JZ4760

Mobile Application Processor

Programming Manual

Release Date: Dec. 14, 2010

北京君正集成电路股份有限公司
Ingenic Semiconductor Co.,Ltd.

JZ4760 Mobile Application Processor

Programming Manual

Disclaimer

This documentation is provided for use with Ingenic products. No license to Ingenic property rights is granted. Ingenic assumes no liability, provides no warranty either expressed or implied relating to the usage, or intellectual property right infringement except as provided for by Ingenic Terms and Conditions of Sale.

Ingenic products are not designed for and should not be used in any medical or life sustaining or supporting equipment.

All information in this document should be treated as preliminary. Ingenic may make changes to this document without notice. Anyone relying on this documentation should contact Ingenic for the current documentation and errata.

Ingenic Semiconductor Co., Ltd.

Room 108, Building A, Information Center, Zhongguancun Software Park
8 Dongbeiwang West Road, Haidian District, Beijing, China,
Tel: 86-10-82826661
Fax: 86-10-82825845
Http: //www.ingenic.cn

CONTENTS

TABLES

FIGURES

1 Overview

JZ4760 is a mobile application processor targeting for multimedia rich and mobile devices like smartphone, tablet computer, mobile digital TV, and GPS. This SOC introduces innovative dual-core architecture to fulfill both high performance mobile computing and high quality video decoding requirements addressed by mobile multimedia devices.

The CPU (Central Processing Unit) core, equipped with 16K instruction cache and 16K data cache operating at 528~600MHz, and full feature MMU function performs OS related tasks. At the heart of the CPU core is XBurst processor engine. XBurst is an industry leading microprocessor core which delivers superior high performance and best-in-class low power consumption. A hardware floating-point unit which compatible with IEEE754 is also included.

The VPU (Video Processing Unit) core is powered with another XBurst processor engine. The SIMD instruction set implemented by XBurst engine, in together with the on chip video accelerating engine and post processing unit, delivers doubled video performance comparing with the single core implementation.

The memory interface supports a variety of memory types that allow flexible design requirements, including glueless connection to SLC NAND flash memory or 4-bit/8-bit/12-bit/16-bit/24-bit ECC MLC/TLC NAND flash memory for cost sensitive applications.

On-chip modules such as audio CODEC, multi-channel SAR-ADC, AC97/I2S controller and camera interface offer designers a rich suite of peripherals for multimedia application. TV encoder unit 10-bits DAC provide composite TV signal output in PAL or NTSC format. The LCD controller support up to 1280x720 output, as well as external HDMI transmitter. WLAN, Bluetooth and expansion options are supported through high-speed SPI and MMC/SD/SDIO host controllers. The TS (Transport stream) interface provides enough bandwidth to connect to an external mobile digital TV demodulator. Other peripherals such as USB OTG and USB 1.1 host, UART and SPI as well as general system resources provide enough computing and connectivity capability for many applications.

1.1 Block Diagram

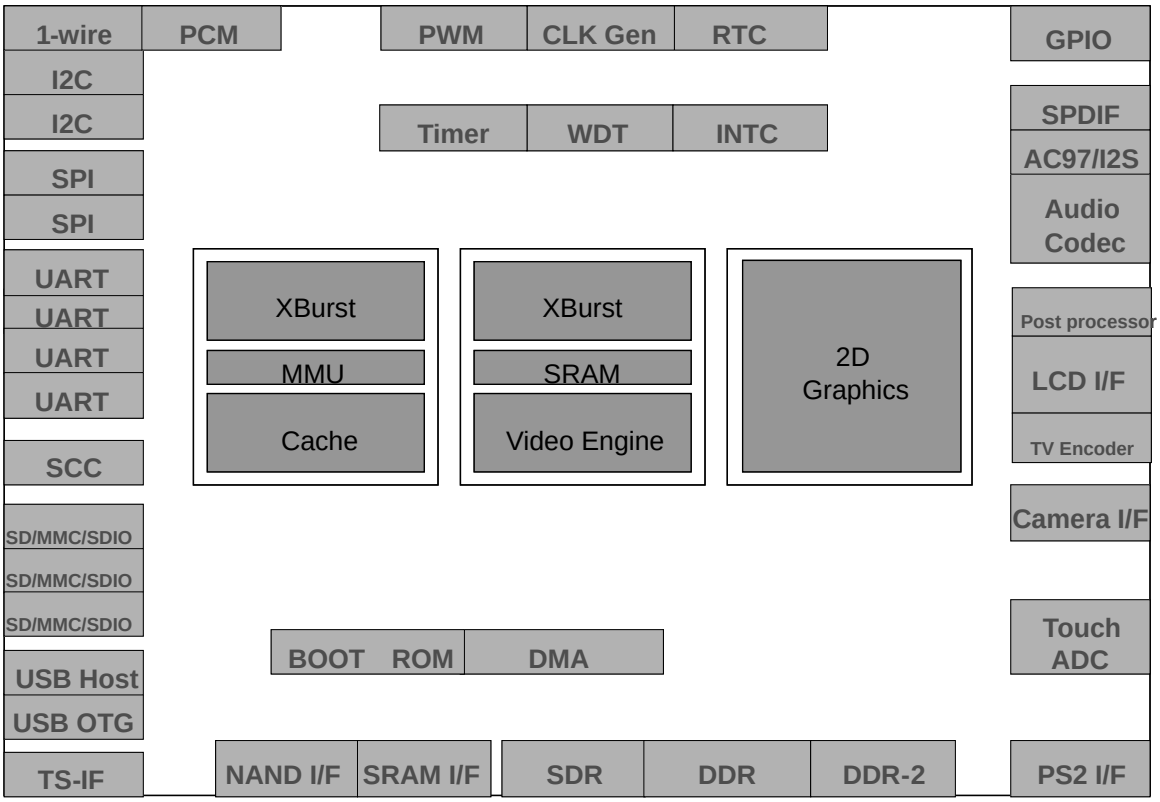


Figure 1-1 JZ4760 Diagram

1.2 Features

1.2.1 CPU Core

- XBurst CPU
  - XBurst® RISC instruction set
  - XBurst® SIMD instruction set
  - XBurst® FPU instruction set supporting both single and double floating point format which are IEEE745 compatible
  - XBurst® 8-stage pipeline micro-architecture up to 600MHz
- MMU
  - 32-entry joint-TLB
  - 4 entry Instruction TLB
  - 4 entry data TLB
- L1 Cache
  - 16K instruction cache
  - 16K data cache
- Hardware debug support
- 16kB tight coupled memory

1.2.2 VPU Core

- XBurst CPU for video processing
  - XBurst® RISC instruction set
  - XBurst® SIMD instruction set
  - XBurst® 8-stage pipeline micro-architecture up to 600MHz
- Video acceleration engine
  - Motion compensation
  - Motion estimation
  - De-block
  - DCT/IDCT for 4x4 block
  - Parser
- 48kB tight coupled memory
- 32kB scratch RAM

1.2.3 GPU Core

- 2D graphic
  - Up to 100M pix/s
  - Up to 1080P
  - Line/Rectangle
  - ROP4/Alpha blending/Filter
  - Rotation (90/180/270 degree)/Mirror
  - 1 Rectangle Clip

1.2.4 Memory Sub-systems

- DDR Controller
  - Support DDR2, DDR, mobile DDR (LPDDR) memory
  - Support x16 and x32 external DDR data width
  - Support clock frequency ratio – (BUS clock) : (DDR clock) = 2:1
  - Support clock frequency ratio – (BUS clock) : (DDR clock) = 1:1
  - Support clock-stop mode
  - Support auto-refresh and self-refresh
  - Support power-down mode and deep-power-down mode
  - Programmable DDR timing parameters
  - Programmable DDR row and column address width
- Static memory interface
  - Direct interface to SRAM, ROM, Burst ROM, and NOR Flash
  - Six chip-select pins for static memory, each can be configured separately
  - Support 8 or 16 bits data width
  - 6 bits address
- NAND flash interface
  - Support 4-bit/8-bit/12-bit/16-bit/24-bit MLC/TLC NAND as well as SLC NAND
  - Support all 8-bit/16-bit NAND Flash devices regardless of density and organization
  - Support automatic boot up from NAND Flash devices
- Synchronous DRAM interface
  - DDR, DDR2 and Mobile DDR
  - Programmable size and base address
  - 32-bit or 16-bit data bus width
  - Multiplexes row/column addresses according to DDR capacity
  - Two-bank or four-bank DDR is supported
  - Supports auto-refresh and self-refresh functions
  - Supports power-down mode to minimize the power consumption of DDR
  - Supports page mode
  - 2 Chip select
- Separate static memory data bus from DRAM one
- BCH Controller
  - Implement data ECC encoding and decoding
- BDMA controller
  - Support up to 3 independent DMA channels
  - Descriptor or No-Descriptor Transfer
  - Transfer data units: byte, 2-byte (half word), 4-byte (word), 16-byte, 32-byte or 64-byte
  - Transfer number of data unit: 1 ~ 224
  - Independent source and target port width: 8-bit, 16-bit, 32-bit
- Direct memory access controller
  - DMA dedicated for memory copy with 3 independent DMA channels
  - DMA dedicated for NAND and ECC with 2 independent DMA channels
  - DMA for others with 10 independent DMA channels

- Descriptor supported
- Transfer data units: 8-bit, 16-bit, 32-bit, 16-byte or 32-byte
- Transfer requests can be: auto-request within DMA; and on-chip peripheral module request
- Interrupt on transfer completion or transfer error
- Supports two transfer modes: single mode or block mode
- Two external DMA channels

- MDMAC controller
  - Support up to 2 independent DMA channels
  - Descriptor or No-Descriptor Transfer
  - Transfer data units: byte, 2-byte (half word), 4-byte (word), 16-byte, 32-byte or 64-byte
  - Transfer number of data unit: 1 ~ 224
  - Independent source and target port width: 8-bit, 16-bit, 32-bit
- The XBurst processor system supports little endian only

1.2.5 AHB Bus Arbiter

- Provide a fair chance for each AHB master to possess the AHB bus
- Fulfill the back-to-back feature of AHB protocol
- Divide two master groups with different privileges supports two arbitrating methods: Round-robin possession for masters in the same group, Preemptive possession for masters with higher privileges

1.2.6 System Devices

- Clock generation and power management
  - On-chip oscillator circuit for an 32768Hz clock and an 12MHz clock
  - On-chip phase-locked loops (PLL) with programmable multiple-ratio. Internal counter are used to ensure PLL stabilize time
  - PLL on/off is programmable by software
  - ICLK, PCLK, HCLK, HHCLK, MCLK and LCLK frequency can be changed separately for software by setting division ratio
  - Supports six low-power modes and function: NORMAL mode; DOZE mode; IDLE mode; SLEEP mode; HIBERNATE mode; and MODULE-STOP function
  - Support module power-down
- RTC (Real Time Clock)
  - 32-bit second counter
  - 1Hz from 32768hz
  - Alarm interrupt
  - Independent power
  - A 32-bits scratch register used to indicate whether power down happens for RTC power
- Interrupt controller
  - Total 32 maskable interrupt sources from on-chip peripherals and external request

through GPIO ports
  - Interrupt source and pending registers for software handling
  - Unmasked interrupts can wake up the chip in sleep or standby mode
- Timer and counter unit with PWM output and/or input edge counter
  - Provide eight separate channels
  - 16-bit A counter and 16-bit B counter with auto-reload function every channel
  - Support interrupt generation when the A counter underflows
  - Three clock sources: RTCLK (real time clock), EXCLK (external clock input), PCLK (APB Bus clock) selected with 1, 4, 16, 64, 256 and 1024 clock dividing selected
  - Every channel has PWM output
- OS timer
  - One channel
  - 32-bit counter and 32-bit compare register
  - Support interrupt generation when the counter matches the compare register
  - Three clock sources: RTCLK (real time clock), EXCLK (external clock input), PCLK (APB Bus clock) selected with 1, 4, 16, 64, 256 and 1024 clock dividing selected
- Watchdog timer
  - 16-bit counter in RTC clock with 1, 4, 16, 64, 256 and 1024 clock dividing selected
  - Generate power-on reset
- Six of them has input signal transition edge counter

1.2.7 Audio/Display/UI Interfaces

- LCD controller
  - Single-panel display in active mode, and single- or dual-panel displays in passive mode
  - 2, 4, 16 grayscales and up to 4096 colors in STN mode
  - 2, 4, 16, 256, 4K, 32K, 64K, 256K and 16M colors in TFT mode
  - 24-bit data bus
  - Support 1,2,4,8 pins STN panel, 16bit, 18bit and 24bit TFT and 8bit I/F TFT
  - Display size up to 1280×720 pixels
  - 256×16 bits internal palette RAM
  - Support ITU601/656 data format
  - Support smart LCD (SRAM-like interface LCD module)
  - Support delta RGB
  - One single color background and two foreground OSD
- TV encoder
  - Support NTSC or PAL
  - Support CVBS signal
  - 10 bits DAC
- EPD controller
  - Supports EINK epds and compatible devices
  - Supports different size of display panel
  - Supports different width of pixel data

- Supports internal DMA operation and register operation

- Image post processor
  - Video frame resize
  - Color space conversion: 420/444/422 YUV to RGB convert
  - Bi-cubic algorithm supported
- Camera interface module
  - Input image size up to 4096×4096 pixels
  - Supports CCIR656 data format
  - YCbCr 4:2:2 and YCbCr 4:4:4 data format
  - Raw data input
  - 64×32 image data receive FIFO with DMA support
- On-chip audio CODEC
  - 24-bit DAC, SNR: 95dB
  - 24-bit ADC, SNR: 90dB
  - Sample rate: 8/9.6/11.025/12/16/22.05/24/32/44.1/48/96kHz
  - L/R channels line input
  - 2 MICs input, differential or single-ended
  - L/R channels headphone output amplifier support up to 16ohm load
  - Capacitor-coupled
  - Mono differential line out
  - Mono 450mW amplifier for speaker out for 8ohm load
- AC97/I2S/SPDIF controller
  - Supports 8, 16, 18, 20 and 24 bit for sample for AC-link and I2S/MSB-Justified format
  - Support 2/4/6/8 channels data out for I2S
  - Support compress data format for SPDIF
  - DMA transfer mode support
  - Support variable sample rate mode for AC-link format
  - Power down mode and two wake-up mode support for AC-link format
  - Programmable Interrupt function support
  - Support the on-chip CODEC
  - Support off-chip CODEC
  - Support off-chip HDMI transmitter audio
- PCM interface
  - Data starts with the frame PCMSYN or one PCMCLK later
  - Support three modes of operation for PCM: Short frame sync mode, Long frame sync mode, Multi-slot mode
  - Data is transferred and received with the MSB first
  - Support master mode and slave mode
  - The PCM serial output data, PCMDOUT, is clocked out using the rising edge of the PCMSCLK
  - The PCM serial input data, PCMDIN, is clocked in on the falling edge of the PCMSCLK.
  - 8/16 bit sample data sizes supported
  - DMA transfer mode supported

  - Two FIFOs for transmit and receive respectively with 16 samples capacity in every direction
- SADC
  - 12-bit, 1Msps/200ksps
  - XP/XN, YP/YN inputs for touch screen
  - Battery voltage inputs for internal/external resistor divider respectively
  - 2 generic input channels
  - 5mW@1Msps, 2.2mW@200ksps
- Support e-paper type panel

1.2.8 On-chip Peripherals

- General-Purpose I/O ports
  - Total GPIO pin number is 166, where 8 are dedicated and all others are shared
  - Each pin can be configured as general-purpose input or output or multiplexed with internal chip functions
  - Each pin can act as a interrupt source and has configurable rising/falling edge or high/low level detect manner, and can be masked independently
  - Each pin can be configured as open-drain when output
  - Each pin can be configured as internal resistor pull-up
- Two I2C bus interfaces
  - Only supports single master mode
  - Supports I2C standard-mode and F/S-mode up to 400 kHz
  - Double-buffered for receiver and transmitter
  - Supports general call address and START byte format after START condition
- Two Synchronous serial interfaces (SSI0, SSI1)
  - Up to 50MHz speed
  - Supports three formats: TI's SSP, National Microwire, and Motorola's SPI
  - Configurable 2 - 17 (or multiples of them) bits data transfer
  - Full-duplex/transmit-only/receive-only operation
  - Supports normal transfer mode or Interval transfer mode
  - Programmable transfer order: MSB first or LSB first
  - 17-bit width, 128-level deep transmit-FIFO and receive-FIFO
  - Programmable divider/prescaler for SSI clock
  - Back-to-back character transmission/reception mode
- One-wire bus interface
  - Overdrive and regular speed
  - Master only
  - LSB first
  - Bit or byte operate modes
- USB 1.1 host interface
  - Open Host Controller Interface (OHCI)-compatible and USB Revision 1.1-compatible
  - Full speed and low speed

  - Embedded USB 1.1 PHY
- USB 2.0 OTG interface
  - Compliant with USB protocol revision 2.0 OTG
  - High speed and full speed supported for device role
  - High speed, full speed and low speed supported for host role
  - Embedded USB OTG PHY
- Three MMC/SD/SDIO controllers (MSC0, MSC1, MSC2)
  - Support automatic boot up from MSC0, which has 4-bit data bus
  - MSC1 with 4-bit data bus
  - Compliant with "The MultiMediaCard System Specification version 4.2"
  - Compliant with "SD Memory Card Specification version 2.0" and "SDIO Card Specification version 1.0" with 1 command channel and 4 data channels
  - Up to 320 Mbps data rate in MSC0
  - Up to 320 Mbps data rate in MSC1
  - Supports up to 10 cards (including one SD card)
  - Maskable hardware interrupt for SD I/O interrupt, internal status, and FIFO status
- Four UARTs (UART0, UART1, UART2, UART3)
  - 5, 6, 7 or 8 data bit operation with 1 or 1.5 or 2 stop bits, programmable parity (even, odd, or none)
  - 32x8bit FIFO for transmit and 32x11bit FIFO for receive data
  - Interrupt support for transmit, receive (data ready or timeout), and line status
  - Supports DMA transfer mode
  - Provide complete serial port signal for modem control functions
  - Support slow infrared asynchronous interface (IrDA)
  - IrDA function up to 115200bps baudrate
  - UART function up to 3.7Mbps baudrate
  - Hardware flow control
- Smart Card Controller
  - Supports normal card and UIM card
  - 8-bit, 16-level receive-/transmit- FIFO
  - Supports asynchronous character (T=0) communication modes
  - Supports asynchronous block (T=1) communication modes
  - Supports setting of clock-rate conversion factor F (372, 512, 558, etc.), and bit-rate adjustment factor D (1, 2, 4, 8, 16, 32, 12, 20, etc.)
  - Supports extra guard time waiting
  - Auto-error detection in T=0 receive mode
  - Auto-character repeat in T=0 transmit mode
  - Transforms inverted format to regular format and vice versa
  - Support stop clock function in some power consuming sensitive applications
- PS/2 keyboard controller
  - Be compatible with 8042
  - Support PS/2 keyboard and mouse
  - Bi-directional synchronous serial transfer operation

- A frame format: 1 start bit, 8 data bits, 1 odd parity bit and 1 stop bit
- Device provides about 20KHz clock to KBC
- KBC has priority over the data line and can inhibit communication from the keyboard/mouse at any time by holding Clock low

- Transport stream slave interface
  - 8-bit or 1-bit data bus selectable
  - Support PID filtering

1.2.9 Bootrom

- 8kB Boot ROM memory

1.3 Characteristic

Item	Characteristic
Process Technology	0.13um CMOS
Power supply voltage	General purpose I/O: 3.3 ± 0.3V DDR I/O for mDDR: 1.8V± 0.2V DDR I/O for DDR: 2.5V± 0.2V DDR I/O for DDR2: 1.8V± 0.2V NAND/SRAM I/O: 1.62V~3.6V RTC I/O: 1.8V± 0.1V Core: 1.2 ± 0.1V
Package	BGA345 14mm x 14mm x 1.1mm, 0.65mm pitch
Operating frequency	528~600MHz

2 CPU Core

Enhanced features of CPU core in JZ4760 include:

- Enhanced MXU implements XBurst SIMD instruction set release I and release II.
- TCSM, tightly coupled shared memory with physical address scope 0x132B0000 ~ 0x132BFFFF.
- PMON, processor performance monitor.
- FPU, floating point unit implemented to improve floating point number processing ability.

2.1 Block Diagram

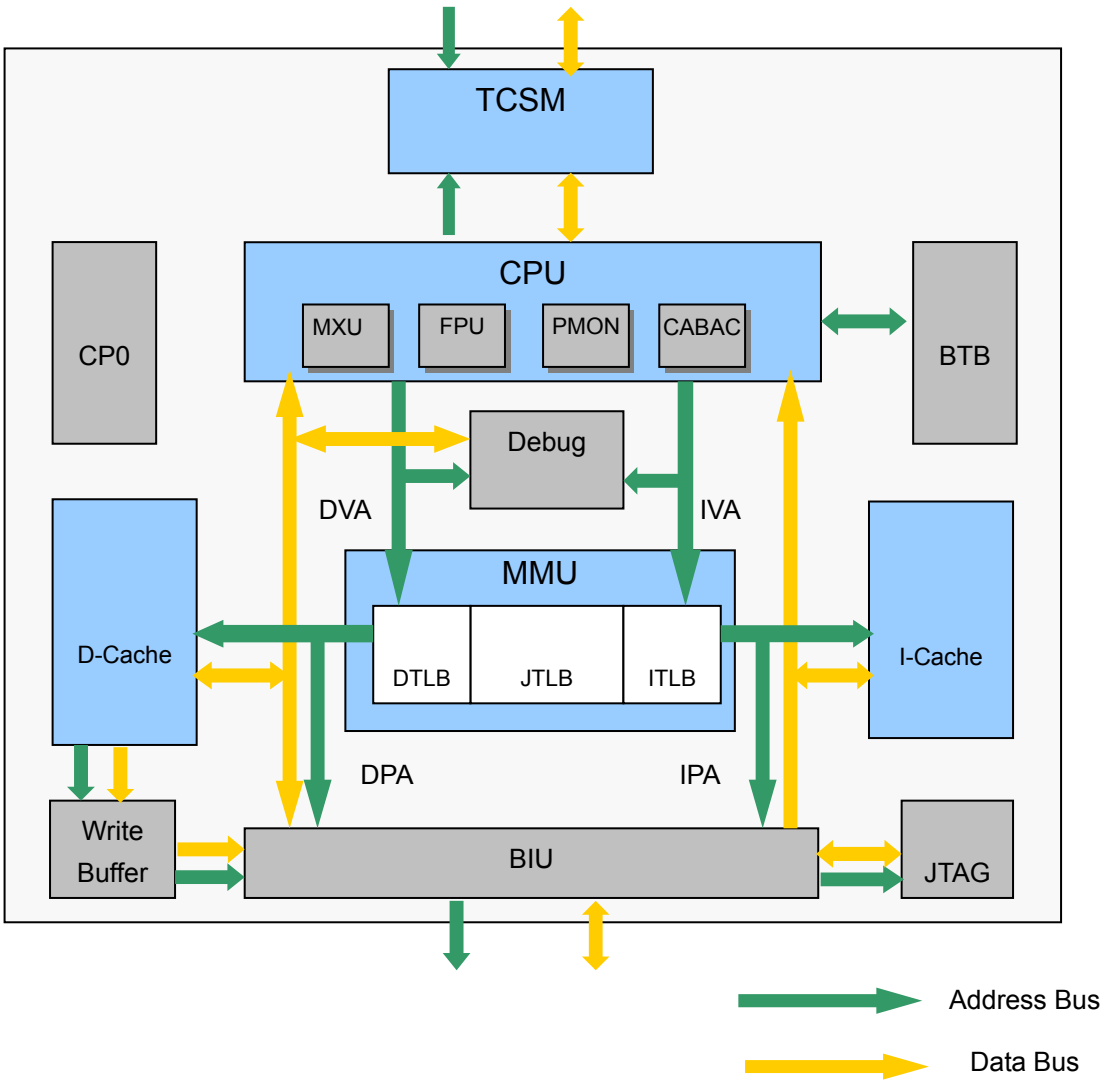


Figure 2-1 Structure of CPU core

2.2 Extra Features of the CPU core in JZ4760

Item	Features
Media Extension Unit (MXU)	● XBurst SIMD instruction set release I and release II ● fully pipelined
Tightly Coupled Sharing Memory (TCSM)	● Size: 16K bytes ● Same clock freqency as L1 cache ● AHB slave interface ● Four banks support up to four simultaneous accesses
Floating Point Unit (FPU)	● Comply with IEEE754 standard ● Support single and double format ● not fully pipelined implementation
CABAC interface	● Part of bitstream processing cooperating CABAC in VPU ● Dedicated CP0 interface is CP0 register number 21, select0~7
Performance Monitor (PMON)	● Real-time monitor ● Dedicated CP0 interface
Processor ID	Value read from CP0.PRId is 0x2ed0024f

Please refer to documents XBurst-ISA and XBurst1_PM for ISA and programming relative details.

2.3 Instruction Cycles

Most instructions have one cycle repeat rate, that is, when the pipeline is fully filled, there is one instruction issued per clock cycle. However, some particular instructions require extra cycles. Following table lists cycle consumption of all instructions belonging to XBurst-ISA implemented in JZ4760.

1st Instruction	2nd Instruction	Cycles	Description
WAIT	**Anyone**	variable	WAIT instruction will be repeatedly executed until an interrupt arise.
MTC0 **TLBWI/TLBWR** **TLBP/TLBR**	**Anyone**	4	3 extra interlock cycles.
CACHE	**Anyone**	2	1 extra interlock cycles.
JMP/BC	**Anyone** **(delay slot)**	4/1	0 cycle penalty when BTB predicts taken and the branch is taken or BTB predicts untaken and the branch is untaken or BTB miss and the branch is untaken. Otherwise, extra 3 cycles penalty.
BCL	**Anyone** **(delay slot)**	5/4/2/1	0 cycle penalty when BTB predicts taken and branch is taken, otherwise: 1 BTB miss, branch is taken, 3 cycles penalty. 2 BTB miss, branch is untaken, 1 cycle penalty. 3 BTB predict taken, branch is untaken, 4 cycles penalty. 4 BTB predict untaken, branch is taken, 3 cycles penalty.
MULT/MULTU **MADD/MADDU** **MSUB/MSUBU**	**MULT/MULTU** **MADD/MADDU** **MSUB/MSUBU**	4	3 extra interlock cycles due to MDU operating hazard.
	MUL/DIV/DIVU	4	3 extra interlock cycles due to MDU operating hazard.
	MFHI/MFLO **MTHI/MTLO**	4	3 extra interlock cycles due to MDU operating hazard.
	Any other	1	No data dependency or hazards exist.
MUL	**MULT/MULTU** **MADD/MADDU** **MSUB/MSUBU**	4	3 extra interlock cycles due to MDU operating hazard.
	MUL/DIV/DIVU	4	3 extra interlock cycles due to MDU operating hazard.
	MFHI/MFLO **MTHI/MTLO**	4	3 extra interlock cycles due to MDU operating hazard.

	Any other	4/1	If the second instruction has RAW data dependency, 3 extra interlock cycles, otherwise, 0 cycle penalty.
DIV/DIVU	**MULT/MULTU MADD/MADDU MSUB/MSUBU MUL/DIV/DIVU**	4~35	3~34 extra interlock cycles determined by characteristic value of divider and dividend.
	MFHI/MFLO	2~34	1~33 interlock cycles determined by characteristic value of divider and dividend.
	Any other	1	No data dependency or hazards exist.
MFHI/MFLO/MFC0	**Anyone**	4/1	If the second instruction has RAW data dependency, 3 extra interlock cycles, otherwise, 0 cycle penalty.
LW/LL LWL/LWR LB/LBHU LH/HU LXW LXH/LXHU LXB/LXBU	**Anyone**	4/1	If the second instruction has RAW data dependency, 3 extra interlock cycles, otherwise, 0 cycle penalty.
D16MUL/D16MULF D16MAC/D16MACF D16MULE/D16MACE	**SIMD instruction**	3/1	If the second SIMD instruction has RAW data dependency, 2 extra interlock cycles, otherwise, 0 cycle penalty.
	Any other	1	No data dependency or hazards exist.
D32ACC/Q16ACC Q8SAD S32MAX/S32MIN D16MAX/D16MIN D32ACCM/D32ASUM Q16ACCM/D16ASUM	**SIMD instruction**	2/1	If the second SIMD instruction has RAW data dependency, 1 extra interlock cycle, otherwise, 0 cycle penalty.
	Any other	1	No data dependency or hazards exist.
S32LDD/S32LDDV S32LDI/S32LDIV S32LDDR/S32LDDVR S32LDIR/S32LDIVR S16LDD/S16DI S8LDD/S8LDI	**SIMD instruction**	2/1	If the second SIMD instruction has RAW data dependency, 1extra interlock cycle, otherwise, 0 cycle penalty.
	Any other	1	No data dependency or hazards exist.
S32I2M	**SIMD instruction**	2/1	If the second SIMD instruction has RAW data dependency, 1extra interlock cycle, otherwise, 0 cycle penalty.
	Any other	1	No data dependency or hazards exist.

S32M2I	**Anyone**	4/1	If the second instruction has RAW data dependency, 3 extra interlock cycles, otherwise, 0 cycle penalty.
S32EXTR **S32EXTRV**	**SIMD instruction**	2/1	If the second SIMD instruction has RAW data dependency, 1extra interlock cycle, otherwise, 0 cycle penalty.
	Any other	1	No data dependency or hazards exist.
Others	**Anyone**	1	

NOTE: JMP denotes J and JR instructions; BC denotes branch conditionally instructions; BCL denotes branch conditionally and likely instructions.

2.4 TCSM

TCSM (tightly coupled shared memory) is a dedicated on-chip SRAM. It serves as an on-chip scratchpad memory, moreover, it acts as a high-speed SRAM for CPU. Through the TCSM, CPU and VPU's AHB masters such as DBlock can exchange data quickly and efficiently. TCSM in JZ4760's CPU core has following features:

- 16K bytes
- The same clock frequency as L1 cache
- Physical address scope from 0x132B,0000 to 0x132B,FFFF
- Four banks support up to four simultaneous accesses if no bank conflicts occurs

Moreover, like the **dseg** section separated from K3 section, another **tcsm** section with 16MB capacity range from 0xF400,0000 to 0xF4FF,FFFF is separated too. This virtual address section is uncacheable and unmappable and can only be accessed by CPU core in kernel mode.

Please note the fact that the capacity of TCSM in JZ4760 is only 16K bytes, which denotes that available virtual address range is from 0xF400,0000 to 0xF400,3FFF and available physical address range is from 0x132B,0000 to 0x132B,3FFF.

2.4.1 TCSM Occupied Available Physical Address Range

Physical Address range 0x132B,0000 ~ 0x132B,FFFF are reserved for TCSM. In Jz4760, physical address range 0x132B,0000 ~ 0x132B,3FFF are available and others are reserved, and corresponding address partition for the four banks are as following:

bank0: 0xF4000000~0xF4000FFF (virtual); 0x132B,0000~0x132B,0FFF (physical)
bank1: 0xF4001000~0xF4001FFF (virtual); 0x132B,1000~0x132B,1FFF (physical)
bank2: 0xF4002000~0xF4002FFF (virtual); 0x132B,2000~0x132B,2FFF (physical)
bank3: 0xF4003000~0xF4003FFF (virtual); 0x132B,3000~0x132B,3FFF (physical)

Therefore, arranging instructions and data into different banks can achieve best access performance. Similarly, using ping-pong buffers located in the separate banks for efficient data exchange between CPU core and other VPU's AHB masters is a better choice.

2.5 PMON

PMON is a simple performance monitor. In JZ4760, following performance relative real-time events can be monitored.

- I-cache miss times
- D-cache miss times
- Total issued instructions
- Discarded instructions
- Pipeline freeze cycles
- CPU clock cycles

A dedicated software interface is devised to manipulate PMON in kernel mode, that is, CP0 Config4 ~ Config7 registers are extended for PMON. Refer to chapter of CP0 in the document XBurst1_PM for detail.

2.5.1 Fundamental

When PMON is enabled (set value 1 to config7.bit8), one preset event pair determined by config7.bit15~bit12 will be continuously monitored until PMON is disabled (set value 0 to config7.bit8). Finally, loading values of CP0.config4~CP0.config6 can get monitored result.

3 VPU Core

Video Processor Unit (VPU) core in this chip is dedicated for video decoding and encoding. VPU embeds an XBurst® CPU core (named AUX in VPU) and application specified hardware accelerators for common video compress/decompress algorithms, which includes IDCT, Motion Compensation, Motion Estimation, De-Block, CABAC. Further more 3 route general purpose DMA enhances data management and transfer efficiency during video encoding/decoding.

XBurst® core's powerful programming agility combining with specified algorithm accelerators' high hotspot processing ability ensures VPU's multi format supporting and high performance ability. This distinctive structure brings us a nice trade-off of DSP's high power consumption and low processing ability with Hardware IP's complicated large logic size and limited format supporting.

Key standards performance of VPU in chip JZ4760:

- RealVideo decoding up to 720P 30fps
- MPEG-2 decoding up to 720P 30fps
- MPEG-4 decoding up to 720P 30fps
- VC-1 decoding up to 720P 30fps
- H.264 decoding up to 720P 30fps
- MPEG-4 encoding up to VGA 20fps

3.1 Block Diagram

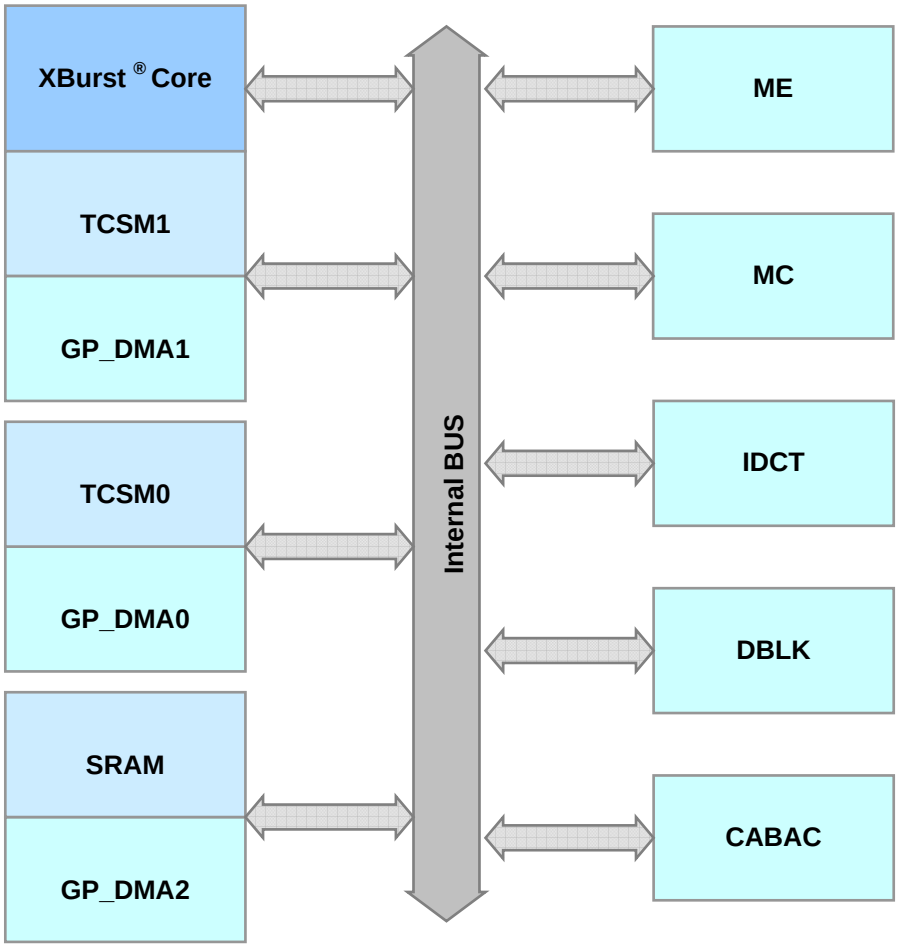


Figure 3-1 VPU Block Diagram

3.2 Features of VPU

Table 3-1 VPU Features

Item	Features
XBurst® core(AUX)	● XBurst-1 CPU – Industry standard RISC instruction set – 32 32-bit general purpose registers, no shadow GPR – 8-stage pipeline – Interlocked implementation – Physical address accessing directly ● Media Extension Unit (MXU) – Ingenic SIMD instruction set II – fully pipelined
Tightly Coupled Sharing Memory (TCSM)	● TCSM0 – Size: 16K bytes – AHB slave interface supports external DMA access ● TCSM1 – Size: 48K bytes – AHB slave interface supports external DMA access **NOTE:** TCSM0 is coupled with J1 externally and serves as a memory interface for VPU, while TCSM1 is coupled with VPU XBurst® core internally.
Scratch RAM (SRAM)	– Size: 32K bytes – AHB slave interface supports external DMA access
Gerenal Purpose DMA(GP_DMA)	● GP_DMA0 – Descriptor based DMA ● GP_DMA1 – Descriptor based DMA ● GP_DMA2 – Descriptor based DMA **NOTE:** GP_DMA0 is coupled with TCSM0, GP_DMA1 is coupled with TCSM1 and GP_DMA2 is coupled with SRAM as well.
Motion Estimation(ME)	– Fast diamond searching strategy – Fixed integral pixel accuracy – Fixed 8x8 searching unit
Motion Compensation (MC)	– Descriptor based task fetching – Programmable processing size from 2x2 to 16x16 – Programmable interplation filter from 2-tap to 8-tap – Programmable sub-pixel accuracy from 1/2-pixel to 1/8-pixel
Inverse DCT(IDCT)	– Descriptor based task fetching – RealVideo 4x4 IDCT support

	– H.264 4x4 DCT/IDCT/Intra prediction support – VC-1 4x4 IDCT support
De-block (DBLK)	– Descriptor based task fetching – RealVideo in loop filter support – H.264 in loop filter support, MBAFF not support
CABAC	– Context-based binary arithmetic decoding support

3.3 AUX

3.3.1 Overview

AUX is a simplified MAIN core without FPU, CACHE, MMU, CP0 and DBG&JTAG, however, it has a dedicated TCSM named TCSM1. As an AHB device of VPU, AUX has several memory mapped registers serving for its controls and communication with MAIN core. Since there is no MMU, AUX CPU core can only access physical address space.

Features:

- The same clock frequency as MAIN core
- Support most XBurst instructions except those ones manipulating CACHE, MMU and CP0
- 48K bytes TCSM1 with the same structure as TCSM in MAIN core
- Available physical address scope of TCSM1 is 0x132C,0000 ~ 0x132C,.BFFF
- Available virtual address scope of TCSM1 (can only be accessed by AUX) is 0xF400,0000 ~ 0xF400,BFFF
- Total six banks of TCSM1 have following address partition
  - Bank0: 0xF400,0000~0xF400,1FFF (virtual); 0x132C,0000~0x132C,1FFF (physical)
  - Bank1: 0xF400,2000~0xF400,3FFF (virtual); 0x132C,2000~0x132C,3FFF (physical)
  - Bank2: 0xF400,4000~0xF400,5FFF (virtual); 0x132C,4000~0x132C,5FFF (physical)
  - Bank3: 0xF400,6000~0xF400,7FFF (virtual); 0x132C,6000~0x132C,7FFF (physical)
  - Bank4: 0xF400,8000~0xF400,9FFF (virtual); 0x132C,8000~0x132C,9FFF (physical)
  - Bank5: 0xF400,A000~0xF400,BFFF (virtual); 0x132C,A000~0x132C,BFFF (physical)

3.3.2 Memory Mapped Registers Definition

The physical address base for the memory-mapped registers of AUX is 0x132A,0000.

3.3.2.1 Control and Status

AUX_CTRL **offset 0x0**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SLEEP																							BTB_INV					MIRQ_EN	NMI_DIS	SW_NMI	SW_RST
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0	1

Bits	Name	Description	R/W
31	SLEEP	AUX_IU sleep status. 1: sleep; 0: no sleep.	R
30:9	Reserved	Writing has no effect, read as zero.	R

8	BTB_INV	Writing 1 can invalid BTB. Writing 0 has no effect, read as zero.	W
7:4	Reserved	Writing has no effect, read as zero.	R
3	MIRQ_EN	1: enable message IRQ to main CPU; 0: disable.	RW
2	NMI_DIS	1: NMI pulse only wakes up AUX_IU from sleep status 0: NMI pulse wakes up AUX_IU meanwhile triggers a NMI exception	RW
1	SW_NMI	Nonmaskable IRQ (NMI). Writing 1 to the field triggers a NMI pulse to AUX_IU. Writing 0 has no effect, read as zero.	W
0	SW_RST	Software reset. 1: reset AUX_IU; 0: do not reset.	RW

NOTES:

1 When NMI or reset excpetion occurs, AUX resumes from PC 0xF4000000.

2 AUX will sleep after it executes a WAIT instruction.

3 When AUX wakes up due to an NMI pulse meanwhile NMI_DIS is 1, AUX just resumes from the next PC of the WAIT instruction.

3.3.2.2 SPINLOCK

AUX_SPINLK **offset 0x4**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																															LOCK	
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0

Bits	Name	Description	R/W
30:2	Reserved	Writing has no effect, read as zero.	R
1:0	LOCK	Lock status.	RW

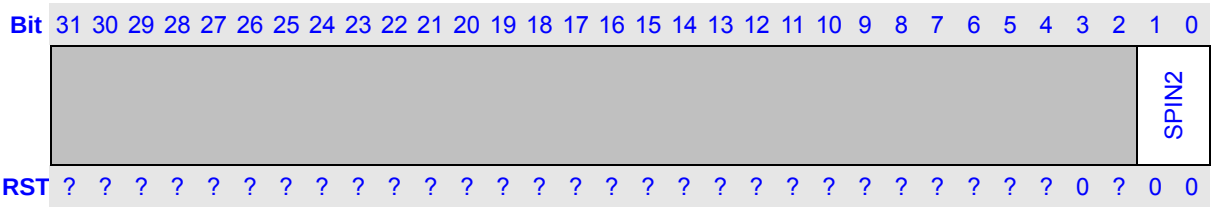
AUX_SPIN1 **offset 0x8**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																															SPIN1	
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	?	0	0

Bits	Name	Description	R/W
30:2	Reserved	Writing has no effect, read as zero.	R
1:0	SPIN1	Reading SPIN1 triggers following special hardware operations. First, value of AUX_SPINLK will be checked, if the value equals zero, the value of SPIN1 will overwrite AUX_SPINLK immediately, otherwise, AUX_SPINLK keeps unchanged. Then	RW

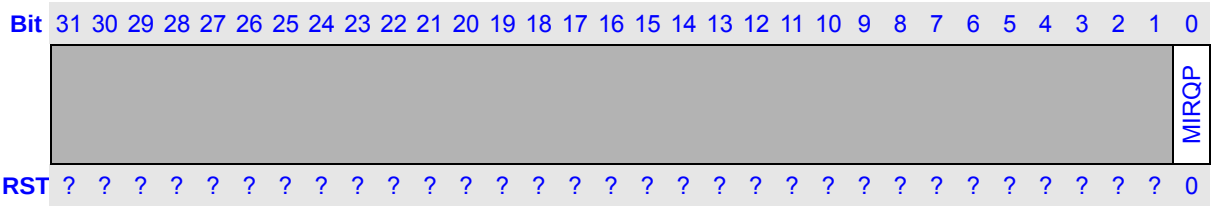
		reading AUX_SPINLK instead of SPIN1 supplies the final read result. Writing SPIN1 is a normal write operation.	

AUX_SPIN2 **offset 0xC**



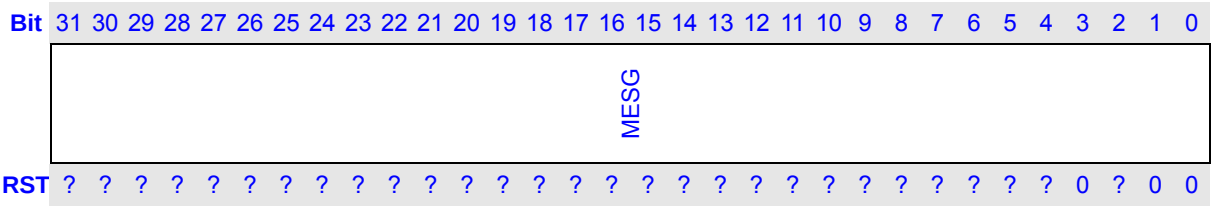
Bits	Name	Description	R/W
30:2	Reserved	Writing has no effect, read as zero.	R
1:0	SPIN2	The operations for SPIN1 also fit SPIN2 except the role of SPIN1 should be replaced by SPIN2.	RW

AUX_MIRQP **0x10**



Bits	Name	Description	R/W
31:1	Reserved	Writing has no effect, read as zero.	R
0	MIRQP	Pending status of MIRQ (messgae IRQ) which can only be set to 1 by HW and be reset to 0 by SW.	RW

AUX_MSG **0x14**



Bits	Name	Description	R/W
31:0	MESG	If MIRQ_EN bit of AUX_CTRL. is value 1, writing the register raises an IRQ routing to the main CPU meanwhile the AUX_MIRQP is set to 1 by HW automatically. The IRQ then keeps active until the register AUX_MIRQP is cleared to 0 by SW.	RW

3.4 TCSM/SRAM

TCSM0/TCSM1/SRAM serves as the VPU control flow and data flow's communication between XBurst® CPU core with specified algorithm hardware accelerators and different hardware accelerators as well.

3.4.1 TCSM/SRAM space usage

Table 3-2 TCSM space usage

	XBurst® J1	XBurst® AUX	HW accelerator
TCSM0	0xF400_0000 ~ 0xF400_3FFF	0x132B_0000 ~ 0x132B_3FFF	0x132B_0000 ~ 0x132B_3FFF
TCSM1	0x132C_0000 ~ 0x132C_BFFF	0xF400_0000 ~ 0xF400_BFFF	0x132C_0000 ~ 0x132C_BFFF
SRAM	0x132D_0000 ~ 0x132D_7FFF	0x132D_0000 ~ 0x132D_7FFF	0x132D_0000 ~ 0x132D_7FFF

NOTES:

1 TCSM1/SRAM's space list for XBurst® J1 is physical address. In actual using it must be translated to its relative virtual address for XBurst® J1's accessing.

3.5 GP_DMA

3.5.1 Overview

GP_DMA is a 2-D data transfer DMA controller, which is tightly coupled with TCSM0/TCSM1/SRAM. Due to this tightly coupling, the data path for transferring should be limited as the following:

Table 3-3 GP_DMA data transfer path

GP_DMA	Validity of data transfer path
GP_DMA0	From other slavers to TCSM0 is valid. From TCSM0 to other slavers is valid. From TCSM0 to TCSM0 is forbiden.
GP_DMA1	From other slavers to TCSM1 is valid. From TCSM1 to other slavers is valid. From TCSM1 to TCSM1 is forbiden.
GP_DMA2	From other slavers to SRAM is valid. From SRAM to other slavers is valid. From SRAM to SRAM is forbiden.

GP_DMA is working under descriptor-based configuration. Its descriptor node is defined as:

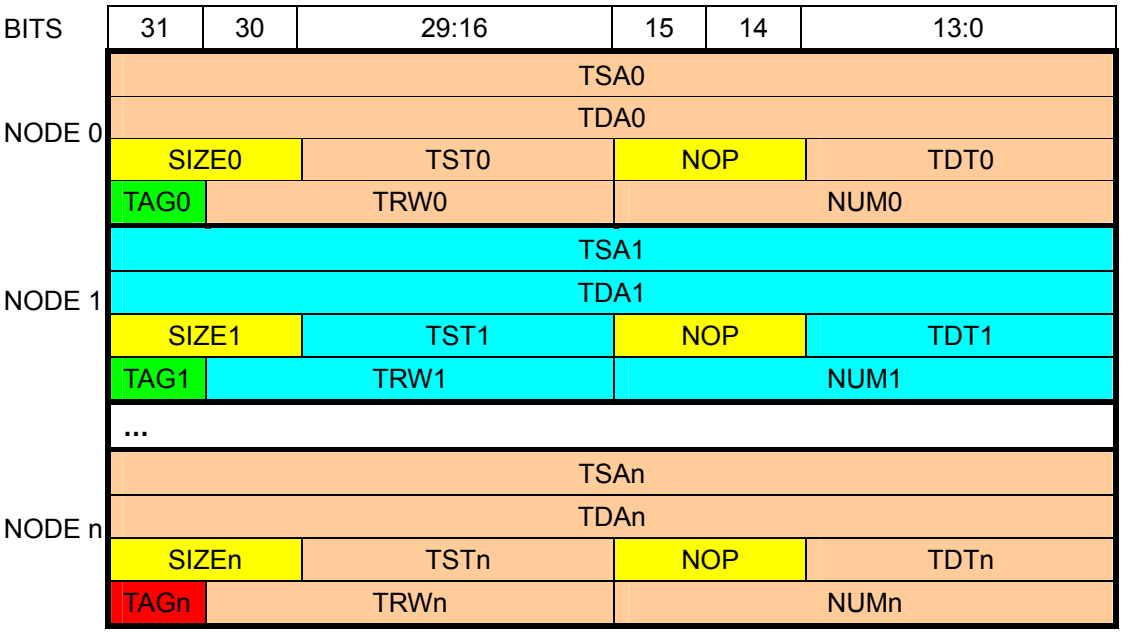


Figure 3-2 GP_DMA descriptor node structure

Table 3-4 GP_DMA descriptor node description

Item	Meaning
TSA	transfer source ADDRESS.
TDA	transfer destination ADDRESS.
TST	transfer source STRIDE.
TDT	transfer destination STRIDE.
TRW	transfer row WIDTH.
NUM	transfer byte NUMBER.
SIZE	transfer size type. 0: word 1: byte 2: half-word
TAG	Transfer link end tag. (GP_DMA parses each node to do data transfer and then go on parsing next adjacent node until it accomplishes a node with TAG equaling 1)

3.5.2 Register Definition

3.5.2.1 Descriptor Head Address (DHA)

DHA 0x0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DHA																														Reserved	
RST																																

Bits	Name	Description	RW
31:2	DHA	Descriptor Head Address.	RW
1:0	Reserved	1-word align.	

3.5.2.2 DMA Status/Command (DCS)

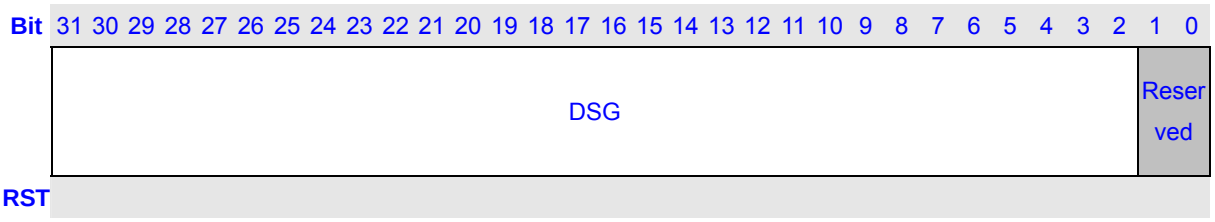
DCS 0x4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BTN																NDN								Reserved					END	RST	SUP
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:16	BTN	Transfer number byte.	R
15:8	NDN	Transfer node number.	R
15~3	Reserved		
2	END	0: GP_DMA in transferring; 1: transmit end, GP_DMA is idle.	R
1	RST	GP_DMA SW reset. GP_DMA would be reset when it was written as 1.	RW
0	SUP	GP_DMA startup.	RW

3.5.2.3 Frame Stride Gate (DSG)

DSG 0x8



Bits	Name	Description	RW
31:2	DSG	DDR frame Stride Gate.	RW
1:0	Reserved	1-word align.	

3.6 Video Acceleration Block

Please refer to documents JZ4760_VPU and programming relative details.

4 GPU Core

4.1 Overview

GPU defines a high-performance 2D raster graphics core that accelerates the 2D graphics display on a variety of consumer devices. Addressable screen sizes range from the smallest cell phones to HD 1080p displays. It is designed to provide powerful graphics at low power consumption and the smallest of silicon footprints. Dynamic power consumption is minimized by extensive use of localized clock gating. The design also includes a 32-bit AHB interface, a 64-bit AXI interface, and support for virtual memory. It accelerates numerous 2D applications, including on-screen display and touch screen user interfaces, graphical user interfaces (GUI) and menu displays, Flash animation, and gaming, as featured in popular consumer devices like:

- Digital cameras
- Personal printers
- Feature phones
- Digital picture frames
- Digital signage
- Portable and in-dash navigation systems
- MIDs and Netbooks
- Set-top boxes
- DTV

GPU supports the following graphics APIs:

- DirectFB (on Linux)
- GDI/DirectDraw (on Windows CE)

4.2 OPERATIONS and FEATURES

4.2.1 Line

The LINE operation draws a line. Coordinates for two points are given: start point and end point. The end point is not drawn. Lines are rendered using the Bresenham algorithm. The Bresenham algorithm has the advantage of using integer arithmetic and has no accumulation of rounding errors. In the case of line, only ROP2 and ROP4 are supported. It operates on pattern and destination. The pattern should have a transparency mask in order to use ROP4. Clipping is supported for lines on a per pixel basis.

4.2.2 Rectangle Fill and Clear

Rectangle fill suffuses a rectangle area with a given color. Essentially rectangle fill is a pattern fill, where an 8x8 pattern is initialized with the specified color. It supports ROP2 and ROP4 with the pattern and destination as its inputs. If ROP4 is used, the pattern should have a transparency mask. Clear is similar to rectangle fill except that it does not does not use a pattern. A 32-bit clear value with 4-bit byte mask is used to fill the entire rectangle area. Both rectangle fill and clear support clipping, which is performed on a per primitive basis.

4.2.3 BitBLT

Bit blit transfers data from one area of a memory (source) to another area of the memory (destination). The source and destination can be from the same or different memories. Both source and destination must be described by a rectangular area. The source and destination rectangles can be the same size (most bit blits are in this nature) or it can be different sizes. In which case, it becomes a stretch or shrink blit. Bit blit supports both ROP2, ROP3, and ROP4 which includes source, destination and pattern, and an optional transparency color. Clipping can be performed on the primitive basis.

4.2.4 Stretch BLT

The STRETCH BLT primitive performs a BitBlit operation with stretch or shrink. The modified Bresenham algorithm is used to generate corresponding coordinates for fast stretching. The stretch factor is specified in a 15.0 fixed-point format. Stretch blit is not allowed to overlap, that is no part of source and destination can share any piece of memory. Non-stretch blits can overlap. For stretch blit clipping is performed on a per pixel basis.

4.2.5 Mono Expansion / Mask BLT

Mono color representation means using just one bit per pixel to represent colors. A typical application for mono color is font drawing. The MONO EXPANSION primitive increases color representation from one bit per pixel to multiple bits per pixel. Monochrome expansion / Masked blit are similar to bit blit with a major difference. It supports a monochrome mask for ROP4 selection. It means that each output pixel can be a combination of source, pattern, monochrome mask (for masked blits) and destination

For monochrome blit, the source and destination rectangle must be the same size.
Monochrome blit does not support overlapping of the source and destination. It is the responsibility of the driver to make sure that it will never execute the command on overlapping source and destination. Mask blit uses a color source from the memory and monochrome mask from the command stream. Monochrome data for the blit comes from the command stream. Clipping is supported and is performed on a per pixel basis.

4.2.6 Filter BLT

FILTER BLT performs high quality scaling, up or down, using an FIR re-sampling filter with up to 9 taps. Sub-pixel coordinates (locations between the pixel grids) are generated by the drawing engine. The filter block in the drawing engine uses the sub-pixel information to select the appropriate filter kernel. A stretch- or shrink-factor of 15.16 fixed-point format is supported. To generate a single destination pixel requires 9 source pixels. An image is scaled in two passes, one for X-dimension (HOR_FILTER_BLT) and the other for Y-dimension (VER_FILTER_BLT). Software sets up the filter kernel/coefficient table and the kernel size, as well as a temporary buffer for storing intermediate results. After the first pass is completed, intermediate results are sent back to memory, and then the second pass starts to scale the first-pass image. Because of this two-step procedure, the throughput of FILTER BLT is lower than that of STRETCH BLT. Also the Filter Kernel Table may need to be reloaded, and some cycles are consumed in calculating the stepping parameters. When the stretch or shrink factor is 1, the filterBlit works as a bitBlit copy. It can be used as format converter in that case, for instance, YUV to RGB converter. To use as a format converter, only one pass (HOR_FILTER_BLT or VER_FILTER_BLT) is needed. To optimize the memory bandwidth, when using filterBlit to do YUV to RGB filtering, the temporary target buffer format can be specified as YUY2 to process Y-dimension filtering (VER_FILTER_BLT). This is to avoid converting YUV to A8R8G8B8 in the 1st vertical pass to reduce the memory bandwidth and increase the pixel processing rate. This is the only special case that GPU may use YUY2 as target format.

4.3 Other Features

4.3.1 Rotation

90° / 180° / 270° / X-Flip / Y-Flip rotation is supported for all primitives.

4.3.2 Transparency Mode

For monochrome expansion:

- Opaque
- Conditional transparency. Transparent if the current pixel matches the specified value

For blits:

- Opaque
- Masked transparency. Transparent if the mask for the current pixel or pattern is zero
- Source Conditional transparency. Transparent if the source pixel is within the specified value range
- Destination Conditional transparency. Transparent if the destination pixel is not within the specified value range

4.3.3 Clipping

One clipping rectangle is supported for all bitBlit primitives.

5 DDR Controller

5.1 Overview

DDRC (DDR Controller) is a general IP which provide an interface to DDR2, DDR, mobile DDR memory. The DDRC IP is designed for SOC usage and is configurable, scalable to meet the requirement of various SOC.

Features:

- Support DDR2, DDR, mobile DDR (LPDDR) memory
- Support x16 and x32 external DDR data width
- Support clock frequency ratio – (BUS clock) : (DDR clock) = 2:1
- Support clock frequency ratio – (BUS clock) : (DDR clock) = 1:1
- Support clock-stop mode
- Support auto-refresh and self-refresh
- Support power-down mode and deep-power-down mode
- Programmable DDR timing parameters
- Programmable DDR row and column address width

5.1.1 Supported DDR SDRAM Types

In the following table, the DDR memory types in green are supported by DDRC.
Row address width 15-bit or more & Column width 11 or more are not supported.

64Mb			
Configuration	16Mb x 4	8Mb x 8	4Mb x 16
Number of Banks	4	4	4
Row address width	12	12	12
Column address width	10	9	8
128Mb			
Configuration	32Mb x 4	16Mb x 8	8Mb x 16
Number of Banks	4	4	4
Row address width	12	12	12
Column address width	11	10	9
256Mb			
Configuration	64Mb x 4	32Mb x 8	16Mb x 16
Number of Banks	4	4	4
Row address width	13	13	13

Column address width	11	10	9
512Mb			
Configuration	128Mb x 4	64Mb x 8	32Mb x 16
Number of Banks	4	4	4
Row address width	13	13	13
Column address width	12	11	10
1Gb			
Configuration	256Mb x 4	128Mb x 8	64Mb x 16
Number of Banks	4	4	4
Row address width	14	14	14
Column address width	12	11	10

5.1.2 Supported DDR2 SDRAM Types

In the following table, the DDR2 memory types in green are supported by DDRC.
All x4 (memory data width is 4-bit) devices are not supported.
Row address width 15-bit or more & Column width 11 or more are not supported.

256Mb			
Configuration	64Mb x 4	32Mb x 8	16Mb x 16
Number of Banks	4	4	4
Row address width	13	13	13
Column address width	11	10	9
512Mb			
Configuration	128Mb x 4	64Mb x 8	32Mb x 16
Number of Banks	4	4	4
Row address width	14	14	13
Column address width	11	10	10
1Gb			
Configuration	256Mb x 4	128Mb x 8	64Mb x 16
Number of Banks	8	8	8
Row address width	14	14	13
Column address width	11	10	10
2Gb			
Configuration	512Mb x 4	256Mb x 8	128Mb x 16
Number of Banks	8	8	8
Row address width	15	15	14
Column address width	11	10	10

5.1.3 Supported LPDDR SDRAM Types

In the following table, the LPDDR memory types in green are supported by DDRC.
Row address width 15-bit or more & Column width 11 or more are not supported.

128Mb			
Configuration		8Mb x 16	4Mb x 32
Number of Banks		4	-
Row address width		12	-
Column address width		9	-
256Mb			
Configuration		16Mb x 16	8Mb x 32
Number of Banks		4	4
Row address width		13	12
Column address width		9	9
512Mb			
Configuration		32Mb x 16	16Mb x 32
Number of Banks		4	4
Row address width		13	13
Column address width		10	9
1Gb			
Configuration		64Mb x 16	32Mb x 32
Number of Banks		8	8
Row address width		14	13
Column address width		10	10

5.1.4 Block Diagram

Following figure shows the functional block diagram of DDRC.

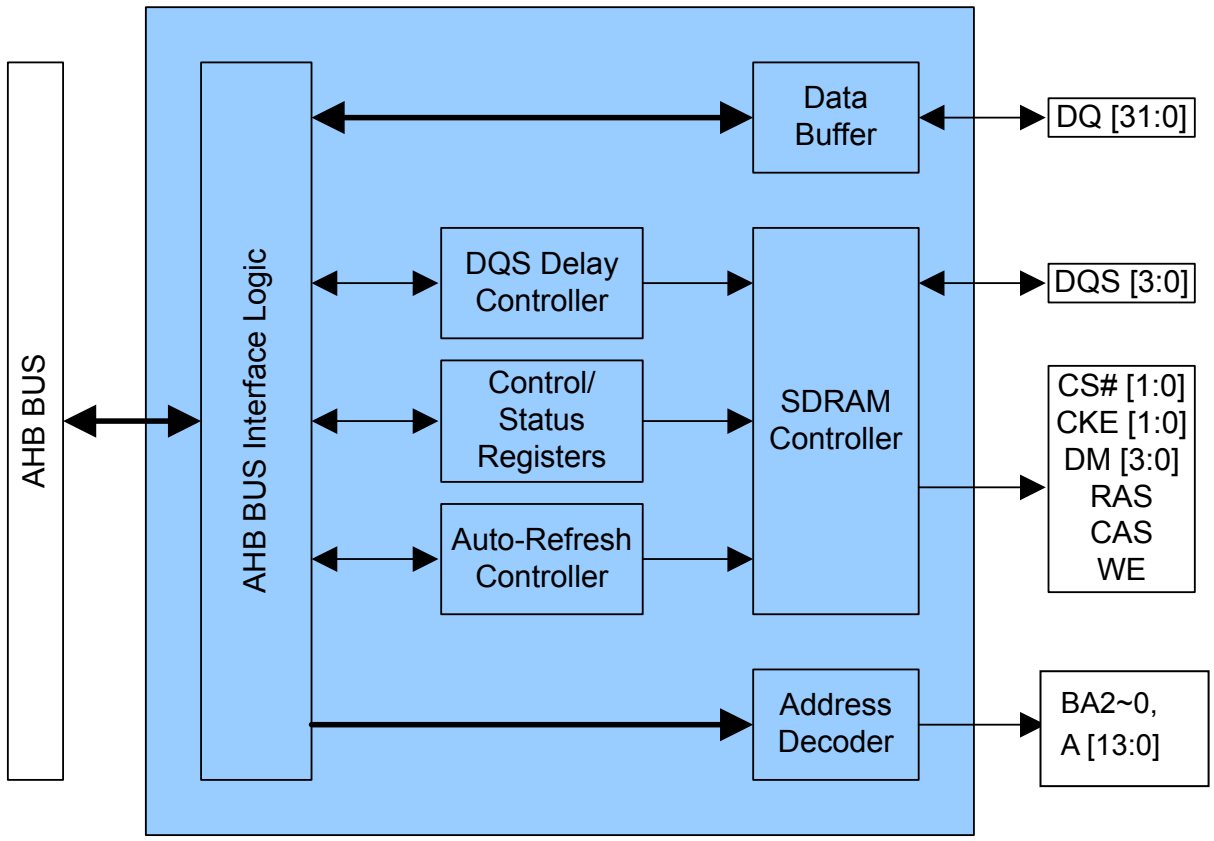


Figure 5-1 DDRC block diagram

5.1.5 Pin Description

Pin Name	I/O	Signal	Description
Data Bus	I/O	DQ31 – DQ0	Data I/O.
Data strobe	I/O	DQS3 –DQS0	Data strobe signal.
Address bus	O	A16–A0	Address output. A16~A14 represent BA2~BA0.
chip select 0	O	CS0#	Chip select signal 0 that indicates the SDRAM bank being accessed.
chip select 1	O	CS1#	Chip select signal 1 that indicates the SDRAM bank being accessed.
Write enable	O	WE#	write enable signal.
Column address strobe	O	CAS#	SDRAM column address strobe signal.
Row address strobe	O	RAS#	SDRAM row address strobe signal.
Data mask 0	O	DM0#	For SDRAM, D7–D0 input mask.
Data mask 1	O	DM1#	For SDRAM, D15–D8 input mask.
Data mask 2	O	DM2#	For SDRAM , D23–D16 input mask.
Data mask 3	O	DM3#	For SDRAM , D31-24 input mask.
Clock enable	O	CKE	Enable the SDRAM clock.
Clock	O	CKO/CKO_	SDRAM clock.

NOTE: SDRAM represent DDR/DDR2/LPDDR (mobile-DDR) SDRAM.

5.2 Register Description

Table 5-1 DDRC Register lists the registers of DDR Controller. All of these registers are 32bit, and each bit of the register represents or controls one interrupt source that list in Table 5-1 DDRC Register.

All DDRC register 32bit access address is physical address.

The physical address base for the address-mapped registers of DDRC is 0x13020000.

Table 5-1 DDRC Register

Name	Address offset	Width	Access	Description
DSTATUS	0x00	32	RW	Status Register
DCFG	0x04	32	RW	DDR Configure Register
DCTRL	0x08	32	RW	DDR Control Register
DLMR	0x0C	32	RW	DDR Load-Mode-Register
DTIMING1	0x10	32	RW	DDR Timing Configure Register 1
DTIMING2	0x14	32	RW	DDR Timing Configure Register 2
DREFCNT	0x18	32	RW	Auto-Refresh Counter
DDQS	0x1C	32	RW	DDR DQS Delay Control Register
DDQSADJ	0x20	32	RW	DDR DQS Delay Adjust Register
DMMAP0	0x24	32	RW	DDR Memory CS0 Map Configure Register
DMMAP1	0x28	32	RW	DDR Memory CS1 Map Configure Register
DDELAYCTRL	0x2C	32	RW	DDR Memory Delay Control Register

5.2.1 DSTATUS

DSTATUS 0x1302000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																								ENDIAN	MISS	DPDN	PDN	AREF	SREF	CKE1	CKE0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits 31~8 : Reserved. Writes to these bits have no effect and always read as 0.

ENDIAN : Read-only, indicate the data endian status.

Bit [7]	Description	Remark
0	Little data Endian	(reset value)
1	Big data Endian	

MISS : Indicate the bus memory-operation address out of DDRC memory mapping area. (this bit can be written)

Bit [6]	Description	Remark
0	No operation miss DDRC memory mapping	(reset value)
1	At last one operation miss DDRC memory mapping	

DPDN : Indicate the deep-power-down status of DDR memory.

Bit [5]	Description	Remark
0	DDR memory is NOT in deep-power-down state	(reset value)
1	DDR memory is in deep-power-down state	

PDN : Indicate the power-down status of DDR memory.

Bit [4]	Description	Remark
0	DDR memory is NOT in power-down state	(reset value)
1	DDR memory is in power-down state	

AREF : Indicate the auto-refresh status of DDR memory.

Bit [3]	Description	Remark
0	DDR memory is NOT in auto-refresh state	(reset value)
1	DDR memory is in auto-refresh state	

SREF : Indicate the self-refresh status of DDR memory.

Bit [2]	Description	Remark
0	DDR memory is NOT in self-refresh state	(reset value)
1	DDR memory is in self-refresh state	

CKE1 : Indicate the CKE1 Pin status of DDR memory.

Bit [1]	Description	Remark
0	CKE1 Pin is low	(reset value)
1	CKE1 Pin is high	

CKE0 : Indicate the CKE0 Pin status of DDR memory.

Bit [0]	Description	Remark
0	CKE0 Pin is low	(reset value)
1	CKE0 Pin is high	

5.2.2 DCFG

Configure the external memory, once set; this register can NOT be changed on-the-fly.

DCFG 0x13020004

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SDR	Reserved	REPRI	ROW1		COL1		BA1	IMBA	DQSMD	BTRUN	Reserved					MISPE	TYPE			ROW0		COL0		CS1EN	CS0EN	CL				BA0	DW
RST	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits 30, 22, 20: Reserved. Writes to these bits have no effect and always read as 0.

SDR : 1: use SDR sdram; 0: use DDR/DDR2/LPDDR sdram.

RDPRI : READ command priority configure. 1: read command has a high priority.
(hardware will avoid read/write command conflict automatically)

MISPE : Miss CS protect. Set 1 to enable.
If software read (or write) a memory space which is not select by any CS, this function will return random data to a read operation (or mask write operation) to avoid system bus be locked. A CS missing flag will set in DSTATUS.

DQSMD : Dqs pin mode. (only for inner test)
1: DQS pin with pull down resist
0: DQS pin without pull down resist

BTRUN : burst terminate enable. (only for mddr /ddr1)
1: enable
0: disable

IMBA :
0: CS0, CS1 connected 2 memory chips which has same ROW, COL, BA configuration.

In this mode, ROW,COL, BA configure both two chips. ROW1,COL1,BA1 are don't care.
1: CS0, CS1 connected 2 memory chips which has different ROW, COL, BA configuration.
ROW, COL, BA refer to CS0; ROW1, COL1, BA1 refer to CS1.

MEM_TYPE : Select external memory device type.
This field is not supported by current DDRC design.

Bit [14:12]	Description	Remark
000	Normal SDR (Single-Data-Rate) SDRAM(Not support)	(reset value)
001	Mobile SDR(Not support)	
010	Normal DDR1 (Double-Data-Rate) SDRAM	
011	Mobile DDR	
100	Normal DDR2	
101	Mobile DDR2(Not support)	
110	Normal DDR3(Not support)	
111	Mobile DDR3(Not support)	

ROW0 : Row Address width. Specify the row address width of external DDR.

Bit [11:10]	Description	Remark
00	12-bit row address is used	(reset value)
01	13-bit row address is used	
10	14-bit row address is used	
11	Reserved	

COL0: Column Address width. Specify the Column address width of external DDR.

Bit [9:8]	Description	Remark
00	8-bit Column address is used	(reset value)
01	9-bit Column address is used	
10	10-bit Column address is used	
11	11-bit Column address is used	

CS1EN : DDR Chip-Select-1 Enable.
If there're ddr memory connected to ddr pin cs1, set CS1EN=1.

Bit [7]	Description	Remark
0	DDR Pin CS1 un-used	(reset value)
1	There're DDR memory connected to CS1	

CS0EN : DDR Chip-Select-0 Enable.

If there're ddr memory connected to ddr pin cs0, set CS0EN=1.

Bit [6]	Description	Remark
0	DDR Pin CS0 un-used	(reset value)
1	There're DDR memory connected to CS0	

CL : CAS Latency.

Bit [5:2]	Description	Remark
0,000	CL = 1 tCK	(reset value)
0,001	CL = 1.5 tCK	
1,001	CL = 2 tCK	
0,010	CL = 2.5 tCK	
1,010	CL = 3 tCK	
0,011	CL = 3.5tCK	
1,011	CL = 4 tCK	
0,100	CL = 4.5 tCK	
1,100	CL = 5 tCK	
0,101	CL = 5.5 tCK	
1,101	CL = 6 tCK	
0,110	CL = 6.5 tCK	
1,110	CL = 7 tCK	
others	Reserved	

BA0 : Bank Address width of DDR memory.

Bit [1]	Description	Remark
0	4 bank device, Pin ba[1:0] valid, ba[2] un-used	(reset value)
1	8 bank device, Pin ba[2:0] valid	

DW : External DDR Memory Data Width.

Specify the external DDR memory data width.

Bit [0]	Description	Remark
0	External memory data width is 16-bit	(reset value)
1	External memory data width is 32-bit	

5.2.3 DCTRL

On the positive edge of START, one command selected by CMD field will be performed.

DCTRL 0x13020008

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																ACTPD	PDT			ACTSTP	Reserved			DPD	SR	UNALIGN	ALH	Reserved	CKE	RESET	
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit 31~16, 10~7,2: Reserved. Writes to these bits have no effect and always read as 0.

ACTSTP : Active Clock-Stop.

0: Clock can be stopped only after all banks be precharged

1: Clock can be stopped with some bank's row actived

ACTPD : Active Power-Down.

Some SDRAM devices support Active-Power-Down.

By default, ACTPD=0, hardware will percharge all active banks before entering Power-Down mode, so called Precharge-Power-Down.

By setting ACTPD=1, hardware drives SDRAM into Power-Down mode without precharge all active banks, some banks are still active in Power-Down mode, so called Active-Power-Down.

Bit [15]	Description	Remark
0	Precharge all banks before entering power-down	(reset value)
1	Do not precharge all banks before entering power-down	

PDT : Power-Down Timer.

When there's no access to DDR memory for a period of time, hardware drives DDR into power-down mode to save power consumption. Hardware can exit Power-Down mode automatically when new access arrives.

If PDT=0, power-down function disabled.

if use power-down, recommend to enable it after DDR initialization finished.

Bit [14:13]	Description	Remark
000	power-down disabled, hardware never drive SDRAM into power-down mode	(reset value)
001	Enter power-down after 8 tCK idle	
010	Enter power-down after 16 tCK idle	
011	Enter power-down after 32 tCK idle	
100	Enter power-down after 64 tCK idle	
101	Enter power-down after 128 tCK idle	

110 - 111	Reserved	

SPD : Stop CKO when into Power-down mode(only use in mobile-ddr mode).
1: stop CKO
0: not stop CKO

SR : Software drive external DDR device entering Self-Refresh mode.
Software set SR=1 drive external DDR device entering self-refresh mode;
Software set SR=0 drive external DDR device exiting self-refresh mode;
In this mode, the CK to external DDR device would be stopped during self-refresh period;
But the clock supply to ddr_controller logic would not stop.
Software can read & write ddr_controller registers in this mode.
Software can NOT read or write memory data in this mode.

NOTE: Since ddr_controller registers are accessed via AXI bus interface, software must guarantee that there's no memory access during self-refresh mode. Otherwise, software can NOT exit this mode, **system would hangup!!**

Bit [5]	Description	Remark
0	Drive external DDR device entering self-refresh mode	(reset value)
1	Drive external DDR device exiting self-refresh mode	

DPD: Software drive external Mobile DDR device entering Deep-Power-Down mode.
Software set DPD = 1 drive external Mobile DDR device entering Deep-Power-Down mode instead of Power-Down mode, when there's no access to DDR memory for a period of time, So you must first enable Power-Down mode (refer to PDT).
Software need to reset DDR controller and re-do a complete initial process to exit Deep-Power-Down mode.
When external device go to Deep-Power-Down mode, it will lose all data store in memory and registers.
The memory chip will disable inner power support to save power.

UNALIGN : Enable unaligned transfer on AXI BUS.

Bit [4]	Description	Remark
0	Disable unaligned transfer on AXI BUS	(reset value)
1	Enable unaligned transfer on AXI BUS	

ALH : Advanced Latency Hiding.
This is a test purpose register.
Some latency timings can be hidden in special cases.

Bit [3]	Description	Remark
0	Disable ALH	(reset value)

1	Enable ALH	

CKE : Control the status of CKE pin.

Write CKE=1 can set CKE pin to HIGH state.

Write CKE=0 would be ignored.

The default value of CKE Pin is low;

CKE0,1 Pins status is represented by DDR_STATUS register.

CAUTION: This register is used only for DDR initializing sequence; software can NOT update this register when DDR memory is in normal working mode.

Bit [1]	Description	Remark
0	Not set CKE Pin High	(reset value)
1	Set CKE Pin HIGH	

RESET : Module reset for ddr_controller.

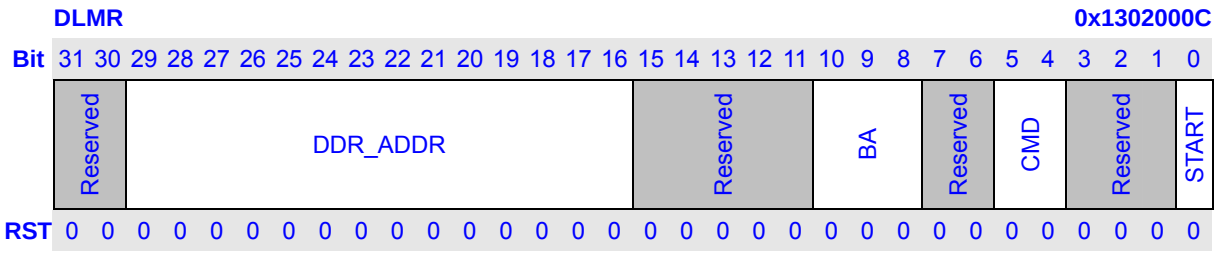
Software reset ddr_controller by setting RESET bit high. Then, software end reset by setting RESET bit low.

Bit [0]	Description	Remark
0	End resetting ddr_controller	(reset value)
1	Resetting ddr_controller	

5.2.4 DLMR

DLMR register is used for initializing the DDR SDRAM memory device.

On the positive edge of START, one command selected by CMD field will be performed.



Bit 31~30, 15~11, 7~6, 3~1: Reserved. Writes to these bits have no effect and always read as 0.

DDR_ADDR : When performing a DDR command, DDR_ADDR[13:0] corresponding to external DDR address Pin A[13:0]; DDR_ADDR[15:14] are reserved.

Bit [29:16]	Description	Remark
0000_0000	corresponding to external DDR address Pin A[13:0]	(reset value)

BA : Bank Address.

When performing a DDR command, BA[2:0] corresponding to external DDR address Pin BA[2:0].

Bit [10:8]	Description	Remark
000	corresponding to external DDR address Pin BA[2:0]	(reset value)

CMD : Select command to process when setting START from low to high.

On the positive edge of START, one of the following commands will be performed.

Bit [5:4]	Description	Remark
00	Precharge one bank / All banks (dependent field : BA, DDR_ADDR)	(reset value)
01	Auto-Refresh	
10	Load Mode Register (dependent field : BA, DDR_ADDR)	
11	Reserved	

START : Start perform a command to external DDR memory.

The command is performed on the positive edge of START; Hardware will clear START bit to zero when command issued out to external DDR memory.

Write 0 to START will be ignored and take no effect;

START=1 means hardware is busy executing current command and can NOT accept new command;

Software must check START=0 before writing 1 to START.

Bit [0]	Description	Remark
0	No command is performed	(reset value)
1	On the positive edge of START, perform a command defined by CMD field	

5.2.5 DTIMING1,2 (DDR Timing Config Register 1, 2)

The timing parameters are identical to the JEDEC DDR Specification.

DTIMMING1 **0x13020010**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	tRAS				Reserved		tRTP		Reserved		tRP		Reserved		tRCD		tRC				Reserved		tRRD		Reserved		tWR		Reserved		tWTR	
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits 27~26, 23, 19, 11~10, 7, 3-2: Reserved. Writes to these bits have no effect and always read as 0.

tRAS : ACTIVE to PRECHARGE command period.

tRAS defines the ACTIVE to PRECHARGE command period to the same bank.

Bit [31:28]	Description	Remark
0000	1 tCK	(reset value)
0001	3 tCK	
0010	5 tCK	
0011	7 tCK	
… …	**… 2 * tRAS + 1 …**	
1101	27 tCK	
1110	29 tCK	
1111	31 tCK	

tRTP : READ to PRECHARGE command period.

Bit [25:24]	Description	Remark
00	1 tCK	(reset value)
01	2 tCK	
10	3 tCK	
11	4 tCK	

tRP : PRECHARGE command period.

tRP defines the PRECHARGE to next command period to the same bank.

Bit [22:20]	Description	Remark
000	1 tCK	(reset value)
001	2 tCK	
010	3 tCK	
011	4 tCK	
100	5 tCK	

101	6 tCK	
110	7 tCK	
111	8 tCK	

tRCD : ACTIVE to READ or WRITE command period.

tRCD defines the ACTIVE to READ/WRITE command period to the same bank.

Bit [18:16]	Description	Remark
000	1 tCK	(reset value)
001	2 tCK	
010	3 tCK	
011	4 tCK	
100	5 tCK	
101	6 tCK	
110	7 tCK	
111	8 tCK	

tRC : ACTIVE to ACTIVE command period.

tRC defines the ACTIVE to ACTIVE command period to the same bank.

Since tRCD + read/write-time + tRP > tRC always match, in most cases, tRC can be disabled.

Bit [15:12]	Description	Remark
0000	1 tCK	(reset value)
0001	3 tCK	
0010	5 tCK	
0011	7 tCK	
… …	**… 2 * tRC + 1 …**	
1101	27 tCK	
1110	29 tCK	
1111	31 tCK	

tRRD : ACTIVE bank A to ACTIVE bank B command period.

tRRD defines the ACTIVE to ACTIVE command period to **different** banks.

Bit [9:8]	Description	Remark
00	Disable tRRD counter	(reset value)
01	2 tCK	
10	3 tCK	
11	4 tCK	

tWR : WRITE Recovery Time defined by register MR of DDR2 memory.

Bit [6:4]	Description	Remark
000	1 tCK	(reset value)
001	2 tCK	
010	3 tCK	
011	4 tCK	
100	5 tCK	
101	6 tCK	
110 - 111	Reserved	

tWTR : WRITE to READ command delay.

Bit [1:0]	Description	Remark
00	1 tCK	(reset value)
01	2 tCK	
10	3 tCK	
11	4 tCK	

DTIMING2 **0x13020014**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													tCKE			tRFC				tMINSR				Reserved				tXP				Reserved		tMRD		
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits 31~19, 7, 3~2 : Reserved. Writes to these bits have no effect and always read as 0.

tCKE : minimum CKE pulse width.

tCKE define the minimum CKE pulse width, include high level and low level.

Bit [18:16]	Description	Remark
000	1 tCK	(reset value)
001	2 tCK	
010	3 tCK	
011	4 tCK	
100	5 tCK	
101	6 tCK	
110	7 tCK	
111	8 tCK	

tRFC : AUTO-REFRESH command period.

tRFC defines the minimum delay after an AUTO-REFRESH command. During tRFC period,

no command can be issued to DDR memory.

Delay Time = **2 * tRFC + 1.**

Bit [15:12]	Description	Remark
0000	1 tCK	(reset value)
0001	3 tCK	
0010	5 tCK	
0011	7 tCK	
… …	**… 2 * tRFC + 1 …**	
1101	27 tCK	
1110	29 tCK	
1111	31 tCK	

* tCK – one DDR memory clock cycle, typical tCK value is 7.5 ns (133MHz clock).

tMINSR : Minimum Self-Refresh / Deep-Power-Down time.

After DDR memory turns into Self-Refresh or Deep-Power-Down mode, it will NOT exit until tMINSR condition meets.

Delay Time = tMINSR * 8 + 1.

Bit [11:8]	Description	Remark
0000	1*8 + 1 tCK	(reset value)
0001	2*8 + 1 tCK	
0010	3*8 + 1 tCK	
0011	4*8 + 1 tCK	
… …	**… tMINSR * 8 + 1 …**	
1101	14*8 + 1 tCK	
1110	15*8 + 1 tCK	
1111	16*8 + 1 tCK	

tXP : EXIT-POWER-DOWN to next valid command period.

tXP defines the EXIT-POWER-DOWN to next valid command period to all banks.

Bit [6:4]	Description	Remark
000	1 tCK	(reset value)
001	2 tCK	
010	3 tCK	
011	4 tCK	
100	5 tCK	
101	6 tCK	
110	7 tCK	
111	8 tCK	

tMRD : Load-Mode-Register to next valid command period.

tMRD defines the Load-Mode-Register to next valid command period.

Bit [1:0]	Description	Remark
00	1 tCK	(reset value)
01	2 tCK	
10	3 tCK	
11	4 tCK	

5.2.6 DREFCNT (DDR Auto-Refresh Counter)

DREFCNT **0x13020018**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								CON								CNT								Reserved				CLK_DIV			REF_EN
RST	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits 31~24, 7-4 : Reserved. Writes to these bits have no effect and always read as 0.

CON : A constant value used to compare with the CNT value.

After reset, CON=0xFF and CNT=0x00;

It is not recommended to set CON=0x00.

CNT : 8-bit counter; When the value of CNT match the value of CON, flag bit EQU is set high and an auto-refresh command will be issued to DDR memory. READ only.

CLK_DIV : Clock Divider.

Divide the dclk to generate a lower frequency of clock to drive the auto-refresh counter. This helps to save power consumption.

When the DDR memory is in self-refresh mode or in deep-power-down mode, disable the clock of auto-refresh counter can save power consumption. Future more, the module clock (ahb_clk & dclk) to DDRC can also be stopped.

Bit [3:1]	Description	Remark
000	dclk / 16	(reset value)
001	dclk / 32	
010	dclk / 64	
011	dclk / 128	
100	dclk / 256	
101	dclk / 512	
110	dclk / 1024	
111	reserved	

REF_EN : Enable Refresh Counter.

Software set REF_EN=1 right after initialize ddr memory.

Bit [29]	Description	Remark
0	Enable auto-refresh counter	(reset value)
1	Disable auto-refresh counter	

5.2.7 DDQS (DDR DQS Delay Control Register)

DDRC contains an on-chip DLL to control the DQS Delay for read data and write data.

DDQS **0x1302001C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		ERROR	READY	Reserved		SRDET	DET	AUTO	Reserved	CLKD						Reserved		WDQS						Reserved		RDQS					
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits 31~30, 27-26, 22, 15-14, 7-6: Reserved. Writes to these bits have no effect and always read as 0.

ERROR : ahb_clk Delay Detect ERROR, read-only.

When hardware detect one ahb_clk cycle delay failed, ERROR is set high;

ERROR is cleared zero when a new detection starts;

ERROR is valid only when READY=1.

Bit [29]	Description	Remark
0	delay detect success	(reset value)
1	delay detect failed	

READY : ahb_clk Delay Detect READY, read-only.

When hardware detect complete, ERROR is set high.

READY is cleared zero when a new detection starts.

Bit [28]	Description	Remark
0	delay detect NOT complete	(reset value)
1	delay detect complete	

SRDET : DDRC auto re-detect and set (if auto = = 1) delay line after clock change.

It will consume extra times in clock change process.

Bit [25]	Description	Remark
0	not enable	(reset value)

1	enable	

AUTO : Hardware auto-detect & set delay line.

Bit [23]	Description	Remark
0	Hardware do NOT auto-set delay line	(reset value)
1	Hardware auto-set delay line after detect success	

DET : Start delay detecting.

Write 1 to START bit starts a new delay detect progress.

When delay detect complete, START is cleared zero by hardware.

START can be used as the BUSY flag. When START=1, it is busy.

Bit [24]	Description	Remark
0	No operation	(reset value)
1	Delay detect in progress, busy	

CLKD : Indicate the number of delay elements needed to delay ¼ Tck.

CLKD is a reference value for setting WDQS and RDQS.

CLKD is set when DLL detection finished.

The range of CLKD: [0, +63].

WDQS : Set the number of delay elements used on the write DQS delay-line.

When WDQS increase one, the delay value of write DQS increase approximately 0.1 ns .

The range of WDQS, RDQS: [0, +63].

NOTE: The delay value of each delay element depends on the technology and the structure of the delay cell; "0.1 ns" is just an example at .18 technology.

RDQS : Set the number of delay elements used on the read DQS delay-line.

When RDQS increase one, the delay value of read DQS increase approximately 0.1 ns.

5.2.8 DDQSADJ (DDR DQS Delay Adjust Register)

DDQSADJ **0x13020020**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DQSCON																Reserved		WSIGN	WDQS							Reserved		RSIGN	RDQS				
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits 15~13, 7-6: Reserved. Writes to these bits have no effect and always read as 0.

DQSCON: DQS detect looping counter threshold. When inner counter equal to

DQSCON, then trigger a DQS detect operation to auto adjust DQS delay line. This inner counter use AUTO_REFRESH's divided clock (refer to DREFCNT). When DQSCON set to 0, this function be disabled.

WSIGN, RSIGN: The adjust value's sign. 0: plus; 1: minus.

WDQS, RDQS: The adjust value for WRITE and READ DQS delay.

WDQS, RDQS can be either positive or negative number.

For negative number, it should be in "complemental code" format;

The range of WDQS, RDQS : [-16, +15].

For READ: DQS_Delay = DDQS.RDQS +/- DDQSADJ.RDQS.

For WRITE: DQS_Delay = DDQS.WDQS +/-DDQSADJ.WDQS.

5.2.9 DMMAP0,1 (DDR Memory Map Config Register)

The physical base address and size of external DDR Memory can be configured by DMMAP register.

The size of external DDR Memory must be: **2^(24+n), n=0, 1, 2, 3, …**

When the following equation is met:

(AXI_BUS_Address[31:24] & MASK[7:0]) == BASE

The DDR Memory is selected.

DMMAP0,1 **0x13020024, 0x13020028**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																	BASE								MASK							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0

Bits 31~16: Reserved. Writes to these bits have no effect and always read as 0.

BASE : base address.

MASK : address mask.

Examples:

1 DDR address space in system memory : 0x2000_0000 ~ 0x2FFF_FFFF (256MB)
BASE=0x20 MASK=0xF0.

2 DDR address space in system memory : 0x5000_0000 ~ 0x57FF_FFFF (128MB)
BASE=0x50 MASK=0xF8.

NOTES:

1 If DDRC is disabled, please set DMMAP=0x0000_FF00 (reset value).

5.2.10 DDELAYCTRL

This register can be configured at any time, but effect after a Auto-Refresh command occur.

DDELAYCTRL1 **0x1302002C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												TSEL		MSEL		HL	QUAR	Reserved							MAUTO	MSIGN	MASK_DELAY_SEL_ADJ						
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DDELAYCTRL2 **0x13020030**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																										MASK_DELAY_SEL						
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

MSEL : Mask delay select.

Bit [17:16]	Description	Remark
00	No delay	(reset value)
01	delay 1 tCK	
10	delay 2 tCK	
11	delay 3 tCK	

HL : Half clock delay select.

Adjust MSEL delay 1/2 tCK

0: delay no change

1: delay reduced 1/2 tCK

QUAR : Quarter clock delay select.

Adjust MSEL delay 1/4 tCK

0: delay no change

1: delay add 1/4 tCK

Msel[1]	Msel[0]	HL	QUAR	DELAY
0	0	1	0	- 0.5 tCK
0	0	1	1	- 0.25 tCK
0	0	0	0	0 tCK
0	0	0	1	0.25 tCK
0	1	1	0	0.5 tCK
0	1	1	1	0.75 tCK
0	1	0	0	1 tCK
0	1	0	1	1.25 tCK
1	0	1	0	1.5 tCK
1	0	1	1	1.75 tCK
1	0	0	0	2 tCK
1	0	0	1	2.25 tCK
1	1	1	0	2.5 tCK
1	1	1	1	2.75 tCK
1	1	0	0	3 tCK
1	1	0	1	3.25 tCK

TSEL : Read delay select.

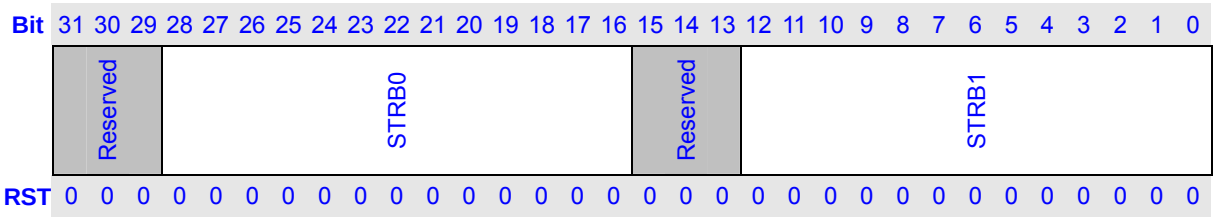
Bit [19:18]	Description	Remark
00	No delay	(reset value)
01	delay 1 tCK	
10	delay 2 tCK	
11	delay 3 tCK	

MASK_DELAY_SEL_ADJ , MASK_DELAY_SEL, MSIGN :

Delay configure for a inner mask, use as DDQS and DDQSADJ.

5.2.11 DSTRB

DSTRB **0x1000100C**



Inner usage.

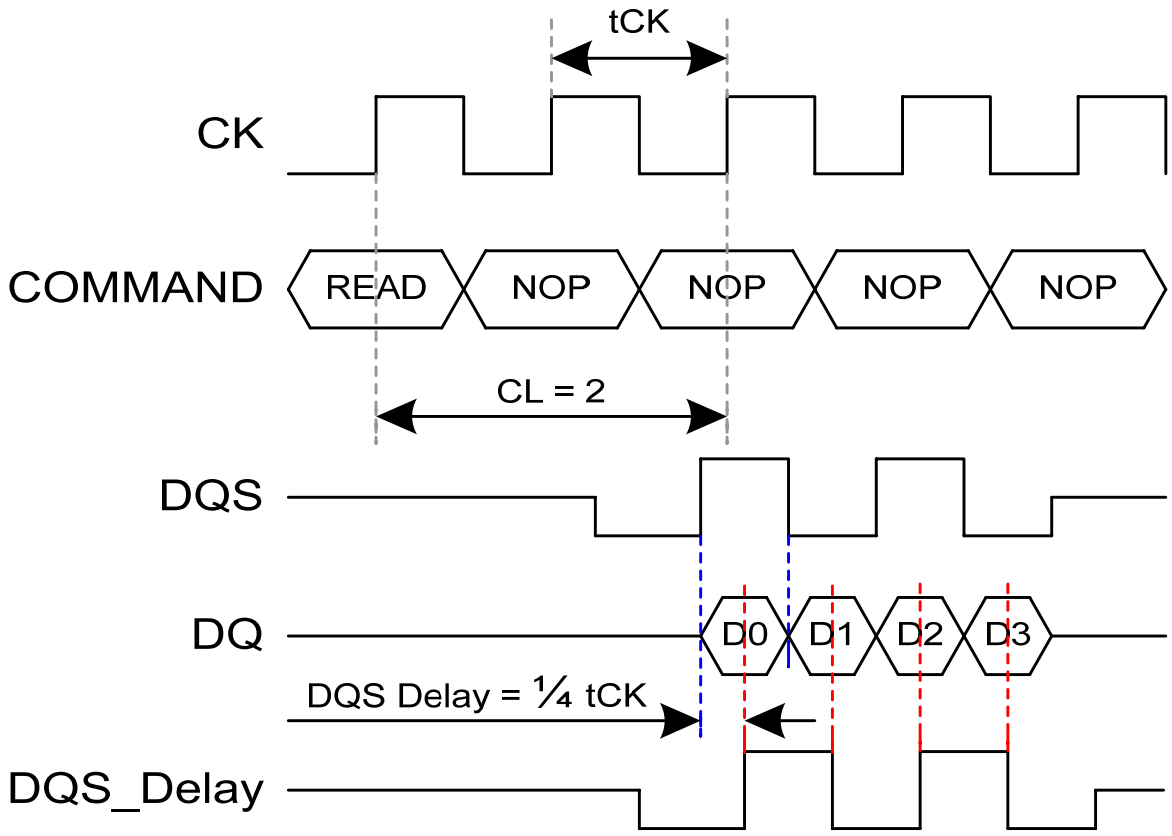
Just for Video Decoder.

5.3 Functional Description

5.3.1 DDR DQS Delay Detect-and-Set Processing

Sub-module “DQS Delay Controller” of DDRC generates DQS_Delay signal to capture data;

The following figure illustrates the DQS_Delay in READ case. The DQS_Delay in WRITE case is similar to READ case.



DQS_Delay is used to capture DQ by its posedge and negedge. DQS_Delay signal is generated by the “DQS Delay Controller” sub-module of DDRC.

There’re delay elements in “DQS Delay Controller”; each delay element adds approximately 0.1 ns delay value to between its input and output. The number of delay elements used can be controlled by DDQS.RDQS and DDQS.WDQS.

Note that the delay value of each delay element changes according to the temperature and voltage. It is recommended to adjust the value of DDQS.RDQS and DDQS.WDQS periodically according to the

temperature change.

The “DQS Delay Controller” provides a mechanism to automatically adjust the DDQS value periodically. Alternatively, this function can be disabled to enable fully manual control.

5.3.2 Detect dclk delay

Setting DDQS.START=1 and DDQS.AUTO=0, hardware do the detect processing one time.
Setting DDQS.START=1 and DDQS.AUTO=1, hardware do the detect-and-set processing one time.
Setting DDQS.START=1 and DDQS.AUTO=2, hardware periodically do the detect-and-set processing after each auto-refresh command.

The detection result stores in DDQS.CLKD. DDQS.CLKD indicates the delay value of half dclk clock cycle (frequency Fdclk=F(DDR CK)). Thus, the delay value:

DQS_Delay = ¼ tCK = ½ DDQS.CLKD <1>

Delay DQS by ¼ tCK means to the ideal case. Actually, there’re always a gap between the ideal value and the real value. The gap comes from many factors such as IC manufacture processing, PCB layout, noise, etc. So, a revised parameter is introduced to equation <1>:

For READ, DQS_Delay = ½ DDQS.CLKD + DDQSADJ.RDQS <2>
For WRITE, DQS_Delay = ½ DDQS.CLKD + DDQSADJ.WDQS <3>

DDQSADJ can be either positive or negative number to add or sub value from DQS_Delay.
After reset, DDQSADJ.RDQS/WDQS = 0.

5.3.3 Set DDQS.RDQS and DDQS.WDQS

When hardware complete a detect processing:
If DDQS.AUTO=0, hardware do NOT set DDQS.RDQS and DDQS.WDQS;
If DDQS.AUTO=1 or 2, hardware set DDQS.RDQS and DDQS.WDQS according to equation <2> and <3>.

5.3.4 Manual Detect-and-Set Processing

DQS delay value can be set manually.

Step 1: Software set DDQS.RDQS and DDQS.WDQS.

Step 2: Software do write- and-read test on DDR Memory, then compare the read data with the write data.

Step 3: Repeat step 1 and 2.

When the tests complete, software can choose the fittest value for RDQS and WDQS.

5.3.5 Handling the DQS delay detection "ERROR"

The number of delay elements for detection dclk : 256;
The number of delay elements for RDQS : 128;
The number of delay elements for WDQS : 128;

DDRC can't do the detect processing successfully when:

Tmlck > 25.6 ns (256 x 0.1 ns = 25.6 ns)
Or: Fdclk < 40 MHz

According to JEDEC DDR Specification:
For normal DDR, CK > 83 MHz;
For mobile DDR, CK > 0 MHz; there have no requirement for the lower range of CK;
In case detection failed, hardware set RDQS and WDQS with max number if DDQS.AUTO ≠ 0.

5.3.6 DDRC and DDR2 Memory Initialization Sequence

5.3.6.1 Example 1

One 512Mb x16 DDR2 device connected on CS0;
No memory device connected on CS1;
DCK = 133 MHz, CL = 3

1 After system reset, wait system clock stable before initialize ddrc.
2 Configure the Clock-Control module for ddrc clocks.
3 DDR Memory device need at least 200us initialization time after power-on before it can accept any command.

```
//-----------------------------------------------------------------------------------------------------
// INIT DDRC
//-----------------------------------------------------------------------------------------------------
```

4 Configure DCFG = 0x.
5 Configure DTIMING1 = 0x.
6 Configure DTIMING1 = 0x.
7 Configure DMMAP0 = 0x.
8 Configure DMMAP1 = 0x0000FF00.

```
//-----------------------------------------------------------------------------------------------------
// INIT DDR memory device
//-----------------------------------------------------------------------------------------------------
```

9 Set CKE Pin HIGH : Configure DCTRL = 0x00000002.
10 PRECHAREG-ALL : Configure DCTRL = 0x.
11 Load-Mode-Register EMR2 : Configure DCTRL = 0x.
12 Load-Mode-Register EMR3 : Configure DCTRL = 0x.
13 Load-Mode-Register EMR1 : Configure DCTRL = 0x.

14 Load-Mode-Register MR with DLL reset : Configure DCTRL = 0x.
15 PRECHAREG-ALL : Configure DCTRL = 0x.
16 AUTO-REFRESH : Configure DCTRL = 0x.
17 AUTO-REFRESH : Configure DCTRL = 0x.
18 Load-Mode-Register MR with DLL reset end : Configure DCTRL = 0x.
19 Load-Mode-Register EMR1 with OCD default : Configure DCTRL = 0x.
20 Load-Mode-Register EMR1 with OCD exit : Configure DCTRL = 0x.
21 Wait at least 200 tCK before next step.

//----------

// Enable Refresh Counter

//----------

22 Enable Refresh Counter : Configure DREFCNT = 0x.
23 AUTO-REFRESH : Configure DCTRL = 0x.

//----------

// DQS Delay Detect

//----------

24 Configure DDQSADJ = 0x.
25 Configure DDQS = 0x.
26 Read register DDQS.
27 configure TSEL form min to max, under each value of TSEL, do 28.
28 Configure {MSEL, HL, QUAR} register, form min delay to max delay (relate to 1.2.7). You need write/read some data by CPU or DMA or other device, to check if the sdram work properly. During this process, record the pass configure, you may found there has several configure pass the test, then chose the one that TSEL min &
{MSEL, HL, QUAR} min passed <= {MSEL,HL,QUAR} <= {MSEL, HL, QUAR}max passed.

//----------

// END INITIALIZING SEQUENCE

//----------

5.4 Change Clock Frequency

To save power consumption, the system clock frequency may be changed frequently according to the application. There're 3 ways to change the clock frequency:

5.4.1 Clock-Stop Mode(only in Mobile-ddr)

CPM will auto drive DDRC to clock-stop mode, when use mobile-ddr.
How to change clock, relate CPM spec.

5.4.2 Manually SELF-REFRESH Mode

DDR can stay in SELF-REFRESH & DEEP-POWER-DOWN mode for a long period of time. System clock frequency can be changed during this time. Even more, the clocks to DDRC module can also be stopped to save power-consumption.
Reference Sequence:

1 Manually issue SELF-REFRESH command to DDR.
2 Change relates register in CPM.
3 Change system clock frequency.
4 Drive DDR exit SELF-REFRESH mode.

5.4.3 CPM driven SELF-REFRESH Mode

CPM will auto drive DDRC to self-refresh mode, when use ddr2, ddr1.
How to change clock, relate CPM spec.

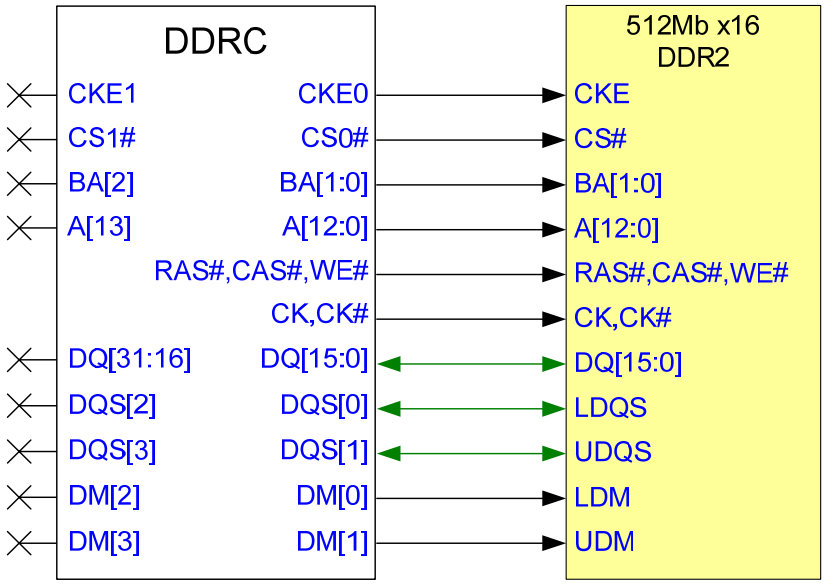
5.5 Data Endian

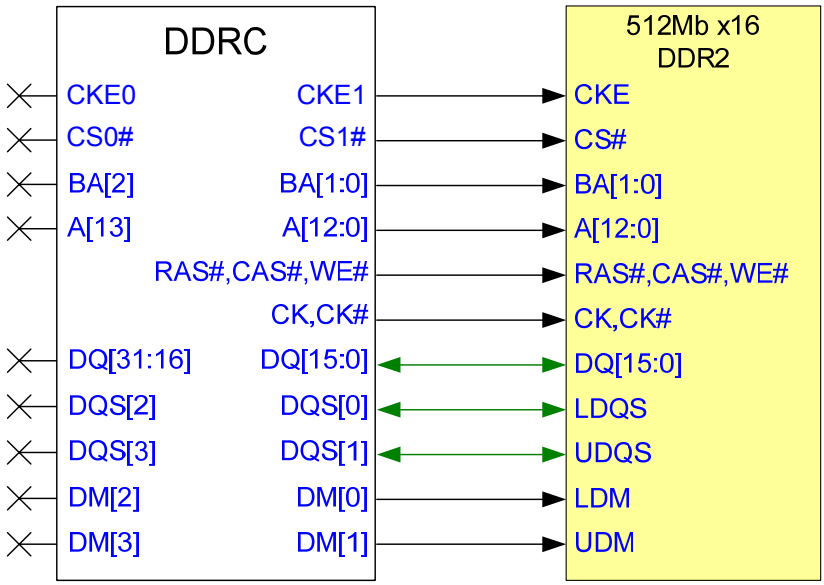
Fix to little Endian

5.6 DDR Connection Diagrams

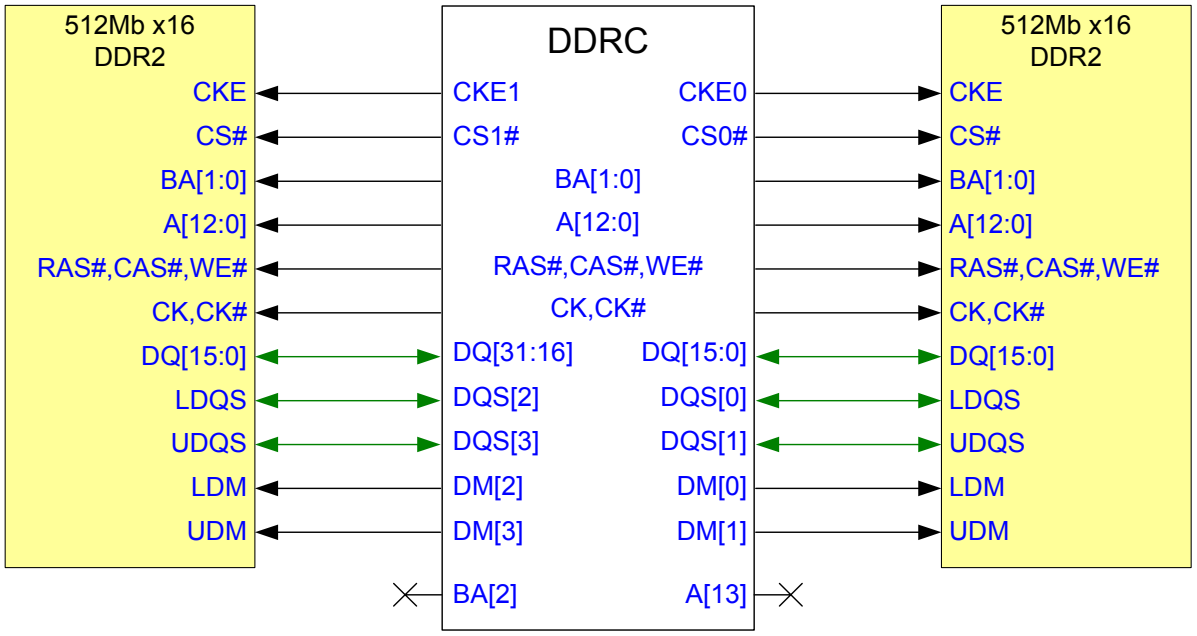
The following diagrams give examples on the connection to external DDR2 devices.
Note not all the possible connections are listed.

5.6.1 Connection to one 512Mb x16 DDR2 device





5.6.2 Connection to two 512Mb x16 DDR2 devices



6 External Memory Controller

6.1 Overview

The External Memory Controller (EMC) divides the off-chip memory space and outputs control signals complying with synchronous DRAM and support direct connection to this processor.

- SDRAM Interface
  - Support 2 chip selection DCS0# and DCS1#
  - Support both 32-bit and 16-bit bus width
  - Support both two-bank and four-bank type SDRAM
  - Support burst operation
  - Support both auto-refresh and self-refresh functions
  - The size and base address of each bank is configurable
  - Multiplexes row/column addresses according to SDRAM capacity
  - Controls timing of SDRAM direct-connection control signals according to register setting
  - Supports power-down mode to minimize the power consumption of SDRAM
  - Support page mode

6.2 Pin Description

Following table lists the EMC pins.

Table 6-1 EMC Pin Description

Pin Name	I/O	Signal	Description
Data Bus	I/O	D31 – D0	Data I/O.
Address bus	O	A25–A0	Address output.
SDRAM chip select	O	DCS0#	Chip select signal that indicates the SDRAM bank being accessed.
SDRAM chip select	O	DCS1#	Chip select signal that indicates the SDRAM bank being accessed.
Column address strobe	O	CAS#	SDRAM column address strobe signal.
Row address strobe	O	RAS#	SDRAM row address strobe signal.
Read/write	O	RD/WR#	Data bus direction designation signal. Also used as SDRAM write enable signal.
Byte enable 0	O	DQM0 /	For SDRAM, D7–D0 selection signal.
Byte enable 1	O	DQM1/	For SDRAM, D15–D8 selection signal.
Byte enable 2	O	DQM2 /	For SDRAM , D23–D16 selection signal.
Byte enable 3	O	DQM3	For SDRAM, D31–D24 selection signal.
SDRAM Clock enable	O	CKE	Enable the SDRAM clock.
SDRAM Clock	O	CKO	SDRAM clock.

6.3 Physical Address Space Map

Both virtual spaces and physical spaces are 32-bit wide in this architecture. Virtual addresses are translated by MMU into physical address which is further divided into several partitions for static memory, SDRAM, and internal I/O.

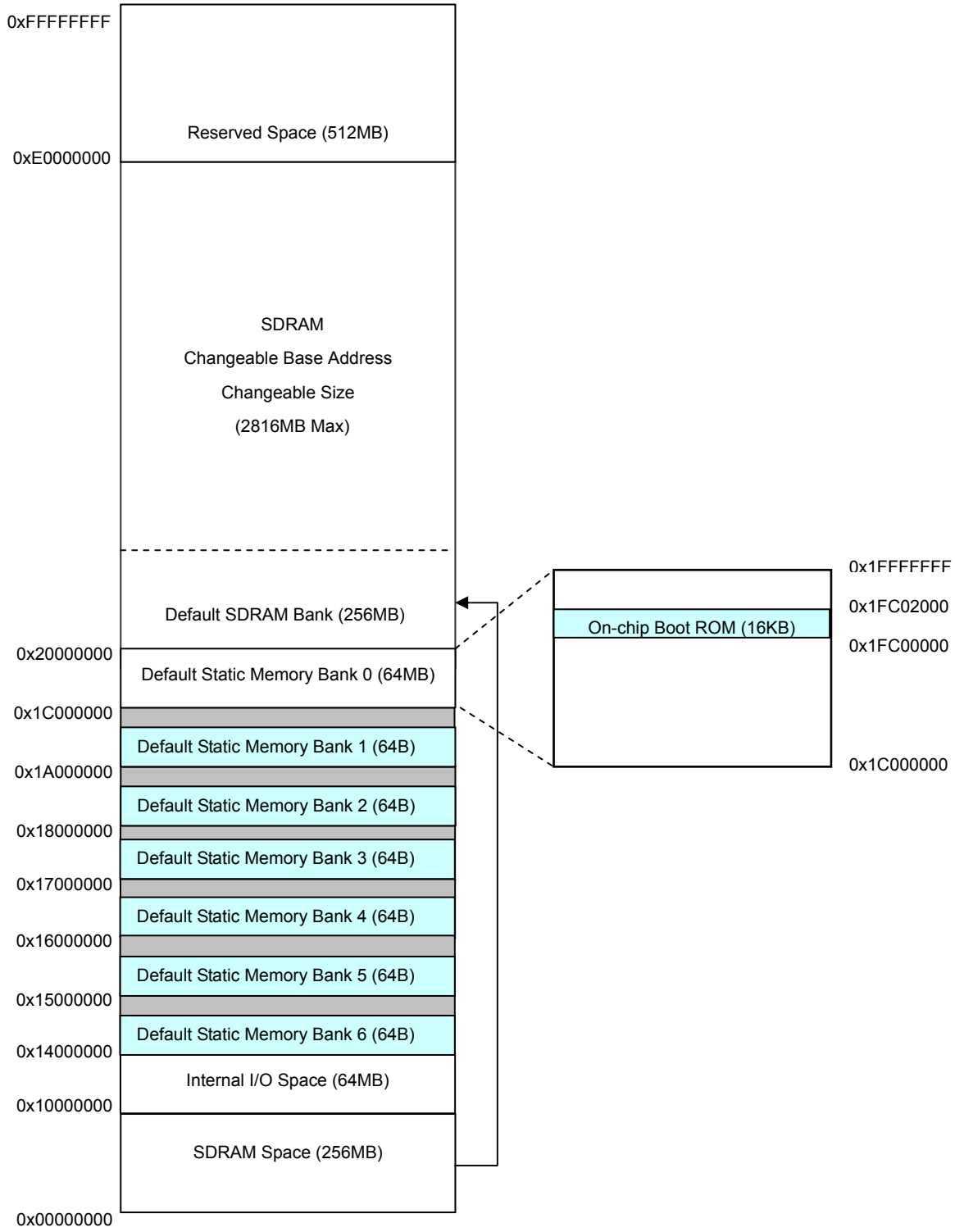


Figure 6-1 Physical Address Space Map

Table 6-2 Physical Address Space Map

Start Address	End Address	Connectable Memory	Capacity
0x00000000	0x0FFFFFFF	SDRAM Memory	256
0x10000000	0x10FFFFFF	I/O Devices on APB Bus	16
0x11000000	0x12FFFFFF	Reserved	32
0x13000000	0x13FFFFFF	I/O Devices on AHB Bus	16
0x14000000	0x1400003F	Static Memory, CS6#	64B
0x14000040	0x14FFFFFF	Reserved	
0x15000000	0x1500003F	Static Memory, CS5#	64B
0x15000040	0x15FFFFFF	Reserved	
0x16000000	0x1600003F	Static Memory, CS4#	64B
0x16000040	0x16FFFFFF	Reserved	
0x17000000	0x1700003F	Static Memory, CS3#	64B
0x17000040	0x17FFFFFF	Reserved	
0x18000000	0x1800003F	Static Memory, CS2#	64B
0x18000040	0x19FFFFFF	Reserved	
0x1A000000	0x1A00003F	Static Memory, CS1#	64B
0x1A000040	0x1BFFFFFF	Reserved	
0x1C000000	0x1FBFFFFF	Reserved	60
0x1FC00000	0x1FC01FFF	On-chip Boot ROM (8kB)	0.008
0x1FC02000	0x1FFFFFFF	Reserved	3.992
0x20000000	0xDFFFFFFF	SDRAM Memory	3072
0xE0000000	0xFFFFFFFF	Reserved	512

The base address and size of each memory banks are configurable. Software can re-configure these memory banks according to the actual connected memories. Following table lists the default configuration after reset.

Table 6-3 Default Configuration of EMC Chip Select Signals

Chip-Select Signal	Connected Memory	Capacity	Memory Width *1	Start Address	End Address
CS1#	Static memory bank 1	64 B	8, 16, 32	0x1A000000	0x1A00003F
CS2#	Static memory bank 2	64 B	8, 16, 32	0x18000000	0x1800003F
CS3#	Static memory bank 3	64 B	8, 16, 32	0x17000000	0x1700003F
CS4#	Static memory bank 4	64 B	8, 16, 32	0x16000000	0x1600003F
CS5#	Static memory bank 5	64 B	8, 16, 32	0x15000000	0x1500003F
CS6#	Static memory bank 6	64 B	8, 16, 32	0x14000000	0x1400003F

NOTES:

1 Data width of static memory banks can be configured to 8, 16 or 32 bits by software.

2 The 8KB address space from H'1FC00000 to H'1FC01FFF in bank 0 is mapped to on-chip boot ROM. The other memory spaces in bank 0 are not used.

3 To support large SDRAM space, EMC re-maps the physical address H'00000000-H07FFFFFF to H'20000000-H'27FFFFFF. Software must configure the SDRAM base address by the re-mapped address.

6.4 SDRAM Interface

The SDRAM controller provides a glueless interface to industry standard SDRAM chip. The SDRAM controller provides two chip selects DCS0~1# supporting 16-bit or 32-bit wide SDRAM.

Both 2-bank and 4-bank SDRAM modules are supported. The bank select signals are always output from the A13 pin and A14 pin of processor.

The SDRAM interface includes the following signals:

- Two chip selects, DCS0#, DCS1#
- Four byte mask signals, DQM3~0#
- 15 multiplexed bank/row/column address signals, A14-A0
- One write enable, RD/WR#
- One column-address strobe CAS#
- One row-address strobe RAS#
- One clock enable CKE
- One clock CKO

The processor performs auto-refresh (CBR) during normal operation and supports self-refreshing SDRAM during sleep, hibernate, and frequency-change modes. An SDRAM power-down mode bit (DMCR[PDM]]) can be set so that the CKO and the clock-enable signal CKE to SDRAM are automatically de-asserted whenever none of the corresponding banks is being accessed.

6.4.1 Register Description

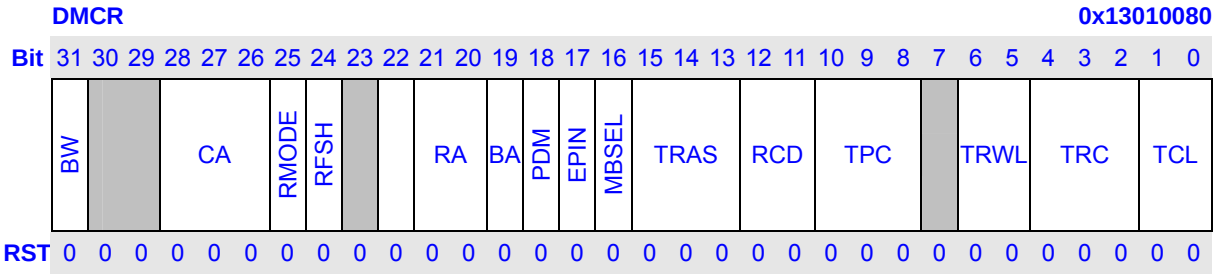
Table 6-4 SDRAM Registers

Name	Description	RW	Reset Value	Address	Access Width
DMCR	DRAM control register	RW	0x0000 0000	0x13010080	32
RTCSR	Refresh time control/status register	RW	0x0000	0x13010084	16
RTCNT	Refresh timer counter	RW	0x0000	0x13010088	16
RTCOR	Refresh time constant register	RW	0x0000	0x1301008C	16
DMAR1	SDRAM bank 0 address configuration register	RW	0x000020F8	0x13010090	32
DMAR2	SDRAM bank 1 address configuration register	RW	0x000028F8	0x13010094	32
SDMR	Mode register of SDRAM bank	W	--	0x1301-xxx (-: 4'b1xxx)	8

6.4.1.1 SDRAM Control Register (DMCR)

DMCR is a 32-bit read/write register that specifies the timing, address multiplexing and refresh control of SDRAM. This enables direct connection of SDRAM without external circuits.

The DMCR is initialized to 0x00000000 by any resets. SDRAM bank should not be accessed until initialization is completed.



Bits	Name	Description	RW
31	BW	Specifies the data bus width of SDRAM. **BW** **Description** 0 Data width is 32 bits (Initial value) 1 Data width is 16 bits	RW
30:29	Reserved	Writes to these bits have no effect and always read as 0.	R
28:26	CA	**Column Address Width**: Specify the column address width of connected SDRAM chip. **CA** **Description** 000 8 bits column address 001 9 bits column address 010 10 bits column address 011 11 bits column address 100 12 bits column address 101 Reserved 110 Reserved 111 Reserved	RW
25	RMODE	**Refresh Mode.** **RMODE** **Description** 0 Auto-refresh 1 Self-refresh	RW
24	RFSH	**Refresh Control.** **RFSH** **Description** 0 No refresh is performed (Initial value) 1 Refresh is performed	RW
23	MRSET	**Mode Register Set:** Set when a SDRAM mode register setting is used. When this bit is 0 and SDRAM mode register is written, a Pre-charge all banks command (PALL) is performed. When this bit is 1 and SDRAM	RW

		mode register is written, a Mode Register Set command (MRS) is performed. **MRSET Description** 0 All-bank pre-charge (Initial value) 1 Mode register setting	
22	Reserved	Writes to these bits have no effect and always read as 0.	R
21:20	RA	**Row Address Width:** Specify the row address width of connected SDRAM. **RA Description** 00 11-bit row address (Initial value) 01 12-bit row address 10 13-bit row address 11 Reserved	RW
19	BA	**Bank Address Width:** Specify the number of bank select signals for one chip select. **BA Description** 0 1-bit bank address is used (2 banks each chip select) (Initial value) 1 2-bit bank address is used (4 banks each chip select)	RW
18	PDM	**Power Down Mode:** Set power-down mode. When power-down mode is set, SDRAM will be driven to power-down mode when it is not accessing and refreshing. Clock supply to SDRAM will be stopped also. **PDM Description** 0 Non-power-down mode (Initial value) 1 Power-down mode	RW
17	EPIN	**CKE Pin Control:** Controls the level of CKE pin. Clearing this bit by software causes a power-down command (if CKOEN of CPM is 1). **CAUTION:** after power-down command, all commands except power-down-exit are prohibited. Setting this bit by software causes a power-down-exit command. Setting EPIN is a part of initializes procedure for SDRAM. **EPIN Description** 0 CKE pin is deserted (Initial value) 1 CKE pin is asserted	RW
16	MBSEL	**Bank Select for Mode Register Load:** It is used to distinguish to load which bank Mode register. **MBSEL Description** 0 Bank 0 (Initial value) 1 Bank 1	RW
15:13	TRAS	**RAS Assertion Time:** When synchronous DRAM is connected, these bits set the minimum CKE negation time after self-refresh command is issued. **TRAS Description**	RW

		000 4 (Initial value) 001 5 010 6 011 7 100 8 101 9 110 10 111 11	
12:11	RCD	**RAS–CAS Delay**: Set the SDRAM bank active-read/write command delay time. **RCD** **Description** 00 1(Initial value) 01 2 10 3 11 4	RW
10:8	TPC	**RAS Precharge Time:** Specify the minimum number of cycles until the next bank active command is output after precharging. **TPC** **Description** 000 1 cycle (Initial value) 001 2 cycles 010 3 cycles 011 4 cycles 100 5 cycles 101 6 cycles 110 7 cycles 111 8 cycles	RW
7	Reserved	Writes to these bits have no effect and always read as 0.	R
6:5	TRWL	**Write Precharge Time:** Set the SDRAM write precharge delay time. In auto-precharge mode, they specify the time until the next bank active command is issued after a write cycle. After a write cycle, the next active command is not issued for a period of TRWL + TPC. **TRWL** **Description** 00 1 cycle (Initial value) 01 2 cycles 10 3 cycles 11 4 cycles	RW
4:2	TRC	**RAS Cycle Time:** For SDRAM, no bank active command is issued during the period TRC after an auto-refresh command. In self-refresh, these bits also specify the delay cycles to be inserted after CKE assertion. **TRC** **Description** 000 1 cycle (Initial value) 001 3 cycle	RW

		010 5 cycle 011 7 cycle 100 9 cycle 101 11 cycle 110 13 cycle 111 15 cycle	
1:0	TCL	**CAS Latency:** Specify the delay from read command to data becomes available at the outputs. **TCL** **Description** 00 Inhibit (Initial value) 01 2 cycles 10 3 cycles 11 Inhibit	RW

6.4.1.2 SDRAM Mode Register (SDMR)

SDMR is written to via the SDRAM address bus and is a 15-bit write-only register. It sets SDRAM mode for SDRAM bank. SDMR is undefined after a reset.

Write to the SDRAM mode register use the address bus rather than the data bus. If the value to be set is X and the SDMR address is Y, the value X is written in the SDRAM mode register by writing in address X + Y. Here Y is 0x8000, X is value for SDRAM configuration. For example X is 0x0022, random data is written to the address offset 0x8022, as a result, 0x0022 is written to the SDMR register. The range for value X is 0x0000 to 0x7FFF.

SDMR **0x1301-000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SDRAM address																															
RST																																

The Mode Register is used to define the specific mode of operation of the SDRAM. This definition includes the section of a burst length, a burst type, a CAS latency, an operating mode and a write burst mode, as shown in following figure.

For Mobile SDR, Extended Mode Register is used to define low power mode.

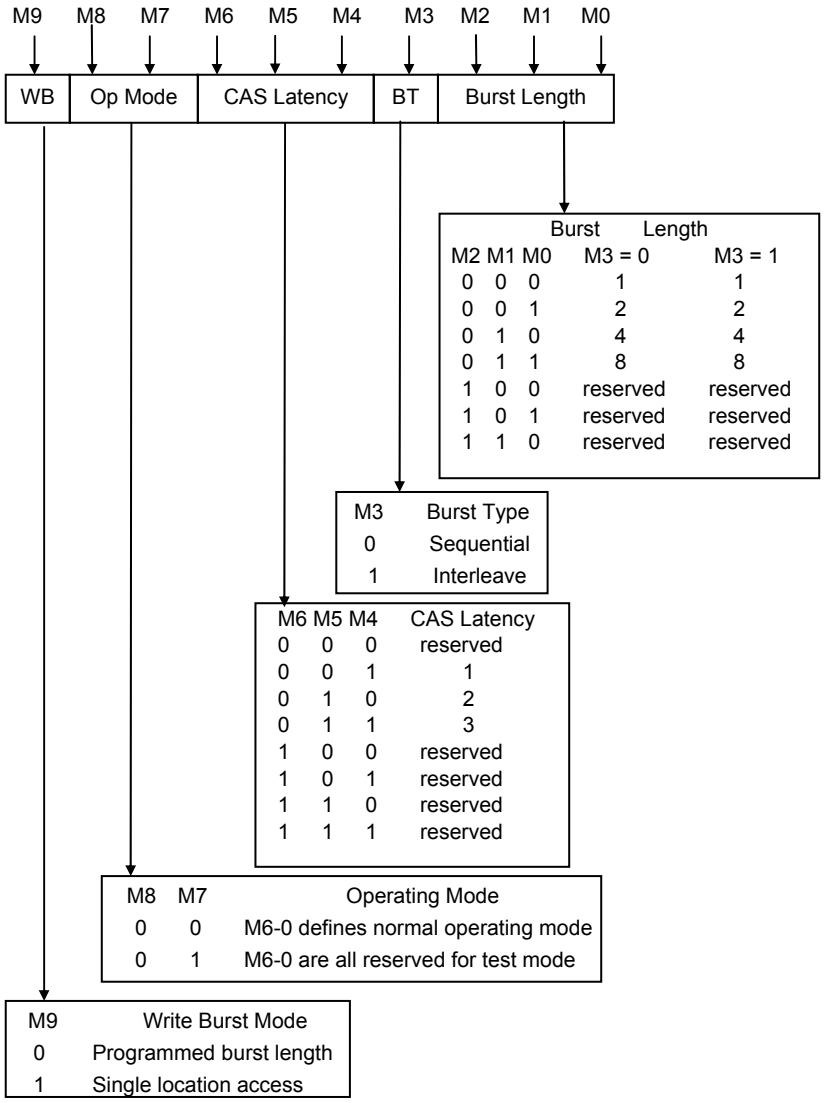


Figure 6-2 Synchronous DRAM Mode Register Configuration

6.4.1.3 Refresh Timer Control/Status Register (RTCSR)

RTCSR is a 16-bit readable/writable register that specifies the refresh cycle and the status of RTCNT. RTCSR is initialized to 0x0000 by a reset.

RTSCR 0x13010084

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							SRF	CMF	Reserved				CKS		
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15:9	Reserved	These bits always read 0. Data written to these bits are ignored.	R
8	SRF	**Self-refresh Flag (SRF):** Status flag that indicates EMC already enter self-refresh sequence. **SRF** **Description** 0 No self-refresh (Initial value) Clear condition: When 0 is written, write 1 is ignored 1 EMC already enter self-refresh sequence Set condition: when EMC enter self-refresh	RW
7	CMF	**Compare-Match Flag (CMF):** Status flag that indicates a match between the refresh timer counter (RTCNT) and refresh time constant register (RTCOR) values. Writes to 1 of this bit have no effect. **CMF** **Description** 0 RTCNT and RTCOR values do not match (Initial value) Clear condition: When 0 is written 1 RTCNT and RTCOR values match Set condition: When RTCNT = RTCOR	RW
2:0	CKS	**Refresh Clock Select Bits:** These bits select the clock input to RTCNT. The source clock is the external bus clock (CKO). The RTCNT count clock is CKO divided by the specified ratio. **CKS** **Description** 000 Disable clock input (Initial value) 001 Bus lock CKO/4 010 CKO/16 011 CKO/64 100 CKO/256 101 CKO/1024 110 CKO/2048 111 CKO/4096	RW

6.4.1.4 Refresh Timer Counter (RTCNT)

RTCNT is a 16-bit read/write register. RTCNT is a 16-bit counter that counts up with input clocks. The clock select bits (CKS2–CKS0) of RTCSR select the input clock. When the refresh bit (RFSH) of the memory control register (DMCR) is set to 1 and the refresh mode is set to auto-refresh, a memory refresh cycle starts when RTCNT matches RTCOR. RTCNT is initialized to 0x0000 by a reset.

RTCNT **0x13010088**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.4.1.5 SSTL18, SSTL2, MDDR & LVTTL combo single-end transceiver control register 1 (PMEMBS1)

This register is used to select output mode and strength of SSTL18, SSTL2, MDDR and LVTTL combo single-end transceiver.

PMEMBS1 **0x13016004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	IENDQS3	IENDQS2	IENDQS1	IENDQS0	IENDQ3	IENDQ2	IENDQ1	IENDQ0	Reserved							SSTL	SSELDQS3		SSELDQS2		SSELDQS1		SSELDQS0		SSELDQ3		SSELDQ2		SSELDQ1		SSELDQ0	
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	IENDQS3	Only use in SSTL mode represent input enable for DQS3. If SSTL = 1 set to 0 If SSTL = 0 set to 1	RW
30	IENDQS2	Only use in SSTL mode represent input enable for DQS2. If SSTL = 1 set to 0 If SSTL = 0 set to 1	RW
29	IENDQS1	Only use in SSTL mode represent input enable for DQS1. If SSTL = 1 set to 0 If SSTL = 0 set to 1	RW
28	IENDQS0	Only use in SSTL mode represent input enable for DQS0. If SSTL = 1 set to 0 If SSTL = 0 set to 1	RW
27	IENDQ3	Only use in SSTL mode represent input enable for DQ[31:24]. If SSTL = 1 set to 0	RW

		If SSTL = 0 set to 1	
26	IENDQ2	Only use in SSTL mode represent input enable for DQ[23:18]. If SSTL = 1 set to 0 If SSTL = 0 set to 1	RW
25	IENDQ1	Only use in SSTL mode represent input enable for DQ[17:8]. If SSTL = 1 set to 0 If SSTL = 0 set to 1	RW
24	IENDQ0	Only use in SSTL mode represent input enable for DQ[7:0]. If SSTL = 1 set to 0 If SSTL = 0 set to 1	RW
23:16	Reserved	These bits always read 0, and written are ignored.	R
16	SSTL	1: DQS and DQ pad work in SSTL mode 0: DQS and DQ pad work in LVCOMS or LVTTL mode	RW
15:14	SSELDQS3	Output mode & strength select for DQS[3].	RW
13:12	SSELDQS2	Output mode & strength select for DQS[2].	RW
11:10	SSELDQS1	Output mode & strength select for DQS[1].	RW
9:8	SSELDQS0	Output mode & strength select for DQS[0].	RW
7:6	SSELDQ3	Output mode & strength select for DQ[31:24].	RW
5:4	SSELDQ2	Output mode & strength select for DQ[23:16].	RW
3:2	SSELDQ1	Output mode & strength select for DQ[15:8].	RW
1:0	SSELDQ0	Output mode & strength select for DQ[7:0].	RW

SSEL configure:

MODE	Power Supply	SSEL [1:0]			
		Reduced Strength		Full Strength	
SSTL18	1.8V	01		11	
SSTL2	2.5V	00		10	
LPDDR	1.8V	00		11	
LVTTL	3.3V	00 (12mA)	01 (16mA)	10 (24mA)	11 (30mA)

6.4.1.6 SSTL18, SSTL2, MDDR & LVTTL combo single-end transceiver control register 0 (PMEMBS0)

This register is used to select output mode and strength of SSTL18, SSTL2, MDDR and LVTTL combo single-end transceiver.

PMEMBS0 **0x13016008**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PDDQS3	PDDQS2	PDDQS1	PDDQS0	PDDQ3	PDDQ2	PDDQ1	PDDQ0	STDQS3	STDQS2	STDQS1	STDQS0	STDQ3	STDQ2	STDQ1	STDQ0	PEDQS3	PEDQS2	PEDQS1	PEDQS0	PEDQ3	PEDQ2	PEDQ1	PEDQ0	PSDQS3	PSDQS2	PSDQS1	PSDQS0	PSDQ3	PSDQ2	PSDQ1	PSDQ0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	PDDQS3	Only use in NO SSTL mode represent INPUT BUFFER POWER DOWN for DQS3. 1: power down; 0: don't power down.	RW
30	PDDQS2	Only use in NO SSTL mode represent INPUT BUFFER POWER DOWN for DQS2. 1: power down; 0: don't power down.	RW
29	PDDQS1	Only use in NO SSTL mode represent INPUT BUFFER POWER DOWN for DQS1. 1: power down; 0: don't power down.	RW
28	PDDQS0	Only use in NO SSTL mode represent INPUT BUFFER POWER DOWN for DQS0. 1: power down; 0: don't power down.	RW
27	PDDQ3	Only use in NO SSTL mode represent INPUT BUFFER POWER DOWN for DQ[31:24]. 1: power down; 0: don't power down.	RW
26	PDDQ2	Only use in NO SSTL mode represent INPUT BUFFER POWER DOWN for DQ[23:18]. 1: power down; 0: don't power down.	RW
25	PDDQ1	Only use in NO SSTL mode represent INPUT BUFFER POWER DOWN for DQ[17:8]. 1: power down; 0: don't power down.	RW
24	PDDQ0	Only use in NO SSTL mode represent INPUT BUFFER POWER DOWN for DQ[7:0]. 1: power down; 0: don't power down.	RW
23	STDQS3	Schmitt trigger select pin for DQS[3]. 1: enable; 0: disable.	RW
22	STDQS2	Schmitt trigger select pin for DQS[2]. 1: enable; 0: disable.	RW
21	STDQS1	Schmitt trigger select pin for DQS[1]. 1: enable; 0: disable.	RW
20	STDQS0	Schmitt trigger select pin for DQS[0]. 1: enable; 0: disable.	RW
19	STDQ3	Schmitt trigger select pin for DQ[31:24]. 1: enable; 0: disable.	RW
18	STDQ2	Schmitt trigger select pin for DQ[23:16]. 1: enable; 0: disable.	RW
17	STDQ1	Schmitt trigger select pin for DQ[15:8]. 1: enable; 0: disable.	RW
16	STDQ0	Schmitt trigger select pin for DQ[7:0]. 1: enable; 0: disable.	RW
15	PEDQS3	Pull up or down enable pin for DQS[3]. 1: enable; 0: disable.	RW
14	PEDQS2	Pull up or down enable pin for DQS[2]. 1: enable; 0: disable.	RW
13	PEDQS1	Pull up or down enable pin for DQS[1]. 1: enable; 0: disable.	RW
12	PEDQS0	Pull up or down enable pin for DQS[0]. 1: enable; 0: disable.	RW

11	PEDQ3	Pull up or down enable pin for DQ[31:24]. 1: enable; 0: disable.	RW
10	PEDQ2	Pull up or down enable pin for DQ[23:16]. 1: enable; 0: disable.	RW
9	PEDQ1	Pull up or down enable pin for DQ[15:8]. 1: enable; 0: disable.	RW
8	PEDQ0	Pull up or down enable pin for DQ[7:0]. 1: enable; 0: disable.	RW
7	PSDQS3	Pull up or down select pin for DQS[3]. 1: pull-up; 0: pull-down.	RW
6	PSDQS2	Pull up or down select pin for DQS[2]. 1: pull-up; 0: pull-down.	RW
5	PSDQS1	Pull up or down select pin for DQS[1]. 1: pull-up; 0: pull-down.	RW
4	PSDQS0	Pull up or down select pin for DQS[0]. 1: pull-up; 0: pull-down.	RW
3	PSDQ3	Pull up or down select pin for DQ[31:24]. 1: pull-up; 0: pull-down.	RW
2	PSDQ2	Pull up or down select pin for DQ[23:16]. 1: pull-up; 0: pull-down.	RW
1	PSDQ1	Pull up or down select pin for DQ[15:8]. 1: pull-up; 0: pull-down.	RW
0	PSDQ0	Pull up or down select pin for DQ[7:0]. 1: pull-up; 0: pull-down.	RW

6.4.1.7 SSTL18, SSTL2, MDDR & LVTTL combo single-end/differential transmitter SSEL register (PMEMOSEL)

This register is used to select strength of SSTL18, SSTL2, MDDR and LVTTL combo single-end/differential transmitter.

PMEMOSEL **0x1301600C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												CKSSEL		CKESSEL		ADDRSSEL		DMSSEL3		DMSSEL2		DMSSEL1		DMSSEL0		CMDSSEL		CSSSEL1		CSSSEL0	
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:20	Reserved	These bits always read 0, and written are ignored.	R
19:18	CKSSEL	Output mode & strength select for CKO.	RW
17:16	CKESSEL	Output mode & strength select for CKE.	RW
15:14	ADDRSSEL	Output mode & strength select for ADDR[16:0].	RW
13:12	DMSSEL3	Output mode & strength select for DM3.	RW
11:10	DMSSEL2	Output mode & strength select for DM2.	RW
9:8	DMSSEL1	Output mode & strength select for DM1.	RW
7:6	DMSSEL0	Output mode & strength select for DM0.	RW
5:4	CMDSSEL	Output mode & strength select for CMD (RAS, CAS,WE).	RW
3:2	CSSSEL1	Output mode & strength select for CS1.	RW
1:0	CSSSEL0	Output mode & strength select for CS0.	RW

SSEL configure:

MODE	Power Supply	SSEL [1:0]			
		Reduced Strength		Full Strength	
SSTL18	1.8V	01		11	
SSTL2	2.5V	00		10	
LPDDR	1.8V	00		11	
LVTTL	3.3V	00 (12mA)	01 (16mA)	10 (24mA)	11 (30mA)

6.4.1.8 SSTL18, SSTL2, MDDR & LVTTL combo single-end/differential transmitter OEN register (PMEMOEN)

This register is used to select output enable signal (low active) of SSTL18, SSTL2, MDDR and LVTTL combo single-end transceiver.

PMEMOEN **0x13016010**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																	CKOEN	BAOEN2	BAOEN1	BAOEN0	AOEN13	AOEN12	AOEN11_0	DMOEN3	DMOEN2	DMOEN1	DMOEN0	CMDOEN	CSOEN1	CSOEN0	CKEOEN
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:15	Reserved	These bits always read 0, and written are ignored.	R
14	CKOEN	output put enable signal for CKO. 1: enable; 0: disable.	RW
13	BAOEN2	output put enable signal for BA[2] (ADDR[16]). 1: enable; 0: disable.	RW
12	BAOEN1	output put enable signal for BA[1] (ADDR[15]). 1: enable; 0: disable.	RW
11	BAOEN0	output put enable signal for BA[0] (ADDR[14]). 1: enable; 0: disable.	RW
10	AOEN13	output put enable signal for ADDR[13]. 1: enable; 0: disable.	RW
9	AOEN12	output put enable signal for ADDR[12]. 1: enable; 0: disable.	RW
8	AOEN11_0	output put enable signal for ADDR[11:0]. 1: enable; 0: disable.	RW
7	DMOEN3	output put enable signal for DM3. 1: enable; 0: disable.	RW
6	DMOEN2	output put enable signal for DM2. 1: enable; 0: disable.	RW
5	DMOEN1	output put enable signal for DM1. 1: enable; 0: disable.	RW
4	DMOEN0	output put enable signal for DM0. 1: enable; 0: disable.	RW
3	CMDOEN	output put enable signal for CMD(RAS,CAS,WE). 1: enable; 0: disable.	RW
2	CSOEN1	output put enable signal for CS1. 1: enable; 0: disable.	RW
1	CSOEN0	output put enable signal for CS0. 1: enable; 0: disable.	RW
0	CKEOEN	output put enable signal for CKE. 1: enable; 0: disable.	RW

6.4.2 Refresh Time Constant Register (RTCOR)

RTCOR is a 16-bit read/write register. The values of RTCOR and RTCNT (bottom 8 bits) are constantly compared. When the refresh bit (RFSH) of the memory control register (DMCR) is set to 1 and the refresh mode bit (RMODE) is set to auto-refresh, a memory refresh cycle starts when RTCNT matches RTCOR. RTCOR is initialized to 0x0000 by a reset.

RTCOR **0x1301008C**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

6.4.2.1 DRAM Bank Address Configuration Register (DMARn, n = 1, 2)

DMARn define the physical address for SDRAM bank0 or bank 1, respectively. Each register contains a base address and a mask. When the following equation is met:

(*physical_address* [31:24] & **MASK$_n$**) == **BASE$_n$**

The bank n is active. The *physical_address* is address output on internal system bus. DRAM bank regions must be programmed so that each bank occupies a unique area of the physical address space. Programming overlapping bank regions will result in unpredictable error.

These registers are initialized by a reset.

DMAR1 **0x13010090**

DMAR2 **0x13010094**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																	BASE								MASK							
RST																	0	0	1	0	0	0	0	0	1	1	1	1	1	0	0	0

Bits	Name	Description	RW
31:16	Reserved	Writes to these bits have no effect and read always as 0.	R
15:8	BASEn	**Address Base:** Defines the base address of SDRAM Bank. The initial values are: DMAR.BASE1 0x20 DMAR.BASE2 0x28	RW
23:20	MASKn	**Address Mask**: Defines the mask of SDRAM Bank. The initial values are:	RW

		DMAR.MASK 0xF8	

6.4.3 Example of Connection

Following figure shows an example of connection of 512K x 16-bit x 2-bank SDRAM.

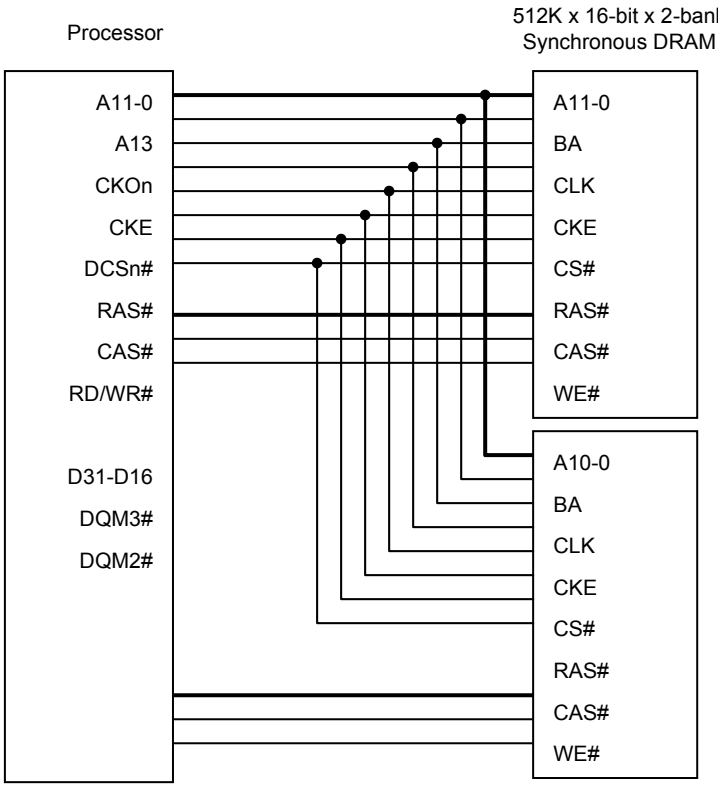


Figure 6-3 Example of Synchronous DRAM Chip Connection (1)

Following figure shows an example of connection of 1M x 16-bit x 4-bank synchronous DRAM.

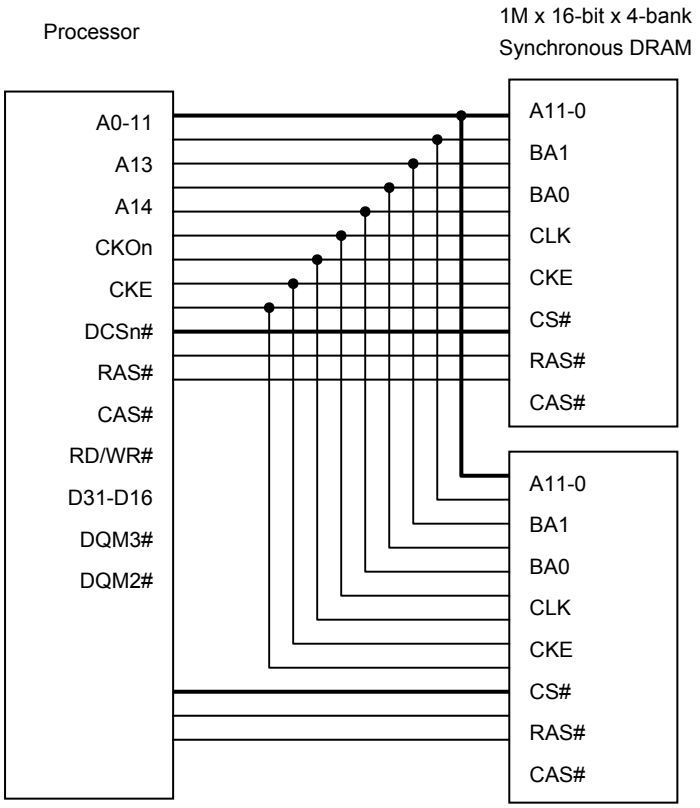


Figure 6-4 Example of Synchronous DRAM Chip Connection (2)

6.4.4 Address Multiplexing

SDRAM can be connected without external multiplexing circuitry in accordance the address multiplex specification bits CA2~0, RA1~0 and BA in DMCR. Table 6-5 shows the relationship between the address multiplex specification bits and the bits output at the address pins.

A14-0 is used as SDRAM address. The original values are always output at these pins.

Table 6-5 SDRAM Address Multiplexing (32-bit data width) *[4]

CA2~0	RA1~0	Output Timing	A0-A9, A10, A11, A12	A13	A14	Note
8 bits	11 bits	Column	A2-A11, L/H*[1], A12, A13	A21	A22	3, 4
		Row	A10-A22			
	12 bits	Column	A2-A11, L/H*[1], A12, A13	A22	A23	3, 4
		Row	A10-A22			
	13 bits	Column	A2-A11, L/H*[1], A12, A13	A23	A24	3, 4
		Row	A10-A22			
9 bits	11 bits	Column	A2-A11, L/H*[1], A12, A13	A22	A23	3, 4
		Row	A11-A23			
	12 bits	Column	A2-A11, L/H*[1], A12, A13	A23	A24	3, 4
		Row	A11-A23			
	13 bits	Column	A2-A11, L/H*[1], A12, A13	A24	A25	3, 4
		Row	A11-A23			
10 bits	11 bits	Column	A2-A11, L/H*[1], A12, A13	A23	A24	3, 4
		Row	A12-A24			
	12 bits	Column	A2-A11, L/H*[1], A12, A13	A24	A25	3, 4
		Row	A12-A24			
	13 bits	Column	A2-A11, L/H*[1], A12, A13	A25	A26	3, 4
		Row	A12-A24			
11 bits	11 bits	Column	A2-A11, L/H*[1], A12, A13	A24	A25	3, 4
		Row	A13-A25,			
	12 bits	Column	A2-A11, L/H*[1], A12, A13	A25	A26	3, 4
		Row	A13-A25,			
	13 bits	Column	A2-A11, L/H*[1], A12-A17	A26	A27	3, 4
		Row	A13-A25,			
12 bits	11 bits	Column	A2-A11, L/H*[1], A12, A13	A25	A26	3, 4
		Row	A14-A26			
	12 bits	Column	A2-A11, L/H*[1], A12, A13	A26	A27	3, 4
		Row	A14-A26			
	13 bits	Column	A2-A11, L/H*[1], A12, A13	A27	A28	3, 4
		Row	A14-A26			

NOTES:

1 L/H is a bit used in the command specification; it is fixed at L or H according to the Access mode.

2 Bank address specification.

3 If one bank select signal is used (BA = 0), take A13 as bank select signal. If two bank select signals are used (BA = 1), take A13 and A14 as bank select signals.

4 The A0 to A14 in table head are output pins. The A2 to A28 in table body are physical address.

6.4.5 SDRAM Command

Commands for SDRAM are specified by RAS#, CAS#, RD/WR and special address signals. The processor accesses SDRAM by using the following subset of standard interface commands.

- Mode Register Set (MRS)
- Bank Activate (ACTV)
- Read (READ)
- Write (WRIT)
- Burst Terminate
- Precharge All Banks (PALL)
- Auto-Refresh (CBR)
- Enter Self-Refresh (SLFRSH)
- No Operation (NOP)

Table 6-6 SDRAM Command Encoding (NOTES:1)

Command	Processor Pins							
	CS#	RAS#	CAS#	RD/WR#	DQM	A14-11, A9-0	A10	Note
INHIBIT	H	X	X	X	X	X	X	
NOP	L	H	H	H	X	X	X	
MRS	L	L	L	L	X	Op-Code		
ACTV	L	L	H	H	X	Bank, Row	X	2
READ	L	H	L	H	L/H	Bank, Col	L	3
WRIT	L	H	L	L	L/H	Bank, Col	L	3
Burst Terminate	L	H	H	L	X	X	X	
PRE	L	L	H	L	X	Bank	L	
PALL	L	L	H	L	X	X	H	
CBR/SLFRSH	L	L	L	H	X	X	X	4

NOTES:

1 CKE is HIGH for all commands shown except SLFRSH.
2 A0-A12 provides row address, and A13-A14 determines which bank is active.
3 A0-A9 provides column address, and A13-A14 determines which bank is being read from or written to.
4 This command is CBR if CKE is HIGH, SLFRSH if CKE is LOW.

6.4.6 SDRAM Timing

The SDRAM bank function is used to support high-speed accesses to the same row address. As SDRAM is internally divided into two or four banks, it is possible to activate one row address in each bank.

When a de-active bank is accessed, an access is performed by issuing an ACTV command following by READ or WRIT command.

When an active bank is accessed and just hit the open row, an access is performed by issuing READ or WRIT command immediately without issuing an ACTV command.

When an active bank is accessed but hit a closed row, a PRE command is first issued to precharge the bank, then the access is performed by issuing an ACTV command followed by a READ or WRIT command.

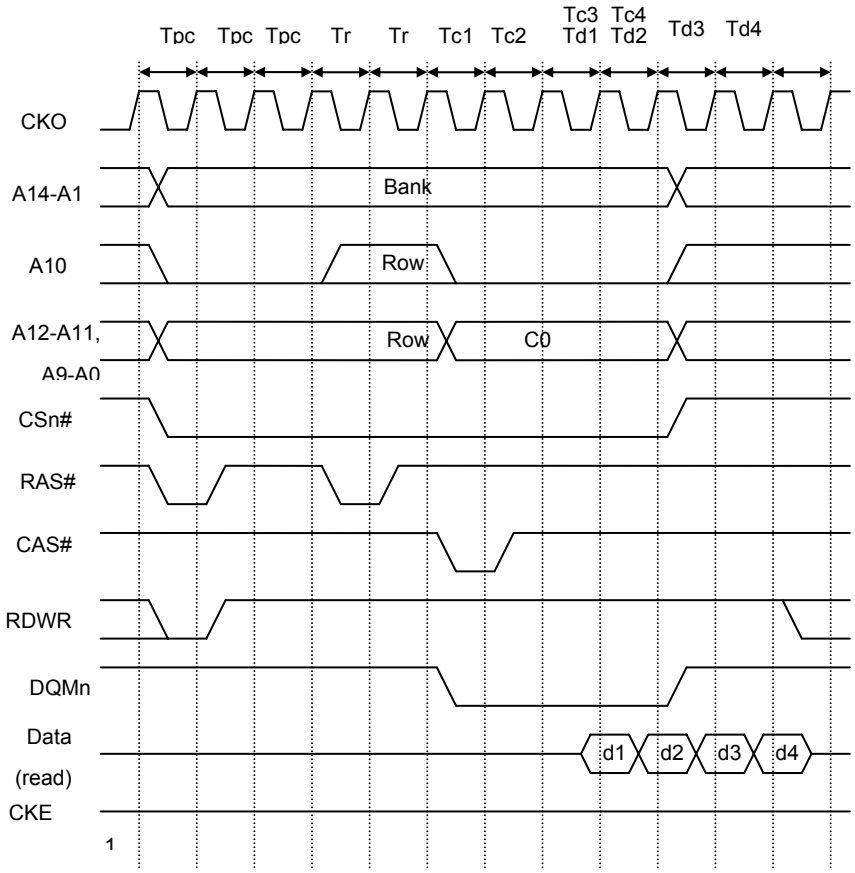
There is a limit on Tras , the time for placing each bank in the active state. If there is no guarantee that there will not be a cache hit and another row address will be accessed within the period in which this value is maintained by program execution, it is necessary to set auto-refresh and set the refresh cycle to no more than the maximum value of Tras. In this way, it is possible to observe the restrictions on the maximum active state time for each bank. If auto-refresh is not used, measures must be taken in the program to ensure that the banks do not remain active for longer than the prescribed time.

Glossary

Tr – row active cycle
Trw – row active wait cycle
Trwl – write latency cycle
Tpc – precharge cycle
TRr – refresh command cycle
Trc – RAS cycle
Trs1 – self refresh cycle 1
Trs2 – self refresh cycle 2
Trs3 – self refresh cycle 3
Tc1 – command cycle 1
Trsw – self refresh wait cycle
Tc2 – command cycle 2
Tc3 – command cycle 3
Tc4 – command cycle 4
Tc5 – command cycle 5
Tc6 – command cycle 6
Tc7 – command cycle 7
Tc8 – command cycle 8
Td1 – data cycle 1
Td2 – data cycle 2

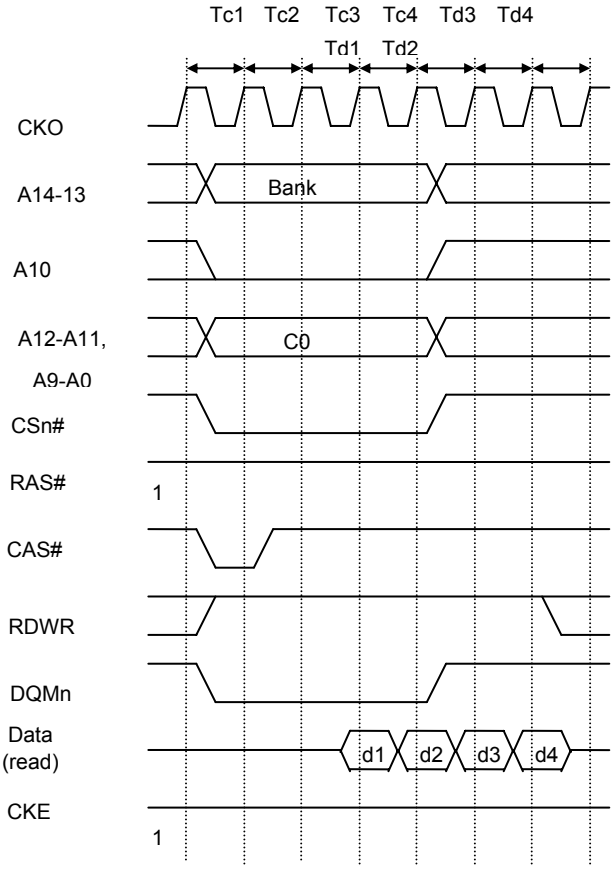
Td3 – data cycle 3
Td4 – data cycle 4
Td5 – data cycle 5
Td6 – data cycle 6
Td7 – data cycle 7
Td8 – data cycle 8
TRp1 – precharge-all cycle 1
TRp2 – precharge-all cycle 2
TRp3 – precharge-all cycle 3
TRp4 – precharge-all cycle 4
TMw1– mode register set cycle 1
TMw2– mode register set cycle 2
TMw3– mode register set cycle 3
TMw4– mode register set cycle 4

Following figures show the timing of 4-beat burst access, 8-beat burst access and single access.



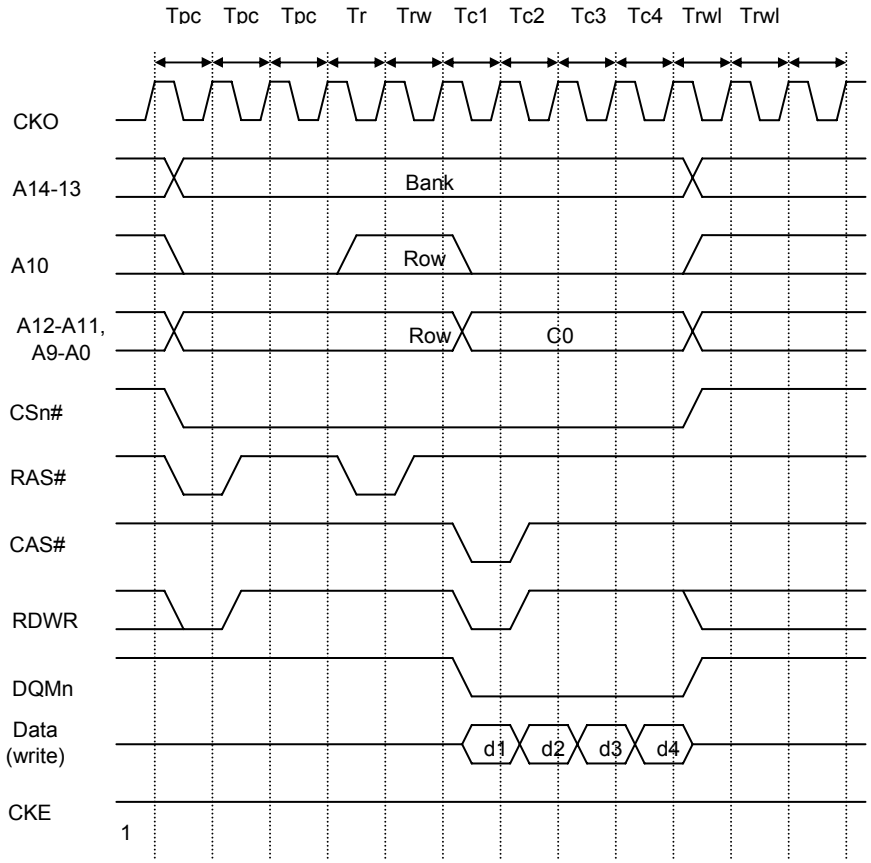
*DMCR: RCD = 1, TCL = 1, TPC = 2

Figure 6-5 Synchronous DRAM 4-beat Burst Read Timing (Different Row)



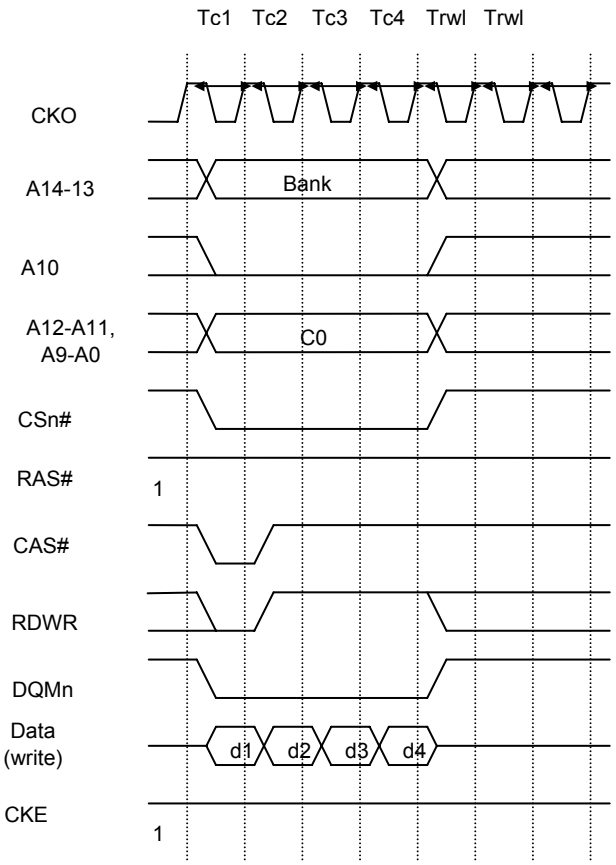
*DMCR: RCD = 1, TCL = 1, TPC = 2

Figure 6-6 Synchronous DRAM 4-beat Burst Read Timing (Same Row)



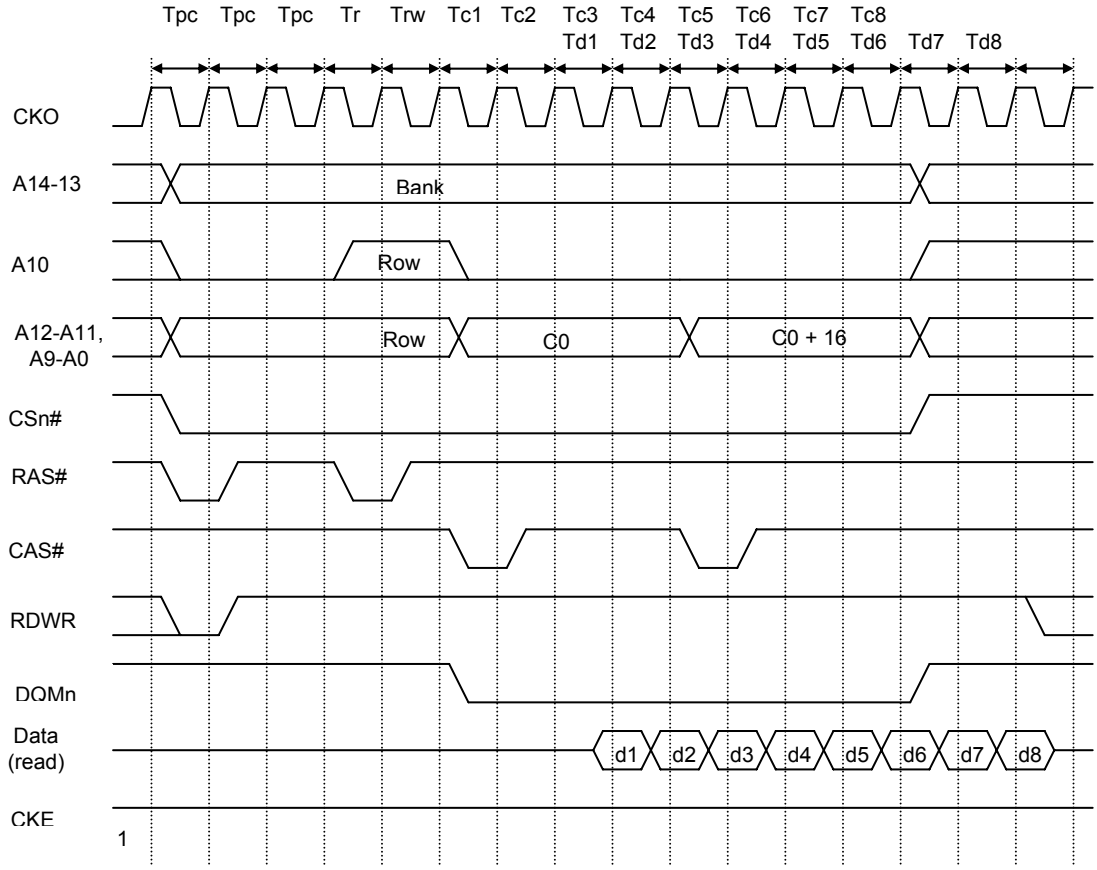
*DMCR: RCD = 1, TCL = 1, TPC = 2, TRWL = 1

Figure 6-7 Synchronous DRAM 4-beat Burst Write Timing (Different Row)



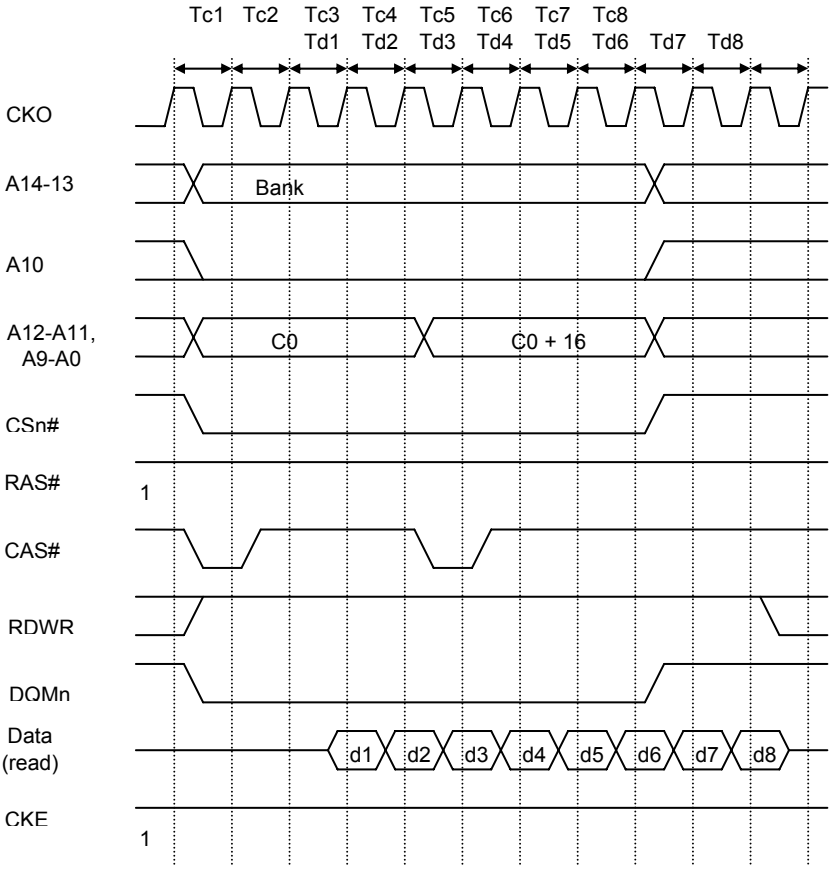
*DMCR: RCD = 1, TCL = 1, TPC = 2, TRWL = 1

Figure 6-8 Synchronous DRAM 4-beat Burst Write Timing (Same Row)



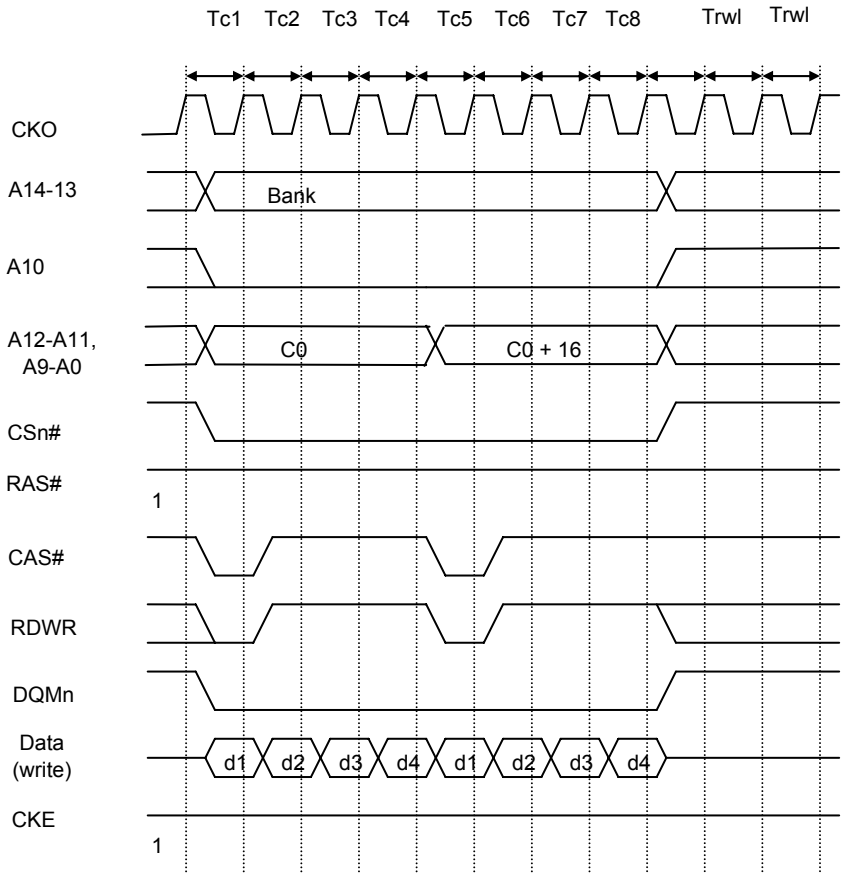
*DMCR: RCD = 1, TCL = 1, TPC = 2

Figure 6-9 Synchronous DRAM 8-beat Burst Read Timing (Different Row)



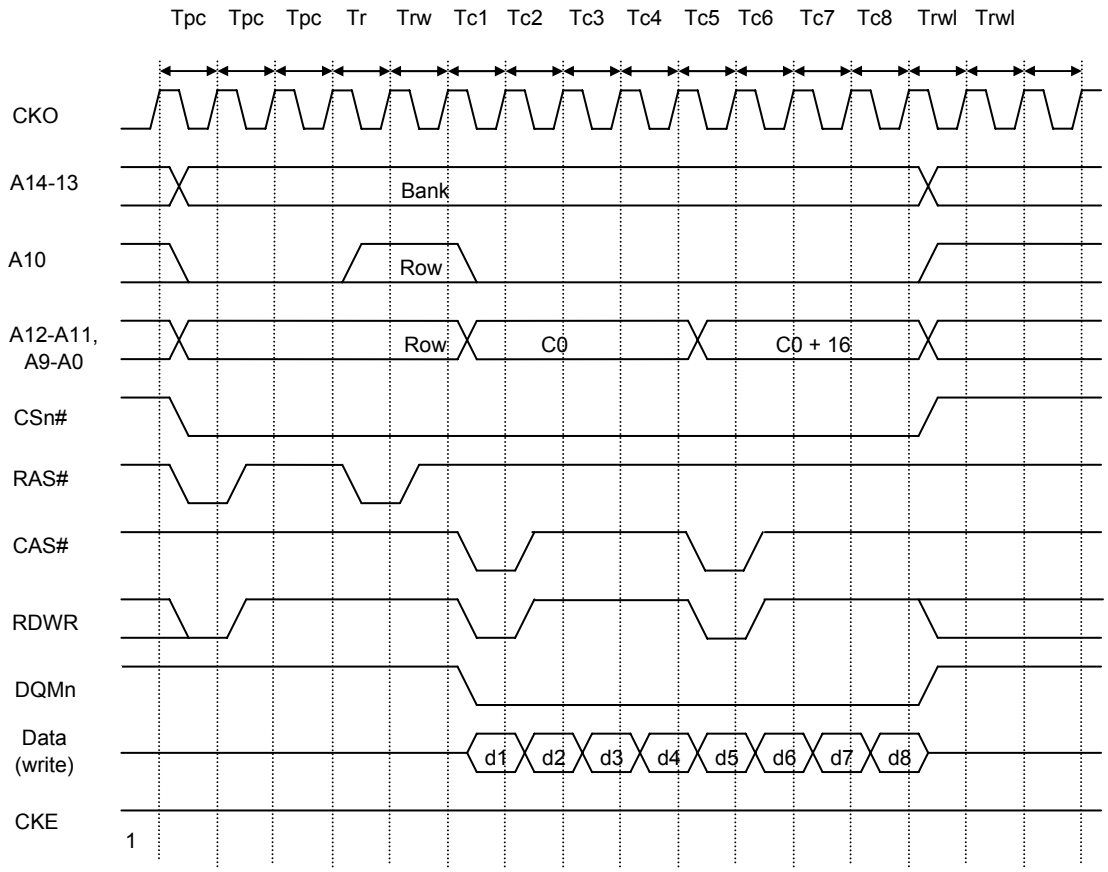
*DMCR: RCD = 1, TCL = 1, TPC = 2

Figure 6-10 Synchronous DRAM 8-beat Burst Read Timing (Same Row)



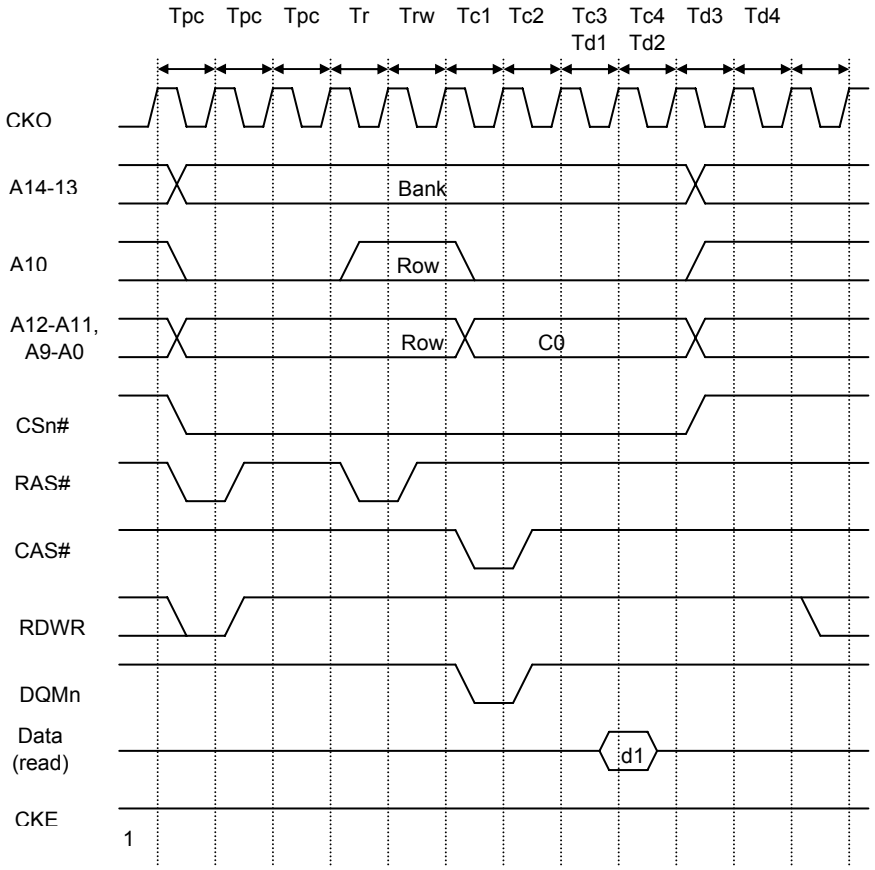
*DMCR: RCD = 1, TCL = 1, TPC = 2, TRWL = 1

Figure 6-11 Synchronous DRAM 8-beat Burst Write Timing (Same Row)



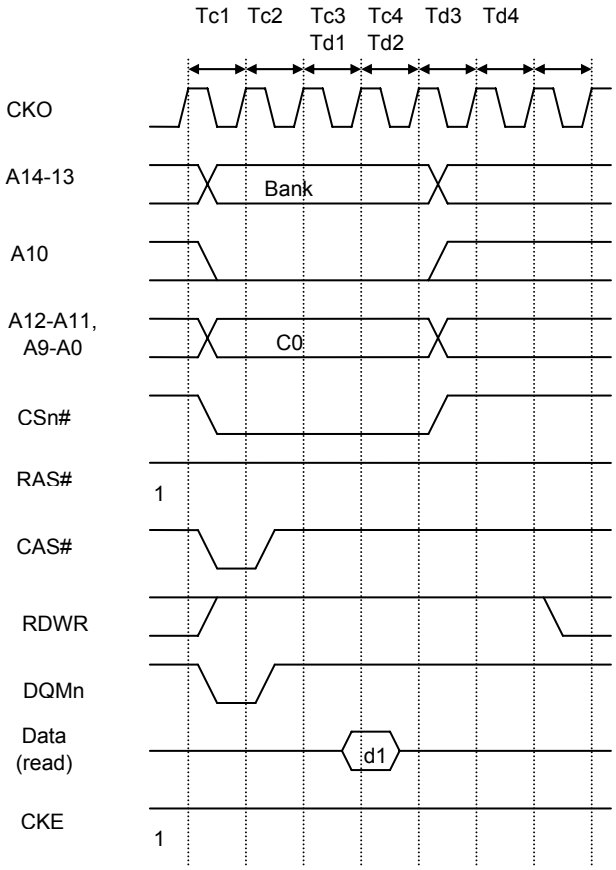
*DMCR: RCD = 1, TCL = 1, TPC = 2, TRWL = 1

Figure 6-12 Synchronous DRAM 8-beat Burst Write Timing (Different Row)



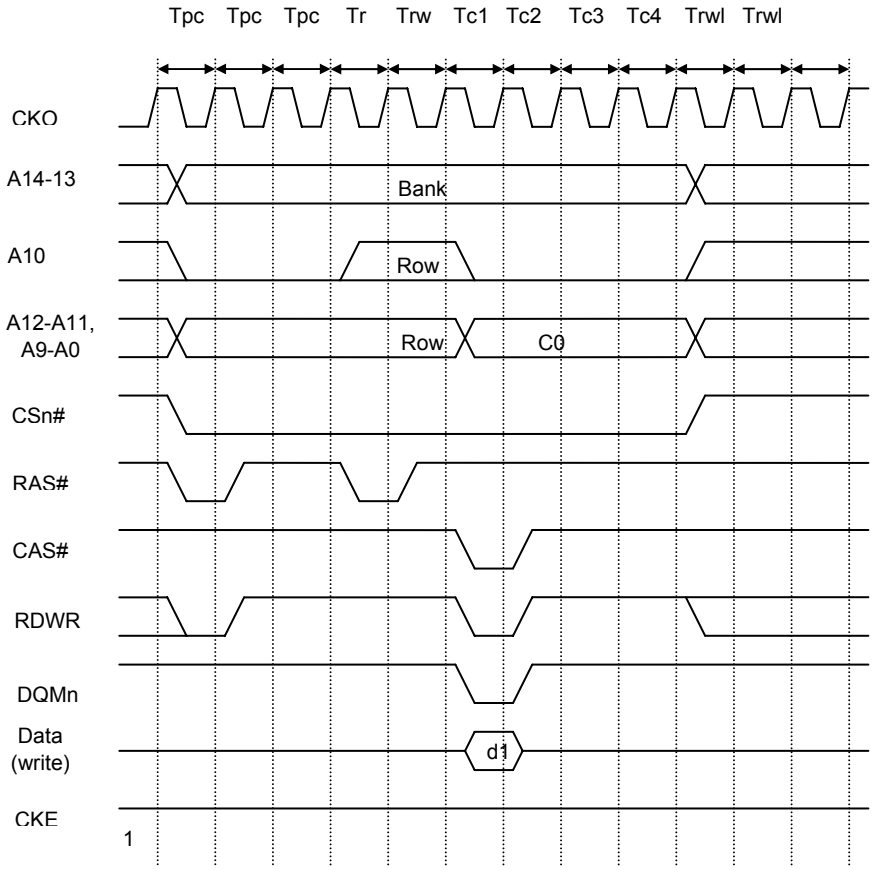
*DMCR: RCD = 1, TCL = 1, TPC = 2

Figure 6-13 Synchronous DRAM Single Read Timing (Different Row)



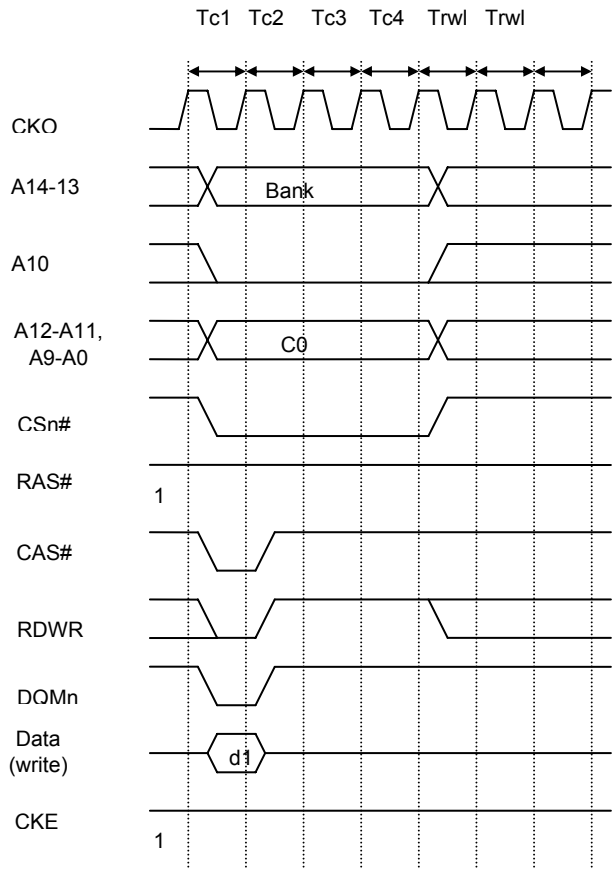
*DMCR: RCD = 1, TCL = 1, TPC = 2

Figure 6-14 Synchronous DRAM Single Read Timing (Same Row)



*DMCR: RCD = 1, TCL = 1, TPC = 2, TRWL = 1

Figure 6-15 Synchronous DRAM Single Write Timing (Different Row)



*DMCR: RCD = 1, TCL = 1, TPC = 2, TRWL = 1

Figure 6-16 Synchronous DRAM Single Write Timing (Same Row)

6.4.7 Power-Down Mode

The SDRAM power-down mode is supported to minimize the power consumption. CKE going to low level when SDRAM is idle/active state will drive SDRAM to precharge/active power-down mode. The clock supplies to SDRAM may be stopped also when CKE keep in low level more than two cycles. When a new access start or a refresh request, CKE is driven to high level and clock supplies is re-enabled. In power-down mode, clock of the accessed SDRAM bank pair is supplied. Clock of the other pair is stopped.

Following figures shows the timing of power-down mode and clock stopping.

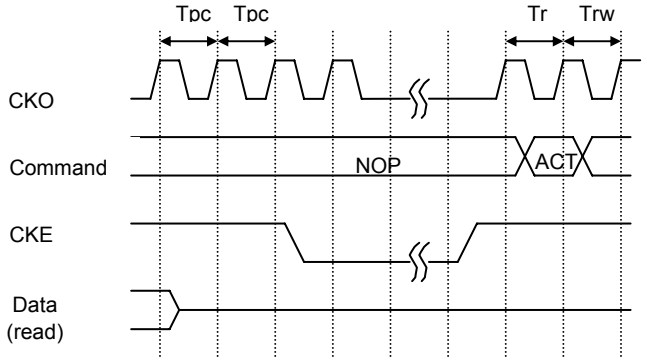


Figure 6-17 SDRAM Power-Down Mode Timing (CKO Stopped)

Following figure shows the power-down mode timing that CKE low level less than two cycles and clock is not stopped.

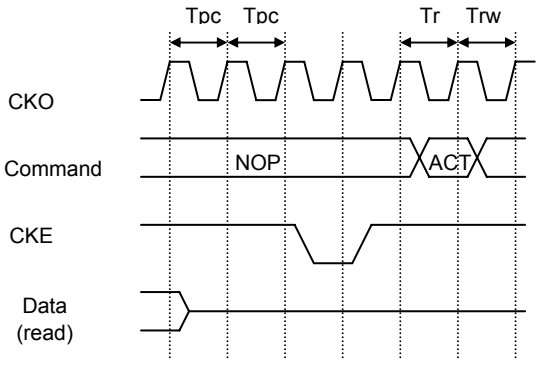


Figure 6-18 SDRAM Power-Down Mode Timing (Clock Supplied)

6.4.8 Refreshing

EMC provide a function for controlling the refresh of synchronous DRAM, Auto-refresh can be performed by clearing the RMODE bit to 0 and setting the RFSH bit to 1 in DMCR. If SDRAM is not accessed for a long period, self-refresh mode can be activated by set both the RMODE bit and the RFSH bit to 1.

6.4.8.1 AUTO-Refresh

Refreshing is performed at intervals determined by the input clock selected by bits CKS2-0 in RTCSR, and the value set in RTCOR. The value of bits CKS2-0 in RTCSR should be set so as to satisfy the refresh interval stipulation for the synchronous DRAM used. First make the settings for RTCOR, RTCNT, and the RMODE and RFSH bits in MCR, and then make the CKS2-CKS0 setting. When the clock is selected by CKS2-CKS0, RTCNT starts counting up from the value at that time. The RTCNT value is constantly compared with the RTCOR value, and if the two values are the same, a refresh request is generated and an auto-refresh is performed. At the same time, RTCNT is cleared to zero and the count-up is restarted. Figure 6-19 shows the auto-refresh cycle operation.

First, a REF command is issued in the TRr cycle. After the TRr cycle, new command output cannot be performed for the duration of the number of cycles specified by the TRC bits in DMCR. The TRC bits must be set so as to satisfy the synchronous DRAM refresh cycle time stipulation (active/active command delay time). Following figure shows the auto-refresh timing when TRC is set to 2.

Auto-refresh is performed in normal operation and sleep mode.

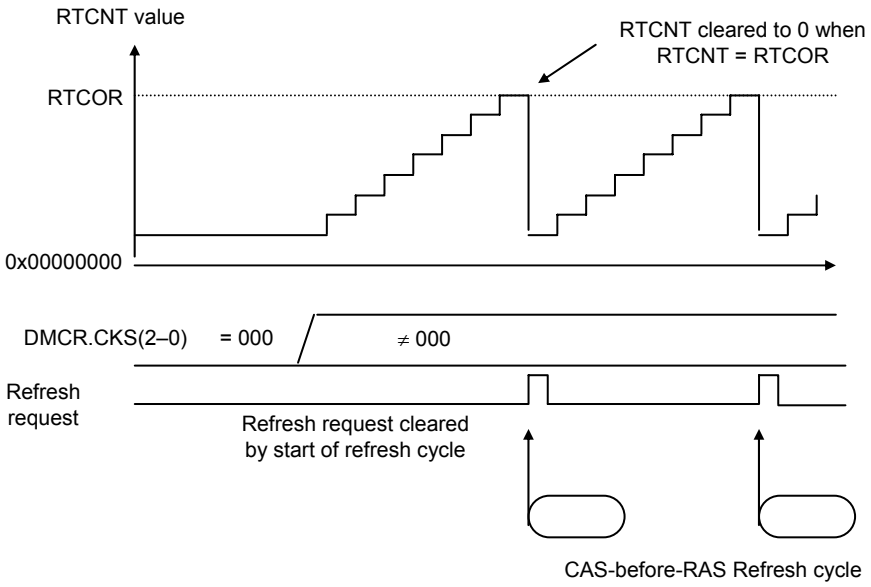


Figure 6-19 Synchronous DRAM Auto-Refresh Operation

A PALL command is issues firstly to precharge all banks. Then a REF command is issued in the TRr cycle.

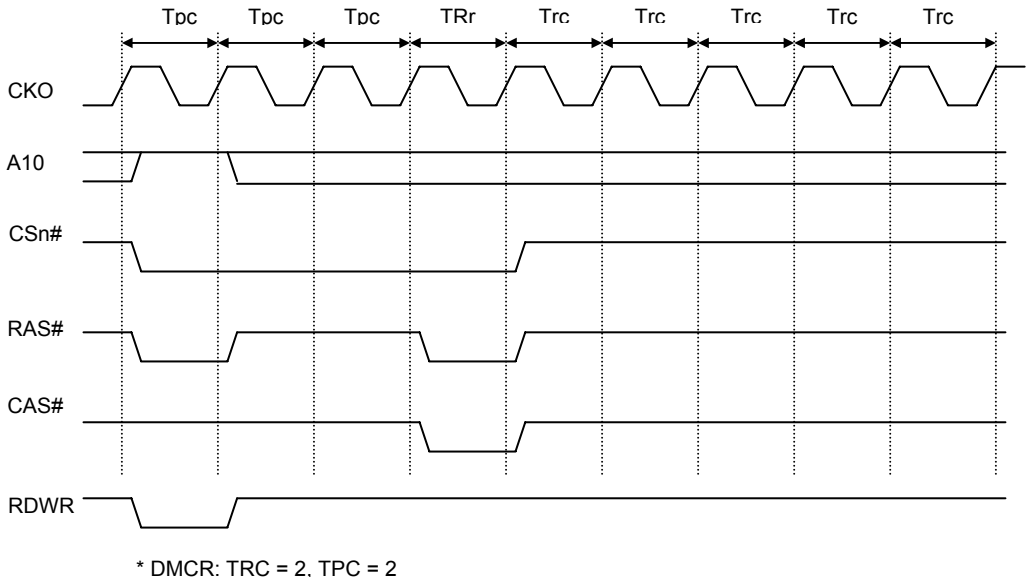


Figure 6-20 Synchronous DRAM Auto-Refresh Timing

6.4.8.2 SELF-Refresh

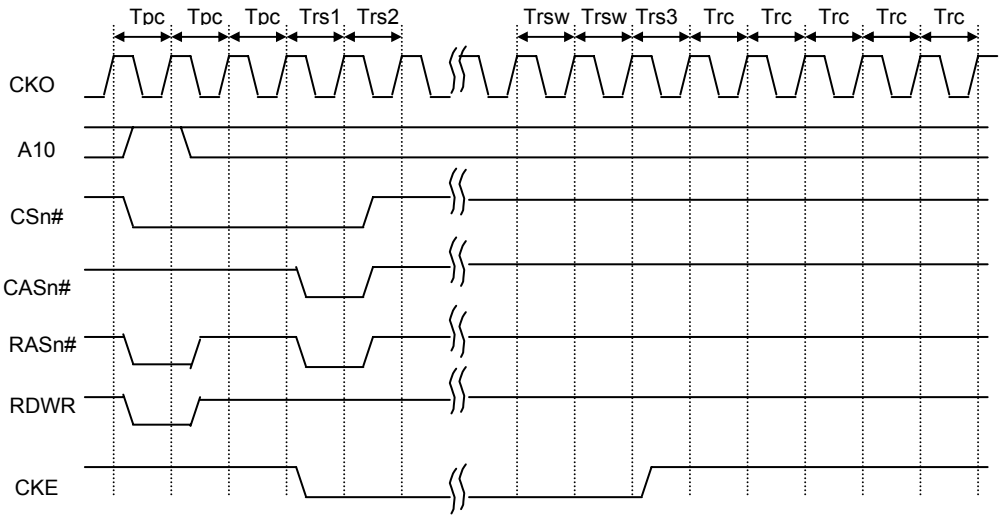
Self-refresh mode is a kind of sleep mode in which the refresh timing and refresh addresses are generated within the SDRAM. Self-refreshing is activated by setting both the RMODE bit and the RFSH bit to 1. The self-refresh state is maintained while the CKE signal is low. SDRAM cannot be accessed while in the self-refresh state. Self-refresh mode is cleared by clearing the RMODE bit to 0. After self-refresh mode has been cleared, command issuance is disabled for the number of cycles specified by the TRC bits in DMCR. Trsw cycles are inserted to meet the minimum CKE negation time specified by the TRAS bits in DMCR. Self-refresh timing is shown in following figure. Settings must be made so that self-refresh clearing and data retention are performed correctly, and auto-refresh is performed at the correct intervals. When self-refreshing is activated from the state in which auto-refreshing is set, or when exiting sleep mode other than through a reset, auto-refreshing is restarted if RFSH is set to 1 and RMODE is cleared to 0 when self-refresh mode is cleared. If the transition from clearing of self-refresh mode to the start of auto-refresh takes time, this time should be taken into consideration when setting the initial value of RTCNT. Making the RTCNT value 1 less than the RTCOR value will enable refreshing to be started immediately. After self-refreshing has been set, the self-refresh state continues even if the chip standby state is entered using the processor's sleep function, and is maintained even after recovery from sleep mode other than through a reset. In the case of a reset, the bus state controller's registers are initialized, and therefore the self-refresh state is cleared.

Self-refreshing is performed in normal operation, in idle mode and in sleep mode. In sleep mode, if RFSH bit in DMCR is 1, self-refresh is always performed in spite of RMODE field in DMCR until sleep mode is canceled.

Relationship between Refresh Requests and Bus Cycle Requests:

If a refresh request is generated during execution of a bus cycle, execution of the refresh is deferred until the bus cycle is completed. If a match between RTCNT and RTCOR occurs while a refresh is waiting to be executed, so that a new Refresh request is generated, the previous refresh request is eliminated. In order for refreshing to be performed normally, care must be taken to ensure that no bus cycle is longer than the refresh interval.

A PALL command is issued firstly to precharge all banks.



* DMCR: TRAS = 0, TRC = 2

Figure 6-21 Synchronous DRAM Self-Refresh Timing

6.4.9 Initialize Sequence

In order to use SDRAM, mode setting must first be performed after powering on. To perform SDRAM initialization correctly, the EMC registers must first be set, followed by a write to the SDRAM mode register.

In SDRAM mode register setting, the address signal value at that time is latched by MRS command. If the value to be set is X, the bus state controller provides for value X to be written to the synchronous DRAM mode register by performing a write to address offset 0x8000 + X for bank 0. In this operation the data is ignored, but the mode write is performed as a byte-size access. To set burst read/write, CAS latency 2 to 3, wrap type = sequential, and burst length 4 supported by the processor, arbitrary data is written in a byte-size access to the following addresses.

Table 6-7 SDRAM Mode Register Setting Address Example (32-bit)

	Bank 0	Bank 1		
CAS latency 2	8022	8022		
CAS latency 3	8032	8032		

Table 6-8 SDRAM Mode Register Setting Address Example (16-bit)

	Bank 0	Bank 1		
CAS latency 2	8011	8011		
CAS latency 3	8019	8011		

The value set in DMCR.MRSET is used to select whether a Pre-charge All Banks command (PALL) or a Mode Register Set command (MRS) is issued. DMCR.MBSEL is used to select Bank 0 or Bank 1 for Mode Register Set. The timing for the Pre-charge All Banks command is shown in Figure 6-22, and the timing for the Mode Register Set command in Figure 6-23.

Before mode register setting, a 200 s idle time (depending on the memory manufacturer) must be guaranteed after powering on requested by the synchronous DRAM. If the reset signal pulse width is greater than this idle time, there is no problem in performing initialize sequence immediately.

First, a pre-CHARGE all bank (PALL) command must be issued by performing a write to address offset 0x8000 + X for bank 0, while DMCR.MRSET = 0, DMCR.MBSEL = 0.

Next the NUMBER of dummy auto-refresh cycles specified by the manufacturer (usually 8) or more must be executed. This is usually achieved automatically while various kinds of initialization are being performed after auto-refresh setting, but a way of carrying this out more dependably is to set a short refresh request generation interval just while these dummy cycles are being executed. With simple read or write access, the address counter in the synchronous DRAM used for auto-refreshing is not initialized, and so the cycle must always be an auto-refresh cycle.

After auto-REFRESH has been executed at least the prescribed number of times, a Mode Register Set command (MRS) is issued in the TMw1 cycle by setting DMCR.MRSET to 1 and DMCR.MBSEL to 0 for bank 0 or DMCR.MBSEL to 1 for bank 1 and performing a write to address offset 0x8000 + X.

An example of SDRAM operation flow is as the following:

1 Disable Bus release.
Write 0x00000000 to BCR.

2 Initialize RTCOR and RTCNT for auto-refresh cycle.
Before configure SDRAM SDMR, SDRAM needs to execute auto-refresh, the number of times depends on the type of SDRAM. It's better to set a short refresh request generation interval here. For example, set RTCOR to 0x0000000F, and set RTCNT 0x00000000.

3 Initialize DMCR for Precharge all bank and auto-refresh.
When DMCR.RMODE=0 and DMCR.RFSH=1, enter auto-refresh mode;
When DMCR.MRSET=0, DMCR.MBSEL=0 (bank 0) or 1 (bank 1), write SDMR will generates Precharge all bank cycle.
DMCR.TPC must be defined for precharge.

4 Disable refresh counter clock.
Write 0x00000000 to RTCSR.

5 Execute Precharge all bank before auto-refresh.
Because DMCR.MRSET=0, DMCR.MBSEL=0 (bank 0) or 1 (bank 1), writing SDMR generates a Precharge all bank cycle, for example, write address (0x13018000).

6 Enable fast refresh counter clock for auto-refresh cycle.
For example, write 0x00000001 to RTCSR.

7 Wait for number of auto-refresh cycles (defined by SDRAM chip).
When RTCSR.CMF=1, it indicates value of RTCOR and RTCNT match and an auto-refresh cycle occurs.

8 Configure DMCR for SDRAM MODE Register Set.
When DMCR.MRSET=1, DMCR.MBSEL=0 (bank 0) or 1 (bank 1), write SDMR generate MRSET cycle.
For example, write 0x059A5231 to DMCR, so that:
Bus-width: 32-bit; Column Address: 9-bit; Row Address: 12-bit; Auto-refresh mode; SDMR Set mode; 4-bank; etc..

9 SDRAM Mode Register Set.
Because DMCR.MRSET=1 and DMCR.MBSEL=0, for example, write address 0x13018022 to configure SDMR as:
Burst Length: 4 burst
Burst Type: Sequential
CAS Latency: 2

10 Set normal auto-refresh counter clock.
For example, write 0x00000005 to RTCSR.

11 Then Read/Write SDRAM can be executed.

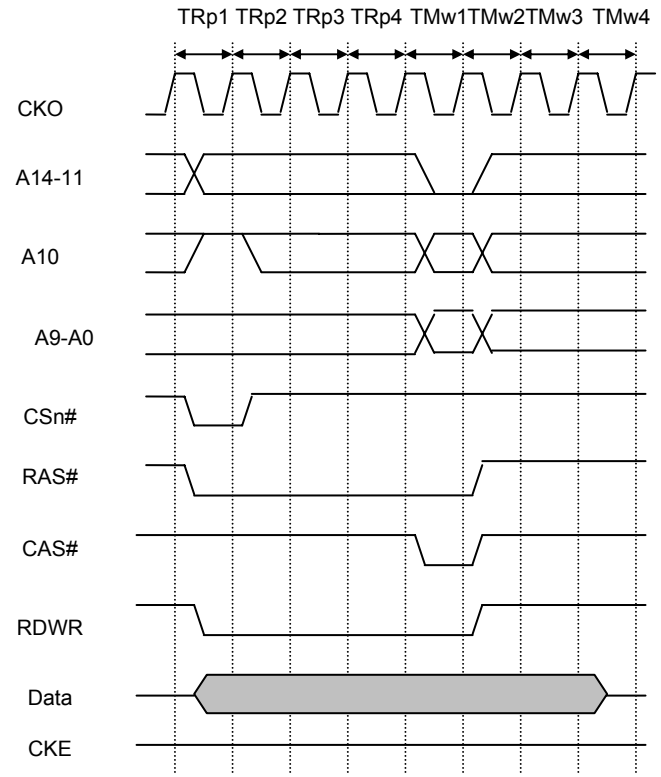


Figure 6-22 SDRAM Mode Register Write Timing 1 (Pre-charge All Banks)

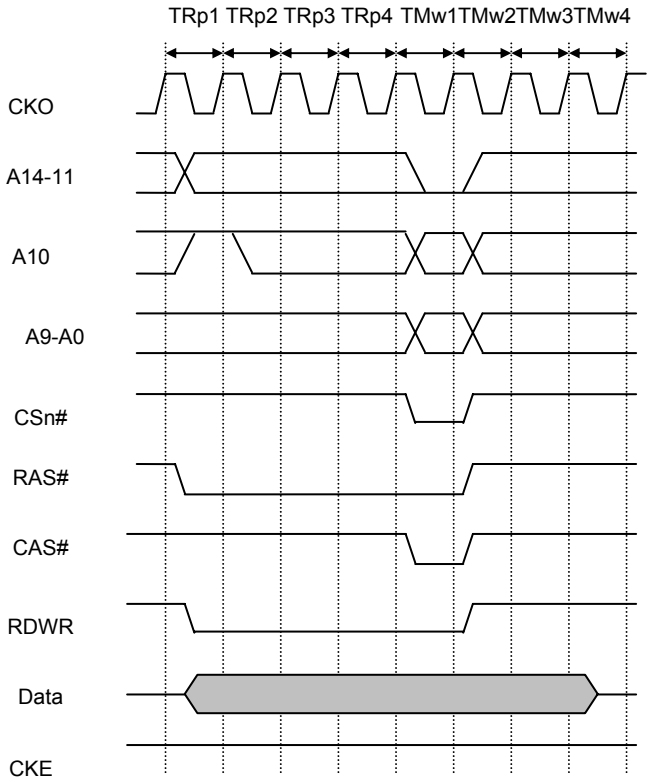
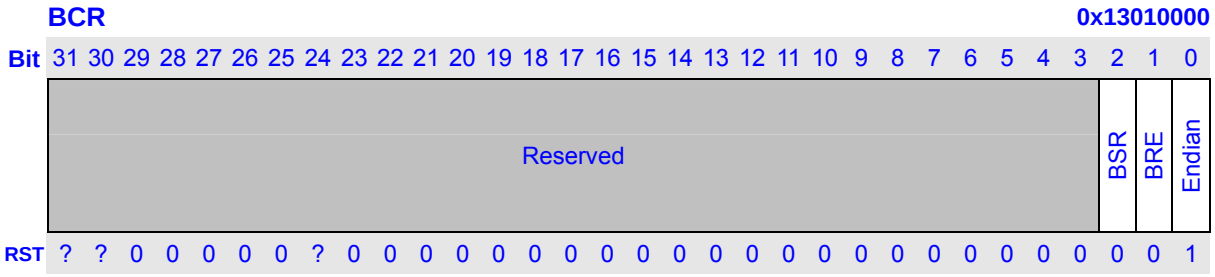


Figure 6-23 SDRAM Mode Register Write Timing 2 (Mode Register Set)

6.5 Bus Control Register (BCR)

BCR is used to specify the behavior of EMC on system bus. It is initialized to 0x00000001 by any reset.

Name	Description	RW	Reset Value	Address	Access Width
BCR	Bus Control Register	RW	0x?0000001	0x13010000	32



Bits	Name	Description	RW
31:3	Reserved	Writes to these bits has no effect and always read as 0.	R
2	BSR	**Bus Share Select.** 0: Nand and SDRAM bus share; 1: Nand and SDRAM bus separate.	RW
1	BRE	**Bus Release Enable**: When clear, once a transaction to EMC begins on the system bus; it must be completed before another transaction starts. When set, the system bus may be released to allow other transaction before EMC prepare the read data or be able to receipt the write data. If slow memory devices are used in the system, setting this bit will improve the efficPDcy of the whole system. The efficPDcy of SDRAM access may be improved by setting this bit. But the power consumption is increased if this bit is set. **BRE Description** 0 The system bus can not be released during an access (Initial value) 1 The system bus can be released during an access	RW
0	Endian	**Endian:** Indicates the system is little-endian.	R

7 External NAND Memory Controller

7.1 Overview

The External NAND Memory Controller (NEMC) divides the off-chip memory space and outputs control signals complying with specifications of various types of static memory and bus interfaces. It enables the connection of static memory such as NAND flash memory, etc. to this processor.

- Static memory interface
  - Support 6 external chip selection CS6~1#. Each bank can be configured separately
  - The size and base address of static memory banks are programmable
  - Direct interface to 8-bit or 16-bit (no byte control) bus width external memory interface devices or external static memory to each bank. Read/Write strobe setup time and hold time periods can be programmed and inserted in an access cycle to enable connection to low-speed memory
  - Wait insertion by WAIT pin
  - Automatic wait cycle insertion to prevent data bus collisions in case of consecutive memory accesses to different banks, or a read access followed by a write access to the same bank

- NAND flash interface
  - Support on CS6~CS1, sharing with static memory bank6~bank1
  - Support most types of NAND flashes, including 8-bit and 16-bit bus width, 512B/2K/4K/8KB page size. For 512B page size, 3 and 4 address cycles are supported. For 2K/4K/8KB page size, 4 and 5 address cycles are supported
  - Support read/erase/program NAND flash memory
  - Support boot from NAND flash

7.2 Pin Description

Following table list the NEMC pins.

Table 7-1 NEMC Pin Description

Pin Name	I/O	Signal	Description
Data Bus	I/O	SD15 – SD0	Data I/O.
Address bus	O	SA5–SA0	Address output.
Static chip select 6 ~ 1	O	CS6~1#	Chip select signal that indicates the static bank being accessed.
Read enable	O	RD# /	For Static memory read enable signal.
Write enable	O	WE# /	Static memory write enable signal.
Wait	I	Wait# /	External wait state request signal for memory-like devices.
NAND flash read enable	O	FRE#	NAND flash read enable signal.
NAND flash write enable	O	FWE#	NAND flash write enable signal.
NAND flash ready/busy	I	FRB#	Indicates NAND flash is ready or busy. (When Nand flash boot, GPC30 is used as FRB# of CS1#)

7.3 Physical Address Space Map

Both virtual spaces and physical spaces are 32-bit wide in this architecture. Virtual addresses are translated by MMU into physical address which is further divided into several partitions for static memory, SDRAM, and internal I/O.

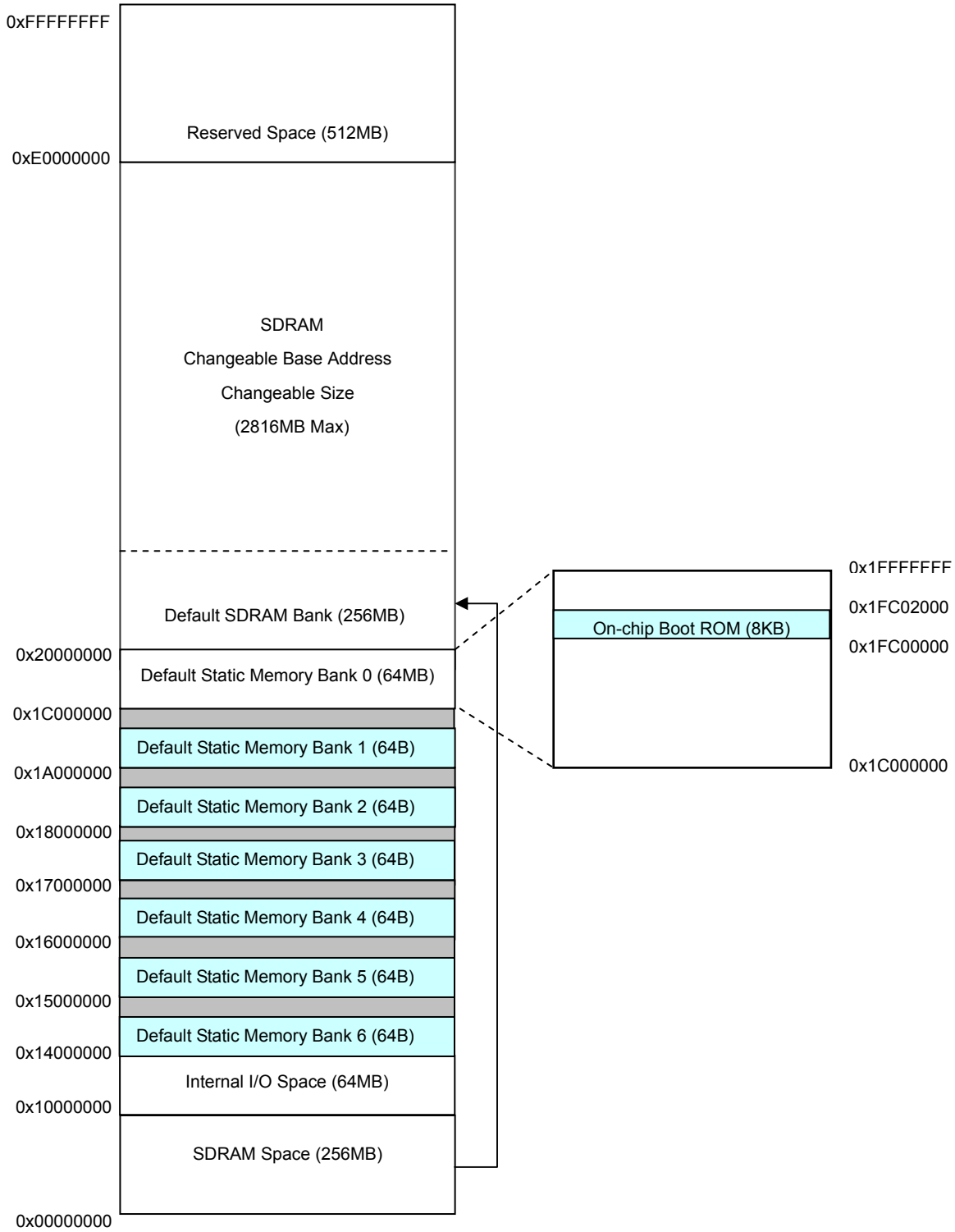


Figure 7-1 Physical Address Space Map

Table 7-2 Physical Address Space Map

Start Address	End Address	Connectable Memory	Capacity
0x00000000	0x0FFFFFFF	SDRAM Memory	256
0x10000000	0x10FFFFFF	I/O Devices on APB Bus	16
0x11000000	0x12FFFFFF	Reserved	32
0x13000000	0x13FFFFFF	I/O Devices on AHB Bus	16
0x14000000	0x1400003F	Static Memory, CS6#	64B
0x14000040	0x14FFFFFF	Reserved	
0x15000000	0x1500003F	Static Memory, CS5#	64B
0x15000040	0x15FFFFFF	Reserved	
0x16000000	0x1600003F	Static Memory, CS4#	64B
0x16000040	0x16FFFFFF	Reserved	
0x17000000	0x1700003F	Static Memory, CS3#	64B
0x17000040	0x17FFFFFF	Reserved	
0x18000000	0x1800003F	Static Memory, CS2#	64B
0x18000040	0x19FFFFFF	Reserved	
0x1A000000	0x1A00003F	Static Memory, CS1#	64B
0x1A000040	0x1BFFFFFF	Reserved	
0x1C000000	0x1FBFFFFF	Reserved	60
0x1FC00000	0x1FC01FFF	On-chip Boot ROM (8kB)	0.008
0x1FC02000	0x1FFFFFFF	Reserved	3.992
0x20000000	0xDFFFFFFF	SDRAM Memory	3072
0xE0000000	0xFFFFFFFF	Reserved	512

The base address and size of each memory banks are configurable. Software can re-configure these memory banks according to the actual connected memories. Following table lists the default configuration after reset.

Table 7-3 Default Configuration of NEMC Chip Select Signals

Chip-Select Signal	Connected Memory	Capacity	Memory Width *1	Start Address	End Address
CS1#	Static memory bank 1	64 B	8, 16, 32	0x1A000000	0x1A00003F
CS2#	Static memory bank 2	64 B	8, 16, 32	0x18000000	0x1800003F
CS3#	Static memory bank 3	64 B	8, 16, 32	0x17000000	0x1700003F
CS4#	Static memory bank 4	64 B	8, 16, 32	0x16000000	0x1600003F
CS5#	Static memory bank 5	64 B	8, 16, 32	0x15000000	0x1500003F
CS6#	Static memory bank 6	64 B	8, 16, 32	0x14000000	0x1400003F

NOTES:

1 Data width of static memory banks can be configured to 8, 16 bits by software.

2 The 8KB address space from H'1FC00000 to H'1FC01FFF in bank 0 is mapped to on-chip boot ROM. The other memory spaces in bank 0 are not used.

7.4 Static Memory Interface

NEMC provides a glueless interface to normal static memory which don't need byte control like SRAM, memory interface IO devices, etc.. It can directly control up to 6 devices using six chip select lines. Additional devices may be supported through external decoding of the address bus.

Each chip select can directly access memory or IO devices that are 8-bits or 16-bits wide. Each device connected to a chip select line has 2 associated registers that control its operation and the access timing to the external device. The Static Memory Control Register SMCRn specifies various configurations for the device. The Static Memory Address Configuration Register SACRn specifies the base address and size for each device, enabling any device to be located anywhere in the physical address range.

The static memory interface includes the following signals:

- Six chip selects, CS6~1#
- Six address signals, SA5-SA0
- One read enable, RD#
- One write enable, WE#
- One wait pin, WAIT#

The SMT field in SMCRn registers specifies the type of memory and BW field specifies the bus width. BOOT_SEL[1:0] pin defines whether system boot from Nor or Nand flash and the page size when boot from Nand flash.

7.4.1 Register Description

Table 7-4 Static Memory Interface Registers

Name	Description	RW	Reset Value	Address	Access Width
SMCR1	Static memory control register 1	RW	0x0FFF7700	0x13410014	32
SMCR2	Static memory control register 2	RW	0x0FFF7700	0x13410018	32
SMCR3	Static memory control register 3	RW	0x0FFF7700	0x1341001C	32
SMCR4	Static memory control register 4	RW	0x0FFF7700	0x13410020	32
SMCR5	Static memory control register 5	RW	0x0FFF7700	0x13410024	32
SMCR6	Static memory control register 6	RW	0x0FFF7700	0x13410028	32
SACR1	Static memory bank 1 address configuration register	RW	0x00001AFE	0x13410034	32
SACR2	Static memory bank 2 address configuration register	RW	0x000018FE	0x13410038	32
SACR3	Static memory bank 3 address configuration register	RW	0x000017FF	0x1341003C	32
SACR4	Static memory bank 4 address configuration register	RW	0x000016FF	0x13410040	32
SACR5	Static memory bank 5 address configuration register	RW	0x000015FF	0x13410044	32
SACR6	Static memory bank 6 address configuration register	RW	0x000014FF	0x13410048	32

7.4.1.1 Static Memory Control Register (SMCR1~6)

SMCR1~6 are 32-bit read/write registers that contain control bits for static memory. On reset, SMCR1~6 are initialized to 0x0FFF7700.

SMCR1	**0x13410014**
SMCR2	**0x13410018**
SMCR3	**0x1341001C**
SMCR4	**0x13410020**
SMCR5	**0x13410024**
SMCR6	**0x13410028**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					STRV				TAW				TBP					TAH				TAS			BW					BL		SMT
RST	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	0	1	1	1	0/x	0/x	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	Writes to these bits have no effect and always read as 0.	R
27:24	STRV	**Static Memory Recovery Time.** Its value is the number of idle cycles (0~15 cycles) inserted between bus cycles when switching from one bank to another bank or between a read access to a write access in the same bank. Its initial value is 0xF (15 cycles).	RW
23:20	TAW	**Access Wait Time.** For normal memory, these bits specify the number of wait cycles to be inserted in read strobe time. For burst ROM, these bits specify the number of wait cycles to be inserted in first data read strobe time. **TAW3~0 Wait cycle Wait# Pin** 0000 0 cycle Ignored 0001 1 cycle Enabled 0010 2 cycles Enabled 0011 3 cycles Enabled 0100 4 cycles Enabled 0101 5 cycles Enabled 0110 6 cycles Enabled 0111 7 cycles Enabled 1000 8 cycles Enabled 1001 9 cycles Enabled 1010 10 cycles Enabled 1011 12 cycles Enabled 1100 15 cycles Enabled 1101 20 cycles Enabled	RW

		1110 25 cycles Enabled 1111 31 cycles Enabled (Initial Value)	
19:16	TBP	**Burst Pitch Time.** For burst ROM, these bits specify the number of wait cycles to be inserted in subsequent access. For normal memory, these bits specify the number of wait cycles to be inserted in write strobe time. **TBP3~0 Wait cycle Wait# Pin** 0000 0 cycle Ignored 0001 1 cycle Enabled 0010 2 cycles Enabled 0011 3 cycles Enabled 0100 4 cycles Enabled 0101 5 cycles Enabled 0110 6 cycles Enabled 0111 7 cycles Enabled 1000 8 cycles Enabled 1001 9 cycles Enabled 1010 10 cycles Enabled 1011 12 cycles Enabled 1100 15 cycles Enabled 1101 20 cycles Enabled 1110 25 cycles Enabled 1111 31 cycles Enabled (Initial Value)	RW
15	Reserved	Writes to these bits have no effect and always read as 0.	R
14:12	TAH	**Address Hold Time.** These bits specify the number of wait cycles to be inserted from negation of read/write strobe to address. **TAH2~0 Wait cycle** 000 0 cycle 001 1 cycle 010 2 cycles 011 3 cycles 100 4 cycles 101 5 cycles 110 6 cycles 111 7 cycles (Initial Value)	RW
11	Reserved	Writes to these bits have no effect and always read as 0.	R
10:8	TAS	**Address Setup Time.** These bits specify the number of wait cycles (0~7 cycles) to be inserted from address to assertion of read/write strobe. **TAS2~0 Wait cycle** 000 0 cycle 001 1 cycle	RW

		010 2 cycles 011 3 cycles 100 4 cycles 101 5 cycles 110 6 cycles 111 7 cycles (Initial Value)	
7:6	BW	**Bus Width.** These bits specify the bus width. this filed is writeable and are initialized to 0 by a reset. **BW1~0 Bus Width** 00 8 bits (Initial Value) 01 16 bits 10 Reserved 11 Reserved	RW
5:3	Reserved	Writes to these bits have no effect and always read as 0. **NOTE:** Don't write Bit3 to 1.	R
2:1	BL	**Burst Length (BL1, BL0).** When Burst ROM is connected; these bits specify the number of burst in an access. These bits are only valid when SMT is set to 1. **BL1~0 Burst Length** 00 4 consecutive accesses. Can be used with 8- or 16-bit bus width (Initial Value) 01 8 consecutive accesses. Can be used with 8- or 16-bit bus width 10 16 consecutive accesses. Can only be used with 8- or 16-bit bus width 11 32 consecutive accesses. Can only be used with 8-bit bus width	
0	SMT	**Static Memory Type (SMT).** This bit specifies the type of static memory. **SMT Description** 0 Normal Memory (Initial Value) 1 Burst ROM	RW

7.4.1.2 Static Bank Address Configuration Register (SACR1~6)

SACR1~6 defines the physical address for static memory bank 1 to 6, respectively. Each register contains a base address and a mask. When the following equation is met:

(*physical_address* [31:24] & **MASK$_n$**) == **BASE$_n$**

The bank n is active. The *physical_address* is address output on internal system bus. Static bank regions must be programmed so that each bank occupies a unique area of the physical address space.

Programming overlapping bank regions will result in unpredictable error. These registers are initialized by a reset.

SACR1	**0x13410034**
SACR2	**0x13410038**
SACR3	**0x1341003C**
SACR4	**0x13410040**
SACR5	**0x13410044**
SACR6	**0x13410048**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																	BASE								MASK								
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0																

Bits	Name	Description	RW
31:16	Reserved	Writes to these bits have no effect and read always as 0.	R
15:8	BASE	**Address Base:** Defines the base address of Static Bank n (n = 1 to 6). The initial values are: SACR1.BASE 0x1A SACR2.BASE 0x18 SACR3.BASE 0x17 SACR4.BASE 0x16 SACR5.BASE 0x15 SACR6.BASE 0x14	RW
23:20	MASK	**Address Mask**: Defines the mask of Static Bank n (n = 1 to 6). The initial values are: SACR1.MASK 0xFE SACR2.MASK 0xFE SACR3.MASK 0xFF SACR4.MASK 0xFF SACR5.MASK 0xFF SACR6.MASK 0xFF	RW

7.4.2 Example of Connection

Following figures shows examples of connection to 16- and 8-bit data width normal memory.

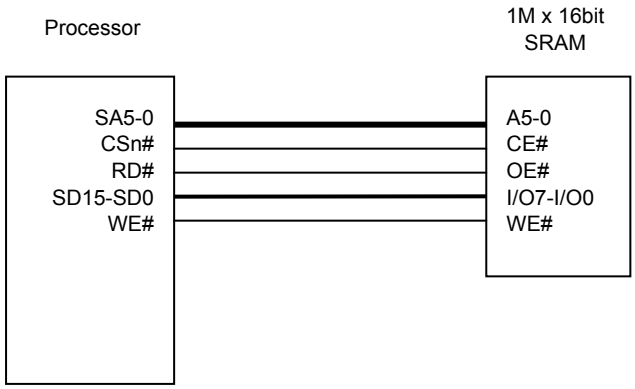


Figure 7-2 Example of 16-Bit Data Width SRAM Connection

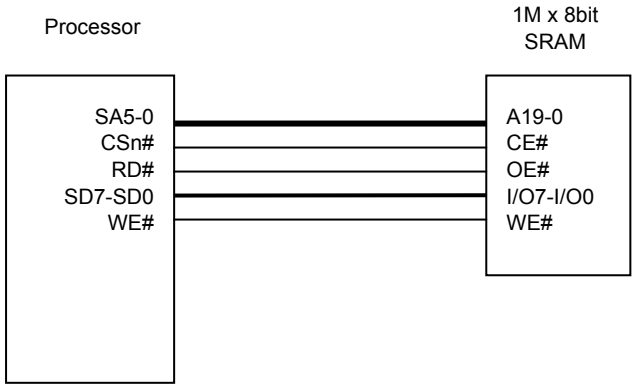


Figure 7-3 Example of 8-Bit Data Width SRAM Connection

7.4.3 Basic Interface

When SMT field in SMCRn (n = 1 to 6) is 0, normal memory (non-burst ROM, Flash, normal SRAM or memory-like device) is connected to bank n. When bank n (n = 1 to 6) is accessed, CSn# is asserted as soon as address is output. In addition, the RD# signal, which can be used as OE#, and write control signals WE# is asserted.

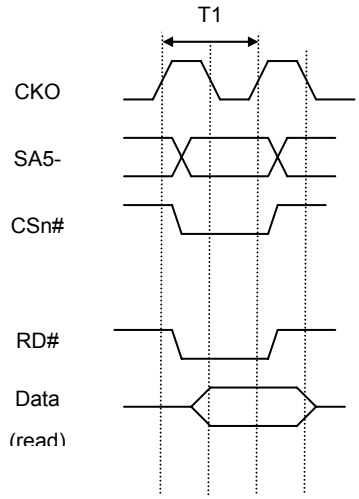
The TAS field in SMCRn is the latency from CSn# to read/write strobe. The TAW3 field is the delay time of RD# in read access. TBP3~0 field is the delay time of WE# and WEn# in write access. In addition, any number of waits can be inserted by means of the external pin (WAIT#). The TAH field is the latency from RD# and WEn# negation to CSn# negation, also the hold time to address and write data.

All kinds of normal memories (non-burst ROM, normal SRAM and Flash) have the same read and write timing. There are some requirements for writes to flash memory. Flash memory space must be un-cacheable and un-buffered. Writes must be exactly the width of the populated Flash devices on the data bus (no byte writes or word writes to a 16-bit bus, and so on). Software is responsible for partitioning commands and data, and writing them out to Flash in the appropriate sequence.

Glossary

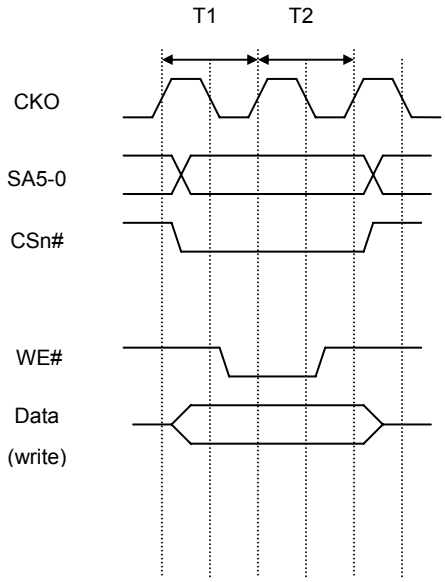
Th – hold cycle
Tw – wait cycle
Ts – setup cycle
T1 – read inherent cycle or first write inherent cycle
T2 – last write inherent cycle
Tb – burst read inherent cycle

Following figures show the timing of normal memory. A no-wait read access is completed in one cycle and a no-wait write access is completed in two cycles. Therefore, there is no negation period in case of access at minimum pitch.



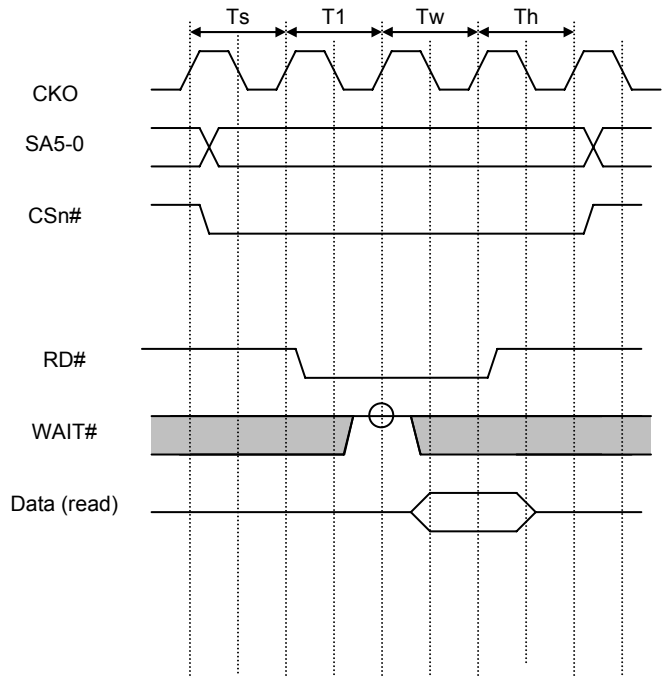
*In this example, SMCRn:MT = 0, TAS = 0, TAW = 0, TAH = 0

Figure 7-4 Basic Timing of Normal Memory Read



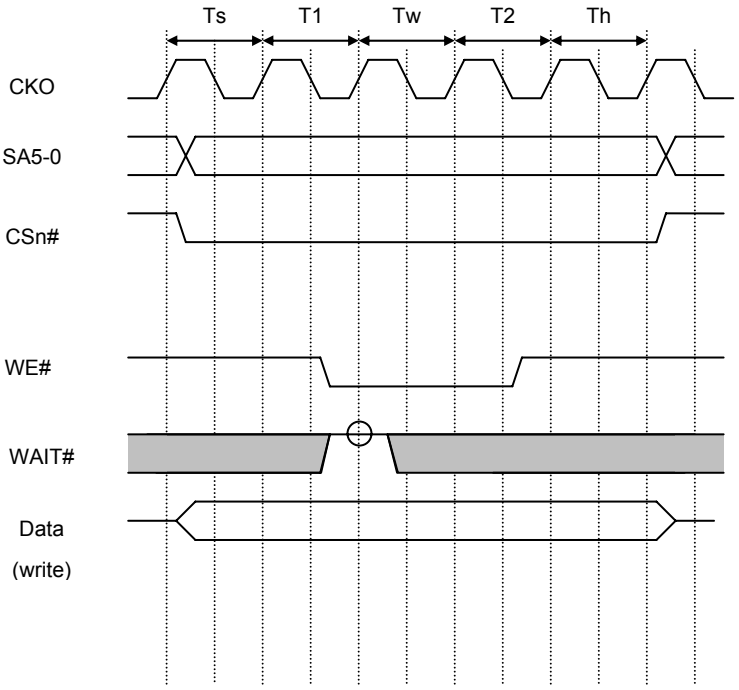
*In this example, SMCRn: SMT = 0, TAS = 0, TBP = 0, TAH = 0

Figure 7-5 Basic Timing of Normal Memory Write



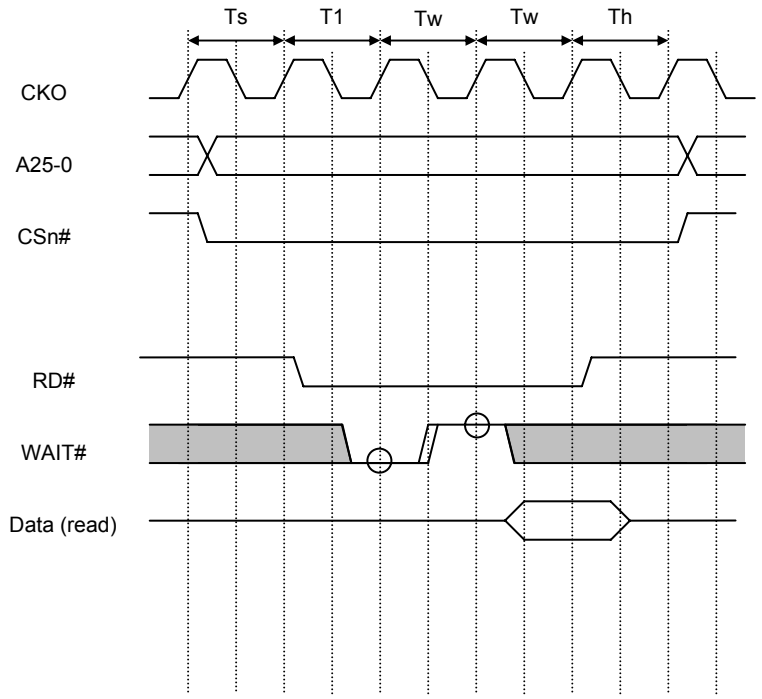
*In this example, SMCRn: SMT = 0, TAS = 1, TAW = 1, TAH = 1

Figure 7-6 Normal Memory Read Timing With Wait (Software Wait Only)



*In this example, SMCRn: SMT = 0, TAS = 1, TBP = 1, TAH = 1

Figure 7-7 Normal Memory Write Timing With Wait (Software Wait Only)



*In this example, SMCRn: SMT = 0, TAS = 1, TAW = 1, TAH=1

Figure 7-8 Normal Memory Read Timing With Wait (Wait Cycle Insertion by WAIT# pin)

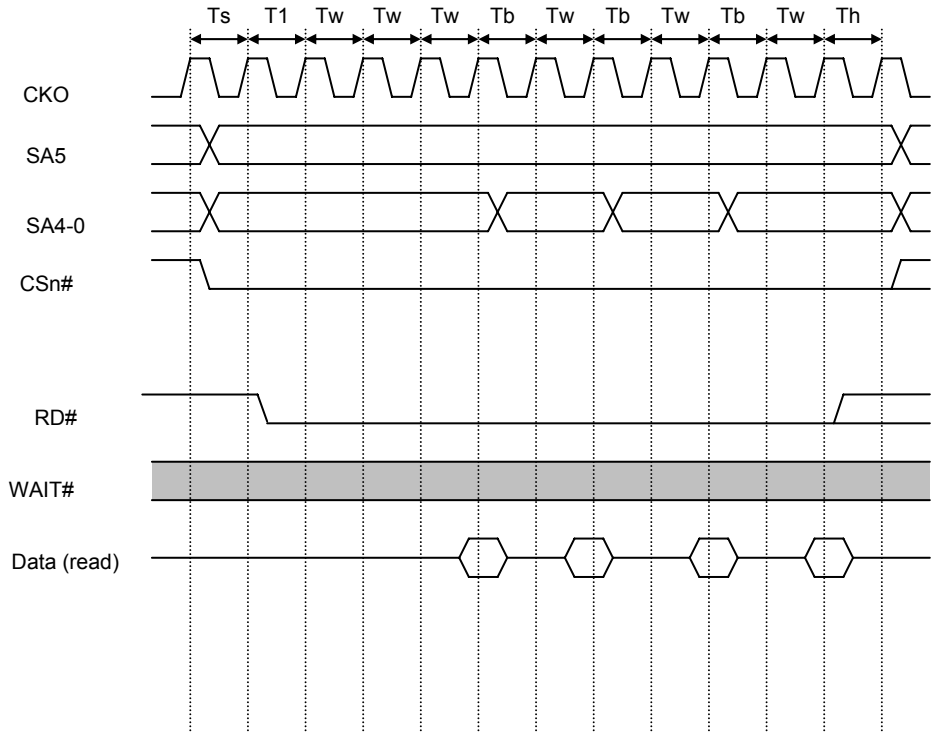
7.4.4 Burst ROM Interface

Setting SMT to 1 in SMCRn allows burst ROM to be connected to bank n (n = 1 to 6). The burst ROM interface provides high-speed access to ROM that has a nibble access function. Basically, access is performed in the same way as for normal memory, but when the first cycle ends, only the address is changed before the next access is executed. When 8-bit burst ROM is connected, the number of consecutive accesses can be set as 4, 8, 16, or 32 with bits BL1~0. When 16-bit ROM is connected, 4, 8, or 16 can be set in the same way.

For burst ROM read, TAW sets the delay time from read strobe to the first data, TBP sets the delay time from consecutive address to data. Burst ROM writes have the same timing as normal memory except TAW instead of TBP is used to set the delay time of write strobe.

WAIT# pin sampling is always performed when one or more wait states are set.

Following figures show the timing of burst ROM.



*In this example, SMT = 1, BL = 0, TAS = 1, TAW = 3,TBP = 1,TAH = 1

Figure 7-9 Burst ROM Read Timing (Software Wait Only)

7.5 NAND Flash Interface

NAND flash can be connected to static memory bank 6~ band 1. Both 8-bit and 16-bit NAND flashes are supported. A mechanism for booting from NAND flash is also supported.

7.5.1 Register Description

Table 7-5 NAND Flash Interface Registers

Name	Description	RW	Reset Value	Address	Access Width
NFCSR	NAND flash control/status register	RW	0x00000000	0x13410050	32
PNCR	NAND PN control register	RW	0x00000000	0x13410100	32
PNDR	NAND PN data register	RW	0x00005AA5	0x13410104	32
BITCNT	NAND bit counter	R	0x00000000	0x13410108	32

7.5.1.1 NAND Flash Control/Status Register (NFCSR)

NFCSR is a 32-bit read/write register that is used to configure NAND flash. It is initialized by any reset.

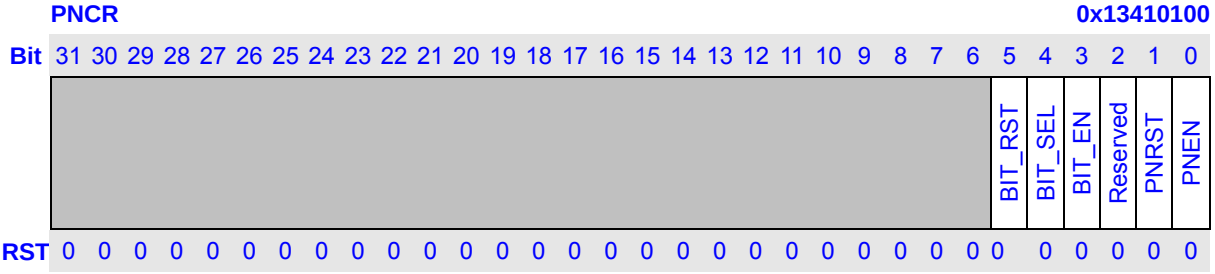
NFCSR 0x13410050

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																					NFCE6	NFE6	NFCE5	NFE5	NFCE4	NFE4	NFCE3	NFE3	NFCE2	NFE2	NFCE1	NFE1
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:16	Reserved	Writes to these bits have no effect and read always as 0.	R
1/3/5/7/9/11	FCEn (n=1,2,3,4,5,6)	**NAND Flash FCE# Assertion Control :** Controls the assertion of NAND Flash FCEn#. When set, FCEn# is always asserted until this bit is cleared. When the NAND flash require FCEn# to be asserted during read busy time, this bit should be set. **FCE** **Description** 0 FCEn# is asserted as normal static chip enable(Initial value) 1 FCEn# is always asserted	RW
0/2/4/6/8/10	NFEn (n=1,2,3,4,5,6)	**NAND Flash Enable:** Specifies if NAND flash is connected to static bank n. When system is configured to boot from NAND flash, this bit is initialized to 1. **NFE** **Description** 0 Static bank n is not used as NAND flash 1 Static bank n is used as NAND flash	RW

7.5.1.2 NAND PN Control Register (PNCR)

PNCR is a 32-bit read/write register that is used to control NAND flash data randomization. It is initialized by any reset.



Bits	Name	Description	RW
31:6	Reserved	Writes to these bits have no effect and read always as 0.	R
5	BIT_RST	**NAND BIT Counter Reset:** Reset Bit counter. When this bit is written to 1, the bit counter is reset to 0. This bit is write-only.	W
4	BIT_SEL	**NAND BIT Counter Select:** Bit counter's counting select. **BIT_SEL Description** 0 Calculate number of "0" in NAND read data 1 Calculate number of "1" in NAND read Data	RW
3	BIT_EN	**NAND BIT Counter Enable:** Enable/disable bit counter counting. **BIT_EN Description** 0 Bit counting is disabled 1 Bit counting is enabled	RW
2	Reserved	Writes to these bits have no effect and read always as 0.	R
1	PNRST	**NAND Flash PN Reset:** Reset seed of randomizer. When this bit is written to 1, the seed of randomizer is reset. This bit is write-only.	W
0	PNEN	**NAND Flash PN Enable:** Specifies if NAND flash read/write data randomization is enabled. This bit is initialized to 0. **PNEN Description** 0 Data randomization is disabled 1 Data randomization is enabled	RW

7.5.1.3 NAND PN Data Register (PNDR)

PNDR is a 23-bit read/write register that is used for seed of randomizer during NAND read/write.

PNDR **0x13410104**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										PNDR																						
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1

7.5.1.4 NAND Bit Counter (BITCNT)

BITCNT is a 32-bit read/write register that is used to counting the number of “1” or “0” (based on BIT_SEL) in Nand read data and keep counting during Nand read till BIT Counter Reset. It is initialized by any reset.

BITCNT **0x13410108**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																BITCNT																
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

7.5.2 NAND Flash Boot Loader

To support boot from NAND flash, 8KB on-chip Boot ROM is implemented. Following figure illustrates the structure of NAND Flash Boot Loader.

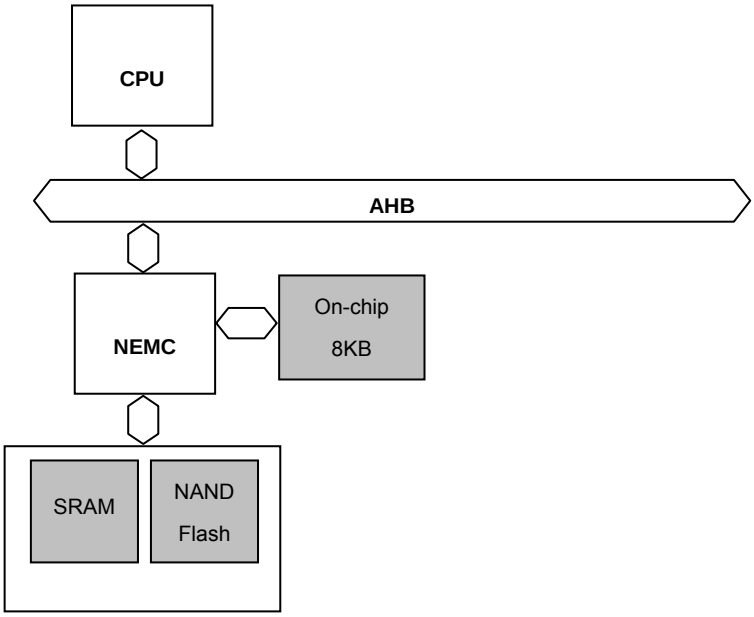


Figure 7-10 Structure of NAND Flash Boot Loader

When system is configured to boot from NAND flash, after reset, the program in Boot ROM is executed and the program will copy the first 8K bytes of NAND flash to internal memory for further initialization.

Generally, the boot code will copy more NAND flash content to DRAM. Then the main program will be executed on DRAM.

When system is configured to boot from NAND flash, software may know the nand flash page size through BOOT_SEL[2:0] pin.

7.5.3 NAND Flash Operation

Set NFEn bit of NAND Flash Control/Status Register (NFCSR) will enable access to NAND flash. The partition of static bank n (n=1~6) is changed as following figure. Writes to any of address space will be translated to NAND flash address cycle. Writes to any of command space will be translated to NAND flash command cycle. **CAUTION:** don't read to address and command space, and these two partitions should be uncacheable. Reads and writes to any of data space will be translated to NAND flash data read/write cycle. DMA access to data space is supported to increase the speed of data read/write. The DMA access cannot exceed the page boundary (512 bytes or 2K bytes) of NAND

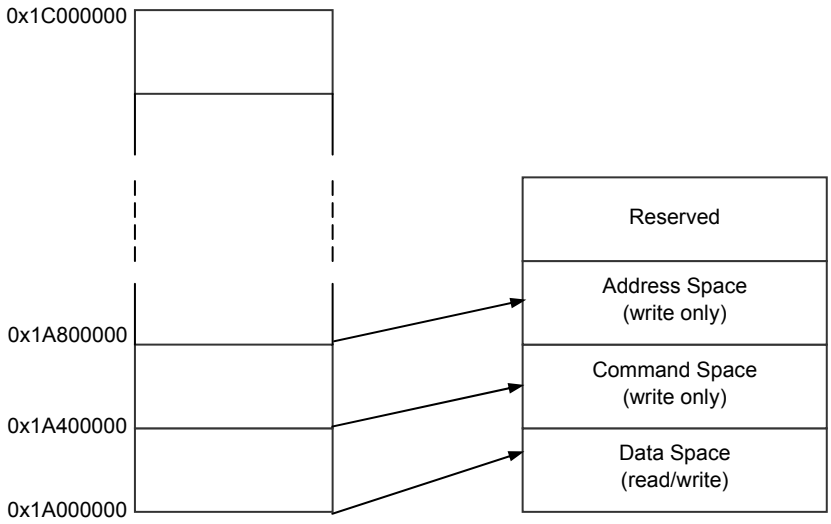


Figure 7-11 Static Bank 1 Partition When NAND Flash is Used (an example)

The timing of NAND flash access is configured by SMCRn and is same as normal static memory timing, except that CSn# is controlled by NFCE bit NFCSR. CSn# is always asserted when NFCE is 1. When NFCE is 0, CSn# is asserted as normal static memory access.

The control signals for direction connection of NAND flash are CSn#, FRE#, FWE#, FRB#(GPIO), A1 and A0. Following figure shows the connection between processor and NAND Flash.

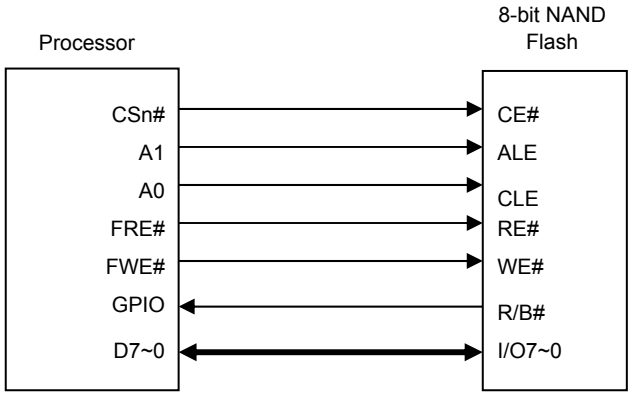


Figure 7-12 Example of 8-bit NAND Flash Connection

8 BCH Controller

8.1 Overview

The BCH Controller implements data ECC encoding and decoding.

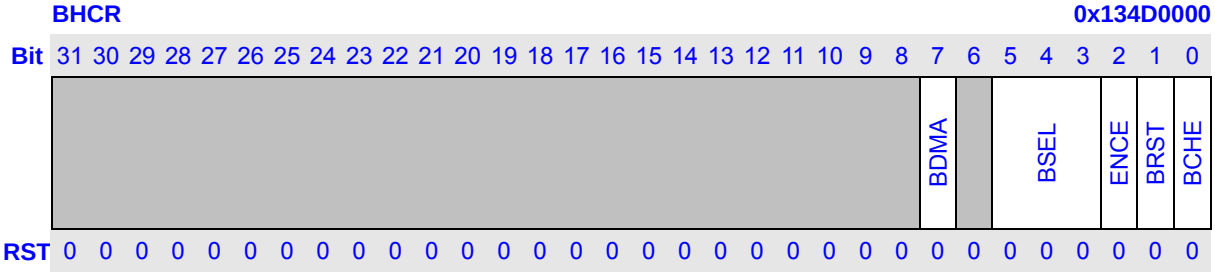
8.2 Register Description

Table 8-1 BCH Registers

Name	Description	RW	Reset Value	Address	Access Width
BHCR	BCH Control register	R	0x00000000	0x134D0000	32
BHCSR	BCH Control Set register	W	Undefined	0x134D0004	32
BHCCR	BCH Control Clear register	W	Undefined	0x134D0008	32
BHCNT	BCH ENC/DEC Count register	RW	0x00000000	0x134D000C	32/16
BHDR	BCH data register	W	Undefined	0x134D0010	8
BHPAR0	BCH Parity 0 register	RW	0x00000000	0x134D0014	32/16/8
BHPAR1	BCH Parity 1 register	RW	0x00000000	0x134D0018	32/16/8
BHPAR2	BCH Parity 2 register	RW	0x00000000	0x134D001C	32/16/8
BHPAR3	BCH Parity 3 register	RW	0x00000000	0x134D0020	32/16/8
BHPAR4	BCH Parity 4 register	RW	0x00000000	0x134D0024	32/16/8
BHPAR5	BCH Parity 5 register	RW	0x00000000	0x134D0028	32/16/8
BHPAR6	BCH Parity 6 register	RW	0x00000000	0x134D002C	32/16/8
BHPAR7	BCH Parity 7 register	RW	0x00000000	0x134D0030	32/16/8
BHPAR8	BCH Parity 8 register	RW	0x00000000	0x134D0034	32/16/8
BHPAR9	BCH Parity 9 register	RW	0x00000000	0x134D0038	32/16/8
BHERR0	BCH Error Report 0 register	R	0x00000000	0x134D003C	32/16
BHERR1	BCH Error Report 1 register	R	0x00000000	0x134D0040	32/16
BHERR2	BCH Error Report 2 register	R	0x00000000	0x134D0044	32/16
BHERR3	BCH Error Report 3 register	R	0x00000000	0x134D0048	32/16
BHERR4	BCH Error Report 4 register	R	0x00000000	0x134D004C	32/16
BHERR5	BCH Error Report 5 register	R	0x00000000	0x134D0050	32/16
BHERR6	BCH Error Report 6 register	R	0x00000000	0x134D0054	32/16
BHERR7	BCH Error Report 7 register	R	0x00000000	0x134D0058	32/16
BHERR8	BCH Error Report 8 register	R	0x00000000	0x134D005C	32/16
BHERR9	BCH Error Report 9 register	R	0x00000000	0x134D0060	32/16
BHERR10	BCH Error Report 10 register	R	0x00000000	0x134D0064	32/16
BHERR11	BCH Error Report 11 register	R	0x00000000	0x134D0068	32/16
BHINT	BCH Interrupt Status register	R	0x00000000	0x134D006C	32
BHINTE	BCH Interrupt Enable register	RW	0x00000000	0x134D0070	32
BHINTES	BCH Interrupt Set register	W	Undefined	0x134D0074	32
BHINTEC	BCH Interrupt Clear register	W	Undefined	0x134D0078	32

8.2.1 BCH Control Register (BHCR)

BHCR is a 32-bit read/write register that is used to configure BCH controller. It is initialized by any reset.



Bits	Name	Description	RW
31:8	Reserved	Writes to these bits have no effect and read always as 0.	R
7	BDMA	**BCH DMA Enable**: It is used to enable or disable dma transfer during correction. **BDMA** **Description** 0 DMA transfer is disabled (Initial value) 1 DMA transfer is enabled	RW
6	Reserved	Writes to these bits have no effect and read always as 0.	R
5:3	BSEL	**BCH Encoding/Decoding Bit Select**: It is used to select the correction algorithm among 4-bit, 8-bit, 12-bit, 16-bit, 20-bit and 24-bit BCH. 000: 4-bit correction (initial value) 001: 8-bit correction 010: 12-bit correction 011: 16-bit correction 100: 20-bit correction 101: 24-bit correction	
2	ENCE	**BCH Encoding/Decoding Select**: It is used to define whether in encoding or in decoding phase when BCH is used. **ENCE** **Description** 0 Decoding (Initial value) 1 Encoding	RW
1	BRST	**BCH Reset:** It is used to reset BCH controller. This bit is cleared automatically by hardware and always read as 0. **BRST** **Description** 0 BCH controller is not reset (Initial value) 1 BCH controller is reset	W
0	BCHE	**BCH Enable:** BCH correction is enable/disable. **BCHE** **Description** 0 BCH is disabled (initial value) 1 BCH is enabled	RW

8.2.2 BCH Control Set Register (BHCSR)

BHCSR is a 32-bit write-only register that is used to set BCH controller to 1.

When write 1 to BHCSR, the corresponding bit in BHCR register is set to 1. Write 0 to BHCSR is ignored.

BHCSR **0x134D0004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																									BDMAS		BSELS			ENCES	BRSTS	BCHES
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:8	Reserved	Writes to these bits have no effect and read always as 0.	R
7	BDMAS	**BCH DMA Enable Set**: It is used to set BHCR.BDMA to 1.	W
6	Reserved	Writes to these bits have no effect and read always as 0.	R
5:3	BSELS	**BCH Encoding/Decoding Bit Select Set**: It is used to set BHCR.BSEL to 1.	W
2	ENCES	**BCH Encoding/Decoding Select Set**: It is used to set BHCR.ENCE to 1.	W
1	BRSTS	**BCH Reset Set:** It is used to set BHCR.BRST to 1.	W
0	BCHES	**BCH Enable Set:** It is used to set BHCR.BCHE to 1.	W

8.2.3 BCH Control Clear Register (BHCCR)

BHCCR is a 32-bit write-only register that is used to clear BCH controller to 0.

When write 1 to BHCCR, the corresponding bit in BHCR register is cleared to 0. Write 0 to BHCCR is ignored.

BHCCR **0x134D0008**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																									BDMAC		BSELC			ENCEC	BRSTC	BCHEC
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

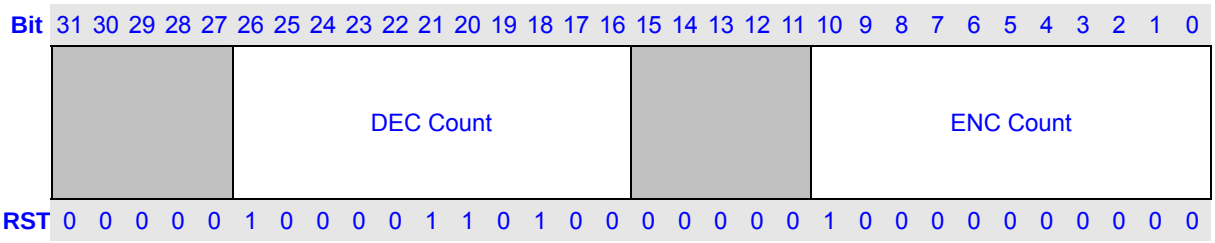
Bits	Name	Description	RW
31:8	Reserved	Writes to these bits have no effect and read always as 0.	R
7	BDMAC	**BCH DMA Enable Clear**: It is used to clear BHCR.BDMA to 0.	W
6	Reserved	Writes to these bits have no effect and read always as 0.	R
5:3	BSELC	**BCH Encoding/Decoding Bit Select Clear**: It is used to clear	W

		BHCR.BSEL to 0.	
2	ENCEC	**BCH Encoding/Decoding Select Clear**: It is used to clear BHCR.ENCE to 0.	W
1	Reserved	Writes to this bit have no effect and read always as 0.	R
0	BCHEC	**BCH Enable Clear:** It is used to clear BHCR.BCHE to 0.	W

8.2.4 BCH ENC/DEC Count Register (BHCNT)

BHCNT is a 32-bit read/write register that is used to indicate the total number of 4-bit data during encoding or decoding. It is initialized by any reset.

BHCNT **0x134D000C**

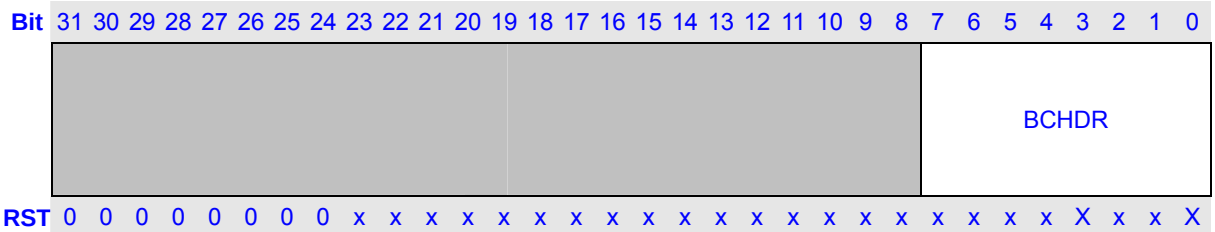


Bits	Name	Description	RW
31:27	Reserved	Writes to these bits have no effect and read always as 0.	R
26:16	DEC Count	**DEC Count**: It is used to indicate total 4-bit data count in BCH decoding which includes data + parity. For example, total data + parity is 538 bytes, the field of DEC Count should be set to 'h434 which is the initial value.	RW
15:11	Reserved	Writes to these bits have no effect and read always as 0.	R
10:0	ENC Count	**ENC Count**: It is used to indicate total byte count in BCH encoding which just includes 4-bit data and should be less and equal to 1996 4-bit (which is equal to 998 Bytes) when 16-bit BCH is selected.	RW

8.2.5 BCH Data Register (BHDR)

BHDR is an 8-bit write-only register that is used to transfer ecc data to BCH.

BHDR **0x134D0010**



8.2.6 BH Parity Register (BHPARn, n=0,1,2,3,4,5,6,7,8,9)

BHPARn (n=0,1,2,3,4,5,6,7,8,9) are all 32-bit read/write register that contains the encoding parity data during BCH correction. It is initialized by any reset and BRST of BHCR.

When 24-bit BCH is selected, BHPAR0~BHPAR9 consist of the 312 bits of parity data and bit 0 of BHPAR0 is the 312th bit of parity data and bit 23 of BHPAR9 is the 1st bit of parity data.

When 20-bit BCH is selected, BHPAR0~BHPAR8 consist of the 260 bits of parity data, and bit 0 of BHPAR0 is the 260th bit of parity data and bit 3 of BHPAR8 is the 1st bit of parity data.

When 16-bit BCH is selected, BHPAR0~BHPAR6 consist of the 208 bits of parity data, and bit 0 of BHPAR0 is the 208th bit of parity data and bit 15 of BHPAR6 is the 1st bit of parity data.

When 12-bit BCH is selected, BHPAR0~BHPAR4 consist of the 156 bits of parity data, and bit 0 of BHPAR0 is the 156th bit of parity data and bit 27 of BHPAR4 is the 1st bit of parity data.

When 8-bit BCH is selected, BHPAR0~BHPAR3 consist of the 104 bits of parity data and bit 0 of BHPAR0 is the 104th bit of parity data and bit 7 of BHPAR3 is the 1st bit of parity data.

Similarly, when 4-bit BCH is selected, the two parity register, BHPAR0 and BHPAR1 together consist of the 52 bits of parity data and bit 0 of BHPAR0 is the 52th bit of parity data and bit 19 of BHPAR1 is the 1st bit of parity data.

BHPAR0		**0x134D0014**
BHPAR1		**0x134D0018**
BHPAR2		**0x134D001C**
BHPAR3		**0x134D0020**
BHPAR4		**0x134D0024**
BHPAR5		**0x134D0028**
BHPAR6		**0x134D002C**
BHPAR7		**0x134D0030**
BHPAR8		**0x134D0034**
BHPAR9		**0x134D0038**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BHPAR0~BHPAR9																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

8.2.7 BCH Error Report Register (BHERRn, n=0,1,2,3,4,5,6,7,8,9,10,11)

BHERRn is 32-bit read/write register that contains the index for each error after BCH decoding. It is initialized by any reset and BRST of BHCR.

BHERR0 contains INDEX0 and INDEX1.

BHERR1 contains INDEX2 and INDEX3.
BHERR2 contains INDEX4 and INDEX5.
BHERR3 contains INDEX6 and INDEX7.
BHERR4 contains INDEX8 and INDEX9.
BHERR5 contains INDEX10 and INDEX11.
BHERR6 contains INDEX12 and INDEX13.
BHERR7 contains INDEX14 and INDEX15.
BHERR8 contains INDEX16 and INDEX17.
BHERR9 contains INDEX18 and INDEX19.
BHERR10 contains INDEX20 and INDEX21.
BHERR11 contains INDEX22 and INDEX23.

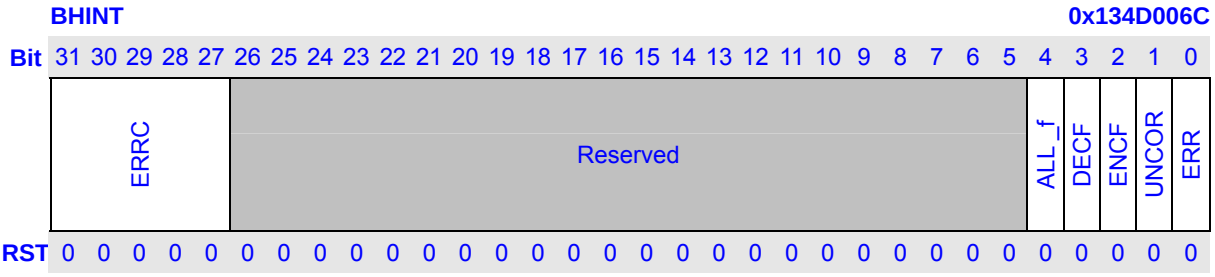
BHERR0	**0x134D003C**
BHERR1	**0x134D0040**
BHERR2	**0x134D0044**
BHERR3	**0x134D0048**
BHERR4	**0x134D004C**
BHERR5	**0x134D0050**
BHERR6	**0x134D0054**
BHERR7	**0x134D0058**
BHERR8	**0x134D005C**
BHERR9	**0x134D0060**
BHERR10	**0x134D0064**
BHERR11	**0x134D0068**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			INDEXn(n=0,2,4,6,8,10,12,14,16,18,20,22)													Reserved			INDEXn(n=1,3,5,7,9,11,13,15,17,19,21,23)													
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:29	Reserved	Writes to these bits have no effect and read always as 0.	R
28:16	INDEXn	**Error Bit Index:** It is used to indicate the location of the error bit. For example, INDEX=2, it means the second bit is an error bit.	R
15:13	Reserved	Writes to these bits have no effect and read always as 0.	R
12:0	INDEXn	**Error Bit Index:** It is used to indicate the location of the error bit. For example, INDEX=2, it means the second bit is an error bit.	R

8.2.8 BCH Interrupt Status Register (BHINT)

BHINT is a 32-bit read-only register that contains the interrupt flag and error count information during BCH correction. It is initialized by any reset. Software write 1 to clear the corresponding bit except ERRC.



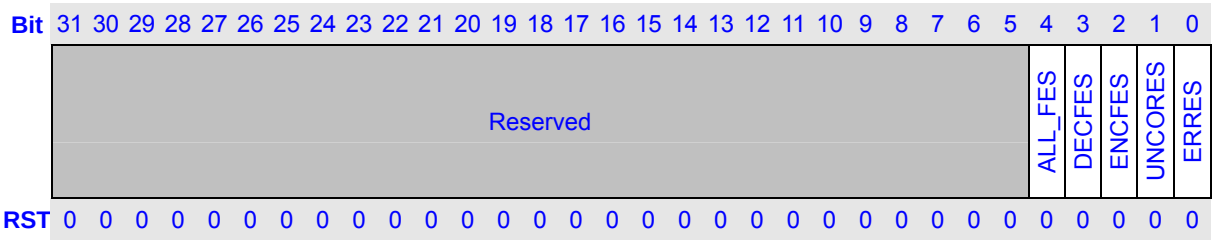
Bits	Name	Description	RW
31:27	ERRC	**Error Count:** It indicates the number of errors in the data block and these bits are also reset by BHCR.BRST bit. **ERRC** **Description** 0 No errors or uncorrection error occurs (Initial value) 1 One error in the data block 2 Two errors in the data block 3 Three errors 4 Four errors 5 Five errors 6 Six errors 7 Seven errors 8 Eight errors … 24 Twenty-four errors	R
27:5	Reserved	Writes to these bits have no effect and read always as 0.	R
4	ALL_f	**ALL_f:** It indicates that all data received during decoding are 0xf. When receiving all 0xf data, BCH doesn't correct the data and no error occurs. **ALL_f** **Description** 0 Not all data (data + parity bytes) are 0xf (Initial value) 1 All data (data + parity bytes) are 0xf	R
3	DECF	**Decoding Finish:** It indicates that hardware finish BCH decoding. **DECF** **Description** 0 Decoding not Finish (Initial value) 1 Decoding Finish	R
2	ENCF	**Encoding Finish:** It indicates that hardware finish BCH encoding. **ENCF** **Description** 0 Encoding not Finish (Initial value) 1 Encoding Finish	R
1	UNCOR	**Uncorrection Error:** It indicates that hardware finish BCH encoding.	R

		UNCOR **Description** 0 No uncorrectable error (Initial value) 1 Uncorrectable error occur	
0	ERR	**Error:** It indicates that hardware detects error bits in data in the data block during BCH decoding. **ERR** **Description** 0 No error (Initial value) 1 Error occur	R

8.2.9 BCH Interrupt Enable Set Register (BHINTES)

BHINTES is a 32-bit write-only register that is used to set BHINTE register. Writing 1 to BHINTES will set the corresponding bit in BHINTE to 1. Writing 0 to BHINTES is ignored.

BHINTES **0x134D0074**



Bits	Name	Description	RW
31:5	Reserved	Writes to these bits have no effect and read always as 0.	R
4	ALL_FES	**ALL_F Interrupt Enable Set:** It is used to set BHINTE.ALL_FE to 1.	W
3	DECFES	**Decoding Finish Interrupt Enable Set:** It is used to set BHINTE.DECFE to 1.	W
2	ENCFES	**Encoding Finish Interrupt Enable Set:** It is used to set BHINTE.ENCFE to 1.	W
1	UNCORES	**Uncorrection Error Interrupt Enable Set:** It is used to set BHINTE.ENCFE to 1.	W
0	ERRES	**Error Interrupt Enable Set:** It is used to set BHINTE.ERRE to 1.	W

8.2.10 BCH Interrupt Enable Clear Register (BHINTEC)

BHINTEC is a 32-bit write-only register that is used to clear BHINTE register. Writing 1 to BHINTEC will clear the corresponding bit in BHINTE to 0. Writing 0 to BHINTEC is ignored.

BHINTEC **0x134D0078**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																											ALL_FEC	DECFEC	ENCFEC	UNCORC	ERREC
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:5	Reserved	Writes to these bits have no effect and read always as 0.	R
4	ALL_FEC	**ALL_F Interrupt Enable Clear:** It is used to clear BHINTE.ALL_FE to 0.	W
3	DECFEC	**Decoding Finish Interrupt Enable Clear:** It is used to clear BHINTE.DECFE to 0.	W
2	ENCFEC	**Encoding Finish Interrupt Enable Clear:** It is used to clear BHINTE.ENCFE to 0.	W
1	UNCOREC	**Uncorrection Error Interrupt Enable Clear:** It is used to clear BHINTE.ENCFE to 0.	W
0	ERREC	**Error Interrupt Enable Clear:** It is used to set BHINTE.ERRE to 0.	W

8.2.11 BCH Interrupt Enable Register (BHINTE)

BHINTE is a 32-bit read/write register that is used to enable/disable interrupts during BCH correction. It is initialized by any reset.

BHINTE **0x134D0070**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																											ALL_FE	DECFE	ENCFE	UNCORE	ERRE
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:5	Reserved	Writes to these bits have no effect and read always as 0.	R
4	ALL_FE	**ALL_F Interrupt Enable:** It is used enable or disable all_f data interrupt. **ALL_FE Description** 0 Disable ALL_F data interrupt (Initial value)	RW

		1 Enable ALL_F data interrupt	
3	DECFE	**Decoding Finish Interrupt Enable:** It is used to enable or disable decoding finish interrupt. **DECFE Description** 0 Disable Decoding Finish Interrupt (Initial value) 1 Enable Decoding Finish Interrupt	RW
2	ENCFE	**Encoding Finish Interrupt Enable:** It is used to enable or disable encoding finish interrupt. **ENCFE Description** 0 Disable Encoding Finish Interrupt (Initial value) 1 Enable Encoding Finish Interrupt	RW
1	UNCORE	**Uncorrection Error Interrupt Enable:** It is used to enable or disable uncorrection error interrupt. **UNCORE Description** 0 Disable Uncorrectable Error interrupt (Initial value) 1 Enable Uncorrectable Error Interrupt	RW
0	ERRE	**Error Interrupt Enable:** It is used to enable or disable error interrupt. **ERRE Description** 0 Disable Error interrupt (Initial value) 1 Enable Error interrupt	RW

8.3 BCH Operation

BCH controller uses BCH(n, k) codes. Here n is less and equal to 8191-bit and k is less and equal to 7879-bit in 24-bit correction, 7931-bit in 20-bit correction, 7983-bit in 16-bit correction, 8035-bit in 12-bit correction, 8087-bit in 8-bit correction and 8139-bit in 4-bit correction. During encoding, hardware will generate 312-bit parity data in 24-bit correction, 260-bit parity data in 20-bit correction, 208-bit parity data in 16-bit correction, 156-bit parity data in 12-bit correction, 104-bit parity data in 8-bit correction or 52-bit parity data in 4-bit correction. Parity data can be read out by cpu or dma. During decoding, if there are error bits in data block, after decoding BHERRn registers will hold the error bit location that can be read by cpu or dma.

8.3.1 Encoding Sequence

BCH encoding can be operated by cpu or dma.

8.3.1.1 CPU

1 Set BHCR.BCHE to 1 to enable BCH controller.
2 Select 24-bit, 20-bit, 16-bit, 12-bit, 8-bit or 4-bit correction by setting BHCR.BSEL.
3 Set BHCR.ENCE to 1 to enable encoding.
4 Set BHCR.BRST to 1 to reset BCH controller.
5 Set BHCNT.ENC_COUNT to data block size in bytes.
6 Byte-write all data block to BHDR.
7 Check BHINTS.ENCF bit or by enabling encoding finish interrupt.
8 When encoding finishes, read out the parity data in BHPARn.

8.3.1.2 DMA

1 Set BHCR.BCHE to 1 to enable BCH controller.
2 Select 24-bit, 20-bit, 16-bit, 12-bit, 8-bit or 4-bit correction by setting BHCR.BSEL.
3 Set BHCR.ENCE to 1 to enable encoding.
4 Set BHCR.BRST to 1 to reset BCH controller.
5 Set BHCNT.ENC_COUNT to data block size in bytes.
6 Set BHCR.BDMA to 1 to select DMA transfer.
7 Start DMA transfer after configuring DMA channel.
8 DMA read data block from system memory and write to BCH controller automatically.
9 DMA will wait BCH encoding request when finishes writing data block.
10 BCH controller will issue encoding request to DMA when encoding ends.
11 DMA start to read out parity data.
12 After parity data is read out, BCH automatically reset itself and clear BHINT.ENCF.

NOTES:

1 When DMA is enabled, software should guarantee not to enable encoding finish interrupt.

8.3.2 Decoding Sequence

BCH decoding can be operated by cpu or dma.

8.3.2.1 CPU

1 Set BHCR.BCHE to 1 to enable BCH controller.
2 Select 24-bit, 20-bit, 16-bit, 12-bit, 8-bit or 4-bit correction by setting BHCR.BSEL.
3 Clear BHCR.ENCE to 0 to enable decoding.
4 Set BHCR.BRST to 1 to reset BCH controller.
5 Set BHCNT.DEC_COUNT to data block size in bytes.
6 Byte-write all data block to BHDR.
7 Check BHINTS.DECF bit or by enabling decoding finish interrupt.
8 When decoding finishes, read out the status in BHINT and error report in BHERRn.

8.3.2.2 Decoding Sequence

1 Set BHCR.BCHE to 1 to enable BCH controller.
2 Select 24-bit, 20-bit, 16-bit, 12-bit, 8-bit or 4-bit correction by setting BHCR.BSEL.
3 Clear BHCR.ENCE to 0 to enable decoding.
4 Set BHCR.BRST to 1 to reset BCH controller.
5 Set BHCNT.DEC_COUNT to data block size in bytes.
6 Set BHCR.BDMA to 1 to select DMA transfer.
7 Start DMA transfer after configuring DMA channel.
8 DMA read data block from system memory and write to BCH controller automatically.
9 DMA will wait BCH decoding request when finishes writing data block.
10 BCH controller will issue decoding request to DMA when decoding ends.
11 DMA start to read out bch int status and error report data and write to memory.
12 If using descriptor DMA, if the data block needs error correction, the current data block.
13 syndrome generation and last data block error correction can be executed in pipeline.
14 automatically by DMA.
15 After status and error report data is read out, BCH automatically reset itself and clear BHINT.DECF and Error status in BHINT.

9 BDMA Controller

BDMA controller (BDMAC) is dedicated to transfer data between BCH, external memories and memory-mapped external devices.

9.1 Features

- Support up to 3 independent DMA channels
- Descriptor or No-Descriptor Transfer
- Transfer data units: byte, 2-byte (half word), 4-byte (word), 16-byte, 32-byte or 64-byte
- Transfer number of data unit: 1 ~ 2^{24}
- Independent source and target port width: 8-bit, 16-bit, 32-bit

9.2 Register Descriptions

Table 9-1 BDMAC Registers

Name	Description	RW	Reset Value	Address	Access Size (bit)
DSA0	DMA Source Address 0	RW	0x0	0x13450000	32
DTA0	DMA Target Address 0	RW	0x0	0x13450004	32
DTC0	DMA Transfer Count 0	RW	0x0	0x13450008	32
DRT0	DMA Request Source 0	RW	0x0	0x1345000C	32
DCS0	DMA Channel Control/Status 0	RW	0x0	0x13450010	32
DCM0	DMA Command 0	RW	0x0	0x13450014	32
DDA0	DMA Descriptor Address 0	RW	0x0	0x13450018	32
DSD0	DMA Stride Address 0	RW	0x0	0x1345001C	32
DSA1	DMA Source Address 1	RW	0x0	0x13450020	32
DTA1	DMA Target Address 1	RW	0x0	0x13450024	32
DTC1	DMA Transfer Count 1	RW	0x0	0x13450028	32
DRT1	DMA Request Source 1	RW	0x0	0x1345002C	32
DCS1	DMA Channel Control/Status 1	RW	0x0	0x13450030	32
DCM1	DMA Command 1	RW	0x0	0x13450034	32
DDA1	DMA Descriptor Address 1	RW	0x0	0x13450038	32
DSD1	DMA Stride Address 1	RW	0x0	0x1345003C	32
DSA2	DMA Source Address 2	RW	0x0	0x13450040	32
DTA2	DMA Target Address 2	RW	0x0	0x13450044	32
DTC2	DMA Transfer Count 2	RW	0x0	0x13450048	32
DRT2	DMA Request Source 2	RW	0x0	0x1345004C	32
DCS2	DMA Channel Control/Status 2	RW	0x0	0x13450050	32
DCM2	DMA Command 2	RW	0x0	0x13450054	32
DDA2	DMA Descriptor Address 2	RW	0x0	0x13450058	32
DSD2	DMA Stride Address 2	RW	0x0	0x1345005C	32
DNT0	DMA Nand Timer 0	RW	0xC	0x134500C0	32
DNT1	DMA Nand Timer 1	RW	0xC	0x134500C4	32
DNT2	DMA Nand Timer 2	RW	0xC	0x134500C8	32

DMAC1	DMA Control 1 Register	R/W	0x0	0x13450300	32
DIRQP1	DMA Interrupt Pending 1	R	0x0	0x13450304	32
DDR1	DMA Doorbell 1 Register	RW	0x0	0x13450308	32
DDRS1	DMA Doorbell Set 1 Register	W	0x0	0x1345030C	32
DCKE1	DMA Clock Enable 1 Register	W	0x0	0x13450310	32

9.2.1 DMA Source Address (DSAn, n = 0 ~ 2)

DSA0, DSA1, DSA2 **0x13450000, 0x13450020, 0x13450040**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SA																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	SA	Source physical address.	RW

9.2.2 DMA Target Address (DTAn, n = 0 ~ 2)

DTA0, DTA1, DTA2 **0x13450004, 0x13450024, 0x13450044**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TA																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	TA	Target physical address.	RW

9.2.3 DMA Transfer Count (DTCn, n = 0 ~ 2)

DTC0, DTC1, DTC2 **0x13450008, 0x13450028, 0x13450048**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								TC																							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:24	Reserved	Write has no effect, read as zero.	R
23:0	TC	When Stride address transfer is disabled: TC hold the number of data unit to transfer and it counts down to 0 at the end; When Stride address transfer is enabled: TC composes of two parts: The lower 16 bits: the number of data unit for sub-block transfer The higher 8 bits: the number of sub-block And both the two parts count down to 0 at the end.	RW

9.2.4 DMA Request Types (DRTn, n = 0 ~ 2)

DRT0, DRT1, DRT2 **0x1345000c, 0x1345002c, 0x1345004c**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																										RT					
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:6	Reserved	Write has no effect, read as zero.	R
5:0	RT	Transfer request type.	RW

Table 9-2 Transfer Request Types

RT5-0	Description
000000	Reserved.
000001	NAND DMA request. (external address → external address)
000010	BCH Encoding DMA request.
000011	BCH Decoding DMA request.
000100	Reserved.
000101	Reserved.
000110	Reserved.
000111	Reserved.
001000	Auto-request. (ignore RDIL3-0, external address → external address)
001001	Reserved.
001010	Reserved.
001011	Reserved.
001100	External request with DREQn. (external address ←→ external device with DACKn)
Other	Reserved.

NOTES:

1 Only auto request can be concurrently selected in all channels with different source and target address.

2 Only channel 1 and channel 2 can handle external request. Channel 2 handles external request 0, and channel 1 handles external request 1.

9.2.5 DMA Channel Control/Status (DCSn, n = 0 ~ 2)

DCS0, DCS1, DCS2 **0x13450010, 0x13450030, 0x13450050**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NDES	DES8	Reserved						CDOA								Reserved								BERR	Reserved		AR	TT	HLT	Reserved	CTE
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	NDES	Descriptor or No-Descriptor Transfer Select. 0: Descriptor Transfer; 1: No-descriptor Transfer.	RW
30	DES8	Descriptor 8 Word. 0: 4-word descriptor; 1: 8-word descriptor.	RW
29:24	Reserved	Write has no effect, read as zero.	R
23:16	CDOA	Copy of offset address of last completed descriptor from that in DMA command register. Software could know which descriptor is just	RW

		completed combining with count terminate interrupt resulted by DCSn.CT. (Ignored in No-Descriptor Transfer)	
15:8	Reserved	Write has no effect, read as zero.	R
7	BERR	BCH error. 0: no BCH error; 1: BCH error within this transfer. (Only channel 0 has this bit for BCH transfer)	RW
6:5	Reserved	Write has no effect, read as zero.	R
4	AR	Address Error. 0: no address error; 1: address error.	RW
3	TT	Transfer Terminate. 0: No-Link Descriptor or No-Descriptor DMA transfer does not end 1: No-Link Descriptor or No-Descriptor DMA transfer end	RW
2	HLT	DMA halt. 0: DMA transfer is in progress; 1: DMA halt.	RW
1	Reserved	Write has no effect, read as zero.	R
0	CTE	Channel transfer enable. 0: disable; 1: enable.	RW

9.2.6 DMA Channel Command (DCMn, n = 0 ~ 2)

DCM0, DCM1, DCM2 **0x13450014, 0x13450034, 0x13450054**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EACKS	EACKM	ERDM		Reserved		BLAST	Reserved	SAI	DAI			Reserved				SP		DP		Reserved		TSZ		NRD	NWR	NAC		Reserved		STDE	TIE	LINK
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	EACKS	External DACK Output Level Select. 0: active high; 1: active low.	RW
30	EACKM	External DACK Output Mode Select. 0: output in read cycle; 1: output in write cycle.	RW
29:28	ERDM	External DREQ Detection Mode Select. 00: Low level detection 01: Falling edge detection 10: High level detection 11: Rising edge detection	RW
27:26	Reserved	Write has no effect, read as zero.	R
25	BLAST	BCH/NAND last. 0: non-last data block for BCH/NAND; 1: last data block for BCH/NAND. (Only channel 0 support BCH transfer; all channel support Nand transfer,	RW

		when it is used for nand, it means the last data block transfer for one nand dma request detection)	
24	Reserved	Write has no effect, read as zero.	R
23	SAI	Source Address Increment. 0: no increment; 1: increment.	RW
22	DAI	Target Address Increment. 0: no increment; 1: increment.	RW
19:16	Reserved	Write has no effect, read as zero.	R
15:14	SP	Source port width. 00: 32-bit; 01: 8-bit; 10: 16-bit; 11: reserved.	RW
13:12	DP	Target port width. 00: 32-bit; 01: 8-bit; 10: 16-bit; 11: reserved. (**NOTE:** for bch transfer encoding, DP only can be 32-bit or 8-bit; for bch decoding, DP only can be 32-bit)	RW
11	Reserved	Write has no effect, read as zero.	R
10:8	TSZ	Transfer Data Size of a data unit. 000: 32-bit; 001: 8-bit; 010: 16-bit; 011: 16-byte; 100: 32-byte; 101: 64-byte; others: reserved.	RW
7	NRD	Direct read nand. 0, non-direct read nand; 1, enable direct read nand.	RW
6	NWR	Direct write nand. 0: non-direct write nand; 1: enable direct write nand.	RW
5	NAC	Nand AL/CL from Data. 0: AL/CL from data is disabled; 1: enable AL/CL from data. (If AL/CL from data is disabled, AL/CL is from address[23:22] written to nand; When AL/CL from data is enabled, bit 31 of the data indicates AL, bit 30 of the data indicates CL; When AL/CL from data is enabled, be sure to set SP to 32-bit, set DP according to SMCR.BW in memory controller.)	RW
4:3	Reserved	Write has no effect, read as zero.	R
2	STDE	Stride Disable/Enable. 0: address stride disable; 1: address stride enable.	RW
1	TIE	Transfer Interrupt Enable. (TIE) 0: disable interrupt; 1: enable interrupt when TT is set to 1.	RW
0	LINK	Descriptor Link Enable. 0: disable; 1: enable. (Ignored in No-Descriptor Transfer)	RW

9.2.7 DMA Descriptor Address (DDAn, n = 0 ~ 2)

This register is ignored in No-Descriptor Transfer.

DDA0, DDA1, DDA2 **0x13450018, 0x13450038, 0x13450058**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DBA																				DOA								Reserved			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:12	DBA	Descriptor Base Address.	RW
11:4	DOA	Descriptor Offset Address. When 4-descriptor is used, DOA is used for the offset address of DDA for next descriptor fetch.	RW
3:0	Reserved	Write has no effect, read as zero.	R

NOTES:

1 When 8-descriptor is used, next descriptor fetch address is from 8th word of last descriptor, that is the 0th~27th bit of the 8th word of last descriptor is mapped to DDA[31:4] for next descriptor fetch.

9.2.8 DMA Stride Address (DSDn, n = 0 ~ 2)

This register is ignored in No-Descriptor Transfer.

When address stride transfer is enabled in Descriptor mode, after a sub-block defined in DTCRn is finished transferring, the source or target stride address will be added up to the corresponding source or target address and the transfer will keep going until the transfer ends which means TC in DTCRn reach 0.

DSD0, DSD1, DSD2 **0x1345001C, 0x1345003C, 0x1345005C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TSD																SSD															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:16	TSD	Target Stride Address.	RW
15:0	SSD	Source Stride Address.	RW

9.2.9 DMA Nand Timer (DNTn, n = 0 ~ 2)

This register is used for nand low pulse detect and for AL and CL from data.

DNT0, DNT1, DNT2 **0x134500C0, 0x134500C4, 0x134500C8**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									DTCT							DNTE	Reserved											DNT			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

Bits	Name	Description	RW
31:23	Reserved	Write has no effect, read as zero.	R
22:16	DTCT	Tail Counter. When Nand AL/CL from data is enabled, the counter indicates the actual word written to nand in the last transfer. It is used for transfer count when nand AL/CL from data is enabled. During the last transfer (DTCR == 1), if DCMR.TSZ is set for 16-byte, 32-byte or 64-byte, when DTCT is not equal to 0, the value in DTCT indicates the actual word number written to nand in the last transfer.	RW
15	DNTE	Nand Detect Timer enable. 0: Nand detect timer disable 1: Nand detect timer enable	RW
14:6	Reserved	Write has no effect, read as zero.	R
5:0	DNT	Nand Detect Timer. When Nand detect timer is enabled, the timer starts running down, when the timer is down to zero, it generates a request to DMA for data transfer.	RW

9.2.10 DMA Control

DMAC1 controls channel 0~2.

DMAC1 **0x13450300**

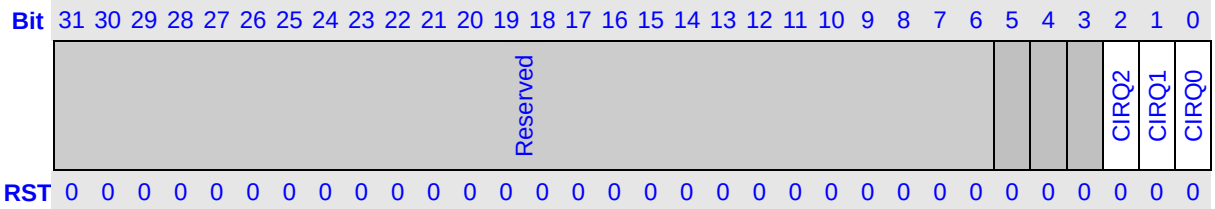
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																						PM	Reserved					HLT	AR	Reserved	DMAE
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:10	Reserved	Write has no effect, read as zero.	R
9:8	PM	Channel priority mode. 00: CH0, CH1 > CH2 01: CH1, CH2 > CH0 10: CH2 > CH0, CH1 11: CH0, CH1, CH2 For example, when PM == 2'b00, it means set1 includes ch0 and ch1 and set2 includes ch2, set 1 has the higher priority than set 2, within one set, channel priority is round robin, that is: ch0→ch1→ch2.	RW
7:4	Reserve	Write has no effect, read as zero.	R
3	HLT	Global halt status, halt occurs in any channel, the bit should set to 1. 0: no halt 1: halt occurred	RW
2	AR	Global address error status, address error occurs in any channel, the bit should be set to 1. 0: no address error 1: address error occurred	RW
1	Reserved	Write has no effect, read as zero.	R
0	DMAE	Global DMA transfer enable. 0: disable DMA channel transfer 1: enable DMA channel transfer	RW

9.2.11 DMA Interrupt Pending (DIRQP)

DMAC supports total 3 pending interrupt which are in DIRQP.

DIRQP **0x13450304**



Bits	Name	Description	RW
31:3	Reserved	Write has no effect, read as zero.	R
2:0	CIRQn	CIRQn (n=0~2) denotes pending status for corresponding channel. 0: no abnormal situation or normal DMA transfer is in progress 1: abnormal situation occurred or normal DMA transfer done	RW

9.2.12 DMA Doorbell (DDR)

DDR supports channel 0~2.

DDR **0x13450308**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																													DB2	DB1	DB0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:3	Reserved	Write has no effect, read as zero.	R
2:0	DBn	DMA Doorbell for each channel, n=0~2, for example DB0 is for DMA channel 0. Software set it to 1 and hardware clears it to 0. 0: disable DMA controller to fetch the first descriptor or DMA controller clears it to 0 as soon as it starts to fetch the descriptor 1: Write 1 to DDS will set the corresponding DBn bit to 1 and enable DMA controller to fetch the first descriptor For example, write 0x00000001 to DDS, DB0 bit is set to 1 and enable DMA channel 0 to fetch the first descriptor. Write 0 to DDS, no meaning.	R

9.2.13 DMA Doorbell Set (DDRS)

DDRS supports channel 0~2.

DDRS **0x1345030c**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																													DBS2	DBS1	DBS0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:3	Reserved	Write has no effect, read as zero.	R
2:0	DBSn	DMA Doorbell Set for each channel. 0: ignore 1: Set the corresponding DBn bit to 1	W

9.2.14 DMA Clock Enable (DCKE)

DCKE supports channel 0~2.

DCKE **0x13450310**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																										DCKE5	DCKE4	DCKE3	DCKE2	DCKE1	DCKE0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:3	Reserved	Write has no effect, read as zero.	R
2:0	DCKEn	DMA Clock Enable for each channel. 0: ignore 1: Set the corresponding DCKEn bit to 1	W

9.3 DMA manipulation

9.3.1 Descriptor Transfer

9.3.1.1 Normal Transfer

To do proper Descriptor DMA transfer, do as following steps:

1 First of all, open channel clock by setting DCKEn register for corresponding channel.

2 Check whether the status of DMA controller are available, that is, for global control (DMAC), ensure that DMAC.AR=0 and DMAC.HLT=0; while for expected channels, ensure that DCSn.AR=0, DCSn.HLT=0, DCSn.TT=0 and DTCn=0.

3 Select 4 word or 8 word descriptor by DCSn.DES8.

4 For Descriptor transfer, guarantee DCSn.NDES=0.

5 Initiate channel request register DRSRn.

6 Build descriptor in memory. Write the first descriptor address in DDAn and the address must be 16Bytes aligned in 4word descriptor and 32Bytes aligned in 8word descriptor. The descriptor address includes two parts: Base and Offset address. If the descriptor is linked, the 32-bit address of next descriptor is composed of 20-bit Base address in DDAn and 8-bit Offset address in DES3.DOA and the four LSB is 0x0. See Table 9-3 for the detailed 4-word descriptor structure.

   NOTE: if stride address transfer is enabled, the address must be 32Bytes aligned because DES4 needs to read out.

7 Set 1 to the corresponding bit in DDR to initiate descriptor fetch.

8 Set DMAC.DMAE=1 and expected DCSn.CTE=1 to launch DAM transfer.

9 Hardware clears the corresponding bit in DDR as soon as it starts to fetch the descriptor.

10 Waits for dma request from peripherals to start dma transfer.

11 After DMAC completes the current descriptor dma transfer, if DES0.Link=0, it sets DCSn.TT to 1. If the interrupt enabled, it will generates the corresponding interrupts.

12 If DES0.LINK=1, after DMAC completes the current descriptor dma transfer and return to fetch the next descriptor and continues dma transfer until completes the descriptor dma transfer which DES0.LINK=0.

13 When transfer end, clr DCSn.CTE to 0 to close the channel, and then clear DCSn.TT bits.

Table 9-3 Descriptor Structure

Word	Bit	Name	Function
1st (DES0)	31	EACKS	External DMA DACKn output polarity select
	30	EACKM	External DMA DACKn output Mode select
	29-28	ERDM	External DMA request detection Mode
	27	EOPM	External DMA End of process mode
	26	Reserved	
	25	BLAST	BCH Last (Only for BCH and Nand transfer)
	24	Reserved	
	23	SAI	Source Address Increment

	22	DAI	Target Address Increment
	21-20	Reserved	
	19-16	RDIL	Request Detection Interval Length
	15-14	SP	Source port width
	13-12	DP	Target port width
	11	Reserved	
	10-8	TSZ	Transfer Data Size
	7	NRD	Direct read nand
	6	NWR	Direct write nand
	5	NAC	Nand AL/CL from data
	4:3	Reserved	
	2	STDE	Stride transfer enable
	1	TIE	Transfer Interrupt Enable
	0	LINK	Descriptor Link Enable
2nd (DES1)	31-0	DSA	Source Address
3rd (DES2)	31-0	DTA	Target Address
4th (DES3)	31-24	DOA	Descriptor Offset address
	23-0	DTC	Transfer Counter
5th (DES4)	31-16	TSD	Target Stride Address
	15-0	SSD	Source Stride Address
6th(DES5)	31-6	Reserved	
	5-0	DRT	DMA Request Type
7th(DES6)	31-23	Reserved	
	22-16	DTCT	Nand tail counter
	15	DNTE	Nand detect timer enable
	14-6	Reserved	
	5-0	DNT	Nand detect timer
8th(DES7)	31-4	DDA	Next descriptor address
	3-0	Reserved	

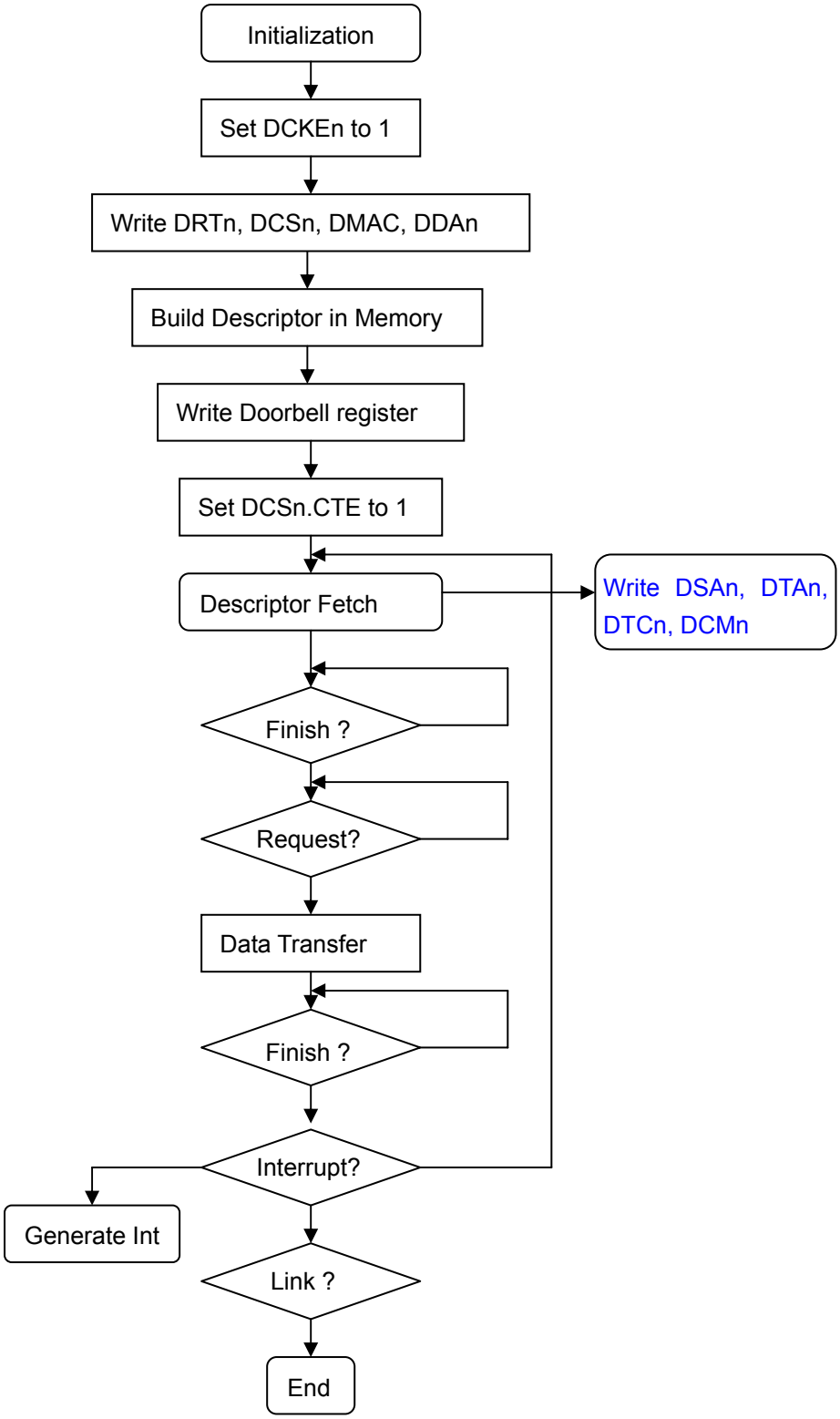


Figure 9-1 Descriptor Transfer Flow

9.3.1.2 Stride Address Transfer

During transfer, source or target address can be not continuous and the source and target stride offset address are showed in DSDn registers.

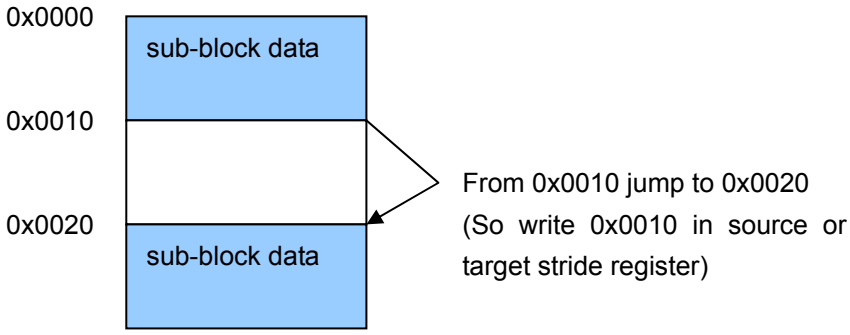


Figure 9-2 Example for Stride Address Transfer

9.3.1.3 BCH DMA Transfer

Channel 0 supports BCH DMA transfer.

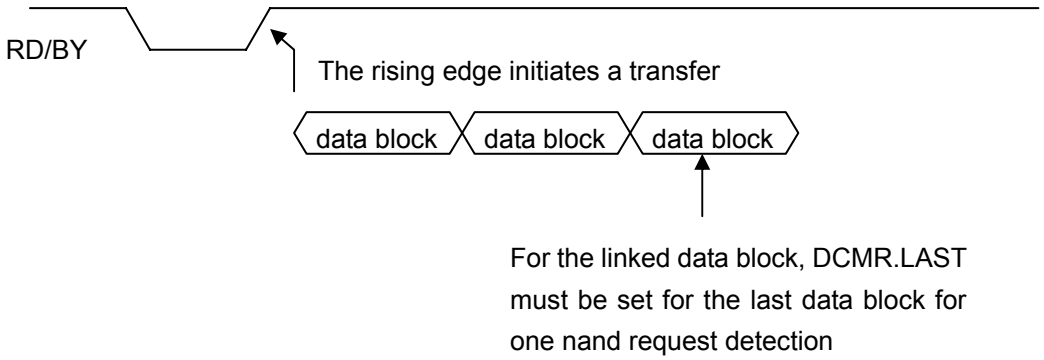
During BCH encoding, DMA read data from memory pointed by DSAR0 and write to BCH data register BHDR, after BCH encoding finishes, DMA write BHINT and BCH parity data BHPAR0~9 respectively to memory pointed by DTAR0, and then DMA clear BHINT and set BCH reset to BCH automatically.

During BCH decoding, DMA read data from memory pointed by DSAR0 and write to BCH data register BHDR, after BCH decoding finishes, if there is error in the data block, DMA will write BHINT, BHERR0~11 to memory pointed by DTAR0 or if there is no error in the data block, DMA will only write BHINT to memory, and then DMA clear BHINT and set BCH reset to BCH. If multiple data block are linked to wait for BCH decoding, data transfer and decoding can be executed in pipeline, that is when the first data block is being decoding, and second data can be transfer to BCH for syndrome generation.

Here one data block means, for encoding, the entire data bytes need encoding, for decoding, the entire data bytes and parity bytes need decoding. DCM.BLAST must be used in descriptor BCH transfer. When one data block is in a continuous memory space, BLAST must be set to 1 for this data block; when one data block is linked in multiple data space, BLAST must be set to 1 for the last data space.

9.3.1.4 Nand Transfer

Two ways are for nand RB detect.
One way is to detect RD/BY rising edge as the following waveform.



The other way is using DNT register. When the rising edge is missed by DMA, DNT timer also can be used for nand RB detect. The timer is used to detect the high level duration of RD/BY signal, when the high level keep high longer than DNT counting periods, then nand transfer request generates.

9.3.2 No-Descriptor Transfer

To do proper DMA transfer, do as following steps:

1. First of all, check whether the status of DMA controller are available, that is, for global control (DMAC), ensure that DMAC.AR=0 and DMAC.HLT=0; while for expected channels, ensure that DCSn.AR=0, DCSn.HLT=0 and DCSn.TT=0 and DTCn=0.
2. For each channel n, initialize DSAn, DTAn, DTCn, DRTn, DCSn, DCMn properly.
3. Set DMAC.DMAE=1 and expected DCSn.CTE=1 and DCSn.NDES=1 to launch DAM transfer.

For a channel with auto-request (DRTn.RT=0x8), the transfer begins automatically when the DCSn.CTE bit and DMAC.DMAE bit are set to 1. While for a channel with other request types, the transfer does not start until a transfer request is issued and detected.

For any channel n, The DTCn value is decremented by 1 for each successful transaction of a data unit. When the specified number of transfer data unit has been completed (DTCn = 0), the transfer ends normally. Meanwhile corresponding bit of DIRQP is set to 1. If DCMn.TIE bit is set to 1, an interrupt request is sent to the CPU. However, during the transfer, if a DMA address error occurs, the transfer is suspended, both DCSn.AR and DMAC.AR are set to 1 as well as corresponding bit of DIRQP. Then an interrupt request is sent to the CPU despite of DCMn.TIE.

Sometimes, for example, an UART parity error occurs for a channel that is transferring data between such UART and another terminal. In the case, both DCSn.HLT and DMAC.HLT are set to 1 and the transfer is suspended. Software should identify halt status by checking such two bits and re-configure

DMA to let DMA rerun properly later.

For non-descriptor BCH transfer, there is no pipeline execution for BCH decoding. DCM.BLAST doesn't need to be set in non-descriptor BCH transfer.

9.4 DMA Requests

DMA transfer requests are normally generated from either the data transfer source or target, but also they can be issued by on-chip peripherals that are neither the source nor the target. There are two DMA transfer request types: auto-request, and on-chip peripheral request. For any channel n, its transfer request type is determined through DRTn.

9.4.1 Auto Request

When there is no explicit transfer request signal available, for example, memory-to-memory transfer or memory to some on-chip peripherals like GPIO, the auto-request mode allows the DMA to automatically generate a transfer request signal internally. Therefore, when DMA initialization done, once the DMAC.DMAE and DCSn.CTE are set to 1, the transfer begins immediately in channel n which DRTn=0x8.

9.4.2 On-Chip Peripheral Request

In the mode, transfer request signals come from on-chip peripherals. All request types except 0x8 (value of DRT) belong to the mode.

NOTES:

1 he transfer byte number for one request detection according to DCMn.RDIL must be equal or less than the byte number according to receive or transmit trigger value of source or target devices.

9.5 Channel Priorities

There are two dma cores, each one supports 6 channels dma transfer. The two cores have the same priority.

In each core, there are two sets: set 1 has the higher priority than set 2, within each set priority is round robin.

Table 9-4 Relationship among DMA transfer connection, request mode & transfer mode

Transfer Connection	Request Mode	Transfer Mode	Data Size (bits)	Channel
External memory or memory-mapped external device and on-chip peripheral module	Auto on-chip	Single	8/16/32 16-byte/32-byte	0~5

9.6 Examples

9.6.1 Memory-to-memory auto request No-Descriptor Transfer

Suppose you want to do memory move between two different memory regions through channel 3, for example, moving 1KB data from address 0x20001000 to 0x20011000, do as following steps:

1 Check if (DMAC.AR==0 && DMAC.HLT==0 && DCS3.AR==0 && DCS3.HLT==0 && DCS3.CT==0 && DCS3.NDES=1 && DTC3==0).
2 If above condition is true, set value 0 to DCS3.CTE to disable the channel 3 temporarily.
3 Set source address 0x20001000 to DSA3 and target address 0x20011000 to DTA3.
4 Suppose the data unit is word, set transfer count number 256 (1024/4) to DTC3.
5 Set auto-request (0x8) to DRT3.
6 Up to now, only the most important channel control register DCM3 is left, set it carefully.
7 Set value 1 to SAI and DAI[*1].
8 Ignore RDIL because in the case there is no explicit request signal can be detected.
9 Set word size (0) to SP and DP[*2].
10 Set value 1 to TIE to let CPU do some post process after the transfer done.
11 Set value 1 to DCS3.CTE and DMAC.DMAE to launch the transfer in channels 3.
12 When the transfer terminates normally (DTC3==0 && DCS3.TT==1), DIRQP.CIRQ3 will automatically be set value 1 and an interrupt request will be sent to CPU.
13 When CPU grants the interrupt request, in the corresponding IRQ handler, software must clear the DCS3.CT to value 0, and the behavior will automatically clear DIRQP.CIRQ3.

NOTES:

1 [*1]: Either source or target is a FIFO, must not enable corresponding address increment.
2 [*2]: When either source or target need be accessed through EMC (external memory controller), the real port with of the device is encapsulated by EMC, so you can set any favorite port with for it despite of the real one.

10 DMA Controller

DMA controller (DMAC) is dedicated to transfer data between on-chip peripherals (MSC, AIC, UART, etc.), external memories, and memory-mapped external devices.

10.1 Features

- Support up to 10 independent DMA channels
- Two independent DMA core, each supports 5 channels
- Descriptor or No-Descriptor Transfer
- Transfer data units: byte, 2-byte (half word), 4-byte (word), 16-byte, 32-byte or 64-byte
- Transfer number of data unit: 1 ~ 2^{24}
- Independent source and target port width: 8-bit, 16-bit, 32-bit
- Two channel priority modes: fixed, round robin

10.2 Register Descriptions

Table 10-1 DMAC Registers

Name	Description	RW	Reset Value	Address	Access Size (bit)
DSA0	DMA Source Address 0	RW	0x0	0x13420000	32
DTA0	DMA Target Address 0	RW	0x0	0x13420004	32
DTC0	DMA Transfer Count 0	RW	0x0	0x13420008	32
DRT0	DMA Request Source 0	RW	0x0	0x1342000C	32
DCS0	DMA Channel Control/Status 0	RW	0x0	0x13420010	32
DCM0	DMA Command 0	RW	0x0	0x13420014	32
DDA0	DMA Descriptor Address 0	RW	0x0	0x13420018	32
DSD0	DMA Stride Address 0	RW	0x0	0x1342001C	32
DSA1	DMA Source Address 1	RW	0x0	0x13420020	32
DTA1	DMA Target Address 1	RW	0x0	0x13420024	32
DTC1	DMA Transfer Count 1	RW	0x0	0x13420028	32
DRT1	DMA Request Source 1	RW	0x0	0x1342002C	32
DCS1	DMA Channel Control/Status 1	RW	0x0	0x13420030	32
DCM1	DMA Command 1	RW	0x0	0x13420034	32
DDA1	DMA Descriptor Address 1	RW	0x0	0x13420038	32
DSD1	DMA Stride Address 1	RW	0x0	0x1342003C	32
DSA2	DMA Source Address 2	RW	0x0	0x13420040	32
DTA2	DMA Target Address 2	RW	0x0	0x13420044	32
DTC2	DMA Transfer Count 2	RW	0x0	0x13420048	32
DRT2	DMA Request Source 2	RW	0x0	0x1342004C	32
DCS2	DMA Channel Control/Status 2	RW	0x0	0x13420050	32
DCM2	DMA Command 2	RW	0x0	0x13420054	32
DDA2	DMA Descriptor Address 2	RW	0x0	0x13420058	32
DSD2	DMA Stride Address 2	RW	0x0	0x1342005C	32
DSA3	DMA Source Address 3	RW	0x0	0x13420060	32
DTA3	DMA Target Address 3	RW	0x0	0x13420064	32
DTC3	DMA Transfer Count 3	RW	0x0	0x13420068	32
DRT3	DMA Request Source 3	RW	0x0	0x1342006C	32
DCS3	DMA Channel Control/Status 3	RW	0x0	0x13420070	32
DCM3	DMA Command 3	RW	0x0	0x13420074	32
DDA3	DMA Descriptor Address 3	RW	0x0	0x13420078	32
DSD3	DMA Stride Address 3	RW	0x0	0x1342007C	32
DSA4	DMA Source Address 4	RW	0x0	0x13420080	32
DTA4	DMA Target Address 4	RW	0x0	0x13420084	32
DTC4	DMA Transfer Count 4	RW	0x0	0x13420088	32
DRT4	DMA Request Source 4	RW	0x0	0x1342008C	32

DCS4	DMA Channel Control/Status 4	RW	0x0	0x13420090	32
DCM4	DMA Command 4	RW	0x0	0x13420094	32
DDA4	DMA Descriptor Address 4	RW	0x0	0x13420098	32
DSD4	DMA Stride Address 4	RW	0x0	0x1342009C	32
DSA6	DMA Source Address 6	RW	0x0	0x13420100	32
DDA6	DMA Target Address 6	RW	0x0	0x13420104	32
DTC6	DMA Transfer Count 6	RW	0x0	0x13420108	32
DRT6	DMA Request Source 6	RW	0x0	0x1342010C	32
DCS6	DMA Channel Control/Status 6	R/W	0x0	0x13420110	32
DCM6	DMA Command 6	RW	0x0	0x13420114	32
DDA6	DMA Descriptor Address 6	RW	0x0	0x13420118	32
DSD6	DMA Stride Address 6	RW	0x0	0x1342011C	32
DSA7	DMA Source Address 7	RW	0x0	0x13420120	32
DDA7	DMA Target Address 7	RW	0x0	0x13420124	32
DTC7	DMA Transfer Count 7	RW	0x0	0x13420128	32
DRT7	DMA Request Source 7	RW	0x0	0x1342012C	32
DCS7	DMA Channel Control/Status 7	R/W	0x0	0x13420130	32
DCM7	DMA Command 7	RW	0x0	0x13420134	32
DDA7	DMA Descriptor Address 7	RW	0x0	0x13420138	32
DSD7	DMA Stride Address 7	RW	0x0	0x1342013C	32
DSA8	DMA Source Address 8	RW	0x0	0x13420140	32
DDA8	DMA Target Address 8	RW	0x0	0x13420144	32
DTC8	DMA Transfer Count 8	RW	0x0	0x13420148	32
DRT8	DMA Request Source 8	RW	0x0	0x1342014C	32
DCS8	DMA Channel Control/Status 8	R/W	0x0	0x13420150	32
DCM8	DMA Command 8	RW	0x0	0x13420154	32
DDA8	DMA Descriptor Address 8	RW	0x0	0x13420158	32
DSD8	DMA Stride Address 8	RW	0x0	0x1342015C	32
DSA9	DMA Source Address 9	RW	0x0	0x13420160	32
DDA9	DMA Target Address 9	RW	0x0	0x13420164	32
DTC9	DMA Transfer Count 9	RW	0x0	0x13420168	32
DRT9	DMA Request Source 9	RW	0x0	0x1342016C	32
DCS9	DMA Channel Control/Status 9	R/W	0x0	0x13420170	32

DCM9	DMA Command 9	RW	0x0	0x13420174	32
DDA9	DMA Descriptor Address 9	RW	0x0	0x13420178	32
DSD9	DMA Stride Address 9	RW	0x0	0x1342017C	32
DSA10	DMA Source Address 10	RW	0x0	0x13420180	32
DDA10	DMA Target Address 10	RW	0x0	0x13420184	32
DTC10	DMA Transfer Count 10	RW	0x0	0x13420188	32
DRT10	DMA Request Source 10	RW	0x0	0x1342018C	32
DCS10	DMA Channel Control/Status 10	R/W	0x0	0x13420190	32
DCM10	DMA Command 10	RW	0x0	0x13420194	32
DDA10	DMA Descriptor Address 10	RW	0x0	0x13420198	32
DSD10	DMA Stride Address 10	RW	0x0	0x1342019C	32
DMAC1	DMA Control 1 Register	R/W	0x0	0x13420300	32
DIRQP1	DMA Interrupt Pending 1	R	0x0	0x13420304	32
DDR1	DMA Doorbell 1 Register	RW	0x0	0x13420308	32
DDRS1	DMA Doorbell Set 1 Register	W	0x0	0x1342030C	32
DCKE1	DMA Clock Enable 1 Register	W	0x0	0x13420310	32
DMAC2	DMA Control 2 Register	R/W	0x0	0x13420400	32
DIRQP2	DMA Interrupt Pending 2	R	0x0	0x13420404	32
DDR2	DMA Doorbell 2 Register	RW	0x0	0x13420408	32
DDRS2	DMA Doorbell Set Register	W	0x0	0x1342040C	32
DCKE2	DMA Clock Enable 2 Register	W	0x0	0x13420410	32

10.2.1 DMA Source Address (DSAn, n = 0 ~ 11)

DSA0, DSA1, DSA2, 0x13420000, 0x13420020, 0x13420040,
DSA3, DSA3, 0x13420060, 0x13420080,
DSA6, DSA7, DSA8, 0x13420100, 0x13420120, 0x13420140,
DSA9, DSA10, 0x13420160, 0x13420180

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SA																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	SA	Source physical address.	RW

10.2.2 DMA Target Address (DTAn, n = 0 ~ 11)

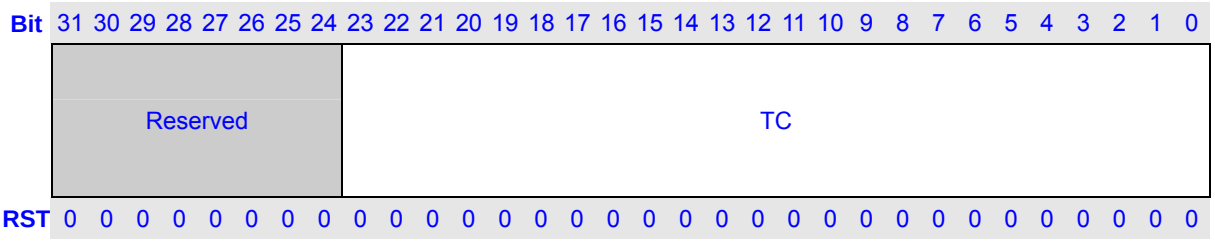
DTA0, DTA1, DTA2, 0x13420004, 0x13420024, 0x13420044,
DTA3, DTA4, 0x13420064, 0x13420084,
DTA6, DTA7, DTA8, 0x13420104, 0x13420124, 0x13420144,
DTA9, DTA10 0x13420164, 0x13420184

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TA																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	TA	Target physical address.	RW

10.2.3 DMA Transfer Count (DTCn, n = 0 ~ 11)

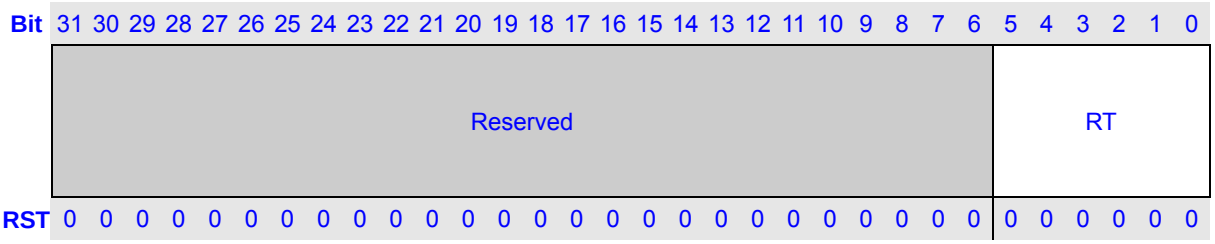
DTC0, DTC1, DTC2, 0x13420008, 0x13420028, 0x13420048,
DTC3, DTC4, 0x13420068, 0x13420088,
DTC6, DTC7, DTC8, 0x13420108, 0x13420128, 0x13420148,
DTC9, DTC10 0x13420168, 0x13420188



Bits	Name	Description	RW
31:24	Reserved	Write has no effect, read as zero.	R
23:0	TC	When Stride address transfer is disabled: TC hold the number of data unit to transfer and it counts down to 0 at the end; When Stride address transfer is enabled: TC composes of two parts: The lower 16 bits: the number of data unit for sub-block transfer The higher 8 bits: the number of sub-block And both the two parts count down to 0 at the end.	RW

10.2.4 DMA Request Types (DRTn, n = 0 ~ 11)

DRT0, DRT1, DRT2, 0x1342000c, 0x1342002c, 0x1342004c,
DRT3, DRT4 0x1342006c, 0x1342008c,
DRT6, DRT7, DRT8, 0x1342010c, 0x1342012c, 0x1342014c,
DRT9, DRT10 0x1342016c, 0x1342018c



Bits	Name	Description	RW
31:6	Reserved	Write has no effect, read as zero.	R
5:0	RT	Transfer request type.	RW

Table 10-2 Transfer Request Types

RT5-0	Description
000000	Reserved
000001	Reserved
000010	Reserved
000011	Reserved
000100	Reserved
000101	Reserved
000110	Reserved
000111	Reserved
001000	Auto-request (ignore RDIL3-0, external address → external address)
001001	TSSI receive-fifo-full transfer request (TS fifo → external address)
001010	Reserved
001011	Reserved
001100	External request with DREQn (external address ←→ external device with DACKn)
001101	Reserved
001110	UART3 transmit-fifo-empty transfer request (external address → UTHR)
001111	UART3 receive-fifo-full transfer request (URBR → external address)
010000	UART2 transmit-fifo-empty transfer request (external address → UTHR)
010001	UART2 receive-fifo-full transfer request (URBR → external address)
010010	UART1 transmit-fifo-empty transfer request (external address → UTHR)
010011	UART1 receive-fifo-full transfer request (URBR → external address)
010100	UART0 transmit-fifo-empty transfer request (external address → UTHR)
010101	UART0 receive-fifo-full transfer request (URBR → external address)
010110	SSI transmit-fifo-empty transfer request
010111	SSI receive-fifo-full transfer request
011000	AIC transmit-fifo-empty transfer request
011001	AIC receive-fifo-full transfer request
011010	MSC transmit-fifo-empty transfer request
011011	MSC receive-fifo-full transfer request
011100	TCU channel n (overflow interrupt, external address→external address space)
011101	SADC transfer request (SADC → external address)
011110	MSC1 transmit-fifo-empty transfer request
011111	MSC1 receive-fifo-full transfer request
100000	SSI1 transmit-fifo-empty transfer request
100001	SSI1 receive-fifo-full transfer request
100010	PM transmit-fifo-empty transfer request
100011	PM receive-fifo-full transfer request
100100	MSC2 transmit-fifo-empty transfer request
100101	MSC2 receive-fifo-full transfer request
Other	Reserved

NOTES:

1 Only auto request can be concurrently selected in all channels with different source and target address.

2 For on-chip device DMA request except TCU, the corresponding source or target address that map to on-chip device must be set as fixed.

10.2.5 DMA Channel Control/Status (DCSn, n = 0 ~ 11)

DCS0, DCS1, DCS2, **0x13420010, 0x13420030, 0x13420050,**
DCS3, DCS4, **0x13420070, 0x13420090,**
DCS6, DCS7, DCS8, **0x13420110, 0x13420130, 0x13420150,**
DCS9, DCS10 **0x13420170, 0x13420190**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NDES	DES8	Reserved						CDOA								Reserved											AR	TT	HLT	Reserved	CTE
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	NDES	Descriptor or No-Descriptor Transfer Select. 0: Descriptor Transfer; 1: No-descriptor Transfer.	RW
30	DES8	Descriptor 8 Word. 0: 4-word descriptor; 1: 8-word descriptor.	RW
29:24	Reserved	Write has no effect, read as zero.	R
23:16	CDOA	Copy of offset address of last completed descriptor from that in DMA command register. Software could know which descriptor is just completed combining with count terminate interrupt resulted by DCSn.CT. (Ignored in No-Descriptor Transfer)	RW
15:5	Reserved	Write has no effect, read as zero.	R
4	AR	Address Error. 0: no address error; 1: address error.	RW
3	TT	Transfer Terminate. 0: No-Link Descriptor or No-Descriptor DMA transfer does not end 1: No-Link Descriptor or No-Descriptor DMA transfer end	RW
2	HLT	DMA halt. 0: DMA transfer is in progress; 1: DMA halt.	RW
1	Reserved	Write has no effect, read as zero.	R
0	CTE	Channel transfer enable. 0: disable; 1: enable.	RW

10.2.6 DMA Channel Command (DCMn, n = 0 ~ 11)

DCM0, DCM1, DCM2, 0x13420014, 0x13420034, 0x13420054,
DCM3, DCM4, 0x13420074, 0x13420094,
DCM6, DCM7, DCM8, 0x13420114, 0x13420134, 0x13420154,
DCM9. DCM10 0x13420174, 0x13420194

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EACKS	EACKM	ERDM		Reserved				SAI	DAI	Reserved		RDIL				SP		DP		Reserved		TSZ		Reserved					STDE	TIE	LINK
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	EACKS	External DACK Output Level Select. 0: active high; 1: active low.	RW
30	EACKM	External DACK Output Mode Select. 0: output in read cycle; 1: output in write cycle.	RW
29:28	ERDM	External DREQ Detection Mode Select. 00: Low level detection 01: Falling edge detection 10: High level detection 11: Rising edge detection	RW
27:24	Reserved	Write has no effect, read as zero.	R
23	SAI	Source Address Increment. 0: no increment; 1: increment.	RW
22	DAI	Target Address Increment. 0: no increment; 1: increment.	RW
19:16	RDIL	Request Detection Interval Length. Set the number of transfer unit between two requests detection in single mode. Please refer to following Table 10-3.	RW
15:14	SP	Source port width. 00: 32-bit; 01: 8-bit; 10: 16-bit; 11: reserved.	RW
13:12	DP	Target port width. 00: 32-bit; 01: 8-bit; 10: 16-bit; 11: reserved.	RW
11	Reserved	Write has no effect, read as zero.	R
10:8	TSZ	Transfer Data Size of a data unit. 000: 32-bit; 001: 8-bit; 010: 16-bit; 011: 16-byte; 100: 32-byte; 101: 64-byte; others: reserved.	RW
7:3	Reserved	Write has no effect, read as zero.	R
2	STDE	Stride Disable/Enable. 0: address stride disable; 1: address stride enable.	RW
1	TIE	Transfer Interrupt Enable. (TIE)	RW

		0: disable interrupt; 1: enable interrupt when TT is set to 1.	
0	LINK	Descriptor Link Enable. 0: disable; 1: enable. (Ignored in No-Descriptor Transfer)	RW

Table 10-3 Detection Interval Length

RDIL	Description
0	Interval length is 0
1	Interval length is 2 transfer unit
2	Interval length is 4 transfer unit
3	Interval length is 8 transfer unit
4	Interval length is 12 transfer unit
5	Interval length is 16 transfer unit
6	Interval length is 20 transfer unit
7	Interval length is 24 transfer unit
8	Interval length is 28 transfer unit
9	Interval length is 32 transfer unit
10	Interval length is 48 transfer unit
11	Interval length is 60 transfer unit
12	Interval length is 64 transfer unit
13	Interval length is 124 transfer unit
14	Interval length is 128 transfer unit
15	Interval length is 200 transfer unit

10.2.7 DMA Descriptor Address (DDAn, n = 0 ~ 11)

This register is ignored in No-Descriptor Transfer.

DDA0, DDA1, DDA2, DDA3, DDA4, DDA6, DDA7, DDA8, DDA9, DDA10 — **0x13420018, 0x13420038, 0x13420058, 0x13420078, 0x13420098, 0x13420118, 0x13420138, 0x13420158, 0x13420178, 0x13420198**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DBA																				DOA								Reserved			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:12	DBA	Descriptor Base Address.	RW
11:4	DOA	Descriptor Offset Address.	RW

3:0	Reserved	Write has no effect, read as zero.	R

10.2.8 DMA Stride Address (DSDn, n = 0 ~ 11)

This register is ignored in No-Descriptor Transfer.

When address stride transfer is enabled in Descriptor mode, after a sub-block defined in DTCRn is finished transferring, the source or target stride address will be added up to the corresponding source or target address and the transfer will keep going until the transfer ends which means TC in DTCRn reach 0.

DSD0, DSD1, DSD2, DSD3, DSD4, DSD6, DSD7, DSD8, DSD9, DSD10 — **0x1342001C, 0x1342003C, 0x1342005C, 0x1342007C, 0x1342009C, 0x1342011C, 0x1342013C, 0x1342015C, 0x1342017C, 0x1342019C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TSD																SSD															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:16	TSD	Target Stride Address.	RW
15:0	SSD	Source Stride Address.	RW

10.2.9 DMA Control

DMAC1 controls channel 0~5 and DMAC2 controls channel 6~11.

DMAC1 — **0x13420300**

DMAC2 — **0x13420400**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FMSC	FSSI	FTSSI	FUART	FAIC	Reserved																	PM	Reserved					HLT	AR	Reserved	DMAE
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	FMSC	MSC Fast DMA mode. 0: normal DMA transfer; 1: fast DMA transfer.	RW
30	FSSI	SSI Fast DMA mode. 0: normal DMA transfer; 1: fast DMA transfer.	RW

29	FTSSI	TSSI Fast DMA mode. 0: normal DMA transfer; 1: fast DMA transfer.	RW
28	FUART	UART Fast DMA mode. 0: normal DMA transfer; 1: fast DMA transfer.	RW
27	FAIC	AIC Fast DMA mode. 0: normal DMA transfer; 1: fast DMA transfer.	RW
26:10	Reserved	Write has no effect, read as zero.	R
9:8	PM	Channel priority mode. 00: CH0, CH1 > CH2, CH3, CH4 01: CH1, CH2 > CH0, CH3, CH4 10: CH2, CH3 > CH0, CH1, CH4 11: CH3, CH4 > CH0, CH1, CH2 For example, when PM == 2'b00, it means set1 includes ch0 and ch1 and set2 includes ch2~ch4, set 1 has the higher priority than set 2, within one set, channel priority is round robin, that is: ch0→ch1→ch2→ch0→ch1→ch3→ch0→ch1→ch4→ch0→ch1.	RW
7:4	Reserve	Write has no effect, read as zero.	R
3	HLT	Global halt status, halt occurs in any channel, the bit should set to 1. 0: no halt 1: halt occurred	RW
2	AR	Global address error status, address error occurs in any channel, the bit should be set to 1. 0: no address error 1: address error occurred	RW
1	Reserved	Write has no effect, read as zero.	R
0	DMAE	Global DMA transfer enable. 0: disable DMA channel transfer 1: enable DMA channel transfer	RW

NOTES:

1 FMSC/FSSI/FTSSI/FUART/FAIC bit either in DMAC1 or in DMAC2 is set, the corresponding dma transfer for MSC(MSC1, MSC2), SSI(SSI1), UART0~3, AIC is in fast dma mode.

10.2.10 DMA Interrupt Pending (DIRQP)

DMAC supports total 12 pending interrupt, 6 of them are in DIRQP and the other 6 are in DIRQP2.

DIRQP **0x13420304**

DIRQP2 **0x13420404**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																										CIRQ5	CIRQ4	CIRQ3	CIRQ2	CIRQ1	CIRQ0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:6	Reserved	Write has no effect, read as zero.	R
5:0	CIRQn	CIRQn (n=0~5) denotes pending status for corresponding channel. 0: no abnormal situation or normal DMA transfer is in progress 1: abnormal situation occurred or normal DMA transfer done	RW

10.2.11 DMA Doorbell (DDR)

DDR supports channel 0~5 and DDR2 supports channel 6~11.

DDR **0x13420308**

DDR2 **0x13420408**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																										DB5	DB4	DB3	DB2	DB1	DB0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:6	Reserved	Write has no effect, read as zero.	R
5:0	DBn	DMA Doorbell for each channel, n=0~5, for example DB0 is for DMA channel 0. Software set it to 1 and hardware clears it to 0. 0: disable DMA controller to fetch the first descriptor or DMA controller clears it to 0 as soon as it starts to fetch the descriptor 1: Write 1 to DDS will set the corresponding DBn bit to 1 and enable DMA controller to fetch the first descriptor For example, write 0x00000001 to DDS, DB0 bit is set to 1 and enable DMA channel 0 to fetch the first descriptor. Write 0 to DDS, no meaning.	R

10.2.12 DMA Doorbell Set (DDRS)

DDRS supports channel 0~5 and DDRS2 supports channel 6~11.

DDRS **0x1342030c**

DDRS2 **0x1342040c**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																										DBS5	DBS4	DBS3	DBS2	DBS1	DBS0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:6	Reserved	Write has no effect, read as zero.	R
5:0	DBSn	DMA Doorbell Set for each channel. 0: ignore 1: Set the corresponding DBn bit to 1	W

10.2.13 DMA Clock Enable (DCKE)

DCKE supports channel 0~5 and DCKE2 supports channel 6~11.

DCKE **0x13420310**

DCKE2 **0x13420410**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																										DCKE5	DCKE4	DCKE3	DCKE2	DCKE1	DCKE0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:6	Reserved	Write has no effect, read as zero.	R
5:0	DCKEn	DMA Clock Enable for each channel. 0: ignore 1: Set the corresponding DCKEn bit to 1	W

10.3 DMA manipulation

10.3.1 Descriptor Transfer

10.3.1.1 Normal Transfer

To do proper Descriptor DMA transfer, do as following steps:

1 First of all, open channel clock by setting DCKEn register for corresponding channel.
2 Check whether the status of DMA controller are available, that is, for global control (DMAC), ensure that DMAC.AR=0 and DMAC.HLT=0; while for expected channels, ensure that DCSn.AR=0, DCSn.HLT=0, DCSn.TT=0, DTCn=0 and DCSn.INV=0.
3 Select 4 word or 8 word descriptor by DCSn.DES8.
4 For Descriptor transfer, guarantee DCSn.NDES=0.
5 Initiate channel request register DRTn.
6 Build descriptor in memory. Write the first descriptor address in DDAn and the address must be 16Bytes aligned in 4word descriptor and 32Bytes aligned in 8word descriptor. The descriptor address includes two parts: Base and Offset address. If the descriptor is linked, the 32-bit address of next descriptor is composed of 20-bit Base address in DDAn and 8-bit Offset address in DES3.DOA and the four LSB is 0x0. See Table 10-4 for the detailed 4-word descriptor structure.
NOTE: if stride address transfer is enabled, the address must be 32Bytes aligned because DES4 needs to read out.
7 Set 1 to the corresponding bit in DDR to initiate descriptor fetch.
8 Set DMAC.DMAE=1 and expected DCSn.CTE=1 to launch DAM transfer.
9 Hardware clears the corresponding bit in DDR as soon as it starts to fetch the descriptor.
10 Waits for dma request from peripherals to start dma transfer.
11 After DMAC completes the current descriptor dma transfer, if DES0.Link=0, it sets DCSn.TT to 1. If the interrupt enabled, it will generates the corresponding interrupts.
12 If DES0.LINK=1, after DMAC completes the current descriptor dma transfer and return to fetch the next descriptor and continues dma transfer until completes the descriptor dma transfer which DES0.LINK=0.
13 When transfer end, clr DCSn.CTE to 0 to close the channel, and then clear DCSn.TT bits.

Table 10-4 Descriptor Structure

Word	Bit	Name	Function
1st (DES0)	31	EACKS	External DMA DACKn output polarity select
	30	EACKM	External DMA DACKn output Mode select
	29-28	ERDM	External DMA request detection Mode
	27	EOPM	External DMA End of process mode
	26-24	Reserved	
	23	SAI	Source Address Increment
	22	DAI	Target Address Increment
	21-20	Reserved	

	19-16	RDIL	Request Detection Interval Length
	15-14	SP	Source port width
	13-12	DP	Target port width
	11	Reserved	
	10-8	TSZ	Transfer Data Size
	7-3	Reserved	
	2	STDE	Stride transfer enable
	1	TIE	Transfer Interrupt Enable
	0	LINK	Descriptor Link Enable
2nd (DES1)	31-0	DSA	Source Address
3rd (DES2)	31-0	DTA	Target Address
4th (DES3)	31-24	DOA	Descriptor Offset address
	23-0	DTC	Transfer Counter
5th (DES4)	31-16	TSD	Target Stride Address
	15-0	SSD	Source Stride Address
6th(DES5)	31-6	Reserved	
	5-0	DRT	DMA Request Type
7th(DES6)	31-0	Reserved	
8th(DES7)	31-0	Reserved	

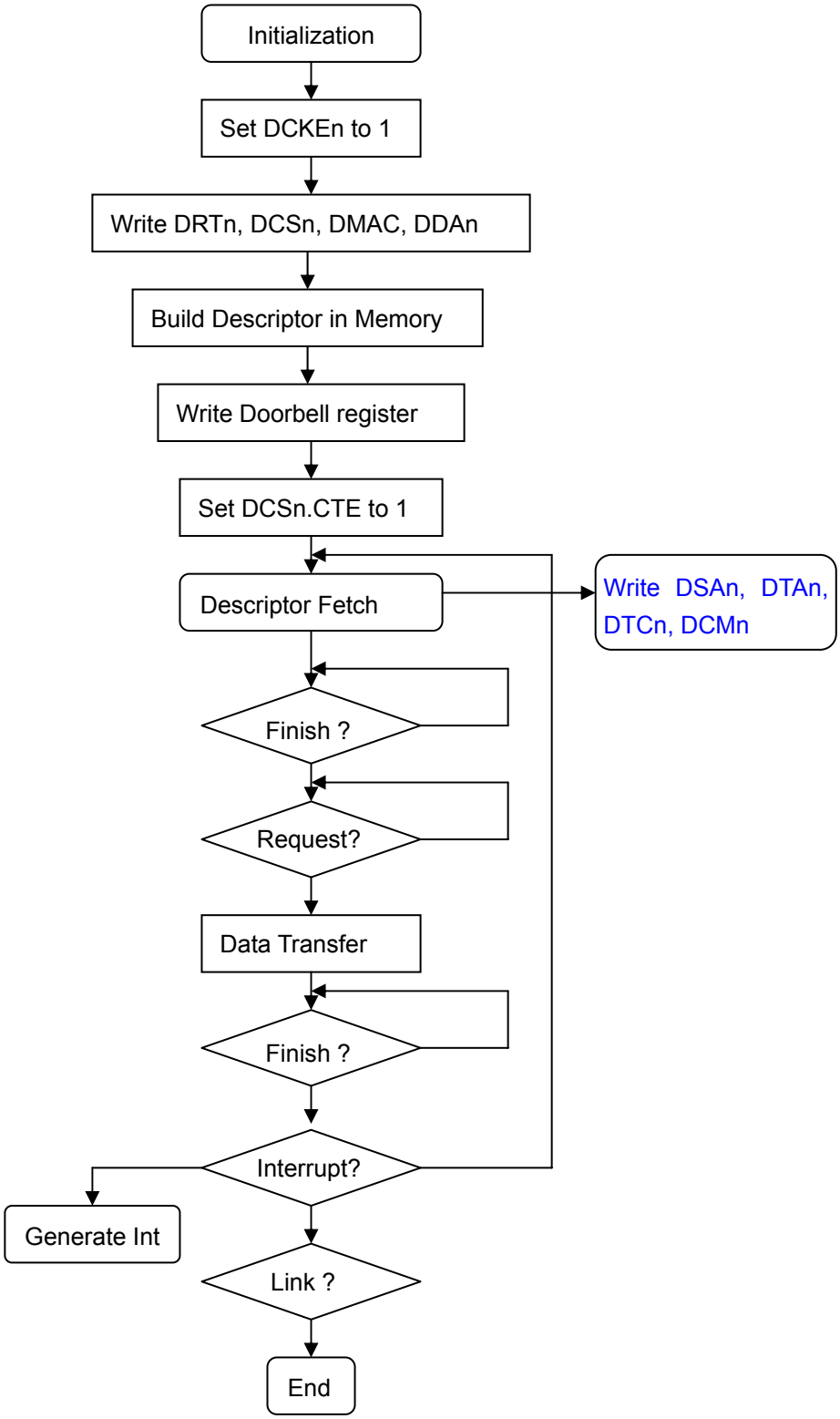


Figure 10-1 Descriptor Transfer Flow

10.3.1.2 Stride Address Transfer

During transfer, source or target address can be not continuous and the source and target stride offset address are showed in DSDn registers.

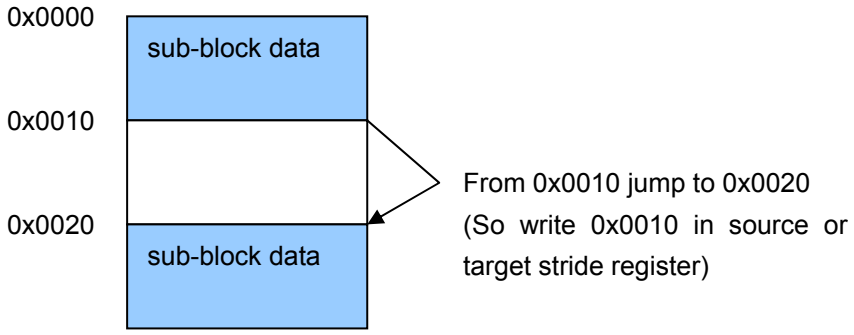


Figure 10-2 Example for Stride Address Transfer

10.3.2 No-Descriptor Transfer

To do proper DMA transfer, do as following steps:

1. First of all, check whether the status of DMA controller are available, that is, for global control (DMAC), ensure that DMAC.AR=0 and DMAC.HLT=0; while for expected channels, ensure that DCSn.AR=0, DCSn.HLT=0 and DCSn.TT=0 and DTCn=0.
2. For each channel n, initialize DSAn, DTAn, DTCn, DRTn, DCSn, DCMn properly.
3. Set DMAC.DMAE=1 and expected DCSn.CTE=1 and DCSn.NDES=1 to launch DMA transfer.

For a channel with auto-request (DRTn.RT=0x8), the transfer begins automatically when the DCSn.CTE bit and DMAC.DMAE bit are set to 1. While for a channel with other request types, the transfer does not start until a transfer request is issued and detected.

For any channel n, The DTCn value is decremented by 1 for each successful transaction of a data unit. When the specified number of transfer data unit has been completed (DTCn = 0), the transfer ends normally. Meanwhile corresponding bit of DIRQP is set to 1. If DCMn.TIE bit is set to 1, an interrupt request is sent to the CPU. However, during the transfer, if a DMA address error occurs, the transfer is suspended, both DCSn.AR and DMAC.AR are set to 1 as well as corresponding bit of DIRQP. Then an interrupt request is sent to the CPU despite of DCMn.TIE.

Sometimes, for example, an UART parity error occurs for a channel that is transferring data between such UART and another terminal. In the case, both DCSn.HLT and DMAC.HLT are set to 1 and the transfer is suspended. Software should identify halt status by checking such two bits and re-configure DMA to let DMA rerun properly later.

10.4 DMA Requests

DMA transfer requests are normally generated from either the data transfer source or target, but also they can be issued by on-chip peripherals that are neither the source nor the target. There are two DMA transfer request types: auto-request, and on-chip peripheral request. For any channel n, its transfer request type is determined through DRTn.

10.4.1 Auto Request

When there is no explicit transfer request signal available, for example, memory-to-memory transfer or memory to some on-chip peripherals like GPIO, the auto-request mode allows the DMA to automatically generate a transfer request signal internally. Therefore, when DMA initialization done, once the DMAC.DMAE and DCSn.CTE are set to 1, the transfer begins immediately in channel n which DRTn=0x8.

10.4.2 On-Chip Peripheral Request

In the mode, transfer request signals come from on-chip peripherals. All request types except 0x8 (value of DRT) belong to the mode.

NOTES:

1 the transfer byte number for one request detection according to DCMn.RDIL must be equal or less than the byte number according to receive or transmit trigger value of source or target devices.

10.5 Channel Priorities

There are two dma cores, each one supports 6 channels dma transfer. The two cores have the same priority.

In each core, there are two sets: set 1 has the higher priority than set 2, within each set priority is round robin.

Table 10-5 Relationship among DMA transfer connection, request mode & transfer mode

Transfer Connection	Request Mode	Transfer Mode	Data Size (bits)	Channel
External memory or memory-mapped external device and on-chip peripheral module	Auto on-chip	Single	8/16/32 16-byte/32-byte/ 64-byte	0~5

10.6 Examples

10.6.1 Memory-to-memory auto request No-Descriptor Transfer

Suppose you want to do memory move between two different memory regions through channel 3, for example, moving 1KB data from address 0x20001000 to 0x20011000, do as following steps:

1 Check if (DMAC.AR==0 && DMAC.HLT==0 && DCS3.AR==0 && DCS3.HLT==0 && DCS3.NDES=1 && DTC3==0).
2 If above condition is true, set value 0 to DCS3.CTE to disable the channel 3 temporarily.
3 Set source address 0x20001000 to DSA3 and target address 0x20011000 to DTA3.
4 Suppose the data unit is word, set transfer count number 256 (1024/4) to DTC3.
5 Set auto-request (0x8) to DRT3.
6 Up to now, only the most important channel control register DCM3 is left, set it carefully:
   - Set value 1 to SAI and DAI[*1].
   - Ignore RDIL because in the case there is no explicit request signal can be detected.
   - Set word size (0) to SP and DP[*2].
   - Set value 1 to TIE to let CPU do some post process after the transfer done.
7 Set value 1 to DCS3.CTE and DMAC.DMAE to launch the transfer in channels 3.
8 When the transfer terminates normally (DTC3==0 && DCS3.TT==1), DIRQP.CIRQ3 will automatically be set value 1 and an interrupt request will be sent to CPU.
9 When CPU grants the interrupt request, in the corresponding IRQ handler, software must clear the DCS3.TT to value 0, and the behavior will automatically clear DIRQP.CIRQ3.

NOTES:

1 [*1]: Either source or target is a FIFO, must not enable corresponding address increment.
2 [*2]: When either source or target need be accessed through EMC (external memory controller), the real port with of the device is encapsulated by EMC, so you can set any favorite port with for it despite of the real one.

11 MDMA Controller

MDMAC controller (MDMAC) is dedicated to transfer data between external memories and memory-mapped external devices.

11.1 Features

- Support up to 2 independent DMA channels
- Descriptor or No-Descriptor Transfer
- Transfer data units: byte, 2-byte (half word), 4-byte (word), 16-byte, 32-byte or 64-byte
- Transfer number of data unit: 1 ~ 2^{24}
- Independent source and target port width: 8-bit, 16-bit, 32-bit

11.2 Register Descriptions

Table 11-1 MDMAC Registers

Name	Description	RW	Reset Value	Address	Access Size (bit)
DSA0	DMA Source Address 0	RW	0x0	0x13030000	32
DTA0	DMA Target Address 0	RW	0x0	0x13030004	32
DTC0	DMA Transfer Count 0	RW	0x0	0x13030008	32
DRT0	DMA Request Source 0	RW	0x0	0x1303000C	32
DCS0	DMA Channel Control/Status 0	RW	0x0	0x13030010	32
DCM0	DMA Command 0	RW	0x0	0x13030014	32
DDA0	DMA Descriptor Address 0	RW	0x0	0x13030018	32
DSD0	DMA Stride Address 0	RW	0x0	0x1303001C	32
DSA1	DMA Source Address 1	RW	0x0	0x13030020	32
DTA1	DMA Target Address 1	RW	0x0	0x13030024	32
DTC1	DMA Transfer Count 1	RW	0x0	0x13030028	32
DRT1	DMA Request Source 1	RW	0x0	0x1303002C	32
DCS1	DMA Channel Control/Status 1	RW	0x0	0x13030030	32
DCM1	DMA Command 1	RW	0x0	0x13030034	32
DDA1	DMA Descriptor Address 1	RW	0x0	0x13030038	32
DSD1	DMA Stride Address 1	RW	0x0	0x1303003C	32
DMAC1	DMA Control 1 Register	R/W	0x0	0x13030300	32
DIRQP1	DMA Interrupt Pending 1	R	0x0	0x13030304	32
DDR1	DMA Doorbell 1 Register	RW	0x0	0x13030308	32
DDRS1	DMA Doorbell Set 1 Register	W	0x0	0x1303030C	32
DCKE1	DMA Clock Enable 1 Register	W	0x0	0x13030310	32

11.2.1 DMA Source Address (DSAn, n = 0 ~ 1)

DSA0, DSA1 **0x13030000, 0x13030020**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SA																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	SA	Source physical address.	RW

11.2.2 DMA Target Address (DTAn, n = 0 ~ 1)

DTA0, DTA1 **0x13030004, 0x13030024**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TA																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	TA	Target physical address.	RW

11.2.3 DMA Transfer Count (DTCn, n = 0 ~ 1)

DTC0, DTC1 **0x13030008, 0x13030028**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								TC																							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:24	Reserved	Write has no effect, read as zero.	R
23:0	TC	When Stride address transfer is disabled: TC hold the number of data unit to transfer and it counts down to 0 at the end; When Stride address transfer is enabled: TC composes of two parts: The lower 16 bits: the number of data unit for sub-block transfer The higher 8 bits: the number of sub-block And both the two parts count down to 0 at the end.	RW

11.2.4 DMA Request Types (DRTn, n = 0 ~ 1)

DRT0, DRT1 **0x1303000c, 0x1303002c**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																										RT					
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:6	Reserved	Write has no effect, read as zero.	R
5:0	RT	Transfer request type.	RW

Table 11-2 Transfer Request Types

RT5-0	Description
000000	Reserved
000001	Reserved
000010	Reserved
000011	Reserved
000100	Reserved
000101	Reserved
000110	Reserved
000111	Reserved
001000	Auto-request (external address → external address)
Other	Reserved

11.2.5 DMA Channel Control/Status (DCSn, n = 0 ~ 1)

DCS0, DCS1 **0x13030010, 0x13030030**

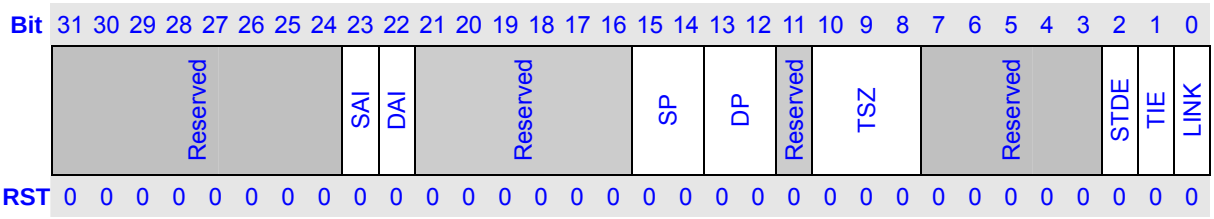
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NDES	DES8	Reserved						CDOA								Reserved											AR	TT	HLT	Reserved	CTE
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	NDES	Descriptor or No-Descriptor Transfer Select. 0: Descriptor Transfer; 1: No-descriptor Transfer.	RW
30	DES8	Descriptor 8 Word. 0: 4-word descriptor; 1: 8-word descriptor.	RW
29:24	Reserved	Write has no effect, read as zero.	R
23:16	CDOA	Copy of offset address of last completed descriptor from that in DMA command register. Software could know which descriptor is just completed combining with count terminate interrupt resulted by DCSn.CT. (Ignored in No-Descriptor Transfer)	RW
15:5	Reserved	Write has no effect, read as zero.	R
4	AR	Address Error. 0: no address error; 1: address error.	RW

3	TT	Transfer Terminate. 0: No-Link Descriptor or No-Descriptor DMA transfer does not end 1: No-Link Descriptor or No-Descriptor DMA transfer end	RW
2	HLT	DMA halt. 0: DMA transfer is in progress; 1: DMA halt.	RW
1	Reserved	Write has no effect, read as zero.	R
0	CTE	Channel transfer enable. 0: disable; 1: enable.	RW

11.2.6 DMA Channel Command (DCMn, n = 0 ~ 1)

DCM0, DCM1 **0x13030014, 0x13030034**



Bits	Name	Description	RW
31:24	Reserved	Write has no effect, read as zero.	R
23	SAI	Source Address Increment. 0: no increment; 1: increment	RW
22	DAI	Target Address Increment. 0: no increment; 1: increment.	RW
19:16	Reserved	Write has no effect, read as zero.	R
15:14	SP	Source port width. 00: 32-bit; 01: 8-bit; 10: 16-bit; 11: reserved.	RW
13:12	DP	Target port width. 00: 32-bit; 01: 8-bit; 10: 16-bit; 11: reserved.	RW
11	Reserved	Write has no effect, read as zero.	R
10:8	TSZ	Transfer Data Size of a data unit. 000: 32-bit; 001: 8-bit; 010: 16-bit; 011: 16-byte; 100: 32-byte; 101: 64-byte; others: reserved.	RW
7:3	Reserved	Write has no effect, read as zero.	R
2	STDE	Stride Disable/Enable. 0: address stride disable; 1: address stride enable.	RW
1	TIE	Transfer Interrupt Enable. (TIE) 0: disable interrupt; 1: enable interrupt when TT is set to 1.	RW
0	LINK	Descriptor Link Enable. 0: disable; 1: enable. (Ignored in No-Descriptor Transfer)	RW

11.2.7 DMA Descriptor Address (DDAn, n = 0 ~ 1)

This register is ignored in No-Descriptor Transfer.

DDA0, DDA1 **0x13030018, 0x13030038**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DBA																				DOA								Reserved			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:12	DBA	Descriptor Base Address.	RW
11:4	DOA	Descriptor Offset Address. When 4-descriptor is used, DOA is used for the offset address of DDA for next descriptor fetch.	RW
3:0	Reserved	Write has no effect, read as zero.	R

NOTES:

1 When 8-descriptor is used, next descriptor fetch address is from 8th word of last descriptor, that is the 0th~27th bit of the 8th word of last descriptor is mapped to DDA[31:4] for next descriptor fetch.

11.2.8 DMA Stride Address (DSDn, n = 0 ~ 1)

This register is ignored in No-Descriptor Transfer.

When address stride transfer is enabled in Descriptor mode, after a sub-block defined in DTCRn is finished transferring, the source or target stride address will be added up to the corresponding source or target address and the transfer will keep going until the transfer ends which means TC in DTCRn reach 0.

DSD0, DSD1 **0x1303001C, 0x1303003C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TSD																SSD															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:16	TSD	Target Stride Address.	RW
15:0	SSD	Source Stride Address.	RW

11.2.9 DMA Control

DMAC1 controls channel 0~1.

DMAC1 0x13030300

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																												HLT	AR	Reserved	DMAE
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:4	Reserved	Write has no effect, read as zero.	R
3	HLT	Global halt status, halt occurs in any channel, the bit should set to 1. 0: no halt 1: halt occurred	RW
2	AR	Global address error status, address error occurs in any channel, the bit should be set to 1. 0: no address error 1: address error occurred	RW
1	Reserved	Write has no effect, read as zero.	R
0	DMAE	Global DMA transfer enable. 0: disable DMA channel transfer 1: enable DMA channel transfer	RW

11.2.10 DMA Interrupt Pending (DIRQP)

DMAC supports total 2 pending interrupt which are in DIRQP.

DIRQP 0x13030304

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																														CIRQ1	CIRQ0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:2	Reserved	Write has no effect, read as zero.	R
1:0	CIRQn	CIRQn (n=0~1) denotes pending status for corresponding channel. 0: no abnormal situation or normal DMA transfer is in progress 1: abnormal situation occurred or normal DMA transfer done	RW

11.2.11 DMA Doorbell (DDR)

DDR supports channel 0~1.

DDR 0x13030308

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																														DB1	DB0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:3	Reserved	Write has no effect, read as zero.	R
2:0	DBn	DMA Doorbell for each channel, n=0~1, for example DB0 is for DMA channel 0. Software set it to 1 and hardware clears it to 0. 0: disable DMA controller to fetch the first descriptor or DMA controller clears it to 0 as soon as it starts to fetch the descriptor 1: Write 1 to DDS will set the corresponding DBn bit to 1 and enable DMA controller to fetch the first descriptor For example, write 0x00000001 to DDS, DB0 bit is set to 1 and enable DMA channel 0 to fetch the first descriptor. Write 0 to DDS, no meaning.	R

11.2.12 DMA Doorbell Set (DDRS)

DDRS supports channel 0~1.

DDRS 0x1303030c

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																														DBS1	DBS0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:2	Reserved	Write has no effect, read as zero.	R
1:0	DBSn	DMA Doorbell Set for each channel. 0: ignore 1: Set the corresponding DBn bit to 1	W

11.2.13 DMA Clock Enable (DCKE)

DCKE supports channel 0~1.

DCKE 0x13030310

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																										DCKE5	DCKE4	DCKE3	DCKE2	DCKE1	DCKE0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:3	Reserved	Write has no effect, read as zero.	R
2:0	DCKEn	DMA Clock Enable for each channel. 0: ignore 1: Set the corresponding DCKEn bit to 1	W

11.3 DMA manipulation

11.3.1 Descriptor Transfer

11.3.1.1 Normal Transfer

To do proper Descriptor DMA transfer, do as following steps:

1 First of all, open channel clock by setting DCKEn register for corresponding channel.

2 Check whether the status of DMA controller are available, that is, for global control (DMAC), ensure that DMAC.AR=0 and DMAC.HLT=0; while for expected channels, ensure that DCSn.AR=0, DCSn.HLT=0, DCSn.TT=0 and DTCn=0.

3 Select 4 word or 8 word descriptor by DCSn.DES8.

4 For Descriptor transfer, guarantee DCSn.NDES=0.

5 Initiate channel request register DRTn.

6 Build descriptor in memory. Write the first descriptor address in DDAn and the address must be 16Bytes aligned in 4word descriptor and 32Bytes aligned in 8word descriptor. The descriptor address includes two parts: Base and Offset address. If the descriptor is linked, the 32-bit address of next descriptor is composed of 20-bit Base address in DDAn and 8-bit Offset address in DES3.DOA and the four LSB is 0x0. See Table 11-3 for the detailed 4-word descriptor structure.

NOTE: if stride address transfer is enabled, the address must be 32Bytes aligned because DES4 needs to read out.

7 Set 1 to the corresponding bit in DDR to initiate descriptor fetch.

8 Set DMAC.DMAE=1 and expected DCSn.CTE=1 to launch DAM transfer.

9 Hardware clears the corresponding bit in DDR as soon as it starts to fetch the descriptor.

10 Waits for dma request from peripherals to start dma transfer.

11 After DMAC completes the current descriptor dma transfer, if DES0.Link=0, it sets DCSn.TT to 1. If the interrupt enabled, it will generates the corresponding interrupts.

12 If DES0.LINK=1, after DMAC completes the current descriptor dma transfer and return to fetch the next descriptor and continues dma transfer until completes the descriptor dma transfer which DES0.LINK=0.

13 When transfer end, clr DCSn.CTE to 0 to close the channel, and then clear DCSn.TT bits.

Table 11-3 Descriptor Structure

Word	Bit	Name	Function
1st (DES0)	31-24	Reserved	
	23	SAI	Source Address Increment
	22	DAI	Target Address Increment
	21-16	Reserved	
	15-14	SP	Source port width
	13-12	DP	Target port width
	11	Reserved	
	10-8	TSZ	Transfer Data Size

	7:3	Reserved	
	2	STDE	Stride transfer enable
	1	TIE	Transfer Interrupt Enable
	0	LINK	Descriptor Link Enable
2nd (DES1)	31-0	DSA	Source Address
3rd (DES2)	31-0	DTA	Target Address
4th (DES3)	31-24	DOA	Descriptor Offset address
	23-0	DTC	Transfer Counter
5th (DES4)	31-16	TSD	Target Stride Address
	15-0	SSD	Source Stride Address
6th(DES5)	31-6	Reserved	
	5-0	DRT	DMA Request Type
7th(DES6)	31-0	Reserved	
8th(DES7)	31-0	Reserved	

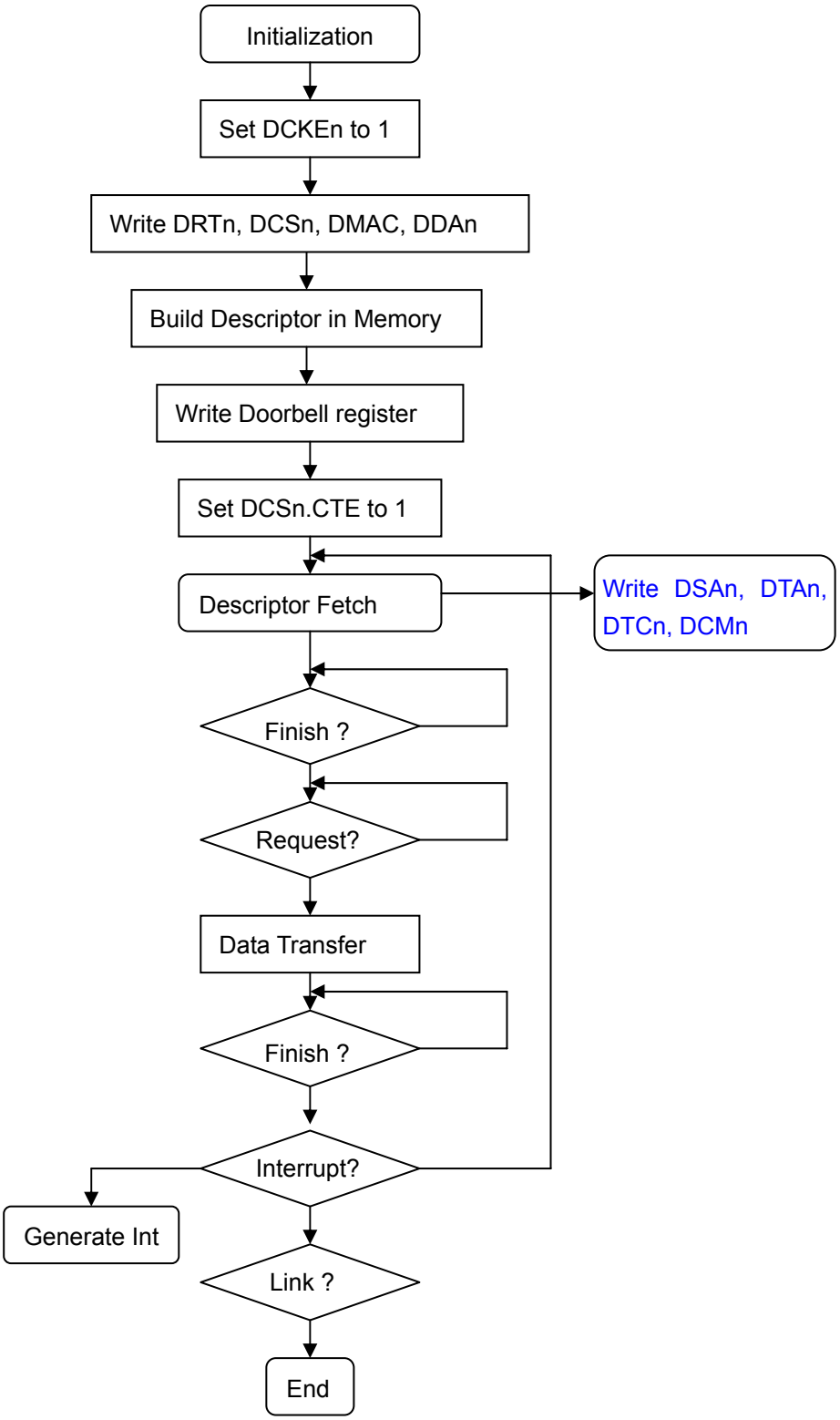


Figure 11-1 Descriptor Transfer Flow

11.3.1.2 Stride Address Transfer

During transfer, source or target address can be not continuous and the source and target stride offset address are showed in DSDn registers.

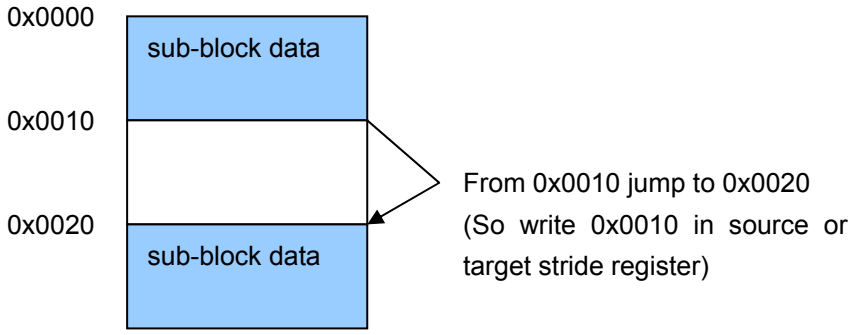


Figure 11-2 Example for Stride Address Transfer

11.3.2 No-Descriptor Transfer

To do proper DMA transfer, do as following steps:

1. First of all, check whether the status of DMA controller are available, that is, for global control (DMAC), ensure that DMAC.AR=0 and DMAC.HLT=0; while for expected channels, ensure that DCSn.AR=0, DCSn.HLT=0 and DCSn.TT=0 and DTCn=0.
2. For each channel n, initialize DSAn, DTAn, DTCn, DRTn, DCSn, DCMn properly.
3. Set DMAC.DMAE=1 and expected DCSn.CTE=1 and DCSn.NDES=1 to launch DMA transfer.

For a channel with auto-request (DRTn.RT=0x8), the transfer begins automatically when the DCSn.CTE bit and DMAC.DMAE bit are set to 1. While for a channel with other request types, the transfer does not start until a transfer request is issued and detected.

For any channel n, The DTCn value is decremented by 1 for each successful transaction of a data unit. When the specified number of transfer data unit has been completed (DTCn = 0), the transfer ends normally. Meanwhile corresponding bit of DIRQP is set to 1. If DCMn.TIE bit is set to 1, an interrupt request is sent to the CPU. However, during the transfer, if a DMA address error occurs, the transfer is suspended, both DCSn.AR and DMAC.AR are set to 1 as well as corresponding bit of DIRQP. Then an interrupt request is sent to the CPU despite of DCMn.TIE.

Sometimes, for example, an UART parity error occurs for a channel that is transferring data between such UART and another terminal. In the case, both DCSn.HLT and DMAC.HLT are set to 1 and the transfer is suspended. Software should identify halt status by checking such two bits and re-configure DMA to let DMA rerun properly later.

11.4 DMA Requests

DMA transfer requests are normally generated from either the data transfer source or target, but also they can be issued by on-chip peripherals that are neither the source nor the target. There are two DMA transfer request types: auto-request, and on-chip peripheral request. For any channel n, its transfer request type is determined through DRTn.

11.4.1 Auto Request

When there is no explicit transfer request signal available, for example, memory-to-memory transfer or memory to some on-chip peripherals like GPIO, the auto-request mode allows the DMA to automatically generate a transfer request signal internally. Therefore, when DMA initialization done, once the DMAC.DMAE and DCSn.CTE are set to 1, the transfer begins immediately in channel n which DRTn=0x8.

11.5 Channel Priorities

Table 11-4 Relationship among DMA transfer connection, request mode & transfer mode

Transfer Connection	Request Mode	Transfer Mode	Data Size (bits)	Channel
External memory or memory-mapped external device	Auto	Single	8/16/32 16-byte/32-byte/ 64-byte	0~5

11.6 Examples

11.6.1 Memory-to-memory auto request No-Descriptor Transfer

Suppose you want to do memory move between two different memory regions through channel 3, for example, moving 1KB data from address 0x20001000 to 0x20011000, do as following steps:

1 Check if (DMAC.AR==0 && DMAC.HLT==0 && DCS3.AR==0 && DCS3.HLT==0 && DCS3.NDES=1 && DTC3==0).
2 If above condition is true, set value 0 to DCS3.CTE to disable the channel 3 temporarily.
3 Set source address 0x20001000 to DSA3 and target address 0x20011000 to DTA3.
4 Suppose the data unit is word, set transfer count number 256 (1024/4) to DTC3.
5 Set auto-request (0x8) to DRT3.
6 Up to now, only the most important channel control register DCM3 is left, set it carefully.
7 Set value 1 to SAI and DAI[*1].
8 Ignore RDIL because in the case there is no explicit request signal can be detected.
9 Set word size (0) to SP and DP[*2].
10 Set value 1 to TIE to let CPU do some post process after the transfer done.
11 Set value 1 to DCS3.CTE and DMAC.DMAE to launch the transfer in channels 3.
12 When the transfer terminates normally (DTC3==0 && DCS3.TT==1), DIRQP.CIRQ3 will automatically be set value 1 and an interrupt request will be sent to CPU.
13 When CPU grants the interrupt request, in the corresponding IRQ handler, software must clear the DCS3.TT to value 0, and the behavior will automatically clear DIRQP.CIRQ3.

NOTES:

1 [*1]: Either source or target is a FIFO, must not enable corresponding address increment.
2 [*2]: When either source or target need be accessed through EMC (external memory controller), the real port with of the device is encapsulated by EMC, so you can set any favorite port with for it despite of the real one.

12 AHB Bus Arbiter

12.1 Overview

AHB bus arbiter is responsible for AHB bus transactions' arbitrating to provide a fair chance for each AHB master to possess the AHB bus. The refined arbiter in JZ4760 adopts a new arbitrating technique to fulfill the back-to-back feature of AHB protocol. Moreover, dividing two master groups with different privileges supports two arbitrating methods:

1 Round-robin possession for masters in the same group
2 Preemptive possession for masters with higher privileges

There are three sets of AHB buses in JZ4760 including AHB0, AHB1 and AHB2, and they all instance this arbiter.

12.2 Register Descriptions

The base address for memory-mapped registers of arbiter are listed below:

AHB0: 0x13000000

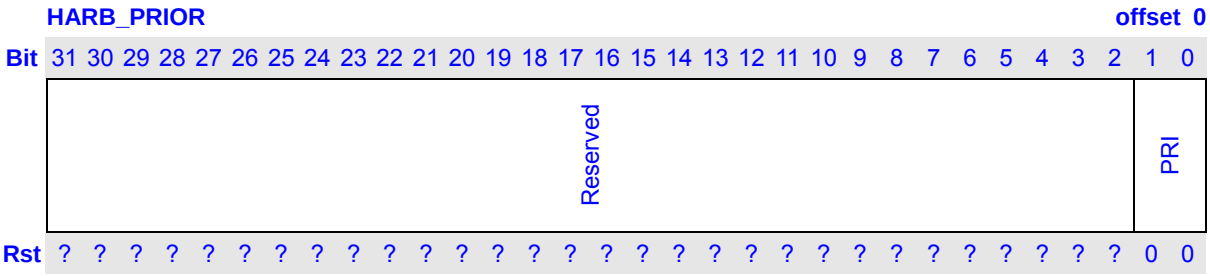
AHB1: 0x13200000

AHB2: 0x13400000

Table 12-1 AHB Bus Arbiter Registers List

Register Name	Offset	Size	R/W	Reset Value	Description
RPIOR	0x00	32	R/W	0x00000000	Master group priority order
CTRL	0x04	32	R/W	0x00000000	AHB monitor control
CLKL	0x08	32	R/W	undefined	AHB clock counter low
EVENT0L	0x0C	32	R/W	undefined	AHB bus event 0 counter low
EVENT1L	0x10	32	R/W	undefined	AHB bus event 1 counter low
EVENTH	0x14	32	R/W	undefined	AHB bus event & clock counter high
WATCHCTRL	0x18	32	RW	0x00000000	AHB bus watch control
WATCHADDR	0x1C	32	RW	undefined	AHB bus watch address
WATCHAMSK	0x20	32	RW	undefined	AHB bus watch address mask
WATCHDATA	0x24	32	RW	undefined	AHB bus watch data
WATCHDMSK	0x28	32	RW	undefined	AHB bus watch data mask

12.2.1 Priority Order Register



Bits	Name	Description	R/W
31:2	Reserved	Write is ignored, read as zero.	R
1:0	PRI	Priority for AHB0: (first 4 masters belong to high privilege group) 0: {bridge, cim, lcd, ipu}, {p1, dma, cabac, p0, mdma0, mdma1, mdma2} 1: {bridge, cim, lcd, p1}, {dma, cabac, p0, mdma0, mdma1, mdma2, ipu} 2: {bridge, lcd, ipu, p1}, {dma, cabac, p0, mdma0, mdma1, mdma2, cim} 3: {bridge, cim, ipu, p1}, {dma, cabac, p0, mdma0, mdma1, mdma2, lcd} Priority for AHB1: (first 3 masters belong to high privilege group) 0: {p0, idct, mc}, {me, dblk, mdma2, mdma1, mdma0, cabac, p1}	RW

		1: {p0, idct, me}, {dblk, mdma2, mdma1, mdma0, cabac, p1, mc} 2: {idct, mc, me}, {dblk, mdma2, mdma1, mdma0, cabac, p1, p0} 3: {p0, mc, me}, {dblk, mdma2, mdma1, mdma0, cabac, p1, idct} Priority for AHB2: (first 3 masters belong to high privilege group) 0: {p0 bridge, otg, gps}, {uhc, eth, dma, p1 bridge} 1: {p0 bridge, otg, uhc}, {eth, dma, p1 bridge, gps} 2: {otg, gps, uhc}, {eth, dma, p1 bridge, p0 bridge} 3: {p0 bridge, gps, uhc}, {eth, dma, p1 bridge, otg}	

12.2.2 Monitor Control Register

HARB_MC offset 4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									M1				M0				EV1				EV0									EV1E	EV0E	CLKE
Rst	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	0	0	0

Bits	Name	Description	R/W
31:24	Reserved	Write is ignored, read as zero.	R
23:20	M1	Monitored Master ID in monitor channel 1[*1].	RW
19:16	M0	Monitored Master ID in monitor channel 0[*1].	RW
15:12	EV1	AHB bus event encoding for monitor channel 1[*2].	RW
11:8	EV0	AHB bus event encoding for monitor channel 0[*2].	RW
7:3	Reserved	Write is ignored, read as zero.	R
2	EV1E	Enable monitor channel 1. 0: disable; 1: enable.	RW
1	EV0E	Enable monitor channel 0. 0: disable; 1: enable.	RW
0	CLKE	AHB clock counting enable. 0: disable; 1: enable.	RW

NOTES:

1 [*1] denotes the masterID encoding is described in the 2568HTable 12-3 AHB0 Master-ID.

2 [*2] the event encoding is described in the _Table 12-2 AHB Bus Monitor Events.

Table 12-2 AHB Bus Monitor Events

Events	Full Name	Comment
0	bus transaction cycles	exclude idle cycles.
1	bus transaction times	count NONSEQ times.
2	grant latency[*3]	count pending request (not granted) cycles.
3	critical grant latency trigger[*4]	Once the grant latency for a bus transaction

		exceeds the critical value preset in the counter low register, the associative counter high register will accumulate 1.
4	single beat transaction times	BURST type is SINGLE.
5	fixed length burst transaction times	BURST type is INCR4/8/16 or WRAP4/8/16.
6	INCR burst transaction times	BURST type is INCR.
7	critical transaction cycles trigger*5	Once the active transaction cycles for a bus transaction exceeds the critical value preset in the counter low register, the associative counter high register will accumulate 1.
8~15	reserved	

NOTES:

1 *3, *4, *5 denotes that such events are undefined when masterID is ALL.

Table 12-3 AHB0 Master-ID

Masters	Full Name
0	CIM
1	LCD
2	IPU
3	CORE1
4	DMA
5	CABAC
6	CORE0
7	MDMA0
8	MDMA1
9	MDMA2
10	AHB0-AHB2 bridge
11~14	Reserved
15	ALL (events triggered by any master should be monitored)

Table 12-4 AHB1 Master-ID

Masters	Full Name
0	CORE0
1	IDCT
2	MC
3	ME
4	DBLK
5	MDMA2
6	MDMA1

7	MDMA0
8	CABAC
9	CORE1
10~14	Reserved
15	ALL (events triggered by any master should be monitored)

Table 12-5 AHB2 Master-ID

Masters	Full Name
0	CORE0 bridge
1	OTG
2	GPS
3	UHC
4	ETH
5	DMA
6	CORE1 bridge
7~14	Reserved
15	ALL (events triggered by any master should be monitored)

12.2.3 AHB Clock Counter Low Register

HARB_CLKL **offset 8**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CLKL																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:0	CLKL	Record the low 32 bits of AHB clock counter.	RW

12.2.4 Event0 Low Register

HARB_EVENT0L **offset 12**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EVENT0L																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:0	EVENT0L	Record the low 32 bits of event 0 counter.	RW

12.2.5 Event1 Low Register

HARB_EVENT1L **offset 16**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EVENT1L																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:0	EVENT1L	Record the low 32 bits of event 1 counter.	RW

12.2.6 Event High Register

HARB_EVENTH **offset 20**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CLKH																EVENT1H								EVENT0H							
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:16	CLKH	Record the high 16 bits of AHB clock counter.	RW
15:8	EVENT1H	Record the high 8 bits of event 1 counter.	RW
7:0	EVENT0H	Record the high 8 bits of event 0 counter.	RW

NOTES:

1 Fields of EVENTH register will not overflow automatically. For example, when EVENT1H reaches 0xFF during monitoring, it remains the value until software modifies it.

12.2.7 AHB Watch Control Register

WATCH_CTRL offset 24

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	WF	RF	MasterID																					IRQE						DE	WE	RE
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	?	?	?	?	?	?	?	0	?	?	?	?	?	0	0	0

Bits	Name	Description	R/W
31	WF	0: no write watch point detected; 1: a write watch point is detected.	RW
30	RF	0: no read watch point detected; 1: a read watch point is detected.	RW
29:16	MasterID	0: ID of the master just triggering watch point, one-hot encoding.	RW
15:9		Reserved, read as zero.	R
8	IRQE	Interrupt enable. 1: if WF or RF is set value 1, an interrupt request arises immediately.	RW
7:3		Reserved, read as zero.	R
2	DE	Watch data enable. 1: enable.	RW
1	WE	Switch of triggering watch point by write. 1: enable.	RW
0	RE	Switch of triggering watch point by read. 1: enable.	RW

12.2.8 AHB Watch Address Register

WATCH_ADDR offset 28

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ADDR																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:0	ADDR	Watch address to be monitored.	RW

12.2.9 AHB Watch Address Mask Register

WATCH_AMSK **offset 32**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MASK																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:0	MASK	if a bit is set 0, which means in WATCH_ADDR, corresponding bit position should be monitored, otherwise, the bit position should be ignored.	RW

12.2.10 AHB Watch Data Register

WATCH_DATA **offset 36**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATA																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:0	DATA	Read or Write data to be monitored.	RW

12.2.11 AHB Watch Data Mask Register

WATCH_DMSK **offset 40**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MASK																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
31:0	MASK	if a bit is set 0, which means in WATCH_DATA, corresponding bit position should be monitored, otherwise, the bit position should be ignored.	RW

13 Clock Reset and Power Controller

13.1 Overview

The Clock & Power management block consists of three parts: Clock control, PLL control, and Power control, Reset control.

The Clock control logic can generate the required clock signals including CCLK for CPU, , HCLK for the AHB0 and AHB1 bus peripherals, H2CLK for the AHB2 bus peripherals, MCLK for DDR or SDRAM Memory and PCLK for the APB bus peripherals. The Chip has two Phase Locked Loops (PLL): for CCLK, HCLK, H2CLK and PCLK, MCLK, GPUCLK, GPSCLK, SSICLK, MSCLK, LPCLK, USBCLK, I2SCLK. The clock control logic can make slow clocks without PLL and connect/disconnect the clock to each peripheral block by software, which will reduce the power consumption.

For the power control logic, there are various power management schemes to keep optimal power consumption for a given task. The power management block can activate four modes: NORMAL mode, DOZE mode, IDLE mode, SLEEP mode.

Support power supply shut down for 3 power domain separately. Software may separately shut down AHB1 module and GPS module. When in Sleep mode, software may shut down J1. Thus, the chip may best reduce leakage current.

For reset control logic, the hardware reset and hibernate reset is extended to more 40ms. It controls or distributes all of the system reset signals.

13.2 Clock Generation UNIT

The clock generation unit (CGU) contains one PLL driven by an external oscillator and the clock generation circuit from which the following clocks are derived:

Signal	Description
CCLK	Fast clock for internal operations such as executing instructions from the cache. It can be gated during doze and idle mode when all the criteria to enter a low power are met.
HCLK	AHB0 and AHB1 High Speed Bus Clock.
H2CLK	AHB2 Speed Bus Clock.
PCLK	APB Speed Bus Clock.
MCLK	Clock for EMC or DDR controller.
SCLK	Clock for NEMC controller.
CKO	DDR or SDRAM Clock.
LPCLK	LCD pixel clock.
TVECLK	TV encoder 27M clock.
CIM_MCLK	Clock output from CIM module.
CIM_PCLK	Clock input to CIM module.
GPUCLK	GPU clock.
GPSCLK	GPS clock.
I2SCLK	I2S codec clock.
BITCLK	AC97 bit clock.
PCMCLK	PCM clock.
MSCCLK	MSC clock.
SSICLK	SSI clock.
TSSICLK	TSSI clock.
EXCLK	12M clock output for UART I2C TCU USB2.0-PHY AUDIO CODEC.

Feature:

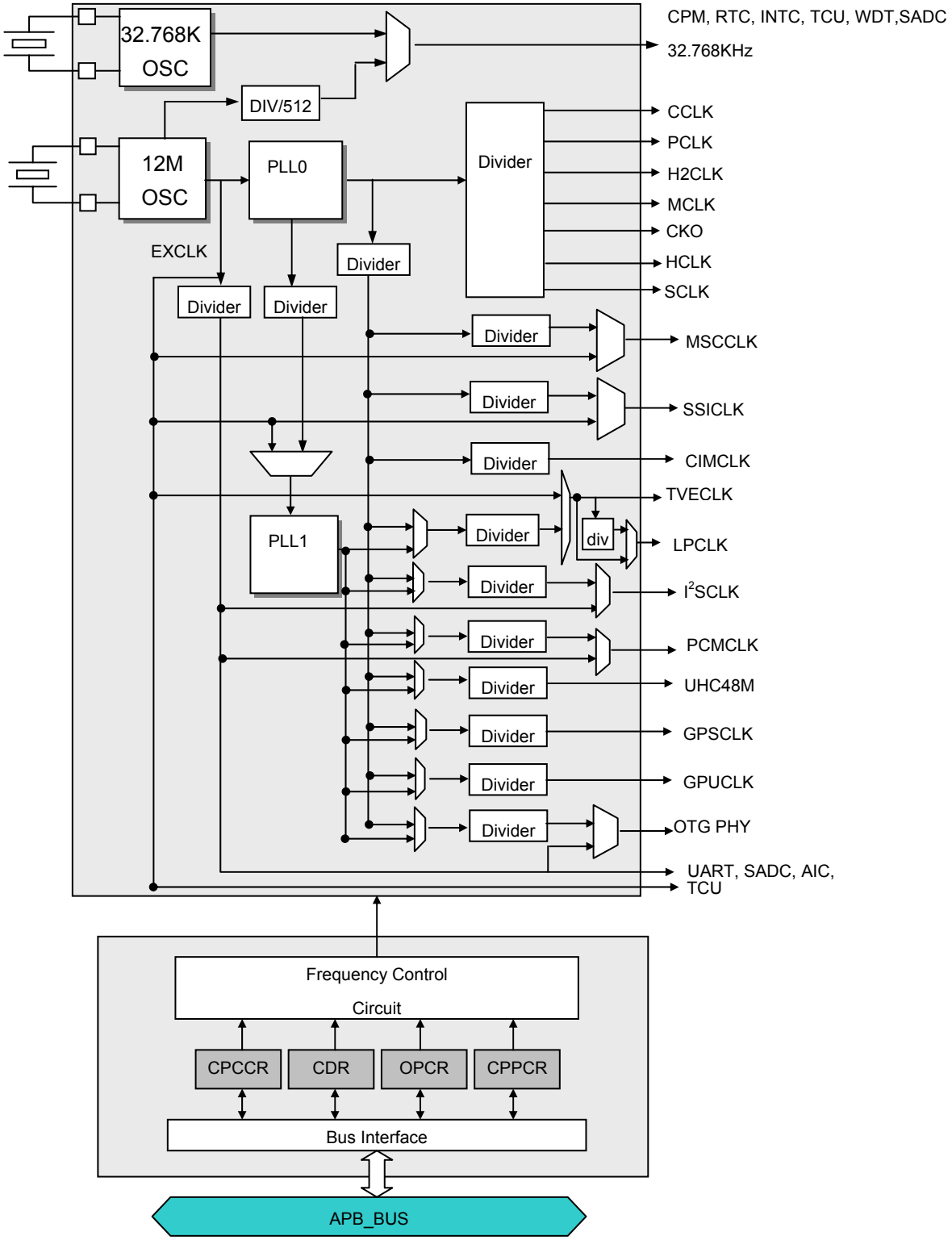
- On-chip 2MHz~27MHZ oscillator circuit
- On-chip 32.768KHZ oscillator circuit
- One two-chip phase-locked loops (PLL) with programmable multiplier
- CCLK, HCLK, PCLK, H2CLK, MCLK , SCLK ,CKO and LPCLK, GPUCLK, GPSCLK, MSCCLK , UHCCLK, SSICLK frequency can be changed separately for software by setting registers
- SSI clock supports 50M clock
- MSC clock supports 50M clock
- Functional-unit clock gating
- Shut down power supply for J1, AHB1 and GPS

13.2.1 Pin Description

Name	I/O	Description
RTCLK_XI	Input	32.768KHZ Oscillator input signal
RTCLK_XO	Output	32.768KHZ Oscillator output signal
EXCLK	Input	Oscillator input signal
EXCLKO	Output	Oscillator output signal
CIM_MCLK	Output	Clock output from CIM module signal
CIM_PCLK	Input	Clock input to CIM module signal
LPCLK	Output	LCD pix clock signal
CKO	Output	SDRAM or DDR clock signal
TSSICLK	Input	TSSI clock signal
BITCLK	Inout	I2S/AC97 bit clock
PCMCLK	Inout	PCM bit clock
MSC_CLK	Output	Clock output For MMC0/SD0 Card signal
SSI_CLK	Output	Clock output from SSI module signal

13.2.2 CGU Block Diagram

Following figure illustrates a block diagram of CGU.



13.2.3 Clock Overview

There is an internal PLL in this chip. PLL input clock is an external input clock EXCLK. Theoretically, EXCLK can be 2MHz ~ 27MHz.

CCLK is CPU clock. It is usually the fastest clock in the chip. This clock represents the chip speed.

HCLK is for on chip high speed peripherals connected to AHB0 and AHB1 bus.

H2CLK is for on chip high speed peripherals connected to AHB2 bus.

PCLK is for on chip slow speed peripherals connected to APB bus.

MCLK is external memory bus clock. MCLK represents the SDRAM or DDR speed.

SCLK is for static memory frequency. SCLK frequency can be equal to H2CLK or SCLK/H2CLK = 1/2.

CCLK, HCLK, H2CLK, PCLK MCLK and SCLK are synchronous clocks that may have different frequencies. They are from the same clock source, the on chip PLL output clock in most cases. HCLK frequency can be equal to CCLK or divided CCLK by an integer. PCLK frequency can be equal to H2CLK or divided H2CLK by an integer. MCLK frequency can be equal to or half of HCLK.

AC97 in AIC module needs a 12.288MHz BIT clock. It is input from the external AC97 CODEC chip or other clock source. I2S and PCM clock are generated from PLL output clock.

Besides PLL input, EXCLK also provides device clock or one of device clocks for many peripherals, such as, UART, TCU, SSI, SADC and WDT.

Device clock of MSC (MMC/SD) is taken from software divided PLL output clock or 12M from oscillator.

USB device and host controllers need a 48MHz USB clock. USB clock can be selected by software divided PLL output clock.

Device clock of SSI is taken from software divided PLL output clock. or 12M from oscillator.

LCD's pixel clock are generated from PLL output clock, which are divided by one independent dividers.

GPU clock and GPS clock are generated from PLL output clock, which are divided by one independent dividers.

The slowest clock is RTCLK, which is usually 12M/512 or 32768Hz.

13.2.4 CGU Registers

All CGU register 32bit access address is physical address.

Table 13-1 CGU Registers Configuration

Name	description	RW	Reset Value	Address	Access Size
CPCCR	Clock Control Register	RW	0x01011100	0x10000000	32
CPPCR	PLL Control Register0	RW	0x28080011	0x10000010	32
CPPSR	PLL switch and status register	RW	0x80000000	0x10000014	32
CPPCR1	PLL Control Register1	RW	0x28080002	0x10000030	32
CPSPR	CPM Scratch Pad Register	RW	0x????????	0x10000034	32
CPSPPR	CPM Scratch Protected Register	RW	0x0000a5a5	0x10000038	32
USBPCR	USB Parameter control register	RW	0x42992198	0x1000003C	32
USBRDT	USB Reset Detect Timer Register	RW	0x00000096	0x10000040	32
USBVBFIL	USB jitter filter Register	RW	0x00000080	0x10000044	32
USBCDR	OTG PHY clock divider Register	RW	0x00000000	0x10000050	32
I2SCDR	I2S device clock divider Register	RW	0x00000000	0x10000060	32
LPCDR	LCD pix clock divider Register	RW	0x00000000	0x10000064	32
MSCCDR	MSC clock divider Register	RW	0x00000000	0x10000068	32
UHCCDR	UHC 48M clock divider Register	RW	0x00000000	0x1000006C	32
SSICDR	SSI clock divider Register	RW	0x00000000	0x10000074	32
CIMCDR	CIM MCLK clock divider Register	RW	0x00000000	0x1000007C	32
GPSCDR	GPS clock divider Register	RW	0x00000000	0x10000080	32
PCMCDR	PCM device clock divider Register	RW	0x00000000	0x10000084	32
GPUCDR	GPU clock divider Register	RW	0x00000000	0x10000088	32

13.2.4.1 Clock Control Register

The Clock Control Register (CPCCR) is a 32-bit read/write register, which controls CCLK, HCLK, H2CLK, PCLK, MCLK and SCLK division ratios. It is initialized to 0x01011100 by any reset. Only word access can be used on CPCCR.

CPCCR **0x10000000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ECS	MEM	Reserved		SDIV				Reserved	CE	PCS	Reserved	H2DIV				MDIV				PDIV				HDIV				CDIV			
RST	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0

<table>
<tr><th>Bits</th><th>Name</th><th>Description</th><th>RW</th></tr>
<tr><td>31</td><td>ECS</td><td>Select the clock source between EXCLK and EXCLK/2 output.
0: clock source is EXCLK
1: clock source is EXCLK/2
The bit is only used to APB device such as UART I2S I2C SSI SADC OTG_PHY etc.
Usually, please don't change the bit when EXCLK is 12M.</td><td>RW</td></tr>
<tr><td>30</td><td>MEM</td><td>0: mobile ddr or sdram memory
1: ddr or ddr2 memory</td><td>RW</td></tr>
<tr><td>29:28</td><td>Reserved</td><td></td><td></td></tr>
<tr><td>27:24</td><td>SDIV</td><td>Divider for Static memory Clock Frequency. Specified the SCLK division ratio.
<table>
<tr><th colspan="4">Bit 27~24: SDIV</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>X1</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>X1/2</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>X1/3</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>X1/4</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>X1/6</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>X1/8</td></tr>
<tr><td colspan="4">Other Value</td><td>Reserved</td></tr>
</table></td><td>RW</td></tr>
<tr><td>23</td><td>Reserved</td><td></td><td></td></tr>
<tr><td>22</td><td>CE</td><td>change enable. If CE is 1, writes on CDIV, H2DIV, HDIV, PDIV, MDIV, SDIV, USBCDR, LPCDR, CIMCDR, MSCCDR, SSICDR, GPSCDR, UHCCDR, GPUCDR , PCMCDR , I2SCDR will start a frequency changing sequence immediately. When CE is 0, writes on CDIV, H2DIV, HDIV, PDIV, MDIV, SDIV, USBCDR, LPCDR, CIMCDR, MSCCDR, SSICDR, GPSCDR, UHCDR, PCMCDR, GPUCDR, I2SCDR will not start a frequency changing sequence immediately. The division ratio is actually updated in PLL multiple ratio changing sequence or PLL Disable Sequence.
0: Division ratios are updated in PLL multiple ratio changing sequence or PLL Disable Sequence
1: Division ratios are updated immediately</td><td>RW</td></tr>
<tr><td>21</td><td>PCS</td><td>PLL out clock source clock selection. It supplies source clock for MSC I2S LCD UHC OTG SSI PCM GPU GPS.
0: divider clock source is PLL output divided by 2
1: divider clock source is PLL output
Software should set the bit according to various needs.</td><td>RW</td></tr>
<tr><td>20</td><td>Reserved</td><td></td><td>R</td></tr>
<tr><td>19:16</td><td>H2DIV</td><td>Divider for AHB2 Clock Frequency. Specified the H2CLK division ratio.
<table>
<tr><th colspan="4">Bit 19~16: H2DIV</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>X1</td></tr>
</table></td><td>RW</td></tr>
</table>

			0	0	0	1	X1/2	
			0	0	1	0	X1/3	
			0	0	1	1	X1/4	
			0	1	0	0	X1/6	
			0	1	0	1	X1/8	
			Other Value				Reserved	
15:12	MDIV	Divider for Memory Clock Frequency. Specified the MCLK division ratio.						RW
			Bit 15~12: MDIV				Description	
			0	0	0	0	X1	
			0	0	0	1	X1/2	
			0	0	1	0	X1/3	
			0	0	1	1	X1/4	
			0	1	0	0	X1/6	
			0	1	0	1	X1/8	
			Other Value				Reserved	
11:8	PDIV	Divider for Peripheral Clock Frequency. Specified the PCLK division ratio.						RW
			Bit 11~8: PDIV				Description	
			0	0	0	0	X1	
			0	0	0	1	X1/2	
			0	0	1	0	X1/3	
			0	0	1	1	X1/4	
			0	1	0	0	X1/6	
			0	1	0	1	X1/8	
			Other Value				Reserved	
7:4	HDIV	Divider for AHB0 Clock Frequency. Specified the HCLK division ratio.						RW
			Bit 7~4: HDIV				Description	
			0	0	0	0	X1	
			0	0	0	1	X1/2	
			0	0	1	0	X1/3	
			0	0	1	1	X1/4	
			0	1	0	0	X1/6	
			0	1	0	1	X1/8	
			Other Value				Reserved	
3:0	CDIV	Divider for CPU Clock Frequency. Specifies the CCLK division ratio.						RW
			Bit 3~0: HDIV				Description	
			0	0	0	0	X1	
			0	0	0	1	X1/2	
			0	0	1	0	X1/3	
			0	0	1	1	X1/4	
			0	1	0	0	X1/6	
			0	1	0	1	X1/8	

			Other Value	Reserved		

13.2.4.2 I2S device clock divider Register

I2S device clock divider Register (I2SCDR) is a 32-bit read/write register that specifies the divider of I2S device clock . This register is initialized to 0x00000000 only by any reset. Only word access can be used on I2SCDR.

I2SCDR **0x10000060**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	I2CS	I2PCS	Reserved																					I2SCDR								
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	I2CS	I2S Clock Source Selection. Selects the I2S clock source between PLL output and pin EXCLK. 0: I2S clock source is EXCLK 1: I2S clock source is PLL output divided by I2SDIV If EXCLK is 12M, please don't change the bit.	R
30	I2PCS	0: select PLL0 clock output 1: select PLL1 clock output	
29:9	Reserved	Writes to these bits have no effect and always read as 0.	R
8:0	I2SCDR	Divider for I2S Frequency. Specified the I2S device clock division ratio, which varies from 1 to 512 (division ratio = I2SCDR + 1). When EXCLK is 12M, don't care the bit.	RW

13.2.4.3 USB clock divider Register

USB clock divider Register (USBCDR) is a 32-bit read/write register that specifies the divider of OTG PHY clock . This register is initialized to 0x00000000 only by any reset. Only word access can be used on USBCDR.

USBCDR **0x10000050**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	UCS	UPCS	Reserved																								USBCDR					
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	UCS	OTG PHY Clock Source Selection. Selects the OTG PHY clock source between PLL output and pin EXCLK. 0: OTG clock source is pin EXCLK 1: OTG clock source is PLL output If EXCLK is 12M, please don't change the bit.	RW
30	UPCS	0: select PLL0 clock output 1: select PLL1 clock output	
29:9	Reserved	Writes to these bits have no effect and always read as 0.	R
5:0	USBCDR	Divider for OTG PHY Clock Frequency. When OTG PHY clock source is PLL (UCS bit is 1), this field specified the OTG PHY clock division ratio, which varies from 1 to 64 (division ratio = USBCDR + 1).	RW

13.2.4.4 LCD pix clock divider Register

LCD pix clock divider Register (LPCDR) is a 32-bit read/write register that specifies the divider of LCD pixel clock (LPCLK). This register is initialized to 0x00000000 only by any reset. Only word access can be used on LPCDR.

LPCDR **0x10000064**

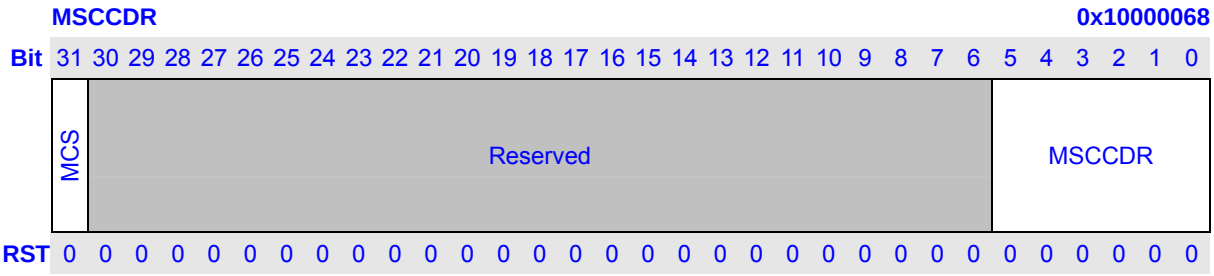
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LSCS	LTCS	LPCS	Reserved																							LPCDR					
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	LSCS	TV encoder Source Pixel Clock Selection. Selects the TV encoder source pixel clock between divider and external clock input.	RW

		0: TV encoder source pixel clock source is PLL divider output 1: TV encoder source clock source is EXCLK PIN	
30	LTCS	LCD TV Encoder or Panel pix clock Selection. 0: pix clock is used as LCD PANEL 1: pix clock is used as TV ENCODER	RW
29	LPCS	0: select PLL0 clock output 1: select PLL1 clock output	RW
29:11	Reserved	Writes to these bits have no effect and always read as 0.	R
10:0	LPCDR	Divider for Pixel Frequency. Specified the LCD pixel clock (LPCLK) division ratio, which varies from 1 to 2048 (division ratio = LPCDR + 1).	RW

13.2.4.5 MSC device clock divider Register

MSC device clock divider Register (MSCCDR) is a 32-bit read/write register that specifies the divider of MSC device clock . This register is initialized to 0x00000000 only by any reset. Only word access can be used on MSCCDR.



Bits	Name	Description	RW
31	MCS	MSC Clock Source Selection. Selects the MSC clock source between PLL output and pin EXCLK. 0: MSC clock source is EXCLK 1: MSC clock source is PLL output divided by MSCCDR	RW
30:5	Reserved	Writes to these bits have no effect and always read as 0.	R
4:0	MSCCDR	Divider for MSC Frequency. Specified the MSC device clock division ratio, which varies from 1 to 32 (division ratio = MSCCDR + 1).	RW

13.2.4.6 UHC device clock divider Register

UHC device clock divider Register (UHCCDR) is a 32-bit read/write register that specifies the divider of UHC 48M device clock . This register is initialized to 0x00000000 only by any reset. Only word access can be used on UHCCDR.

UHCCDR **0x1000006C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	UPCS	Reserved																											UHCCDR				
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	UHPCS	0: select PLL0 clock output 1: select PLL1 clock output	
30:4	Reserved	Writes to these bits have no effect and always read as 0.	
3:0	UHCCDR	Divider for UHC Frequency. Specified the UHC 48M device clock division ratio, which varies from 1 to 16 (division ratio = UHCCDR + 1).	RW

13.2.4.7 SSI device clock divider Register

SSI device clock divider Register (SSICDR) is a 32-bit read/write register that specifies the divider of SSI device clock . This register is initialized to 0x00000000 only by any reset. Only word access can be used on SSICDR.

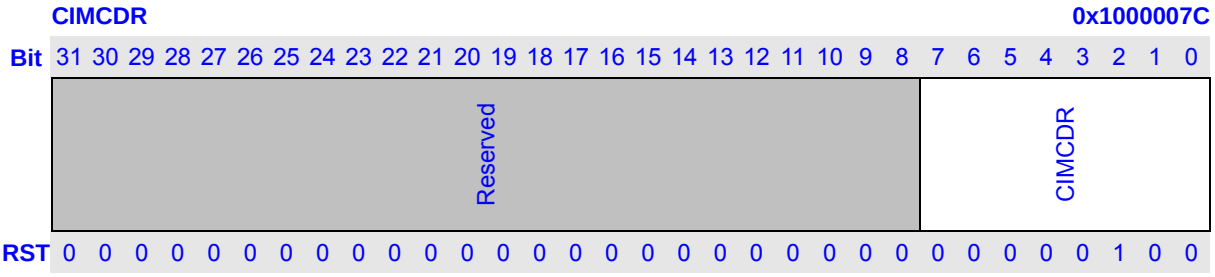
SSICDR **0x10000074**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SCS	Reserved																											SSICDR				
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	SCS	SSI Clock Source Selection. Selects the SSI clock source between PLL output and pin EXCLK. 0: SSI clock source is EXCLK 1: SSI clock source is PLL output divided by SSICDR	R
31:4	Reserved	Writes to these bits have no effect and always read as 0.	R
3:0	SSICDR	Divider for SSI Frequency. Specified the SSI device clock division ratio, which varies from 1 to 32 (division ratio = SSICDR + 1).	RW

13.2.4.8 CIM MCLK clock divider Register

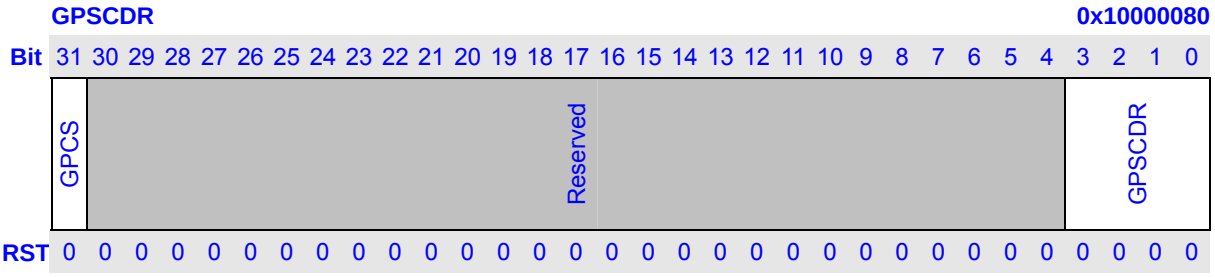
CIM mclk clock divider Register (CIMCDR) is a 32-bit read/write register that specifies the divider of CIM mclk clock (CIM_MCLK). This register is initialized to 0x00000000 only by any reset. Only word access can be used on CIMCDR.



Bits	Name	Description	RW
31:8	Reserved	Writes to these bits have no effect and always read as 0.	R
7:0	CIMCDR	Divider for CIM MCLK Frequency. Specified the CIM MCLK clock (CIM_MCLK) division ratio, which varies from 1 to 256 (division ratio = CIMCDR + 1).	RW

13.2.4.9 GPS clock divider Register

GPS clock divider Register (GPSCDR) is a 32-bit read/write register that specifies the divider of GPS device clock (GPSCLK). This register is initialized to 0x00000000 only by any reset. Only word access can be used on GPSCDR.



Bits	Name	Description	RW
31	GPCS	0: select PLL0 clock output 1: select PLL1 clock output	R
30:4	Reserved	Writes to these bits have no effect and always read as 0.	R
3:0	GPSCDR	Divider for GPS clock Frequency. Specified the GPS clock division ratio, which varies from 1 to 16 (division ratio = GPSCDR + 1).	RW

13.2.4.10 PCM device clock divider Register

PCM device clock divider Register (PCMCDR) is a 32-bit read/write register that specifies the divider of PCM device clock . This register is initialized to 0x00000000 only by any reset. Only word access can be used on PCMCDR.

PCMCDR **0x10000084**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PCMS	PCMPCS	Reserved																					PCMCDR								
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	PCMS	PCM source clock Selection. 0: PCM source clock is pin EXCLK 1: PCM source clock is PLL divider output	RW
30	PCMPCS	0: select PLL0 divider output 1: select PLL1 divider ouput	
30:9	Reserved	Writes to these bits have no effect and always read as 0.	R
8:0	PCMCDR	Divider for PCM Frequency. Specified the PCM device clock division ratio, which varies from 1 to 512 (division ratio = PCMCDR + 1).	RW

13.2.4.11 GPU clock divider Register

GPU clock divider Register (GPUCDR) is a 32-bit read/write register that specifies the divider of GPU clock . This register is initialized to 0x00000000 only by any reset. Only word access can be used on GPUCDR.

GPUCDR **0x10000088**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	GPCS	Reserved																												GPUCDR		
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	GPCS	0: select PLL0 divider output 1: select PLL1 divider ouput	RW
30:3	Reserved	Writes to these bits have no effect and always read as 0.	R
2:0	GPUCDR	Divider for GPU Frequency. Specified the GPU clock division ratio, which varies from 1 to 8 (division ratio = GPUCDR + 1).	RW

13.2.4.12 CPM Scratch Pad Protected Register

The Scratch Pad Protected Register is reset to 0x0000a5a5. When CPSPPR value equals to 0x00005a5a, software can write the CPSPR.

CPSPPR 0x10000038

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																CPSPPR															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1	1	0	1	0	0	1	0	1

Bits	Name	Description	RW
31:16	Reserved		R
15:0	CPSPPR	The value is only = 0x00005a5a, software can write the CPSPR.	RW

13.2.4.13 CPM Scratch Pad Register

The Scratch Pad Register is a 32-bit read/write register that allows software to preserve some critical data . It is not initialized by poweron and WDT reset.

CPPSR 0x10000034

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CPPSR																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

13.2.4.14 USB Parameter Control Register

The USBPCR is a 32-bit read/write register that allows software to control OTG PHY some functions. It is initialized to 0x42992198.

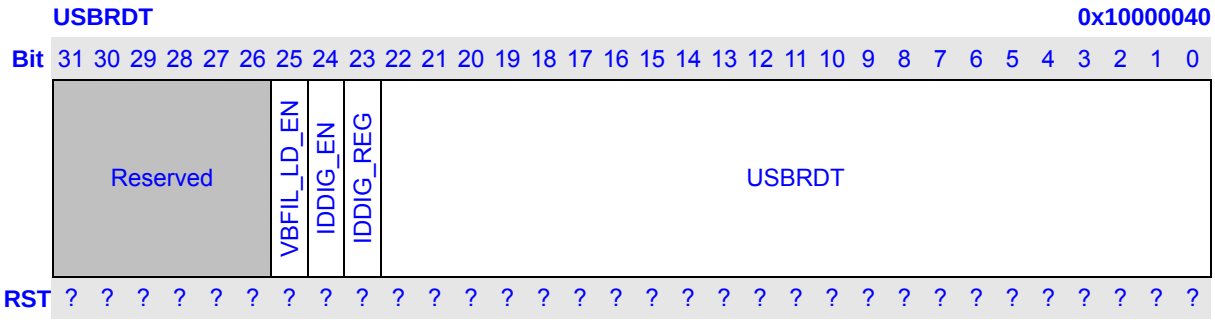
USBPCR 0x1000003C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	USB_MODE	AVLD_REG	IDPULLUP_MASK		INCR_MASK	CLK12_EN	COMONONN	VBUSVLDEXT	VBUSVLDEXTSEL	POR	SIDDQ	OTGDISABLE	COMPDISTUNE		OTGTUNE		SQRXTUNE		TXFSLSTUNE		TXPREMTUNE	TXRISETUNE	TXVREFTUNE								
RST	0	1	0	0	0	0	1	0	1	0	0	1	1	0	0	1	0	0	1	0	0	0	0	1	1	0	1	1	0	1	0	1

Bits	Name	Description	RW
31	USB_MODE	0: work as USB device 1: work as OTG	RW
30	AVLD_REG	This bit is used to set “avalid”(VBUS above A-device session threshold) signal.	RW
29:28	IDPULLUP_MASK	These 2 bits control “idpullup” signal in otg mode. 2’b1x: “idpullup” always active 2’b01: “idpullup” always active when usb suspend 2’b00: use “idpullup” from otg controller	RW
27	INCR_MASK	This bit controls whether the ahb interface enhancement for “incr transfer” takes effect. Set this bit to 0 will active the enhancement.	RW
26	CLK12_EN	OTG PHY reference clock enable.	RW
25	COMMONNN	This bit is the OTG PHY common block power down control signal. 0: The common blocks remain powered in suspend mode 1: The common blocks are powered down in suspend mode	RW
24	VBUSVLDEXT	This bit controls OTG PHY VBUSVLDEXT signal.	RW
23	VBUSVLDEXTSEL	This bit controls OTG PHY VBUSVLDEXTSEL signal.	RW
22	POR	This bit controls OTG PHY power on reset.	RW
21	SIDDQ	This bit is the OTG PHY analog blocks power down signal.	RW
20	OTG_DISABLE	This bit is the power control for otg block in OTG PHY.	RW
19:17	COMPDISTUNE	These bits control disconnect threshold adjustment. 3’b111 +4.5% 3’b110 +3% 3’b101 +1.5% 3’b100 Default 3’b011 -1.5% 3’b010 -3% 3’b001 -4.5% 3’b000 -6%	RW
16:14	OTGTUNE	These bits control VBUS valid threshold adjustment. 3’b111 +4.5% 3’b110 +3% 3’b101 +1.5% 3’b100 Default 3’b011 -1.5% 3’b010 -3% 3’b001 -4.5%	RW

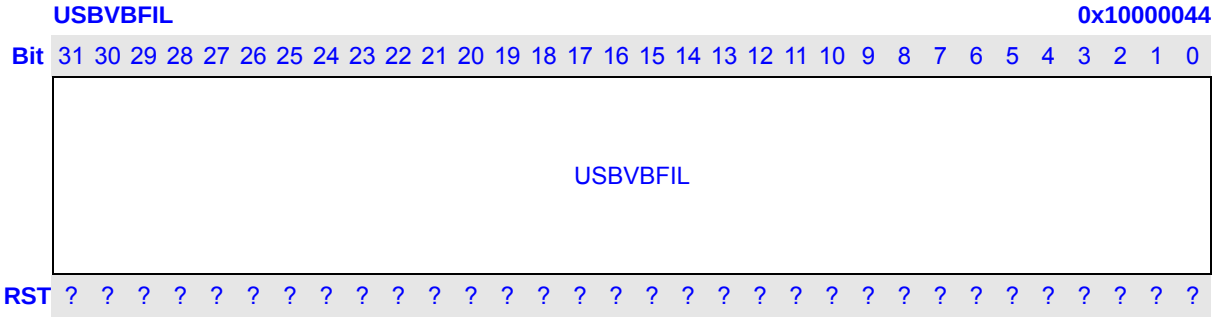
		3’b000	-6%	
13:11	SQRXTUNE	These bits control squelch threshold adjustment.		RW
		3’b111	-15%	
		3’b110	-10%	
		3’b101	-5%	
		3’b100	Default	
		3’b011	+5%	
		3’b010	+10%	
		3’b001	+15%	
		3’b000	+20%	
10:7	TXFSLSTUNE	These bits control FS/LS source impedance adjustment.		RW
		4’b1111	-5%	
		4’b0111	-2.5%	
		4’b0011	Default	
		4’b0001	+2.5%	
		4’b0000	+5%	
6	TXPREEMPHTUNE	This bit controls HS transmitter Pre-emphasis enable. 1: enable 0: disable		RW
5:4	TXRISETUNE	These bits controls transmitter rise/fall time adjustment.		RW
		2’b11	Default	
		2’b10	+4%	
		2’b01	+8%	
		2’b00	+12%	
3:0	TXVREFTUNE	These bits control HS DC voltage level adjustment.		RW
		4’b1111	+12.5%	
		4’b1110	+11.25%	
		4’b1101	+10%	
		4’b1100	+8.75%	
		4’b1011	+7.5%	
		4’b1010	+6.255	
		4’b1001	+5%	
		4’b1000	+3.75%	
		4’b0111	+2.5%	
		4’b0110	+1.25%	
		4’b0101	Default	
		4’b0100	-1.25%	
		4’b0011	-2.5%	
		4’b0010	-3.75%	
		4’b0001	-5%	
		4’b0000	-6.25%	

13.2.4.15 USB Reset Detect Timer Register



Bits	Name	Description	RW
31:26	Reserved		R
25	VBFIL_LD_EN	VBUS filter data load enable.	RW
24	IDDIG_EN	This bit indicates using IDDIG_REG to control “iddig” signal.	RW
23	IDDIG_REG	This bit controls “iddig” when IDDIG_REG_EN = 1’b1.	RW
22:0	USBRDT	These bits control USB reset detect time.	RW

13.2.4.16 USB VBUS Jitter Filter Register



Bits	Name	Description	RW
31:0	USBVBFIL	These bits controls VBUS jitter filter time.	RW

13.2.4.17 PLL Control Register0

The PLL Control Register (CPPCR) is a 32-bit read/write register, which controls PLL multiplier, on/off state and stabilize time. It is initialized to 0x28080020 only by any reset. Only word access can be used on CPPCR.

CPPCR 0x10000010

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	PLLM									PLLN				PLLOD		LOCK0	ENLCOK	Reserve			PLLS	PLLBP	PLLEN	PLLST							
RST	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Bits	Name	Description	RW
31	Reserved		RW
30:24	PLLM	the PLL feedback 7-bit divider.	RW
23:22	Reserved		
21:18	PLLN	the PLL input 4-bit divider.	RW
17:16	PLLOD	00: divide by 1 01: divide by 2 10: divide by 4 11: divide by 8	RW
15	LOCK0	0: the PLL output is stable 1: the PLL output is not stable Software should clear this bit to 0, when this bit equal to 1, it indicates that PLL hadn't stable previously , it is only used to debug.	RW
14	ENLOCK	PLL Stable Method. 0: not use PLL LOCK method 1: use PLL LOCK method Software should not change this bit usually, it is only used to debug.	RW
13:11	Reserved	Writes to these bits have no effect and always read as 0.	R
10	PLLS	PLL Stabilize Flag. 0: PLL is off or not stable 1: PLL is on and stable	R
9	PLLBP	PLL Bypass. If PLLEN is 1, set this bit to1 will bypass PLL. The PLL is still running background but the source of associated dividers is switched to 12-M. If PLLEN is 0, set this bit to 1 has no effect. If PLLEN is 1, clear this bit to 0 will switch the source of associated dividers to PLL output.	RW
8	PLLEN	PLL Enable. When PLLEN is set to 1, PLL starts to lock phase. After PLL stabilizes, PLLS bit is set. If PLLBP is 0, the source of associated dividers, is switched to PLL output. When PLLEN is clear to 0, PLL is shut off and the source of associated dividers is switched to 12-MHz in spite of PLLBP bit.	RW
7:0	PLLST	PLL Stabilize Time. Specifies the PLL stabilize time by unit of RTCCLK (approximate 32kHz) cycles. It is used when change PLL multiplier or change PLL from off to on. It is initialized to H'20.	RW

13.2.4.18 PLL Control Register1

The PLL Control Register (CPPCR1) is a 32-bit read/write register, which controls PLL multiplier, on/off state and stabilize time. It is initialized to 0x28080002 only by any reset. Only word access can be used on CPPCR1.

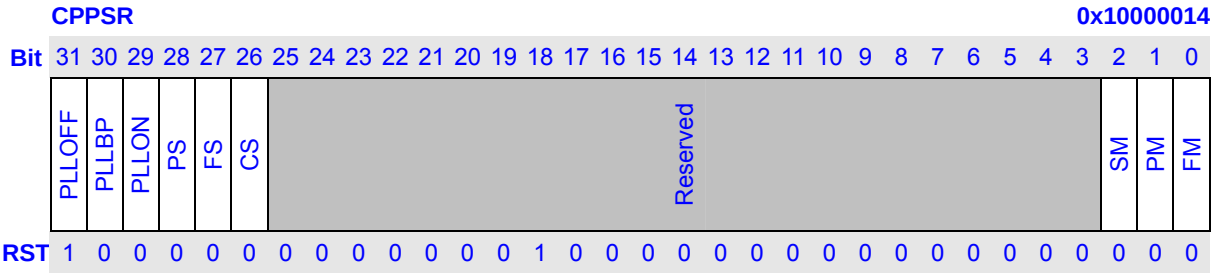
CPPCR1 **0x10000030**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	PLL1M							Reserved	PLL1N					PLL1OD		P1SCS	P1SDIV						Reserved	PLL1EN	PLL1S	Reserved				LOCK1	PLL1OFF	PLL1ON
RST	0	0	1	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Bits	Name	Description	RW
31	Reserved		RW
30:24	PLL1M	the PLL1 feedback 7-bit divider.	RW
22	Reserved		
21:18	PLL1N	the PLL1 input 4-bit divider.	RW
17:16	PLL1OD	00: divide by 1 01: divide by 2 10: divide by 4 11: divide by 8	RW
15	P1SCS	0: select EXCLK as PLL1 input clock 1: select PLL0 divided clock as PLL1 input clock	RW
14:9	P1SDIV	PLL0 clock dividers as PLL1 input clock.	
8	Reserved		
7	PLL1EN	PLL1 Enable. When PLL1EN is set to 1, PLL1 starts to lock phase. After PLL1 stabilizes, PLL1S bit is set. When PLL1EN is clear to 0, PLL1 is shut off.	RW
6	PLL1S	PLL1 Stabilize Flag. 0: PLL1 is off or not stable 1: PLL 1is on and stable	R
5:3	Reserved		
2	LOCK1	0: the PLL output is stable 1: the PLL output is not stable Software should clear this bit to 0, when this bit equal to 1, it indicates that PLL hadn't stable previously , it is only used to debug.	RW
1	PLLOFF	0 : PLL1 doesn't enter shut off state 1: PLL1 is in shut off state	R
0	PLLON	0: PLL1 doesn't enter on state 1: PLL1 is in on state	R

13.2.4.19 PLL Switch and Status Register

The PLL Switch and Status Register (CPPSR) is a 32-bit read/write register, which controls the clock switch ,frequency change mode and reflect the PLL and clock switch Status .It is initialized to 0x80000000 by any reset. Only word access can be used on CPPSR.



Bits	Name	Description	RW
31	PLLOFF	0 : PLL doesn't enter shut off state 1: PLL is in shut off state	R
30	PLLBP	0: PLL doesn't enter by pass state 1: PLL is in by pass state	R
29	PLLON	0: PLL doesn't enter on state 1: PLL is in on state	R
28	PS	0: disable PLL or no change PLL parameters 1: enable PLL or change PLL parameters have finished The bit is asserted to 1 auto by hardware . when software concerns this bit, at first software write 0 to the bit, then read the status bit until to 1.	RW
27	FS	Indicate the change frequency has finished . the bit only reflect CDIV, HDIV, MDIV, PDIV , H2DIV, SDIV change. 0: no change CDIV, HDIV, MDIV, PDIV, H2DIV, SDIV 1: change clock parameters have finished when software concerns this bit, at first software write 0 to the bit, then read the status bit until to 1.	RW
26	CS	Indicate the clock switch has finished, the bit reflects when PLL switch to EXCLK or EXCLK to PLL. 0: no clock switch 1: clock switch has finished. when software concerns this bit, at first software write 0 to the bit, then read the status bit until to 1.	RW
25:3	Reserved		R
2	SM	When cdiv hdiv mdiv pdiv ,h2div ,sdiv change, whether cclk h1clk h0clk mclk pclk are all stopped. 0: hardware control 1: when frequency changes, above clocks are all stopped	RW
1	PM	Clock switch mode. When PLL switch to EXCLK or EXCLK switch to PLL. 0: slow mode	RW

		1: fast mode	
0	FM	Clock frequency change mode. Only to CDIV MDIV HDIV PDIV. 0: slow mode 1: fast mode	RW

13.2.5 PLL Operation

The PLL developed as a macro cell for clock generator. It can generate a stable high-speed clock from a slower clock signal. The output frequency is adjustable and can be up to 1500MHz. The PLL integrates a phase frequency detector (PFD), a low pass filter (LPF), a voltage controlled oscillator (VCO) and other associated support circuitry. All fundamental building blocks as well as fully programmable dividers are integrated on the core. It is useful for clock multiplication of stable crystal oscillator sources and for de-skew clock signals.

The PLL block diagram is shown in following figure:

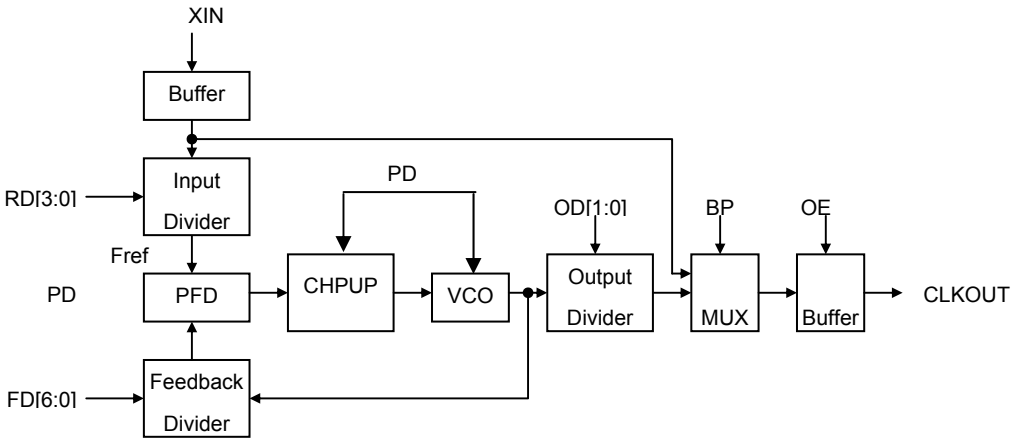


Figure 13-1 Block Diagram of PLL

13.2.5.1 PLL Configuration

PLL Divider Value Setting

There are 3 divider values (N, M and NO) to set the PLL output clock frequency CLKOUT:

1 Input Divider Value N.

N = PLLN of CPPCR

2 Feedback Divider Value M.

M = PLLM of CPPCR

3 Output Divider Value NO.

Output Divider Setting (OD)	Output Divider Value (NO)
0	1
1	2
2	4
3	8

4 The PLL output frequency, CLK_OUT, is determined by the ratio set between the value set in the input divider and the feedback divider. PLL output frequency CLK_OUT is calculated from the following equations:

CLKOUT = XIN x (M / N) x (1/ NO)

M = M1*2 + M2*4 + M3*8 + M4*16 + M5*32 + M6*64 + M7*128
N = N0*1 + N1*2 + N2*4 + N3*8
NO = $2^{od0+2od1}$

Where:

CLK_OUT represents the output frequency
XIN represents PLL input frequency
N represents input divider value
M represents feedback divider value
NO represents output divider value

< Attention >

1) 1MHZ ≤ XIN/N ≤ 50MHZ.
2) 500MHZ ≤ CLK_OUT x NO ≤ 1500MHZ.
3) M>=4; N>=2.

13.2.6 Main Clock Division Change Sequence

Main clock (CCLK, HCLK, H2CLK, PCLK MCLK SCLK) frequencies can be changed separately or simultaneously by changing division ratio. Following conditions must be obeyed:

1 CCLK must be integral multiple of HCLK, H2CLK.
2 HCLK must be equal to MCLK or twice of MCLK.
3 HCLK must be equal to H2CLK or twice of H2CLK.
4 HCLK H2CLK MCLK SCLK must be integral multiple of PCLK.
5 SCLK must be equal to H2CLK or twice of H2CLK.

Don't violate this limitation, otherwise unpredictable error may occur.

In normal mode, if CE bit of CPCCR is 1, changing CDIV, HDIV, H2DIV, PDIV , MDIV, SDIV will start a Division Change Sequence immediately. If CE bit of CPCCR is 0, changing CDIV, HDIV, H2DIV, PDIV MDIV SDIV will not start Division Change Sequence.

13.2.7 Change Other Clock Frequencies

The divider of LCD pixel clock (LPCLK), I2S device clock, SSI device clock, MSC device clock ,USB clock, UHC clock, PCM clock, GPS and GPU clock can be changed by programming LPCDR , I2SCDR, SSICDR, MSCCDR , USBCDR, UHCCDR, PCMCDR, GPSCDR and GPUCDR respectively.

Change LPCDR I2SCDR SSICDR MSCCDR , USBCDR, UHCCDR, PCMCDR, GPSCDR, GPUCDR as following steps:

1 Stop related devices with clock-gate function. Clock supplies to the devices are stopped.
2 Change LPCDR, I2SCDR, SSICDR, MSCCDR, USBCDR, UHCCDR, PCMCDR, GPSCDR, GPUCDR . If CE is 1, clock frequencies are changed immediately. If CE is 0, clock frequencies are not changed until PLL Multiplier Change Sequence is started.
3 Cancel above clock-gate function.

13.2.8 Change Clock Source Selection

USB, I2S device clocks, PCM device clock, LCD pix clock , MSC clock and SSI clock can be selected from two sources. Before change clock source, corresponding devices should be stopped using clock-gate function.

1 When USB clock source is changed (UCS bit of CPCCR), USB clock should be stopped.
2 When I2S clock source is changed (I2CS bit of CPCCR), AIC should be stopped.
3 When LCD pix clock source is changed (LSCS LTCS bit of LPCDR), LCD should be stopped.
4 When MSC clock source is changed (MCS of MSCCDR), MSC should be stopped.
5 When SSI clock source is changed (SCS of SSICDR), SSI should be stopped.

When UCS, I2CS , LSCS, LTCS, MCS, SCS bit is changed, clock source is changed immediately.

When PCS of CPPCR is changed, the LCD AIC MSC SSI clock should be stopped.
When ECS of CPCCR is changed, the UART SADC I2C clock should be stopped.

When P1SCS of CPPCR1 or P1SDIV is changed, the corresponding module should be stopped.

13.2.9 Two PLL Source Selection

USB, I2S, PCM, GPS, GPU, LCD, UHC source clock can be selected from PLL0 or PLL1. Before change clock source, corresponding devices should be stooped using clock-gate function.

13.2.10 EXCLK Oscillator

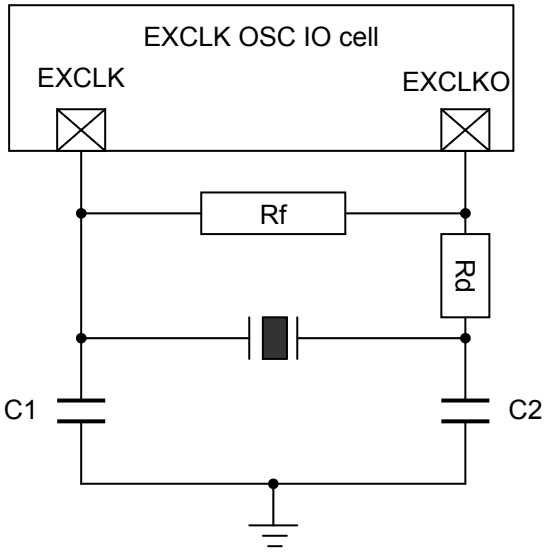


Figure 13-2 Oscillating circuit for fundamental mode

To turn on the oscillator, the oscillating circuit must provide the negative resistance (-Re) at least five times the equivalent series resistance (ESR) of the crystal sample. For larger -Re value, faster turn on the crystal. Higher gm provides larger -Re therefore can start-up the crystal with higher ESR for the same load capacitance (CL). However, it's required higher power consumption.

There are two key parameters to turn on oscillator. Which are CL and the maximum ESR at the target frequency? By reducing the CL, the -Re can be increased thus; shorter turn on time can be achieved. However, if CL is too small, the deviation from the target frequency will increase because of the capacitance variation. So, a trade-off relationship between short turn on time and small frequency deviation in deciding CL value. The smaller ESR of the crystal sample will reduce turn on time but the price is higher. The typical CL and ESR values for difference target frequencies are listed in Table 13-2.

Table 13-2 Typical CL and the corresponding maximum ESR

Target Frequency (Hz)	2M ~ 3M	3M ~ 6M	6M ~ 10M	10M ~ 20M
CL (pf)	25	20	16	12
Maximum ESR (ohm)	1K	400	100	80

Figure 13-2 shows the oscillating circuit is connected with the oscillator I/O cell. Components feedback resistor (Rf), damping resistor (Rd), C1 and C2 are used to adjust the turn on time, keep stability and accurate of the oscillator.

Rf is used to bias the inverter in the high gain region. It cannot be too low or the loop may not oscillate.

For mega Hertz range applications, Rf of 1Mohm is applied.
Rd is used to increase stability, low power consumption, suppress the gain in high frequency region and also reduce -Re of the oscillator. Thus, proper Rd cannot be too large to cease the loop oscillating.

C1 and C2 are deciding regard to the crystal or resonator CL specification. In the steady state of oscillating, CL is defined as (C1*C2)/(C1+C2). Actually, the I/O ports, bond pad, and package pin all contribute the parasitic capacitance to C1 and C2. Thus, CL can be rewrite to (C1'*C2')/(C1'+C2'), where C1'=(C1+Cin,stray) and C2'=(C2+Cout,stray). In this case, the required C1 and C2 will be reduced.

Notice, this oscillating circuit is for parallel resonate but not series resonate. Because C1, C2, Rd and Rf are varying with the crystal specifications; therefore there is no single magic number of all the applications.

13.3 Power Manager

In the Low-Power mode, part or whole processor is halted. This will reduce power consumption. The Power Management Controller contains low-power mode control and reset sequence control.

13.3.1 Low-Power Modes and Function

The processor supports six low-power modes and function:

- NORMAL mode
  In Normal mode, all peripherals and the basic blocks including power management block, the CPU core, the bus controller, the memory controller, the interrupt controller, DMA, and the external master may operate completely. But, the clock to each peripheral, except the basic blocks, can be stopped selectively by software to reduce the power consumption.

- DOZE mode
  DOZE mode is entered by setting DOZE bit of LCR to 1. In DOZE mode, clock is burst to CPU core and the clock duty is set by DUTY field of LCR. DOZE mode is canceled by reset, interrupt or clearing DOZE bit to 0. Continuous clock is supplied immediately after DOZE mode is canceled. The other Clocks except CCLK run continuously in DOZE mode.

- IDLE mode
  In IDLE mode, the clock to the CPU core is stopped except the bus controller, the memory controller, the interrupt controller, and the power management block. To exit the IDLE mode, the any interrupts should be activated.

- SLEEP mode
  In SLEEP mode, all clocks except RTC clock are disabled. PLL is disabled also. SLEEP mode is canceled by reset or interrupt. When SLEEP mode is canceled, PLL is restarted , the PLL needs clock stabilization time (PLL lock time). This PLL stabilization time is automatically inserted by the internal logic with lock time count register. and all clocks start operating after PLL stability time.

- CLOCK GATE function
  CLOCK GATE function is used to gate specified on-chip module when it is not used. Set specified CLKG0~40 bits in CLKGR will enter specified CLK gate function. CLOCK gate function is canceled by reset or clearing specified CLKGR0~40 to 0.

- Power down Mode
  In order to reduce power leakage, software may shut down power supply for AHB1 and GPS module. When system enters into SLEEP mode, the software may shut down power for J1 according to OPCR.PD bit.

13.3.2 Register Description

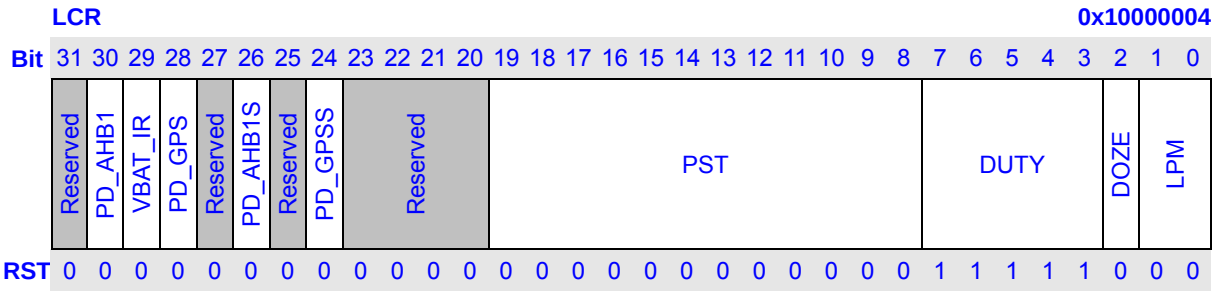
All PMC register 32bit access address is physical address.

Table 13-3 Power/Reset Management Controller Registers Configuration

Name	description	RW	Initial Value	Address	Access Size
LCR	Low Power Control Register	RW	0x000000F8	0x10000004	32
PSWC0ST	Power Switch Chain0 Start Time	RW	0x00000000	0x10000090	32
PSWC1ST	Power Switch Chain1 Start Time	RW	0x00000000	0x10000094	32
PSWC2ST	Power Switch Chain2 Start Time	RW	0x00000000	0x10000098	32
PSWC3ST	Power Switch Chain3 Start Time	RW	0x00000000	0x1000009c	32
CLKGR0	Clock Gate Register0	RW	0x3FFFFFE0	0x10000020	32
OPCR	Oscillator and Power Control Register	RW	0x00001570	0x10000024	32
CLKGR1	Clock Gate Register1	RW	0x0000017F	0x10000028	32

13.3.2.1 Low Power Control Register

The Low Power Control Register (LCR) is a 32-bit read/write register that controls low-power mode status. It is initialized to 0x000000F8 by any reset.



Bits	Name	Description	RW
31	Reserved		R
30	PD_AHB1	Power Down Module AHB1. 0: not shut down power supply to AHB1 1: shut down power supply to AHB1	RW
29	VBAT_IR	Use internal resistor network. 0: directly measure VBAT on pin AUX2 1: measure VBAT on pin VBAT. (use internal resistors). Details pls refer sadc spec.	RW
28	PD_GPS	Power Down Module GPS. 0: not shut down power supply to GPS 1: shut down power supply to GPS	RW
27	Reserved		R

26	PD_AHB1S	AHB1 power down status. 0: AHB1 module not shut down 1: AHB1 module has entered shut down mode	R
25	Reserved		R
24	PD_GPSS	GPS power down status. 0: GPS module not shut down 1: GPS module has entered shut down mode	R
21:20	Reserved		R
19:8	PST	Power stability Time. Specifies the Power stabilize time by unit of RTCCLK (approximate 32kHz) cycles.	RW
7:3	DUTY	CPU Clock Duty. Control the CPU clock duty in doze mode. When the DUTY field is 0x1F, the clock is always on and when it is zero, the clock is always off. Set the DUTY field to 0 when the CPU will be disabled for an extended amount of time. 00000: 0/31 duty-cycle 00001: 1/31 duty-cycle 00010: 2/31 duty-cycle … 11111: 31/31 duty-cycle	RW
2	DOZE	Doze Mode. Control the doze mode. When doze mode is canceled, this bit is cleared to 0 automatically. 0: Doze mode is off 1: Doze mode is on	RW
1:0	LPM	Low Power Mode. Specifies which low-power mode will be entered when SLEEP instruction is executed. Bit 1~0: **00**: IDLE mode will be entered when SLEEP instruction is executed **01**: SLEEP mode will be entered when SLEEP instruction is executed **10**: Reserved **11**: Reserved	RW

13.3.2.2 Power Switch Chain0 Start Time Register

PSWC0ST **0x10000090**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																				PSWC0ST											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.3.2.3 Power Switch Chain1 Start Time Register

PSWC1ST																														0x10000094
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																				PSWC1ST											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.3.2.4 Power Switch Chain2 Start Time Register

PSWC2ST																														0x10000098
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																				PSWC2ST											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.3.2.5 Power Switch Chain3 Start Time Register

PSWC3ST																														0x1000009C
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																				PSWC3ST											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

NOTES:

1 The Start Time by the unit of PCLK cycles.

13.3.2.6 Clock Gate Register0

The Clock Gate Register (CLKGR0) is a 32-bit read/write register that controls the CLOCK GATE function of peripherals. It is reset to 0x2FFFFFE0.

CLKGR																														0x10000020
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CLKGR0																															
RST	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0

<table>
<tr><th>Bits</th><th>Name</th><th>Description</th><th>RW</th></tr>
<tr><td>31:0</td><td>CLKGR0</td><td>Clock gate Bits. Controls the clock supplies to some peripherals. If set, clock supplies to associated devices are stopped, and registers of the device cannot be accessed also.

<table>
<tr><th>Bit</th><th>Module</th><th>Description</th></tr>
<tr><td>31</td><td>EMC</td><td></td></tr>
<tr><td>30</td><td>DDR</td><td></td></tr>
<tr><td>29</td><td>IPU</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>28</td><td>LCD</td><td></td></tr>
<tr><td>27</td><td>TVE</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>26</td><td>CIM</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>25</td><td>MDMA</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>24</td><td>UHC</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>23</td><td>MAC</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>22</td><td>GPS</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>21</td><td>DMAC</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>20</td><td>SSI2</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>19</td><td>SSI1</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>18</td><td>UART3</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>17</td><td>UART2</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>16</td><td>UART1</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>15</td><td>UART0</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>14</td><td>SADC</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>13</td><td>KBC</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>12</td><td>MSC2</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>11</td><td>MSC1</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>10</td><td>OWI</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>9</td><td>TSSI</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>8</td><td>AIC</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>7</td><td>SCC</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>6</td><td>I2C1</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>5</td><td>I2C0</td><td>After reset period, the clock is stopped.</td></tr>
<tr><td>4</td><td>SSI0</td><td></td></tr>
<tr><td>3</td><td>MSC0</td><td></td></tr>
<tr><td>2</td><td>OTG</td><td></td></tr>
<tr><td>1</td><td>BCH</td><td></td></tr>
<tr><td>0</td><td>NEMC</td><td></td></tr>
</table>
</td><td>RW</td></tr>
</table>

13.3.2.7 Clock Gate Regiater1

The Clock Gate Register (CLKGR1) is a 32-bit read/write register that controls the CLOCK GATE function of peripherals. It is reset to 0x000003FF.

CLKGR1 0x10000028

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																						CLKGR									
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1

Bits	Name	Description	RW
31:8	Reserved		RW
8:0	CLKGR1	Clock gate Bits. Controls the clock supplies to some peripherals. If set, clock supplies to associated devices are stopped, and registers of the device cannot be accessed also.	

Bit	Module	Description
9	GPU	After reset period, the clock is stopped.
8	PCM	After reset period, the clock is stopped.
7	AHB1	After reset period, the clock is stopped.
6	CABAC	After reset period, the clock is stopped.
5	SRAM	After reset period, the clock is stopped.
4	DCT	After reset period, the clock is stopped.
3	ME	After reset period, the clock is stopped.
2	DBLK	After reset period, the clock is stopped.
1	MC	After reset period, the clock is stopped.
0	BDMA	After reset period, the clock is stopped.

13.3.2.8 Oscillator and Power Control Register (OPCR)

The Oscillator and Power Control Register is a 32-bit read/write register that specifies some special controls to oscillator and analog block. It is initialized to 0x00001570 by reset.

OPCR 0x10000024

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																O1ST								SPENDN	GPSEN	SPENDH	O1SE		ERCS	Reserved	USBM
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	1	1	0	0	0	0	0

Bits	Name	Description	RW
31:16	Reserved	Writes to these bits have no effect and always read as 0.	R
15:8	O1ST	EXCLK Oscillator Stabilize Time. This filed specifies the EXCLKoscillator stabilize time by unit of 16 RTCCLK periods (oscillator stable time O1ST × 16 / 32768) cycles. It is initialized to H'15.	RW
7	SPENDN	force OTG phy to enter suspend mode. 0: OTG phy has forced to entered SUSPEND mode 1: OTG phy hasn't forced to entered SUSPEND mode	R
6	GPSEN	0: Disable GPS module 1: Enable GPS module	RW
5	SPENDH	Force UHC phy to enter suspend mode. 0: UHC phy hasn't forced to entered SUSPEND mode 1: UHC phy has forced to entered SUSPEND mode	RW
4	O1SE	EXCLK Oscillator Sleep Mode Enable. This filed controls the state of the EXCLK oscillator in Sleep mode. 0: EXCLK oscillator is disabled in Sleep mode 1: EXCLK oscillator is enabled in Sleep mode	RW
3	PD	The fief controls the state P0 in Sleep mode. 0: The P0 not power down in Sleep mode 1: The P0 power down in Sleep mode	RW
2	ERCS	EXCLK/512 clock and RTCLK clock selection. 0: select EXCLK/512 division ration clock 1: select RTCLK clock the clock only output to CPM INTC SSI TCU etc.	RW
1:0	Reserved		R

13.3.3 Doze Mode

Firstly, software should set the DUTY bits of LCR. Then set DOZE bit of LCR to 1 to enter doze mode. When slot controller of PMC indicates that the CPU clock's time-slot has expired, CPU is halted but its register contents are retained. During doze mode, program can modify clock duty-cycle according to core resource requirement. Clock control is in increments of approximately 3% (1/31).

Doze is exited by software, interrupt, reset or SLEEP instruction.

13.3.4 IDLE Mode

In normal mode, when LPM bits in LCR are 0 and SLEEP instruction is executed, the processor enters idle mode. CPU is halted but its register contents are retained All critical application must be finished and peripherals must be configured to generate interrupts when they need CPU attention.

The procedure of entering sleep mode is shown blow:

1 Set LPM bits in LCR to 0.

2 Executes SLEEP instruction.
3 When current operation of CPU core has finished and CPU core is idle, CCLK supply to CPU core is stopped.

IDLE mode is exited by an interrupt (IRQ or on-chip devices) or a reset.

13.3.5 SLEEP Mode

In normal mode, when LPM bits in LCR is 1 and SLEEP instruction is executed, the processor enter SLEEP mode. CPU and on-chip devices are halted, except some wakeup-logic. PLL is shut off. Clock output from CKO pin is also stopped. SDRAM content is preserved by driving into self-refresh state. CPU registers and on-chip devices registers contents are retained.

Before enter SLEEP mode, software should ensure that all peripherals are not running. The procedure of entering SLEEP mode is shown blow:

1 Set LPM bit in LCR to 1.
2 Execute a SLEEP instruction.
3 When current access on system bus complete, the arbiter will not grant any following request. EMC will drive SDRAM from auto-refresh mode to self-refresh mode.
4 When system bus is idle state and SDRAM is self-refresh mode, internal clock supplies are stopped.
5 SLEEP mode can be exited by an interrupt (IRQ or on-chip devices), WDT reset or a poweron reset via the RESETP pin.

13.3.6 Power Down Mode

When PD_AHB1/PD_GPS bit in LCR is 1 , the processor enters shut down AHB1/GPS module power sequence.

When PD_AHB1S/PD_GPSS bit in LCR is 1, it indicates that the AHB1/GPS module has been shut off. The leakage current of AHB1/GPS is reduced almost to 0.

When enter sleep mode, when PD bits in OPCR is 1, the J1 power supply would be shut off. The leakage current of J1 is reduced almost to 0.

The procedure of entering Power Down mode is shown blow:

1 set proper values for PSWC0ST, PSWC1ST, PSWC2ST, PSWC3ST.
2 set PD_AHB1/PD_GPS bit in LCR to 1.
3 wait until PD_AHB1S/PD_GPSS = 1.
4 When need for supply power for AHB1/GPS, set PD_AHB1/PD_GPS in LCR to 0.
5 wait until PD_AHB1S/PD_GPSS = 0.
6 the hardware auto generate RESET signal to the module, the software must again config the

module as the same to POWER ON Reset.

13.4 Reset Control Module

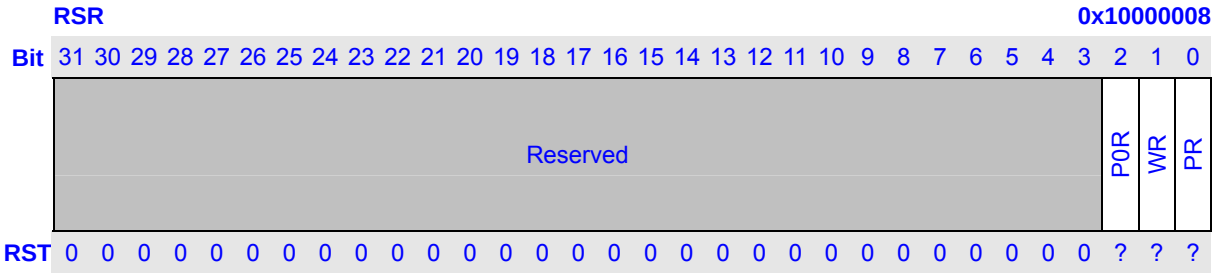
13.4.1 Register Description

All RCM register 32bit access address is physical address.

Name	description	RW	Initial Value	Address	Access Size
RSR	Reset Status Register	RW	0x????????	0x10000008	32

13.4.1.1 Reset Status Register (RSR)

The Reset Status Register (RSR) is a 32-bit read/write register which records last cause of reset. Each RSR bit is set by a different source of reset. Please refer to Reset Sequence Control for reset sources description.



Bits	Name	Description	RW
31:2	Reserved	Writes to these bits have no effect and always read as 0.	R
2	P0R	P0 power up Reset. It indicates that P0 has been shut down, now it has been power up. When P0 reset is detected, P0R is set and remains set until software clears it or another reset occurs. This bit can only be written with 0. Write with 1 will be ignored. 0: P0 reset has not occurred since the last time the software clears this bit 1: P0 reset has occurred since the last time the software clears this bit	RW
1	WR	WDT Reset. When a WDT reset is detected, WR is set and remains set until software clears it or another reset occurs. This bit can only be written with 0. Write with 1 will be ignored. 0: WDT reset has not occurred since the last time the software clears this bit 1: WDT reset has occurred since the last time the software clears this bit	RW
0	PR	Power On Reset. When a poweron reset via PRESET pin is detected, PR is set and remains set until software clears it or another reset occurs. This bit can only be written with 0. Write with 1 is ignored. 0: Power on reset has not occurred since the last time the software clears	RW

		this bit 1: Power on reset has occurred since the last time the software clears this bit	

13.4.2 Power On Reset

Power on reset is generated when PRESET pin is driven to low. Internal reset is asserted immediately. All pins return to their reset states. The Power on reset is extended to 40MS.

PRESET pin must be held low until power stabilizes and the EXCLK oscillator stabilize. CPU and peripherals are clocked by EXCLK oscillator output directly. PLL is reset to off state. All internal modules are initialized to their predefined reset states.

13.4.3 WDT Reset

WDT reset is generated when WDT overflow. Internal reset is asserted within two RTCCLK cycles. All pins return to their reset states.

Then WDT reset source is cleared because of internal reset. The internal reset is asserted for about 10 milliseconds. CPU and peripherals are clocked by EXCLK oscillator output directly. PLL is reset to off state.

14 Real Time Clock

14.1 Overview

The Real-Time Clock (RTC) unit can be operated in either chip main power is on or the main power is down but the RTC power is still on. In this case, the RTC power domain consumes only a few micro watts power.

The RTC contains a 32768Hz oscillator, the real time and alarm logic, and the power down and wakeup control logic.

14.1.1 Features

RTC module has following features:

- Embedded 32768Hz oscillator for 32k clock generation with an external 32k crystal
- RTCLK selectable from the oscillator or from the divided clock of EXCLK, so that 32k crystal can be absent if the hibernating mode is not needed
- 32-bits second counter
- Programmable and adjustable counter to generate accurate 1 Hz clock
- Alarm interrupt, 1Hz interrupt
- Stand alone power supply, work in hibernating mode
- Power down controller
- Alarm wakeup
- External pin wakeup with up to 2s glitch filter

14.1.2 Signal Descriptions

RTC has 5 signal IO pins and 1 power pin. They are listed and described in.

Pin Names	Pin Loc	IO	IO Cell Char.	Pin Description	Power
RTCLK		AI	32768Hz	RTCLK: 32768 clock input or OSC input	VDD_{RTC}
RTCLKO		AO		RTCLKO: OSC output	VDD_{RTC}
PWRON		AO	~2mA, Open-Draw	PWRON: Power on/off control of main power	VDD_{RTC}
WKUP		AI	Schmitt	WKUP: Wake signal after main power down	VDD_{RTC}
PPRST_		AI	Schmitt	PPRST_: RTC power on reset and RESET-KEY reset input	VDD_{RTC}
VDDRTC		P		VDDRTC: 3.3V power for RTC and hibernating mode controlling that never power down	-

RTCLK/RTCLKO pins. We have an embedded oscillator for 32768Hz crystal. These two pins are the crystal XTALI and XTALO connection pins. If an input clock is used instead, please input it to

RTCLKO pin.

If do not use any clock, hibernate mode will be NOT available any more, and the time will lose if power down.

PWRON pin: this pin is used to control the main power on/off. Output low voltage means off and high voltage means on.

WKUP pin: hibernating mode wakeup input.

PPRST_ pin: This pin should be set to low voltage only in two cases.

- When RTC power is turned on (so that whole chip is power on)
- A RESET-KEY is pressed

14.2 Register Description

Table 14-1 Registers for real time clock

Name	Description	RW	Reset Value	Address	Access Size
RTCCR	RTC Control Register	RW	0x00000081[1][2]	0x10003000	32
RTCSR	RTC Second Register	RW	0x????????	0x10003004	32
RTCSAR	RTC Second Alarm Register	RW	0x????????	0x10003008	32
RTCGR	RTC Regulator Register	RW	0x0???????	0x1000300C	32

NOTES:

1 Unless otherwise stated, the reset value is for PPRST_ and Hibernating wakeup reset. WDT reset doesn't change the value.

2 The reset value can be either of 0x00000081, 0x00000091, 0x00000089, 0x00000099.

Table 14-2 Registers for hibernating mode

Name	Description	RW	Reset Value	Address	Access Size
HCR	Hibernate Control Register	RW	0x00000000[1]	0x10003020	32
HWFCR	Wakeup filter counter Register in Hibernate mode	RW	0x0000???0	0x10003024	32
HRCR	Hibernate reset counter Register in Hibernate mode	RW	0x00000??0	0x10003028	32
HWCR	Wakeup control Register in Hibernate mode	RW	0x00000008[1][2]	0x1000302C	32
HWRSR	Wakeup Status Register in Hibernate mode	RW	0x00000000[1]	0x10003030	32
HSPR	Scratch pattern register	RW	0x????????	0x10003034	32
WENR	Write enable pattern register	RW	0x00000000	0x1000303C	32

NOTES:

1 Unless otherwise stated, the reset value is for PPRST_ and Hibernating wakeup reset. WDT reset doesn't change the value.

2 The reset value can be either 0x00000008 or 0x0000000C.

All these registers, include those for real time clock and for hibernating mode control, except otherwise stated, are implemented in RTCLK clock domain. When write to these registers, it needs about 1 ~ 2 RTCLK cycles to actually change the register's value and needs another RTCLK cycle to allow the next write access. A bit RTCCR.WRDY is used to indicate it. When RCR.WRDY is 1, it means the previous write is finished, a right value can be read from the target register, and a new write access can be issued. So before any write access, please make sure RCR.WRDY = 1.

14.2.1 RTC Control Register (RTCCR)

RTCCR contains bits to configure the real time clock features. Unless otherwise stated, the reset value is for PPRST_ and Hibernating wakeup reset. WDT reset doesn't change the value.

RTCCR 0x10003000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																								WRDY	1HZ	1HZIE	AF	AIE	AE	SELEXC	RTCE
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1[1]	0	0[1]	?	?	0	0[1]	1

NOTES:

1 These bits are reset in all resets: PPRST_ input pin reset, hibernating reset and WDT reset.

Bits	Name	Description	RW
31:7	Reserved	Writes to these bits have no effect and always read as 0.	R
7	WRDY	Write ready flag. It is 0 when a write is currently processing and the value has not been written to the writing target register. No write to any RTC registers can be issued in this case, or the result is undefined. The read value from the target register is also undefined. The reading is meaningful and another write can be issued when it is 1. Please reference to descriptions in14.2._ for some more details. This bit is read only and write to it is ignored.	R
6	1HZ	1Hz flag. This bit is set by hardware once every 1 second through the 1Hz pulse if the real time clock is enabled (RTCCR.RTCE = 1). This bit can be cleared by software. Write 1 to this bit is ignored.	RW
5	1HZIE	1Hz interrupt enable. Writing to this bit takes effect immediately without delay. 1HZIE: 0 – 1Hz interrupt is disabled. 1 – 1Hz interrupt is enabled. RTC issues interrupt when 1HZ bit is set.	RW
4	AF	Alarm flag. This bit is set by hardware when alarm match (RTCSR = RTCSAR) is found and alarm is enabled (RTCCR.AE = 1) and the real time clock is enabled (RTCCR.RTCE = 1). This bit can be cleared by software. Write 1 to this bit is ignored. Writing to this bit takes effect immediately.	RW
3	AIE	Alarm interrupt enable. AIE: 0 – Alarm interrupt is disabled. 1 – Alarm interrupt is enabled. RTC issues interrupt when AF is set.	RW

<table>
<tr><td>2</td><td>AE</td><td>Alarm enable.<table><tr><th>AE</th><th>Description</th></tr><tr><td>0</td><td>Alarm function is disabled.</td></tr><tr><td>1</td><td>Alarm function is enabled.</td></tr></table></td><td>RW</td></tr>
<tr><td>1</td><td>SELEXC</td><td>The divided EXCLK is selected as RTCLK in rtc-hiber module.<table><tr><th>SELEXC</th><th>Description</th></tr><tr><td>0</td><td>OSC32K or RTCLK input clock is selected as RTCLK in rtc-hiber module.</td></tr><tr><td>1</td><td>The divided EXCLK is selected as RTCLK in rtc-hiber module.</td></tr></table>NOTE: If do not use any 32Khz clock (either input clock or using crystal), hibernate mode will be NOT available any more, and the time will lose if power down.
CPM.OPCR.ERCS must be 0, when using SELEXC = 1.
When the main chip power down, SELEXC will be 0 in internal circuit, in this time, RTCLK will use OSC32K clock.</td><td>RW</td></tr>
<tr><td>0</td><td>RTCE</td><td>Real time clock enable.<table><tr><th>RTCE</th><th>Description</th></tr><tr><td>0</td><td>Real time clock function is disabled.</td></tr><tr><td>1</td><td>Real time clock function is enabled.</td></tr></table></td><td>RW</td></tr>
</table>

14.2.2 RTC Second Register (RTCSR)

RTCSR is a 32-bit width second counter. It can be read and write by software. It is increased by 1 at every 1Hz pulse if the real time clock is enabled (RTCCR.RTCE = 1). When read, it should be read continued more than once and take the value if the adjacent results are the same. RTCSR is not initialized by any reset.

RTCSR **0x10003004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RTCSR																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

14.2.3 RTC Second Alarm Register (RTCSAR)

RTCSAR serves as a second alarm register. Alarm flag (RTCCR.AF) is set to 1 when the RTCSR equals the RTCSAR in the condition of alarm is enabled (RTCCR.AE = 1) and the real time clock is enabled (RTCCR.RTCE = 1). RTCSAR can be read and write by software and is not initialized by any reset.

RTCSAR **0x10003008**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RTCSAR																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

14.2.4 RTC Regulator Register (RTCGR)

RTCGR is serves as the real time clock regulator, which is used to adjust the interval of the 1Hz pulse.

RTCGR **0x1000300C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LOCK	Reserved					ADJC										NC1HZ															
RST	0[1]	0	0	0	0	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

NOTES:

1 This bit is reset in all resets: PPRST_ input pin reset, hibernating reset and WDT reset.

Bits	Name	Description	RW
31	LOCK	Lock bit. This bit is used to safeguard the validity of the data written into the RTCGR register. Once it is set, write to RTCGR is ignored. This bit can only be set by software and cleared by (any type of) resets. LOCK: 0 – Write to RTCGR is allowed. LOCK: 1 – Write to RTCGR is forbidden.	RW
30:26	Reserved	Writes to these bits have no effect and always read as 0.	R
25:16	ADJC	This field specifies how many times it needs to add one 32kHz cycle for the 1Hz pulse interval in every 1024 1Hz pulses. In other word, among every 1024 1Hz pulses, ADJC number of them are trigged in every (NC1HZ + 2) 32kHz clock cycles, (1024 – ADJC) number of them are trigged in every (NC1HZ + 1) 32kHz clock cycles.	RW
15:0	NC1HZ	This field specifies the number plus 1 of the working 32kHz clock cycles are contained in the 1Hz pulse interval. In other word, 1Hz pulse is trigged every (NC1HZ + 1) 32kHz clock cycles, if RTCGR.ADJC = 0.	RW

14.2.5 Hibernate Control Register (HCR)

HCR contains the bit to control the main chip power on/off.

HCR 0x10003020

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																															PD
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:1	Reserved	Writes to these bits have no effect and always read as 0.	R
0	PD	Power down or power on bit. Besides writing by CPU, this bit will be set to 1 if an unknown reason main power supply off is detected. This bit controls the PWRON pin level. When co-working with some external components, this bit is used for power management of this chip. It is supposed when 1 is written to this bit, the main power supply of the chip, except RTC power, will be shut down immediately. After this bit is set to 1, all registers in RTC module, except RTCCR.1HZ and RTCCR.1HZIE, cannot be changed by write access. This bit is cleared by reset pin reset and hibernating reset. The later one is asserted by wakeup procedure. PD: 0 – PWRON: VDDRTC – No power down, keep power on. PD: 1 – PWRON: 0 V – Power down enable, turn power off.	RW

14.2.6 HIBERNATE mode Wakeup Filter Counter Register (HWFCR)

The HIBERNATE mode Wakeup Filter Counter Register (HWFCR) is a 32-bit read/write register .It filters the glitch generated by a dedicated wakeup pin. The HRCR is initialized by PPRST_ and WDT reset.

HWFCR **0x10003024**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																HWFCR											Reserved					
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	?	?	?	?	?	?	?	?	?	?	?	0	0	0	0	0

Bits	Name	Description	RW
31:16	Reserved	Writes to these bits have no effect and always read as 0.	R
15:5	HWFCR	Wakeup pin effective minimum time in number of 32 RTCLK cycles, used as glitch filter logic. Maximum of 2 seconds if the RTCLK is 32768Hz If this value is configured to 0, and the pin keeps low longer than 15 RTCLK periods, it wakes up RTC from Hibernate.	RW
4:0	Reserved	Writes to these bits have no effect and always read as 0.	R

14.2.7 Hibernate Reset Counter Register (HRCR)

The Hibernate Reset Counter Register is a 32-bit read/write register that specifies hibernate reset assertion time. The HRCR is initialized by PPRST_ and WDT reset.

HRCR **0x10003028**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																				HRCR							Reserved					
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	?	?	?	?	?	?	?	0	0	0	0	0

Bits	Name	Description	RW
31:12	Reserved	Writes to these bits have no effect and always read as 0.	R
11:5	HRCR	HIBERNATE Reset waiting time. Number of 32 RTCLK cycles. Maximum 125 ms if the RTCLK is 32768Hz. If this value is configured to 0, it will generate 31 RTCLK HIBERNATE Reset.	RW
4:0	Reserved	Writes to these bits have no effect and always read as 0.	R

14.2.8 HIBERNATE Wakeup Control Register (HWCR)

The HIBERNATE Wakeup Control Register is a 32-bit read/write register that controls real time clock alarm wake up enable. The reset value is for PPRST_ and Hibernating wakeup reset. WDT reset doesn't change the value.

HWCR **0x1000302C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																												EPDET	WKUPVL	Reserved	EALM
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	?	0	0

Bits	Name	Description	RW
31:4	Reserved	Writes to these bits have no effect and always read as 0.	R
3	EPDET	Power detect enable. 0: disable 1: enable (default)	RW
2	WKUPVL	RTC Alarm wakeup pin valid level. 0: Low effective. WKUP pin in low level to wake up chip 1: High effective. WKUP pin in high level to wake up chip	RW
1	Reserved	Writes to these bits have no effect and always read as 0.	RW
0	EALM	RTC Alarm wakeup enable. 0: disable 1: enable	RW

14.2.9 HIBERNATE Wakeup Status Register (HWRSR)

The HIBERNATE Wakeup Status Register is a 32-bit read/write register that reflects wakeup status bits.

HWRSR **0x10003030**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																							APD	Reserved		HR	PPR	Reserved		PIN	ALM
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	?	0	0	1[2]	1[1]	0	0	?	?

NOTES:

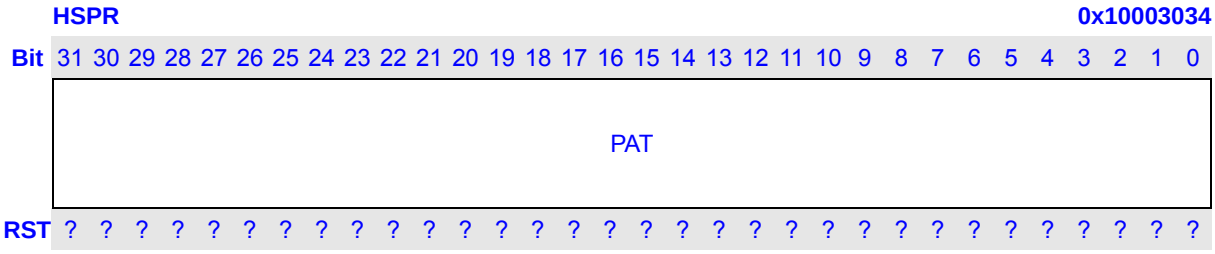
1 This reset value only for PPRST_. It is undefined in case of other resets.

2 This reset value only for HRST_. It is undefined in case of other resets.

<table>
<tr><th>Bits</th><th>Name</th><th>Description</th><th>RW</th></tr>
<tr><td>31:9</td><td>Reserved</td><td>Writes to these bits have no effect and always read as 0.</td><td>R</td></tr>
<tr><td>8</td><td>APD</td><td>Accident power down. When the software has not set to HIBERNATE state, the core power is down, then an accident power down is detected. APD is set and remains set until software clears it. This bit can only be written with 0. Write with 1 is ignored.
<table>
<tr><th>HR</th><th>Description</th></tr>
<tr><td>0</td><td>Accident power down has not occurred since the last time the software clears this bit.</td></tr>
<tr><td>1</td><td>Accident power down has occurred since the last time the software clears this bit.</td></tr>
</table></td><td>RW</td></tr>
<tr><td>7:6</td><td>Reserved</td><td>Writes to these bits have no effect and always read as 0.</td><td>R</td></tr>
<tr><td>5</td><td>HR</td><td>Hibernate Reset. When a Hibernate reset detected, HR is set and remains set until software clears it or another reset occurs. This bit can only be written with 0. Write with 1 is ignored.
<table>
<tr><th>HR</th><th>Description</th></tr>
<tr><td>0</td><td>Hibernate reset has not occurred since the last time the software clears this bit.</td></tr>
<tr><td>1</td><td>Hibernate reset has occurred since the last time the software clears this bit.</td></tr>
</table></td><td>RW</td></tr>
<tr><td>4</td><td>PPR</td><td>PAD PIN Reset. When a PPRST_ is detected, PPR is set and remains set until software clears it or another reset occurs. This bit can only be written with 0. Write with 1 is ignored.
<table>
<tr><th>PPR</th><th>Description</th></tr>
<tr><td>0</td><td>PPRST_ reset has not occurred since last time the software clears this bit.</td></tr>
<tr><td>1</td><td>PPRST_ reset has occurred since last time the software clears this bit.</td></tr>
</table></td><td>RW</td></tr>
<tr><td>3:2</td><td>Reserved</td><td>Writes to these bits have no effect and always read as 0.</td><td>R</td></tr>
<tr><td>1</td><td>PIN</td><td>Wakeup Pin Status bit. The bit is cleared when chip enters hibernating mode. It is set when exit the hibernating mode by wakeup pin. This bit can only be written with 0. Write with 1 is ignored.</td><td>RW</td></tr>
<tr><td>0</td><td>ALM</td><td>RTC Alarm Status bit. The bit is cleared when chip enters hibernating mode. It is set when exit the hibernating mode by alarm. This bit can only be written with 0. Write with 1 is ignored.</td><td>RW</td></tr>
</table>

14.2.10 Hibernate Scratch Pattern Register (HSPR)

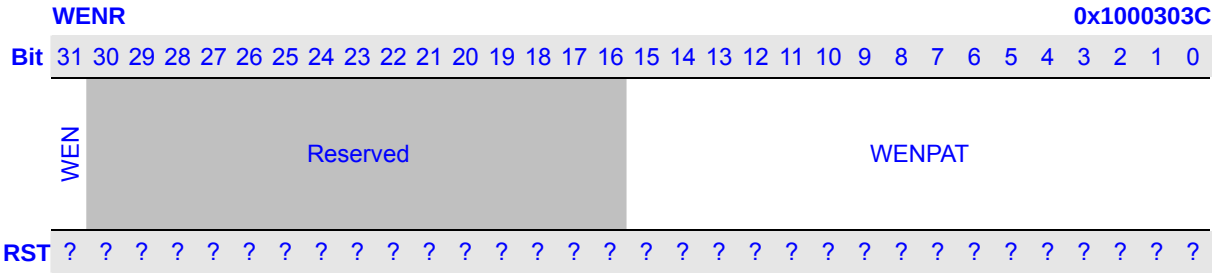
This is a scratch register used to hold a pattern. The software can check the pattern is kept to know whether RTC power has ever been down and whether it is needed to setup the real time clock.



Bits	Name	Description	RW
31:0	PAT	The pattern.	RW

14.2.11 Write Enable Pattern Register (WENR)

This is a scratch register used to hold a pattern. The software can check the pattern is kept to know whether RTC power has ever been down and whether it is needed to setup the real time clock.



Bits	Name	Description	RW
31	WEN	The write enable flag. If the WENPAT is 0xA55A then this bit will be 1. When the WEN changes to 1, the RTCCR, RTCSR, RTCSAR, RTCGR, HCR, HWFCR, HRCR, HWCR, HWRSR, HSPR registers could be changed. But RTCCR.SELEXC, RTCCR.HZIE, RTCCR.WRDY may change in any time. This bit is read only and write to it is ignored. There is an exception, when system does NOT have RTC 32Khz crystal. MUST write 1 to RTCCR.SELEXC before write to any value to any other registers. **WEN** / **Description** 0 — Other RTC registers is locked, write these registers will be ignored. 1 — Other RTC registers can be changed.	R
30:16	Reserved	Writes to these bits have no effect and always read as 0.	R

15:0	WENPAT	The write enable pattern. Before writing any value to RTCCR, RTCSR, RTCSAR, RTCGR, HCR, HWFCR, HRCR, HWCR, HWRSR, HSPR registers, write 0xA55A to WENPAT to set these register writable. If this value is ok, WEN will change to 1. But RTCCR.SELEXC and RTCCR.HZIE are writable in any time. These bits are write-only, always read as 0.	W

14.3 Time Regulation

Because of the inherent inaccuracy of crystal and other variables, the time counter may be inaccurate. This requires a slight adjustment. The application processor, through the RTCGR, lets you adjust the 1Hz time base to an error of less than 1ppm. Such that if the Hz clock were set to be 1Hz, there would be an error of less than 5 seconds per month.

To determine the value programmed into the RTCGR, you must first measure the output frequency at the oscillator multiplex (approximately 32 kHz) using an accurate time base, such as a frequency counter. This clock is externally visible by selecting the alternate function of GPIO[?]
To gain access to the clock, program this pin as an output and then switch to the alternate function. To trim the clock, divide the output of the oscillator by an integer value and fractional adjust it by periodically deleting clocks from the stream driving this integer divider.

After the true frequency of the oscillator is known, it must be split into integer and fractional portions. The integer portion of the value (minus one) is loaded into the NC1HZ field of the RTCGR.

The fractional part of the adjustment is done by periodically deleting clocks from the clock stream driving the Hz divider. The trim interval period is hardwired to be 1024 1Hz clock cycles (approximately 17 minutes). The number of clocks (represented by ADJC field of RTCGR) are deleted from the input clock stream per trim interval. If ADJC is programmed to be zero, then no trim operations occur and the RTC is clocked with the raw 32 kHz clock. The relationship between the Hz clock frequency and the nominal 32 kHz clock (f1 and f32K, respectively) is shown in the following equation.

$$f1 = \frac{2^{10} \times (NC1HZ + 1)}{2^{10} \times (NC1HZ + 1) + ADJC} \times \frac{f32k}{NC1HZ + 1}$$

f1 = actual frequency of 1Hz clock
f32k = frequency of either 32.768KHz crystal output or 3.6864MHz crystal output further divided down to 32.914KHz

14.3.1 HIBERNATE Mode

First make sure RTCCR.SELEXC is 0.
When Software writes 1 to PD bit of HCR, the system at once enters HIBERNATE mode. The powers of CORE and IO are disconnected by PWRON pin, no power consumption to core and IO. When a wakeup event occurs, the core enters through a hibernate reset. Only CPM wake up logic and RTC is operating in HIBERNATE mode.

14.3.1.1 Procedure to Enter HIBERNATE mode

Before enter HIBERNATE mode, software must complete following steps:

1 Finish the current operation and preserve all data to flash.
2 Configure the wake-up sources properly by configure HWCSR.
3 Set HIBERNATE MODE. (Set PD bit in HCR to 1).

14.3.1.2 Procedure to Wake-up from HIBERNATE mode

1 The internal hibernate reset signal will be asserted if one of the wake-up sources is issued.
2 Check RSR to determine what caused the reset.
3 Check PIN/ALM bits of HWCSR in order to know whether or not the power-up is caused by which wake-up from HIBERNATE mode.
4 Configure the SDRAM memory controller.
5 Recover the data from flash.

14.4 Clock select

There could be two clock input to RTC internal clock called rtclk. One is OSC32k clock; the other is EXCLK/512.

The software MUST make sure the RTC run in valid clock configuration.

Table 14-3 Clock select registers

RTCCR.SELEXC	CPM.ERCS	Description	Valid
0	0	RTC use OSC32K clock	OK
0	1		OK
1	0	RTC use EXCLK/512 clock	OK
1	1	RTC will lost clock (Not Valid)	NO

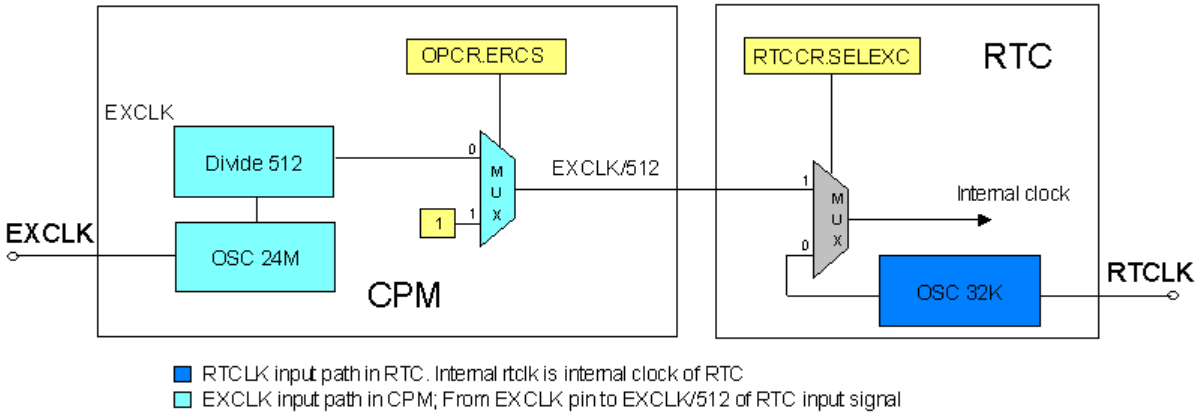


Figure 14-1 RTC clock selection path

Changing RTCLK sequence:

1 There are both 32KHz crystal and 24Mhz EXCLK crystal connected, so RTCLK input path has 32Khz clock.

In this case, there is no need to change internal clock, so do NOT change SELEXC all the time.

2 There is no 32KHz crystal connected but only 24Mhz EXCLK crystal connected, so RTCLK input path has no clock.

In this case, should flow the sequence below to change internal clock:

a Set OPCR.ERCS of CPM to 1; close EXCLK/512 to RTC.

b Set CLKGR.RTC of CPM to 1; close PCLK to RTC.

c Set RTCCR.SELEXC to 1; change internal clock to EXCLK/512.

d Wait two clock period of clock.

e Clear OPCR.ERCS of CPM to 0; open EXCLK/512 to RTC.

f Clear CLKGR.RTC of CPM to 0; open PCLK to RTC.

g Configure all RTC registers but RTCCR.SELEXC.

h Check RTCCR.SELEXC == 1.

i IF YES, finish this sequence; IF NO, do step (1) again.

NOTES:

1 If using HIBERNATE mode, MUST have both 32KHz crystal (or input 32Khz clock) and 24Mhz EXCLK crystal connected, or RTC time will be insignificant.

15 Interrupt Controller

15.1 Overview

This chapter describes the interrupt controller included in the XBurst Processor, explains its modes of operation, and defines its registers. The interrupt controller controls the interrupt sources available to the processor and contains the location of the interrupt source to allow software to determine source of all interrupts. It also determines whether the interrupts cause an IRQ to occur and masks the interrupts.

Features:

- Total 64 interrupt sources
- Each interrupt source can be independently enabled
- Priority mechanism to indicate highest priority interrupt
- All the registers are accessed by CPU
- Unmasked interrupts can wake up the chip in sleep mode

15.2 Register Description

Table 15-1 lists the registers of Interrupt Controller. All of these registers are 32bit, and each bit of the register represents or controls one interrupt source that list in Table 15-1.

All INTC register 32bit access address is physical address.

Table 15-1 INTC Register

Name	Description	RW	Reset Value	Address	Access Size
ICSR0	Interrupt controller Source Register	R	0x00000000	0x10001000	32
ICMR0	Interrupt controller Mask Register	RW	0xFFFFFFFF	0x10001004	32
ICMSR0	Interrupt controller Mask Set Register	W	0x????????	0x10001008	32
ICMCR0	Interrupt controller Mask Clear Register	W	0x????????	0x1000100C	32
ICPR0	Interrupt controller Pending Register	R	0x00000000	0x10001010	32
ICSR1	Interrupt controller Source Register	R	0x00000000	0x10001020	32
ICMR1	Interrupt controller Mask Register	RW	0xFFFFFFFF	0x10001024	32
ICMSR1	Interrupt controller Mask Set Register	W	0x????????	0x10001028	32
ICMCR1	Interrupt controller Mask Clear Register	W	0x????????	0x1000102C	32
ICPR1	Interrupt controller Pending Register	R	0x00000000	0x10001030	32

15.2.1 Interrupt Controller Source Register (ICSR0)

This register contains all the interrupts' status. A "1" indicates that the corresponding interrupt is pending . A "0" indicates that the interrupt is not pending now. The register is read only.

ICSR0 **0x10001000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LCD	CIM	IPU	GPS	TCU0	TCU1	TCU2	DMA0	DMA1	MDMA	OTG	UHC	ETH	SADC	GPIO0	GPIO1	GPIO2	GPIO3	GPIO4	GPIO5	KBC	BDMA	TSSI	SSI0	SSI1	GPU	UART0	UART1	UART2	UART3	I2C0	I2C1
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits Of ICSR0	Description
0	The corresponding interrupt source is not pending.
1	The corresponding interrupt source is pending.

15.2.2 Interrupt Controller Source Register (ICSR1)

This register contains all the interrupts' status. A "1" indicates that the corresponding interrupt is pending . A "0" indicates that the interrupt is not pending now. The register is read only.

ICSR1 **0x10001020**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																					HARB2	HARB0	PCM	BCH	SCC	MSC0	MSC1	MSC2	AIC	OWI	RTC
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits Of ICSR1	Description
0	The corresponding interrupt source is not pending.
1	The corresponding interrupt source is pending.

15.2.3 Interrupt Controller Mask Register (ICMR0)

This register is used to mask the interrupt input sources and defines which active sources are allowed to generate interrupt requests to the processor. Its value can be changed either by writing ICMSR and ICMCR or by writing itself. The masked interrupts are invisible to the processor.

ICMR0 **0x10001004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LCD	CIM	IPU	GPS	TCU0	TCU1	TCU2	DMA0	DMA1	MDMA	OTG	UHC	ETH	SADC	GPIO0	GPIO1	GPIO2	GPIO3	GPIO4	GPIO5	KBC	BDMA	TSSI	SSI0	SSI1	GPU	UART0	UART1	UART2	UART3	I2C0	I2C1
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits Of ICMR0	Description
0	The corresponding interrupt is not masked.
1	The corresponding interrupt is masked.

15.2.4 Interrupt Controller Mask Register (ICMR1)

This register is used to mask the interrupt input sources and defines which active sources are allowed to generate interrupt requests to the processor. Its value can be changed either by writing ICMSR and ICMCR or by writing itself. The masked interrupts are invisible to the processor.

ICMR1 **0x10001024**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																					HARB2	HARB0	PCM	BCH	SCC	MSC0	MSC1	MSC2	AIC	OWI	RTC
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits Of ICMR1	Description
0	The corresponding interrupt is not masked.
1	The corresponding interrupt is masked.

15.2.5 Interrupt Controller Mask Set Register (ICMSR0)

This register is used to set bits in the interrupt mask register. This register is write only.

ICMSR0 **0x10001008**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LCD	CIM	IPU	GPS	TCU0	TCU1	TCU2	DMA0	DMA1	MDMA	OTG	UHC	ETH	SADC	GPIO0	GPIO1	GPIO2	GPIO3	GPIO4	GPIO5	KBC	BDMA	TSSI	SSI0	SSI1	GPU	UART0	UART1	UART2	UART3	I2C0	I2C1
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits Of ICMSR0	Description
0	Ignore.
1	Will set the corresponding interrupt mask bit.

15.2.6 Interrupt Controller Mask Set Register (ICMSR1)

This register is used to set bits in the interrupt mask register. This register is write only.

ICMSR1 **0x10001028**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																					HARB2	HARB0	PCM	BCH	SCC	MSC0	MSC1	MSC2	AIC	OWI	RTC
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits Of ICMSR1	Description
0	Ignore.
1	Will set the corresponding interrupt mask bit.

15.2.7 Interrupt Controller Mask Clear Register (ICMCR0)

This register is used to clear bits in the interrupt mask register. This register is write only.

ICMCR0 **0x1000100C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LCD	CIM	IPU	GPS	TCU0	TCU1	TCU2	DMA0	DMA1	MDMA	OTG	UHC	ETH	SADC	GPIO0	GPIO1	GPIO2	GPIO3	GPIO4	GPIO5	KBC	BDMA	TSSI	SSI0	SSI1	GPU	UART0	UART1	UART2	UART3	I2C0	I2C1
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits Of ICMCR0	Description
0	Ignore.
1	Will clear the corresponding interrupt mask bit.

15.2.8 Interrupt Controller Mask Clear Register (ICMCR1)

This register is used to clear bits in the interrupt mask register. This register is write only.

ICMCR1 **0x1000102C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																					HARB2	HARB0	PCM	BCH	SCC	MSC0	MSC1	MSC2	AIC	OWI	RTC
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits Of ICMCR1	Description
0	Ignore.
1	Will clear the corresponding interrupt mask bit.

15.2.9 Interrupt Controller Pending Register (ICPR0)

This register contains the status of the interrupt sources after masking. This register is read only.

ICPR0 **0x10001010**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LCD	CIM	IPU	GPS	TCU0	TCU1	TCU2	DMA0	DMA1	MDMA	OTG	UHC	ETH	SADC	GPIO0	GPIO1	GPIO2	GPIO3	GPIO4	GPIO5	KBC	BDMA	TSSI	SSI0	SSI1	GPU	UART0	UART1	UART2	UART3	I2C0	I2C1
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits Of ICPR0	Description
0	The corresponding interrupt is not active or is masked.
1	The corresponding interrupt is active and is not masked to the processor.

NOTES:

1 Reserved bits in ICMR0, ICMSR0 and ICMCR0 are normal bits to be written into and read out.

2 Reserved bits in ICSR and ICPR are read-only and always 0.

15.2.10 Interrupt Controller Pending Register (ICPR1)

This register contains the status of the interrupt sources after masking. This register is read only.

ICPR1 **0x10001030**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																					HARB2	HARB0	PCM	BCH	SCC	MSC0	MSC1	MSC2	AIC	OWI	RTC
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits Of ICPR1	Description
0	The corresponding interrupt is not active or is masked.
1	The corresponding interrupt is active and is not masked to the processor.

NOTES:

1 Reserved bits in ICMR1, ICMSR1 and ICMCR1 are normal bits to be written into and read out.

2 Reserved bits in ICSR1 and ICPR1 are read-only and always 0.

15.3 Software Considerations

The interrupt controller is reflecting the status of interrupts sources in the peripheral .

Software should perform the task - determine the interrupt source from in ICPRx. In this chip, pending interrupts have two levels in structure. Interrupting module in the system that contains more than one interrupt sources need software to determine how to service it by reading interrupt status registers within it.

In the interrupt handler, the serviced interrupt source needs to be cleared in the interrupting device. In order to make certain the cleared source request status has been reflected at the corresponding ICPRx bit, software should wait enough time before exiting interrupt state.

The procedure is described following:

1 Interrupt generated.
2 CPU query interrupt sources, saves the current environment and then goes to interrupt common service routine.
3 Get ICPRx.
4 Find the highest priority interrupt and vector it. (The software decides which one has the highest priority)
5 Mask the chosen interrupt by writing the register ICMSRx.
6 Enable the system interrupt to allow the interrupt nesting.(software decided)
7 Execute the interrupt handler and unmask it by writing the register ICMCRx when exit the handler.
8 CPU restores the saved environment and exits the interrupt state.

16 Timer/Counter Unit

16.1 Overview

The TCU (Timer/Counter with PWM output) contains 6 channels of 16-bit programmable timers (timers 0 to 5). They can be used as Timer or PWM.

TCU has the following features:

- There are two modes of TCU for the eight channels
  - TCU1: Channel 0, 3,4, 5, 6,and 7
  - TCU2: Channel 1,2
- Six independent channels, each consisting of
  - Counter
  - Data register (FULL and HALF)
  - Control register
- Independent clock for each counter, selectable by software
  - PCLK, EXTAL and RTCCLK can be used as the clock for counter
  - The division ratio of the clock can be set to 1, 4, 16, 64, 256 and 1024 by software
- FULL interrupt and HALF interrupt can be generated for each channel using the compare data registers
  - Timer 0-7 can be used as PWM (Set the initial signal level)
  - Timer 0,3-7 can be used as a counter to count external signal (like trackball)
  - Timer 5 has separated interrupt
  - Timer 0-4 and timer 6-7 have one interrupt in common
  - OST uses interrupt 0, Timer 0 uses interrupt 1, and Timer 1-7 uses interrupt 2
- The difference between TCU1 and TCU2
  - TCU1: It cannot work in sleep mode, but operated easily
  - TCU2: It can work in sleep mode, but operated more complicated than TCU1

16.2 Pin Description

Table 16-1 PWM Pins Description

Name	I/O	Description
PWM [7:0]	Output	PWM channel output signals.

16.3 Register Description

In this section, we will describe the registers in timer. Following table lists all the registers definition. All timer register's 32bit address is physical address. And detailed function of each register will be described below.

Name	Description	RW	Reset Value	Address	Access Size
TSTR	Timer Status Register	R	0x00000000	0x100020F0	32
TSTSR	Timer Status Set Register	W	0x????????	0x100020F4	32
TSTCR	Timer Status Clear Register	W	0x????????	0x100020F8	32
TSR	Timer STOP Register	R	0x00000000	0x1000201C	32
TSSR	Timer STOP Set Register	W	0x00000000	0x1000202C	32
TSCR	Timer STOP Clear Register	W	0x0000	0x1000203C	32
TER	Timer Counter Enable Register	R	0x0000	0x10002010	16
TESR	Timer Counter Enable Set Register	W	0x????	0x10002014	16
TECR	Timer Counter Enable Clear Register	W	0x????	0x10002018	16
TFR	Timer Flag Register	R	0x003F003F	0x10002020	32
TFSR	Timer Flag Set Register	W	0x????????	0x10002024	32
TFCR	Timer Flag Clear Register	W	0x????????	0x10002028	32
TMR	Timer Mask Register	R	0x00000000	0x10002030	32
TMSR	Timer Mask Set Register	W	0x????????	0x10002034	32
TMCR	Timer Mask Clear Register	W	0x????????	0x10002038	32
TDFR0	Timer Data FULL Register 0	RW	0x????	0x10002040	16
TDHR0	Timer Data HALF Register 0	RW	0x????	0x10002044	16
TCNT0	Timer Counter 0	RW	0x????	0x10002048	16
TCSR0	Timer Control Register 0	RW	0x0000	0x1000204C	16
TDFR1	Timer Data FULL Register 1	RW	0x????	0x10002050	16
TDHR1	Timer Data HALF Register 1	RW	0x????	0x10002054	16
TCNT1	Timer Counter 1	RW	0x????	0x10002058	16
TCSR1	Timer Control Register 1	RW	0x0000	0x1000205C	16
TDFR2	Timer Data FULL Register 2	RW	0x????	0x10002060	16
TDHR2	Timer Data HALF Register 2	RW	0x????	0x10002064	16
TCNT2	Timer Counter 2	RW	0x????	0x10002068	16
TCSR2	Timer Control Register 2	RW	0x0000	0x1000206C	16
TDFR3	Timer Data FULL Register 3	RW	0x????	0x10002070	16
TDHR3	Timer Data HALF Register 3	RW	0x????	0x10002074	16
TCNT3	Timer Counter 3	RW	0x????	0x10002078	16
TCSR3	Timer Control Register 3	RW	0x0000	0x1000207C	16
TDFR4	Timer Data FULL Register 4	RW	0x????	0x10002080	16
TDHR4	Timer Data HALF Register 4	RW	0x????	0x10002084	16
TCNT4	Timer Counter 4	RW	0x????	0x10002088	16

TCSR4	Timer Control Register 4	RW	0x0000	0x1000208C	16
TDFR5	Timer Data FULL Register 5	RW	0x????	0x10002090	16
TDHR5	Timer Data HALF Register 5	RW	0x????	0x10002094	16
TCNT5	Timer Counter 5	RW	0x????	0x10002098	16
TCSR5	Timer Control Register 5	RW	0x0000	0x1000209C	16
TDFR6	Timer Data FULL Register 6	RW	0x????	0x100020A0	16
TDHR6	Timer Data HALF Register 6	RW	0x????	0x100020A4	16
TCNT6	Timer Counter 6	RW	0x????	0x100020A8	16
TCSR6	Timer Control Register 6	RW	0x0000	0x100020AC	16
TDFR7	Timer Data FULL Register 7	RW	0x????	0x100020B0	16
TDHR7	Timer Data HALF Register 7	RW	0x????	0x100020B4	16
TCNT7	Timer Counter 7	RW	0x????	0x100020B8	16
TCSR7	Timer Control Register 7	RW	0x0000	0x100020BC	16

16.3.1 Timer Control Register (TCSR)

The TCSR is a 16-bit read/write register. It contains the control bits for each channel. It is initialized to 0x00 by any reset.

TCSR0, TCSR1, TCSR2, TCSR3, TCSR4, TCSR5 TCSR6, TCSR7 — **0x1000204C, 0x1000205C, 0x1000206C, 0x1000207C, 0x1000208C, 0x1000209C, 0x100020AC, 0x100020BC**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					CLRZ	SD	INITL	PWM_EN	PWM_IN_EN	PRESCALE			EXT_EN	RTC_EN	PCK_EN
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15:11	Reserved	These bits always read 0, and written are ignored.	R
10	CLRZ	Clear counter to 0. It is only used in TCU2 mode. Writing 1 to this bit will clear the counter to 0. When the counter is finished setting to 0, it will be cleared by hardware. Writing 0 to this bit will be ignored.	RW
9	SD	Shut Down (SD) the PWM output. It is only used in TCU1 mode. 0: Graceful shutdown 1: Abrupt shutdown Graceful shutdown: The output level for PWM output will keep the level after the comparison match of FULL. Abrupt shutdown: The output level for PWM output will keep the level.	RW

<table>
<tr><td>8</td><td>INITL</td><td>Selects an initial output level for PWM output.
1: High
0: Low</td><td>RW</td></tr>
<tr><td>7</td><td>PWM_EN</td><td>PWM output pin control bit.
1: PWM pin output enable
0: PWM pin output disable, and the PWM pin will be set to the initial level according to INITL</td><td>RW</td></tr>
<tr><td>6</td><td>PWM_IN_EN</td><td>PWM input mode enable.
Set to 1 to enable this function.
In this function, PWM pin need to set as input in GPIO to receive external signal, EXT_EN, RTC_EN, PCK_EN need to set 0.
And TCNT became a counter to count this signal’s both edges. (This bit in TCSR1, 2 is reserved)</td><td>RW</td></tr>
<tr><td>5:3</td><td>PRESCALE</td><td>These bits select the TCNT count clock frequency. Don’t change this field when the channel is running.
<table>
<tr><td>Bit 2</td><td>Bit1</td><td>Bit 0</td><td>Description</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>Internal clock: CLK/1</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>Internal clock: CLK/4</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>Internal clock: CLK/16</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>Internal clock: CLK/64</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>Internal clock: CLK/256</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>Internal clock: CLK/1024</td></tr>
<tr><td colspan="3">110~111</td><td>Reserved</td></tr>
</table></td><td>RW</td></tr>
<tr><td>2</td><td>EXT_EN</td><td>Select EXTAL as the timer clock input.
1: Enable
0: Disable</td><td>RW</td></tr>
<tr><td>1</td><td>RTC_EN</td><td>Select RTCCLK as the timer clock input.
1: Enable
0: Disable</td><td>RW</td></tr>
<tr><td>0</td><td>PCK_EN</td><td>Select PCLK as the timer clock input.
1: Enable
0: Disable</td><td>RW</td></tr>
</table>

NOTES:

1 The input clock of timer and the PCLK should keep to the rules as follows:

Input clock of timer: IN_CLK	**Clock generated from the frequency divider (PRESCALE): DIV_CLK**
PCK_EN == 0, RTC_EN == 1 and EXT_EN == 0 (IN_CLK = RTCCLK)	$f_{DIV\_CLK} < ½ f_{PCLK}$
PCK_EN == 0, RTC_EN == 0 and EXT_EN == 1 (IN_CLK = EXTAL)	$f_{DIV\_CLK} < ½ f_{PCLK}$

PCK_EN == 1, RTC_EN == 0 and EXT_EN == 0 (IN_CLK = PCLK)	ANY

16.3.2 Timer Data FULL Register (TDFR)

The comparison data FULL registers TDFR is used to store the data to be compared with the content of the up-counter TCNT. This register can be directly read and written. (Default: indeterminate) But it is not suggested changing when counter is working in TCU2 mode.

TDFR0, TDFR1, TDFR2, TDFR3, TDFR4, TDFR5, TDFR6, TDFR7 — **0x10002040, 0x10002050, 0x10002060, 0x10002070, 0x10002080, 0x10002090, 0x100020A0, 0x100020B0**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TDFR															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

16.3.3 Timer Data HALF Register (TDHR)

The comparison data HALF registers TDHR is used to store the data to be compared with the content of the up-counter TCNT. This register can be directly read and written. (Default: indeterminate) But it is not suggested changing when counter is working in TCU2 mode.

TDHR0, TDHR1, TDHR2, TDHR3, TDHR4, TDHR5, TDHR6, TDHR7 — **0x10002044, 0x10002054, 0x10002064, 0x10002074, 0x10002084, 0x10002094, 0x100020A4, 0x100020B4**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TDHR															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

16.3.4 Timer Counter (TCNT)

TCNT is a 16-bit read/write register. The up-counter TCNT can be reset to 0 by software and counts up using the prescaler output clock. When TCNT count up to equal to TDFR, it will reset to 0 and continue to count up.

TCU1: The counter data can be read out at any time. The data can be written at any time. This makes it possible to change the interrupt and/or clock output cycles temporarily. (Default: indeterminate)

TCU2: The counter data can be read out at any time, but you should read TSTR.REALn to check

whether the data is real data or not. The data can only be written before counter is started, and the counter clock is pclk. But it can be cleared to 0 by setting TCSR.CLRZ to 1, and if the counter is really cleared, TCSR.CLRZ will be set to 0 by hardware.

TCNT0, TCNT1, TCNT2, TCNT3, TCNT4, TCNT5, TCNT6, TCNT7 — **0x10002048, 0x10002058, 0x10002068, 0x10002078, 0x10002088, 0x10002098, 0x100020A8, 0x100020B8**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TCNT															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

16.3.5 Timer Counter Enable Register (TER)

The TER is a 16-bit read-only register. It contains the counter enable control bits for each channel. It is initialized to 0x0000 by any reset. It can only be set by register TESR and TECR. Since the timer enable control bits are located in the same addresses, two or more timers can be started at the same time.

TER — **0x10002010**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OSTEN	Reserved							TCEN 7	TCEN 6	TCEN 5	TCEN 4	TCEN 3	TCEN 2	TCEN 1	TCEN 0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15	OSTEN	Enable the counter in OST. 1: Begin counting up 0: Stop counting up	
14:8	Reserved	These bits always read 0, and written are ignored.	R
7	TCEN 7	Enable the counter in timer 7. 1: Begin counting up 0: Stop counting up	R
6	TCEN 6	Enable the counter in timer 6. 1: Begin counting up 0: Stop counting up	R
5	TCEN 5	Enable the counter in timer 5. 1: Begin counting up 0: Stop counting up	R
4	TCEN 4	Enable the counter in timer 4.	R

		1: Begin counting up 0: Stop counting up	
3	TCEN 3	Enable the counter in timer 3. 1: Begin counting up 0: Stop counting up	R
2	TCEN 2	Enable the counter in timer 2. 1: Begin counting up 0: Stop counting up	R
1	TCEN 1	Enable the counter in timer 1. 1: Begin counting up 0: Stop counting up	R
0	TCEN 0	Enable the counter in timer 0. 1: Begin counting up 0: Stop counting up	R

16.3.6 Timer Counter Enable Set Register (TESR)

The TCCSR is a 32-bit write-only register. It contains the counter enable set bits for each channel. Since the timer enable control set bits are located in the same addresses, two or more timers can be started at the same time.

TESR 0x10002014

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OSTST	Reserved							TCST 7	TCST 6	TCST 5	TCST 4	TCST 3	TCST 2	TCST 1	TCST 0
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
15	OSTST	Set OSTEN bit of TER. 1: Set OSTEN bit to 1 0: Ignore	W
14:8	Reserved	These bits always read 0, and written are ignored.	W
7	TCST 7	Set TCEN 7 bit of TER. 1: Set TCEN 5 bit to 1 0: Ignore	W
6	TCST 6	Set TCEN 6 bit of TER. 1: Set TCEN 5 bit to 1 0: Ignore	W
5	TCST 5	Set TCEN 5 bit of TER. 1: Set TCEN 5 bit to 1 0: Ignore	W

4	TCST 4	Set TCEN 4 bit of TER. 1: Set TCEN 4 bit to 1 0: Ignore	W
3	TCST 3	Set TCEN 3 bit of TER. 1: Set TCEN 3 bit to 1 0: Ignore	W
2	TCST 2	Set TCEN 2 bit of TER. 1: Set TCEN 2 bit to 1 0: Ignore	W
1	TCST 1	Set TCEN 1 bit of TER. 1: Set TCEN 1 bit to 1 0: Ignore	W
0	TCST 0	Set TCEN 0 bit of TER. 1: Set TCEN 0 bit to 1 0: Ignore	W

16.3.7 Timer Counter Enable Clear Register (TECR)

The TECR is a 32-bit write-only register. It contains the counter enable clear bits for each channel. Since the timer enable clear bits are located in the same addresses, two or more timers can be stop at the same time.

TECR **0x10002018**

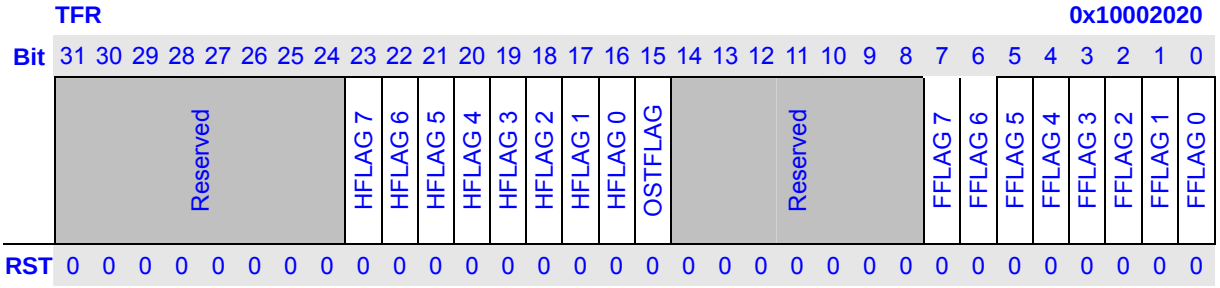
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OSTCL	Reserved							TCCL 7	TCCL 6	TCCL 5	TCCL 4	TCCL 3	TCCL 2	TCCL 1	TCCL 0
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
15	OSTCL	Set OSTEN bit of TER. 1: Set OSTEN 5 bit to 0 0: Ignore	W
14:8	Reserved	These bits always read 0, and written are ignored.	W
7	TCCL 7	Set TCEN 7 bit of TER. 1: Set TCEN 6 bit to 0 0: Ignore	W
6	TCCL 6	Set TCEN 7 bit of TER. 1: Set TCEN 6 bit to 0 0: Ignore	W
5	TCCL 5	Set TCEN 5 bit of TER. 1: Set TCEN 5 bit to 0	W

		0: Ignore	
4	TCCL 4	Set TCEN 4 bit of TER. 1: Set TCEN 4 bit to 0 0: Ignore	W
3	TCCL 3	Set TCEN 3 bit of TER. 1: Set TCEN 3 bit to 0 0: Ignore	W
2	TCCL 2	Set TCEN 2 bit of TER. 1: Set TCEN 2 bit to 0 0: Ignore	W
1	TCCL 1	Set TCEN 1 bit of TER. 1: Set TCEN 1 bit to 0 0: Ignore	W
0	TCCL 0	Set TCEN 0 bit of TER. 1: Set TCEN 0 bit to 0 0: Ignore	W

16.3.8 Timer Flag Register (TFR)

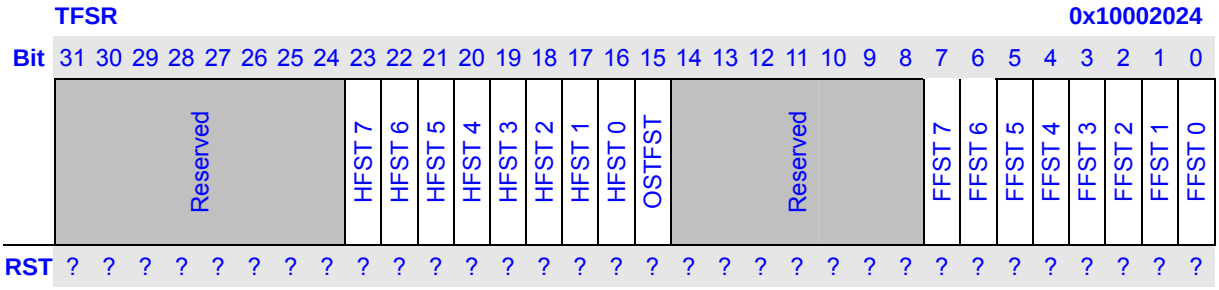
The TFR is a 32-bit read-only register. It contains the comparison match flag bits for all the channels. It can also be set by register TFSR and TFCR. It is initialized to 0x00000000 by any reset.



Bits	Name	Description	RW
31:24	Reserved	These bits always read 0, and written are ignored.	R
23:16	HFLAG 7~0	HALF comparison match flag. (TCNT = TDHR) 1: Comparison match 0: Comparison not match	R
15	OSTFLAG	OST comparison match flag. (OSTCNT = OSTDR) 1: Comparison match 0: Comparison not match	R
14:8	Reserved	These bits always read 0, and written are ignored.	R
7:0	FFLAG 7~0	FULL comparison match flag. (TCNT = TDFR) 1: Comparison match 0: Comparison not match	R

16.3.9 Timer Flag Set Register (TFSR)

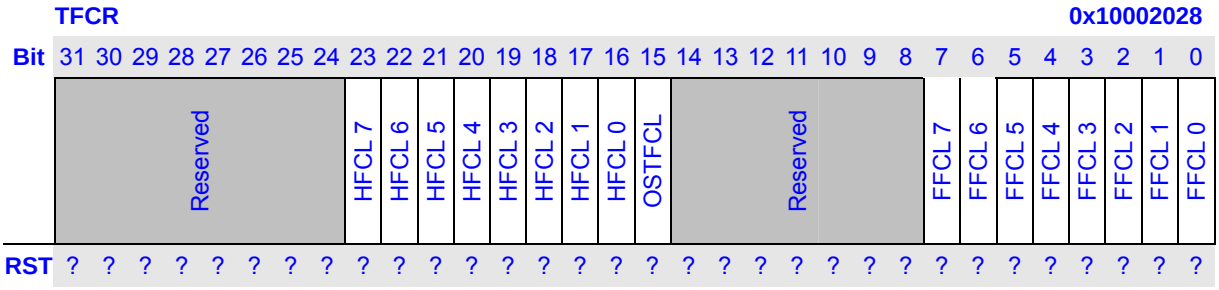
The TFSR is a 32-bit write-only register. It contains the comparison match flag set bits for all the channels.



Bits	Name	Description	RW
31:24	Reserved	-	-
23:16	HFST 7~0	Set HFLAG n bit of TFR. 1: Set HFLAG n bit to 1 0: Ignore	W
15	OSTFST	Set OSTFLAG n bit of TFR. 1: Set OSTFLAG n bit to 1 0: Ignore	W
14:8	Reserved	-	-
7:0	FFST 7~0	Set FFLAG n bit of TFR. 1: Set FFLAG n bit to 1 0: Ignore	W

16.3.10 Timer Flag Clear Register (TFCR)

The TFCR is a 32-bit write-only register. It contains the comparison match flag clear bits for all the channels.



Bits	Name	Description	RW
31:24	Reserved	-	-
23:16	HFCL 7~0	Set HFLAG n bit of TFR. 1: Set FFLAG n bit to 0	W

		0: Ignore	
15	OSTFCL	Set OSTFLAG n bit of TFR. 1: Set OSTFLAG n bit to 0 0: Ignore	W
14:8	Reserved	-	-
7:0	FFCL 7~0	Set FFLAG n bit of TFR. 1: Set FFLAG n bit to 0 0: Ignore	W

16.3.11 Timer Mask Register (TMR)

The TMR is a 32-bit read-only register. It contains the comparison match flag bits for all the channels. It is initialized to 0x003F003F by any reset. It can only be set by register TMSR and TMCR.

TMR **0x10002030**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								HMASK 7	HMASK 6	HMASK 5	HMASK 4	HMASK 3	HMASK 2	HMASK 1	HMASK 0	OSTMASK	Reserved							FMASK 7	FMASK 6	FMASK 5	FMASK 4	FMASK 3	FMASK 2	FMASK 1	FMASK 0
RST	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Bits	Name	Description	RW
31:24	Reserved	These bits always read 0, and written are ignored.	R
23:16	HMASK 7~0	HALF comparison match interrupt mask. 1: Comparison match interrupt mask 0: Comparison match interrupt not mask	R
15	OSTMASK	OST comparison match interrupt mask. 1: Comparison match interrupt mask 0: Comparison match interrupt not mask	R
14:8	Reserved	These bits always read 0, and written are ignored.	R
7:0	FMASK 7~0	FULL comparison match interrupt mask. 1: Comparison match interrupt mask 0: Comparison match interrupt not mask	R

16.3.12 Timer Mask Set Register (TMSR)

The TMSR is a 32-bit write-only register. It contains the comparison match flag set bits for all the channels.

TMSR 0x10002034

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								HMST 7	HMST 6	HMST 5	HMST 4	HMST 3	HMST 2	HMST 1	HMST 0	OSTMST	Reserved							FMST 7	FMST 6	FMST 5	FMST 4	FMST 3	FMST 2	FMST 1	FMST 0
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
31:24	Reserved	-	-
23:16	HMST 7~0	Set HMASK n bit of TMR. 1: Set HMASK n bit to 1 0: Ignore	W
15	OSTMST	Set OSTMASK n bit of TMR. 1: Set OSTMASK n bit to 1 0: Ignore	W
14:8	Reserved	-	-
7:0	FMST 7~0	Set FMASK n bit of TMR. 1: Set FMASK n bit to 1 0: Ignore	W

16.3.13 Timer Mask Clear Register (TMCR)

The TMCR is a 32-bit write-only register. It contains the comparison match flag clear bits for all the channels.

TMCR 0x10002038

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								HMCL 7	HMCL 6	HMCL 5	HMCL 4	HMCL 3	HMCL 2	HMCL 1	HMCL 0	OSTMCL	Reserved							FMCL 7	FMCL 6	FMCL 5	FMCL 4	FMCL 3	FMCL 2	FMCL 1	FMCL 0
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
31:22	Reserved	-	-
23:16	HMCL 7~0	Set HMASK n bit of TMR. 1: Set HMASK n bit to 0	W

		0: Ignore	
15	OSTMCL	Set OSTMASK n bit of TMR. 1: Set OSTMASK n bit to 0 0: Ignore	W
14:8	Reserved	-	-
7:0	FMCL 7~0	Set FMASK n bit of TMR. 1: Set FMASK n bit to 0 0: Ignore	W

16.3.14 Timer Stop Register (TSR)

The TSR is a 32-bit read-only register. It contains the timer stop control bits for each channel, WDT and OST. It is initialized to 0x00000000 by any reset. It can only be set by register TSSR and TSCR. If set, clock supplies to timer n / WDT / OST is stopped, and registers of the timer / WDT / OST cannot be accessed also.

TSR 0x1000201C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDTS	OSTS	Reserved							STOP 7	STOP 6	STOP 5	STOP 4	STOP 3	STOP 2	STOP 1	STOP 0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:17	Reserved	These bits always read 0, and written are ignored.	R
16	WDTS	1: The clock supplies to WDT is stopped 0: The clock supplies to WDT is supplied	R
15	OSTS	1: The clock supplies to OST is stopped 0: The clock supplies to OST is supplied	R
14:8	Reserved	These bits always read 0, and written are ignored.	R
7	STOP 7	1: The clock supplies to timer 7 is stopped 0: The clock supplies to timer 7 is supplied	R
6	STOP 6	1: The clock supplies to timer 6 is stopped 0: The clock supplies to timer 6 is supplied	R
5	STOP 5	1: The clock supplies to timer 5 is stopped 0: The clock supplies to timer 5 is supplied	R
4	STOP 4	1: The clock supplies to timer 4 is stopped 0: The clock supplies to timer 4 is supplied	R
3	STOP 3	1: The clock supplies to timer 3 is stopped 0: The clock supplies to timer 3 is supplied	R
2	STOP 2	1: The clock supplies to timer 2 is stopped 0: The clock supplies to timer 2 is supplied	R

1	STOP 1	1: The clock supplies to timer 1 is stopped 0: The clock supplies to timer 1 is supplied	R
0	STOP 0	1: The clock supplies to timer 0 is stopped 0: The clock supplies to timer 0 is supplied	R

16.3.15 Timer Stop Set Register (TSSR)

The TCSR is an 32-bit write-only register. It contains the timer stop set bits for each channel, WDT and OST. Since the timer stop control set bits are located in the same addresses, two or more timers can be started at the same time.

TSSR **0x1000202C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDTSS	OSTSS	Reserved							STPS 7	STPS 6	STPS 5	STPS 4	STPS 3	STPS 2	STPS 1	STPS 0
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
31:17	Reserved	-	-
16	WDTSS	Set WDTS bit of TSR. 1: Set WDTS bit to 1 0: Ignore	W
15	OSTSS	Set OSTS bit of TSR. 1: Set OSTS bit to 1 0: Ignore	W
14:8	Reserved	-	-
7	STPS 7	Set STOP 7 bit of TSR. 1: Set STOP 7 bit to 1 0: Ignore	W
6	STPS 6	Set STOP 6 bit of TSR. 1: Set STOP 6 bit to 1 0: Ignore	W
5	STPS 5	Set STOP 5 bit of TSR. 1: Set STOP 5 bit to 1 0: Ignore	W
4	STPS 4	Set STOP 4 bit of TSR. 1: Set STOP 4 bit to 1 0: Ignore	W
3	STPS 3	Set STOP 3 bit of TSR. 1: Set STOP 3 bit to 1 0: Ignore	W

2	STPS 2	Set STOP 2 bit of TSR. 1: Set STOP 2 bit to 1 0: Ignore	W
1	STPS 1	Set STOP 1 bit of SR. 1: Set STOP 1 bit to 1 0: Ignore	W
0	STPS 0	Set STOP 0 bit of TSR. 1: Set STOP 0 bit to 1 0: Ignore	W

16.3.16 Timer Stop Clear Register (TSCR)

The TSCR is an 32-bit write-only register. It contains the timer stop clear bits for each channel, WDT and OST. Since the timer stop clear bits are located in the same addresses, two or more timers can be stop at the same time.

TSCR **0x1000203C**

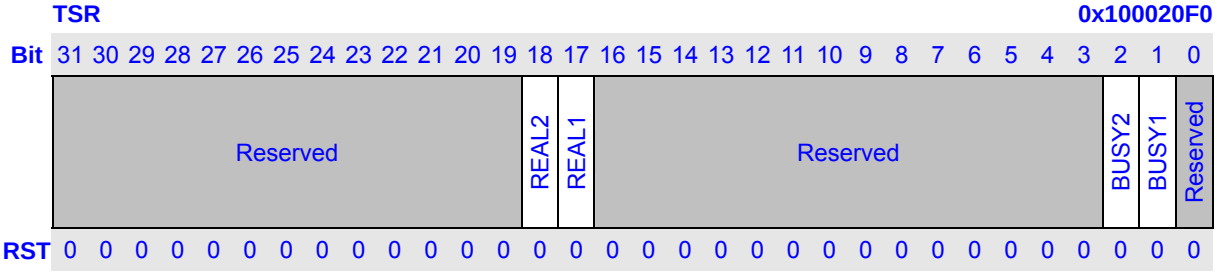
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															WDTSC	OSTSC	Reserved							STPC 7	STPC 6	STPC 5	STPC 4	STPC 3	STPC 2	STPC 1	STPC 0
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
31:17	Reserved	-	-
16	WDTSC	Set WDTS bit of TSR. 1: Set WDTS bit to 0 0: Ignore	W
15	OSTSC	Set OSTS bit of TSR. 1: Set OSTS bit to 0 0: Ignore	W
14:8	Reserved	-	-
7	STPC 7	Set STOP 7 bit of TSR. 1: Set STOP 7 bit to 0 0: Ignore	W
6	STPC 6	Set STOP 6 bit of TSR. 1: Set STOP 6 bit to 0 0: Ignore	W
5	STPC 5	Set STOP 5 bit of TSR. 1: Set STOP 5 bit to 0 0: Ignore	W
4	STPC 4	Set STOP 4 bit of TSR.	W

		1: Set STOP 4 bit to 0 0: Ignore	
3	STPC 3	Set STOP 3 bit of TSR. 1: Set STOP 3 bit to 0 0: Ignore	W
2	STPC 2	Set STOP 2 bit of TSR. 1: Set STOP 2 bit to 0 0: Ignore	W
1	STPC 1	Set STOP 1 bit of TSR. 1: Set STOP 1 bit to 0 0: Ignore	W
0	STPC 0	Set STOP 0 bit of TSR. 1: Set STOP 0 bit to 0 0: Ignore	W

16.3.17 Timer Status Register (TSTR)

The TSTR is a 32-bit read-only register. It contains the status of channel in TCU2 mode. The register can be written by setting register TSTSR and TSTCR.



Bits	Name	Description	RW
31:19	Reserved	These bits always read 0, and written are ignored.	R
18	REAL 2	1: The value read from counter 2 is a real value 0: The value read from counter 2 is a false value	R
17	REAL1	1: The value read from counter 1 is a real value 0: The value read from counter 1 is a false value	R
16:3	Reserved	These bits always read 0, and written are ignored.	R
2	BUSY 2	1: The counter 2 is busy now 0: The counter 2 is ready now	R
1	BUSY1	1: The counter 1 is busy now 0: The counter 1 is ready now	R
0	Reserved	These bits always read 0, and written are ignored.	R

16.3.18 Timer Status Set Register (TSTSR)

The TSTSR is a 32-bit write-only register. It contains the timer status set bits for each channel.

TSTSR **0x100020F4**

Bit	31:19	18	17	16:3	2	1	0
	Reserved	REALS 2	REALS 1	Reserved	BUSYS 2	BUSYS 1	Reserved
RST	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:19	Reserved	These bits always read 0, and written are ignored.	R
18	REALS 2	Set REAL 2 bit of TSTR. 1: Set REAL 2 bit to 1 0: Ignore	R
17	REALS 1	Set REAL 1 bit of TSTR. 1: Set REAL 1 bit to 1 0: Ignore	R
16:3	Reserved	These bits always read 0, and written are ignored.	R
2	BUSYS 2	Set BUSY 2 bit of TSTR. 1: Set BUSY 2 bit to 1 0: Ignore	R
1	BUSYS 1	Set BUSY 1 bit of TSTR. 1: Set BUSY 1 bit to 1 0: Ignore	R
0	Reserved	These bits always read 0, and written are ignored.	R

16.3.19 Timer Status Clear Register (TSTCR)

The TSTCR is a 32-bit write-only register. It contains the timer status clear bits for each channel.

TSTCR **0x100020F8**

Bit	31:19	18	17	16:3	2	1	0
	Reserved	REALC 2	REALC 1	Reserved	BUSYC 2	BUSYC 1	Reserved
RST	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:19	Reserved	These bits always read 0, and written are ignored.	R
18	REALC 2	Clear REAL 2 bit of TSTR.	R

		1: Clear REAL 2 bit to 1 0: Ignore	
17	REALC 1	Clear REAL 1 bit of TSTR. 1: Clear REAL 1 bit to 1 0: Ignore	R
16:3	Reserved	These bits always read 0, and written are ignored.	R
2	BUSYC 2	Clear BUSY 2 bit of TSTR. 1: Clear BUSY 2 bit to 1 0: Ignore	R
1	BUSYC 1	Clear BUSY 1 bit of TSTR. 1: Clear BUSY 1 bit to 1 0: Ignore	R
0	Reserved	These bits always read 0, and written are ignored.	R

16.4 Operation

16.4.1 Basic Operation in TCU1 Mode

The value of TDFR should be bigger than TDHR, and the minimum settings are TDHR = 0 and TDFR = 1. In this case, the timer output clock cycle is the input clock × 1/2. If TDHR > TDFR, no comparison TFHR signal is generated.

Before the timer counter begin to count up, we need to do as follows:

If you want to use PWM you should set TCSR.PWM_EN to be 0 before you initial TCU.

1 Initial the configuration.
   a Writing TCSR.INITL to initialize PWM output level.
   b Writing TCSR.SD to setting the shutdown mode (Abrupt shutdown or Graceful shutdown).
   c Writing TCSR.PRESCALE to set TCNT count clock frequency.
   d Setting TCNT, TDHR and TDFR.

2 Enable the clock.
   a Writing TCSR.PWM_EN to set whether enable PWM or disable PWM.
   b Writing TCSR.EXT_EN, TCSR.RTC_EN or TCSR.PCK_EN to 1 to select the input clock and enable the input clock. Only one of TCSR.EXT_EN, TCSR.RTC_EN and TCSR.PCK_EN can be set to 1.

After initialize the register of timer, we should start the counter as follows:

3 Enable the counter.
   Setting the TESR.TCST bit to 1 to enable the TCNT.

NOTES:

1 The input clock and PCLK should follow the rules advanced before.

16.4.2 Disable and Shutdown Operation in TCU1 Mode

1 Setting the TECR.TCCL bit to 1 to disable the TCNT.

16.4.3 Basic Operation in TCU2 Mode

The value of TDFR should be bigger than TDHR, and the minimum settings are TDHR = 0 and TDFR = 1. In this case, the timer output clock cycle is the input clock × 1/2. If TDHR > TDFR, no comparison TFHR signal is generated.

Initial state is that TCSR.PRESCALE=0, TCSR.PWM_EN=0 and TCENR=0.

1 Reset the TCU.
   a Writing TCSR.PCK_EN to 1 to select pclk as the input clock.
   b Set TCSR.CLRZ to 1 to clear TCNT or set TCNT to an initial value.
   c Writing TCSR.PCK_EN to 0 to close the input clock.

2 Initial the configuration.
   a Setting TDHR and TDFR.
   b Writing TCSR.INITL to initialize PWM output level (if used PWM).
   c Writing TCSR.PRESCALE to set TCNT count clock frequency.
   d Writing TCSR.EXT_EN, TCSR.RTC_EN or TCSR.PCK_EN to 1 to select the input clock and enable the input clock. Only one of TCSR.EXT_EN, TCSR.RTC_EN and TCSR.PCK_EN can be set to 1.
   e Writing TCSR.PWM_EN to set whether enable PWM or disable PWM.

After initialize the register of timer, we should start the counter as follows:

3 Setting the TESR.TCST bit to 1 to enable the TCNT.

NOTE:

You can clear the counter when counter is working.

1 Set TCSR.CLRZ to 1 to clear TCNT.
2 Wait till TSTR.BUSY = 0, that is the counter have been cleared.

You can enable PWM or disable PWM the counter when counter is working.

1 Set TCSR.PWM_EN to 1 to enable PWM.
2 Set TCSR.PWM_EN to 0 to disable PWM.

16.4.4 Disable and Shutdown Operation in TCU2 Mode

1 Writing TCSR.PWM_EN to 0 to disable PWM.
2 Setting the TECR.TCCL bit to 1 to disable the TCNT.
3 Wait till TSTR.BUSY = 0, that is the reset of counter is finished.

16.4.5 Read Counter in TCU2 Mode

If you want to read the data from register TCNT when the TCU is working, you can check TSTR.REAL whether it is good or not. It is suggested that:

1 If TSTR.REAL==1, the data read is available.
2 If TSTR.REAL==0, reread the counter till TSTR.REAL==1, the data read is available.
3 If TSTR.REAL is always 0, you can read some data, and lose some data that is quick different from the others. Then choose a data from them as the available data.

NOTES:

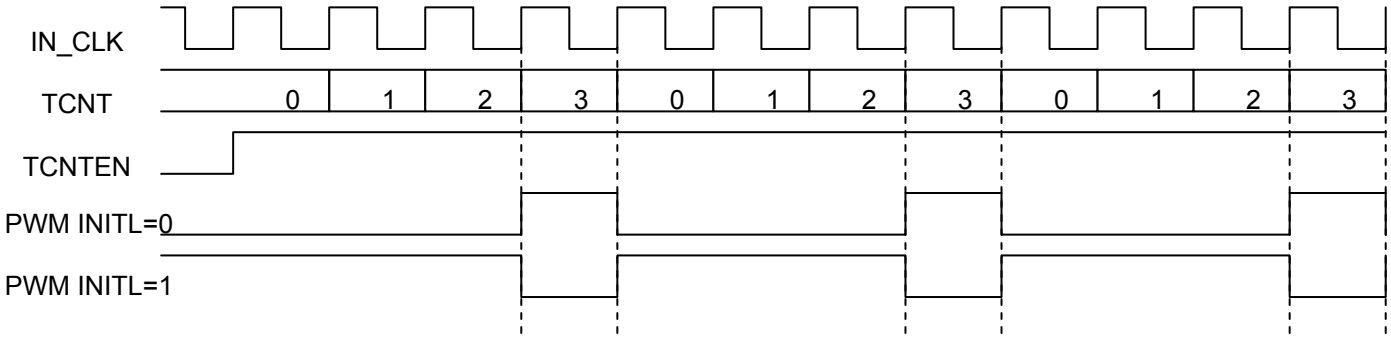
1 It suggested that (1), (2) is often used when the counter clock is very slow.
2 It suggested that (3) is often used when the counter clock is very fast.

16.4.6 Pulse Width Modulator (PWM)

Timer 0~7 can be used as Pulse Width Modulator (PWM). The PWM can be used to control the back

light inverter or adjust bright or contrast of LCD panel.

FULL comparison match signal and HALF comparison match signal can determine an attribute of the PWM_OUT waveform. FULL comparison match signal specifies the clock cycle for the PWM module clock. HALF comparison match signal specifies the duty ratio for the PWM module clock.



16.4.7 Trackball Input Waveform Detect

Timer 0, 3~7 can be used as a waveform edge counter to count both positive edge and negative edge of an external input waveform. For example, a trackball device's input. 4 timers will need to count all four directions (up, down, left, right). You need configure relate GPIO (set relate 4 PWM IO as input) and set relate TCSR.PWM_IN_EN to 1. Both relate TDFR and TDHR need to set to 0xFFFF, unless you need a special interrupt when the counter hit TDFR or TDHR. The counter will clear to 0 when hit TDFR.

Before the timer counter begin to count up, we need to do as follows:

1 Initial the configuration.
   a Writing TCSR.SD to setting the shutdown mode. (Abrupt shutdown or Graceful shutdown)
   b Writing TCSR.PRESCALE to set to 0.
   c Setting TCNT, TDHR and TDFR.

2 Enable the clock.
   a Writing TCSR.PWM_EN to disable PWM.
   b Writing TCSR.EXT_EN, TCSR.RTC_EN and TCSR.PCK_EN to 0, TCSR.PWM_IN_EN to 1 to select the input clock and enable the input clock.

After initialize the register of timer, we should start the counter as follows:

3 Enable the counter.
   Setting the TESR.TCST bit to 1 to enable the TCNT.

NOTES:

1 The input clock and PCLK should follow the rules advanced before.

17 Operating System Timer

17.1 Overview

The OST (Operating System Timer) contains one 32-bit programmable timer. It can be used as operating system timer.

OST has the following features:

- OST includes
  - 32-bit Counter
  - 32-bit Compare Data Register
  - Control Register
- Independent clock for each counter, selectable by software
  - PCLK, EXTAL and RTCCLK can be used as the clock for counter
  - The division ratio of the clock can be set to 1, 4, 16, 64, 256 and 1024 by software
- Match interrupt can be generated for OST using the compare data registers
  - Interrupt flag and interrupt mask is same with TCU in TCU spec

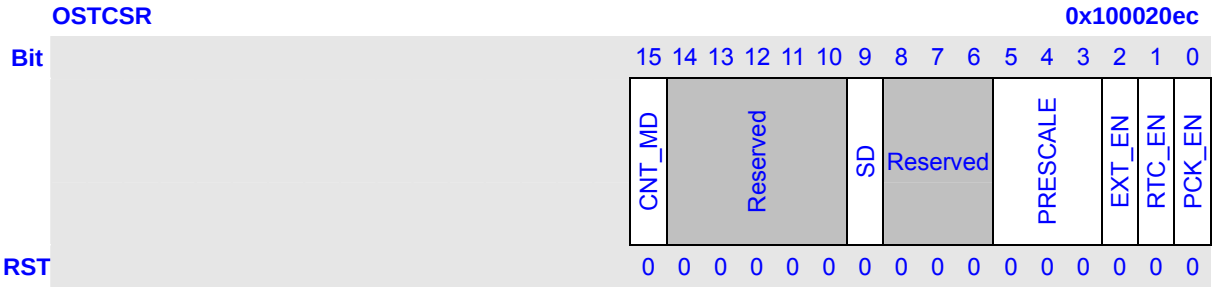
17.2 Register Description

In this section, we will describe the registers in OST. Following table lists all the registers definition. All OST register's 32bit address is physical address. And detailed function of each register will be described below.

Name	Description	RW	Reset Value	Address	Access Size
OSTDR	Operating System Timer Data Register	RW	0x????????	0x100020e0	32
OSTCNT	Operating System Timer Counter	RW	0x????????	0x100020e8	32
OSTCSR	Operating System Timer Control Register	RW	0x0000	0x100020ec	16

17.2.1 Operating System Control Register (OSTCSR)

The TCSR is a 16-bit read/write register. It contains the control bits for OST. It is initialized to 0x00 by any reset.



Bits	Name	Description	RW
15	CNT_MD	Counter mode choose bit. 0: When the value counter is equal to compare value, the counter will be cleared, and increase from 0 1: When the value counter is equal to compare value, the counter will go on increasing till overflow, and then increase from 0	
14:6	Reserved	These bits always read 0, and written are ignored.	R
9	SD	Shut Down (SD) the PWM output. It is only used in TCU1 mode. 0: Graceful shutdown (only used when CNT_MD = 0) 1: Abrupt shutdown	RW
5:3	PRESCALE	These bits select the TCNT count clock frequency.	RW

Bit 2	Bit1	Bit 0	Description
0	0	0	Internal clock: CLK/1
0	0	1	Internal clock: CLK/4
0	1	0	Internal clock: CLK/16
0	1	1	Internal clock: CLK/64

		1	0	0	Internal clock: CLK/256	
		1	0	1	Internal clock: CLK/1024	
		110~111			Reserved	
2	EXT_EN	Select EXTAL as the timer clock input. 1: Enable 0: Disable				RW
1	RTC_EN	Select RTCCLK as the timer clock input. 1: Enable 0: Disable				RW
0	PCK_EN	Select PCLK as the timer clock input. 1: Enable 0: Disable				RW

NOTES:

1 The input clock of timer and the PCLK should keep to the rules as follows:

Input clock of timer: IN_CLK	**Clock generated from the frequency divider (PRESCALE): DIV_CLK**
PCK_EN == 0, RTC_EN == 1 and EXT_EN == 0 (IN_CLK = RTCCLK)	$f_{DIV\_CLK} < ½ f_{PCLK}$
PCK_EN == 0, RTC_EN == 0 and EXT_EN == 1 (IN_CLK = EXTAL)	$f_{DIV\_CLK} < ½ f_{PCLK}$
PCK_EN == 1, RTC_EN == 0 and EXT_EN == 0 (IN_CLK = PCLK)	ANY

17.2.2 Operating System Timer Data Register (OSTDR)

The operating system timer data register OSTDR is used to store the data to be compared with the content of the operating system timer up-counter OSTCNT. This register can be directly read and written. (Default: indeterminate)

OSTDR **0x100020e0**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OSTDR																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

17.2.3 Operating System Timer Counter (OSTCNT)

The operating system timer counter (OSTCNT) is a 32-bit read/write counter. The up-counter OSTCNT can be set by software and counts up using the prescaler output clock. The data can be read out at any time. The counter data can be written at any time. (Default: indeterminate)

OSTCNT **0x100020e8**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OSTCNT																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

17.3 Operation

17.3.1 Basic Operation

Before the timer counter begins to count up, we need to do as follows:

1 Initial the configuration.
   a Writing TCSR.SD to setting the shutdown mode (Abrupt shutdown or Graceful shutdown).
   b Writing OSTCSR.PRESCALE to set OSTCNT count clock frequency.
   c Setting OSTCNT and OSTDR.

2 Enable the clock.
   Writing OSTCSR.EXT_EN, OSTCSR.RTC_EN or OSTCSR.PCK_EN to 1 to select the input clock and enable the input clock. Only one of OSTCSR.EXT_EN, OSTCSR.RTC_EN and OSTCSR.PCK_EN can be set to 1.

After initialize the register of timer, we should start the counter as follows:

3 Enable the counter.
   Setting the TESR.OSTCST bit to 1 to enable the OSTCNT.

NOTES:

1 The input clock and PCLK should follow the rules advanced before.

17.3.2 Disable and Shutdown Operation

1 Setting the TECR.OSTCCL bit to 1 to disable the OSTCNT.

18 Watchdog Timer

18.1 Overview

The watchdog timer is used to resume the processor whenever it is disturbed by malfunctions such as noise and system errors. The watchdog timer can generate the reset signal.

Features:

- Generates WDT reset
- A 16-bit Data register and a 16-bit counter
- Counter clock uses the input clock selected by software
  - PCLK, EXTAL and RTCCLK can be used as the clock for counter
  - The division ratio of the clock can be set to 1, 4, 16, 64, 256 and 1024 by software

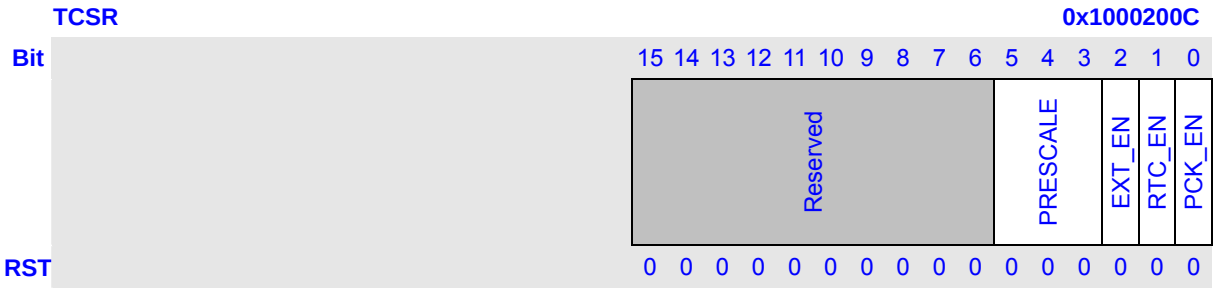
18.2 Register Description

In this section, we will describe the registers in WDT. Following table lists all the registers definition. All WDT register's 32bit address is physical address. And detailed function of each register will be described below.

Name	Description	RW	Reset Value	Address	Access Size
TDR	Watchdog Timer Data Register	RW	0x????	0x10002000	16
TCER	Watchdog Counter Enable Register	RW	0x00	0x10002004	8
TCNT	Watchdog Timer Counter	RW	0x????	0x10002008	16
TCSR	Watchdog Timer Control Register	RW	0x0000	0x1000200C	16

18.2.1 Watchdog Control Register (TCSR)

The TCSR is a 16-bit read/write register. It contains the control bits for WDT. It is initialized to 0x00 by any reset.



Bits	Name	Description	RW
15:6	Reserved	These bits always read 0, and written are ignored.	R
5:3	PRESCALE	These bits select the TCNT count clock frequency.	RW
2	EXT_EN	Select EXTAL as the timer clock input. 1: Enable 0: Disable	RW
1	RTC_EN	Select RTCCLK as the timer clock input. 1: Enable	RW

PRESCALE:

Bit 2	Bit1	Bit 0	Description
0	0	0	Internal clock: CLK/1
0	0	1	Internal clock: CLK/4
0	1	0	Internal clock: CLK/16
0	1	1	Internal clock: CLK/64
1	0	0	Internal clock: CLK/256
1	0	1	Internal clock: CLK/1024
110~111			Reserved

		0: Disable	
0	PCK_EN	Select PCLK as the timer clock input. 1: Enable 0: Disable	RW

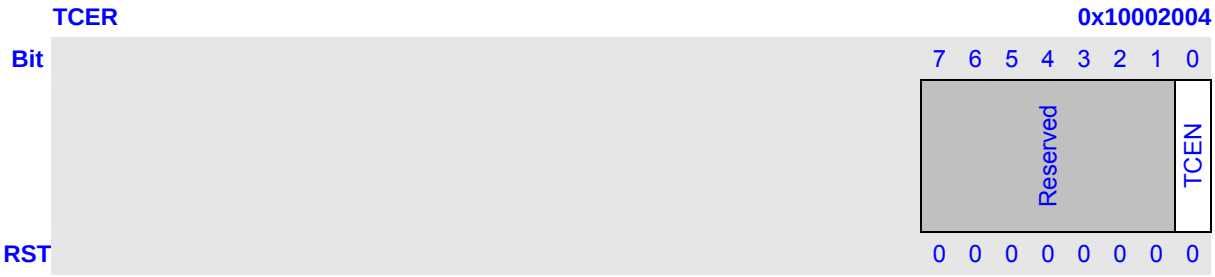
NOTES:

1 The input clock of timer and the PCLK should keep to the rules as follows:

Input clock of timer: IN_CLK	**Clock generated from the frequency divider (PRESCALE): DIV_CLK**
PCK_EN == 0, RTC_EN == 1 and EXT_EN == 0 (IN_CLK = RTCCLK)	$f_{DIV\_CLK} < ½ f_{PCLK}$
PCK_EN == 0, RTC_EN == 0 and EXT_EN == 1 (IN_CLK = EXTAL)	$f_{DIV\_CLK} < ½ f_{PCLK}$
PCK_EN == 1, RTC_EN == 0 and EXT_EN == 0 (IN_CLK = PCLK)	ANY

18.2.2 Watchdog Enable Register (TCER)

The TCER is an 8-bit read/write register. It contains the counter enable control bits for watchdog. It is initialized to 0x00 by any reset.



Bits	Name	Description	RW
7:1	Reserved	These bits always read 0, and written are ignored.	R
0	TCEN	Counter enable control. 0: Timer stop 1: Timer running	RW

18.2.3 Watchdog Timer Data Register (TDR)

The watchdog timer data register TDR is used to store the data to be compared with the content of the watchdog timer up-counter TCNT. This register can be directly read and written. (Default: indeterminate)

TDR																0x10002000
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TDR															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

18.2.4 Watchdog Timer Counter (TCNT)

The watchdog timer counter (TCNT) is a 16-bit read/write counter. The up-counter TCNT can be reset to 0 by software and counts up using the prescaler output clock. When TCNT count up to equal to TDR, the comparison match signal will be generated and a WDT reset is generated. The data can be read out at any time. The counter data can be written at any time. (Default: indeterminate)

TCNT																0x10002008
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TCNT															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

18.3 Watchdog Timer Function

The following describes steps of using WDT:

1 Setting the PRESCALE of input clock in register TCSR.
2 Set register TDR and TCNT.
3 Select the input clock and enable the input clock in register TCSR.

After initialize the register of timer, we should start the counter as follows:

4 Set TCEN bit in TCER to 1. The counter TCNT begins to count.
5 If TCNT = TDR, a WDT reset will be generated.

NOTES:

1 The input clock and PCLK should follow the rules advanced before.
2 The clock of WDT can be stopped by setting register TSR, and register TSR can only be set by register TSSR or TSCR. The content of register TSR, TSSR and TSCR can be found in TCU spec.

19 LCD Controller

19.1 Overview

The JZ integrated LCD controller has the capabilities to driving the latest industry standard STN and TFT LCD panels. It also supports some special TFT panels used in consuming electronic products. The controller performs the basic memory based frame buffer and palette buffer to LCD panel data transfer through use of a dedicated DMA controller. Temporal dithering (frame rate modulation) is supported for STN LCD panels. And OSD is also supported for LCD controller.

Features:

- Basic Features
  - Support PAL/NTSC TV out. 3-components (YUV) TV out (refer TVE spec). VGA
  - Support CCIR601/656 data format
  - Single and Dual panel displays in STN mode
  - Single panel displays in TFT mode
  - Display size up to 1280x720@60Hz (BPP24)
  - Internal palette RAM 256x16 bits

- Colors Supports
  - Encoded pixel data of 1, 2, 4, 8 or 16 BPP in STN mode
  - Support 2, 4, 16 grayscales and up to 4096 colors in STN mode
  - Encoded pixel data of 1, 2, 4, 8, 16, 18 or 24 BPP in TFT mode
  - Support 65,536(65K), 262,144(260K) and up to 16,777,216 (16M) colors in TFT mode

- Panel Supports
  - Support single STN panel and dual STN panel with 1, 2, 4, 8 data output pins
  - Support 16-bit parallel TFT panel
  - Support 18-bit parallel TFT panel
  - Support 24-bit serial TFT panel with 8 data output pins
  - Support 24-bit parallel TFT panel
  - Support Delta RGB panel

- OSD Supports
  - Supports one single color background
  - Supports two foregrounds, and every size can be set for each foreground
  - Supports one transparency for the whole graphic
  - Supports one transparency for each pixel in one graphic
  - Supports color key and mask color key

19.2 Pin Description

Table 19-1 LCD Controller Pins Description

Name	I/O	Description
Lcd_pclk	Input/Output	Display device pixel clock
Lcd_vsync	Input/Output	Display device vertical synchronize pulse
Lcd_hsync	Input/Output	Display device horizontal synchronize pulse
Lcd_de	Output	Display device is STN: AC BIAS Pin Display device is NOT STN: data enable Pin
Lcd_d[17:0]	Output	Display device data pins
lcd_lo6_o[5:0]	Output	Display device data pins use in 24 bit parallel mode
Lcd_spl*[1]	Output	Programmable special pin for generating control signals
Lcd_cls*[1]	Output	Programmable special pin for generating control signals
Lcd_ps*[1]	Output	Programmable special pin for generating control signals
Lcd_rev*[1]	Output	Programmable special pin for generating control signals

NOTES:

1 The mode and timing of special pin Lcd_spl, Lcd_cls, Lcd_ps and Lcd_rev can be seen in part 1.7 LCD Controller Pin Mapping.

19.3 Block Diagram

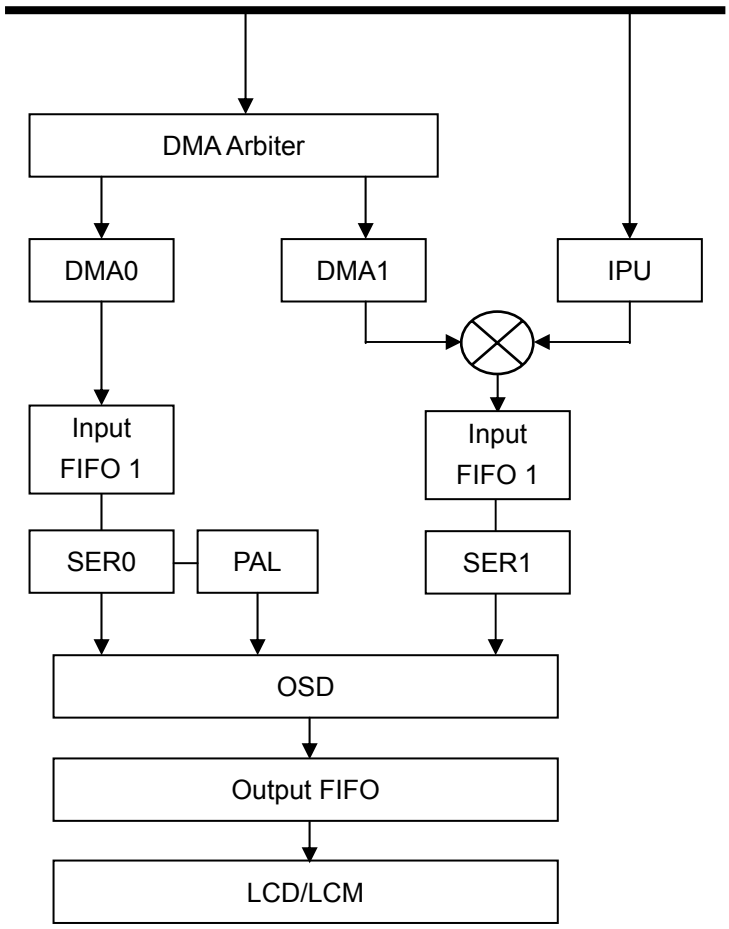


Figure 19-1 Block Diagram when use OSD mode

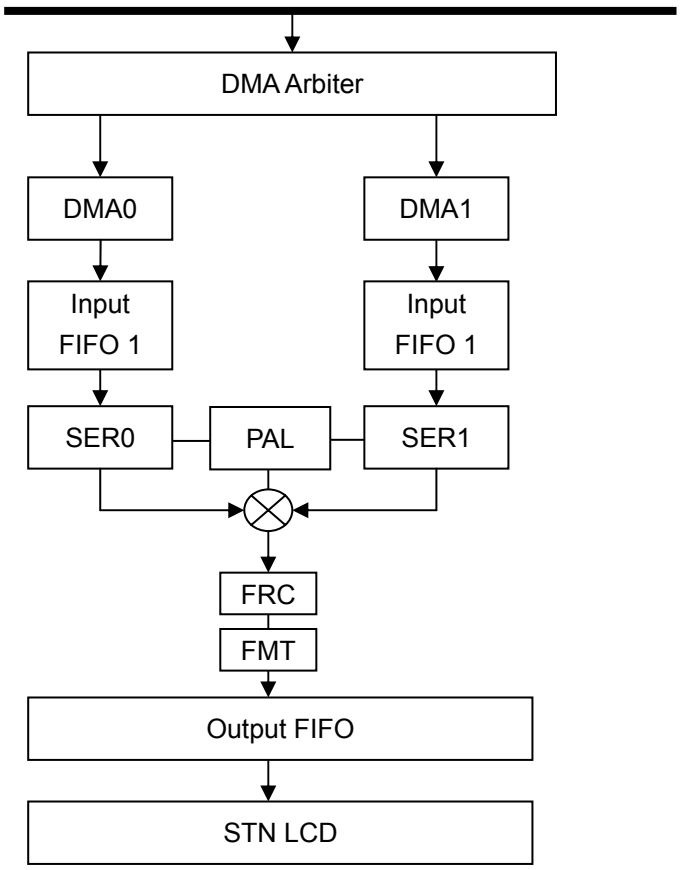


Figure 19-2 Block Diagram of STN mode (not use OSD)

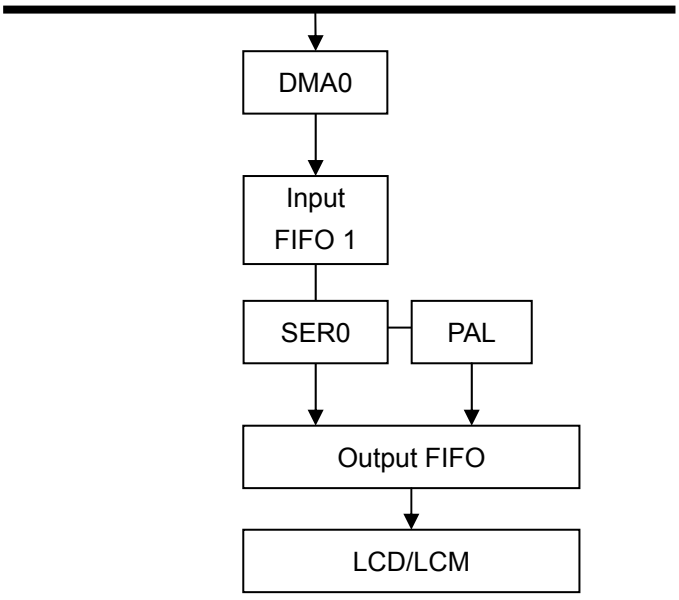


Figure 19-3 Block Diagram of TFT mode (not use OSD)

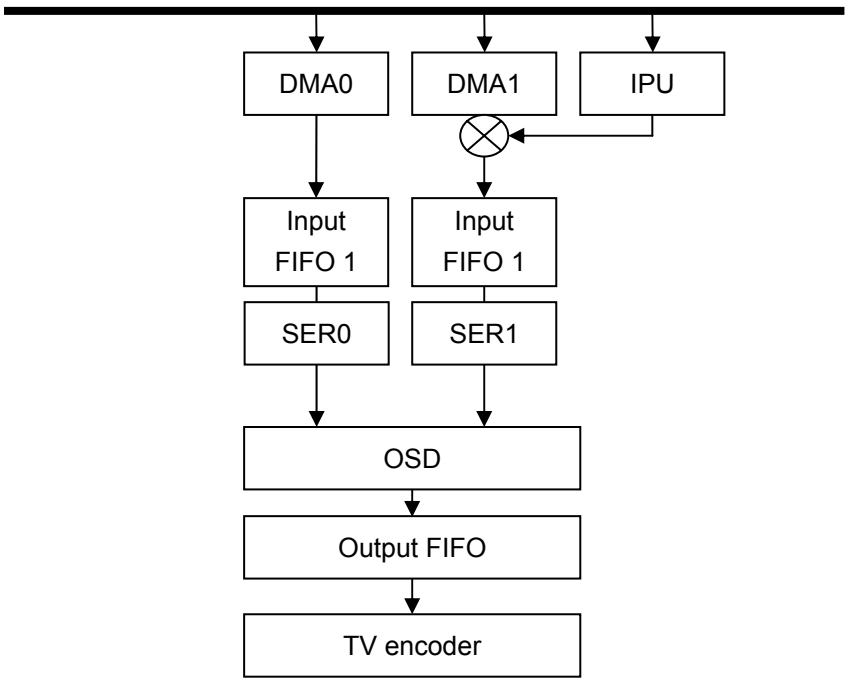


Figure 19-4 Block Diagram of TV interface

19.4 LCD Display Timing

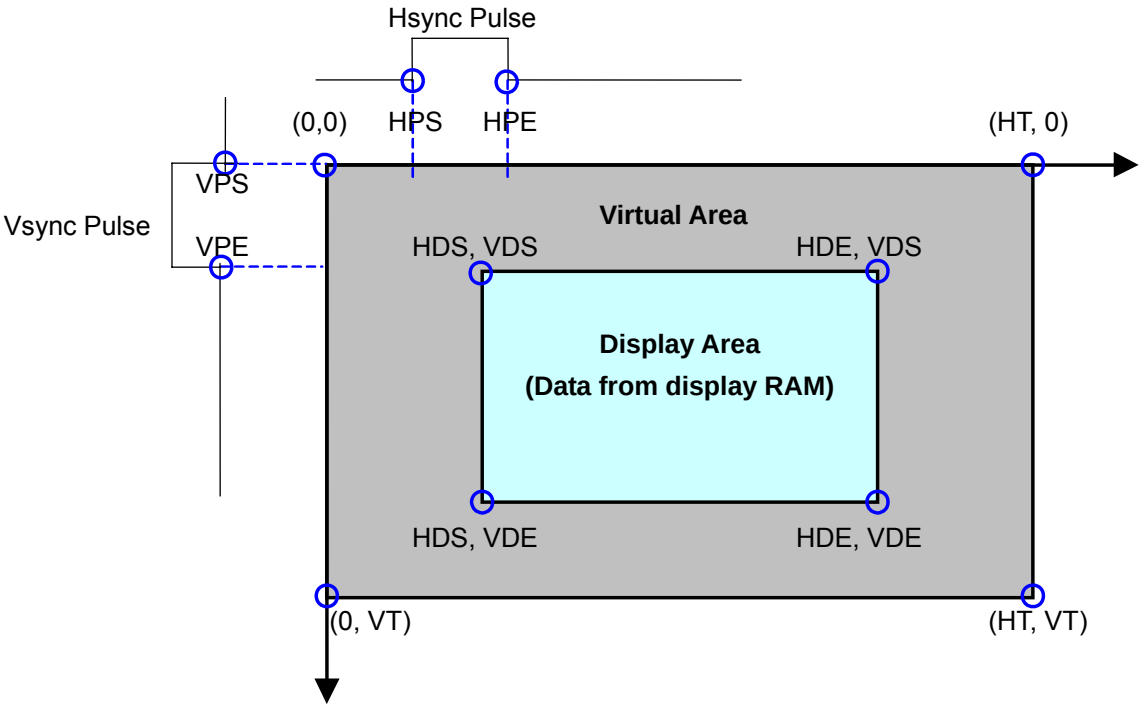


Figure 19-5 Display Parameters

NOTES:

1 VPS === 0
VSYNC pulse always start at point (0,0)

2 H: Horizontal V: Vertical T: Total
D: Display Area P: Pulse
S: Start point E: End point

In the (H, V) Coordinates:

1 The gray rectangle (0, 0) to (HT, VT) is “Virtual Area”.
2 The blue rectangle (HDS, VDS) to (HDE, VDE) is “Display Area”.
3 VPS, VPE defines the VSYNC signal timing. (VPS always be zero)
4 HPS, HPE defines the HSYNC signal timing.

All timing parameters start with “H” is measured in lcd_pclk ticks.
All timing parameters start with “V” is measured in lcd_hsync ticks.

This diagram describes the general LCD panel parameters, these can be set via the registers that describes in next section.

19.5 TV Encoder Timing

Some of Video Encoders for TV (Tele Vision) require interlaced timing interface.

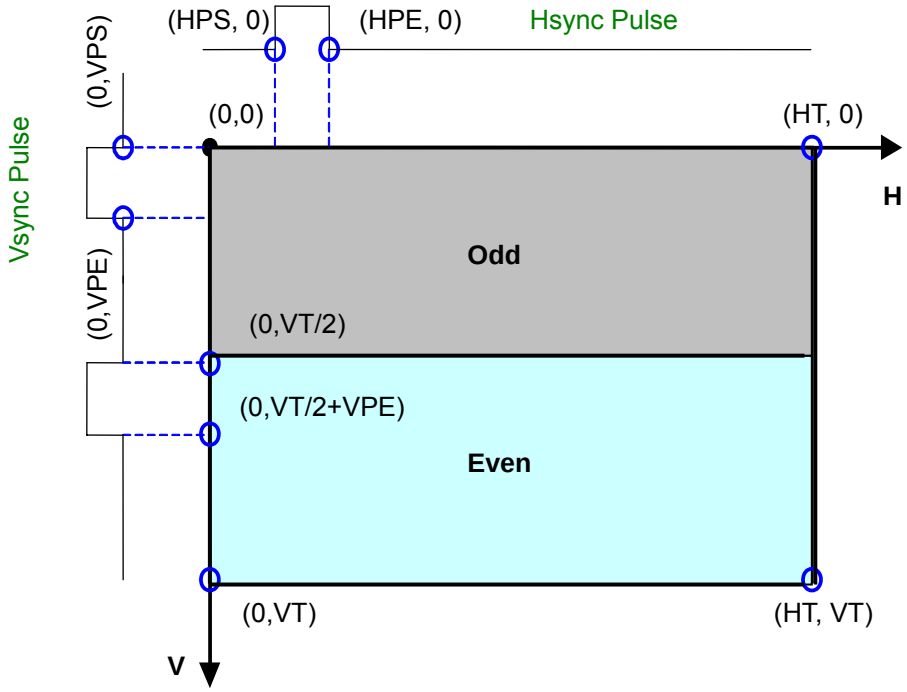


Figure 19-6 TV-Encoder Display Parameters

NOTES:

1 Even Field contains one more blank line.
E.g. for standard PAL timing, Odd filed has 312 lines while even field has 313 lines.

2 Interlace mode generates 2 vsync pulses for each field. The second vsync starts at (VT/2), ends at (VT/2 + VPE).

3 Display Area & Virtual Area has the same size. VDS=HDS=0, VDE=VT, HDE=HT.

19.6 OSD Graphic

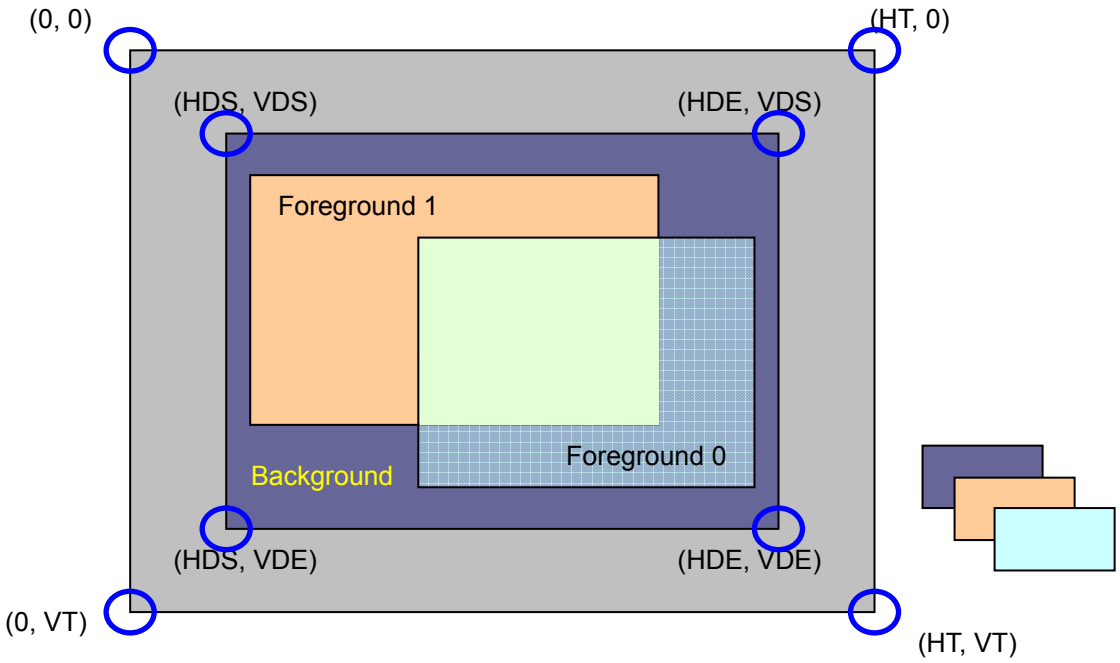


Figure 19-7 OSD Graphic

NOTES:

1 Background is one single color and the size is the full screen.

2 The size of foregrounds can be every size smaller than background.

3 The order of the graphic is as follows:

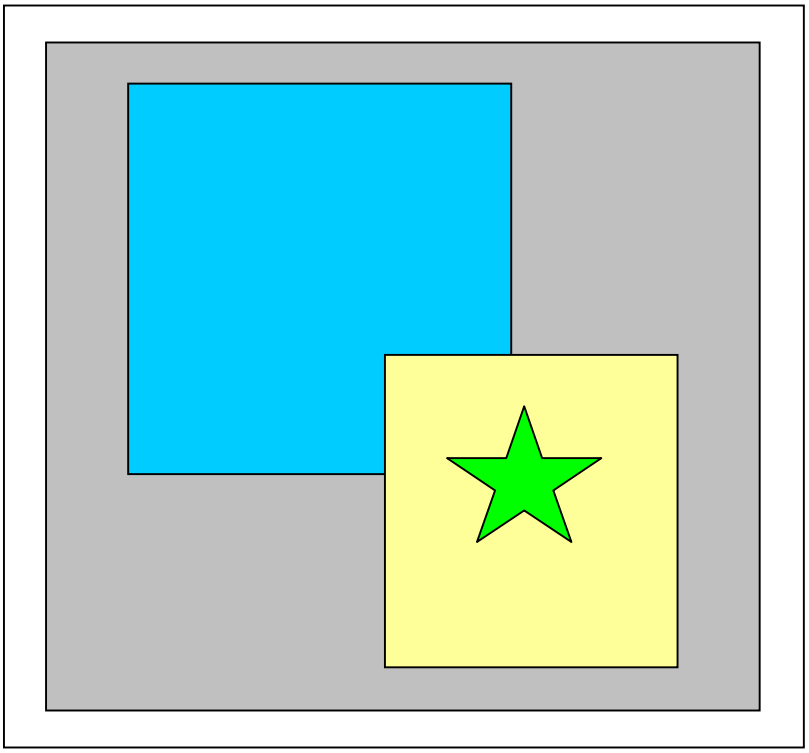
- a Top layer: Foreground 0
- b Middle layer: Foreground 1
- c Bottom layer: Background

19.6.1 Color Key

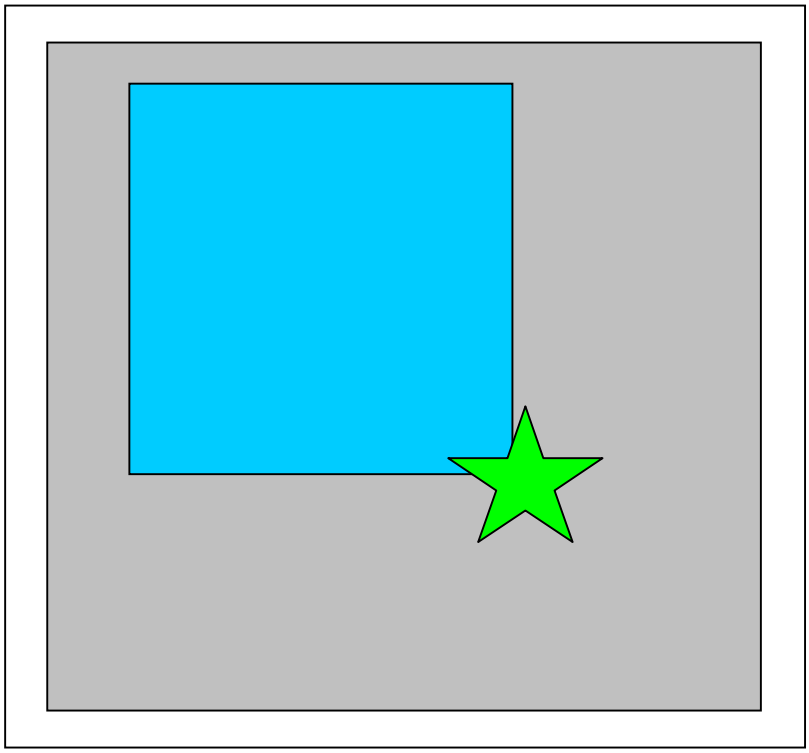
This function gives user a method to implement irregular display window. User can make foreground 0 and foreground 1 to different shape. The color key has two implements mode that called color key and mask color key.

Color Key mode is mean to mask a chosen color and show others.

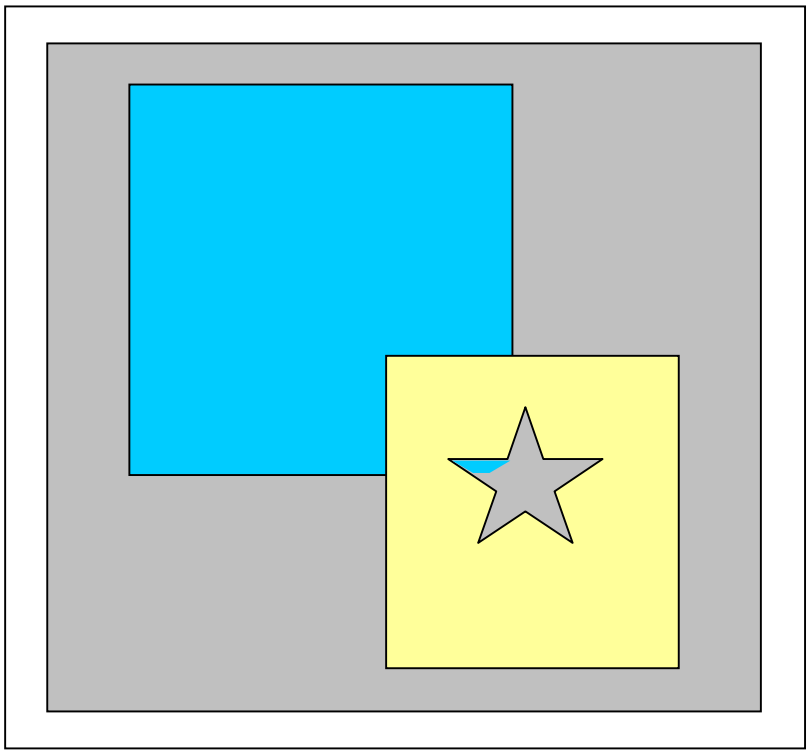
Mask Color Key mode is mean to only show a chosen color and mask others.



Not use color key function

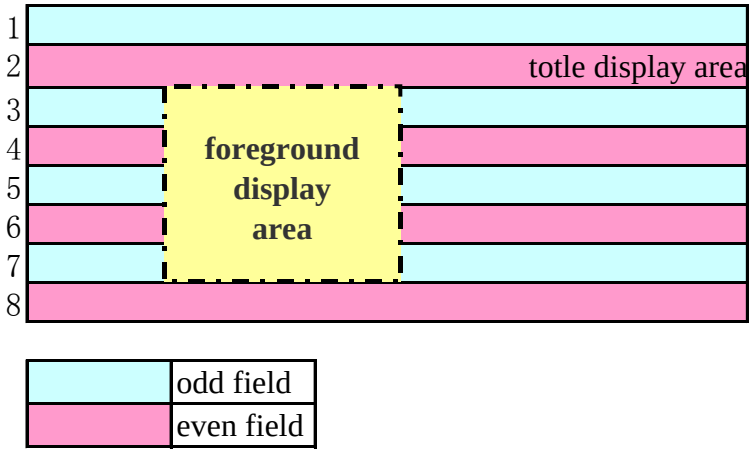


Color key mode



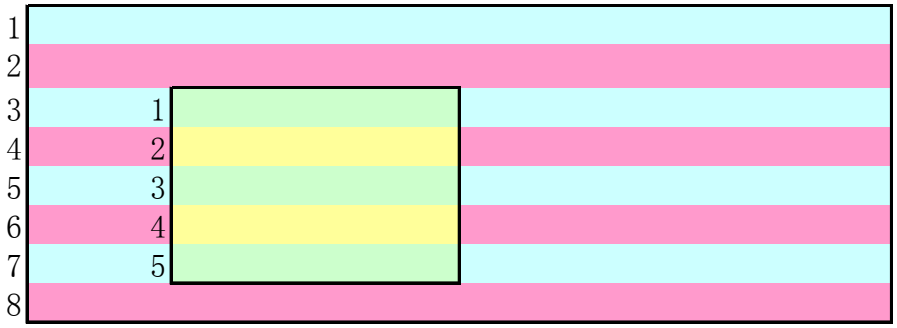
Mask color key mode

19.7 TV Graphic



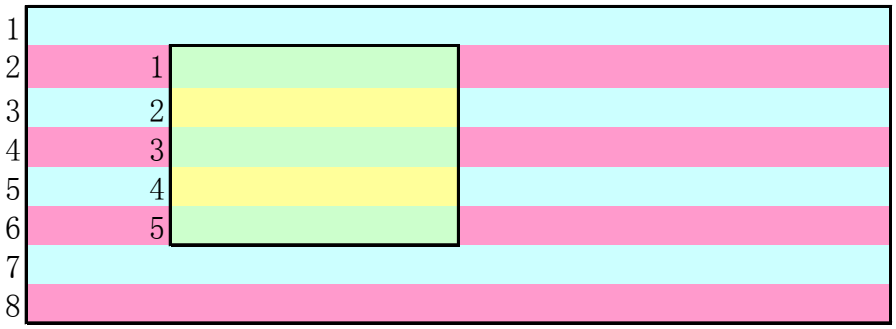
	odd field
	even field

19.7.1 Different Display Field



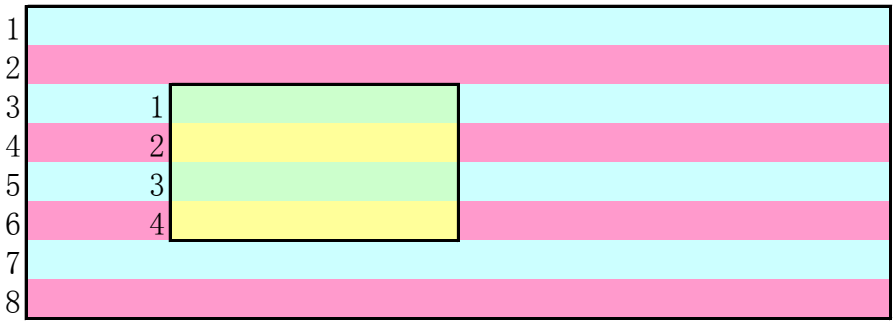
	foreground data odd field
	foreground data even field
	totle display area odd field
	totle display area even field

foreground data
odd field first, 3 line
even field, 2 line



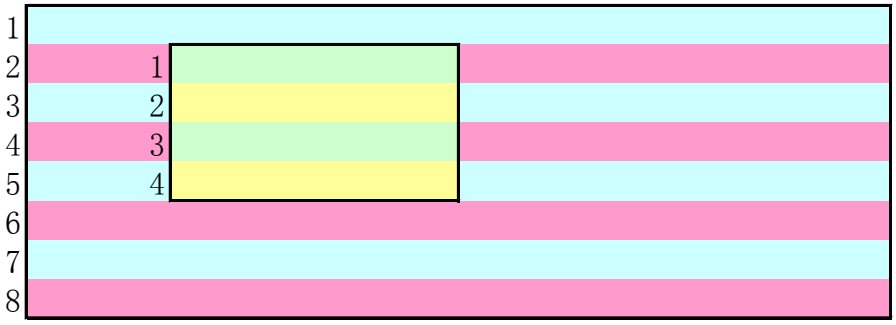
	foreground data odd field
	foreground data even field
	totle display area odd field
	totle display area even field

foreground data
even field first, 2 line
odd field, 3 line



	foreground data odd field
	foreground data even field
	totle display area odd field
	totle display area even field

foreground data
odd field first, 2 line
even field, 2 line



	foreground data odd field
	foreground data even field
	totle display area odd field
	totle display area even field

foreground data
even field first, 2 line
odd field, 2 line

19.8 Register Description

Table 19-2 LCD Controller Registers Description

Name	RW	Reset Value	Address	Access Size
LCDCFG	RW	0x00000000	0x13050000	32
LCDCTRL	RW	0x00000000	0x13050030	32
LCDSTATE	RW	0x00000000	0x13050034	32
LCDOSDC	RW	0x0000	0x13050100	16
LCDOSDCTRL	RW	0x0000	0x13050104	16
LCDOSDS	RW	0x0000	0x13050108	16
LCDBGC	RW	0x00000000	0x1305010C	32
LCDKEY0	RW	0x00000000	0x13050110	32
LCDKEY1	RW	0x00000000	0x13050114	32
LCDALPHA	RW	0x00	0x13050118	8
LCDIPUR	RW	0x00000000	0x1305011C	32
LCDRGBC	RW	0x0000	0x13050090	16
LCDVAT	RW	0x00000000	0x1305000C	32
LCDDAH	RW	0x00000000	0x13050010	32
LCDDAV	RW	0x00000000	0x13050014	32
LCDXYP0	RW	0x00000000	0x13050120	32
LCDXYP1	RW	0x00000000	0x13050124	32
LCDSIZE0	RW	0x00000000	0x13050128	32
LCDSIZE1	RW	0x00000000	0x1305012C	32
LCDVSYNC	RW	0x00000000	0x13050004	32
LCDHSYNC	RW	0x00000000	0x13050008	32
LCDPS[*1]	RW	0x00000000	0x13050018	32
LCDCLS[*1]	RW	0x00000000	0x1305001C	32
LCDSPL[*1]	RW	0x00000000	0x13050020	32
LCDREV[*1]	RW	0x00000000	0x13050024	32
LCDIID	R	0x00000000	0x13050038	32
LCDDA0	RW	0x00000000	0x13050040	32
LCDSA0	R	0x00000000	0x13050044	32
LCDFID0	R	0x00000000	0x13050048	32
LCDCMD0	R	0x00000000	0x1305004C	32
LCDOFFS0	R	0x00000000	0x13050060	32
LCDPW0	R	0x00000000	0x13050064	32
LCDCNUM0	R	0x00000000	0x13050068	32
LCDDESSIZE0	R	0x00000000	0x1305006C	32
LCDDA1[*2]	RW	0x00000000	0x13050050	32
LCDSA1[*2]	R	0x00000000	0x13050054	32

LCDFID1[*2]	R	0x00000000	0x13050058	32
LCDCMD1[*2]	R	0x00000000	0x1305005C	32
LCDOFFS1[*2]	R	0x00000000	0x13050070	32
LCDPW1[*2]	R	0x00000000	0x13050074	32
LCDCNUM1[*2]	R	0x00000000	0x13050078	32
LCDDESSIZE1[*2]	R	0x00000000	0x1305007C	32
LCDXYP0_PART2	RW	0x00000000	0x130501F0	32
LCDSIZE0_PART2	RW	0x00000000	0x130501F4	32
LCDDA0_PART2	RW	0x00000000	0x130501C0	32
LCDSA0_PART2	R	0x00000000	0x130501C4	32
LCDFID0_PART2	R	0x00000000	0x130501C8	32
LCDCMD0_PART2	R	0x00000000	0x130501CC	32
LCDOFFS0_PART 2	R	0x00000000	0x130501E0	32
LCDPW0_PART2	R	0x00000000	0x130501E4	32
LCDCNUM0_PART 2[*3]	R	0x00000000	0x130501E8	32
LCDDESSIZE0_PA RT2	R	0x00000000	0x130501EC	32

NOTES:

1 [*1]: These registers are only used for SPECIAL TFT panels.

2 [*2]: These registers are only used for Dual Panel STN panels and use DMA channel 1 in OSD mode for TFT panels.

19.8.1 Configure Register (LCDCFG)

LCDCFG **0x13050000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	LCDPIN	TVEPEH		NEWDES	PALBP	TVEN	RECOVER	DITHER	PSM	CLSM	SPLM	REVM	HSYNM	PCLKM	INVDAT	SYNDIR	PSP	CLSP	SPLP	REVP	HSP	PCP	DEP	VSP	18/16	24	PDW	MODE				
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	LCDPIN[*1]	LCD PIN Select bit. It is used to choose the function of LCD PINS or SLCD PINS. The function of pins is as follows. **LCDPIN** / **PIN SELECT** 0 / LCD PIN 1 / SLCD PIN	RW

30	TVEPEH	TVE PAL enable extra_halfline signal.	RW
29		**KEEP THIS BIT TO 0.**	RW
28	NEWDES	indicate use new 8 words descriptor or not. 0: use old 4 words descriptor 1: use new 8 words descriptor (add LCDOFFSx, LCDPWx, LCDCUNMx, LCDDESSIZEx) OSD mode use 8 word descriptor.	RW
27	PALBP	Indicate bypass pal in BPP8, and in OSD mode, set this bit to 1 is also bypass data format and alpha blending. 0: use PAL; 1: not use PAL.	RW
26	TVEN	Indicate the terminal is LCD panel or TV.	RW
25	RECOVER	Auto recover when output FIFO under run. 0: disable, 1: enable.	RW
24	DITHER	Dither function. (use when 24bpp data output to a 18/16bit panel) 0: disable, 1: enable. Dither function use to make the picture misty, when you show a static picture with few color, strongly recommend you not use it. When you use this function both static and dynamic picture, strongly recommend you to set the static picture with 16/18BPP color.	RW
23	PSM	PS signal mode bit. 1: disabled, 0: enabled.	RW
22	CLSM	CLS signal mode bit. 1: disabled, 0: enabled.	RW
21	SPLM	SPL signal mode bit. 1: disabled, 0: enabled.	RW
20	REVM	REV signal mode bit. 1: disabled, 0: enabled.	RW
19	HSYNM	H-Sync signal polarity choice function. 1: disabled, 0: enabled.	RW
18	PCLKM	Dot clock signal polarity choice function. 1: disabled, 0: enabled.	RW
17	INVDAT	Inverse output data. 0: normal, 1: inverse.	RW
16	SYNDIR	V-Sync and H-Sync direction. 0: output, 1: input.	RW
15	PSP	PS pin reset state.	RW
14	CLSP	CLS pin reset state.	RW
13	SPLP	SPL pin reset state.	RW
12	REVP	REV pin reset state.	RW
11	HSP	H-Sync polarity. 0: active high, 1: active low.	RW
10	PCP	Pix-clock polarity. 0: data translations at rising edge 1: data translations at falling edge	RW
9	DEP	Data Enable polarity. 0 : active high; 1: active low.	RW
8	VSP	V-Sync polarity. 0: leading edge is rising edge 1: leading edge is falling edge	RW
7	18/16	18-bit TFT Panel or 16-bit TFT Panel. This bit will be available when MODE [3:2] is equal to 0 and 24[6] is equal to 0. 0: 16-bit TFT Panel 1: 18-bit TFT Panel	RW

<table>
<tr><td>6</td><td>24</td><td>Set this bit to 1 for 24-bit TFT Panel .</td><td>RW</td></tr>
<tr><td>5:4</td><td>PDW</td><td>STN pins utilization.
<table>
<tr><th></th><th>Signal Panel</th></tr>
<tr><td>00</td><td>Lcd_d[0]</td></tr>
<tr><td>01</td><td>Lcd_d[0:1]</td></tr>
<tr><td>10</td><td>Lcd_d[0:3]</td></tr>
<tr><td>11</td><td>Lcd_d[0:7]</td></tr>
<tr><td></td><td>Dual-Monochrome Panel</td></tr>
<tr><td>00</td><td>Reserved</td></tr>
<tr><td>01</td><td>Reserved</td></tr>
<tr><td>10</td><td>Upper panel: lcd_d[3:0], lower panel: lcd_d[11:8]</td></tr>
<tr><td>11</td><td>Upper panel: lcd_d[7:0], lower panel: lcd_d[15:8]</td></tr>
</table></td><td>RW</td></tr>
<tr><td>3:0</td><td>MODE</td><td>Display Device Mode Select/Output mode.
<table>
<tr><th></th><th>LCD Panel</th></tr>
<tr><td>0000</td><td>Generic 16-bit/18-bit Parallel TFT Panel</td></tr>
<tr><td>0001</td><td>Special TFT Panel Mode1</td></tr>
<tr><td>0010</td><td>Special TFT Panel Mode2</td></tr>
<tr><td>0011</td><td>Special TFT Panel Mode3</td></tr>
<tr><td>0100</td><td>Non-Interlaced TV out</td></tr>
<tr><td>0101</td><td>Reserved</td></tr>
<tr><td>0110</td><td>Interlaced TV out</td></tr>
<tr><td>0111</td><td>Reserved</td></tr>
<tr><td>1000</td><td>Single-Color STN Panel</td></tr>
<tr><td>1001</td><td>Single-Monochrome STN Panel</td></tr>
<tr><td>1010</td><td>Dual-Color STN Panel</td></tr>
<tr><td>1011</td><td>Dual-Monochrome STN Panel</td></tr>
<tr><td>1100</td><td>8-bit Serial TFT</td></tr>
<tr><td>1101</td><td>LCM</td></tr>
<tr><td>1110</td><td>Reserved</td></tr>
<tr><td>1111</td><td>Reserved</td></tr>
</table></td><td>RW</td></tr>
</table>

NOTES:

*1:

LCDPIN	PIN25	PIN24	PIN23	PIN22	PIN21	PIN20	PIN19	PIN18	PIN17-0
0	LCD PCLK	LCD VSYNC	LCD HSYNC	LCD DE	LCD REV	LCD PS	LCD CLS	LCD SPL	LCD D [17:0]
1	SLCD CLK	SLCD CS	SLCD RS						SLCD D [17:0]

1 The direction of PIN25 is set by register LPCDR.LCS in CPM SPEC.

2 The direction of PIN23 and PIN23 are set by register LCDCFG.SYNDIR.

19.8.2 Control Register (LCDCTRL)

LCDCTRL **0x13050030**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PINMD	Reserved	BST		RGB	OFUP	FRC		PDD									DACTE	EOFM	SOFM	OFUM	IFUM0	IFUM1	LDDM	QDM	BEDN	PEDN	DIS	ENA	BPP		
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	PINMD	This register set Pin distribution in 16-bit parallel mode. 0: 16-bit data correspond with LCD_D[15:0] 1: 16-bit data correspond with LCD_D[17:10], LCD_D[8:1]	RW
30	Reserved	These bits always read 0, and written are ignored.	R
29:28	BST	Burst Length Selection. 00: 4 word 01: 8 word 10: 16 word 11: 32 word	RW
27	RGB	Bpp16 RGB mode. 0: RGB565; 1: RGB555. In OSD mode, this bit configure the foreground 0. If use parallel 18 bit，set this bit to 0.	RW
26	OFUP	Output FIFO under run protection. 0: disable; 1: enable.	RW
25:24	FRC	STN FRC Algorithm Selection. 00: 16 grayscale 01: 4 grayscale 10: 2 grayscale 11: Reserved	RW
23:16	PDD	Load Palette Delay Counter.	RW
15		keep this bit to 0.	
14	DACTE	DAC loop back test.	RW
13	EOFM	Mask end of frame interrupt. 0: INT-disabled; 1:INT-enabled.	RW
12	SOFM	Mask start of frame interrupt. 0: INT-disabled; 1: INT-enabled.	RW
11	OFUM	Mask out FIFO under run interrupt. 0: INT-disabled; 1: INT-enabled.	RW
10	IFUM0	Mask in FIFO 0 under run interrupt. 0: INT-disabled; 1: INT-enabled.	RW
9	IFUM1	Mask in FIFO 1 under run interrupt. 0: INT-disabled; 1: INT-enabled.	RW
8	LDDM	Mask LCD disable done interrupt. 0: INT-disabled; 1: INT-enabled.	RW
7	QDM	Mask LCD quick disable done interrupt. 0: INT-disabled, 1:	RW

		INT-enabled.	
6	BEDN	Endian selection. 0: same as system Endian; 1: reverse endian format.	RW
5	PEDN	Endian in byte. 0: msb first; 1: lsb first.	RW
4	DIS	Disable controller indicate bit. 0: enable; 1: in disabling or disabled.	RW
3	ENA	Enable controller. 0: disable; 1: enable.	W
2:0	BPP	Bits Per Pixel. 000: 1bpp 001: 2bpp 010: 4bpp 011: 8bpp 100: 15/16bpp 101: 18bpp/24bpp 110: 24bpp compressed 111: 30bpp In OSD mode, those bits configure the foreground 0.	RW

19.8.3 Status Register (LCDSTATE)

LCDSTATE **0x13050034**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																									QD	Reserved	EOF	SOF	OUF	IFU0	IFU1	LDD
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7	QD	LCD Quick disable. 0: not been quick disabled; 1: quick disabled done.	RW
6	Reserved	These bits always read 0, and written are ignored.	R
5	EOF	End of Frame indicate bit.	RW
4	SOF	Start of Frame indicate bit.	RW
3	OUF	Out FIFO under run.	RW
2	IFU0	In FIFO 0 under run.	RW
1	IFU1	In FIFO 1 under run.	RW
0	LDD	LCD disable. 0: not been normal disabled; 1: been normal disabled.	RW

19.8.4 OSD Configure Register (LCDOSDC)

LCDOSDC **0x13050100**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SOFM1	EOFM1	Reserved		SOFM0	EOFM0	Reserved					F1EN	F0EN	ALPHAEN	ALPHAMD	OSDEN
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15	SOFM1	Start of frame interrupt mask for foreground 1.	RW
14	EOFM1	End of frame interrupt mask for foreground 1.	RW
13:12	Reserved	These bits always read 0, and written are ignored.	R
11	SOFM0	Start of frame interrupt mask for foreground 0.	RW
10	EOFM0	End of frame interrupt mask for foreground 0.	RW
9:5	Reserved	These bits always read 0, and written are ignored.	R
4	F1EN	1: Foreground 1 is enabled 0: Foreground 1 is disabled	RW
3	F0EN	1: Foreground 0 is enabled 0: Foreground 0 is disabled.*When use slcd, F0EN must set 1.	RW
2	ALPHAEN	1: Alpha blending is enabled 0: Alpha blending is disabled	RW
1	ALPHAMD	Alpha blending mode. 0: One transparency for the whole graphic, and the LCDALPHA register is used for transparency 1: One transparency for each pixel in one graphic, and the alpha value is coming from each pixel data	RW
0	OSDEN	OSD mod enable. 1: enabled. And you can use F0 F1 0: disabled	RW

19.8.5 OSD Control Register (LCDOSDCTRL)

LCDOSDCTRL **0x13050104**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	IPU	Reserved										RGB	CHANGES	OSDBPP		
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15	IPU	Indicate use IPU or DMA channel 1 to transport data to FIFO 1. This bit is only use in OSD mode. 0: use DMA channel 1 1: use IPU	RW
13:5	Reserved	These bits always read 0, and written are ignored.	
4	OSDRGB	Bpp16 RGB mode. 0: RGB565;1: RGB555. This bit only use in OSD mode to configure foreground 1.	RW
3	CHANGES	Change configure flag, when software need change the foreground0 and foreground1's enable/position/size, it need set this bit to 1. When hardware finishes the change, It will clear this bit to 0. DO NOT set this bit when you needed change size or position. AND make sure the reconfigure value is different to the old one. Only one of these (F0's position, F1's position, F0's size, F1's size) could be change in one time. Refer to 1.14.6.	RW
2:0	OSDBPP	Bits Per Pixel of OSD channel 1. (this channel cannot use palette) 000 Reserved 001 Reserved 010 Reserved 011 Reserved 100 15/16bpp 101 18bpp/24bpp 110 24bpp compressed 111 30bpp Those bits only use in OSD mode to configure display window 1.	RW

19.8.6 OSD State Register (LCDOSDS)

LCDOSDS **0x13050108**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SOF1	EOF1	Reserved		SOF0	EOF0	Reserved									READY
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15	SOF1	Start of frame flag for foreground 1.	RW
14	EOF1	End of frame flag for foreground 1.	RW
13:12	Reserved	These bits always read 0, and written are ignored.	R
11	SOF0	Start of frame flag for foreground 0.	RW

10	EOF0	End of frame flag for foreground 0.	RW
9:1	Reserved	These bits always read 0, and written are ignored.	R
0	READY	Ready for accept the change. When this bit set 1, the software can change the descriptor's LCDDESSIZE0, 1 to change the foreground size. This bit will clear by hardware when the change is finished.	R

19.8.7 Background Color Register (LCDBGC)

LCDBGC **0x1305010C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								Red [7:0]								Green [7:0]								Blue [7:0]							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:27	Reserved	These bits always read 0, and written are ignored.	R
23:16	Red	Red part or Y part of background.	RW
15:8	Green	Green part or Cb part of background.	RW
7:0	Blue	Blue part or Cr part of background	RW

19.8.8 Foreground Color Key Register 0 (LCDKEY0)

LCDKEY0 **0x13050110**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	KEYEN	KEYMD	Reserved						Red [7:0]								Green [7:0]								Blue [7:0]							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	KEYEN	The enable bit of color key for foreground 0.	RW
30	KEYMD	Color key mod. 0: color key; 1: mask color key.	RW
29:27	Reserved	These bits always read 0, and written are ignored.	R
23:16	Red	Red part of color key for foreground 0.	RW
15:8	Green	Green part of color key for foreground 0.	RW
7:0	Blue	Blue part of color key for foreground 0.	RW

19.8.9 Foreground Color Key Register 1 (LCDKEY1)

LCDKEY1 **0x13050114**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	KEYEN	KEYMD	Reserved						Red [7:0]								Green [7:0]								Blue [7:0]							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	KEYEN	The enable bit of color key for foreground 1.	RW
30	KEYMD	Color key mod. 0: color key; 1: mask color key.	RW
29:27	Reserved	These bits always read 0, and written are ignored.	R
23:16	Red	Red part of color key for foreground 1.	RW
15:8	Green	Green part of color key for foreground 1.	RW
7:0	Blue	Blue part of color key for foreground 1.	RW

19.8.10 ALPHA Register (LCDALPHA)

LCDALPHA **0x13050118**

Bit	7	6	5	4	3	2	1	0
	ALPHA [7:0]							
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:0	ALPHA	The alpha value for one graphic with one transparency.	RW

The formula of alpha blending is as follows:

$$NewPixel = \frac{[(256 - Alpha) * (Foreground1\_or\_background) + Alpha * Froeground0 + 128]}{256}$$

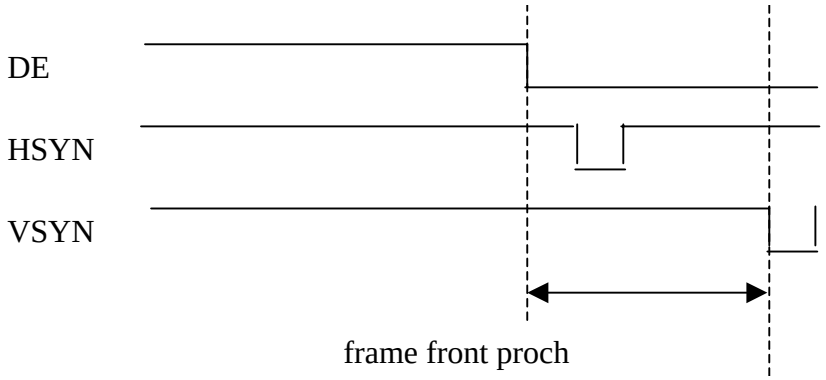
Note that foreground 1 must be overlay background.

19.8.11 IPU Restart (LCDIPUR)

LCDIPUR 0x1305011C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	IPUREN	Reserved							IPUR																							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	IPUREN	IPU restart function enable. 0:disable; 1:enable.	RW
30:24	Reserved	These bits always read 0, and written are ignored.	RW
23:0	IPUR	This register is indicating when one frame is end, how long the panel can wait for the next frame data from IPU. In common, set this number larger than frame front porch and near to ((HT-0) X (VPE-VPS))/3. This signal only use when foreground1 work in IPU mode. Trigger IPU transfer the last frame again to avoid output FIFO under run.	RW



19.8.12 RGB Control (LCDRGBC)

LCDRGBC 0x13050090

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RGBDM	DMM	Reserved					YCC	Reserved	OddRGB			Reserved	EvenRGB		
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15	RGBDM	RGB with dummy data enable. Only useful for RGB serial mode. If this bit set to 1, the one pixel	RW

<table>
<tr><td></td><td></td><td>include 4 clock periods, that Red, Green, Blue and Dummy data.
Dummy is equal to 0.
0: Disable; 1: Enable.</td><td></td></tr>
<tr><td>14</td><td>DMM</td><td>RGB dummy mode.
0: R-G-B-Dummy
1: Dummy-R-G-B</td><td></td></tr>
<tr><td>13:9</td><td>Reserved</td><td>These bits always read 0, and written are ignored.</td><td>RW</td></tr>
<tr><td>8</td><td>YCC</td><td>Change RGB to YCbCr.
0: not change; 1: change to YUV.
This bit only use in OSD mode. Change RGB data to YCbYCr and sent to TV encoder.
Please notice that the data will be translated as 16 bits parallel. And only half of it will be transfer. (YCb or YCr in one pixel). If you not use OSD mode and TV encoder, please set this bit to 0.
When use this function with IPU transfer data to an interlaced TV, please set IPU output as RGB 888, and OSDBPP to 24. or IPU output data as PACKAGE(YCbYCr) and OSDBPP to 16.</td><td>RW</td></tr>
<tr><td>7</td><td>Reserved</td><td>These bits always read 0, and written are ignored.</td><td>RW</td></tr>
<tr><td>6:4</td><td>OddRGB</td><td>Odd line serial RGB data arrangement, useful for RGB serial mode only. *Please notice that you must set 000 when use 16/18parallel mode.
<table>
<tr><th></th><th>RGB mode</th></tr>
<tr><td>000</td><td>RGB</td></tr>
<tr><td>001</td><td>RBG</td></tr>
<tr><td>010</td><td>GRB</td></tr>
<tr><td>011</td><td>GBR</td></tr>
<tr><td>100</td><td>BRG</td></tr>
<tr><td>101</td><td>BGR</td></tr>
<tr><td>110</td><td>Reserved</td></tr>
<tr><td>111</td><td>Reserved</td></tr>
</table></td><td>RW</td></tr>
<tr><td>3</td><td>Reserved</td><td>These bits always read 0, and written are ignored.</td><td>RW</td></tr>
<tr><td>2:0</td><td>EvenRGB</td><td>Even line serial RGB data arrangement, useful for RGB serial mode only. *Please notice that you must set 000 when use 16/18parallel mode.
<table>
<tr><th></th><th>RGB mode</th></tr>
<tr><td>000</td><td>RGB</td></tr>
<tr><td>001</td><td>RBG</td></tr>
<tr><td>010</td><td>GRB</td></tr>
<tr><td>011</td><td>GBR</td></tr>
<tr><td>100</td><td>BRG</td></tr>
<tr><td>101</td><td>BGR</td></tr>
<tr><td>110</td><td>Reserved</td></tr>
</table></td><td>RW</td></tr>
</table>

			111	Reserved		

19.8.13 Virtual Area Setting (LCDVAT)

LCDVAT **0x1305000C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				HT												Reserved				VT											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	HT	Horizontal Total size. (in dot clock, sum of display area and blank space)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	VT	Vertical Total size. (in line clock, sum of display area and blank space)	RW

19.8.14 Display Area Horizontal Start/End Point (LCDDAH)

LCDDAH **0x13050010**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				HDS												Reserved				HDE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	HDS	Horizontal display area start. (in dot clock)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	HDE	Horizontal display area end. (in dot clock)	RW

19.8.15 Display Area Vertical Start/End Point (LCDDAV)

LCDDAV **0x13050014**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				VDS												Reserved				VDE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	VDS	Vertical display area start position. (in line clock)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	VDE	Vertical display area end position. (in line clock)	RW

19.8.16 Foreground 0 XY Position Register (LCDXYP0)

LCDXYP0 **0x13050120**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				YPOS												Reserved				XPOS											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	YPOS	The Y position of top-left part for foreground 0.	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	XPOS	The X position of top-left part for foreground 0.	RW

19.8.17 Foreground 0 PART2 XY Position Register (LCDXYP0_PART2)

LCDXYP0_PART2 **0x130501F0**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				YPOS												Reserved				XPOS											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	YPOS	The Y position of top-left part for foreground 0 PART2.	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	XPOS	The X position of top-left part for foreground 0 PART2.	RW

19.8.18 Foreground 1 XY Position Register (LCDXYP1)

LCDXYP1 **0x13050124**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				YPOS												Reserved				XPOS											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	YPOS	The Y position of top-left part for foreground 1.	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	XPOS	The X position of top-left part for foreground 1.	RW

19.8.19 Foreground 0 Size Register (LCDSIZE0)

LCDSIZE0 **0x13050128**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Height												Reserved				Width											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	Height	The height of foreground 0.	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	Width	The width of foreground 0.	RW

When use TVE interlaced mode, please set the area of F0 and F1 aligned with BST.

19.8.20 Foreground 0 PART2 Size Register (LCDSIZE0_PART2)

LCDSIZE0_PART2 **0x130501F4**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Height												Reserved				Width											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	Height	The height of foreground 0 PART2.	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	Width	The width of foreground 0 PART2.	RW

19.8.21 Foreground 1 Size Register (LCDSIZE1)

LCDSIZE1 **0x1305012C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				Height												Reserved				Width											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	Height	The height of foreground 1.	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	Width	The width of foreground 1.	RW

19.8.22 Vertical Synchronize Register (LCDVSYNC)

LCDVSYNC **0x13050004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				VPS												Reserved				VPE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	VPS	V-Sync Pulse start position, fixed to 0. (in line clock)	R
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	VPE	V-Sync Pulse end position. (in line clock)	RW

19.8.23 Horizontal Synchronize Register (LCDHSYNC)

LCDHSYNC 0x13050008

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				HPS												Reserved				HPE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	HPS	H-Sync pulse start position. (in dot clock)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	HPE	H-Sync pulse end position. (in dot clock)	RW

19.8.24 PS Signal Setting (LCDPS)

LCDPS 0x13050018

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				PSS												Reserved				PSE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	PSS	PS signal start position. (in dot clock) In STN mode, PS signal is ignored. But this register is used to define the AC BIAS signal. AC BIAS signal will toggle very N lines per frame. PSS defines the Toggle position.	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	PSE	PS signal end position. (in dot clock) In STN mode, PSE defines N, which described in PSS.	RW

19.8.25 CLS Signal Setting (LCDCLS)

LCDCLS **0x1305001C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				CLSS												Reserved				CLSE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	CLSS	CLS signal start position. (in dot clock)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	CLSE	CLS signal end position. (in dot clock)	RW

19.8.26 SPL Signal Setting (LCDSPL)

LCDSPL **0x13050020**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				SPLS												Reserved				SPLE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	SPLS	SPL signal start position. (in dot clock)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	SPLE	SPL signal end position. (in dot clock)	RW

In test mode this register use to keep TV encoder module's output data: comp_luma([25:16]) and chroma([9:0]).

19.8.27 REV Signal Setting (LCDREV)

LCDREV **0x13050024**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				REVS												Reserved															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	REVS	REV signal start position. (in dot clock)	RW
15:0	Reserved	These bits always read 0, and written are ignored.	R

19.8.28 Interrupt ID Register (LCDIID)

LCDIID is a read-only register that contains a copy of the Frame ID register (LCDFID) from the descriptor currently being processed when a start of frame (SOF) or end of frame (EOF) interrupt is generated. LCDIID is written to only when an unmasked interrupt of the above type is signaled and there are no other unmasked interrupts in the LCD controller pending. As such, the register is considered to be sticky and will be overwritten only when the signaled interrupt is cleared by writing the LCD controller status register. For dual-panel displays, LCDIID is written only when both channels have reached a given state.

LCDIID is written with the last channel to reach that state. (i.e. LCDFID of the last channel to reach SOF would be written in LCDIID if SOF interrupts are enabled). Reserved bits must be written with zeros and reads from them must be ignored.

LCDIID **0x13050038**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	IID																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	IID	A copy of Frame ID register, which transferred from Descriptor.	RW

19.8.29 Descriptor Address Register0, 1 (LCDDA0, 1, LCDDA0_PART2)

A frame descriptor is a 4-word block, aligned on 4-word (16-byte) boundary, in external memory:

WORD [0] contains the physical address for next LCDDAx
WORD [1] contains the physical address for LCDSAx
WORD [2] contains the value for LCDFIDx
WORD [3] contains the value for LCDCMDx

Software must write the physical address of the first descriptor to LCDDAx before enabling the LCD Controller. Once the LCD Controller is enabled, the first descriptor is read, and all 4 registers are written by the DMAC. The next frame descriptor pointed to by LCDDAx is loaded into the registers for the associated DMA channel after all data for the current descriptor has been transferred.

NOTES:

1 If only one frame buffer is used in external memory, the LCDDAx field (word [0] of the frame descriptor) must point back to itself. That is to say, the value of LCDDAx is the physical address of itself.

Read/write registers LCDDA0 and LCDDA1, corresponding to DMA channels 0 and 1, contain the physical address of the next descriptor in external memory. The DMAC fetches the descriptor at this location after finishing the current descriptor. On reset, the bits in this register are zero. The target address must be aligned to 16-byte boundary. Bits [3:0] of the address must be zero.

LCDDA0, LCDDA1, LCDDA0_PART2 **0x13050040, 0x13050050, 0x130501C0**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DA0, 1																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	DA0, 1	Next descriptor physical address. And descriptor structure as following: WORD [0]: next descriptor physical address WORD [1]: the buffer physical address WORD [2]: the buffer ID value (Only for debug) WORD [3]: the buffer property. The value is same as LCDCMD	RW

19.8.30 Source Address Register0, 1 (LCDSA0, 1, LCDSA0_PART2)

Registers LCDSA0 and LCDSA1, corresponding to DMA channels 0 and 1, contain the physical address of frame buffer or palette buffer in external memory. The address must be aligned on a 4, 8, or 16 word boundary according to register LCDCTRL.BST. If this descriptor is for palette data, LCDSA0 points to the memory location of the palette buffer. If this descriptor is for frame data, LCDSAx points to the memory location of the frame buffer. This address is incremented by hardware as the DMAC

fetches data from memory. If desired, the Frame ID Register can be used to hold the initial frame source address.

LCDSA0, LCDSA1, LCDSA0_PART2 **0x13050044, 0x13050054, 0x130501C4**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SA0, 1																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	SA0, 1	Buffer start address. (Only for driver debug)	R

19.8.31 Frame ID Register0，1 (LCDFID0, 1, LCDFID0_PART2)

Registers LCDFID0 and LCDFID1, corresponding to DMA channels 0 and 1, contain an ID field that describes the current frame. The particular use of this field is up to the software. This ID register is copied to the LCD Controller Interrupt ID Register when an interrupt occurs.

LCDFID0, LCDFID1, LCDFID0_PART2 **0x13050048, 0x13050058, 0x130501C8**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FID0, 1																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	FID0, 1	Frame ID. (Only for debug)	R

19.8.32 DMA Command Registers (LCDCMDx, LCDCMD0_PART2)

LCDCMD0, LCDCMD1, LCDCMD0_PART2 **0x1305004C, 0x1305005C, 0x130501CC**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SOFINT	EOFINT	CMD	PAL	Reserved												LEN															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	SOFINT	Enable start of frame interrupt. When SOFINT =1, the DMAC sets the start of frame bit	R

		(LCDSTATE.SOF) when starting a new frame. The SOF bit is set after a new descriptor is loaded from memory and before the palette/frame data is fetched. In dual-panel mode, LCDSTATE.SOF is set only when both channels reach the start of frame and both frame descriptors have SOFINT set. SOFINT must not be set for palette descriptors in dual-panel mode, since only one channel is ever used to load the palette descriptor.	
30	EOFINT	Enable end of frame interrupt. When EOFINT =1, the DMAC sets the end of frame bit (LCDSTATE.EOF) after fetching the last word in the frame buffer. In dual-panel mode, LCDSTATE.EOF is set only when both channels reach the end of frame and both frame descriptors have EOFINT set. EOFINT must not be set for palette descriptors in dual-panel mode, since only one channel is ever used to load the palette descriptor.	R
29	CMD	It is used to distinguish command and data in lcm mode. And it is only loaded via DMA channel 0. 1: The data is command 0: The data is data	R
28	PAL	The descriptor contains a palette buffer. PAL indicates that data being fetched will be loaded into the palette RAM. If PAL =1, the palette RAM data is loaded via DMA channel 0 as follows: In bpp1, 2, 4, 8 modes, software must load the palette at least once after enabling the LCD. In bpp16 mode, PAL must be 0.	R
27:24	Reserved	These bits always read 0, and written are ignored.	R
23:0	LEN	The buffer length value. (in WORD) The LEN bit field determines the number of bytes of the buffer size pointed by LCDSAx. LEN = 0 is not valid. DMAC fetch data according to LEN. Each time one or more word(s) been fetched, LEN is decreased automatically. Software can read LEN.	R

19.8.33 DMA OFFSIZE Registers (LCDOFFSx, LCDOFFS0_PART2)

LCDOFFS0, LCDOFFS1, LCDOFFS0_PART2 **0x13050060, 0x13050070, 0x130501E0**

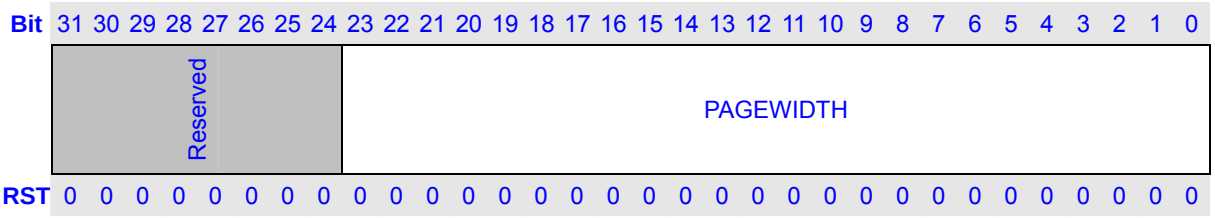
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								OFFSIZE																							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
23:0	OFFSIZE0, 1 OFFSIZE0_P	OFFSIZE value for DMA 0,1. Indicate the offset in word. *please notice that when you need OFFSIZE function, to set this reg	R

	ART2	to an un-zero value and also need to set LCDPW0, 1 to indicate how much word in one line of this frame.	

19.8.34 DMA Page Width Registers (LCDPWx, LCDPW0_PART2)

LCDPW0, LCDPW1, LCDPW_PART2 **0x13050064, 0x13050074, 0x130501E4**

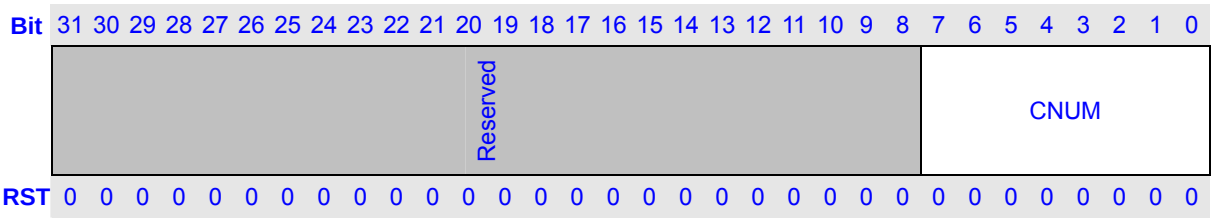


Bits	Name	Description	RW
23:0	PAGEWIDTH0, 1 PAGEWIDTH0_PART2	Page width for DMA 0,1. * When you set LCDOFFS.OFFSIZE0/1 to 0, you need not care the PAGEWIDTH0/1.	R

19.8.35 DMA Commend Counter Register0, 1 (LCDCNUM0,1)

When LCDCMD.CMD = 1, **0x13050068, 0x13050078** is use as LCDCNUM0, 1 LCDCNUM0_PART2 are not used now, set it to 0

LCDCNUM0, LCDCNUM1, LCDCNUM0_PART2 **0x13050068, 0x13050078, 0x130501E8**

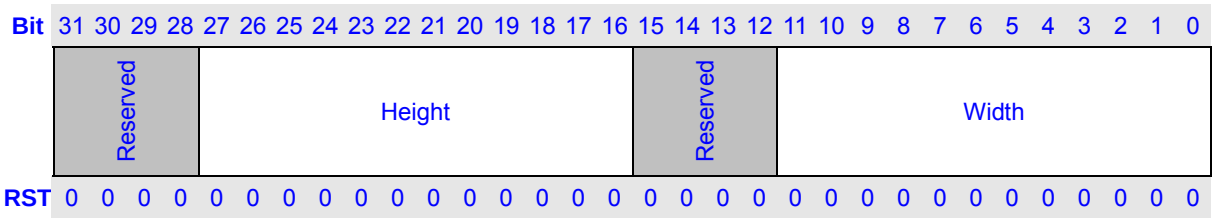


Bits	Name	Description	RW
7:0	CNUM0,1	Commands' number in this frame transfer by DMA. (only use in Smart LCD mode)	R

19.8.36 Foreground x Size in Descriptor (LCDDESSIZEx, LCDDESSIZE0_PART2)

When LCDCMD.CMD = 0, **0x1305006C, 0x1305007C** is use as LCDDESSIZE0, 1, to indicator the next frame foreground0, 1's size.

LCDDESSIZE0, 1, LCDDESSIZE0_PART2 **0x1305006C, 0x1305007C, 0x130501EC**



Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	Height	The height of foreground 0.	R
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	Width	The width of foreground 0.	R

19.9 LCD Controller Pin Mapping

There are several mapping schemes for different LCD panels.

19.9.1 TFT and CCIR Pin Mapping

Pin	Generic 8-bit Serial TFT	Generic 18/16-bit Parallel TFT		Special TFT 1 18/16-bit Parallel		Special TFT 2 18/16-bit Parallel		Special TFT 3 18/16-bit Parallel		CCIR656 8-bit	CCIR601 16-bit
Lcd_pclk/ Slcd_clk	CLK	CLK		DCLK		CLK		HCLK		CLK	CLK
Lcd_vsync/ Slcd_cs	VSYNC	VSYNC		SPS		GSRT		STV		VSYNC	VSYNC
Lcd_hsync/ Slcd_rs	HSYNC	HSYNC		LP		GPCK		STH		HSYNC	HSYNC
Lcd_de	DE	DE		-		-		-		-	-
Lcd_ps	-	-		Pulse		Toggle		Toggle		-	-
Lcd_cls	-	-		Pulse		Pulse		Pulse		-	-
Lcd_rev	-	-		Toggle		Toggle		Toggle		-	-
Lcd_spl	-	-		Pulse		Pulse		**Toggle**		-	-
Lcd_dat17	-	R5	-	R5	-	R5	-	R5	-	-	-
Lcd_dat16	-	R4	-	R4	-	R4	-	R4	-	-	-
Lcd_dat15	-	R3	R5	R3	R5	R3	R5	R3	R5	-	D15
Lcd_dat14	-	R2	R4	R2	R4	R2	R4	R2	R4	-	D14
Lcd_dat13	-	R1	R3	R1	R3	R1	R3	R1	R3	-	D13
Lcd_dat12	-	R0	R2	R0	R2	R0	R2	R0	R2	-	D12
Lcd_dat11	-	G5	R1	G5	R1	G5	R1	G5	R1	-	D11
Lcd_dat10	-	G4	G5	G4	G5	G4	G5	G4	G5	-	D10
Lcd_dat9	-	G3	G4	G3	G4	G3	G4	G3	G4	-	D9
Lcd_dat8	-	G2	G3	G2	G3	G2	G3	G2	G3	-	D8
Lcd_dat7	R7/G7/B7	G1	G2	G1	G2	G1	G2	G1	G2	D7	D7
Lcd_dat6	R6/G6/B6	G0	G1	G0	G1	G0	G1	G0	G1	D6	D6
Lcd_dat5	R5/G5/B5	B5	G0	B5	G0	B5	G0	B5	G0	D5	D5
Lcd_dat4	R4/G4/B4	B4	B5	B4	B5	B4	B5	B4	B5	D4	D4
Lcd_dat3	R3/G3/B3	B3	B4	B3	B4	B3	B4	B3	B4	D3	D3
Lcd_dat2	R2/G2/B2	B2	B3	B2	B3	B2	B3	B2	B3	D2	D2
Lcd_dat1	R1/G1/B1	B1	B2	B1	B2	B1	B2	B1	B2	D1	D1
Lcd_dat0	R0/G0/B0	B0	B1	B0	B1	B0	B1	B0	B1	D0	D0

TFT 24 bit parallel mode/16 bit parallel mode2

Pin	16 bit Parallel mode2	24 bit Parallel
Lcd_pclk/ Slcd_clk	CLK	CLK
Lcd_vsync/Slcd_cs	VSYNC	VSYNC
Lcd_hsync/Slcd_rs	HSYNC	HSYNC
Lcd_de	DE	DE
Lcd_ps	-	-
Lcd_cls	-	-
Lcd_rev	-	-
Lcd_spl	-	-
Lcd_dat17	R7	R7
Lcd_dat16	R6	R6
Lcd_dat15	R5	R5
Lcd_dat14	R4	R4
Lcd_dat13	R3	R3
Lcd_dat12	G7	R2
Lcd_dat11	G6	G7
Lcd_dat10	G5	G6
Lcd_dat9	0 (NC for panel)	G5
Lcd_dat8	G4	G4
Lcd_dat7	G3	G3
Lcd_dat6	G2	G2
Lcd_dat5	B7	B7
Lcd_dat4	B6	B6
Lcd_dat3	B5	B5
Lcd_dat2	B4	B4
Lcd_dat1	B3	B3
Lcd_dat0	0 (NC for panel)	B2
Lcd_lo6_o[5]	0	R1
Lcd_lo6_o[4]	0	R0
Lcd_lo6_o[3]	0	G1
Lcd_lo6_o[2]	0	G0
Lcd_lo6_o[1]	0	B1
Lcd_lo6_o[0]	0	B0

19.9.2 Single Panel STN Pin Mapping

Pin	Color STN	Mono STN			
	PDW=3	PDW=0	PDW=1	PDW=2	PDW=3
Lcd_pclk	CLK	CLK	CLK	CLK	CLK
Lcd_vsync	VSYNC	VSYNC	VSYNC	VSYNC	VSYNC
Lcd_hsync	HSYNC	HSYNC	HSYNC	HSYNC	HSYNC
Lcd_de	BIAS	BIAS	BIAS	BIAS	BIAS
Lcd_ps	-	-	-	-	-
Lcd_cls	-	-	-	-	-
Lcd_rev	-	-	-	-	-
Lcd_spl	-	-	-	-	-
Lcd_dat17	-	-	-	-	-
Lcd_dat16	-	-	-	-	-
Lcd_dat15	-	-	-	-	-
Lcd_dat14	-	-	-	-	-
Lcd_dat13	-	-	-	-	-
Lcd_dat12	-	-	-	-	-
Lcd_dat11	-	-	-	-	-
Lcd_dat10	-	-	-	-	-
Lcd_dat9	-	-	-	-	-
Lcd_dat8	-	-	-	-	-
Lcd_dat7	D7	-	-	-	D7
Lcd_dat6	D6	-	-	-	D6
Lcd_dat5	D5	-	-	-	D5
Lcd_dat4	D4	-	-	-	D4
Lcd_dat3	D3	-	-	D3	D3
Lcd_dat2	D2	-	-	D2	D2
Lcd_dat1	D1	-	D1	D1	D1
Lcd_dat0	D0	D0	D0	D0	D0

19.9.3 Dual Panel STN Pin Mapping

Pin	Color STN	Mono STN			
	PDW=3	PDW=0	PDW=1	PDW=2	PDW=3
Lcd_pclk	CLK	-	-	CLK	CLK
Lcd_vsync	VSYNC	-	-	VSYNC	VSYNC
Lcd_hsync	HSYNC	-	-	HSYNC	HSYNC
Lcd_de	BIAS	-	-	BIAS	BIAS
Lcd_ps	-	-	-	-	-
Lcd_cls	-	-	-	-	-
Lcd_rev	-	-	-	-	-
Lcd_spl	-	-	-	-	-
Lcd_dat17	-	-	-	-	-
Lcd_dat16	-	-	-	-	-
Lcd_dat15	UD7	-	-	-	UD7
Lcd_dat14	UD6	-	-	-	UD6
Lcd_dat13	UD5	-	-	-	UD5
Lcd_dat12	UD4	-	-	-	UD4
Lcd_dat11	UD3	-	-	UD3	UD3
Lcd_dat10	UD2	-	-	UD2	UD2
Lcd_dat9	UD1	-	-	UD1	UD1
Lcd_dat8	UD0	-	-	UD0	UD0
Lcd_dat7	LD7	-	-	-	LD7
Lcd_dat6	LD6	-	-	-	LD6
Lcd_dat5	LD5	-	-	-	LD5
Lcd_dat4	LD4	-	-	-	LD4
Lcd_dat3	LD3	-	-	LD3	LD3
Lcd_dat2	LD2	-	-	LD2	LD2
Lcd_dat1	LD1	-	-	LD1	LD1
Lcd_dat0	LD0	-	-	LD0	LD0

19.9.4 Data mapping to GPIO function.

pin name in LCD	mapping to GPIO function
Lcd_dat17/Slcd_dat17	lcd_r7
Lcd_dat16/Slcd_dat16	lcd_r6
Lcd_dat15/Slcd_dat15	lcd_r5
Lcd_dat14/Slcd_dat14	lcd_r4
Lcd_dat13/Slcd_dat13	lcd_r3
Lcd_dat12/Slcd_dat12	lcd_r2
Lcd_dat11/Slcd_dat11	lcd_g7
Lcd_dat10/Slcd_dat10	lcd_g6
Lcd_dat9/Slcd_dat9	lcd_g5
Lcd_dat8/Slcd_dat8	lcd_g4
Lcd_dat7/Slcd_dat7	lcd_g3
Lcd_dat6/Slcd_dat6	lcd_g2
Lcd_dat5/Slcd_dat5	lcd_b7
Lcd_dat4/Slcd_dat4	lcd_b6
Lcd_dat3/Slcd_dat3	lcd_b5
Lcd_dat2/Slcd_dat2	lcd_b4
Lcd_dat1/Slcd_dat1	lcd_b3
Lcd_dat0/Slcd_dat0	lcd_b2
Lcd_lo6_o[5]	lcd_r1
Lcd_lo6_o[4]	lcd_r0
Lcd_lo6_o[3]	lcd_g1
Lcd_lo6_o[2]	lcd_g0
Lcd_lo6_o[1]	lcd_b1
Lcd_lo6_o[0]	lcd_b0

19.10 Display Timing

19.10.1 General 16-bit and 18-bit TFT Timing

This section shows the general 16-bit and 18-bit TFT LCD timing diagram, the polarity of signal “Vsync”, “Hsync”, and “PCLK” can be programmed correspond to the LCD panel specification.

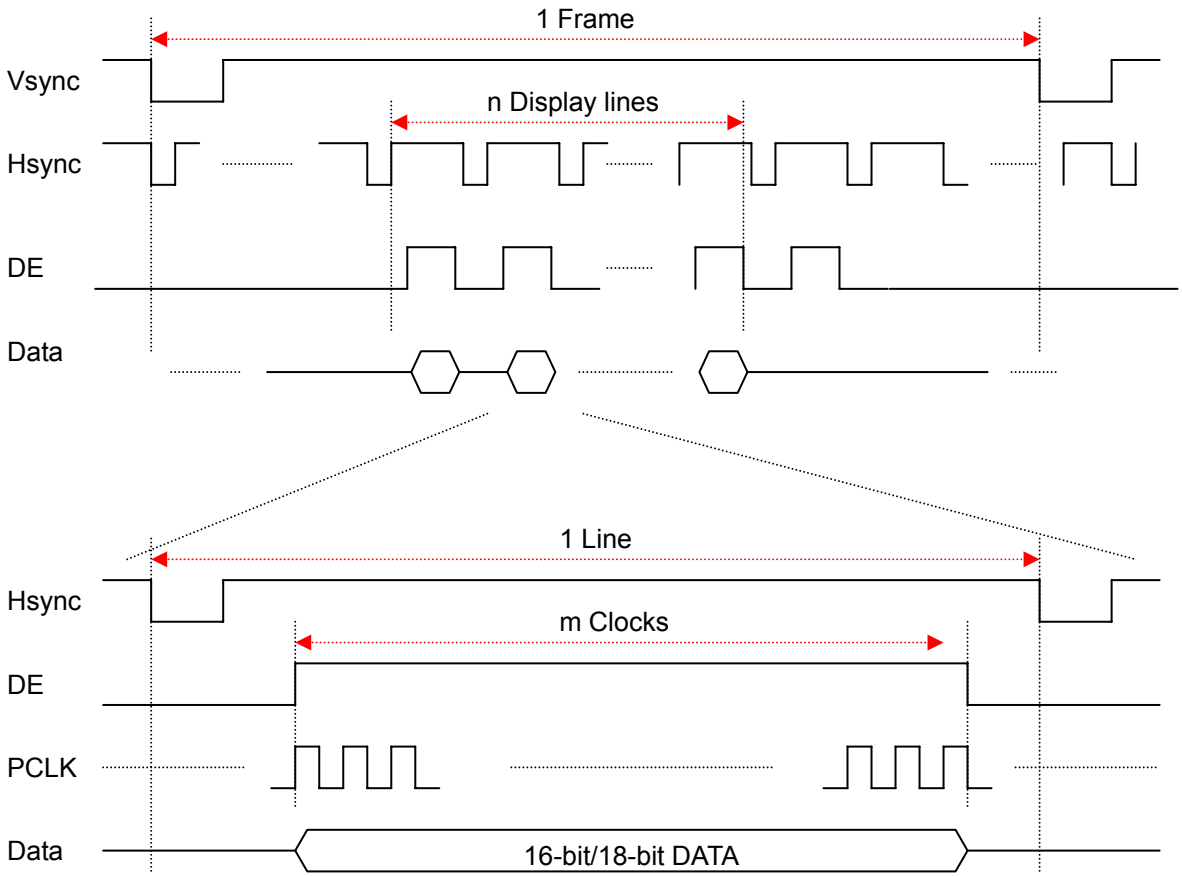


Figure 19-8 General 16-bit and 18-bit TFT LCD Timing

19.10.2 8-bit Serial TFT Timing

This section shows the 8-bit serial TFT LCD timing diagram, the polarity of signal “Vsync”, “Hsync”, and “PCLK” can be programmed correspond to the LCD panel specification.

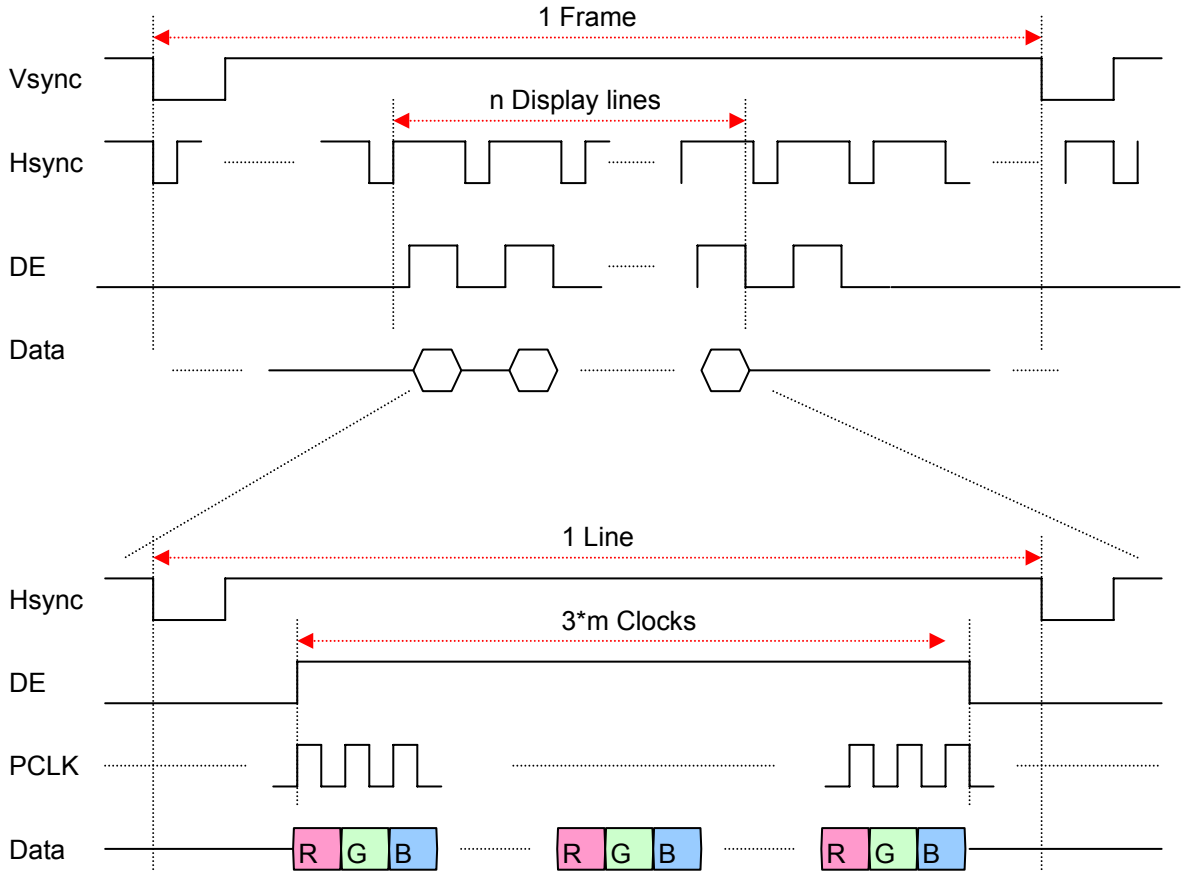


Figure 19-9 8-bit serial TFT LCD Timing (24bpp)

19.10.3 Special TFT Timing

Based on the general TFT LCD support, this controller also provides 4 special signals that can be programmed to general some special timing used for some panel. All 4 signals are worked in two modes: pulse mode and toggle mode. Signal “CLS” is fixed in pulse mode, and “REV” in toggle mode. The work mode of signals “SPL” and “PS” are defined in the special TFT LCD mode 1 to mode 3, either pulse mode or toggle mode. The position and polarity of these 4 signals can be programmed via registers. The Figures show the two modes as follows: (The toggle mode of signal “SPL” is different with the others signal. “SPL” does toggle after display line.)

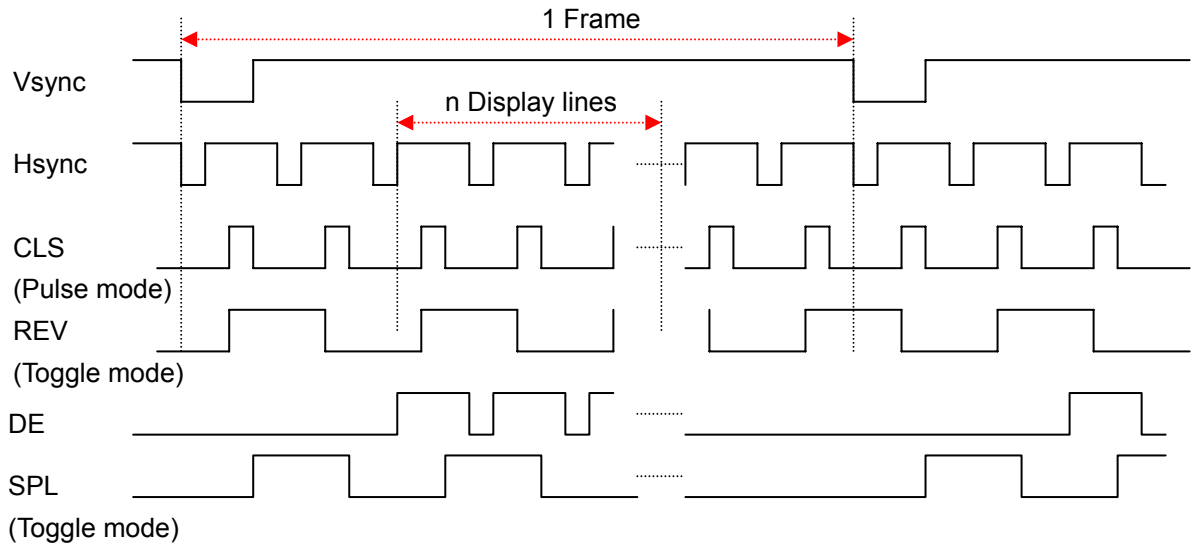


Figure 19-10 Special TFT LCD Timing 1

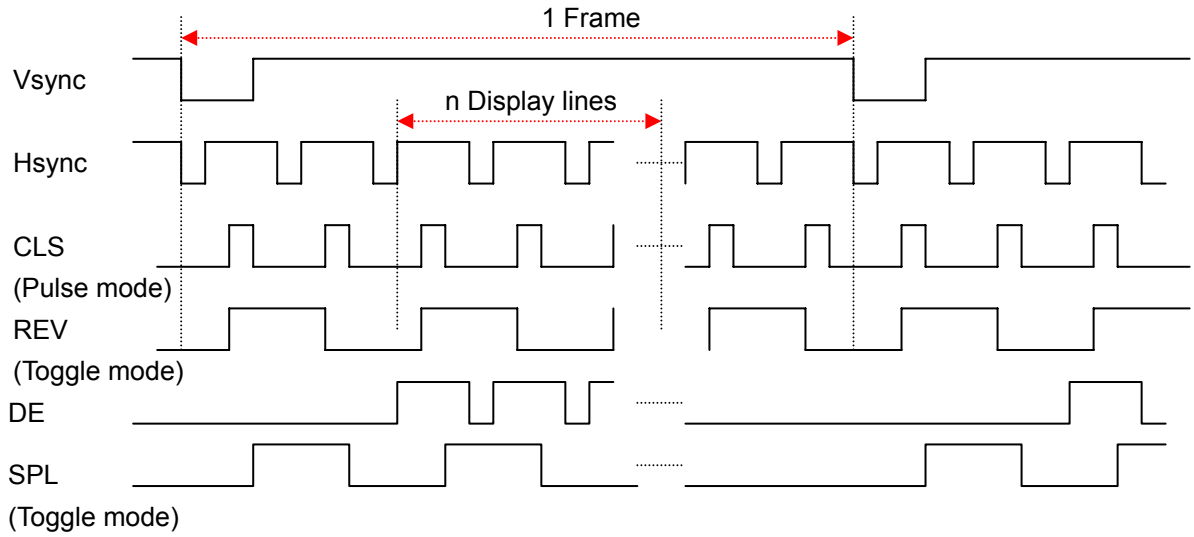


Figure 19-11 Special TFT LCD Timing 2

These two Figures show the timing of pulse mode and toggle mode, the pulse mode timing is same and the toggle mode timing is different. Timing 1 shows the condition when the total lines in 1 frame is

odd (the number of display is even and the number of blank is odd), so the phase of REV inverse at the first line of each frame and the phase of SPL dose not inverse at the first line of each frame. Timing 2 shows the condition when the total lines in 1 frame is even (the number of display is even and the number of blank is even), so the phase of REV and SPL dose not inverse at the first line of each frame.

When LCDC is enabled ,there will be a null line to be add before transferring data to LCD panel. So the toggle mode exept SPL signal of special 3 TFT mode is when reset level is high,the first valid edge will be rising edge. SPL signal of special 3 TFT mode is when reset level is high,the first valid edge will be falling edge.

19.10.4 Delta RGB panel timing

This section shows the Delta RGB timing diagram, the polarity of signal “Vsync”, “Hsync”, and “PCLK” can be programmed. And the odd/even line RGB order also can be programmed correspond to the LCD panel specification.

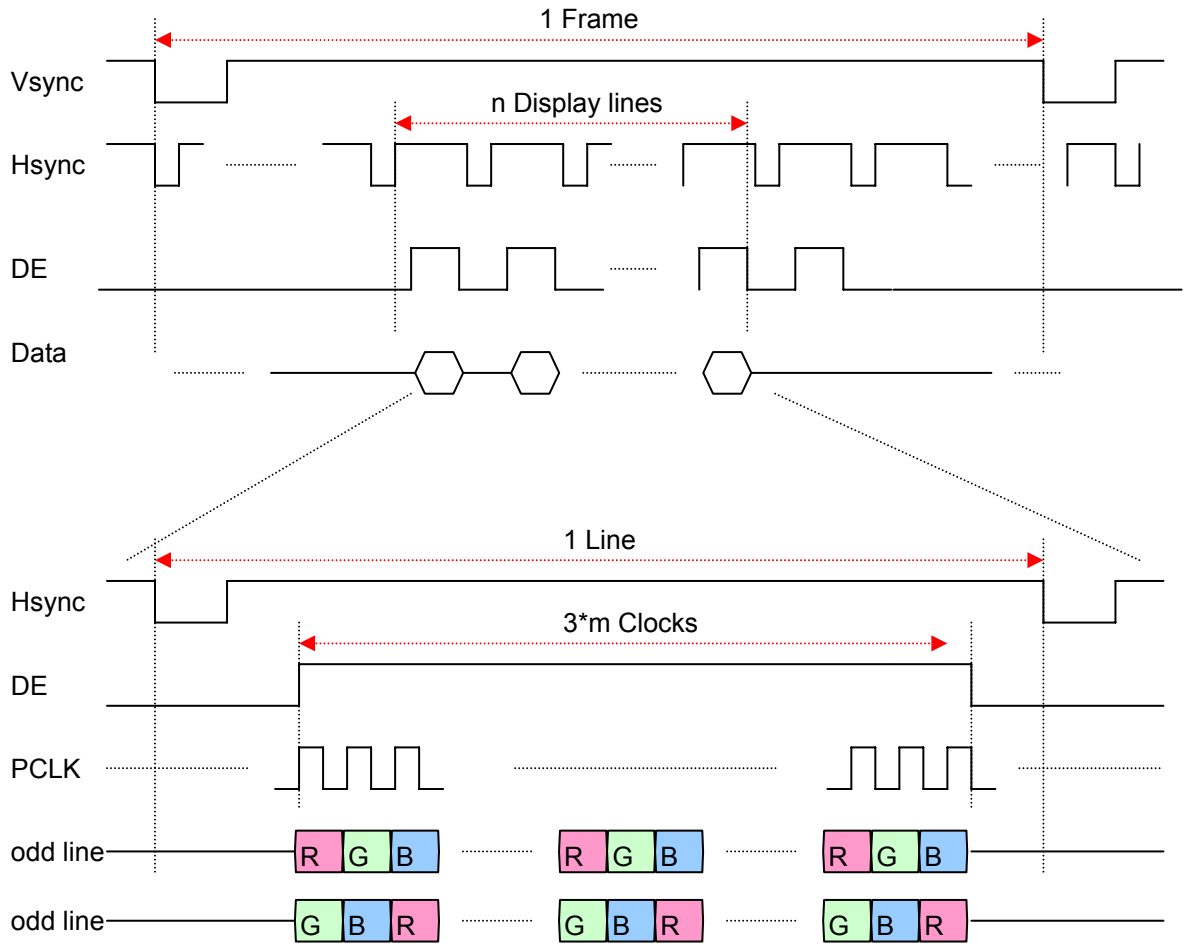
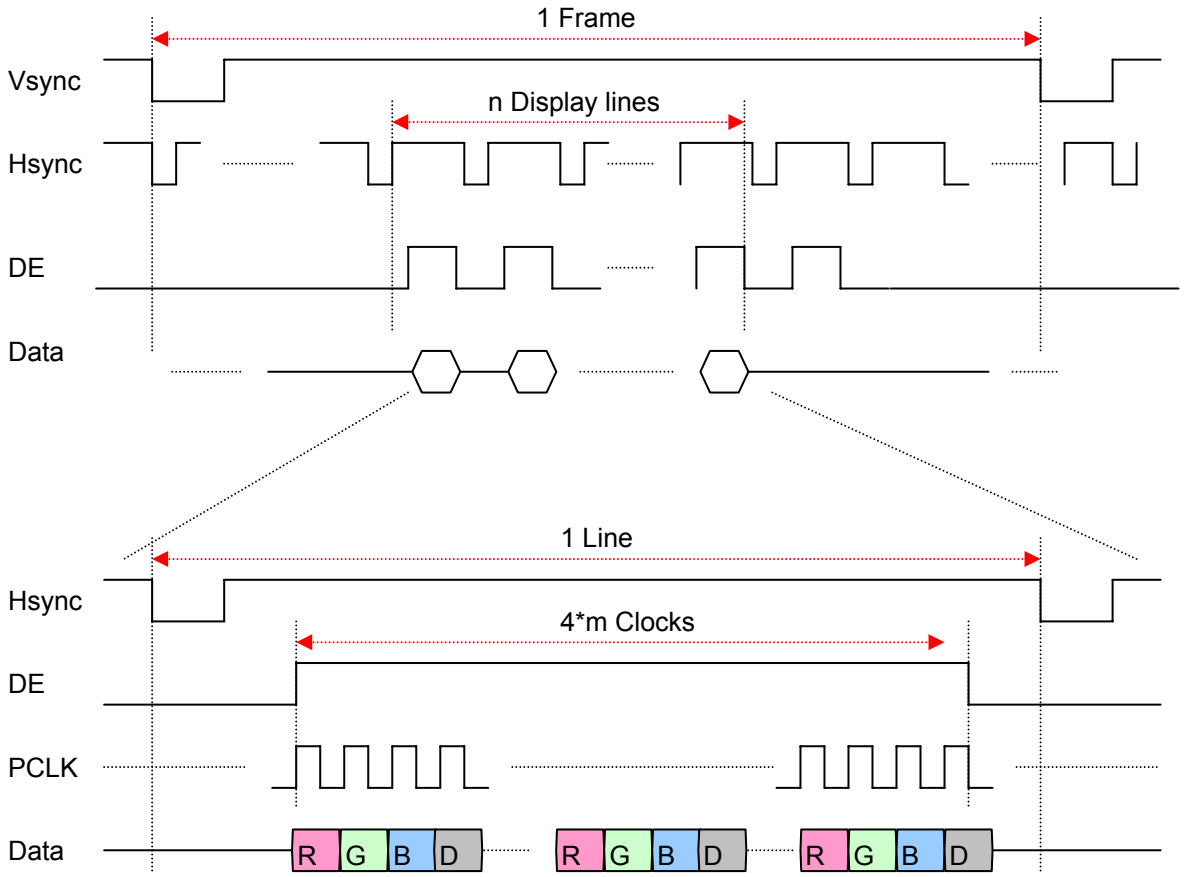


Figure 19-12 Delta RGB timing

19.10.5 RGB Dummy mode timing

This section shows the RGB Dummy diagram, the polarity of signal “Vsync”, “Hsync”, and “PCLK” can be programmed.



***Dummy = 0**

Figure 19-13 RGB Dummy timing

19.11 Format of Palette

This LCD controller contains a palette RAM with 256-entry x 16-bit used only for BPP8, BPP4, BPP2 and BPP1. Palette RAM data is loaded directly from the external memory palette buffer by DMAC channel 0. Each word of palette buffer contains 2 palette entries.

- In 8-bpp modes, palette buffer size is128 words.
- In 4-bpp modes, palette buffer size is 8 words.
- In 2-bpp modes, palette buffer size is 2 words.
- In 1-bpp modes, palette buffer size is 1 word.
- In 16/18/24-bpp modes, has no palette buffer.

Palette buffer base address	Bit: 31 . . . 16	Bit: 15 . . . 0
Palette entry	Entry-1 bit: 15 . . . 0	Entry-0 bit: 15 . . . 0
Palette buffer base address + 4	Bit: 31 . . . 16	Bit: 15 . . . 0
Palette entry	Entry-3 bit: 15 . . . 0	Entry-2 bit: 15 . . . 0
Palette buffer base address + 8	Bit: 31 . . . 16	Bit: 15 . . . 0
Palette entry	Entry-5 bit: 15 . . . 0	Entry-4 bit: 15 . . . 0

19.11.1 STN

For STN Panel, 16-bpp pixel data is encoded with RGB 565 or RGB 555.
Please refer to register LCDCTRL.RGB.

BPP 16, RGB 565, pixel encoding for STN Panel:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0

BPP 16, RGB 555, pixel encoding for STN Panel:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	R4	R3	R2	R1	R0	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0

19.11.2 TFT

BPP 16, RGB 565, pixel encoding for TFT Panel:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0

NOTES:

1 For BPP 16, 18, 24, palette is bypass.

19.12 Format of Frame Buffer

19.12.1 16bpp

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0

19.12.2 18bpp

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	R5	R4	R3	R2	R1	R0	0	0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
G5	G4	G3	G2	G1	G0	0	0	B5	B4	B3	B2	B1	B0	0	0

19.12.3 24bpp

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	R7	R6	R5	R4	R3	R2	R1	R0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
G7	G6	G5	G4	G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0

19.12.4 16bpp with alpha

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
A7	A6	A5	A4	A3	A2	A1	A0	0	0	0	0	0	0	0	0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
R5	R4	R3	R2	R1	G5	G4	G3	G2	G1	G0	B5	B4	B3	B2	B1

19.12.5 18bpp with alpha

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
A7	A6	A5	A4	A3	A2	A1	A0	R5	R4	R3	R2	R1	R0	0	0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
G5	G4	G3	G2	G1	G0	0	0	B5	B4	B3	B2	B1	B0	0	0

19.12.6 24bpp with alpha

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
A7	A6	A5	A4	A3	A2	A1	A0	R7	R6	R5	R4	R3	R2	R1	R0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
G7	G6	G5	G4	G3	G2	G1	G0	B7	B6	B5	B4	B3	B2	B1	B0

19.12.7 24bpp compressed

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BLUE 1 [7:0]								RED 0 [7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
GREEN 0 [7:0]								BLUE 0 [7:0]							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GLEEN 2 [7:0]								BLUE 2 [7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RED 1 [7:0]								GLEEN 1 [7:0]							

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RED 3 [7:0]								GLEEN 3 [7:0]							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BLUE3 [7:0]								RED2 [7:0]							

19.13 Format of Data Pin Utilization

19.13.1 Mono STN

In Mono STN mode, data pin pixel ordering of one LCD screen row. Column 0 is the first pixel of a screen row.

Upper panel								
Panel data width	**Col0**	**Col1**	**Col2**	**Col3**	**Col4**	**Col5**	**Col6**	**Col7**
1 bit	D0	D0	D0	D0	D0	D0	D0	D0
2 bit	D1	D0	D1	D0	D1	D0	D1	D0
4 bit	D3	D2	D1	D0	D3	D2	D1	D0
8 bit	D7	D6	D5	D4	D3	D2	D1	D0
Lower panel (dual-panel mode)								
4 bit	D11	D10	D9	D8	D11	D10	D9	D8
8 bit	D15	D14	D13	D12	D11	D10	D9	D8

19.13.2 Color STN

In Color STN mode, data pin pixel ordering of one LCD screen row. Column 0 is the first pixel of a screen row.

Upper panel							
Col0 (R)	**Col0 (G)**	**Col0 (B)**	**Col1 (R)**	**Col1 (G)**	**Col1 (B)**	**Col2 (R)**	**Col2 (G)**
D7	D6	D5	D4	D3	D2	D1	D0
Lower panel (dual-panel mode)							
D15	D14	D13	D12	D11	D10	D9	D8

19.13.3 18-bit Parallel TFT

Col0 (RGB)																	
D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

19.13.4 16-bit Parallel TFT

Col0 (RGB)															
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

19.13.5 8-bit Serial TFT (24bpp)

Col0 (R)							
D7	D6	D5	D4	D3	D2	D1	D0
Col0 (G)							
D7	D6	D5	D4	D3	D2	D1	D0
Col0 (B)							
D7	D6	D5	D4	D3	D2	D1	D0

19.14 LCD Controller Operation

19.14.1 Set LCD Controller AHB Clock and Pixel Clock

The LCD Controller has 2 clock input: AHB clock and pixel clock. The both clocks are generated by CPM (Clock and Power Manager). The frequency of the 2 clocks can be set by CPM registers. lcdc 's AHB clock is equal to AHB0 clock (HCLK in CPM spec), and CPM.LPCDR set LCD pixel clock division ratio. Please refer to CPM spec for detail.

LCD AHB clock is the LCD controller's internal clock while LCD pixel clock is output to drive LCD panel. There have 2 rules for LCD clocks:

1 For TFT Panel, the frequency of LCD AHB clock must be at least 1.5 times of LCD pixel clock.
2 For STN Panel, the frequency of LCD AHB clock must be at least 3 times of LCD pixel clock.

LCD panel determines the frequency of LCD pixel clock.

19.14.2 Enabling the Controller

If the LCD controller is being enabled for the first time after system reset or sleep reset, all of the LCD registers must be programmed as follows:

1 Write the frame descriptors and, if needed, the palette descriptor to memory.
2 Program the entire LCD configuration registers except the Frame Descriptor Address Registers (LCDDAx) and the LCD Controller enable bit (LCDCTRL.ENA).
3 Program LCDDAx with the memory address of the palette/frame descriptor.
4 Enable the LCD controller by writing to LCDCTRL.ENA.

If the LCD controller is being re-enabled, there has not been a reset since the last programming; only the registers LCDDAx and LCDCTRL.ENA need to be reprogrammed. The LCD Controller Status Register (LCDSTATE) must also be written to clear any old status flags before re-enabling the LCD controller.

Once the LCD controller has been enabled, do not write new values to LCD registers except LCDCTRL.ENA or DIS or LCDDA0/1 or LCDOSDC.F0/1EN .

19.14.3 Disabling the Controller

The LCD controller can be disabled in two ways: regular and quick.

1 Regular disabling.
Regular disabling is accomplished by setting the disable bit, LCDCTRL.DIS. The other bits in LCDCTRL must not be changed — read the register, set the DIS bit, and rewrite the register. This method causes the LCD controller to stop cleanly at the end of a frame. The LCD Disable Done bit, LCDSTATE.LDD, is set when the LCD controller finishes displaying the last frame, and the enable bit, LCDCTRL.ENA, is cleared automatically by hardware.
LCDCTRL.DIS must be set zero when enabling the controller.

2 Quick disabling.

Quick disabling is accomplished by clearing the enable bit, LCDCTRL.ENA. The LCD controller will finish any current DMA transfer, stop driving the panel, setting the LCD Quick Disable bit (LCDSTATE.QD) and shut down immediately. This method is intended for situations such as a battery fault, where system bus traffic has to be minimized immediately so the processor can have enough time to store critical data to memory before the loss of power. The LCD controller must not be re-enabled until the QD bit is set, indicating that the quick shutdown is complete. Do not set the DIS bit when a quick disabling command has been issued.

NOTES:

1 It is strongly recommended that software set the "LCD Module Stop Bit" in PMC to shut down LCDC clock supply to save power consumption after disable LCDC. Please refer to PMC for detailed information.

19.14.4 Resetting the Controller

At reset, the LCD Controller is disabled. All LCD Controller Registers are reset to the conditions shown in the register descriptions.

19.14.5 Frame Buffer & Palette Buffer

The starting address of frame buffer stored in external memory must be aligned to 4, 8 or 16 words boundary according to register LCDCTRL.BST. The length of buffer must be multiple of word (32-bit).

If LCDCTRL .BST = 0, align frame and palette buffer to 16 word boundary
If LCDCTRL .BST = 1, align frame and palette buffer to 8 word boundary
If LCDCTRL .BST = 2, align frame and palette buffer to 4 word boundary

One frame buffer contains encoded pixel data of multiple of screen lines; each line of encoded pixel data must be aligned to word boundary. If the length of a line is not the multiple of word, extra bits must be applied to reach a word boundary. It is suggested that the extra bits to be set zero.

19.14.6 CCIR601/CCIR656

CCIR601: just as 16bit-parallel output.

CCIR656: need external encoder, or software designer need give digital blanking data and timing reference signal in data buffer.

19.14.7 OSD Operation

1 Normal process.

a Configuration.

* LCDCFG and LCDCTRL

* LCDOSDC and LCDOSDCTRL
* LCDRGBC and LCDIPUR

b Set Color.
* LCDBGC, LCDKEY0, LCDKEY1, LCDALHPA

c Set Display.
* LCDVAT, LCDDAH, LCDDAV
* LCDXYP0, LCDXYP1, LCDSIZE0, LCDSIZE1
* LCDVSYNC, LCDHSYNC

d Set DMAC.
* LCDIID
* LCDDA0, LCDSA0, LCDFID0, LCDCMD0, LCDOFFS0, LCDPW0, LCDCNUM0, LCDDESSIZE0
* LCDDA1, LCDSA1, LCDFID1, LCDCMD1, LCDOFFS1, LCDPW1, LCDCNUM1, LCDDESSIZE1

e Enable LCDC.

f Check the state from register LCDSTATE and LCDOSDS.

2 Reconfigure OSD.

If foreground0 and foreground1 (enable, position, size)need to reconfigure during display process, there has two methods:

Method1:(recommend in TFT and SLCD)

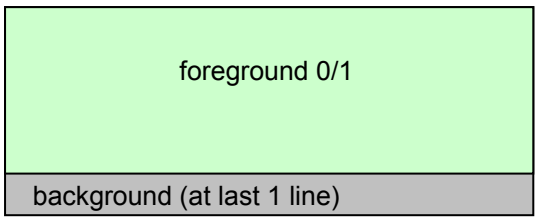
a Reconfigure the relate Register after disable LCDC.
b In TFT mode, use normal disable to avoid lcd panel flicker.
c In SLCD mode, use quick disable (smart LCD could keep the frame by its inner buffer).
d After disable LCDC, you can reconfigure any register/descriptor, but please make sure this process is quick enough in TFT mode (less than the interval between two frames).

Method2:

Dynamic reconfigure the register:

You can reconfigure some register(LCDOSDC.F0/1EN) during display process but there some rule you must follow:

a Foreground 0/1 are at last 1 line less than background.



b Foreground 0 and foreground 1's data can not less than 33 words(except 0 word).
Or you only can change those register after disable LCDC.

c When use TFT panel. During the display process, you can re-configure the LCDOSDC.F0EN, LCDCOSDC.F1EN; (You can not change them when use SLCD or TVE) but the new configuration will recognized by LCDC module after finished a complete frame. If you need to re-configure LCDOSDCTRL.IPU to select IPU or DMA channel 1, you need to follow the process below:

- Quick or Normal disable LCDC. (SLCD only can use Quick disable)
- Configure the LCDOSDCTRL to set IPUEN, and then enable LCD.

To change IPU to DMA1 you can :

- Quick or Normal disable LCDC. (SLCD only can use Quick disable)
- Configure the LCDOSDCTRL to set IPUEN = 0, and then enable LCD.

3 During the display process, while foreground 1 use IPU, to change size of foreground 1 you need follow the step shown bellow.

a Quick or Normal disable LCDC. (SLCD only can use Quick disable).

b Configure the IPU, and LCDSIZE1.

c Run IPU and enable LCDC.

4 You CAN NOT change BPP or OSDBPP during the display process. if you want to change them first you should disable LCDC, change the BPP or OSDBPP and then enable LCDC.
If you need not use Foreground0 during the whole display process. set BPP to 5.

5 You can change LCDSIZE0/1 during display process without disable LCD controller.

method 1:

a Set LCDCOSDC.F0/1EN = 0. (follow the rule above)

b Re-configure LCDSIZE0/1 (and the relate DMA0/1 descriptor),
then set LCDOSDCTRL.CHANGE = 1.

c Wait until CHANGE = 0 and then set LCDOSDC.F0/1EN = 1.

method 2:

a Set LCDOSDCTRL.CHANGE = 1.

b Wait until LCDOSDS. READY = 1.

c Change relate DMA0/1 channel descriptor.

d Wait until LCDOSDCTRL.CHANGE = 0.

*Please notice that in TVE (not include VGA)and SLCD only use method 2.

6 You can change LCDXY0/1 during display process without disable LCD controller.

Method 1:

a Set LCDOSDC.F0/1EN = 0 (follow the rule above).

b Change LCDXYPOS0/1 and then set LCDOSDCTRL.CHANGE = 1.

c Wait until LCDOSDCTRL.CHANGE = 0.

Method 2:

a Change LCDXYPOS0/1.

b Set LCDOSDCTRL.CHANGE = 1.

c Wait until LCDOSDCTRL.CHANGE = 0.

* Please notice that in TVE (not include VGA) and SLCD only use method 2.

* Please notice that if you do not change foreground 0/1's size and position, keep LCDOSDCTRL.CHANGE = 0. And you can only change one of them in one time.

7 How to "close/open" foreground0 and foreground1?

Method 1:

a Set LCDOSDCTRL.CHANGE = 1.

b Wait until LCDOSDS. READY = 1.

c Direct change LCDOSDC.F0/1EN.

d Wait until LCDOSDCTRL.CHANGE = 0.

Method 2:

Change foreground0/1 size to 0 Without change LCDOSDC.F0/1EN.

Method 3: (recommend)

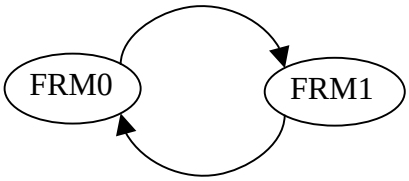
Normal disable LCDC, and change LCDOSDC.F0/1EN. Use normal disable need to wait LCDSTATE.LDD, and set relate register soon, to make sure the LCD panel are not flicker.

* Please notice that in TVE (not include VGA) and SLCD only use method 2,3. And strongly suggest that DO NOT close both foreground0 and 1 or set both foreground0 and 1 's size to 0.

19.14.8 Descriptor Operation

1 TFT panel

Not use palette: you can use only one descriptor or several connected descriptor. As which shown below.



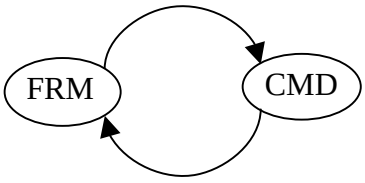
Use palette: add one PAL descriptor at the beginning of descriptor chain.

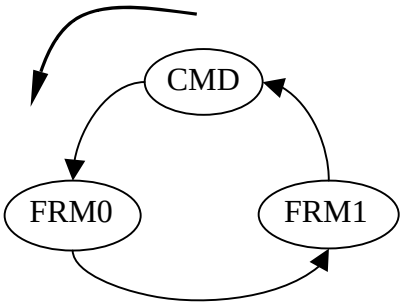
When you need to change palette during the display you need follow the steps shown below.

*Please notice that you **cannot** disable foreground 0 during the whole process. and also You can not change PAL when Foreground 0' s area == 0 or not enable LCDOSDC.F0EN.

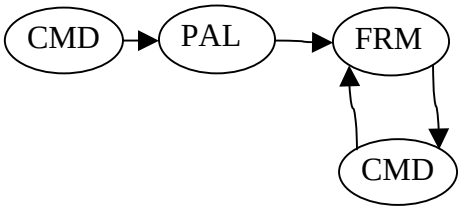
2 SLCD

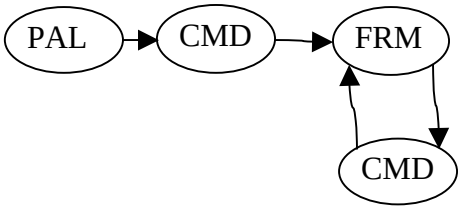
Not use palette:



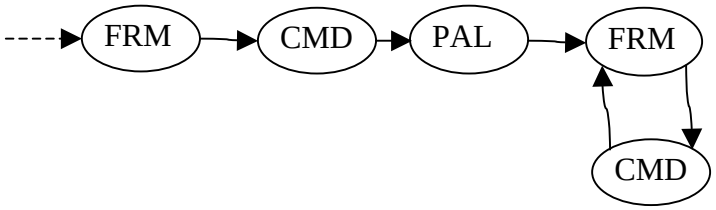


Use palette:





Change palette:



You can not change PAL when Foreground 0' s area == 0. Or not enable LCDOSDC.F0EN and during you change PAL, you can not change F0 or F1's size.

19.14.9 IPU direct connect mode

When you use IPU direct connect mode, you need to:

1 Open IPU early than LCDC.
2 Use normal disable in TFT mode, and use quick disable in SLCD/TVE mode.
3 When you use normal disable you need to wait IPU frame end flag.
4 When you use quick disable you must not wait IPU frame end flag, and must reset IPU before restart LCDC and IPU.
5 In SLCD mode, you can first wait IPU frame end flag, then quick stop LCDC. Then you need not reset IPU before restart LCDC and IPU.

* "IPU frame end flag" please refer to IPU spec.

19.14.10VGA output

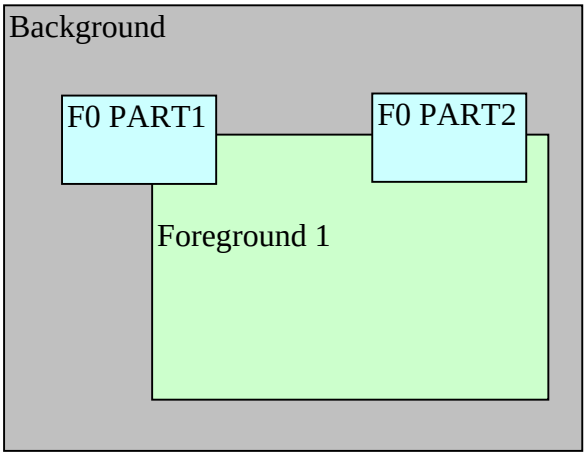
When you use VGA output you need:

1 Open all channel of DAC. (refer to TVEDAC spec)
2 Set TVEN to 0.
3 Disable LCD panel pins (except HSYNC/VSYNC) for save power. (refer to GPIO spec)

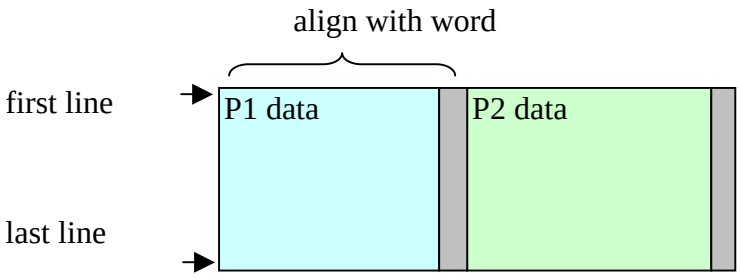
19.14.11Foreground 0 divide mode

In divide mode the original register of foreground 0 position and size are correspond to F0 PART1, the additional (named with "_part2") registers correspond to F0 PART2.
LCDOSDC.F0EN correspond the total foreground 0 （part1 and part2）and each part has a F0PxEN to enable.
F0EN, F0P2EN, F0P1EN, and part2's position/size can be reconfigure during display process. (refer to 1.12.6)

MODE 1: LCDOSDC.F0DIVMD = 0. F0P2MD = 1.

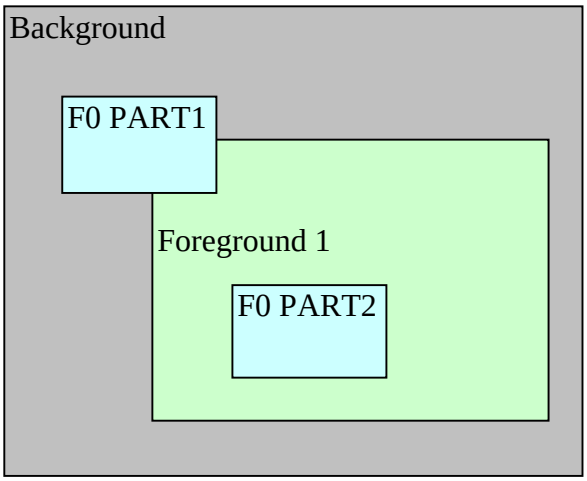


Foreground 0 divided into 2 parts, and PART1, PART2 must begin with same line and has the same height. They can have different width but cannot overlay each other.
They can use only one descriptor (connect to it self).
The two parts data must "combination" in one data buffer as follow:



*Please notice that in this mode, you need to disable LCDC before reconfigure foreground0/1's Register.

MODE 2: LCDOSDC.F0DIVMD = 1. F0P2MD = 0. F0P1EN = 1. F0P2EN = 1.



Foreground 0 divided into 2 parts, and PART1, PART2 can have different width and height but cannot overlay each other. PART2 must below PART1 they also cannot have any superposition in vertical.

PART1 and PART2 use independent descriptor refer to descriptor register with “part2”.

20 Smart LCD Controller

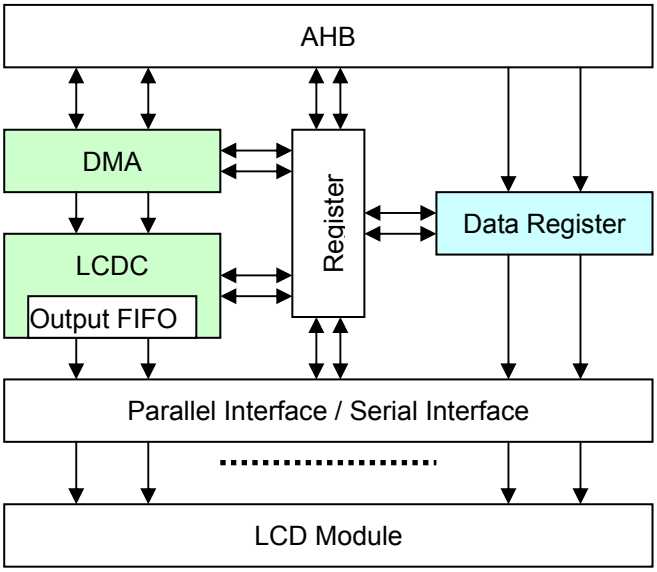
20.1 Overview

The Smart LCD Controller affords an interface to transfer data from the LCD controller to the LCD Module. It supports DMA operation and register operation.

Features:

- Supports a large variety of LCD Module from different vendors
- Supports parallel and serial interfaces
- Supports different size of display panel
- Supports different width of pixel data
- Supports internal DMA operation and register operation
- Supports Write Operation. Read Operation is not supported

20.2 Structure



*Please notice that the command only can transfer by DMA channel 0. No matter the DMA channel 1 or IPU are use or not.

20.3 Pin Description

Table 20-1 SLCD Pins Description

Name	I/O	Description	Interface
SLCD_RS	O	Command/Data Select Signal. The polarity of the signal can be programmable.	Serial: RS Parallel: RS
SLCD_CS	O	Data Sample Signal. The polarity of the signal can be programmable.	Serial: CS Parallel: Sample Data with the edge of CS
SLCD_CLK	O	The clock of SLCD. The polarity of the clock can be programmable.	Serial or not used
SLCD_DAT*1 [17:0]	O	The data of SLCD. Relate to 1.9.4 Data mapping to GPIO function.	Serial: SLCD_DAT [15] Parallel: 18bit SLCD_DAT [17:0] 16bit SLCD_DAT [15:0] 8bit SLCD_DAT [7:0]
LCD_LO6_O	O	24 bit parallel SLCD RGB (or 24 bit command) low bit ([17:16],[9:8],[1:0]) output Relate to 1.9.4 Data mapping to GPIO function.	

NOTES:

1 ***1:** SLCD_DAT [15] is also use as data pin for serial. The SLCD pins are shared with LCDC. You can see the set of register LCDCFG.LCDPIN in LCDC spec.

20.4 Register Description

In this section, we will describe the registers in Smart LCD controller. Following table lists all the registers definition. All register's 32bit address is physical address. And detailed function of each register will be described below.

Name	Description	RW	Reset Value	Address	Access Size
MCFG	SLCD Configure Register	RW	0x0000	0x130500A0	32
MCTRL	SLCD Control Register	RW	0x00	0x130500A4	8
MSTATE	SLCD Status Register	RW	0x00	0x130500A8	8
MDATA	SLCD Data Register	RW	0x00000000	0x130500AC	32

.

20.4.1 SLCD Configure Register (MCFG)

The register MCFG is used to configure SLCD.

MCFG 0x130500A0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			DWIDTH			CWIDTH		Reserved			CSPLY	RSPLY	Reserved	CLKPLY	TYPE
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15:13	Reserved	These bits always read 0, and written are ignored.	R
12:10	DWIDTH*1	Data Width. DWIDTH / Data Width 000 18-bit once Parallel/Serial 001 16-bit once Parallel/Serial 010 8-bit third time Parallel 011 8-bit twice Parallel 100 8-bit once Parallel/Serial 101 24-bit once Parallel 111 9-bit twice Parallel 110 Reserved *Please notice that you can only use 24-bit parallel command when use 24-bit parallel data. (The command may not 24-bit but need put them as 24-bit in memory(one command use one word))	RW

<table>
<tr><td>9:8</td><td>CWIDTH*1</td><td>Command Width.<table><tr><th>CWIDTH</th><th>Command Width</th></tr><tr><td>00</td><td>16-bit once / 9bit once</td></tr><tr><td>01</td><td>8-bit once</td></tr><tr><td>10</td><td>18-bit once</td></tr><tr><td>11</td><td>24-bit once</td></tr></table>*Please notice that you can only use 24-bit parallel command when use 24-bit parallel data. (The command may not 24-bit but need put them as 24-bit in memory (one command use one word))</td><td>RW</td></tr>
<tr><td>7:5</td><td>Reserved</td><td>These bits always read 0, and written are ignored.</td><td>R</td></tr>
<tr><td>4</td><td>CSPLY</td><td>CS Polarity. (CS initial level will be different from CS Polarity)
0: Active Level is Low
1: Active Level is High</td><td>RW</td></tr>
<tr><td>3</td><td>RSPLY</td><td>RS Polarity.
0: Command RS = 0, Data RS = 1
1: Command RS = 1, Data RS = 0</td><td>RW</td></tr>
<tr><td>2</td><td>Reserved</td><td>These bits always read 0, and written are ignored.</td><td>R</td></tr>
<tr><td>1</td><td>CLKPLY</td><td>LCD_CLK Polarity.
0: Active edge is Falling
1: Active edge is Rising</td><td>RW</td></tr>
<tr><td>0</td><td>TTYPE</td><td>Transfer Type.
0: Parallel
1: Serial</td><td>RW</td></tr>
</table>

NOTES:

1 *1: The set of DWIDTH and CWIDTH should keep to the rules as follows:

Interface Mode	Command Width	Data Width	Color
Parallel	18-bit	18-bit once	R6G6B6
	16-bit	16-bit once	R5G6B5
		9-bit twice	
	9-bit	9-bit twice	
	8-bit	8-bit once	
		8-bit twice	
		8-bit third times	
Serial	18-bit	18-bit once	
	16-bit	16-bit once	
	9-bit	9-bit twice	
	8-bit	8-bit once	
		8-bit twice	
		8-bit third times	

20.4.2 SLCD Control Register (MCTRL)

MCTRL is SLCD Control Register.

MCTRL 0x130500A4

Bit	7	6	5	4	3	2	1	0
	Reserved					DMASTART	DMAMODE	DMATXEN
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:3	Reserved	These bits always read 0, and written are ignored.	R
2	DMAMODE	SLCD descriptor DMA mode select. 0: DMA will continually transfer data follow descriptor chain 1: DMA will stop when one descriptor finished	
1	DMASTART	Only use when DMAMODE = 1, set 1 to restart DMA transfer.	
0	DMATXEN	SLCD DMA Transfer Enable. This bit is only used for DMA automatic transfer: – This bit starts the automatic transfer of image data from system memory to LCDM. – When DMAC finishes transferring the data, and the MSTATE.BUSY bit is 0, you can clear DMATXEN bit to stop DMA mode.	RW

20.4.3 SLCD Status Register (MSTATE)

The register of MSTATE is SLCD status register.

MSTATE 0x130500A8

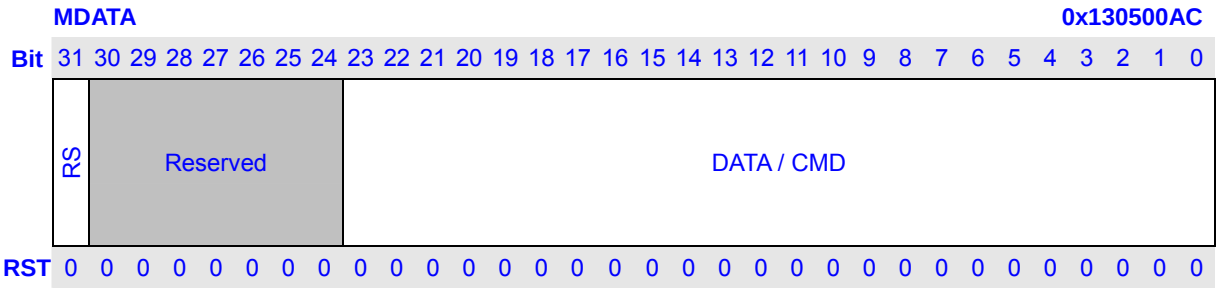
Bit	7	6	5	4	3	2	1	0
	Reserved							BUSY
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:1	Reserved	These bits always read 0, and written are ignored.	R
0	BUSY	Transfer is working or not. This bit will be set to 1 when transfer is working. It will be cleared by hardware when transfer is finished.	RW

		0: not busy 1: busy	

20.4.4 SLCD Data Register (MDATA)

The register MDATA is used to send command or data to LCM. When RS=0, the low 24-bit is used as command. When RS=1, the low 24-bit is used as data.



Bits	Name	Description	RW
31	RS	The RS bit of data register is used to decide the meanings of the low 24-bit. 0: data 1: command	RW
30:24	Reserved	These bits always read 0, and written are ignored.	R
23:0	DATA/CMD	Data or Command Register.	RW

20.5 System Memory Format

20.5.1 Data format

you can configure these registers according to LCDC module.

20.5.2 Command Format

1 18-bit command

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
X	X	X	X	X	X	X	X	X	X	X	X	X	X	C17	C16
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0

2 16-bit command

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0

3 9-bit command once

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
X	X	X	X	X	X	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
X	X	X	X	X	X	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0

4 8-bit command once

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
C7	C6	C5	C4	C3	C2	C1	C0	C7	C6	C5	C4	C3	C2	C1	C0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
C7	C6	C5	C4	C3	C2	C1	C0	C7	C6	C5	C4	C3	C2	C1	C0

5 8-bit command twice (Command = command part + data part)
*Please notice that when you use this kind command, set CWIDTH as 8bit once and set the LCDCNUM.CNUM as doubled the real command number.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D7	D6	D5	D4	D3	D2	D1	D0	C7	C6	C5	C4	C3	C2	C1	C0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
D7	D6	D5	D4	D3	D2	D1	D0	C7	C6	C5	C4	C3	C2	C1	C0

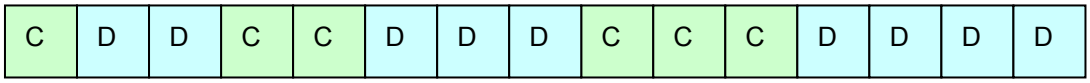
20.6 Transfer Mode

Two transfer modes can be used: DMA/IPU Transfer Mode and Data Register Transfer Mode. In DMA/IPU mode, always transfer commands by DMA 0.

20.6.1 DMA Transfer Mode

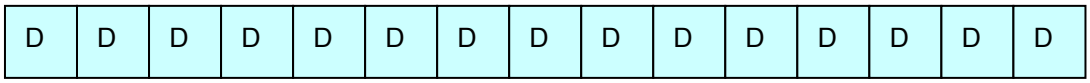
Command and data can be recognized by RS bit coming from memory. The format of DMA transfer can be as follows:

1 Command and Data



*Please notice that the command only can insert between two complete frame and the number of command is 0~255.

2 Only Data



*You can also not use command but you still need to use a command descriptor and set the CNUM = 0.

Because DMA transfer mode only can work in OSD mode, you need to configure the panel according OSD mode:

1 Configuration.
* LCDCFG and LCDCTRL
* LCDOSDC and LCDOSDCTRL
* LCDRGBC and LCDIPUR

2 Set Color.
* LCDBGC, LCDKEY0, LCDKEY1, LCDALHPA

3 Set Display.
* LCDVAT, LCDDAH, LCDDAV
* LCDXYP0, LCDXYP1, LCDSIZE0, LCDSIZE1
* LCDVSYNC, LCDHSYNC

4 Set DMAC.
* LCDIID
*LCDDA0, LCDSA0, LCDFID0, LCDCMD0, LCDOFFS0, LCDPW0, LCDCNUM0,

LCDDESSIZE0
*LCDDA1, LCDSA1, LCDFID1, LCDCMD1, LCDOFFS1, LCDPW1, LCDCNUM1, LCDDESSIZE1

5 Enable slcd DMA.

6 Enable LCDC.

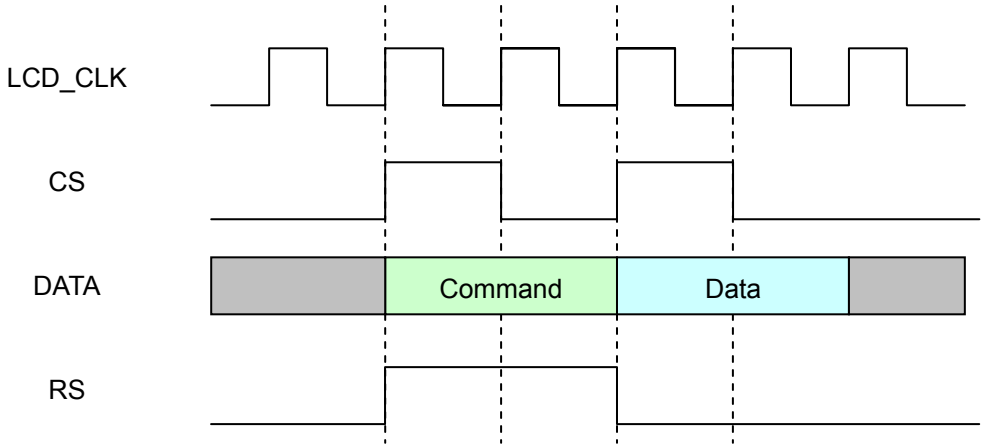
20.6.2 Register Transfer Mode

Each time you can write a command or a data to the register, then it will transfer the RS signal and data or command to LCM. Command and data can be recognized by RS bit coming from data register. The format of data register transfer can be as follows:

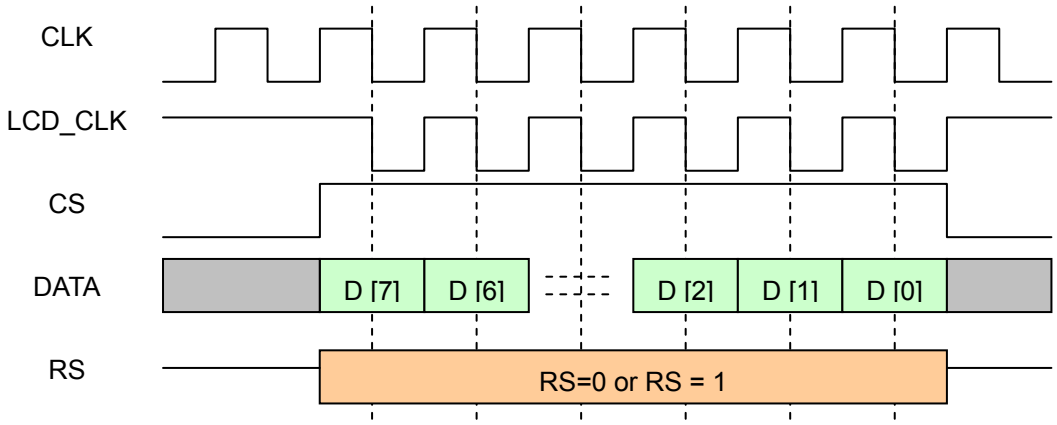
Command	RS [31] = 1	XXX [30:n+1]	Command [n:0]
Data	RS [31] = 0	XXX [30:24]	Data [23:0]
	RS [31] = 0	XXX [30:16]	Data [15:0]
	RS [31] = 0	XXX [30:9]	Data [8:0]
	RS [31] = 0	XXX [30:8]	Data [7:0]

20.7 Timing

20.7.1 Parallel Timing



20.7.2 Serial Timing



20.8 Operation Guide

20.8.1 DMA Operation

1 Start DMA transfer.

a Set LCDCFG.MODE to 1101 to choose LCM.

b Set LCDCTRL.BST to choose burst length for transferring.

c Set register LCDIID0, LCDDA0, LCDSA0, LCDFID0, LCDCMD0, LCDOFFS0, LCDPW0, LCDCNUM0, LCDDESSIZE0 to initial internal DMA.

d Also set register LCDIID1, LCDDA1, LCDSA1, LCDFID1, LCDCMD1, LCDOFFS1, LCDPW1, LCDCNUM1, LCDDESSIZE1 when use DMA channel 1 in OSD mode.

e Set MCFG to configure SLCDC.

f Before starting DMA, Wait for MSTATE.BUSY == 0.

g Set MCTRL.DMATXEN to 1 to prepare DMA transfer.

Note that if you don't want to stop DMA transfer, you need not to check MSTATE.BUSY.

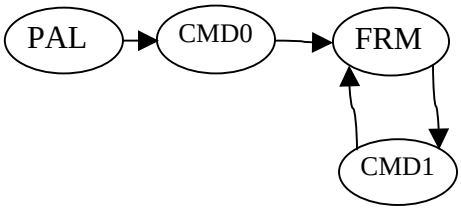
h Set LCDCCTRL.ENA to 1 to start LCDC internal DMA.

i The LCDC internal DMA will transfer data to SLCDC, and SLCDC transfer data to LCM.

Repeat this step till you want to close the SLCDC to transfer data to LCM Panel.

*Please notice that use and only use DMA0 to transfer command no matter use DMA0 to transfer frame data or not.

One recommend descriptor chain (CMD0 with CNUM>0 and CMD1 with CNUM=0):



2 Stop DMA transfer.

a Set LCDCCTRL.ENA to 0 to stop LCDC internal DMA at once.

b Wait till MSTATE.BUSY is set to 0 by hardware.

MSTATE.BUSY == 1: there is data in the FIFO waited for transferring to LCM.

MSTATE.BUSY == 0: all data in the FIFO have finished transferring to LCM.

c Set MCTRL.DMATXEN to 0 to stop DMA transfer.

3 Restart DMA transfer.

When MCTRL.DMATXEN is set to 0, and then you want to restart DMA transfer at once, you should ensure that MCTRL.DMATXEN must keep 0 at least three cycles of PIXCLK.

20.8.2 Register Operation

1 Set MCFG to configure SLCD.
2 Wait for MSTATE.BUSY == 0.
3 Set MDATA register.
4 Wait for MSTATE.BUSY == 0.
5 Set MDATA register.
6 Wait for MSTATE.BUSY == 0.
7 … …

21 TV Encoder

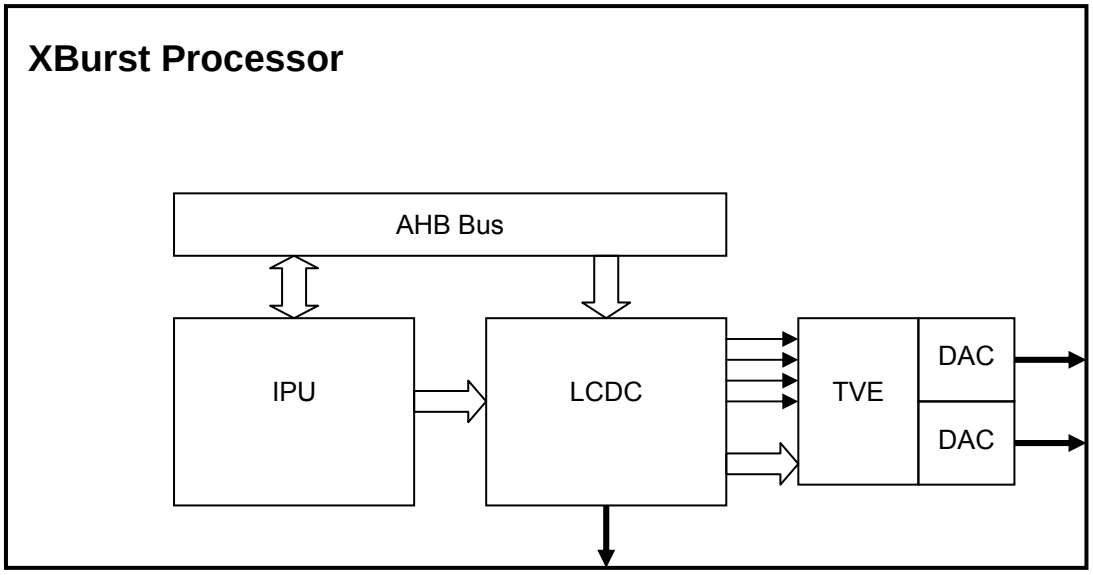
21.1 Overview

The TV Encoder enables the data for LCD panel showing in TV screen.

Features:

- CVBS and S-video output
- PAL and NTSC supported

21.2 Structure



21.3 Pin Description

Table 21-1 TVE Pins Description

Name	I/O	Description	Interface
YCMP	AO	CVBS or Luma of S-Video analog output	
C	AO	Chroma of S-Video analog output	

21.4 Register Description

TVE memory mapped registers are put together with LCD controller, occupied address area of 'H13050140 ~ 'H130501FF. Following table lists all the registers definition. All register's 32bit address is physical address. And detailed function of each register will be described below.

Name	Description	RW	Reset Value	Address	Size
TVECR	TV Encoder Control register	RW	0x01040301	0x13050140	32
FRCFG	Frame configure register	RW	0x00170271	0x13050144	32
SLCFG1	TV signal level configure register 1	RW	0x0320011A	0x13050150	32
SLCFG2	TV signal level configure register 2	RW	0x012800F0	0x13050154	32
SLCFG3	TV signal level configure register 3	RW	0x00000048	0x13050158	32
LTCFG1	Line timing configure register 1	RW	0x00143F4E	0x13050160	32
LTCFG2	Line timing configure register 2	RW	0x05A0103D	0x13050164	32
CFREQ	Chrominance sub-carrier frequency configure register	RW	0x2A098ACB	0x13050170	32
CPHASE	Chrominance sub-carrier phase configure register	RW	0x00000001	0x13050174	32
CCFG	Chrominance filter configure register	RW	0x3B3B8989	0x13050178	32
WSSCR	Wide screen signal control register	RW	0x00000070	0x13050180	32
WSSCFG1	Wide screen signal configure register 1	RW	0x00000000	0x13050184	32
WSSCFG2	Wide screen signal configure register 2	RW	0x00000000	0x13050188	32
WSSCFG3	Wide screen signal configure register 3	RW	0x00000000	0x1305018C	32

21.4.1 TV Encoder Control Register (TVECR)

This register is used to control TV encoder.

TVECR **0x13050140**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						YUV	ECVBS	DAPD3	DAPD2	DAPD1	DAPD	Reserved	YCDLY			CGAIN		CBW		Reserved		SYNCT	PAL	FINV	ZBLACK	CR1ST	CLBAR	Reserved			SWRST
RST	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1

Bits	Name	Description	RW
31:29	Reserved	These bits always read 0, and written are ignored.	R
28	Reserved	These bits always read 0, and written are ignored.	R

<table>
<tr><td>27:26</td><td>Reserved</td><td>These bits always read 0, and written are ignored.</td><td>R</td></tr>
<tr><td>25</td><td>YUV</td><td>set this bit to 1 to enable yuv output.</td><td>RW</td></tr>
<tr><td>24</td><td>ECVBS</td><td>Enable CVBS (Composite Video Baseband Signal) output. This bit is used to choose the TVE output signal format between CVBS and S-Video.<table><tr><th>ECVBS</th><th>Description</th></tr><tr><td>1</td><td>TVE outputs CVBS format signal to TV</td></tr><tr><td>0</td><td>TVE outputs S-Video format signal to TV</td></tr></table></td><td>RW</td></tr>
<tr><td>23</td><td>DAPD3</td><td>DAC 3 power down. When it is 1, power down DAC 3, the Cr of components video, or the BLUE of VGA.</td><td>RW</td></tr>
<tr><td>22</td><td>DAPD2</td><td>DAC 2 power down. When it is 1, power down DAC 2, the chroma of S-Video, or the Cb of components video, or the GREEN of VGA.</td><td>RW</td></tr>
<tr><td>21</td><td>DAPD1</td><td>DAC 1 power down. When it is 1, power down DAC 1, the CVBS, or the luma of S-Video, the Y of components video, or the RED of VGA.</td><td>RW</td></tr>
<tr><td>20</td><td>DAPD</td><td>DAC power down. When it is 0, power down all DACs.</td><td>RW</td></tr>
<tr><td>19</td><td>Reserved</td><td>These bits always read 0, and written are ignored.</td><td>R</td></tr>
<tr><td>18:16</td><td>YCDLY</td><td>(internal used only)</td><td>RW</td></tr>
<tr><td>15:14</td><td>CGAIN</td><td>Chrominance modulated signal gain factor setting when it is added to luminance signal in composite output format.<table><tr><th>CGAIN</th><th>Description</th></tr><tr><td>00</td><td>1</td></tr><tr><td>01</td><td>1/4</td></tr><tr><td>10</td><td>1/2</td></tr><tr><td>11</td><td>3/4</td></tr></table></td><td>RW</td></tr>
<tr><td>13:12</td><td>CBW</td><td>Bandwidth setting for chrominance filter.<table><tr><th>CBW</th><th>Description</th></tr><tr><td>00</td><td>Narrow band</td></tr><tr><td>01</td><td>Wide band</td></tr><tr><td>10</td><td>Extra wide band</td></tr><tr><td>11</td><td>Ultra wide band</td></tr></table></td><td>RW</td></tr>
<tr><td>11:10</td><td>Reserved</td><td>These bits always read 0, and written are ignored.</td><td>R</td></tr>
<tr><td>9</td><td>SYNCT</td><td>Choose the sequence of field synchronizing pulses duration.<table><tr><th>SYNCT</th><th>Description</th></tr><tr><td>0</td><td>The duration of sequence of field synchronizing pulses is 3 H, where H is a line period. Set SYNCT to this for NTSC TV set</td></tr><tr><td>1</td><td>The duration of sequence of field synchronizing pulses is 2.5 H. Set SYNCT to this for PAL TV set</td></tr></table></td><td>RW</td></tr>
<tr><td>8</td><td>PAL</td><td>Set this to 1 for PAL TV set, 0 for NTSC TV set.</td><td>RW</td></tr>
<tr><td>7</td><td>FINV</td><td>When this bit is 1, invert top and bottom fields.</td><td>RW</td></tr>
<tr><td>6</td><td>ZBLACK</td><td>Black of luminance (Y) input is 0. Set this bit to 1 if the input video</td><td>RW</td></tr>
</table>

		luminance data for black is 0. Set this bit to 0 if the input video luminance data for black is 16. When this bit is 0, the Y input data will be clamped to ≥ 16.	
5	CR1ST	This bit described the Cb and Cr data order in input video. **ECVBS** / **Description** 0 / Cb comes before Cr, which is ITU656 standard 1 / Cr comes before Cb	RW
4	CLBAR	Color bar mode. In this mode, a color bar picture is output to TV. **CLBAR** / **Description** 0 / Output input video to TV 1 / Output color bar to TV	RW
3	Reserved	These bits always read 0, and written are ignored.	R
2	Reserved	These bits always read 0, and written are ignored.	
1	Reserved	These bits always read 0, and written are ignored.	R
0	SWRST	Software reset. When set this bit to 1, TVE is reset.	RW

21.4.2 Frame configure register (FRCFG)

This register is used to configure line in a frame.

FRCFG **0x13050144**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								L1ST								Reserved						NLINE									
RST	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	1	0	0	1	1	1	0	0	0	1

Bits	Name	Description	RW
31:24	Reserved	These bits always read 0, and written are ignored.	R
23:16	L1ST	This field defines the first active video line of a field. The reset value is 23 in decimal. The frame active video line number is (NLINE – 1 – 2 * L1ST). The top and bottom field line number is a half of the frame line number.	RW
15:10	Reserved	These bits always read 0, and written are ignored.	R
9:0	NLINE	This field defines number of lines per-frame. The reset value is **625** in decimal.	RW

21.4.3 Signal level configure register 1, 2 and 3 (SLCFG1, SLCFG2, SLCFG3)

These registers are used to configure the TV signal level in difference phases.

SLCFG1 **0x13050150**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						WHITEL										Reserved						BLACKL									
RST	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	1	0

Bits	Name	Description	RW
31:26	Reserved	These bits always read 0, and written are ignored.	R
25:16	WHITEL	Signal level for white color. The reset value is 800 in decimal.	RW
15:10	Reserved	These bits always read 0, and written are ignored.	R
9:0	BLACKL	Signal level for black color. The reset value is 282 in decimal.	RW

SLCFG2 **0x13050154**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved						VBLANKL										Reserved						BLANKL									
RST	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0

Bits	Name	Description	RW
31:26	Reserved	These bits always read 0, and written are ignored.	R
25:16	VBLANKL	Signal level in vertical blank period. The reset value is 296 in decimal.	RW
15:10	Reserved	These bits always read 0, and written are ignored.	R
9:0	BLANKL	Signal level in other blank period. The reset value is 240 in decimal.	RW

SLCFG3 **0x13050158**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																								SYNCL							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0

Bits	Name	Description	RW
31:8	Reserved	These bits always read 0, and written are ignored.	R
7:0	SYNCL	Signal level in sync period. The reset value is 72 in decimal.	RW

21.4.4 Line timing configure register 1 and 2 (LTCFG1, LTCFG2)

These registers are used to configure timing period in a line.

LTCFG1 **0x13050160**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved											FRONTP					Reserved	HSYNCW							Reserved	BACKP						
RST	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	1	1	1	1	1	1	0	1	0	0	1	1	1	0

Bits	Name	Description	RW
31:21	Reserved	These bits always read 0, and written are ignored.	R
20:16	FRONTP	Front porch width, 16 cycles of 13.5MHz for 525 line system and 20 cycles for 625 line.	RW
15	Reserved	These bits always read 0, and written are ignored.	R
14:8	HSYNCW	HSYNC width in cycles of 13.5MHz. The reset value is 63 in decimal.	RW
7	Reserved	These bits always read 0, and written are ignored.	R
6:0	BACKP	Back porch width in cycles of 13.5MHz. The reset value is 78 in decimal.	RW

LTCFG2 **0x13050164**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					ACTLIN											Reserved	PREBW							Reserved	BURSTW						
RST	0	0	0	0	0	1	0	1	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	1	1	0	1

Bits	Name	Description	RW
31:27	Reserved	These bits always read 0, and written are ignored.	R
26:16	ACTLIN	Active line length in cycles of 27MHz. The reset value is 1440 in decimal, which represent 720 pixels per line.	RW
15:13	Reserved	These bits always read 0, and written are ignored.	R
12:8	PREBW	Pre-burst width. The width after HSYNC and before the burst signals of back porch in cycles of 27MHz. The reset value is **16** in decimal.	RW
7	Reserved	These bits always read 0, and written are ignored.	R
6:0	BURSTW	The sub-carrier burst width inside back porch in cycles of 27MHz. The reset value is **61** in decimal.	RW

21.4.5 Chrominance configure registers (CFREQ, CPHASE, CFCFG)

This register is used to define chrominance sub-carrier frequency.

CFREQ **0x13050170**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CFREQ																															
RST	0	0	1	0	1	0	1	0	0	0	0	0	1	0	0	1	1	0	0	0	1	0	1	0	1	1	0	0	1	0	1	1

Bits	Name	Description	RW
32	CFREQ	Chrominance sub-carrier frequency.	RW

This register is used to define chrominance sub-carrier phase.

CPHASE **0x13050174**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	INITPH								ACTPH								Reserved														CCRSTP	
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	Name	Description	RW
31:24	INITPH	Initial phase of chrominance sub-carrier. Corresponding to upper 8 bits of CFREQ.	RW
23:16	ACTPH	This is added to chrominance sub-carrier angle (corresponding to upper 8 bits of CFREQ) in case of active video period.	RW
15:2	Reserved	These bits always read 0, and written are ignored.	R
1:0	CCRSTP	Chrominance clock reset period. After the reset, chrominance clock is set to INITPH. Besides this, chrominance clock is reset also to INITPH in case of chip reset.	RW

CCRSTP	Description
00	Every 8 field
01	Every 4 fields
10	Every 2 lines
11	Never

This register is used to configure chrominance filter.

CCFG **0x13050178**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CBBA								CRBA								CBGAIN								CRGAIN							
RST	0	0	1	1	1	0	1	1	0	0	1	1	1	0	1	1	1	0	0	0	1	0	0	1	1	0	0	0	1	0	0	1

Bits	Name	Description	RW
31:24	CBBA	Cb amplitude for burst period. The reset value is 59 in decimal, which corresponding to 59*4 = 236 ≈ (WHITEL – BLANKL) * 4 / 10 (±10%) = 224 ± 22 ≈ (WHITEL – BLANKL) * 3 / 7 (±3%) = 240 ± 7	RW
23:16	CRBA	Cr amplitude for burst period. The reset value is 59 in decimal. In PAL mode CRBA value is 59 in decimal and in NTSC mode CRBA value is 0 in decimal.	RW
15:8	CBGAIN	Cb gain. The reset value is 137 in decimal. CBGAIN=128 means no changing to the incoming Cb data.	RW
7:0	CRGAIN	Cr gain. The reset value is 137 in decimal. CRGAIN=128 means no changing to the incoming Cr data.	RW

21.5 Switch between LCD panel and TV set

LCD panel ➔ TV set switch

Step 1. Configure TVE (CVBS/S-Video, N/P, and etc), enable DAC.

Step 2. Disable LCDC. If data is from IPU, stop IPU. Then LCD panel is turned off.

Step 3. Configure LCDC for output via TVE.

Step 4. Configure TVE and LCDC pixel clock and enable TVE clock (CPM).

Step 5. If data is from IPU, start IPU. Then start LCDC.

Step 6. Enable TVE (TVECR.SWRST=0). Then data stream from LCDC is output to TV set via TVE.

TV set ➔ LCD panel switch

Step 1. Disable TVE (TVECR.SWRST=1). Then no signal is output to TV set.

Step 2. Disable TVE clock (CPM), and disable DAC.

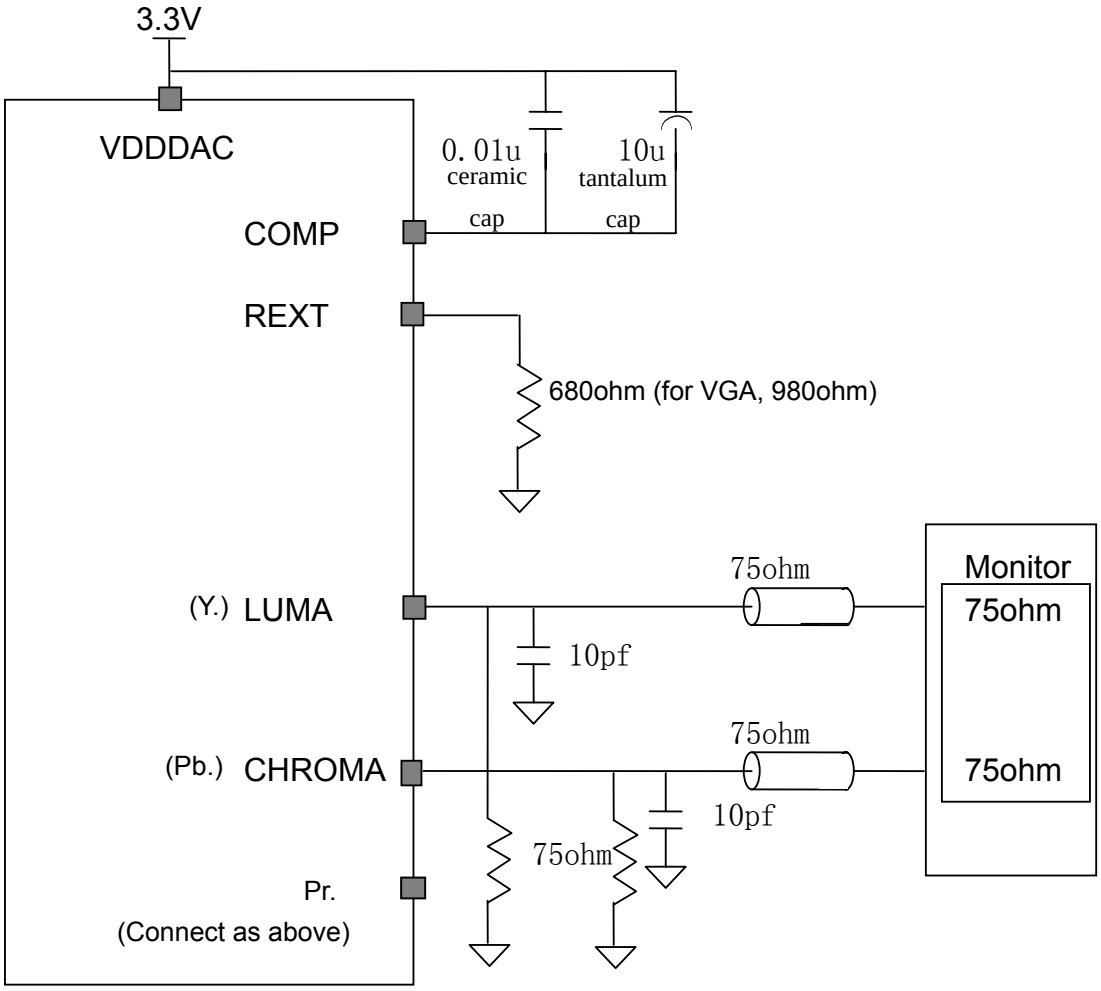
Step 3. Disable LCDC. If data is from IPU, stop IPU.

Step 4. Configure LCDC pixel clock. Configure LCDC for output to LCD panel.

Step 5. If data is from IPU, start IPU. Start LCDC. Then LCD panel is work.

21.6 DAC

21.6.1 DAC Connection



21.6.2 DAC DC Character

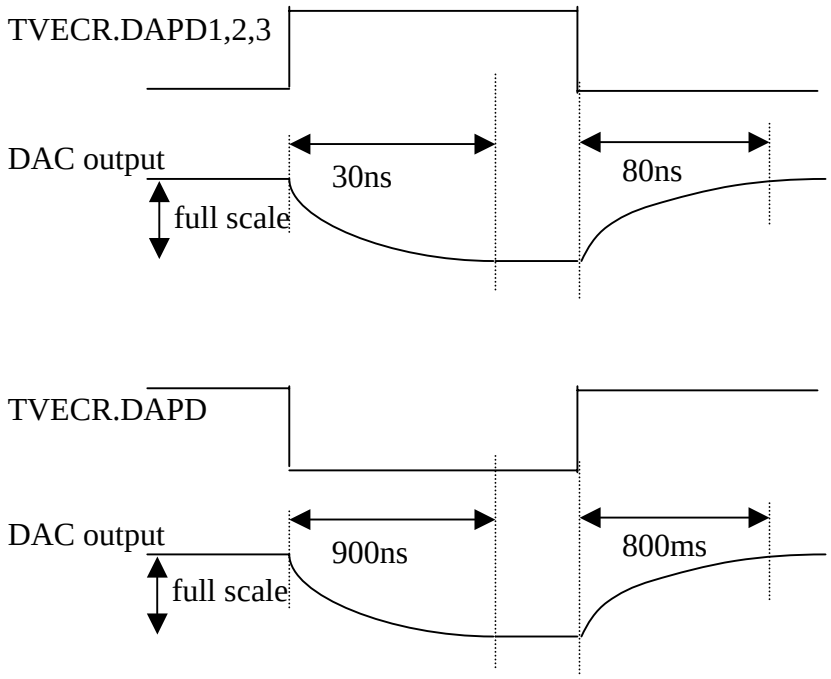
VDDDAC =3.3V; DVDD=1.8V; RL=37.5ohm, CL=10Pf; Temp = 25℃

Parameter	Symbol	Min	Type	Max	Unit
Operating voltage range	VDDDAC	3.0	3.3	3.6	V
Max output voltage	DVDD	--	1.278	--	V
DAC resolution		--	10	--	bits
Integral non-linearity error	INL	--	±1LSB	±1.5LSB	LSB
Differential non-linearity erro	DNL	--	±0.5LSB	±1LSB	LSB

21.6.3 DAC Power Down Setup Time

As the output current's max value per channel is 34.1mA, keep the DAC power down when you not use TV encoder.

The relate parameter is measured when VDDDAC = 3.3V and the temperature is 100 ℃.



22 EPD Controller

22.1 Overview

The EPD Controller affords an interface to transfer data from the LCD controller to the EPD.

Features:

- Supports EPD and compatible devices
- Supports different size of display panel
- Supports different width of pixel data
- Supports internal DMA operation and register operation

22.2 Pin Description

Table 22-1 EPD Pins Description

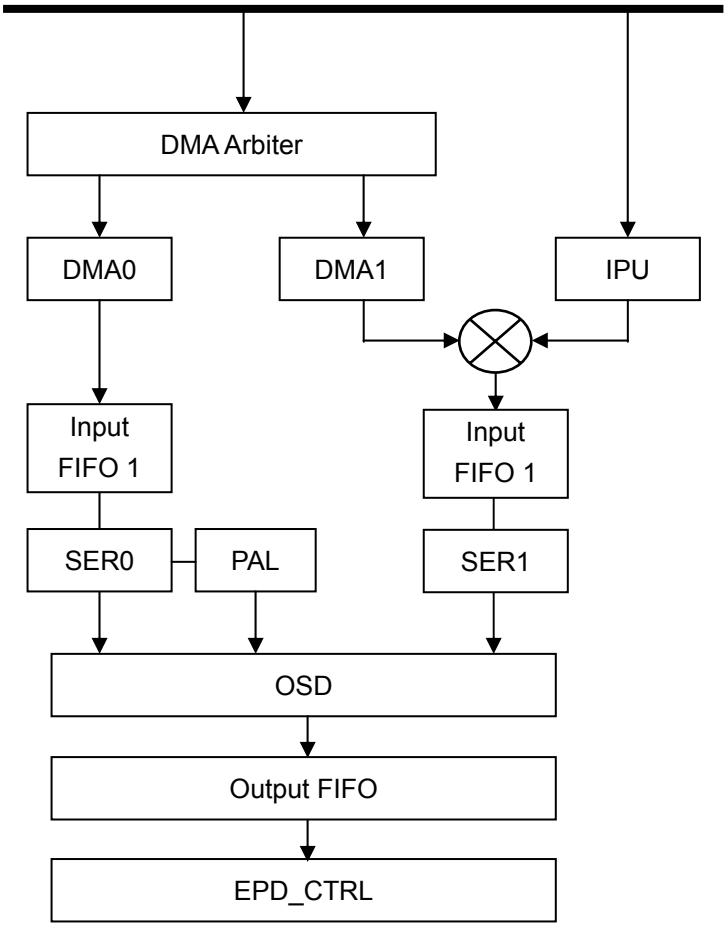
Name	I/O	Description
sdclk	Output	Source driver clock
sdce[7:0]	Output	Source driver chip enable
sdoe	Output	Source driver output enable
sdle	Output	Source driver date latch enable
sddo[15:0]	Output	Source driver data
sdshr	Output	Source driver data shift direction
gdclk	Output	Gate driver clock
gdsp	Output	Gate driver start pulse
gdoe	Output	Gate driver output enable
gdrl	Output	Gate driver data shift direction
pwr[7:0]	Output	Power management control signals
pwrcom	Output	Power management control signal

22.3 Pin Mapping

Pin	EPD
Lcd_pclk/ Slcd_clk	sdclk
Lcd_vsync/Slcd_cs	gdclk
Lcd_hsync/Slcd_rs	sdle
Lcd_de	sdoe
Lcd_ps	
Lcd_cls	
Lcd_rev	
Lcd_spl	
Lcd_dat17	sddo15
Lcd_dat16	sddo14
Lcd_dat15	sddo13
Lcd_dat14	sddo12
Lcd_dat13	sddo11
Lcd_dat12	sddo10
Lcd_dat11	sddo9
Lcd_dat10	sddo8
Lcd_dat9	sddo7
Lcd_dat8	sddo6
Lcd_dat7	sddo5
Lcd_dat6	sddo4
Lcd_dat5	sddo3
Lcd_dat4	sddo2
Lcd_dat3	sddo1
Lcd_dat2	sddo0
Lcd_dat1	sdshr
Lcd_dat0	gdoe
Lcd_lo6_o[5]	sdce7
Lcd_lo6_o[4]	sdce6
Lcd_lo6_o[3]	sdce1
Lcd_lo6_o[2]	sdce0
Lcd_lo6_o[1]	gdsp
Lcd_lo6_o[0]	gdrl
PB9	pwrcom
PB10	pwr0
PB11	pwr1

PB12	sdce2
PB13	sdce3
PB14	sdce4
PB15	sdce5
PB16	pwr2
PB17	pwr3
PD12	pwr4
PD13	pwr5
PE6	pwr6
PE7	pwr7

22.4 Block Diagram

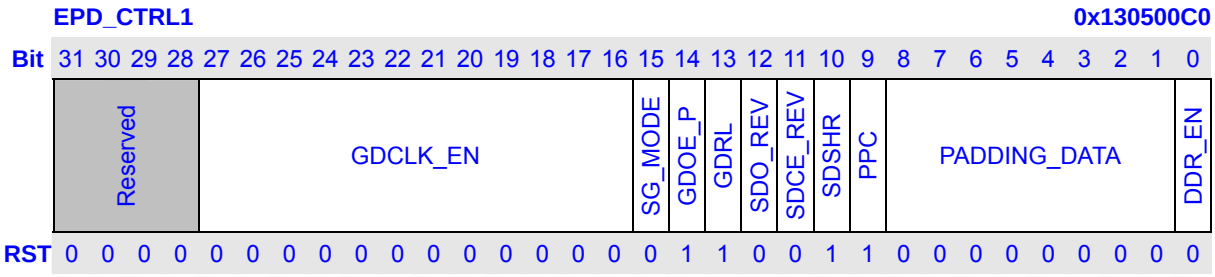


22.5 Register Description

Table 22-2 EPD Controller Registers Description

Name	RW	Reset Value	Address	Access Size
EPD_CTRL1	RW	0x00006600	0x130500C0	32
EPD_CTRL2	RW	0x00020000	0X130500C4	32
EPD_CTRL3	RW	0x00000000	0X130500C8	32
EPD_CTRL4	RW	0xff000000e	0X130500CC	32
EPD_CTRL5	RW	0x00000000	0X130500D0	32
EPD_CTRL6	RW	0x00000000	0X130500D4	32
EPD_CTRL7	RW	0x00000000	0X130500D8	32
EPD_CTRL8	RW	0x00000000	0X130500DC	32
EPD_CTRL9	RW	0x00000000	0X130500E0	32
EPD_VAT	RW	0x00000000	0x1305000C	32
EPD_DAH	RW	0x00000000	0x13050018	32
EPD_DAV	RW	0x00000000	0x1305001C	32
EPD_FSYNC	RW	0x00000000	0x13050004	32
EPD_LSYNC	RW	0x00000000	0x13050008	32

22.5.1 EPD Control1 Register (EPD_CTRL1)



Bits	Name	Description	RW
31:28	Reserved[1]	These bits always read 0, and written are ignored.	R
27:16	GDCLK_EN	Gate driver clock start point.	RW
15	SG_MODE	Source and gate driver timing mode select. 0: MODE 0; 1: MODE 1.	RW
14	GDOE_P	Gate driver output enable polarity.	RW
13	GDRL	Gate driver shift direction. 0: from omax to omin; 1: from omin to omax.	RW
12	SDO_REV	This bit select the parallel pixels output arrangement. 1: reversed; 0: not reversed.	RW
11	SDCE_REV	This bit configures the enable sequence for multiple chip	RW

		selects. 1: reversed; 0: not reversed.	
10	SDSHR	Source driver serial to parallel data shift direction. 1: from omin to omax; 0: from omax to omin.	RW
9	PPC	Pixels per clock. 0: 8; 1: 4.	RW
8:1	PADDING_DATA	Pixels needed to pad / 4.	RW
0	DDR_EN	Double data rate enable.	RW

22.5.2 EPD Control2 Register (EPD_CTRL2)

EPD_CTRL2 **0x130500C4**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PWR_DL01												PWR_OFF	PWR_ON	PCOM_P	SDCE1					SDOS								SDCN			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:20	PWR_DL01	Delay from PWR0 to PWR1. (in gdclk)	RW
19	PWR_OFF	Power off trigger.	RW
18	PWR_ON	Power on trigger.	RW
17	PCOM_P	PWRCOM polarity.	RW
16:13	SDCE1	Source driver chip enable start bits.	RW
12:4	SDOS	Source driver output size. (pixel number / 2)	RW
3:0	SDCN	Source driver chip number.	RW

22.5.3 EPD Control3 Register (EPD_CTRL3)

EPD_CTRL3 **0x130500C8**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				PWR_DL23												Reserved				PWR_DL12											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	PWR_DL23	Delay from PWR2 to PWR3. (in gdclk)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R

11:0	PWR_DL12	Delay from PWR1 to PWR2. (in gdclk)	RW

22.5.4 EPD Control4 Register (EPD_CTRL4)

EPD_CTRL4 **0x130500CC**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PWR7_P	PWR6_P	PWR5_P	PWR4_P	PWR3_P	PWR2_P	PWR1_P	PWR0_P	Reserved							PWRDN_M	DMA_M	FE_M	FCE_M	PWRUP_M	FE_INT_SEL	F_CANCEL	F_EN	Reserved			PADDING_SWAP	SDCE_P	SDLE_P	SDOE_P	GDCLK_P	GDSP_P
RST	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

Bits	Name	Description	RW
31	PWR7_P	PWR7 polarity.	RW
30	PWR6_P	PWR6 polarity.	RW
29	PWR5_P	PWR5 polarity.	RW
28	PWR4_P	PWR4 polarity.	RW
27	PWR3_P	PWR3 polarity.	RW
26	PWR2_P	PWR2 polarity.	RW
25	PWR1_P	PWR1 polarity.	RW
24	PWR0_P	PWR0 polarity.	RW
23:17	Reserved	These bits always read 0, and written are ignored.	R
16	PWRDN_M	Power down process finish interrupt mask.	RW
15	DMA_M	EPD dma stop interrupt mask.	RW
14	FE_M	Frame update finish interrupt mask.	RW
13	FCE_M	Frame cancel finish interrupt mask.	RW
12	PWRUP_M	Power up process finish interrupt mask.	RW
11	FE_INT_SEL	Frame end interrupt select. 1: int for each update; 0: int for each frame.	RW
10	F_CANCEL	Frame update cancel enable.	RW
9	F_EN	Frame update enable.	RW
8:6	Reserved	These bits always read 0, and written are ignored.	R
5	PADDING_SWAP	Source driver swap padding pixels. Used in combination with SDSHR.	RW

P_Swap	SDSHR	Padding data location
0	0	Padded at the end of the last chip
0	1	Padded at the beginning of the first chip
1	0	Padded at the beginning of the first chip
1	1	Padded at the end of last chip

4	SDCE_P	Source driver chip enable polarity.	RW
3	SDLE_P	Source driver latch enable polarity.	RW
2	SDOE_P	Source driver output enable polarity.	RW
1	GDCLK_P	Gate driver clock polarity.	RW
0	GDSP_P	Gate driver start pulse polarity.	RW

22.5.5 EPD Control5 Register (EPD_CTRL5)

EPD_CTRL5 **0x130500D0**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				GDCLK_DIS												MAX_FRM								Reserved		DMA_MODE	OBPP			OMODE	EPD_EN
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	GDCLK_DIS	Gate driver clock disable point.	RW
15:8	MAX_FRM	Work along with DMA_MODE, indicate max DMA transfer frame.	RW
7:6	Reserved	These bits always read 0, and written are ignored.	R
5	DMA_MODE	1: DMA will work in a auto-stop mode, when transfer frame equal to MAX_FRM, DMA will stopped, until you set MCTRL.DMASTART = 1 to restart a DMA transfer loop.(this bit valid when MCTRL.DMAMODE = 1)	RW
4:2	OBPP	Output bits-per-pixel. 1: 2bpp; 2: 4bpp.	RW
1	OMODE	Parallel output data size. 0: 8bit;1: 16bit.	RW
0	EPD_EN	Enable EPD controller.	RW

22.5.6 EPD Control6 Register (EPD_CTRL6)

EPD_CTRL6 **0x130500D4**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				GDSP_DIS												Reserved				GDSP_EN											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	GDSP_DIS	Gate driver start pulse disable point.	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	GDSP_EN	Gate driver start pulse start point.	RW

22.5.7 EPD Control7 Register (EPD_CTRL7)

EPD_CTRL7 **0x130500D8**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				SDOE_DIS												Reserved				SDOE_EN											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	SDOE_DIS	Source driver output enable end point.	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	SDOE_EN	Source driver output enable start point.	RW

22.5.8 EPD Control8 Register (EPD_CTRL8)

EPD_CTRL8 **0x130500DC**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				PWR_DL45												Reserved				PWR_DL34											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	PWR_DL45	Delay from PWR4 to PWR5. (in gdclk)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	PWR_DL34	Delay from PWR3 to PWR4. (in gdclk)	RW

22.5.9 EPD Control9 Register (EPD_CTRL9)

EPD_CTRL9 **0x130500E0**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				PWR_DL67												Reserved				PWR_DL56											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	PWR_DL67	Delay from PWR6 to PWR7. (in gdclk)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	PWR_DL56	Delay from PWR5 to PWR6. (in gdclk)	RW

22.5.10 Virtual Area Setting (EPD_VAT)

EPD_VAT **0x1305000C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				HT												Reserved				VT											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	HT	Horizontal Total size. (in SD clock, sum of display area and blank space)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	VT	Vertical Total size. (in GD clock, sum of display area and blank space)	RW

22.5.11 Display Area Horizontal Start/End Point (EPD_DAH)

EPD_DAH **0x13050018**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				HDS												Reserved				HDE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	HDS	Horizontal display area start. (in SD clock)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	HDE	Horizontal display area end. (in SD clock)	RW

22.5.12 Display Area Vertical Start/End Point (EPD_DAV)

EPD_DAV **0x1305001C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				VDS												Reserved				VDE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	VDS	Vertical display area start position. (in GD clock)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	VDE	Vertical display area end position. (in GD clock)	RW

22.5.13 Frame Synchronize Register (EPD_FSYNC)

EPD_FSYNC **0x13050004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				VPS												Reserved				VPE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	VPS	Frame sync start position, fixed to 0. (in GD clock)	R
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	VPE	Frame sync end position. (in GD clock)	RW

22.5.14 Line Synchronize Register (EPD_LSYNC)

EPD_LSYNC **0x13050008**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				HPS												Reserved				HPE											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	These bits always read 0, and written are ignored.	R
27:16	HPS	Line sync start position. (in SD clock)	RW
15:12	Reserved	These bits always read 0, and written are ignored.	R
11:0	HPE	Line sync end position. (in SD clock)	RW

22.6 Operation Guide

1 The EPD controller supports different size of display panels. The display area can be set by the following registers: EPD_VAT, EPD_DAH, EPD_DAV, EPD_FSYNC, and EPD_LSYNC.
2 Set EPD control parameters: signal polarity, power delay time, frame update number, .etc.
3 Set relate LCDC configure registers.
   a Configuration.
   * LCDCFG and LCDCTRL ()
   * LCDOSDC and LCDOSDCTRL (foreground0, 1 ‘s BPP only support BPP2, BPP4)
   * LCDXYP0, LCDXYP1, LCDSIZE0, LCDSIZE1 (foreground0,1 must cover total display area)
   * LCDALPHA (disable ALPHA)
   * EPD_CTRL5()
   b Set DMAC.
   * LCDIID
   * LCDDA0, LCDSA0, LCDFID0, LCDCMD0, LCDOFFS0, LCDPW0, LCDCNUM0, LCDDESSIZE0
   * LCDDA1, LCDSA1, LCDFID1, LCDCMD1, LCDOFFS1, LCDPW1, LCDCNUM1, LCDDESSIZE1
   c Enable LCDC.
4 Trigger power up process by setting register EPD_CTRL2.PWR_ON.
5 Enable frame update by setting register EPD_CTRL4.F_EN.
6 Power off after receiving the frame end interrupt by setting EPD_CTRL2.PWR_OFF.

23 Image Process Unit

23.1 Overview

IPU (Image process unit) contains Resize and CSC (color space conversion), which is used for image post processing.

23.1.1 Feature

- Location: AHB bus
- Input format
  - Separate frame: YUV /YCbCr (4:2:0, 4:2:2, 4:4:4, 4:1:1), RGB888
  - Packaged data: YUV422, **RGB888, RGB565, RGB555, YUV444**
- Output data format
  - RGB (565, 555, 888, AAA)
  - Packaged data YUV422
- Color convention coefficient: configurable (CSC enable)
- Minimum input image size (pixel): 4x4
- Maximum input image size (pixel): 4095x4095
- Maximum output image size (pixel)
  - Width: up to 4095 (without vertical resizing)
    up to **1280** (with vertical resizing)
  - Height: up to 4095
- Image resizing
  - Support bilinear
  - 0 and bi-cube zooming mode
  - Up scaling ratios up to 1:31 in fractional steps with 1/32 accuracy
  - Down scaling ratios up to 31:1 in fractional steps with 1/32 accuracy

**For more details, refer to* 1268H2606H*Special Instruction.*

23.2 Block

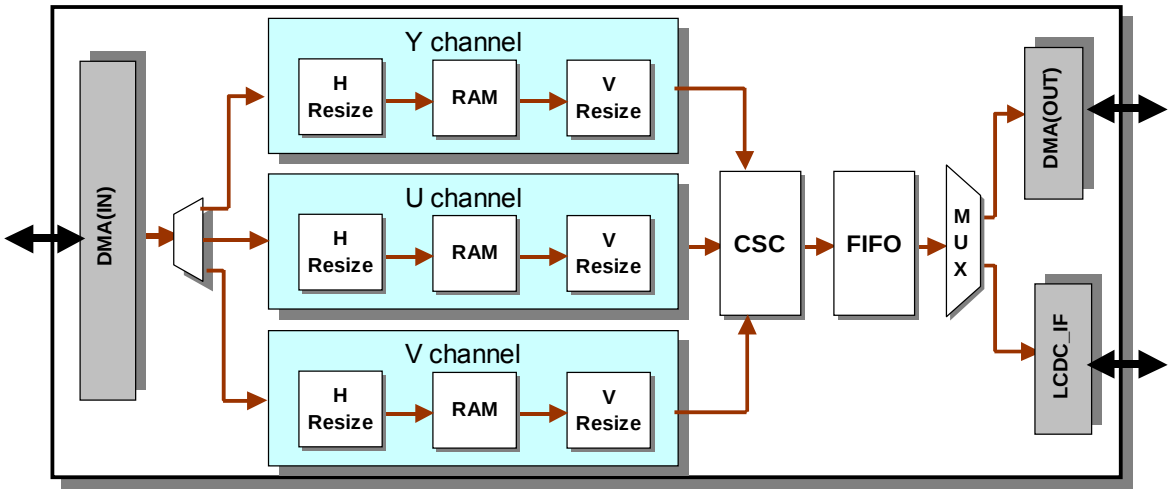


Figure 23-1 The Block about the IPUData flow

23.3 Data flow

23.3.1 Input data

- Separated YUV (or YcbCr/RGB888; the following use YUV for convenience) Frame case: Y, U, V data would be fetched from external memory by DMA burst read operation.
- Packaged YUV422 (or RGB888/ RGB565/ RGB555/YUV444) case: Packaged YUV(RGB888/ RGB565/ RGB555/YUV444) data would be fetched from external memory by DMA burst read operation.

23.3.2 Output data

- DMA output to external memory case: The output data format could be RGB (565, 555, 888) or YUV (package422), and the data would be stored to the external memory by DMA burst write operation.
- Flow into LCDC case: The output data format can be RGB or YUV (package), and the transfer would not cross AHB BUS.

23.3.3 Resize Coefficients LUT

The resize coefficients look up table is preset by software according to specific format (see for 2607H 23.4.30, 2608H 23.4.31 detail). There are 2 tables to support independent horizontal and vertical scaling. Each table has 32 entries that can accommodate up to 32 coefficient-groups.

23.4 Registers Descriptions

The physical address base for the address-mapped registers of IPU is **0x13080000.**
The following table will show all the register address.

Table 23-1 register list

NAME	Offset	Descript
1269H**IPU_CONTROL**	**0x0**	IPU global controller
1270H**IPU_STATUS**	**0x4**	IPU global status register
1271H**D_FMT**	**0x8**	IPU data format register
1272H**Y_ADDR**	**0xC**	Source Y (or R) base address
1273H**U_ADDR**	**0x10**	Source U (or G) base address
1274H**V_ADDR**	**0x14**	Source V (or B) base address
1275H**IN_FM_GS**	**0x18**	Input Geometric Size (height and width)
1276H**Y_STRIDE**	**0x1C**	Source Y frame stride
1277H**UV_STRIDE**	**0x20**	Source U and V frame stride
1278H**OUT_ADDR**	**0x24**	Result frame base address
1279H**OUT_GS**	**0x28**	Result frame size (height and width)
1280H**OUT_STRIDE**	**0x2C**	Result frame stride
1281H**RSZ_COEF_INDEX**	**0x30**	Resize Coefficients Table Index
1282H**CSC_C0_COEF**	**0x34**	Color conversion Coefficient
1283H**CSC_C1_COEF**	**0x38**	Color conversion Coefficient
1284H**CSC_C2_COEF**	**0x3C**	Color conversion Coefficient
1285H**CSC_C3_COEF**	**0x40**	Color conversion Coefficient
1286H**CSC_C4_COEF**	**0x44**	Color conversion Coefficient
1287H**HRSZ_COEF_LUT**	**0x48**	Horizontal Resize Coefficients Look Up Table
1288H**VRSZ_COEF_LUT**	**0x4C**	Vertical Resize Coefficients Look Up Table
1289H**CSC_OFSET_PARA**	**0x50**	Color conversion offset Coefficient
1290H**SRC_TLB_ADDR**	**0x54**	Base address of the source Y's physical address map table
1291H**DEST_TLB_ADDR**	**0x58**	Base address of the destination's physical address map table
1292H**TLB_MONITOR**	**0x60**	TLB monitor
1293H**IPU_ADDR_CTRL**	**0x64**	IPU address set controller
1294H**Y_ADDR_N**	**0x84**	Source Y base address of next frame
1295H**U_ADDR_N**	**0x88**	Source U base address of next frame
1296H**V_ADDR_N**	**0x8C**	Source V base address of next frame
1297H**OUT_ADDR_N**	**0x90**	Result frame base address of next frame
1298H**SRC_TLB_ADDR_N**	**0x94**	Base address of the source Y's physical address map table for next frame
1299H**DEST_TLB_ADDR_N**	**0x98**	Base address of the destination's physical address map table for next frame
1300H**TLB_CTRL**	**0x68**	TLB controller

23.4.1 IPU Control Register

IPU_CONTROL **0x0**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										BUS_OPT	CONF_MODE	ADDR_SEL	BURST_SEL	ZOOM_SEL	DFIX_SEL	FIELD_SEL	FIELD_CONF	DISP_SEL	DPAGE_MAP	SPAGE_MAP	LCDC_SEL	SPKG_SEL				IPU_RST	FM_IRQ_EN	CSC_EN	VRSZ_EN	HRSZ_EN	IPU_RUN	CHIP_EN
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:23	Reserved	Writing has no effect, read as zero.	R
22	BUS_OPT	IPU DMA bus optimization selector. 0: no bus optimization 1: bus optimization when BURST_SEL is 1	RW
21	CONF_MODE	IPU configure mode selector. 0: IPU's registers can be changed any time 1: IPU's registers only can be changed when it is not busy	RW
20	ADDR_SEL*3	IPU address mode selector. 0: IPU source and destination address only can be modified when IPU is free, just like it is in XBurst JZ4750 processor 1: IPU source and destination address can be modified anytime	RW
19	BURST_SEL	IPU DMA burst mode selector. 1: 32 burst (DDR is suggested) 0: 16 burst	RW
18	ZOOM_SEL	IPU rooming mode selector. 1: bi-cube 0: bilinear	RW
17	DFIX_ SEL	Fixed destination address choose. (Valid when LCDC_SEL == 0) 0: not use the fixed address 1: use the fixed address	RW
16	FIELD_SEL *1	Destination field choose. (Valid when FIELD_CONF_EN == 1) 0: top field 1: bottom field	RW
15	FIELD_CONF_EN *1	FIELD_SEL mask. 0: do not change FIELD_SEL 1: configure FIELD_SEL	W

		Read as zero.	
14	DISP_ SEL	Destination display choose. 0: frame display mode 1: field display mode	RW
13	DPAGE_MAP	Destination address page mapping choose. 0: not use the page mapping 1: use the page mapping	RW
12	SPAGE_MAP	Source address page mapping choose. 0: not use the page mapping 1: use the page mapping	RW
11	LCDC_SEL	Output data destination choose. 0: output to external memory 1: output to LCDC FIFO	RW
10	SPKG_SEL	Input data case choose. 0: Separated Frame 1: Packaged Frame	RW
9:8	Reserved	Read as 0.	R
7	IPU_STOP*4	Stop IPU. 1: stop IPU. When stop IPU, the end flag will be pull up to 1.	W
6	IPU_RST *2	Reset IPU. Writing 1: reset IPU; 0: no effect. Read as zero.	W
5	FM_IRQ_EN	Frame process finish interrupt enable. 1: enable; 0: disable.	RW
4	CSC_EN	CSC enable. 1: enable; 0: disable.	RW
3	VRSZ_EN	Vertical Resize enable. 1: enable; 0: disable.	RW
2	HRSZ_EN	Horizontal Resize enable. 1: enable; 0: disable.	RW
1	IPU_RUN	Run the IPU. 1: run. Software just can set 1 to IPU_RUN.	RW
0	CHIP_EN	IPU chip enable. 1: enable; 0: disable.	RW

NOTES:

1 *[1]: The FIELD_SEL will work when the DISP_ SEL is 1, which indicates the IPU is under the field display mode. And the IPU will output the picture from the initial field (top or bottom) to the next field (bottom or top) automatically when the current field has been outputted or stopped. The initial field can be configured by setting the FIELD_SEL to 0 or 1 with FIELD_CONF_EN is 1. The FIELD_CONF_EN is just the trigger that controls the FIELD_SEL‘s valuation.

2 *[2]: Setting 1 to IPU_RST can reset all registers except the CHIP_EN immediately, but user must make sure the IPU is free when need to assert IPU_RST.

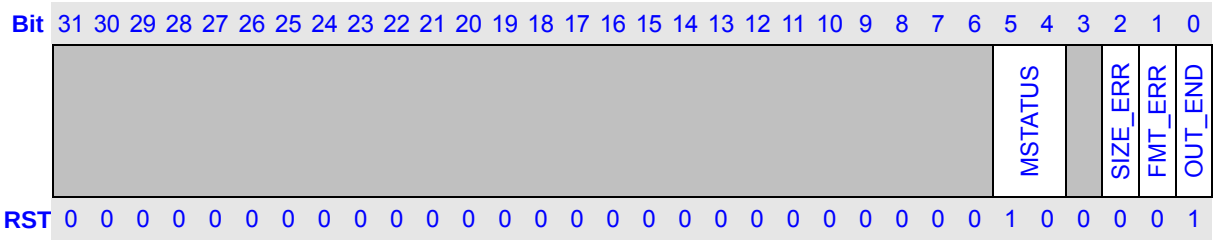
3 *[3]: When ADDR_SEL is set to 0, the address set method is the same as XBurst JZ4750 processor, and the frame address of IPU can be set just like the way in XBurst JZ4750 processor, which limits the address setting time to IPU none working period (after frame

end-flag). When the ADD_SEL is 1, the above limitation is released. The addresses of IPU can be changed at anytime. It just needs to set the correspond bits in IPU_ADDR_CTRL to 1 to tell IPU that new address can be used, after the addresses are changed.

4 *[4]: The IPU_STOP is used to stop IPU in anytime in save mode. When the IPU_STOP has been written to 1, the IPU need some time to stop. And the user can monitor the IPU_STATUS.OUT_END to make sure the IPU has been stopped.

23.4.2 IPU Status Register

IPU_STATUS **0x4**



Bits	Name	Description	R/W
31:2	Reserved	Writing has no effect, read as zero.	R
5:4	MSTATUS	IPU main status. 00: IPU is free and waiting the CPU or LCDC to get the IPU's control 01: IPU is running now 10: IPU is under the control of CPU 11: reserved	R
3	Reserved	Writing has no effect, read as zero.	R
2	SIZE_ERR	The size error flag. 1: size error; 0: size ok.	R
1	FMT_ERR	IPU format error flag. 1: format error; 0: format OK.	R
0	OUT_END *1	Output termination flag. 1: finished; 0: not finished.	R/W

NOTES:

1 *1: If **IPU_CONTROL**.FM_IRQ_EN has been set 1, once OUT_END is set value 1 which denotes a frame's post process done, an low level active interrupt request will be issued until corresponding software handler read IPU_STATUS and clean end flag.

When the **IPU_CONTROL**.FM_LCDC_SEL has been set 1, and the IPU has finished one transfer, the LCDC and CPU need to occupy the IPU control. The IPU will monitor the request signal from LCDC and the read signal from the CPU, then it will determine whether dre-configure itself by the CPU if the CPU read first or output the same frame to LCDC again if the LCDC get the control.

23.4.3 IPU address control register

IPU_ADDR_CTRL **0x64**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																											PTD_RY	PTS_RY	D_RY	V_RY	U_RY	Y_RY
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Bits	Name	Description	R/W
31:6	Reserved	Writing has no effect, read as zero.	R
5	PTD_READY	New destination TLB base address ready. (Only used when IPU_CONTROL.ADDR_SEL==1&&IPU_CONTROL.DPAGE_MAP == 1)	RW
4	PTS_READY	New source TLB base address ready. (only used when IPU_CONTROL.ADDR_SEL==1&&IPU_CONTROL.SPAGE_MAP == 1)	RW
3	D_READY	New destination address ready.	RW
2	V_READY	New source V address ready.	RW
1	U_READY	New source U address ready.	RW
0	Y_READY	New source Y address ready.	RW

NOTES:

1 When the xx_READY bit has been set 1, the IPU will use the new corresponding address in the next frame, and use the old address value whose corresponding bit in **IPU_ADDR_CTRL** is 0.

2 **IPU_ADDR_CTRL** works when **IPU_CONTROL. ADDR_SEL** is 1.

3 When the **IPU** has fetched the new address, it will clear the **IPU_ADDR_CTRL** to 0.

23.4.4 Data Format Register

D_FMT **0x8**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
							RGB_888	RGB_OUT_OFT			OUT_FMT			YUV_PKG_OUT_OFT													IN_RGB_FMT		IN_OFT		IN_FMT	
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

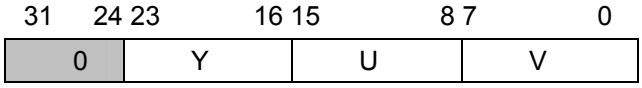
Bits	Name	Description	R/W
31:26	Reserved	Writing has no effect, read as zero.	R

25	RGB_888_ OUT_FMT	RGB888 output format indicator. (only used in RGB888 out OUT_FMT == 010) 0: the low 24 bits will be the pixel in a word 1: the high 24 bits will be the pixel in a word	RW
24:22	RGB_OUT_ OFT	Output data packaged offset. (only used in RGB out OUT_FMT != 011) 000: RGB 001: RBG 010: GBR 011: GRB 100: BRG 101: BGR Others: reserved	RW
21:19	OUT_FMT	Indicates the destination data format. 000: RGB555 001: RGB565 010: RGB888 011: YUV422 package 100: RGBAAA(R(or G or B) is 10 bits wide)	RW
18:16	YUV_PKG_ OUT_OFT	Output data packaged offset. (only used in CSC disable case and in the source is YUV422 packaged case (**IPU_CONTROL.** SPKG_SEL == 1)) 000: Y1UY0V 001: Y1VY0U 010: UY1VY0 011: VY1UY0 100: Y0UY1V 101: Y0VY1U 110: UY0VY1 111: VY0UY1	RW
15: 6	Reserved	Writing has no effect, read as zero.	R
3:2	IN_OFT	Input data packaged offset. (when the source is YUV422 packaged case (**IPU_CONTROL.** SPKG_SEL == 1 && IN_FMT == 01)) 00: Y1UY0V 01: Y1VY0U 10: UY1VY0 11: VY1UY0	RW
1:0	IN_FMT	Indicates the source data format. When **IPU_CONTROL.**SPKG_SEL == 0 00: YUV 4:2:0 01: YUV 4:2:2 10: YUV 4:4:4 11: YUV 4:1:1 When **IPU_CONTROL.**SPKG_SEL == 1 00: RGB555 01: YUV 4:2:2 10: RGB888 or YUV444 11: RGB565	RW

NOTES:

1 When the source frame is packed RGB, the **IPU_CONTROL.** SPKG_SEL must be 1 and IN_FMT must be 10, and when the source frame is packed YUV, the **IPU_CONTROL.** SPKG_SEL must be 1 and IN_FMT must be 01.

2 When the source frame is packed YUV444, the data format about a pixel should be as following:



23.4.5 Input Y Data Address Register

Y_ADDR **0xc**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Y_ADDR																														0	0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:0	Y_ADDR *1	In separated Frame case, it indicates the source Y (or R) data buffer's start address. In source YUV422 package case, it indicates the start Address of the packaged Frame.	RW

NOTES:

1 When the **IPU_CONTROL**.SPAGE_MAP == 1, the Y_ADDR should be the start virtual address.

2 Y_ADDR should be **word align**.

23.4.6 Input U Data Address Register

U_ADDR **0x10**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	U_ADDR																														0	0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:0	U_ADDR *1	The source U data buffer's start address of separated frame case.	RW

NOTES:

1 When the IPU_CONTROL.SPAGE_MAP == 1, the U_ADDR should be the start virtual address.

2 U_ADDR should be **word align**.

23.4.7 Input V Data Address Register

V_ADDR **0x14**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	V_ADDR																														0	0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:0	V_ADDR	The source V data buffer's start address of separated Frame case.	RW

NOTES:

1 When the IPU_CONTROL.SPAGE_MAP == 1, the V_ADDR should be the start virtual address.

2 V_ADDR should be word align.

23.4.8 Input source TLB base address

SRC_TLB_ADDR **0x54**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SRC_TLB_ADDR																														0	0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:0	SRC_TLB_ ADDR	The SOURCE TLB base address. (This register will act when the IPU_CONTROL.PAGE_MAP==1)	RW

23.4.9 Destination TLB base address

DEST_TLB_ADDR **0x58**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DEST_TLB_ADDR																														0	0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:0	DEST_TLB_ ADDR	The destination frame's TLB base address. (This register will work when the IPU_CONTROL.DPAGE_MAP==1)	RW

23.4.10 TLB monitor

TLB_MONITOR **0x60**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																							MIS_CNT									
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:11	Reserved	Read as 0.	
10:0	MIS_CNT	The TLB cache miss counter.	RW

23.4.11 TLB controller

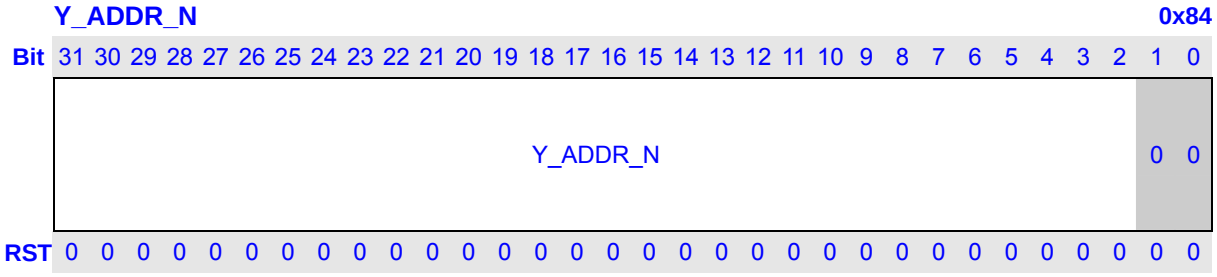
TLB_CTRL **0x68**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									DEST_ BURST				DEST_PAGE												SRC_ BURST				SRC_PAGE			
RST	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

Bits	Name	Description	R/W
31:25	Reserved	Read as 0.	
23:20	DEST_BURST	The destination TLB burst length.	RW

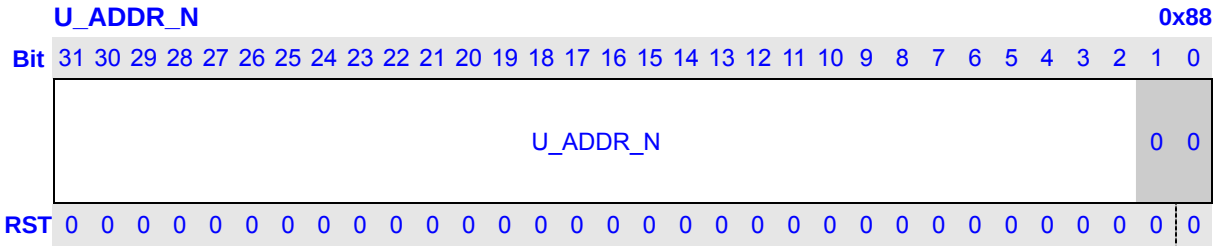
		1: 1; 2:2; 4:4; 8:8.	
19:16	DEST_PAGE	The destination TLB page size. 1: 4K; 2: 8K; 4:16K; 8:32K.	RW
15:4	Reserved	Read as 0.	
7:4	SRC_BURST	The sources TLB burst length. 1: 1; 2:2; 4:4; 8:8.	RW
3:0	SRC_PAGE	The source TLB page size. 1: 4K; 2: 8K; 4:16K; 8:32K.	RW

23.4.12 Input Y Data Address of next frame Register



Bits	Name	Description	R/W
31:0	Y_ADDR_N	In separated Frame case, it indicates the source Y (or R) data buffer's start address of the next frame. In package case, it indicates the start address of the packaged Frame of the next frame.	R

23.4.13 Input U Data Address of next frame Register



Bits	Name	Description	R/W
31:0	U_ADDR_N	The source U (G) data buffer's start address of the next frame in separated frame case.	RW

NOTES:

1 When the IPU_CONTROL.SPAGE_MAP == 1, the U_ADDR_N will be the start virtual address.

23.4.14 Input V Data Address of next frame Register

V_ADDR_N **0x8C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	V_ADDR_N																														0	0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:0	V_ADDR_N	The source V (B) data buffer's start address of the next frame in separated frame case.	RW

NOTES:

1 When the IPU_CONTROL.SPAGE_MAP == 1, the V_ADDR_N will be the start virtual address.

23.4.15 Source TLB base address of next frame

SRC_TLB_ADDR_N **0x94**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SRC_TLB_ADDR_N																														0	0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:0	SRC_TLB_ADDR_N	The TLB base address about the next source frame's data which will be DMA in.	R

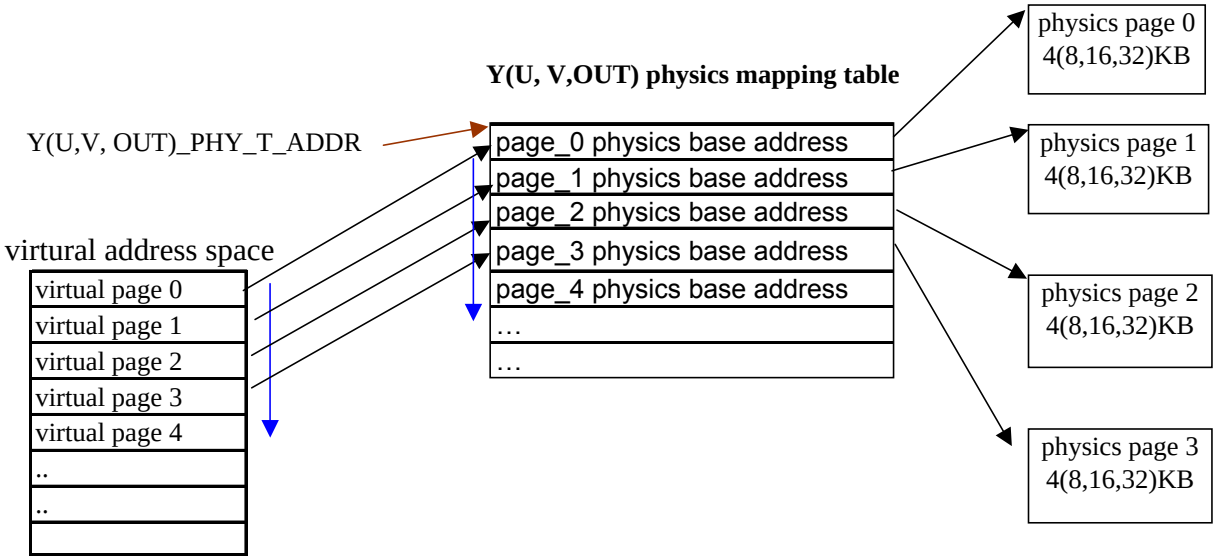
23.4.16 Destination TLB base address of next frame

DEST_TLB_ADDR_N **0x98**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DEST_TLB_ADDR_N																														0	0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

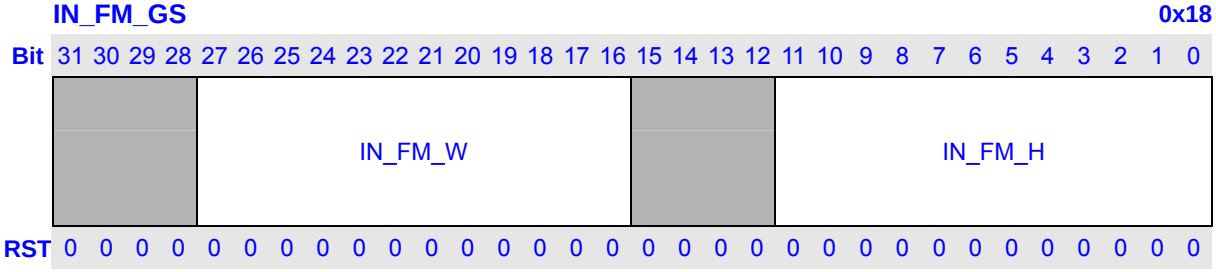
Bits	Name	Description	R/W
31:0	DEST_TLB_ADDR_N	The TLB base address about the next frame's data that will be DMA out.	R

23.4.17 ADDRESS Mapping



The Y (U, V, OUT)_PHY_T_ADDR should store the **base address** of the Y (U, V, OUT) physics-mapping table. In the Y (U, V, OUT) physics-mapping table, it should contain different physics page base address, and the physics page must be 4(8, 16, 32)KB align.

23.4.18 Input Geometric Size Register

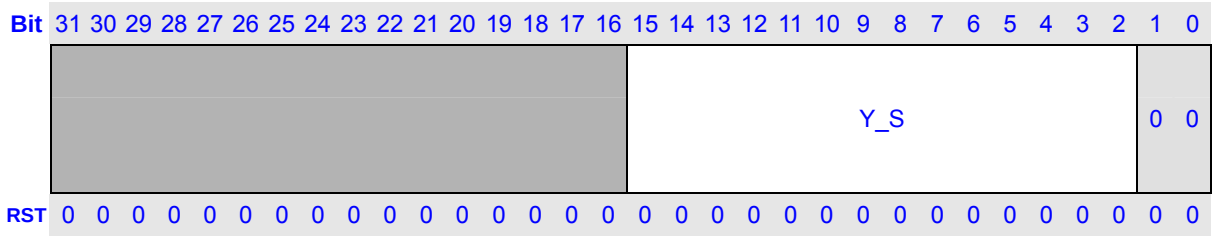


Bits	Name	Description	R/W
31:28	Reserved	Writing has no effect, read as zero.	R
27:16	IN_FM_W	The width of the input frame (unit: byte). Y data width is the same as this value while U/V or Cb/Cr data width should do relatively zoom in according to the source data format. And **in the source package pattern**, this value should be the Y data width also.	RW
15:12	Reserved	Writing has no effect, read as zero.	R

11:0	IN_FM_H	The height of the input frame (unit: byte). Y data width is same as this value while U/V or Cb/Cr data width should do relatively zoom in according to the source data format.	RW

23.4.19 Input Y Data Line Stride Register

Y_STRIDE **0x1c**



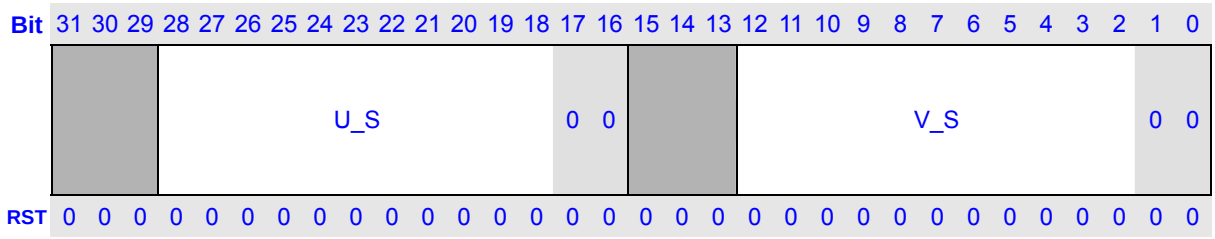
Bits	Name	Description	R/W
31:16	Reserved	Writing has no effect, read as zero.	R
15:0	Y_S	The line stride of the source Y data in the external memory of separated Frame case or packaged Frame stride(Unit: byte).	RW

NOTES:

1 Y_S should be world align.

23.4.20 Input UV Data Line Stride Register

UV_STRIDE **0x20**



Bits	Name	Description	R/W
31:29	Reserved	Writing has no effect, read as zero.	R
28:16	U_S	The line stride of the source U data in the external memory.	RW
15:13	Reserved	Writing has no effect, read as zero.	R
12:0	V_S	The line stride of the source V data in the external memory.	RW

NOTES:

1 U_S and V_S should be word align and unit is byte.

23.4.21 Output Frame Start Address Register

OUT_ ADDR																														0x24
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OUT_ ADDR																														0	0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:0	OUT_ ADDR *1	The output buffer's start address.	RW

NOTES:

1 *1: When the IPU_CONTROL.DPAGE_MAP == 1, the OUT_ADDR should be the start virtual address.

2 it should be word align.

23.4.22 Output Data Address of next frame Register

OUT_ ADDR_N																														0x90
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	OUT_ ADDR_N																														0	0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:0	OUT_ ADDR *1	The output buffer's start address.	RW

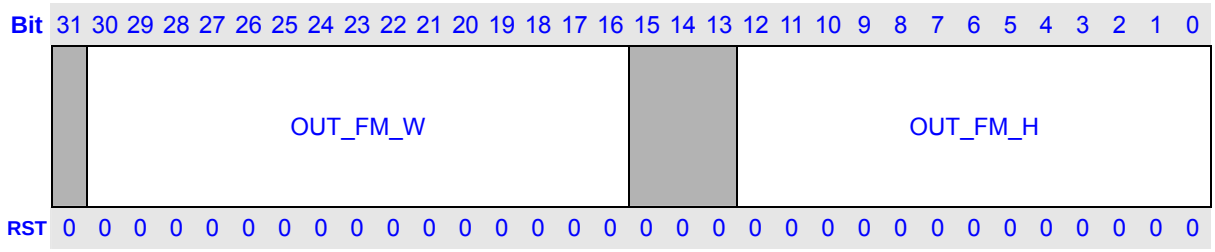
NOTES:

1 *1: When the IPU_CONTROL.DPAGE_MAP == 1, the OUT_ADDR should be the start virtual address.

2 it should be word align.

23.4.23 Output Geometric Size Register

OUT_GS **0x28**



Bits	Name	Description	R/W
31	Reserved	Writing has no effect, read as zero.	R
30:16	OUT_FM_W	The width of the output destination frame (unit: byte).	RW
15:13	Reserved	Writing has no effect, read as zero.	R
12:0	OUT_FM_H	The height of the output destination frame (unit: byte).	RW

NOTES:

1 In the package YUV out pattern, the OUT_FM_W should be the **pixel number** in a line.

2 In the **RGB** out pattern, the OUT_FM_W should be the **data space** width in the RAM.

3 In the out package pattern, the OUT_FM_W **must be** even number, and otherwise IPU will not run.

e.g.

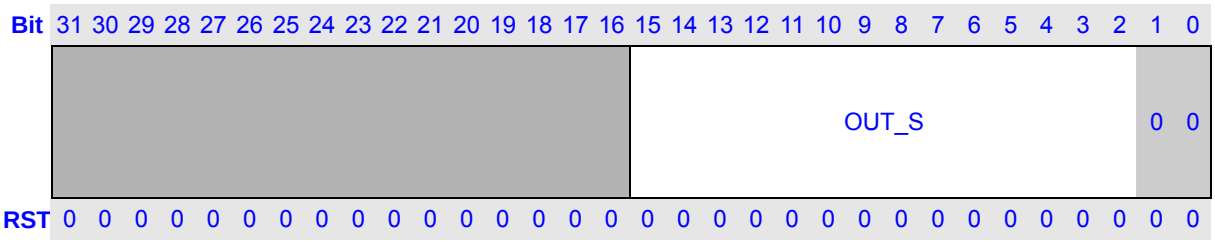
Package YUV out: 480x273 (OUT_FM_W: 480 OUT_FM_H: 273)

RGB 888 (or AAA) out: 480x273 (OUT_FM_W: 480*4 OUT_FM_H: 273)

RGB 555 (or 565) out: 480x273 (OUT_FM_W: 480*2 OUT_FM_H: 273)

23.4.24 Output Data Line Stride Register

OUT_STRIDE **0x2c**



Bits	Name	Description	R/W
31:16	Reserved	Writing has no effect, read as zero.	R
15:0	OUT_S	The line stride of the destination data buffer in the external memory(Unit: byte).	RW

23.4.25 CSC C0 Coefficient Register

CSC_C0_COEF **0x34**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																					C0_COEF											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	C0_COEF	The C0 coefficient of the YUV/YCbCr to RGB conversion. C0_COEF = [C0 * 1024 + 0.5].	RW
NOTE: R = C0*(Y –LUMA_OF) + C1*(Cr-CHROM_OF) G = C0*(Y – LUMA_OF) – C2*(Cb-CHROM_OF) – C3*(Cr-CHROM_OF) B = C0*(Y – LUMA_OF) + C4*(Cb-CHROM_OF)			

23.4.26 CSC C1 Coefficient Register

CSC_C1_COEF **0x38**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																					C1_COEF											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	C1_COEF	The C1 coefficient of the YUV/YCbCr to RGB conversion. C1_COEF = [C1 * 1024 + 0.5].	RW
NOTE: R = C0*(Y – LUMA_OF) + C1*(Cr-CHROM_OF) G = C0*(Y – LUMA_OF) – C2*(Cb-CHROM_OF) – C3*(Cr-CHROM_OF) B = C0*(Y – LUMA_OF) + C4*(Cb-CHROM_OF)			

23.4.27 CSC C2 Coefficient Register

CSC_C2_COEF 0x3C

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																					C2_COEF											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	C2_COEF	The C2 coefficient of the YUV/YCbCr to RGB conversion. C2_COEF = [C2 * 1024 + 0.5].	RW

NOTE:

R = C0*(Y – LUMA_OF) + C1*(Cr-CHROM_OF)

G = C0*(Y – LUMA_OF) – C2*(Cb-CHROM_OF) – C3*(Cr-CHROM_OF)

B = C0*(Y – LUMA_OF) + C4*(Cb-CHROM_OF)

23.4.28 CSC C3 Coefficient Register

CSC_C3_COEF 0x40

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																					C3_COEF											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	C3_COEF	The C3 coefficient of the YUV/YCbCr to RGB conversion. C3_COEF = [C3 * 1024 + 0.5].	RW

NOTE:

R = C0*(Y – LUMA_OF) + C1*(Cr-CHROM_OF)

G = C0*(Y – LUMA_OF) – C2*(Cb-CHROM_OF) – C3*(Cr-CHROM_OF)

B = C0*(Y – LUMA_OF) + C4*(Cb-CHROM_OF)

23.4.29 CSC C4 Coefficient Register

CSC_C4_COEF **0x44**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																					C4_COEF										
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:12	Reserved	Writing has no effect, read as zero.	R
11:0	C4_COEF	The C4 coefficient of the YUV/YCbCr to RGB conversion. C4_COEF = [C4 * 1024 + 0.5].	RW

NOTE:

R = C0*(Y – LUMA_OF) + C1*(Cr-CHROM_OF)

G = C0*(Y – LUMA_OF) – C2*(Cb-CHROM_OF) – C3*(Cr-CHROM_OF)

B = C0*(Y – LUMA_OF) + C4*(Cb-CHROM_OF)

23.4.30 Resize Coefficients Table Index Register

RSZ_COEF_INDEX **0x30**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													HE_IDX															VE_IDX				
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

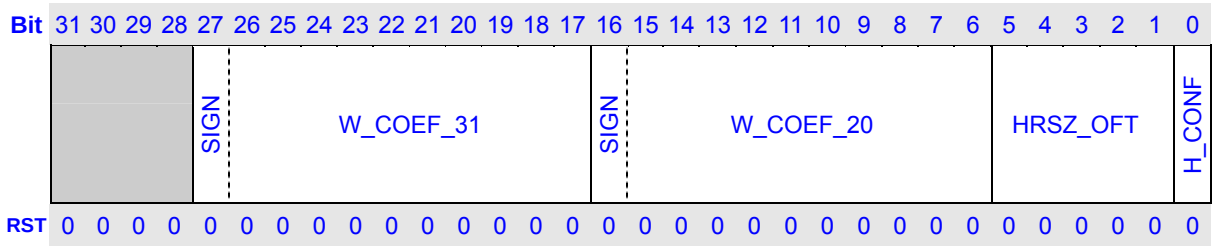
Bits	Name	Description	R/W
31:21	Reserved	Writing has no effect, read as zero.	R
20:16	HE_IDX *1	Indicates the end address of the horizontal resize look up table.	RW
15:5	Reserved	Writing has no effect, read as zero.	R
4:0	VE_IDX *1	Indicates the end address of the vertical resize look up table.	RW

NOTES:

1 The HE_IDX (VE_IDX) should be the depth of the horizontal (vertical) resize look up table **minus 1**, and how to get HE_IDX or VE_IDX, please refer to 3.34.

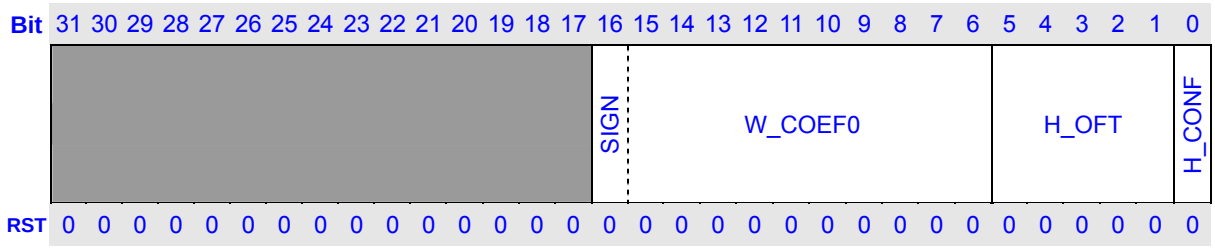
23.4.31 Horizontal Resize Coefficients Look Up Table Register group

HRSZ_COEF_LUT (bi-cube) **0x48**



Bits	Name	Description	R/W
15:12	Reserve	Writing has no effect, read as zero.	W
27:17	W_COEF_31	Weighting coefficients for pix_3 or pix_1.	W
16:6	W_COEF_20	Weighting coefficients for pix_2 or pix_0.	W
5:1	HRSZ_ OFT	Horizontal Resize pixel offset.	W
0	H_CONF	Start to configure the horizontal resize table, read as zero: 1: start.	W

HRSZ_COEF_LUT (bilinear) **0x48**



Bits	Name	Description	R/W
31:10	Reserve	Writing has no effect, read as zero.	R
16:6	W_COEF0	Weighting coefficients for pix_0.	W
5:1	H_OFT	Horizontal Resize pixel offset.	W
0	H_CONF	Start to configure the horizontal resize table, read as zero: 1: start.	W

NOTES:

1 The **bilinear zooming** weighting coefficients should be calculated as following: because it is 11 bits length, a one bit is the sign-bit, so, that is to say the precision is 1/512.

For up-scaling,

$W_k = 1 – (k*n/m – [k*n/m]), k = 0, 1, … m-1.$

For down-scaling,

```
for (t=0, k=0; k < n; k++)
{
```

If ([(t*n+1)/m] – k >=1) { W_k = 0;}
else if ((t*n+1)/m – k == 0) { W_k = 1; t++;}
else { W_k = 1 – ((t*n+1)/m – [t*n/m]); t++;}
}

W_COEF_k = [512 * W_k] (stands for get the rounding integer, [20.33] = 20 while [20.66] = 21)

Here n stands for original pixel points, m stands for pixel points after resize. For example down-scaling 5:3, n = 5, m = 3. Moreover, m and n are prime, that is, for example 8:2 should be converted to 4:1.

When IPU_CONTROL.RSZ_EN set as 1 and m:n = 1:1, all coefficients should be calculated as up-scale case.

2 The **bi-cube zooming** weighting coefficients should be calculate as following: because it is 10 bits length, that is to say the precision is 1/512.

Step1:

For up-scaling,

W_k = 1 – (k*n/m – [k*n/m]), k = 0, 1, … m-1.

For down-scaling,

for (t=0, k=0; k < n; k++)
{
If ([(t*n+1)/m] – k >=1) { W_k = 0;}
else if ((t*n+1)/m – k == 0) { W_k = 1; t++;}
else { W_k = 1 – ((t*n+1)/m – [t*n/m]); t++;}
}

W_COEF_k = [512 * W_k] (stands for get the rounding integer, [20.33] = 20 while [20.66] = 21)

Here n stands for original pixel points, m stands for pixel points after resize. For example down-scaling 5:3, n = 5, m = 3. Moreover, m and n are prime, that is, for example 8:2 should be converted to 4:1.

When IPU_CONTROL.RSZ_EN set as 1 and m:n = 1:1, all coefficients should be calculated as up-scale case.

Step 2:

After calculate the Wk, then using the following rule to get the coefficients:

```
double SinXDivX(double x)
{
    const float a = -1; //a can be a=-2,-1,-0.75,-0.5 and so on to control the blur level
    double x2=x*x;
    double x3=x2*x;
    if (x<=1)
    return (a+2)*x3 - (a+3)*x2 + 1;
    else if (x<=2)
    return a*x3 - (5*a)*x2 + (8*a)*x - (4*a);
    else
    return 0;
```

}

W0 = 512*SinXDivX(1+Wk) W1 = 512*SinXDivX(Wk)

W2 = 512*SinXDivX(1-Wk) W3 = 512*SinXDivX(2-Wk)

Step 3:

And then the zooming weight coefficient should set to IPU as following:

Prepare: Set H_CONF to 1

a set W4n+1 & W4n+0

b set W4n+3 & W4n+2

Index(n)	step	W0	W1	step	W2	W3
0	0	-34	129	1	100	-19
1	2	-45	45	3	77	-33
2	4	-12	122	5	94	-56
3	6	-13	230	7	123	-77
4	8	-23	11	9	45	-100
5	10	-19	87	11	69	-90
6	12	-12	79	13	148	-8

3 Calculate the H_OFT.

Step 1: calculate the line in enable flag (IN_EN) and out enable flag (OUT_EN) table.

IN_EN: In down scale case, IN_EN always equals 1.

In up scale case,

For (i=0, k=0; k < m; k++) {

If(i<= k*n/m) { IN_EN [k] = 1; i++;}

else { IN_EN [k] =0;}

OUT_EN: In up scale case, OUT_EN always equals 1.

In down scale case,

For (t=0, k=0; k < n; k++) {

If([(t*n+1)/m] – k >=1)

OUT_EN [k] = 0;

else {OUT_EN[k]=1; t++;}

}

Step 2: calculate the H_MAX_LUT.

H_MAX_LUT = max (m, n) – 1

*Step 3: C*alculate the LUT.

```
int hoft_table_buf[33];
int hcoef_table_buf[33];
int voft_table_buf[33];
int vcoef_table_buf[33];
int *hoft_table = &hoft_table_buf[1];
int *hcoef_table = &hcoef_table_buf[1];
int *voft_table = &voft_table_buf[1];
int *vcoef_table = &vcoef_table_buf[1];
```

```
int in_oft_tmp = 0;
int hcoef_real_heiht = 0   ;
int vcoef_real_heiht = 0   ;
int coef_tmp = 0   ;
j = -1 ;
 for (i=0; i<=H_MAX_LUT+1; i++)
 {
      if ( h_lut[i].out_n )
      {
           hoft_table[j] = (h_lut[i].in_n == 0)? 0: in_oft_tmp;
           hcoef_table[j] = coef_tmp;
           coef_tmp = h_lut[i].coef;
           in_oft_tmp = h_lut[i].in_n==0? in_oft_tmp : h_lut[i].in_n ;
           j++;
       }
       else
       in_oft_tmp = h_lut[i].in_n + in_oft_tmp;
 }
if ( h_lut[0].out_n )
{
     hoft_table[j] = (h_lut[0].in_n == 0)? 0: in_oft_tmp;        \
     hcoef_table[j] = coef_tmp;
 }
j++;
hcoef_real_heiht = j;
 RSZ_COEF_INDEX. HE_IDX = j –1;
```

Step 4: Calculate the last table of resize coefficient.

Bilinear:

```
for (cnt =0 ; cnt<hcoef_real_heiht ; cnt ++)
      HRSZ_COEF_LUT_bilinear[cnt] = (hcoef_table[cnt], hoft_table[cnt]);
```

Bicube:

```
 int sinxdivx_table_8[(2<<9)+1];
for ( i = 0 ; i < (2<<9); ++i)
 {
     sinxdivx_table_8[i] = (int)(0.5 + 512*sinxdivx(i*(1.0/512)));
 }
int u_8;
for (i = 0 ; i <hcoef_real_heiht; i++ )
{
       int au_8[4];
       u_8 = 512 - hcoef_table[i];
       cube_hcoef_table[i][0] = sinxdivx_table_8[(1<<9)+u_8];
       cube_hcoef_table[i][1] = sinxdivx_table_8[u_8];
```

```
        cube_hcoef_table[i][2] = sinxdivx_table_8[(1<<9)-u_8];
        cube_hcoef_table[i][3] = sinxdivx_table_8[(2<<9)-u_8];
    }
for (cnt =0 ; cnt<hcoef_real_heiht ; cnt ++)
        HRSZ_COEF_LUT_bicube[cnt] = (cube_hcoef_table[cnt], hoft_table[cnt]);
```

4 Following are two examples of setting LUT in bilinear scale mode.

Resize coefficients for 7:3:

W	W_COEF	IN_EN	OUT_EN	Pixel 1	Pixel 2	OUT
2/3	341	1	1	P [0]	P [1]	P [0] * 2/3 + P [1] * 1/3
0	0	1	0	P [1]	P [2]	No new pixel out
1/3	171	1	1	P [2]	P [3]	P [2] * 1/3 + P [3] * 2/3
0	0	1	0	P [3]	P [4]	No new pixel out
0	0	1	0	P [4]	P [5]	No new pixel out
1	512	1	1	P [5]	P [6]	P [5] * 1 + P [6] * 0
0	0	1	0	P [6]	P [7]	No new pixel out

Parameter set to IPU is following:

index	W	W_COEF	OFSET	Pixel1	Pixel 2	OUT
0	2/3	341	2	P [0]	P [1]	P [0] * 2/3 + P [1] * 1/3
1	1/3	171	3	P [2]	P [3]	P [2] * 1/3 + P [3] * 2/3
2	1	512	2	P [5]	P [6]	P [5] * 1 + P [6] * 0

Resize coefficients for 3:5:

W	W_COEF	IN_EN	OUT_EN	Pixel 1	Pixel 2	OUT
1	512	1	1	P [0]	P [1]	P [0] * 1 + P [1] * 0
2/5	205	0	1	P [0]	P [1]	P [0] * 2/5 + P [1] * 3/5
4/5	410	1	1	P [1]	P [2]	P [1] * 4/5 + P [2] * 1/5
1/5	102	0	1	P [1]	P [2]	P [1] * 1/5 + P [2] * 4/5
3/5	307	1	1	P [2]	P [3]	P [2] * 3/5 + P [3] * 2/5

The parameter set to IPU is as following:

index	W	W_COEF	OFSET	Pixel 1	Pixel 2	OUT
0	1	512	0	P [0]	P [1]	P [0] * 1 + P [1] * 0
1	2/5	205	1	P [0]	P [1]	P [0] * 2/5 + P [1] * 3/5
2	4/5	410	0	P [1]	P [2]	P [1] * 4/5 + P [2] * 1/5
3	1/5	102	1	P [1]	P [2]	P [1] * 1/5 + P [2] * 4/5
4	3/5	307	1	P [2]	P [3]	P [2] * 3/5 + P [3] * 2/5

23.4.32 Vertical Resize Coefficients Look Up Table Register group

VRSZ_COEF_LUT (bi-cube) **0x4C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					SIGN	W_COEF_31										SIGN	W_COEF_20										VRSZ_OFT					V_CONF
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:28	Reserve	Writing has no effect, read as zero.	W
27:17	W_COEF_31	Weighting coefficients for pix_3 or pix_1.	W
16:6	W_COEF_20	Weighting coefficients for pix_2 or pix_0.	W
5:1	VRSZ_OFT	Vertical Resize pixel offset.	W
0	V_CONF	Start to configure the vertical resize table, read as zero: 1: start.	W

VRSZ_COEF_LUT (bilinear) **0x4C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																W_COEF0											V_OFT					V_CONF
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
31:17	Reserve	Writing has no effect, read as zero.	R
16:6	W_COEF0	Weighting coefficients for pix_0.	W
5:1	V_OFT	Vertical Resize pixel offset.	W
0	V_CONF	Start to configure the vertical resize table, read as zero: 1: start.	W

NOTES:

1 refer to Horizontal HRSZ_COEF_LUT.

23.4.33 Calculation for Resized width and height

For software, to preset correct value for register OUT_GS, please refer to following formula.
Set IW stand for original input frame width, IH stand for original input frame height, OW stand for new frame width after resize, OH stand for new frame height after resize.

In Up-scale case (n < m):

If [(IW - 1) * (m/n)] * (n/m) ==(IW-1) then
    OW = [(IW - 1) * (m/n)] + 1;
Else OW = [(IW - 1) * (m/n)] + 2;
If [(IH - 1) * (m/n)] * (n/m) == (IH-1) then
    OH = [(IH - 1) * (m/n)] + 1;
Else OH = [(IH - 1) * (m/n)] + 2;

In Down-scale case (n>m):

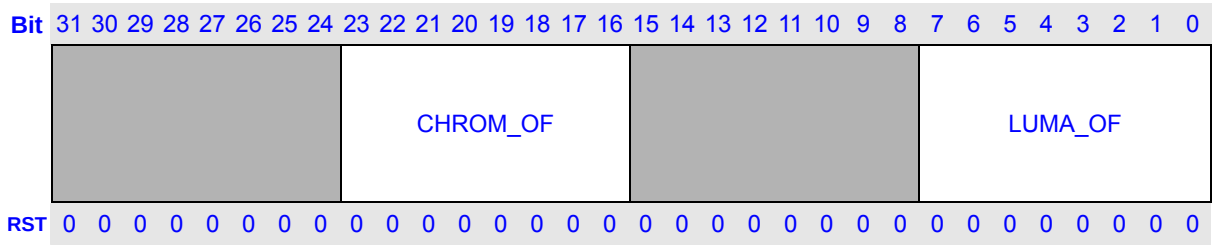
If [(IW - 1) * (m/n)] * (n/m) ==(IW-1) then
    OW = [(IW - 1) * (m/n)];
Else OW = [(IW - 1) * (m/n)] + 1;
If [(IH - 1) * (m/n)] * (n/m) ==(IH-1) then
    OH = [(IH - 1) * (m/n)];
Else OH = [(IH - 1) * (m/n)] + 1;

For example:
A 36x46 frame with the horizontal resize ratio of 4:5 (up-scale) and vertical resize ratio of 7:6 (down-scale), by the expressions above we get its new size after resize from the following process.

For Width: [(36 - 1) * (5/4)] * (4/5) = 34.4 ≠ (36-1)
    So OW = [(36 - 1) * (5/4)] + 2 = 45
For Height: [(46 - 1) * (6/7)] * (7/6) = 44.33 ≠ (46 - 1)
    So OH = [(46 - 1) * (6/7)] + 1 = 39

23.4.34 CSC Offset Parameter Register

CSC_OFSET_PARA **0x50**



Bits	Name	Description	R/W
31:24	Reserved	Writing has no effect, read as zero.	R
23:16	CHROM_OF	Chroma offset value.	RW

15:8	Reserved	Writing has no effect, read as zero.	R
7:0	LUMA_OF	Luma offset value.	RW

NOTE:

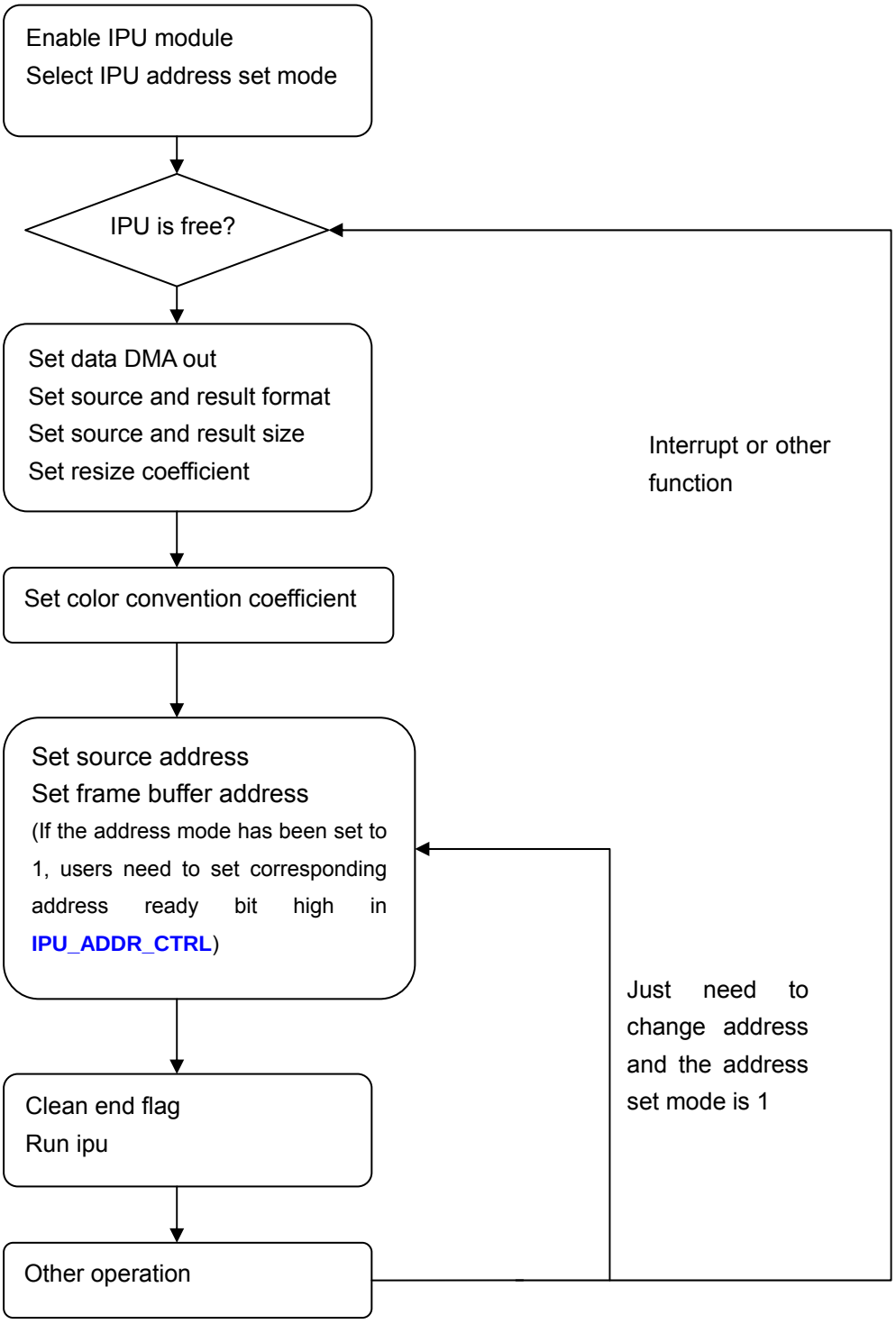
R = C0*(Y – LUMA_OF) + C1*(Cr-CHROM_OF)

G = C0*(Y – LUMA_OF) – C2*(Cb-CHROM_OF)– C3*(Cr-CHROM_OF)

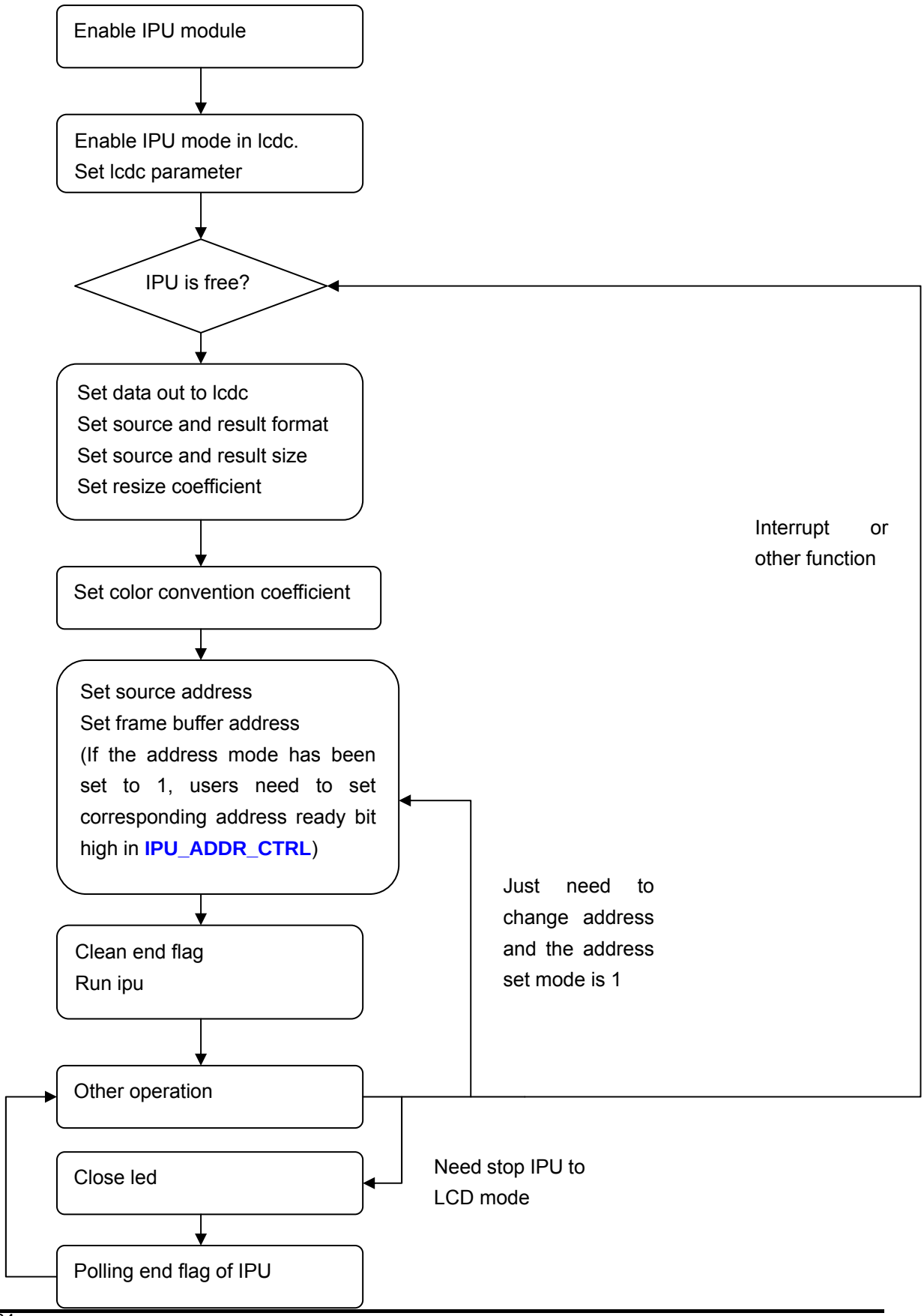
B = C0*(Y – LUMA_OF) + C4*(Cb-CHROM_OF)

23.5 IPU Operation Flow

23.5.1 Data out to frame buffer



23.5.2 Data out to lcdc



23.5.3 Operation example

Table 23-2 no mapping mode

Step	Action
Base	Chip_enable()
Base_0	ipu_addr_sel(1);
0	Do { } while {!polling_end_flag}
1	set_primary_ctrl(VRSZ_ENABLE, HRSZ_ENABLE, CSC_EN, irq_en); //
2	set_source_ctrl(source_pkg_sel, SPAGE_SEL) ;
3	set_out_ctrl(lcdc_sel, DPAGE_SEL, DISP_SEL, FIELD_SEL, FIELD_CONF_EN) ;
4	set_scale_ctrl(V_SCALE, H_SCALE);
5	set_ipu_fmt(RGB888_OUT_FMT, OUT_OFT_RGB, OUT_FMT, OUT_Y1UY0V , IN_OF_YUYV, IN_FM_YUV444);
6	set_inframe_gsize(FIN_W, FIN_H, FIN_Y_STRIDE, FIN_U_STRIDE, FIN_V_STRIDE);

7	set_y_addr((unsigned int)fin_y & 0x1FFFFFFF);
	set_u_addr((unsigned int)fin_y & 0x1FFFFFFF);
	set_v_addr((unsigned int)fin_y & 0x1FFFFFFF);

8	set_outframe_gsize(FOUT_W, FOUT_H , FOUT_STRIDE);
9	set_out_addr((unsigned int)fout & 0x00000FFF);
9A	set_addr_ready(0xFF); **NOTE:** this step is necessary when ipu address set mode is 1.

10	set_csc_c0(YUV_CSC_C0);
	set_csc_c1(YUV_CSC_C1);
	set_csc_c2(YUV_CSC_C2);
	set_csc_c3(YUV_CSC_C3);
	set_csc_c4(YUV_CSC_C4);

11	set_csc_ofset_para (128, 0) ;
12	set_rsz_lut_end(H_MAX_LUT-1, V_MAX_LUT-1);
13	start_hlut_coef_write(); **NOTE:** This step is necessary before write new LUT.

14	for (i=0;i<H_MAX_LUT;i++) {
	set_hrsz_lut_coef(h_lut[i].coef, h_lut[i].in_n, h_lut[i].out_n);
	}

15	start_vlut_coef_write(); **NOTE:** This step is necessary before write new LUT.

16	for (i=0;i<V_MAX_LUT;i++) {
	set_vrsz_lut_coef(v_lut[i].coef, v_lut[i].in_n, v_lut[i].out_n);
	}

17	Clean_end_flag(); run_ipu();

Table 23-3 mapping mode

Step	Action

Prepare	y_phy_table[0] = ((unsigned int)fin_y & 0x0FFFF000) | 0x20000000 ;
	u_phy_table[0] = ((unsigned int)fin_u & 0x0FFFF000) | 0x20000000 ;
	v_phy_table[0] = ((unsigned int)fin_v & 0x0FFFF000) | 0x20000000 ;
	out_phy_table[0] = ((unsigned int)fout & 0x0FFFF000) | 0x20000000 ;
	for (i =1; i<100; i++){
	y_phy_table[i] = y_phy_table[i-1] + 4096 ;
	u_phy_table[i] = u_phy_table[i-1] + 4096 ;
	v_phy_table[i] = v_phy_table[i-1] + 4096 ;
	out_phy_table[i] = out_phy_table[i-1] + 4096 ;
	}

Base	Chip_enable()
Base_0	ipu_addr_sel(1);
0	Do { } while {!polling_end_flag}
1	set_primary_ctrl(VRSZ_ENABLE, HRSZ_ENABLE, CSC_EN, irq_en); //
2	set_source_ctrl(source_pkg_sel, SPAGE_SEL) ;
3	set_out_ctrl(lcdc_sel, DPAGE_SEL, DISP_SEL, FIELD_SEL, FIELD_CONF_EN) ;
4	set_scale_ctrl(V_SCALE, H_SCALE);
5	set_ipu_fmt(RGB888_OUT_FMT, OUT_OFT_RGB, OUT_FMT, OUT_Y1UY0V , IN_OF_YUYV, IN_FM_YUV444);
6	set_inframe_gsize(FIN_W, FIN_H, FIN_Y_STRIDE, FIN_U_STRIDE, FIN_V_STRIDE);

7	set_y_addr((unsigned int)fin_y & 0xFFF);
	set_u_addr((unsigned int)fin_y & 0xFFF);
	set_v_addr((unsigned int)fin_y & 0xFFF);

8	set_outframe_gsize(FOUT_W, FOUT_H , FOUT_STRIDE);
9	set_out_addr((unsigned int)fout & 0x00000FFF);
10	set_y_phy_t_addr((unsigned int)y_phy_table & 0x1FFFFFFF) ; set_u_phy_t_addr((unsigned int)u_phy_table & 0x1FFFFFFF) ; set_v_phy_t_addr((unsigned int)v_phy_table & 0x1FFFFFFF) ; set_out_phy_t_addr((unsigned int)out_phy_table & 0x1FFFFFFF) ;
10A	set_addr_ready(0xFF); **NOTE:** this step is necessary when ipu address set mode is 1.

11	set_csc_c0(YUV_CSC_C0);
	set_csc_c1(YUV_CSC_C1);
	set_csc_c2(YUV_CSC_C2);
	set_csc_c3(YUV_CSC_C3);
	set_csc_c4(YUV_CSC_C4);

12	set_csc_ofset_para (128, 0) ;
13	set_rsz_lut_end(H_MAX_LUT-1, V_MAX_LUT-1);

```
14      start_hlut_coef_write();
        NOTE: This step is necessary before write new LUT.
15        for (i=0;i<H_MAX_LUT;i++) {
            set_hrsz_lut_coef(h_lut[i].coef, h_lut[i].in_n, h_lut[i].out_n);
          }
16        start_vlut_coef_write();
        NOTE: This step is necessary before write new LUT.
17        for (i=0;i<V_MAX_LUT;i++) {
            set_vrsz_lut_coef(v_lut[i].coef, v_lut[i].in_n, v_lut[i].out_n);
          }
18      Clean_end_flag();
        run_ipu();
```

23.6 Special Instruction

A1. Resizing size feature

Input size (W x H)		Output size (W x H)	
Min	4x4	Disable vertical scale	Min: 4x4
			Max: 4095x4095
Max	4095x4095	Enable vertical scale	Min: 4x4
			Max: 1280x4095

A2. Color convention feature

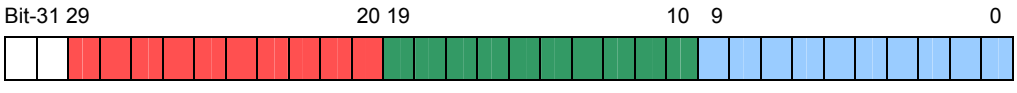
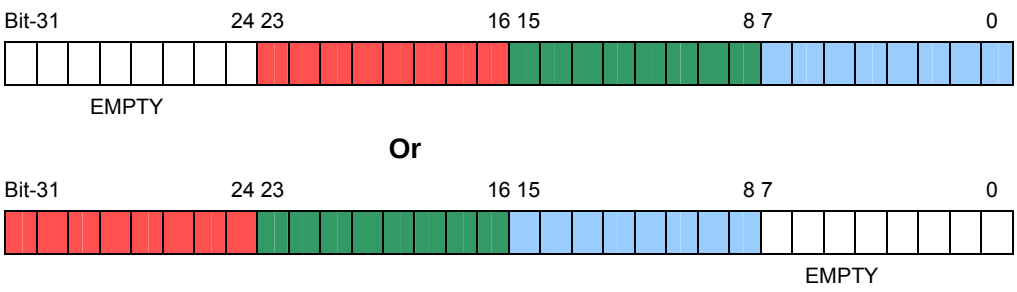
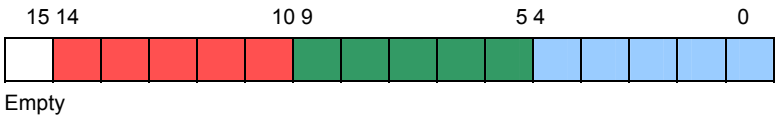
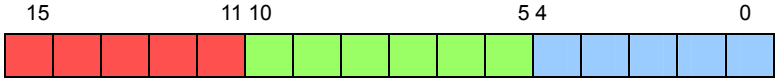
Source format	Output format	Parameter configure (necessary)
RGB	RGB	IPU_CONTROL.CSC_EN =0 IPU_CONTROL. SPKG_SEL = 0 or 1
		D_FMT. IN_FMT
		D_FMT.OUT_FMT = 2'b00, 2'b01, 2'b10
YUV	RGB	IPU_CONTROL.CSC_EN =1
		IPU_CONTROL. SPKG_SEL
		D_FMT. IN_FMT
		D_FMT. IN_OFT ***(IPU_CONTROL. SPKG_SEL == 1)***
		D_FMT.OUT_FMT = 2'b00, 2'b01, 2'b10
		D_FMT.RGB_OUT_OFT.
		CSC_C0 (1,2,3,4)_COEF, CSC_OFSET_PARA
YUV	YUV (package)	IPU_CONTROL.CSC_EN =0
		IPU_CONTROL. SPKG_SEL
		D_FMT. IN_FMT
		D_FMT. IN_OFT ***(IPU_CONTROL. SPKG_SEL == 1)***
		D_FMT.OUT_FMT = 2'b11

A3. YUV/YCbCr to RGB CSC Equations

Input data	Matrix	CSC_COEF
YUV	R = C0*(Y – X0) + C1*(V-128)	CSC_C0_COEF = 0x400
	G = C`0*(Y – X0) – C2*(U-128) – C3*(V-128)	CSC_C1_COEF= 0x59C
	B = C0*(Y – X0) + C4*(U-128)	CSC_C2_COEF = 0x161
	X0: 0	CSC_C3_COEF = 0x2DC

	C0: 1 C1: 1.4026 C2: 0.3444 C3: 0.7144 C4: 1.7730	CSC_C4_COEF = 0x718
YCbCr	R = C0*(Y – X0) + C1*(Cr-128) G = C0*(Y – X0) – C2*(Cb-128) – C3*(Cr-128) B = C0*(Y – X0) + C4*(Cb-128) X0: 16 C0: 1.164 C1: 1.596 C2: 0.391 C3: 0.813 C4: 2.018	CSC_C0_COEF = 0x4A8 CSC_C1_COEF = 0x662 CSC_C2_COEF = 0x191 CSC_C3_COEF = 0x341 CSC_C4_COEF = 0x811

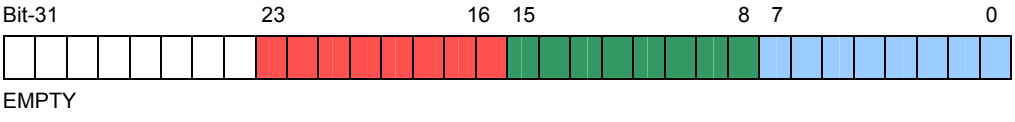
A4. Output data package format (RGB order)

Format	Package
RGBAAA	
RGB888	
RGB555	
RGB565	

NOTES:

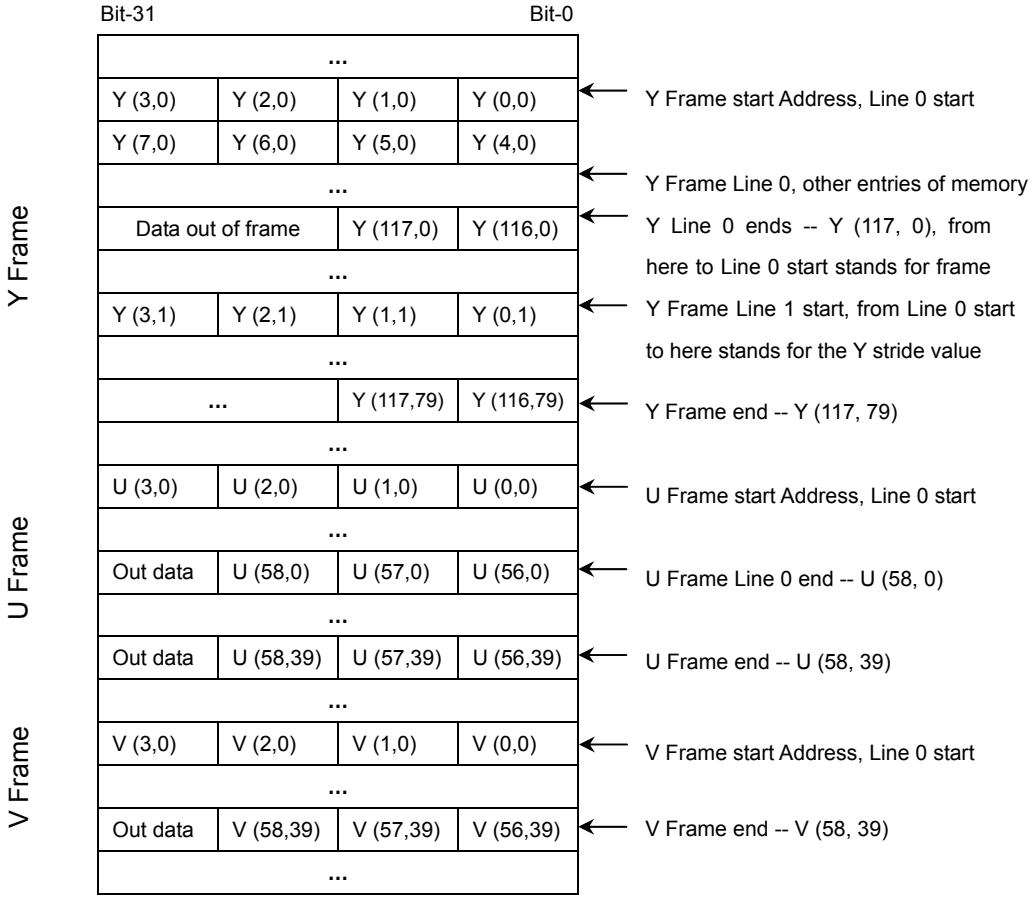
1 All R/G/B data are little-endian type; all the empty bits in the above figure are filled with 0.

A5. Input data package format (RGB order)

Format	Package
RGBPACK	

A6. Source Data storing format in external memory (separated YUV Frame)

Example: YUV420 118x80 frame



NOTES:

1 Every line's start address should be word aligned.

2 All pixel data should be stored as little-endian format.

3 Destination data (RGB) storing format in external memory is similar with above figure, but RGB555 and RGB565 frame's every line start address can be half-word aligned. (RGB888 frame still need word aligned)

24 Camera Interface Module

24.1 Overview

The camera interface module (CIM) supports commonly available CMOS or CCD type image sensors. The CIM sources the digital image stream through a common 8-bit parallel digital protocol. The CIM can directly connect to external CMOS image sensors and ITU656 standard video decoders.

24.1.1 Features

- Input image size up to 4096x4096 pixels
- Max. VGA for image preview
- Max. VGA for video record
- Integrated DMA
- Supported data format: YCbCr 4:4:4, YCbCr 4:2:2 and other formats
- Supports ITU656 (YCbCr 4:2:2) input
- Configurable CIM_VSYNC and CIM_HSYNC signals: active high/low
- Configurable CIM_PCLK: active edge rising/falling
- 64x33 image data receive FIFO (RXFIFO)
- PCLK max. 80MHz
- Output format: csc mode is YCbCr 4:2:2, bypass mode is the input data format
- Configurable output order

24.1.2 Pin Description

Table 24-1 Camera Interface Pins Description

Name	I/O	Description
CIM_MCLK	O	CIM work clock
CIM_PCLK	I	Pixel clock from Image Sensor
CIM_VSYNC	I	Vertical synchronous from Image Sensor
CIM_HSYNC	I	Horizontal synchronous from Image Sensor
CIM_DATA[7:0]	I	Data bus from Image Sensor

24.2 CIM Special Register

The special registers are for CIM to configure and control the interface and DMA operation. The table below lists these registers.

Table 24-2 CIM Registers

Name	RW	Reset Value	Address	Access Size
CIMCFG	RW	0x00000080	0x13060000	32
CIMCR	RW	0x00000000	0x13060004	32
CIMST	RW	0x00000000	0x13060008	32
CIMIID	R	0x00000000	0x1306000C	32
CIMRXFIFO	R	0x????????	0x13060010	32
CIMDA	RW	0x00000000	0x13060020	32
CIMFA	R	0x00000000	0x13060024	32
CIMFID	R	0x00000000	0x13060028	32
CIMCMD	R	0x00000000	0x1306002C	32
CIMSIZE	RW	0x00000000	0x13060030	32
CIMOFFSET	RW	0x00000000	0x13060034	32

24.2.1 CIM Configuration Register (CIMCFG)

CIMCFG **0x13060000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												ORDER		DF		INV_DAT	VSP	HSP	PCP	BURST_TYPE		DUMMY	E_VSYNC	LM	PACK			FP	BYPASS	DSM	
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:20	Reserved		RW
19:18	ORDER	Input data stream order. 00: YCbCr 4:4:4 — YCbCr; ITU656/YCbCr 4:2:2 — Y_0CbY_1Cr 01: YCbCr 4:4:4 — YCrCb; ITU656/YCbCr 4:2:2 — Y_0CrY_1Cb 10: YCbCr 4:4:4 — CbCrY; ITU656/YCbCr 4:2:2 — CbY_0CrY_1 11: YCbCr 4:4:4 — CrCbY; ITU656/YCbCr 4:2:2 — CrY_0CbY_1	RW
17:16	DF	Input data format. 00: reserved 01: YCbCr 4:4:4 10: YCbCr 4:2:2	RW

		11: ITU656 YCbCr 4:2:2	
15	INV_DAT	Inverse every bit of input data. 0: not inverse; 1: inverse.	RW
14	VSP	VSYNC polarity selection. When VSYNC signal is input from pin CIM_VSYNC, this bit specifies the VSYNC signal active level and leading edge. When VSYNC is retrieved from SAV&EAV, this bit is ignored. 0: VSYNC signal active high, VSYNC signal leading edge is rising edge 1: VSYNC signal active low, VSYNC signal leading edge is falling edge	RW
13	HSP	Specifies the HSYNC signal active level and leading edge. 0: HSYNC signal active high, HSYNC signal leading edge is rising edge 1: HSYNC signal active low, HSYNC signal leading edge is falling edge	RW
12	PCP	Specifies the PCLK working edge. 0: Data is sampled by PCLK rising edge 1: Data is sampled by PCLK falling edge	RW
11:10	BURST_ TYPE	DMA burst type. 00: INCR4 01: INCR8 10: INCR16 11: Reserved It is suggested using INCR8; if AHB works at high speed, INCR16 is suggested.	RW
9	DUMMY	DUMMY zero function. When DUMMY is 1, CIM hardware adds one byte zero to every 3 input data bytes to form 32-bit data. 0: DUMMY zero function disabled 1: DUMMY zero function enabled	9
8	E_VSYN C	External / internal VSYNC selection. When DSM is ITU656Progressive Mode, VSYNC can be external (provided by sensor) or internal (retrieved from SAV&EAV). This bit only valid for ITU656Progressive Mode; In other DSM modes, this bit is ignored. 0: Internal VSYNC mode, pin CIM_VSYNC is ignored 1: External VSYNC mode, VSYNC is provided by image sensor via pin CIM_VSYNC	RW
7	LM	Line Mode for ITU656. 0: EAV is before SAV in each line 1: SAV is before EAV in each line	RW
6:4	PACK	Data packing mode, pack 8-bit input data into 32-bit data for FIFO.	6:4

		<table><tr><th>PACK</th><th>Bypass Mode</th><th>CSC Mode</th></tr><tr><td>3'b000</td><td>0x 11 22 33 44</td><td>0x Y_0 Cb Y_1 Cr</td></tr><tr><td>3'b001</td><td>0x 22 33 44 11</td><td>0x Cb Y_1 Cr Y_0</td></tr><tr><td>3'b010</td><td>0x 33 44 11 22</td><td>0x Y_1 Cr Y_0Cb</td></tr><tr><td>3'b011</td><td>0x 44 11 22 33</td><td>0x Cr Y_0 Cb Y_1</td></tr><tr><td>3'b100</td><td>0x 44 33 22 11</td><td>0x Cr Y_1 Cb Y_0</td></tr><tr><td>3'b101</td><td>0x 33 22 11 44</td><td>0x Y_1 Cb Y_0 Cr</td></tr><tr><td>3'b110</td><td>0x 22 11 44 33</td><td>0x Cb Y_0 Cr Y_1</td></tr><tr><td>3'b111</td><td>0x 11 44 33 22</td><td>0x Y_0 Cr Y_1 Cb</td></tr></table> In this table, 0x11, 0x22, 0x33 and 0x44 mean the received data from the sensor, 0x11 is received first and 0x44 is received last, and Y0 is received before Y1.	
3	FP	Field flag polarity selection. When ITU656 progressive stream is input, this bit specifies the field flag active level. When other modes are used, this bit is ignored. 0: Field flag active low 1: Field flag active high	
2	BYPASS	0: enable CIM CSC 1: disable CIM CSC The formula for csc from RGB888 to YCbCr444 is: Y = R *0.299000 + G * 0.587000 + B * 0.114000 Cb = -R * 0.168736 - G * 0.331264 + B * 0.500000 + 128 Cr = R * 0.500000 - G * 0.418688 - B * 0.081312 + 128	
1:0	DSM	Data sample mode. Please refer to the table below. <table><tr><th>DSM</th><th>Description</th></tr><tr><td>2'b00</td><td>ITU656Progressive Mode</td></tr><tr><td>2'b01</td><td>ITU656Interlace Mode</td></tr><tr><td>2'b10</td><td>Gated Clock Mode</td></tr><tr><td>2'b11</td><td>Reserved</td></tr></table>	RW

24.2.2 CIM Control Register (CIMCR)

CIMCR **0x13060004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EEOF_LINE												FRC				DMA_EEOFM	WIN_En	VDDM	DMA_SOFM	DMA_EOFM	DMA_STOPM	RF_TRIGM	RF_OFM	DMA_SYNC	RF_TRIG				DMA_EN	RF_RST	ENA
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

<table>
<tr><th>Bits</th><th>Name</th><th>Description</th><th>RW</th></tr>
<tr><td>31:20</td><td>EEOF_LINE</td><td>When EEOF_LINE lines data has been transferred of a frame, the EEOF flag will be set, and the EEOF interrupt will occur.</td><td>R</td></tr>
<tr><td>19:16</td><td>FRC</td><td>CIM frame rate control. Specifies the sampling frame data rate. If FRC = N, CIM sampling one frame of every N+1 frames from the sensor. In this way, CIM reduces the frame rate of sensor. Another way to reduce frame rate is to decrease the MCLK frequency output to the image sensor.
<table>
<tr><th>FRC</th><th>Description</th></tr>
<tr><td>4’b0000</td><td>Sample every frame from the sensor</td></tr>
<tr><td>4’b0001</td><td>Sample 1 frame of every 2 frames from the sensor</td></tr>
<tr><td>……</td><td>……</td></tr>
<tr><td>4’b1110</td><td>Sample 1 frame of every 15 frames from the sensor</td></tr>
<tr><td>4’b1111</td><td>Sample 1 frame of every 16 frames from the sensor</td></tr>
</table></td><td>RW</td></tr>
<tr><td>15</td><td>DMA_EEOFM</td><td>The control bit to enable EEOF interrupt.</td><td>RW</td></tr>
<tr><td>14</td><td>WIN_En</td><td>To enable window-image. Used to indicate whether the registers CIMSIZE and CIMOFFSET work or not.
0: the value in CIMSIZE and CIMOFFSET will be ignored
1: the value in CIMSIZE and CIMOFFSET will be used</td><td>R</td></tr>
<tr><td>13</td><td>VDDM</td><td>The control bit to enable VDD interrupt.
0: disable; 1: enable.</td><td>RW</td></tr>
<tr><td>12</td><td>DMA_SOFM</td><td>The control bit to enable DMA_SOF interrupt.
0: disable; 1: enable.</td><td>RW</td></tr>
<tr><td>11</td><td>DMA_EOFM</td><td>The control bit to enable DMA_EOF interrupt.
0: disable; 1: enable.</td><td></td></tr>
<tr><td>10</td><td>DMA_STOPM</td><td>The control bit to enable DMA_STOP interrupt.
0: disable; 1: enable.</td><td>RW</td></tr>
<tr><td>9</td><td>RF_TRIGM</td><td>The control bit to enable RXF_TRIG interrupt.
0: disable; 1: enable.</td><td>RW</td></tr>
<tr><td>8</td><td>RF_OFM</td><td>The control bit to enable RXF_OF interrupt.
0: disable; 1: enable.</td><td>RW</td></tr>
<tr><td>7</td><td>DMA_SYNC</td><td>The control bit to enable DAM synchronization.
0: The valid data input to CIM will be transferred by DMA to external memory
1: When a new descriptor-DMA transfer starts with writing CIMDA, a frame synchronization will be done, and the data in RXFIFO will be ignored.</td><td>RW</td></tr>
<tr><td>6:3</td><td>RF_TRIG</td><td>Specifies the trigger value of RXFIFO.</td><td>RW</td></tr>
</table>

			CIMCFG.BURST_TYPE: INCR4 — RF_TRIG = n: Trigger value is (n + 1) * 4 INCR8 — Trigger value is (n + 1) * 8 INCR16 — Trigger value is (n + 1) * 16 **NOTE:** Trigger value should be less then 64, and n is suggested to 0.	
2	DMA_EN	Enable / disable the DMA function. 0: disable DMA; 1: enable DMA.		RW
1	RF_RST	RXFIFO software reset. Setting 1 to RXF_RST can reset RXFIFO immediately. Setting 0 to RXF_RST can stop resetting RXFIFO. After reset, RXFIFO is empty.		RW
0	ENA	Enable / disable the CIM module. Setting 1 to ENA can enable CIM. When CIM is working, clear ENA to 0 can stop CIM immediately. 0: CIM is not enabled, or disable CIM immediately 1: CIM is enabled, or enabling CIM		RW

24.2.3 CIM Status Register (CIMST)

CIMST 0x13060008

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																								DMA_EEOF	DMA_SOF	DMA_EOF	DMA_STOP	RF_OF	RF_TRIG	RF_EMPTY	VDD
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0

Bits	Name	Description	RW
31:8	Reserved		R
7	DMA_EEOF	When set to 1, indicate the DMA has transferred CIMCTRL.EEOF_LINE lines data of a frame. Write 0 to this bit to clear.	
6	DMA_SOF	When set to 1, Indicate the DMA start a transfer from RXFIFO to a frame buffer. Write 0 to this bit to clear.	RW
5	DMA_EOF	When set to 1, indicate the DMA complete a transfer from RXFIFO to a frame buffer. Write 0 to this bit to clear.	RW
4	DMA_STOP	When set to 1, indicate the DMA complete transferring data and stop the operation. Can generate interrupt if CIMCR.DMA_STOPM bit is	RW

		set. Write 0 to this bit to clear.	
3	RF_OF	RXFIFO over flow. When RXFIFO over flow happens, RXF_OF is set 1. Can generate interrupt if CIMCR.RF_OFM bit is set. Write 0 to this bit to clear.	RW
2	RF_TRIG	RXFIFO trigger. Indicates whether RXFIFO meet the trigger value or not. When the valid data number in RXFIFO reaches the trig value, RXF_TRIG is set 1; when the valid data number in RXFIFO do not reach the trig value, RXF_TRIG is set 0. Can generate interrupt if CIMCR.RF_TRIGM bit is set. 0: RXFIFO does not meets the trigger value 1: RXFIFO meets the trigger value	R
1	RF_EMPTY	RXFIFO empty. Indicates whether RXFIFO is empty or not. After reset, RXFIFO is empty, and RXF_EMPTY is 1. 0: RXFIFO is not empty 1: RXFIFO is empty	R
0	VDD	CIM disable done. Indicate this module is disabled after clear the CIMCR.ENA bit to disable the CIM module. Can generate interrupt if CIMCR.DMA_VDDM bit is set. 0: CIM has not been disabled 1: CIM has been disabled Write 0 to this bit to clear.	RW

24.2.4 CIM Interrupt ID Register (CIMIID)

CIMIID **0x1306000C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FID																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	FID	Interrupt frame ID. Contains a copy of the Frame ID register (CIMFID) from the descriptor currently being processed when a DMA_SOF or DMA_EOF interrupt is generated. CIMIID is written to only when CIMCMD.SOFINT or CIMCMD.EOFINT is high. As such, the register is considered to be sticky and will be overwritten only when the associated interrupt is cleared by writing the CIM state register.	R

24.2.5 CIM RXFIFO Register (CIMRXFIFO)

CIMRXFIFO																														0x13060010
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Data																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
31:0	Data	This register provides a port for software to read image data directly. When the software start CIM with DMA_EN=1, this register should not be read. Otherwise, the DMA data may be damaged.	R

24.2.6 CIM Descriptor Address (CIMDA)

CIMIDA																														0x13060020
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NDA																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	NDA	Next descriptor physical address in external memory. DMAC gets the next descriptor according to it after finishing the current one. The target address Bits [3:0] must be zero to be aligned to 16-byte boundary.	RW

24.2.7 CIM Frame buffer Address Register (CIMFA)

CIMIFA																														0x13060024
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FPA																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	FPA	Frame buffer physical address in external memory. When starts CIM, DMA transfers data from RXFIFO to frame buffer. This address is increased by hardware automatically. Bits [4:0] must be zero to be aligned to 32-byte boundary.	R

NOTES:

1 CIMFA comes from DMA Descriptor, so here it is read-only.

24.2.8 CIM Frame ID Register (CIMFID)

CIMFID **0x13060028**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FID																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	FID	Frame ID. The particular use of this field is up to the software. This ID will be copied to the CIMIID register when an interrupt occurs.	R

NOTES:

1 CIMFID comes from DMA Descriptor, so here it is read-only.

24.2.9 CIM DMA Command Register (CIMCMD)

CIMFID **0x1306002C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SOFINT	EOFINT	EEOFINTEn	STOP	Reserved												LEN															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	SOFINTEn	Interrupt enable for DMA starting a frame-buffer transfer. 1: DMA will set CIMSTATE.DMA_SOF when start of a frame-buffer transfer When one frame uses several buffers, it is suggested to set	R

		SOFINTEn of first buffer only.	
30	EOFINTEn	Interrupt enable for DMA ending a frame-buffer transfer. 1: DMA will set CIMSTATE.DMA_EOF when CIMCMD.LEN is decreased to 0, which means end of a frame-buffer transfer When one frame uses several buffers, it is suggested to set EOFINTEn of last buffer only.	R
29	EEOFINTEn	Interrupt enable for DMA issuing an earlier eof interrupt.	R
28	STOP	DMA stop. When DMA complete transferring data, STOP bit decides whether DMA should loading next descriptor or not. 0: DMA start loading next descriptor 1: DMA stopped, and CIMSTATE.DMA_STOP bit is set 1 by hardware	R
27	OFRCVEN	Auto recovery enable when there is RXFIFO overflow. 0: No auto recovery when overflow occurs, thus the software should do something 1: Auto recovery enable, the hardware will correct the overflow	
26:24	Reserved		R
23:0	LEN	Length of transfer in words. Indicate the number of words to be transferred by DMA to a frame buffer. LEN = 0 is not valid. DMA transfers data according to LEN. Each time one or more word(s) been transferred, LEN is decreased automatically.	R

24.2.10 CIM Window-image Size (CIMSIZE)

CIMFID **0x13060030**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																LPF												Reserved					
																													PPL			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:29	Reserved		R
28:16	LPF	Lines per frame for CIM output.	RW
15:13	Reserved		R
12:0	PPL	Pixels per line for CIM output. PPL must be multiples of 2. In fact, the number of CIM output data in word is equal to PPL/2.	RW

24.2.11 CIM Image Offset (CIMOFFSET)

CIMFID 0x13060034

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				V_OFFSET												Reserved				H_OFFSET											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved		R
27:16	V_OFFSET	Vertical offset.	RW
15:12	Reserved		R
11:0	H_OFFSET	Horizontal offset. It should be an even number.	RW

24.3 CIM Data Sampling Modes

CIM module supports several types of data sampling mode. The modes and the corresponding signals used are shown in the following diagram:

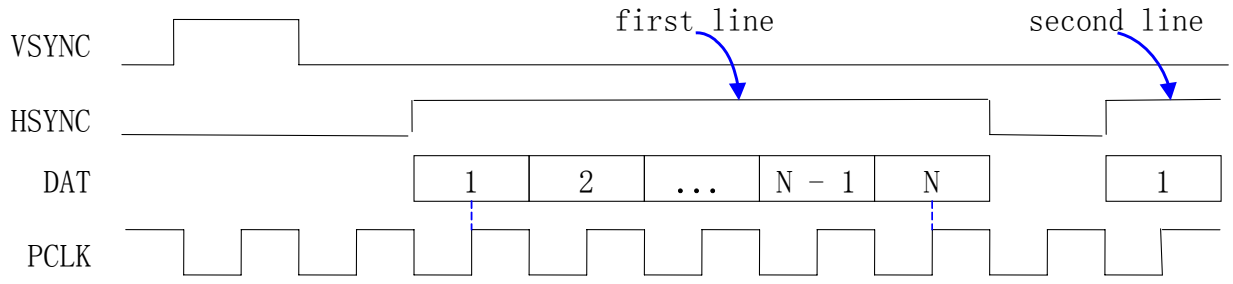
The modes and the corresponding signals used:

Mode \ Signals	CIM_VSYNC	CIM_HSYNC	CIM_PCLK	CIM_DATA
Gated Clock Mode	Y	Y	Y	Y
ITU656 Interlace Mode	N	N	Y	Y
ITU656 Progressive Mode	N	N	Y	Y

24.3.1 Gated Clock Mode

CIM_VSYNC, CIM_HSYNC, and CIM_PCLK signals are used in this mode.

A frame starts with VSYNC leading edge, and then HSYNC goes active and holds the entire line. Data is sampled at the valid edge of PCLK when HSYNC is active; that means, HSYNC functions like “data enable” signal. Please refer to the figure below.



Gated Clock Mode Input Timing Diagram

The VSYNC leading edge, HSYNC active HIGH or LOW, PCLK valid edges are programmable.

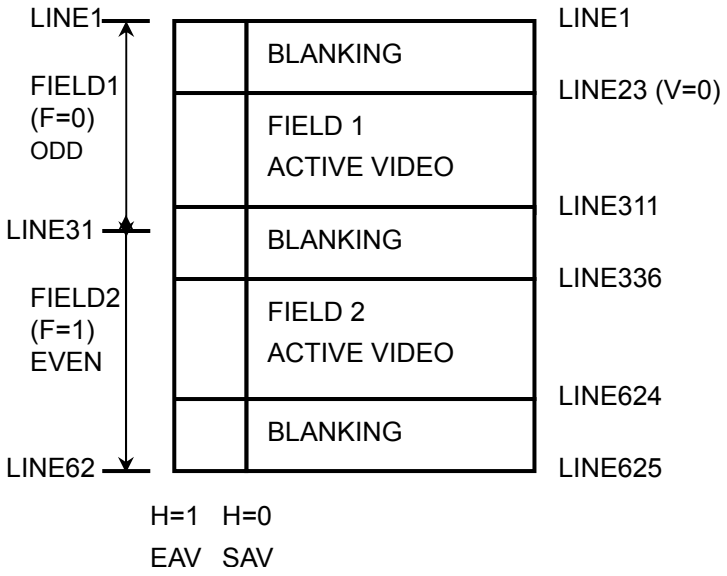
24.3.2 ITU656 Interlace Mode

In this mode, CIM_PCLK and CIM_DAT signals are used, CIM_VSYNC, CIM_HSYNC signals are ignored.

CIM utilizes the SAV & EAV code within ITU656data stream to get active video data.

The following diagrams and tables are quoted from ITU656standard. Only the PAL format is shown. CIM supports both NTSC and PAL formats. For more information about ITU656, please refer to ITU656 standard.

24.3.2.1 PAL Vertical Timing



LINE NUMBER	F	V	H (EAV)	H (SAV)	P0, P1, P2, P3
1-22	0 Field 1	1: blanking	1: in EAV, to indicate the end of active video	0: in SAV, to indicate the start of active video	Protection bits
23-310		0: video data			
311-312		1: blanking			
313-335	1 Field 2	1: blanking			
336-623		0: video data			
624-625		1: blanking			

Figure 24-1 Typical BT.656 Vertical Blanking Intervals for 625/50 Video Systems

24.3.2.2 PAL Horizontal Timing

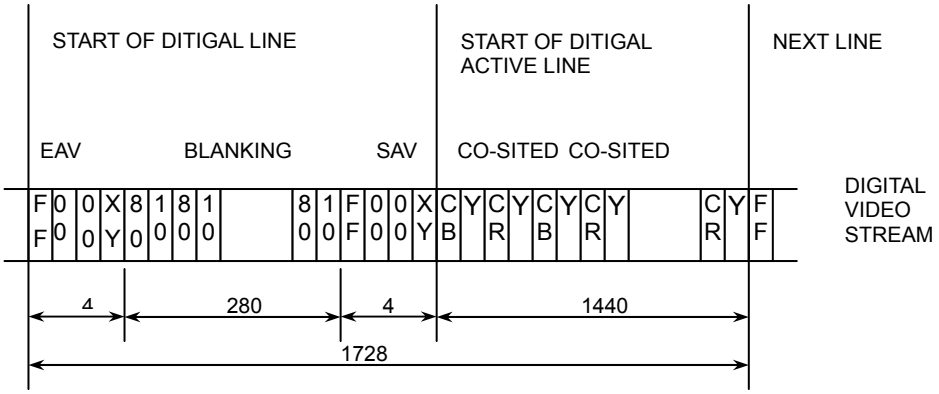


Figure 24-2 BT.656 8-BIT Parallel Interface Data Format for 625/50 Video Systems

24.3.2.3 Coding for SAV and EAV

Data Pin Number	1st Byte 0xFF	2nd Byte 0x00	3rd Byte 0x00	4th Byte 0xXY
7 (MSB)	1	0	0	1
6	1	0	0	F
5	1	0	0	V
4	1	0	0	H
3	1	0	0	P3
2	1	0	0	P2
1	1	0	0	P1
0 (LSB)	1	0	0	P0

24.3.2.4 Coding for Protection Bits

F	V	H	P3	P2	P1	P0
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	1	1
1	0	1	1	0	1	0
1	1	0	1	1	0	0
1	1	1	0	0	0	1

24.3.3 ITU656 Progressive Mode

CIM_PCLK and CIM_DAT signals are used in this mode. CIM_HSYNC signal is ignored.

CIM_VSYNC is optional in this mode. When the start of frame information is retrieved from SAV and EAV, it is known as internal VSYNC mode. When CIM_VSYNC is provided by sensor directly, it is known as external VSYNC mode. CIM supports both internal and external VSYNC modes.

ITU656Progressive Mode is a kind of Non-Interlace Mode. The image data are encoded within only one field. The F-bit of SAV and EAV are ignored. Most sensors support ITU656Progressive Mode.

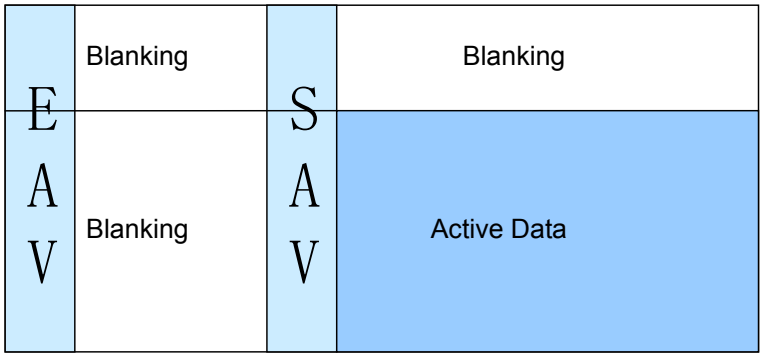


Figure 24-3 ITU656 Progressive Mode

24.4 DMA Descriptors

A DMA descriptor is a 4-word block corresponding to the four DMA registers – CIMDA, CIMFA, CIMFID, and CIMCMD, aligned on 4-word (16-byte) boundary, in external memory:

word [0] contains the physical address for next CIMDA
word [1] contains the physical address for CIMFA
word [2] contains the value for CIMFID
word [3] contains the value for CIMCMD

Software must write the physical address of the first descriptor to CIMDA before enabling the CIM. Once the CIM is enabled, the first descriptor is read, and all 4 registers are written by the DMAC. The next DMA descriptor pointed to by CIMDA is loaded into the registers after all data for the current descriptor has been transferred.

NOTES:

1 If only one frame buffer is used in external memory, the CIMDA field (word [0] of the DMA descriptor) must point back to itself. That is to say, the value of CIMDA is the physical address of itself.

24.5 Interrupt Generation

CIM has next interrupt sources:

Step 1. RXFIFO FULL Interrupt. (RF_TRIG)

When the valid data number of RXFIFO reaches trigger value, CIMST.RF_TRIG bit is set. At the same time, if RF_TRIGM is 1, RF_TRIG interrupt is generated.

Step 2. RXFIFO Over Flow Interrupt. (RF_OF)

When the valid data number of RXFIFO reaches 32 and one more data are written to RXFIFO, CIMST.RF_OF bit is set. At the same time, if RF_OFM is 1, RF_OF interrupt is generated.

Step 3. DMA Start Of Frame Data Transferring Interrupt. (DMA_SOF)

When the CIMCMD.SOFINT bit is 1 and DMA start transferring the first data from RXFIFO to frame buffer, CIMST.DMA_SOF bit is set. At the same time, if DMA_SOFM is 1, DMA_SOF interrupt is generated.

Step 4. DMA End Of Frame Data Transferring Interrupt. (DMA_EOF)

When the CIMCMD.EOFINT bit is 1 and DMA complete transferring the last data from RXFIFO to frame buffer, CIMST.DMA_EOF bit is set. At the same time, if DMA_EOFM is 1, DMA_EOF interrupt is generated.

Step 5. DMA Stop Transferring Interrupt. (DMA_STOP)

When the CIMCMD.STOP bit is 1 and DMA complete transferring the last data from RXFIFO to frame buffer, CIMST.DMA_STOP bit is set. At the same time, if DMA_STOPM is 1, DMA_STOP interrupt is generated.

Step 6. CIM Disable Done Interrupt. (VDD)

When disable the module by clearing the CIMCR.ENA, the module should be disabled after transferring current valid data. Then set the CIMST.VDD bit, at the same time, if VDDM is set, VDD interrupt is generated.

24.6 Software Operation

24.6.1 Enable CIM with DMA

Step 1. Configure register CIMCFG.
Step 2. Prepare frame buffer and descriptors.
Step 3. Configure register CIMDA.
Step 4. Clear state register: write 0 to register CIMSTATE.
Step 5. Reset RXFIFO: configure register CIMCTRL with DMA_EN=1, RXF_RST=1, ENA=0.
Step 6. Stop resetting RXFIFO: configure register CIMCTRL with DMA_EN=1, RXF_RST=0, ENA=0.
Step 7. Enable CIM: configure register CIMCTRL with DMA_EN=1, RXF_RST=0, ENA=1.

24.6.2 Enable CIM without DMA

1 Configure register CIMCFG.
2 Clear state register: write 0 to register CIMSTATE.
3 Reset RXFIFO: configure register CIMCTRL with DMA_EN=0, RXF_RST=1, ENA=0.
4 Stop resetting RXFIFO: configure register CIMCTRL with DMA_EN=0, RXF_RST=0, ENA=0.
5 Enable CIM: configure register CIMCTRL with DMA_EN=0, RXF_RST=0, ENA=1.

24.6.3 Disable CIM

Method 1:

Step 1. Configure register CIMCTRL with RXF_RST=0, ENA=0. // quick disable
Step 2. Clear state register: write 0 to register CIMSTATE.

Method 2:

When DMA is enabled, the following sequence is recommended:

Step 1. Configure descriptor with STOP = 1.
Step 2. Wait DMA_STOP interrupt, then write 0 to CIMCTRL.ENA.
Step 3. Clear state register: write 0 to register CIMSTATE.

25 Internal CODEC Interface

25.1 Overview

This chapter describes the embedded audio CODEC in the processor and related software interface.

This embedded CODEC is an I2S audio CODEC. AIC module is an interface to this CODEC in audio data replaying and recording. Several memory mapped registers are used to access this embedded CODEC, and write/read these registers could access the CODEC's internal control and configure registers that is using 12 MHz clock.

25.1.1 Features

The following are internal CODEC features:

- 24 bits ADC and DAC
- Headphone load up to 16 Ohm
- Sample frequency supported: 8k, 9.6k, 12k, 11.025k, 12k, 16k, 22.05k, 24k, 32k, 44.1k, 48k, and 96k
- Two MIC input, 85db SNR
- Stereo line input
- Separate power-down modes for ADC and DAC path with several shutdown modes
- Reduction of audible glitches systems: Pop Reduction system, Soft Mute mode

TBD = parameter or document section to be defined later on
TBC = parameter or document section subject to change
TO BE COMPLETED = section to be filled or subject to change

25.1.2 Signal Descriptions

CODEC has max 13 analog signal IO pins and 4 power pins on chip. They are listed and described in the flowing table.

Table 25-1 CODEC signal IO pin description

Pin Names	IO	Pin Description	Power
MICP1	AI	Microphone mono differential analog input 1 (MIC1), positive pin.	AVDCDC
MICN1	AI	Microphone mono differential analog input 1 (MIC1), negative pin.	AVDCDC
MICP2	AI	Microphone mono differential analog input 2 (MIC2), positive pin.	AVDCDC
MICN2	AI	Microphone mono differential analog input 2 (MIC2), negative pin.	AVDCDC
MICBIAS	AO	Microphone bias.	AVDCDC
AIL	AI	Left line input.	AVDCDC
AIR	AI	Right line input.	AVDCDC
AOLP	AO	Differential line output, positive pin.	AVDCDC
AOLN	AO	Differential line output, negative pin.	AVDCDC
AOBTLP	AO	Differential BTL output, positive pin.	AVDBTL
AOBTLN	AO	Differential BTL output, negative pin.	AVDBTL
AOHPL	AO	Left headphone out.	AVDHP
AOHPR	AO	Right headphone out.	AVDHP
VCAP	AO	Voltage Reference Output. An electrolytic capacitor more than 10μF in parallel with a 0.1μF ceramic capacitor attached from this pin to AVSCDC eliminates the effects of high frequency noise.	AVDCDC
AVDHP	P	Headphone amplifier power, 3.3V.	-
AVSHP	P	Headphone amplifier ground.	-
AVDBTL	P	BTL amplifier power, 3.3V.	-
AVSBTL	P	BTL amplifier ground.	-
AVDCDC	P	CODEC analog power, 3.3V, inter signal VREFP.	-
AVSCDC	P	CODEC analog ground, inter signal VREFN.	-
HPSENSE	AI	Headphone jack sense.	AVDHP

NOTES:

1 AVDHP = 3.3v (typ). AVDCDC= 3.3v (typ) AVDBTL= 3.3v (typ).

2 Inter signal VREFP is connected to AVDCDC, inter signal VREFN is connected to AVSCDC.

3 Please refer to data sheet of the chip for details.

25.1.3 Block Diagram

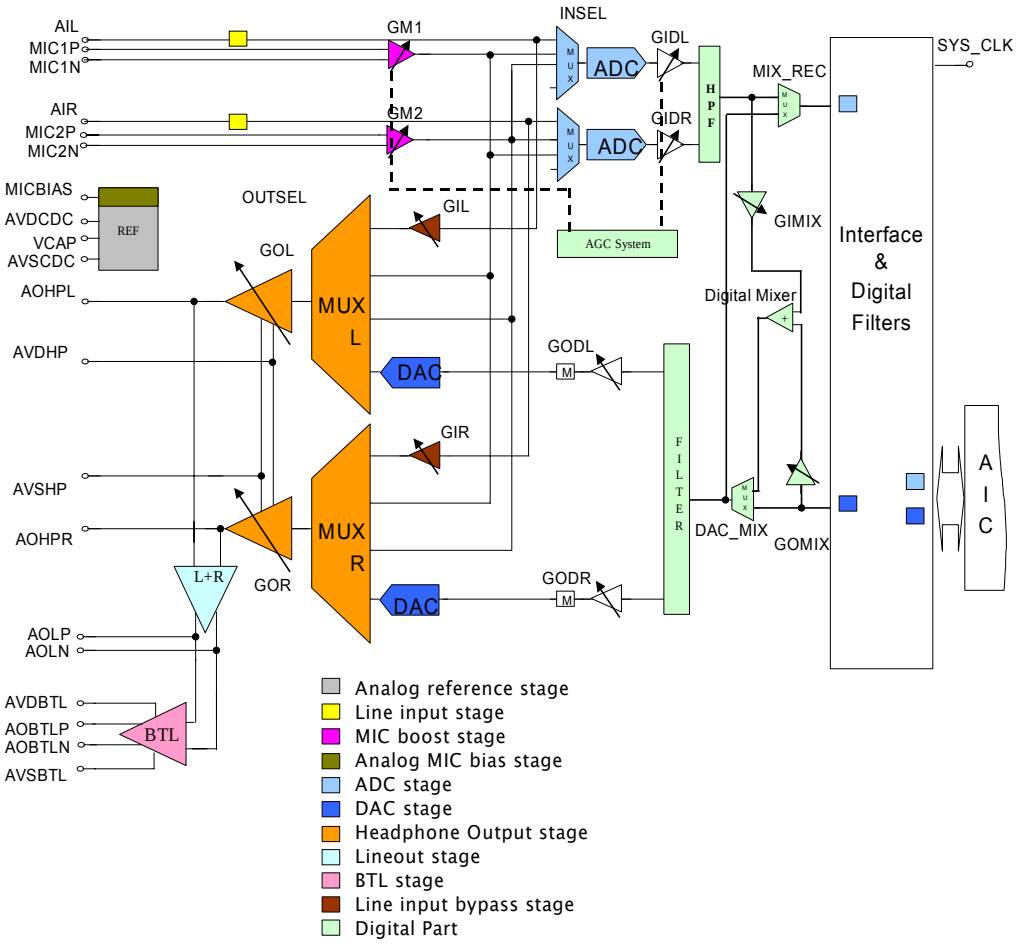


Figure 25-1 CODEC block diagram

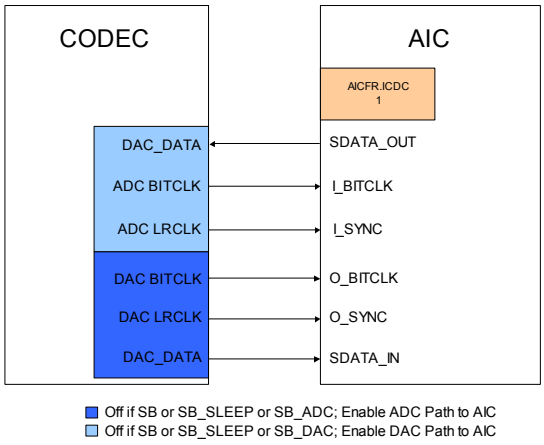


Figure 25-2 Internal CODEC works with AIC

25.2 Mapped Register Descriptions

The internal CODEC software interface includes 2 registers. They are mapped in IO memory address space of AIC module so that program can access them to control the operations of the CODEC.

Table 25-2 Internal CODEC Mapped Registers Description (AIC Registers)

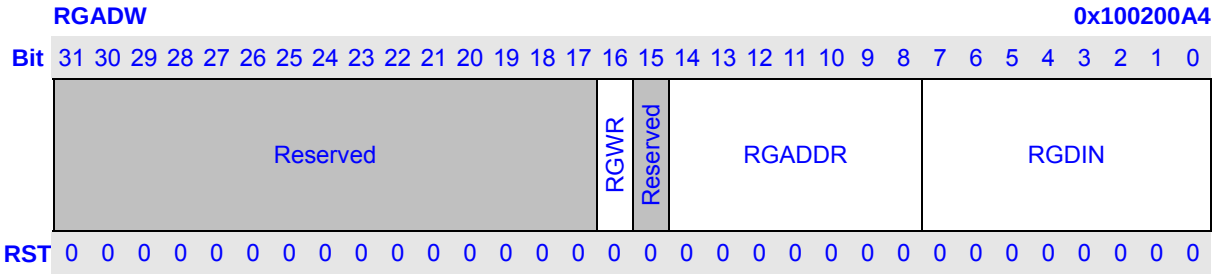
Name	Description	RW	Reset value	Address	Size
RGADW	Address, data in and write command for accessing to internal registers of internal embedded CODEC.	RW	0x00000000	0x100200A4	32
RGDATA	The read out data and interrupt request status of Internal registers data in the internal embedded CODEC.	R	0x00000000	0x100200A8	32

NOTES:

1 All these registers are AIC Registers, because they are mapped in AIC IO memory address.
2 RGADW contains data, address and write command to the internal registers of the internal CODEC.
3 RGDATA returns the internal register value of the internal CODEC and interrupt request status.

25.2.1 CODEC internal register access control (RGADW)

RGADW contains address, data and write command to the internal registers of the internal embedded CODEC.



Bits	Name	Description	RW
31:17	Reserved	Writes to these bits have no effect and always read as 0.	R
16	RGWR	Write 1 to this bit issues writing to CODEC's internal register process. This bit keeps value 1 until the current writing process is finished. A register read or a new register writing process cannot be issued before the previous writing process finished. In another word, it should not write to RGADW before RGADW.RGWR becomes 0. A writing process takes max of 0.17us plus 1 PCLK cycle. Write 0 to this bit is ignored.	RW
15	Reserved	Writes to these bits have no effect and always read as 0.	R
14:8	RGADDR	When it issues a writing to CODEC's internal register command, i.e. RGWR=1, this field specifies the register's address. In addition, this field also decides the address of the register's data out at any time.	RW
7:0	RGDIN	When it issues a writing to CODEC's internal register command, i.e. RGWR=1, this field contains the data to be written to the register.	RW

NOTES:

1 It is strong suggesting verifying the data (using read RGDATA below) after writing it to internal register of CODEC. When RGDATA returns the right data which writing to the address, the writing process is finish.

2 Please notice that AIC needs SYS_CLK (refers to AIC spec), when write new value to or read from CODEC internal registers.

25.2.2 CODEC internal register data output (RGDATA)

RGDATA returns the internal register value of the internal embedded CODEC and interrupt request status.

RGDATA **0x100200A8**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																							IRQ	RGDOUT							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:9	Reserved	Writes to these bits have no effect and always read as 0.	R
8	IRQ	This field returns the internal embedded CODEC's interrupt request. **IRQ** / **Description** 0 / **No CODEC's interrupt request found** 1 / **CODEC's interrupt request is pending**	R
7:0	RGDOUT	This field returns the value of the internal register in internal embedded CODEC. As the RGADW.RGADDR field specifies the register's address.	R

NOTES:

1 AIC needs SYS_CLK (refers to AIC spec), when write new value to or read from CODEC internal registers.

25.3 Operation

The internal embedded CODEC is controlled its internal registers. These registers can be accessed by through memory-mapped registers, RGADW and RGDATA, just like L3 bus or I2C bus for an external CODEC. AIC's BITCLK and SYNC are from/to the CODEC and is controlled by CKCFG.SELAD register. The audio data transferring, i.e. audio replaying and recording, is down by AIC. AIC still takes the role of I2S controller. We will refer to many AIC operations and registers in the following audio operation descriptions, please reference to AIC spec for the details.

This is a guide for software.

25.3.1 Access to internal registers of the embedded CODEC

The embedded CODEC is controlled through its internal registers. RGADW and RGDATA are used to write to and read from these registers. Here are some examples.

Example 1. Write to a CODEC internal register.

Step 1: RGADW.RGWR == 0.
Step 2: If not, go to step 1.
Step 3: Write to RGADW and make it.
RGADW.RGDIN = <data to be written to the register>.
RGADW.RGADDR = <the register's address >.
Step 4: Write to RGADW to commit the writing operation.
RGADW.RGWR = 1.

Example 2. Read from a CODEC internal register.

Step 1: RGADW.RGWR == 0.
Step 2: If not, go to step 1.
Step 3: write to RGADW and make it.
RGADW.RGWR = 0.
RGADW.RGDIN = <don't care>.
RGADW.RGADDR = <the register's address>.
Step 4: read RGDATA.DOUT, which returns the register's content.

25.3.2 CODEC controlling and typical operations

This section is some typical operations. We are assumed the power supply of CODEC is on, and CODEC is in STANDBY mode, CRR is configured for audio Ramping system.

Before using any of these operations, make sure AIC is configured properly as list below:

1 Make AIC to use internal CODEC mode:
AICFR.ICDC = 1; Use internal CODEC.
AICFR.AUSEL = 1; Use I2S mode.
AICFR.BCKD = 0; CODEC input BIT_CLK to AIC.

AICFR.SYNCD = 0; CODEC input SYNC to AIC.
I2SCR.AMSL = 1; Use I2S operation mode.
I2SCR.ESCLK = 1; Open SYS_CLK to internal CODEC.(if using PLL Clock)

2 Make sure AICCR.FLUSH = 0; AICFR.RST = 0; AICCR.ENLBF = 0.
3 Clear AICSR.ROR, AICSR.TUR, AICSR.RFS, AICSR.TFS = 0 to 0.
4 Set proper value to AICCR.M2S; AICCR.ENDSW; AICCR.ASVTSU.
5 Set AICFR.ENB to 1; Open AIC.

When using DMA mode, configure AICFR.RFTH, AICCR.RDMS or AICFR.TFTH, AICCR.TDMS.
Configure TX-FIFO and interrupt means setting proper value to AICFR.TFTH, clear AICCR.ETFS to 0, and clear AICCR.ETUR to 0.
Configure RX-FIFO and interrupt means setting proper value to AICFR.RFTH, clear AICCR.ERFS to 0 and clear AICCR.EROR to 0.
When configure interrupt, software must handle all the interrupt. So all interrupt is recommended disabled as shown above.
CODEC shares the interrupt with AIC module.

The register or register bit of CODEC will use the same form as the Mapped registers, but software should use the method in the section "Mapped Register Descriptions" to access this registers.

More details are listed in the CODEC guide.

25.3.3 Power saving

There are many power modes in CODEC. In every working mode, it should close stages (parts) of CODEC for saving power.

The power diagram is shown in “CODEC Power Diagram"; please refer to “CODEC Operating modes”.

25.3.4 Pop noise and the reduction of it

Please refre to “Ramping system note” and “Anti-pop operation sequences” for details.

25.3.4.1 Reference open step

1 Init play.
  Step 0: Open DMA and two AIC modules Clocks in CPM.CLKGR.
  Step 1: Configure AIC as slave and using inter CODEC mode.
    AICFR.ICDC = 1; Use internal CODEC.
    AICFR.AUSEL = 1; Use I2S mode.
    AICFR.BCKD = 0; CODEC input BIT_CLK to AIC.
    AICFR.SYNCD = 0; CODEC input SYNC to AIC.
    I2SCR.AMSL = 1; Use I2S operation mode.
    I2SCR.ESCLK = 1; Open SYS_CLK to internal CODEC.
  Step 2: Configure DMA as slave mode using internal CODEC.
2 Open.
  Step 0: Enable DMA Channel Clock.
  Step 1: Configure AIC sample size and sample rate. Configure AIC Output FIFO Threshold.
  Step 2: Configure DMA.
  Step 3: Configure CODEC.
3 Write.
  Step 0: Enable DMA Channel Clock.
  Step 1: Configure AIC.
  Step 2: Configure DMA.
  Step 3: Configure CODEC.
4 Read.
  Step 0: Enable DMA Channel Clock.
  Step 1: Configure AIC.
  Step 2: Configure DMA.
  Step 3: Configure CODEC.
5 Close.
6 End.

NOTES:

1 SB_DAC Control the internal OBIT_CLK from CODEC to AIC, First turn it on when write data

(replay).

2 SB_ADC Control the internal IBIT_CLK from CODEC to AIC, First turns it on when read data (record).

25.4 Timing parameters

Parameter	Condition	Min.	Typ.	Max.	Unit
Tsbyu	Cext = 10uF/100nF +/-20%		250	500	ms
Tshd_adc	Cext = 10uF/100nF +/-20%		200		ms
Tshd_dac	Cext = 10uF/100nF +/-20%		400	900	ms

NOTES:

1 Tsbyu is the reference wake-up time after complete power down.

2 Tshd_adc is the ADC wake-up time after sleep mode.

3 Tshd_dac is DAC wake-up time after sleep mode.

25.5 AC & DC parameters

Votages:

AVSHP and AVSCDC are connected to analog ground.

AVDHP = 3.3v (typ).

AVDCDC= 3.3v (typ).

Currents:

Mode	Currents
1 Complete down (Static)	$I_{AVDCDC} + I_{AVDHP} < 5\mu A$ (TBC)
2 SLEEP mode (Static)	TBD
3 SLEEP mode with SYS_CLK(Static)	TBD
4 Playback to AOHPR/AOHPL(Silence)	$2\ mA < I_{AVDCDC} + I_{AVDHP} < 8\ mA$
5 Record from AIL/AIR(Silence)	$1.5\ mA < I_{AVDCDC} + I_{AVDHP} < 6\ mA$
6 Playback with Record (4 + 5 Silence)	$3\ mA < I_{AVDCDC} + I_{AVDHP} < 10\ mA$
7 Playback to AOHPR/AOHPL(Digital Full Scale)	TBD
8 Record from AIL/AIR(2.8Vpp)	TBD
9 Playback with Record (7 + 8 Full Scale)	TBD

Current value is at AVDCDC = AVDHP = 3.3 V.

Chip pin Name	MAX Current across I/O @ AVDCDC = AVDHP = 3.3 V
AVDCDC	< 20 mA in normal working mode
AVSCDC	< 20 mA in normal working mode
AVDHP	< 160 mA in normal working mode
	< 1400 mA in case of short circuit
AVSHP	< 160 mA in normal working mode
	< 1400 mA in case of short circuit
AVDBTL	< 400 mA in normal working mode
AVSBTL	< 400 mA in normal working mode
VCAP	< 2 mA in normal working mode
MICP1, MICN1	< 2 mA in normal working mode
MICP2, MICN2	< 2 mA in normal working mode
MICBIAS	< 5 mA in normal working mode
AOHPL	< 80 mA in normal working mode
	< 1200 mA in case of short circuit
AOHPR	< 80 mA in normal working mode
	< 1200 mA in case of short circuit
AOBTLP, AOBTLN	< 400 mA in normal working mode
AIL, AIR	< 2 mA in normal working mode
AOLP, AOLN	< 1 mA in normal working mode
HPSENSE	< 1 mA in normal working mode

The current in case of short circuit is the max value. This current is only sink or drawn until the short circuit detection system acts.

Please refer to Chip Datasheet for more details.

25.6 CODEC internal Registers

Register Name	Function	Address	Reset value
SR	Status Register	00000 / 0x0 / 00	00
AICR	Audio Interface Control	00001 / 0x1 / 01	FC
CR1	Control Register 1	00010 / 0x2 / 02	1B
CR2	Control Register 2	00011 / 0x3 / 03	20
CR3	Control Register 3	00100 / 0x4 / 04	00
CR4	Control Register 4	00101 / 0x5 / 05	80
CCR1	Control Clock Register 1	00110 / 0x6 / 06	00
CCR2	Control Clock Register 2	00111 / 0x7 / 07	00
PMR1	Power Mode Register 1	01000 / 0x8 / 08	FF
PMR2	Power Mode Register 2	01001 / 0x9 / 09	1F
ICR	Interrupt Control Register	01010 / 0xA / 10	3F
IFR	Interrupt Flag Register	01011 / 0xB / 11	00
CGR1	Control Gain Register 1	01100 / 0xC / 12	06
CGR2	Control Gain Register 2	01101 / 0xD / 13	06
CGR3	Control Gain Register 3	01110 / 0xE / 14	06
CGR4	Control Gain Register 4	01111 / 0xF / 15	06
CGR5	Control Gain Register 5	10000 / 0x10 / 16	00
CGR6	Control Gain Register 6	10001 / 0x11 / 17	00
CGR7	Control Gain Register 7	10010 / 0x12 / 18	00
CGR8	Control Gain Register 8	10011 / 0x13 / 19	00
CGR9	Control Gain Register 9	10100 / 0x14 / 20	00
AGC1	Automatic Gain Control 1	10101 / 0x15 / 21	34
AGC2	Automatic Gain Control 2	10110 / 0x16 / 22	07
AGC3	Automatic Gain Control 3	10111 / 0x17 / 23	44
AGC4	Automatic Gain Control 4	11000 / 0x18 / 24	1F
AGC5	Automatic Gain Control 5	11001 / 0x19 / 25	00
MIX1	Digital Mixer 1	11010 / 0x1A / 26	00
MIC2	Digital Mixer 2	11011 / 0x1B / 27	00

25.6.1 CODEC internal registers

25.6.1.1 SR: Status Register

Register Name: SR Register Address: 0x0

bit7-R-0	bit6-R-0	bit5-R-0	bit4-R-0	Bit3-R-0	bit2-R-0	bit1-R-0	bit0-R-0
ACK_PON	ACK_IRQ	Reserved					

Bits	Field	Description
7	ACK_PON	Acknowledge status bit after power on. Read 0 = reset value Read 1 = codec is ready to operate
6	ACK_IRQ	Acknowledge status bit after IRQ sending. Read 0 = reset value Read 1 = codec has requested an interrupt (IRQ signal activated)

25.6.1.2 AICR: Audio Interface Control Register

Register Name: AICR Register Address: 0x1

bit7-RW-1	bit6-RW-1	bit5-RW-1	bit4-RW-1	bit3-RW-1	bit2-RW-1	bit1-RW-0	bit0-RW-0
DAC_ADWL		ADC_ADWL		DAC_SERIAL	ADC_SERIAL	DAC_I2S	ADC_I2S

Bits	Field	Description
7:4	DAC_ADWL[1:0] ADC_ADWL[1:0]	Audio Data Word Length for respectively DAC and ADC. Read / Write 00: 16-bit word length data 01: 18-bit word length data 10: 20-bit word length data 11: 24-bit word length data
3	DAC_SERIAL	Selection of DAC digital serial audio interface. Read / Write 0: Parallel interface 1: Serial interface
2	ADC_SERIAL	Selection of ADC digital serial audio interface. Read / Write 0: Parallel interface 1: Serial interface
1	DAC_I2S	Working mode of DAC serial mode. (only relevant when serial interface is selected) Read/Write 0: DSP mode 1: I2S mode
0	ADC_I2S	Working mode of ADC serial mode. (only relevant when serial interface is selected) Read/Write 0: DSP mode 1: I2S mode

NOTES:

1 DAC_I2S and ADC_I2S should be configured to 1.

25.6.1.3 CR1: Control Register 1

Register Name: CR1 Register Address: 0x2

bit7-RW-0	bit6-RW-0	Bit5-RW-0	bit4-RW-1	bit3-RW-1	bit2-RW-0	bit1-RW-1	bit0-RW-1
LOAD	-	HP_MUTE	LINEOUT_MUTE	BTL_MUTE	-	OUTSEL	

Bits	Field	Description
7	LOAD	Selection of load impedance value for ramp generation. Read/Write 0: 16 Ohm / 220uF 1: 10k Ohm / 1uF
5	HP_MUTE	Headphone output signal disabled. Read/Write 0: signal applied to headphone outputs 1: no signal on headphone outputs, acts as a mute signal
4	LINEOUT_ MUTE	Differential line output mute mode. Read/Write 0: mute inactive, signal applied to line output 1: no signal on line output
3	BTL_MUTE	BTL mute mode. Read/Write 0: mute inactive, signal applied to BTL input 1: no signal on BTL input
1-0	OUTSEL	Output Amplifier input selection. Read/Write * If MICSTEREO = 0 00: Microphone 1 input to right and left channels 01: Microphone 2 input to right and left channels 10: Bypass path enabled 11: DAC output enabled * If MICSTEREO = 1 00: Microphone 1 input to right channel and microphone 2 input to left channel 01: Microphone 2 input to right channel and microphone 1 input to left channel 10: Bypass path enabled 11: DAC output enabled

25.6.1.4 CR2: Control Register 2

Register Name: CR2 Register Address: 0x3

bit7-RW-0	Bit6-RW-0	Bit5-RW-1	Bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
MONO	-	DAC_MUTE	-	-	-	NOMAD	DAC_RIGHT_ONLY

Bits	Field	Description
7	MONO	Digital stereo-to-mono conversion for DAC path. Read/Write 0: stereo 1: mono
5	DAC_MUTE	DAC soft mute mode. Read/Write 0: mute inactive, digital input signal transmitted to the DAC 1: puts the DAC in soft mute mode
1	NOMAD	Low power mode for Head Phone. Read/Write 0: DOCK mode 1: NOMAD mode
0	DAC_RIGHT_ONLY	Deactivation of DAC left channel. Read/Write 0: DAC left channel active 1: DAC left channel inactive Note that when DAC left channel is deactivated, right channel data are transmitted to left channel analog headphone output instead. To avoid any audible pop, it is required to put the DAC in soft mute mode before modifying the DAC_RIGHT_ONLY bit.

25.6.1.5 CR3: Control Register 3

Register Name: CR3 Register Address: 0x4

bit7-RW-0	bit6-RW-0	bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-0	Bit1-RW-0	bit0-RW-0
-	-	-	-	INSEL		MICSTEREO	MICDIFF

Bits	Field	Description
3:2	INSEL[1:0]	Selection of the signal converted by the ADC. Read/Write * If MICSTEREO = 0 00: Microphone 1 input to right and left channels (CODEC automatically considers that ADC_RIGHT_ONLY equals '1' to optimize power consumption)

		01: Microphone 2 input to right and left channels (CODEC automatically considers that ADC_RIGHT_ONLY equals '1' to optimize power consumption) 10: Line input 11: Reserved for further use *If MICSTEREO = 1 00: Microphone 1 input to right channel and microphone 2 input to left channel 01: Microphone 2 input to right channel and microphone 1 input to left channel 10: Line input 11: Reserved for further use The microphone input to left channel depends on stereo or mono mode.
1	MICSTEREO	Microphone input mode selection. Read/Write 0: Microphone mono inputs 1: Microphone stereo inputs This signal affects INSEL[1:0]. Refer to its description.
0	MICDIFF	Microphone input mode selection. Read/Write 0: Microphone single-ended inputs 1: Microphone differential inputs

25.6.1.6 CR4: Control Register 4

Register Name: CR4 Register Address: 0x5

bit7-RW-1	bit6-RW-0	bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-0	Bit1-RW-0	bit0-RW-0
ADC_HPF	-	-	-	-	-	-	ADC_RIGHT_ONLY

Bits	Field	Description
7	ADC_HPF	ADC High Pass Filter enable. Read/Write 0: inactive 1: enables the ADC High Pass Filter
0	ADC_RIGHT_ONLY	Deactivation of ADC left channel. Read/Write 0: ADC left channel active 1: ADC left channel inactive

25.6.1.7 CCR1: Control Clock Register 1

Register Name: CCR1 Register Address: 0x6

bit7-RW-0	bit6-RW-0	Bit5-RW-0	bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
-				CRYSTAL			

Bits	Field	Description
3:0	CRYSTAL [3:0]	Selection of the SYS_CLK frequency. Read/Write The sampling frequency value is given in the CRYSTAL[3:0] table.

CRYSTAL[3:0]	Master Clock Frequency
0000	12 MHz
0001	13 MHz
….	Reserved for further use
1111	Reserved for further use

NOTES:

1 This register should be configured to 0x00 for setting the internal 12Mhz master clock SYS_CLK (default).

25.6.1.8 CCR2: Control Clock Register 2

Register Name: CCR2 Register Address: 0x7

bit7-RW-0	bit6-RW-0	Bit5-RW-0	bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
DAC_FREQ				ADC_FREQ			

Bits	Field	Description
7:4	DAC_FREQ[3:0]	Selection of the DAC sampling rate (Fs). Read/Write The sampling frequency value is given in the FREQ[3:0] table.
3:0	ADC_FREQ[3:0]	Selection of the ADC sampling rate (Fs). Read/Write The sampling frequency value is given in the FREQ[3:0] table.

NOTES:

1 Please refer to section Sample frequency: FREQ.

25.6.1.9 PMR1: Power Mode Register 1

Register Name: PMR1 Register Address: 0x8

bit7-RW-1	bit6-RW-1	Bit5-RW-1	bit4-RW-1	bit3-RW-1	bit2-RW-1	bit1-RW-1	bit0-RW-1
SB	SB_SLEEP	SB_AIP	SB_LINE	SB_MIC1	SB_MIC2	SB_BYPASS	SB_MICBIAS

Bits	Field	Description
7	SB	Complete power-down mode. Read/Write 0: normal mode (active) 1: complete power-down
6	SB_SLEEP	Sleep mode. Read/Write 0: normal mode (active) 1: sleep mode
5	SB_AIP	Complete AIP power-down mode. Read/Write 0: active 1: power-down
4	SB_LINE	Stereo Line input conditioning circuitry power-down mode. Read/Write 0: active 1: power-down
3	SB_MIC1	Analog MIC1 Input conditioning circuitry power-down mode. Read/Write 0: active 1: power-down
2	SB_MIC2	Analog MIC2 Input conditioning circuitry power-down mode. Read/Write 0: active 1: power-down
1	SB_BYPASS	Analog Line input (Bypass) conditioning circuitry power-down mode. Read/Write 0: active 1: power-down
0	SB_MICBIAS	Microphone biasing buffer power-down. Read/Write 0: active 1: power-down

NOTES:

1 Please refer to section CODEC Operating modes.

25.6.1.10 PMR2: Power Mode Register 2

Register Name: PMR2 Register Address: 0x9

bit7-RW-0	bit6-RW-0	bit5-RW-0	bit4-RW-1	bit3-RW-1	bit2-RW-1	bit1-RW-1	bit0-RW-1
-	-	-	SB_ADC	SB_HP	SB_BTL	SB_LOUT	SB_DAC

Bits	Field	Description
4	SB_ADC	ADC power-down mode. Read/Write 0: active 1: power-down
3	SB_HP	Headphone output stage power-down mode. Read/Write 0: headphone output stage is active 1: power-down
2	SB_BTL	BTL output conditioning circuitry power-down mode. Read/Write 0: active 1: power-down
1	SB_LOUT	Line out + BTL conditioning circuitry power-down mode. Read/Write 0: active 1: power-down
0	SB_DAC	DAC power-down mode. Read/Write 0: active 1: power-down

25.6.1.11 ICR: Interrupt Control Register

Register Name: ICR Register Address: 0xA

bit7-RW-0	bit6-RW-0	Bit5-RW-1	bit4-RW-1	bit3-RW-1	bit2-RW-1	bit1-RW-1	bit0-RW-1
INT_FORM		JACK_MASK	SCMC_MASK	RUP_MASK	RDO_MASK	GUP_MASK	GDO_MASK

Bits	Field	Description
7:6	INT_FORM[1:0]	Waveform and polarity of the IRQ signal. Read/Write 00: The generated IRQ is a high level 01: The generated IRQ is a low level 10: The generated IRQ is a high level pulse with an 8 SYS_CLK cycles duration

		11: The generated IRQ is a low level pulse with an 8 SYS_CLK cycles duration
5	JACK_MASK	Mask for the JACK_EVENT flag. 0: interrupt enabled 1: interrupt masked (no IRQ generation)
4	SCMC_MASK	Mask for the SCMC flag. 0: interrupt enabled 1: interrupt masked (no IRQ generation)
3	RUP_MASK	Mask for the RUP flag. 0: interrupt enabled 1: interrupt masked (no IRQ generation)
2	RDO_MASK	Mask for the RDO flag. 0: interrupt enabled 1: interrupt masked (no IRQ generation)
1	GUP_MASK	Mask for the GUP flag. 0: interrupt enabled 1: interrupt masked (no IRQ generation)
0	GDO_MASK	Mask for the GDO flag. 0: interrupt enabled 1: interrupt masked (no IRQ generation)

NOTES:

1 When an interrupt is masked, the event do not generates any change on the IRQ signal, but the corresponding flag value is set to '1' in the IFR register.

2 When the IRQ signal is active on level, the IRQ signal is set to the inactive level while the bits IFR[5:0] & (!ICR[5:0]) equals '0'.

3 When the IRQ signal is a pulse, the IRQ signal is set to the inactive state until a new non-masked event occurs in IFR[5:0] or until a masked event is unmasked.

4 SYS_CLK must not be stopped in order to propagate IRQ signal.

25.6.1.12 IFR: Interrupt Flag Register

Register Name: IFR Register Address: 0xB

bit7-RW-0	Bit6-R-0	bit5-RW-0	bit4-R-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
-	JACK	JACK_EVENT	SCMC	RUP	RDO	GUP	GDO

Bits	Field	Description
6	JACK	Output Jack plug detection status. 0: no jack 1: output jack present
5	JACK_EVENT	Event on output Jack plug detection status.

		0: no event 1: event detected (due to JACK flag change to '0' or '1'). Write 1 to Reset of the flag.
4	SCMC	Output short circuit detection status. 0: inactive 1: indicates that a short circuit has been detected by the output stage Write 1 to Update of the flag.
3	RUP	End of output stage ramp up flag. 1: the ramp-up sequence is completed (output stage is active). Write 1 to Reset of the flag.
2	RDO	End of output stage ramp down flag. 1: the ramp-down sequence is completed (output stage in stand-by mode) Write 1 to Reset of the flag.
1	GUP	End of mute gain up sequence flag. 1: the mute sequence is completed; the DAC input signal is transmitted to the DAC path Write 1 to Reset of the flag.
0	GDO	End of mute gain down sequence flag. 1: the mute sequence is completed, a 0 DC signal is transmitted to the DAC path Write 1 to Reset of the flag.

NOTES:

1 The flags RAMP_UP_DONE, RAMP_DOWN_DONE, GAIN_UP_DONE and GAIN_DOWN_DONE can be reset after 4 cycles of SYS_CLK.

2 Interpretation of any unspecified point is absolutely up to the designer of analog part, so it is need to pay an attention to using this flags in section "Anti-pop operation sequences".

25.6.1.13 GCR1: Control Gain Register 1

Register Name: GCR1 Register Address: 0xC

bit7-RW-0	bit6-RW-0	Bit5-RW-0	bit4-RW-0	bit3-RW-0	bit2-RW-1	Bit1-RW-1	bit0-RW-0
RLGO	-	-	GOR				

Bits	Field	Description
7	RLGO	HP amplifier gain coupling. Read/Write 0: Right and left channels gains are independent 1: Right and left channels gain track right channel gain
4:0	GOR[4:0]	Right channel HP amplifier gain programming value.

NOTES:

1 Please refer to section "Programmable attenuation: GO" for more details.

25.6.1.14 GCR2: Control Gain Register 2

Register Name: GCR2 Register Address: 0xD

bit7-RW-0	bit6-RW-0	Bit5-RW-0	bit4-RW-0	bit3-RW-0	bit2-RW-1	Bit1-RW-1	bit0-RW-0
-	-	-	GOL				

Bits	Field	Description
4:0	GOL[4:0]	Left channel HP amplifier gain programming value.

NOTES:

1 Please refer to section "Programmable attenuation: GO" for more details.

25.6.1.15 GCR3: Control Gain Register 3

Register Name: GCR3 Register Address: 0xE

bit7-RW-0	bit6-RW-0	Bit5-RW-0	bit4-RW-0	bit3-RW-0	bit2-RW-1	bit1-RW-1	bit0-RW-0
RLGI	-	-	GIR				

Bits	Field	Description
7:5	RLGI	Analog bypass gain coupling. Read/Write 0: Right and left channels gains are independent 1: Right and left channels gain track right channel gain
4:0	GIR[4:0]	Right channel Line in gain programming value.

NOTES:

1 Please refer to section "Programmable Bypass path attenuation: GI" for more details.

25.6.1.16 GCR4: Control Gain Register 4

Register Name: GCR4 Register Address: 0xF

bit7-RW-0	Bit6-RW-0	Bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-1	bit1-RW-1	bit0-RW-0
-	-	-	GIL				

Bits	Field	Description
4:0	GIL[4:0]	Left channel Line in gain programming value.

NOTES:

1 Please refer to section "Programmable Bypass path attenuation: GI" for more details.

25.6.1.17 GCR5: Control Gain Register 5

Register Name: GCR5 Register Address: 0x10

bit7-RW-0	Bit6-RW-0	Bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
RLGOD	-	-	GODR				

Bits	Field	Description
7	RLGOD	DAC digital gain coupling. Read/Write 0: Right and left channels gains are independent 1: Right and left channels gain track right channel gain
4:0	GODR[4:0]	Right channel DAC digital gain programming value.

NOTES:

1 Please refer to section "Programmable digital attenuation: GOD" for more details.

25.6.1.18 GCR6: Control Gain Register 6

Register Name: GCR6 Register Address: 0x11

bit7-RW-0	Bit6-RW-0	Bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
-	-	-	GODL				

Bits	Field	Description
4:0	GODL[4:0]	Left channel DAC digital gain programming value.

NOTES:

1 Please refer to section "Programmable digital attenuation: GOD" for more details.

25.6.1.19 GCR7: Control Gain Register 7

Register Name: GCR7 Register Address: 0x12

bit7-RW-0	Bit6-RW-0	Bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
-	-	GIM1			GIM2		

Bits	Field	Description
7	GIM1[2:0]	Microphone 1 boost stage gain programming value.
4:0	GIM2[2:0]	Microphone 2 boost stage gain programming value.

NOTES:

1 Please refer to section "Programmable boost gain: GIM".

25.6.1.20 GCR8: Control Gain Register 8

Register Name: GCR8 Register Address: 0x13

bit7-RW-0	Bit6-RW-0	Bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
RLGID	-	-	GIDR				

Bits	Field	Description
7:6	RLGID	ADC digital gain coupling. Read/Write 0: Right and left channels gains are independent 1: Right and left channels gain track right channel gain
4:0	GIDR[4:0]	Right channel ADC digital gain programming value.

NOTES:

1 Please refer to the section "Programmable input attenuation amplifier: GID".

25.6.1.21 GCR9: Control Gain Register 9

Register Name: GCR9 Register Address: 0x14

bit7-RW-0	Bit6-RW-0	Bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
-	-	-	GIDL				

Bits	Field	Description
4:0	GIDL[4:0]	Left channel ADC digital gain programming value.

NOTES:

1 Please refer to the section "Programmable input attenuation amplifier: GID".

25.6.1.22 AGC1: Automatic Gain Control Register 1

Register Name: AGC1 Register Address: 0x15

bit7-RW-0	bit6-RW-0	bit5-RW-1	bit4-RW-1	bit3-RW-0	bit2-RW-1	bit1-RW-0	bit0-RW-0
AGC_EN	Reserved	TARGET				Reserved	

Bits	Field	Description
7	AGC_EN	Selection of the AGC system. 0: inactive 1: enables the automatic level control
6	Reserved	This bit are not used, when read is 0.
5:2	TARGET	Target output level of the ADC. 0000: -6dB 0001: -7.5dB … by step of 1.5 dB 1111: - 28.5dB
1:0	Reserved	These bits are not used, when read is 00.

NOTES:

1 Please refer to section "AGC system guide" for more details.

25.6.1.23 AGC2: Automatic Gain Control Register 2

Register Name: AGC2 Register Address: 0x16

bit7-RW-0	bit6-RW-0	bit5-RW-0	bit4-RW-0	bit3-RW-0	bit2-RW-1	bit1-RW-1	bit0-RW-1
NG_EN	NG_THR			HOLD			

Bits	Field	Description
7	NG_EN	Selection of the Noise Gate system. 0: inactive 1: enables the noise gate system
6:4	NG_THR	Noise Gate Threshold value. Input level (dB) < Noise Gate Level (dB). 000: -72 dB 001: -66 dB … by step of 6dB 111: -30 dB
3:0	HOLD	Hold time before starting AGC adjustment to the TARGET value. 0000: 0ms

		0001: 2 ms 0010: 4 ms … Time Step x2 1111: 32.768s

NOTES:

1 Please refer to section "AGC system guide" for more details.

25.6.1.24 AGC3: Automatic Gain Control Register 3

Register Name: AGC3 Register Address: 0x17

bit7-RW-0	bit6-RW-1	bit5-RW-0	bit4-RW-0	bit3-RW-0	bit2-RW-1	bit1-RW-0	bit0-RW-0
ATK				DCY			

Bits	Field	Description
7:4	ATK	Attack Time - Gain Ramp Down. 0000: 32 ms 0001: 64 ms … by step of 32 ms 1111: 512 ms
3:0	DCY	Decay Time - Gain Ramp up. 0000: 32 ms 0001: 64 ms … by step of 32 ms 1111: 512 ms

NOTES:

1 DCY and ATK registers values are delays between each step of gain.

2 Please refer to section "AGC system guide" for more details.

25.6.1.25 AGC4: Automatic Gain Control Register 4

Register Name: AGC4 Register Address: 0x18

Bit7-RW-0	bit6-RW-0	bit5-RW-0	bit4-RW-1	bit3-RW-1	bit2-RW-1	bit1-RW-1	bit0-RW-1
Reserved			AGC_MAX				

Bits	Field	Description
7	Reserved	These bits are not used, when read is 000.
4:0	AGC_MAX	Maximum Gain Value to apply to the ADC path

AGC_MAX	Gain Value	AGC_MAX	Gain Value	AGC_MAX	Gain Value	AGC_MAX	Gain Value
00000	0	01000	12	10000	23	11000	32
00001	1.5	01001	13.5	10001	23	11001	33.5
00010	3	01010	15	10010	23	11010	35
00011	4.5	01011	16.5	10011	24.5	11011	36.5
00100	6	01100	18	10100	26	11100	38
00101	7.5	01101	19.5	10101	27.5	11101	39.5
00110	9	01110	21	10110	29	11110	41
00111	10.5	01111	22.5	10111	30.5	11111	42.5
AGC_MAX	Gain Value	AGC_MAX	Gain Value	AGC_MAX	Gain Value	AGC_MAX	Gain Value
00000	0	01000	12	10000	23	11000	32
00001	1.5	01001	13.5	10001	23	11001	33.5
00010	3	01010	15	10010	23	11010	35
00011	4.5	01011	16.5	10011	24.5	11011	36.5
00100	6	01100	18	10100	26	11100	38
00101	7.5	01101	19.5	10101	27.5	11101	39.5
00110	9	01110	21	10110	29	11110	41
00111	10.5	01111	22.5	10111	30.5	11111	42.5

NOTES:

1 Please refer below table for AGC_MAX setup.

2 Please refer to section "AGC system guide" for more details.

25.6.1.26 AGC5: Automatic Gain Control Register 5

Register Name: AGC5 Register Address: 0x19

bit7-RW-0	bit6-RW-0	bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
Reserved			AGC_MIN				

Bits	Field	Description
7:5	Reserved	These bits are not used, when read is 000.
4:0	AGC_MIN	Maximum Gain Value to apply to the ADC path.

NOTES:

1 Please refer to below table for AGC_MIN setup.

AGC_MIN	Gain Value	AGC_ MIN	Gain Value	AGC_ MIN	Gain Value	AGC_ MIN	Gain Value
00000	0	01000	12	10000	23	11000	32
00001	1.5	01001	13.5	10001	23	11001	33.5
00010	3	01010	15	10010	23	11010	35
00011	4.5	01011	16.5	10011	24.5	11011	36.5
00100	6	01100	18	10100	26	11100	38
00101	7.5	01101	19.5	10101	27.5	11101	39.5
00110	9	01110	21	10110	29	11110	41
00111	10.5	01111	22.5	10111	30.5	11111	42.5

2 Please refer to section "AGC system guide" for more details.

25.6.1.27 MIX1: Digital Mixer 1

Register Name: MIX1 Register Address: 0x1A

bit7-RW-0	bit6-RW-0	bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
MIX_REC1	MIX_REC0	-	GIMIX[4:0]				

Bits	Field	Description
7:6	MIX_REC	Mixer mode on ADC Path. Read/Write 00: Record input only 01: Record input + DAC 10: Reserved for further use 11: Reserved for further use
4:0	GIMIX[4:0]	Mixer gain for input path. Read/Write 00000: 0dB 00001: -1dB … by step of 1dB 11111: -31dB

25.6.1.28 MIX2: Digital Mixer 2

Register Name: MIX2 Register Address: 0x1B

bit7-RW-0	bit6-RW-0	bit5-RW-0	Bit4-RW-0	bit3-RW-0	bit2-RW-0	bit1-RW-0	bit0-RW-0
DAC_MIX1	DAC_MIX0	-	GOMIX[4:0]				

Bits	Field	Description
7:6	DAC_MIX[1:0]	Mixer mode on DAC Path. Read/Write 00: Playback DAC only 01: Playback DAC + ADC 10: Reserved for further use 11: Reserved for further use
4:0	GOMIX[4:0]	Mixer gain for DAC path. Read/Write 00000: 0dB 00001: -1dB … by step of 1dB 11111: -31dB

25.7 Programmable gains

This section helps you to configure the programmable gain amplifier in the CODEC.
Internal signal VREFP is connected to AVDCDC Pin and internal signal VREFN is connected to AVSCDC Pin.

In this section, VREF equals to (VREFP – VREFN).

25.7.1.1 Programmable boost gain: GIM

The following table gives the relation between the gain and the input level for the microphone input amplifier when GI = 0000.

GIM [2:0]	Gain value (dB)	Maximum input amplitude
000	0	0.85*VREF
001	4	0.536*VREF
010	8	0.338*VREF
011	12	0.213*VREF
100	16	0.134*VREF
101	20	0.085*VREF
110	20	0.085*VREF
111	20	0.085*VREF

NOTES:

1 Maximum analog input amplitude value is given in Vpp differential.
2 Maximum analog input amplitude is referenced as Full Scale (FS). After conversion, the corresponding digital code of the output value varies from 0x7FFF down to 0x8000 for a 16-bit word. When the analog input amplitude is greater than FS, the dynamic characteristics are not guaranteed.
3 When a change occurs on GIDi inputs, data are valid on the digital output after about 64 sample periods. If the HPF is activated, data are valid after about 64 sample periods but the offset cancellation is not still completed at this time due to its internal time constant.

25.7.1.2 Programmable input gain amplifier: GID

The digital gain of ADC path may be programmed through the registers bits GIDL[3:0] and GIDR[3:0]. The value of the gain is programmable from 0 to 23dB with a pitch of 1dB.

The gain and input levels are obtained according to the following table:

GID[4:0]	Decimal	Gain (dB)	Maximum input amplitude
0 0 0 0 0	0	0	0.85*VREF

0 0 0 0 1	1	1	0.757*VREF
0 0 0 1 0	2	2	0.6021*VREF
		…	
x x y z t	I	i	0.85 / {10^(i/20)} * VREF
		…	
1 0 1 1 1	23	23	0.06 * VREF
1 1 0 0 0	24	23	0.06 * VREF
		…	
1 1 1 1 1	31	23	0.06 * VREF

NOTES:

1 The last column of the table gives the maximum analog input to be applied on the MICi inputs. The value is given in Vpp differential. These values refer to the external voltage reference VREF equals to (VREFP – VREFN). The voltage levels depend on the VREF voltage.

25.7.1.3 Programmable digital attenuation: GOD

The attenuation of DAC output amplifier may be programmed independently for the both channels through the registers bits GODL[4:0] and GODR[4:0].

The value of the gain GODL/R is programmable from +0 to –31dB with 1 dB pitch. The gain and output levels are obtained according to the following table:

GOD*[4:[0]					Decimal decoded value	Gain Value (dB)
0	0	0	0	0	0	0
0	0	0	0	1	1	-1
					…	
0	0	1	1	0	6	-3
					…	
1	1	1	1	0	30	-30
1	1	1	1	1	31	-31

25.7.1.4 Programmable attenuation: GO

The attenuation of Headphone output amplifier may be programmed independently for the both channels through the registers bits GOL[4:0] and GOR[4:0].

The value of the gain GOL/R is programmable from +6 to –25dB with 1 dB pitch. The gain and output levels are obtained according to the following table:

GO*[4:0]					Decimal decoded value	Gain Value (dB)	Maximal PGAT input amplitude (Vpp)	Maximal PGAT output amplitude (Vpp)
0	0	0	0	0	0	+6	0.425*VREF	0.85*VREF
0	0	0	0	1	1	+5	0.478*VREF	0.85*VREF
					…		…	…
0	0	1	0	1	5	+1	0.757*VREF	0.85*VREF
0	0	1	1	0	6	0	0.85*VREF	0.85*VREF
0	0	1	1	1	7	-1	0.85*VREF	0.757*VREF
					…		…	…
1	1	1	1	0	30	-24	0.85*VREF	0.054*VREF
1	1	1	1	1	31	-25	0.85*VREF	0.048*VREF

NOTES:

1 When headphone driver is loaded by a 16 Ohm load, setting GOL/R = 0 is possible. However, set GOL/R to 12 at maximum to preserve dynamic performances. The output stage is sized to support a 50mA current and no more.

2 The last column of the table gives the analog output voltage delivered on the outputs and corresponding to a digital input at FS (Full Scale). The value is given in Vpp single-ended.

3 These values refer to the external voltage reference VREF equals to (VREFP – VREFN). The voltage levels depend on the VREF voltage.

25.7.1.5 Programmable Bypass path attenuation: GI

The analog input gain may be programmed through GIL/R[4:0].

The value of the gain is programmable from +6 to -25dB with a pitch of 1dB. The gain and input levels are obtained according to the following table:

GI*[4:0]					Decimal decoded value	Gain value (dB)	Maximum input amplitude (Vpp) (FS)
0	0	0	0	0	0	+6	0.425*VREF
0	0	0	0	1	1	+5	0.478*VREF
0	0	0	1	0	2	+4	0.536*VREF
x	y	z	t	u	i	i+6	0.85/{10^((i+6)/20)}*VREF
0	0	1	1	0	6	0	0.85*VREF
					…		0.85*VREF

1	1	1	1	1	31	-25	0.85*VREF

The last column of the table gives the maximum analog input to be applied on the line inputs. The value is given in Vpp. These values refer to the external voltage reference VREF equals to (VREFP – VREFN). The voltage levels depend on the VREF voltage.

25.7.1.6 Programmable digital mixer gain: GIMIX and GOMIX

The following table gives the relation between the gain and the input level for the microphone input amplifier when GI = 0000.

GIMIX[4:0] or GOMIX[4:0]	Gain value (dB)
00000	0
00001	-1
00010	-2
00011	-3
…	…
11101	-29
11110	-30
11111	-31

25.7.1.7 Gain refresh strategy

GI and GO gains are controlled through the control interface. To avoid sound artifacts, the gain increases or decreases each time the gain stage output crosses the zero value. Tcrossout time-out counter forces the gain to be updated if a zero crossing event doesn't occur. After each gain step, zero crossing events are ignored during at least Tcrossmin.

In case that gain coupling between both left and right channels is active (LRGi different of RLGi), gain stepping of each channel is independent from the other depending on zero crossing event occurrence.

The duration of Tcrossout and Tcrossmin are given below:

If SYS_CLK =12Mhz (default)

Tcrossout (ms) =21.8, Tcrossmin (ms) =0.171

If SYS_CLK = 13Mhz

Tcrossout (ms) =20.2, Tcrossmin (ms) =0.158

25.8 Configuration of the headphone output stage

In cap-coupled connection, codec uses the LOAD register bit to control the ramping duration). Inappropriate setting will lead to a too long or too fast ramping and will create audio artifacts.

To prevent pop-up noise generation due to floating nodes when no load is plugged in the jack connector, it is required to add some resistor devices that act as pull down function (named Rhpl and Rhpr in section “Headphone connnection” and section “Required external components”).

Its value has to be determined as following:

Working Mode	Load resistor and bypass capacitor values	LOAD value	Rhpdo value
Driving Headphone	16 Ohm / 220uF	0	470 Ohm typ.
Driving Lineout	10k Ohm / 1uF	1	4.7k Ohm max.

25.9 Out-of-band noise filtering

An internal analog Low Pass Filter at the DAC output is designed to remove the out-of-band noise generated by the delta sigma modulation (Noise Shaper). The internal LPF reduces the amount of energy contained in the wide band part (> 24 kHz) of the output signal. The out-of-band noise, when not removed, can be damageable in some high quality applications.

This filter is always working and does not need configure.

25.10 Output short-circuit protection (headphone output)

Analog short-circuit protection in the output stage has been implemented to prevent excessive current from flowing through AOHPL, AOHPR output pins. This prevents the output stage from over-heating.

The system detects the following cases:

- Abnormal headphone load

25.10.1.1 Indication of the short circuit detection

When such an overload is detected on one of AOHPL, AOHPR output pins,

- An interrupt is sent on the IRQ pin and the SCMC flag in the IFR register is set to ‘1’.
- Internally to codec:

  Automatic power-down of the 2 output amplifiers (AOHPL, AOHPR signals) when a short-circuit is detected on AOHPL or AOHPR pin (SCMC flag set to ‘1’).

25.10.1.2 Reset of short circuit detection

The following sequence has to be to apply:

1 Mask the interrupt by writing ‘1’ in the Interrupt Control Register.

2 Handle the cause of short-circuit according to the events presented in following paragraphs (Capacitor-coupled headphone connection).

3 Update the short-circuit flag by writing ‘1’ in the Interrupt Flag Register.

4 Check the reset of flag by reading the Interrupt Flag Register. The bit must be equal to ‘0’. If it remains at '1', that means that short-circuit is not resolved.

5 Enable the interrupt by writing ‘0’ in the Interrupt Control Register.

25.10.1.3 Capacitor-coupled headphone connection

It is up to the application to put the output stage in power down mode (SB_HP = ‘1’), to put codec in sleep or complete power-down mode, to reset it.

The short-circuit will be solved by the following events:

- Removal of the inserted jack (needs the use of HPSENSE pins)
- Reset of codec (NRST signal)
- Putting the output stage in power-down mode (SB_HP=1)
- Putting codec in sleep mode (SB_SLEEP=1)
- Putting codec in complete power-down mode (SB=1)

25.11 Sampling frequency: FREQ

The sampling frequency value is given in the FREQ[3:0] table below.

FREQ[3:0]	Sampling Rate (Fs)
0000	96kHz
0001	48kHz
0010	44.1kHz
0011	32kHz
0100	24kHz
0101	22.05kHz
0110	16kHz
0111	12kHz
1000	11.025kHz
1001	8kHz
1010	Reserved for further use
….	….
1111	Reserved for further use

25.12 Programmable data word length

The Data Word Length block (DWL) allows selecting the length of the input data and of the output data between 24-/20-/18-/16-bit thanks to AICR.DAC_ADWL and AICR.ADC_ADWL (respectively for the DAC and ADC paths) in accordance with the following table:

*ADWL[1:0]	Word length
0 0	16-bit word length data
0 1	18-bit word length data
1 0	20-bit word length data
1 1	24-bit word length data

The size of the buses is always 24 bits, but the input/output data only use the number of MSB programmed with ADWL. The LSB are considered as ‘0’ in input and set to ‘0’ in output.

The capability to use a data word length of 16 bits is kept for compatibility with standard applications.

25.13 Ramping system note

An internal mechanism is used to reduce output glitches when the headphone stage enters or leaves the power-down mode.

When the SB_HP is set to '1', the headphone output voltages (AOHPL, AOHPR) are slowly decreased in the same time from AVDHP/2 down to 0.

When the SB_HP is set to '0', the headphone output voltages (AOHPL, AOHPR) are slowly increased in the same time from 0 to AVDHP/2.

After power supplies ramp up, the CODEC start its internal initialization sequence and set SR.ACK_PON register bit once completed.
An interrupt request is sent when the ramp completes.

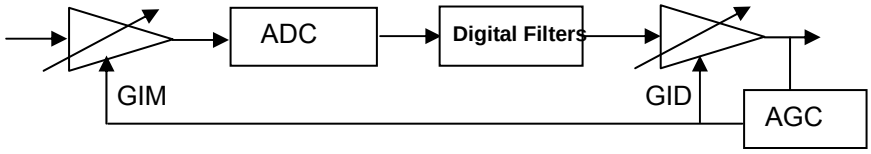
Do not change the level of SB_HP as long as the sequence due to the previous change is not complete or working not guaranteed.

In order to prevent audible glitch, it is required to power-down the output stage (SB_HP=1) before changing the output load with CR1.LOAD.

Please refer to "Anti-pop operation sequences" for details.

25.14 AGC system guide

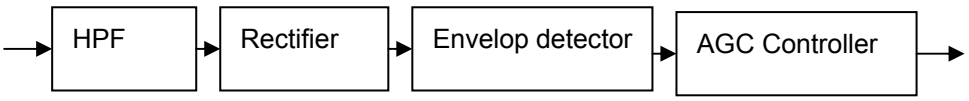
For the microphone input to ADC path, an Automatic Gain Control (AGC) system allows to optimize the signal swing at the input of the ADC.



The AGC circuit compares the output of the ADC to a level and increases or decreases the gain of the microphone preamplifier and the digital gain to compensate. The full dynamic range of the ADC can be used automatically if the audio from the microphone is to be output digitally through the ADC.

The AGC_EN register bit enables the AGC system, in this case INSEL must be equal to “00” or “01”.

AGC Block Diagram



The AGC system is used at the MIC input.
The HPF filter characteristics: Cut Frequency =300 Hz.
In the in AGC mode, the system of gain control will directly assign the values of the gains GIDL, GIDR and GIM1 (or GIM2).

25.14.1 AGC operating mode

TARGET sets the desired ADC output range level. The AGC system adapts the gain stages (GID and GIM) in order to best reach this target. AGC_MAX and AGC_MIN fix the limits of the gain variation.
Please refer to “CODEC Operating modes” for the AGC System diagram in the “CODEC Power Diagram”.

In order that the AGC system should not alter the dynamic content of the signal (voice “tonic” for instance) by continuously adapting the gain to fit the target level, the time between two consecutives gain adjustments is modifiable by the HOLD register value.

After this delay:

- If the output level is lower than TARGET, the gain is increased step by step in accordance to the DCY register value.
- If the output level is higher than TARGET, the gain is decreased step by step in accordance to the ATK register value.

The following figure illustrates the behavior of AGC system:

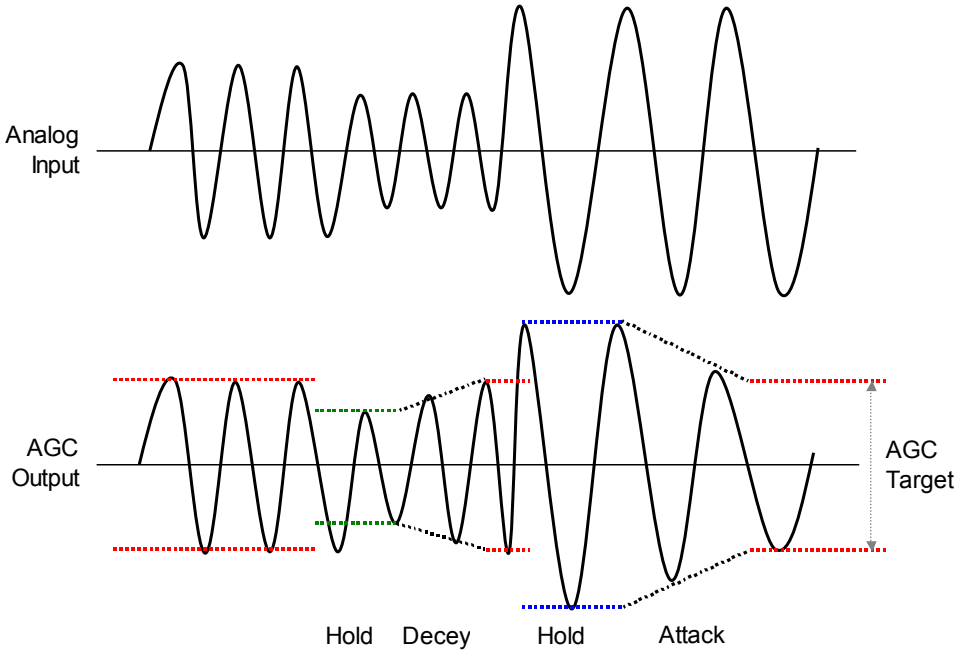


Figure 25-3 AGC adjusting waves

A noise-gating feature, enabled by the NG_EN register bit, prevents gain increases when no signal or small signal is present at the input. The noise gate threshold is set by the NG_THR register value. The following graph shows a more detailed application.

The following graph summarizes the operations and shows more details.

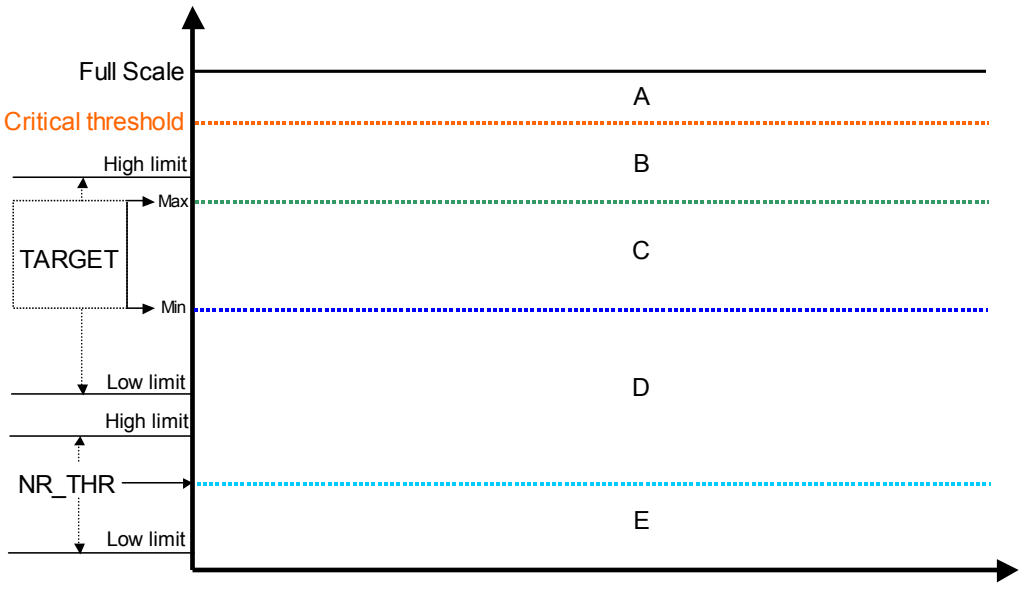


Figure 25-4 AGC adjust areas

The areas from A to E are deferent working area of AGC system, which is listing below:

A: If the signal level is in this critical area: the AGC system decreases quickly the gain at the input of the ADC until the signal goes under the critical threshold.

B: If the signal level remains in this area after the HOLD delay: the AGC system decreases the gain at the input of the ADC until the signal reaches the target area with a slope defined by AGC3.ATK register value.

C: If the signal level is in this area: the AGC system does not perform gain adjustment.

D: If the signal level remains in this area after the HOLD delay: the AGC system increases gain at the input of the ADC until the signal reach the target area with a slope defined by AGC3.DCY register value.

E: If the signal level is in this range: the AGC system considers the signal as noise and does not perform gain adjustment.

25.15 Digital Mixer description

CODEC includes a digital mixer which provides a loopback of the ADC output to the DAC and Headphone output and a loopback of the mixer output to the record path.

Two gains GIMIX[4:0] and GOMIX[4:0] control each input of the mixer to adapt the amplitude of the mixed signal. A zero-crossing detection is included on each gain stage to minimize the zipper noise.

A digital multiplexer allows choosing between the ADC signal and the mixer output signal on the record path.

Another digital multiplexer allows choosing between the DAC signal and the mixer output signal on the playback path.

Please refer to “CODEC Operating modes” for the digital mixer diagram in the “CODEC Power Diagram”.

25.16 CODEC Operating modes

Different operating modes are available:

- Power-up mode: During power on time, CODEC is in this mode.
- Reset mode: When NRST is low, CODEC is in this mode.
- Soft mute mode: When DAC_MUTE is 1, CODEC is in this mode.
- Complete Power-down mode: After RESET, CODEC is in this mode.
- SLEEP modes: When SB_SLEEP is 1, CODEC is in this mode.
- Normal mode: When CODEC is not in above mode, it is in this mode. This mode has three modes: RECORD mode, REPLAY mode, RECORD_REPLAY mode.

The power diagram is shown below.

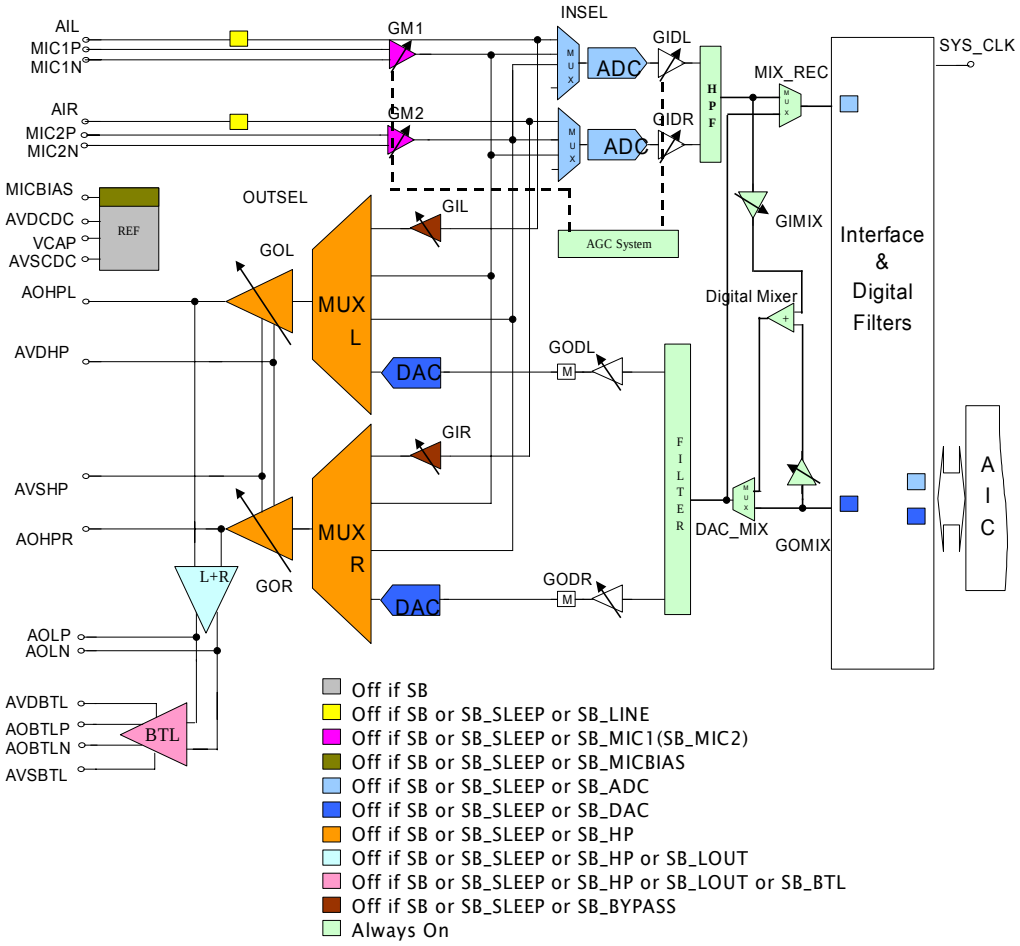


Figure 25-5 CODEC Power Diagram

There are many power parts of CODEC. Any part could be powered down independently.

25.16.1 Power-On mode and Power-Off mode

When the power supply ramps up, CODEC enters the power-on mode. During the reset, the CODEC is put in stand-by in order to reduce audible pops.

The CODEC doesn't handle the power supply ramp down on itself. The software has to turn the CODEC in complete stand-by mode before the power supply starts to ramp down.

25.16.2 RESET mode

The reset input signal is asynchronous; the reset minimum duration is one SYS_CLK cycle.
During the power-up mode and system reset, the CODEC goes into Reset mode.
After system reset the CODEC will exit Reset mode and go to STANDBY mode.

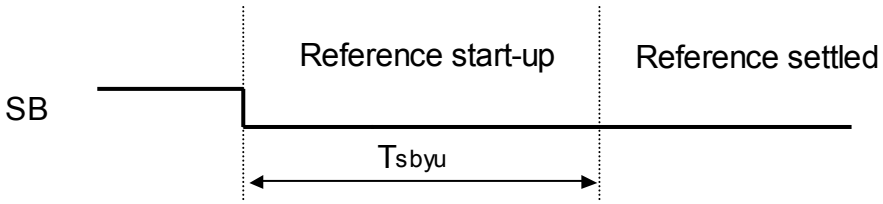
NOTES:

1 Except during the power-up mode, do NOT perform any reset in order to avoid audible pops.
2 Resetting the CODEC during normal operating mode will turn instantaneously the CODEC in STANDBY mode. This will lead to generate a large audible pop.

25.16.3 STANDBY mode

CODEC goes to STANDBY mode when the SB register bit equals 1, and all functions including ADC path, DAC path and analog references will stop and whole CODEC is shutdown for saving power. CODEC is complete down in this mode.

During the STANDBY mode, the power consumption is reduced to a minimum, so it is also called Complete Power-Down mode. When SB is set to '0', CODEC leaves the STANDBY mode. It is necessary to wait some time before the CODEC references settle. This time is called Tsbyu. When CODEC reference settled, it is in SLEEP mode.



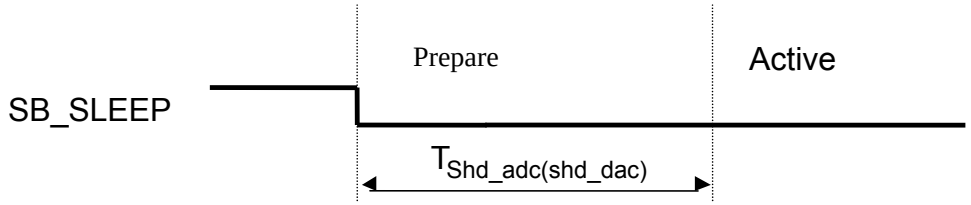
Please refer to the section "Timing parameters" for the Tsbyu Value.

25.16.4 SLEEP mode

When SB_SLEEP is set to 1, CODEC enters in sleep mode. The logical part and the analog functions, except the voltage and biasing references, enter in power-down mode. So, the power consumption is reduced without penalizing the start-up time.

When SB_SLEEP falls, CODEC leaves the corresponding STANDBY mode; it is necessary to wait

some time before the CODEC reaches the normal mode. Depending on the selected mode, this time is either called Tshd_adc (SB_ADC=0) for the ADC path or Tshd_dac (SB_DAC=0) for the DAC path.



Please refer to the section "Timing parameters" for the Tshd_adc and Tshd_dac Value.

25.16.5 Soft Mute mode

Soft Mute mode is used in order to reduce audible parasites when before the DAC enters or after leaves the Normal mode. Set the DAC_MUTE register bit to 1, it will go to Soft Mute mode.

Set DAC_MUTE to 1 puts the DAC in Soft Mute mode. The CODEC decreases progressively the digital gain from 0dB to -∞. When the gain down sequence is completed, the signal of the DAC is equal to 0 whatever the value of the digital input data is. Then CODEC generates an interrupt and if ICR.GDD_MASK is 0, and set IFR.GAIN_DOWN_DONE register bit to 1.

During Soft Mute mode, the DAC is still converting but the output final voltages (AOL, AOR) are equal to VREF/2, so the differential of the Headphone voltage is zero that cause no sound output.

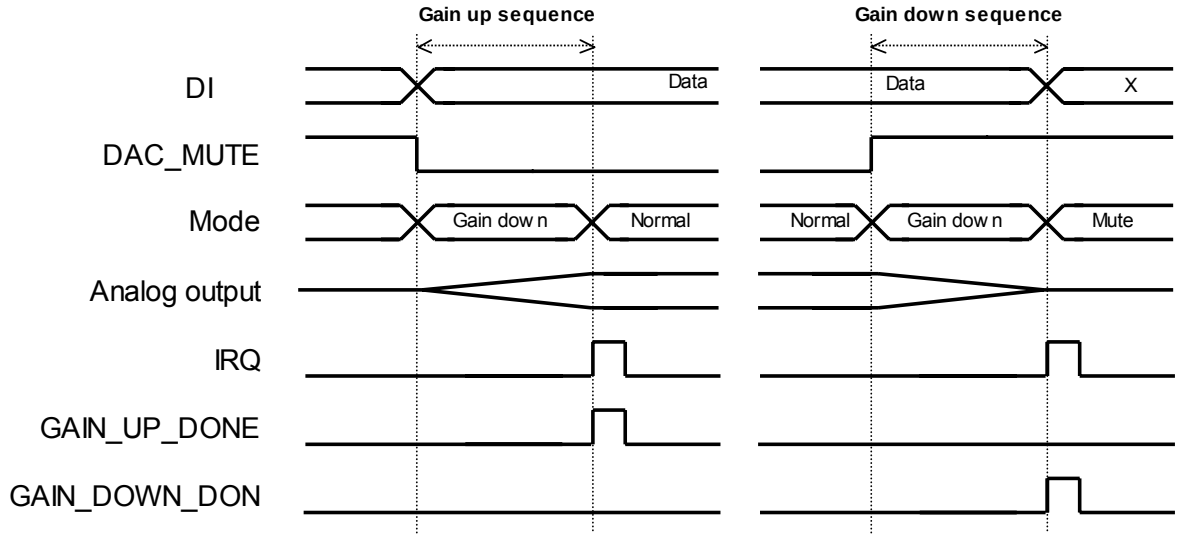


Figure 25-6 Gain up and gain down sequence

In the opposite, when DAC_MUTE is set to 0, the DAC leaves the Soft Mute mode by increasing progressively the digital gain from -∞ to 0dB. When the gain up sequence is completed, the DAC returns in Normal mode. The CODEC then generates an interrupt and if ICR.GDD_MASK is 0, and set

IFR.GAIN_UP_DONE register bit to 1.
After exiting Soft Mute mode, the DAC output will flow the DAC input data, and there is sound in the Headphone.

The duration of gain down and gain up sequences are nearly independent of Fs as shown below:

Fs(kHz)	Time(ms)	Fs(kHz)	Time(ms)	Fs(kHz)	Time(ms)
96	17.72	24	17.25	11.025	17.73
48	17.72	22.05	17.73	9.6	17.98
44.1	17.73	16	17.25	8	17.25
32	17.96	12	17.25		

NOTES:

1 Do NOT change the value of DAC_MUTE while the effect of the previous change is not reached, or the working is not guaranteed.
2 Do NOT enter in stand-by mode while the gain sequence is not completed, or the working is not guaranteed.

25.16.6 Power-Down mode and ACTIVE mode

Twelve stand-by inputs allow putting independently the different parts of CODEC into Power-Down mode.

When all SB_*=1 except SB=0 and SB_SLEEP=0, the CODEC is in ACTIVE mode, it's ready for play sound or record sound. But still need follow the anti-pop start or stop sequence. Please refer to "Start up sequence" and "Shutdown sequence".

25.16.7 Working modes summary

Different working modes are sum-up in the following table (non exhaustive table):

Working Mode		SB	SB_SLEEP	SB_DAC	SB_HP	SB_LOUT	SB_BTL	SB_ADC	SB_MICBIAS	SB_MIC1	SB_MIC2	SB LINE	SB_BTPASS	INSEL[1:0]	OUTSEL[1:0]	MICSTEREO	DAC_RIGHT_ONLY	ADC_RIGHT_ONLY	DAC_MUTE	HP_MUTE	LINEOUT_MUTE	BTL_MUTE	NOMAD
0. Reset / Complete Power-off		1	1	1	1	1	1	1	1	1	1	1	1	00	00	0	0	0	1	1	1	1	0
1. SLEEP		0	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
2. RECORD	Mono mic1 input	0	0	-	-	-	-	0	0	0	-	-	-	00	-	0	-	1	-	-	-	-	-
	Line input	0	0	-	-	-	-	0	-	-	-	0	-	10	-	-	-	0	-	-	-	-	-
	Stereo mic input, mic1 to right channel	0	0	-	-	-	-	0	0	0	0	-	-	00	-	1	-	0	-	-	-	-	-
4. REPLAY	DAC to headphone (16 ohm)	0	0	0	0	1	1	-	-	-	-	-	-	-	11	-	0	-	0	0	-	-	1
	DAC to differential lineout (mono, 10kOhm)	0	0	0	0	0	1	-	-	-	-	-	-	-	11	-	0	-	0	0	0	-	0
	DAC to BTL	0	0	0	0	0	0	-	-	-	-	-	-	-	11	-	0	-	0	0	0	0	0
5.Bypass	Line to headphone (16 Ohm)	0	0	-	0	1	1	-	-	-	-	-	0	-	10	-	-	-	-	0	-	-	1
	Line to line out (10k Ohm)	0	0	-	0	1	1	-	-	-	-	-	0	-	10	-	-	-	-	0	-	-	0
	Line to differential line out (Mono, 10k Ohm)	0	0	-	0	0	1	-	-	-	-	-	0	-	10	-	-	-	-	0	0	-	0
	Line to BTL out	0	0	-	0	0	0	-	-	-	-	-	0	-	10	-	-	-	-	0	0	0	0
6.Sidetone	Stereo mic to 16 Ohm headphone out, mic 1 to right channel	0	0	-	0	1	1	-	0	0	0	-	-	-	00	1	-	-	-	0	-	-	1
	Stereo Mic to BTL out, mic2 to right channel	0	0	-	0	0	0	-	0	0	0	-	-	-	01	1	-	-	-	0	0	-	0

NOTES:

1 The '-' means don't care this bit, but most of them should be set to 1 for reduce power.

25.17 SYS_CLK turn-off and turn-on

The main clock of CODEC is called SYS_CLK, which is generated in CPM module and called MCLK. During the SLEEP mode and the complete power-down mode, the main clock SYS_CLK may be stopped to reduce the power consumption to the leakage currents only. In other modes, the main clock SYS_CLK must not be stopped.

The main clock SYS_CLK must not be stopped until CODEC has reached the complete power-down mode and must be restarted before leaving the power-down mode.

After SYS_CLK restarts, it is required to wait 4 SYS_CLK cycles before reading or writing the registers.

When SYS_CLK is turned off (SB_SLEEP=1 or SB=1), writing on register values are not taken into account, register values are not up to date when read and interrupts not generated until SYS_CLK turns on.

25.18 Requirements on outputs and inputs selection and power-down modes

The following rules must be respected in order not to damage performances and to keep the functionality:

- If SB_BYPASS is set to 1, OUTSEL must not be equal to 10.
- If SB_LINE is set to 1, INSEL must not be equal to 10.
- If SB_MIC1 is set to 1, MICSTEREO must be equal to 0, INSEL and OUTSEL must not be equal to 00.
- If SB_MIC2 is set to 1, MICSTEREO must be equal to 0, INSEL and OUTSEL must not be equal to 01.
- If SB_DAC is set to 1, OUTSEL must not be equal to 11.

25.19 Anti-pop operation sequences

The main idea of this section is to describe the sequences to perform to minimize the audible pop to the minimum for the headphone output.

Due to the large number of stand-by combinations and to be the most flexible, the handling of the sequence from one working mode to another is left to the software. So for helping the software designer in this task, some specific sequences are automatically performed by CODEC and an interrupt mechanism (IRQ signal and associated registers) warns the application when these sequences end.

25.19.1 Initialization and configuration

To use the embedded CODEC with AIC, several AIC registers should be set up the below register of AIC before start the CODEC:

AICFR.ICDC = 1
AICFR.AUSEL = 1
AICFR.BCKD = 0
AICFR.SYNCD = 0
I2SCR.AMSL = 0
I2SCR.ESCLK = 1

25.19.2 Start up sequence (DAC)

This sequence is from Power-on mode to CODEC REPLAY mode.
The output sound is driving by DAC.
The intent of the following sequence is to prevent for large audible glitches due to the system start-up with the CODEC.

Before this sequence, setup the AIC properly.

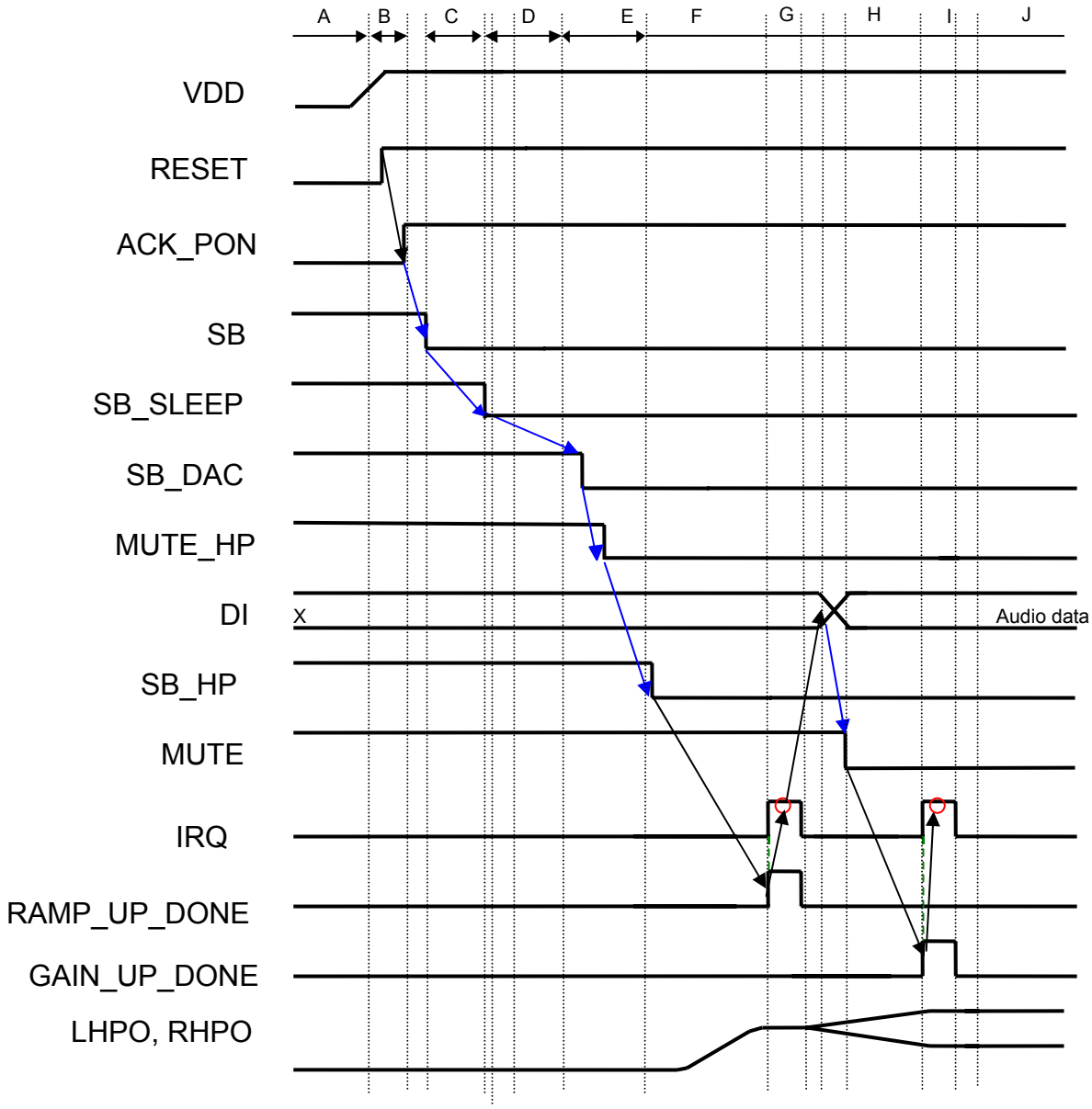


Figure 25-7 Start up sequence

NOTES:

1 The sequences in **blue** are manually handled by the software.

2 The sequences in **black** are automatically handled by the CODEC.

3 **Red** circles are interrupts automatically generated by the CODEC.

SEQUENCE:

A Initial state.
The power supply is off.

B Power supply ramp up.

The RESET of CODEC is '0' during system reset. The CODEC starts its internal initialization sequence and set ACK_PON register bit once completed.

C Starting of CODEC reference.

The software turns the CODEC on SLEEP mode by clearing SB register bit to 0. The duration equals Tsbyu. After waiting the Tsbyu duration (for example, on event generated by a timer at the application level), the CODEC is in SLEEP mode, the ADC and DAC path are ready to be turn to active mode.

D Go from SLEEP mode to active.

The application turns on the DAC by clearing SB_SLEEP register bits to 0.

E Turn on DAC.

Once after leaving SLEEP mode, the application turns on the DAC (SB_DAC=0) and after 0.5 ms switch the analog mute signal of the port to activate to 0 (MUTE_HP=0) .

F Ramp up cycle.

After waiting 1 ms, the application turns on the headphone output stage (SB_HP=0).

G Ramp up IRQ generation.

Once the ramp up cycle completes, the CODEC sets the RUP_DONE flag to 1 and generates an interrupt.

H IRQ handling and gain up cycle.

The application handles the interrupt, resets the RUP_DONE flag by writing 1 on it and releases the mute of the DAC (DAC_MUTE=0). In the same time, the application sends valid audio data to the CODEC DAC.

I Gain up IRQ generation.

Once the gain up cycle completes, the CODEC sets the GUP_DONE flag to 1 and generates an interrupt.

J IRQ handling and DAC active mode.

The application handles the interrupt and resets the GUP_DONE flag by writing 1 on. The CODEC DAC path is now fully activated.

25.19.3 Shutdown sequence (DAC)

This sequence is from CODEC REPLAY mode to STANDBY mode.
The output sound is driving by DAC.

The intent of the following sequence is to prevent for large audible glitches due to the system shutdown with the CODEC.

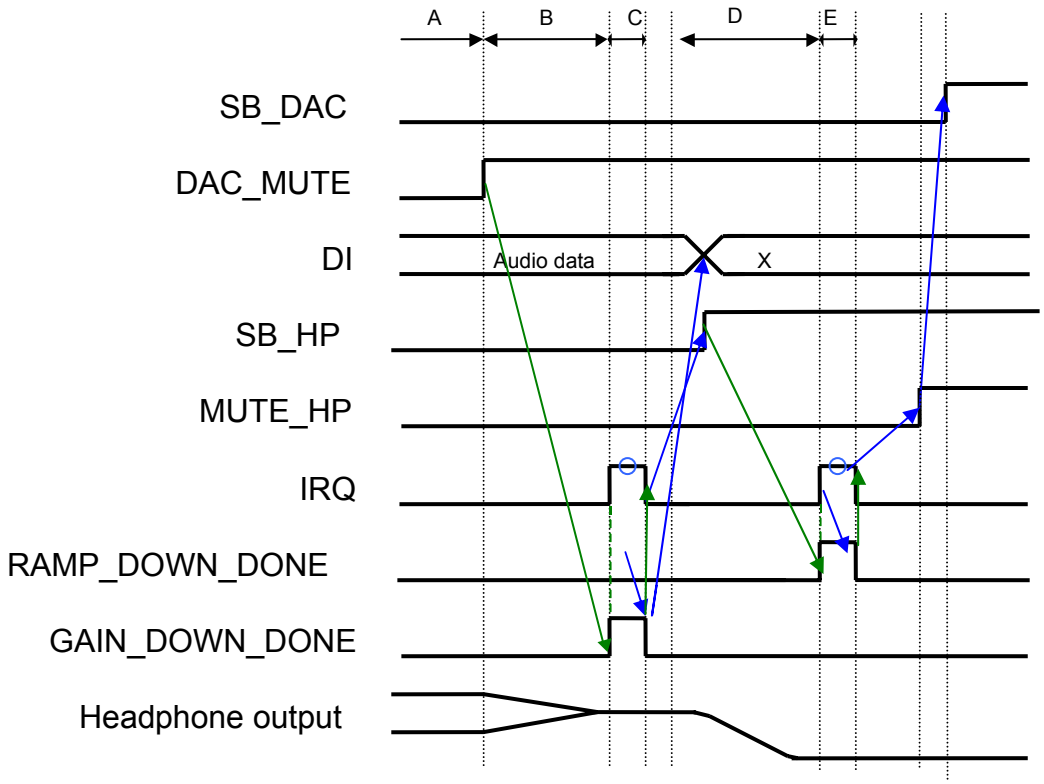


Figure 25-8 Shutdown sequence

NOTES:

1 The sequences in blue are handled by the software.
2 The sequences in black are automatically handled by the CODEC.

SEQUENCE:

A Initial state.
The power supply is on, CODEC DAC path is fully activated.

B Gain down cycle.
The application activates the mute of the DAC (DAC_MUTE=1). Once the gain down cycle completes, the CODEC sets the GDO_DONE flag to 1 and generates an interrupt.

C Gain down IRQ handling and ramp down cycle.
The application handles the interrupt and resets the GDO_DONE flag by writing 1 on it. The application can then stop sending audio data and turns off the headphone output stage

(SB_HP=1).

D Ramp down IRQ generation.

Once the ramp down cycle completes, the CODEC sets the RDO_DONE flag to 1 and generates an interrupt.

E IRQ handling.

The application handles the interrupt and resets the RDO_DONE flag by writing 1 on it. Then, the application can activate the analog mute (MUTE_HP=1). Finally, the application turns off the DAC path (SB_DAC=1) to be in sleep mode or turn off the CODEC (SB_SLEEP=1, SB=1).

25.19.4 Start up sequence (Line input)

This sequence is from Power-on mode to CODEC REPLAY mode.

The output sound is driving by Line input.

The intent of the following sequence is to prevent for large audible glitches due to the system start-up with the CODEC.

SEQUENCE:

A initial state.

DAC or Line in channel is already in use, valid analog audio signals are available at the input of the switch matrix.

B initializing output port.

The application first set the line in and headphone gain stages to their minimum value (gain automatically forced when the port is in power-down mode). This setting is taken into account in few clocks cycles. Set the MUTE_HP=0, Then the application turns on the headphone output stages (SB_HP = 0).

C Ramp up IRQ generation.

Once the ramp up cycle completes, the CODEC sets the RUP_DONE flag to 1 and generates an interrupt.

D Ramp up IRQ handling and line in stage gain up.

The application handles the interrupt and resets the RUP_DONE flag by writing 1 on it. The application then set the line in gain stage to the wished value.

The maximum duration of the gain ramping equals Trlinemax

$$Trlinemax = N1 * Tcrossout,$$

N1 is the number of line in gain steps.

Please Refer to section "Gain refresh strategy" for the value of Tcrossout.

E Headphone stage gain up.

The application set the headphone gain stage to the wished value. The maximum duration of the gain ramping equals Troutmax:

$$Troutmax = N2 * Tcrossout,$$

N2 is the number of headphone gain steps.

F active mode.

The signal path is now fully activated.

25.19.5 Shutdown sequence (Line input)

This sequence is from CODEC REPLAY mode to STANDBY mode.
The output sound is driving by Line input.

The intent of the following sequence is to prevent for large audible glitches due to the system shutdown with the CODEC.

SEQUENCE:

A active mode.
The signal path is now fully activated.

B headphone stage gain down.
The application set the headphone gain stage to the minimum value. The maximum duration of the gain ramping equals Tdoutmax

$$Tdoutmax = N3 * Tcrossout$$

N3 is the number of headphone gain steps.
Please Refer to section "Gain refresh strategy" for the value of Tcrossout.

C line in stage gain down.
The application set the line in gain stage to the minimum value. The maximum duration of the gain ramping equals Tdlinemax

$$Tdlinemax = N4 * Tcrossout$$

N4 is the number of headphone gain steps.

D Ramp down cycle.
Then, the application can activate the analog mute (MUTE_HP=1) and turns off the headphone output stages (SB_HP=1).

E Ramp down IRQ generation.
Once the ramp up cycle completes, the CODEC sets the RDO_DONE flag to '1' and generates an interrupt.

F Ramp down IRQ handling.
The application handles the interrupt and resets the RDO_DONE flag by writing '1' on it. The signal path is now off.

25.20 Circuits design suggestions

This section lists a few PCB design suggestions with difference using mode.

25.20.1 Avoid quiet ground common currents

25.20.1.1 References pins

To work properly, CODEC requires few additional external components.

CODEC includes an internal voltage reference. To insure a correct common mode biasing of the internal components, an additional voltage VCAP is used. This requires connecting two decoupling capacitors (Cext) between the pin VCAP and AVSCDC. One 10uF low ESR (ceramic or tantalum) and one 100nF ceramic have to be used. The ceramic capacitor has to be kept as close as possible to IC package (closer than 0.2 inch).

25.20.1.2 Power supply pins

CODEC analog power supplies require external decoupling capacitors.

For each power supply pin, one 100nF ceramic capacitor has to be used. This ceramic capacitor has to be kept as close as possible to IC package (closer than 0.2 inch). One low ESR (ceramic or tantalum) capacitor has to be used to decouple the analog power supply provided to the CODEC. Its value depends on the power supply generator; its typical value is between 1uF and 10uF. Ideally use separate ground planes for analog and digital parts.

Connect all ground pins with thick traces to power plane in order to ensure lowest impedance connections.

AVSCDC must be connected to the PCB analog single point reference (star connection) ground (AGND).

25.20.2 Headphone connection (Capacitor-coupled)

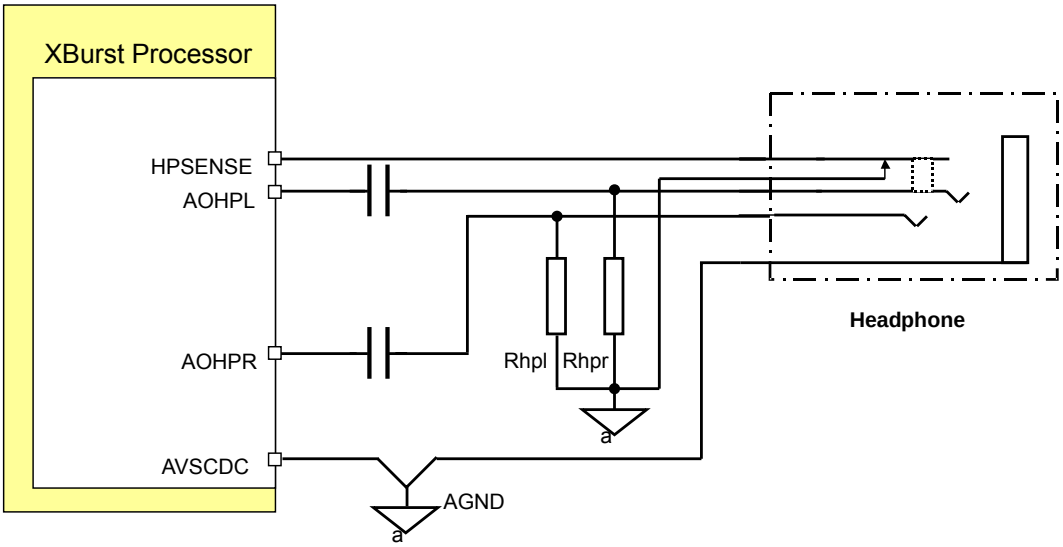


Figure 25-9 Capacitor-coupled connection

The AOHPL and AOHPR pins are connected to the headphone through an external bypass capacitor which is a DC blocking capacitors.

This capacitor is called C_L. When the headphone resistance R_L is 16 Ohm, The tantalum blocking capacitor C_L is 220 uF.

The DC value of the signal AOHPL or AOHPR equals to AVDCDC/2.

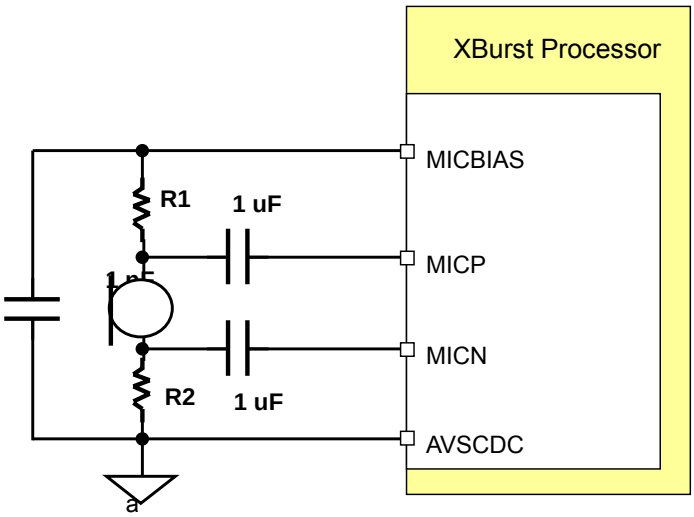
The ground of the headphone is connected to AGND, which is the PCB analog single point reference (star connection) ground.

25.20.3 Microphone connection

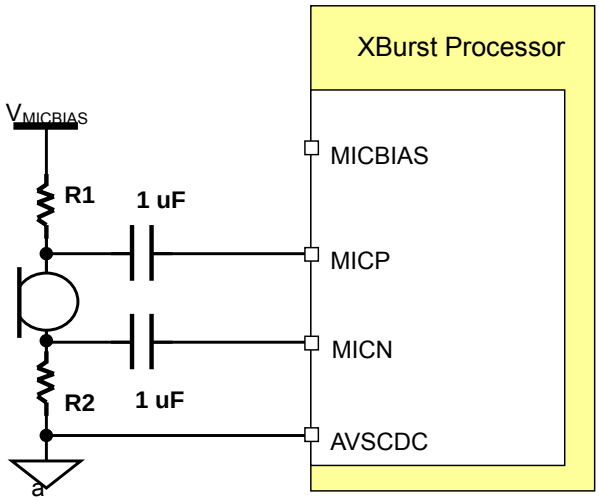
The optimal performance for the SNR is obtained in differential Microphone inputs with a FS input level corresponding to the following values: the peak-to-peak amplitude of the signal is 0.28V, corresponding to 0.085*VREF Vpp.

We recommend customer to use differential MIC input for better performance.

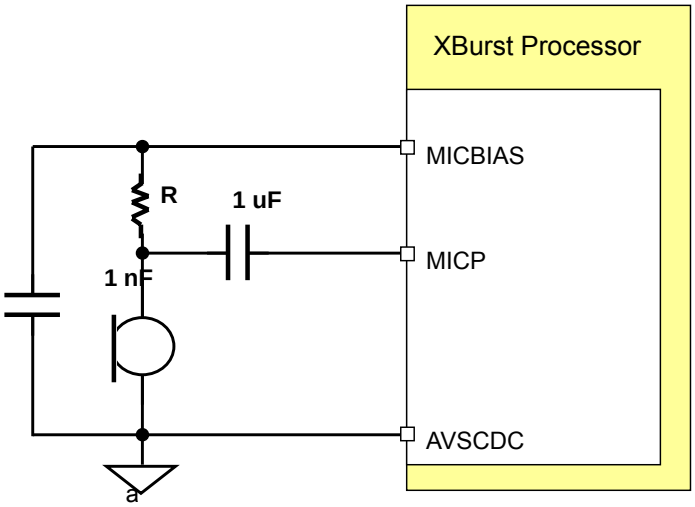
Application schematic with differential MIC input (using MICBIAS pin):



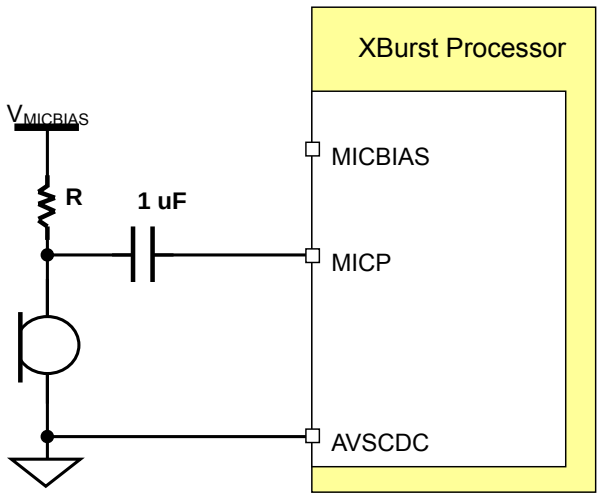
Application schematic with differential MIC input (Vmicbias generated on board):



Application schematic with single-ended MIC input (using MICBIAS pin):



Application schematic with single-ended MIC input (Vmicbias generated on board):



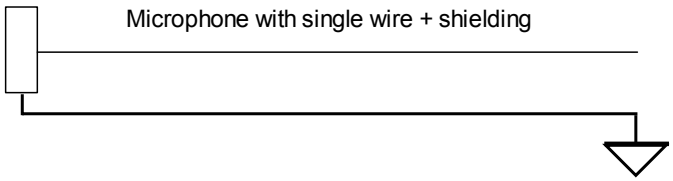
In single-ended connection, one external resistor (R) has to be used to bias the electret microphone. In differential connection, a pair of external resistor (R1, R2) has to be used to bias the electret microphone. The resistors value relation between them is R1 = R2 = R/2.

Specific value of resistor (R, commonly from 2.2k Ohm to 4.7k Ohm) and Vmicbias (if generated on board, usually from 1 to 2V or more) depends on the selected EC (Electret Condenser) microphone. The 1nf decoupling capacitance used in MICBIAS pin removes high frequency noise of the chip.

Setting SB_MIC1/SB_MIC2 to 1 will close microphone input path for saving power, also setting SB_MICBIAS to 1 will close MICBIAS stage and the MICBIAS output voltage will be zero.

MICBIAS output voltage scales with AVDCDC, equals to 5/6*AVDCDC (typical 2.75v).
MICBIAS output current is 4mA max.
MICBIAS output noise is 40uVrms max.

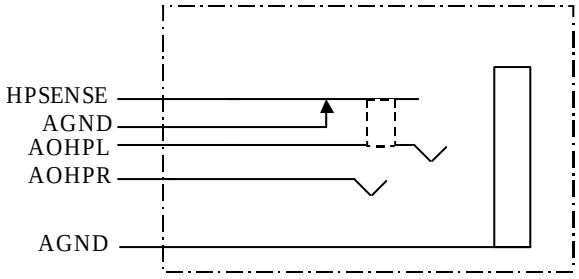
This configuration is better suited for microphone with single wire + shielding.



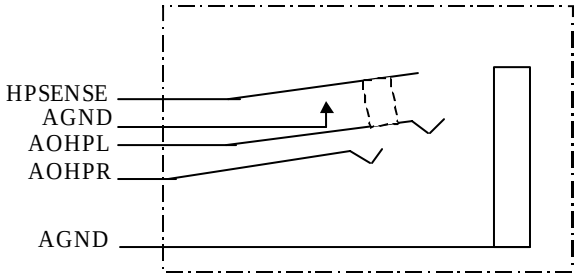
The AVSCDC Pin is connected the analog quiet reference ground in the chip (refers to Grounds and analog signal references). So the ground of MIC must be connected to AVSCDC using a star connection.

25.20.4 Description of the connections to the jack

When the jack is inserted, “sense” and “ground” are disconnected.



No jack plugged: the switch acts as a short-circuit.



Jack plugged: the switch acts as an open circuit.

25.20.4.1 Grounds and analog signal references

In order to limit the parasitic disturbances from the AVSHP and AVSBTL output power supplies to inter VREFN analog quiet ground(which is using AVSCDC pin), should use the following principle to distribute the grounds.

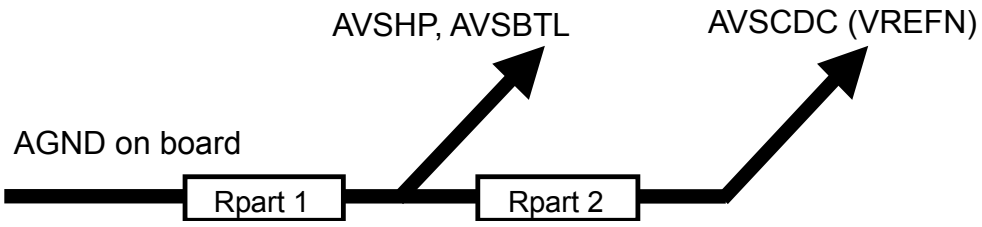


Figure 25-10 Ground distributing

Minimize the values of the connections parasitic resistance Rpar1, Rpar2.

Take a special care for Rpar1 in order to limit the disturbance from the output stages (AVSHP and AVSBTL) to the signal reference (VREFN).

The reference of the input signals must be connected to VREFN (internal quiet ground which using the AVSCDC pin) using a star connection.

25.20.5 PCB considerations

To work properly, CODEC analog power supplies require external decoupling capacitors.

In the VCAP pin, one 10uF low ESR (ceramic or tantalum) called C2 and one 100nF ceramic called C1 have to be used. The ceramic capacitor has to be kept as close as possible to IC package (closer than 0.2 inch).

For each power supply pin, one 100nF ceramic capacitor has to be used. The capacitor used in AVDCDC pin is called C4, the capacitor used in AVDHP pin is C6, and the capacitor used in AVDBTL pin is called C7. These ceramic capacitors have to be kept as close as possible to IC package (closer than 0.2 inch).

One low ESR (ceramic or tantalum) capacitor called C3 has to be used to decouple the analog power supply provided to the CODEC. Its value depends on the power supply generator; its typical value is between 1uF and 10uF. Ideally use separate ground planes for analog and digital parts.

C1, C2, C3, C4, C5, C6 and C7 are defined in section "Required external components".

The reference PCB design is shown below:

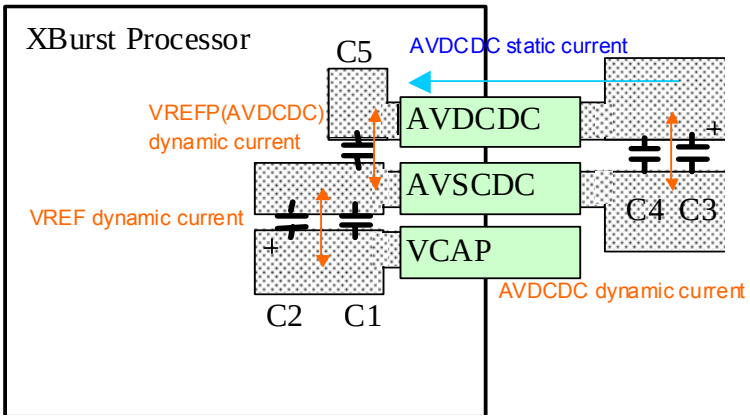


Figure 25-11 the bottom corner of chip PCB Layer

This is just an example reference diagram. You should change and select the PCB layer and route with your design constraints.

25.20.5.1 Required external components

The following table summarizes the external components required for a proper working of CODEC, except those used for the analog input and output signals.

Name	Description	Typical Value	Unit
C1	Ceramic reference decoupling capacitor. Cext.	100	nF
C2	Tantalum reference decoupling capacitor. Cext.	10	uF
C3	Tantalum analog power supply decoupling capacitor.	1 to 10	uF
C4	Ceramic AVDCDC decoupling capacitor.	100	nF
C5	Ceramic inter signal VREFP decoupling capacitor (can be shared with C4).	100	nF
C6	Ceramic AVDHP decoupling capacitor. Not shown in section PCB considerations.	100	nF
C7	Ceramic AVDBTL decoupling capacitor. Not shown in section PCB considerations.	100	nF
C8	MICBIAS decoupling capacitor, Refer to section "Microphone connection".	1	nF
C9, C10	External bypass capacitor, for DC blocking, Refer to section "Headphone connection (Capacitor-coupled)".	220	uF
Rhpl, Rhpr	Headphone jack pull-down resistors, Refer to section "Headphone connection (Capacitor-coupled)".	470 or 4.7K	Ohm
R	In single-ended connection, one external resistor (R) has to be used to bias the electret microphone. Refer to section "Headphone connection (Capacitor-coupled)".	2.2K ~ 4.7K	Ohm

26 AC97/I2S/SPDIF Controller

26.1 Overview

This chapter describes the AIC (AC'97 and I^2S Controller) included in this processor.

The AIC supports the Audio Codec '97 Component Specification 2.3 for AC-link format and I2S or IIS (for inter-IC sound), a protocol defined by Philips Semiconductor. Both normal I2S and the MSB-justified I2S formats are supported by AIC.

AIC consists of buffers, status registers, control registers, serializers, and counters for transferring digitized audio between the processor's system memory and an internal I2S CODEC, an external AC97 or I2S CODEC. AIC can record digitized audio by storing the samples in system memory. For playback of digitized audio or production of synthesized audio, the AIC retrieves digitized audio samples from system memory and sends them to a CODEC through the serial connection with AC-link or I2S formats. The internal or external digital-to-analog converter in the CODEC then converts the audio samples into an analog audio waveform. The audio sample data can be stored to and retrieved from system memory either by the DMA controller or by programmed I/O.

The AC-link is a synchronous, fixed-rate serial bus interface for transferring CODEC register control and status information in addition to digital audio. Where both normal I2S and MSB-justified-I2S work with a variety of clock rates, which can be obtained either by dividing the PLL clock by two programmable dividers or from an external clock source.

For I2S systems that support the L3 control bus protocol, additional pins are required to control the external CODEC. CODECs that use an L3 control bus require 3 signals: L3_CLK, L3_DATA, and L3_MODE for writing bytes into the L3 bus register. The AIC supports the L3 bus protocol via software control of the general-purpose I/O (GPIO) pins. The AIC does not provide hardware control for the L3 bus protocol.

To control the internal CODEC, internal CODEC Spec can be referenced.

SPDIF (Sony/Philips Digital Interconnect Format) is a digital audio interface. The transmission medium can be either electrical or optical. Supports IEC60958 two-channel PCM audio and IEC61937 multi-channel compressed audio (Dolby Digital, DTS, etc.).

This chapter describes the programming model for the AIC. The information in this chapter requires an understanding of the AC'97 specification, Revision 2.3.

26.1.1 Block Diagram

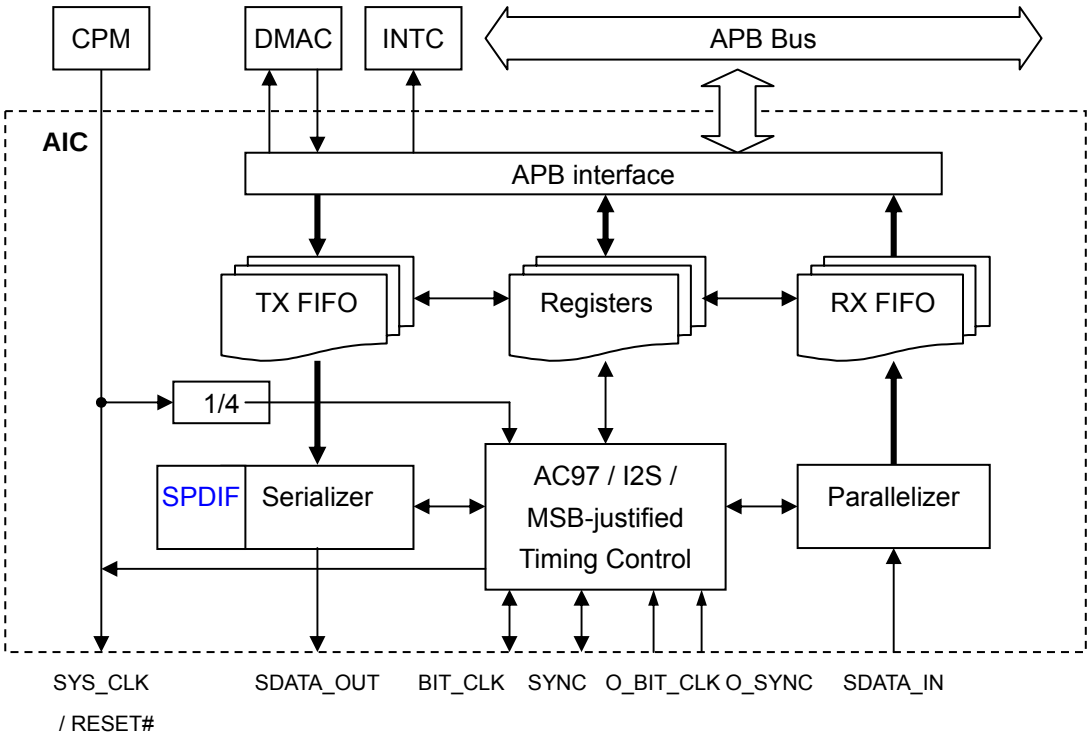


Figure 26-1 AIC Block Diagram

The O_BIT_CLK and O_SYNC ports are only used by inter CODEC.

26.1.2 Features

AIC support following AC97/I2S/SPDIF features:

- AC-link (AC97) features
  - Up to 20 bit audio sample data sizes supported
  - DMA transfer mode supported
  - Stop serial clock supported
  - Programmable Interrupt function supported
  - Support mono PCM data to stereo PCM data expansion on audio play back
  - Support endian switch on 16-bits normal audio samples play back
  - Support variable sample rate in AC-link format
  - Multiple channel output and double rated supported for AC-link format
  - Power Down Mode and two Wake-Up modes Supported for AC-link format

- I2S features
  - 8, 16, 18, 20 and 24 bit audio sample data sizes supported, 16 bits packed sample data is supported
  - Up to 8 channels sample data supported
  - DMA transfer mode supported
  - Stop serial clock supported
  - Programmable Interrupt function supported
  - Support mono PCM data to stereo PCM data expansion on audio play back
  - Support endian switch on 16-bits normal audio samples play back
  - Internal programmable or external serial clock and optional system clock supported for I2S or MSB-Justified format
  - Internal I2S CODEC supported
  - Two FIFOs for transmit and receive respectively

- SPDIF features
  - 8, 16, 18, 20 and 24 bit audio sample data sizes supported
  - DMA transfer mode supported
  - Stop serial clock supported
  - Programmable Interrupt function supported
  - Support IEC60958 two-channel PCM audio
  - Support IEC61937 multi-channel compressed audio
  - Support consumer mode and only support transmitter mode
  - Profession mode is not supported
  - The User data bit is ‘0’as it is not supported in the chip
  - Support sampling frequency from 32kHz to 192kHz

26.1.3 Interface Diagram

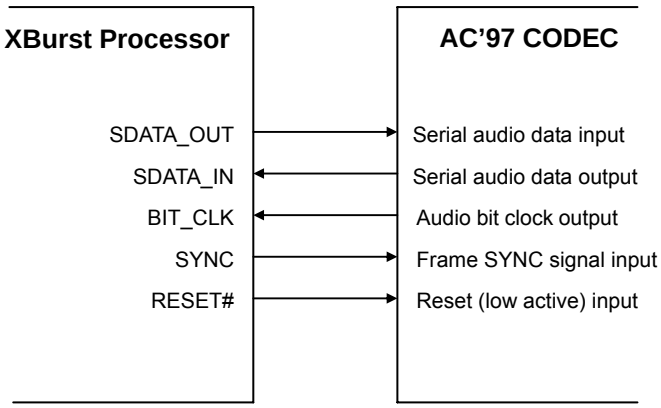


Figure 26-2 Interface to an External AC'97 CODEC Diagram

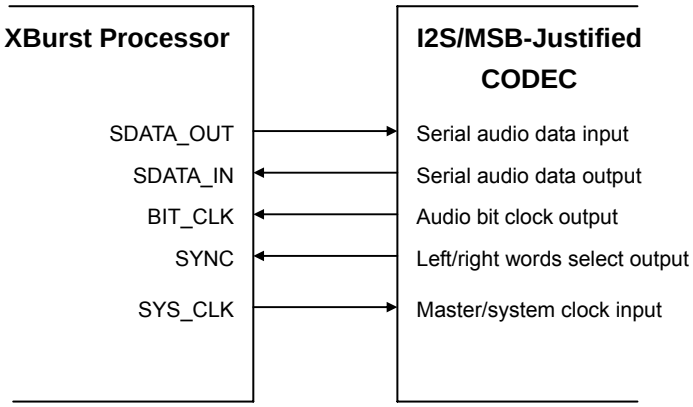


Figure 26-3 Interface to an External Master Mode I2S/MSB-Justified CODEC Diagram

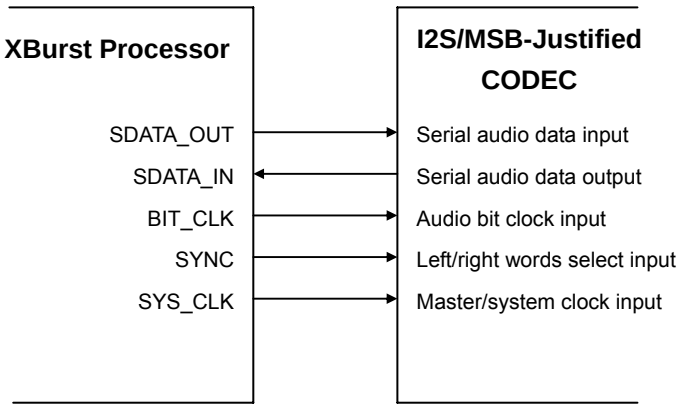


Figure 26-4 Interface to an External Slave Mode I2S/MSB-Justified CODEC Diagram

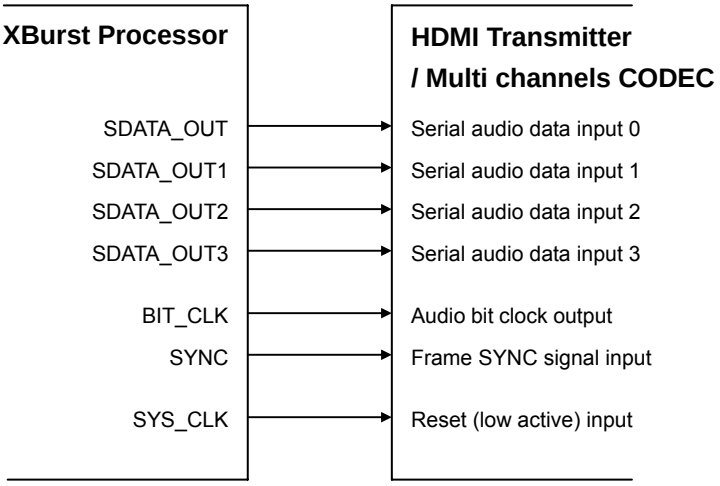


Figure 26-5 Interface to a HDMI Transmitter via I2S Diagram

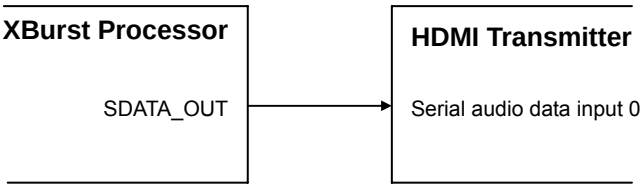


Figure 26-6 Interface to a HDMI Transmitter via SPDIF Diagram

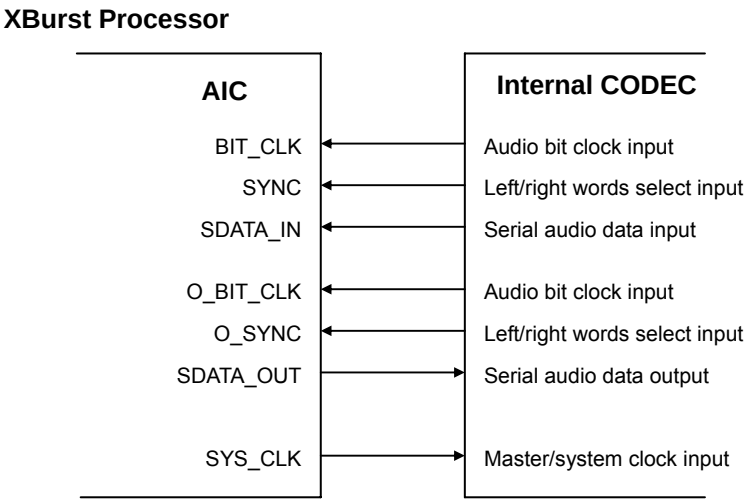


Figure 26-7 Interface to an internal Master Mode I2S CODEC Diagram

Please refer to the related CODEC specification for the details.

26.1.4 Signal Descriptions

There are all 5 pins used to connect between AIC and an external audio CODEC device. If an internal CODEC is used, these pins are not needed. Please refer to Chip Spec. They are listed and described

in Table 26-1.

Table 26-1 AIC Pins Description

Function Name	PIN Name	I/O	Description
RESET# **SYS_CLK**	SCLK_RSTN	O	RESET#: AC-link format, active-low CODEC reset. SYS_CLK: I2S/MSB-Justified formats, supply system clock to CODEC.
BIT_CLK	BCLK	I I/O	12.288 MHz bit-rate clock input for AC-link, and sample rate dependent bit-rate clock input/output for I2S/MSB-Jistified.
SYNC	SYNC	O	48-kHz frame indicator and synchronizer for AC-link format.
		I/O	Indicates the left- or right-channel for I2S/MSB-Justified format.
SDATA_OUT3	SDATO3	O	Serial audio output data I2S line 3.
SDATA_OUT2	SDATO2	O	Serial audio output data I2S line 2.
SDATA_OUT1	SDATO1	O	Serial audio output data I2S line 1.
SDATA_OUT	SDATO	O	Serial audio output data to CODEC / I2S line 0 / SPDIF output.
SDATA_IN	SDATI	I	Serial audio input data from CODEC.

The O_BIT_CLK and O_SYNC signals are not connected to any pin for only using by internal CODEC.

26.1.4.1 RESET# / SYS_CLK Pin

RESET# is AC97 active-low CODEC reset, which outputs to CODEC. The CODEC's registers are reset when this RESET# is asserted. This pin is useful only in AC-link format. If AIC is disabled, it retains the high.

SYS_CLK outputs the system clock to CODEC. This pin is useful only in I2S/MSB-justified format. It generates a frequency between approximately 2.048 MHz and 24.576 MHz by dividing down the PLL clock with a programmable divisor. This frequency can be 256, 384, 512 and etc. times of the audio sampling frequency. Or it can be set to a wanted frequency. If AIC is disabled, it retains the high.

26.1.4.2 BIT_CLK Pin

BIT_CLK is the serial data bit rate clock, at which AC97/I2S data moves between the CODEC and the processor. One bit of the serial data is transmitted or received each BIT_CLK period. It is fixed to 12.288 MHz in AC-link format, which inputs from the CODEC. In I2S and MSB-justified format it inputs from the CODEC in slave mode and outputs to CODEC in master mode. In the master mode, the clock is generated internally that is 64 times the sampling frequency. Table 26-7 lists the available sampling frequencies based on an internal clock source. If AIC is disabled, AICFR.AUSEL and AICFR.BCKD determine the direction. And it retains the low if it is output and the state is undefined if it is input.

26.1.4.3 SYNC Pin

In AC-link format, SYNC provides frame synchronization, fixed to 48kHz, by specifying beginning of an audio sample frame and outputs to CODEC. In I2S/MSB-Justified formats, SYNC is used to indicate left- or right-channel sample data and toggled in sample rate frequency. It outputs to CODEC in master mode and inputs from CODEC in slave mode. If AIC is disabled, AICFR.AUSEL and AICFR.BCKD determine the direction. And it retains the low if it is output and the state is undefined if it is input.

26.1.4.4 SDATA_OUT Pin

SDATA_OUT is AIC output data pin, which outputs AC97/I2S serial audio data, SPDIF serial data or data of AC97 CODEC register control to an external audio CODEC device.
If in multi channels mode, it outputs the first two channels serial data.
If AIC is disabled, it retains the low.

SDATA_OUTn (n = 1,2,3) is AIC output data pin, which outputs multi channels serial audio data.

26.1.4.5 SDATA_IN Pin

SDATA_IN is AIC inputs data pin, which inputs serial audio data or data of AC97 CODEC register status from an external audio CODEC device. If AIC is disabled, its state is undefined.

26.2 Register Descriptions

AIC software interface includes 13 registers and 1 FIFO data port. They are mapped in IO memory address space so that program can access them to control the operation of AIC and the outside CODEC.

Table 26-2 AIC Registers Description

Name	Description	RW	Reset value	Address	Size
AICFR	AIC Configuration Register	RW	0x07100000	0x10020000	32
AICCR	AIC Common Control Register	RW	0x01240000	0x10020004	32
ACCR1	AIC AC-link Control Register 1	RW	0x00000000	0x10020008	32
ACCR2	AIC AC-link Control Register 2	RW	0x00000000	0x1002000C	32
I2SCR	AIC I2S/MSB-justified Control Register	RW	0x00000000	0x10020010	32
AICSR	AIC FIFO Status Register	RW	0x00000008	0x10020014	32
ACSR	AIC AC-link Status Register	RW	0x00000000	0x10020018	32
I2SSR	AIC I2S/MSB-justified Status Register	RW	0x00000000	0x1002001C	32
ACCAR	AIC AC97 CODEC Command Address Register	RW	0x00000000	0x10020020	32
ACCDR	AIC AC97 CODEC Command Data Register	RW	0x00000000	0x10020024	32
ACSAR	AIC AC97 CODEC Status Address Register	R	0x00000000	0x10020028	32
ACSDR	AIC AC97 CODEC Status Data Register	R	0x00000000	0x1002002C	32
I2SDIV	AIC I2S/MSB-justified Clock Divider Register	RW	0x00000003	0x10020030	32
AICDR	AIC FIFO Data Port Register	RW	0x????????	0x10020034	32
SPENA	SPDIF Enable Register	RW	0x00	0x10020080	8
SPCTRL	SPDIF Control Register	RW	0x0003	0x10020084	16
SPSTATE	SPDIF Status Register	RW	0x0000	0x10020088	16
SPCFG1	SPDIF Configure 1 Register	RW	0x00000000	0x1002008C	32
SPCFG2	SPDIF Configure 2 Register	RW	0x00000000	0x10020090	32
SPFIFO	SPDIF FIFO Register	W	0x????????	0x10020094	32
CKCFG	Clock Configure for the embedded CODEC to AIC	RW	0x00000000 0x00000002	0x100200A0	32
RGADW	Address, data in and write command for accessing to internal registers of embedded CODEC	RW	0x00000000	0x100200A4	32
RGDATA	The read out data and interrupt request status of Internal registers	R	0x00000000	0x100200A8	32

	data in the embedded CODEC				

1 AICFR is used to control FIFO threshold, AC-link or I2S/MSB-justified selection, AIC reset, master/slave selection, and AIC enable.
2 AICCR is used to control DMA mode, FIFO flush, interrupt enable, internal loop-back, play back and recording enable. It also controls sample size and signed/unsigned data transfer.
3 ACCR1 is used to reflect/control valid incoming/outgoing slots of AC97.
4 ACCR2 is used to control interrupt enable, output/input sample size, and alternative control of RESET#, SYNC and SDATA_OUT pins in AC-link.
5 I2SCR is used to control BIT_CLK stop, audio sample size, I2S or MSB-justified selection in I2S/MSB-justified.
6 AICSR is used to reflect FIFOs status.
7 ACSR is used to reflect the status of the connected external CODEC in AC-link.
8 I2SSR is used to reflect AIC status in I2S/MSB-justified.
9 ACCAR and ACCDR are used to hold address and data for AC-link CODEC register read/write.
10 ACSAR and ACSDR are used to receive AC-link CODEC registers address and data.
11 I2SDIV is used to set clock divider for BIT_CLK generating in I2S/MSB-justified format.
12 AICDR is act as data input/output port to/from transmit/receive FIFO when write/read.
13 CKCFG, RGADW and RGDATA are used to access internal CODEC, please refer to CODEC Spec.

26.2.1 AIC Configuration Register (AICFR)

AICFR contains bits to control FIFO threshold, AC-link or I2S/MSB-justified selection, AIC reset, master/slave selection, and AIC enable.

AICFR **0x10020000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				RFTH				Reserved			TFTH					Reserved									LSMP	ICDC	AUSEL	RST	BCKD	SYNCD	ENB
RST	0	0	0	0	0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:28	Reserved	Writes to these bits have no effect and always read as 0.	R
27:24	RFTH	Receive FIFO threshold for interrupt or DMA request. The RFTH valid value is 0 ~ 15. This value represents a threshold value of (RFTH + 1) * 2. When the sample number in receive FIFO, indicated by AICSR.RFL, is great than or equal to the threshold value, AICSR.RFS is set. Larger RFTH value provides lower DMA/interrupt request frequency but have more risk to involve receive FIFO overflow. The optimum value is system dependent.	RW
23:21	Reserved	Writes to these bits have no effect and always read as 0.	R
20:16	TFTH	Transmit FIFO threshold for interrupt or DMA request. The TFTH valid value 0 ~ 31. This value represents a threshold value of TFTH * 2. When the sample number in transmit FIFO, indicated by AICSR.TFL, is less than or equal to the threshold value, AICSR.TFS is set. Smaller TFTH value provides lower DMA/interrupt request frequency but have more risk to involve transmit FIFO underflow. The optimum value is system dependent.	RW
15:7	Reserved	Writes to these bits have no effect and always read as 0.	R
6	LSMP	Select between play last sample or play ZERO sample in TX FIFO underflow. ZERO sample means sample value is zero. This bit is better be changed while audio replay is stopped. **LSMP** / **CODEC used** 0 / **Play ZERO sample when TX FIFO underflow.** 1 / **Play last sample when TX FIFO underflow.**	RW
5	ICDC	Internal CODEC used. Select between internal or external CODEC. **ICDC** / **CODEC used** 0 / **External CODEC.** 1 / **Internal CODEC.**	RW
4	AUSEL	Audio Unit Select. Select between AC-link and I2S/MSB-justified. Change this bit in case of BIT_CLK is stopped (I2SCR.STPBK = 1). **AUSEL** / **Selected**	RW

		0 **Select AC-link format.** 1 **Select I2S/MSB-justified format.**	
3	RST	Reset AIC. Write 1 to this bit reset AIC registers and FIFOs except AICFR and I2SDIV register. Writing 0 to this bit has no effect and this bit is always reading 0.	W
2	BCKD	BIT_CLK Direction. This bit specifies input/output direction of BIT_CLK. It is only valid in I2S/MSB-justified format. When AC-link format is selected, BIT_CLK is always input and this bit is ignored. Change this bit in case of BIT_CLK is stopped (I2SCR.STPBK = 1). **BCKD** / **BIT_CLK Direction** 0 / BIT_CLK is input from an external source. 1 / BIT_CLK is generated internally and driven out to the CODEC.	RW
1	SYNCD	SYNC Direction. This bit specifies input/output direction of SYNC in I2S/MSB-justified format. When AC-link format is selected, SYNC is always output and this bit is ignored. Change this bit in case of BIT_CLK is stopped (I2SCR.STPBK = 1). **SYNCD** / **SYNC Direction** 0 / SYNC is input from an external source. 1 / SYNC is generated internally and driven out to the CODEC.	RW
0	ENB	Enable AIC function. This bit is used to enable or disable the AIC function. **ENB** / **Description** 0 / **Disable AIC Controller.** 1 / **Enable AIC Controller.**	RW

The BCKD bit (bit 2) and SYNCD bit (bit 1) configure the mode of I2S/MSB-justified interface. This is compliant with I2S specification.

BCKD	SYNCD	Description
0 (input)	**0 (input)**	**AIC roles the slave of I2S/MSB-justified interface.**
	1 (output)	**AIC roles the master with external serial clock source of I2S/MSB-justified interface.**
1 (output)	**0 (input)**	**Reserved.**
	1 (output)	**AIC roles the master of I2S/MSB-justified interface.**

26.2.2 AIC Common Control Register (AICCR)

AICCR contains bits to control DMA mode, FIFO flush, interrupt enable, internal loop-back, play back and recording enable. It also controls sample size and signed/unsigned data transfer.

AICCR **0x10020004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		EN2OLD	PACK16	Reserved		CHANNEL		Reserved		OSS			ISS			RDMS	TDMS	Reserved		M2S	ENDSW	ASVTSU	TFLUSH	RFLUSH	EROR	ETUR	ERFS	ETFS	ENLBF	ERPL	EREC
RST	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:30	Reserved	Writes to these bits have no effect and always read as 0.	R
29	Reserved	keep this value to 0 in normal use.	
28	PACK16	Output Sample data 16bit packed mode select. This bit reflects that one word contains two sample data or only one sample data with LSB align. The packed mode is only support 16bit sample size. **PACK16** / **Sample Size** 0 / Unpacked data mode. One word only contains one 16bit sample data aligned LSB. 1 / Packed data mode. One word contains two 16 bit sample data.	RW
27	Reserved	Writes to this bit have no effect and always read as 0.	R
26:24	CHANNEL	Output Channel Number Select. These bits reflect output data channels from AIC to device. The data supported are: 1(mono), 2(stereo), 4, 6 and 8 channels. The sample data is LSB-justified in memory/register. **CHANNEL** / **Sample Size** 0x0 / 1 channel, mono, Only SDATA0 used. 0x1 / 2 channels, stereo, Only SDATA0 used. 0x2 / Reserved. 0x3 / 4 channels, SDATA0 and SDATA1 used. 0x4 / Reserved. 0x5 / 6 channels, SDATA0 to SDATA2 used. 0x6 / Reserved. 0x7 / 8 channels, SDATA0 to SDATA3 used.	RW
23:22	Reserved	Writes to these bits have no effect and always read as 0.	R
21:19	OSS	Output Sample Size. These bits reflect output sample data size from	RW

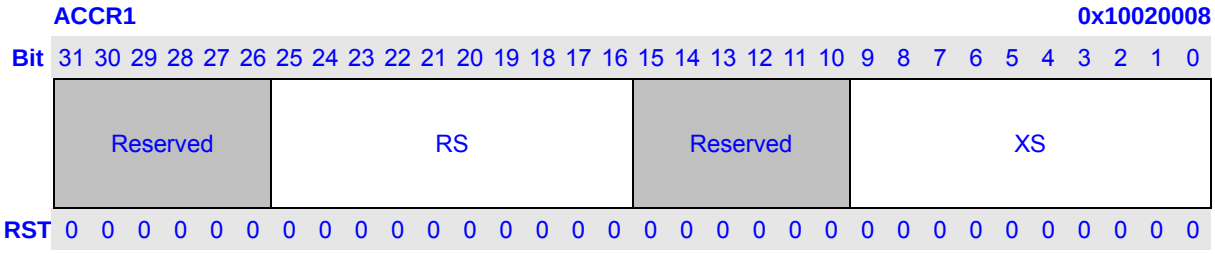
<table>
<tr><td></td><td></td><td>memory or register. The data sizes supported are: 8, 16, 18, 20 and 24 bits. The sample data is LSB-justified in memory/register.
<table>
<tr><th>OSS</th><th>Sample Size</th></tr>
<tr><td>0x0</td><td>8 bit.</td></tr>
<tr><td>0x1</td><td>16 bit.</td></tr>
<tr><td>0x2</td><td>18 bit.</td></tr>
<tr><td>0x3</td><td>20 bit.</td></tr>
<tr><td>0x4</td><td>24 bit.</td></tr>
<tr><td>0x5~0x7</td><td>Reserved.</td></tr>
</table></td><td></td></tr>
<tr><td>18:16</td><td>ISS</td><td>Input Sample Size. These bits reflect input sample data size to memory or register. The data sizes supported are: 8, 16, 18, 20 and 24 bits. The sample data is LSB-justified in memory/register.
<table>
<tr><th>ISS</th><th>Sample Size</th></tr>
<tr><td>0x0</td><td>8 bit.</td></tr>
<tr><td>0x1</td><td>16 bit.</td></tr>
<tr><td>0x2</td><td>18 bit.</td></tr>
<tr><td>0x3</td><td>20 bit.</td></tr>
<tr><td>0x4</td><td>24 bit.</td></tr>
<tr><td>0x5~0x7</td><td>Reserved.</td></tr>
</table></td><td>RW</td></tr>
<tr><td>15</td><td>RDMS</td><td>Receive DMA enable. This bit is used to enable or disable the DMA during receiving audio data.
<table>
<tr><th>RDMS</th><th>Receive DMA</th></tr>
<tr><td>0</td><td>Disabled.</td></tr>
<tr><td>1</td><td>Enabled.</td></tr>
</table></td><td>RW</td></tr>
<tr><td>14</td><td>TDMS</td><td>Transmit DMA enable. This bit is used to enable or disable the DMA during transmit audio data.
<table>
<tr><th>TDMS</th><th>Transmit DMA</th></tr>
<tr><td>0</td><td>Disabled.</td></tr>
<tr><td>1</td><td>Enabled.</td></tr>
</table></td><td>RW</td></tr>
<tr><td>13:12</td><td>Reserved</td><td>Writes to these bits have no effect and always read as 0.</td><td>R</td></tr>
<tr><td>11</td><td>M2S</td><td>Mono To Stereo. This bit control whether to do mono to stereo sample expansion in play back. When this bit is set, every outgoing sample data in the steam plays in both left and right channels. This bit should only be set in 2 channels configuration. It takes effective immediately when the bit is changed. Change this before replay started.
<table>
<tr><th>M2S</th><th>Description</th></tr>
<tr><td>0</td><td>No mono to stereo expansion.</td></tr>
<tr><td>1</td><td>Do mono to stereo expansion.</td></tr>
</table></td><td>RW</td></tr>
<tr><td>10</td><td>ENDSW</td><td>Endian Switch. This bit control endian change on outgoing 16-bits size audio sample by swapping high and low bytes in the sample data.</td><td>RW</td></tr>
</table>

			ENDSW / **Description** 0 / **No change on outgoing sample data.** 1 / **Swap high and low byte for outgoing 16-bits size sample data.**	
9	ASVTSU	Audio Sample Value Transfer between Signed and Unsigned data format. This bit is used to control the signed ←→unsigned data transfer. If it is 1, the incoming and outgoing audio sample data will be transferred by toggle its most significant bit. **ASVTSU** / **Description** 0 / **No audio sample value signed ←→unsigned transfer.** 1 / **Do audio sample value signed ←→unsigned transfer.**		RW
8	TFLUSH	Transmit FIFO Flush. Write 1 to this bit flush transmit FIFOs to empty. Writing 0 to this bit has no effect and this bit is always reading 0.		W
7	RFLUSH	Receive FIFO Flush. Write 1 to this bit flush receive FIFOs to empty. Writing 0 to this bit has no effect and this bit is always reading 0.		W
6	EROR	Enable ROR Interrupt. This bit is used to control the ROR interrupt enable or disable. **EROR** / **ROR Interrupt** 0 / **Disabled.** 1 / **Enabled.**		RW
5	ETUR	Enable TUR Interrupt. This bit is used to control the TUR interrupt enable or disable. **ETUR** / **TUR Interrupt** 0 / **Disabled.** 1 / **Enabled.**		RW
4	ERFS	Enable RFS Interrupt. This bit is used to control the RFS interrupt enable or disable. **ERFS** / **RFS Interrupt** 0 / **Disabled.** 1 / **Enabled.**		RW
3	ETFS	Enable TFS Interrupt. This bit is used to control the TFS interrupt enable or disable. **ETFS** / **TFS Interrupt** 0 / **Disabled.** 1 / **Enabled.**		RW
2	ENLBF	Enable AIC Loop Back Function. This bit is used to enable or disable the internal loop back function of AIC, which is used for test only. When the AIC loop back function is enabled, normal audio replay/record functions are disabled.		RW

			ENLBF	**Description**
			0	AIC Loop Back Function is Disabled.
			1	AIC Loop Back Function is Enabled.
1	ERPL	Enable Playing Back function. This bit is used to disable or enable the audio sample data transmitting.		
			ERPL	**Description**
			0	**AIC Playing Back Function is Disabled.**
			1	**AIC Playing Back Function is Enabled.**
0	EREC	Enable Recording Function. This bit is used to disable or enable the audio sample data receiving.		
			EREC	**Description**
			0	**AIC Recording Function is Disabled.**
			1	**AIC Recording Function is Enabled.**

26.2.3 AIC AC-link Control Register 1 (ACCR1)

ACCR1 contains bits to reflect/control valid incoming/outgoing slots of AC97. It is used only in AC-link format.



Bits	Name	Description	RW
31:26	Reserved	Writes to these bits have no effect and always read as 0.	R
25:16	RS	Receive Valid Slots. These bits are used to indicate which incoming slots are valid. Slot 3 is mapped to bit 16 or RS[0], slot 4 to bit 17 or RS[1] and so on. When write to this field, a bit 1 means we expect a PCM data in the corresponding slot, a bit 0 means the corresponding slot PCM data will be discarded. When read from this field, a bit 1 means we receive an expected valid PCM data in the corresponding slot. This field should be written before record started. **RS[n] Value** / **Description** 0 / **Slot n+3 is invalid.** 1 / **Slot n+3 has valid PCM data.**	RW
15:10	Reserved	Writes to these bits have no effect and always read as 0.	R
9:0	XS	Transmit Valid Slots. These bits making up slots map to the valid bits in the AC'97 tag (slot 0 on SDATA_OUT) and indicate which outgoing slots have valid PCM data. Bit 0 or XS[0] maps to slot 3, bit 1 or XS[1] to slot 4 and so on. Setting the corresponding bit indicates to AIC to take an audio sample from transmit FIFO to fill the respective slot. And it indicates to the CODEC that valid PCM data will be in the respective slot. The number of valid bits will designate how many words will be pulled out of the FIFO per audio frame. This field should be written before record and replay started. **XS[n] Value** / **Description** 0 / **Slot n+3 is invalid.** 1 / **Slot n+3 has valid PCM data.**	RW

26.2.4 AIC AC-link Control Register 2 (ACCR2)

ACCR2 contains bits to control interrupt enable, output/input sample size, and alternative control of RESET#, SYNC and SDATA_OUT pins in AC-link. It is valid only in AC-link format.

ACCR2 **0x1002000C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved													ERSTO	ESADR	ECADT	Reserved												SO	SR	SS	SA
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:19	Reserved	Writes to these bits have no effect and always read as 0.	R
18	ERSTO	Enable RSTO Interrupt. This bit is used to control the RSTO interrupt enable or disable. **ERSTO** / **RSTO Interrupt** 0 / **Disabled.** 1 / **Enabled.**	RW
17	ESADR	Enable SADR Interrupt. This bit is used to control the SADR interrupt enable or disable. **ESADR** / **SADR Interrupt** 0 / **Disabled.** 1 / **Enabled.**	RW
16	ECADT	Enable CADT Interrupt. This bit is used to control the CADT interrupt enable or disable. **ECADT** / **CADT Interrupt** 0 / **Disabled.** 1 / **Enabled.**	RW
15:4	Reserved	Writes to these bits have no effect and always read as 0.	R
3	SO	SDATA_OUT output value. When SA is 1, this bit controls SDATA_OUT pin voltage level, 0 for low, 1 for high; otherwise, it is ignored.	RW
2	SR	RESET# pin level. When AC-link is selected, this bit is used to drive the RESET# pin. **SR** / **RESET# Pin Voltage Level** 0 / High. 1 / Low.	RW
1	SS	SYNC value. When this bit is read, it returns the actual value of SYNC. When SA is 1, write value controls SYNC pin value. When SA is 0, write to it is ignored.	RW
0	SA	SYNC and SDATA_OUT Alternation. This bit is used to determine the driven signal of SYNC and SDATA_OUT. When SA is 0, SYNC and SDATA_OUT being driven AIC function logic; otherwise, SYNC is	RW

controlled by the SS and SDATA_OUT is controlled by the SO. The true table of SYNC is described in following.

SA	SS	Description
0	0	When read, indicated SYNC is 0.
		When write, not effect.
	1	When read, indicated SYNC is 1.
		When write, not effect.
1	0	When read, indicated SYNC is 0.
		When write, SYNC is driven to 0.
	1	When read, indicated SYNC is 1.
		When write, SYNC is driven to 1.

26.2.5 AIC I2S/MSB-justified Control Register (I2SCR)

I2SCR contains bits to control BIT_CLK stop, audio sample size, I2S or MSB-justified selection in I2S/MSB-justified. It is valid only in I2S/MSB-justified format.

I2SCR 0x10020010

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														RFIRST	SWLH	Reserved			STPBK	Reserved							ESCLK	Reserved			AMSL
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:18	Reserved	Writes to these bits have no effect and always read as 0.	R
17	RFIRST	Send R channel first in stereo mode. This bit should only be set in 2 channels configuration. The frame is LR like or RL like. It takes effective immediately when the bit is changed. Change this before replay started. **RFIRST / Description** 0 – **Send L channel first (LR).** 1 – **Send R channel first (RL).**	RW
16	SWLH	Switch LR channel in 16bit-packed stereo mode. This bit control whether the low address data is L or R. This bit should only be set in 2 channels configuration and 16bit-packed mode. That means it can only valid with packed mode (PACK16 =1) and 2 channels (CHANNEL=0x1). It takes effective immediately when the bit is changed. Change this before replay started. **SWLH / Description** 0 – **16 bit LSB and 16bit MSB is not switched.** 1 – **16 bit LSB and 16bit MSB is switched.**	RW
15:13	Reserved	Writes to these bits have no effect and always read as 0.	R
12	STPBK	Stop BIT_CLK. It is used to stop the BIT_CLK in I2S/MSB-justified format. When AC-link is selected, all of its operations are ignored. **STPBK / Description** 0 – **BIT_CLK is not stopped.** 1 – **BIT_CLK is stopped.** Please set this bit to 1 to stop BIT_CLK when change AICFR.AUSEL and AICFR.BCKD.	RW

<table>
<tr><td>11:5</td><td>Reserved</td><td>Writes to these bits have no effect and always read as 0.</td><td>R</td></tr>
<tr><td>4</td><td>ESCLK</td><td>Enable SYSCLK output. When this bit is 1, the SYSCLK outputs to chip outside is enabled. Else, the clock is disabled.</td><td>RW</td></tr>
<tr><td>3:1</td><td>Reserved</td><td>Writes to these bits have no effect and always read as 0.</td><td>R</td></tr>
<tr><td>0</td><td>AMSL</td><td>Specify Alternate Mode (I2S or MSB-Justified) Operation.
<table>
<tr><th>AMSL</th><th>Description</th></tr>
<tr><td>0</td><td>Select I2S Operation Mode.</td></tr>
<tr><td>1</td><td>Select MSB-Justified Operation Mode.</td></tr>
</table></td><td>RW</td></tr>
</table>

26.2.6 AIC Controller FIFO Status Register (AICSR)

AICSR contains bits to reflect FIFOs status. Most of the bits are read-only except two, which can be written a 0.

AICSR 0x10020014

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				RFL				Reserved										TFL						Reserved	ROR	TUR	RFS	TFS	Reserved		
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:30	Reserved	Writes to these bits have no effect and always read as 0.	R
29:24	RFL	Receive FIFO Level. The bits indicate the amount of valid PCM data in Receive FIFO. RFL Value: 0x00 ~ 0x20 — RFL valid PCM data in receive FIFO. RFL Value: 0x21 ~ 0x3F — Reserved.	R
23:14	Reserved	Writes to these bits have no effect and always read as 0.	R
13:8	TFL	Transmit FIFO Level. The bits indicate the amount of valid PCM data in Transmit FIFO. TFL Value: 0x00 ~ 0x20 — TFL valid PCM data in transmit FIFO. TFL Value: 0x21 ~ 0x3F — Reserved.	R
7	Reserved	Writes to these bits have no effect and always read as 0.	R
6	ROR	Receive FIFO Over Run. This bit indicates that receive FIFO has or has not experienced an overrun. ROR 0: When read, indicates over-run has not been found. / When write, clear itself. ROR 1: When read, indicates data has even been written to full receive FIFO. / When write, not effects.	RW
5	TUR	Transmit FIFO Under Run. This bit indicates that transmit FIFO has or has not experienced an under-run. TUR 0: When read, indicates under-run has not been found. / When write, clear itself. TUR 1: When read, indicates data has even been read from empty transmit FIFO. / When write, not effects.	RW
4	RFS	Receive FIFO Service Request. This bit indicates that receive FIFO level	R

<table>
<tr><td></td><td></td><td>is or not below receive FIFO threshold, which is controlled by AICFR.RFTH. When RFS is 1, it may trigger interrupt or DMA request depends on the interrupt enable and DMA setting.
<table>
<tr><th>RFS</th><th>Description</th></tr>
<tr><td>0</td><td>Receive FIFO level below RFL threshold.</td></tr>
<tr><td>1</td><td>Receive FIFO level at or above RFL threshold.</td></tr>
</table></td><td></td></tr>
<tr><td>3</td><td>TFS</td><td>Transmit FIFO Service Request. This bit indicates that transmit FIFO level is below Transmit FIFO threshold, which is controlled by AICFR.TFTH. When TFS is 1, it may trigger interrupt or DMA request depends on the interrupt enable and DMA setting.
<table>
<tr><th>TFS</th><th>Description</th></tr>
<tr><td>0</td><td>Transmit FIFO level exceeds TFL threshold.</td></tr>
<tr><td>1</td><td>Transmit FIFO level at or below TFL threshold.</td></tr>
</table></td><td>R</td></tr>
<tr><td>2:0</td><td>Reserved</td><td>Writes to these bits have no effect and always read as 0.</td><td>R</td></tr>
</table>

26.2.7 AIC AC-link Status Register (ACSR)

ACSR contains bits to reflect the status of the connected external CODEC in AC-link format. Bits in this register are read-only in general, except some of them can be written a 0.

ACSR **0x10020018**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved										SLTERR	CRDY	CLPM	RSTO	SADR	CADT	Reserved															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:22	Reserved	Writes to these bits have no effect and always read as 0.	R
21	SLTERR	Hardware detects a Slot Error. This bit indicates an error in SLOTREQ bits on incoming data from external CODEC is detected. The error can be: (1) find 1 in a SLOTREQ bit, which corresponding to an inactive slot; (2) all active slots should be request in the same time by SLOTREQ, but an exception is found. All errors are accumulated to ACSR.SLTERR by hardware until software clears it. Software writes 0 clear this bit and write 1 has no effect.	RW
20	CRDY	External CODEC Ready. This bit is derived from the CODEC Ready bit of Slot 0 in SDATA_IN, and it indicates the external AC97 CODEC is ready or not. **CRDY** / **Description** 0 / CODEC is not ready. 1 / CODEC is ready.	R
19	CLPM	External CODEC Low Power Mode. This bit indicates the external CODEC is switched to low power mode or BIT_CLK is active from CODEC after wake up. **CLPM** / **Description** 0 / BIT_CLK is active. 1 / CODEC is switched to low power mode.	R
18	RSTO	External CODEC Registers Read Status Time Out. This bit indicates that the read status time out is detected or not. It is set to 1 if the data not return in 4 frames after a CODEC registers read command issued. **RSTO** / **Description** 0 / When read, indicates time out has not occurred. 1 / When read, indicates read status time out found. Write 0 clear this bit and write 1 is ignored. When RSTO is 1, it may trigger an interrupt depends on the interrupt enable setting.	RW
17	SADR	External CODEC Registers Status Address and Data Received. This bit indicates that address and data of an external AC '97 CODEC register	RW

<table>
<tr><td></td><td></td><td>has or has not been received.<table><tr><th>SADR</th><th>Description</th></tr><tr><td>0</td><td>When read, indicates no register address/data received.</td></tr><tr><td>1</td><td>When read, indicates address/data received.</td></tr></table>Write 0 clear this bit and write 1 is ignored. When SADR is 1, it may trigger an interrupt depends on the interrupt enable setting.</td><td></td></tr>
<tr><td>16</td><td>CADT</td><td>Command Address and Data Transmitted. This bit indicates that a CODEC register reading/writing command transmission has completed or not.<table><tr><th>CADT</th><th>Description</th></tr><tr><td>0</td><td>When read, indicates the command has not done.</td></tr><tr><td>1</td><td>When read, indicates the command has done.</td></tr></table>Write 0 clear this bit and write 1 is ignored. When CADT is 1, it may trigger an interrupt depends on the interrupt enable setting.</td><td>RW</td></tr>
<tr><td>15:0</td><td>Reserved</td><td>Writes to these bits have no effect and always read as 0.</td><td>R</td></tr>
</table>

26.2.8 AIC I2S/MSB-justified Status Register (I2SSR)

I2SSR is used to reflect AIC status in I2S/MSB-justified. It is a read-only register.

I2SSR 0x1002001c

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																										CHBSY	TBSY	RBSY	BSY	Reserved	
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:3	Reserved	Writes to these bits have no effect and always read as 0.	R
5	CHBSY	AIC Transmitter busy in I2S/MSB-justified format.(Multi-channel status) CHBSY: 0 — AIC Transmitter part is idle or disabled. CHBSY: 1 — AIC Transmitter part currently is transmitting or receiving a frame.	R
4	TBSY	AIC Transmitter busy in I2S/MSB-justified format. TBSY: 0 — AIC Transmitter part is idle or disabled. TBSY: 1 — AIC Transmitter part currently is transmitting or receiving a frame.	R
3	RBSY	AIC Receiver busy in I2S/MSB-justified format. RBSY: 0 — AIC Receiver part is idle or disabled. RBSY: 1 — AIC Receiver part currently is transmitting or receiving a frame.	R
2	BSY	AIC busy in I2S/MSB-justified format. BSY: 0 — AIC controller is idle or disabled. BSY: 1 — AIC controller currently is transmitting or receiving a frame.	R
1:0	Reserved	Writes to these bits have no effect and always read as 0.	R

26.2.9 AIC AC97 CODEC Command Address & Data Register (ACCAR, ACCDR)

ACCAR and ACCDR are used to hold register address and data for external AC-link CODEC register read/write operation through SDATA_OUT. The format of ACCAR.CAR and ACCDR.CDR is compliant with AC'97 Component Specification 2.3 where ACCAR.CAR[19] of "1" specifies CODEC register read operation, of "0" specifies CODEC register write operation. The write access to ACCAR and ACCDR signals AIC to issue this operation. Please reference to _0 for software flow. These registers are valid only in AC-link. It is ignored in I2S/MSB-justified format.

ACCAR **0x10020020**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												CAR																			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:20	Reserved	Writes to these bits have no effect and always read as 0.	R
19:0	CAR	Command Address Register. This is used to hold 20-bit AC '97 CODEC register address transmitted in SDATA_OUT slot 1. After this field is write, it should not be write again until the operation is finished.	RW

ACCDR **0x10020024**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												CDR																			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:20	Reserved	Writes to these bits have no effect and always read as 0.	R
19:0	CDR	Command Data Register. This is used to hold 20-bit AC'97 CODEC register data transmitted in SDATA_OUT slot 2. After this field is write, it should not be write again until the operation is finished.	RW

26.2.10 AIC AC97 CODEC Status Address & Data Register (ACSAR, ACSDR)

ACSAR and ACSDR are used to receive the external AC-link CODEC registers address and data from SDATA_IN. When AIC receives CODEC register status from SDATA_IN, it set ACSR.SADR bit and put the address and data to ACSAR.SAR and ACSDR.SDR. Please reference to _0 for software flow. These registers are valid only in AC-link format and are ignored in I2S/MSB-justified format.

ACSAR **0x10020028**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												SAR																			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:20	Reserved	Writes to these bits have no effect and always read as 0.	R
19:0	SAR	CODEC Status Address Register. This is used to receive 20-bit AC ’97 CODEC status address from SDATA_IN slot 1. Which reflect the register index for which data is being returned? The write operation is ignored.	R

ACSDR **0x1002002C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												SDR																			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:20	Reserved	Writes to these bits have no effect and always read as 0.	R
19:0	SDR	CODEC Status Data Register. This is used to receive 20-bit AC ’97 CODEC status data from SDATA_IN slot 2. The register data of external CODEC is returned. The write operation is ignored.	R

26.2.11 AIC I2S/MSB-justified Clock Divider Register (I2SDIV)

I2SDIV is used to set clock divider to generated BIT_CLK from SYS_CLK in I2S/MSB-justified format.

I2SDIV **0x10020030**

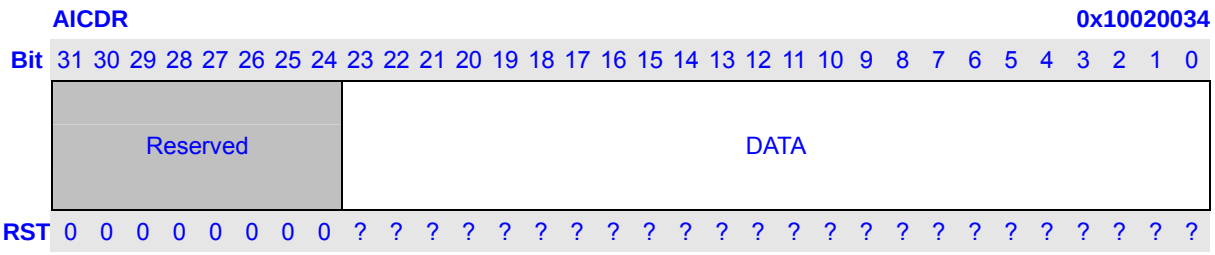
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																												DV			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Bits	Name	Description	RW
31:4	Reserved	Writes to these bits have no effect and always read as 0.	R
3:0	DV	Audio BIT_CLK clock divider value minus 1. I2SDIV.DV is used to control the generating of BIT_CLK from dividing SYS_CLK. The dividing value should be even and I2SDIV.DV should be set to the dividing value minus 1. So I2SDIV.DV bit0 is fixed to 1. BIT_CLK frequency is fixed to 64 f_S in AIC, where f_S is the audio sample frequency. I2SDIV.DV depends on SYS_CLK frequency f_{SYS_CLK}, which is selected according to external CODEC's requirement and internal PLL frequency. Please reference to 1.4.10 Serial Audio Clocks and Sampling Frequencies for further description.	RW

26.2.12 AIC FIFO Data Port Register (AICDR)

AICDR is act as data input port to transmit FIFO when write and data output port from receive FIFO when read, one audio sample every time. The FIFO width is 24 bits. Audio sample with size N that is less than 24 is located in LSB N-bits. The sample size is specified by ACCR2.OASS and ACCR2.IASS in AC-link, and by I2SCR.WL in I2S/MSB-justified. The sample order is specified by ACCR1.XS and ACCR1.RS in AC-link. In I2S/MSB-justified, the left channel sample is prior to the right channel sample.

Care should be taken to monitor the status register to insure that there is room for data in the FIFO when executing a program read or write transaction. This is taken care automatically in DMA.

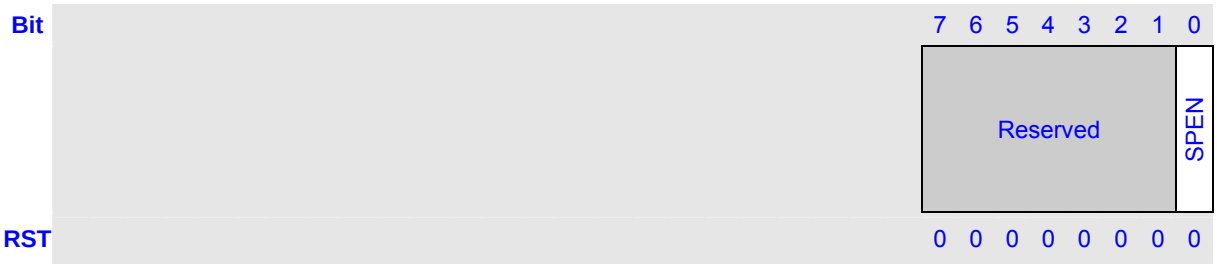


Bits	Name	Description	RW
31:24	Reserved	Writes to these bits have no effect and always read as 0.	R
23:0	DATA	FIFO port. When write to it, data is push to the transmit FIFO. When read from it, data is pop from the receiving FIFO.	RW

26.2.13 SPDIF Enable Register (SPENA)

The register SPENA is used to trigger SPDIF transmitter to work.

SPENA 0x10020080



Bits	Name	Description	RW
7:1	Reserved	These bits always read 0, and writing operations are ignored.	R
0	SPEN	Enable / disable the SPDIF transmitter. 0: SPDIF transmitter is disabled 1: SPDIF transmitter is enabled	RW

26.2.14 SPDIF Control Register (SPCTRL)

The register SPCTRL is used to control SPDIF to work.

SPCTRL **0x10020084**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DMA_EN	D_TYPE	SIGN_N	INVALID	SFT_RST	SPDIF_I2S	Reserved								M_TRIG	M_FFUR
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

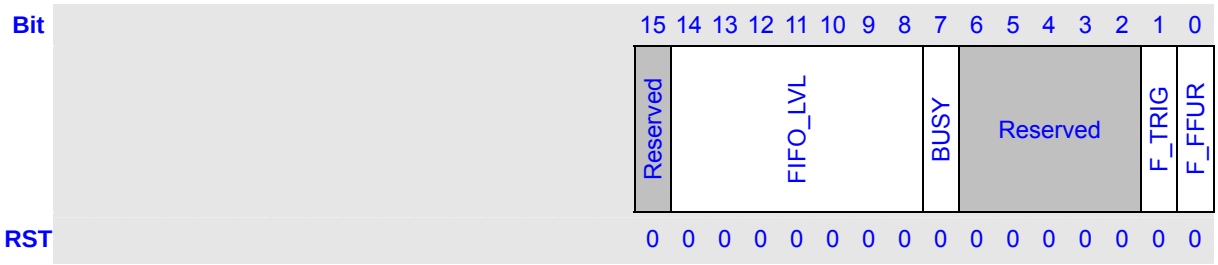
Bits	Name	Description	RW
15	DMA_EN	DMA transmitter enable bit. 0: DMA transmitter disable 1: DMA transmitter enable	RW
14	D_TYPE	If the bit number of data is less than16, the data in memory is as follows: 0: XXXXXXXXXXXXXXXX Data 0 XXXXXXXXXXXXXXXX Data 1 1: Data 1 Data 0 Data 3 Data 2	RW
13	SIGN_N	Signed to unsigned or not. If it is 1, the incoming and outgoing audio sample data will be transferred by toggle its most significant bit. 0: Not transfer 1: Do transfer	RW
12	INVALID	Data invalid bit. The data transmitted on SPDIF is valid or not. 0: Valid 1: Invalid	RW
11	SFT_RST	SPDIF FIFO software-reset. Set it to 1 and later it will be cleared by hardware auto. When SFT_RST returns back to 0, the FIFO finish reset. 0: Stop reset 1: Start reset	RW
10	SPDIF_I2S	Choose SPDIF or I2S. 0: I2S 1: SPDIF	
9:2	Reserved	These bits always read 0, and writing operations are ignored.	R
1	M_TRIG	Trigger interrupt mask. 0: Enabled 1: Masked	RW

0	M_FFUR	FIFO underrun interrupt mask. 0: Enabled 1: Masked	RW

26.2.15 SPDIF State Register (SPSTATE)

The register SPSTATE is used to keep the state of SPDIF.

SPSTATE **0x10020088**



Bits	Name	Description	RW
15	Reserved	These bits always read 0, and writing operations are ignored.	R
14:8	FIFO_LVL	FIFO level. The bits indicate the amount of valid data in FIFO.	R
7	BUSY	SPDIF busy bit. 0: SPDIF is not working. 1: SPDIF is working.	R
6:2	Reserved	These bits always read 0, and writing operations are ignored.	R
1	F_TRIG	Trigger flag. 0: Not active 1: Active	R
0	F_FFUR	FIFO underrun flag. 0: Not active 1: Active	RW

26.2.16 SPDIF Configure 1 Register (SPCFG1)

The register SPCFG1 is used to configure SPDIF.

SPCFG1 **0x1002008C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved														INIT_LVL	ZRO_VLD	Reserved		TRIG		SRC_NUM				CH1_NUM				CH2_NUMI			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:18	Reserved	These bits always read 0, and writing operations are ignored.	R
17	INIT_LVL	Initial level set bit. 0: SPDIF initial level is low. 1: SPDIF initial level is high.	
16	ZRO_VLD	The valid bit of channel state is 0 or 1 when play ZERO sample under FIFO underflow. 0: Valid 1: Invalid	RW
15:14	Reserved	These bits always read 0, and writing operations are ignored.	R
13:12	TRIG	Specify the trigger value of FIFO.	RW
11:8	SRC_NUM	Source number. 0000:Unspecified 0001~1111:1~15	RW
7:4	CH1_NUM	Channel 1 number. 0000:Unspecified 0001~1111:A~O	RW
3:0	CH2_NUM	Channel 2 number. 0000:Unspecified 0001~1111:A~O	RW

TRIG	Description
00	Trigger Value is 4
01	Trigger Value is 8
10	Trigger Value is 16
11	Trigger Value is 32

26.2.17 SPDIF Configure 2 Register (SPCFG2)

The register SPCFG2 is used to configure SPDIF.

SPCFG2 0x10020090

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved		FS				ORG_FRQ				SAMPL_WL			MAX_WL	CLK_ACU		CAT_CODE								CH_MD		Reserved		PRE	COPY_N	AUDIO_N	CON_PRO
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:30	Reserved	These bits always read 0, and writing operations are ignored.	R
29:26	FS	Sampling frequency. 0000:44.1kHz 0010:48kHz 0011:32kHz 1010:96kHz 1110:192kHz Others: Reference IEC60958-3	RW
25:22	ORG_FRQ	Original sampling frequency. 1111:44.1kHz 1101:48kHz 1100:32kHz 0101:96kHz 0001:192kHz Others: Reference IEC60958-3	RW
21:19	SAMPL_WL	Sample word length. When MAX_WL=1 001:20 bit 110:21 bit 010:22 bit 100:23 bit 101:24 bit Others: reserved When MAX_WL=0 001:16 bit 110:17 bit 010:18 bit 100:19 bit 101:20 bit Others: reserved	RW

18	MAX_WL	Maximum audio sample word length. 0:20 bit 1:24 bit	RW
17:16	CLK_ACU	Clock Accuracy of transmitted clock. 00: Level II 01: Level I 10: Level III 11: Interface frame rate not matched to sampling frequency	RW
15:8	CAT_CODE	Category code. Reference IEC60958-3 for full details. 00 indicates “general” mode.	RW
7:6	CH_MD	Channel mode choose bit. 00: Mode 0 01~11: Reserved	RW
5:4	Reserved	These bits always read 0, and writing operations are ignored.	R
3	PRE	Pre-emphasis set bit. 0: None 1: 15us/15us	RW
2	COPY_N	Copyright set bit. 0: Copyright is asserted 1: Copyright is not asserted	RW
1	AUDIO_N	Linear PCM identification bit. 0: Audio sample word represents linear PCM samples 1: Audio sample word used for other purpose	RW
0	CON_PRO	Consumer mode and professional mode choose bit. 0: Consumer mode 1: Professional mode Professional is not supported in the chip.	RW

26.2.18 SPDIF FIFO Register (SPFIFO)

The register SPCFG1 is used to configure SPDIF.

SPFIFO **0x10020094**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								DATA																							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:24	Reserved	These bits always read 0, and writing operations are ignored.	R
23:0	DATA	FIFO port. When write to it, data is push to the transmit FIFO. Read from it as 0.	W

26.3 Serial Interface Protocol

26.3.1 AC-link serial data format

Following figures are AC-link serial data format. Audio data is MSB adjusted, regardless of 8, 16, 18, 20, 24 bits sample size. When a 24-bits sample is transmitted, the LSB 4-bits are truncated. When try to record 24-bits sample, 4-bits of 0 are appended in LSB. Please reference to “AC ’97 Component Specification Revision 2.3, 2002”, provided by Intel Corporation, for details of AC ’97 architecture and AC-link specification.

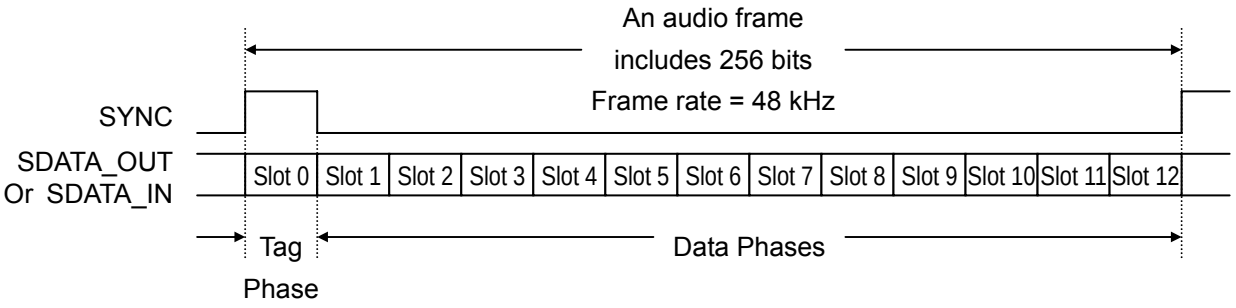


Figure 26-8 AC-link audio frame format

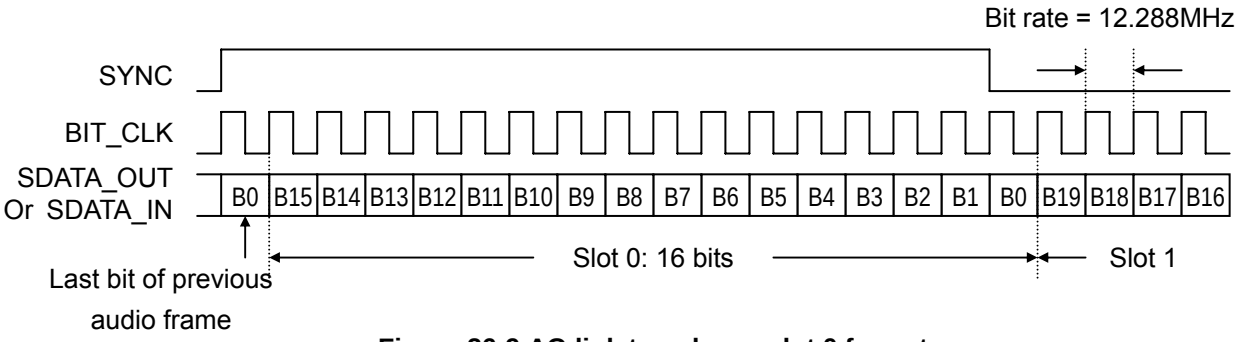


Figure 26-9 AC-link tag phase, slot 0 format

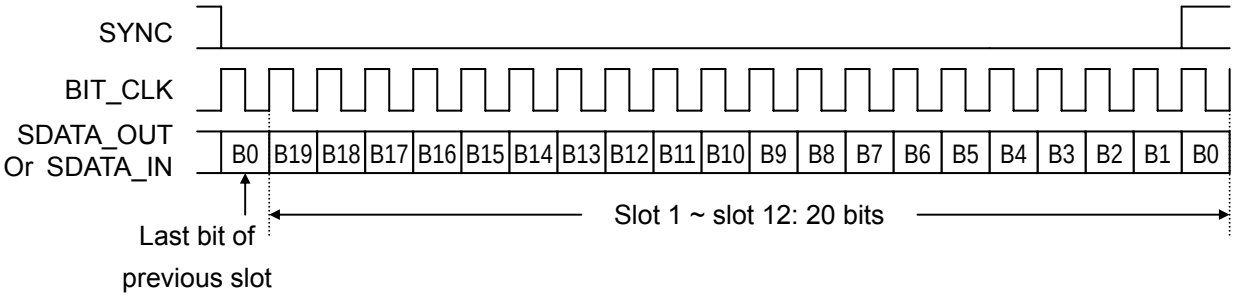


Figure 26-10 AC-link data phases, slot 1 ~ slot 12 format

26.3.2 I2S and MSB-justified serial audio format

Normal I2S and MSB-justified are similar protocols for digitized stereo audio transmitted over a serial path.

The BIT_CLK supplies the serial audio bit rate, the basis for the external CODEC bit-sampling logic. Its frequency is 64 times the audio sampling frequency. Divided by 64, the resulting 8 kHz to 48 kHz or even higher signal signifies timing for left and right serial data samples passing on the serial data paths. This left/right signal is sent to the CODEC on the SYNC pin. Each phase of the left/right signal is accompanied by one serial audio data sample on the data pins SDATA_IN and SDATA_OUT.

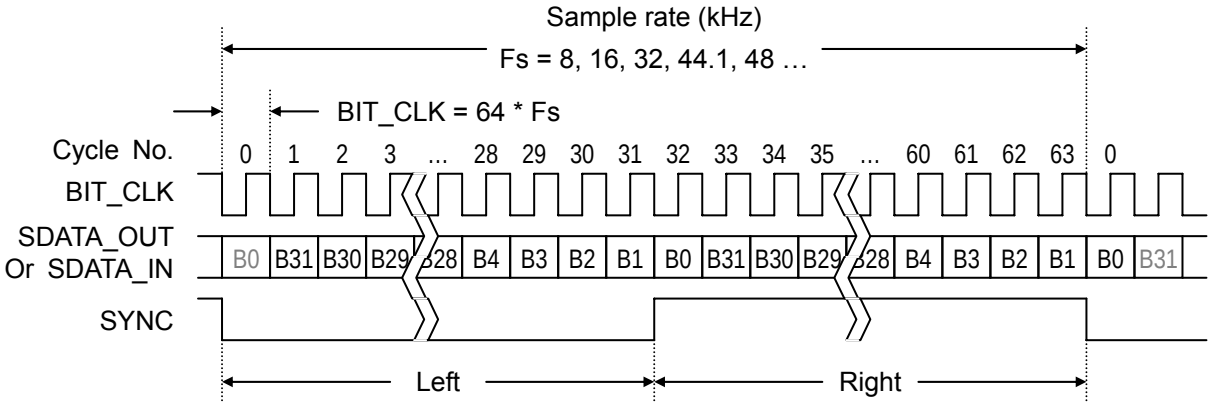


Figure 26-11 I2S data format (A: LR mode)

In the A: LR mode, first send the left channel in a stereo frame. One Left slot and one Right slot make a sample frame. It is the normal mode of I2S.

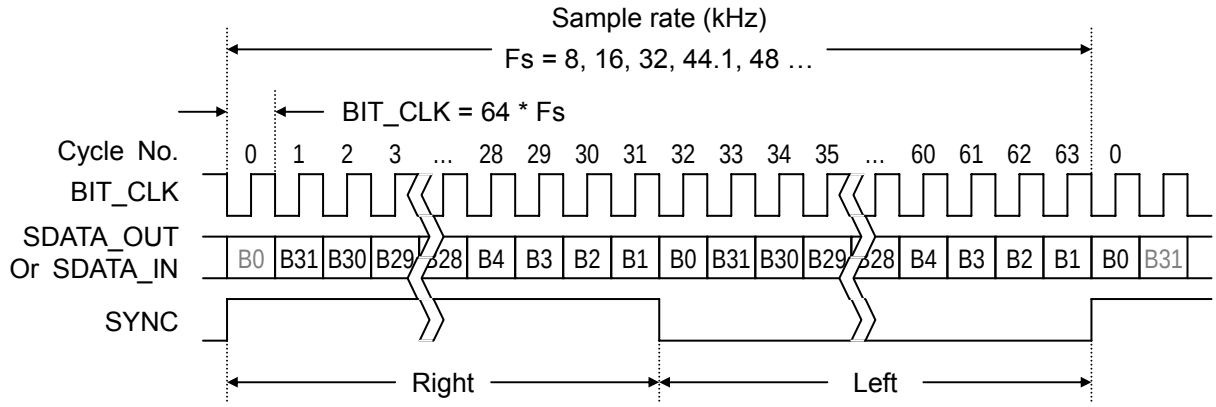


Figure 26-12 I2S data format (B: RL mode)

In the B: RL mode, first send the right channel in a stereo frame. One Right slot and one Left slot make a sample frame. It is used in same CODEC.

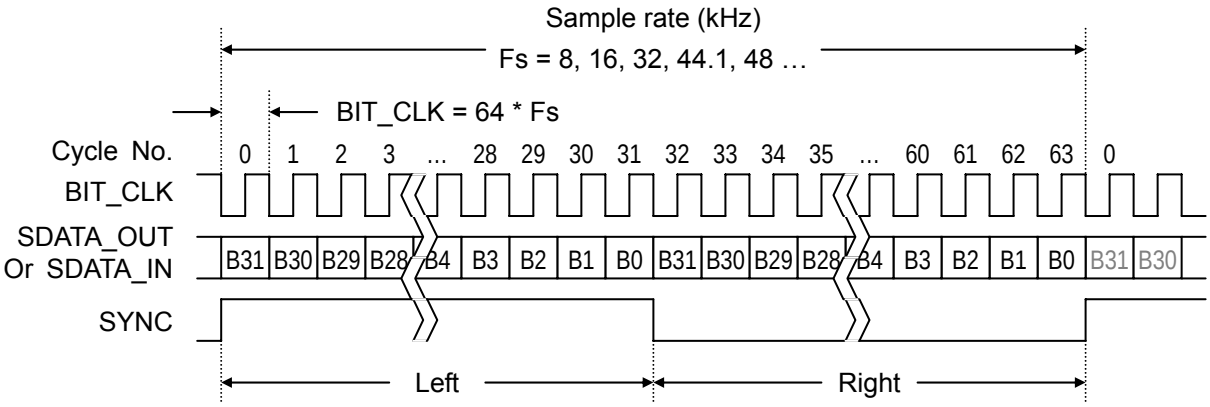


Figure 26-13 MSB-justified data format (C: LR mode)

In the C: LR mode, first send the left channel in a stereo frame. One Left slot and one Right slot make a sample frame. It is the normal mode in MSB-justified.

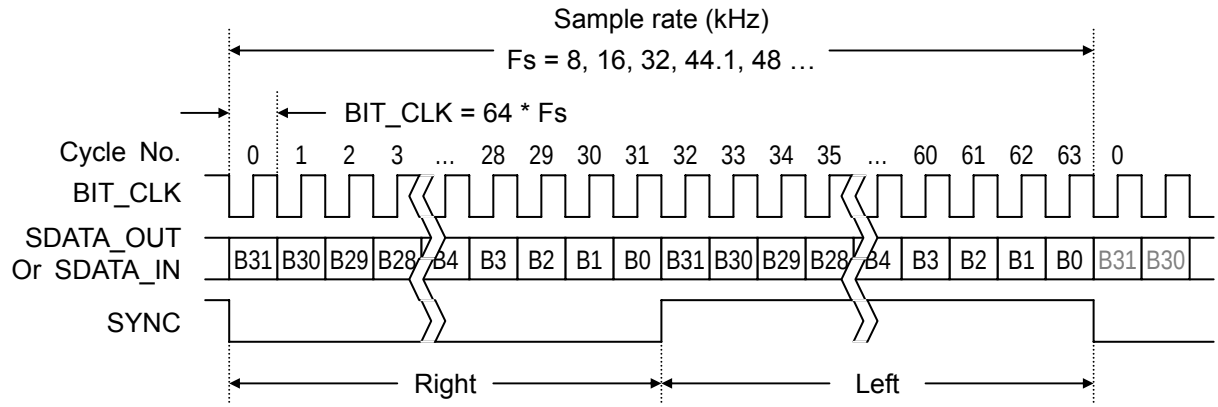


Figure 26-14 MSB-justified data format (D: RL mode)

In the D: RL mode, first send the right channel in a stereo frame. One Right slot and one Left slot make a sample frame.

Figure 26-11 and Figure 26-13 provide timing diagrams that show formats for the normal I2S and MSB-justified modes of operations. Data is sampled on the rising edge of the BIT_CLK and data is sent out on the falling edge of the BIT_CLK.

Data is transmitted and received in frames of 64 BIT_CLK cycles(If BIT_CLK is generated internally). Each frame consists of a left sample and a right sample. Each sample holds 8, 16, 18, 20 or 24 bits of valid data. The LSB other bits of each sample is padded with zeroes.

In the normal I2S mode, the SYNC is low for the left sample and high for the right sample. Also, the MSB of each data sample lags behind the SYNC edges by one BIT_CLK cycle.

In the MSB-justified mode, the SYNC is high for the left sample and low for the right sample. Also, the MSB of each data sample is aligned with the SYNC edges.

When use with the internal CODEC, the BIT_CLK and SYNC signals also with O_BIT_CLK and O_SYNC signals are provided by the internal CODEC from the SYSCLK, which is enabled by I2SCR.ESCLK and configured to 12MHz clock using CPM.

26.3.3 Audio sample data placement in SDATA_IN/SDATA_OUT

The placement of audio sample in incoming/outgoing serial data stream for all formats support in AIC is MSB (Most Significant Bit) justified. Suppose n bit sample composed by

S_{n-1}	S_{n-2}	…	S_2	S_1	S_0

Table 26-3 described the how sample data bits are transferred.

Table 26-3 Sample data bit relate to SDATA_IN/SDATA_OUT bit

AC-link Format						I2S/MSB-Justified Format					
SDATA IN/OUT	Audio Sample Size (bit)					SDATA IN/OUT					
	8	16	18	20	24		8	16	18	20	24
B19	S7	S15	S17	S19	S23	B31	S7	S15	S17	S19	S23
B18	S6	S14	S16	S18	S22	B30	S6	S14	S16	S18	S22
B17	S5	S13	S15	S17	S21	B29	S5	S13	S15	S17	S21
B16	S4	S12	S14	S16	S20	B28	S4	S12	S14	S16	S20
B15	S3	S11	S13	S15	S19	B27	S3	S11	S13	S15	S19
B14	S2	S10	S12	S14	S18	B26	S2	S10	S12	S14	S18
B13	S1	S9	S11	S13	S17	B25	S1	S9	S11	S13	S17
B12	S0	S8	S10	S12	S16	B24	S0	S8	S10	S12	S16
B11	0	S7	S9	S11	S15	B23	0	S7	S9	S11	S15
B10	0	S6	S8	S10	S14	B22	0	S6	S8	S10	S14
B9	0	S5	S7	S9	S13	B21	0	S5	S7	S9	S13
B8	0	S4	S6	S8	S12	B20	0	S4	S6	S8	S12
B7	0	S3	S5	S7	S11	B19	0	S3	S5	S7	S11
B6	0	S2	S4	S6	S10	B18	0	S2	S4	S6	S10
B5	0	S1	S3	S5	S9	B17	0	S1	S3	S5	S9
B4	0	S0	S2	S4	S8	B16	0	S0	S2	S4	S8
B3	0	0	S1	S3	S7	B15	0	0	S1	S3	S7
B2	0	0	S0	S2	S6	B14	0	0	S0	S2	S6
B1	0	0	0	S1	S5	B13	0	0	0	S1	S5
B0	0	0	0	S0	S4	B12	0	0	0	S0	S4
						B11	0	0	0	0	S3
						B10	0	0	0	0	S2
						B9	0	0	0	0	S1
						B8	0	0	0	0	S0
						B7~ B0	0	0	0	0	0

If in 16 bits packed mode, the data transferred is the same as the 16 bits normal mode as shown above. But there are two samples in one word.

26.3.4 SPDIF Protocol

SPDIF block format is shown below:

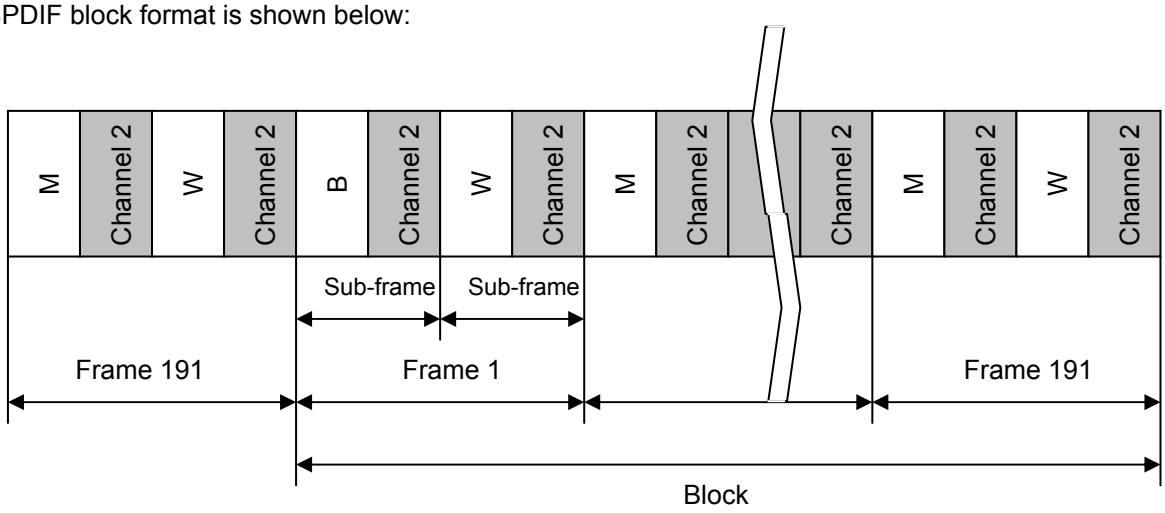


Figure 26-15 Block format

Sub-frame format in PCM mode is shown below:

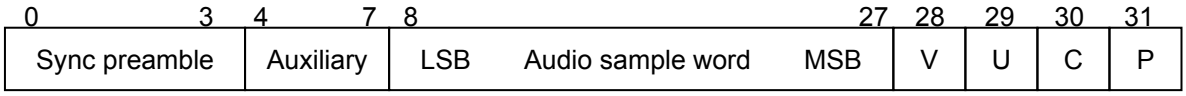


Figure 26-16 Sub-frame format in PCM mode

Sub-frame format in non-PCM mode is shown below:

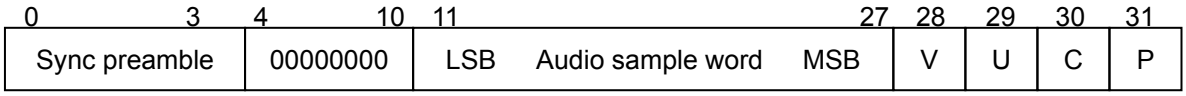


Figure 26-17 Sub-frame format in non-PCM mode

26.4 AC97/I2S Operation

The AIC can be accessed either by the processor using programmed I/O instructions or by the DMA controller. The processor uses programmed I/O instructions to access the AIC and can access the following types of data.

The AIC memory mapped registers data—All registers are 32 bits wide and are aligned to word boundaries.
AIC controller FIFO data—An entry is placed into the transmit FIFO by writing to the I2S controller's Serial Audio Data register (AICDR). Writing to AICDR updates a transmit FIFO entry. Reading AICDR flushes out a receive FIFO entry.
The external CODEC registers for I2S CODEC—CODEC registers can be accessed through the L3 bus. The L3 bus operation is emulated by software controlling three GPIO pins.
The external CODEC registers for AC97 CODEC—An AC97 audio CODEC can contain up to sixty-four 16-bit registers. A CODEC uses a 16-bit address boundary for registers. The AIC supplies access to the CODEC registers through several registers.
The internal CODEC registers can be accessed via memory-mapped registers in the CODEC.

The DMA controller can only access the FIFOs. Accesses are made through the data registers, as explained in the previous paragraph. The DMA controller responds to the following DMA requests made by the I2S controller:

The transmit FIFO request is based on the transmit trigger-threshold (AICFR.TFTH) setting. See 2583H0 for further details regarding AICFR.TFTH.
The receive FIFO request is based on the receive trigger-threshold (AICFR.RFTH) setting. See 2584H0 for further details regarding AICFR.RFTH.

Before operation to AIC, you may need to set proper PIN function selection from GPIO using if the pin is shared with GPIO.

Please also reference to "AC '97 Component Specification Revision 2.3, 2002" when deal with AIC AC-link operations.

26.4.1 Initialization

At power-on or other hardware reset (WDT and etc), AIC is disabled. Software must initiate AIC and the internal or external CODEC after power-on or reset. If errors found in data transferring, or in other places, software must initial AIC and optional, the internal or external CODEC. Here is the initial flow:

1 Select internal or external CODEC (AICFR.ICDC).
2 If external CODEC is selected, select AC-link or I2S/MSB-Justified (AICFR.AUSEL). If internal CODEC is used, select I2S/MSB-Justified format (AICFR.AUSEL=1). If the resettlement without involving link format and architecture changing, this step can be skip.
3 If I2S/MSB-Justified is selected, select between I2S and MSB-Justified (I2SCR.AMSL).
4 Decide BIT_CLK direction (AICFR.BCKD) and SYNC direction (AICFR.SYNCD).
5 If BIT_CLK is configured as output, BIT_CLK divider I2SDIV.DV must be set to what correspond with the values as shown in 2585HTable 26-7. And the clock selection and the divider between PLL clock out and AIC also must be set (CFCR.I2S and I2SCDR in CPM). If internal CODEC is used, select 12MHz clock input (via set proper value in CFCR.I2S and I2SCDR), I2S format (I2SCR.AMSL=0), input BIT_CLK (AICFR.BCKD=0), input SYNC (AICFR.SYNCD=0).
6 Enable AIC by write 1 to AICFR.ENB.
7 If it needs to reset AIC registers and flush FIFOs, write 1 to AICFR.RST. If it need only flush FIFOs, write 1 to AICCR.FLUSH. BIT_CLK must exist here and after.
8 In AC-link format, issue a warm or cold CODEC reset.
9 In AC-link format, configure AC ’97 CODEC via ACCAR and ACCDR registers. If the resettlement doesn’t involving AC’97 CODEC registers changing, this step can be skipped.
10 In case of external CODEC with I2S/MSB-Justified format, configure I2S/MSB-justified CODEC via the control bus connected to the CODEC, for instance I2C or L3, depends on CODEC. In case of internal CODEC, configure CODEC via CODEC’s memory mapped registers. If the resettlement without involving I2S/MSB-justified CODEC or ADC/DAC function changing, this step can be skip.

26.4.2 AC '97 CODEC Power Down

AC '97 CODEC can be placed in a low power mode. When the CODEC's power-down register (26h), is programmed to the appropriate value, the CODEC will be put in a low power mode and both BIT_CLK and SDATA_IN will be brought to and held at a logic low voltage level.

Once powered down, re-activation of the AC-link via re-assertion of the SYNC signal must not occur for a minimum of four audio frame times following the frame in which the power down was triggered. When AC-link powers up it indicates readiness via the CODEC Ready bit (input slot 0, bit 15).

26.4.3 Cold and Warm AC ’97 CODEC Reset

AC-link reset operations occur when the system is initially powered up, when resuming from a lower powered sleep state, and in response to critical subsystem failures that can only be recovered from with a reset.

26.4.3.1 Cold AC ‘97 CODEC Reset

A cold reset is achieved by asserting RESET# for the minimum specified time. By driving RESET# low, BIT_CLK, and SDATA_IN will be activated, or re-activated as the case may be, and all AC ‘97 CODEC registers will be initialized to their default power on reset values.

RESET# is an asynchronous AC ‘97 CODEC input.

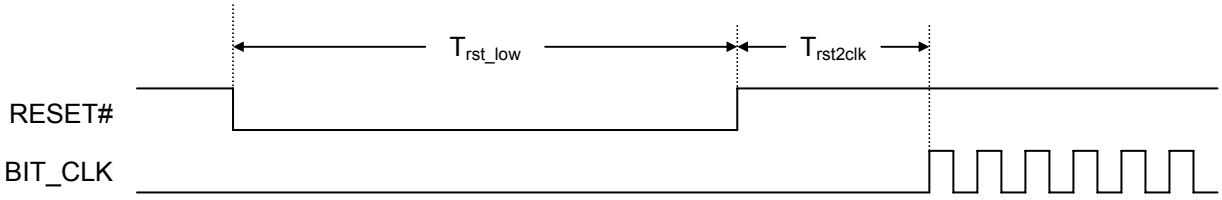


Figure 26-18 Cold AC ’97 CODEC Reset Timing

Table 26-4 Cold AC ’97 CODEC Reset Timing parameters

Parameter	Symbol	Min	Type	Max	Units
RESET# active low pulse width	T_{rst_low}	1.0	-	-	μs
RESET# inactive to BIT_CLK startup delay	$T_{rst2clk}$	162.8	-	-	ns

26.4.3.2 Warm AC '97 CODEC Reset

A warm AC'97 reset will re-activate the AC-link without altering the current AC'97 register values. Driving SYNC high for a minimum of 1 µs in the absence of BIT_CLK signals a warm reset.

Within normal audio frames SYNC is a synchronous AC '97 CODEC input. However, in the absence of BIT_CLK, SYNC is treated as an asynchronous input used in the generation of a warm reset to AC '97 CODEC.

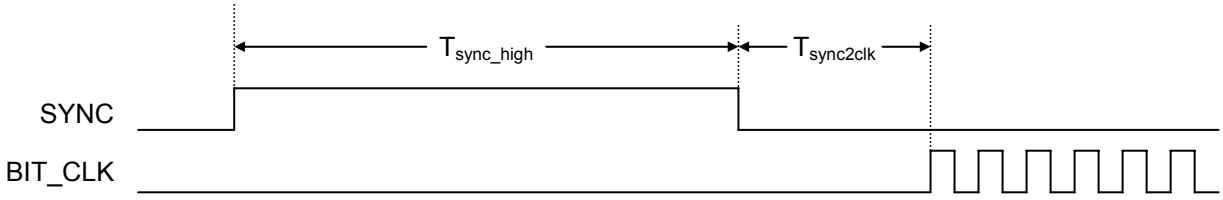


Figure 26-19 Warm AC '97 CODEC Reset Timing

Table 26-5 Warm AC '97 CODEC Reset Timing Parameters

Parameter	Symbol	Min	Type	Max	Units
SYNC active high pulse width	T_{sync_high}	1.0	-	-	Ms
SYNC inactive to BIT_CLK startup delay	$T_{sync2clk}$	162.8	-	-	Ns

26.4.4 External CODEC Registers Access Operation

The external audio CODEC can be configured/controlled by its internal registers. To access these registers, an I2S/MSB-justified CODEC usually employs L3 bus, SPI bus, I2C bus or other control bus. The L3 bus operation can be emulated by software by using 3 GPIO pins of the chip. For AC '97, "AC '97 Component Specification" defines the CODEC register access protocol. Several registers are provided in AIC to accomplish this task.

The ACCAR and ACCDR are used to send a register accessing request command to external AC'97 CODEC. The ACSAR and ACSDR are used to receive a register's content from external AC'97 CODEC. The register accessing request and the register's content returning is asynchronous.

The AC'97 CODEC register accessing request flow:

1 If ACSR.CADT is 0, wait for 25.4μs. If no previous accessing request, this step can be skip.
2 Clear ACSR.CADT.
3 If read access, write read-command and register address to ACCAR, if write access, write write-command and register address to ACCAR and write data to ACCDR. Any order of write ACCAR and ACCDR is OK.
4 Polling for ACSR.CADT changing to 1, which means the request has been send to CODEC via AC-link.

The AC'97 CODEC register content receiving flow by polling:

1 Polling for ACSR.SADR changing to 1.
2 Read the CODEC register's address from ACSAR and content from ACSDR.
3 Clear ACSR.SADR.

The AC'97 CODEC register content receiving flow by interrupt:

1 Before accessing request, clear ACSR.SADR and set ACCR2.ESADR.
2 Waiting for the interrupt. When the interrupt is found, check if ACSR.SADR is 1, if not, repeat this step again.
3 Read the CODEC register's address from ACSAR and content from ACSDR.
4 Clear ACSR.SADR.

26.4.5 Audio Replay

Outgoing audio sample data (from AIC to CODEC) is written to AIC transmit FIFO from processor via store instruction or from memory via DMA. AIC then takes the data from the FIFO, serializes it, and sends it over the serial wire SDATA_OUT to an external CODEC or over an internal wire to an internal CODEC.

The audio transmission is enabled automatically when the AIC is enabled by set AICFR.ENB. But all replay data is zero at this time except both of the following conditions are true:

1 AICCR.ERPL must be 1. If AICCR.ERPL is 0, value of zero is send to CODEC even if there are samples in transmit FIFO.
2 At least one audio sample data in the transmit FIFO. If the transmit FIFO is empty, value of zero or last sample depends on AICFR.LSMP, is send to CODEC even if AICCR.ERPL is 1.

Here is the audio replay flow:

1 Configure the CODEC as needed.
2 Configure sample size by AICCR.OSS.
3 Configure sample channels (AICCR.CHANNEL).
4 If sample size is configured 16 bit, select packed or unpacked mode (AICCR.PACK16).
5 If two channels is configured, select the right-channel-first sample data or not (I2SCR.RFIRST).
6 If two channels is configured, select the sample data switched or not (I2SCR.SWLH).
7 Configure sample rate by clock dividers (for I2S/MSB-Justified format with BIT_CLK is provided internally) or by CODEC registers (for AC-link or BIT_CLK provided by external CODEC) or by accessing CODEC internal registers (for internal CODEC).
8 For AC-link, configure replay channels by ACCR1.XS.
9 Some other configurations: mono to stereo, endian switch, signed/unsigned data transfer, transmit FIFO configuration, play ZERO or last sample when TX FIFO under-run, and etc.
10 Write 1 to AICCR.ERPL.
   It is suggested that at least a frame of PCM data is pre-filled in the transmit FIFO to prevent FIFO under-run flag (AICSR.TUR).
   But when using internal CODEC, write first frame of PCM data to transmit FIFO till TX FIFO under-run (AICSR.TUR is set to 1), otherwise left/right channel may be switched.
11 Fill sample data to the transmit FIFO. Repeat this till finish all sample data. In this procedure, please control the FIFO to make sure no FIFO under-run and other errors happen. When the transmit FIFO under-run, noise or pause may be heard in the audio replay, AICSR.TUR is 1, and if AICCR.ETUR is 1, AIC issues an interrupt. Please reference to 2586H26.4.7 for detail description on FIFO.
12 Waiting for AICSR.TFL change to 0. So that all samples in the transmit FIFO has been replayed, then we can have a clean start up next time.
13 Write 0 to AICCR.ERPL.

NOTES:

1 Before replaying Open ADC BITCLK and close it to generating Record internal circuit reset

when using internal CODEC.

26.4.6 Audio Record

Incoming audio sample data (from CODEC to AIC) is received from SDATA_IN (for an external CODEC) or an internal wire (for an internal CODEC) serially and converted to parallel word and stored in AIC receive FIFO. Then the data can be taken from the FIFO to processor via load instruction or to memory via DMA.

The audio recording is enabled automatically when the AIC is enabled by set AICFR.ENB. But all received data is discarded at this time except both of the following conditions are true:

1 AICCR.EREC must be 1. If AICCR.EREC is 0, the received data is discarded even if there are rooms in the receive FIFO.
2 At least one room left in the receive FIFO. If the receive FIFO is full, the received data is discarded even if AICCR.EREC is 1.

Here is the audio record flow:

1 Configure the CODEC as needed.
2 Configure sample size by AICCR.ISS.
3 Configure sample rate by clock dividers (for I2S/MSB-Justified format with BIT_CLK is provided internally) or by CODEC registers (for AC-link or BIT_CLK provided by external CODEC) or by CODEC memory mapped registers (for internal CODEC).
4 Some other configurations: signed/unsigned data transfer, receive FIFO configuration, and etc.
5 Write 1 to AICCR.EREC. Make sure there are rooms available in the receive FIFO before set AICCR.EREC. Usually, it should empty the receive FIFO by fetch data from it before set AICCR.EREC.
6 Take sample data form the receive FIFO. Repeat this till the audio finished. In this procedure, please control the FIFO to make sure no FIFO over-run and other errors happen. When the receive FIFO over-run, same recorded audio samples will be lost, AICSR.ROR is 1, and if AICCR.EROR is 1, AIC issues an interrupt. Please reference to 26.4.7 for detail description on FIFO. For AC-link, ACCR1.RS tells which channels are recorded.
   When using internal CODEC, the first data should be ignored.
7 Write 0 to AICCR.EREC.
8 Take sample data from the receive FIFO until AICSR.RFL change to 0. So that all samples in the receive FIFO has been taken away, then we can have a clean start up next time. When the receive FIFO is empty, read from it returns zero.

26.4.7 FIFOs operation

AIC has two FIFOs, one for transmit audio sample and one for receive. All AIC played/recorded audio sample data is taken from/send to transmit/receive FIFOs. The RX FIFO is in 24 bits width and 32 entries depth, one entry for keeping one audio sample regardless of the sample size. The RX FIFO is

in 32 bits width and 64 entries depth, one entry for keeping one audio sample regardless of the sample size, but in 16 bits packed mode, one entry for keeping two audio samples. AICDR.DATA provides the access point for processor/DMA to write to transmit FIFO and read from receive FIFO. One time access to AICDR.DATA process one sample. The sample data should be put in LSB (Least Significant Bit) in memory or processor registers. For transmitting, bits exceed sample are discarded. For receiving, these bits are set to 0. Figure 26-20 illustrates the FIFOs access.

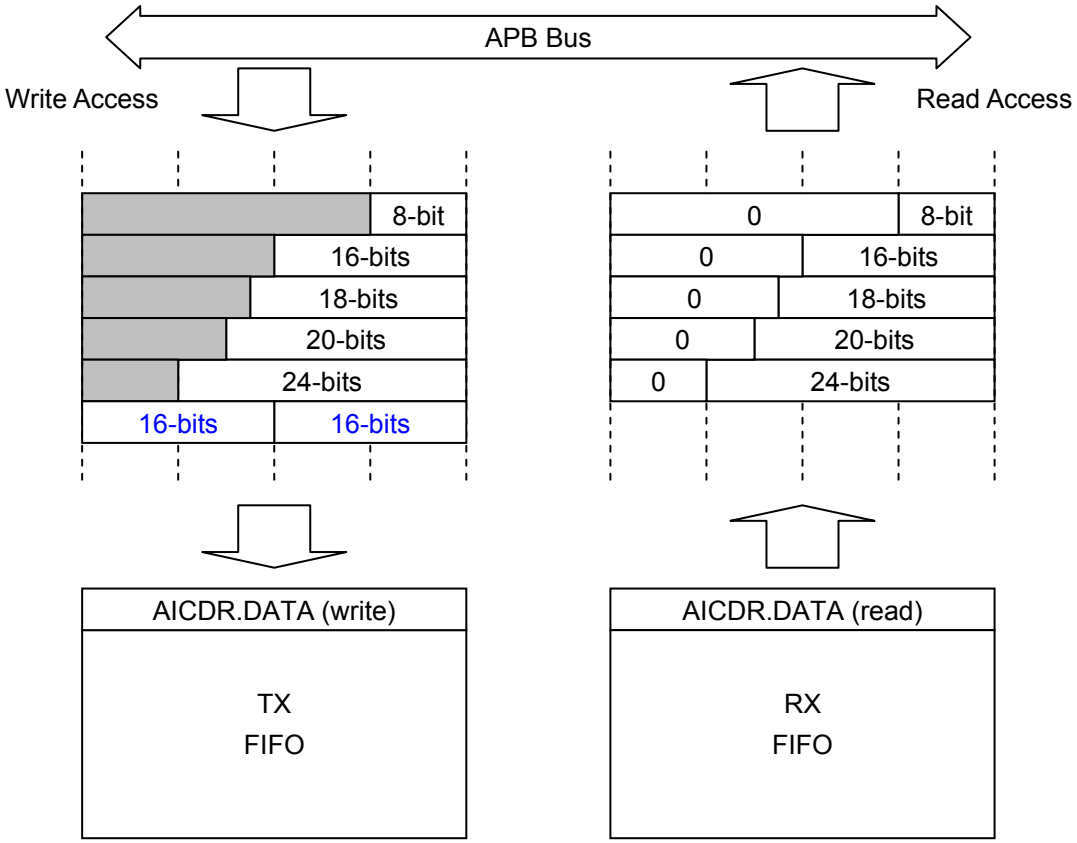


Figure 26-20 Transmitting/Receiving FIFO access via APB Bus

The software and bus initiator must guarantee the right sample placement at the bus.

In case of DMA bus initiator, one 24, 20, 18 bits audio sample must occupies one 32-bits word in memory, so 32-bits width DMA must be used. One 16 bits sample occupies one 16-bits half word in memory, so 16-bits width DMA must be used. One 8-bits sample occupies one byte in memory, and use 8-bits width DMA except 16bits packed mode. If in 16 bits packed mode, Two 16 bits sample occupies one 32-bits word in memory, so 32-bits width DMA must be used.

In case of processor bus initiator, any type of the audio sample must occupy one CPU general-purpose register at LSB, and read/write from/to AICDR.DATA with 32-bits load/store instruction. When process small sample size, 16-bits or 8-bits, software may need to do the data

pack/unpack except 16 bits packed mode. In the 16bits packed mode, the sample data is packed, and two 16 bits audio samples occupy one CPU general-purpose register.
The AICFR.TFTH and AICFR.RFTH are used to set the FIFO level thresholds, which are the trig levels of DMA request and/or FIFO service interrupt. The AICFR.TFTH and AICFR.RFTH should be set to proper value; too small or too big are not good. When AICFR.RFTH is too small, or AICFR.TFTH is too big, the DMA burst length or the number of sample can be processed by processor is too small, which harms the bus or processor efficiency. When AICFR.RFTH is too big or AICFR.TFTH is too small, the bus or the interrupt latency left for under-run/over-run is too small, which may causes replay/record errors.

AICSR.TUR is set to 1 during transmit under-run conditions. If AICCR.ETUR is 1, this can trigger an interrupt. During transmit under-run conditions, zero or last sample is continuously sent out across the serial link. Transmit under-run can occur under the following conditions:

1 Valid transmit data is still available in memory, but the DMA controller/processor starves the transmit FIFO, as it is busy servicing other higher-priority tasks.
2 The DMA controller/processor has transferred all valid data from memory to the transmit FIFO.

AICSR.ROR is set to 1 during receive over-run conditions. If AICCR.EROR is 1, this can trigger an interrupt. During receive over-run conditions, data sent by the CODEC is lost and is not recorded.

When replay/record two channels data, the left channel is default the first data in FIFOs and in the serial link. If multiple channels in AC-link are used, the channel sample order is follows the slot order. In 16bits packed mode, could configure that the left channel is the first data or the right channel. By default, the 16 bits LSB is left channel, 16 bits MSB is the right channel. But it also could be switched the Left or the Right channel (I2SCR.SWLH).

26.4.8 Data Flow Control

There are three approaches provided to control/synchronize the audio incoming/outgoing data flow.

26.4.8.1 Polling and Processor Access

AICSR.RFL and AICSR.TFL reflect how many samples exist in receiving and transmitting FIFOs. Through read these register fields, processor can detect when there are samples in receiving FIFO in audio record and then load them from the RX-FIFO, and when there are rooms in transmitting FIFO in audio replay and then store samples to the TX-FIFO.

Polling approach is in very low efficiency and is not recommended.

26.4.8.2 Interrupt and Processor Access

Set proper values to AICFR.TFTH and AICFR.RFTH, the FIFO interrupts trig thresholds. Set AICCR.ETFS and/or AICCR.ERFS to 1 to enable transmitting and/or receiving FIFO level trigger interrupts. When the interrupt found, it means there are rooms or samples in the TX or RX FIFO, and processor can store or load samples to or from the FIFO.

Interrupt approach is more efficient than polling approach.

26.4.8.3 DMA Access

Audio data is real time stream, though it is in low data bandwidth, usually less than 1.2Mbps. DMA approach is the most efficient and is the recommended approach.

To enable DMA operation, set AICCR.TDMS and AICCR.RDMS to 1 for transmit and receive respectively. It also needs to allocate two channels in DMA controller for data transmitting and receiving respectively. Please reference to the processor's DMA controller spec for the details.

The AICFR.TFTH and AICFR.RFTH are used to set the transmitting and receiving FIFO level thresholds, which determine the issuing of DMA request to DMA controller. To respond the request, DMAC initiator and controls the data movement between memory and TX/RX FIFO.

26.4.9 Audio Samples format

26.4.9.1 16 bits packed mode

One channel (mono) mode and two channels (stereo) mode:

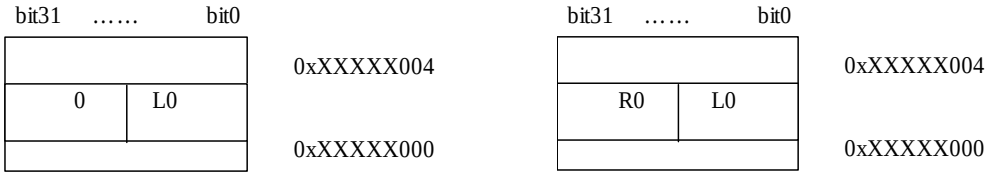


Figure 26-21 One channel (Left) and Two channels (right) mode (16 bits packed mode)

Four channels mode and six channels mode:

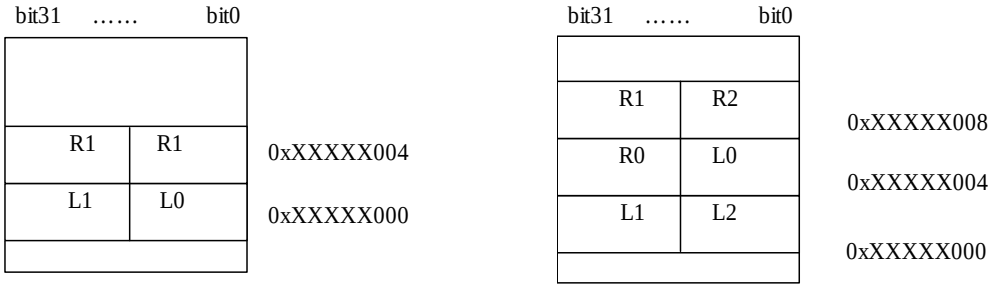


Figure 26-22 Four channels (Left) and Six channels (right) mode (16 bits packed mode)

Eight channels mode:

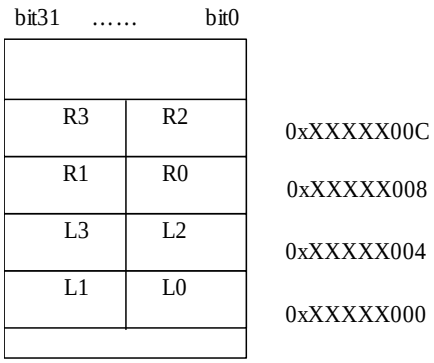


Figure 26-23 Eight channels mode (16 bits packed mode)

26.4.9.2 Normal mode.

One channel (Mono) and two channels (stereo) mode:

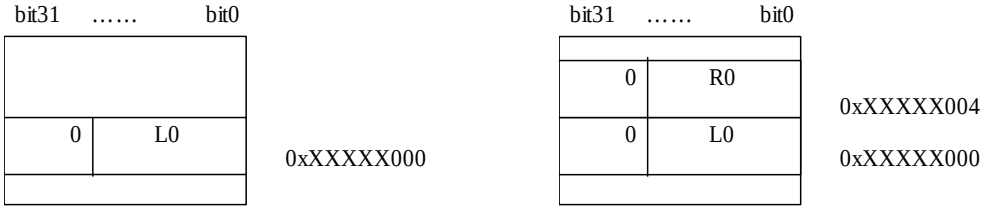


Figure 26-24 One channel (Left) and Two channels (right) mode

Four channels mode and six channels mode:

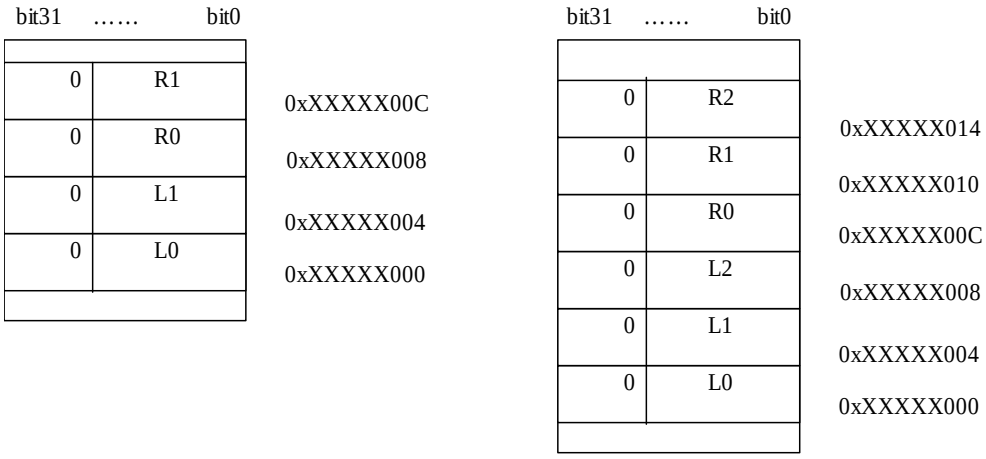


Figure 26-25 Four channels (Left) and Six channels (right) mode

Eight channel mode:

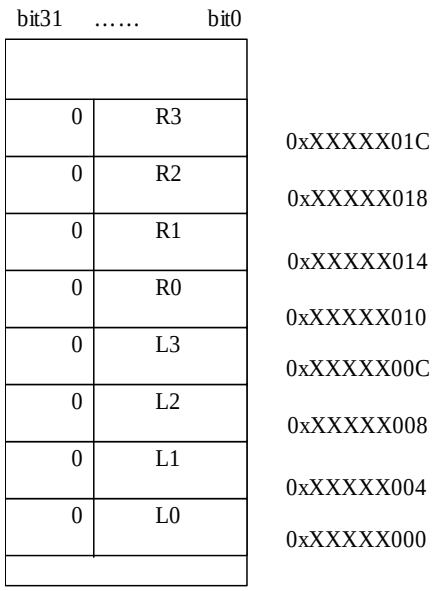


Figure 26-26 Eight channels mode

26.4.10 Serial Audio Clocks and Sampling Frequencies

For internal CODEC, CODEC module containing the audio CODEC circuit/logic and corresponding controlling registers. CODEC needs a 12MHz clock from CPM called SYS_CLK and provides I_BITCLK, O_BITCLK and I_SYNC, O_SYNC (left-right clock which is the sample rate as ADC or DAC) to AIC for outgoing and incoming audio respectively. These clocks change when change the sample rate in CODEC controlling registers. When using internal CODEC, must configure SYNC and BIT_CLK as input, more details refers to CODEC Spec.

For AC-link, the bit clock is input from chip external and is fixed to 12.288MHz. The sample frequency of 48kHz is supported in nature. Variable Sample Rate feature is supported in this AIC. If the CODEC supports this feature, sample rate other than 48kHz audio data can be replay directly. Otherwise, software has to do the rate transfer to replay other sample rate audio data. Double rate, 96kHz or even 88.2kHz audio is also supported with proper CODEC.

Following are for BIT_CLK/SYS_CLK configuration in I2S/MSB-Justified format with external CODEC.

The BIT_CLK is the rate at which audio data bits enter or leave the AIC. BIT_CLK can be supplied either by the CODEC or an internally PLL. If it is supplied internally, BIT_CLK is configured as output pins, and is supplied out to the CODEC. If BIT_CLK is supplied by the CODEC, then it is configured as an input pin. Register bit AICFR.BCKD is used to select BIT_CLK direction.

The audio sampling frequency is the frequency of the SYNC signal, which must be 1/64 of BIT_CLK, $f_{BIT\_CLK} = 64\ f_{S.}$ But SYNC signal frequency is not fixed when using internal CODEC.

SYS_CLK is only for CODEC. It usually takes one of the two roles, as CODEC master clock input or as CODEC over-sampling clock input. If SYS_CLK roles as CODEC master clock input, it usually should be set to a fixed frequency according to CODEC requirement but independent to audio sample rate. In this case, usually there is a PLL in the CODEC and CODEC roles master mode. See Figure 26-3 for the interface diagram. This is the recommended AIC CODEC system configuration.

If SYS_CLK roles as CODEC over-sampling clock, its frequency is usually 4, 6, 8 or 12 times of BIT_CLK frequency, which are 256, 384, 512 and 768 times of audio sample rates. Table 26-6 lists the relation between sample rate, BIT_CLK and SYS_CLK frequencies.

Table 26-6 Audio Sampling rate, BIT_CLK and SYS_CLK frequencies

Sample Rate f_S (kHz)	BIT_CLK (MHz) $f_{BIT\_CLK} = 64\ f_S$	SYS_CLK (MHz)			
		256 f_S	384 f_S	512 f_S	768 f_S
48	3.072	12.288	18.432	24.576	36.864
44.1	2.8224	11.2896	16.9344	22.5792	33.8688
32	2.048	8.192	12.288	16.384	24.576
24	1.536	6.144	9.216	12.288	18.432
22.05	1.4112	5.6448	8.4672	11.2896	16.9344
16	1.024	4.096	6.144	8.192	12.288
11.025	0.7056	2.8224	4.2336	5.6448	8.4672
8	0.512	2.048	3.072	4.096	6.144

In this processor, SYS_CLK can be selected from EXCLK or generated by dividing the PLL output clock in a CPM divider controlled by I2SCDR. If BIT_CLK is chosen as an output, another divider in AIC is used to divide SYS_CLK for it.

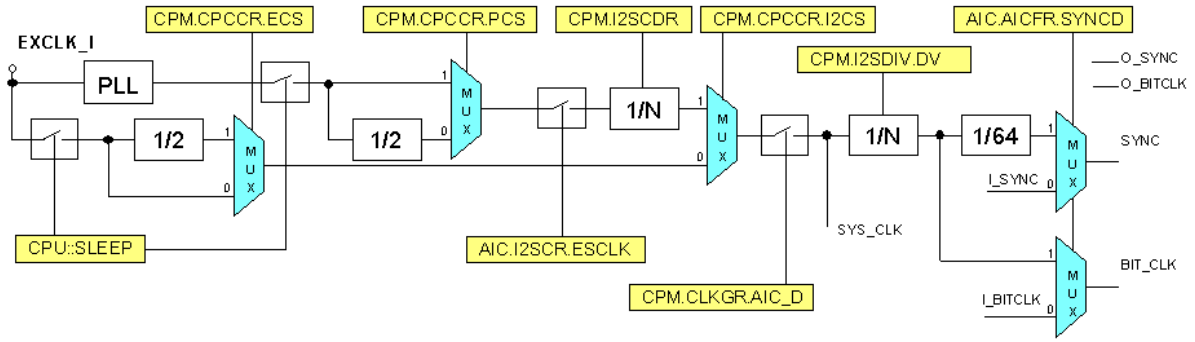


Figure 26-27 SYS_CLK, BIT_CLK and SYNC generation scheme

The setting of I2SDIV.DV is shown in Table 26-7.

Table 26-7 BIT_CLK divider setting

I2SDIV.DV	f_{SYS_CLK}	f_{BIT_CLK}	f_{SYS_CLK} / f_{BIT_CLK}
0x1	128 f_S	64 f_S	2
0x2	196 f_S	64 f_S	3
0x3	256 f_S	64 f_S	4
0x5	384 f_S	64 f_S	6
0x7	512 f_S	64 f_S	8
0xB	768 f_S	64 f_S	12

As we observe in Table 26-6, if SYS_CLK is taken as over-sampling clock by CODEC, the common multiple of all SYS_CLK frequencies is much bigger than the PLL output clock frequency. To generate all different SYS_CLK frequencies, one approach is change PLL frequency according to sample rate.

This is not realistic, since frequently change PLL frequency during normal operation is not recommended.

Another approach is to found some approximate common multiples of all SYS_CLK frequencies according to the fact that there tolerance in audio sample rate. Take f_{SYS_CLK} = 256 f_S, Table 26-8 list most frequencies, which are less than 400MHz, with relatively small sample rate errors. It is suggested to set PLL frequency as close to the frequencies listed as possible, then use clock dividers to generate different SYS_CLK/BIT_CLK for different sample rate.

Table 26-8 Approximate common multiple of SYS_CLK for all sample rates

Approximate Common Frequency (MHz)	Max Error Caused in Audio Sample Rate (%)
123.53	0.53
147.11	0.24
170.68	0.79
235.5	0.87
247.06	0.53
270.64	0.11
280.56	0.73
294.22	0.24
305.14	0.67
317.79	0.53
329.57	0.66
341.35	0.79
347	0.85
353.13	0.90
358.79	0.69
370.59	0.53
382.96	0.54
394.17	0.24

Take PLL = 270.64 MHz as an example, Table 26-9 lists the divider settings for various sample rates.

Table 26-9 CPM/AIC clock divider setting for various sampling rate if PLL = 270.64MHz

Sample Rate (kHz)	I2SCDR	I2SDIV.DV	Sample Rate Error (%)
48	1	11	0.11
44.1	1	12	-0.11
32	0	33	0.11
24	1	22	0.11
22.05	1	24	-0.11

16	1	33	0.11
12	1	44	0.11
11.025	1	48	-0.11
8	1	66	0.11

For an EXCLK clock frequency, try to generate PLL frequencies as close to the frequencies listed in Table 26-8 as possible. Table 26-10 lists the PLL parameters and audio sample errors at different PLL frequencies for EXCLK at 12MHz.

Table 26-10 PLL parameters and audio sample errors for EXCLK=12MHz

PLL			Max Sample Rate Error
M	N	Freq (MHz)	
103	10	123.6	0.59%
49	4	147	0.31%
128	9	170.67	0.79%
157	8	235.5	0.87%
103	5	247.2	0.59%
65	3	260	0.82%
45	2	270	0.35%
203	9	270.67	0.12%
113	5	271.2	0.32%
187	8	280.5	0.75%
237	10	284.4	0.81%
49	2	294	0.31%
178	7	305.14	0.67%
53	2	318	0.60%
302	11	329.45	0.70%
256	9	341.33	0.79%
318	11	346.91	0.88%
206	7	353.14	0.90%
299	10	358.8	0.69%
247	8	370.5	0.55%
351	11	382.91	0.55%
230	7	394.29	0.27%

The BIT_CLK should be stopped temporary when change the divider settings, or when change BIT_CLK source (from internal or external), to prevent clock glitch. Register I2SCR.STPBK is provided to assist the task. When I2SCR.STPBK = 1, BIT_CLK is disabled no matter whether it is generated internally or inputted from the external source. The operation flow is described in following:

1 Stop all replay/record by clear AICCR.ERPL and AICCR.EREC.

2 Polling I2SSR.BSY till it is 0.

3 Stop the BIT_CLK by write 1 to I2SCR.STPBK.
4 Operations concerning BIT_CLK.
5 Resume the BIT_CLK by write 0 to I2SCR.STPBK.

26.4.11 Interrupts

The following status bits, if enabled, interrupt the processor:

- Receive FIFO Service (AICSR.RFS). It's also DMA Request.
- Transmit FIFO Service (AICSR.TFS). It's also DMA Request.
- Transmit Under-Run (AICSR.TUR).
- Receive Over-Run (AICSR.ROR).
- Command Address and Data Transmitted, AC-link only (ACSR.CADT).
- External CODEC Registers Status Address and Data Received, AC-link only (ACSR.SADR).
- External CODEC Registers Read Status Time Out, AC-link only (ACSR.RSTO).

For further details, see the corresponding register description sections.

26.5 SPDIF Guide

26.5.1 Set SPDIF clock frequency

Set SPDIF clock frequency is as same as i2s clock.

26.5.2 PCM audio mode operation (Reference IEC60958)

1 Set SPCFG1 and SPCFG2 to configure SPDIF transmitter.
   a Set SPCFG2.CON_PRO to 0 to choose consumer mode.
   b Set SPCFG2. AUDIO_N to 0 to choose linear PCM audio data mode.
   c Set SPCFG1.XXX to configure SPDIF.
   d Set SPCFG2.XXX to configure SPDIF.
2 Set SPCTRL.DMA_EN to choose DMA mode or CPU mode.
3 Set SPCTRL.SIGN_N to choose whether to transfer the most significant bit by toggle or not.
4 Set SPCTRL. SFT_RST to 1 reset FIFO.
5 Wait SPCTRL. SFT_RST set to be set 0 by hardware.
6 Set SPCTRL.M_TRIG and SPCTRL.M_FFUR to enable or disable the interrupt.
7 Set SPCTRL.INVALID 1 or 0 to set the V bit of sub-frame.
8 Set SPENA.SPEN to 1 to Enable SPDIF to transmitter.

26.5.3 Non-PCM mode operation (Reference IEC61937)

1 Set SPCFG1 and SPCFG2 to configure SPDIF transmitter.
   a Set SPCFG2.CON_PRO to 0 to choose consumer mode.
   b Set SPCFG2.AUDIO_N to 1 to choose non-PCM mode.
   c Set SPCFG1.SRC_NUM to 0.
   d Set SPCFG1.CH1_NUM to 0.
   e Set SPCFG1.CH2_NUM to 0.
   f Set SPCFG2.PRE to 0.
   g Set SPCFG2.CH_MD to 0.
   h Set SPCFG2.ORG_FRQ to 0.
   i Set SPCFG2.SAMPL_WL to 0.
   j Set SPCFG2.MAX_WL to 0.
   k Set SPCFG1.XXX to configure SPDIF.
   l Set SPCFG2.XXX to configure SPDIF.
2 Set SPCTRL.DMA_EN to choose DMA mode or CPU mode.
3 Set SPCTRL.SIGN_N to choose whether to transfer the most significant bit by toggle or not.
4 Set SPCTRL. SFT_RST to 1 reset FIFO.
5 Wait SPCTRL. SFT_RST to be set to 0 by hardware.
6 Set SPCTRL.M_TRIG and SPCTRL.M_FFUR to enable or disable the interrupt.
7 Set SPCTRL.INVALID 1 or 0 to set the V bit of sub-frame.

8 Set SPENA.SPEN to 1 to Enable SPDIF to transmitter.

26.5.4 Disable operation

1 Set SPENA.SPEN to 0 to disable SPDIF to transmitter.
2 Wait SPSTATE.BUSY to be set to 0 by hardware.
3 You can do other operation.

27 PCM Interface

27.1 Overview

The PCM has the following features:

- Data starts with the frame PCMSYN or one PCMCLK later
- Support three modes of operation for PCM
  - Short frame sync mode
  - Long frame sync mode
  - Multi-slot mode
- Data is transferred and received with the MSB first
- Support master mode and slave mode
- The PCM serial output data, PCMDOUT, is clocked out using the rising edge of the PCMSCLK
- The PCM serial input data, PCMDIN, is clocked in on the falling edge of the PCMSCLK
- 8/16 bit sample data sizes supported
- DMA transfer mode supported
- Two FIFOs for transmit and receive respectively with 16 samples capacity in every direction

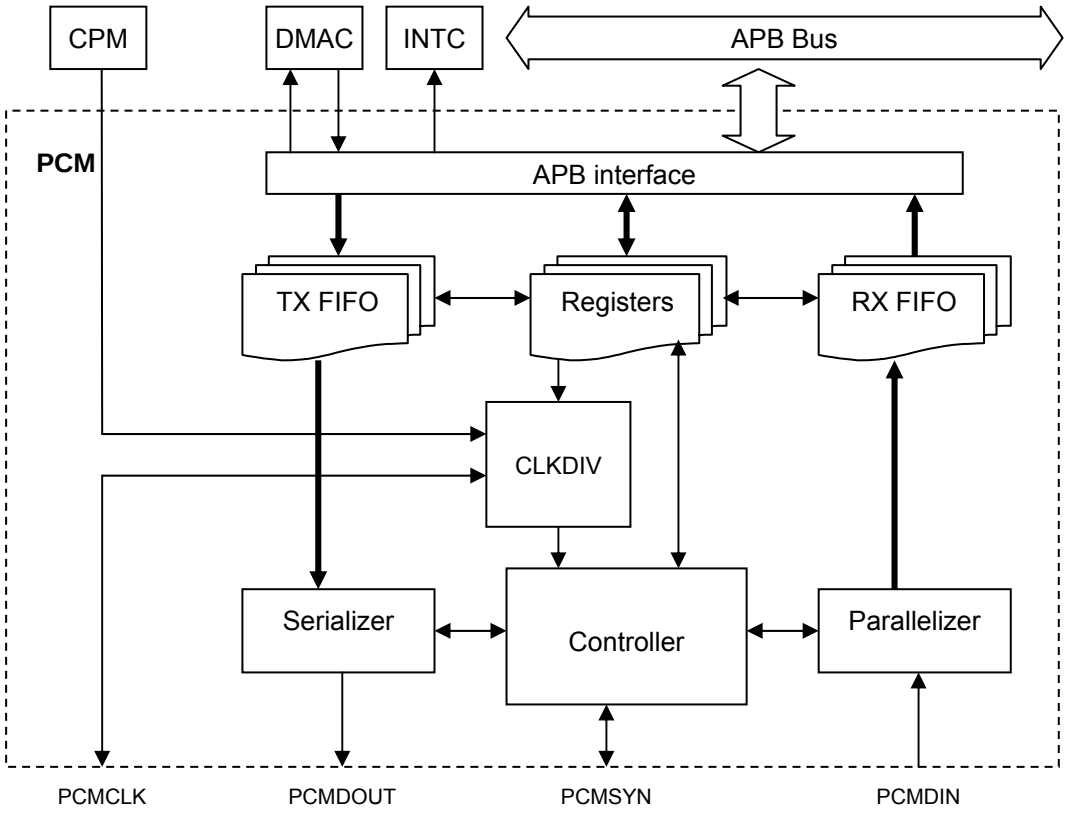
27.2 Pin Description

There are all 4 pins used to connect between PCM interface and an external device. They are listed and described in Table 27-1.

Table 27-1 PCM Interface Pins Description

Name	I/O	Description
PCMCLK	Input/Output	PCM Serial clock Line signal input/output
PCMSYN	Input/Output	PCM sync signal input/output
PCMDOUT	Output	PCM Serial data output
PCMDIN	Input	PCM Serial data input

27.3 Block Diagram



27.4 Register Description

Table 27-2 PCM Registers Description

Name	Description	RW	Reset Value	Address	Size
PCMCTL	PCM Control Register	RW	0x00000000	0x10071000	32
PCMCFG	PCM Configure Register	RW	0x00000110	0x10071004	32
PCMDP	PCM FIFO Data Port Register	RW	0x00000000	0x10071008	32
PCMINTC	PCM Interrupt Control Register	RW	0x00000000	0x1007100c	32
PCMINTS	PCM Interrupt Status Register	RW	0x00000100	0x10071010	32
PCMDIV	PCM Clock Divide Register	RW	0x00000001	0x10071014	32

27.4.1 PCM Control Register (PCMCTL)

PCMCTL 0x00000000

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																						ERDMA	ETDMA	LSMP	ERPL	EREC	FLUSH	RST	SYNEN	CLKEN	PCMEN
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:10	Reserved	These bits always read as 0. Write data to these bits are ignored.	RW
9	ERDMA	Receive DMA Enable. This bit is used to enable or disable the DMA during receiving audio data. ERDMA: 0 = Disabled.; 1 = Enabled.	RW
8	ETDMA	Transmit DMA Enable. This bit is used to enable or disable the DMA during transmit audio data. ETDMA: 0 = Disabled.; 1 = Enabled.	RW
7	LSMP	Select between play last sample or play ZERO sample in TX FIFO underflow. ZERO sample means sample value is zero. LSMP: 0 = Play ZERO sample when TX FIFO underflow.; 1 = Play last sample when TX FIFO underflow.	RW
6	ERPL	Enable Playing Back function. This bit is used to disable or enable the audio sample data transmitting.	RW

			ERPL: 0 — PCM Playing Back Function is Disabled. ERPL: 1 — PCM Playing Back Function is Enabled.	
5	EREC	Enable Recording Function. This bit is used to disable or enable the audio sample data receiving. EREC: 0 — PCM Recording Function is Disabled. EREC: 1 — PCM Recording Function is Enabled.		RW
4	FLUSH	FIFO Flush. Write 1 to this bit flush transmit/receive FIFOs to empty. Writing 0 to this bit has no effect and this bit is always reading 0.		W
3	RST	Reset PCM. Write 1 to this bit reset PCM registers and FIFOs. Writing 0 to this bit has no effect and this bit is always reading 0.		W
2	Reserved	These bits always read as 0. Write data to these bits are ignored.		R
1	CLKEN	Enable the serial clock division logic. Must be HIGH for the PCM to operate.		RW
0	PCMEN	Enable PCM function. This bit is used to enable or disable the PCM function. PCMENB: 0 — Disable PCM Controller. PCMENB: 1 — Enable PCM Controller.		RW

27.4.2 PCM Configuration Register (PCMCFG)

PCMCFG 0x00000004

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																	SLOT		ISS	OSS	IMSBPOS	OMSBPOS	RFTH				TFTH				PCMMOD
RST																		0	0	0	0	0	0	1	0	0	0	1	0	0	0	0

Bits	Name	Description	RW
31:15	Reserved	These bits always read as 0. Write data to these bits are ignored.	R
14:13	SLOT	Controls the amount of valid PCM timeslot in one PCMSYN frame.	RW
12	ISS	Input Sample Size. These bits reflect input sample data size to memory or register. The data sizes supported are: 8/16bits. The sample data is LSB-justified in memory/register. ISS: 0 — 8 bit ISS: 1 — 16 bit	RW
11	OSS	Output Sample Size. These bits reflect output sample data size from	RW

		memory or register. The data sizes supported are: 8/16 bits. The sample data is LSB-justified in memory/register. **OSS** / **Sample Size** 0 / 8 bit 1 / 16 bit	
10	IMSBPOS	Controls the position of the MSB bit in the serial input stream relative to the PCMSYN signal. 0: MSB is captured on the falling edge of PCMCLK during the same cycle that PCMSYNC is high 1: MSB is captured on the falling edge of PCMCLK during the cycle after the PCMSYNC is high	RW
9	OMSBPOS	Controls the position of the MSB bit in the serial output stream relative to the PCMSYN signal. 0: MSB sent during the same clock that PCMSYN is high 1: MSB sent on the next PCMSCLK cycle after PCMSYNC is high	RW
8:5	RFTH	Receive FIFO threshold for interrupt or DMA request. Determines when the RFS flags go active for the RXFIFO. When the sample number in receive FIFO, indicated by PCMINTS.RFL, is great than the threshold value, PCMINTS.RFS is set. Larger RFTH value provides lower DMA/interrupt request frequency but have more risk to involve receive FIFO overflow. The optimum value is system dependent.	RW
4:1	TFTH	Transmit FIFO threshold for interrupt or DMA request. Determines when the TFS flags go active for the TXFIFO. When the sample number in transmit FIFO, indicated by PCMINTS.TFL, is less than the threshold value, PCMINTS.TFS is set. Smaller TFTH value provides lower DMA/interrupt request frequency but have more risk to involve transmit FIFO underflow. The optimum value is system dependent.	RW
0	PCMMOD	PCM mode select. 0: Master mode; 1: Slave mode.	RW

27.4.3 PCM FIFO DATA PORT REGISTER (PCMDP)

PCMDP **0x00000008**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATA																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	DATA	FIFO port. When write to it, data is push to the transmit FIFO. When read from it, data is pop from the receiving FIFO.	RW

27.4.4 PCM INTERRUPT CONTROL REGISTER (PCMINTC)

PCMINTCR 0x0000000c

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																												ETFS	ETUR	ERFS	EROR
RST																													0	0	0	0

Bits	Name	Description	RW
31:4	Reserved	These bits always read as 0. Write data to these bits are ignored.	R
3	ETFS	Enable TFS Interrupt. This bit is used to control the TFS interrupt enable or disable. ETFS / TFS Interrupt 0 – Disabled. 1 – Enabled.	RW
2	ETUR	Enable TUR Interrupt. This bit is used to control the TUR interrupt enable or disable. ETUR / TUR Interrupt 0 – Disabled. 1 – Enabled.	RW
1	ERFS	Enable RFS Interrupt. This bit is used to control the RFS interrupt enable or disable. ERFS / RFS Interrupt 0 – Disabled. 1 – Enabled.	RW
0	EROR	Enable ROR Interrupt. This bit is used to control the ROR interrupt enable or disable. EROR / ROR Interrupt 0 – Disabled. 1 – Enabled.	RW

27.4.5 PCM INTERRUPT STATUS REGISTER (PCMINTS)

PCMINTS **0x00000010**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																	RSTS	TFL					TFS	TUR	RFL					RFS	ROR
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:15	Reserved	These bits always read as 0. Write data to these bits are ignored.	R
14	RSTS	Soft reset / flush state. 0: Nothing / reset or flush operation has completed 1: reset or flush operation has not completed	R
13:9	TFL	Transmit FIFO Level. The bits indicate the amount of valid PCM data in Transmit FIFO.	R
8	TFS	Transmit FIFO Service Request. This bit indicates that transmit FIFO level exceeds TFL threshold which is controlled by PCMCFG.TFTH When TFS is 1, it may trigger interrupt or DMA request depends on the interrupt enable and DMA setting. **TFS** / **Description** 0 / Transmit FIFO level exceeds TFL threshold. 1 / Transmit FIFO level at or below TFL threshold.	RW
7	TUR	Transmit FIFO Under Run. This bit indicates that transmit FIFO has or has not experienced an under-run. **TUR** / **Description** 0 / When read, indicates under-run has not been found. 0 / When write, clear itself. 1 / When read, indicates data has even been read from empty transmit FIFO. 1 / When write, not effects.	RW
6:2	RFL	Receive FIFO Level. The bits indicate the amount of valid PCM data in Receive FIFO.	R
1	RFS	Receive FIFO Service Request. This bit indicates that receive FIFO level is or not below RFL threshold which is controlled by PCMCFG.RFTH. When RFS is 1, it may trigger interrupt or DMA request depends on the interrupt enable and DMA setting. **RFS** / **Description** 0 / Receive FIFO level below RFL threshold.	RW

			1	Receive FIFO level at or above RFL threshold.		
0	ROR	Receive FIFO Over Run. This bit indicates that receive FIFO has or has not experienced an overrun.				RW

ROR	Description
0	When read, indicates over-run has not been found.
	When write, clear itself.
1	When read, indicates data has even been written to full receive FIFO.
	When write, not effects.

27.4.6 PCM CLOCK DIVIDE REGISTER (PCMDIV)

PCMDIV **0x00000014**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															SYNL						SYNDIV					CLKDIV					
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:27	Reserved	These bits always read as 0. Write data to these bits are ignored.	R
16:11	SYNL	Controls the length that the PCMSYN based upon the PCMCLK. The length of PCMSYN = (SYNL + 1) * PCMCLK cycle.	RW
10:6	SYNDIV	Controls the frequency of the PCMSYN signal based upon the PCMCLK. PCMSYN = PCMCLK / 8 (SYNDIV + 1).	RW
5:0	CLKDIV	PCMCLK clock divider value minus 1. Controls the divider used to create the PCMCLK based upon the CPM_PCM_SYSCLK. PCMCLK = CPM_PCM_SYSCLK / (CLKDIV + 1).	RW

27.5 PCM Interface Timing

The following figures show the timing relationship for the PCM transfers, Note in all cases. In master mode, the PCMCLK is derived from dividing the input clock, CPM_PCM_SYSCLK, and the PCMSYN is divided depended on the PCMCLK. In slave mode, the PCMCLK and PCMSYN are input from the external device. Data is sampled on the falling edge of the PCMCLK and sent out on the rising edge of the PCMCLK. The PCMSYN signal determines when the next data sample is to be transferred between the controller and the external device. Also, the PCMSYN signal as seen in the figure can be one bit time or a long bit time controlled by PCMDIV.SYNL. The PCMSYN frequency controlled by PCMDIV.SYNDIV is usually the sample rate. There are some variations controlled by PCMCFG.ISS, PCMCFG.OSS and PCMCFG.SLOT to accommodate 8 / 16bit sample sizes and multi-slot transmission.

27.5.1 Short Frame SYN

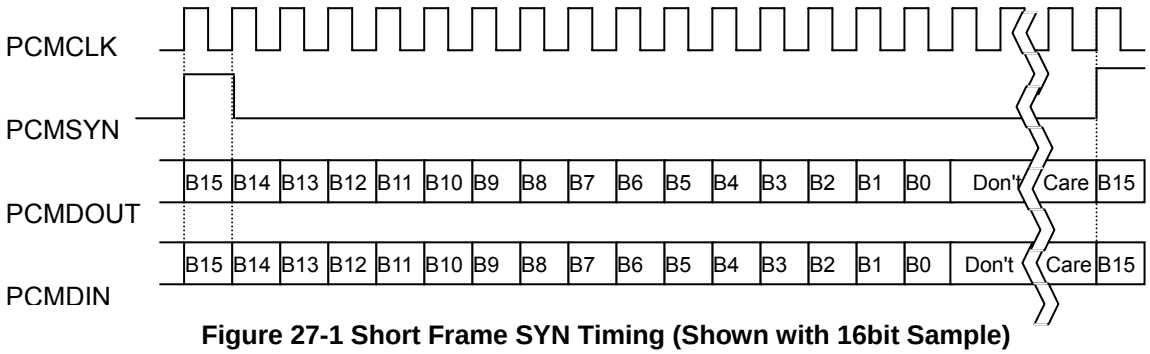


Figure 27-1 Short Frame SYN Timing (Shown with 16bit Sample)

NOTES:

1 Figure 27-1 shows a PCM transfer with the MSB configured to be coincident with the PCMSYN.

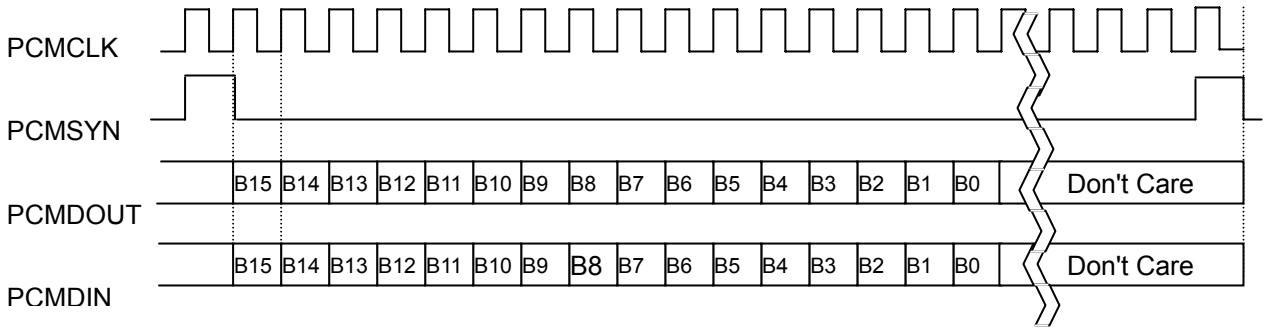


Figure 27-2 Short Frame SYN Timing (Shown with 16bit Sample)

NOTES:

1 Figure 27-2 shows a PCM transfer with the MSB configured one shift clock after the PCMSYN.

27.5.2 Long Frame SYN

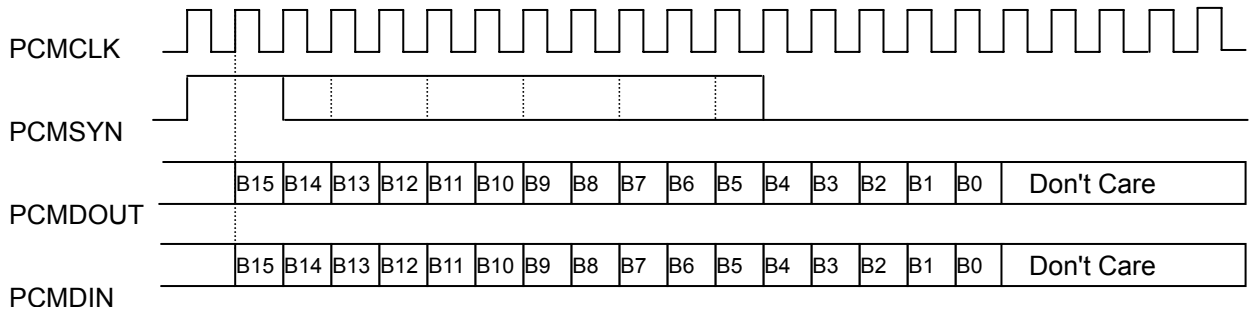


Figure 27-3 Long Frame SYN Timing (Shown with 16bit Sample)

NOTES:

1 Figure 27-3 shows a PCM transfer with the MSB configured one shift clock after the PCMSYN.

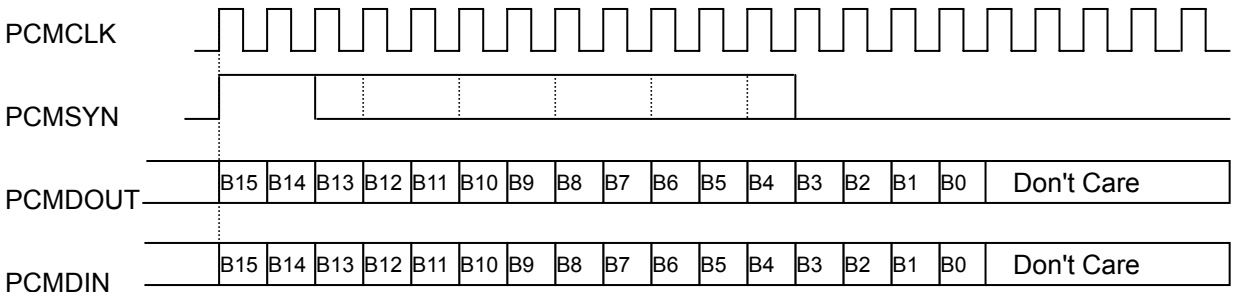


Figure 27-4 Long Frame SYN Timing (Shown with 16bit Sample)

NOTES:

1 Figure 27-4 shows a PCM transfer with the MSB configured to be coincident with the PCMSYN.

27.5.3 Multi-Slot Operation

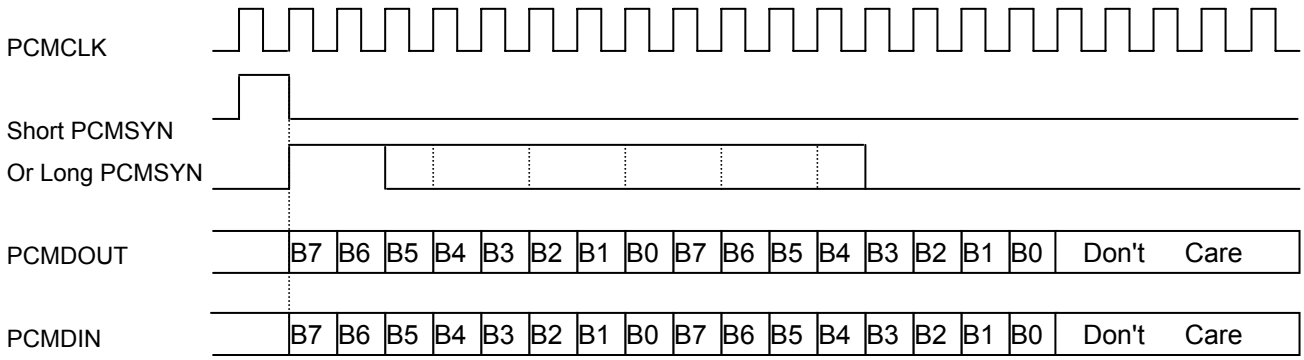
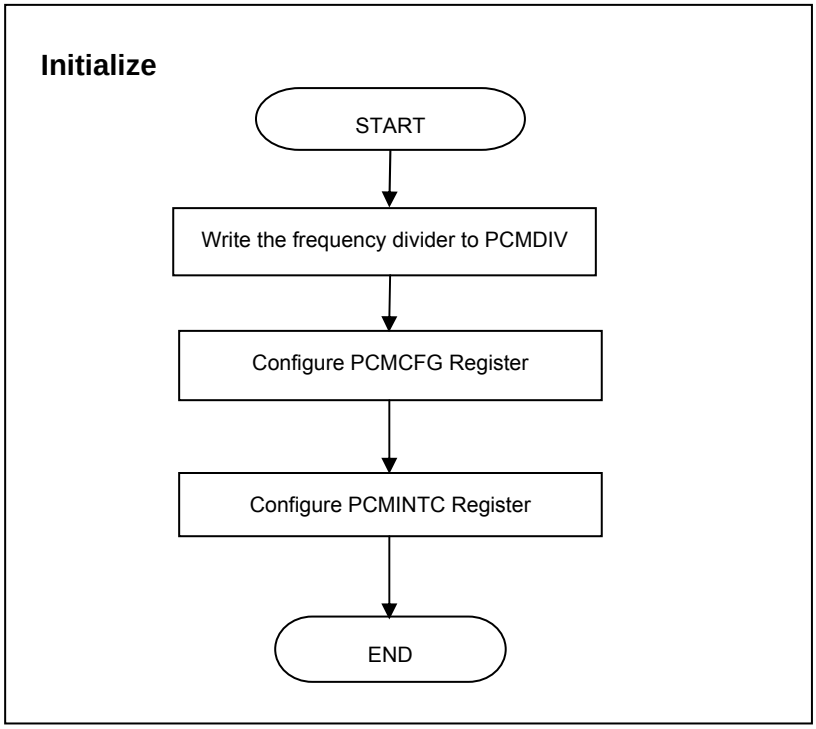


Figure 27-5 Multi-Slot Frame SYN Timing (Shown with two Slots and 8bit Sample)

27.6 PCM Operation

27.6.1 PCM Initialization

At power-on or other hardware reset (WDT and etc), PCM is disabled. Software must initiate PCM after power-on or reset.



For further details, see the corresponding register description sections.

27.6.2 Audio Replay

Outgoing audio sample data is written to PCM transmit FIFO from processor via store instruction or from memory via DMA. PCM then takes the data from the FIFO, serializes it, and sends it over the serial wire PCMDOUT to an external DEVICE.

The audio transmission is enabled automatically when the PCM is enabled by set PCMCTL.PCMEN. And PCMCTL.ERPL must be 1. If PCMCTL.ERPL is 0, value of zero is sent to external DEVICE even if there are samples in transmit FIFO. At least one audio sample data in the transmit FIFO. If the transmit FIFO is empty, value of zero or last sample depends on AICFR.LSMP, is send to external DEVICE even if PCMCTL.ERPL is 1.

Here is the audio replay flow:

1 Configure the external DEVICE as needed.
2 Initialize PCM and configure the register.
3 Write 1 to PCMCTL.PCMEN and PCMCTL.CLKEN.
4 Fill sample data to the transmit FIFO. Repeat this till finish all sample data. In this procedure,

please control the FIFO to make sure no FIFO under-run and other errors happen. When the transmit FIFO under-run, noise or pause may be heard in the audio replay, PCMINTS.TUR is 1, and if PCMINTC.ETUR is 1, PCM issues an interrupt. Please reference to 27.6.4 for detail description on FIFO.

5 Write 1 to PCMCTL.ERPL. It is suggested that at least a frame of PCM data is pre-filled in the transmit FIFO to prevent FIFO under-run flag (PCMINTS.TUR).

6 Waiting for PCMINTS.TFL change to 0. So that all samples in the transmit FIFO has been replayed, then we can have a clean start and write 0 to PCMCTL.ERPL.

27.6.3 Audio Record

Incoming audio sample data is received from PCMDIN serially and converted to parallel word and stored in PCM receive FIFO. Then the data can be taken from the FIFO to processor via load instruction or to memory via DMA.

The audio recording is enabled automatically when the PCM is enabled by set PCMCTL.PCMEN, And PCMCTL.EREC must be 1. If PCMCTL.EREC is 0, the received data is discarded even if there are rooms in the receive FIFO. At least one room left in the receive FIFO. If the receive FIFO is full, the received data is discarded even if PCMCTL.EREC is 1.

Here is the audio record flow:

1 Configure the external DEVICE as needed.
   a Initialize PCM and configure the register.
   b Write 1 to PCMCTL.PCMEN and PCMCTL.CLKEN.

2 Write 1 to PCMCTL.EREC. Make sure there are rooms available in the receive FIFO before set PCMCTL.EREC. Usually, it should empty the receive FIFO by fetch data from it before set PCMCTL.EREC.

3 Take sample data form the receive FIFO. Repeat this till the audio finished. In this procedure, please control the FIFO to make sure no FIFO over-run and other errors happen. When the receive FIFO over-run, same recorded audio samples will be lost, PCMINTS.ROR is 1, and if PCMINTC.EROR is 1, PCM issues an interrupt. Please reference to 27.6.4 for detail description on FIFO.

4 Write 0 to AICCR.EREC.

5 Take sample data from the receive FIFO until PCMINTS.RFL change to 0. So that all samples in the receive FIFO has been taken away, then we can have a clean start up next time. When the receive FIFO is empty, read from it returns zero.

27.6.4 FIFOs operation

PCM has two FIFOs, one for transmitting and one for receiving. The FIFOs are in 16 bits width and 16 entries depth, one entry for keep one sample regardless of the sample size. PCMDP.DATA provides the access point for processor/DMA to write to transmit FIFO and read from receive FIFO. One time access to PCMDP.DATA process one sample. The sample data should be put in LSB (Least Significant

Bit) in memory or processor registers. For transmitting, bits exceed sample are discarded. For receiving, these bits are set to 0. Figure 6 illustrates the FIFOs access.

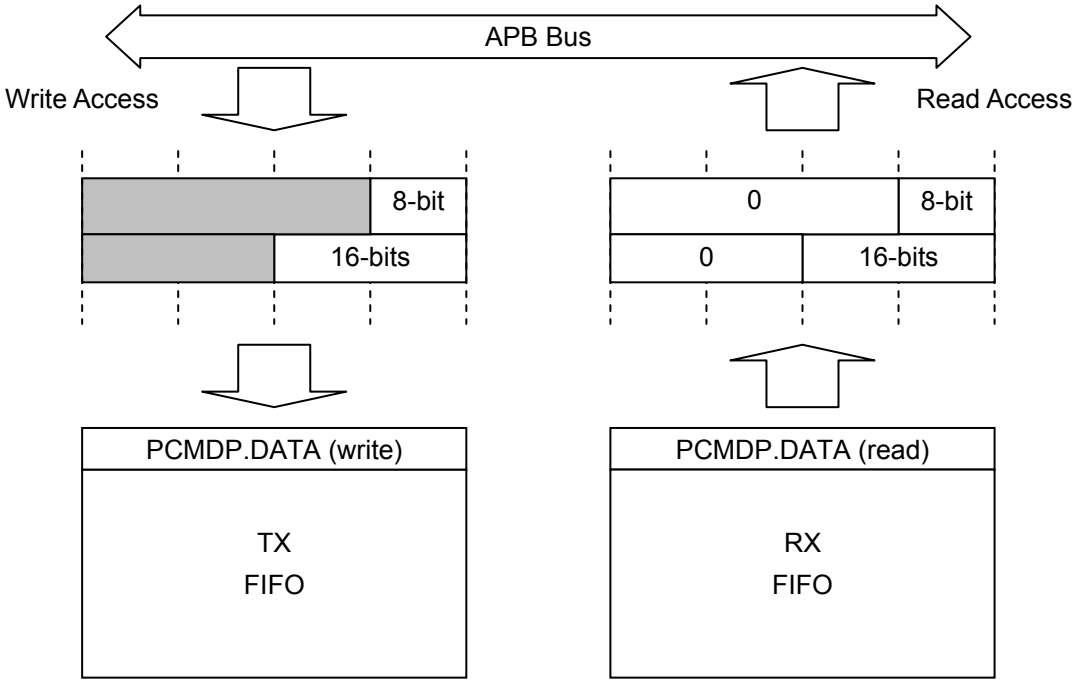


Figure 27-6 Transmitting/Receiving FIFO access via APB Bus

The software and bus initiator must guarantee the right sample placement at the bus.
In case of DMA bus initiator, One 16 bits sample occupies one 16-bits half word in memory, so 16-bits width DMA must be used. One 8-bits sample occupies one byte in memory, and use 8-bits width DMA.

27.6.5 Data Flow Control

There are three approaches provided to control/synchronize the incoming/outgoing data flow.

27.6.5.1 Polling and Processor Access

PCMINTS.RFL and PCMINTS.TFL reflect how many samples exist in receiving and transmitting FIFOs. Through read these register fields, processor can detect when there are samples in receiving FIFO and then load them from the RxFIFO, and when there are rooms in transmitting FIFO and then store samples to the TxFIFO.

Polling approach is in very low efficiency and is not recommended.

27.6.5.2 Interrupt and Processor Access

Set proper values to PCMCFG.TFTH and PCMCFG.RFTH, the FIFO interrupts trig thresholds. Set PCMINTC.ETFS and/or PCMINTC.ERFS to 1 to enable transmitting and/or receiving FIFO level trigger interrupts. When the interrupt found, it means there are rooms or samples in the TX or RX FIFO, and processor can store or load samples to or from the FIFO.

Interrupt approach is more efficient than polling approach.

27.6.5.3 DMA Access

To enable DMA operation, set PCMCTL.ERDMA and PCMCTL.ETDMA to 1 for transmit and receive respectively. It also needs to allocate two channels in DMA controller for data transmitting and receiving respectively. Please reference to DMAC spec for the details.

The PCMCFG.TFTH and PCMCFG.RFTH are used to set the transmitting and receiving FIFO level thresholds, which determine the issuing of DMA request to DMA controller. To respond the request, DMAC initiator and controls the data movement between memory and TX/RX FIFO.

27.6.6 PCM Serial Clocks and Sampling Frequencies

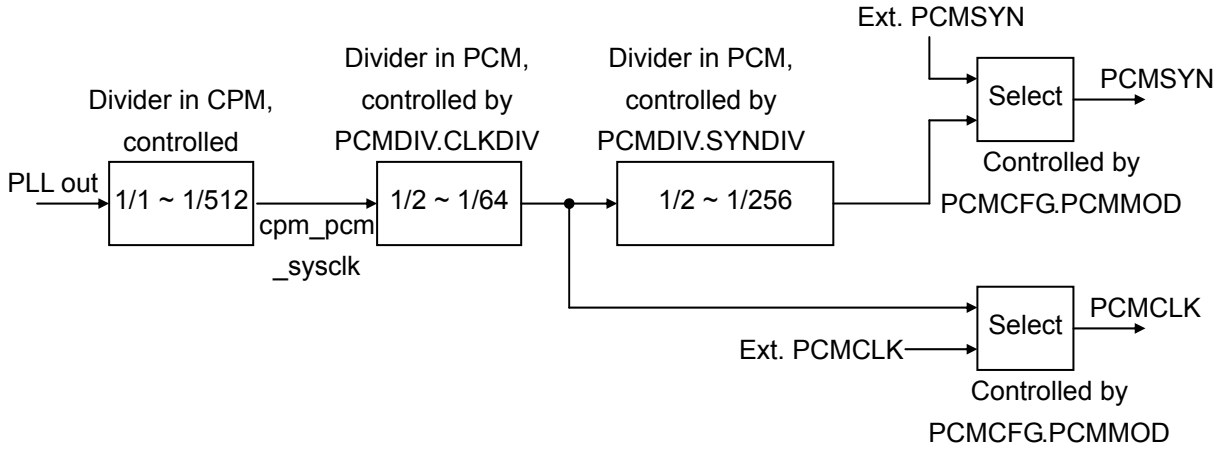


Figure 27-7 PCMCLK and PCMSYN generation scheme

27.6.7 Interrupts

The following status bits, if enabled, interrupt the processor:

- Receive FIFO Service (PCMINTS.RFS). It's also DMA Request.
- Transmit FIFO Service (PCMINTS.TFS). It's also DMA Request.
- Transmit Under-Run (PCMINTS.TUR).
- Receive Over-Run (PCMINTS.ROR).

For further details, see the corresponding register description sections.

28 SAR A/D Controller

28.1 Overview

The A/D in JZ4760 is CMOS low-power dissipation 10bit touch screen SAR analog to digital converter. It operates with 3.3/1.2V power supply. It is developed as an embedded high resolution ADC targeting to the 13nm CMOS process and has wide application in portable electronic devices, high-end home entertainment center, communication systems and so on.

The SAR A/D controller is dedicated to control A/D to work at three different modes: Touch Screen (measure pen position and pen down pressure), Battery (check the battery power), and three auxiliary input. Touch Screen can transfer the data to memory through the DMA or CPU. Battery and three auxiliary inputs can transfer the data to memory through CPU.

Features:

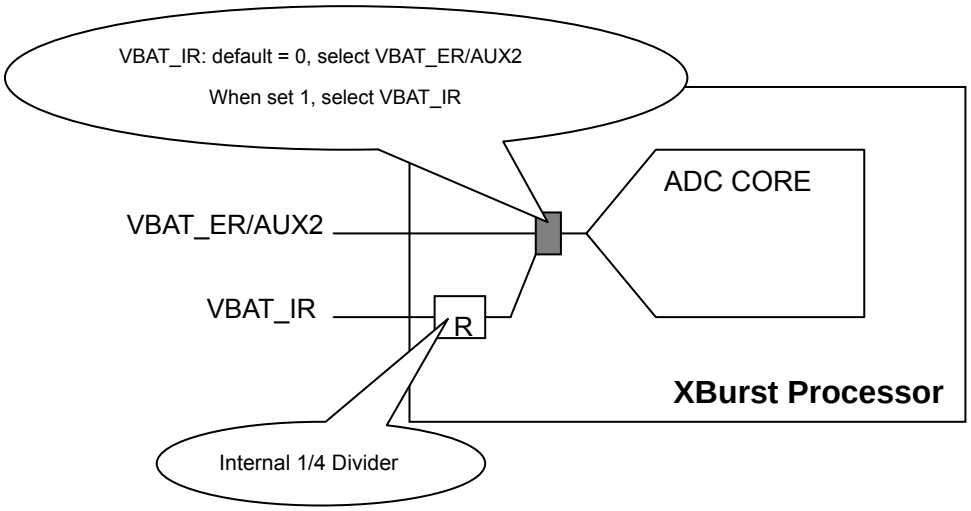
- 8 Channels
- Resolution: 12-bit
- Integral nonlinearity: ±2 LSB
- Differential nonlinearity: ±4 LSB
- Resolution/speed: up to 1Msps
- Max Frequency: 200k
- Low power dissipation: 1mA(worst)
- Support touch screen measurement (Through pin XP, XN, YP, YN)
- Support voltage measurement (Through pin VBAT_IR, VBAT_ER/AUX2)
- Support three auxiliary input (Through pin AUX, AUX1, VBAT_ER/AUX2)
- Separate Channel Conversion Mode
- Single-end and Differential Conversion Mode
- Auto X/Y and X/Y/Z1/Z2 position measurement

28.2 Pin Description

Table 28-1 SADC Pin Description

Name	I/O	Description
XN	AI	Touch screen analog differential X- position input
YN	AI	Touch screen analog differential Y- position input
XP	AI	Touch screen analog differential X- position input
YP	AI	Touch screen analog differential Y- position input
VBAT_IR	AI	Analog input for VBAT measurement using internal resistors *
AUX	AI	Auxiliary Input
AUX1	AI	Auxiliary Input
VBAT_ER/AUX2	AI	VBAT direct input * / Auxiliary Input

* Users who already deployed divider resistors on board level can use VBAT_ER/AUX2 to direct measure the battery value.
Also chip provides a build-in 1/4 divider for VBAT measurement on VBAT_IR pin. To measure battery using internal resistor network, please write register: CPM.VBAT_IR and connect Battery output to VBAT_IR pin. Details about register location please refer CPM spec.



28.3 Register Description

In this section, we will describe the registers in SAR A/D controller. Following table lists all the register definitions. All registers' 32bit addresses are physical addresses. And detailed function of each register will be described below.

Table 28-2 SADC Register Description

Name	Description	RW	Reset Value	Address	Access Size
ADENA	ADC Enable Register	RW	0x00	0x10070000	8
ADCFG	ADC Configure Register	RW	0x0002000C	0x10070004	32
ADCTRL	ADC Control Register	RW	0x3F	0x10070008	8
ADSTATE	ADC Status Register	RW	0x00	0x1007000C	8
ADSAME	ADC Same Point Time Register	RW	0x0000	0x10070010	16
ADWAIT	ADC Wait Time Register	RW	0x0000	0x10070014	16
ADTCH	ADC Touch Screen Data Register	RW	0x00000000	0x10070018	32
ADVDAT	ADC VBAT Data Register	RW	0x0000	0x1007001C	16
ADADAT	ADC AUX Data Register	RW	0x0000	0x10070020	16
ADCLK	ADC Clock Divide Register	RW	0x00000000	0x10070028	32
ADFLT	ADC Filter Register	RW	0x0000	0x10070024	16

28.3.1 ADC Enable Register (ADENA)

The register ADENA is used to trigger A/D to work.

ADENA **0x10070000**

Bit	7	6	5	4	3	2	1	0
	POWDN	SLP_MD	Reserved			TCHEN	VBATEN	AUXEN
RST	1	0	0	0	0	0	0	0

Bits	Name	Description	RW
7	POWDN	SADC Power control bit. 1: SADC power down 0: SADC power on When POWDN is set to 0, wait at least 50ms to enable SADC.	RW
6	SLP_MD	SLEEP Mode Control. 1:Enter sleep mode 0:Exit sleep mode	RW
5:3	Reserved	These bits always read 0, and written are ignored.	R
2	TCHEN	Touch Screen Enable Control. 0: disable 1: enable	RW
1	VBATEN	VBAT Enable Control. No matter TCHEN is 1 or 0, VBATEN can be set to 1 to sample the voltage of battery, and when the value of voltage is ready, PBATEN will be cleared by hardware auto.	RW
0	AUXEN	AUX n Enable Control. No matter TCHEN is 1 or 0, AUXEN can be set to 1 to sample the voltage of AUX, AU1 or VBAT_ER/AUX2, and when the value of voltage is ready. AUXEN will be cleared by hardware auto.	RW

NOTES:

1 TCHEN, VBATEN and AUXEN can be set to 1 at the same time. The priority of the three mode is AUX > VBAT > TCH.

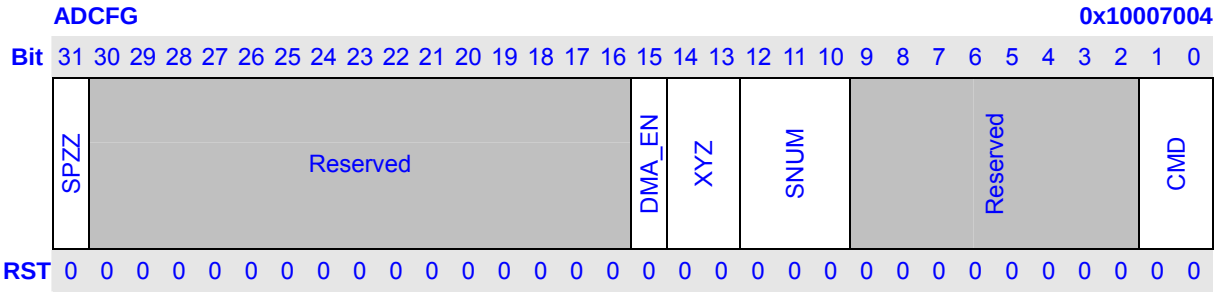
2 SLP_MD, TCHEN can be set to 1 at the same time. The priority of the three mode is SLP_MD > TCH.

3 When VBATEN and AUXEN are all 0, SLP_MD can be set to 1.

4 When VBAT_ER/AUX2 is used as an AUX input, use AUXEN to start measure.
When VBAT_ER/AUX2 is used as a battery input, use VBATEN to start measure.

28.3.2 ADC Configure Register (ADCFG)

The register ADCFG is used to configure the A/D.



Bits	Name	Description	RW
31	SPZZ*1	The $X_dY_dZ_mZ_n$ of different point measure can be different. But the $X_dY_dZ_mZ_n$ of the same point measure can be same or different. 0: The $X_dY_dZ_mZ_n$ of the same point measure is all the same (X_dY_dZ1Z2, X_dY_dZ1Z2, X_dY_dZ1Z2, X_dY_dZ1Z2 … X_dY_dZ1Z2) 1: The $X_dY_dZ_mZ_n$ of the same point measure maybe different (X_dY_dZ1Z2, X_dY_dZ3Z4, X_dY_dZ3Z4, X_dY_dZ1Z2 … X_dY_dZ1Z2)	RW
30:16	Reserved	These bits always read 0, and written are ignored.	R
15	DMA_EN	When A/D is used as Touch Screen (CMD=1100), DMA_EN is used as follows: 0: The sample data is read by CPU 1: The sample data is read by DMA	RW
14:13	XYZ	When A/D is used in Touch Screen mode, XYZ is used as follows: XYZ | Measure 00 | $X_s \rightarrow Y_s$ 01 | $X_d \rightarrow Y_d$ 10 | $X_d \rightarrow Y_d \rightarrow Z1_d \rightarrow Z2_d$ or $X_d \rightarrow Y_d \rightarrow Z3_d \rightarrow Z4_d$ 11 | Reserved	RW
12:10	SNUM	The number of repeated sampling one point. When A/D is used as Touch Screen (CMD=1100), SNUM is used as follows: SNUM | Number 000 | 1 001 | 2 010 | 3 011 | 4 100 | 5 101 | 6 110 | 8 111 | 9	RW
9:2	Reserved	These bits always read 0, and written are ignored.	R

<table>
<tr>
<td>1:0</td>
<td>CMD</td>
<td>CMD is used to choose the current sample command when ADENA.AUXEN is set to 1.
<table>
<tr><th>CMD</th><th>Function</th></tr>
<tr><td>00</td><td>Measure AUX voltage</td></tr>
<tr><td>01</td><td>Measure AUX1 voltage</td></tr>
<tr><td>10</td><td>Measure VBAT_ER/AUX2 voltage</td></tr>
<tr><td>11</td><td>Reserved</td></tr>
</table>
</td>
<td>RW</td>
</tr>
</table>

NOTES:

1 [*1]: X_s, Y_s means the reference mode of X, Y is single-end mode.
X_d, Y_d, $Z1_d$, $Z2_d$, $Z3_d$, Z4d means the reference mode of X, Y, Z1, Z2, Z3, Z4 is differential mode.
When you measure Xs you need to make sure that X-plate is driven by external DC power.
When you measure Ys you need to make sure that Y-plate is driven by external DC power.

28.3.3 ADC Control Register (ADCTRL)

The register ADCTRL is used to control A/D to work.

ADCTRL **0x10070008**

Bit	7	6	5	4	3	2	1	0
	Reserved		SLPENDM	PENDM	PENUM	DTCHM	VRDYM	ARDYM
RST	0	0	1	1	1	1	1	1

Bits	Name	Description	RW
7:6	Reserved	These bits always read 0, and written are ignored.	R
5	SLPENDM	In SLEEP mode pen down interrupt mask. 0: enabled 1: masked	RW
4	PENDM	Pen down interrupt mask. 0: enabled 1: masked	RW
3	PENUM	Pen up interrupt mask. 0: enabled 1: masked	RW
2	DTCHM	Touch Screen Data Ready interrupt mask. 0: enabled 1: masked	RW
1	VRDYM	VBAT data ready interrupt mask. 0: enabled 1: masked	RW
0	ARDYM	AUX data ready interrupt mask. 0: enabled 1: masked	RW

28.3.4 ADC Status Register (ADSTATE)

The register ADSTATE is used to keep the status of A/D.

ADSTATE **0x1007000C**

Bit	7	6	5	4	3	2	1	0
	SLP_RDY	Reserved	SLPEND	PEND	PENU	DTCH	VRDY	ARDY
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7	SLP_RDY	Sleep state bit. 1:The set of sleep mode is ready 0:The set of sleep mode is not ready	R
6	Reserved	These bits always read 0, and written are ignored.	R
5	SLPEND	In SLEEP mode pen down interrupt flag. Write 1 will clear this bit. 1: active 0: not active	RW
4	PEND	Pen down interrupt flag. Write 1 will clear this bit. 1: active 0: not active	RW
3	PENU	Pen up interrupt flag. Write 1 will clear this bit. 1: active 0: not active	RW
2	DTCH	Touch screen data ready interrupt flag. Write 1 will clear this bit. 1: active 0: not active	RW
1	VRDY	VBAT data ready interrupt flag. Write 1 will clear this bit. 1: active 0: not active	RW
0	ARDY	AUX data ready interrupt flag. Write 1 will clear this bit. 1: active 0: not active	RW

28.3.5 ADC Same Point Time Register (ADSAME)

The register ADSAME is used to store the interval time between repeated sampling the same point. The clock of the counter is clk_us.

ADSAME **0x10070010**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SCNT															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.3.6 ADC Wait Pen Down Time Register (ADWAIT)

The register ADWAIT is used to store the interval time of wait pen down. And the register can be used as the interval time among the different point. The clock of the counter is clk_ms.

ADWAIT **0x10070014**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	WCNT															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

28.3.7 ADC Touch Screen Data Register (ADTCH)

The read-only ADTCH is corresponded to 2x32 bit FIFO, it keep the sample data for touch screen. 0~11 bits are data, 15 bit is data type. 16~27 bits are data, 31 bit is data type. When write to the register, DATA will be clear to 0.

ADTCH **0x10070018**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TYPE1	Reserved			TDATA1												TYPE0	Reserved			TDATA0											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	TYPE1	Type of the Touch Screen Data1. When A/D is used as Touch Screen, ADCFG.XYZ=10. TYPE1=1: X_d→Y_d→Z1→Z2 TYPE1=0: X_d→Y_d→Z3→Z4 When A/D is used as Touch Screen, ADCFG.XYZ=00 or XYZ=01, TYPE1=0.	RW
30:28	Reserved	These bits always read 0, write operation will be ignored.	R
27:16	TDATA1	The concert data of touch screen A/D.	RW
15	TYPE0	Type of the Touch Screen Data2. When A/D is used as Touch Screen, ADCFG.XYZ=10. TYPE0=1: X_d→Y_d→Z1→Z2 TYPE0=0: X_d→Y_d→Z3→Z4 When A/D is used as Touch Screen, ADCFG.XYZ=00 or XYZ=01, TYPE0=0.	RW
14:12	Reserved	These bits always read 0, write operation will be ignored.	R
11:0	TDATA0	The concert data of touch screen A/D.	RW

NOTES:

1 When A/D is used as Touch Screen, ADCFG.XYZ=00.
The format of touch screen data is as follows:

Type1	Reserved	Data1	Type0	Reserved	Data0
0	000	Y_s	0	000	X_s

2 When A/D is used as Touch Screen, ADCFG.XYZ=01.
The format of touch screen data is as follows:

Type1	Reserved	Data1	Type0	Reserved	Data0
0	000	Y_d	0	000	X_d

3 When A/D is used as Touch Screen, ADCFG.XYZ=10.TYPE=1.
The format of touch screen data is as follows:

Type1	Reserved	Data1	Type0	Reserved	Data0
1	000	Y_d	1	000	X_d
1	000	$Z2_d$	1	000	$Z1_d$

Users need to read twice to get the whole data. The first time reading gets the data Y_d and X_d. The second time reading gets the data $Z2_d$ and $Z1_d$.

The touch pressure measurement formula is as follows: (You can use formula 1 or formula 2.)

$$R_{TOUCH} = R_{X\text{-}Plate} \bullet \frac{X\text{-}Position}{4096}\left(\frac{Z_2}{Z_1} - 1\right) \quad (1)^{*1}$$

$$R_{TOUCH} = \frac{R_{X\text{-}Plate} \bullet X\text{-}Position}{4096}\left(\frac{4096}{Z_1} - 1\right) - R_{Y\text{-}Plate} \bullet \left(1 - \frac{Y\text{-}Position}{4096}\right) \quad (2)^{*1}$$

4 When A/D is used as Touch Screen, ADCFG.XYZ=10.TYPE=0.

The format of touch screen data is as follows:

Type1	Reserved	Data1	Type0	Reserved	Data0
0	000	Y_d	0	000	X_d
0	000	$Z4_d$	0	000	$Z3_d$

Users need to read twice to get the whole data. The first time reading gets the data Y_d and X_d. The second time reading gets the data $Z4_d$ and $Z3_d$.

The touch pressure measurement formula is as follows: (You can use formula 3 or formula 4.)

$$R_{TOUCH} = R_{Y\text{-}Plate} \bullet \frac{Y\text{-}Position}{4096}\left(\frac{Z_4}{Z_3} - 1\right) \quad (3)^{*1}$$

$$R_{TOUCH} = \frac{R_{Y\text{-}Plate} \bullet Y\text{-}Position}{4096}\left(\frac{4096}{Z_3} - 1\right) - R_{X\text{-}Plate} \bullet \left(1 - \frac{X\text{-}Position}{4096}\right) \quad (4)^{*1}$$

NOTES:

1 *1: To determine pen or finger touch, the pressure of the touch needs to be determined. Generally, it is not necessary to have very high performance for this test; therefore, the 8-bit resolution mode is recommended (however, calculations will be shown here are in 10bit resolution mode).

$R_{X\text{-}plate}$: Total X-axis resistor value (about 200Ω~ 600Ω)

$R_{Y\text{-}plate}$: Total Y-axis resistor value (about 200Ω~ 600Ω)

X-Position: X-axis voltage sample value

Y-Position: Y-axis voltage sample value

Z1, Z2: Z1, Z2 voltage sample value

Z3, Z4: Z3, Z4 voltage sample value

28.3.8 ADC VBAT Data Register (ADVDAT)

The read-only ADVDAT is a 16-bit register, it keep the sample data of VBAT. 0~11 bits are data.

ADBDAT																0x1007001C
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				VDATA											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15:12	Reserved	These bits always read 0, write operation will be ignored.	R
11:0	VDATA	Data of VBAT A/D convert. When write to the register, DATA will be clear to 0.	RW

There are 2 pins available for V_{BAT} voltage measure, selected by register CPM.VBAT_IR:

VBAT_ER/AUX2:

Users who already deployed resistor divider on board level can direct measure on this pin.

Select this mode by CPM.VBAT_IR = 0.

$$V_{BAT\text{-}ER} = \frac{VDATA}{4096 \bullet [external\_divider\_ratio]} \bullet 2.5V$$

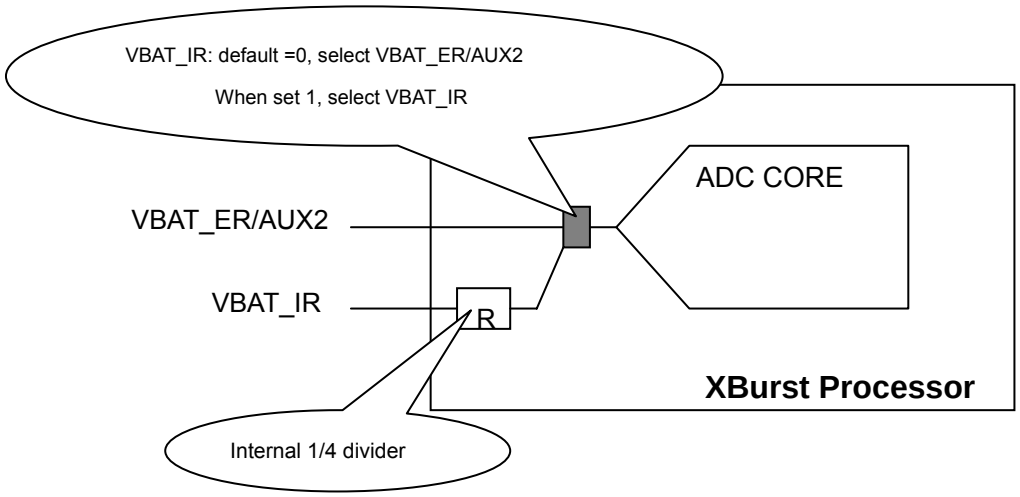
VBAT_IR:

Users who use chip build-in resistor divider (ratio = 1/4) should measure on this pin.

Select this mode by CPM.VBAT_IR = 1.

$$V_{BAT\text{-}IR} = \frac{VDATA}{1024} \bullet 2.5V$$

Details about register CPM.VBAT_I location please refer to CPM spec.



28.3.9 ADC AUX Data Register (ADADAT)

The read-only ADADAT is a 16-bit register, it keep the sample data. 0~11 bits are data.

ADSDAT **0x10070020**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				ADATA											
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15:12	Reserved	These bits always read 0, write operation will be ignored.	R
11:0	ADATA	Data of AUX. When write to the register, DATA will be clear to 0.	RW

The measured voltage V_{AUX} (V_{AUX}, V_{AUX1} and $V_{VBAT\_ER/AUX2}$) is as follows:

$$V_{SAD} = \frac{ADATA}{4096} \bullet 3.3V$$

28.3.10 ADC Clock Divide Register (ADCLK)

The register ADCLK is used to set the A/D's clock dividing number.

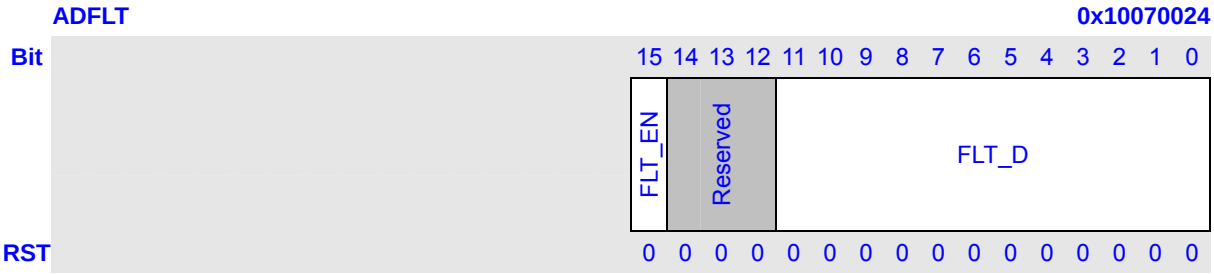
ADCLK **0x10070028**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CLKDIV_MS																CLKDIV_US								CLKDIV							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:16	CLKDIV_MS	Dividing number to get ms clock from ADC clock. ms_clk = us_clk / (CLK_MS +1).	RW
15:8	CLKDIV_US	Dividing number to get us clock from ADC clock. us_clk = adc_clk / (CLKDIV_US+1). 0 ≤CLKDIV_10 ≤127.	RW
7:0	CLKDIV	Dividing number to get ADC clock from device clock. The A/D works at the frequency between 20KHz and 200KHz. If CLKDIV = N, Then the freq of adc_clk = dev_clk / (N+1). 0 ≤ N ≤ 255.	RW

28.3.11 ADC Filter Register (ADFLT)

ADC Filter Register ADFLT is used for filter out the no valid point for Touch Screen control.



Bits	Name	Description	RW
15	FLT_EN	Filter enable bit. 1: Filter function enable 0: Filter function disable	RW
14:12	Reserved	These bits always read 0, write operation will be ignored.	R
11:0	FLT_D	Filter Data.	RW

NOTES:

1 When ADFLT.FLT_EN is set to 1,

If (|Z2-Z1|> ADFLT.FLT_D), X=Y=Z1=Z2=0;

If (|Z4-Z3|> ADFLT.FLT_D), X=Y=Z3=Z4=0;

28.4 SAR A/D Controller Guide

The following describes steps of using SAR-ADC.

28.4.1 Power Down Mode

1 Initial value of ADENA.POWDN == 1, SADC is in power down state.
2 Exit Power Down mode:
Before issue control to SADC, set ADENA.POWDN = 0 to power on SADC.
Wait for at least 50ms for SADC exiting power down mode.
3 Enter Power Down mode:
Disable Touch Screen, VBAT and AUX first, then set ADENA.POWDN = 1.

28.4.2 SLEEP mode

1 If the register ADCLK has not been set before, set ADCLK.CLKDIV, ADCLK.CLKDIV_US and ADCLK.CLKDIV_MS to set A/D clock frequency.
2 DSTATE.SLP_RDY = 0, then ADENA.SLP_MD = 1.
When ADSTATE.SLP_RDY == 1, the Touch Screen has entered the SLEEP mode.
3 Enable “in SLEEP mode pen down interrupt” and mask all other interrupts:
ADSTATE.SLPEND = 0, ADCTRL.SLPENDM = 0.
4 When “in SLEEP mode pen down interrupt” happened:
a Close “in SLEEP mode pen down interrupt”:
ADCTRL.SLPENDM = 1, ADSTATE.SLPEND = 0.
b Switch from SLEEP to NORMAL:
ADSTATE.SLP_RDY = 0, ADENA.SLP_MD = 0.
When ADSTATE.SLP_RDY == 1, the Touch Screen has exited the SLEEP mode.
5 Execute Touch Screen / VBAT / AUX operations at normal mode.

28.4.3 VBAT Sample Operation

1 Set ADCLK.CLKDIV, ADCLK.CLKDIV_US and ADCLK.CLKDIV_MS to set A/D clock frequency.
2 If use pin VBAT_ER/AUX2 with onboard voltage divider, CPM.VBAT_IR=0;
Else if use pin VBAT_IR, CPM.VBAT_IR = 1.
3 ADENA.VBATEN = 1, to enable the channel.
4 When ADSTATE.VRDY == 1, read the sample data from ADVDAT.
Then the VBATEN will be auto set to 0.

28.4.4 AUX Sample Operation

1 Set ADCFG.CMD to choose one CMD. (AUX, AUX1 or VBAT_ER/AUX2).
2 Set ADCLK.CLKDIV, ADCLK.CLKDIV_US and ADCLK.CLKDIV_MS to set A/D clock frequency.

3 ADENA.AUXEN = 1 to enable the channel.

4 When ADSTATE. ARDY == 1, read the sample data from ADADAT.
Then the AUXEN will be auto set to 0.

28.4.5 A Sample Touch Screen Operation

(Pen Down → Sample some data of several points → Pen Up)

1 ADCTRL = 0x1f to mask all the interrupt of SADC.

2 DMA_EN = 1 to choose DMA or CPU to read the sample data out.

3 Set ADCFG.SPZZ and ADCFG.XYZ to choose sample mode.

a X_s→Y_s (Single-end X→Single-end Y).

b X_d→Y_d (Differential X→Differential Y).

c X_d→Y_d→$Z1_d$→$Z2_d$ or X_d→Y_d→$Z3_d$→$Z4_d$ (Reference register ADCFG.SPZZ)
(Differential X→Differential Y→Differential Z1→Differential Z2 or
Differential X→Differential Y→Differential Z3→Differential Z4).

4 Set ADCLK.CLKDIV, ADCLK.CLKDIV_US and ADCLK.CLKDIV_MS to set A/D clock frequency.

5 Set ADWAIT to decide the wait time of pen down and the interval time between sampling different points. This time delay is necessary because when pen is put down or pen position change, there should be some time to wait the pen down signal to become stable.

6 Set ADSAME to decide the interval time between repeated sampling the same point.
User can repeat sampling one point to get the most accurate data.

7 ADCTRL.PENDM = 0 to enable the pen down interrupt of touch panel.

8 ADENA.TCHEN = 1 to start touch panel.

9 When pen down interrupt happened:
ADCTRL.PENDM = 1, ADSTATE.PEND = 0 to close pen down interrupt.
ADSTATE.PENDU = 0, ADCTRL.PENUM = 0 to enable pen up interrupt.

10 When pen down interrupt happened, the SAR ADC is sampling data.
When ADSTATE.DTCH == 1, read the sample data from ADTCH.
The SAR ADC will not sample the next point until the whole data of the one point are read (no matter by CPU or DMA). If ADCFG.XYZ is mode one and mode two, user only needs to read once to get the whole data. In other modes, user needs to read twice to get the whole data.

11 Repeat (10) till pen up interrupt happened.

12 When pen up interrupt happened, enable pen down interrupt:
ADCTRL.PENUM = 1, ADSTATE.PENU = 0, Then ADSTATE.PEND = 0, ADCTRL.PENDM = 0.

13 Wait pen down interrupt and repeat from (9).

14 ADENA.TCHEN = 0 to shut down the touch screen at anytime.
If the last point is not sampled completely, user can abandon it.

28.4.6 Disable Touch Screen

1 The initial condition is that ADENA.TCHEN == 1, ADENA.VBATEN == 0, ADENA.AUXEN == 0.
2 set ADENA.TCHEN = 0.
3 Read ADENA.TCHEN till it is set to 0 by hardware, then Touch Screen is fully disabled.

28.4.7 Use TSC to support keypad

SADC TSC function can apply to a keypad, if touch screen is not used. Suppose the keypad is a NxM matrix, where X direction has N key columns and Y direction has M key rows. Kij is used to indicate the key in ith column from left to right and jth row from bottom to top, where i=0~(N-1) and j=0~(M-1). Figure 28-1 is a 6x5 keypad circuit. The blue color is for X direction network and pink color is for Y. The networks are composed by resistors and metal line. These two networks should be connected to SADC 4 pins: XP/XN/YP/YN as illustrated in the figure. The gray circle is the key. When no key pressing, X network and Y network is open circuit. When a key is pressed, the X network and Y network is shorted under the key position.

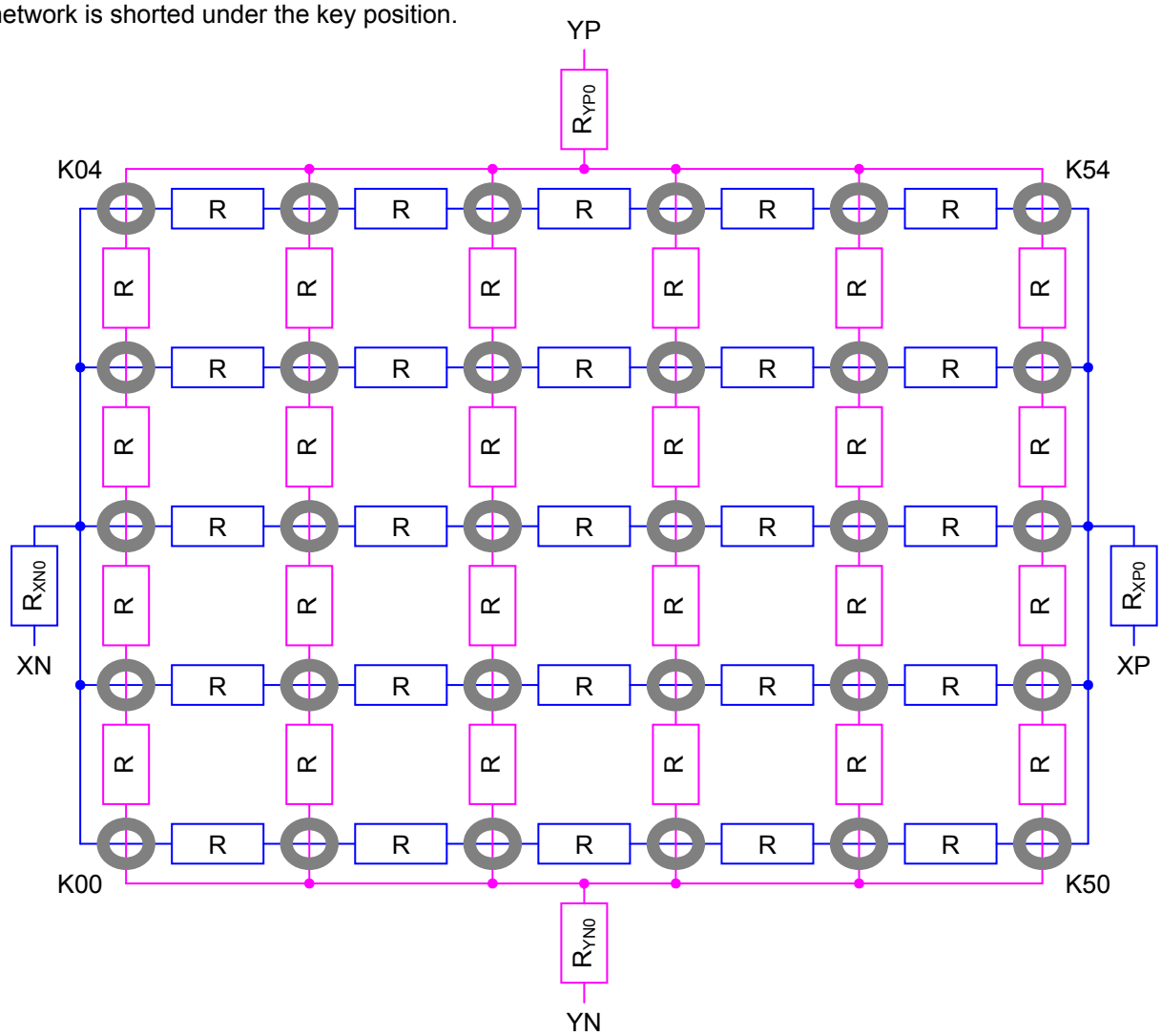


Figure 28-1 6x5 keypad circuit

When SADC is in waiting for pen-down status (C=1100), the equivalent circuit is show in Figure 28-2. When the key is not pressed, XP is open and the PEN is pulled to VDDADC, which is logic 1. When the key Kij is pressed, the circuit is: VDDADC→(10kΩ resistor)→R_{XP}→R_{YN}→VSSADC.

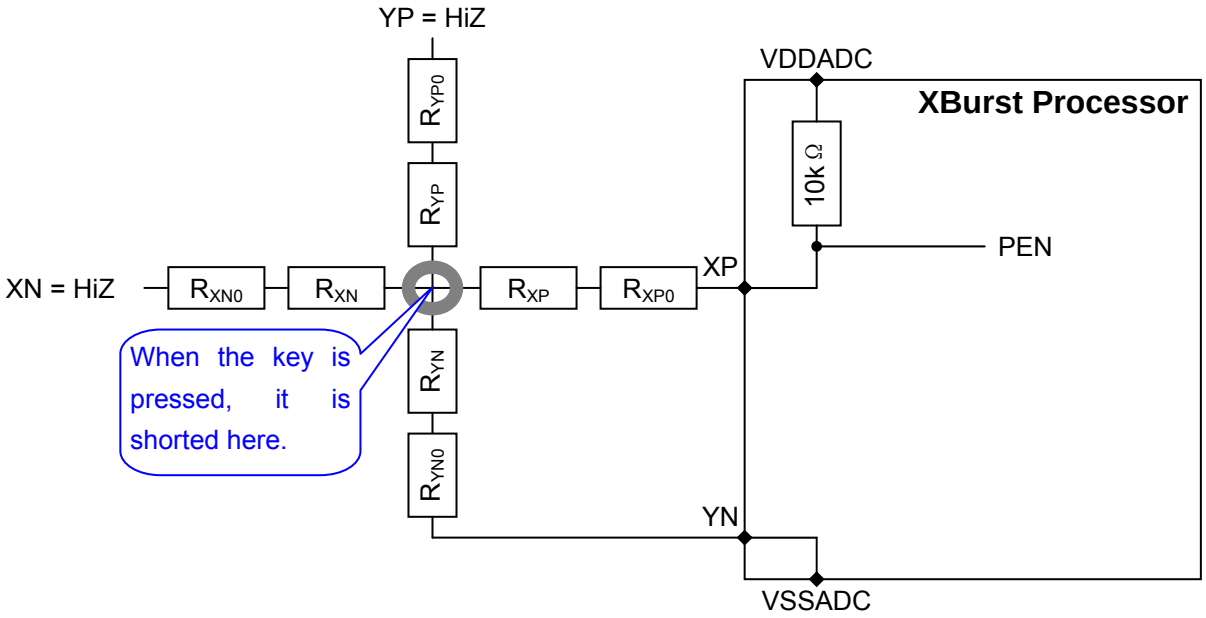


Figure 28-2 Wait for pen-down (C=1100) circuit

Where

$$R_{XP} = \frac{(N-1)^2 - i^2}{M \times (N-1-i) + 2i} \times R$$

$$R_{YN} = \frac{j \times (2M-2-j)}{N \times j + 2M - 2 - 2j} \times R$$

To ensure logic 0 at PEN in this case, following formula should be obeyed.

$$R_{XP} + R_{YN} + R_{XP0} + R_{YN0} \le 3k\Omega \quad (1)$$

It is suggested the value of N and M is as close to each other as possible. For N=2~20, M=2~20 and M=(N-1, N or N+1), we found

$$R_{XP} + R_{YN} < 2.7 \times R \quad (2)$$

After key pressing is found, the key Kij location, columns and row, should be measured by using C=0010 and C=0011 respectively. The equivalent circuits are show in Figure 28-3 and Figure 28-4, where

$$R_{X0} = \frac{N-1}{M-1} \times R$$

$$R_{Y0} = \frac{M-1}{N-1} \times R$$

$$R_{XNi} = i \times R$$

$$R_{XPi} = (N-1-i) \times R$$

$$R_{YNj} = j \times R$$

$$R_{YPj} = (M-1-j) \times R$$

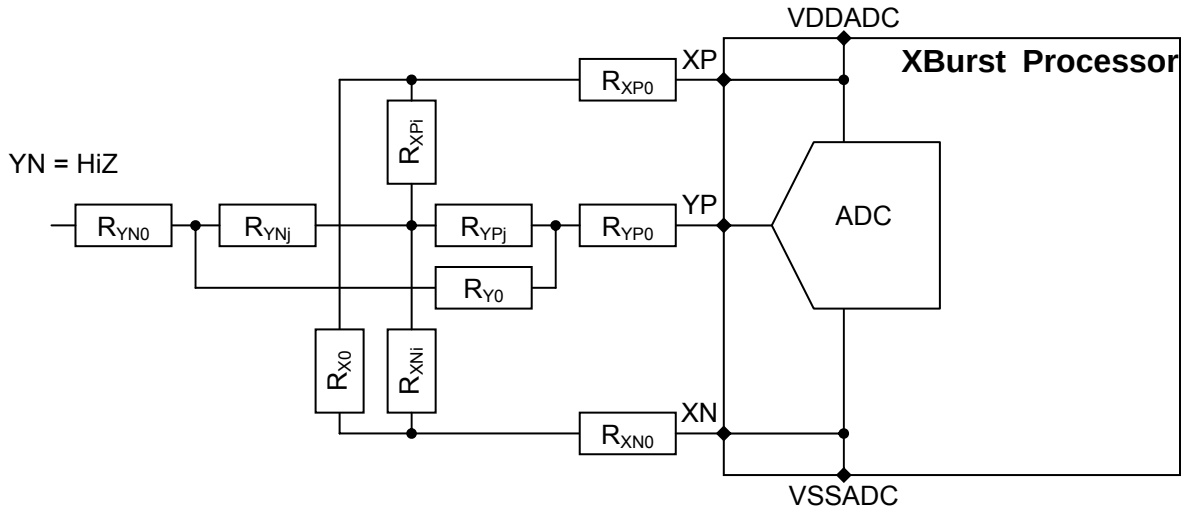


Figure 28-3 Measure X-position (C=0010) circuit

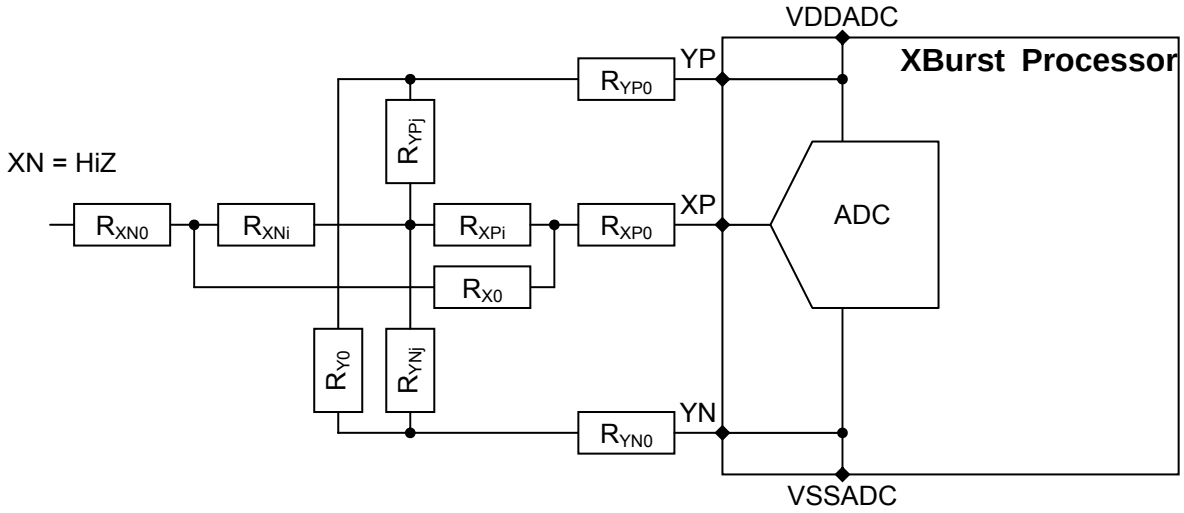


Figure 28-4 Measure Y-position (C=0011) circuit

So for Kij pressing, we should get ADC converted number Ni and Nj for i and j respectively.

$$Ni = \frac{R_{XN0} + \frac{i}{M}R}{R_{XN0} + \frac{N-1}{M}R + R_{XP0}} \times 4096$$

$$Nj = \frac{R_{YN0} + \frac{j}{N}R}{R_{YN0} + \frac{M-1}{N}R + R_{YP0}} \times 4096$$

It is required the resistor between XP and XN in case of C=0010, between YP and YN in case of C=0011, must be ≥200Ω and it better be ≥500Ω. Also consider the requirement in formula (1) and (2) above, we suggest to put R_{XP0} = R_{XN0} = R_{YP0} = R_{YN0} = 50Ω or 100Ω, put R = 500Ω ~ 1kΩ.

To use the keypad, the software should set:

ADENA.TCHEN = 1

ADCFG.XYZ = 10

The operation is similar to touch screen.

29 General-Purpose I/O Ports

29.1 Overview

General Purpose I/O Ports (GPIO) is used in generating and capturing application-specific input and output signals. Each port can be programmed as an output, an input or function port that serves certain peripheral. As input, pull up/down can be enabled/disabled for the port and the port also can be configured as level or edge tripped interrupt source.

Features:

- Each port can be configured as an input, an output or an alternate function port
- Each port can be configured as an interrupt source of low/high level or rising/falling edge triggering. Every interrupt source can be masked independently
- Each port has an internal pull-up or pull-down resistor connected. The pull-up/down resistor can be disabled
- GPIO output 6 interrupts, 1 for every group, to INTC

29.1.1 GPIO Port A Summary

Table 29-1 GPIO Port A summary

Bit N	PA N	Pull (U/D)	Shared Function Port Selected by				
			0	1	2	3	Note
0	00	U	sd0(io)	-	-	-	
1	01	U	sd1(io)	-	-	-	
2	02	U	sd2(io)	-	-	-	
3	03	U	sd3(io)	-	-	-	
4	04	U	sd4(io)	-	-	-	
5	05	U	sd5(io)	-	-	-	
6	06	U	sd6(io)	-	-	-	
7	07	U	sd7(io)	-	-	-	
8	08	U	sd8(io)	-	-	-	
9	09	U	sd9(io)	-	-	-	
10	10	U	sd10(io)	-	-	-	
11	11	U	sd11(io)	-	-	-	
12	12	U	sd12(io)	-	-	-	
13	13	U	sd13(io)	-	-	-	
14	14	U	sd14(io)	-	-	-	
15	15	U	sd15(io)	-	-	-	
16	16	U	rd_(o)	-	-	-	
17	17	U	we_(o)	-	-	-	
18	18	U	fre_(o)	msc0_clk(o)	ssi0_clk(o)	-	
19	19	U	fwe_(o)	msc0_cmd(io)	ssi0_ce0_(o)	-	
20	20	U	msc0_d0(io)	ssi0_dr(i)	-	-	1
21	21	U	cs1_(o)	msc0_d1(io)	ssi0_dt(o)	-	
22	22	U	cs2_(o)	msc0_d2(io)	-	-	
23	23	U	cs3_(o)	msc0_d3(io)	-	-	
24	24	U	cs4_(o)	-	-	-	
25	25	U	cs5_(o)	-	-	-	
26	26	U	cs6_(o)	-	-	-	
27	27	U	wait_(i)	-	-	-	
28	28	U	dreq0(i)	-	-	-	
29	29	U	dack0(o)	owi(io)	-	-	
30	30	U	-	-	-	-	6
31	31	U	-	-	-	-	7

29.1.2 GPIO Port B Summary

Table 29-2 GPIO Port B summary

Bit N	PB N	Pull (U/D)	Shared Function Port Selected by				
			0	1	2	3	Note
0	00	U	sa0_cl(o)	-	-	-	8
1	01	U	sa1_al(o)	-	-	-	9
2	02	U	sa2(o)	-	-	-	
3	03	U	sa3(o)	-	-	-	
4	04	U	sa4(o)	dreq1(i)	-	-	
5	05	U	sa5(o)	dack1(o)	-	-	
6	06	U	cim_pclk(i)	tsclk(i)	ssi1_dr(i)	msc2_d0(io)	
7	07	U	cim_hsyn(i)	tsfrm(i)	ssi1_clk(o)	msc2_clk(o)	
8	08	U	cim_vsyn(i)	tsstr(i)	ssi1_ce0_(o)	msc2_cmd(io)	
9	09	U	cim_mclk(o)	tsfail(i)	ssi1_dt(o)	epd_pwc(o)	
10	10	D	cim_d0(i)	tsdi0(i)	-	epd_pwr0(o)	
11	11	D	cim_d1(i)	tsdi1(i)	-	epd_pwr1(o)	
12	12	U	cim_d2(i)	tsdi2(i)	-	epd_sce2_(o)	
13	13	U	cim_d3(i)	tsdi3(i)	-	epd_sce3_(o)	
14	14	U	cim_d4(i)	tsdi4(i)	-	epd_sce4_(o)	
15	15	U	cim_d5(i)	tsdi5(i)	-	epd_sce5_(o)	
16	16	D	cim_d6(i)	tsdi6(i)	-	epd_pwr2(o)	
17	17	D	cim_d7(i)	tsdi7(i)	-	epd_pwr3(o)	
18	18	U	-	-	-	-	
19	19	U	-	-	-	-	
20	20	U	msc2_d0(io)	ssi0_dr(i)	ssi1_dr(i)	tsdi0(i)	
21	21	U	msc2_d1(io)	ssi0_dt(o)	ssi1_dt(o)	tsdi1(i)	
22	22	U	tsdi2(i)	-	-	-	
23	23	U	tsdi3(i)	-	-	-	
24	24	U	tsdi4(i)	-	-	-	
25	25	U	tsdi5(i)	-	-	-	
26	26	U	tsdi6(i)	-	-	-	
27	27	U	tsdi7(i)	-	-	-	
28	28	U	msc2_clk(o)	ssi0_clk(o)	ssi1_clk(o)	tsclk(i)	
29	29	U	msc2_cmd(io)	ssi0_ce0_(o)	ssi1_ce0_(o)	tsstr(i)	
30	30	U	msc2_d2(io)	ssi0_gpc(o)	ssi1_gpc(o)	tsfail(i)	
31	31	U	msc2_d3(io)	ssi0_ce1_(o)	ssi1_ce1_(o)	tsfrm(i)	

29.1.3 GPIO Port C Summary

Table 29-3 GPIO Port C summary

Bit N	PC N	Pull (U/D)	Shared Function Port Selected by				
			0	1	2	3	Note
0	00	U	lcd_b0(o)	lcd_rev(o)	-	-	
1	01	U	lcd_b1(o)	lcd_ps(o)	-	-	
2	02	U	lcd_b2(o)	-	-	-	
3	03	U	lcd_b3(o)	-	-	-	
4	04	U	lcd_b4(o)	-	-	-	
5	05	U	lcd_b5(o)	-	-	-	
6	06	U	lcd_b6(o)	-	-	-	
7	07	U	lcd_b7(o)	-	-	-	
8	08	U	lcd_pclk(o)	-	-	-	
9	09	U	lcd_de(o)	-	-	-	
10	10	U	lcd_g0(o)	lcd_spl(o)	-	-	
11	11	U	lcd_g1(o)	-	-	-	
12	12	U	lcd_g2(o)	-	-	-	
13	13	U	lcd_g3(o)	-	-	-	
14	14	U	lcd_g4(o)	-	-	-	
15	15	U	lcd_g5(o)	-	-	-	
16	16	U	lcd_g6(o)	-	-	-	
17	17	U	lcd_g7(o)	-	-	-	
18	18	U	lcd_hsyn(io)	-	-	-	
19	19	U	lcd_vsyn(io)	-	-	-	
20	20	U	lcd_r0(o)	lcd_cls(o)	-	-	
21	21	U	lcd_r1(o)	-	-	-	
22	22	U	lcd_r2(o)	-	-	-	
23	23	U	lcd_r3(o)	-	-	-	
24	24	U	lcd_r4(o)	-	-	-	
25	25	U	lcd_r5(o)	-	-	-	
26	26	U	lcd_r6(o)	-	-	-	
27	27	U	lcd_r7(o)	-	-	-	
28	28	U	uart2_rxd(i)	-	-	-	
29	29	U	uart2_cts_(i)	-	-	-	
30	30	U	uart2_txd(o)	-	-	-	
31	31	U	uart2_rts_(o)	-	-	-	

29.1.4 GPIO Port D Summary

Table 29-4 GPIO Port D summary

Bit N	PD N	Pull (U/D)	Shared Function Port Selected by				
			0	1	2	3	Note
0	00	U	pcm_dout(o)	-	-	-	
1	01	U	pcm_clk(io)	-	-	-	
2	02	U	pcm_syn(io)	-	-	-	
3	03	U	pcm_din(i)	-	-	-	
4	04	U	ps2_mclk(io)	-	-	-	
5	05	U	ps2_mdata(io)	-	-	-	
6	06	U	ps2_kclk(io)	-	-	-	
7	07	U	ps2_kdata(io)	-	-	-	
8	08	U	scc_data(io)	-	-	-	
9	09	U	scc_clk(o)	-	-	-	
10	10	U	pwm6(io)	-	-	-	
11	11	U	pwm7(io)	-	-	-	
12	12	D	uart3_rxd(i)	bclk(io)	ssi1_dt(o)	epd_pwr4(o)	
13	13	D	sync(io)	msc2_d0(io)	ssi1_dr(i)	epd_pwr5(o)	
14	14	U	-	-	-	-	
15	15	U	-	-	-	-	
16	16	U	-	-	-	-	
17	17	U	boot_sel0(i)	-	-	-	2,5
18	18	U	boot_sel1(i)	-	-	-	3,5
19	19	U	boot_sel2(i)	-	-	-	4,5
20	20	U	msc1_d0(io)	ssi0_dr(i)	ssi1_dr(i)	-	
21	21	U	msc1_d1(io)	ssi0_dt(o)	ssi1_dt(o)	-	
22	22	U	msc1_d2(io)	ssi0_gpc(o)	ssi1_gpc(o)	-	
23	23	U	msc1_d3(io)	ssi0_ce1_(o)	ssi1_ce1_(o)	-	
24	24	U	msc1_clk(o)	ssi0_clk(o)	ssi1_clk(o)	-	
25	25	U	msc1_cmd(io)	ssi0_ce0_(o)	ssi1_ce0_(o)	-	
26	26	U	uart1_rxd(i)	-	-	-	
27	27	U	uart1_cts_(i)	-	-	-	
28	28	U	uart1_txd(o)	-	-	-	
29	29	U	uart1_rts_(o)	-	-	-	
30	30	U	i2c0_sda(io)	-	-	-	
31	31	U	i2c0_sck(io)	-	-	-	

29.1.5 GPIO Port E Summary

Table 29-5 GPIO Port E summary

Bit N	PE N	Pull (U/D)	Shared Function Port Selected by				
			0	1	2	3	Note
0	00	D	pwm0(io)	-	-	-	
1	01	D	pwm1(o)	-	-	-	
2	02	U	pwm2(o)	-	-	-	
3	03	U	pwm3(io)	-	-	-	
4	04	U	pwm4(io)	-	-	-	
5	05	U	pwm5(io)	uart3_txd(o)	sclk_rstn(o)	-	
6	06	U	sdati(i)	msc2_cmd(io)	ssi1_ce0_(o)	epd_pwr6(o)	
7	07	D	sdato(o)	msc2_clk(o)	ssi1_clk(o)	epd_pwr7(o)	
8	08	U	uart3_cts_(i)	-	-	-	
9	09	U	uart3_rts_(o)	-	-	-	
10	10	D	otg_drvvbus(o)	-	-	-	
11	11	U	sdato1(o)	-	-	-	
12	12	U	sdato2(o)	-	-	-	
13	13	U	sdato3(o)	-	-	-	
14	14	U	ssi0_dr(i)	ssi1_dr(i)	-	-	
15	15	U	ssi0_clk(o)	ssi1_clk(o)	-	-	
16	16	U	ssi0_ce0_(o)	ssi1_ce0_(o)	-	-	
17	17	U	ssi0_dt(o)	ssi1_dt(o)	-	-	
18	18	U	ssi0_ce1_(o)	ssi1_ce1_(o)	-	-	
19	19	U	ssi0_gpc(o)	ssi1_gpc(o)	-	-	
20	20	U	msc0_d0(io)	msc1_d0(io)	msc2_d0(io)	-	
21	21	U	msc0_d1(io)	msc1_d1(io)	msc2_d1(io)	-	
22	22	U	msc0_d2(io)	msc1_d2(io)	msc2_d2(io)	-	
23	23	U	msc0_d3(io)	msc1_d3(io)	msc2_d3(io)	-	
24	24	U	msc0_d4(io)	msc1_d4(io)	msc2_d4(io)	-	
25	25	U	msc0_d5(io)	msc1_d5(io)	msc2_d5(io)	-	
26	26	U	msc0_d6(io)	msc1_d6(io)	msc2_d6(io)	-	
27	27	U	msc0_d7(io)	msc1_d7(io)	msc2_d7(io)	-	
28	28	U	msc0_clk(o)	msc1_clk(o)	msc2_clk(o)	-	
29	29	U	msc0_cmd(io)	msc1_cmd(io)	msc2_cmd(io)	-	
30	30	U	i2c1_sda(io)	-	-	-	
31	31	U	i2c1_sck(io)	-	-	-	

29.1.6 GPIO Port F Summary

Table 29-6 GPIO Port F summary

Bit N	PF N	Pull (U/D)	Shared Function Port Selected by				
			0	1	2	3	Note
0	00	U	uart0_rxd(i)	gps_clk(i)	ssi1_dr(i)	msc2_d0(io)	
1	01	U	uart0_cts_(i)	gps_mag(i)	ssi1_ce0_(o)	msc2_cmd(io)	
2	02	U	uart0_rts_(o)	gps_sig(i)	ssi1_clk(o)	msc2_clk(o)	
3	03	U	uart0_txd(o)	-	ssi1_dt(o)	-	
4	04	D	-	-	-	-	
5	05	D	-	-	-	-	
6	06	D	-	-	-	-	
7	07	D	-	-	-	-	
8	08	D	-	-	-	-	
9	09	D	-	-	-	-	
10	10	D	-	-	-	-	
11	11	D	-	-	-	-	

NOTES:

1 If NAND flash is used, this pin must be used as NAND FRB.

2 PD17: GPIO group D bit 17 is used as BOOT_SEL0 input during boot.

3 PD18: GPIO group D bit 18 is used as BOOT_SEL1 input during boot.

4 PD19: GPIO group D bit 19 is used as BOOT_SEL2 input during boot.

5 BOOT_SEL2, BOOT_SEL1, BOOT_SEL0 are used to select boot source and function during the processor boot.

6 PA30: GPIO group A bit 30 can only be used as input and interrupt, no pull-up and pull-down. It is also used to select the function between PS2 and JTAG of JTAG/UART3/PS2 Pins （ TCK_UART3_RTS__PS2_MCLK, TMS_UART3_CTS__PS2_MDATA , TDI_UART3_RxD_PS2_KCLK and TDO_UART3_TxD_PS2_KDATA），which share the same set of pins.
When PA30.function1 is false, select JTAG function.
When PA30.function1 is true, select PS2 function.

7 PA31: GPIO group A bit 31. No corresponding pin exists for this GPIO.
It is only used to select the function between UART and JTAG of JTAG/UART3/PS2 Pins (TCK_UART3_RTS__PS2_MCLK,TMS_UART3_CTS__PS2_MDATA, TDI_UART3_RxD_PS2_KCLK and TDO_UART3_TxD_PS2_KDATA）, which share the same set of pins, by using register PASEL [31].
When PASEL [31] =0, select JTAG function.
When PASEL [31] =1, select UART function.

8 If NAND flash is used, this pin must be used as NAND CLE.

9 If NAND flash is used, this pin must be used as NAND ALE.

29.2 Registers Description

Table 29-7 summarized all memory-mapped registers, which can be programmed to operate GPIO port and alternate function port sharing configuration.

All registers are in 32-bits width. Usually, 1 bit in the register affects a corresponding GPIO port and every GPIO port can be operated independently.

Table 29-7 GPIO Registers

Name	Description	RW	Reset Value	Address	Size
GPIO PORT A					
PAPIN	PORT A PIN Level Register	R	0x00000000	0x10010000	32
PADAT	PORT A Data Register	R	0x00000000	0x10010010	32
PADATS	PORT A Data Set Register	W	0x????????	0x10010014	32
PADATC	PORT A Data Clear Register	W	0x????????	0x10010018	32
PAIM	PORT A Interrupt Mask Register	R	0xFFFFFFFF	0x10010020	32
PAIMS	PORT A Interrupt Mask Set Register	W	0x????????	0x10010024	32
PAIMC	PORT A Interrupt Mask Clear Register	W	0x????????	0x10010028	32
PAPE	PORT A PULL Disable Register	R	0x00000000	0x10010030	32
PAPES	PORT A PULL Disable Set Register	W	0x????????	0x10010034	32
PAPEC	PORT A PULL Disable Clear Register	W	0x????????	0x10010038	32
PAFUN	PORT A Function Register	R	0x00000000	0x10010040	32
PAFUNS	PORT A Function Set Register	W	0x????????	0x10010044	32
PAFUNC	PORT A Function Clear Register	W	0x????????	0x10010048	32
PASEL	PORT A Select Register	R	0x00000000	0x10010050	32
PASELS	PORT A Select Set Register	W	0x????????	0x10010054	32
PASELC	PORT A Select Clear Register	W	0x????????	0x10010058	32
PADIR	PORT A Direction Register	R	0x00000000	0x10010060	32
PADIRS	PORT A Direction Set Register	W	0x????????	0x10010064	32
PADIRC	PORT A Direction Clear Register	W	0x????????	0x10010068	32
PATRG	PORT A Trigger Register	R	0x00000000	0x10010070	32
PATRGS	PORT A Trigger Set Register	W	0x????????	0x10010074	32
PATRGC	PORT A Trigger Clear Register	W	0x????????	0x10010078	32
PAFLG	PORT A FLAG Register	R	0x00000000	0x10010080	32
PAFLGC	PORT A FLAG Clear Register	W	0x????????	0x10010014	32
GPIO PORT B					
PBPIN	PORT B PIN Level Register	R	0x00000000	0x10010100	32
PBDAT	PORT B Data Register	R	0x00000000	0x10010110	32
PBDATS	PORT B Data Set Register	W	0x????????	0x10010114	32
PBDATC	PORT B Data Clear Register	W	0x????????	0x10010118	32
PBIM	PORT B Interrupt Mask Register	R	0xFFFFFFFF	0x10010120	32
PBIMS	PORT B Interrupt Mask Set Register	W	0x????????	0x10010124	32

PBIMC	PORT B Interrupt Mask Clear Register	W	0x????????	0x10010128	32
PBPE	PORT B PULL Enable Register	R	0x00000000	0x10010130	32
PBPES	PORT B PULL Enable Set Register	W	0x????????	0x10010134	32
PBPEC	PORT B PULL Enable Clear Register	W	0x????????	0x10010138	32
PBFUN	PORT B Function Register	R	0x00000000	0x10010140	32
PBFUNS	PORT B Function Set Register	W	0x????????	0x10010144	32
PBFUNC	PORT B Function Clear Register	W	0x????????	0x10010148	32
PBSEL	PORT B Select Register	R	0x00000000	0x10010150	32
PBSELS	PORT B Select Set Register	W	0x????????	0x10010154	32
PBSELC	PORT B Select Clear Register	W	0x????????	0x10010158	32
PBDIR	PORT B Direction Register	R	0x00000000	0x10010160	32
PBDIRS	PORT B Direction Set Register	W	0x????????	0x10010164	32
PBDIRC	PORT B Direction Clear Register	W	0x????????	0x10010168	32
PBTRG	PORT B Trigger Register	R	0x00000000	0x10010170	32
PBTRGS	PORT B Trigger Set Register	W	0x????????	0x10010174	32
PBTRGC	PORT B Trigger Clear Register	W	0x????????	0x10010178	32
PBFLG	PORT B FLAG Register	R	0x00000000	0x10010180	32
PBFLGC	PORT B FLAG Clear Register	W	0x????????	0x10010114	32
GPIO PORT C					
PCPIN	PORT C PIN Level Register	R	0x00000000	0x10010200	32
PCDAT	PORT C Data Register	R	0x00000000	0x10010210	32
PCDATS	PORT C Data Set Register	W	0x????????	0x10010214	32
PCDATC	PORT C Data Clear Register	W	0x????????	0x10010218	32
PCIM	PORT C Interrupt Mask Register	R	0xFFFFFFFF	0x10010220	32
PCIMS	PORT C Interrupt Mask Set Register	W	0x????????	0x10010224	32
PCIMC	PORT C Interrupt Mask Clear Register	W	0x????????	0x10010228	32
PCPE	PORT C PULL Enable Register	R	0x00000000	0x10010230	32
PCPES	PORT C PULL Enable Set Register	W	0x????????	0x10010234	32
PCPEC	PORT C PULL Enable Clear Register	W	0x????????	0x10010238	32
PCFUN	PORT C Function Register	R	0x00000000	0x10010240	32
PCFUNS	PORT C Function Set Register	W	0x????????	0x10010244	32
PCFUNC	PORT C Function Clear Register	W	0x????????	0x10010248	32
PCSEL	PORT C Select Register	R	0x00000000	0x10010250	32
PCSELS	PORT C Select Set Register	W	0x????????	0x10010254	32
PCSELC	PORT C Select Clear Register	W	0x????????	0x10010258	32
PCDIR	PORT C Direction Register	R	0x00000000	0x10010260	32
PCDIRS	PORT C Direction Set Register	W	0x????????	0x10010264	32
PCDIRC	PORT C Direction Clear Register	W	0x????????	0x10010268	32
PCTRG	PORT C Trigger Register	R	0x00000000	0x10010270	32
PCTRGS	PORT C Trigger Set Register	W	0x????????	0x10010274	32
PCTRGC	PORT C Trigger Clear Register	W	0x????????	0x10010278	32

PCFLG	PORT C FLAG Register	R	0x00000000	0x10010280	32
PCFLGC	PORT C FLAG Clear Register	W	0x????????	0x10010214	32
GPIO PORT D					
PDPIN	PORT D PIN Level Register	R	0x00000000	0x10010300	32
PDDAT	PORT D Data Register	R	0x00000000	0x10010310	32
PDDATS	PORT D Data Set Register	W	0x????????	0x10010314	32
PDDATC	PORT D Data Clear Register	W	0x????????	0x10010318	32
PDIM	PORT D Interrupt Mask Register	R	0xFFFFFFFF	0x10010320	32
PDIMS	PORT D Interrupt Mask Set Register	W	0x????????	0x10010324	32
PDIMC	PORT D Interrupt Mask Clear Register	W	0x????????	0x10010328	32
PDPE	PORT D PULL Enable Register	R	0x00000000	0x10010330	32
PDPES	PORT D PULL Enable Set Register	W	0x????????	0x10010334	32
PDPEC	PORT D PULL Enable Clear Register	W	0x????????	0x10010338	32
PDFUN	PORT D Function Register	R	0x00000000	0x10010340	32
PDFUNS	PORT D Function Set Register	W	0x????????	0x10010344	32
PDFUNC	PORT D Function Clear Register	W	0x????????	0x10010348	32
PDSEL	PORT D Select Register	R	0x00000000	0x10010350	32
PDSELS	PORT D Select Set Register	W	0x????????	0x10010354	32
PDSELC	PORT D Select Clear Register	W	0x????????	0x10010358	32
P\DDIR	PORT D Direction Register	R	0x00000000	0x10010360	32
PDDIRS	PORT D Direction Set Register	W	0x????????	0x10010364	32
PDDIRC	PORT D Direction Clear Register	W	0x????????	0x10010368	32
PDTRG	PORT D Trigger Register	R	0x00000000	0x10010370	32
PDTRGS	PORT D Trigger Set Register	W	0x????????	0x10010374	32
PDTRGC	PORT D Trigger Clear Register	W	0x????????	0x10010378	32
PDFLG	PORT D FLAG Register	R	0x00000000	0x10010380	32
PDFLGC	PORT D FLAG Clear Register	W	0x????????	0x10010314	32
GPIO PORT E					
PEPIN	PORT E PIN Level Register	R	0x00000000	0x10010400	32
PEDAT	PORT E Data Register	R	0x00000000	0x10010410	32
PEDATS	PORT E Data Set Register	W	0x????????	0x10010414	32
PEDATC	PORT E Data Clear Register	W	0x????????	0x10010418	32
PEIM	PORT E Interrupt Mask Register	R	0xFFFFFFFF	0x10010420	32
PEIMS	PORT E Interrupt Mask Set Register	W	0x????????	0x10010424	32
PEIMC	PORT E Interrupt Mask Clear Register	W	0x????????	0x10010428	32
PEPE	PORT E PULL Enable Register	R	0x00000000	0x10010430	32
PEPES	PORT E PULL Enable Set Register	W	0x????????	0x10010434	32
PEPEC	PORT E PULL Enable Clear Register	W	0x????????	0x10010438	32
PEFUN	PORT E Function Register	R	0x00000000	0x10010440	32
PEFUNS	PORT E Function Set Register	W	0x????????	0x10010444	32
PEFUNC	PORT E Function Clear Register	W	0x????????	0x10010448	32

PESEL	PORT E Select Register	R	0x00000000	0x10010450	32
PESELS	PORT E Select Set Register	W	0x????????	0x10010454	32
PESELC	PORT E Select Clear Register	W	0x????????	0x10010458	32
PEDIR	PORT E Direction Register	R	0x00000000	0x10010460	32
PEDIRS	PORT E Direction Set Register	W	0x????????	0x10010464	32
PEDIRC	PORT E Direction Clear Register	W	0x????????	0x10010468	32
PETRG	PORT E Trigger Register	R	0x00000000	0x10010470	32
PETRGS	PORT E Trigger Set Register	W	0x????????	0x10010474	32
PETRGC	PORT E Trigger Clear Register	W	0x????????	0x10010478	32
PEFLG	PORT E FLAG Register	R	0x00000000	0x10010480	32
PEFLGC	PORT E FLAG Clear Register	W	0x????????	0x10010414	32
GPIO PORT F					
PFPIN	PORT F PIN Level Register	R	0x00000000	0x10010500	32
PFDAT	PORT F Data Register	R	0x00000000	0x10010510	32
PFDATS	PORT F Data Set Register	W	0x????????	0x10010514	32
PFDATC	PORT F Data Clear Register	W	0x????????	0x10010518	32
PFIM	PORT F Interrupt Mask Register	R	0x00FFFFFF	0x10010520	32
PFIMS	PORT F Interrupt Mask Set Register	W	0x????????	0x10010524	32
PFIMC	PORT F Interrupt Mask Clear Register	W	0x????????	0x10010528	32
PFPE	PORT F PULL Enable Register	R	0x00000000	0x10010530	32
PFPES	PORT F PULL Enable Set Register	W	0x????????	0x10010534	32
PFPEC	PORT F PULL Enable Clear Register	W	0x????????	0x10010538	32
PFFUN	PORT F Function Register	R	0x00000000	0x10010540	32
PFFUNS	PORT F Function Set Register	W	0x????????	0x10010544	32
PFFUNC	PORT F Function Clear Register	W	0x????????	0x10010548	32
PFSEL	PORT F Select Register	R	0x00000000	0x10010550	32
PFSELS	PORT F Select Set Register	W	0x????????	0x10010554	32
PFSELC	PORT F Select Clear Register	W	0x????????	0x10010558	32
PFDIR	PORT F Direction Register	R	0x00000000	0x10010560	32
PFDIRS	PORT F Direction Set Register	W	0x????????	0x10010564	32
PFDIRC	PORT F Direction Clear Register	W	0x????????	0x10010568	32
PFTRG	PORT F Trigger Register	R	0x00000000	0x10010570	32
PFTRGS	PORT F Trigger Set Register	W	0x????????	0x10010574	32
PFTRGC	PORT F Trigger Clear Register	W	0x????????	0x10010578	32
PFFLG	PORT F FLAG Register	R	0x00000000	0x10010580	32
PFFLGC	PORT F FLAG Clear Register	W	0x????????	0x10010514	32

NOTES:

1 PX**** in the description of register as follows means PA****, PB****, PC****, PD****, PE**** and PF****.

29.2.1 PORT PIN Level Register (PxPIN)

PAPIN, PBPIN, PCPIN, PDPIN, PEPIN and PFPIN are six 32-bit PORT PIN level registers. They are read-only registers.

PAPIN, PBPIN, PCPIN, PDPIN, PEPIN, PFPIN — **0x10010000, 0x10010100, 0x10010200, 0x10010300, 0x10010400, 0x10010500**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PINL31	PINL30	PINL29	PINL28	PINL27	PINL26	PINL25	PINL24	PINL23	PINL22	PINL21	PINL20	PINL19	PINL18	PINL17	PINL16	PINL15	PINL14	PINL13	PINL12	PINL11	PINL10	PINL09	PINL08	PINL07	PINL06	PINL05	PINL04	PINL03	PINL02	PINL01	PINL00
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
n	PINL n	Where n = 0 ~ 31 and PINL n = PINL0 ~ PINL31. The PORT PIN level can be read by reading PINL n bit in register PXPIN.	R

PAPIN bits 31-0 correspond to PA31-0; PBPIN to PB31-0; PCPIN to PC31-0; PDPIN to PD31-0; PEPIN to PE31-0 and PFPIN to PF 31-0.

29.2.2 PORT Data Register (PxDAT)

PADAT, PBDAT, PCDAT, PDDAT, PEDAT and PFDAT are six 32-bit PORT DATA registers. They are read-only registers.

PADAT, PBDAT, PCDAT, PDDAT, PEDAT, PFDAT — **0x10010010, 0x10010110, 0x10010210, 0x10010310, 0x10010410, 0x10010510**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATA31	DATA30	DATA29	DATA28	DATA27	DATA26	DATA25	DATA24	DATA23	DATA22	DATA21	DATA20	DATA19	DATA18	DATA17	DATA16	DATA15	DATA14	DATA13	DATA12	DATA11	DATA10	DATA09	DATA08	DATA07	DATA06	DATA05	DATA04	DATA03	DATA02	DATA01	DATA00
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
N	DATA n	Where n = 0 ~ 31 and DATA n = DATA0 ~ DATA31. The register is used as GPIO data register. When GPIO is used as interrupt the register is no used.	R

PADAT bits 31-0 correspond to PA31-0; PBDAT to PB31-0; PCDAT to PC31-0; PDDAT to PD31-0; PEDAT to PE31-0 and PFDAT to PF 31-0.

29.2.3 PORT Data Set Register (PxDATS)

PADATS, PBDATS, PCDATS, PDDATS, PEDATS and PFDATS are six 32-bit PORT DATA set registers. They are write-only registers.

PADATS, PBDATS, PCDATS, PDDATS, PEDATS, PFDATS — **0x10010014, 0x10010114, 0x10010214, 0x10010314, 0x10010414, 0x10010514**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATAS31	DATAS30	DATAS29	DATAS28	DATAS27	DATAS26	DATAS25	DATAS24	DATAS23	DATAS22	DATAS21	DATAS20	DATAS19	DATAS18	DATAS17	DATAS16	DATAS15	DATAS14	DATAS13	DATAS12	DATAS11	DATAS10	DATAS09	DATAS08	DATAS07	DATAS06	DATAS05	DATAS04	DATAS03	DATAS02	DATAS01	DATAS00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	DATAS n	Writing 1 to DATAS n will set DATA n to 1 in register PXDAT. Writing 0 to DATAS n will no use.	W

PADATS bits 31-0 correspond to PA31-0; PBDATS to PB31-0; PCDATS to PC31-0; PDDATS to PD31-0; PEDATS to PE31-0 and PFDATS to PF 31-0.

29.2.4 PORT Data Clear Register (PxDATC)

PADATC, PBDATC, PCDATC, PDDATC, PEDATC and PFDATC are six 32-bit PORT DATA clear registers. They are write-only registers.

PADATC, PBDATC, PCDATC, PDDATC, PEDATC, PFDATC — **0x10010018, 0x10010118, 0x10010218, 0x10010318, 0x10010418, 0x10010518**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATAC31	DATAC30	DATAC29	DATAC28	DATAC27	DATAC26	DATAC25	DATAC24	DATAC23	DATAC22	DATAC21	DATAC20	DATAC19	DATAC18	DATAC17	DATAC16	DATAC15	DATAC14	DATAC13	DATAC12	DATAC11	DATAC10	DATAC09	DATAC08	DATAC07	DATAC06	DATAC05	DATAC04	DATAC03	DATAC02	DATAC01	DATAC00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	DATAC n	Writing 1 to DATAC n will set DATA n to 0 in register PXDAT. Writing 0 to DATAC n will no use.	W

PADATC bits 31-0 correspond to PA31-0; PBDATC to PB31-0; PCDATC to PC31-0; PDDATC to PD31-0; PEDATC to PE31-0 and PFDATC to PF 31-0.

29.2.5 PORT Mask Register (PxIM)

PAIM, PBIM, PCIM, PDIM, PEIM and PFIM are six 32-bit PORT MASK registers. They are read-only registers.

PAIM, PBIM, PCIM, PDIM. PEIM, PFIM																																0x10010020, 0x10010120, 0x10010220, 0x10010320, 0x10010420, 0x10010520
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MASK31	MASK30	MASK29	MASK28	MASK27	MASK26	MASK25	MASK24	MASK23	MASK22	MASK21	MASK20	MASK19	MASK18	MASK17	MASK16	MASK15	MASK14	MASK13	MASK12	MASK11	MASK10	MASK09	MASK08	MASK07	MASK06	MASK05	MASK04	MASK03	MASK02	MASK01	MASK00
RST	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	Name	Description	R/W
n	MASK n	Where n = 0 ~ 31 and MASK n = MASK0 ~ MASK31. MASK n is used for mask the interrupt of GPIO n. 0: Enable the pin as an interrupt source 1: Disable the pin as an interrupt source	R

PAIM bits 31-0 correspond to PA31-0; PBIM to PB31-0; PCIM to PC31-0; PDIM to PD31-0; PEIM to PE31-0 and PFIM to PF 31-0.

29.2.6 PORT Mask Set Register (PxIMS)

PAIMS, PBIMS, PCIMS, PDIMS, PEIMS and PFIMS are six 32-bit PORT MASK set registers. They are write-only registers.

PAIMS, PBIMS, PCIMS, PDIMS, PEIMS, PFIMS																																0x10010024, 0x10010124, 0x10010224, 0x10010324, 0x10010424, 0x10010524
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MASKS31	MASKS30	MASKS29	MASKS28	MASKS27	MASKS26	MASKS25	MASKS24	MASKS23	MASKS22	MASKS21	MASKS20	MASKS19	MASKS18	MASKS17	MASKS16	MASKS15	MASKS14	MASKS13	MASKS12	MASKS11	MASKS10	MASKS09	MASKS08	MASKS07	MASKS06	MASKS05	MASKS04	MASKS03	MASKS02	MASKS01	MASKS00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	MASKS n	Writing 1 to MASKS n will set MASK n to 1 in register PXIM. Writing 0 to MASKS n will no use.	W

PAIMS bits 31-0 correspond to PA31-0; PBIMS to PB31-0; PCIMS to PC31-0; PDIMS to PD31-0; PEIMS to PE31-0 and PFIMS to PF 31-0.

29.2.7 PORT Mask Clear Register (PxIMC)

PAIMC, PBIMC, PCIMC, PDIMC, PEIMC and PFIMC are six 32-bit PORT MASK clear registers. They are write-only registers.

PAIMS, PBIMC, PCIMC, PDIMC, PEIMC, PFIMC — **0x10010028, 0x10010128, 0x10010228, 0x10010328, 0x10010428, 0x10010528**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MASKC31	MASKC30	MASKC29	MASKC28	MASKC27	MASKC26	MASKC25	MASKC24	MASKC23	MASKC22	MASKC21	MASKC20	MASKC19	MASKC18	MASKC17	MASKC16	MASKC15	MASKC14	MASKC13	MASKC12	MASKC11	MASKC10	MASKC09	MASKC08	MASKC07	MASKC06	MASKC05	MASKC04	MASKC03	MASKC02	MASKC01	MASKC00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	MASKC n	Writing 1 to MASKC n will set MASK n to 0 in register PXIM. Writing 0 to MASKC n will no use.	W

PAIMC bits 31-0 correspond to PA31-0; PBIMC to PB31-0; PCIMC to PC31-0; PDIMC to PD31-0; PEIMC to PE31-0 and PFIMC to PF 31-0.

29.2.8 PORT PULL Disable Register (PxPE)

PAPE, PBPE, PCPE, PDPE, PEPE and PFPE are six 32-bit PORT PULL disable registers. They are read-only registers.

PAPE, PBPE, PCPE, PDPE, PEPE, PFPE — **0x10010030, 0x10010130, 0x10010230, 0x10010330, 0x10010430, 0x10010530**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PULL31	PULL30	PULL29	PULL28	PULL27	PULL26	PULL25	PULL24	PULL23	PULL22	PULL21	PULL20	PULL19	PULL18	PULL17	PULL16	PULL15	PULL14	PULL13	PULL12	PULL11	PULL10	PULL09	PULL08	PULL07	PULL06	PULL05	PULL04	PULL03	PULL02	PULL01	PULL00
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
N	PULL n	Where n = 0 ~ 31 and PULL n = PULL0 ~ PULL31. PULL n is used for setting the port to be PULL UP or PULL DOWN enable. 1: No pull up or pull down resistor connects to the port 0: An internal pull up or pull down resistor connects to the port. Up or down is pin dependence.	R

PAPE bits 31-0 correspond to PA31-0; PBPE to PB31-0; PCPE to PC31-0; PDPE to PD31-0; PEPE to PE31-0 and PFPE to PF 31-0.

29.2.9 PORT PULL Set Register (PxPES)

PAPES, PBPES, PCPES, PDPES, PEPES and PFPES are six 32-bit PORT PULL set registers. They are write-only registers.

PAPES, PBPES, PCPES, PDPES, PEPES, PFPES — **0x10010034, 0x10010134, 0x10010234, 0x10010334, 0x10010434, 0x10010534**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PULLS31	PULLS30	PULLS29	PULLS28	PULLS27	PULLS26	PULLS25	PULLS24	PULLS23	PULLS22	PULLS21	PULLS20	PULLS19	PULLS18	PULLS17	PULLS16	PULLS15	PULLS14	PULLS13	PULLS12	PULLS11	PULLS10	PULLS09	PULLS08	PULLS07	PULLS06	PULLS05	PULLS04	PULLS03	PULLS02	PULLS01	PULLS00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	PULLS n	Writing 1 to PULLS n will set PULL n to 1 in register PXPE. Writing 0 to PULLS n will no use.	W

PAPES bits 31-0 correspond to PA31-0; PBPES to PB31-0; PCPES to PC31-0; PDPES to PD31-0; PEPES to PE31-0 and PFPES to PF 31-0.

29.2.10 PORT PULL Clear Register (PxPEC)

PAPEC, PBPEC, PCPEC, PDPEC, PEPEC and PFPEC are six 32-bit PORT PULL clear registers. They are write-only registers.

PAPES, PBPEC, PCPEC, PDPEC, PEPEC, PFPEC — **0x10010038, 0x10010138, 0x10010238, 0x10010338, 0x10010438, 0x10010538**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	PULLC31	PULLC30	PULLC29	PULLC28	PULLC27	PULLC26	PULLC25	PULLC24	PULLC23	PULLC22	PULLC21	PULLC20	PULLC19	PULLC18	PULLC17	PULLC16	PULLC15	PULLC14	PULLC13	PULLC12	PULLC11	PULLC10	PULLC09	PULLC08	PULLC07	PULLC06	PULLC05	PULLC04	PULLC03	PULLC02	PULLC01	PULLC00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	PULLC n	Writing 1 to PULLC n will set PULL n to 0 in register PXPE. Writing 0 to PULLC n will no use.	W

PAPEC bits 31-0 correspond to PA31-0; PBPEC to PB31-0; PCPEC to PC31-0; PDPEC to PD31-0; PEPEC to PE31-0 and PFPEC to PF 31-0.

29.2.11 PORT Function Register (PxFUN)

PAFUN, PBFUN, PCFUN, PDFUN, PEFUN and PFFUN are six 32-bit PORT function registers. They are read-only registers.

PAFUN, PBFUN, PCFUN, PDFUN, PEFUN, PFFUN — **0x10010040, 0x10010140, 0x10010240, 0x10010340, 0x10010440, 0x10010540**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FUN31	FUN30	FUN29	FUN28	FUN27	FUN26	FUN25	FUN24	FUN23	FUN22	FUN21	FUN20	FUN19	FUN18	FUN17	FUN16	FUN15	FUN14	FUN13	FUN12	FUN11	FUN10	FUN09	FUN08	FUN07	FUN06	FUN05	FUN04	FUN03	FUN02	FUN01	FUN00
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
n	FUN n	Where n = 0 ~ 31 and FUN n = FUN0 ~ FUN31. In most cases, port is shared with one or more peripheral functions. FUN n controls the owner of the port n. 0: GPIO or Interrupt 1: Alternate Function	R

PAFUN bits 31-0 correspond to PA31-0; PBFUN to PB31-0; PCFUN to PC31-0; PDFUN to PD31-0; PEFUN to PE31-0 and PFFUN to PF 31-0.

29.2.12 PORT Function Set Register (PxFUNS)

PAFUNS, PBFUNS, PCFUNS, PDFUNS, PEFUNS and PFFUNS are six 32-bit PORT function set registers. They are write-only registers.

PAFUNS, PBFUNS, PCFUNS, PDFUNS, PEFUNS, PFFUNS — **0x10010044, 0x10010144, 0x10010244, 0x10010344, 0x10010444, 0x10010544**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FUNS31	FUNS30	FUNS29	FUNS28	FUNS27	FUNS26	FUNS25	FUNS24	FUNS23	FUNS22	FUNS21	FUNS20	FUNS19	FUNS18	FUNS17	FUNS16	FUNS15	FUNS14	FUNS13	FUNS12	FUNS11	FUNS10	FUNS09	FUNS08	FUNS07	FUNS06	FUNS05	FUNS04	FUNS03	FUNS02	FUNS01	FUNS00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	FUNS n	Writing 1 to FUNS n will set FUN n to 1 in register PXFUN. Writing 0 to FUNS n will no use.	W

PAFUNS bits 31-0 correspond to PA31-0; PBFUNS to PB31-0; PCFUNS to PC31-0; PDFUNS to PD31-0; PEFUNS to PE31-0 and PFFUNS to PF 31-0.

29.2.13 PORT Function Clear Register (PxFUNC)

PAFUNC, PBFUNC, PCFUNC, PDFUNC, PEFUNC and PFFUNC are six 32-bit PORT function clear registers. They are write-only registers.

PAFUNC, PBFUNC, PCFUNC, PDFUNC, PEFUNC, PFFUNC — **0x10010048, 0x10010148, 0x10010248, 0x10010348, 0x10010448, 0x10010548**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FUNC31	FUNC30	FUNC29	FUNC28	FUNC27	FUNC26	FUNC25	FUNC24	FUNC23	FUNC22	FUNC21	FUNC20	FUNC19	FUNC18	FUNC17	FUNC16	FUNC15	FUNC14	FUNC13	FUNC12	FUNC11	FUNC10	FUNC09	FUNC08	FUNC07	FUNC06	FUNC05	FUNC04	FUNC03	FUNC02	FUNC01	FUNC00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	FUNC n	Writing 1 to FUNC n will set FUN n to 0 in register PXFUN. Writing 0 to FUNC n will no use.	W

PAFUNC bits 31-0 correspond to PA31-0; PBFUNC to PB31-0; PCFUNC to PC31-0; PDFUNC to PD31-0; PEFUNC to PE31-0 and PFFUNC to PF 31-0.

29.2.14 PORT Select Register (PxSEL)

PASEL, PBSEL, PCSEL, PDSEL, PESEL and PFSEL are six 32-bit PORT select registers. They are read-only registers.

PASEL, PBSEL, PCSEL, PDSEL, PESEL, PFSEL — **0x10010050, 0x10010150, 0x10010250, 0x10010350, 0x10010450, 0x10010550**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SEL31	SEL30	SEL29	SEL28	SEL27	SEL26	SEL25	SEL24	SEL23	SEL22	SEL21	SEL20	SEL19	SEL18	SEL17	SEL16	SEL15	SEL14	SEL13	SEL12	SEL11	SEL10	SEL09	SEL08	SEL07	SEL06	SEL05	SEL04	SEL03	SEL02	SEL01	SEL00
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
n	SEL n	Where n = 0 ~ 31 and SEL n = SEL0 ~ SEL31. When PXFUN = 0: 0: GPIO 1: Interrupt When PXFUN = 1: 0: Alternate Function 0 of group 1: Alternate Function 1 of group	R

PASEL bits 31-0 correspond to PA31-0; PBSEL to PB31-0; PCSEL to PC31-0; PDSEL to PD31-0;

PESEL to PE31-0 and PFSEL to PF 31-0.

29.2.15 PORT Select Set Register (PxSELS)

PASELS, PBSELS, PCSELS, PDSELS, PESELS and PFSELS are six 32-bit PORT select set registers. They are write-only registers.

PASELS, PBSELS, PCSELS, 0x10010054, 0x10010154, 0x10010254,
PDSELS, PESELS, PFSELS 0x10010354, 0x10010454, 0x10010554

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SELS31	SELS30	SELS29	SELS28	SELS27	SELS26	SELS25	SELS24	SELS23	SELS22	SELS21	SELS20	SELS19	SELS18	SELS17	SELS16	SELS15	SELS14	SELS13	SELS12	SELS11	SELS10	SELS09	SELS08	SELS07	SELS06	SELS05	SELS04	SELS03	SELS02	SELS01	SELS00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	SELS n	Writing 1 to SELS n will set SEL n to 1 in register PXSEL. Writing 0 to SELS n will no use.	W

PASELS bits 31-0 correspond to PA31-0; PBSELS to PB31-0; PCSELS to PC31-0; PDSELS to PD31-0; PESELS to PE31-0 and PFSELS to PF 31-0.

29.2.16 PORT Select Clear Register (PxSELC)

PASELC, PBSELC, PCSELC, PDSELC, PESELC and PFSELC are six 32-bit PORT select clear registers. They are write-only registers.

PASELC, PBSELC, PCSELC, 0x10010058, 0x10010158, 0x10010258,
PDSELC, PESELC, PFSELC 0x10010358, 0x10010458, 0x10010558

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SELC31	SELC30	SELC29	SELC28	SELC27	SELC26	SELC25	SELC24	SELC23	SELC22	SELC21	SELC20	SELC19	SELC18	SELC17	SELC16	SELC15	SELC14	SELC13	SELC12	SELC11	SELC10	SELC09	SELC08	SELC07	SELC06	SELC05	SELC04	SELC03	SELC02	SELC01	SELC00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	SELC n	Writing 1 to SELC n will set SEL n to 0 in register PXSEL. Writing 0 to SELC n will no use.	W

PASELC bits 31-0 correspond to PA31-0; PBSELC to PB31-0; PCSELC to PC31-0; PDSELC to PD31-0; PESELC to PE31-0 and PFSELC to PF 31-0.

29.2.17 PORT Direction Register (PxDIR)

PADIR, PBDIR, PCDIR, PDDIR, PEDIR and PFDIR are six 32-bit PORT direction registers. They are read-only registers.

PADIR, PBDIR, PCDIR, PDDIR, PEDIR, PFDIR — **0x10010060, 0x10010160, 0x10010260, 0x10010360, 0x10010460, 0x10010560**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DIR31	DIR30	DIR29	DIR28	DIR27	DIR26	DIR25	DIR24	DIR23	DIR22	DIR21	DIR20	DIR19	DIR18	DIR17	DIR16	DIR15	DIR14	DIR13	DIR12	DIR11	DIR10	DIR09	DIR08	DIR07	DIR06	DIR05	DIR04	DIR03	DIR02	DIR01	DIR00
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
n	DIR n	Where n = 0 ~ 31 and DIR n = DIR0 ~ DIR31. DIR n is used for setting the direction of port or setting the trigger direction of interrupt trigger. GPIO Direction: (GPIO Function) 0: INPUT 1: OUTPUT Interrupt Trigger Direction: (Interrupt Function) PXTRG = 0: 0: Low Level Trigger 1: High Level Trigger PXTRG =1: 0: Falling Edge Trigger 1: Rising Edge Trigger	R

PADIR bits 31-0 correspond to PA31-0; PBDIR to PB31-0; PCDIR to PC31-0; PDDIR to PD31-0; PEDIR to PE31-0 and PFDIR to PF 31-0.

29.2.18 PORT Direction Set Register (PxDIRS)

PADIRS, PBDIRS, PCDIRS, PDDIRS, PEDIRS and PFDIRS are six 32-bit PORT direction set registers. They are write-only registers.

PADIRS, PBDIRS, PCDIRS, PDDIRS, PEDIRS, PFDIRS — **0x10010064, 0x10010164, 0x10010264, 0x10010364, 0x10010464, 0x10010564**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DIRS31	DIRS30	DIRS29	DIRS28	DIRS27	DIRS26	DIRS25	DIRS24	DIRS23	DIRS22	DIRS21	DIRS20	DIRS19	DIRS18	DIRS17	DIRS16	DIRS15	DIRS14	DIRS13	DIRS12	DIRS11	DIRS10	DIRS09	DIRS08	DIRS07	DIRS06	DIRS05	DIRS04	DIRS03	DIRS02	DIRS01	DIRS00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	DIRS n	Writing 1 to DIRS n will set DIR n to 1 in register PXDIR. Writing 0 to DIRS n will no use.	W

PADIRS bits 31-0 correspond to PA31-0; PBDIRS to PB31-0; PCDIRS to PC31-0; PDDIRS to PD31-0; PEDIRS to PE31-0 and PFDIRS to PF 31-0.

29.2.19 PORT Direction Clear Register (PxDIRC)

PADIRC, PBDIRC, PCDIRC, PDIRC, PEDIRC and PFDIRC are six 32-bit PORT direction clear registers. They are write-only registers.

PADIRS, PBDIRC, PCDIRC, PDDIRC, PEDIRC, PFDIRC — **0x10010068, 0x10010168, 0x10010268, 0x10010368, 0x10010468, 0x10010568**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DIRC31	DIRC30	DIRC29	DIRC28	DIRC27	DIRC26	DIRC25	DIRC24	DIRC23	DIRC22	DIRC21	DIRC20	DIRC19	DIRC18	DIRC17	DIRC16	DIRC15	DIRC14	DIRC13	DIRC12	DIRC11	DIRC10	DIRC09	DIRC08	DIRC07	DIRC06	DIRC05	DIRC04	DIRC03	DIRC02	DIRC01	DIRC00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	DIRC n	Writing 1 to DIRC n will set DIR n to 0 in register PXDIR. Writing 0 to DIRC n will no use.	W

PADIRC bits 31-0 correspond to PA31-0; PBDIRC to PB31-0; PCDIRC to PC31-0; PDDIRC to PD31-0; PEDIRC to PE31-0 and PFDIRC to PF 31-0.

29.2.20 PORT Trigger Register (PxTRG)

PATRG, PBTRG, PCTRG, PDTRG, PETRG and PFTRG are six 32-bit PORT trigger registers. They are read-only registers.

PATRG, PBTRG, PCTRG, PDTRG, PETRG, PFTRG — **0x10010070, 0x10010170, 0x10010270, 0x10010370, 0x10010470, 0x10010570**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TRIG31	TRIG30	TRIG29	TRIG28	TRIG27	TRIG26	TRIG25	TRIG24	TRIG23	TRIG22	TRIG21	TRIG20	TRIG19	TRIG18	TRIG17	TRIG16	TRIG15	TRIG14	TRIG13	TRIG12	TRIG11	TRIG10	TRIG09	TRIG08	TRIG07	TRIG06	TRIG05	TRIG04	TRIG03	TRIG02	TRIG01	TRIG00
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
N	TRIG n	Where n = 0 ~ 31 and TRIG n = TRIG00 ~ TRIG31. TRIG n is used for setting the trigger mode for interrupt. When GPIO is used as interrupt function: 0: Level Trigger Interrupt 1: Edge Trigger Interrupt When GPIO is used as alternate function: 0: Alternate Function Group 0 1: Alternate Function Group 1	R

PATRG bits 31-0 correspond to PA31-0; PBTRG to PB31-0; PCTRG to PC31-0; PDTRG to PD31-0; PETRG to PE31-0 and PFTRG to PF 31-0.

29.2.21 PORT Trigger Set Register (PxTRGS)

PATRGS, PBTRGS, PCTRGS, PDTRGS, PETRGS and PFTRGS are six 32-bit PORT trigger set registers. They are write-only registers.

PATRGS, PBTRGS, PCTRGS, PDTRGS, PETRGS, PFTRGS — **0x10010074, 0x10010174, 0x10010274, 0x10010374, 0x10010474, 0x10010574**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TRIGS31	TRIGS30	TRIGS29	TRIGS28	TRIGS27	TRIGS26	TRIGS25	TRIGS24	TRIGS23	TRIGS22	TRIGS21	TRIGS20	TRIGS19	TRIGS18	TRIGS17	TRIGS16	TRIGS15	TRIGS14	TRIGS13	TRIGS12	TRIGS11	TRIGS10	TRIGS09	TRIGS08	TRIGS07	TRIGS06	TRIGS05	TRIGS04	TRIGS03	TRIGS02	TRIGS01	TRIGS00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
N	TRIGS n	Writing 1 to TRIGS n will set TRIG n to 1 in register PXTRG. Writing 0 to TRIGS n will no use.	W

PATRGS bits 31-0 correspond to PA31-0; PBTRGS to PB31-0; PCTRGS to PC31-0; PDTRGS to PD31-0; PETRGS to PE31-0 and PFTRGS to PF 31-0.

29.2.22 PORT Trigger Clear Register (PxTRGC)

PATRGC, PBTRGC, PCTRGC, PDTRGC, PETRGC and PFTRGC are six 32-bit PORT trigger clear registers. They are write-only registers.

PATRGC, PBTRGC, PCTRGC, PDTRGC, PETRGC, PFTRGC — **0x10010078, 0x10010178, 0x10010278, 0x10010378, 0x10010478, 0x10010578**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TRIGC31	TRIGC30	TRIGC29	TRIGC28	TRIGC27	TRIGC26	TRIGC25	TRIGC24	TRIGC23	TRIGC22	TRIGC21	TRIGC20	TRIGC19	TRIGC18	TRIGC17	TRIGC16	TRIGC15	TRIGC14	TRIGC13	TRIGC12	TRIGC11	TRIGC10	TRIGC09	TRIGC08	TRIGC07	TRIGC06	TRIGC05	TRIGC04	TRIGC03	TRIGC02	TRIGC01	TRIGC00
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	R/W
n	TRIGC n	Writing 1 to TRIGC n will set TRIG n to 0 in register PXTRG. Writing 0 to TRIGC n will no use.	W

PATRGC bits 31-0 correspond to PA31-0; PBTRGC to PB31-0; PCTRGC to PC31-0; PDTRGC to PD31-0; PETRGC to PE31-0 and PFTRGC to PF 31-0.

29.2.23 PORT FLAG Register (PxFLG)

PAFLG, PBFLG, PCFLG, PDFLG, PEFLG and PFFLG are six 32-bit GPIO FLAG registers. They are read-only registers.

PAFLG, PBFLG, PCFLG, PDFLG, PEFLG, PFFLG — **0x10010080, 0x10010180, 0x10010280, 0x10010380, 0x10010480, 0x10010580**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLAG31	FLAG30	FLAG29	FLAG28	FLAG27	FLAG26	FLAG25	FLAG24	FLAG23	FLAG22	FLAG21	FLAG20	FLAG19	FLAG18	FLAG17	FLAG16	FLAG15	FLAG14	FLAG13	FLAG12	FLAG11	FLAG10	FLAG09	FLAG08	FLAG07	FLAG06	FLAG05	FLAG04	FLAG03	FLAG02	FLAG01	FLAG00
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
n	FLAG n	Where n = 0 ~ 31 and FLAG n = FLAG00 ~ FLAG31. FLAG n is interrupt flag bit for checking the interrupt whether to happen. When GPIO is used as interrupt function and the interrupt happened, the FLAG n in PXFLG will be set to 1.	R

PAFLG bits 31-0 correspond to PA31-0; PBFLG to PB31-0; PCFLG to PC31-0; PDFLG to PD31-0; PEFLG to PE31-0 and PFFLG to PF 31-0.

29.2.24 PORT FLAG Clear Register (PxFLGC)

PAFLGC, PBFLGC, PCFLGC, PDFLGC, PEFLGC and PFFLGC are six 32-bit GPIO FLAG Clear registers. They are read-only registers.

PAFLGC, PBFLGC, PCFLGC, PDFLGC, PEFLGC, PFFLGC — **0x10010014, 0x10010114, 0x10010214, 0x10010314, 0x10010414,0x10010514**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FLAGC31	FLAGC30	FLAGC29	FLAGC28	FLAGC27	FLAGC26	FLAGC25	FLAGC24	FLAGC23	FLAGC22	FLAGC21	FLAGC20	FLAGC19	FLAGC18	FLAGC17	FLAGC16	FLAGC15	FLAGC14	FLAGC13	FLAGC12	FLAGC11	FLAGC10	FLAGC09	FLAGC08	FLAGC07	FLAGC06	FLAGC05	FLAGC04	FLAGC03	FLAGC02	FLAGC01	FLAGC00
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
n	FLAGC n	When GPIO is used as interrupt function and when write 1 to the bit, the bit FLAG n in PXFLG will be cleared.	R

PAFLGC bits 31-0 correspond to PA31-0; PBFLGC to PB31-0; PCFLGC to PC31-0; PDFLGC to PD31-0; PEFLGC to PE31-0 and PFFLGC to PF 31-0.

29.3 Program Guide

PXFUN	PXTRG	PXSEL	GPIO	INT	FUN0	FUN1	FUN2	FUN3
0	X	0	Y					
0	X	1		Y				
1	0	0			Y			
1	0	1				Y		
1	1	0					Y	
1	1	1						Y

29.3.1 GPIO Function Guide

1 Set PXFUN to 0 by writing 1 to register PXFUNC.
2 Set PXSEL to 0 by writing 1 to register PXSELC.
3 Set PXDIR to choose the direction of GPIO.
4 Others.
   a You can read the PORT PIN level by reading register PXPIN.
   b You can use register PXDAT as normal data register. The register can be set by register PXDATS and PXDATC.
   c You can config PXPE to enable internal pull-up/down resistor or not.

29.3.2 Alternate Function Guide

1 Set PXFUN to 0 by writing 1 to register PXFUNC. (Ready state)
2 Set PXTRG to 0 to choose the alternate function group 0 by writing 1 to register PXTRGC.
  Set PXTRG to 1 to choose the alternate function group 1 by writing 1 to register PXTRGS.
3 Set PXSEL to choose the alternate function 0 of group by writing 1 to register PXSELC.
  Set PXSEL to choose the alternate function 1 of group by writing 1 to register PXSELS.
4 Set PXFUN to choose the function of alternate function by writing 1 to register PXFUNS.

29.3.3 Interrupt Function Guide

First you should keep GPIO status.

1 Set PXIM by writing 1 to register PXIMS.
2 Set PXTRG to choose the interrupt trigger mode by writing 1 to register PXTRGS or PXTRGC.
3 Set PXFUN to choose the function of GPIO / Interrupt by writing 1 to register or PXFUNC.
4 Set PXSEL to choose the Interrupt function by writing 1 to register PXSELS.
5 Set PXDIR to choose the direction of interrupt trigger by writing 1 to register PXDIRS or PXDIRC.
6 Set the PXFLGC register to clear the interrupt flag.
7 Clear PXIM by writing 1 to register PXIMC to enable the GPIO interrupt.
8 Others.

You should check the level interrupt whether to happen as follows:

a When the PIN level read from register PXPIN is the same with what you have set in register PXTRG and PXDIR, and then the level interrupt happened.

b When the PIN level read from register PXPIN is different from what you have set in register PXTRG and PXDIR, and then the level interrupt did not happen.

29.3.4 Disable Interrupt Function Guide

1 Set PXIM by writing 1 to register PXIMS.
2 Set PXTRG to 0 by writing 1 to register PXTRGC.
3 Set PXDIR to 0 by writing 1 to register PXDIRC.
4 Set PXFUN to 0 by writing 1 to register or PXFUNC.
5 Set PXSEL to 0 by writing 1 to register PXSELC.

30 I2C Controller

30.1 Overview

The I2C bus is a two-wire serial interface, consisting of a serial data line (SDA) and a serial clock (SCL). These wires carry information between the devices connected to the bus. Each device is recognized by a unique address and can operate as either a “transmitter” or “receiver,” depending on the function of the device. Devices can also be considered as masters or slaves when performing data transfers. A master is a device that initiates a data transfer on the bus and generates the clock signals to permit that transfer. At that time, any device addressed is considered a slave. Operating as a master and slave simultaneously is not supported.

30.1.1 Features

- Two-wire I2C serial interface – consists of a serial data line (SDA) and a serial clock (SCL)
- Two speeds
  - Standard mode (100 Kb/s)
  - Fast mode (400 Kb/s)
- Device clock is identical with pclk
- Programmable SCL generator
- Master or slave I2C operation
- 7-bit addressing
- 2-level transmit and receive buffers
- Interrupt operation
- The number of devices that you can connect to the same I2C-bus is limited only by the maximum bus capacitance of 400pF
- APB interface

30.1.2 Pin Description

Table 30-1 I2C Pin Description

Name	Width	IO	Description
SDA	1-bit	IO	I2C serial data
SCL	1-bit	IO	I2C serial clock

30.2 Registers

30.2.1 Registers Memory Map

A read operation to an address location that contains unused bits results in a 0 value being returned on each of the unused bits.

Table 30-2 Registers Memory Map

Name	Address Offset	Description	Width	R/W	Reset
I2CCON	0x00	I2C control	7bits	RW	0x7D
I2CTAR	0x04	I2C target address	13bits	RW	0x1055
I2CSAR	0x08	I2C slave address	10bits	RW	0x55
I2CDC	0x10	I2C data buffer and command	9bits	RW	0x000
I2CSHCNT	0x14	Standard speed I2C SCL high count	16bits	RW	0x0190
I2CSLCNT	0x18	Standard speed I2C SCL low count	16bits	RW	0x01d6
I2CFHCNT	0x1C	Fast speed I2C SCL high count	16bits	RW	0x003c
I2CFLCNT	0x20	Fast speed I2C SCL low count	16bits	RW	0x0082
I2CINTST	0x2C	I2C Interrupt Status	12bits	R	0x0
I2CINTM	0x30	I2C Interrupt Mask	12bits	R/W	12'h8ff
I2CRXTL	0x38	I2C RxFIFO Threshold	8 bits	R/W	0x0
I2CTXTL	0x3C	I2C TxFIFO Threshold	8 bits	R/W	0x0
I2CCINT	0x40	Clear Combined and Individual Interrupts	1 bit	R	0x0
I2CCRXUF	0x44	Clear RXUF Interrupt	1 bit	R	0x0
I2CCRXOF	0x48	Clear RX_OVER Interrupt	1 bit	R	0x0
I2CCTXOF	0x4C	Clear TX_OVER Interrupt	1 bit	R	0x0
I2CCRXREQ	0x50	Clear RDREQ Interrupt	1 bit	R	0x0
I2CCTXABT	0x54	Clear TX_ABRT Interrupt	1 bit	R	0x0
I2CCRXDN	0x58	Clear RX_DONE Interrupt	1 bit	R	0x0
I2CCACT	0x5c	Clear ACTIVITY Interrupt	1 bit	R	0x0
I2CCSTP	0x60	Clear STOP Interrupt	1 bit	R	0x0
I2CCSTT	0x64	Clear START Interrupt	1 bit	R	0x0
I2CCGC	0x68	Clear GEN_CALL Interrupt	1 bit	R	0x0

I2CENB	0x6C	I2C Enable	1 bit	R/W	
I2CST	0x70	I2C Status register	7 bits	R	0x6
I2CABTSRC	0x80	I2C Transmit Abort Status Register	16 bits	R	0x0
I2CSDASU	0x94	I2C SDA Setup Register	8 bits	R/W	0x64
I2CACKGC	0x98	I2C ACK General Call Register	1 bit	R/W	0x0
I2CENBST	0x9C	I2C Enable Status Register	3 bits	R	0x0

30.2.2 Registers and Fields Description

30.2.2.1 I2CCON

I2CCON **BASE + 0x0000**

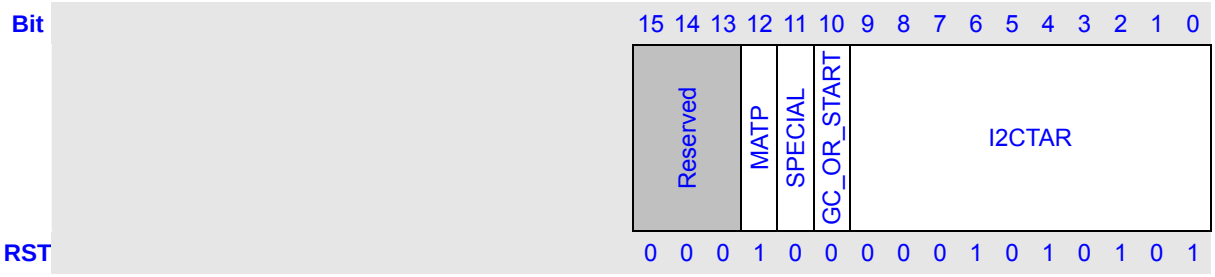
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									SLVDIS	REST	MATP	SATP	SPD		MD
RST	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	1

Bits	Name	Description	R/W
15:7	Reserved	Reserved.	N/A
6	SLVDIS	This bit controls whether I2C has its slave disabled after reset. 0: slave is enabled 1: slave is disabled	R/W
5	REST	Determines whether RESTART conditions may be sent when acting as a master. 0: disable 1: enable	R/W
4	MATP	This bit controls whether the I2C starts its transfers in 7- or 10-bit addressing mode when acting as a master. The function of this bit is handled by bit 12 of I2CTAR register. 0: 7-bit addressing 1: 10-bit addressing	R
3	SATP	When acting as a slave, this bit controls whether the I2C responds to 7- or 10-bit addresses. 0: 7-bit addressing 1: 10-bit addressing	R/W
2:1	SPD	These bits control at which speed the I2C operates. 0: standard mode (100 kbps)	R/W

		1: fast mode (400 kbps)	
0	MD	This bit controls whether the I2C master is enabled. 0: master disabled 1: master enabled	R/W

30.2.2.2 I2CTAR

I2CTAR BASE + 0x0004



Bits	Name	Description	R/W
15:13	Reserved	Reserved.	R
12	MATP	This bit controls whether the I2C starts its transfers in 7- or 10-bit addressing mode when acting as a master. 0: 7-bit addressing 1: 10-bit addressing	RW
11	SPECIAL	This bit indicates whether software performs a General Call or START BYTE command. 0: ignore bit 10 GC_OR_START and use I2CTAR normally 1: perform special I2C command as specified in GC_OR_START bit	RW
10	GC_OR_START	If bit 11 (SPECIAL) is set to 1, then this bit indicates whether a General Call or START byte command is to be performed by the I2C. 0: General Call Address – after issuing a General Call, only writes may be performed. Attempting to issue a read command results in setting bit 6 (TX_ABRT) of the I2CINTST register. The I2C remains in General Call mode until the SPECIAL bit value (bit 11) is cleared. 1: START BYTE	RW
9:0	I2CTAR	This is the target address for any master transaction. When transmitting a General Call, these bits are ignored. To generate a START BYTE, the CPU needs to write only once into these bits. If the I2CTAR and I2CSAR are the same, loopback exists but the FIFOs are shared between master and slave, so full loopback is not feasible. Only one direction loopback mode is supported (simplex), not duplex. A master cannot transmit to itself; it can transmit to only a slave.	RW

NOTES:

1 It is not necessary to perform any write to this register if I2C is enabled as an I2C slave only.

30.2.2.3 I2CSAR

I2CSAR BASE + 0x0008

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							I2CSAR								
RST	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1

Bits	Name	Description	R/W
15:10	Reserved	Reserved.	N/A
9:0	I2CSAR	The I2CSAR holds the slave address when the I2C is operating as a slave. For 7-bit addressing, only I2CSAR [6:0] is used. This register can be written only when the I2C interface is disabled. Writes at other times have no effect.	RW

NOTES:

1 It is not necessary to perform any write to this register if I2C is enabled as an I2C master only.

30.2.2.4 I2CDC

I2C Rx/Tx Data Buffer and Command Register; this is the register the CPU writes to when filling the TX FIFO and the CPU reads from when retrieving bytes from RX FIFO.

I2CDC BASE + 0x0010

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							CMD	DAT							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:9	Reserved	Reserved.	R
8	CMD	This bit controls whether a read or a write is performed. This bit does not control the direction when the I2C acts as a slave. It controls only the direction when it acts as a master. 1: Read 0: Write	RW

7:0	DAT	This register contains the data to be transmitted or received on the I2C bus.	RW

30.2.2.5 I2CSHCNT

I2CSHCNT **BASE + 0x0014**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	I2CSHCNT															
RST	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0

Bits	Name	Description	R/W
15:0	I2CSHCNT	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. The register sets the SCL clock high-period count for standard speed. This register can be written only when the I2C interface is disabled. Writes at other times have no effect. SCL high time of i2c is (I2CSHCNT + 8) i2c_clk periods.	RW

NOTES:

1 Minimum value allowed for the I2CSHCNT registers is 6.

30.2.2.6 I2CSLCNT

I2CSLCNT **BASE + 0x0018**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	I2CSLCNT															
RST	0	0	0	0	0	0	0	1	1	1	0	1	0	1	1	0

Bits	Name	Description	R/W
15:0	I2CSLCNT	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for standard speed. This register can be written only when the I2C interface is disabled. Writes at other times have no effect. SCL low time of i2c is (I2CSLCNT + 1) i2c_clk periods.	RW

NOTES:

1 Minimum value that can be programmed in the I2CSLCNT registers is 8.

30.2.2.7 I2CFHCNT

I2CFHCNT BASE + 0x001C

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	I2CFHCNT															
RST	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0

Bits	Name	Description	R/W
15:0	I2CFHCNT	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for fast speed. This register can be written only when the I2C interface is disabled. Writes at other times have no effect.	RW

NOTES:

1 Minimum value allowed for the I2CSHCNT registers is 6.

30.2.2.8 I2CFLCNT

I2CFLCNT BASE + 0x0020

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	I2CFLCNT															
RST	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0

Bits	Name	Description	R/W
15:0	I2CFLCNT	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for fast speed. This register can be written only when the I2C interface is disabled. Writes at other times have no effect.	RW

NOTES:

1 Minimum value that can be programmed in the I2CSLCNT registers is 8.

30.2.2.9 I2CINTST

Each bit in this register has a corresponding mask bit in the I2CINTM register. These bits are cleared by reading the matching interrupt clear register.

I2CINTST **BASE + 0x002C**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				IGC	ISTT	ISTP	IACT	RXDN	TXABT	RDREQ	TXEMP	TXOF	RXFL	RXOF	RXUF
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:12	Reserved	Reserved.	N/A
11	IGC	Set only when a General Call address is received and it is acknowledged. It stays set until it is cleared either by disabling I2C or when the CPU reads bit 0 of the I2CCGC register. I2C stores the received data in the Rx buffer.	R
10	ISTT	Indicates whether a START or RESTART condition has occurred on the I2C interface regardless of whether I2C is operating in slave or master mode.	R
9	ISTP	Indicates whether a STOP condition has occurred on the I2C interface regardless of whether I2C is operating in slave or master mode.	R
8	IACT	This bit captures I2C activity and stays set until it is cleared. There are four ways to clear it: 1 Disabling the I2C 2 Reading the I2CCACT register 3 Reading the I2CCINT register 4 System reset Once this bit is set, it stays set unless one of the four methods is used to clear it. Even if the I2C module is idle, this bit remains set until cleared, indicating that there was activity on the bus.	R
7	RXDN	When the I2C is acting as a slave-transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. This occurs on the last byte of the transmission, indicating that the transmission is done.	R
6	TXABT	This bit indicates if I2C, as an I2C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I2C master or an I2C slave, and is referred to as a “transmit abort”. When this bit is set to 1, the I2CABTSRC register indicates the reason why the transmit abort takes places. **NOTE:** The I2C flushes/resets/empties the TX FIFO whenever this bit is set. The TX FIFO remains in this flushed state until the register	R

		I2CCTXABT is read. Once this read is performed, the TX FIFO is then ready to accept more data bytes from the APB interface.	
5	RDREQ	This bit is set to 1 when I2C is acting as a slave and another I2C master is attempting to read data from I2C. The I2C holds the I2C bus in a wait state (SCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the I2CDC register. This bit is set to 0 just after the processor reads the I2CCRREQregister.	R
4	TXEMP	This bit is set to 1 when the transmit buffer is empty. It is automatically cleared by hardware when the buffer is not empty. When the I2CENB bit 0 is 0, the TX FIFO is flushed and held in reset. There the TX FIFO looks like it has no data within it, so this bit is set to 1, provided there is activity in the master or slave state machines. When there is no longer activity, then with i2c_en=0, this bit is set to 0.	R
3	TXOF	Set during transmit if the transmit buffer is filled to I2CTX_BUFFER_DEPTH and the processor attempts to issue another I2C command by writing to the I2CDC register. When the module is disabled, this bit keeps its level until the master or slave state machines go into idle, and when i2c_en goes to 0, this interrupt is cleared.	R
2	RXFL	Set when the receive buffer reaches 2. It is automatically cleared by hardware when buffer level goes below the threshold. If the module is disabled (I2CENB [0] =0), the RX FIFO is flushed and held in reset; therefore the RX FIFO is not full. So this bit is cleared once the I2CENB bit 0 is programmed with a 0, regardless of the activity that continues.	R
1	RXOF	Set if the receive buffer is completely filled to 2 and an additional byte is received from an external I2C device. The I2C acknowledges this, but any data bytes received after the FIFO is full are lost. If the module is disabled (I2CENB[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when i2c_en goes to 0, this interrupt is cleared.	R
0	RXUF	Set if the processor attempts to read the receive buffer when it is empty by reading from the I2CDC register. If the module is disabled (I2CENB[0]=0), this bit keeps its level until the master or slave state machines go into idle, and when i2c_en goes to 0, this interrupt is cleared.	R

30.2.2.10 I2CINTM

These bits mask their corresponding interrupt status bits. They are active high; a value of 1 prevents a bit from generating an interrupt.

I2CINTM BASE + 0x0030

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				MIGC	MISTT	MISTP	MIACT	MRXDN	MTXABT	MRDREQ	MTXEMP	MTXOF	MRXFL	MRXOF	MRXUF
RST	0	0	0	0	1	0	0	0	1	1	1	1	1	1	1	1

Bits	Name	Description	R/W
15:12	Reserved	Reserved.	N/A
11	MIGC	These bits mask their corresponding interrupt status bits in the I2CINTST register.	RW
10	MISTT		RW
9	MISTP		RW
8	MIACT		RW
7	MRXDN		RW
6	MTXABT		RW
5	MRDREQ		RW
4	MTXEMP		RW
3	MTXOF		RW
2	MRXFL		RW
1	MRXOF		RW
0	MRXUF		RW

30.2.2.11 I2CRXTL

I2CRXTL BASE + 0x0038

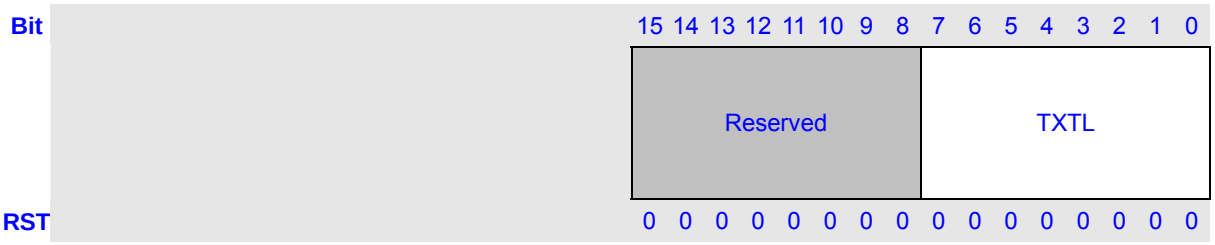
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								RXTL							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:8	Reserved	Reserved.	N/A
7:0	RXTL	Receive FIFO Threshold Level. Controls the level of entries (or above) that triggers the RxFIFO full interrupt. The valid range is 0-255, with the additional restriction that	RW

		hardware does not allow this value to be set to a value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 1 entry, and a value of 255 sets the threshold for 256 entries.	

30.2.2.12 I2CTXTL

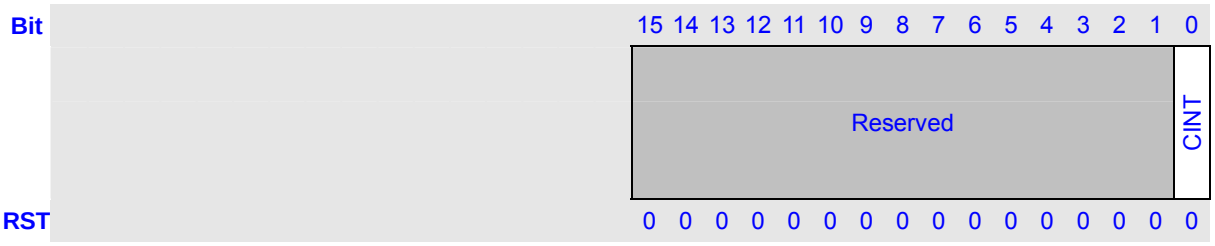
I2CTXTL BASE + 0x003C



Bits	Name	Description	R/W
15:8	Reserved	Reserved.	N/A
7:0	TXTL	Transmit FIFO Threshold Level. Controls the level of entries (or below) that triggers the TxFIFO empty interrupt. The valid range is 0-255, with the additional restriction that it may not be set to value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 0 entries, and a value of 255 sets the threshold for 255 entries.	RW

30.2.2.13 I2CCINT

I2CCINT BASE + 0x0040



Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CINT	Read this register to clear the combined interrupt, all individual interrupts, and the I2CABTSRC register. This bit does not clear hardware clearable	R

		interrupts but software clearable interrupts. Refer to Bit 9 of the I2CABTSRC register for an exception to clearing I2CABTSRC.	

30.2.2.14 I2CCRXUF

I2CCRXUF BASE + 0x0044

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CRXUF
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CRXUF	Read this register to clear the RXUF interrupt (bit 0) of the I2CINTST register.	R

30.2.2.15 I2CCRXOF

I2CCRXOF BASE + 0x0048

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CRXUF
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CRXUF	Read this register to clear the RXUF interrupt (bit 0) of the I2CINTST register.	R

30.2.2.16 I2CCTXOF

I2CCTXOF BASE + 0x004C

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CTXOF
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CTXOF	Read this register to clear the TX_OVER interrupt (bit 3) of the I2CINTST register.	R

30.2.2.17 I2CCRXREQ

I2CCRXREQ BASE + 0x0050

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CLRDREQ
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CLRDREQ	Read this register to clear the RDREQ interrupt (bit 5) of the I2CINTST register.	R

30.2.2.18 I2CCTXABT

I2CCTXABT BASE + 0x0054

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CTXABT
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CTXABT	Read this register to clear the TX_ABRT interrupt (bit 6) of the I2CINTST register, and the I2CABTSRC register. This also releases the TX FIFO from the flushed/reset state, allowing more writes to the TX FIFO. Refer to Bit 9 of the I2CABTSRC register for an exception to clearing I2CABTSRC.	R

30.2.2.19 I2CCRXDN

I2CCRXDN BASE + 0x0058

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CRXDN
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CRXDN	Read this register to clear the RX_DONE interrupt (bit 7) of the I2CINTST register.	R

30.2.2.20 I2CCACT

I2CCACT BASE + 0x005C

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CACT
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CACT	Reading this register clears the ACTIVITY interrupt if the I2C is not active anymore. If the I2C module is still active on the bus, the ACTIVITY interrupt bit continues to be set. It is automatically cleared by hardware if the module is disabled and if there is no further activity on the bus. The value read from this register to get status of the ACTIVITY interrupt (bit 8)	R

		of the I2CINTST register.	

30.2.2.21 I2CCSTP

I2CCSTP BASE + 0x0060

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CLRDREQ
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CSTP	Read this register to clear the STOP interrupt (bit 9) of the I2CINTST register.	R

30.2.2.22 I2CCSTT

I2CCSTT BASE + 0x0064

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CSTP
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CSTP	Read this register to clear the START interrupt (bit 10) of the I2CINTST register.	R

30.2.2.23 I2CCGC

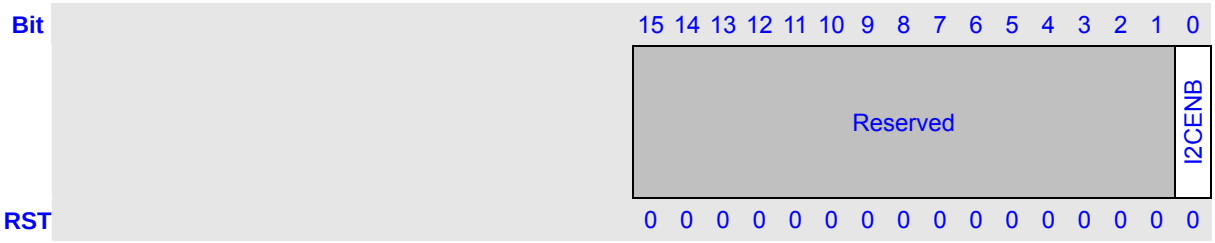
I2CCGC BASE + 0x0068

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															CGC
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	CGC	Read this register to clear the GEN_CALL interrupt (bit 11) of I2CINTST register.	R

30.2.2.24 I2CENB

I2CENB BASE + 0x006C



Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	I2CENB	Controls whether the I2C is enabled. 0: Disables I2C (TX and RX FIFOs are held in an erased state) 1: Enables I2C Software can disable I2C while it is active. However, it is important that care be taken to ensure that I2C is disabled properly. When I2C is disabled, the following occurs: 1 The TX FIFO and RX FIFO get flushed. 2 Status bits in the I2CINTST register are still active until I2C goes into IDLE state. If the module is transmitting, it stops as well as deletes the contents of the transmit buffer after the current transfer is complete. If the module is receiving, the I2C stops the current transfer at the end of the current byte and does not acknowledge the transfer.	R

30.2.2.25 I2CST

This is a read-only register used to indicate the current transfer status and FIFO status. The status register may be read at any time. None of the bits in this register request an interrupt.

When the I2C is disabled by writing 0 in bit 0 of the I2CENB register:

- Bits 1 and 2 are set to 1
- Bits 3 and 4 are set to 0

When the master or slave state machine goes to idle and ic_en=0:

- Bits 5 and 6 are set to 0

I2CST BASE + 0x0070

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved									SLVACT	MSTACT	RFF	RFNE	TFE	TFNF	ACT
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Bits	Name	Description	R/W
15:7	Reserved	Reserved.	R
6	SLVACT	Slave FSM Activity Status. 0: Slave FSM is in IDLE state 1: Slave FSM is not in IDLE state	R
5	MSTACT	Master FSM Activity Status. 0: Master FSM is in IDLE state 1: Master FSM is not in IDLE state	R
4	RFF	Receive FIFO Completely Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0: Receive FIFO is not full 1: Receive FIFO is full	R
3	RFNE	Receive FIFO Not Empty. This bit is set when the receive FIFO contains one or more entries; it is cleared when the receive FIFO is empty. 0: Receive FIFO is empty 1: Receive FIFO is not empty	R
2	TFE	Transmit FIFO Completely Empty. When the transmit FIFO is completely empty, this bit is set. When it contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0: Transmit FIFO is not empty 1: Transmit FIFO is empty	R
1	TFNF	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0: Transmit FIFO is full 1: Transmit FIFO is not full	R
0	ACT	I2C Activity Status. The OR of SLVACT and MSTACT bits.	R

30.2.2.26 I2CABTSRC

This register has 16 bits that indicate the source of the TX_ABRT bit. Except for Bit 9, this register is cleared whenever the I2CCTXABT register or the I2CCINT register is read. To clear Bit 9, the source of the SBYTE_NORSTRT must be fixed first; RESTART must be enabled (I2CCON[5]=1), the

SPECIAL bit must be cleared (I2CTAR[11]), or the GC_OR_START bit must be cleared (I2CTAR[10]). Once the source of the SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the SBYTE_NORSTRT is not fixed before attempting to clear this bit, Bit 9 clears for one cycle and is then re-asserted.

I2CABTSRC **BASE + 0x0080**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SLVRD_INTX	SLV_ARBLOST	SLVFLUSH_TXFIFO	ARB_LOST	ABRT_MASTER_DIS	ABRT_10B_RD_NORSTRT	SBYTE_NORSTRT	ABRT_HS_NORSTRT	SBYTE_ACKDET	ABRT_HS_ACKDET	ABRT_GCALL_READ	ABRT_GCALL_N	ABRT_TXDATA_N OACK	ABRT_10ADDR2_NOACK	ABRT_10ADDR1_NOACK	ABRT_7B_ADDR_NOACK
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	R/W
15	SLVRD_INTX	1: When the processor side responds to a slave mode request for data to be transmitted to a remote master and user writes a 1 in CMD (bit 8) of I2CDC register.	R
14	SLV_ARBLOST	1: Slave lost the bus while transmitting data to a remote master. I2CABTSRC[12] is set at the same time. **NOTE:** Even though the slave never “owns” the bus, something could go wrong on the bus. This is a fail safe check. For instance, during a data transmission at the low-to-high transition of SCL, if what is on the data bus is not what is supposed to be transmitted, then I2C no longer own the bus. Reset value: 0x0.	R
13	SLVFLUSH_TXFIFO	1: Slave has received a read command and some data exists in the TX FIFO so the slave issues a TX_ABRT interrupt to flush old data in TX FIFO. Reset value: 0x0.	R
12	ARB_LOST	1: Master has lost arbitration, or if I2CABTSRC[14] is also set, then the slave transmitter has lost arbitration. **NOTE:** I2C can be both master and slave at the same time. Reset value: 0x0.	R
11	ABRT_MASTER_DIS	1: User tries to initiate a Master operation with the Master mode disabled. Reset value: 0x0.	R

10	ABRT_10B_RD_NORSTRT	1: The restart is disabled (I2CRESTART_EN bit (I2CCON[5]) = 0) and the master sends a read command in 10-bit addressing mode. Reset value: 0x0.	R
9	SBYTE_NORSTRT	To clear Bit 9, the source of the SBYTE_NORSTRT must be fixed first; restart must be enabled (I2CCON[5]=1), the SPECIAL bit must be cleared (I2CTAR[11]), or the GC_OR_START bit must be cleared (I2CTAR[10]). Once the source of the SBYTE_NORSTRT is fixed, then this bit can be cleared in the same manner as other bits in this register. If the source of the SBYTE_NORSTRT is not fixed before attempting to clear this bit, bit 9 clears for one cycle and then gets re-asserted. 1: The restart is disabled (I2CRESTART_EN bit (I2CCON[5]) = 0) and the user is trying to send a START Byte. Reset value: 0x0.	R
8	ABRT_HS_NORSTRT	1: The restart is disabled (I2CRESTART_EN bit (I2CCON[5]) = 0) and the user is trying to use the master to transfer data in High Speed mode. Reset value: 0x0.	R
7	SBYTE_ACKDET	1: Master has sent a START Byte and the START Byte was acknowledged (wrong behavior). Reset value: 0x0.	R
6	ABRT_HS_ACKDET	1: Master is in High Speed mode and the High Speed Master code was acknowledged (wrong behavior). Reset value: 0x0.	R
5	ABRT_GCALL_READ	1: I2C in master mode sent a General Call but the user programmed the byte following the General Call to be a read from the bus (I2CDC[9] is set to 1). Reset value: 0x0.	R
4	ABRT_GCALL_NOACK	1: I2C in master mode sent a General Call and no slave on the bus acknowledged the General Call. Reset value: 0x0.	R
3	ABRT_TXDATA_NOACK	1: This is a master-mode only bit. Master has received an acknowledgement for the address, but when it sent data byte(s) following the address, it did not receive an acknowledge from the remote slave(s). Reset value: 0x0.	R
2	ABRT_10ADDR2_NOACK	1: Master is in 10-bit address mode and the second address byte of the 10-bit address was not acknowledged by any slave. Reset value: 0x0.	R
1	ABRT_10ADDR1_NOACK	1: Master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave.	R

		Reset value: 0x0.	
0	ABRT_7B_ADDR_NOACK	1: Master is in 7-bit addressing mode and the address sent was not acknowledged by any slave. Reset value: 0x0.	R

30.2.2.27 I2CSDASU

This register controls the amount of time delay (in terms of number of ic_clk clock periods) introduced in the rising edge of SCL, relative to SDA changing, when I2C services a read request in a slave-transmitter operation. The relevant I2C requirement is tSU:DAT (note 4) as detailed in the I2C Bus Specification.

NOTES:

1 The length of setup time is calculated using [(I2CSDASU - 1) * (ic_clk_period)], so if the user requires 10 ic_clk periods of setup time, they should program a value of 11.

I2CSDASU **BASE + 0x0094**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								SDASU							
RST	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0

Bits	Name	Description	R/W
15:8	Reserved	Reserved.	N/A
7:0	SDASU	SDA Setup. It is recommended that if the required delay is 1000ns, then for an ic_clk frequency of 10 MHz, I2CSDASU should be programmed to a value of 11.	R/W

30.2.2.28 I2CACKGC

The register controls whether I2C responds with a ACK or NACK when it receives an I2C General Call address.

I2CACKGC **BASE + 0x0098**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															ACKGC
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	Name	Description	R/W
15:1	Reserved	Reserved.	N/A
0	ACKGC	ACK General Call. When set to 1, I2C responds with a ACK (by asserting ic_data_oe) when it receives a General Call. When set to 0, the I2C does not generate General Call interrupts.	R/W

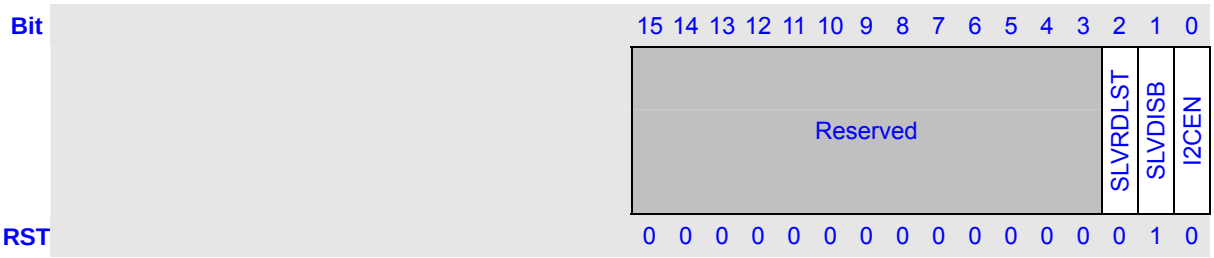
30.2.2.29 I2CENBST

The register is used to report the I2C hardware status when the I2CENB register is set from 1 to 0; that is, when I2C is disabled.

If I2CENB has been set to 1, bits 2:1 are forced to 0, and bit 0 is forced to 1.

If I2CENB has been set to 0, bits 2:1 is only be valid as soon as bit 0 is read as '0'.

I2CENBST **BASE + 0x009C**



Bits	Name	Description	R/W
15:3	Reserved	Reserved.	N/A
2	SLVRDLST	Slave Received Data Lost. This bit indicates if a Slave-Receiver operation has been aborted with at least one data byte received from an I2C transfer due to the setting of I2CENB from 1 to 0. When read as 1, I2C is deemed to have been actively engaged in an aborted I2C transfer (with matching address) and the data phase of the I2C transfer has been entered, even though a data byte has been responded with a NACK. **NOTES:** 1 If the remote I2C master terminates the transfer with a STOP condition before the I2C has a chance to NACK a transfer, and I2CENB has been set to 0, then this bit is also set to 1. When read as 0, I2C is deemed to have been disabled without being actively involved in the data phase of a Slave-Receiver transfer. 2 The CPU can safely read this bit when IC_EN (bit 0) is read as 0. Reset value: 0x0.	R
1	SLVDISB	Slave Disabled While Busy (Transmit, Receive). This bit indicates if a potential or active Slave operation has been aborted due to the setting of the I2CENB register from 1 to 0. This bit is set when the CPU writes a 0 to the I2CENB register while: (a) I2C is receiving the address byte of the Slave-Transmitter operation from a remote master;	R

		OR, (b) address and data bytes of the Slave-Receiver operation from a remote master. When read as 1, I2C is deemed to have forced a NACK during any part of an I2C transfer, irrespective of whether the I2C address matches the slave address set in I2C (I2CSAR register) OR if the transfer is completed before I2CENB is set to 0 but has not taken effect. **NOTE:** If the remote I2C master terminates the transfer with a STOP condition before the I2C has a chance to NACK a transfer, and I2CENB has been set to 0, then this bit will also be set to 1. When read as 0, I2C is deemed to have been disabled when there is master activity, or when the I2C bus is idle. **NOTE:** The CPU can safely read this bit when IC_EN (bit 0) is read as 0. Reset value: 0x0.	
0	I2CEN	ic_en Status. This bit always reflects the value driven on the output port ic_en. When read as 1, I2C is deemed to be in an enabled state. When read as 0, I2C is deemed completely inactive. **NOTE:** The CPU can safely read this bit anytime. When this bit is read as 0, the CPU can safely read SLVRDLST (bit 2) and SLVDISB (bit 1). Reset value: 0x0.	R

30.3 Operating Flow

This section provides information on the following topics:

- "Slave Mode Operation"
- "Master Mode Operation"
- "Disabling I2C"

NOTES:

1 It is important to note that the I2C should only be set to operate as an I2C Master, or I2C Slave, but not both simultaneously. This is achieved by ensuring that bit 6 (I2CSLAVE_DISABLE) and 0 (I2CMASTER_MODE) of the I2CCON register are never set to 0 and 1, respectively.

30.3.1 I2C Behavior

The I2C can be controlled via software to be either:

- An I2C master only, communicating with other I2C slaves;

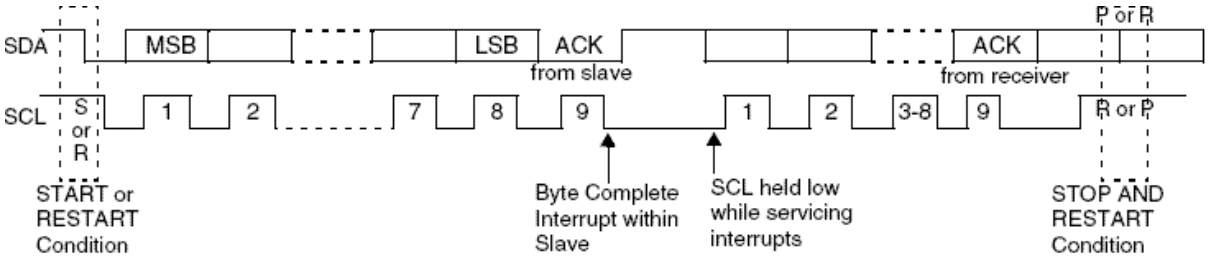
OR

- An I2C slave only, communicating with one more I2C masters.

The master is responsible for generating the clock and controlling the transfer of data. The slave is responsible for either transmitting or receiving data to/from the master. The device that is receiving data, which can be either a master or a slave, sends the acknowledgement of data. As mentioned previously, the I2C protocol also allows multiple masters to reside on the I2C bus and uses an arbitration procedure to determine bus ownership.

Each slave has a unique address that is determined by the system designer. When a master wants to communicate with a slave, the master transmits a START/RESTART condition that is then followed by the slave's address and a control bit (R/W) to determine if the master wants to transmit data or receive data from the slave. The slave then sends an acknowledge (ACK) pulse after the address.

If the master (master-transmitter) is writing to the slave (slave-receiver), the receiver gets one byte of data. This transaction continues until the master terminates the transmission with a STOP condition. If the master is reading from a slave (master-receiver), the slave transmits (slave-transmitter) a byte of data to the master, and the master then acknowledges the transaction with the ACK pulse. This transaction continues until the master terminates the transmission by not acknowledging (NACK) the transaction after the last byte is received, and then the master issues a STOP condition or addresses another slave after issuing a RESTART condition.



30.3.2 Master Mode Operation

This section includes the following topics:

- "Initial Configuration"
- "Dynamic I2CTAR or I2C10BITADDR_MASTER Update"
- "Master Transmit and Master Receive"

30.3.2.1 Configuration

To use the I2C as a master perform the following steps:

1 Disable the I2C by writing 0 to the I2CENB register. And wait for the I2CENBST.I2CEN = 0.

2 Write to the I2CCON register to set the maximum speed mode supported (bits 2:1) and the desired address mode of the I2C master-initiated transfers, either 7-bit or 10-bit addressing (bit 4).

3 Set the expected SCL frequency. I2CCON.SPD = 2'b01, only I2CSHCNT and I2CSLCNT are needed to be configured; I2CCON.SPD = 2'b10, only I2CFHCNT and I2CFLCNT are needed to be configured.

Supposed:

Tscl: I2C SCL period

Ti2c_clk: I2C device clock period

Tmin_scl_l: Protocol minimum SCL low time

Tmin_scl_h: Protocol minimum SCL high time

Then can get the equation:

Tscl = Ti2c_clk * ((I2C*HCNT + 8) + (I2C*LCNT + 1))

And the following conditions should be meted:

(I2C*HCNT + 8) * Ti2c_clk >= Tmin_scl_h

(I2C*LCNT + 1) * Ti2c_clk >= Tmin_scl_l

I2C*HCNT >= 6

I2C*LCNT >= 8

4 Write to the I2CTAR register the address of the I2C device to be addressed. It also indicates whether a General Call or a START BYTE command is going to be performed by I2C. The desired speed of the I2C master-initiated transfers, either 7-bit or 10-bit addressing, is controlled by the I2C10BITADDR_MASTER bit field (bit 12).

5 Enable the I2C by writing a 1 in the I2CENB register. And wait for the I2CENBST.I2CEN = 1.

6 Now write the transfer direction and data to be sent to the I2CDC register. If the I2CDC register is written before the I2C is enabled, the data and commands are lost as the buffers are kept cleared when I2C is not enabled.

NOTES:

1 For multiple I2C transfers, perform additional writes to the TX FIFO such that the TX FIFO does not become empty during the I2C transaction. If the TX FIFO is completely emptied at any stage, then further writes to the TX FIFO results in an independent I2C transaction.

30.3.2.2 Dynamic I2CTAR or I2C10BITADDR_MASTER Update

The I2C supports dynamic updating of the I2CTAR (bits 9:0) and I2C10BITADDR_MASTER (bit 12) bit fields of the I2CTAR register. You can dynamically write to the I2CTAR register provided the following conditions are met:

1 I2C is not enabled (I2CENB=0);

OR

2 I2C is enabled (I2CENB=1);
AND
I2C is NOT engaged in any Master (tx, rx) operation (I2CST[5]=0); AND
I2C is enabled to operate in Master mode (I2CCON[0]=1);
AND
there are NO entries in the TX FIFO (I2CST[2]=1)

30.3.2.3 Master Transmit and Master Receive

The I2C supports switching back and forth between reading and writing dynamically. To transmit data, write the data to be written to the lower byte of the I2C Rx/Tx Data Buffer and Command Register (I2CDC). The *CMD* bit [8] should be written to 0 for I2C write operations.

Subsequently, a read command may be issued by writing “don’t cares” to the lower byte of the I2CDC register, and a 1 should be written to the *CMD* bit.

30.3.3 Slave Mode Operation

This section includes the following procedures:

- “Initial Configuration”
- “Slave-Transmitter Operation for a Single Byte”
- “Slave-Receiver Operation for a Single Byte”
- “Slave-Transfer Operation For Bulk Transfers”

30.3.3.1 Initial Configuration

To use the I2C as a slave, perform the following steps:

1 Disable the I2C by writing a '0' to bit 0 of the I2CENB register.

2 Write to the I2CSAR register (bits 9:0) to set the slave address. This is the address to which the I2C responds.

3 Write to the I2CCON register to specify which type of addressing is supported (7- or 10-bit by setting bit 3). Enable the I2C in slave-only mode by writing a '0' into bit 6 (I2CSLAVE_DISABLE) and a '0' to bit 0 (MASTER_MODE).

NOTE:

Slaves and masters do not have to be programmed with the same type of addressing 7- or 10-bit address. For instance, a slave can be programmed with 7-bit addressing and a master with 10-bit addressing, and vice versa.

4 Enable the I2C by writing a '1' in bit 0 of the I2CENB register.

NOTE:

Depending on the reset values chosen, steps 2 and 3 may not be necessary because the reset values can be configured. For instance, if the device is only going to be a master, there would be no need to set the slave address because you can configure I2C to have the slave disabled after reset and to enable the master after reset. The values stored are static and do not need to be reprogrammed if the I2C is disabled.

30.3.3.2 Slave-Transmitter Operation for a Single Byte

When another I2C master device on the bus addresses the I2C and requests data, the I2C acts as a slave-transmitter and the following steps occur:

1 The other I2C master device initiates an I2C transfer with an address that matches the slave address in the I2CSAR register of the I2C.

2 The I2C acknowledges the sent address and recognizes the direction of the transfer to indicate that it is acting as a slave-transmitter.

3 The I2C asserts the RDREQ interrupt (bit 5 of the I2CINTST register) and holds the SCL line low. It is in a wait state until software responds.

If the RDREQ interrupt has been masked, due to I2CINTM[5] register (MRDREQ bit field) being set to 0, then it is recommended that a hardware and/or software timing routine be used to instruct the CPU to perform periodic reads of the I2CINTST register.

a Reads that indicate I2CINTST[5] (RDREQ bit field) being set to 1 must be treated as the equivalent of the RDREQ interrupt being asserted.

b Software must then act to satisfy the I2C transfer.

c The timing interval used should be in the order of 10 times the fastest SCL clock period the I2C can handle. For example, for 400 kb/s, the timing interval is 25us.

NOTE:

The value of 10 is recommended here because this is approximately the amount of time required for a single byte of data transferred on the I2C bus.

4 If there is any data remaining in the TX FIFO before receiving the read request, then the I2C asserts a TX_ABRT interrupt (bit 6 of the I2CINTST register) to flush the old data from

the TX FIFO.

NOTE:

Because the I2C's TX FIFO is forced into a flushed/reset state whenever a TX_ABRT event occurs, it is necessary for software to release the I2C from this state by reading the I2CCTXABT register before attempting to write into the TX FIFO. See register I2CINTST for more details.

If the TX_ABRT interrupt has been masked, due to of I2CINTM[6] register (MTX_ABRT bit field) being set to 0, then it is recommended that re-using the timing routine (described in the previous step), or a similar one, be used to read the I2CINTST register.

a Reads that indicate bit 6 (TXABT) being set to 1 must be treated as the equivalent of the TX_ABRT interrupt being asserted.

b There is no further action required from software.

c The timing interval used should be similar to that described in the previous step for the I2CINTST[5] register.

5 Software writes to the I2CDC register with the data to be written (by writing a '0' in bit 8).

6 Software must clear the RDREQ and TX_ABRT interrupts (bits 5 and 6, respectively) of the I2CINTST register before proceeding.

If the RDREQ and/or TX_ABRT interrupts have been masked, then clearing of the I2CINTST register will have already been performed when either the RDREQ or TXABT bit has been read as 1.

7 The I2C releases the SCL and transmits the byte.

8 The master may hold the I2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

30.3.3.3 Slave-Receiver Operation for a Single Byte

When another I2C master device on the bus addresses the I2C and is sending data, the I2C acts as a slave-receiver and the following steps occur:

1 The other I2C master device initiates an I2C transfer with an address that matches the I2C's slave address in the I2CSAR register.

2 The I2C acknowledges the sent address and recognizes the direction of the transfer to indicate that the I2C is acting as a slave-receiver.

3 I2C receives the transmitted byte and places it in the receive buffer.

NOTE:

If the RX FIFO is completely filled with data when a byte is pushed, then an overflow occurs and the I2C continues with subsequent I2C transfers. Because a NACK is not generated, software must recognize the overflow when indicated by the I2C (by the RXOF bit in the I2CINTST register) and take appropriate actions to recover from lost data. Hence, there is a real time constraint on software to service the RX FIFO before the latter overflow as there is no way to re-apply pressure to the remote transmitting master. You must select a deep enough RX FIFO depth to satisfy the interrupt service interval of their system.

4 I2C asserts the RX_FULL interrupt (I2CINTST[2] register).
If the RX_FULL interrupt has been masked, due to setting I2CINTM[2] register to 0 or setting I2CTX_TL to a value larger than 0, then it is recommended that a timing routine (described in "Slave-Transmitter Operation for a Single Byte" on page 57) be implemented for periodic reads of the "I2CST" on page 136 register. Reads of the I2CST register, with bit 3 (RFNE) set at 1, must then be treated by software as the equivalent of the RX_FULL interrupt being asserted.
5 Software may read the byte from the I2CDC register (bits 7:0).
6 The other master device may hold the I2C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

30.3.3.4 Slave-Transfer Operation For Bulk Transfers

In the standard I2C protocol, all transactions are single byte transactions and the programmer responds to a remote master read request by writing one byte into the slave's TX FIFO.

When a slave (slave-transmitter) is issued with a read request (RDREQ) from the remote master (master-receiver), at a minimum there should be at least one entry placed into the slave-transmitter's TX FIFO.

I2C is designed to handle more data in the TX FIFO so that subsequent read requests can take that data without raising an interrupt to get more data. Ultimately, this eliminates the possibility of significant latencies being incurred between raising the interrupt for data each time had there been a restriction of having only one entry placed in the TX FIFO.
This mode only occurs when I2C is acting as a slave-transmitter. If the remote master acknowledges the data sent by the slave-transmitter and there is no data in the slave's TX FIFO, the I2C holds the I2C SCL line low while it raises the read request interrupt (RDREQ) and waits for data to be written into the TX FIFO before it can be sent to the remote master.

If the RDREQ interrupt is masked, due to bit 5 (MRDREQ) of the I2CINTST register being set to 0, then it is recommended that a timing routine be used to activate periodic reads of the I2CINTST register. Reads of I2CINTST that return bit 5 (RDREQ) set to 1 must be treated as the equivalent of the RDREQ interrupt referred to in this section.

The RDREQ interrupt is raised upon a read request, and like interrupts, must be cleared when exiting the interrupt service handling routine (ISR). The ISR allows you to either write 1 byte or more than 1 byte into the TX FIFO. During the transmission of these bytes to the master, if the master acknowledges the last byte. Then the slave must raise the RDREQ again because the master is requesting for more data.

30.3.4 Disabling I2C

The register I2CENBST is added to allow software to unambiguously determine when the hardware

has completely shutdown in response to the I2CENB register being set from 1 to 0.

30.3.4.1 Procedure

1 Define a timer interval (ti2c_poll) equal to the 10 times the signaling period for the highest I2C transfer speed used in the system and supported by I2C. For example, if the highest I2C transfer mode is 400 kb/s, then this ti2c_poll is 25us.

2 Define a maximum time-out parameter, MAX_T_POLL_COUNT, such that if any repeated polling operation exceeds this maximum value, an error is reported.

3 Execute a blocking thread/process/function that prevents any further I2C master transactions to be started by software, but allows any pending transfers to be completed.

NOTE:

This step can be ignored if I2C is programmed to operate as an I2C slave only.

4 The variable POLL_COUNT is initialized to zero.

5 Set I2CENB to 0.

6 Read the I2CENBST register and test the I2C_EN bit (bit 0). Increment POLL_COUNT by one. If POLL_COUNT >= MAX_T_POLL_COUNT, exit with the relevant error code.

7 If I2CENBST[0] is 1, then sleep for ti2c_poll and proceed to the previous step. Otherwise, exit with a relevant success code.

31 Synchronous Serial Interface

31.1 Overview

The SSI is a full-duplex synchronous serial interface and can connect to a variety of external analog-to-digital (A/D) converters, audio and telecom codecs, and other devices that use serial protocols for transferring data. The SSI supports National's Microwire, Texas Instruments Synchronous Serial Protocol (SSP), and Motorola's Serial Peripheral Interface (SPI) protocol.

The SSI operates in master mode (the attached peripheral functions as a slave) and supports serial bit rates from 7.2 KHz to 54 MHz. Serial data formats may range from 2 to 17 bits in length. The SSI provides 128 entries deep x 17 bits wide transmit and receive data FIFOs.

The FIFOs may be loaded or emptied by the Central Processor Unit (CPU) using programmed I/O, or DMA transfers while receiving or transmitting.

Features:

- 3 protocols support: National's Microwire, TI's SSP, and Motorola's SPI
- Full-duplex or transmit-only or receive-only operation
- Programmable transfer order: MSB first or LSB first
- 128 entries deep x 17 bits wide transmit and receive data FIFOs
- Configurable normal transfer mode or Interval transfer mode
- Programmable clock phase and polarity for Motorola's SSI format
- Two slave select signal (SSI_CE_ / SSI_CE2_) supporting up to 2 slave devices
- Back-to-back character transmission/reception mode
- Loop back mode for testing

31.2 Pin Description

Table 31-1 Micro Printer Controller Pins Description

Name	I/O	Description
SSI_CLK	Output	Serial bit-rate clock
SSI_CE_	Output	First slave select enable
SSI_CE2_	Output	Second slave select enable
SSI_GPC	Output	General purpose control signal to external chip
SSI_DT	Output	Transmit data (serial data out)
SSI_DR	Input	Receive data (serial data in)

SSI_CLK is the bit-rate clock driven from the SSI to the peripheral. SSI_CLK is toggled only when data is actively being transmitted and received.

SSI_CE_ or SSI_CE2_ are the framing signal, indicating the beginning and the end of a serialized data word.

SSI_DT and SSI_DR are the Transmit and Receive serial data lines.

SSI_GPC is general-purpose control signal, synchronized with SSI_CLK, can be used for LCD control.

31.3 Register Description

The SSI has seven registers: one data, two control, one status, one bit-rate control, and two interval control registers. The table lists these registers.

Table 31-2 SSI Serial Port Registers

Name	RW	Reset Value	Address	Access Size
SSIDR	RW	0x??	0x10043000	32
SSICR0	RW	0x0000	0x10043004	16
SSICR1	RW	0x00087860	0x10043008	32
SSISR	RW	0x00000098	0x1004300C	32
SSIITR	RW	0x0000	0x10043010	16
SSIICR	RW	0x00	0x10043014	8
SSIGR	RW	0x0000	0x10043018	16

NOTES:

1 There two SSI controller. SSI0 whose base address is 0x100430xx and SSI1 whose base address is 0x100450xx.

31.3.1 SSI Data Register (SSIDR)

SSIDR **0x10043000**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved															GPC/D16	Transmit/Receive Data															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
31:17	Reserved		R
16	GPC/D16	This bit can be used as normal data bus bit 16 or GPC bit alternatively. When it is used as normal data bus bit, it's readable / writable; when SSI_GPC is used, it is GPC bit for SSI_GPC pin output and it's write-only.	RW
15:0	Transmit/ Receive Data	Data word to be written to/read from Transmit/Receive FIFO. When the transfer frame length is less than 17-bit, received data is automatically right justified in the receive-FIFO and the upper unused bits are filled with '0'. For transmission, the upper unused bits of the data written into SSIDR is ignored by the transmit logic. (**NOTE:** "upper unused bits" does not include the SSIDR.GPC bit. National microwire format includes format 1 and format2, when national	RW

		microwire format 2 is selected, Bit 16 of SSIDR is defined as read/write operation judge bit, if it is 0, bit 15~0 represent one read command; if it is 1, bit 15~0 represent one write command and following is the written data. So the maximum length of one command (is defined in MCOM) is 16, the maximum length of one written or read data (is defined in FLEN) can be 17. Transmit-FIFO only contain one read operation command once, or one write operation command and its data once, after transmit-FIFO is empty, next command can be filled in transmit-FIFO.	

31.3.2 SSI Control Register0 (SSICR0)

SSICR0 0x10043004

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SSIE	TIE	RIE	TEIE	REIE	LOOP	RFINE	RFINC	EACLRUN	FSEL	Reserved		TFMODE	TFLUSH	RFLUSH	DISREV
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15	SSIE	This bit is used to enable/disable SSI module. 0: disable; 1: enable. Clearing SSIE will not reset SSI FIFO, SSICR0, SSICR1, SSIGR, SSIITR and SSIICR automatically. Software should ensure the FIFOs/registers are properly configured and be flush/reset manually when necessary before enabling SSI.	RW
14	TIE	This bit enables/disables the transmit-FIFO half-empty interrupt TXI. 0: disable; 1: enable.	RW
13	RIE	This bit enables/disables the receive-FIFO half-full interrupt RXI. 0: disable; 1: enable.	RW
12	TEIE	This bit enables/disables the transmit-error interrupt TEI. 0: disable; 1: enable.	RW
11	REIE	This bit enables/disables the receive-error interrupt REI. 0: disable; 1: enable.	RW
10	LOOP	Used for test purpose. In loop mode, the output of SSI transmit shift register is connected to input of SSI receive shift register internally. The data received should be the same as the data transmitted. And do not output any valid signals on the pins. 0: normal SSI mode; 1: LOOP mode.	RW

<table>
<tr><td>9</td><td>RFINE</td><td>This bit enables/disables receive finish control function.
0: disable; 1: enable.
For SSICR1.FMAT = B’10 (National Microwire format 1 is selected), SSICR0.RFINE must be 0.</td><td rowspan="2">The receive finish condition list below:<table><tr><th>RFINE</th><th>RFINC</th><th>Receive Finish Condition</th></tr><tr><td>0</td><td>x</td><td>Same as transmit completion condition (transmit-fifo is empty and SSICR1.UNFIN = 0)</td></tr><tr><td>1</td><td>0</td><td>Receive continue</td></tr><tr><td>1</td><td>1</td><td>Receive finish</td></tr></table></td><td>RW</td></tr>
<tr><td>8</td><td>RFINC*</td><td>Receive finish control bit.
0: receive continue
1: receive finished</td><td>RW</td></tr>
<tr><td>7</td><td>EACLRUN</td><td colspan="2">0: don’t auto clear under flag, software clear under
1: software auto clear under flag when tfifo don’t empty</td><td>RW</td></tr>
<tr><td>6</td><td>FSEL</td><td colspan="2">This bit sets the frame signal to be used for slave select. The unselected frame signal always output invalid level.
0: SSI_CE_ is selected
1: SSI_CE2_ is selected</td><td>RW</td></tr>
<tr><td>5:4</td><td>Reserved</td><td colspan="2"></td><td>R</td></tr>
<tr><td>3</td><td>TFMODE</td><td colspan="2">0: new fifo empty mode
1: old fifo empty mode</td><td>RW</td></tr>
<tr><td>2</td><td>TFLUSH</td><td colspan="2">Flush the transmit FIFO when set to 1. Always return 0 when read.</td><td>RW</td></tr>
<tr><td>1</td><td>RFLUSH</td><td colspan="2">Flush the receive FIFO when set to 1. Always return 0 when read.</td><td>RW</td></tr>
<tr><td>0</td><td>DISREV</td><td colspan="2">This bit enables/disables receive function. 0: enable; 1: disable.</td><td>RW</td></tr>
</table>

NOTES:

1 ***:** When transmitting finished or for receive-only operation, transmit function can be disabled and this bit is used to control receiving completion, and the SSI will consume less power.
When the finish condition is set, the receiving will complete after present character is completely shifted in, then the SSI will stop the SSI_CLK and negate the SSI_CE_ / SSI_CE2_ if necessary. To make sure present transfer is completed, user must read and get SSISR.END = 1 (or SSISR.BUSY = 0).

31.3.3 SSI Control Register1 (SSICR1)

SSICR1 0x10043008

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FRMHL		TFVCK		TCKFI		LFST	ITFRM	UNFIN		FMAT		TTRG				MCOM				RTRG				FLEN						PHA	POL
RST	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	0	0	0	0	0

Bits	Name	Description	RW
31:30	FRMHL	Frame valid level select, FRMHL [1: 0] correspond to SSI_CE2_ and SSI_CE_ respectively. **FRMHL[1:0] / Description** 00 — SSI_CE_ is low level valid and SSI_CE2_ is low level valid — Initial value 01 — SSI_CE_ is high level valid and SSI_CE2_ is low level valid 10 — SSI_CE_ is low level valid and SSI_CE2_ is high level valid 11 — SSI_CE_ is high level valid and SSI_CE2_ is high level valid	RW
29:28	TFVCK	Time from frame valid to clock start, that provide programmable time delay from frame (SSI_CE_ /SSI_CE2_) assert edge to SSI_CLK leading edge. When TFVCK = B'00, the time is fixed half SSI_CLK or one SSI_CLK cycle according to SSICR1.POL and SSICR1.PHA configuration. For SSICR1.FMAT = B'01, SSICR1.TFVCK is ignored. **TFVCK[1:0] / Description** 00 — Ignore (default half or one SSI_CLK cycle delay time) — Initial value 01 — 1 more SSI_CLK cycle delay time is added 10 — 2 more SSI_CLK cycle delay time is added 11 — 3 more SSI_CLK cycle delay time is added	RW
27:26	TCKFI	Time from clock stop to frame invalid, provide programmable time delay from SSI_CLK last edge to frame (SSI_CE_ /SSI_CE2_) negate edge. When TCKFI = B'00, the time is fixed one SSI_CLK or half SSI_CLK cycle according to SSICR1.POL and SSICR1.PHA configuration. For SSICR1.FMAT = B'01, SSICR1.TFVCK is ignored. **TCKFI[1:0] / Description**	RW

		00	Ignore (default half or one SSI_CLK cycle delay time)	Initial value	
		01	1 more SSI_CLK cycle delay time is added		
		10	2 more SSI_CLK cycle delay time is added		
		11	3 more SSI_CLK cycle delay time is added		
25	LFST	Set to LSB first or MSB first when transfer. 0: MSB first; 1: LSB first.			RW
24	ITFRM	Frame during interval, selects if the Frame (SSI_CE_ /SSI_CE2_) signal is negated or not during interval time at Interval Mode (SSICR1.FMAT = B'00 and SSIITR.IVLTM ≠ H'0000). It's ignored at Normal Mode. 0: SSI_CE_ /SSI_CE2_ de-asserts during interval time at Interval Mode 1: SSI_CE_ /SSI_CE2_ keeps asserted during interval time at Interval Mode			RW
23	UNFIN	This bit controls whether the SSI finishes transmission or wait for data filling (underrun happen) after all data in transmit-FIFO are sent out during transfer. This bit must be cleared to 0 when SSICR1.FMAT = B'01. (TI's SSP format) 0: Transmit-FIFO empty means end of transmission 1: Transmission didn't finish when transmit-FIFO is empty, SSI underrun error would occur and SSI waits for data filling; SSI_CLK and SSI_CE_ /SSI_CE2_ keeps asserted, SSI_CLK stop at the current level **NOTE:** For transmit-FIFO empty before any transfer after SSI enabled, if SSICR1.UNFIN = 1 or SSICR0.RFINE = 0, SSI will wait till transmit-FIFO isn't empty then start to transfer and no underrun error will occur; if SSICR1.UNFIN = 0 and SSICR0.RFINE = 1, after transmit-FIFO become empty, SSI will start a receive-only transfer.			RW
22	Reserved				R
21:20	FMAT	These bits set the operating transfer format.			RW
		FMAT[1:0]	**Description**		
		00	Motorola's SPI format	Initial value	
		01	TI's SSP format		
		10	National Microwire 1 format		
		11	National Micowire 2 format		
19:16	TTRG	These bits define the transmit-FIFO half-empty threshold value, which when equal or less characters left in transmit-FIFO, the SSISR.TFHE will be set to '1'. 0000: less than or equal to 1 n: less than or equal to nx8			RW
15:12	MCOM	When SSICR1.FMAT = B'10 or B'11 (National Microwire format 1 or 2 is			RW

<table>
<tr><td></td><td></td><td>selected), this bit decides the length of command from 1-bit to 16-bit. The length of written or read data is defined in FLEN. For SSICR1.FMAT ≠ B’10 or B’11, this bit is ignored.
<table>
<tr><th>MCOM[1:0]</th><th>Description</th><th></th></tr>
<tr><td>0000</td><td>1-bit command selected</td><td></td></tr>
<tr><td>0001</td><td>2-bit command selected</td><td></td></tr>
<tr><td>0010</td><td>3-bit command selected</td><td></td></tr>
<tr><td>0011</td><td>4-bit command selected</td><td></td></tr>
<tr><td>0100</td><td>5-bit command selected</td><td></td></tr>
<tr><td>0101</td><td>6-bit command selected</td><td></td></tr>
<tr><td>0110</td><td>7-bit command selected</td><td></td></tr>
<tr><td>0111</td><td>8-bit command selected</td><td>Initial value</td></tr>
<tr><td>1000</td><td>9-bit command selected</td><td></td></tr>
<tr><td>1001</td><td>10-bit command selected</td><td></td></tr>
<tr><td>1010</td><td>11-bit command selected</td><td></td></tr>
<tr><td>1011</td><td>12-bit command selected</td><td></td></tr>
<tr><td>1100</td><td>13-bit command selected</td><td></td></tr>
<tr><td>1101</td><td>14-bit command selected</td><td></td></tr>
<tr><td>1110</td><td>15-bit command selected</td><td></td></tr>
<tr><td>1111</td><td>16-bit command selected</td><td></td></tr>
</table></td><td></td></tr>
<tr><td>11:8</td><td>RTRG
(SSI1)</td><td>These bits define the receive-FIFO half-full threshold value, which when equal or more characters received in receive-FIFO, the SSISR.RFHF will be set to ‘1’.
0000: more than or equal to 1
n: more than or equal to nx8</td><td>RW</td></tr>
<tr><td>9:8</td><td>RTRG
(SSI0)</td><td>These bits define the receive-FIFO half-full threshold value, which when equal or more characters received in receive-FIFO, the SSISR.RFHF will be set to “1”.
<table>
<tr><th>RTRG[1:0]</th><th>Description</th><th></th></tr>
<tr><td>00</td><td>more than or equal to 1</td><td>Initial value</td></tr>
<tr><td>01</td><td>more than or equal to 4</td><td></td></tr>
<tr><td>10</td><td>more than or equal to 8</td><td></td></tr>
<tr><td>11</td><td>more than or equal to 14</td><td></td></tr>
</table></td><td>RW</td></tr>
<tr><td>7:4</td><td>FLEN</td><td>These bits set the bit length of every character to be transmitted/received. The maximum data length can be configured is 17 bits. For data length longer than 17 bits (multiples of the SSICR1.FLEN configured length), the software should ensure properly processing. When SSI_GPC pin is used, the FLEN shouldn’t be configured as B’1111 (17-bit data). When TI SSP mode is selected (FMAT = 2’b01), 2-bit data length (FLEN = 4’b0000) isn’t supported.
<table>
<tr><th>MCOM[1:0]</th><th>Description</th><th></th></tr>
<tr><td>0000</td><td>2-bit data</td><td></td></tr>
</table></td><td>RW</td></tr>
</table>

		0001	3-bit data		
		0010	4-bit data		
		0011	5-bit data		
		0100	6-bit data		
		0101	7-bit data		
		0110	8-bit data	Initial value	
		0111	9-bit data		
		1000	10-bit data		
		1001	11-bit data		
		1010	12-bit data		
		1011	13-bit data		
		1100	14-bit data		
		1101	15-bit data		
		1110	16-bit data		
		1111	17-bit data		
3:2	Reserved				R
1	PHA	This bit sets the phase of the SSI_CLK from the beginning of a data frame for Motorola's SPI format (SSICR1.FMAT = B'00). 0: The leading edge of SSI_CLK is used to sample data from SSI_DR after the SSI_CE_ /SSI_CE2_ goes valid, it is initial value 1: The leading edge of SSI_CLK is used to drive data onto SSI_DT after the SSI_CE_ /SSI_CE2_ goes valid			RW
0	POL	This bit sets SSI_CLK's idle state polarity for Motorola's SPI format. (SSICR1.FMAT = B'00). 0: SSI_CLK keeps low level when idle, when SSI_CE_ /SSI_CE2_ goes valid the leading clock edge is a rising edge, it is initial value 1: SSI_CLK keeps high level when idle, when SSI_CE_ /SSI_CE2_ goes valid the leading clock edge is a falling edge			RW

31.3.4 SSI Status Register1 (SSISR)

SSISR **0x1004300C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									TFIFO-NUM								RFIFO-NUM								END	BUSY	TFF	RFE	TFHE	RFHF	UNDR	OVER
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0

Bits	Name	Description	RW
31:24	Reserved		R

23:16	TFIFO-NUM	These bits indicate the Characters Number in Transmit-FIFO.	R
15:8	RFIFO-NUM	These bits indicate the Characters Number in Transmit-FIFO.	R
7	END	This bit indicates transfer end status. It is the inverse of SSISR.BUSY when transfer is in process, but it'll keep cleared at interval time before transfer is completed. It'll be set when transfer finished.	R
6	BUSY	This bit indicates SSI's working status. 0: SSI is idle or at interval time 1: Transmission and/or reception is in process	R
5	TFF	This bit denotes transmit-FIFO is full or not. 0: Transmit-FIFO is not full 1: Transmit-FIFO is full	R
4	RFE	This bit denotes receive-FIFO is empty or not. 0: Receive-FIFO is not empty 1: Receive-FIFO is empty	R
3	TFHE	This bit denotes whether the characters number in transmit-FIFO being less or equal to SSICR1.TTRG. 0: The data in transmit-FIFO is more than the condition set by SSICR1.TTRG 1: The data in transmit-FIFO meets the condition set by SSICR1.TTRG, If SSICR0.TIE = 1, it will generate SSI TXI interrupt	R
2	RFHF	This bit denotes whether the characters number in receive-FIFO being more or equal to the number set by SSICR1.RTRG. 0: The data in receive-FIFO is less than the condition set by SSICR1.RTRG 1: The data in receive-FIFO meets the condition set by SSICR1.RTRG, If SSICR0.RIE = 1, it will generate SSI RXI interrupt	R
1	UNDR	Transmit-FIFO underrun status. When underrun happens, SSI set this bit and keeps the current status of SSI_CLK and SSI_CE_/SSI_CE2_, waiting for transmit-FIFO filling. 0: Underrun has not occurred 1: Underrun has occurred, when SSICR0.TEIE is set, it will generate SSI TEI interrupt. Write '0' to clear this bit, writing '1' has no effect	RW
0	OVER	Receive-FIFO overrun status, new received data will lose. 0: Overrun has not occurred 1: Overrun has occurred; When SSICR0.REIE is set, it will generate SSI REI interrupt. Write '0' to clear this bit, writing '1' has no effect	RW

31.3.5 SSI Interval Time Control Register (SSIITR)

SSIITR 0x10043010

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CNTCLK	IVLTM														
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15	CNTCLK	Counting clock source select. 0: Use SSI bit clock (SSI_CLK) as the interval counter clock source 1: Use 32K clock as the interval counter clock source	RW
14:0	IVLTM	Interval time set, set the cycle number of counting clock source for desired interval time. When SSIITR.IVLTM = 0x0000, normal mode is selected, and SSIITR.CNTCLK and SSIICR are ignored. When SSIITR.IVLTM ≠ 0x0000, interval mode is selected. The interval time is calculated as follows: *Interval time ≈ [CNTCLK clock period] * [Value of IVLTM]* The actual interval time is as follow: When SSIITR.CNTCLK = 0: *Interval time = [CNTCLK clock period] * [Value of IVLTM] + 3 * device_clock period* When SSIITR.CNTCLK = 1: *Interval time ≥ [CNTCLK clock period] * [Value of IVLTM + 1] + 1 * device_clock period;* *Interval time ≤ [CNTCLK clock period] * [Value of IVLTM + 2] + 2 * device_clock period*	RW

31.3.6 SSI Interval Character-per-frame Control Register (SSIICR)

SSIICR 0x10043014

Bit	7	6	5	4	3	2	1	0
						ICC		
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:3	Reserved		R
2:0	ICC	Sets the fixed number of characters to be transmitted / received each time during SSI_CLK changing (and SSI_CE_ / SSI_CE2_ asserting) in interval mode for SSICR1.FMAT = B'00 (Motorola's SPI format is selected). SSIICR is ignored for SSICR1.FMAT ≠ B'00. The desired transfer number of characters-per-frame is (SSIICR set value + 1).	RW

31.3.7 SSI Clock Generator Register (SSIGR)

SSIGR 0x10043018

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
									CGV							
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15:8	Reserved		R
7:0	CGV	Sets the frequency of serial bit clock (SSI_CLK). The serial bit clock (SSI_CLK) is generated by dividing device-clock as follows: F_{SSI_CLK} = [Frequency of device clock] / (2 * (CGV + 1)) Device clock is generated in CPM module. The value in SSIGR can be set from 0 to 255, and initialized to 0x0000 on power-on reset.	RW

31.4 Functional Description

Serial data is transferred between the processor and external peripheral through FIFO buffers in the SSI. Data transfers to system memory are handled by either the CPU (using programmed I/O) or by DMA. Operation is full duplex - separate buffers and serial data paths permit simultaneous transfers to and from the external peripheral.

Programmed I/O transmits and receives data directly between the CPU and the transmit/receive FIFO's. The DMA controller transfers data during transmit and receive operations between memory and the FIFO's.

Transmit data is written by the CPU or DMA to the SSI's transmit FIFO. The SSI then takes the data from the FIFO, serializes it, and transmits it via the SSI_DT signal to the peripheral. Data from the peripheral is received via the SSI_DR signal, converted to parallel words and is stored in the Receive FIFO. Read operations automatically target the receive FIFO, while write operations write data to the transmit FIFO. Both the transmit and receive FIFO buffers are 128 entries deep by 17 bits wide. As the received data fills the receive FIFO, a programmable threshold triggers an interrupt to the Interrupt Controller. If enabled, an interrupt service routine responds by identifying the source of the interrupt and then performs one or several read operations from the inbound (receive) FIFO buffer.

31.5 Data Formats

Four signals are used to transfer data between the processor and external peripheral. The SSI supports three formats: Motorola SPI, Texas Instruments SSP, and National Microwire. Although they have the same basic structure the three formats have significant differences, as described below:

1 SSI_CE_/SSI_CE2_ varies for each protocol as follows:
   - For SPI and Microwire formats, SSI_CE_/SSI_CE2_ functions as a chip select to enable the external device (target of the transfer), and is held active-low during the data transfer.
   - For SSP format, this signal is pulsed high for one serial bit-clock period at the start of each frame.

2 SSI_CLK varies for each protocol as follows:
   - For Microwire, both transmit and receive data sources switch data on the falling edge of SSI_CLK, and sample incoming data on the rising edge.
   - For SSP, transmit and receive data sources switch data on the rising edge of SSI_CLK, and sample incoming data on the falling edge.
   - For SPI, the user has the choice of which edge of SSI_CLK to use for switching outgoing data, and for sampling incoming data. In addition, the user can move the phase of SSI_CLK, shifting its active state one-half period earlier or later at the start and end of a frame.

While SSP and SPI are full-duplex protocols, Microwire uses a half-duplex master-slave messaging protocol. At the start of a frame, a 1 or 2-byte control message is transmitted from the controller to the peripheral. The peripheral does not send any data. The peripheral interprets the message and, if it is a READ request, responds with requested data, one clock after the last bit of the requesting message.

The serial clock (SSI_CLK) only toggles during an active frame. At other times it is held in an inactive or idle state, as defined by its specified protocol.

31.5.1 Motorola's SPI Format Details

31.5.1.1 General Single Transfer Formats

The figures below show the timing of general single transfer format.

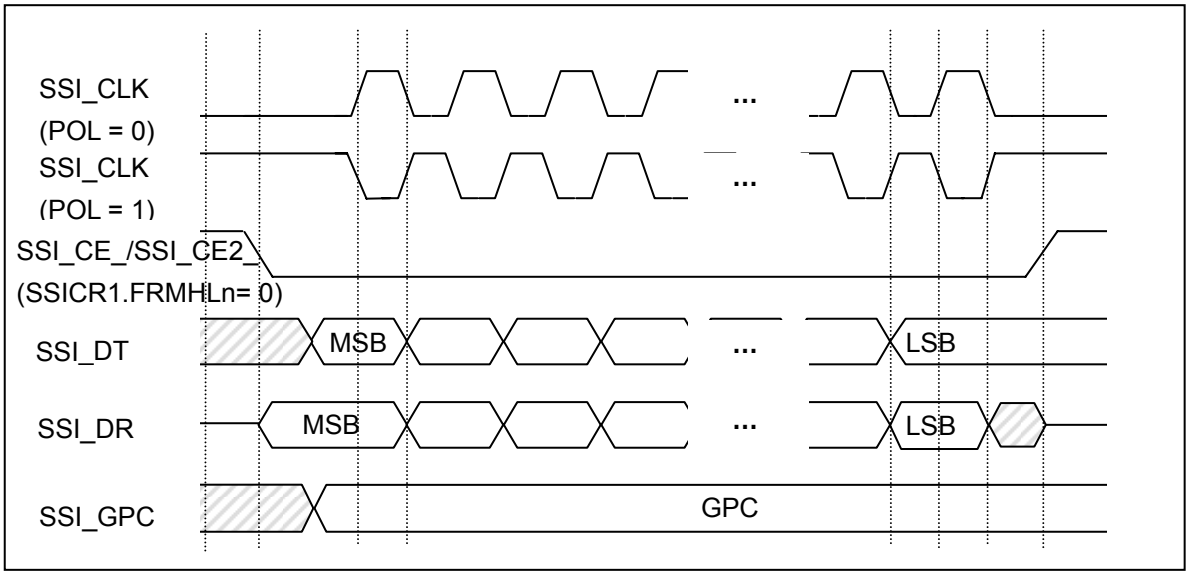


Figure 31-1 SPI Single Character Transfer Format (PHA = 0)

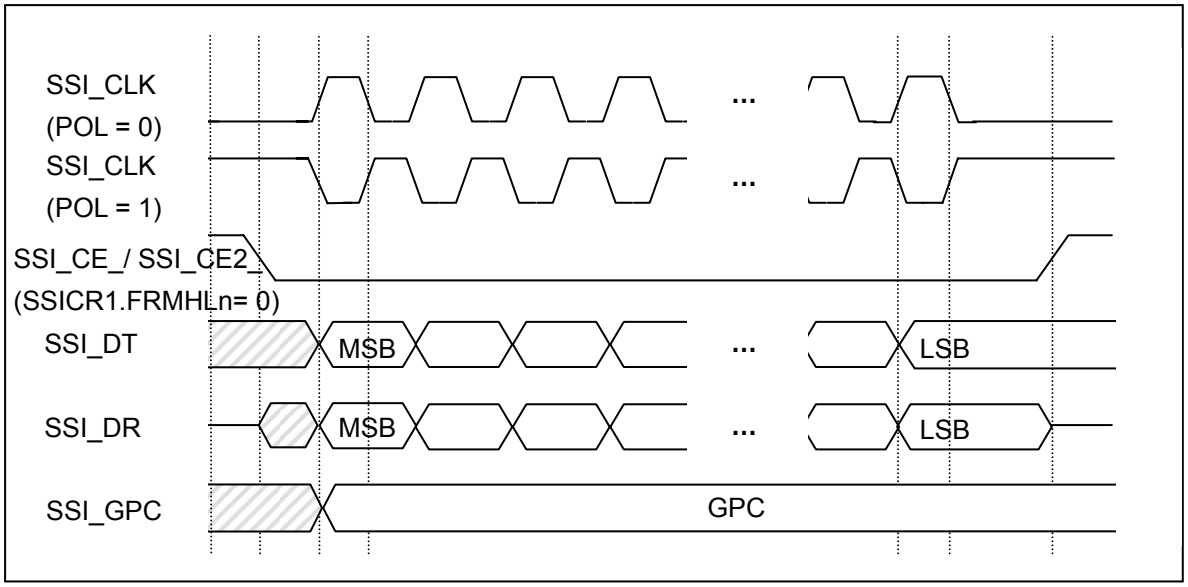


Figure 31-2 SPI Single Character Transfer Format (PHA = 1)

For SSICR1.PHA = 0, when SSICR1.TFVCK = B'00, hardware ensures the first clock edge appears one SSI_CLK period after SSI_CE_ / SSI_CE2_ goes valid; when SSICR1.TCKFI = B'00, hardware ensures the SSI_CE_ / SSI_CE2_ negated half SSI_CLK period after last clock change edge; when SSICR1.TFVCK ≠ B'00 or SSICR1.TCKFI ≠ B'00, 1/2/3 more clock cycles are inserted.

For SSICR1.PHA = 1, when SSICR1.TFVCK = B'00, hardware ensures the first clock edge appears half SSI_CLK period after SSI_CE_ / SSI_CE2_ goes valid; when SSICR1.TCKFI = B'00, hardware ensures the SSI_CE_ / SSI_CE2_ negated one SSI_CLK period after last clock change edge; when SSICR1.TFVCK ≠ B'00 or SSICR1.TCKFI ≠ B'00, 1/2/3 more clock cycles are inserted.

Data is sampled from SSI_DR at every rising edge (when PHA = 0, POL = 0 or PHA = 1, POL = 1) or at every falling edge (when PHA = 0, POL = 1 or PHA = 1, POL = 0). According to SPI protocol, input data on SSI_DR should be stable at every sample clock edge.

Drive data onto SSI_DT at every rising edge (when PHA = 0, POL = 1 or PHA = 1, POL = 0) or at every falling edge (when PHA = 0, POL = 0 or PHA = 1, POL = 1).

31.5.1.2 Back-to-Back Transfer Formats

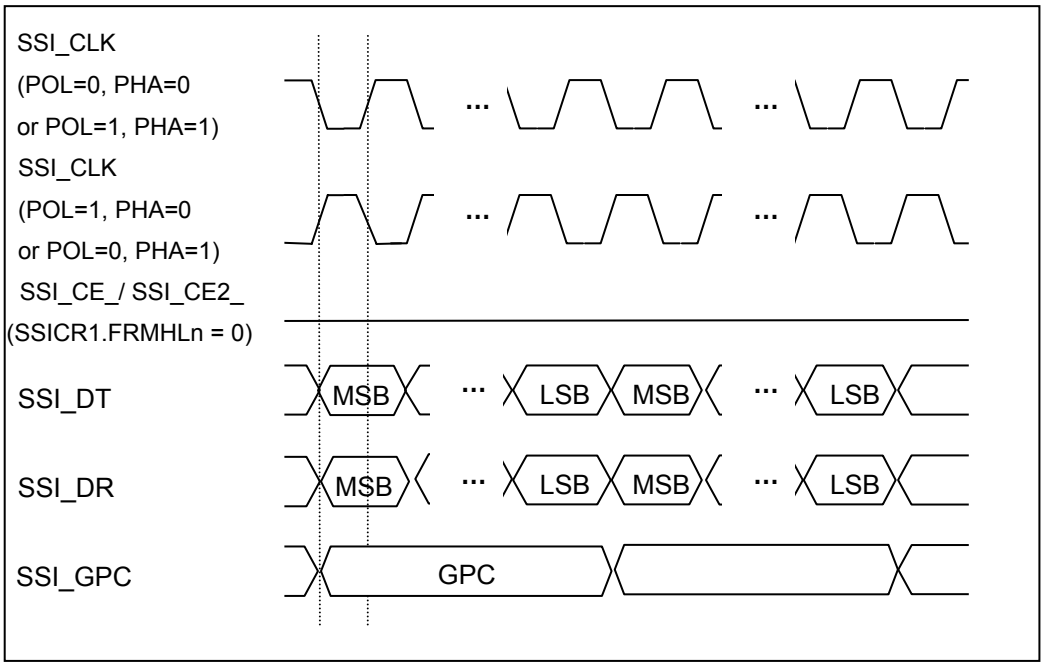


Figure 31-3 SPI Back-to-Back Transfer Format

For Motorola's SPI format transfers those continuous characters are exchanged during SSI_CE_ / SSI_CE2_ being valid, the timing is illustrated in the figure (SSICR1.LFST = 0).

Back-to-back transfer is performed as transmit-only/full-duplex operation when transmit-FIFO is not empty before the completion of the last character's transfer or performed as receive-only operation.

31.5.1.3 Frame Interval Mode Transfer Format

When in interval mode (SSIITR. IVLTM ≠ '0'), SSI always wait for an interval time (SSIITR.IVLTM), transfer fixed number of characters (SSIICR), then repeats the operation.

When SSICR0.RFINE = 1, if transmit-FIFO is still empty after the interval time, receive-only transfer will occur.

During interval-wait time, SSI stops SSI_CLK, and when SSICR1.ITFRM = 0 it negates the SSI_CE_ / SSI_CE2_, when SSICR1.ITFRM = 1 it keeps asserting the SSI_CE_ / SSI_CE2_.

For transfers finished with transmit-FIFO empty, if the SSI transmit-FIFO is empty before fixed number of characters being loaded to transfer (SSICR1.UNFIN must be 1), then the SSI will set SSISR.UNDR = 1; if enabled, it'll send out a SSI underrun interrupt. At the same time, SSI will hold the SSI_CE_ / SSI_CE2_ and SSI_CLK signals at current status and wait for the transmit-FIFO filling. The SSI will continue transfer after transmit-FIFO being filled. The SSI always stops after completion of fixed number of characters' transfer (SSICR1.UNFIN must be 0) with transmit-FIFO empty.

For transfers finished by SSICR0.RFINC being valid set, the SSI will stop after finished current character transfer and needn't wait for a whole completion of fixed number of characters' transfer.

Two Interval transfer mode are illustrated in the following figures. In these timing diagram, SSICR1.PHA = 0, SSICR1.POL = 0 and SSIICR = 0.

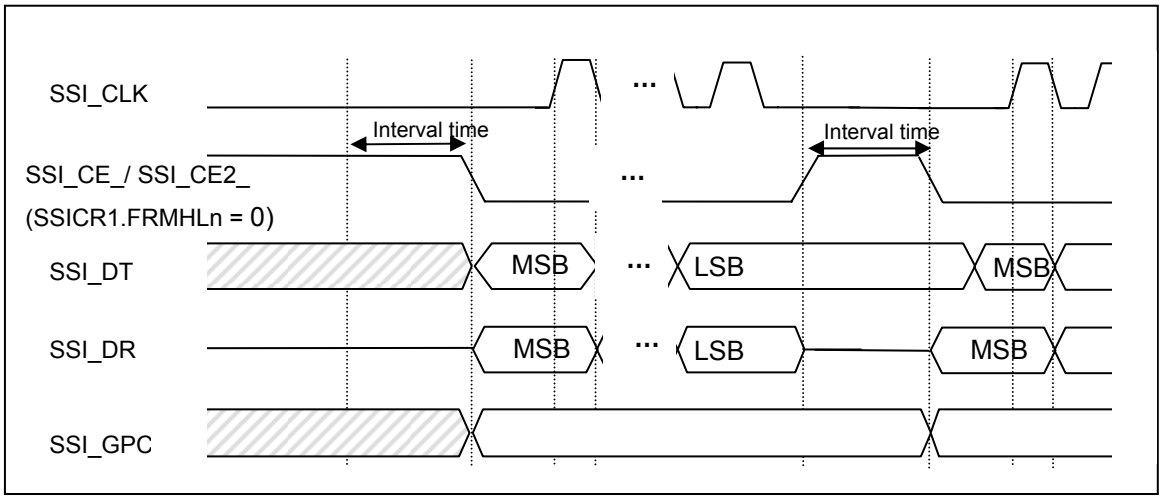


Figure 31-4 SPI Frame Interval Mode Transfer Format (ITFRM = 0, LFST = 0)

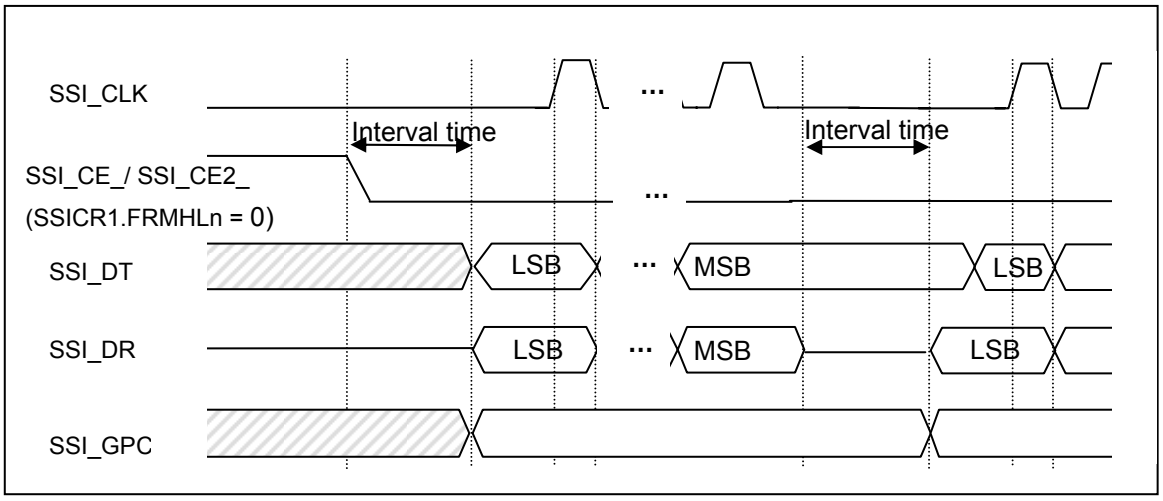


Figure 31-5 SPI Frame Interval Mode Transfer Format (ITFRM = 1, LFST = 1)

31.5.2 TI's SSP Format Details

In this format, each transfer begins with SSI_CE_ pulsed high for one SSI_CLK period. Then both master and slave drive data at SSI_CLK's rising edge and sample data at the falling edge. Data are transferred with MSB first or LSB first. At the end of the transfer, SSI_DT retains the value of the last bit sent through the next idle period.

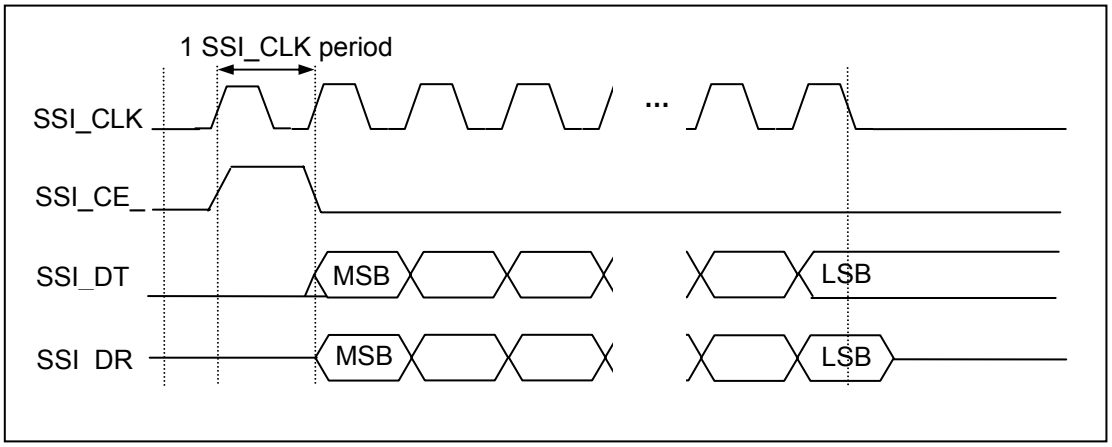


Figure 31-6 TI's SSP Single Transfer Format

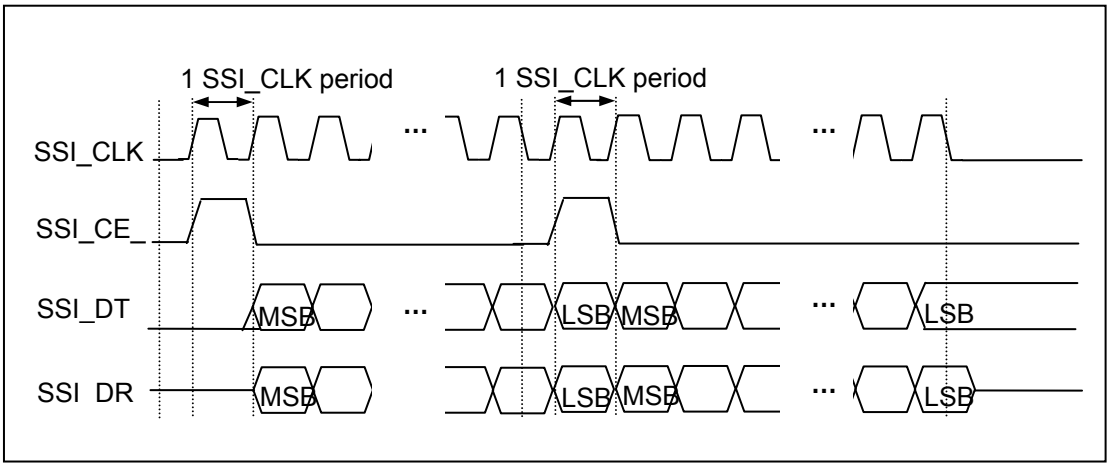


Figure 31-7 TI's SSP Back-to-back Transfer Format

31.5.3 National Microwire Format Details

It supports format 1 and format 2. If format 1 is selected, both master and slave drive data at SSI_CLK falling edge and sample data at the rising edge. If format 2 is selected, master drive and sample data at SSI_CLK falling edge, slave drive and sample data at SSI_CLK rising edge. SSI_CLK goes high midway through the command's most significant bit (or LSB) and continues to toggle at the bit rate. One bit clock (format 1) or half one bit clock (format 2) period after the last command bit, the external slave must return the serial data requested, with most significant bit first (or LSB first) on SSI_DR. SSI_CE_ / SSI_CE2 de-asserts high half clock (SSI_CLK) period (and 1/2/3 additional clock periods) later. Format 1 support back-to-back transfer, the start and end of back-to-back transfers are similar to those of a single transfer. However, SSI_CE_ / SSI_CE2 remains asserted throughout the transfer. The end of a character data on SSI_DR is immediately followed by the start of the next command byte on SSI_DT.

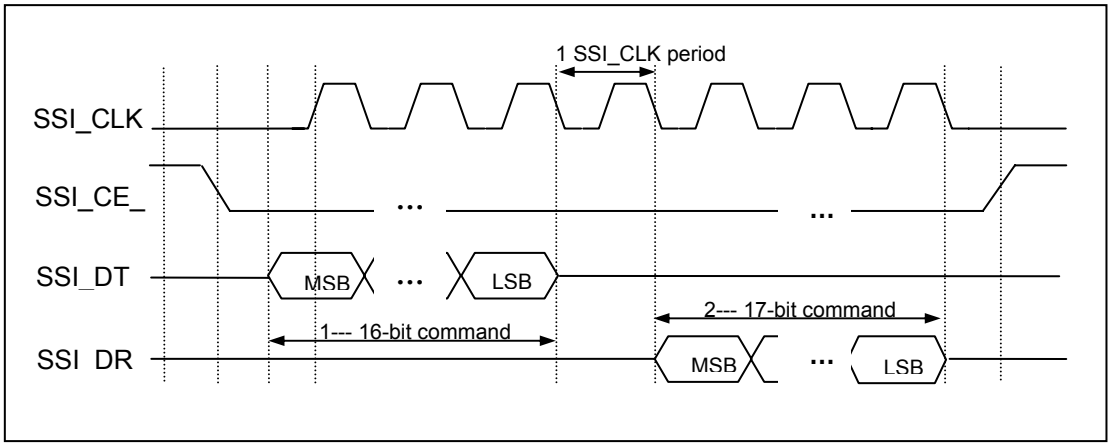


Figure 31-8 National Microwire Format 1 Single Transfer

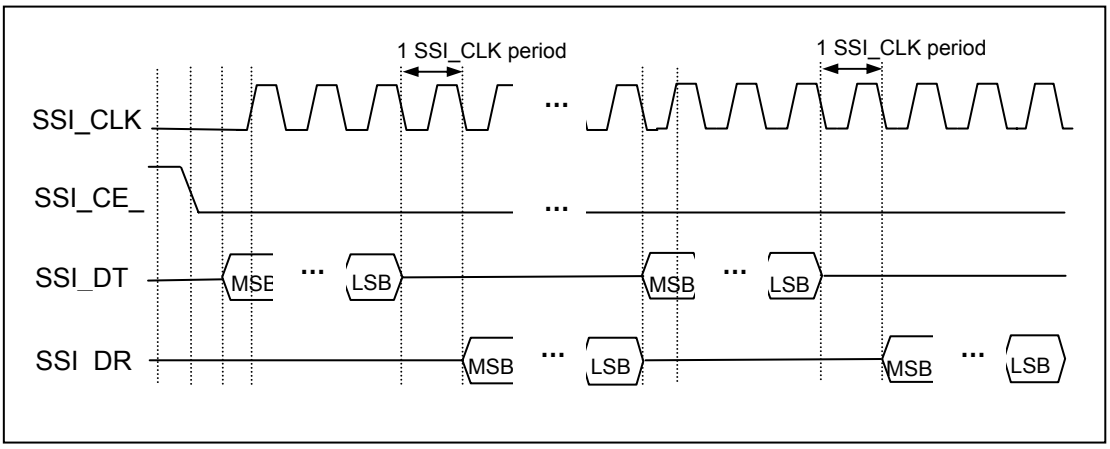


Figure 31-9 National Microwire Format 1 Back-to-back Transfer

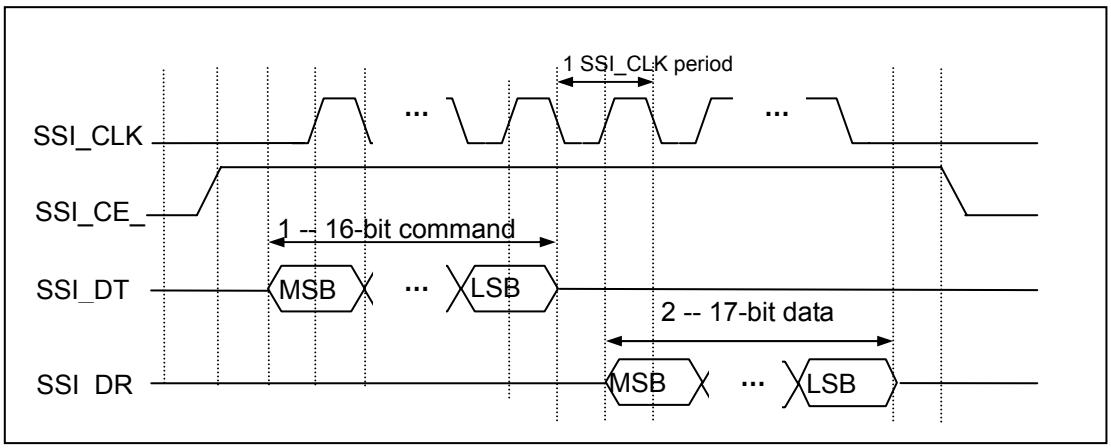


Figure 31-10 National Microwire Format 2 Read Timing

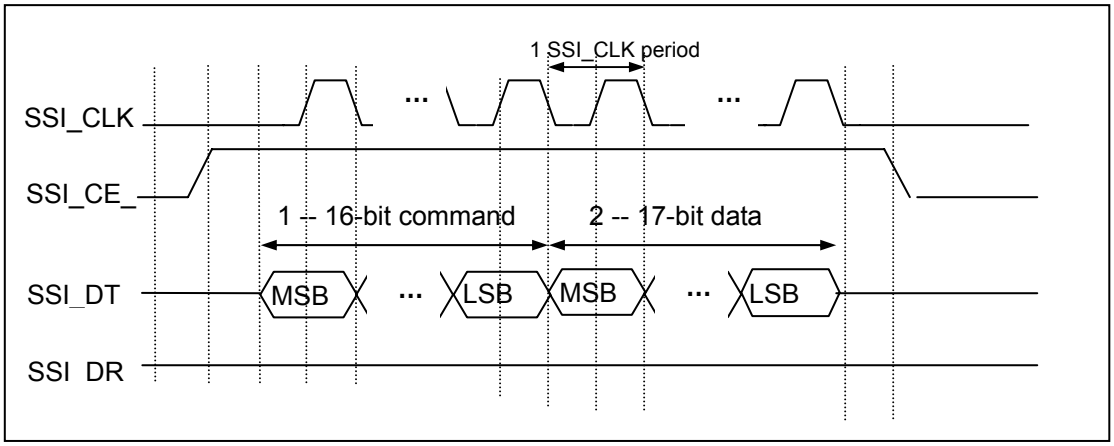


Figure 31-11 National Microwire Format 2 Write Timing

31.6 Interrupt Operation

In SSI, there are TXI, RXI, TEI and REI total 4 interrupts, all these interrupts are combined together to make one SSI interrupt, which can be masked by writing '1' into corresponding mask bit in INTC interrupt mask register (IMR).

Table 31-3 SSI Interrupts

Operation	Condition	Flag Bit	Mask Bit	Interrupt	DMAC Activation
Transmit	T-FIFO is half-empty or less	SSISR.TFHE	SSICR0.TIE	**TXI**	Possible
	Transmit underrun error	SSISR.UNDR	SSICR0.TEIE	**TEI**	Impossible
Receive	R-FIFO is half-full or more	SSISR.RFHF	SSICR0.RIE	**RXI**	Possible
	Receive overrun error	SSISR.OVER	SSICR0.REIE	**REI**	Impossible

Either SSISR.TFHE or SSISR.RFHF can activate DMA transferring when corresponding individual interrupt mask bit in SSICR0 is cleared (masked) and DMA is enabled and configured.

32 One-Wire Bus Interface

32.1 Overview

The OWI has the following features:

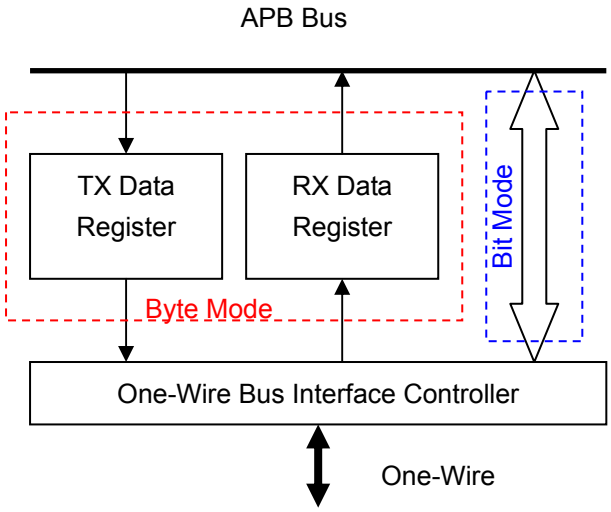
- Support 1-wire bus protocol
- Support Overdrive speed mode and Regular speed mode
- Data is transferred with the LSB first
- Support bit operate mode and byte operate mode
- OWI is the only master on the bus

32.2 Pin Description

Table 32-1 One-Wire Controller Pins Description

Name	I/O	Description
OWDAT	Input/Output	One-Wire Data signal

32.3 Structure



32.4 Register Description

Table 32-2 OWI Registers Description

Name	Description	RW	Reset Value	Address	Access Size
OWCFG	Configure Register	RW	0x00	0x10072000	8
OWCTL	Control Register	RW	0x00	0x10072004	8
OWSTS	Status Register	RW	0x00	0x10072008	8
OWDAT	Data Register	RW	0x00	0x1007200C	8
OWDIV	Clock Divide Register	RW	0x00	0x10072010	8

32.4.1 One-Wire Configure Register (OWCFG)

OWCFG **0x10072000**

Bit	7	6	5	4	3	2	1	0
	MODE	RDDATA	WRDATA	RDST	WR1/RD	WR0	RST	ENA
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7	MODE	OWI mode select. 0: Regular speed mode 1: Overdrive speed mode	RW
6	RDDATA	Receive a byte from one-wire bus. This bit is cleared when receive data is complete. The value of received data is stored in OWDAT, and is valid after RDDATA is self-cleared. 0:Do nothing/Receive data is completed 1:Receive data from one-wire bus and stored in OWDAT	RW
5	WRDATA	Transmit the data in OWDAT. This bit is cleared when the transmission of data is complete. 0: Do nothing/Transmission of data completed 1: Transmit the data in OWDAT	RW
4	RDST	Read status. This bit is valid after the WR1/RD bit is self-cleared. 0: 0 was sampled during a read 1: 1 was sampled during a read	R
3	WR1/RD	Write 1/ Read. This bit is cleared when the write of the bit is complete. The valve of one wire can be read, since the Write 1 and Read timing are identical. The value of the read bit is stored in RDST, and is valid after WR1/RD is self-cleared. 0: Do nothing/Write 1 sequence completed	RW

		1: Generate Write 1 sequence on line	
2	WR0	Write 0 on line. This bit is cleared after the presence is determined. 0: Do nothing/Write 0 sequence completed 1: Generate Write 0 sequence on line	RW
1	RST	Reset presence pulse. This bit is cleared after the presence is determined. 0: Do nothing. Reset pulse completed 1: Generate reset pulse and sample slaves presence pulse	RW
0	ENA	Enable of OWI operation. 1: Write 1 to this bit to enable the OWI operation 0: Write 0 to this bit to disable the OWI operation	RW

NOTES:

1 To make the OWI operate normally, only one of the RST, RDDATA, WRDATA, WR1/RD and WR0 is equal to 1.

32.4.2 One-Wire Control Register (OWCTL)

OWCTL **0x10072004**

Bit	7	6	5	4	3	2	1	0
	Reserved					EBYTE	EBIT	ERST
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:3	Reserved	These bits always read as 0. Write data to these bits are ignored.	R
2	EBYTE	Enable byte write / read interrupt.	RW
1	EBIT	Enable bit write / read interrupt.	RW
0	ERST	Enable reset sequence finished interrupt.	RW

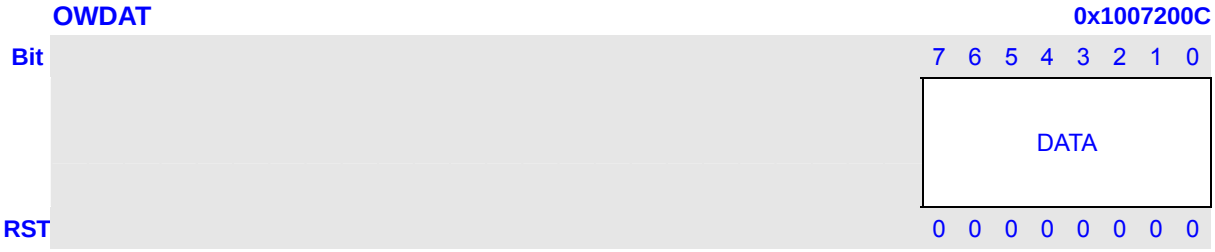
32.4.3 One-Wire Status Register (OWSTS)

OWSTS **0x10072008**

Bit	7	6	5	4	3	2	1	0
	PST	Reserved				BYTE_RDY	BIT_RDY	PST_RDY
RST	0	0	0	0	0	0	0	0

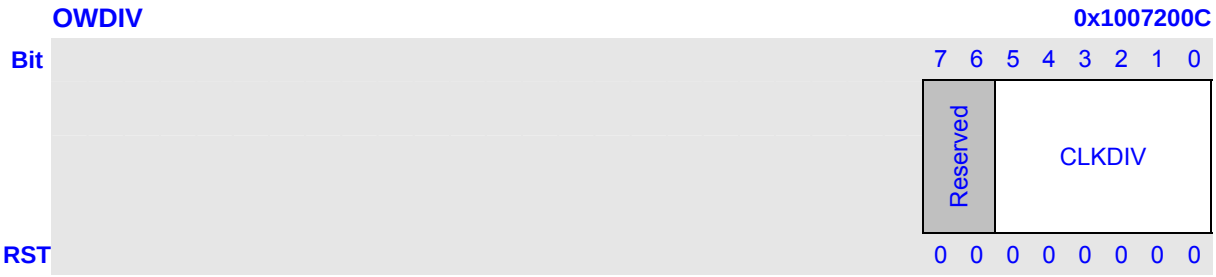
Bits	Name	Description	RW
7	PST	Whether the 1-wire bus has device or not. This bit is valid after the RST bit is self-cleared. 1:The 1-wire bus has device on it 0:The 1-wire bus has no device on it	RW
6:3	Reserved	These bits always read as 0. Write data to these bits are ignored.	R
2	BYTE_RDY	Have received or transmitted a data.	RW
1	BIT_RDY	Have received or transmitted a bit.	RW
0	PST_RDY	Have finished a reset pulse.	RW

32.4.4 One-Wire Data Register (OWDAT)



Bits	Name	Description	RW
7:0	DATA	Store the received data and is valid after RDDATA is self-cleared. Prepare the transmission data for transmitting.	RW

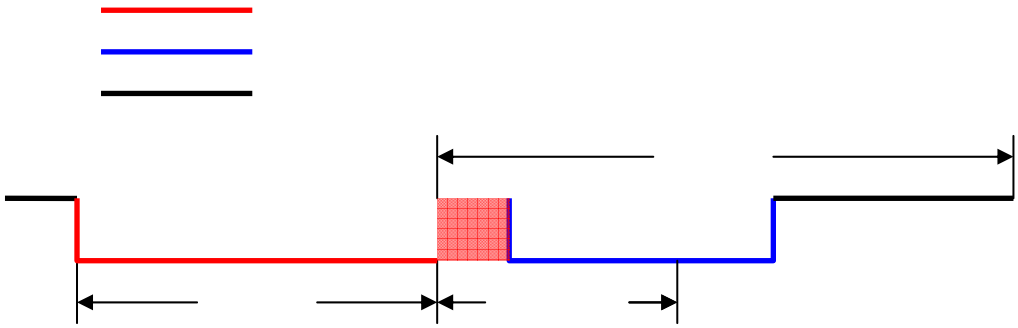
32.4.5 One-Wire Clock Divide Register (OWDIV)



Bits	Name	Description	RW
7:6	Reserved	These bits always read as 0. Write data to these bits are ignored.	
5:0	CLKDIV	Controls the divider used to create the DEV_CLK based upon the CPM_OWI_SYSCLK. When the OWI work in the regular speed mode: 1 MHz = DEV_CLK = CPM_OWI_SYSCLK / (CLKDIV + 1). When the OWI work in the overdrive speed mode: 4 MHz = DEV_CLK = CPM_OWI_SYSCLK / (CLKDIV + 1).	RW

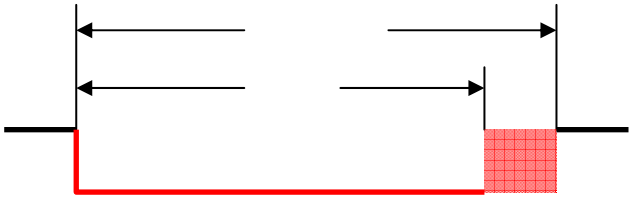
32.5 One-Wire Bus Protocol

32.5.1 Reset Timing and ACK Timing



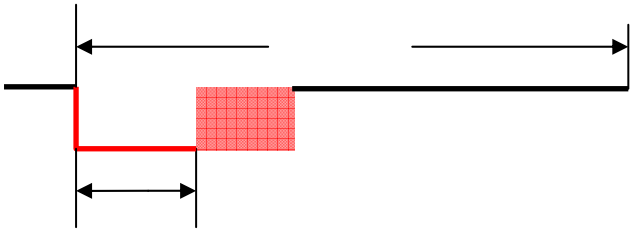
	RSTH	RSTL	RSTS
Regular Speed mode	512us	512us	68us
Overdrive Speed mode	64us	64us	8us

32.5.2 Write 0 Timing



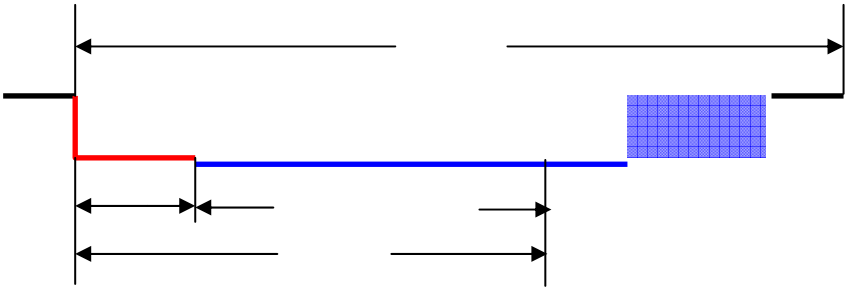
	SLOT	LOW0
Regular Speed mode	128us	100us
Overdrive Speed mode	16us	12us

32.5.3 Write 1 Timing



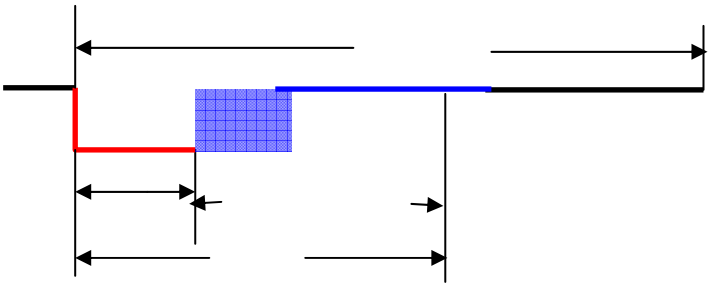
	SLOT	LOW1
Regular Speed mode	128us	5us
Overdrive Speed mode	16us	1.25us

32.5.4 Read0 Timing



	SLOT	LOWR	RDS
Regular Speed mode	128us	5us	13us
Overdrive Speed mode	16us	1.25us	1.75us

32.5.5 Read1 Timing



	SLOT	LOWR	RDS
Regular Speed mode	128us	5us	13us
Overdrive Speed mode	16us	1.25us	1.75us

32.6 One-Wire Operation Guide

- **Interrupt operation guide:**
  1 Write the frequency divider to I2CGR.
  2 Set OWSTS to clear the flag.
  3 Set OWCTL to enable the operation interrupt.
  4 Select OWI mode (Regular speed mode or Overdrive speed mode).
  5 Set RST, RDDATA, WRDATA, WR1/RD or WR0 to choose one kind of operation.
  6 Set ENA to enable OWI.
  7 Wait till the interrupt happened, and the operation is finished.
  8 If you want to disable OWI when OWI is working, you should set ENA to 0 and till ENA is really set to 0, then you can do next operation.

- **CPU operation guide:**
  1 Write the frequency divider to I2CGR.
  2 Select OWI mode (Regular speed mode or Overdrive speed mode).
  3 Set RST, RDDATA, WRDATA, WR1/RD or WR0 to choose one kind of operation.
  4 Set ENA to enable OWI.
  5 Wait the ENA is cleared, and the operation is finished.
  6 If you want to disable OWI when OWI is working, you should set ENA to 0 and till ENA is really set to 0, then you can do next operation.

33 USB Host Controller

33.1 Overview

This chapter describes the Universal Serial Bus host controller (UHC) implemented in the XBurst processor.

The Universal Serial Bus (USB) supports serial data exchanges between a host computer and a variety of simultaneously accessible peripherals. The attached peripherals share USB bandwidth through a host-scheduled, token-based protocol. Peripherals can be attached, configured, used, and detached, while the host and other peripherals continue operation.

Familiarity with the *Universal Serial Bus Specification*, Revision 1.1 and the OHCI specification are necessary to fully understand the material contained in this section.

Features:

- USB Rev. 1.1 compatible
- Supports both low-speed and full-speed USB devices
- Open Host Controller Interface (OHCI) Rev 1.0 compatible
- Root hub supports two data ports

33.2 Pin Description

Table 33-1 UHC Pins Description

Name	Type	Description
DPLS0	Inout	Data Positive to Port 0
DPLS1	Inout	Data Positive to Port 1
DMNS0	Inout	Data Minus to Port 0
DMNS1	Inout	Data Minus to Port 1

33.3 Register Description

The Host Controller (HC) contains a set of on-chip operational registers which are mapped into a noncacheable portion of the system addressable space. These registers are used by the Host Controller Driver (HCD). According to the function of these registers, they are divided into four partitions, specifically for Control and Status, Memory Pointer, Frame Counter and Root Hub. All of the registers should be read and written as words.

Register Name	Description	RW	Reset Value	Address	Access Size
HcRevision	Control and Status group	R	0x00000010	0x13430000	32
HcControl		RW	0x00000000	0x13430004	32
HcCommandStatus		RW	0x00000000	0x13430008	32
HcInterruptStatus		RW	0x00000000	0x1343000C	32
HcInterruptEnable		RW	0x00000000	0x13430010	32
HcInterruptDisable		RW	0x00000000	0x13430014	32
HcHCCA	Memory pointer group	RW	0x00000000	0x13430018	32
HcPeriodCurrentED		R	0x00000000	0x1343001C	32
HcControlHeadED		RW	0x00000000	0x13430020	32
HcControlCurrentED		RW	0x00000000	0x13430024	32
HcBulkHeadED		RW	0x00000000	0x13430028	32
HcBulkCurrentED		RW	0x00000000	0x1343002C	32
HcDoneHead		R	0x00000000	0x13430030	32
HcFmInterval	Frame counter group	RW	0x00002EDF	0x13430034	32
HcFmRemaining		R	0x00000000	0x13430038	32
HcFmNumber		R	0x00000000	0x1343003C	32
HcPeriodicStart		RW	0x00000000	0x13430040	32
HcLSThreshold		RW	0x00000628	0x13430044	32
HcRhDescriptorA	Root hub group	R/W	0x02000902	0x13430048	32
HcRhDescriptorB		RW	0x00060000	0x1343004C	32
HcRhStatus		RW	0x00000000	0x13430050	32
HcRhPortStatus 1		RW	0x00000100	0x13430054	32
HcRhPortStatus 2		RW	0x00000100	0x13430058	32

NOTES:

1 *Open HCI – Open Host Controller Specification for USB* for details of the each register.

33.4 Introduction

The Host Controller is the device which is located between the USB bus and the Host Controller Driver in the OpenHCI architecture. The Host Controller is charged with processing all of the Data Type lists built by the Host Controller Driver. Additionally, the USB Root Hub is attached to the Host Controller.

The main functions as following:

- **USB States**: the Host Controller Operation with respect to the possible USB Bus states.
- **Frame Management**: all aspects of managing the 1-ms USB Frame.
- **List Processing**: the main function of the Host Controller. the detailed processing of the HCD-built Data Type lists.
- **Interrupt Processing**: the interrupt events tracked by the Host Controller and how the Host Controller provides interrupts for those events.
- **Root Hub**: the Root Hub support.

34 OTG Controller

34.1 Overview

This chapter describes the USB On-The-Go (OTG) implemented in the JZ4760 processor.

The Universal Serial Bus (USB) supports serial data exchanges between a host computer and a variety of simultaneously accessible portable peripherals. Many of these portable devices would benefit a lot from being able to communicate to each other over the USB interface. And OTG make this possible. An OTG device can plays the role of both host and device.

Familiarity with the *Universal Serial Bus Specification*, Revision 1.1 and OTG supplement are necessary to fully understand the material contained in this section.

Features:

- Complies with the USB 2.0 standard for high-speed (480 Mbps) functions and with the *On-The-Go* supplement to the USB 2.0 specification
- Operates either as the function controller of a high- /full-speed USB peripheral or as the host/peripheral in point-to-point or multi-point communications with other USB functions
- Supports Session Request Protocol (SRP) and Host Negotiation Protocol (HNP)
- UTMI+ Level 3 Transceiver Interface
- Soft connect/disconnect
- 5 DMA channels
- Supports control, interrupt, ISO and bulk transfer

34.2 Pin Description

Table 34-1 OTG Pins Description

Name	Type	Description
DP	Inout	Data Positive
DM	Inout	Data Minus
ID	Inout	Identification
drvvbus	Out	Charge pump enable

34.3 Register Description

The OTG Controller (OTGC) contains a set of on-chip operational registers which are mapped into a noncacheable portion of the system addressable space. These registers are used by the OTG Controller Driver.

Table 34-2 OTG Registers Description

Name	Description	RW		Reset Value	Address	Access Size
		CPU	USB			
FAddr	Function address register	RW	R	8'h00	0x13440000	8
Power	Power management register	RW	RW	8'h20	0x13440001	8
IntrTx	Interrupt register for Endpoint 0 plus TX Endpoints 1 to 15	R	S	16'h0000	0x13440002	16
IntrRx	Interrupt register for Rx Endpoints 1 to 15	R	S	16'h0000	0x13440004	16
IntrTxE	Interrupt enable register for IntrTx	RW	R	16'hffff	0x13440006	16
IntrRxE	Interrupt enable register for IntrRx	RW	R	16'hfffe	0x13440008	16
IntrUSB	Interrupt register for common USB interrupts	R	S	8'h00	0x1344000a	8
IntrUSBE	Interrupt enable register for IntrUSB	RW	R	8'h06	0x1344000b	8
Frame	Frame Number	R	W	16'h0000	0x1344000c	16
Index	Index register for selecting the endpoint status and control registers	RW	R	4'h0	0x1344000e	4
TestMode	Enables the USB 2.0 test modes	RW	R	8'h00	0x1344000f	8
TxMaxP	Maximum packet size for peripheral TX endpoint (Index register set to select Endpoints 1 – 15 only)	RW	R	16'h0000	0x13440010	16
CSR0L/H	Control Status register for Endpoint 0 (Index register set to select Endpoint 0)	RW	RW	16'h0000	0x13440012	16
TxCSRL/H	Control Status register for peripheral TX endpoint (Index register set to select Endpoints 1 – 15)					
RxMaxP	Maximum packet size for	RW	R	16'h00	0x13440014	16

	peripheral Rx endpoint (Index register set to select Endpoints 1 – 15 only)			00		
RxCSRL/H	Control Status register for peripheral Rx endpoint (Index register set to select Endpoints 1 – 15 only)	RW	RW	16'h0000	0x13440016	16
Count0	Number of received bytes in Endpoint 0 FIFO (Index register set to select Endpoint 0)	R	W	7'h00	0x13440018	7
RxCount	Number of bytes to be read from peripheral Rx endpoint FIFO (Index register set to select Endpoints 1 – 15)					
Type0	Defines the speed of Endpoint 0 (Index register set to select Endpoint 0)	RW	R	8'h0	0x1344001a	8
TxType	Sets the transaction protocol, speed and peripheral endpoint number for the host TX endpoint (Index register set to select Endpoints 1 – 15)					
NakLimit0	Sets the NAK response timeout on Endpoint 0 (Index register set to select Endpoint 0)	RW	R	8'h00	0x1344001b	8
TxInterval	Sets the polling interval for Interrupt/ISOC transactions or the NAK response timeout on Bulk transactions for host TX endpoint (Index register set to select Endpoints 1 – 15 only)					
RxType	Sets the transaction protocol, speed and peripheral endpoint number for the host Rx endpoint (Index register set to select Endpoints 1 – 15 only)	RW	R	8'h00	0x1344001c	8
RxInterval	Sets the polling interval for Interrupt/ISOC transactions or the NAK response timeout on Bulk transactions for host Rx endpoint (Index register set to select Endpoints 1 – 15 only)	RW	R	8'h00	0x1344001d	8

ConfigData	Returns details of core configuration (Index register set to select Endpoint 0)	R	R	8'h??	0x1344001f	8
FifoSize	Returns the configured size of the selected Rx FIFO and TX FIFOs (Endpoints 1 – 15 only)					
FIFOx	FIFOs for Endpoints 0 – 15	RW	-	32'h??	0x13440020	32
DevCtl	OTG device control register	RW	RW	8'h80	0x13440060	8
MISC	Miscellaneous Register	RW	RW	8'h00	0x13440061	8
VControl	UTMI+ PHY Vendor registers	R	R	4'h?	0x13440068	4
Vstatus	UTMI+ PHY Vendor registers	R	R	8'h??	0x1344006a	8
HWVers	Hardware Version Number Register	R	R	16'h????	0x1344006c	16
EPInfo	Information about numbers of TX and Rx endpoints	R	R	8'h??	0x13440078	8
RAMInfo	Information about the width of the RAM and the number of DMA channels	R	R	8'h??	0x13440079	8
LinkInfo	Information about delays to be applied	RW	R	8'h5c	0x1344007a	8
VPLen	Duration of the VBus pulsing charge	RW	R	8'h3c	0x1344007b	8
HS_EOF1	Time buffer available on High-Speed transactions	RW	R	8'h80	0x1344007c	8
FS_EOF1	Time buffer available on Full-Speed transactions	RW	R	8'h77	0x1344007d	8
LS_EOF1	Time buffer available on Low-Speed transactions	RW	R	8'h72	0x1344007e	8
SoftRst	Soft reset	RW	R	8'h00	0x1344007f	8
TxFuncAddr	Transmit Endpoint *n* Function Address (Host Mode only)	RW	R	7'h00	0x13440080 +8*n	7
TxHubAddr	Transmit Endpoint *n* Hub Address (Host Mode only)	RW	R	8'h00	0x13440082 +8*n	8
TxHubPort	Transmit Endpoint *n* Hub Port (Host Mode only)	RW	R	7'h00	0x13440083 +8*n	7
RxFuncAddr	Receive Endpoint *n* Function Address (Host Mode only)	RW	R	7'h00	0x13440084 +8*n	7
RxHubAddr	Receive Endpoint *n* Hub Address (Host Mode only)	RW	R	8'h00	0x13440086 +8*n	8
RxHubPort	Receive Endpoint *n* Hub Port (Host Mode only)	RW	R	7'h00	0x13440087 +8*n	7
DMA_INTR	DMA Interupt register	RW	S	8'h00	0x13440200	8

DMA_CNTL	DMA Control Register for DMA channel n (channel 1 thru 8)	RW	R	11'h00	0x13440204 +(n-1)*0x10	11
DMA_ADDR	DMA Address Register for DMA channel n (channel 1 thru 8)	RW	RW	32'h00000000	0x13440208 +(n-1)*0x10	32
DMA_COUNT	DMA Count Register for DMA channel n (channel 1 thru 8)	RW	RW	32'h00000000	0x1344020c +(n-1)*0x10	32
RqPktCount	Number of requested packets for Receive Endpoint *n* (Endpoints 1 – 15 only)	RW	RW	16'h0000	0x13440300 +4*n	16
RxDPktBufDis	Double Packet Buffer Disable register for Rx Endpoints 1 to 15	RW	R	16'h0000	0x13440340	16
TxDPktBufDis	Double Packet Buffer Disable register for TX Endpoints 1 to 15	RW	R	16'h0000	0x13440342	16
C_T_UCH	This register sets the Chirp Timeout Timer	RW	-	16'h?	0x13440344	16
C_T_HSRTN	This register sets the delay from the end of High Speed resume signaling to enable UTM normal operating mode	RW	-	16'h?	0x13440346	16
C_T_HSBT	HS Timeout Adder	RW	R	4'h0	0x13440348	4

NOTES:

1 In the following bit descriptions:

'r' means that the bit is read only 'rw' means that the bit can be both read and written.

'set' means that the bit can only be written to set it 'r/set' means that the bit can be read or set but it can't be cleared.

'clear' means that the bit can only be written to clear it 'r/clear' means that the bit can be read or cleared but it can't be set.

'self-clearing' means the bit will be cleared automatically when the associated action has been executed.

34.4 Common registers

34.4.1 FAddr

FAddr is an 8-bit register that should be written with the 7-bit address of the peripheral part of the transaction.

When the MUSBHDRC is being used in Peripheral mode (DevCtl.D2=0), this register should be written with the address received through a SET_ADDRESS command, which will then be used for decoding the function address in subsequent token packets.

NOTES:

1 Peripheral Mode Only!!

FAddr **0x13440000**

Bit	7	6	5	4	3	2	1	0
	Unused	FAddr						
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
7	-	Unused. Always 0.	R	-
6:0	FAddr	The function address.	RW	R

34.4.2 Power

Power is an 8-bit register that is used for controlling Suspend and Resume signaling, and some basic operational aspects of the MUSBHDRC.

Power **0x13440001**

Bit	7	6	5	4	3	2	1	0
	ISO Update	Soft Conn	HS Enab	HS Mode	Reset	Resume	Suspend Mode	Enable SuspendM
RST	0	0	1	0	0	0	0	0

Bits	Name	Description	CPU	USB
7	ISO Update	When set by the CPU, the MUSBHDRC will wait for an SOF token from the time TxPktRdy is set before sending the packet. If an IN token is received before an SOF token, then a zero length data packet will be sent. **NOTE:** *Only valid in Peripheral Mode. Also, this bit only affects endpoints performing Isochronous transfers.*	P: RW H: -	P: R H: -
6	Soft Conn	If Soft Connect/Disconnect feature is enabled, then the USB D+/D- lines are enabled when this bit is set by the CPU and tri-stated when this bit is cleared by the CPU. **NOTE:** *Only valid in Peripheral Mode.*	P: RW H: -	P: R H: -
5	HS Enab	When set by the CPU, the MUSBHDRC will negotiate for High-speed mode when the device is reset by the hub. If not set, the device will only operate in Full-speed mode.	P: RW H: RW	P: R H: R
4	HS Mode	When set, this read-only bit indicates High-speed mode successfully negotiated during USB reset. In Peripheral Mode, becomes valid when USB reset completes (as indicated by USB reset interrupt). In Host Mode, becomes valid when Reset bit is cleared. Remains valid for the duration of the session. **NOTE:** Allowance is made for Tiny-J signaling in determining the transfer speed to select.	P: R H: R	P: RW H: RW
3	Reset	This bit is set when Reset signaling is present on the bus. **NOTE:** This bit is Read/Write from the CPU inHost Mode but Read-Only in Peripheral Mode.	P: R H: RW	P: RW H: RW
2	Resume	Set by the CPU to generate Resume signaling when the function is in Suspend mode. The CPU should clear this bit after 10 ms (a maximum of 15 ms) to end Resume signaling. In Host mode, this bit is also automatically set when Resume signaling from the target is detected while the MUSBHDRC is suspended.	P: RW H: RW	P: R H: R
1	Suspend Mode	In Host mode, this bit is set by the CPU to enter Suspend mode. In Peripheral mode, this bit is set on entry into Suspend mode. It is cleared when the CPU reads the interrupt register, or sets the Resume bit above.	P: R H: set	P: RW H: clr
0	Enable SuspendM	Set by the CPU to enable the SUSPENDM output.	P: RW H: RW	P: R H: R

34.4.3 IntrTx

IntrTx is a 16-bit read-only register that indicates which interrupts are currently active for Endpoint 0 and the Tx Endpoints 1–15.

NOTES:

1 Bits relating to endpoints that have not been configured will always return 0. Note also that all active interrupts are cleared when this register is read.

IntrTx **0x13440002**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EP15Tx	EP14Tx	EP13Tx	EP12Tx	EP11Tx	EP10Tx	EP9Tx	EP8Tx	EP7Tx	EP6Tx	EP5Tx	EP4Tx	EP3Tx	EP2Tx	EP1Tx	EP0
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15	EP15Tx	Tx Endpoint 15 interrupt.	R	Set
14	EP14Tx	Tx Endpoint 14 interrupt.	R	Set
13	EP13Tx	Tx Endpoint 13 interrupt.	R	Set
12	EP12Tx	Tx Endpoint 12 interrupt.	R	Set
11	EP11Tx	Tx Endpoint 11 interrupt.	R	Set
10	EP10Tx	Tx Endpoint 10 interrupt.	R	Set
9	EP9Tx	Tx Endpoint 9 interrupt.	R	Set
8	EP8Tx	Tx Endpoint 8 interrupt.	R	Set
7	EP7Tx	Tx Endpoint 7 interrupt.	R	Set
6	EP6Tx	Tx Endpoint 6 interrupt.	R	Set
5	EP5Tx	Tx Endpoint 5 interrupt.	R	Set
4	EP4Tx	Tx Endpoint 4 interrupt.	R	Set
3	EP3Tx	Tx Endpoint 3 interrupt.	R	Set
2	EP2Tx	Tx Endpoint 2 interrupt.	R	Set
1	EP1Tx	Tx Endpoint 1 interrupt.	R	Set
0	EP0	Tx Endpoint 0 interrupt.	R	Set

34.4.4 IntrRx

IntrRx is a 16-bit read-only register that indicates which of the interrupts for Rx Endpoints 1 – 15 are currently active.

NOTES:

1 Bits relating to endpoints that have not been configured will always return 0. Note also that all active interrupts are cleared when this register is read.

INTRRX 0x13040004

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EP15Rx	EP14Rx	EP13Rx	EP12Rx	EP11Rx	EP10Rx	EP9Rx	EP8Rx	EP7Rx	EP6Rx	EP5Rx	EP4Rx	EP3Rx	EP2Rx	EP1Rx	-
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15	EP15Rx	Rx Endpoint 15 interrupt.	R	Set
14	EP14Rx	Rx Endpoint 14 interrupt.	R	Set
13	EP13Rx	Rx Endpoint 13 interrupt.	R	Set
12	EP12Rx	Rx Endpoint 12 interrupt.	R	Set
11	EP11Rx	Rx Endpoint 11 interrupt.	R	Set
10	EP10Rx	Rx Endpoint 10 interrupt.	R	Set
9	EP9Rx	Rx Endpoint 9 interrupt.	R	Set
8	EP8Rx	Rx Endpoint 8 interrupt.	R	Set
7	EP7Rx	Rx Endpoint 7 interrupt.	R	Set
6	EP6Rx	Rx Endpoint 6 interrupt.	R	Set
5	EP5Rx	Rx Endpoint 5 interrupt.	R	Set
4	EP4Rx	Rx Endpoint 4 interrupt.	R	Set
3	EP3Rx	Rx Endpoint 3 interrupt.	R	Set
2	EP2Rx	Rx Endpoint 2 interrupt.	R	Set
1	EP1Rx	Rx Endpoint 1 interrupt.	R	Set
0	-	Unused, always returns 0.	R	R

34.4.5 IntrTxE

IntrTxE is a 16-bit register that provides interrupt enable bits for the interrupts in IntrTx. On reset, the bits corresponding to Endpoint 0 and the Tx endpoints included in the design are set to 1, while the remaining bits are set to 0.

NOTES:

1 Bits relating to endpoints that have not been configured will always return 0.

IntrTxE 0x13440006

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EP15Tx	EP14Tx	EP13Tx	EP12Tx	EP11Tx	EP10Tx	EP9Tx	EP8Tx	EP7Tx	EP6Tx	EP5Tx	EP4Tx	EP3Tx	EP2Tx	EP1Tx	EP0
RST	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	Name	Description	CPU	USB
15	EP15Tx	Tx Endpoint 15 interrupt enable.	RW	R
14	EP14Tx	Tx Endpoint 14 interrupt enable.	RW	R
13	EP13Tx	Tx Endpoint 13 interrupt enable.	RW	R
12	EP12Tx	Tx Endpoint 12 interrupt enable.	RW	R
11	EP11Tx	Tx Endpoint 11 interrupt enable.	RW	R
10	EP10Tx	Tx Endpoint 10 interrupt enable.	RW	R
9	EP9Tx	Tx Endpoint 9 interrupt enable.	RW	R
8	EP8Tx	Tx Endpoint 8 interrupt enable.	RW	R
7	EP7Tx	Tx Endpoint 7 interrupt enable.	RW	R
6	EP6Tx	Tx Endpoint 6 interrupt enable.	RW	R
5	EP5Tx	Tx Endpoint 5 interrupt enable.	RW	R
4	EP4Tx	Tx Endpoint 4 interrupt enable.	RW	R
3	EP3Tx	Tx Endpoint 3 interrupt enable.	RW	R
2	EP2Tx	Tx Endpoint 2 interrupt enable.	RW	R
1	EP1Tx	Tx Endpoint 1 interrupt enable.	RW	R
0	EP0	Tx Endpoint 0 interrupt enable.	RW	R

34.4.6 IntrRxE

IntrRxE is a 16-bit register that provides interrupt enable bits for the interrupts in IntrRx. On reset, the bits corresponding to the Rx endpoints included in the design are set to 1, while the remaining bits are set to 0.

NOTES:

1 Bits relating to endpoints that have not been configured will always return 0.

IntrRxE **0x13440008**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EP15Rx	EP14Rx	EP13Rx	EP12Rx	EP11Rx	EP10Rx	EP9Rx	EP8Rx	EP7Rx	EP6Rx	EP5Rx	EP4Rx	EP3Rx	EP2Rx	EP1Rx	-
RST	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Bits	Name	Description	CPU	USB
15	EP15Rx	Rx Endpoint 15 interrupt enable.	RW	R
14	EP14Rx	Rx Endpoint 14 interrupt enable.	RW	R
13	EP13Rx	Rx Endpoint 13 interrupt enable.	RW	R
12	EP12Rx	Rx Endpoint 12 interrupt enable.	RW	R
11	EP11Rx	Rx Endpoint 11 interrupt enable.	RW	R

10	EP10Rx	Rx Endpoint 10 interrupt enable.	RW	R
9	EP9Rx	Rx Endpoint 9 interrupt enable.	RW	R
8	EP8Rx	Rx Endpoint 8 interrupt enable.	RW	R
7	EP7Rx	Rx Endpoint 7 interrupt enable.	RW	R
6	EP6Rx	Rx Endpoint 6 interrupt enable.	RW	R
5	EP5Rx	Rx Endpoint 5 interrupt enable.	RW	R
4	EP4Rx	Rx Endpoint 4 interrupt enable.	RW	R
3	EP3Rx	Rx Endpoint 3 interrupt enable.	RW	R
2	EP2Rx	Rx Endpoint 2 interrupt enable.	RW	R
1	EP1Rx	Rx Endpoint 1 interrupt enable.	RW	R
0	-	Unused, always returns 0.	R	R

34.4.7 IntrUSB

IntrUSB is an 8-bit read-only register that indicates which USB interrupts are currently active. All active interrupts will be cleared when this register is read.

IntrUSB **0x1344000a**

Bit	7	6	5	4	3	2	1	0
	VBus Error	Sess Req	Discon	Conn	SOF	Reset/Babble	Resume	Suspend
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
7	Vbus Error	Set when VBus drops below the VBus Valid threshold during a session. *Only valid when MUSBHDRC is 'A' device.*	R	Set
6	Sess Req	Set when Session Request signaling has been detected. *Only valid when MUSBHDRC is 'A' device.*	R	Set
5	Discon	Set in Host mode when a device disconnect is detected. Set in Peripheral mode when a session ends. *Valid at all transaction speeds.*	R	Set
4	Conn	Set when a device connection is detected. *Only valid in Host mode. Valid at all transaction speeds.*	R	Set
3	SOF	Set when a new frame starts.	R	Set
2	Reset	Set in Peripheral mode when Reset signaling is detected on the D2 e bus.	R	Set
	Babble	Set in Host mode when babble is detected.		

1	Resume	Set when Resume signaling is detected on the bus while the MUSBHDRC is in Suspend mode.	R	Set
0	Suspend	Set when Suspend signaling is detected on the bus. *Only valid in Peripheral mode.*	R	Set

34.4.8 IntrUSBE

IntrUSBE is an 8-bit register that provides interrupt enable bits for each of the interrupts in IntrUSB.

IntrUSBE **0x1344000b**

Bit	7	6	5	4	3	2	1	0
	VBus Error	Sess Req	Discon	Conn	SOF	Reset/Babble	Resume	Suspend
RST	0	0	0	0	0	1	1	0

Bits	Name	Description	CPU	USB
7	Vbus Error	Interrupt enable.	RW	R
6	Sess Req	Interrupt enable.	RW	R
5	Discon	Interrupt enable.	RW	R
4	Conn	Interrupt enable.	RW	R
3	SOF	Interrupt enable.	RW	R
2	Reset Babble	Interrupt enable.	RW	R
1	Resume	Interrupt enable.	RW	R
0	Suspend	Interrupt enable.	RW	R

34.4.9 Frame

Frame is a 16-bit read-only register that holds the last received frame number.

Frame **0x1344000C**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0					Frame Number										
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15:11	-	Always 0.	R	W
10:0	Frame Number	Frame number.	R	W

34.4.10 Index

Each Tx endpoint and each Rx endpoint have their own set of control/status registers located between 100h – 1FFh. In addition one set of Tx control/status and one set of Rx control/status registers appear at 10h – 19h. Index is a 4-bit register that determines which endpoint control/status registers are accessed.

Before accessing an endpoint's control/status registers at 10h – 19h, the endpoint number should be written to the Index register to ensure that the correct control/status registers appear in the memory map.

Index **0x1344000e**

Bit	3	2	1	0
	Selected Endpoint			
RST	0	0	0	0

Bits	Name	Description	CPU	USB
3:0	Selected Endpoint	Selected endpoint.	RW	R

34.4.11 TestMode

Testmode is an 8-bit register that is primarily used to put the MUSBHDRC into one of the four test modes for High-speed operation described in the USB 2.0 specification – in response to a SET FEATURE: TESTMODE command. It is not used in normal operation.

NOTES:

1 Only one of Bits D0 – D6 should be set at any time.

TestMode **0x1344000f**

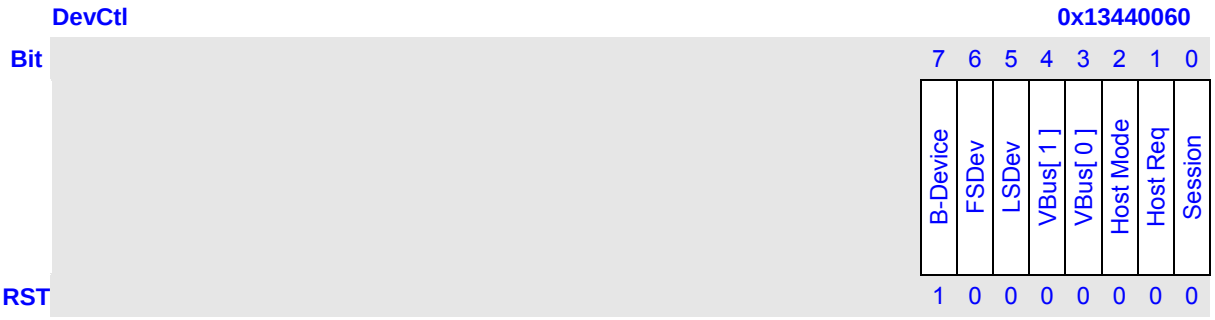
Bit	7	6	5	4	3	2	1	0
	Force_Host	FIFO_Access(self-clr)	Force_FS	Force_HS	Test_Packet	Test_K	Test_J	Test_SE0_NAK
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
7	Force_Host	The CPU sets this bit to instruct the core to enter Host mode when the Session bit is set, regardless of whether it is connected to any peripheral. The state of the CID input, HostDisconnect and LineState signals are ignored. The core will then remain in Host mode until the Session bit is cleared, even if a device is disconnected, and if the Force_Host bit remains set, will re-enter Host mode the next time the Session bit is set. While in this mode, the status of the HOSTDISCON signal from the PHY may be read from bit 7 of the DevCtl register. The operating speed is determined from the Force_HS and Force_FS bits as follows: <table><tr><th>Force_HS</th><th>Force_FS</th><th>Operating Speed</th></tr><tr><td>0</td><td>0</td><td>Low Speed</td></tr><tr><td>0</td><td>1</td><td>Full Speed</td></tr><tr><td>1</td><td>0</td><td>High Speed</td></tr><tr><td>1</td><td>1</td><td>Undefined</td></tr></table>	RW	R
6	FIFO_Access	The CPU sets this bit to transfer the packet in the Endpoint 0 Tx FIFO to the Endpoint 0 Rx FIFO. It is cleared automatically.	Set	R
5	Force_FS	The CPU sets this bit either in conjunction with bit 7 above or to force the MUSBHDRC into Full-speed mode when it receives a USB reset.	RW	R
4	Force_HS	The CPU sets this bit either in conjunction with bit 7 above or to force the MUSBHDRC into High-speed mode when it receives a USB reset.	RW	R
3	Test_Packet	*(High-speed mode)* The CPU sets this bit to enter the Test_Packet test mode. In this mode, the MUSBHDRC repetitively transmits on the bus a 53-byte test packet, the form of which is defined in the *Universal Serial Bus Specification* Revision 2.0, Section 7.1.20 (and in the MUSBHDRC Programmer's Guide). **NOTE:** The test packet has a fixed format and must be loaded into the Endpoint 0 FIFO before the test mode is entered.	RW	R

2	Test_K	*(High-speed mode)* The CPU sets this bit to enter the Test_K test mode. In this mode, the MUSBHDRC transmits a continuous K on the bus.	RW	R
1	Test_J	*(High-speed mode)* The CPU sets this bit to enter the Test_J test mode. In this mode, the MUSBHDRC transmits a continuous J on the bus.	RW	R
0	Test_SE0_NAK	*(High-speed mode)* The CPU sets this bit to enter the Test_SE0_NAK test mode. In this mode, the MUSBHDRC remains in High-speed mode but responds to any valid IN token with a NAK.	RW	R

34.4.12 DevCtl

DevCtl is an 8-bit register that is used to select whether the MUSBHDRC is operating in Peripheral mode or in Host mode, and for controlling and monitoring the USB VBus line.



Bits	Name	Description	CPU	USB
7	B-Device	This Read-only bit indicates whether the MUSBHDRC is operating as the 'A' device or the 'B' device. 0 => 'A' device; 1 => 'B' device. *Only valid while a session is in progress.* **NOTE:** If the core is in Force_Host mode (i.e. a session has been started with Testmode.D7 = 1), this bit will indicate the state of the HOSTDISCON input signal from the PHY.	R	RW

<table>
<tr><td>6</td><td>FSDev</td><td>This Read-only bit is set when a full-speed or high-speed device has been detected being connected to the port. (High-speed devices are distinguished from full-speed by checking for high-speed chirps when the device is reset.) Only valid in Host mode.</td><td>R</td><td>RW</td></tr>
<tr><td>5</td><td>LSDev</td><td>This Read-only bit is set when a low-speed device has been detected being connected to the port. Only valid in Host mode.</td><td>R</td><td>RW</td></tr>
<tr><td>4:3</td><td>VBus</td><td>These Read-only bits encode the current VBus level as follows:
<table><tr><th>D4</th><th>D3</th><th>Meaning</th></tr><tr><td>0</td><td>0</td><td>Below SessionEnd</td></tr><tr><td>0</td><td>1</td><td>Above SessionEnd, below AValid</td></tr><tr><td>1</td><td>0</td><td>Above AValid, below VBusValid</td></tr><tr><td>1</td><td>1</td><td>Above VBusValid</td></tr></table></td><td>R</td><td>RW</td></tr>
<tr><td>2</td><td>Host Mode</td><td>This Read-only bit is set when the MUSBHDRC is acting as a Host.</td><td>R</td><td>RW</td></tr>
<tr><td>1</td><td>Host Req</td><td>When set, the MUSBHDRC will initiate the Host Negotiation when Suspend mode is entered. It is cleared when Host Negotiation is completed. See Section 15. (‘B’ device only)</td><td>RW</td><td>R/clr</td></tr>
<tr><td>0</td><td>Session</td><td>When operating as an ‘A’ device, this bit is set or cleared by the CPU to start or end a session. When operating as a ‘B’ device, this bit is set/cleared by the MUSBHDRC when a session starts/ends. It is also set by the CPU to initiate the Session Request Protocol. When the MUSBHDRC is in Suspend mode, the bit may be cleared by the CPU to perform a software disconnect.</td><td>RW</td><td>RW</td></tr>
</table>

34.5 Indexed Register

NOTES:

1 The action of the following registers when the selected endpoint has not been configured is undefined.

34.5.1 CSR0

CSR0 is a 16-bit register that provides control and status bits for Endpoint 0.

NOTES:

1 The interpretation of the register depends on whether the MUSBHDRC is acting as a peripheral or as a host. Users should also be aware that the value returned when the register is read reflects the status attained e.g. as a result of writing to the register. (with the Index register set to 0)

Peripheral Mode:

CSR0 **0x13440012**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							FlushFIFO (self-clr)	Serviced SetupEnd (self-clr)	Serviced RxPktRdy (self-clr)	SendStall (self-clr)	SetupEnd	DataEnd (self-clr)	SentStall	TxPktRdy (self-clr)	RxPktRdy
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15:9	Reserved	*Unused. Return 0 when read.*	R	R
8	FlushFIFO	The CPU writes a 1 to this bit to flush the next packet to be transmitted/read from the Endpoint 0 FIFO. The FIFO pointer is reset and the TxPktRdy/RxPktRdy bit (below) is cleared. **NOTE:** FlushFIFO has no effect unless TxPktRdy/RxPktRdy is set.	Set	R
7	Serviced SetupEnd	The CPU writes a 1 to this bit to clear the SetupEnd bit. It is cleared automatically.	Set	R
6	Serviced RxPKtRdy	The CPU writes a 1 to this bit to clear the RxPktRdy bit. It is cleared automatically.	Set	R

5	SendStall	The CPU writes a 1 to this bit to terminate the current transaction. The STALL handshake will be transmitted and then this bit will be cleared automatically.	Set	R
4	SetupEnd	This bit will be set when a control transaction ends before the DataEnd bit has been set. An interrupt will be generated and the FIFO flushed at this time. The bit is cleared by the CPU writing a 1 to the ServicedSetupEnd bit.	R	Set
3	DataEnd	The CPU sets this bit: 1 When setting TxPktRdy for the last data packet. 2 When clearing RxPktRdy after unloading the last data packet. 3 When setting TxPktRdy for a zero length data packet. It is cleared automatically.	Set	R
2	SentStall	This bit is set when a STALL handshake is transmitted. The CPU should clear this bit.	R/clr	Set
1	TxPKtRdy	The CPU sets this bit after loading a data packet into the FIFO. It is cleared automatically when a data packet has been transmitted. An interrupt is also generated at this point (if enabled).	R/set	R
0	RxPKtRdy	This bit is set when a data packet has been received. An interrupt is generated when this bit is set. The CPU clears this bit by setting the ServicedRxPktRdy bit.	R	Set

Host Mode:

CSR0 **0x13440012**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved							FlushFIFO (self-clr)	NAK Timeout	Status Pkt	ReqPkt	Error	SetupPkt	RxStall	TxPktRdy	RxPktRdy
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15:9	Reserved	*Unused. Return 0 when read.*	R	R
8	FlushFIFO	The CPU writes a 1 to this bit to flush the next packet to be transmitted/read from the Endpoint 0 FIFO. The FIFO pointer is reset and the TxPktRdy/RxPktRdy bit (below) is cleared. **NOTE:** FlushFIFO has no effect unless TxPktRdy/RxPktRdy is set.	Set	R
7	NAK Timeout	This bit will be set when Endpoint 0 is halted following the receipt of NAK responses for longer than the time set by the NAKLimit0 register. The CPU should clear this bit to allow the endpoint to continue.	R/clr	Set
6	StatusPkt	The CPU sets this bit at the same time as the TxPktRdy or ReqPkt bit is set, to perform a status stage transaction. Setting this bit ensures that the data toggle is set to 1 so that a DATA1 packet is used for the Status Stage transaction.	RW	R
5	ReqPkt	The CPU sets this bit to request an IN transaction. It is cleared when RxPktRdy is set.	RW	RW
4	Error	This bit will be set when three attempts have been made to perform a transaction with no response from the peripheral. The CPU should clear this bit. An interrupt is generated when this bit is set.	R	Set
3	SetupPkt	The CPU sets this bit, at the same time as the TxPktRdy bit is set, to send a SETUP token instead of an OUT token for the transaction.	RW	RW
2	RxStall	This bit is set when a STALL handshake is received. The CPU should clear this bit.	R/clr	Set
1	TxPktRdy	The CPU sets this bit after loading a data packet into the FIFO. It is cleared automatically when the data packet has been transmitted. An interrupt is generated (if enabled) when the bit is cleared.	R/set	Clr

0	RxPktRdy	This bit is set when a data packet has been received. An interrupt is generated (if enabled) when this bit is set. The CPU should clear this bit when the packet has been read from the FIFO.	R/clr	RW

34.5.2 Count0

Count0 is a 7-bit read-only register that indicates the number of received data bytes in the Endpoint 0 FIFO. The value returned changes as the contents of the FIFO change and is only valid while RxPktRdy (CSR0.D0) is set.

Count0 **0x13440018**

Bit	6	5	4	3	2	1	0
	Endpoint 0 Rx Count						
RST	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
6:0	Endpoint 0 Rx Count	Number of received data bytes in the endpoint 0 FIFO.	R	W

34.5.3 ConfigData

ConfigData is an 8-bit Read-Only register that returns information about the selected core configuration.

ConfigData **0x1344001f**

Bit	7	6	5	4	3	2	1	0
	MPRxE	MPTxE	BigEndian	HBRxE	HBTxE	DynFIFOSizing	SoftConE	UTMI Data Width
RST	?	?	?	?	?	?	?	?

Bits	Name	Description	CPU	USB
7	MPRxE	When set to '1', automatic amalgamation of bulk packets is selected.	R	R
6	MPTxE	When set to '1', automatic splitting of bulk packets is selected.	R	R

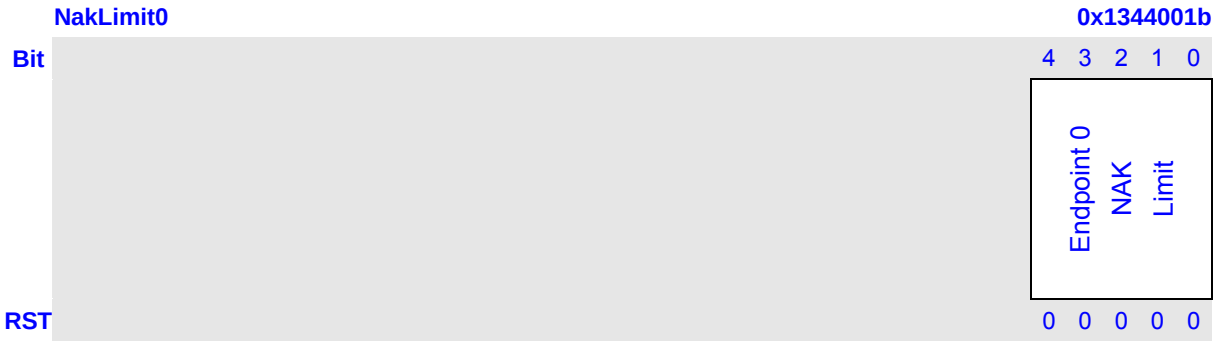
5	BigEndian	When set to '1' indicates Big Endian ordering is selected.	R	R
4	HBRxE	When set to '1' indicates High-bandwidth Rx ISO Endpoint Support selected.	R	R
3	HBTxE	When set to '1' indicates High-bandwidth Tx ISO Endpoint Support selected.	R	R
2	DynFIFOSizing	When set to '1' indicates Dynamic FIFO Sizing option selected.	R	R
1	SoftConE	When set to '1' indicates Soft Connect/Disconnect option selected.	R	R
0	UTMI Data Width	Indicates selected UTMI+ data width. 0 => 8 bits; 1 => 16 bits.	R	R

34.5.4 NakLimit0 (Host Mode Only)

NAKLimit0 is a 5-bit register that sets the number of frames/microframes (High-Speed transfers) after which Endpoint 0 should timeout on receiving a stream of NAK responses. (Equivalent settings for other endpoints can be made through their TxInterval and RxInterval registers.
The number of frames/microframes selected is $2^{(m-1)}$ (where *m* is the value set in the register, valid values 2 – 16). If the host receives NAK responses from the target for more frames than the number represented by the Limit set in this register, the endpoint will be halted.

NOTES:

1 A value of 0 or 1 disables the NAK timeout function.



Bits	Name	Description	CPU	USB
4:0	Endpoint 0 NAK Limit	Endpoint 0 NAK limit.	RW	R

34.5.5 TxMaxP

The TxMaxP register defines the maximum amount of data that can be transferred through the selected Tx endpoint in a single operation. There is a TxMaxP register for each Tx endpoint (except Endpoint 0).

Bits 10:0 define (in bytes) the maximum payload transmitted in a single transaction. The value set can be up to 1024 bytes but is subject to the constraints placed by the USB Specification on packet sizes for Bulk, Interrupt and Isochronous transfers in Full speed and High-speed operations.

Where the option of High-bandwidth Isochronous/Interrupt endpoints or of packet splitting on Bulk endpoints has been taken when the core is configured, the register includes either 2 or 5 further bits that define a multiplier *m* which is equal to one more than the value recorded.

In the case of Bulk endpoints with the packet splitting option enabled, the multiplier *m* can be up to 32 and defines the maximum number of 'USB' packets (i.e. packets for transmission over the USB) of the specified payload into which a single data packet placed in the FIFO should be split, prior to transfer. (If the packet splitting option is not enabled, D15–D13 is not implemented and D12–D11 (if included) is ignored.) **NOTE:** The data packet is required to be an exact multiple of the payload specified by bits 10:0, which is itself required to be either 8, 16, 32, 64 or (in the case of High Speed transfers) 512 bytes.

For Isochronous/Interrupt endpoints operating in High-Speed mode and with the High-bandwidth option enabled, *m* may only be either 2 or 3 (corresponding to bit 11 set or bit 12 set, respectively) and it specifies the maximum number of such transactions that can take place in a single microframe. If either bit 11 or bit 12 is non-zero, the MUSBHDRC will automatically split any data packet written to the FIFO into up to 2 or 3 'USB' packets, each containing the specified payload (or less). The maximum payload for each transaction is 1024 bytes, so this allows up to 3072 bytes to be transmitted in each microframe. (For Isochronous/Interrupt transfers in Full-speed mode or if High-bandwidth is not enabled, bits 11 and 12 are ignored.)

The value written to bits 10:0 (multiplied by *m* in the case of high-bandwidth Isochronous/Interrupt transfers) must match the value given in the *wMaxPacketSize* field of the Standard Endpoint Descriptor for the associated endpoint (see *USB Specification* Revision 2.0, Chapter 9). A mismatch could cause unexpected results. The total amount of data represented by the value written to this register (specified payload × *m*) must not exceed the FIFO size for the Tx endpoint, and should not exceed half the FIFO size if double-buffering is required.

If this register is changed after packets have been sent from the endpoint, the Tx endpoint FIFO should be completely flushed (using the FlushFIFO bit in TxCSR) after writing the new value to this register.

TxMaxP **0x13440010**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	m-1					Maximum Payload/transaction										
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15/12:11	m-1	Multiplier.	RW	R
10:0	Maximum Payload/transaction	Maximum payload transmitted in a single transaction.	RW	R

34.5.6 TxCSR

TxCSR is a 16-bit register that provides control and status bits for transfers through the currently-selected Tx endpoint. There is a TxCSR register for each configured Tx endpoint (not including Endpoint 0).

NOTES:

1 The interpretation of the register depends on whether the MUSBHDRC is acting as a peripheral or as a host. Users should also be aware that the value returned when the register is read reflects the status attained e.g. as a result of writing to the register.

Peripheral Mode

TxCSR **0x13440012**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	AutoSet	ISO	Mode	DMAReqEnab	FrcDataTog	DMAReqMode	Reserved		IncompTx	ClrDataTog	SentStall	SendStall	FlushFIFO(sefl-clr)	UnderRun	FIFO NotEmpty	TxPktRdy
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15	AutoSet	If the CPU sets this bit, TxPktRdy will be automatically set when data of the maximum packet size (value in TxMaxP) is loaded into the Tx FIFO. If a packet of less than the maximum packet size is loaded, then TxPktRdy will have to be set manually. **NOTE:** *Should not be set for high-bandwidth Isochronous endpoints.*	RW	R

14	ISO	The CPU sets this bit to enable the Tx endpoint for Isochronous transfers, and clears it to enable the Tx endpoint for Bulk or Interrupt transfers. **NOTE:** *This bit only has any effect in Peripheral mode. In Host mode,* it always returns zero.	RW	R
13	Mode	The CPU sets this bit to enable the endpoint direction as Tx, and clears the bit to enable it as Rx. **NOTE:** This bit *only* has any effect where the same endpoint FIFO is used for both Tx and Rx transactions.	RW	R
12	DMAReqEnab	The CPU sets this bit to enable the DMA request for the Tx endpoint.	RW	R
11	FrcDataTog	The CPU sets this bit to force the endpoint data toggle to switch and the data packet to be cleared from the FIFO, regardless of whether an ACK was received. This can be used by Interrupt Tx endpoints that are used to communicate rate feedback for Isochronous endpoints.	RW	R
10	DMAReqMode	The CPU sets this bit to select DMA. Request Mode 1 and clears it to select DMA. Request Mode 0. **NOTE:** This bit must not be cleared either before or in the same cycle as the above DMAReqEnab bit is cleared.	RW	R
9:8	Reserved	Unused, always return 0.	R	R
7	InCompTx	When the endpoint is being used for high-bandwidth Isochronous/Interrupt transfers, this bit is set to indicate where a large packet has been split into 2 or 3 packets for transmission but insufficient IN tokens have been received to send all the parts. **NOTE:** *In anything other than a high-bandwidth transfer, this bit will always return 0.*	R/clr	Set
6	ClrDataTog	The CPU writes a 1 to this bit to reset the endpoint data toggle to 0.	Set	R/clr

5	SentStall	This bit is set when a STALL handshake is transmitted. The FIFO is flushed and the TxPktRdy bit is cleared (see below). The CPU should clear this bit.	R/clr	Set
4	SendStall	The CPU writes a 1 to this bit to issue a STALL handshake to an IN token. The CPU clears this bit to terminate the stall condition. **NOTE:** *This bit has no effect where the endpoint is being used for Isochronous transfers.*	RW	R
3	FlushFIFO	The CPU writes a 1 to this bit to flush the latest packet from the endpoint Tx FIFO. The FIFO pointer is reset, the TxPktRdy bit (below) is cleared and an interrupt is generated. May be set simultaneously with TxPktRdy to abort the packet that is currently being loaded into the FIFO. **NOTE:** FlushFIFO has no effect unless TxPktRdy is set. Also note that, if the FIFO is double-buffered, FlushFIFO may need to be set twice to completely clear the FIFO.	Set	R
2	UnderRun	The USB sets this bit if an IN token is received when the TxPktRdy bit not set. The CPU should clear this bit.	R/clr	Set
1	FIFONotEmpty	The USB sets this bit when there is at least 1 packet in the Tx FIFO.	R/clr	Set
0	TxPktRdy	The CPU sets this bit after loading a data packet into the FIFO. It is cleared automatically when a data packet has been transmitted. An interrupt is also generated at this point (if enabled). TxPktRdy is also automatically cleared (but no interrupt is generated) prior to loading a second packet into a double-buffered FIFO.	R/set	Clr

Host Mode:

TxCSR **0x13440012**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	AutoSet	Reserved	Mode	DMAReqEnab	FrcDataTog	DMAReqMode	Reserved		NAK Timeout	ClrDataTog	RxStall	Reserved	FlushFIFO (self-lcr)	Error	FIFONotEmpty	TxPktRdy
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15	AutoSet	If the CPU sets this bit, TxPktRdy will be automatically set when a packet of the maximum packet size (TxMaxP) is loaded into the Tx FIFO. If a packet of less than the maximum packet size is loaded, then TxPktRdy will have to be set manually. **NOTE:** *Should not be set for high-bandwidth Isochronous endpoints.*	RW	R
14	Reserved	Unused, always returns zero.	RW	R
13	Mode	The CPU sets this bit to enable the endpoint direction as Tx, and clears it to enable the endpoint direction as Rx. **NOTE:** *This bit only has any effect where the same endpoint FIFO is used for both Tx and Rx transactions.*	RW	R
12	DMAReqEnab	The CPU sets this bit to enable the DMA request for the Tx endpoint.	RW	R
11	FrcDataTog	The CPU sets this bit to force the endpoint data toggle to switch and the data packet to be cleared from the FIFO, regardless of whether an ACK was received. This can be used by Interrupt Tx endpoints that are used to communicate rate feedback for Isochronous endpoints.	RW	R
10	DMAReqMode	The CPU sets this bit to select DMA Request Mode 1 and clears it to select DMA Request Mode 0. **NOTE:** *This bit must not be cleared either before or in the same cycle as the above DMAReqEnab bit is cleared*.	RW	R
9:8	Reserved	Unused, always returns 0.	R	R

7	NAKTimeout	This bit will be set when the Tx endpoint is halted following the receipt of NAK responses for longer than the time set as the NAK Limit by the TxInterval register. The CPU should clear this bit to allow the endpoint to continue. **NOTE:** *Valid only for Bulk endpoints.*	R/clr	Set
6	ClrDataTog	The CPU writes a 1 to this bit to reset the endpoint data toggle to 0.	Set	R/clr
5	RxStall	This bit is set when a STALL handshake is received. When this bit is set, any DMA request that is in progress is stopped, the FIFO is completely flushed and the TxPktRdy bit is cleared (see below). The CPU should clear this bit.	R/clr	Set
4	Reserved	Unused, always returns 0.	R	R
3	FlushFIFO	The CPU writes a 1 to this bit to flush the latest packet from the endpoint Tx FIFO. The FIFO pointer is reset, the TxPktRdy bit (below) is cleared and an interrupt is generated. May be set simultaneously with TxPktRdy to abort the packet that is currently being loaded into the FIFO. **NOTE:** FlushFIFO has no effect unless TxPktRdy is set. Also note that, if the FIFO is double-buffered, FlushFIFO may need to be set twice to completely clear the FIFO.	Set	R
2	Error	The USB sets this bit when 3 attempts have been made to send a packet and no handshake packet has been received. When the bit is set, an interrupt is generated, TxPktRdy is cleared and the FIFO is completely flushed. The CPU should clear this bit. *Valid only when the endpoint is operating in Bulk or Interrupt mode.*	R/clr	RW
1	FIFONotEmpty	The USB sets this bit when there is at least 1 packet in the Tx FIFO.	R/clr	Set

0	TxPktRdy	The CPU sets this bit after loading a data packet into the FIFO. It is cleared automatically when a data packet has been transmitted. An interrupt is also generated at this point (if enabled). TxPktRdy is also automatically cleared (but no interrupt is generated) prior to loading a second packet into a double-buffered FIFO.	R/set	Clr

34.5.7 RxMaxP

The RxMaxP register defines the maximum amount of data that can be transferred through the selected Rx endpoint in a single operation. There is a RxMaxP register for each Rx endpoint (except Endpoint 0).

Bits 10:0 define (in bytes) the maximum payload transmitted in a single transaction. The value set can be up to 1024 bytes but is subject to the constraints placed by the USB Specification on packet sizes for Bulk, Interrupt and Isochronous transfers in Full speed and High-speed operations.

Where the option of High-bandwidth Isochronous/Interrupt endpoints or of combining Bulk packets has been taken when the core is configured, the register includes either 2 or 5 further bits that define a multiplier *m* which is equal to one more than the value recorded.

For Bulk endpoints with the packet combining option enabled, the multiplier *m* can be up to 32 and defines the number of USB packets of the specified payload which are to be combined into a single data packet within the FIFO. (If the packet splitting option is not enabled, D15–D13 is not implemented and D12–D11 (if included) is ignored.)

For Isochronous/Interrupt endpoints operating in High-Speed mode and with the High-bandwidth option enabled, *m* may only be either 2 or 3 (corresponding to bit 11 set or bit 12 set, respectively) and it specifies the maximum number of such transactions that can take place in a single microframe. If either bit 11 or bit 12 is non-zero, the MUSBHDRC will automatically combine the separate USB packets received in any microframe into a single packet within the Rx FIFO. The maximum payload for each transaction is 1024 bytes, so this allows up to 3072 bytes to be received in each microframe. (For Isochronous/Interrupt transfers in Full-speed mode or if High-bandwidth is not enabled, bits 11 and 12 are ignored.)

The value written to bits 10:0 (multiplied by *m* in the case of high-bandwidth Isochronous/Interrupt transfers) must match the value given in the *wMaxPacketSize* field of the Standard Endpoint Descriptor for the associated endpoint (see *USB Specification* Revision 2.0, Chapter 9). A mismatch could cause unexpected results.

The total amount of data represented by the value written to this register (specified payload × *m*) must

not exceed the FIFO size for the OUT endpoint, and should not exceed half the FIFO size if double-buffering is required.

RxMaxP **0x13440014**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	m-1					Maximum Payload/transaction										
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15/12 :11	m-1	Multiplier.	RW	R
10:0	Maximum Payload/transaction	Maximum payload transmitted in a single transaction.	RW	R

34.5.8 RxCSR

RxCSR is an 16-bit register that provides control and status bits for transfers through the currently-selected Rx endpoint. There is an RxCSR register for each configured Rx endpoint (not including Endpoint 0).

NOTES:

1 The interpretation of the register depends on whether the MUSBHDRC is acting as a peripheral or as a host. Users should also be aware that the value returned when the register is read reflects the status attained e.g. as a result of writing to the register.

Peripheral Mode:

RxCSR **0x13440016**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	AutoClear	ISO	DMAReqEnab	DisNyet	DMAReqMode	Reserved		IncompRx	ClrDataTog	SentStall	SendStall	FlushFIFO(sefi-clr)	DataError	OverRun	FIFOFull (self-clr)	RxPktRdy
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15	AutoClear	If the CPU sets this bit then the RxPktRdy bit will be automatically cleared when a packet of RxMaxP bytes has been unloaded from the Rx FIFO. When packets of less than the maximum packet size are unloaded, RxPktRdy will have to be cleared manually. **NOTE:** *Should not be set for highbandwidth* Isochronous endpoints.	RW	R
14	ISO	The CPU sets this bit to enable the Rx endpoint for Isochronous transfers, and clears it to enable the Rx endpoint for Bulk/Interrupt transfers.	RW	R
13	DMAReqEnab	The CPU sets this bit to enable the DMA request for the Rx endpoint.	RW	R
12	DisNyet	The CPU sets this bit to disable the sending of NYET handshakes. When set, all successfully received Rx packets are ACK'd including at the point at which the FIFO becomes full. **NOTE:** *This bit only has any effect in High-speed mode, in which mode it should be set for all Interrupt endpoints.*	RW	R
11	DMAReqMode	The CPU sets this bit to select DMA. Request Mode 1 and clears it to select DMA. Request Mode 0.	RW	R
10:9	Reserved	Unused, always return zero.	R	R
8	IncompRx	This bit is set in a high-bandwidth Isochronous transfer if the packet in the Rx FIFO is incomplete because parts of the data were not received. It is cleared when RxPktRdy is cleared. **NOTE:** *In anything* other than a high-bandwidth Isochronous transfer, this bit will always return 0.	R	Set
7	ClrDataTog	The CPU writes a 1 to this bit to reset the endpoint data toggle to 0.	R	R/clr
6	SentStall	This bit is set when a STALL handshake is transmitted. The CPU should clear this bit.	R/clr	Set

5	SendStall	The CPU writes a 1 to this bit to issue a STALL handshake. The CPU clears this bit to terminate the stall condition. **NOTE:** *This bit has no effect where the endpoint is being used for Isochronous transfers.*	RW	R
4	FlushFIFO	The CPU writes a 1 to this bit to flush the next packet to be read from the endpoint Rx FIFO. The FIFO pointer is reset and the RxPktRdy bit (below) is cleared. **NOTE:** FlushFIFO has no effect unless RxPktRdy is set. Also note that, if the FIFO is double-buffered, FlushFIFO may need to be set twice to completely clear the FIFO.	Set	R
3	DataError	This bit is set when RxPktRdy is set if the data packet has a CRC or bit-stuff error. It is cleared when RxPktRdy is cleared. **NOTE:** *This bit is only valid when the endpoint is operating in ISO mode. In Bulk mode, it always returns zero.*	R	Set
2	OverRun	This bit is set if an OUT packet cannot be loaded into the Rx FIFO. The CPU should clear this bit. **NOTE:** *This bit is only valid when the endpoint is operating in ISO mode. In Bulk mode, it always returns zero.*	R/clr	Set
1	FIFOFull	This bit is set when no more packets can be loaded into the Rx FIFO.	R	Set
0	RxPktRdy	This bit is set when a data packet has been received. The CPU should clear this bit when the packet has been unloaded from the Rx FIFO. An interrupt is generated when the bit is set.	R/clr	Set

Host Mode:

RxCSR **0x13440016**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	AutoClear	AutoReq	DMAReqEnab	DisNyet	DMAReqMode	Reserved		IncompRx	ClrDataTog	RxStall	ReqPkt	FlushFIFO(sefl-clr)	DataError/NAKTimeout	Error	FIFOFull (self-clr)	RxPktRdy
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15	AutoClear	If the CPU sets this bit then the RxPktRdy bit will be automatically cleared when a packet of RxMaxP bytes has been unloaded from the Rx FIFO. When packets of less than the maximum packet size are unloaded, RxPktRdy will have to be cleared manually. **NOTE:** *Should not be set for highbandwidth Isochronous endpoints.*	RW	R
14	AutoReq	If the CPU sets this bit, the ReqPkt bit will be automatically set when the RxPktRdy bit is cleared.	RW	R
13	DMAReqEnab	The CPU sets this bit to enable the DMA request for the Rx endpoint.	RW	R
12	DisNyet	The CPU sets this bit to disable the sending of NYET handshakes. When set, all successfully received Rx packets are ACK'd including at the point at which the FIFO becomes full. **NOTE:** *This bit only has any effect in High-speed mode, in which mode it should be set for all Interrupt endpoints.*	RW	R
11	DMAReqMode	The CPU sets this bit to select DMA. Request Mode 1 and clears it to select DMA. Request Mode 0.	RW	R
10:9	Reserved	Unused, always return 0.	R	R

8	IncompRx	This bit will be set in a high-bandwidth Isochronous transfer if the packet received is incomplete. It will be cleared when RxPktRdy is cleared. **NOTE:** *If USB protocols are followed correctly, this bit should never be set. The bit becoming set indicates a failure of the associated Peripheral device to behave correctly. (In anything other than a high-bandwidth Isochronous transfer, this bit will always return 0.)*	R	Set
7	ClrDataTog	The CPU writes a 1 to this bit to reset the endpoint data toggle to 0.	Set	R/clr
6	RxStall	When a STALL handshake is received, this bit is set and an interrupt is generated. The CPU should clear this bit.	R/clr	Set
5	ReqPkt	The CPU writes a 1 to this bit to request an IN transaction. It is cleared when RxPktRdy is set.	RW	RW
4	FlushFIFO	The CPU writes a 1 to this bit to flush the next packet to be read from the endpoint Rx FIFO. The FIFO pointer is reset and the RxPktRdy bit (below) is cleared. **NOTE:** FlushFIFO has no effect unless RxPktRdy is set. Also note that, if the FIFO is double-buffered, FlushFIFO may need to be set twice to completely clear the FIFO.	Set	R
3	DataError/NAKTimeout	When operating in ISO mode, this bit is set when RxPktRdy is set if the data packet has a CRC or bit-stuff error and cleared when RxPktRdy is cleared. In Bulk mode, this bit will be set when the Rx endpoint is halted following the receipt of NAK responses for longer than the time set as the NAK Limit by the RxInterval register. The CPU should clear this bit to allow the endpoint to continue.	R/clr	Set

2	Error	The USB sets this bit when 3 attempts have been made to receive a packet and no data packet has been received. The CPU should clear this bit. An interrupt is generated when the bit is set. **NOTE:** *This bit is only valid when the Tx endpoint is operating in Bulk or Interrupt mode. In ISO mode, it always returns zero.*	R/clr	Set
1	FIFOFull	This bit is set when no more packets can be loaded into the Rx FIFO.	R	Set
0	RxPktRdy	This bit is set when a data packet has been received. The CPU should clear this bit when the packet has been unloaded from the Rx FIFO. An interrupt is generated when the bit is set.	R/clr	Set

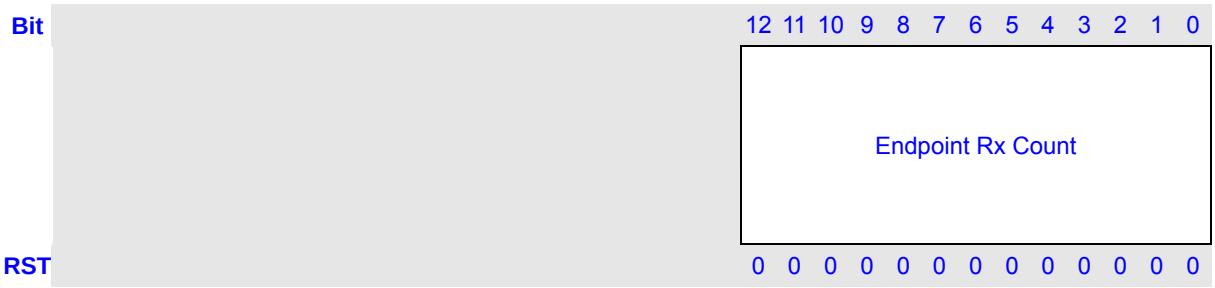
34.5.9 RxCount

RxCount is a 13-bit read-only register that holds the number of received data bytes in the packet in the Rx FIFO.

NOTES:

1 The value returned changes as the FIFO is unloaded and is only valid while RxPktRdy (RxCSR.D0) is set.

RxCount **0x13440018**



Bits	Name	Description	CPU	USB
12:0	Endpoint Rx Count	Number of received data bytes in the packet in the Rx FIFO.	R	W

34.5.10 TxType (Host Mode Only)

TxType is a 6-bit register that should be written with the endpoint number to be targeted by the endpoint in the lower 4 bits, and the transaction protocol to use for the currently-selected Tx endpoint in the upper 2 bits. There is a TxType register for each configured Tx endpoint (except Endpoint 0).

TxType 0x1344001a

Bit	5	4	3	2	1	0
	Protocol		Target Endpoint Number			
RST	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
5:4	Protocol	The CPU should set this to select the required protocol for the Tx endpoint. 00: Illegal 01: Isochronous 10: Bulk 11: Interrupt	RW	R
3:0	Target Endpoint Number	The CPU should set this value to the endpoint number contained in the Tx endpoint descriptor returned to the MUSBHDRC during device enumeration.	RW	R

34.5.11 TxInterval (Host Mode Only)

TxInterval is an 8-bit register that, for Interrupt and Isochronous transfers, defines the polling interval for the currently-selected Tx endpoint. For Bulk endpoints, this register sets the number of frames/microframes after which the endpoint should timeout on receiving a stream of NAK responses. There is a TxInterval register for each configured Tx endpoint (except Endpoint 0).

In each case the value that is set defines a number of frames/microframes (High Speed transfers), as follows:

Transfer Type	Speed	Valid Values (m)	Interpretation
Interrupt	LS or FS	1-255	Polling interval is *m* frames
	HS	1-16	Polling interval is $2^{(m-1)}$ microframes
Isochronous	FS or HS	1-16	Polling interval is $2^{(m-1)}$ frames/microframes
Bulk	FS or HS	2-16	NAK Limit is $2^{(m-1)}$ frames/microframes **NOTE:** A value of 0 or 1 disables the NAK timeout function.

TxInterval 0x1344001b

Bit	7	6	5	4	3	2	1	0
	Tx Polling Interval/ NAK Limit (m)							
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
7:0	Tx Polling Interval/NAK Limit	For interrupt and isochronous transfers, defines the polling interval for the currently selected Tx endpoint. For Bulk endpoints, this register sets the number of frames/microframes after which the endpoint should timeout on receiving a stream of NAK responses.	RW	R

34.5.12 RxType (Host Mode Only)

RxType is a 6-bit register that should be written with the endpoint number to be targeted by the endpoint in the lower 4 bits, and the transaction protocol to use for the currently-selected Rx endpoint in the upper 2 bits. There is an RxType register for each configured Rx endpoint (except Endpoint 0).

RxType 0x1344001c

Bit	5	4	3	2	1	0
	Protocol		Target Endpoint Number			
RST	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
5:4	Protocol	The CPU should set this to select the required protocol for the Rx endpoint. 00: Illegal 01: Isochronous 10: Bulk 11: Interrupt	RW	R
3:0	Target Endpoint Number	The CPU should set this value to the endpoint number contained in the Rx endpoint descriptor returned to the MUSBHDRC during device enumeration.	RW	R

34.5.13 RxInterval

RxInterval is an 8-bit register that, for Interrupt and Isochronous transfers, defines the polling interval for the currently-selected Rx endpoint. For Bulk endpoints, this register sets the number of frames/microframes after which the endpoint should timeout on receiving a stream of NAK responses. There is a RxInterval register for each configured Rx endpoint (except Endpoint 0). In each case the value that is set defines a number of frames/microframes (High Speed transfers), as follows:

Transfer Type	Speed	Valid Values (m)	Interpretation
Interrupt	LS or FS	1-255	Polling interval is *m* frames
	HS	1-16	Polling interval is $2^{(m-1)}$ microframes
Isochronous	FS or HS	1-16	Polling interval is $2^{(m-1)}$ frames/microframes
Bulk	FS or HS	2-16	NAK Limit is $2^{(m-1)}$ frames/microframes **NOTE:** A value of 0 or 1 disables the NAK timeout function.

RxInterval **0x1344001d**

Bit	7	6	5	4	3	2	1	0
	Rx Polling Interval/ NAK Limit (m)							
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
7:0	Rx Polling Interval/NAK Limit	For interrupt and isochronous transfers, defines the polling interval for the currently selected Tx endpoint. For Bulk endpoints, this register sets the number of frames/microframes after which the endpoint should timeout on receiving a stream of NAK responses.	RW	R

34.5.14 FifoSize

FIFOSize is an 8-bit Read-Only register that returns the sizes of the FIFOs associated with the selected additional Tx/Rx endpoints. The lower nibble encodes the size of the selected Tx endpoint FIFO; the upper nibble encodes the size of the selected Rx endpoint FIFO. Values of 3 – 13 correspond to a FIFO size of 2^n bytes (8 – 8192 bytes). If an endpoint has not been configured, a value of 0 will be displayed. Where the Tx and Rx endpoints share the same FIFO, the Rx FIFO size will be encoded as 0xF.

NOTES:

1 The register only has this interpretation when the Index register is set to select one of Endpoints 1 – 15 and Dynamic Sizing is not selected. It has a special interpretation when the Index register is set to select Endpoint 0 (see Section 5.3.3), while the result returned is not valid where Dynamic FIFO sizing is used. (Index register set to select Endpoints 1 – 15 only)

FifoSize **0x1344001f**

Bit	7	6	5	4	3	2	1	0
	Rx FIFO Size				Tx FIFO Size			
RST	?	?	?	?	?	?	?	?

Bits	Name	Description	CPU	USB
7:4	Rx FIFO Size	the sizes of the FIFOs associated with the selected additional Rx endpoints.	R	R
3:0	Tx FIFO Size	the sizes of the FIFOs associated with the selected additional Tx endpoints.	R	R

34.5.15 FIFOx

This address range provides 16 addresses for CPU access to the FIFOs for each endpoint. Writing to these addresses loads data into the TxFIFO for the corresponding endpoint. Reading from these addresses unloads data from the RxFIFO for the corresponding endpoint.

The address range is 20h – 5Fh and the FIFOs are located on 32-bit double-word boundaries (Endpoint 0 at 20h, Endpoint 1 at 24h ... Endpoint 15 at 5Ch).

34.6 Additional Multipoint Control / Status Registers

The following subsections detail additional control and status registers that are only valid when the multipoint option is enabled in the configuration GUI. If the multipoint option is not enabled these reregisters should not be accessed.

34.6.1 TxFuncAddr / RxFuncAddr

NOTE: REQUIRED IN HOST MODE!

TxFuncAddr and RxFuncAddr are 7-bit read/write registers that record the address of the target function that is to be accessed through the associated endpoint (EP*n*). TxFuncAddr needs to be defined for each TX endpoint that is used; RxFuncAddr needs to be defined for each Rx endpoint that is used.

NOTE: TxFuncAddr must be defined for Endpoint 0. The RxFuncAddr register does not exist on EP0.

TxFuncAddr/RxFuncAddr **0x13440080+8*n / 0x13440084+8*n**

Bit	7	6	5	4	3	2	1	0
	Addr of Target Function							
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
6:0	Addr of Target Function	read/write registers that record the address of the target function that is to be accessed through the associated endpoint.	RW	R

34.6.2 TxHubAddr/RxHubAddr

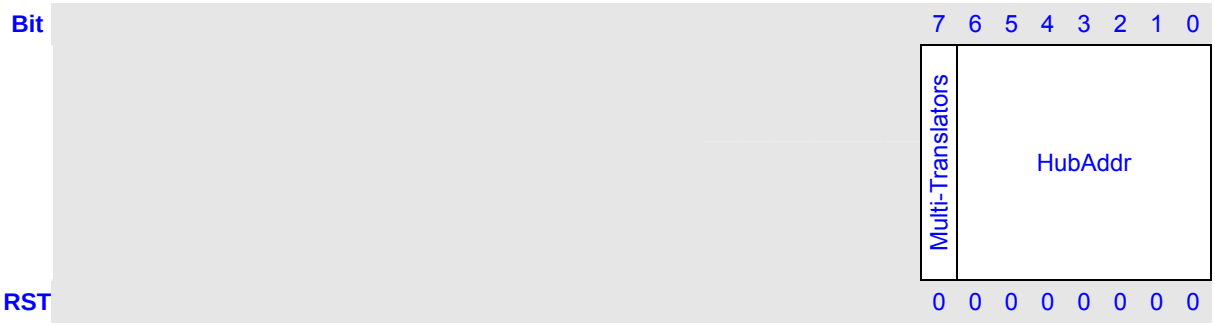
NOTE: RELEVANT IN HOST MODE ONLY!

TxHubAddr and RxHubAddr are 8-bit read/write registers which, like TxHubPort and RxHubPort, only need to be written where a full- or low-speed device is connected to TX/Rx Endpoint EP*n* via a high-speed USB 2.0 hub which carries out the necessary transaction translation to convert between high-speed transmission and full-/low-speed transmission. In such circumstances:

- the lower 7 bits should record the address of this USB 2.0 hub.
- the top bit should record whether the hub has multiple transaction translators (set to '0' if single transaction translator; set to '1' if multiple transaction translators).

NOTE: If Endpoint 0 is connected to a hub, then TxHubAddr must be defined for EP0. The RxHubAddr register does not exist on EP0.

TxHubAddr/RxHubAddr **0x13440082+8*n / 0x13440086+8*n**



Bits	Name	Description	CPU	USB
7	Multi-Translators	0: single transaction translator 1: multiple transaction translators	RW	R
6:0	HubAddr	Hub Address.	RW	R

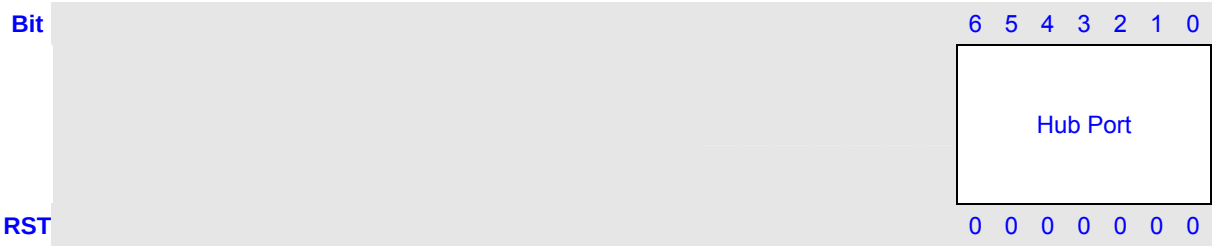
34.6.3 TxHubPort / RxHubPort

NOTE: RELEVANT IN HOST MODE ONLY!

TxHubPort and RxHubPort only need to be written where a full- or low-speed device is connected to TX/Rx Endpoint EP*n* via a high-speed USB 2.0 hub which carries out the necessary transaction translation. In such circumstances, these 7-bit read/write registers need to be used to record the port of that USB 2.0 hub through which the target associated with the endpoint is accessed.

NOTE: If Endpoint 0 is connected to a hub, then TxHubPort must be defined. The RxHubPort register does not exist on EP0.

TxHubPort/RxHubPort **0x13440083+8*n / 0x13440087+8*n**



Bits	Name	Description	CPU	USB
6:0	Hub Port	record the port of that USB 2.0 hub through which the target associated with the endpoint is accessed.	RW	R

34.7 Additional Control/Status Registers

34.7.1 VControl

NOTE: WRITE ONLY!

VControl is a UTMI+ PHY Vendor register that may optionally be included in the core when the core is configured. Its size is also configurable and may be up to 32 bits. The structure of the register is up to the system designer, though users should note that the UTMI+ specification defines a 4-bit VControl register.

Vcontrol **0x13440068**

Bit	3	2	1	0
	Vcontrol			
RST	?	?	?	?

Bits	Name	Description	CPU	USB
3:0	VControl	UTMI+ PHY Vendor registers.	RW	-

34.7.2 VStatus

NOTE: READ ONLY!

VStatus is a UTMI+ PHY Vendor register that may optionally be included in the core when the core is configured. Its size is also configurable and may be up to 32 bits. The structure of the register is up to the system designer, though users should note that the UTMI+ specification defines an 8-bit VStatus register.

VStatus **0x1344006b**

Bit	7	6	5	4	3	2	1	0
	VStatus							
RST	?	?	?	?	?	?	?	?

Bits	Name	Description	CPU	USB
7:0	VStatus	UTMI+ PHY Vendor registers.	RW	-

34.7.3 Hwvers

HWVers register is a 16-bit read-only register that returns information about the version of the RTL from which the core hardware was generated, in particular the RTL version number (v*xx*.*yyy*).

Hwvers **0x1344006c**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RC	xx					yyy									
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	CPU	USB
15	RC	Set to '1' if RTL used from a Release Candidate rather than from a full release of the core.	R	R
14:10	xx	Major Version Number. (Range 0 – 31)	R	R
9:0	yyy	Minor Version Number. (Range 0 – 999)	R	R

34.8 Additional Configuration Registers

34.8.1 EPInfo

This 8-bit read-only register allows read-back of the number of TX and Rx endpoints included in the design.

EPInfo **0x13440078**

	7	6	5	4	3	2	1	0
	RxEdps				TxEdps			
RST	?	?	?	?	?	?	?	?

Bits	Name	Description	CPU	USB
7:4	RxEdps	The number of Rx endpoints implemented in the design.	R	R
3:0	TxEdps	The number of TX endpoints implemented in the design.	R	R

34.8.2 RAMInfo

This 8-bit read-only register provides information about the width of the RAM.

RAMInfo **0x13440079**

Bit	7	6	5	4	3	2	1	0
	DMAChans				RAMBits			
RST	?	?	?	?	?	?	?	?

Bits	Name	Description	CPU	USB
7:4	DMAChans	The number of DMA channels implemented in the design.	R	R
3:0	RAMBits	The width of the RAM address bus.	R	R

34.8.3 LinkInfo

This 8-bit register allows some delays to be specified.

LinkInfo								0x1344007a
Bit	7	6	5	4	3	2	1	0
	WTCon				WTID			
	0	1	0	1	1	1	0	0

Bits	Name	Description	CPU	USB
7:4	WTCon	Sets the wait to be applied to allow for the user's connect/disconnect filter in units of 533.3ns. (The default setting corresponds to 2.667μs)	RW	R
3:0	WTID	Sets the delay to be applied from IDPULLUP being asserted to IDDIG being considered valid in units of 4.369ms. (The default setting corresponds to 52.43ms)	RW	R

34.8.4 VPLen

This 8-bit register sets the duration of the VBus pulsing charge.

VPlen								0x1344007b
Bit	7	6	5	4	3	2	1	0
	VPLen							
RST	0	0	1	1	1	1	0	0

Bits	Name	Description	CPU	USB
7:0	VPLen	Sets the duration of the VBus pulsing charge in units of 546.1 μs. (The default setting corresponds to 32.77ms)	RW	R

34.8.5 HS_EOF1

This 8-bit register sets the minimum time gap that is to be allowed between the start of the last transaction and the EOF for High-speed transactions.

HS_EOF1								0x1344007c
Bit	7	6	5	4	3	2	1	0
	HS_EOF1							
RST	1	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
7:0	HS_EOF1	Sets for High-speed transactions the time before EOF to stop beginning new transactions, in units of 133.3ns. (The default setting corresponds to 17.07μs)	RW	R

34.8.6 FS_EOF1

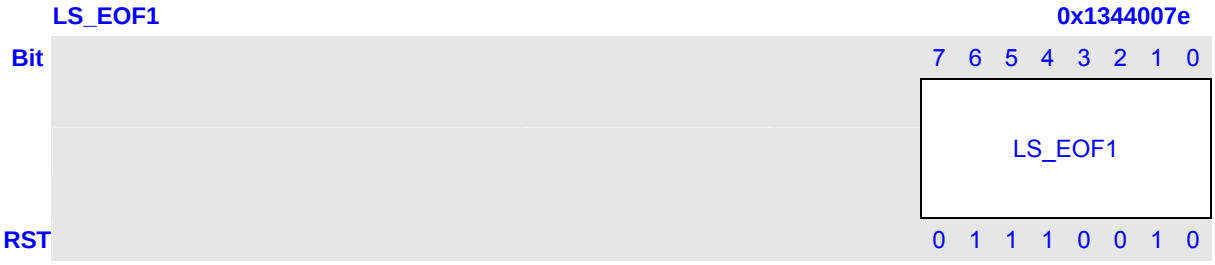
This 8-bit register sets the minimum time gap that is to be allowed between the start of the last transaction and the EOF for Full-speed transactions.

FS_EOF1								0x1344007d
Bit	7	6	5	4	3	2	1	0
	FS_EOF1							
RST	0	1	1	1	0	1	1	1

Bits	Name	Description	CPU	USB
7:0	FS_EOF1	Sets for Full-speed transactions the time before EOF to stop beginning new transactions, in units of 533.3ns. (The default setting corresponds to 63.46μs)	RW	R

34.8.7 LS_EOF1

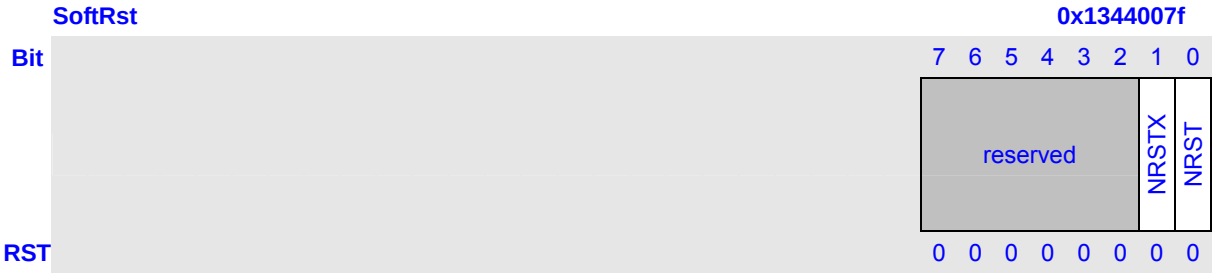
This 8-bit register sets the minimum time gap that is to be allowed between the start of the last transaction and the EOF for Low-speed transactions.



Bits	Name	Description	CPU	USB
7:0	LS_EOF1	Sets for Low-speed transactions the time before EOF to stop beginning new transactions, in units of 1.067μs. (The default setting corresponds to 121.6μs)	RW	R

34.8.8 SoftRst

This 8-bit register will assert LOW the output reset signals NRSTO and NRSTOX. This register is self clearing and will be reset by the input NRST.



Bits	Name	Description	CPU	USB
7:2	reserved	Unused, always returns zero.	-	-
1	NRSTX	The default value of this bit is 1'b0; When a 1 is written to this bit, the output NRSTXO will be asserted (LOW) within a minimum delay of 7 cycles of the CLK input. The output NRSTXO will be asynchronously asserted and synchronously de-asserted with respect to XCLK. This register is self clearing and will be reset by the input NRST.	RW	R

0	NRST	The default value of this bit is 1'b0; When a 1 is written to this bit, the output NRSTO will be asserted (LOW) within a minimum delay of 7 cycles of the CLK input. The output NRSTO will be asynchronously asserted and synchronously de-asserted with respect to CLK. This register is self clearing and will be reset by the input NRST.	RW	R

34.9 Extended Registers

34.9.1 RqPktCnt

NOTE: HOST MODE ONLY!

For each Rx Endpoint 1 – 15, the MUSBMHDRC provides a 16-bit RqPktCount register. This read/write register is used in Host mode to specify the number of packets that are to be transferred in a block transfer of one or more Bulk packets of length MaxP to Rx Endpoint *n*. The core uses the value recorded in this register to determine the number of requests to issue where the AutoReq option (included in the RxCSR register) has been set.

NOTE: Multiple packets combined into a single bulk packet within the FIFO count as one packet.

RqPktCnt **0x13440300+4*n**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	RqPktCnt															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15:0	RqPktCnt	Sets the number of packets of size MaxP that are to be transferred in a block transfer. *Only used in Host mode when AutoReq is set. Has no effect in Peripheral mode or when AutoReq is not set.*	RW	RW

34.9.2 RxDPktBufDis

Rx DPktBufDis is a 16-bit register that indicates which of the Rx endpoints have disabled the double packet buffer functionality described in section 8.4.2.2 of the MUSBMHDRC Product Specification.

NOTE: Bits relating to endpoints that have not been configured may be asserted by writing a '1' their respective register; however the disable bit will have no observable effect.

RxDPktBufDis **0x13440340**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EP15_Rx_Dis	EP14_Rx_Dis	EP13_Rx_Dis	EP12_Rx_Dis	EP11_Rx_Dis	EP10_Rx_Dis	EP9_Rx_Dis	EP8_Rx_Dis	EP7_Rx_Dis	EP6_Rx_Dis	EP5_Rx_Dis	EP4_Rx_Dis	EP3_Rx_Dis	EP2_Rx_Dis	EP1_Rx_Dis	Unused
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15	EP15_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 15.	RW	R
14	EP14_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 14.	RW	R
13	EP13_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 13.	RW	R
12	EP12_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 12.	RW	R
11	EP11_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 11.	RW	R
10	EP10_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 10.	RW	R
9	EP9_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 9.	RW	R
8	EP8_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 8.	RW	R
7	EP7_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 7.	RW	R
6	EP6_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 6.	RW	R
5	EP5_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 5.	RW	R
4	EP4_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 4.	RW	R
3	EP3_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 3.	RW	R
2	EP2_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 2.	RW	R
1	EP1_Rx_Dis	Rx Double Packet Buffer Disable for Endpoint 1.	RW	R
0	Unused	Reserved.	R	R

34.9.3 TxDPktBufDis

Tx DPktBufDis is a 16-bit register that indicates which of the TX endpoints have disabled the double packet buffer functionality described in section 8.4.1.2 of the MUSBMHDRC Product Specification.

NOTES:

1 Bits relating to endpoints that have not been configured may be asserted by writing a '1' their respective register; however the disable bit will have no observable effect.

TxDPktBufDis **0x13440342**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EP15_Tx_Dis	EP14_Tx_Dis	EP13_Tx_Dis	EP12_Tx_Dis	EP11_Tx_Dis	EP10_Tx_Dis	EP9_Tx_Dis	EP8_Tx_Dis	EP7_Tx_Dis	EP6_Tx_Dis	EP5_Tx_Dis	EP4_Tx_Dis	EP3_Tx_Dis	EP2_Tx_Dis	EP1_Tx_Dis	Unused
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
15	EP15_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 15.	RW	R
14	EP14_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 14.	RW	R
13	EP13_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 13.	RW	R
12	EP12_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 12.	RW	R
11	EP11_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 11.	RW	R

10	EP10_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 10.	RW	R
9	EP9_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 9.	RW	R
8	EP8_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 8.	RW	R
7	EP7_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 7.	RW	R
6	EP6_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 6.	RW	R
5	EP5_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 5.	RW	R
4	EP4_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 4.	RW	R
3	EP3_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 3.	RW	R
2	EP2_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 2.	RW	R
1	EP1_Tx_Dis	Tx Double Packet Buffer Disable for Endpoint 1.	RW	R
0	Unused	Reserved.	R	R

34.9.4 C_T_UCH

This register sets the chirp timeout. This number when multiplied by 4 represents the number of XCLK cycles before the timeout occurs. That is, if XCLK is 30MHz, this number represents the number of 133ns time intervals before the timeout occurs. If XCLK is 60MHz, this number represents the number of 67ns time intervals before the timeout occurs. Although this bit is written by the host in the CLK domain, the counter that utilizes this value is in the XCLK domain. No time domain crossing is provided as the value in this register is a static. The default value is the value of the compiler directive of the same name located in the configuration file musbmhdrc_cfg.v.

C_T_UCH 0x13440344

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	C_T_UCH															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	CPU	USB
15:0	C_T_UCH	Configurable Chirp Timeout timer; The default value is determined by compiler directive in musbhsfc_xcfg.v file. The default value is 203Ah if the host PHY data width is 16 bits (XCLK is 30MHz) and 4074h if the PHY data width is 8 bits (XCLK is 60Mhz) corresponding to a delay of 1.1ms.	RW	-

34.9.5 C_T_HHSRTN

This register sets the delay from the end of High Speed resume signaling (acting as a Host) to enable the UTM normal operating mode. This number when multiplied by 4 represents the number of XCLK cycles before the timeout occurs. That is, if XCLK is 30MHz, this number represents the number of

33.3ns time intervals before the timeout occurs. If XCLK is 60MHz, this number represents the number of 16.7ns time intervals before the timeout occurs. Although this bit is written by the host in the CLK domain, the counter that utilizes this value is in the XCLK domain. No time domain crossing is provided as the value in this register is a static. The default value is the value of the compiler directive of the same name located in the configuration file musbmhdrc_cfg.v.

C_T_HHSRTN **0x13440346**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	C_T_HHSRTN															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	CPU	USB
15:0	C_T_HHSRTN	The delay from the end of High Speed resume signaling to enabling UTM normal operating mode. The default value is determined by compiler directive in musbhsfc_xcfg.v file. The default value is 2F3h if the host PHY data width is 16 bits (XCLK is 30MHz) and 5E6h if the PHY data width is 8 bits (XCLK is 60Mhz) corresponding to a delay of 100us.	RW	-

34.9.6 C_T_HSBT

Per USB 2.0, Section 7.1.19.2, a high-speed host or device expecting a response to a transmission must not timeout the transaction if the interpacket delay is less than 736 bit times, and it must timeout the transaction if no signaling is seen within 816 bit times. This register represents the value to be added to the mimumum high speed timeout period of 736 bit times. The timeout period can be increased in increments of 64 high speed bit times (133 ns). There are 16 possible values. By default, the adder is 0 thus setting the high speed timeout to its minimum value. Use of this register will allow the high speed timeout to be set to values that are greater the maximum specified in USB 2.0 making the MUSBMHDRC non-compliant.

C_T_HSBT **0x13440348**

Bit	3	2	1	0
	C_T_HSBT			
RST	0	0	0	0

<table>
<tr><th>Bits</th><th>Name</th><th>Description</th><th>CPU</th><th>USB</th></tr>
<tr><td>3:0</td><td>C_T_HSBT</td><td>The value added to the minimum High Speed Timeout period (736 bit times) in increments of 64 High Speed bit times. This allows the turn around timeout period to be set to 16 possible values as follows:
<table>
<tr><th>Register Value</th><th>HS Turnaround Timeout (HS Bit times)</th><th>HS Turnaround Timeout (us)</th></tr>
<tr><td>0</td><td>736</td><td>1.534</td></tr>
<tr><td>1</td><td>800</td><td>1.667</td></tr>
<tr><td>2</td><td>864</td><td>1.801</td></tr>
<tr><td>3</td><td>928</td><td>1.934</td></tr>
<tr><td>4</td><td>992</td><td>2.067</td></tr>
<tr><td>5</td><td>1056</td><td>2.201</td></tr>
<tr><td>6</td><td>1120</td><td>2.334</td></tr>
<tr><td>7</td><td>1184</td><td>2.467</td></tr>
<tr><td>8</td><td>1248</td><td>2.601</td></tr>
<tr><td>9</td><td>1312</td><td>2.734</td></tr>
<tr><td>10</td><td>1376</td><td>2.868</td></tr>
<tr><td>11</td><td>1440</td><td>3.001</td></tr>
<tr><td>12</td><td>1504</td><td>3.134</td></tr>
<tr><td>13</td><td>1568</td><td>3.268</td></tr>
<tr><td>14</td><td>1632</td><td>3.401</td></tr>
<tr><td>15</td><td>1696</td><td>3.534</td></tr>
</table></td><td>RW</td><td>R</td></tr>
</table>

34.10 DMA Registers

34.10.1 DMA_INTR

This register provides an interrupt for each DMA channel. This interrupt register is cleared when read. When any bit of this register is set, the output DMA_NINT is asserted low. Events that cause interrupts to be set is described in section 17 (The optional DMA Controller description). Bits in this register will only be set if the DMA Interrupt Enable bit for the corresponding channel is enabled (register DMA_CNTL.D3).

DMA_INTR **0x13440200**

Bit	7	6	5	4	3	2	1	0
	CH8_INT	CH7_INT	CH6_INT	CH5_INT	CH4_INT	CH3_INT	CH2_INT	CH1_INT
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
7	CH8_INT	Channel 8 DMA Interrupt.	RW	Set
6	CH7_INT	Channel 7 DMA Interrupt.	RW	Set
5	CH6_INT	Channel 6 DMA Interrupt.	RW	Set
4	CH5_INT	Channel 5 DMA Interrupt.	RW	Set
3	CH4_INT	Channel 4 DMA Interrupt.	RW	Set
2	CH3_INT	Channel 3 DMA Interrupt.	R	Set
1	CH2_INT	Channel 2 DMA Interrupt.	R	Set
0	CH1_INT	Channel 1 DMA Interrupt.	R	Set

34.10.2 DMA_CNTL

This register is only available if the MUSBMHDRC is configured to use at least one internal DMA channel. This register provides the DMA transfer control for each channel. The enabling, transfer direction, transfer mode, the DMA burst modes are all controlled by this register.

DMA_CNTL **0x13440204+(n-1)*0x10**

Bit	10	9	8	7	6	5	4	3	2	1	0
	DMA_BRSTM		DMA_ERR	DMAEP				DMAIE	DMAMODE	DMA_DIR	DMA_EN
RST	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
10:9	DMA_BRSTM	Burst Mode. 00 : (Burst Mode 0 : Bursts of unspecified length) 01: (Burst Mode 1 : INCR4 or unspecified length) 10: (Burst Mode 2 : INCR8, INCR4 or unspecified length) 11: (Burst Mode 3 : INCR16, INCR8, INCR4 or unspecified length)	RW	R
8	DMA_ERR	Bus Error Bit. Indicates that a bus error has been observed on the input. AHB_HRESPM[1:0]. This bit is cleared by software.	RW	RW
7:4	DMAEP	The endpoint number this channel is assigned to.	RW	R
3	DMAIE	DMA Interrupt Enable.	RW	R
2	DMAMODE	This bit selects the DMA Transfer Mode. 0: DMA Mode0 Transfer 1: DMA Mode1 Transfer	RW	R
1	DMA_DIR	This bit selects the DMA Transfer Direction. 0: DMA Write (RX Endpoint) 1: DMa Read (TX Endpoint)	RW	R
0	DMA_ENAB	This bit enables the DMA transfer and will cause the transfer to begin.	RW	R

34.10.3 DMA_ADDR

This register identifies the current memory address of the corresponding DMA channel. The Initial memory address written to this register must have a value such that its modulo 4 value is equal to 0. That is, DMA_ADDR[1:0] must be equal to 2'b00. The lower two bits of this register are read only and cannot be set by software. As the DMA transfer progresses, the memory address will increment as bytes are transferred.

DMA_ADDR **0x13440208+(n-1)*0x10**

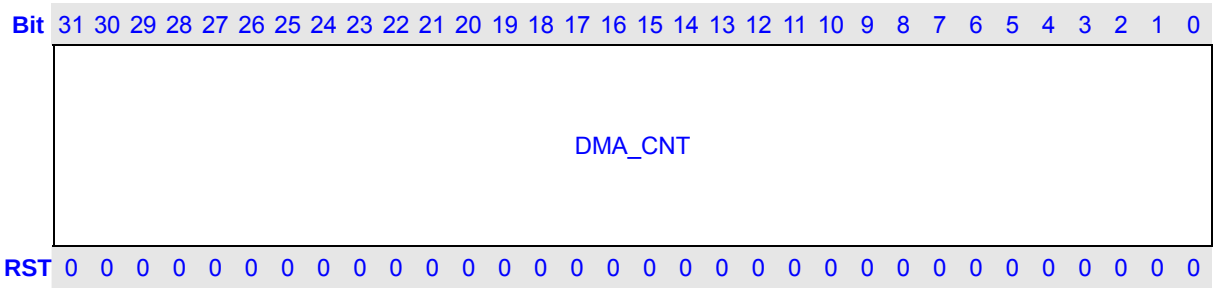
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DMA_ADDR																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	CPU	USB
31:0	DMA_ADDR	The DMA memory address. **Note that the initial memory address written to this register must have a value such that it's modulo 4 value is equal to 0. That is, DMA_ADDR[1:0] must be equal to 2'b00. The lower two bits of this register are read only and cannot be set by software.**	RW	RW

34.10.4 DMA_COUNT

This register identifies the current DMA count of the transfer. Software will set the initial count of the transfer which identifies the entire transfer length. As the count progresses this count is decremented as bytes are transferred.

DMA_CNT **0x1344020c+(n-1)*0x10**



31:0	DMA_CNT	The DMA memory address for the corresponding DMA channel. **NOTE:** If DMA is enabled with a count of 0, the bus will not be requested and a DMA interrupt will be generated.	RW	RW

35 MMC/SD CE-ATA Controller

35.1 Overview

The MultiMediaCard (MMC) is a universal low cost data storage and communication media that is designed to cover a wide area of applications such as electronic toys, organizers, PDAs, smart phones, and so on.

The Secure Digital (SD) card is an evolution of MMC, It is specifically designed to meet the security, capacity, performance, and environmental requirements inherent in newly emerging audio and video consumer electronic devices. The physical form factor, pin assignment, and data transfer protocol are forward compatible with the MultiMediaCard with some additions. An SD card can be categorized as SD memory or SD I/O card, commonly known as SDIO. A memory card invokes a copyright protection mechanism that complies with the security of the SDMI standard and is faster and capable of higher memory capacity. The SDIO card provides high-speed data I/O with low-power consumption for mobile electronic devices.

For CE-ATA detail protocol , please referred to WWW.CE-ATA.ORG.

Features of the MSC Controller include the following:

- Fully compatible with the *MMC System Specification version 4.2*
- Fully compatible with the *SD Memory Card Specification 2.0* and *SD I/O Specification 1.0* with 1 command channel and 4 data channels
- Consumer Electronics Advanced Transport Architecture (CE-ATA – version 1.1)
- 20-80 Mbps maximum data rate
- Support MMC data width 1bit ,4bit and 8bit
- Built-in programmable frequency divider for MMC/SD bus
- Maskable hardware interrupt for SDIO interrupt, internal status and FIFO status
- 32-entry x 32-bit built-in data FIFO
- Multi-SD function support including multiple I/O and combined I/O and memory
- IRQ supported enable card to interrupt MMC/SD controller
- Single or multi block access to the card including erase operation
- Stream access to the MMC card
- Supports SDIO read wait, interrupt detection during 1-bit or 4-bit access
- Supports CE-ATA digital protocol commands
- Support Command Completion Signal and interrupt to CPU
- Command Completion Signal disable feature
- The maximum block length is 4096bytes

35.2 Block Diagram

MSC Controller Block Diagram

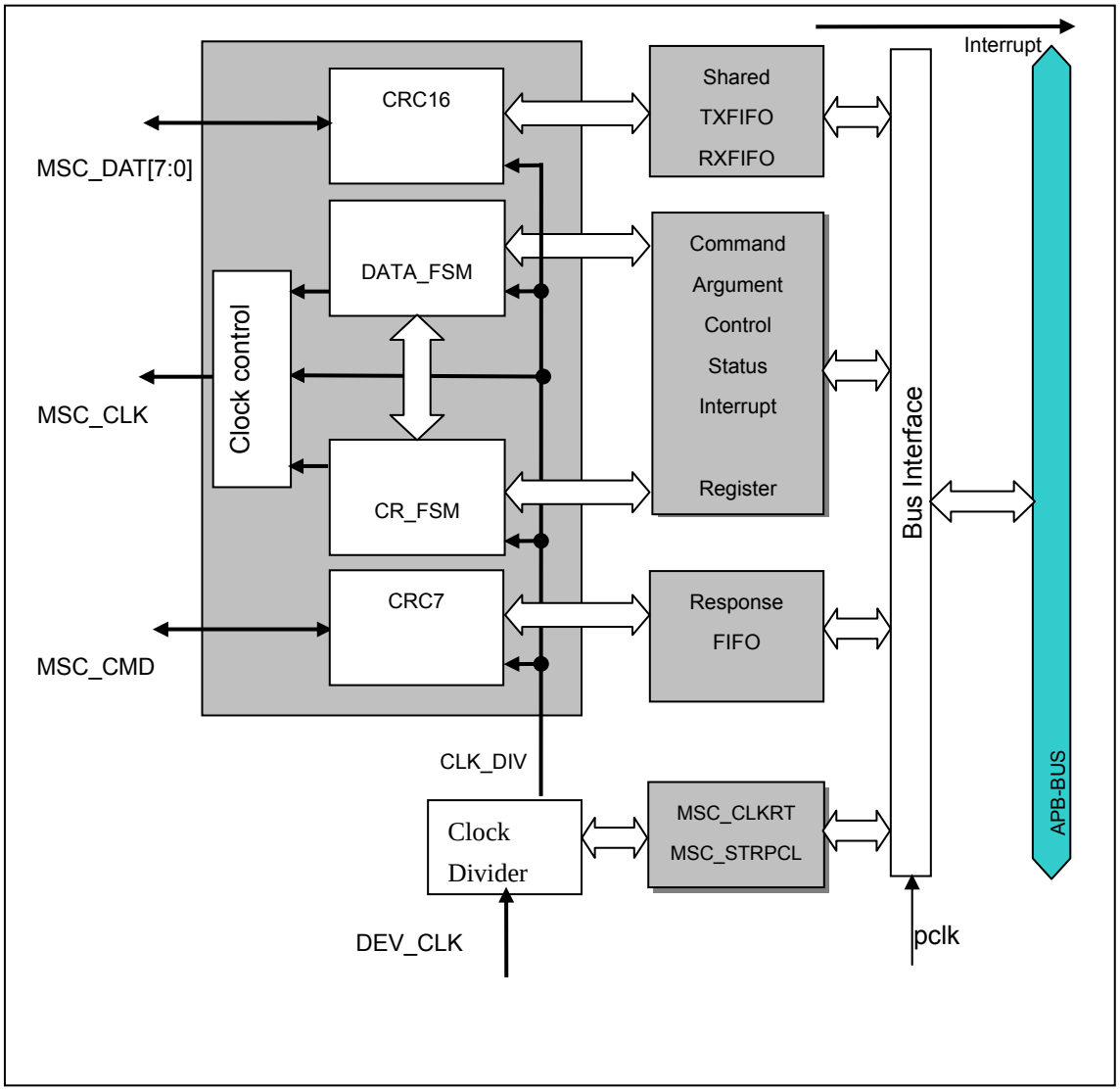


Figure 35-1 MMC/SD CE-ATA Controller Block Diagram

35.3 MMC/SD Controller Signal I/O Description

MSC and the card communication over the CMD and DATA line is base on command and data bit streams which are initiated by a start bit and terminated by a stop bit.

Command: a command is a token, which starts an operation. A command is sent from MSC either to a single card (addressed command) or to all connected cards (broadcast command). A command is transferred serially on the CMD line. Each command token is preceded by a start bit (‘0’) and succeeded by an end bit (‘1’). The total length is 48 bits and protected by CRC bits.

Table 35-1 Command Token Format

Bit position	**47**	**46**	**[45 : 40]**	**[39 : 8]**	**[7 : 1]**	**0**
Width (bits)	1	1	6	32	7	1
Value	0	1	X	X	x	1
Description	Start bit	Transmission bit	Command index	argument	CRC7	End bit

Response: a response is a token which is sent from an addressed card, or (synchronously) from all connected cards, to MSC as an answer to a previously received command. A response is transferred serially on the CMD line. Response tokens have varies coding schemes depending on their content.

Data: data can be transferred from the card to MSC or vice versa. Data is transferred via the data line. Data transfers to/from the SD Memory Card are done in blocks. Data blocks always succeeded by CRC bits. Single and multiple block operations are defined. Note that the Multiple Block operation mode is better for faster write operation. A multiple block transmission is terminated when a stop command follows on the CMD line. Data transfer can be configured by the MSC to use single or multiple data lines.

Table 35-2 MMC/SD Data Token Format

Description	**Start bit**	**Data**	**CRC16**	**End bit**
Stream Data	0	X	no CRC	1
Block Data	0	X	X	1

35.4 Register Description

The MMC-SD-CE_ATA controller is controlled by a set of registers that the application configures before every operation. The Table 35-3 lists all the MSC registers.

Table 35-3 MMC/SD Controller Registers Description

Name	RW	Reset Value	Address	Access Size
MSC_CTRL0	W	0x0000	0x10021000	16
MSC_STAT0	R	0x00000040	0x10021004	32
MSC_CLKRT0	RW	0x0000	0x10021008	16
MSC_CMDAT0	RW	0x00000000	0x1002100C	32
MSC_RESTO0	RW	0x40	0x10021010	16
MSC_RDTO0	RW	0xFFFF	0x10021014	32
MSC_BLKLEN0	RW	0x0000	0x10021018	16
MSC_NOB0	RW	0x0000	0x1002101C	16
MSC_SNOB0	R	0x????	0x10021020	16
MSC_IMASK0	RW	0x00FF	0x10021024	32
MSC_IREG0	RW	0x0000	0x10021028	16
MSC_CMD0	RW	0x00	0x1002102C	8
MSC_ARG0	RW	0x00000000	0x10021030	32
MSC_RES0	R	0x????	0x10021034	16
MSC_RXFIFO0	R	0x????????	0x10021038	32
MSC_TXFIFO0	W	0x????????	0x1002103C	32
MSC_LPM0	RW	0x00000000	0x10021040	32
MSC_CTRL1	W	0x0000	0x10022000	16
MSC_STAT1	R	0x00000040	0x10022004	32
MSC_CLKRT1	RW	0x0000	0x10022008	16
MSC_CMDAT1	RW	0x00000000	0x1002200C	32
MSC_RESTO1	RW	0x40	0x10022010	16
MSC_RDTO1	RW	0xFFFF	0x10022014	32
MSC_BLKLEN1	RW	0x0000	0x10022018	16
MSC_NOB1	RW	0x0000	0x1002201C	16
MSC_SNOB1	R	0x????	0x10022020	16
MSC_IMASK1	RW	0x00FF	0x10022024	32
MSC_IREG1	RW	0x0000	0x10022028	16
MSC_CMD1	RW	0x00	0x1002202C	8
MSC_ARG1	RW	0x00000000	0x10022030	32
MSC_RES1	R	0x????	0x10022034	16
MSC_RXFIFO1	R	0x????????	0x10022038	32
MSC_TXFIFO1	W	0x????????	0x1002203C	32
MSC_LPM1	RW	0x00000000	0x10022040	32

35.4.1 MMC/SD Control Register (MSC_CTRL)

MSC_CTRL0 0x10021000

MSC_CTRL1 0x10022000

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								SEND_CCSD	SEND_AD_CCSD	Reserved		EXIT_MULTIPLE	EXIT_TRANSFER	START_READWAIT	STOP_READWAIT	RESET	START_OP	STRPCL	
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15	SEND_CCSD	0: clear bit 1: Send Command Completion Signal Disable (CCSD) to CE_ATA device when set, host sends CCSD to CE_ATA device. Software set the bit only if current command is expecting CCS and interrupts are enabled in CE_ATA devices. Once the CCSD pattern is sent to device, host automatically clears the SEND_CCSD bit.	W
14	SEND_AS_CCSD	0: clear bit 1: send internally generated stop after sending CCSD to CE_ATA device When set, host automatically sends internally-generated STOP command(CMD12) to CE_ATA device. After sending CMD12, Auto Command Done (ACD) is set and generates interrupt to CPU. After sending the CCSD, controller automatically clears the SEND_AS_CCSD bit.	W
13:8	Reserved		R
7	EXIT_MULTIPLE	If CMD12 or CMD52 (I/O abort) is to be sent to terminate multiple block read/write in advance, set this bit to 1. 0: No effect 1: Exit from multiple block read/write	W
6	EXIT_TRANSFER	**Only used for SDIO suspend/resume and MMC stream read.** For SDIO, after suspend is accepted, set this bit with 1. For MMC, after the expected number of data are received, set this bit with 1. 0: No effect 1: Exit from multiple block read/write after suspend is accepted, or exit from stream read	W
5	START_READWAIT	Only used for SDIO ReadWait. Start the ReadWait cycle. 0: No effect	W

		1: Start ReadWait	
4	STOP_READWAIT	Only used for SDIO ReadWait. Stop the ReadWait cycle. 0: No effect 1: Start ReadWait	W
3	RESET	Resets the MMC/SD controller. 0: No effect 1: Reset the MMC/SD controller	W
2	START_OP	This bit is used to start the new operation. When starting the clock, this bit can be 1. When stopping the clock, this bit can only be 0. 0: Do nothing 1: Start the new operation	W
1:0	CLOCK_CONTROL	These bits are used to start or stop clock. 00: Do nothing 01: Stop MMC/SD clock 10: Start MMC/SD clock 11: Reserved	W

35.4.2 MSC Status Register (MSC_STAT)

MSC_STAT0 **0x10021004**

MSC_STAT1 **0x10022004**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	AUTO_CMD_DONE	RESERVED															IS_RESETTING	SDIO_INT_ACTIVE	PRG_DONE	DATA_TRAN_DONE	END_CMD_RES	DATA_FIFO_AFULL	IS_READWAIT	CLK_EN	DATA_FIFO_FULL	DATA_FIFO_EMPTY	CRC_RES_ERR	CRC_READ_ERROR	CRC_WRITE_ERROR		TIME_OUT_RES	TIME_OUTREAD
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

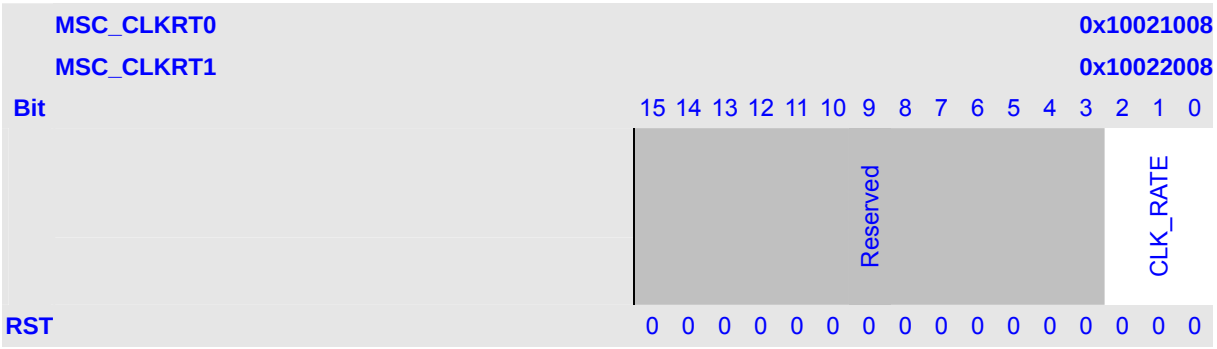
Bits	Name	Description	RW
31	AUTO_CMD_DONE	Indicate that the stop command (CMD12) that is internally generated by controller has finished.	R
30:16	Reserved		R
15	IS_RESETTING	MSC is resetting after power up or MSC_STRPCL[RESET] is written with 1. 0: Reset has been finished 1: Reset has not been finished	R
14	SDIO_INT_ACTIVE	Indicates whether an interrupt is detected at the SD I/O	R

		card. A separate acknowledge command to the card is required to clear this interrupt. 0: No interrupt detected 1: The interrupt from SDIO is detected	
13	PRG_DONE	Indicates whether card has finished programming. 0: Card has not finished programming and is busy 1: Card has finished programming and is not busy	R
12	DATA_TRAN_DONE	Indicates whether data transmission to card has completed. 0: Data transmission to card has not completed 1: Data transmission to card has completed	R
11	END_CMD_RES	End command-response sequence or command sequence. 0: Command and response/no-response sequence has not completed 1: Command and response/no-response sequence has completed	R
10	DATA_FIFO_AFULL	Indicates whether data FIFO is almost full (The number of words >= 15). For reading data from card, use this bit. 0: Data FIFO is not full 1: Data FIFO is full	R
9	IS_READWAIT	Indicates whether SDIO card has entered ReadWait State. 0: Card has not entered ReadWait 1: Card has entered ReadWait	R
8	CLK_EN	Clock enabled. 0: Clock is off 1: Clock is on	R
7	DATA_FIFO_FULL	Indicates whether data FIFO is full. For reading data from card, do not use this bit, because it almost keeps to be 0. 0: Data FIFO is not full 1: Data FIFO is full	R
6	DATA_FIFO_EMPTY	Indicates whether data FIFO is empty. 0: Data FIFO is not empty 1: Data FIFO is empty	R
5	CRC_RES_ERR	Response CRC error. 0: No error on the response CRC 1: CRC error occurred on the response	R
4	CRC_READ_ERROR	CRC read error. 0: No error on received data 1: CRC error occurred on received data	R
3:2	CRC_WRITE_ERROR	CRC write error. 00: No error on transmission of data	R

		01: Card observed erroneous transmission of data 10: No CRC status is sent back 11: Reserved	
1	TIME_OUT_RES	Response time out. 0: Card response has not timed out 1: Card response has time out	R
0	TIME_OUT_READ	Read time out. 0: Card read data has not timed out 1: Card read data has timed out	R

35.4.3 MSC Clock Rate Register (MSC_CLKRT)

The MSC_CLKRT register specifies the frequency division of the MMC/SD bus clock. The software is responsible for setting this register.



Bits	Name	Description	RW
15:3	Reserved		R
2:0	CLK_RATE	Clock rate. 000: CLK_SRC 001: 1/2 of CLK_SRC 010: 1/4 of CLK_SRC 011: 1/8 of CLK_SRC 100: 1/16 of CLK_SRC 101: 1/32 of CLK_SRC 110: 1/64 of CLK_SRC 111: 1/128 of CLK_SRC	WR

35.4.4 MMC/SD Command and Data Control Register (MSC_CMDAT)

MSC_CMDAT0 **0x1002100C**

MSC_CMDAT1 **0x1002200C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CCS_EXPECTED	READ_CEATA	Reserved												SDIO_PDRT	SEND_AS_STOP	RTRG		TTRG		IO_ABORT	BUS_WIDTH		DMA_EN	INIT	BUSY	STREAM_BLOCK	WRITE_READ	DATA_EN	RESPONSE_FORMAT		
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	CCS_EXPECTED	0: interrupts are not enabled in CE-ATA device, or commands does not expect CCS from device 1: interrupts are enabled in CE_ATA device, or RW_BLK command expects command completion signal from device If the command expects Command Completion Signal (CCS) from the device, the software should set the control bit. It is auto cleared 0 by hardware.	RW
30	READ_CEATA	0: host is not performing read access (RW_BLK or RW_REG) towards CE_ATA device 1: host id performing read access (RW_BLK or RW_REG) towards CE_ATA device Software should set the bit to indicate that CE_ATA device is being accessed for read transfer. The bit is used to disable read data timeout indication while performing CE_ATA read transfers. It is auto cleared 0 by hardware.	RW
29:18	Reserved		R
17	SDIO_PRDT	Determine whether SDIO interrupt is 2 cycle or extend more cycle when data block last is transferred. 0: more cycle (like single block) 1: exact 2 cycle	RW
16	SEND_AS_STOP	0: no stop command sent at end of data transfer 1: send stop command at end of data transfer when stop command has finished, it is auto cleared 0 by hardware.	RW
15:14	RTRG	These bits set the receive FIFO half-empty threshold value, when the number of transmit FIFO >= threshold	RW

		value , RXFIFO_RD_REQ will be set to 1. 00 : more than or equal to 8 01: more than or equal to 16 10: more than or equal to 24 11: reserved	
13:12	TTRG	These bits set the transmit FIFO half-empty threshold value, when the number of transmit FIFO < threshold value , TXFIFO_WR_REQ will be set to 1. 00 : less than 8 01: less than 16 10: less than 24 11: reserved	RW
11	STOP_ABORT	Specifies the current command is used to abort data transfer. 0: Nothing 1: The current command is used to abort transfer it is auto cleared 0 by hardware.	WR
10:9	BUS_WIDTH	Specifies the width of the data bus. 00: 1-bit 01: Reserved 10: 4-bit 11: 8bit	WR
8	DMA_EN	DMA mode enables. When DMA mode is used, this bit is also a mask on RXFIFO_RD_REQ and TXFIFO_WR_REQ interrupts. 0: Program I/O 1: DMA mode	WR
7	INIT	80 initialization clocks. 0: Do not precede command sequence with 80 clocks 1: Precede command sequence with 80 clocks	W
6	BUSY	Specifies whether a busy signal is expected after the current command. This bit is for no data command/response transactions only. 0: Not expect a busy signal 1: Expects a busy signal. If the response is R1b, then set it	WR
5	STREAM_BLOCK	Stream mode. 0: Data transfer of the current command sequence is not in stream mode 1: Data transfer of the current command sequence is in stream mode	WR
4	WRITE_READ	Specifies that the data transfer of the current command is a read or write operation. 0: Specifies that the data transfer of the current command	WR

		is a read operation 1: Specifies that the data transfer of the current command is a write operation	
3	DATA_EN	Specifies whether the current command includes a data transfer. It is also used to reset RX_FIFO and TX_FIFO. 0: No data transfer with current command 1: Has data transfer with current command. It is also used to reset RX_FIFO and TX_FIFO	WR
2:0	RESPONSE_FORMAT	These bit specify the response format for the current command. 000: No response 001: Format R1 and R1b 010: Format R2 011: Format R3 100: Format R4 101: Format R5 110: Format R6 111: Format R7	WR

35.4.5 MMC/SD Response Time Out Register (MSC_RESTO)

MSC_RESTO0 **0x10021010**

MSC_RESTO1 **0x10022010**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved								RES_TO							
RST	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bits	Name	Description	RW
15:8	Reserved		R
7:0	RES_TO	Specifies the number of MSC_CLK clock counts between the command and when the MMC/SD controller turns on the time-out error for the received response. The default value is 64.	WR

35.4.6 MMC/SD Read Time Out Register (MSC_RDTO)

MSC_RDTO0 **0x10021014**

MSC_RDTO1 **0x10022014**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	READ_TO																															
RST	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	Name	Description	RW
31:0	READ_TO	Specifies the number of clocks between the command and when the MMC/SD host controller turns on the time-out error for the received data. The unit is MSC_CLK.	WR

35.4.7 MMC/SD Block Length Register (MSC_BLKLEN)

MSC_BLKLEN0 **0x10021018**

MSC_BLKLEN1 **0x10022018**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								BLK_LEN								
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15:0	BLK_LEN	Specifies the number of bytes in a block, and is normally set to 0x200 for MMC/SD data transactions. The value Specified in the cards CSD.	WR

35.4.8 MSC/SD Number of Block Register (MSC_NOB)

MSC_NOB0 **0x1002101C**

MSC_NOB1 **0x1002201C**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								NOB								
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15:0	NOB	Specifies the number of blocks in a data transfer. One block is a possibility.	WR

35.4.9 MMC/SD Number of Successfully-transferred Blocks Register (MSC_SNOB)

In block mode, the MSC_SNOB register records the number of successfully transferred blocks. If the last block has CRC error, this register also summaries it. It is used to query blocks for multiple block transfer.

MSC_SNOB0 **0x10021020**
MSC_SNOB1 **0x10022020**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MSC_SNOB															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
15:0	MSC_SNOB	Specify the number of successfully transferred blocks for a multiple block transfer.	R

35.4.10 MMC/SD Interrupt Mask Register (MSC_IMASK)

MSC_IMASK0 **0x100210240**
MSC_IMASK1 **0x10022024**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	AUTO_CMD_DONE	DATA_FIFO_FULL	DATA_FIFO_EMP	CRC_RES_ERR	CRC_READ_ERR	CRC_WRITE_ERR	TIME_OUT_RES	TIME_OUT_READ	SDIO	TXFIFO_WR_REQ	RXFIFO_RD_REQ	Reserved	END_CMD_RES	PRG_DONE	DATA_TRAN_DONE
RST	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	Name	Description	RW
31:16	Reserved		R
15	AUTO_CMD_DONE	Mask the interrupt Auto Cmd Done (ACD). 0: Not masked 1: Masked	RW
14	DATA_FIFO_FULL	0: Not masked 1: Masked	RW

13	DATA_FIFO_EMP	0: Not masked 1: Masked	RW
12	CRC_RES_ERR	0: Not masked 1: Masked	RW
11	CRC_READ_ERR	0: Not masked 1: Masked	RW
10	CRC_WRITE_ERR	0: Not masked 1: Masked	RW
9	TIME_OUT_RES	0: Not masked 1: Masked	RW
8	TIME_OUT_READ	0: Not masked 1: Masked	RW
7	SDIO	Mask the interrupt from the SD I/O card. 0: Not masked 1: Masked	WR
6	TXFIFO_WR_REQ	Mask the Transmit FIFO write request interrupt. 0: Not masked 1: Masked	WR
5	RXFIFO_RD_REQ	Mask the Receive FIFO read request interrupt. 0: Not masked 1: Masked	WR
4:3	Reserved		R
2	END_CMD_RES	Mask the End command response interrupt. 0: Not masked 1: Masked	WR
1	PRG_DONE	Mask the Programming done interrupt. 0: Not masked 1: Masked	WR
0	DATA_TRAN_DONE	Mask the Data transfer done interrupt. 0: Not masked 1: Masked	WR

35.4.11 MMC/SD Interrupt Register (MSC_IREG)

The MSC_IREG register shows the currently requested interrupt. The FIFO request interrupts, TXFIFO_WR_REQ, and RXFIFO_RD_REQ are masked off with the DMA_EN bit in the MSC_CMDAT register. The software is responsible for monitoring these bit in program I/O mode.

MSC_IREG0 **0x10021028**

MSC_IREG1 **0x10022028**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved	AUTO_CMD_DONE	DATA_FIFO_FULL	DATA_FIFO_EMP	CRC_RES_ERR	CRC_READ_ERR	CRC_WRITE_ERR	TIME_OUT_RES	TIME_OUT_READ	SDIO	TXFIFO_WR_REQ	RXFIFO_RD_REQ	Reserved	END_CMD_RES	PRG_DONE	DATA_TRAN_DONE
RST		0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15	AUTO_CMD_DONE	indicate Auto Cmd Done (ACD) interrupt. 0: the interrupt is not detected 1: the interrupt is detected	RW
14	DATA_FIFO_FULL	Indicate data FIFO is full interrupt. 0: the interrupt is not detected 1: the interrupt is detected	R
13	DATA_FIFO_EMP	Indicate data FIFO is empty interrupt. 0: the interrupt is not detected 1: the interrupt is detected	R
12	CRC_RES_ERR	Indicate response CRC error interrupt. 0: the interrupt is not detected 1: the interrupt is detected	RW
11	CRC_READ_ERR	Indicate CRC read error interrupt. 0: the interrupt is not detected 1: the interrupt is detected	RW
10	CRC_WRITE_ERR	Indicate CRC write error interrupt. 0: the interrupt is not detected 1: the interrupt is detected	RW
9	TIME_OUT_RES	Indicate response time out interrupt. 0: the interrupt is not detected 1: the interrupt is detected	RW
8	TIME_OUT_READ	Indicate read time out interrupt. 0: the interrupt is not detected 1: the interrupt is detected	RW
7	SDIO	Indicates whether the interrupt from SDIO is detected.	R

		0: The interrupt from SDIO is not detected 1: The interrupt from SDIO is detected	
6	TXFIFO_WR_REQ	Transmit FIFO write request. Set if data FIFO becomes half empty. (the number of words is < 8) 0: No Request for data Write to MSC_TXFIFO 1: Request for data write to MSC_TXFIFO	R
5	RXFIFO_RD_REQ	Receive FIFO read request. Set if data FIFO becomes half full (the number of words is >= 8) or the entries in data FIFO are the last read data. 0: No Request for data read from MSC_RXFIFO 1: Request for data read from MSC_RXFIFO	R
4:3	Reserved		R
2	END_CMD_RES	Indicates whether the command/response sequence has been finished. 0: The command/response sequence has not been finished 1: The command/response sequence has been finished Write 1 to clear.	WR
1	PRG_DONE	Indicates whether card has finished programming. 0: Card has not finished programming and is busy 1: Card has finished programming and is no longer busy Write 1 to clear.	WR
0	DATA_TRAN_DONE	Indicates whether data transfer is done. Note that for stream read/write, only when CMD12 (STOP_TRANS) has been sent, is this bit set. 0: Data transfer is not complete 1: Data transfer has completed or an error has occurred Write 1 to clear.	WR

35.4.12 MMC/SD Command Index Register (MSC_CMD)

MSC_CMD0 **0x1002102C**

MSC_CMD1 **0x1002202C**

Bit	7	6	5	4	3	2	1	0
			Reserved	CMD_INDEX				
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:6	Reserved		R
5:0	CMD_INDEX	Specifies the command index to be executed.	WR

35.4.13 MMC/SD Command Argument Register (MSC_ARG)

MSC_ARG0																														0x10021030	
MSC_ARG1																														0x10022030	
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ARG																															
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:0	ARG	Specifies the argument for the current command.	WR

35.4.14 MMC/SD Response FIFO Register (MSC_RES)

The read-only MMC/SD Response FIFO register (RES_FIFO) holds the response sent back to the MMC/SD controller after every command. The size of this FIFO is 8 x 16-bit. The RES FIFO does not contain the 7-bit CRC for the response. The Status for CRC checking and response time-out status is in the status register, MSC_STAT.

The first halt-word read from the response FIFO is the most significant halt-word of the received response.

MSC_RES0																														0x10021034	
MSC_RES1																														0x10022034	
Bit																	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
																	DATA															
RST																	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
15:0	DATA	Contains the response to every command that is sent by the MMC/SD controller. The size of this FIFO register is 8 x 16-bit.	R

35.4.15 MMC/SD Receive Data FIFO Register (MSC_RXFIFO)

The MSC_RXFIFO is used to read the data from a card. It is read-only to the software, and is read on 32-bit boundary. The size of this FIFO is 16 x 32-bit.

MSC_RXFIFO0																														0x10021038	
MSC_RXFIFO1																														0x10022038	
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATA																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
31:0	DATA	One word of read data. The size of this FIFO is 16 x 32-bit.	R

35.4.16 MMC/SD Transmit Data FIFO Register (MSC_TXFIFO)

The MSC_TXFIFO is used to write the data to a card. It is write-only to the software, and is written on 32-bit boundary. The size of this FIFO is 16 x 32-bit.

MSC_TXFIFO0																														0x1002103C	
MSC_TXFIFO1																														0x1002203C	
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATA																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
31:0	DATA	One word of write data. The size of this FIFO is 16 x 32-bit.	W

35.4.17 MMC/SD Low Power Mode Register (MSC_LPM)

The MSC_LPM is used to control whether MSC controller enters Low-Power Mode.

MSC_LPM0																														0x10021040	
MSC_LPM1																														0x10022040	
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved																															LPM
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:1	Reserved		R
0	LPM	0 : Non –Low Power Mode 1: Low-Power Mode. Stop clock when card in idle (should be normally set to only MMC and SD cards. For SDIO cards, if interrupts must be detected, clock should not be stopped) When software sets the bit, MSC clock can auto be stopped. **NOTE:** when set the bit, the start_clock and stop clock can be not use.	RW

35.5 MMC/SD Functional Description

All communication between system and cards is controlled by the MSC. The MSC sends commands of two type: broadcast and addressed (point-to-point) commands.

Broadcast commands are intended for all cards, command like “Go_Idle_State”, “Send_Op_Cond”, “All_send_CID” and “Set_relative_Addr” are using way of broadcasting. During Broadcast mode, all cards are in open-drain mode, to avoid bus contention.

After Broadcast commands “Set_relative_Addr” issue, cards are enter standby mode, and Addressed command will be used from now on, in this mode, CMD/DAT will return to push-pull mode, to have maximum driving for maximum operation frequency.

The MMC and the SD are similar product. Besides the 4x bandwidth and the built-in encryption, they are being programmed similarly.

The MMC/SD controller (MSC) is the interface between the software and the MMC/SD bus. It is responsible for the timing and protocol between the software and the MMC/SD bus. It consists of control and status registers, a 16-bit response FIFO that is 8 entries deep, and one 32-bit receive/transmit data FIFOs that are 16 entries deep. The registers and FIFOs are accessible by the software.

MSC also enable minimal data latency by buffering data and generating and checking CRCs.

35.5.1 MSC Reset

The MMC/SD controller (MSC) can be reset by a hardware reset or software reset. All registers and FIFO controls are set to their default values after any reset.

35.5.2 MSC Card Reset

The command Go_Idle_State, CMD0 is the software reset command for MMC and SD Memory Card, and sets each card into Idle State regardless of the current card state; while in SDIO card, CMD52 is used to write IO reset in CCCR. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

35.5.3 Voltage Validation

All cards shall be able to establish communication with the host using any operation voltage in the maximal allowed voltage range specified in this standard. However, the support minimum and maximum values for Vdd are defined in Operation Conditions register (OCR) and many not cover the whole range. Cards that store the CID and CSD data in the payload memory would be able to communicate these information only under data transfer Vdd conditions. That means if host and card

have non compatible Vdd ranges, the card will not be able to complete the identification cycle, nor to send CSD data.

Therefore, a special command Send_Op_cont (CMD1 for MMC), SD_Send_Op_Cont (CMD41 for SD Memory) and IO_Send_Op_Cont (CMD5 for SDIO) are designed to provide a mechanism to identify and reject cards which do not match the Vdd range desired by the host. This is accomplished by the host sending the required Vdd voltage window as the operand of this command. Cards which can not perform data transfer in the specified range must discard themselves from further bus operations and go into Inactive State. By omitting the voltage range in the command, the host can query each card and determine the common voltage range before sending out-of-range cards into the Inactive State. This query should be used if the host is able to select a common voltage range or if a notification to the application of non usable cards in the stack is desired.

35.5.4 Card Registry

Card registry on MCC and SD card are different.

For SD card, Identification process start at clock rate Fod, while CMD line output drives are push-pull drivers instead of open-drain. After the bus is activated the host will request the cards to send their valid operation conditions. The response to ACMD41 is the operation condition register of the card. The same command shall be send to all of the new cards in the system. Incompatible cards are sent into Inactive State. The host then issue the command All_Send_CID (CMD2) to each card and get its unique card identification (CID) number. Card that is unidentified, that is, which is in Ready State, send its CID number as the response. After the CID was sent by the card it goes into Identification State. Thereafter, the host issues Send_Relative_Addr (CMD3) asks the card to publish a new relative card address (RCA), which is shorter that CID and which will be used to address the card in the future data transfer mode. Once the RCA is received the card state changes to the Stand-by State. At this point, if the host wants that the card will have another RCA number, it may ask the card to publish a new number by sending another Send_Relative_Addr command to the card. The last published RCA is the actual RCA of the card. The host repeats the identification process, that is, the cycles with CMD2 and CMD3 for each card in the system.

In MMC, the host starts the card identification process in open-drain mode with the identification clock rate Fod. The open drain driver stages on the CMD line allow parallel card operation during card identification. After the bus is active the host will request the cards to send their valid operation conditions (CMD1). The response to CMD1 is the ‘wired or’ operation on the condition restrictions of all cards in the system. Incompatible cards are sent into Inactive State. The host then issues the broadcast command All_Send_CID (CMD2), asking all cards for their unique card identification (CID) number. All unidentified cards, that is, those which are in Ready State, simultaneously start sending their CID numbers serially, while bit-wise monitoring their outgoing bitstream. Those cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods stop sending their CID immediately and must wait for the next identification cycle. Since CID is unique for each card, only one card can be successfully send its full CID to the host. This card then

goes into Identification State. Thereafter, the host issues Set_Relative_Addr (CMD3) to assign to this card a relative card address (RCA). Once the RCA is received the card state changes to the Stand-by State, and the card does not react to further identification cycles, and its output switches from open-drain to push-pull. The host repeat the process, which is CM2 and CMD3, until the host receive time-out condition to recognize completion of the identification process.

35.5.5 Card Access

35.5.5.1 Block Access, Block Write and Block Read

During block write (CMD24-27) one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block by the host. A card supporting block write shall always be able to accept a block of data defined by WRITE_BL_LEN. If the CRC fails, the card shall indicate the failure on the DAT line; the transferred data will be discarded and not written, and all further transmitted blocks (in multiple block write mode) will be ignored.

Programming of the CID and CSD registers does not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card will report an error and not change any register contents. Some cards may require long and unpredictable times to write a block of data. After receiving a block of data and completing the CRC check, the card will begin writing and hold the DAT line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host may poll the status of the card with a SEND_STATUS command (CMD13) at any time, and the card will respond with its status. The status bit READY_FOR_DATA indicates whether the card can accept new data or whether the write process is still in progress). The host may deselect the card by issuing CMD7 (to select a different card) which will displace the card into the Disconnect State and release the DAT line without interrupting the write operation. When reselecting the card, it will reactivate busy indication by pulling DAT to low if programming is still in progress and the write buffer is unavailable.

Block read is similar to stream read, except the basic unit of data transfer is a block whose maximizes is defined in the CSD (READ_BL_LEN). If READ_BL_PARTIAL is set, smaller blocks whose starting and ending address are entirely contained within one physical block (as defined by READ_BL_LEN) may also be transmitted. Unlike stream read, a CRC is appended to the end of each block ensuring data transfer integrity. CMD17 (READ_SINGLE_BLOCK) initiates a block read and after completing the transfer, the card returns to the Transfer state. CMD18 (READ_MULTIPLE_BLOCK) starts a transfer of several consecutive blocks. Blocks will be continuously transferred until a stop command is issued. If the host uses partial blocks whose accumulated length is not block aligned and block misalignment is not allowed, the card shall detect a block misalignment at the beginning of the first mis-aligned block, set the ADDRESS_ERROR error bit in the status register, abort transmission and wait in the Data State for a stop command.

35.5.5.2 Stream Access, Stream Write and Stream Read (MMC Only)

Stream write (CMD20) starts the data transfer from the host to the card beginning from the starting address until the host issues a stop command. Since the amount of data to be transferred is not determined in advance, CRC can not be used. If the end of the memory range is reached while sending data and no stop command has been sent by the host, all further transferred data is discarded.

There is a stream oriented data transfer controlled by READ_DAT_UNTIL_STOP (CMD11). This command instructs the card to send its payload, starting at a specified address, until the host sends a STOP_TRANSMISSION command (CMD12). The stop command has execution delay due to the serial command transmission. The data transfer stops after the end bit of the stop command. If the end of the memory range is reached while sending data and no stop command has been sent yet by the host, the contents of the further transferred payload is undefined.

35.5.5.3 Erase, Group Erase and Sector Erase (MMC Only)

It is desirable to erase many sectors simultaneously in order to enhance the data throughput. Identification of these sectors is accomplished with the TAG_* commands. Either an arbitrary set of sectors within a single erase group, or an arbitrary selection of erase groups may be erase at one time, but not both together. That is, the unit of measure for determining an erase is either a sector or an erase group. If a set of sectors must be erased, all selected sectors must lie within the same erase group. To facilitate selection, a first command with the starting address is followed by a second command with the final address, and all sectors (or groups) within this range will be selected for erase.

35.5.5.4 Wide Bus Selection/Deselection

Wide Bus (4 bit bus width) operation mode may be selected / deselected using ACMD6. The default bus width after power up or GO_IDLE (CMD0) is 1 bit bus width. ACMD6 command is valid in ‘trans state’ only. That means the bus width may be changed only after a card was selected (CMD7).

35.5.6 Protection Management

Three write protect methods are supported in the host for Cards, Card internal write protect (Card’s responsibility), Mechanical write protect switch (Host responsibility only) and Password protection card lock operation.

35.5.6.1 Card Internal Write Protection

Card data may be protected against either erase or write. The entire card may be permanently write protected by the manufacturer or content provider by setting the permanent or temporary write protect bits in the CSD. For cards which support write protection of groups of sectors by setting the WP_GRP_SIZE sectors as specified in the CSD), and the write protection may be changed by the application. The SET_WRITE_PROT command sets the write protection of the addressed

write-protect group, and the CLR_WRITE_PROT command clears the write protection of the addressed write-protect group.

The SEND_WRITE_PROT command is similar to a single block read command. The card shall send a data block containing 32 write protection bits (representing 32 write protect groups starting at the specified address) followed by 16 CRC bits. The address field in the write protect commands is a group address in byte units. The card will ignore all LSB's below the group size.

35.5.6.2 Mechanical write protect switch

A mechanical sliding tablet on the side of the card will be used by the user to indicate that a given card is write protected or not. If the sliding tablet is positioned in such a way that the window is open that means the card is write protected. If the window is close the card is not write protected.

A proper, matched, switch on the socket side will indicated to the host that the card is write protected or not. It is the responsibility of the host to protect the card. The position of the write protect switch is un-known to the internal circuitry of the card.

35.5.6.3 Password Protect

The password protection feature enables the host to lock a card while providing a password, which later will be used for unlocking the card. The password and its size is kept in an 128-bit PWD and 8-bit PWD_LEN registers, respectively. These registers are non-volatile so that a power cycle will not erase them.

Locked cards respond to (and execute) all commands in the basic command class (class 0) and "lock card" command class. Thus the host is allowed to reset, initialize, select, query for status, etc., but not to access data on the card. If the password was previously set (the value of PWD_LEN is not 0) will be locked automatically after power on. Similar to the existing CSD and CID register write commands the lock/unlock command is available in "trans_state" only. This means that it does not include an address argument and the card must be selected before using it. The card lock/unlock command has the structure and bus transaction type of a regular single block write command. The transferred data block includes all the required information of the command (password setting mode, PWD itself, card lock/unlock etc.). The following table describes the structure of the command data block.

Table 35-4 Command Data Block Structure

Byte #	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	Rsv	Rsv	Rsv	Rsv	ERASE	LOCK_UNLOCK	CLR_PWD	SET_PWD
1	PWDS_LEN							
2	Password Data							
…								

PWDS_LEN + 1	

ERASE – 1 Defines Forced Erase Operation (all other bits shall be 0) and only the command byte is sent.

LOCK/UNLOCK – 1=Locks the card. 0=Unlock the card (note that it is valid to set this bit together with SET_PWD but it is not allowed to set it together with CLR_PWD).

CLR_PWD – 1=Clears PWD.

SET_PWD – 1=Set new password to PWD.

PWD_LEN – Defines the following password length (in bytes).

PWD – The password (new or currently used depending on the command).

The data block size shall be defined by the host before it send the card lock/unlock command. This will allow different password sizes.

The following paragraphs define the various lock/unlock command sequences:

Lock command sequences:

1 Setting the Password.

a Select a card (CMD7), if not previously selected already.

b Define the block length (CMD16), given by the 8bit card lock/unlock mode, the 8 bits password size (in bytes), and the number of bytes of the new password. In case that a password replacement is done, then the block size shall consider that both passwords, the old and the new one, are sent with the command.

c Send Card Lock/Unlock command with the appropriate data block size on the data line including 16-bit CRC. The data block shall indicate the mode (SET_PWD), the length (PWD_LEN) and the password itself. In case that a password replacement is done, then the length value (PWD_LEN) shall include both passwords, the old and the new one, and the PWD field shall include the old password (currently used) followed by the new password.

d In case that the sent old password is not correct (not equal in size and content) then LOCK_UNLOCK_FAILED error bit will be set in the status register and the old password does not change. In case that PWD matches the sent old password then the given new password and its size will be saved in the PWD and PWD_LEN fields, respectively.

NOTE:

the password length register (PWD_LEN) indicates if a password is currently set. When it equals 0 there is no password set. If the value of PWD_LEN is not equal to zero the card will lock itself after power up. It is possible to lock the card immediately in the current power session by setting the LOCK/UNLOCK bit (while setting the password) or sending additional command for card lock.

2 Reset the password.

a Select a card (CMD7), if not previously selected already.

b Define the block length (CMD16), given by the 8-bit card lock/unlock mode, the 8-bit password size (in bytes), and the number of bytes of the currently used password.

c Send the card lock/unlock command with the appropriate data block size on the data line including 16-bit CRC. The data block shall indicate the mode CLR_PWD, the length (PWD_LEN) and the password (PWD) itself (LOCK/UNLOCK bit is don't care). If the PWD and PWD_LEN is set to 0. If the password is not correct then the LOCK_UNLOCK_FAILED error bit will be set in the status register.

3 Locking a card.

a Select a card (CMD7), if not previously selected already.

b Define the block length (CMD16), given by the 8-bit card lock/unlock mode, the 8-bit password size (in bytes), and the number of bytes of currently used password.

c Send the card lock/unlock command with the appropriate data block size on the data line including 16-bit CRC. The data block shall indicate the mode LOCK, the length (PWD_LEN) and the password (PWD) itself.

If the PWD content equals to the sent password then the card will be locked and the card-locked status bit will be set in the status register. If the password is not correct then LOCK_UNLOCK_FAILED error bit will be set in the status register.

NOTE:

it is possible to set the password and to lock the card in the same sequence. In such case the host shall perform all the required steps for setting the password (as described above) including the bit LOCK set while the new password command is sent. If the password was previously set (PWD_LEN is not 0), then the card will be locked automatically after power on reset. An attempt to lock a locked card or to lock a card that does not have a password will fail and the LOCK_UNLOCK_FAILED error bit will be set in the status register.

Unlock command sequences:

1 Unlocking the card.

a Select a card (CMD7), if not previously selected already.

b Define the block length (CMD16), given by the 8-bit card lock/unlock mode, the 8-bit password size (in bytes), and the number of bytes of the currently used password.

c Send the card lock/unlock command with the appropriate data block size on the data line including 16-bit CRC. The data block shall indicate the mode UNLOCK, the length (PWD_LEN) and the password (PWD) itself.

If the PWD content equals to the sent password then the card will be unlocked and the card-locked status bit will be cleared in the status register. If the password is not correct then the LOCK_UNLOCK_FAILED error bit will be set in the status register.

NOTE:

the unlocking is done only for the current power session. As long as the PWD is not

cleared the card will be locked automatically on the next power up. The only way to unlock the card is by clearing the password. An attempt to unlock an unlocked card will fail and LOCK_UNLOCK_FAILED error bit will be set in the status register.

2 Forcing Erase.

In case that the user forgot the password (the PWD content) it is possible to erase all the card data content along with the PWD content. This operation is called Forced Erase.

a Select a card (CMD7), if not previously selected already.

b Define the block length (CMD16) to 1 byte (8bit card lock/unlock command). Send the card lock/unlock command with the appropriate data block of one byte on the data line including 16-bit CRC. The data block shall indicate the mode ERASE (the ERASE bit shall be the only bit set).

If the ERASE bit is not the only bit in the data field then the LCOK_UNLOCK_FAILED error bit will be set in the status register and the erase request is rejected. If the command was accepted then ALL THE CARD CONTENT WILL BE ERASED including the PWD and PWD_LEN register content and the locked card will get unlocked.

An attempt to force erase on an unlocked card will fail and LOCK_UNLOCK_FAILED error bit will be set in the status register.

35.5.7 Card Status

The response format R1 contains a 32-bit field named card status. This field is intended to transmit the card's status information (which may be stored in a local status register) to the host. If not specified otherwise, the status entries are always related to the previous issued command.

Table below defines the different entries of the status. The type and clear condition fields in the table are abbreviate as follows:

Type:

E: Error bit.

S: Status bit..

R: Detected and set for the actual command response.

X: Detected and set during command execution. The host must poll the card by issuing the status command in order to read these bits.

Clear Condition:

A: According to the card current state.

B: Always related to the previous command. Reception of a valid command will clear it (with a delay of one command).

C: Clear by read.

Table 35-5 Card Status Description

Bits	Identifier	Type	Description	Clear Condition
31	OUT_OF_RAGE	E R	The command's argument was out of the allowed range for this card. 0: No Error 1: Error	C
30	ADDRESS_ERROR	E R X	A misaligned address which did not match the block length was used in the command. 0: No Error 1: Error	C
29	BLOCK_LEN_ERROR	E R	The transferred block length is not allowed for this, or the number of transferred bytes does not match the block length. 0: No Error 1: Error	C
28	ERASE_SEQ_ERROR	E R	An error in the sequence of erase commands occurred. 0: No Error 1: Error	C
27	ERASE_PARAM	E X	An invalid selection of sectors or groups for erase occurred. 0: No Error 1: Error	C
26	WP_VIOLATION	E R X	Attempt to program a write protected block. 0: No Protected 1: Protected	C
25	CARD_IS_LOCKED	S X	When set, signals that the card is locked by the host. 0: Card unlocked 1: Card locked	A
24	LOCK_UNLOCK_FAILED	E R X	Set when a sequence or password error has been detected in lock/unlock card command or if there was an attempt to access a locked card. 0: No Error 1: Error	C

23	COM_CRC_ERROR	E R	The CRC check of the previous command failed. 0: No Error 1: Error	B
22	ILLEGAL_COMMAND	E R	Command not legal for the card state. 0: No Error 1: Error	B
21	CARD_ECC_FAILED	E X	Card internal ECC was applied but failed to correct the data. 0: normal 1: failure	C
20	CC_ERROR	E R X	Internal card controller error. 0: No Error 1: Error	C
19	ERROR	E R X	A general or an unknown error occurred during the operation. 0: No Error 1: Error	C
18	UNDERRUN	E X	The card could not sustain data transfer in stream read mode. 0: No Error 1: Error	C
17	OVERRUN	E X	The card could not sustain data programming in stream write mode. 0: No Error 1: Error	C
16	CID/CSD_OVERWRITE	E R X	Can be either one of the following errors. 0: No Error 1: Error	C
15	WP_ERASE_SKIP	S X	Only partial address space was erased due to existing write protected blocks. 0: No Protected 1 : Protected	C
14	CARD_ECC_DISABLED	S X	The command has been executed without using the internal ECC. 0: enabled 1: disabled	A

13	ERASE_RESET	S R	An erase sequence was cleared before executing because an out of erase sequence command was received. 0: normal 1: set	C
12:9	CURRENT_STATE	S X	The state of the card when receiving the command. If the command execution causes a state change, it will be visible to the host in the response to the next command. The four bits are interpreted as binary coded number between 0 and 15. 0: idle 1: ready 2: ident 3: stby 4: tran 5: data 6: rcv 7: prg 8 : dis (9 – 15) : rsv	B
8	READY_FOR_DATA	S X	Corresponds to buffer empty signaling on the bus. 0: No Ready 1: Ready	A
7:6	Reserved	-	-	-
5	APP_CMD	S R	The card will expect ACMD, or indication that the command has been interpreted as ACMD. 0: Disable 1: Enable	C
4:0	Reserved	-	-	-

35.5.8 SD Status

The SD status contains status bits that are related to the SD card proprietary features and may be used for future application specific usage. The size of the SD status is one data block of 512bit. The content of this register is transmitted to the Host over the DAT bus along with 16-bit CRC. The SD status is sent to the host over the DAT bus if ACMD13 is sent (CMD55 followed with CMD13). ACMD13 can be sent to a card only in 'tran_state' (card is selected). SD status structure is described in below.

The same abbreviation for *type* and *clear condition* were used as for the Card Status above.

Table 35-6 SD Status Structure

Bits	Identifier	Type	Description	Clear Condition
511:510	DAT_BUS_WIDTH	S R	Shows the currently defined data bus width that was defined by SET_BUS_WIDTH command. 00: 1 (default) 01: Reserved 10: 4 bit width 11: Reserved	A
509	SECURED_MODE	S R	Card is in Secured Mode of operation. 0: Not in the Mode 10: In the mode	A
508:496	Reserved.			
495:480	SD_CARD_TYPE	S R	All 0, is SD Memory cards.	A
479:448	SIZE_OF_PROTECTED_AREA	S R	Size of protected area.	A
447:312	Reserved.			
311:0	Reserved for manufacturer.			

35.5.9 SDIO

I/O access differs from memory in that the registers can be written and read individually and directly without a FAT file structure or the concept of blocks (although block access is supported). These registers allow access to the IO data, control of the IO function, and report on status or transfer I/O data to and from the host.

Each SDIO card may have from 1 to 7 functions plus one memory function built into it. A function is a self contained I/O device. I/O functions may be identical or completely different from each other. All I/O functions are organized as a collection of registers, and there is a maximum of 131,072 registers possible for each I/O function.

35.5.9.1 SDIO Interrupts

In order to allow the SDIO card to interrupt the host, and interrupt function is added to a pin on the SD interface. Pin number 8 which is used as DAT[1] when operating in the 4 bit SD mode is used to signal the card's interrupt to the host. The use of interrupt is optional for each card or function within a card. The SDIO interrupt is "level sensitive", that is, the interrupt line must be held active (low) until it is either recognized and acted upon by the host or de-asserted due to the end of the Interrupt Period. Once the host has serviced the interrupt, it is cleared via an IO write to the appropriate bit in the CCCR.

The interrupt output of all SDIO cards is active low. This host controller provides pull-up resistors on all data lines DAT[3:0].

As Pin 8 of the card is shared between the IRQ and DAT[1] use in the 4 bit SD mode, and interrupt shall only be sent by the card and recognized by the host during a specific time. The time that a low on Pin 8 will be recognized as an interrupt is defined as the Interrupt Period.

The host here will only sample the level of Pin 8 (DAT[1]/IRQ) into the interrupt detector during the Interrupt Period. At all other times, the host will ignore the level on Pin 8. Note that the Interrupt Period is applicable for both memory and IO operations. The definition of the Interrupt Period is different for operations with single block and multiple block data transfer.

35.5.9.2 SDIO Suspend/Resume

Within a multi-function SDIO or a Combo (Mix IO and Memory) card, there are multiple devices (I/O and memory) that must share access to the SD bus. In order to allow the sharing of access to the host among multiple devices, SDIO and combo cards can implement the optional concept of suspend/resume. In a card supports suspend/resume, the host may temporarily halt a data transfer operation to one function or memory (suspend) in order to free the bus for a higher priority transfer to a different function of memory. Once this higher-priority transfer is complete, the original transfer is re-started where it left off (resume). The host controller here is supported by all IO functions except zero, and the memory of a combo card, and can suspend multiple transactions and resume them in any order desired. IO function zero does not support suspend/resume.

The procedure used to perform the Suspend/Resume operation on the SD bus is:

- The host determines which function currently used the DAT[] line(s).
- The host requests the lower priority or slower transaction to suspend.
- The host checks for the transaction suspension to complete.
- The host begins the higher priority transaction.
- The host waits for the completion of the higher priority transaction.
- The host restores the suspended transaction.

35.5.9.3 SDIO Read Wait

The optional Read Wait (RW) operation is defined only for the SD 1-bit and 4-bit modes. The read wait operation allows a host to signal a card that it is doing a read multiple (CMD53) operation to temporarily stall the data transfer while allowing the host to send commands to any function within the SDIO device. To determine if a card supports the Read Wait protocol, the host must test capability bits in CCCR. The timing for Read Wait is base on the Interrupt Period.

35.5.10 Clock Control

The software should guarantee that the card identification process starts in open-drain mode with the

clock rate fod (0 ~ 400khz). In addition, the software should also make the card into interrupt mode with fod (only for MMC). The commands that require fod are CMD0, CMD1, CMD2, CMD3, CMD5, CMD40 and ACMD41. In data transfer mode, the MSC controller can operate card with clock rate fpp (0 ~ 25Mhz).

35.5.11 Application Specified Command Handling

The MultiMediaCard/SD system is designed to provide a standard interface for a variety applications types. In this environment it is anticipate that there will be a need for specific customers/applications features. To enable a common way of implementing these features, two types of generic commands are defined in the standard: Application Specific Command, ACMD, and General Command, GEN_CMD.

GEN_CMD, this command, when received by the card, will cause the card to interpret the following command as an application specific command, ACMD. The ACMD has the same structure as of regular MultiMediaCard standard commands and it may have the same CMD number. The card will recognize it as ACMD by the fact that it appears after APP_CMD.

The only effect of the APP_CMD is that if the command index of the, immediately, following command has an ACMD overloading, the none standard version will used. If, as an example, a card has a definition for ACMD13 but not for ACMD7 then, if received immediately after APP_CMD command, Command 13 will be interpreted as the non standard ACMD13 but, command 7 as the standard CMD7.

In order to use one of the manufacturer specific ACMD's the host will:

1 Send APP_CMD. The response will have the APP_CMD bit (new status bit) set signaling to the host that ACMD is now expected.
2 Send the required ACMD. The response will have the APP_CMD bit set, indicating that the accepted command was interpreted as ACMD. If a non-ACMD is sent then it will be respected by the card as normal MultiMediaCard command and the APP_CMD bit in the Card Status stays clear.

If a non valid command is sent (neither ACMD nor CMD) then it will be handled as a standard MultiMediaCard illegal command error.

The bus transaction of the GEN_CMD is the same as the single block read or write commands (CMD24 or CMD17). The difference is that the argument denotes the direction of the data transfer (rather than the address) and the data block is not a memory payload data but has a vendor specific format and meaning.

The card shall be selected ('tran_state') before sending CMD56. The data block size is the BLOCK_LEN that was defined with CMD16. The response to CMD56 will be R1b (card status + busy indication).

35.6 MMC/SD Controller Operation

35.6.1 Data FIFOs

The controller FIFOs for the response tokens, received data, and transmitted data are MSC_RES, MSC_RXFIFO, and MSC_TXFIFO, respectively. These FIFOs are accessible by the software and are described in the following paragraphs.

35.6.1.1 Response FIFO (MSC_RES)

The response FIFO, MSC_RES, contains the response received from an MMC/SD card after a command is sent from the controller. MSC_RES is a read-only, 16-bit, and 8-entry deep FIFO.

The FIFO will hold all possible response lengths. Responses that are only one byte long are located on the LSB of the 16-bit entry in the FIFO. The first half-word read from the response FIFO is the most significant half-word of the received response. For example, if the response format is R1, then the response read from RES_FIFO is bit [47:32], bit[31:16], bit[15:0] and in the third half-word only the low 8-bit is effective response [15:8] and the high 8-bit is ignored. If the response format is R2, then the response read from MSC_RES is bit [135:8] and needs reading 8 times.

The FIFO does not contain the response CRC. The status of the CRC check is in the status register, MSC_STAT.

35.6.1.2 Receive/Transmit Data FIFO (MSC_RXFIFO/MSC_TXFIFO)

The receive data FIFO and transmit data FIFO share one 16-entry x 32-bit FIFO, because at one time data are only received or are only transmitted. If it is used to receive data, it is called MSC_RXFIFO and read-only. If it is used to transmit data, it is called MSC_TXFIFO and write-only.

Data FIFO and its controls are cleared to a starting state after a system reset or at the beginning of the operations which include data transfer. (MSC_CMDAT[DATA_EN] == 1)

If at any time MSC_RXFIFO becomes full and the data transmission is not complete, the controller turns the MSC_CLK off to prevent any overflows. When the clock is off, data transmission from the card stops until the clock is turned back on. After MSC_RXFIFO is not full, the controller turns the clock on to continue data transmission. The full status of the FIFO is registered in the MSC_STAT [DATA_FIFO_FULL] bit.
If at any time MSC_TXFIFO becomes empty and the data transmission is not complete, the controller turns the MSC_CLK off to prevent any underrun. When the clock is off, data transmission to the card stops until the clock is turned back on. When MSC_TXFIFO is no longer empty, the controller automatically restarts the clock. The empty status of the FIFO is registered in the MSC_STAT [DATA_FIFO_EMPTY] bit.

The FIFO is readable on word (32-bit) boundaries. The max read/written number is 16 words. The

controller can correctly process big-endian and little-endian data.

Because at the beginning of the operation which include data transfer (MSC_CMDAT [DATA_EN] == 1), Data FIFO and its controls are cleared, software should guarantee data in FIFO have been read/written before beginning a new command.

35.6.2 DMA and Program I/O

Software may communicate to the MMC controller via the DMA or program I/O.

To access MSC_RXFIFO/MSC_TXFIFO with the DMA, the software must program the DMA to read or write the FIFO with source port width 32-bit, destination port width 32-bit, transfer data size 32-byte, transfer mode single. For example, to write 64 bytes of data to the MSC_TXFIFO, the software must program the DMA as follows:

```
DMA_DCTRn = 2                // Write 2 32-bytes (64 bytes)
DMA_DCCRn[SWDH] = 0          // source port width is 32-bit
DMA_DCCRn[DWDH] = 0          // destination port width is 32-bit
DMA_DCCRn[DS] = 4            // transfer data size is 32-byte
DMA_DCCRn[TM] = 4            // transfer mode is single
DMA_DCCRn[RDIL] = 0          // request detection interval length is 0
```

The number of 32-bytes should be calculated from the number of transferred bytes as follows:

The number of words = (The number of bytes + 31) / 32

If the number of transferred bytes is not the multiple of 4, the controller can correctly process endian.

The DMA trigger level is 8 words, that is to say, the DMA read trigger is when data words in MSC_RXFIFO is >= 8 and the DMA write trigger is when data words in MSC_TXFIFO is < 8. Software can also configure DMA registers based on requirements, but the above 32-byte transfer data size is most efficient.

With program I/O, the software waits for the MSC_IREG [RXFIFO_RD_REQ] or MSC_IREG [TXFIFO_WR_REQ] interrupts before reading or writing the respective FIFO.

NOTES:

1. The MSC_CMDAT [DMA_EN] bit must be set to a 1 to enable communication with the DMA and it must be set to a 0 to enable program I/O.
2. DMA can be enabled only after MSC_CMDAT is written, because MSC_CMDAT [DATA_EN] is used to reset TX/RXFIFO.

35.6.3 Start and Stop clock

The software stops the clock as follows:

1 Write MSC_STRPCL with 0x01 to stop the MMC/SD bus clock.
2 Wait until MSC_STAT[CLK_EN] becomes zero.

To start the clock the software writes MSC_STRPCL with 0x02.

35.6.4 Software Reset

Reset includes the MSC reset and the card reset.

The MSC reset is through MSC_STRPCL [RESET] bit.

The card reset is to make the card into idle state. CMD0 (GO_IDLE_STATE) sets the MMC and SD memory cards into idle state. CMD52 (IO_RW_DIRECT, with argument 0x88000C08) reset the SD I/O card. The MMC/SD card are initialized with a default relative card address (RCA = 0x0001 for MMC and RCA = 0x0000 for SD) and with a default driver stage register setting (lowest speed, highest driving current capability).

The following registers must be set before the clock is started:

Step 1. Stop the clock.
Step 2. Set MSC_STRPCL register to 0x08 to reset MSC.
Step 3. Wait while MSC_STAT [IS_RESETTING] is 1.
Step 4. Set MSC_CMD with CMD0.
Step 5. Update the MSC_CMDAT register as follows:
  a Write 0x0000 to MSC_CMDAT [RESPONSE_FORMAT].
  b Clear the MSC_CMDAT [DATA_EN] bit.
  c Clear the MSC_CMDAT [BUSY] bit.
  d Clear the MSC_CMDAT [INIT] bit.
Step 6. Start the clock.
Step 7. Start the operation. (write MSC_STRPCL with 0x04)
Step 8. Wait for the END_CMD_RES interrupt.
Step 9. Set MSC_CMD with CMD52.
Step 10. Set MSC_ARG with 0x88000C08.
Step 11. Update the MSC_CMDAT register as follows:
  a Write 0x005 to MSC_CMDAT [RESPONSE_FORMAT].
  b Clear the MSC_CMDAT [DATA_EN] bit.
  c Clear the MSC_CMDAT [BUSY] bit.
  d Clear the MSC_CMDAT [INIT] bit.
Step 12. Start the operation.
Step 13. Wait for the END_CMD_RES interrupt.

35.6.5 Voltage Validation and Card Registry

At most 10 MMC and 1 SD (either SDMEM or SDIO) can be inserted MMC/SD bus at the same time, and their voltage validation and card registry steps are different, so the software should be

programmed as follows:

Step 1. Check whether SDIO card is inserted.

Step 2. Check whether SDMEM card is inserted.

Step 3. Check whether MMC cards are inserted.

35.6.5.1 Check SDIO

The commands are sent as follows:

Step 1. (Optional) Send CMD52 (IO_RW_DIRECT) with argument 0x88000C08 to reset SDIO card.

Step 2. Send CMD5 (IO_SEND_OP_CMD) to validate voltage.

Step 3. If the response is correct and the number of IO functions > 0, then continue, else go to check SDMEM.

Step 4. If C-bit in the response is ready (the initialization has finished), go to 6.

Step 5. Send CMD5 (IO_SEND_OP_CMD) to validate voltage, then go to 4.

Step 6. If memory-present-bit in the response is true, then it is a combo card (SDIO + Memory), else it is only a SDIO card.

Step 7. If it is a combo card, go to check SDMEM to initialize the memory part.

Step 8. Send CMD3 (SET_RELATIVE_ADDR) to let the card publish a RCA. The RCA is returned from the response.

Step 9. If do not accept the new RCA, go to 8, else record the new RCA.

Step 10. Go to check MMC, because we can assure that there is no SDMEM card.

35.6.5.2 Check SDMEM

If there is no SDIO card or there is a combo card, continue to check SDMEM.

The commands are sent as follows:

Step 1. (Optional) Send CMD0 (GO_IDLE_STATE) to reset MMC and SDMEM card. This command has no response.

Step 2. Send CMD55. Here the default RCA 0x0000 is used for CMD55.

Step 3. If the response is correct (CMD55 has response), then continue, else go to check MMC.

Step 4. Send ACMD41 (SD_SEND_OP_CMD) to validate voltage (the general OCR value is 0x00FF8000).

Step 5. If the initialization has finished, go to 7. (The response is the OCR register and it includes a status information bit (bit [31]). This status bit is set if the card power up procedure has been finished. As long as the card is busy, the corresponding bit[31] is set to LOW.)

Step 6. Send CMD55 and ACMD41 to validate voltage, and then go to 5.

Step 7. Send CMD2 (ALL_SEND_CID) to get the card CID.

Step 8. Send CMD3 (SET_RELATIVE_ADDR) to let card publish a RCA. The RCA is returned from the response.

Step 9. If do not accept the new RCA, go to 8, else record the new RCA.

Step 10. Go to check MMC.

35.6.5.3 Check MMC

Because there may be several MMC card, so some steps (5 ~ 8) should be repeated several times.

The commands are sent as follows:

Step 1. Send CMD1 (SEND_OP_CMD) to validate voltage (the general OCR value is 0x00FF88000).
Step 2. If the response is correct, then continue, else goto 9.
Step 3. If the initialization has finished, go to 5. (The response is the OCR register and it includes a status information bit (bit [31]). This status bit is set if the card power up procedure has been finished. As long as the card is busy, the corresponding bit[31] is set to LOW.)
Step 4. Send CMD1 (SEND_OP_CMD) to validate voltage, and then go to 3.
Step 5. Send CMD2 (ALL_SEND_CID) to get the card CID.
Step 6. If the response timeout occurs, goto 9.
Step 7. Send CMD3 (SET_RELATIVE_ADDR) to assign the card a RCA.
Step 8. If there are other MMC cards, then go to 5.
Step 9. Finish.

35.6.6 Single Data Block Write

In a single block write command, the following registers must be set before the operation is started:

Step 1. Set MSC_NOB register to 0x0001.
Step 2. Set MSC_BLKLEN to the number of bytes per block.
Step 3. Update the MSC_CMDAT register as follows:
  a Write 0x001 to MSC_CMDAT [RESPONSE_FORMAT].
  b Write 0x2 to MSC_CMDAT [BUS_WIDTH] if the card is SD, else clear it.
  c Set the MSC_CMDAT [DATA_EN] bit.
  d Set the MSC_CMDAT [WRITE_READ] bit.
  e Clear the MSC_CMDAT [STREAM_BLOCK] bit.
  f Clear the MSC_CMDAT [BUSY] bit.
  g Clear the MSC_CMDAT [INIT] bit.
Step 4. Start the operation.
Step 5. Write MSC_IMASK with some value to unmask the expected interrupts.

Then the software must perform the following steps:

Step 1. Wait for the MSC_IREG [END_CMD_RES] interrupt.
Step 2. Wait for the MSC_IREG [DATA_TRAN_DONE] interrupt.
At the same time write data to the MSC_TXFIFO and continue until all of the data have been written to the FIFO.
Step 3. Wait for MSC_IREG [PROG_DONE] interrupt. This interrupt indicates that the card has finished programming. Certainly software may start another command sequence on a different card.
Step 4. Read the MSC_STAT register to verify the status of the transaction (i.e. CRC error status).

To address a different card, the software sends a select command to that card by sending a basic no data command and response transaction. To address the same card, the software must wait for MSC_IREG [PROG_DONE] interrupt. This ensures that the card is not in the busy state.

In addition, CMD26 (PROGRAM_CID), CMD27 (PROGRAM_CSD), CMD42 (LOCK/UNLOCK), CMD56 (GEN_CMD: write) and CMD53 (single_block_write) operations are similar to single block write.

35.6.7 Single Block Read

In a single block read command, the following registers must be set before the operation is started:

Step 1. Set MSC_NOB register to 0x0001.
Step 2. Set MSC_BLKLEN register to the number of bytes per block.
Step 3. Update the following bits in the MSC_CMDAT register:
  a Write 0x001 to MSC_CMDAT [RESPONSE_FORMAT].
  b Write 0x2 to MSC_CMDAT [BUS_WIDTH] if the card is SD, else clear it.
  c Set the MSC_CMDAT [DATA_EN] bit.
  d Clear the MSC_CMDAT [WRITE_READ] bit.
  e Clear the MSC_CMDAT [STREAM_BLOCK] bit.
  f Clear the MSC_CMDAT [BUSY] bit.
  g Clear the MSC_CMDAT [INIT] bit.
Step 4. Start the operation.
Step 5. Write MSC_IMASK with some value to unmask the expected interrupts.

Then the software must perform the following steps:

Step 1. Wait for the MSC_IREG [END_CMD_RES] interrupt.
Step 2. Wait for the MSC_IREG [DATA_TRAN_DONE] interrupt.
  At the same time read data from the MSC_RXFIFO as data becomes available in the FIFO, and continue reading until all data is read from the FIFO.
Step 3. Read the MSC_STAT register to verify the status of the transaction (i.e. CRC error status).

In addition, CMD30 (SEND_WRITE_PROT), ACMD13 (SD_STATUS), CMD56 (GEN_CMD-read), ACMD51 (SEND_SCR) and CMD53 (single_block_read) are similar to single block read.

35.6.8 Multiple Block Write

The multiple block write mode is similar to the single block write mode, except that multiple blocks of data are transferred. Each block is the same length. All the registers are set as they are for the single block write, except that the MSC_NOB register is set to the number of blocks to be written.

The multiple block write mode also requires a stop transmission command, CMD12, after the data is transferred to the card. After the MSC_IREG [DATA_TRAN_DONE] interrupt occurs, the software must

program the controller register to send a stop data transmission command.

If multiple block write with pre-defined block count (refer to MMC spec v-3.3) is used, CMD12 should not be sent.

For SDIO card, CMD53 (multiple_block_write) is also similar, but when IO abort (CMD52) is sent, MSC_CMDAT [IO_ABORT] should be 1.

Table 35-7 How to stop multiple block write

Operation	Stop condition	Software processing
Open-ended or SDIO infinite	After write MSC_NOB blocks	1 Wait for DATA_TARN_DONE interrupt. 2 Send CMD12 or CMD52. (IO abort) 3 Wait for END_CMD_RES and PRG_DONE interrupt.
Open-ended or SDIO infinite	Stop writing in advance (not write MSC_NOB blocks)	1 Set MSC_STRPCL [EXIT_MULTIPLE]. 2 Wait for DATA_TRAN_DONE interrupt. 3 Send CMD12 or CMD52. (IO abort) 4 Wait for END_CMD_RES and PRG_DONE interrupt.
Predefined block or SDIO finite	After writing MSC_NOB blocks	1 Wait for DATA_TRAN_DONE interrupt.
Predefined block or SDIO finite	Stop writing in advance (not write MSC_NOB blocks)	1 Set MSC_STRPCL [EXIT_MULTIPLE]. 2 Wait for DATA_TRAN_DONE interrupt. 3 Send CMD12 or CMD52. (IO abort) 4 Wait for END_CMD_RES and PRG_DONE interrupt.

35.6.9 Multiple Block Read

The multiple blocks read mode is similar to the single block read mode, except that multiple blocks of data are transferred. Each block is the same length. All the registers are set as they are for the single block read, except that the MSC_NOB register is set to the number of blocks to be read.

The multiple blocks read mode requires a stop transmission command, CMD12, after the data from the card is received. After the MSC_IREG [DATA_TRAN_DONE] interrupt has occurred, the software must program the controller registers to send a stop data transmission command.

If multiple block read with pre-defined block count (refer to MMC spec v-3.3) is used, CMD12 should not be sent.

For SDIO card, CMD53 (multiple_block_read) is also similar, but when IO abort (CMD52) is sent, MSC_CMDAT [IO_ABORT] should be 1.

Table 35-8 How to stop multiple block read

Operation	Stop condition	Software processing
Open-ended or SDIO infinite	After reading MSC_NOB blocks	1 Wait for DATA_TRAN_DONE interrupt. 2 Send CMD12 or CMD52. (IO abort) 3 Wait for END_CMD_RES interrupt.
Open-ended or SDIO infinite	Stop reading in advance (not write MSC_NOB blocks)	1 Set MSC_STRPCL [EXIT_MULTIPLE]. 2 Wait for DATA_TRAN_DONE interrupt. 3 Send CMD12 or CMD52. (IO abort) 4 Wait for END_CMD_RES interrupt.
Predefined block or SDIO finite	After reading MSC_NOB blocks	1 Wait for DATA_TRAN_DONE interrupt.
Predefined block or SDIO finite	Stop reading in advance (not write MSC_NOB blocks)	1 Set MSC_STRPCL [EXIT_MULTIPLE]. 2 Wait for DATA_TRAN_DONE interrupt. 3 Send CMD12 or CMD52. (IO abort) 4 Wait for END_CMD_RES interrupt.

35.6.10 Stream Write (MMC)

In a stream write command, the following registers must be set before the operation is started:

1 Update MSC_CMDAT register as follows:
   a Write 0x001 to the MSC_CMDAT [RESPONSE_FORMAT].
   b Clear the MSC_CMDAT [BUS_WIDTH] because only MMC support stream write.
   c Set the MSC_CMDAT [DATA_EN] bit.
   d Set the MSC_CMDAT [WRITE_READ] bit.
   e Set the MSC_CMDAT [STREAM_BLOCK] bit.
   f Clear the MSC_CMDAT [BUSY] bit.
   g Clear the MSC_CMDAT [INIT] bit.
2 Start the operation.
3 Write MSC_IMASK with some value to unmask the expected interrupts.

Then the software must perform the following steps:

1 Wait for the MSC_IREG [END_CMD_RES] interrupt.
2 Write data to the MSC_TXFIFO and continue until all of the data is written to the Data FIFO.
3 Stop clock. Wait until MSC_STAT[CLK_EN] becomes 0. The clock must be stopped.
4 Set the command registers for a stop transaction command (CMD12) and other registers.
5 Start the clock and start the operation.
6 Wait for the MSC_IREG [END_CMD_ERS] interrupt.
7 Wait for the MSC_IREG [DATA_TRAN_DONE] interrupt.
8 Wait for the MSC_IREG [PRG_DONE] interrupt. This interrupt indicates that the card has finished programming. Certainly software may start another command sequence on a different card.

9 Read the MSC_STAT register to verify the status of the transaction.

To address a different card, the software must send a select command to that card by sending a basic no data command and response transaction. To address the same card, the software must wait for MSC_IREG [PRG_DONE] interrupt. This ensures that the card is not in the busy state.

If partial blocks are allowed (if CSD parameter WRITE_BL_PARTIAL is set) the data stream can start and stop at any address within the card address space, otherwise it shall start and stop only at block boundaries. If WRITE_BL_PARTIAL is not set, 16 more stuff bytes need to be written after the useful written data, otherwise only write the useful written data.

35.6.11 Stream Read (MMC)

In a stream read command, the following registers must be set before the operation is turned on:

1 Update the MSC_CMDAT register as follows:
   a Write 0x01 to the MSC_CMDAT [RESPONSE_FORMAT].
   b Clear the MSC_CMDAT [BUS_WIDTH] because only MMC support stream read.
   c Clear the MSC_CMDAT [WRITE_READ] bit.
   d Set the MSC_CMDAT [STREAM_BLOCK] bit.
   e Clear the MSC_CMDAT [BUSY] bit.
   f Clear the MSC_CMDAT [INIT] bit.
2 Start the operation.
3 Write MSC_IMASK with some value to unmask the expected interrupts.

Then the software must perform the following steps:

1 Wait for the MSC_IREG [END_CMD_RES] interrupt.
2 Read data from the MSC_RXFIFO and continue until all of the expected data has been read from the FIFO.
3 Write MSC_STRPCL [EXIT_TRANSER] with 1. If MSC_STAT[DATA_FIFO_FULL] is 1, then read MSC_RXFIFO to make it not full. Because if data FIFO is full, MSC_CLK is stopped. Here, the data FIFO contains useless data.
4 Set the command registers for a stop transaction command (CMD12) and send it. There is no need to stop the clock.
5 Wait for the MSC_IREG [END_CMD_RES].
6 Wait for the MSC_IREG [DATA_TRAN_DONE] interrupt.
7 Read the MSC_STAT register to verify the status of the transaction.

35.6.12 Erase, Select/Deselect and Stop

For CMD7 (SELECT/DESELECT_CARD), CMD12 (STOP_TRANSMISSION) and CMD38 (ERASE), the following registers must be set before the operation is started:

1 Update the MSC_CMDAT register as follows:
   a Write 0x01 to the MSC_CMDAT [RESPONSE_FORMAT].

b Clear the MSC_CMDAT [DATA_EN] bit.
c Clear the MSC_CMDAT [WRITE_READ] bit.
d Clear the MSC_CMDAT [STREAM_BLOCK] bit.
e Set the MSC_CMDAT [BUSY] bit.
f Clear the MSC_CMDAT [INIT] bit.

2 Start the operation.
3 Write MSC_IMASK with some value to unmask the expected interrupts.

Then the software must perform the following steps:

1 Wait for the MSC_IREG [END_CMD_RES] interrupt.
2 Wait for the MSC_IREG [PRG_DONE] interrupt. If CMD12 is sent to terminate data read operation, then there is no need to wait for MSC_IREG [PRG_DONE] interrupt. This interrupt indicates that the card has finished programming. Certainly software may start another command sequence on a different card.

35.6.13 SDIO Suspend/Resume

The actual suspend/resume steps are as follows:

1 During data transfer, send CMD52 to require suspend. BR and RAW flag should be 1.
2 If BS flag in the response is 0, then suspend has been accepted and goto 4.
3 Send CMD52 to query card status. R flag should be 1. Go to 2.
4 Write MSC_STRPCL [EXIT_TRANSFER] with 1.
5 Wait for the MSC_IREG [DATA_TRAN_DONE] interrupt.
6 Read MSC_NOB, MSC_SNOB and etc, save them into variables.
7 Set registers for high priority transfer and start it.
8 Wait until high priority transfer is finished.
9 Restore registers from variables, but MSC_NOB should be (MSC_NOB – MSC_SNOB).
10 Send CMD52 to require resume. FSx should be resumed function number.

35.6.14 SDIO ReadWait

The actual ReadWait steps are as follows:

1 During multiple block read, read MSC_SNOB. If MSC_SNOB is nearby or equal to MSC_NOB, no need to use ReadWait.
2 Write MSC_STRPCL [START_READWAIT] with 1.
3 Wait until MSC_STAT [IS_READWAIT] becomes 1.
4 Send CMD52 to query card status.
5 Write MSC_STRPCL [STOP_READWAIT] with 1.

35.6.15 Operation and Interrupt

The software can use polling-status method to operate the MMC/SD card, but this is not the proposed method, because its performance is very low. The proposed method is to use interrupt. Generally there

are fixed necessary steps to finish each command. The steps are as follows:

1 (Optional) Stop clock. Poll CLK_EN.
2 Fill the registers (MSC_CMD, MSC_CMDAT, MSC_ARG, MSC_CLKRT, and etc).
3 (Optional) Start clock.
4 Start the operation. Wait for the MSC_IREG [END_CMD_RES] interrupt.
5 Wait for the MSC_IREG [DATA_TRAN_DONE] interrupt.
6 Send STOP_TRANS (CMD12) or I/O abort (CMD52). Wait for the MSC_IREG [END_CMD_ERS] interrupt.
7 Wait for the MSC_IREG [DATA_TRAN_DONE] interrupt.
8 Wait for the MSC_IREG [PRG_DONE] interrupt.

Table 35-9 The mapping between Commands and Steps

Index	Abbreviation	1	2	3	4	5	6	7	8	Comments
CMD0	GO_IDLE_STATE	Y	Y	Y	Y					
CMD1	SEND_OP_COND	Y	Y	Y	Y					
CMD2	ALL_SEND_CID	Y	Y	Y	Y					
CMD3	SET_RELATIVE_ADDR	Y	Y	Y	Y					
CMD4	SET_DSR	Y	Y	Y	Y					
CMD7	SELECT/DSELECT_CARD	Y	Y	Y	Y				Y	
CMD9	SEND_CID	Y	Y	Y	Y					
CMD10	SEND_CSD	Y	Y	Y	Y					
CMD11	READ_DAT_UNTIL_STOP	Y	Y	Y	Y		Y	Y		
CMD12	STOP_TRANSMISSION	Y	Y	Y	Y				Y	
CMD13	SEND_STATUS	Y	Y	Y	Y					
CMD15	GO_INACTIVE_STATE	Y	Y	Y	Y					
CMD16	SET_BLOCKLEN	Y	Y	Y	Y					
CMD17	READ_SINGLE_BLOCK	Y	Y	Y	Y	Y				
CMD18	READ_MULTIPLE_BLOCK	Y	Y	Y	Y	Y	Y			Open-ended
CMD18	READ_MULTIPLE_BLOCK	Y	Y	Y	Y	Y				Predefine blocks
CMD20	WRITE_DAT_UNTIL_STOP	Y	Y	Y	Y		Y	Y	Y	
CMD23	SET_BLOCK_COUNT	Y	Y	Y	Y					
CMD24	WRITE_SINGLE_BLOCK	Y	Y	Y	Y	Y			Y	
CMD25	WRITE_MULTIPLE_BLOCK	Y	Y	Y	Y	Y	Y		Y	Open-ended
CMD25	WRITE_MULTIPLE_BLOCK	Y	Y	Y	Y	Y			Y	Predefine blocks
CMD26	PROGRAM_CID	Y	Y	Y	Y	Y			Y	
CMD27	PROGRAM_CSD	Y	Y	Y	Y	Y			Y	
CMD28	SET_WRITE_PROT	Y	Y	Y	Y				Y	
CMD29	CLR_WRITE_PROT	Y	Y	Y	Y				Y	
CMD30	SEND_WRITE_PROT	Y	Y	Y	Y	Y				
CMD32	ERASE_WR_BLOCK_START	Y	Y	Y	Y					
CMD33	ERASE_WR_BLOCK_END	Y	Y	Y	Y					

CMD35	ERASE_GROUP_START	Y	Y	Y	Y					
CMD36	ERASE_GROUP_END	Y	Y	Y	Y					
CMD38	ERASE	Y	Y	Y	Y				Y	
CMD39	FAST_IO	Y	Y	Y	Y					
CMD40	GO_IRQ_STATE	Y	Y	Y	Y					
CMD42	LOCK/UNLOCK	Y	Y	Y	Y	Y			Y	
CMD55	APP_CMD	Y	Y	Y	Y					
CMD56	GEN_CMD	Y	Y	Y	Y	Y				Read
CMD56	GEN_CMD	Y	Y	Y	Y	Y			Y	Write
ACMD6	SET_BUS_WIDTH	Y	Y	Y	Y					
ACMD13	SD_STATUS	Y	Y	Y	Y	Y				
ACMD22	SEND_NUM_WR_BLOCKS	Y	Y	Y	Y					
ACMD23	SET_WR_BLOCK_COUNT	Y	Y	Y	Y					
ACMD41	SD_SEND_OP_COND	Y	Y	Y	Y					
ACMD42	SET_CLR_CARD_DETECT	Y	Y	Y	Y					
ACMD51	SEND_SCR	Y	Y	Y	Y	Y				

NOTES:

1 For stream read/write, STOP_CMD is sent after finishing data transfer. For write, STOP_CMD is with the last six bytes. For read, STOP_CMD is sent after receiving data and card sends some data which MSC ignores.

36 UART Interface

36.1 Overview

This chapter describes the universal asynchronous receiver/transmitter (UART) serial ports. There are three UARTs: All UARTs use the same programming model. Each of the serial ports can operate in interrupt based mode or DMA-based mode.

The Universal asynchronous receiver/transmitter (UART) is compatible with the 16550-industry standard and can be used as slow infrared asynchronous interface that conforms to the Infrared Data Association (IrDA) serial infrared specification 1.1.

36.1.1 Features

- Full-duplex operation
- 5-, 6-, 7- or 8-bit characters with optional no parity or even or odd parity and with 1, 1½, or 2 stop bits
- 32x8 bit transmit FIFO and 32x11bit receive FIFO
- Independently controlled transmit, receive (data ready or timeout), line status interrupts
- Internal diagnostic capability Loopback control and break, parity, overrun and framing-error is provided
- Separate DMA requests for transmit and receive data services in FIFO mode
- Supports modem flow control by software or hardware
- Slow infrared asynchronous interface that conforms to IrDA specification

36.1.2 Pin Description

Table 36-1 UART Pins Description

Name	Type	Description
RxD	Input	Receive data input
TxD	Output	Transmit data output
CTS_	Input	Clear to Send — Modem Transmission enabled
RTS_	Output	Request to Send — UART Transmission request

NOTES:

1 UART2, UART0 support RxD, TxD, RTS_, CTS_, UART1 supports only RxD, TxD.

36.2 Register Descriptions

All UART register 32-bit access address is physical address. When ULCR.DLAB is 0, URBR, UTHR and UIER can be accessed; When ULCR.DLAB is 1, UDLLR and UDLHR can be accessed.

Table 36-2 UART Registers Description

Name	Description	RW	Reset Value	Address	Access Size
URBR0	UART Receive Buffer Register 0	R	0x??	0x10030000	8
UTHR0	UART Transmit Hold Register 0	W	0x??	0x10030000	8
UDLLR0	UART Divisor Latch Low Register 0	RW	0x00	0x10030000	8
UDLHR0	UART Divisor Latch High Register 0	RW	0x00	0x10030004	8
UIER0	UART Interrupt Enable Register 0	RW	0x00	0x10030004	8
UIIR0	UART Interrupt Identification Register 0	R	0x01	0x10030008	8
UFCR0	UART FIFO Control Register 0	W	0x00	0x10030008	8
ULCR0	UART Line Control Register 0	RW	0x00	0x1003000C	8
UMCR0	UART Modem Control Register 0	RW	0x00	0x10030010	8
ULSR0	UART Line Status Register 0	R	0x00	0x10030014	8
UMSR0	UART Modem Status Register 0	R	0x00	0x10030018	8
USPR0	UART Scratchpad Register 0	RW	0x00	0x1003001C	8
ISR0	Infrared Selection Register 0	RW	0x00	0x10030020	8
UMR0	UART M Register 0	RW	0x00	0x10030024	8
UACR0	UART Add Cycle Register 0	RW	0x00	0x10030028	16
URBR1	UART Receive Buffer Register 1	R	0x??	0x10031000	8
UTHR1	UART Transmit Hold Register 1	W	0x??	0x10031000	8
UDLLR1	UART Divisor Latch Low Register 1	RW	0x00	0x10031000	8
UDLHR1	UART Divisor Latch High Register 1	RW	0x00	0x10031004	8
UIER1	UART Interrupt Enable Register 1	RW	0x00	0x10031004	8
UIIR1	UART Interrupt Identification Register 1	R	0x01	0x10031008	8
UFCR1	UART FIFO Control Register 1	W	0x00	0x10031008	8
ULCR1	UART Line Control Register 1	RW	0x00	0x1003100C	8
UMCR1	UART Modem Control Register 1	RW	0x00	0x10031010	8
ULSR1	UART Line Status Register 1	R	0x00	0x10031014	8
UMSR1	UART Modem Status Register 1	R	0x00	0x10031018	8
USPR1	UART Scratchpad Register 1	RW	0x00	0x1003101C	8
ISR1	Infrared Selection Register 1	RW	0x00	0x10031020	8
UMR1	UART M Register 1	RW	0x00	0x10031024	8
UACR1	UART Add Cycle Register 1	RW	0x00	0x10031028	16
URBR2	UART Receive Buffer Register 2	R	0x??	0x10032000	8
UTHR2	UART Transmit Hold Register 2	W	0x??	0x10032000	8
UDLLR2	UART Divisor Latch Low Register 2	RW	0x00	0x10032000	8
UDLHR2	UART Divisor Latch High Register 2	RW	0x00	0x10032004	8

UIER2	UART Interrupt Enable Register 2	RW	0x00	0x10032004	8
UIIR2	UART Interrupt Identification Register 2	R	0x01	0x10032008	8
UFCR2	UART FIFO Control Register 2	W	0x00	0x10032008	8
ULCR2	UART Line Control Register 2	RW	0x00	0x1003200C	8
UMCR2	UART Modem Control Register 2	RW	0x00	0x10032010	8
ULSR2	UART Line Status Register 2	R	0x00	0x10032014	8
UMSR2	UART Modem Status Register 2	R	0x00	0x10032018	8
USPR2	UART Scratchpad Register 2	RW	0x00	0x1003201C	8
ISR2	Infrared Selection Register 2	RW	0x00	0x10032020	8
UMR2	UART M Register 2	RW	0x00	0x10032024	8
UACR2	UART Add Cycle Register 2	RW	0x00	0x10032028	16

36.2.1 UART Receive Buffer Register (URBR)

The read-only URBR is corresponded to one level 11bit buffer in non-FIFO mode and a 32x11bit FIFO that holds the character(s) received by the UART. Bits in URBR are right justified when being configured to use fewer than eight bits, and the rest of most significant data bits are zeroed and the most significant three bits of each buffer are the status for the character in the buffer. If ULSR.DRY is 0, don't read URBR, otherwise wrong operation may occur.

URBR0, URBR1, URBR2 — **0x10030000, 0x10031000, 0x10032000 (DLAB = 0)**

Bit	7	6	5	4	3	2	1	0
	URBR							
RST	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
7:0	URBR	8-bit UART receive read data.	R

36.2.2 UART Transmit Hold Register (UTHR)

The write-only UTHR is corresponded to one leve 8 bit buffer in non-FIFO mode and a 32x8bit FIFO in FIFO mode that holds the data byte(s) to be transmitted next.

UTHR0, UTHR1, UTHR2 — **0x10030000, 0x10031000, 0x10032000 (DLAB = 0)**

Bit	7	6	5	4	3	2	1	0
	UTHR							
RST	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
7:0	UTHR	8-bit UART transmit write hold data.	W

36.2.3 UART Divisor Latch Low/High Register (UDLLR / UDLHR)

UART Divisor Latch registers, UDLLR/UDLHR together compose the divisor for the programmable baud rate generator that can take the UART device clock and divide it by 1 to (2^{16} – 1).

The UART device source clock is EXCLK or EXCLK/2 that is determined by CPCCR.ECS. UDLHR/UDLLR stores the high/low 8-bit of the divisor respectively. Load these divisor latches during initialization to ensure that the baud rate generator operates properly. If both Divisor Latch registers are 0, the 16X clock stops.

If you don't set UMR and UACR, UART will work at normal mode with the specified frequency. The relationship between baud rate and the value of Divisor is shown by the formula when UMR and UACR are not set:

Baud Rate = (UART device clock) / (16 * **Divisor**)

UDLLR0, UDLLR1, UDLLR2 — **0x10030000, 0x10031000, 0x10032000 (DLAB = 1)**

Bit	7	6	5	4	3	2	1	0
	Divisor Latch Low 8-bit							
RST	0	0	0	0	0	0	0	0

UDLHR0, UDLHR1, UDLHR2									0x10030004, 0x10031004, 0x10032004 (DLAB = 1)
Bit	7	6	5	4	3	2	1	0	
	Divisor Latch High 8-bit								
RST	0	0	0	0	0	0	0	0	

36.2.4 UART Interrupt Enable Register (UIER)

The UART Interrupt Enable Register (UIER) contains the interrupt enable bits for the five types of interrupts (receive data ready, timeout, line status, and transmit data request, and modem status) that set a value in UIIR.

UIER0, UIER1, UIER2									0x10030004, 0x100301004, 0x10032004 (DLAB = 0)
Bit	7	6	5	4	3	2	1	0	
	Reserved			RTOIE	MSIE	RLSIE	TDRIE	RDRIE	
RST	0	0	0	0	0	0	0	0	

Bits	Name	Description	RW
7:5	Reserved	Always read 0, write is ignored.	R
4	RTOIE	**Receive Timeout Interrupt Enable.** 0: Disable the receive timeout interrupt 1: Enable the receive timeout interrupt Timeout means the URDR (FIFO mode) is not empty but no character has received for a period of time T: T (bits) = 4 X Word length + 12.	RW
3	MSIE	**Modem Status Interrupt Enable.** 0: Disable the modem status interrupt 1: Enable the modem status interrupt	RW
2	RLSIE	**Receive Line Status Interrupt Enable.** 0: Disable receive line status interrupt 1: Enable receive line status interrupt	RW
1	TDRIE	**Transmit Data Request Interrupt Enable.** 0: Disable the transmit data request interrupt 1: Enable the transmit data request interrupt	RW
0	RDRIE	**Receive Data Ready Interrupt Enable.** 0: Disable the receive data ready interrupt 1: Enable the receive data ready interrupt	RW

36.2.5 UART Interrupt Identification Register (UIIR)

The read-only UART Interrupt Identification Register (UIIR) records the prioritized pending interrupt source information. Its initial value after power-on reset is 0x01.

UIIR0, UIIR1, UIIR2 **0x10030008, 0x10031008, 0x10032008**

Bit	7	6	5	4	3	2	1	0
	FFMSEL		Reserved		INID			INPEND
RST	0	0	0	0	0	0	0	1

Bits	Name	Description	RW
7:6	FFMSEL	**FIFO Mode Select.** 0b00: Non-FIFO mode 0b01: Reserved 0b10: Reserved 0b11: FIFO mode	R
5:4	Reserved	Always read 0, write is ignored.	R
3:1	INID	**Interrupt Identifier.** These bits identify the current highest priority pending interrupt. INID | Description 0b000 | Modem Status 0b001 | Transmit Data Request 0b010 | Receive Data Ready 0b011 | Receive Line Status 0b100 | Reserved 0b101 | Reserved 0b110 | Receive Time Out 0b111 | Reserved See Table 36-3 for details.	R
0	INPEND	**Interrupt Pending.** 0: interrupt is pending 1: No interrupt pending	R

Table 36-3 UART Interrupt Identification Register Description

UIIR.INID	Interrupt Set/Clear Cause			
	Priority	Type	Source	Clear Condition
0b0001	—	None	No pending interrupt	—
0b0110	1st Highest	Receive Line Status	Overrun, Parity, Frame Error, Break Interrupt, and FIFO Error (DMA mode only)	Reading ULSR or empty all the error characters in DMA mode
0b0100	2nd Highest	Receive Data Ready	FIFO mode: Trigger threshold was reached Non-FIFO mode: URBR full	FIFO mode: Reading URBR till below trigger threshold. Non-FIFO mode: Empty URBR
0b1100	2nd Highest	Receive Timeout	FIFO mode only: URBR not empty but no data read in for a period of time	Reset receive buffer by setting UFCR.RFRT to 1 or Reading URBR
0b0010	3rd Highest	Transmit Data Request	FIFO mode: Empty location in UTHR equal to half or more than half Non-FIFO mode: UTHR empty	FIFO mode: Data number in UTHR more than half Non-FIFO mode: Writing UTHR
0b0000	4th Highest	Modem Status	Modem CTS_ pin status change	Reading UMSR

36.2.6 UART FIFO Control Register (UFCR)

The write-only register UFCR contains the control bits for receive and transmit FIFO.

UFCR0, UFCR1, UFCR2 **0x10030008, 0x10031008, 0x10032008**

Bit	7	6	5	4	3	2	1	0
	RDTR		Reserved	UME	DME	TFRT	RFRT	FME
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:6	RDTR	**Receive Buffer Data Number Trigger.** These bits are used to select the trigger level for the receive data ready interrupt in FIFO mode. 0b00: 1 0b01: 8 0b10: 16 0b11: 24	W

5	Reserved	Always read 0, write is ignored.	R
4	UME	**UART Module Enable.** 0: Disable UART 1: Enable UART	W
3	DME	**DMA Mode Enable.** 0: Disable DMA mode 1: Enable DMA mode	W
2	TFRT	**Transmit Holding Register Reset.** 0: Not reset 1: Reset transmit FIFO	W
1	RFRT	**Receive Buffer Reset.** 0: Not reset 1: Reset receive FIFO	W
0	FME	**FIFO Mode Enable.** Set this bit before the trigger levels. 0: non-FIFO mode 1: FIFO mode	W

36.2.7 UART Line Control Register (ULCR)

The ULCR defines the format for UART data transmission.

ULCR0. ULCR1, ULCR2, ULCR3 **0x1003000C, 0x1003100C, 0x1003200C**

Bit	7	6	5	4	3	2	1	0
	DLAB	SBK	STPAR	PARM	PARE	SBLS	WLS	
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7	DLAB	**Divisor Latch Access Bit.** 0: Enable to access URBR, UTHR or UIER 1: Enable to access UDLLR or UDLHR	W
6	SBK	**Set Break.** Causes a break condition (at least one 0x00 data) to be transmitted to the receiving UART. Acts only on the TXD pin and has no effect on the transmit logic. 0: No effect on TXD output 1: Forces TXD output to 0	W
5	STPAR	**Sticky Parity.** Setting this bit forces parity location to be opposite of PARM bit when PARE is 1 (it is ignored when PARE is 0).	W

		0: Disable Sticky parity 1: Enable Sticky parity (opposite o f PARM bit)	
4	PARM	**Parity Odd/Even Mode Select.** If PARE = 0, PARM is ignored. 0: Odd parity 1: Even parity	W
3	PARE	**Parity Enable.** Enables a parity bit to be generated on transmission or checked on reception. 0: No parity 1: Parity	W
2	SBLS	**Stop Bit Length Select.** Specifies the number of stop bits transmitted and received in each character. When receiving, the receiver checks only the first stop bit. 0: 1 stop bit 1: 2 stop bits, except for 5-bit character then 1-1/2 bits	W
1:0	WLS	**Word Length Select.** 0b00: 5-bit character 0b01: 6-bit character 0b10: 7-bit character 0b11: 8-bit character	W

36.2.8 UART Line Status Register (ULSR)

The read-only ULSR indicates status information during the data transfer. Receive error information in ULSR[4:1] remains set until software reads ULSR and it must be read before the error character is read.

ULSR0, ULSR1, ULSR2 **0x10030014, 0x10031014, 0x10032014**

Bit	7	6	5	4	3	2	1	0
	FIFOE	TEMP	TDRQ	BI	FMER	PARER	OVER	DRY
RST	0	1	1	0	0	0	0	0

Bits	Name	Description	RW
7	FIFOE	**FIFO Error Status. (FIFO mode only)** FIFOE is set when there is at least one kind of receive error (parity, frame, overrun, break) for any of the characters in receive buffer. FIFOE is reset when all error characters are read out of the buffer. During DMA transfer, the error interrupt generates when FIFOE is 1, and	R

		no receive DMA request generates even when data in receive buffer reaches the trigger threshold until all the error characters are read out. In non-DMA mode, FIFOE set does not generate error interrupt. 0: No error data in receive buffer or non-FIFO mode 1: One or more error character in receive buffer	
6	TEMP	**Transmit Holding Register Empty.** Set when both UTHR and shift register are empty. It is cleared when either the UTHR or the shift register contains a data character. 0: There is data in the transmit shifter and UTHR 1: All the data in the transmit shifter and UTHR has been shifted out	R
5	TDRQ	**Transmit Data Request.** Set when UTHR has half or more empty location (FIFO mode) or empty (non-FIFO mode). When both UIER.TDRIE and TDRQ are 1, transmit data request interrupt generates or during DMA transfer, DMA request to the DMA controller generates when UIER.TDRIE is 0 and TDRQ is 1. 0: There is one (non-FIFO mode) or more than half data (FIFO mode) in UTHR 1: None data (non-FIFO mode) or half or less than half data (FIFO mode) in UTHR	R
4	BI	**Break Interrupt.** BI is set when the received data input is held low for longer than a full-word transmission time (the total time of start bit + data bits + parity bit + stop bits). BI is cleared when the processor reads the ULSR. In FIFO mode, only one character equal to 0x00 is loaded into the FIFO regardless of the length of the break condition. BI shows the break condition for the character at the front of the FIFO, not the most recently received character. 0: No break signal has been received 1: Break signal received	R
3	FMER	**Framing Error.** Set when the bit following the last data bit or parity bit is detected to be 0. If the ULCR had been set for two or one and half stop bits, the other stop bits are not checked except the first one. In FIFO mode, FMER shows a framing error for the character at the front of the receive buffer, not for the most recently received character. Cleared when the processor reads the ULSR. 0: No framing error 1: Invalid stop bit has been detected	R
2	PARER	**Parity Error.**	R

		Indicates that the received data character does not have the correct even or odd parity, as selected by the even parity select bit. PARER is set upon detection if a parity error and is cleared when the processor reads the ULSR. In FIFO mode, PARER shows a parity error for the character at the front of the FIFO, not the most recently received character. 0: No parity error 1: Parity error has occurred	
1	OVER	**Overrun Error.** Set when both receive buffer and shifter are full and new data is received which will be lost. Cleared when the processor reads the ULSR. 0: No data has been lost 1: Receive data has been lost	R
0	DRY	**Data Ready.** Set when a complete incoming character has been received into the Receive Buffer registers. DRY is cleared when the receive buffer is read (non-FIFO mode) or when the buffer is empty or when the buffer is reset by setting UFCR.RFRT to 1. 0: No data has been received 1: Data is available in URBR	R

36.2.9 UART Modem Control Register (UMCR)

The UMCR uses the modem control pins RTS_ and CTS_ to control the interface with a modem or data set. UMCR also controls the loopback mode. Loopback mode must be enabled before the UART is enabled.

UMCR0, UMCR1, UMCR2 **0x10030010, 0x10031010, 0x10032010**

Bit	7	6	5	4	3	2	1	0
	MDCE	FCM	Reserved	LOOP	Reserved	Reserved	RTS	Reserved
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7	MDCE	**Modem Control Enable.** 0: Modem function is disabled 1: Modem function is enabled	W
6	FCM	**Flow Control Mode.** 0: Flow control by software 1: Flow control by hardware	
5	Reserved	Always read 0, write is ignored.	R

4	LOOP	**Loop Back.** This bit is used for diagnostic testing of the UART. When LOOP is 1, TXD output pin is set to a logic 1 state, RXD is disconnected from the pin, and the output of the transmitter shifter register is looped back into the receiver shift register input internally, similar to CTS_ and RTS_ pins and the RTS bit of the UMCR is connected to CTS bit of UMSR respectively. Loopback mode must be selected before the UART is enabled. 0: Normal operation mode 1: Loopback-mode UART operation	W
3	Reserved	Always read 0, write is ignored.	R
2	Reserved	Always read 0, write is ignored.	R
1	RTS	**Request To Send.** This bit can control the RTS_ output state. 0: RTS_ force to high 1: RTS_ force to low	W
0	Reserved	Always read 0, write is ignored.	R

36.2.10 UART Modem Status Register (UMSR)

The read-only UMSR provides the current state of the control lines from the modem to the processor. They are cleared when the processor reads UMSR.

UMSR0, UMSR1, UMSR2 **0x10030018, 0x10031018, 0x10032018**

Bit	7	6	5	4	3	2	1	0
	Reserved			CTS	Reserved			CCTS
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7	Reserved	Always read 0, write is ignored.	R
6	Reserved	Always read 0, write is ignored.	R
5	Reserved	Always read 0, write is ignored.	R
4	CTS	**Status of Clear To Send.** When MDCE bit is 1, this bit is the complement of CTS_ input. If Loop bit of UMCR is 1, this bit is equivalent to RTS bit of UMCR. 0: CTS_ pin is 1 1: CTS_ pin is 0	R
3	Reserved	Always read 0, write is ignored.	R
2	Reserved	Always read 0, write is ignored.	R
1	Reserved	Always read 0, write is ignored.	R

0	CCTS	**Change status of CTS_.** When MDCE bit is 1, this bit indicates the state change on CTS_ pin. 0: No state change on CTS_ pin since last read of UMSR 1: A change occurs on the state of CTS_ pin	R

36.2.11 UART Scratchpad Register

This Scratchpad register is used as a scratch register for the programmer and has no effect on the UART.

USPR **0x1003001C**

Bit	7	6	5	4	3	2	1	0
	Scratch Data							
RST	?	?	?	?	?	?	?	?

36.2.12 Infrared Selection Register (ISR)

The ISR is used to configure the slow-infrared (SIR) interface that is provided in each UART to support two-way wireless communication using infrared transmission that conforms to the IrDA serial infrared specification 1.1. The maximum frequency is up to 115.2kbps.

ISR **0x10030020**

Bit	7	6	5	4	3	2	1	0
	Reserved			RDPL	TDPL	XMODE	RCVEIR	XMITIR
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:5	Reserved	Always read 0, write is ignored.	R
4	RDPL	**Receive Data Polarity.** 0: Slow-infrared (SIR) interface decoder takes positive pulses as zeros 1: SIR decoder takes negative pulses as zeros	W
3	TDPL	**Transmit Data Polarity.** 0: SIR encoder generates a positive pulse for a data bit of zero 1: SIR encoder generates a negative pulse for a data bit of zero	W
2	XMODE	**Transmit Pulse Width Mode.** Set when the transmit encoder needs to generate 1.6us pulses (that are 3/16 of a bit-time at 115.2 kbps). Cleared when the transmit encoder needs to generate 3/16 of a bit-time	W

		wide according to current baud rate. 0: Transmit pulse width is 3/16 of a bit-time wide 1: Transmit pulse width is 1.6 us	
1	RCVEIR	**Receiver SIR Enable.** This bit is used to select the signal from the RXD pin is processed by the IrDA decoder before it is fed to the UART (RCVEIR = 1) or bypass IrDA decoder and is fed directly to the UART (RCVEIR = 0). 0: Receiver is in UART mode 1: Receiver is in SIR mode	W
0	XMITIR	**Transmitter SIR Enable.** This bit is used to select TXD output pin is processed by the IrDA encoder before it is fed to the device pin (XMITIR = 1) or bypass IrDA encoder and is fed directly to the device pin (XMITIR = 0). **NOTE:** disable infrared LED before XMITIR is set, otherwise a false start bit may occur. 0: Transmitter is in UART mode 1: Transmitter is in SIR mode	W

36.2.13 UART M Register (UMR)

UMSR0, UMSR1, UMSR2 **0x10030024, 0x10031024, 0x10032024**

Bit	7	6	5	4	3	2	1	0
	Reserved			M				
RST	0	0	0	0	0	0	0	0

M is the value of UMR register.
It will take UART at least M cycles to transmit one bit and receiver to receive one bit.
It will take UART at most M+1 cycles for transmitter to transmit one bit and receiver to receive one bit.

36.2.14 UART Add Cycle Register (UACR)

UACR0, UACR1, UACR2 **0x10030028, 0x10031028, 0x10032028**

Bit	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved				AC							
RST	0	0	0	0	0	0	0	0	0	0	0	0

If nth bit of the register is 1, it will take UART M+1 cycles to transmit or receive the bit of date for transmit or receive.
If the register is 12'h0, UART will receive or transmit a bit by M cycle.
If the register is 12'hfff, UART will receive or transmit a bit by M+1 cycle.
For the detail to see For any frequency clock to use the Uart.

36.3 Operation

The following sections describe the UART operations that include flow of configuration, data transmission, data reception, and Infrared mode.

36.3.1 UART Configuration

Before UART starts to transfer data or changing transfer format, configuration must be done to define the transfer format. The sample flow is as the following:

In FIFO mode, set FME bit of UFCR to 1, reset receive and transmit FIFO, then initialize the UART as described below:

1 Clear UFCR.UME to 0.
2 Set value in UDLL/UDHR to generate the baud rate clock.
3 Set data format in ULCR.
4 If it is in FIFO MODE, set FME bit and other FIFO control in UFCR, reset receive and transmit FIFO, otherwise skip item 4.
5 Set each interrupt enable bit in UIER in interrupt-based transfer or set UFCR.DME in DMA-based transfer (DMA transfer is FIFO mode only), then set UFCR.UME.

36.3.2 Data Transmission

After configuration, UART is ready for data transfer. For data transmission, refer to the following procedure:

1 Read ULSR.TDRQ (interrupt disable) or wait for transmit data request interrupt (interrupt enable), if TDRQ = 1 or transmit data request interrupt generates, that means there is enough empty location in UTHR for new data.
2 If ULSR.TDRQ is 1 or get the transmit data request interrupt, write transmit data to UTHR to start transmission.
3 Do item 1 and item 2 if there are more data waiting for transmit.
4 After all necessary data are written to UTHR, wait ULSR.TEMP = 1, that means all data completely transmitted.
5 If it is necessary to send break, set ULCR.SBK and at least wait for 1-bit interval time to send a valid break, then clear ULCR.SBK.
6 Clear UME bit to finish UART transmission.

36.3.3 Data Reception

After configuration, UART is ready for data transfer. For data reception, refer to the following sample procedure:

1 Read ULSR.DRY (interrupt disable) or wait for receive data request interrupt (interrupt enable), if ULSR.DRY =1 or receive data request interrupt generates, that means URBR has one data (non-FIFO mode) or data in URBR reaches the trigger value. (FIFO mode)
2 If ULSR.DRY = 1 or receive data request interrupt generates, then read ULSR.FIFOE or see if

there is error interrupt, if FIFOE = 1, it means received data has receive error, then go to error handler, other wise go to item 3.

3 Read one received data in URBR (non-FIFO mode) or data equal to trigger value in URBR. (FIFO mode)

4 Check whether all data received: check whether ULSR.DRY = 0, in FIFO mode and interrupt is enabled, timeout interrupt may generate, when timeout interrupt generates, read URBR till ULSR.DRY = 0.

5 Clear UFCR.UME to end data reception when all data are received and ULSR.DRY = 0.

36.3.4 Receive Error Handling

A sample error handling flow is as the following:

1 If ULSR.FIFOE = 1, it means there is receive error in received data, then check what error it is.

2 If ULSR.OVER = 1, go to OVER error handling.

3 If ULSR.BI = 1, go to Break handling.

4 If ULSR.FMER = 1, go to Frame error handling.

5 If PARER = 1, go to PARER error handling.

36.3.5 Modem Transfer

When UMCR.MDCE = 1, modem control is enabled. Transfer flow can be stopped and restarted by software through RTS_ and CTS_ pin. When UART transmitter detects low level on CTS_ pin, it stops transmission and TxD pin goes to mark state after finishing transmitting the current character until it detects CTS_ pin goes back to high level. RTS_ pin is output to receiving UART and its state can be controlled by setting UMCR.RTS bit, that is, setting UMCR.RTS to 1, RTS_ pin is low level output that means UART is ready to receive data, on the contrary, it means UART currently can't receive more data.

36.3.6 DMA Transfer

UART can operate in DMA-based (UFCR.DME = 1, FIFO mode only), that is, dma request initiated by UART takes the place of interrupt request and transmission/reception is carried out using DMA instead of CPU. Be sure that software guarantee to disable transmit and receive interrupt except timeout and error interrupts.

During DMA transfer, if an interrupt occurs, software must first read the ULSR to see if an error interrupt exists, then check the UIIR for the source of the interrupt and if DMA channel is already halt because of the error indicator from UART, then disable DMA channel and read out all the error data from receive FIFO. Software re-set and re-enable DMA and data transfer by DMA will re-start.

36.3.7 Slow IrDA Asynchronous Interface

Each UART supports slow infra-red (SIR) transmission and reception by setting ISR.XMITIR and ISR.RCVEIR to 1 (make sure the two bits are not set to 1 at the same time because SIR can't operate full-duplex). According to the IrDA 1.1, data rate is limited at a maximum value of 115.2Kbps.

In SIR transmit mode, the transmit pulse comes out at a rate of 3/16 (when the transmit data bit is zero); in SIR receive mode, the receiver must detect the 3/16 pulsed period to recognize a zero value (an active high or low pulse is demodulation to 0, and no pulse is demodulation to 1).

Compared to normal UART, there are some limitations to SIR, that is, each character is fixed to 8-bit data width, no parity and 1 stop bit and modem function is ignored. The IrDA 1.1 specifies a minimum 10ms latency after an optical node ceases transmitting before its receiver recovers its receiving function and software must guarantee this delay.

In the IrDA 1.1 specification, communication must start up at the rate of 9600bps, but then allows the link to negotiate higher (or lower) data rates if supported by both ends. However, the communication rate will not automatically change. Change, if necessary, is performed by software.

36.3.8 For any frequency clock to use the UART

NOTE: if you don't set M register and UACR the UART work at normal mode with the specified frequencies. To use other frequency you should to set M register and UACR to right value.

1 The Improving

Following changes are made:

a One bit is composed by M CLK_{BR} cycles, which can be 4~1024.

b Some extra CLK_{BR} cycles can be inserted in some bits in one frame, so that like M has fraction.

For instance:

$CLK_{BR} = CLK_{DEV} / N$ N = 1, 2, …

$CLK_{BR} = CLK_{DEV}$ = 4MHz

Band rate = 460800

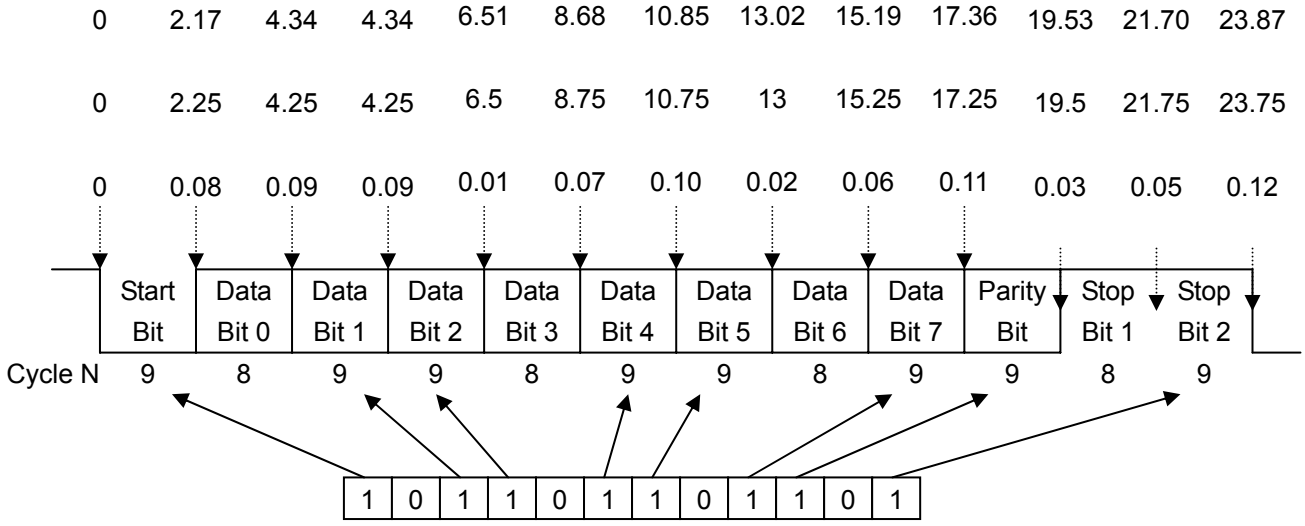
In accurate

$M_a = 8.681$

We take

M = 8, with 8 extra cycles in every frame

A 12-bit register is used to indicate where to insert the extra cycles.



For transmission, in theory, the biggest error is half of CLK_{BR} cycle, which is 0.125us here.

2 To set UMR register

$CLK_{BR} = CLK_{DEV} / N$

$M_a = CLK_{BR}$/band rate

M is modem of M_a.

Write M to Mregister.

Considering the power and the robust quality, for M form 6 to 32 is you better select by set the UDLR.

The max error

$$\frac{0.5 / CLK_{BR}}{M_a / CLK_{BR}} = 0.5/M_a < 0.5/M$$

M	4	8	16	32	64
error/W_{bit}	12.5%	6.25%	3.125%	1.56%	0.78%

3 To set UACR value

For each bit of it means:

0: means not to add additional cycle to the bit that UART is prepare to transmit or receive, in another word, you will to use M cycles to transmit or receive the bit

1: means to add additional cycle to the bit that UART is prepare to transmit or receive, in another word, you will to use M+1 cycles to transmit or receive the bit

To set UACR value you must ensure that the max error of each bit should be less than $0.5P_{BR}$.

For example：M_a-M =0.15; M+1-M_a=0.85;

Write UMR 8

Write UMR 408

cycle/bit	：	M，	M，	M，	M+1，	M，	M，	M，	M，	M，	M，	M+1，	M
UACR	：	0	0	0	1	0	0	0	0	0	0	1	0

37 Smart Card Controller

37.1 Overview

Smart Card Controller (SCC) interface is a primary device and communications interface for kinds of IC cards. The SCC interface supports communication with smart cards as specified in standard ISO7816-3. There are two SCC interfaces integrated in this SOC. And they perform the same functions. The SCC interface supports T=0 and T=1 protocol defined in ISO7816-3.

Software controls the session between the SCC interface and the card by SCC registers. Choosing protocol type and parameters, receiving and sending a byte to/from the card, activating/deactivating the card, and similar operations are accomplished with read/write operations to SCC registers. Transforming byte convention (inverse to direct and vice-versa, according to the session convention) is performed within SCC. Hence, software does not have to perform format inversion before character receipt. The SCC interface provides functionality to support the above standards, but it is the responsibility of software to ensure the standards are met.

Features:

- Supports normal card and UIM card
- 8-bit, 16-level receive-/transmit- FIFO
- Supports asynchronous character (T=0) communication modes
- Supports asynchronous block (T=1) communication modes
- Supports setting of clock-rate conversion factor F (372, 512, 558, etc.), and bit-rate adjustment factor D (1, 2, 4, 8, 16, 32, 12, 20, etc.)
- Supports extra guard time waiting
- Auto-error detection in T=0 receive mode
- Auto-character repeat in T=0 transmit mode
- Transforms inverted format to regular format and vice versa
- Support stop clock function in some power consuming sensitive applications

37.2 Pin Description

Table 37-1 Smart Card Controller Pins Description

Name	I/O	Description
SCC_CLK	Output	Serial clock connects SCC and the card
SCC_DAT	Input/Output	Data communication pin

37.3 Register Description

Table 37-2 Smart Card Controller Registers Description

Name	RW	Reset Value	Address	Access Size
SCCDR	RW	0x??	0x10041000	8
SCCFDR	R	0x00	0x10041004	8
SCCCR	RW	0x00000000	0x10041008	32
SCCSR	RW	0x8000	0x1004100C	16
SCCTFR	RW	0x0173	0x10041010	16
SCCEGTR	RW	0x00	0x10041014	8
SCCECR	RW	0x00000000	0x10041018	32
SCCRTOR	RW	0x00	0x1004101C	8

37.3.1 Transmit/Receive FIFO Data Register (SCCDR)

SCCDR 0x10041000

Bit	7	6	5	4	3	2	1	0
	DR							
RST	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
7:0	DR	Data port of HW FIFO.	RW

37.3.2 FIFO Data Count Register (SCCFDR)

SCCFDR 0x10041004

Bit	7	6	5	4	3	2	1	0
	Reserved			R				
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:5	Reserved		R
4:0	R	Characters resisted in FIFO.	R

37.3.3 Control Register (SCCCR)

SCCCR 0x10041008

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SCCE	TRS	T2R	Reserved			FDIV		FLUSH	Reserved					TRIG		TP	CONV	TXIE	RXIE	TENDIE	RTOIE	ECIE	EPIE	RETIE	EOIE	Reserved		TSEND	PX		CLKSTP
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31	SCCE	Enables or disables SCC. When disable it, the SCC_CLK will be stopped.	RW
30	TRS	Transmit or Receive Select. 0: Reception mode; 1: Transmission mode.	RW
29	T2R	Auto-T2R support. T2R means Transmit turn to Reception. 0: controlled by SW; 1: controlled by HW.	RW
28:26	Reserved		R
25:24	FDIV	Frequency Divider Select. 00: Same as device clk 01: Half of device clk 10: ¼ of device clk 11: Reserved	RW
23	FLUSH	Flush FIFO. 0: Does not empty the Rx/Tx FIFO; 1: Empty the Rx/Tx FIFO.	RW
22:18	Reserved		R
17:16	TRIG	Receive/Transmit FIFO trigger. 00: 1 01: 4 10: 8 11: 14	RW
15	TP	Communicate protocol. 0: (T=0); 1: (T=1).	RW
14	CONV	Card data transfer convention. 0: LSB first; 1: MSB first and inverted.	RW
13	TXIE	Tx FIFO counter meets the trigger value interrupt enable bit.	RW
12	RXIE	Rx FIFO counter meets the trigger value interrupt enable bit.	RW
11	TENDIE	Transmission finished interrupt enable bit. (Both FIFO and transmitter are empty)	RW
10	RTOIE	Reception timeout interrupt enable bit.	RW
9	ECIE	ETU counter overflow interrupt enable bit.	RW
8	EPIE	Parity error interrupt enable bit.	RW
7	RETIE	Re-transmitting 3 times interrupt enable bit.	RW
6	EOIE	Receive overrun error interrupt enable bit.	RW
5:4	Reserved		R

3	TSEND	Receive TS character stage finished bit. 0: TS character has not been received; 1: TS character has been received.	RW
2:1	PX	Parameter X. SCC stop clock mode selection. 00 – Does not support SCC clock stop 01 – SCC_CLK stops at state low 10 – SCC_CLK stops at state high 11 – Reserved	RW
0	CLKSTP	SCC clock stop. 0:SCC has left or is leaving clock stop mode; 1: SCC has entered or is entering clock stop mode.	RW

37.3.4 Status Register (SCCSR)

SCCSR 0x1004100C

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TRANS	Reserved		ORER	RTO	PER	TFTG	RFTG	TEND	Reserved		RETR_3	Reserved			ECNTO
RST	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
15	TRANS	Transfer status. Specifies whether the transfer is stopped or not.	R
14:13	Reserved		R
12	ORER	Receive overrun error.	RW
11	RTO	Reception timeout.	R
10	PER	Parity error.	RW
9	TFTG	Hit Tx FIFO trigger.	R
8	RFTG	Hit Rx FIFO trigger.	R
7	TEND	Transmission end. Both Tx FIFO and transmitter are empty.	RW
6:5	Reserved		R
4	RETR_3	Re-transmit exceed 3 times.	RW
3:1	Reserved		R
0	ECNTO	ETU counter overflow.	RW

37.3.5 Transmission Factor Register (SCCTFR)

SCCTFR 0x10041010

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved					FD										
RST	0	0	0	0	0	0	0	1	0	1	1	1	0	0	1	1

Bits	Name	Description	RW
15:11	Reserved		R
10:0	FD	Value of F/D. The initial value is 0x173(371).	RW

37.3.6 Extra Guard Timer Register (SCCEGTR)

SCCEGTR 0x10041014

Bit	7	6	5	4	3	2	1	0
	EGTR							
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:0	EGTR	The register is corresponding with N value of ISO7816-3. Which indicates the extra-guard time of a transmission?	RW

37.3.7 ETU Counter Value Register (SCCECR)

SCCECR 0x10041018

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved												EC																			
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:20	Reserved		R
19:0	EC	ETU counter. Write operation will clear the internal counter automatically.	RW

37.3.8 Reception Timeout Register (SCCRTOR)

SCCRTOS								0x1004101C
Bit	7	6	5	4	3	2	1	0
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:0	RTO	Retry times when parity error detected.	RW

38 TS Slave Interface (TSSI)

38.1 Overview

The TS Slave Interface (TSSI) in JZ4760 is used to connect DTV Demodulator. It supports MPEG-2 Transport Stream (TS) as its input.

Features:

- Support both parallel mode and serial mode for TS data transfer
- TSDI0 or TSDI7 can be used to transfer data in serial mode
- The order of data in one byte supports LSB at first or MSB at first
- The order of data in one word supports LSB at first or MSB at first
- Input control signals and data can be either active high or active low
- Support using either positive or negative edge of TSCLK
- Support PID filtering function
- Up to 33 PID filters can be used when PID filtering function is enabled
- Support adding data 0 before or after transport stream

38.2 Pin Description

Table 38-1 TSSI Pin Description

Name	I/O	Description
TSDI0~7	I	TS data bus
TSFAIL	I	TS packet uncorrectable
TSCLK	I	TS clock
TSFRM	I	TS data valid
TSSTR	I	TS packet start

38.3 Register Description

In this section, we will describe the registers in TSSI. Following table lists all the register definitions. All registers' 32bit addresses are physical addresses. And detailed function of each register will be described below.

Table 38-2 TSSI Register Description

Name	Description	RW	Reset Value	Address	Access Size
TSENA	TSSI Enable Register	RW	0x08	0x10073000	8
TSCFG	TSSI Configure Register	RW	0x04FF	0x10073004	16
TSCTRL	TSSI Control Register	RW	0x03	0x10073008	8
TSSTAT	TSSI State Register	RW	0x00	0x1007300C	8
TSFIFO	TSSI FIFO Register	R	0x????????	0x10073010	32
TSPEN	TSSI PID Enable Register	RW	0x00000000	0x10073014	32
TSPID0	TSSI PID Filter Register 0	RW	0x00000000	0x10073020	32
TSPID1	TSSI PID Filter Register 1	RW	0x00000000	0x10073024	32
TSPID2	TSSI PID Filter Register 2	RW	0x00000000	0x10073028	32
TSPID3	TSSI PID Filter Register 3	RW	0x00000000	0x1007302C	32
TSPID4	TSSI PID Filter Register 4	RW	0x00000000	0x10073030	32
TSPID5	TSSI PID Filter Register 5	RW	0x00000000	0x10073034	32
TSPID6	TSSI PID Filter Register 6	RW	0x00000000	0x10073038	32
TSPID7	TSSI PID Filter Register 7	RW	0x00000000	0x1007303C	32
TSPID8	TSSI PID Filter Register 8	RW	0x00000000	0x10073040	32
TSPID9	TSSI PID Filter Register 9	RW	0x00000000	0x10073044	32
TSPID10	TSSI PID Filter Register 10	RW	0x00000000	0x10073048	32
TSPID11	TSSI PID Filter Register 11	RW	0x00000000	0x1007304C	32
TSPID12	TSSI PID Filter Register 12	RW	0x00000000	0x10073050	32
TSPID13	TSSI PID Filter Register 13	RW	0x00000000	0x10073054	32
TSPID14	TSSI PID Filter Register 14	RW	0x00000000	0x10073058	32
TSPID15	TSSI PID Filter Register 15	RW	0x00000000	0x1007305C	32
TSNUM	TSSI Data Number Register	RW	0x00	0x10073018	8
TSDTR	TSSI Data Trigger Register	RW	0x7F	0x1007301C	8

38.3.1 TSSI Enable Register (TSENA)

The register TSENA is used to trigger TSSI to work.

TSENA **0x10073000**

Bit	7	6	5	4	3	2	1	0
	SFT_RST	Reserved		FAIL	PEN_0	PID_EN	DMA_EN	ENA
RST	0	0	0	0	1	0	0	0

Bits	Name	Description	RW
7	SFT_RST	TSSI FIFO software-reset. Set it to 1 and later it will be cleared by hardware auto. 0: Stop reset 1: Start reset	RW
6:5	Reserved	These bits always read 0, and writing operations are ignored.	R
4	FAIL	0：FAIL signal is decided by TSSI Demodulator 1：FAIL signal is equal to 1	
3	PEN_0	Choose PID filter enable for PID=0. 0: not enable 1: enable	RW
2	PID_EN	Enable / disenable the PID filtering function. 0: PID filtering function is not enabled 1: PID filtering function is enabled	RW
1	DMA_EN	Enable / disenable the DMA mode. 0: DMA mode is not enabled (CPU only) 1: DMA mode is enabled	RW
0	ENA	Enable / disenable the TSSI module. 0: TSSI is not enabled 1: TSSI is enabled	RW

38.3.2 TSSI Configure Register (TSCFG)

The register TSCFG is used to configure the TSSI.

TSCFG **0x10073004**

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	F_TRIGV		Reserved		TRANS_MD		EDN_WD	EDN_BT	TSDI_H	USE_0	TSCLK_CH	PARAL	TSCLK_P	TSFRM_H	TSSTR_H	TSFAIL_H
RST	0	0	0	0	0	1	0	0	1	1	1	1	1	1	1	1

Bits	Name	Description	RW
15:14	F_TRIGV	Specify the trigger value of FIFO. F_TRIGV: Description 0: Trigger Value is 4 1: Trigger Value is 8 2: Trigger Value is 16 3: Trigger Value is 32	RW
13:12	Reserved	These bits always read 0, and writing operations are ignored.	R
11:10	TRANS_MD	Choose the mode of adding data 0. TRANS_MD: Description 0: Add data 0 before transport stream 1: Add data 0 after transport stream 2: Do not add data 0 3: Reserved	RW
9	EDN_WD[*1]	The order of data in word.	RW
8	EDN_BT[*1]	The order of data in byte.	RW
7	TSDI_H	Choose the polarity of TSDI0~7. 0: TSDI0~7 is active low 1: TSDI0~7 is active high	RW
6	USE_0	USE_0 is only used in SERIAL mode. (TSCFG.PARAL=0) 0: Use TSDI7 to transfer data 1: Use TSDI0 to transfer data	RW
5	TSCLK_CH	Choose how to use TSCLK. 0: When $f_{pclk} > 3f_{TSCLK}$ 1: When $f_{pclk} > 2f_{TSCLK}$	RW
4	PARAL	Choose the working mode of TSSI. 0: Serial Mode 1: Parallel Mode	RW
3	TSCLK_P	This bit is used to determine which edge of TSCLK is used when TSSI is sampling data. 0: Use the negative edge of TSCLK	RW

		1: Use the positive edge of TSCLK	
2	TSFRM_H	Choose the polarity of TSFRM. 0: TSFRM is active low 1: TSFRM is active high	RW
1	TSSTR_H	Choose the polarity of TSSTR. 0: TSSTR is active low 1: TSSTR is active high	RW
0	TSFAIL_H	Choose the polarity of TSFAIL. 0: TSFAIL is active low 1: TSFAIL is active high	RW

NOTE:

*1: END_BT and END_WD in register TSCFG.

	Byte0 Bit0-7	Byte1 Bit0-7	Byte2 Bit0-7	Byte3 Bit0-7	Byte4 Bit0-7	Byte5 Bit0-7	Byte6 Bit0-7	Byte7 Bit0-7
END_WD=0	Byte0	Byte1	Byte2	Byte3				
	Byte4	Byte5	Byte6	Byte7				
END_WD=1	Byte3	Byte2	Byte1	Byte0				
	Byte7	Byte6	Byte5	Byte4				
END_BT=0	Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7
END_BT=1	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

38.3.3 TSSI Control Register (TSCTRL)

The register TSCTRL is used to control TSSI to work.

TSCTRL **0x10073008**

Bit	7	6	5	4	3	2	1	0
	Reserved					F_DTRM	F_OVRNM	F_TRIGM
RST	0	0	0	0	0	0	1	1

Bits	Name	Description	RW
7:3	Reserved	These bits always read 0, and writing operations are ignored.	R
2	F_DTRM	FIFO data trigger interrupt mask. 0: enabled 1: masked	RW
1	F_OVRNM	FIFO overrun interrupt mask. 0: enabled 1: masked	RW
0	F_TRIGM	FIFO trigger interrupt mask. 0: enabled 1: masked	RW

38.3.4 TSSI State Register (TSSTAT)

The register TSSTAT is used to keep the state of TSSI.

TSSTAT 0x1007300C

Bit	7	6	5	4	3	2	1	0
	Reserved					F_DTR	F_OVRN	F_TRIG
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7:3	Reserved	These bits always read 0, and writing operations are ignored.	R
2	F_DTR	FIFO data trigger interrupt flag. 1: active 0: not active	RW
1	F_OVRN	FIFO overrun interrupt flag. 1: active 0: not active	RW
0	F_TRIG	FIFO trigger interrupt flag. 1: active 0: not active	RW

38.3.5 TSSI FIFO Register (TSFIFO)

The register TSFIFO is corresponded to TSSI FIFO.

TSFIFO **0x10073010**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	DATA																															
RST	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

38.3.6 TSSI PID Enable Register (TSPEN)

The register TSPEN is used to control the PID filtering.

TSPEN **0x10073014**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EN_151	EN_141	EN_131	EN_121	EN_111	EN_101	EN_91	EN_81	EN_71	EN_61	EN_51	EN_41	EN_31	EN_21	EN_11	EN_01	EN_150	EN_140	EN_130	EN_120	EN_110	EN_100	EN_90	EN_80	EN_70	EN_60	EN_50	EN_40	EN_30	EN_20	EN_10	EN_00
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:16	EN_x1 (x=15~0)	PID filter enable for TSPIDx.PID1. 0: not enable 1: enable	RW
15:0	EN_x0 (x=15~0)	PID filter enable for TSPIDx.PID0. 0: not enable 1: enable	RW

38.3.7 TSSI PID Filter Registers (TSPID0~15)

The registers TSPID0~15 are used to store PID values that need to be filtered from MPEG-2 TS.

TSPID0~15 **0x10073020~0x1007305C**

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	Reserved			PID1													Reserved			PID0												
RST	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
31:29	Reserved	These bits always read 0, and writing operations are ignored.	R
28:16	PID1	Set the PID value that needs to be filtered.	RW
15:13	Reserved	These bits always read 0, and writing operations are ignored.	R
12:0	PID0	Set the PID value that needs to be filtered.	RW

38.3.8 TSSI Data Number Register (TSNUM)

The register TSNUM is used to show the current number of the data in FIFO.

TSNUM **0x10073018**

Bit	7	6	5	4	3	2	1	0
	Reserved				DNUM			
RST	0	0	0	0	0	0	0	0

Bits	Name	Description	RW
7	Reserved	These bits always read 0, and writing operations are ignored.	R
6:0	DNUM	The current number of data in FIFO. The range of the data number is from 0 to 127.	RW

38.3.9 TSSI Data Trigger Register (TSDTR)

The register TSDTR is used to trigger the FIFO level interrupt when the current data number in the FIFO is more than the value in TSDTR.

TSDTR **0x1007301C**

Bit	7	6	5	4	3	2	1	0
	Reserved				DTR			
RST	0	1	1	1	1	1	1	1

Bits	Name	Description	RW
7	Reserved	These bits always read 0, and writing operations are ignored.	R
6:0	DTRG	The trigger number of FIFO. The range of the data number is from 0 to 127.	RW

38.4 TSSI Timing

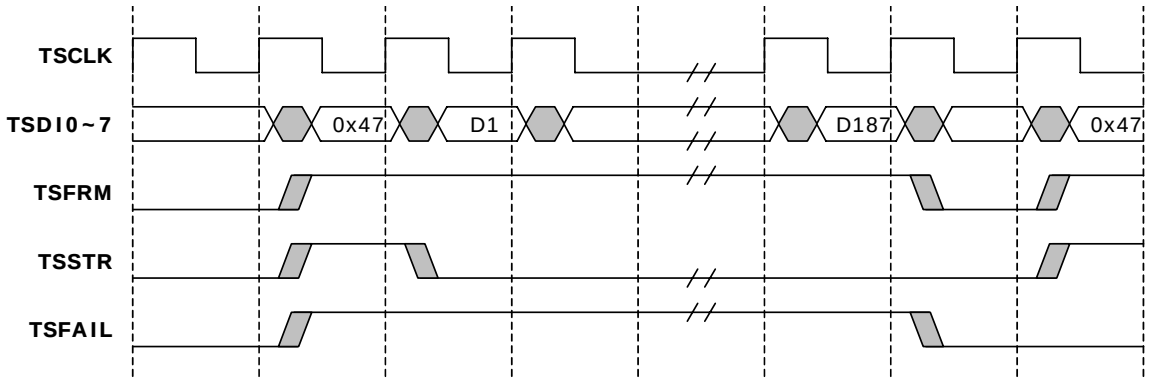


Figure 38-1 Timing waveform in parallel mode

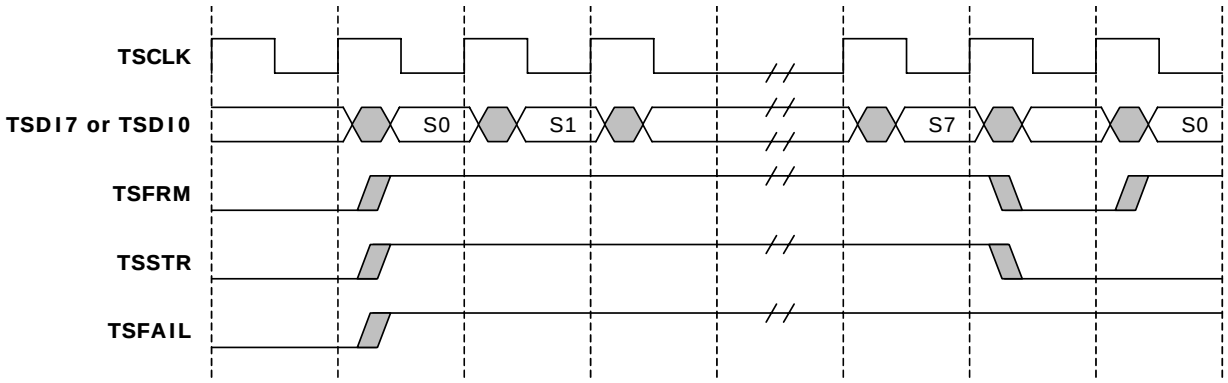


Figure 38-2 Timing waveform in serial mode

38.5 TSSI Guide

38.5.1 TSSI Operation without PID Filtering Function

1 Set TSCTRL to 0x03 to mask all interrupts.
2 Set TSCFG to choose the working mode of TSSI and the trigger value of FIFO.
3 Set TSENA.PID_EN to 0 to turn off the PID filtering function.
4 Set TSENA.DMA_EN to 1 or 0 to decide whether to use the DMA mode or not.
5 Write 0x00 to TSSTAT clear all interrupt flag.
6 Set TSCTRL to 0x00 to enable all interrupts.
7 Set TSENA.ENA to 1 to turn on TSSI module.

38.5.2 TSSI Operation with PID Filtering Function

1 Set TSCTRL to 0x03 to mask all interrupts.
2 Set TSCFG to choose the working mode of TSSI and the trigger value of FIFO.
3 Set TSENA.PID_EN to 1 to turn on the PID filtering function.
4 Set TSENA.DMA_EN to 1 or 0 to decide whether to use the DMA mode or not.
5 Write 0x00 to TSSTAT clear all interrupt flag.
6 Set TSCTRL to 0x00 to enable all interrupts.
7 Set TSENA.ENA to 1 to turn on TSSI module.
8 Change TSPID registers and then set TSPID to enable the PID filter.
9 When PID in TS package is equal to the valve in TSPID register, the TS package will be got.

39 PS/2 Keyboard Controller

39.1 Overview

The PS/2 keyboard controller (KBC) is designed to provide the functions to a keyboard or to a PS/2 mouse. KBC receives serial data from the keyboard or mouse, checks the parity of the data, and presents the data to the system as a byte of data in its output buffer. Then, the controller will assert an interrupt to the system when data are placed in its output buffer. The keyboard and PS/2 mouse are required to acknowledge all data transmissions. No transmission should be sent to the keyboard or PS/2 mouse until acknowledge is received for the previous byte sent.

Features:

- Be compatible with 8042
- Support PS/2 keyboard and mouse
- Bi-directional synchronous serial transfer operation
- A frame format: 1 start bit, 8 data bits, 1 odd parity bit and 1 stop bit
- Device provides about 20KHz clock to KBC
- KBC has priority over the data line and can inhibit communication from the keyboard/mouse at any time by holding Clock low

39.2 Pin Description

Name	TYPE	Description
KCLK	IO	Keyboard Clock pin
KDAT	IO	Keyboard Data pin
MCLK	IO	Mouse Clock pin
MDAT	IO	Mouse Data pin

39.3 Register Description

All KBC register access address is KBC physical address.

Name	Description	RW	Reset Value	Address	Access Size
KTDR	KBC Transmit Data Register	W	0x????????	0x10062060	8
KRDR	KBC Receive Data Register	R	0x????????	0x10062060	8
KCCR	KBC Command Register	W	0x????????	0x10062064	8
KCSR	KBC Status Register	R	0x00	0x10062064	8

39.3.1 KBC Transmit Data Register (KTDR)

KTDR is a write-only input buffer. The data is written to KTDR by CPU, and then KBC read out this data from this KTDR and this data will be sent out in KDAT or MDAT pin serially. After the data in KTDR is sent out completely, KTDR becomes empty. If KTDR isn't empty, software shouldn't write KTDR again, otherwise, the new data covers the old data. According to IBF bit in KCSR, software knows whether KTDR is empty or not.

KTDR **0x10062060**

Bit	7	6	5	4	3	2	1	0
	KTDR							
RST	?	?	?	?	?	?	?	?

39.3.2 KBC Receive Data Register (KRDR)

KRDR is a read-only output buffer. A byte from KDAT or MDAT pin is written to KRDR and KBC request CPU to read out this data from KRDR. Before the data in KRDR is read out by CPU, KCLK or MCLK pin is pulled low by KBC to prevent keyboard or mouse from transmitting data to KBC. After CPU reads out the data in KRDR, KBC release KCLK or MCLK pin, then keyboard or mouse can transmit data to KBC.

KRDR **0x10062060**

Bit	7	6	5	4	3	2	1	0
	KRDR							
RST	?	?	?	?	?	?	?	?

39.3.3 KBC Command Register (KCCR)

KCCR is a write-only register. Writing KCCR doesn't write to any specific register, but sends a command for the KBC to interpret. If the command accepts a parameter, this parameter is sent to KTDR. Likewise, any results returned by the command may be read from KRDR.
The usable command to KBC is described in back text.

KCCR								0x10062064
Bit	7	6	5	4	3	2	1	0
	KCCR							
RST	?	?	?	?	?	?	?	?

39.3.4 KBC Status Register (KCSR)

KCSR								0x100062064
Bit	7	6	5	4	3	2	1	0
	PER	TOT	MOF	INF	AD2	SYF	ITF	OTF
RST	?	?	?	?	?	?	?	?

Bits	Name	Description	RW
7	PER	Parity Error, if this bit is set to 1, it indicates that the data received from the keyboard or mouse had even parity. After a reading KRDR operation, PER bit is cleared to 0 automatically. 0: Odd parity received 1: Even parity received	R
6	TOT	Time-Out, when this bit is set to 1, it indicates that a transmission was started by the keyboard or mouse but did not finish within the receive time-out delay (2ms), or that a transmission was started by KBC but the byte transmitted was not clocked out within the specified time limit (15ms). When a receive time-out occurs, KBC places a hex FF in the output buffer. When a transmit time-out occurs, KBC places a hex FE in the output buffer. After a reading KRDR operation, TOT bit is cleared to 0 automatically. 0: No Time-out 1: Time-out occurs	R
5	MOF	Mouse Output Buffer Full. This bit works in conjunction with OF bit. When this bit and OTF bit are set to 1, mouse data is in the output buffer and MOF interrupt is asserted if INT2 bit in command byte is 1. When this bit is	R

		0 and OTF is set to 1, keyboard or KBC command response data is in the output buffer. After a reading KRDR operation, MOF bit is cleared to 0 automatically. 0: The data in output buffer isn't mouse data 1: The data in output buffer is mouse data	
4	INF	Inhibit Flag, this bit indicates whether or not keyboard communication is inhibited. If this bit is set to 0, KCLK pin is pulled low, so keyboard is inhibited. 0: Keyboard is inhibited 1: Keyboard isn't inhibited	R
3	AD2	Address Line A2. This bit indicates which register was last written to. If this bit is 0, the last write operation is writing KTDR, otherwise is KCCR. 0: KTDR was last written 1: KCCR was last written	R
2	SYF	System Flag. This bit is set to 0 or 1 by writing to the system flag bit (SYS bit in command byte). This bit is cleared to 0 after a power-on reset. 0: SYS bit in command byte is 0 1: SYS bit in command byte is 1	R
1	ITF	Input Buffer Full. This bit indicates whether input buffer is full. Writing KTDR operation for transmitting data to keyboard or mouse set this bit to 1. After the data in KTDR is sent out completely, this bit is cleared to 0. 0: Input buffer is empty 1: Input buffer is full	R
0	OTF	Output Buffer Full. This bit indicates whether output buffer is full. When a byte keyboard or mouse data is put to KRDR, or KBC's command response data is put to KRDR, this bit is set to 1 and OBF interrupt is asserted if INT bit in command byte is 1. After a reading KRDR operation, OTF bit is cleared to 0 automatically. 0: Output buffer is empty 1: Output buffer is full	R

39.4 KBC Commands

Commands are sent to the keyboard controller by writing to port 0x64 (KCCR). Command parameters are written to port 0x60 (KTDR) after command is sent. Results are returned on port 0x60 (KRDR). Always test the IBF flag before writing commands or parameters to KBC.

39.4.1 Write command byte (0x60)

KBC has a command byte register that is used to configure KBC. Command byte's parameter via KTDR is written in command byte register. Command byte defined as follows:

Write command

Bit	7	6	5	4	3	2	1	0
	Reserved	SCT	MEN	KEN	Reserved	SYF	MINT	OINT
RST	0	1	1	1	0	0	0	0

Bits	Name	Description	RW
7	Reserved	Writes to these bits have no effect and always read as 0.	R
6	SCT	Scan Codes Translate. When this bit is set to 1, KBC translates the incoming keyboard scan codes to scan set 1. When this bit is set to 0, KBC passes the incoming scan codes without translation. Following power-on or a keyboard reset, the keyboard transmits using scan code set 2. 0: Disable translation 1: Enable translation	RW
5	MEN	Disable Mouse, setting this bit to 1 disables the mouse interface by driving the MCLK pin low. Mouse data can't be received while the mouse interface is disabled. Clearing this bit to 0 enables the mouse interface by releasing the MCLK pin. 0: Enable mouse interface 1: Disable mouse interface	RW
4	KEN	Disable Keyboard, setting this bit to 1 disables the keyboard interface by driving the KCLK pin low. Keyboard data can't be received while the keyboard interface is disabled. Clearing this bit to 0 enables the keyboard interface by releasing the KCLK pin. 0: Enable keyboard interface 1: Disable keyboard interface	RW
3	Reserved	Writes to these bits have no effect and always read as 0.	R
2	SYF	System Flag. Software sets this bit to tell KBC perform power-on or "warm boot" test/initialization. This bit can be used to manually set/clear SYS flag in Status register.	RW

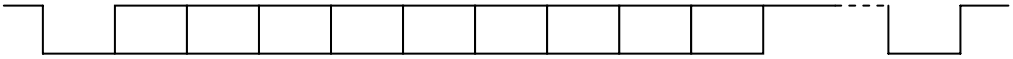
		0: Tell KBC to perform power-on test/initialization 1: Tell KBC to perform “warm boot” tests/initialization	
1	MINT	Mouse Output Buffer Full Interrupt. When set, MOF interrupt is generated when mouse data is available. 0: Disable MOF interrupt 1: Enable MOF interrupt	RW
0	OINT	Output Buffer Full Interrupt. When set, OTF interrupt is generated when data is available in the output buffer. 0: Disable OTF interrupt 1: Enable OTF interrupt	RW

39.4.2 COMMANDS

COMMAND	FUNCTION
20h	Read Command Byte of Keyboard Controller.
60h	Write Command Byte of Keyboard Controller.
A4h	Test Password. Returns 0Fah if Password is loaded. Returns 0F1h if Password is not loaded. Current , KBC doesn’t support password function, 0F1h always returned.
A7h	Disable mouse interface.
A8h	Enable mouse interface.
A9h	Mouse interface test. 00: No error detected 01: MCLK is stuck low 02: MCLK is stuck high 03: MDAT is stuck low 04: MDAT is stuck high
AAh	Self-test. Returns 055h if self test succeeds.
ABh	interface test. 00: No error detected 01: KCLK is stuck low 02: KCLK is stuck high 03: KDAT is stuck low 04: KDAT is stuck high
ADh	Disable keyboard interface.
AEh	Enable keyboard interface.
C0h	read its input port and send the data in its output buffer.
D2h	Write keyboard output buffer.
D3h	Write mouse output buffer.

D4h	Write mouse input buffer.
E0h	Reports the status of the test inputs.

39.5 Frame format



1 start bit: This is always 0.

8 data bits: least significant bit first.

1 parity bit: (odd parity).

1 stop bit: This is always 1.

1 acknowledge bit: This is always 0 (Host-to-device communication only)

39.6 Transmit flow

1 transmit start.
2 check if ITF == 0 then 3 else return 1.
3 Disable keyboard or mouse.
4 read KCSR.
5 check if OTF == 0 then 6 else read KRDR.
6 check if mouse transmission, then 7 else 8.
7 write D4 to KCCR, then 9.
8 clear SCT to 0.
9 write KTDR.
10 enable keyboard or mouse.
11 transmit end.

39.7 Receive flow

1 Receive start.
2 Read KCSR.
3 check if OTF == 1, then 4 else 1.
4 check if PER == 1, then 5 else 6.
5 check if MOF ==1 , then go to Mouse parity error handle else go to Keyboard parity error handle.
6 check if TOT ==1, then 7 else 8.
7 check if MOF == 1, then go to Mouse time-out error handle or go to Keyboard time-out handle.
8 check if MOF == 1, then 9 else 10.
9 read KRDR for getting mouse data, then 11.
10 read KRDR for getting keyboard data.
11 receive end.

40 XBurst Boot ROM Specification

The JZ4760 contains an internal 8KB boot ROM. The CPU boots from the boot ROM after reset.

40.1 Boot Select

The boot sequence of the JZ4760 is controlled by boot_sel [2:0]. The configuration is shown as follow:

Table 40-1 Boot Configuration of JZ4760

boot_sel[2:0]	Boot method
111	NAND boot @ CS1
100	SD boot @ MSC0
101	SPI boot @ SPI0/CE0
011	NOR boot @ CS4 (just for FPGA testing)
110	USB boot @ USB 2.0 device, EXTCLK=12MHz
000	USB boot @ USB 2.0 device, EXTCLK=13MHz
001	USB boot @ USB 2.0 device, EXTCLK=26MHz
010	USB boot @ USB 2.0 device, EXTCLK=19.2MHz

40.2 Boot Procedure

After reset, the boot program on the internal boot ROM executes as follows:

1 Disable all interrupts and read boot_sel[2:0] to determine the boot method.

2 If it is boot from NAND flash, 4 flags at the beginnig of NAND are read to know the NAND information including bus width(8 or 16 bits), page cycle(2 or 3 cycles) and its page size(512B, 2KB, 4KB or 8KB). Then 8KB code are read out from NAND to cache, if the 8KB reading failed, the next 8KB backup in NAND will be read. Then branch to cache at 12 bytes offset.

3 There 8KB backup reading failed, the 8KB backup at 64^{th}, 128^{th}, 192^{th} , …, and finally 1280^{th} page will be tried in consecutive order.

4 If it is boot from MMC/SD card at MSC0, its function pins MSC0_D0, MSC0_CLK, MSC0_CMD are initialized, the boot program loads the 8KB code from MMC/SD card to cache and jump to it. Only one data bus which is MSC0_D0 is used. The clock EXTCLK/128 is used initially. When reading data, the clock EXTCLK/2 is used.

5 If it is boot from USB, a block of code will be received through USB cable connected with host PC and be stored in cache. Then branch to this area in cache.

NOTE: The JZ4760's cache is 16KB, its address is from 0x80000000 to 0x80004000.

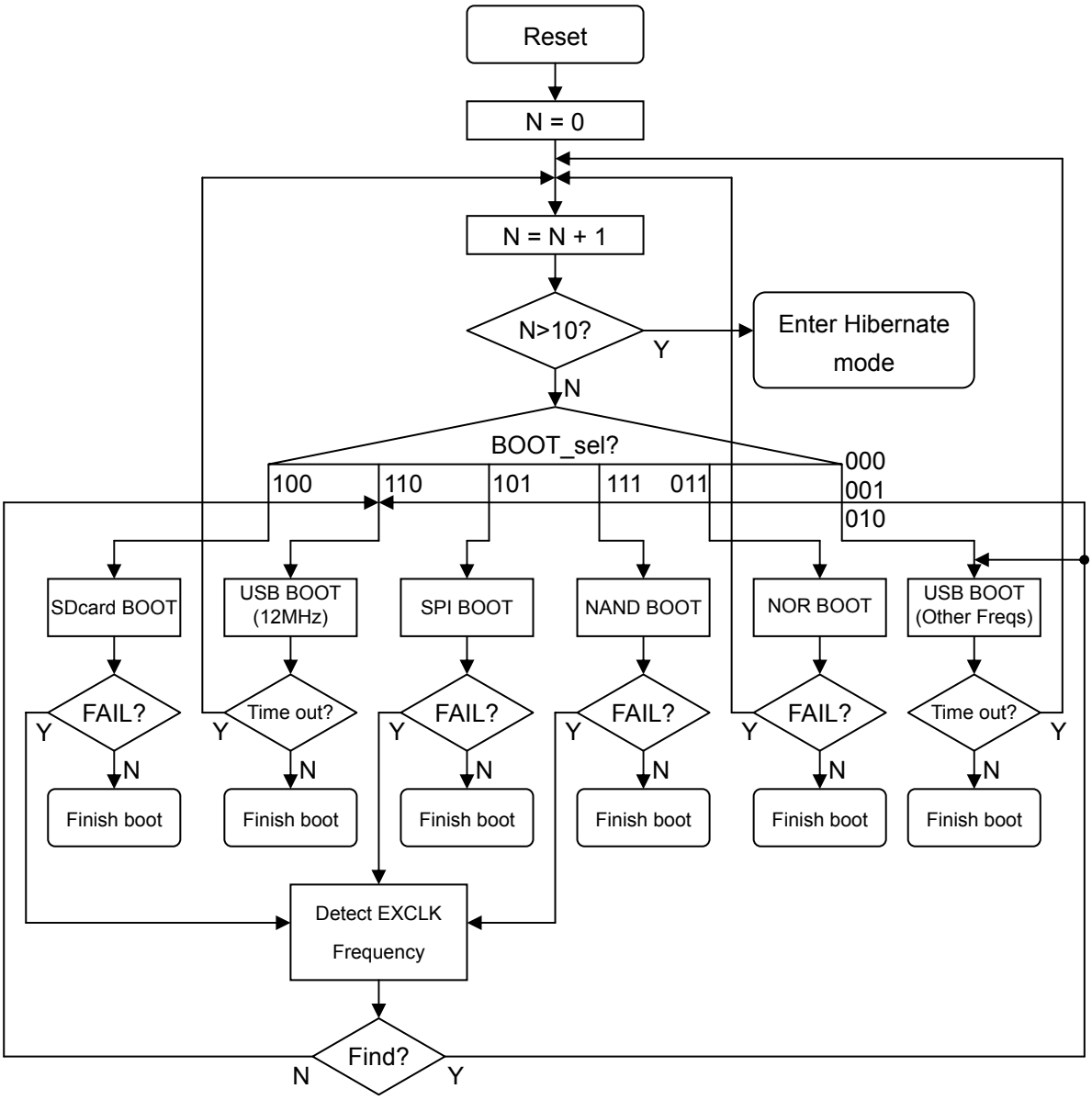


Figure 40-1 Boot sequence diagram of JZ4760

40.3 NAND Boot Specification

If CPU boots from NAND flash (CS1), the boot ROM will read 4 flags from NAND flash to know the NAND information including bus width(8 or 16 bits), page cycle(2 or 3 cycles) and its page size(512, 2KB, 4KB or 8KB bytes).

The content and definition of the 4 flags are shown as follow:

Table 40-2 The definition of 4 flags in NAND flash

Name	Location (in byte)	length (in byte)	Value	Description
buswidth_flag	0-63	64	0x55 or 0xaa	Bus width. 0x55: 8bit bus width 0xaa: 16bit bus width
rowcycle_flag	64-95	32	0x55 or 0xaa	The number of bytes of row cycles. 0x55: 2-byte row cycles 0xaa: 3-byte row cycles
pagesize_flag1	96-127	32	0x55 or 0xaa	pagesize_flag1 pagesize_flag0 pagesize(byte). 0x55 0x55 512 0x55 0xaa 2048 0xaa 0x55 4096 0xaa 0xaa 8192
pagesize_flag0	128-159	32	0x55 or 0xaa	

The buswidth_flag containing 64 bytes locates at the beginning of NAND, if the bus width of NAND is 8 bit, the buswidth_flag should be filled with 0x55 for all 64 bytes, or else it should be filled with 0xaa. The rowcycle_flag containing 32 bytes locates behind the buswidth_flag, if the number of bytes of row cycles is 2, the flag should be filled with 0x55 for all 32 bytes, or else it should be filled with 0xaa. The pagesize_flag1 and pagesize_flag0 each containing 32 bytes locate behind the rowcycle_flag, which value should be filled is determined by the page size of NAND. Please refer to table 2. Totally, 160 bytes are allocated for the 4 flags.

At first, the first 256 bytes (which is a PN* unit) in NAND containing 4 flags will be read out to a buffer assuming the bus width of NAND is 8 bit. The buswidth_flag will be get from the buffer to detect the page size (whether 512B or not) and bus width of nand. If there is no 0x55 or 0xaa in buswidth_flag, 64^{th}, 128^{th}, 192^{th}, ..., 1280^{th} page will be tried in sequence. If failed at 1280^{th} page, bootrom will jump to usb_boot. If bus width is 16 bit, 256 bytes will be read again to the buffer mentioned above. If buswidth_flag is valid, the rowcycle_flag will be obtained from the buffer to know the number of row cycles. At last, pagesize_flag1 and pagesize_flag0 will be obtained from the buffer to know precise page size.

8KB codes in NAND will be loaded up to dcache and transferred to icache and branch to icache at 160 bytes offset. Hardware PN and 24-bit BCH ECC will be used for every 256 bytes during reading. The

ECC(39 bytes per 256 bytes data) stores in the data area of a NAND page behind the page storing code data. If no ECC error is detected or ECC error is correctable(number of error bits <= 24), NAND boots successfully. If uncorrectable error occurred, next 8KB backup at 64 pages behind will be tried. 64th, 128th, 192th, ..., 1280th page will be tried in sequence. If failed at 1280th page, bootrom will jump to usb_boot.

The distribution and structure of the boot code in NAND is shown as Figure 40-2.
The procedure of the JZ4760 NAND boot is shown as Figure 40-3.

NOTES:

1 PN is short for pseudorandom noise which is used for supporting TLC (three-level cell) NAND.

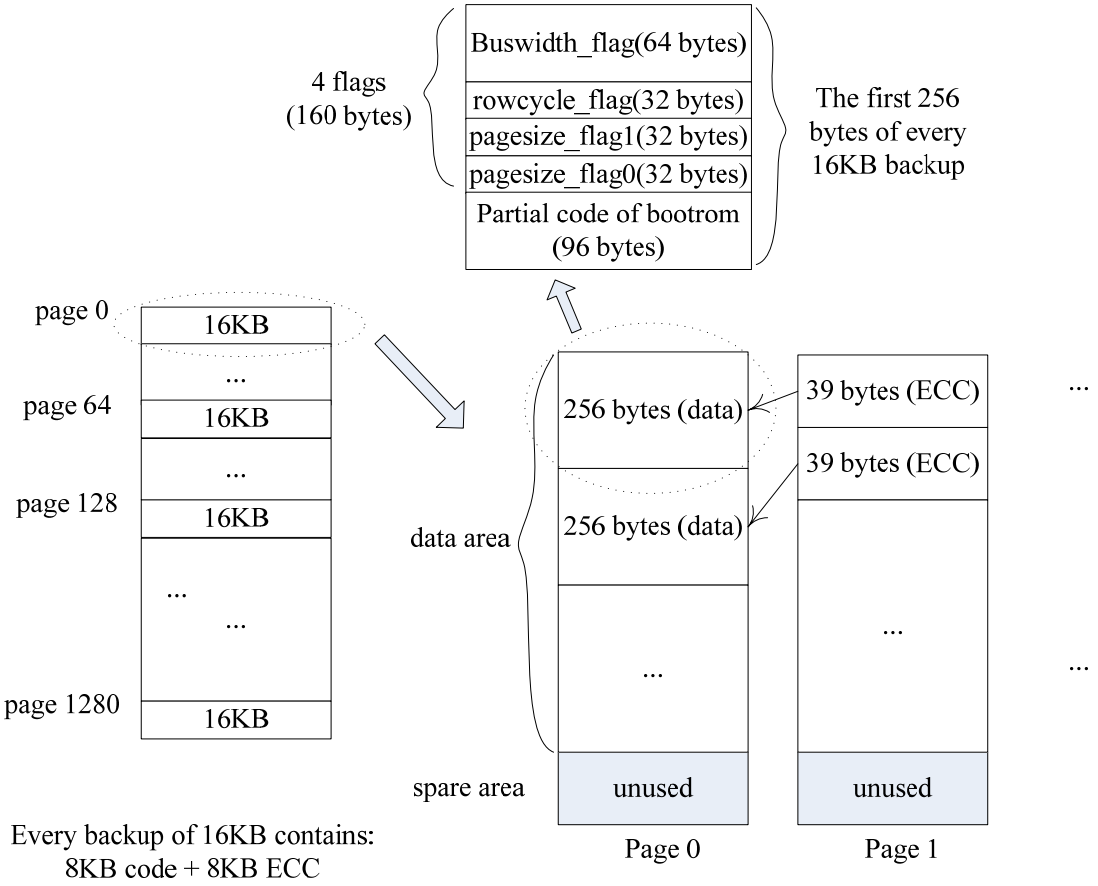


Figure 40-2 the distribution and structure of the boot code in NAND

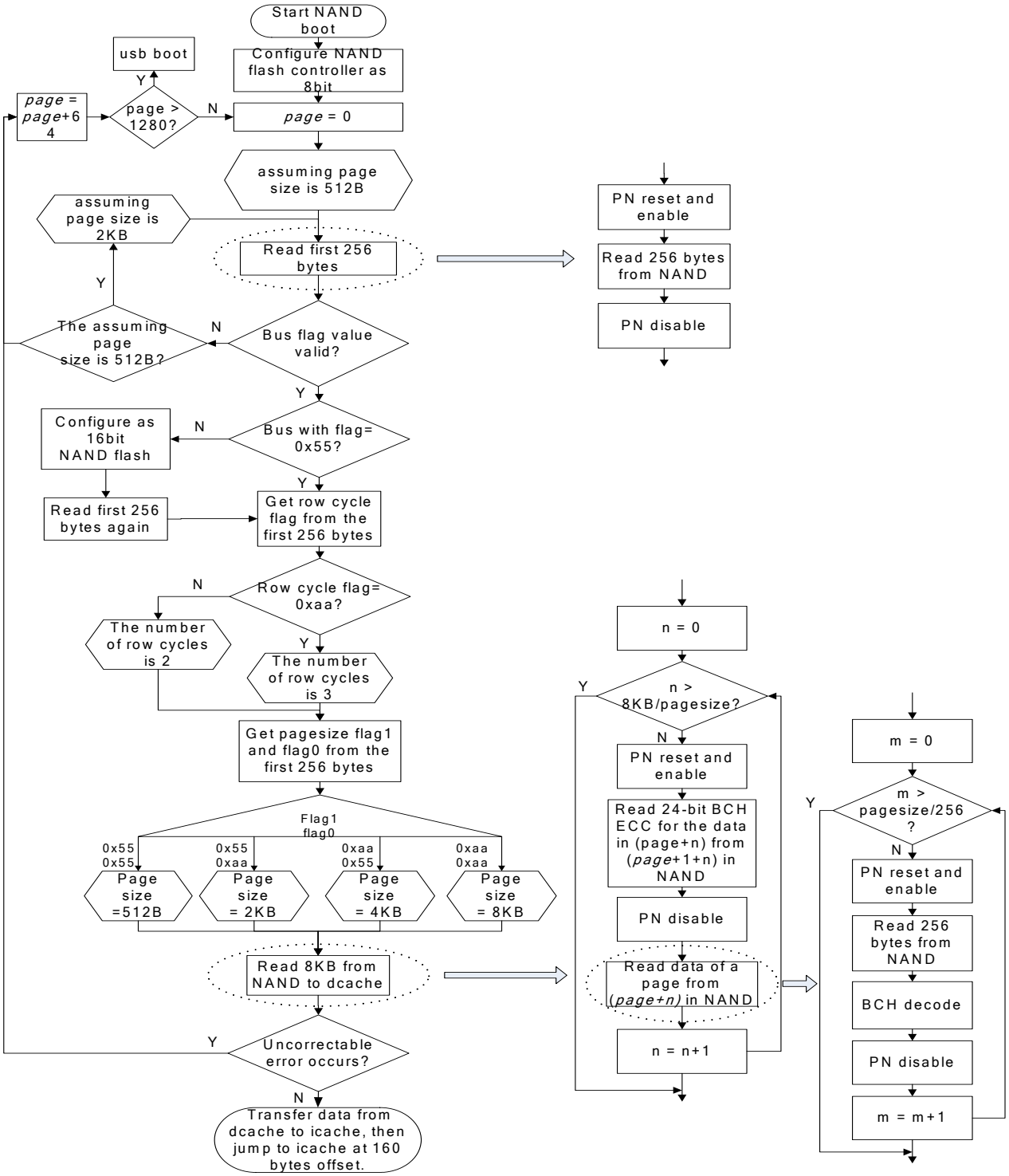


Figure 40-3 JZ4760 NAND Boot Procedure

40.4 USB Boot Specification

When boot_sel[2:0] is selected as USB boot, the internal boot ROM downloads user program from the USB port to internal SRAM and branches to the internal SRAM to execute the program.

JZ4760 supports the external main crystal whose frequency is 12MHz, 13MHz, 19.2MHz, 24MHz or 26MHz.

The boot program supports both high-speed (480MHz) and full-speed (12MHz) transfer modes. The boot program uses the following two transfer types.

Table 40-3 Transfer Types Used by the Boot Program

Transfer Type	Description
Control Transfer	Used for transmitting standard requests and vendor requests.
Bulk Transfer	Used for responding to vendor requests and transmitting a user program.

The following figure shows an overview of the USB communication flow.

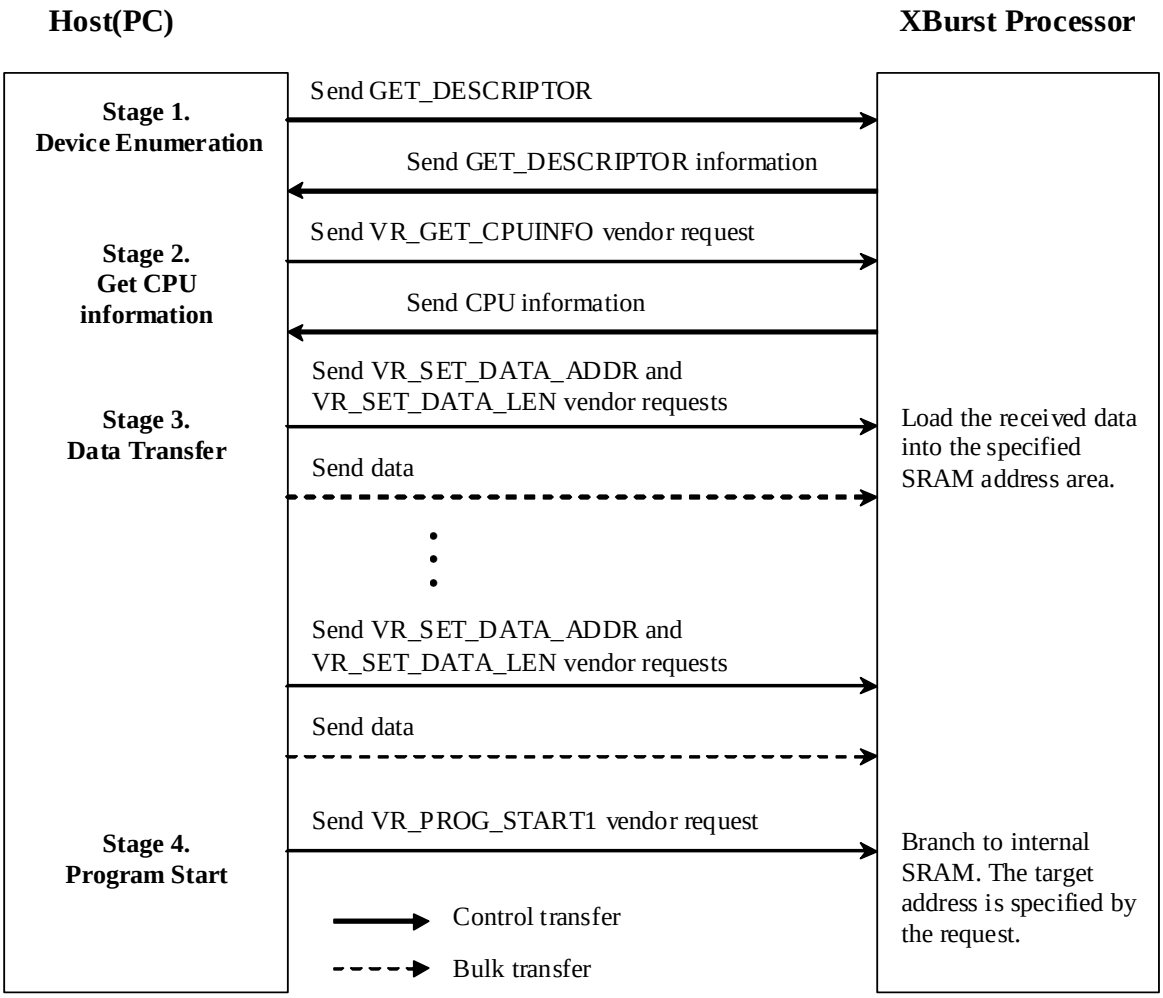


Figure 40-4 USB Communication Flow

The vendor ID and product ID for the USB boot device are 0x601A and 0x4760 respectively. The Configuration for USB is for Control Endpoint 0 with Max Packet Size equals 64 bytes, Bulk IN at Endpoint 1 with Max Packet Size equals 512 bytes in high-speed and 64 bytes in full-speed, Bulk OUT at Endpoint 1 with Max Packet Size equals 512 bytes in high-speed and 64 bytes in full-speed.

The USB boot program provides six vendor requests through control endpoint for user to download/upload data to/from device, and to branch to a target address to execute user program. The six vendor requests are VR_GET_CPU_INFO (0x00), VR_SET_DATA_ADDRESS (0x01), VR_SET_DATA_LENGTH (0x02), VR_FLUSH_CACHES (0x03), VR_PROGRAM_START1 (0x04) and VR_PROGRAM_START2 (0x05). User program is transferred through Bulk IN or Bulk OUT endpoint.

When JZ4760 is reset with boot_sel[2:0] equals 110b, 000b, 001b or 010b, the internal boot ROM will switch to USB boot mode and wait for USB requests from host. After connecting the USB device port

to host, host will recognize the connection of a USB device, and start device enumeration. After finishing the device enumeration, user can send VR_GET_CPU_INFO (0x00) to query the device CPU information. If user wants to download/upload a program to/from device, two vendor requests VR_SET_DATA_ADDRESS (0x01) and VR_SET_DATA_LENGTH (0x02) should be sent first to tell the device the address and length in byte of the subsequent transferring data. Then data can be transferred through bulk-out/bulk-in endpoint. After this first stage program has been transferred to device, user can send vendor request VR_PROGRAM_START1 (0x04) to let the CPU to execute the program. This first stage program must not greater than 16KB and is normally used to init GPIO and SDRAM of the target board. At the end of the first stage program, it can return back to the internal boot ROM by jumping to ra ($31) register. Thus user can download a new program to the SDRAM of the target board like the first stage, and send vendor request VR_FLUSH_CACHES (0x03) and VR_PROGRAM_START2 (0x05) to let the CPU to execute the new program. Next figure is the typical procedure of USB boot.

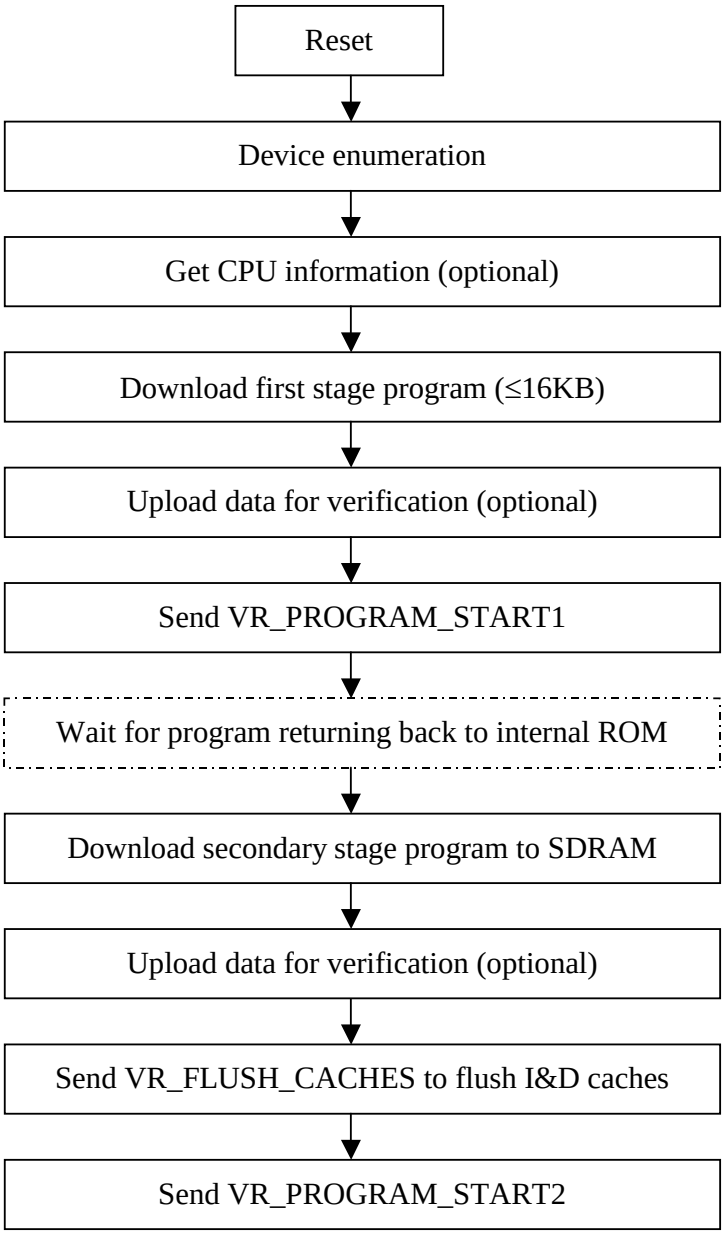


Figure 40-5 Typical Procedure of USB Boot

Following tables list all the vendor requests that USB boot program supports:

Table 40-4 Vendor Request 0 Setup Command Data Structure

Offset	Field	Size	Value	Description
0	bmRequestType	1	40H	D7 0: Host to Device. D6-D5 2: Vendor. D4-D0 0: Device.
1	bRequest	1	00H	VR_GET_CPU_INFO: get CPU information.
2	wValue	2	0000H	Not in used.
4	wIndex	2	0000H	Not in used.
6	wLength	2	0008H	8 bytes.

Table 40-5 Vendor Request 1 Setup Command Data Structure

Offset	Field	Size	Value	Description
0	bmRequestType	1	40H	D7 0: Host to Device. D6-D5 2: Vendor. D4-D0 0: Device.
1	bRequest	1	01H	VR_SET_DATA_ADDRESS: set address for next bulk-in/bulk-out transfer.
2	wValue	2	xxxxH	MSB (bit[31:16]) of the data address.
4	wIndex	2	xxxxH	LSB (bit[15:0]) of the data address.
6	wLength	2	0000H	Not in used.

Table 40-6 Vendor Request 2 Setup Command Data Structure

Offset	Field	Size	Value	Description
0	bmRequestType	1	40H	D7 0: Host to Device. D6-D5 2: Vendor. D4-D0 0: Device.
1	bRequest	1	02H	VR_SET_DATA_LENGTH: set length in byte for next bulk-in/bulk-out transfer.
2	wValue	2	xxxxH	MSB (bit[31:16]) of the data length.
4	wIndex	2	xxxxH	LSB (bit[15:0]) of the data length.
6	wLength	2	0000H	Not in used.

Table 40-7 Vendor Request 3 Setup Command Data Structure

Offset	Field	Size	Value	Description
0	bmRequestType	1	40H	D7 0: Host to Device. D6-D5 2: Vendor. D4-D0 0: Device.
1	bRequest	1	03H	VR_FLUSH_CACHES: flush I-Cache and D-Cache.
2	wValue	2	0000H	Not in used.
4	wIndex	2	0000H	Not in used.
6	wLength	2	0000H	Not in used.

Table 40-8 Vendor Request 4 Setup Command Data Structure

Offset	Field	Size	Value	Description
0	bmRequestType	1	40H	D7 0: Host to Device. D6-D5 2: Vendor. D4-D0 0: Device.
1	bRequest	1	04H	VR_PROGRAM_START1: transfer data from D-Cache to I-Cache and branch to address in I-Cache. **NOTE:** After downloading program from host to device for the first time, you can only use this request to start the program. Since the USB boot program will download data to D-Cache after reset. This request will transfer data from D-Cache to I-Cache and execute the program in I-Cache.
2	wValue	2	xxxxH	MSB (bit[31:16]) of the program entry point.
4	wIndex	2	xxxxH	LSB (bit[15:0]) of the program entry point.
6	wLength	2	0000H	Not in used.

Table 40-9 Vendor Request 5 Setup Command Data Structure

Offset	Field	Size	Value	Description
0	bmRequestType	1	40H	D7 0: Host to Device. D6-D5 2: Vendor. D4-D0 0: Device.
1	bRequest	1	05H	VR_PROGRAM_START2: branch to target address directly.
2	wValue	2	xxxxH	MSB (bit[31:16]) of the program entry point.
4	WIndex	2	xxxxH	LSB (bit[15:0]) of the program entry point.
6	WLength	2	0000H	Not in used.

40.5 MMC/SD Boot Specification

If CPU boots from MSC0, the boot program will load 8KB code starting at sector 0 from MMC/SD card to cache. First the boot program initializes MSC0_D0, MSC0_CLK, MSC0_CMD as function pins. Only one data pin MSC0_D0 is used. Then the boot program sends CMD55 to test if it's SD or MMC card and initializes the card. At last it loads 8KB code from the card to cache and branches to execute the code in cache.

When initializing the card, the clock of EXTCLK/128 is used. And when reading data, the clock of EXTCLK/2 is used.

The procedure of the JZ4760 MMC/SD boot is shown as follow:

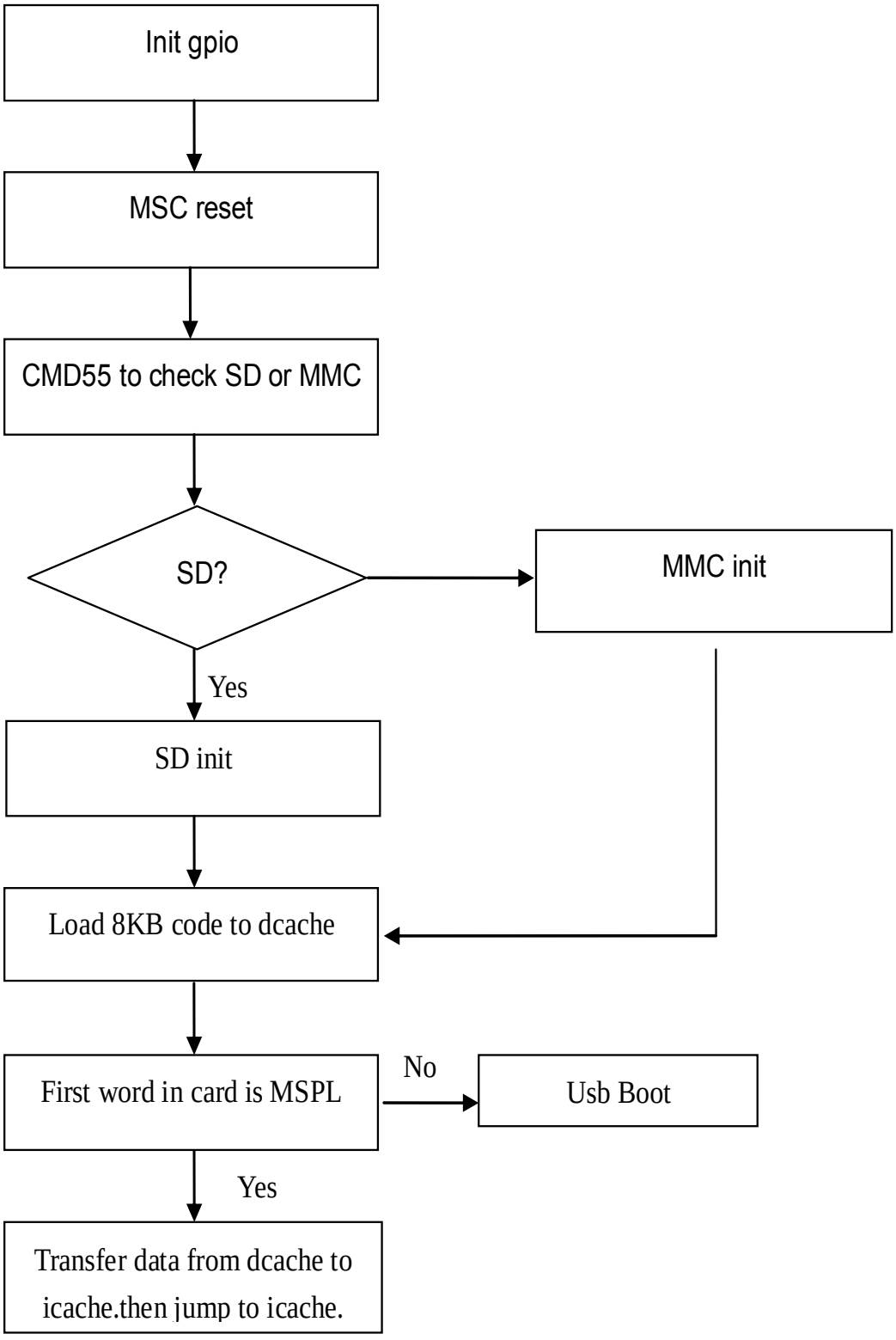


Figure 40-6 JZ4760 MMC/SD Boot Procedure

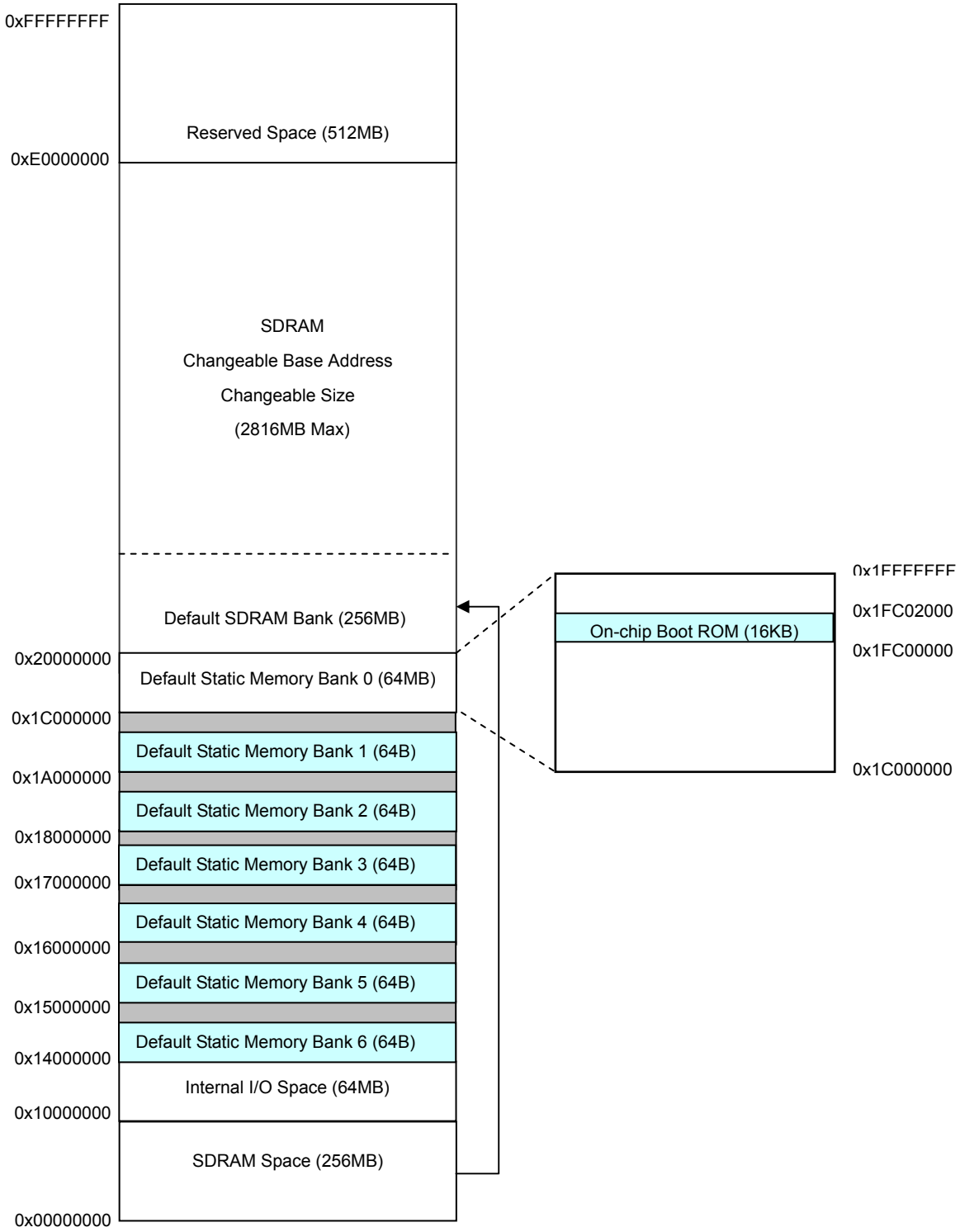
41 Memory Map and Registers

41.1 Physical Address Space Allocation

This chapter describes the physical address map, memory-mapped regions for every block in the JZ4760 processor. Both logical space and physical space of the JZ4760 are 32 bits wide. The 4Gbyte physical space is divided into several partitions for external memory, PCMCIA and internal I/O devices. Table 41-1 shows the basic physical memory map:

Table 41-1 JZ4760 Processor Physical Memory Map

Start Address	End Address	Size (MB)	Function
0x00000000	0x0FFFFFFF	256	SDRAM Memory
0x10000000	0x10FFFFFF	16	I/O Devices on APB Bus
0x11000000	0x12FFFFFF	32	Reserved
0x13000000	0x13FFFFFF	16	I/O Devices on AHB Bus
0x14000000	0x1400003F	64B	Static Memory, CS6#
0x14000040	0x14FFFFFF		Reserved
0x15000000	0x1500003F	64B	Static Memory, CS5#
0x15000040	0x15FFFFFF		Reserved
0x16000000	0x1600003F	64B	Static Memory, CS4#
0x16000040	0x16FFFFFF		Reserved
0x17000000	0x1700003F	64B	Static Memory, CS3#
0x17000040	0x17FFFFFF		Reserved
0x18000000	0x1800003F	64B	Static Memory, CS2#
0x18000040	0x19FFFFFF		Reserved
0x1A000000	0x1A00003F	64B	Static Memory, CS1#
0x1A000040	0x1BFFFFFF		Reserved
0x1C000000	0x1FBFFFFF	60	Reserved
0x1FC00000	0x1FC01FFF	0.008	On-chip Boot ROM (8kB)
0x1FC02000	0x1FFFFFFF	3.992	Reserved
0x20000000	0xDFFFFFFF	3072	SDRAM Memory
0xE0000000	0xFFFFFFFF	512	Reserved



NOTES:

1 Data width of static memory banks can be configured to 8, 16 or 32 bits by software.

2 The 8KB address space from H'1FC00000 to H'1FC01FFF in bank 0 is mapped to on-chip boot ROM. The other memory spaces in bank 0 are not used.

3 To support large SDRAM space, EMC re-maps the physical address H'00000000-H07FFFFFF to H'20000000-H'27FFFFFF. Software must configure the SDRAM base address by the re-mapped address.

The JZ4760 processor AHB bus devices are mapped at the addresses based at 0x13000000, and each device is allocated for 64KB space. Table 41-2 lists the complete addresses:

Table 41-2 AHB0 Bus Devices Physical Memory Map

Module	Start Address	End Address	Size (KB)	Description
HARB	0x13000000	0x1300FFFF	64	AHB Bus Arbiter
EMC	0x13010000	0x1301FFFF	64	External SDR Controller
DDRC	0x13020000	0x1302FFFF	64	External DDR Controller
MDMAC	0x13030000	0x1303FFFF	64	Memory copy DMA Controller
LCDC TVE	0x13050000	0x1305FFFF	64	LCD Controller TV Encoder
CIM	0x13060000	0x1306FFFF	64	Camera Interface Module
IPU	0x13080000	0x1308FFFF	64	Image Process Unit

Table 41-3 AHB1 Bus Devices Physical Memory Map

Module	Start Address	End Address	Size (KB)	Description
HARB	0x13200000	0x1320FFFF	64	AHB Bus Arbiter
DMAGP0	0x13210000	0x1321FFFF	64	2D-DMA Controller 0
DMAGP1	0x13220000	0x1322FFFF	64	2D-DMA Controller 1
DMAGP2	0x13230000	0x1323FFFF	64	2D-DMA Controller 2
MC	0x13250000	0x1325FFFF	64	Motion Compensation
ME	0x13260000	0x1326FFFF	64	Motion Estimation
DEBLK	0x13270000	0x1327FFFF	64	De-Block
IDCT	0x13280000	0x1328FFFF	64	Invert DCT for 4x4 block
CABAC	0x13290000	0x1329FFFF	64	CABAC
TCSM0	0x132B0000	0x132BFFFF	64	Tightly coupled sram 0
TCSM1	0x132C0000	0x132CFFFF	64	Tightly coupled sram 1
SRAM	0x132D0000	0x132DFFFF	64	General purpose sram

Table 41-4 AHB2 Bus Devices Physical Memory Map

Module	Start Address	End Address	Size (KB)	Description
HARB	0x13400000	0x1340FFFF	64	AHB Bus Arbiter
NEMC	0x13410000	0x1341FFFF	64	External Normal Memory / Boot ROM / OTP Controller
DMAC	0x13420000	0x1342FFFF	64	DMA Controller
UHC	0x13430000	0x1343FFFF	64	USB 1.1 Host Controller
UDC	0x13440000	0x1344FFFF	64	USB 2.0 Device Controller
BDMAC	0x13450000	0x1345FFFF	64	BCH/NAND DMA Controller
GPS	0x13480000	0x1348FFFF	64	GPS Baseband
ETHC	0x134B0000	0x134BFFFF	64	ETH Controller
BCH	0x134D0000	0x134DFFFF	64	BCH Controller

The JZ4760 processor APB bus devices are based at 0x10000000, and each device is allocated for 4KB space. Table 41-5 lists the complete addresses:

Table 41-5 APB Bus Devices Physical Memory Map

Module	Start Address	End Address	Size (KB)	Description
CPM	0x10000000	0x10000FFF	4	Clocks and Power Manager
INTC	0x10001000	0x10001FFF	4	Interrupt Controller
TCU OST WDT	0x10002000	0x10002FFF	4	Timer/Counter Unit Operating System Timer Watchdog Timer
RTC	0x10003000	0x10003FFF	4	Real-Time Clock
GPIO	0x10010000	0x10010FFF	4	General-Purpose I/O
AIC CODEC	0x10020000	0x10020FFF	4	AC97/I2S/SPDIF Controller Embedded CODEC
MSC0	0x10021000	0x10021FFF	4	MMC/SD 0 Controller
MSC1	0x10022000	0x10022FFF	4	MMC/SD 1 Controller
MSC2	0x10023000	0x10023FFF	4	MMC/SD 2 Controller
UART0	0x10030000	0x10030FFF	4	UART 0

UART1	0x10031000	0x10031FFF	4	UART 1
UART2	0x10032000	0x10032FFF	4	UART 2
UART3	0x10033000	0x10033FFF	4	UART 3
SCC	0x10040000	0x10040FFF	4	Smart Card Controller
SSI0	0x10043000	0x10043FFF	4	Synchronous Serial Interface 0
SSI1	0x10044000	0x10044FFF	4	Synchronous Serial Interface 1
SSI2	0x10045000	0x10045FFF	4	Synchronous Serial Interface 2
I2C0	0x10050000	0x10050FFF	4	I2C 0 Bus Interface
I2C1	0x10051000	0x10051FFF	4	I2C 1 Bus Interface
PS2	0x10060000	0x10060FFF	4	PS/2 Mouse/Keyboard Controller
SADC	0x10070000	0x10070FFF	4	SAR A/D Controller
OWI	0x10072000	0x10072FFF	4	One-Wire Bus Interface
TSSI	0x10073000	0x10073FFF	4	TS Slave Interface