Amlogic U-boot 使用说明

2009.05.19 V0.02.r256

1. U-boot 概述

U-Boot，全称 Universal Boot Loader，是遵循 GPL 条款的开放源码项目。其源码目录、编译形式与 Linux 内核很相似。使用优点：

a) 开放源码；
b) 支持多种嵌入式操作系统内核；
c) 支持多个处理器系列；
d) 较高的可靠性和稳定性；
e) 高度灵活的功能设置，适合 U-Boot 调试、操作系统不同引导要求、产品发布等；
f) 丰富的设备驱动源码，如串口、以太网、SDRAM、FLASH、LCD、NVRAM、EEPROM、RTC、键盘等；
g) 较为丰富的开发调试文档与强大的网络技术支持；

2. U-boot 源码

目前，Amlogic U-boot 是在 U-boot 2009.03 版本基础上移植，代码的 url：https://svn-bj.amlogic.com/svn/model_ae/aml-uboot。

U-Boot 主要目录结构：

- board 目标板相关文件，主要包含 SDRAM、FLASH 驱动，外设 IO 配置等；
- common 独立于处理器体系结构的通用代码，如内存大小探测与故障检测；
- cpu 与处理器相关的文件。如 mpc8xx 子目录下含串口、网口、LCD 驱动及中断初始化等文件；
- driver 通用设备驱动，如 CFI FLASH 驱动（目前对 INTEL FLASH 支持较好）
- doc U-Boot 的说明文档；
- examples 可在 U-Boot 下运行的示例程序；如 hello_world.c,timer.c；
- include U-Boot 头文件；尤其 configs 子目录下与目标板相关的配置头文件是移植过程中经常要修改的文件；
- lib_arc ARC 处理器体系相关的文件；
- net 与网络功能相关的文件目录，如 bootp，nfs，tftp；
- tools 用于创建.bin 和.img 文件等的工具；
- fs 文件系统；

3. 实现功能

支持 CPU：

- Nike

支持目标板：

- dvbc_8218_ts_ref_v1.0

已实现功能：

a) Uart 驱动，包括 Stdin、Stdout、Stderr；
b) SPI Nor flash 读写操作；
c) Nand flash；
d) IPL；
e) Loader；
f) SD/SDHC/MMC Card 驱动；
g) Nike Ethernet 驱动，支持 TFTP、BOOTP、DHCP 等功能；
h) Mkimage Tool；
i) 存取启动参数；
j) FAT 文件系统；

4. 使用步骤

4.1. 编译

编译 u-boot 需要使用交叉编译器 arc-elf32-gcc，编译环境在 linux 下进行。如果是 windows 机器，可以在 cygwin 下进行。交叉编译器及相关工具需另外获取，并要将相关的路径加入到系统环境变量 PATH 中。编译步骤如下：

a) cygwin 下，编译路径指向 U-boot 根目录；
b) 清除以往编译结果：make distclean；
c) 配置目标：make xxx_config
d) 编译链接：make all

4.2. 调试

配置\aml-uboot\trunk\include\configs***.h 文件中的 Uart 通信相关参数，如系统时钟、波特率、校验位、数据位、停止位等。重新编译 u-boot，将目标板与 PC 串口连接。

调试使用“make down”命令将编译生成.out 文件通过 Jtag 下载到目标板 memory 中。运行 u-boot，通信成功建立后，在 PC 端可以使用串口调试工具（如超级终端）调试目标板。

4.3. Flash 烧写

目前支持对 NAND 和 SPI NOR Flash 读写操作。

- 通过串口将文件烧写到 SPI Flash 中。以下操作均在串口调试工具中进行。
  a) 发送命令：loady offset，通过串口 load 二进制文件到 memory 的 offset 处；
  b) 使用 PC 端串口调试工具中的 Send Ymodem file 功能，把需要烧写的 bin 文件发送到目标板的 memory 中；
  c) 发送命令：sf probe 2，探测 SPI Flash；

d) 发送命令：sf erase offset len，对 SPI Flash 相关区域檫写；
e) 发送命令：sf write addr offset len，对 memory 地址 addr（即步骤 a 中的 offset）开始长度 len 的数据写到 SPI Flash 偏移 offset 处开始的位置；

- 从内存中烧录数据到 NAND，写之前必须擦除
  a) nand erase 0 300000, 从 0 地址开始擦除 3MB 空间
  b) nand write 0x1c00000 0 300000, 烧录从 0x1c00000 开始的 3MB 数据到 NAND 的 0 地址
  c) nand read 0x1c00000 0 300000，从 NAND 0 地址读取 3MB 数据到 0x1c00000

4.4. Loader

- Loader：
  - bootelf：从内存中直接启动 ELF 文件
    exg: 下载 ELF 文件到地址 0x200000，输入命令 bootelf 0x200000 即可执行 ELF 格式的 kernel
  - bootm：从内存中启动 uimage
    exg：假设 uimage 位于内存地址 0x1c00000，输入 bootm 0x1c00000 启动 uimage
  - nboot：从 NAND 中 load uimage
    exg：输入 nboot 0x1c00000 0，把 uimage 从 NAND 的 0 地址加载到 0x1c00000

- 启动 uimage 文件的制作

mkimage 工具可以用来制作不压缩或者压缩的可启动映象文件。mkimage 在制作映象文件的时候，是在原来的可执行映象文件的前面加上一个 0x40 字节的头，记录参数所指定的信息，这样 uboot 才能识别这个映象是针对哪个 CPU 体系结构的，哪个 OS 的，哪种类型，加载内存中的哪个位置， 入口点在内存的那个位置以及映象名是什么等。

mkimage –A arch –O os –T type -C compress -a loadaddr –e entrypoint \
-n name –d datafile[:datafile] outputimage

How to Loader Linux Kernel:

1.arc-elf32-objcopy.exe -O binary -R .note -R .comment -S vmlinux vmlinux.bin

2.gzip -9 vmlinux.bin

3.mkimage -A arc -O linux -T kernel -C gzip -a 0x82000000 -e 0x82002000 -n "Linux Kernel Image" -d vmlinux.ok.bin.gz uImage

4.Download uImage to 0x1c00000 or other right place from TFTP or NAND or JTAG

5.bootm 0x1c00000

4.5. 移植

将 U-boot 移植到新的目标板上，主要修改以下几方面内容：

- 在 cpu 目录中添加新的 cpu 的 IPL 启动文件、timer、Interrupt、chipclock 等方面的文件；
- 在\include\asm-arc\下添加新 cpu 的头文件；
- 在\board\aml\下添加新 board 的文件夹，添加与目标板硬件相关操作的源文件。
  mkconfig.mk：CPU 级别基本配置参数；
  u-boot.lds：u-boot 链接命令文件；
  config.mk：TEXT_BASE 定义了 u-boot 自身 text 段的起始地址；
- 在\include\configs\下添加目标板头文件，配置相关参数；
- 根据目标板情况，调整外设驱动；

4.6. 网络连接

以 TFTP 服务器－客户端为例，说明 u-boot 相关的网络应用。局域网内，TFTP 下载速度可达 450KB/s。

a) 首先，在 PC 端安装 TFTP 服务器软件，如 TFTPD32；
b) 设定服务器端软件参数。以 TFTPD32 为例，主要参数有 Base Directory、TFTP Security、TFTP Port、TFTP Timeout 等；
c) 将需要 load 的 image 文件 copy 到 Base Directory 下；
d) 修改\aml-uboot\trunk\include\configs***.h 文件，配置目标板相关网络参数。如 mac、ip、gatewapip、netmask、serverip 等；
e) 重新编译 u-boot；
f) 将目标板连接到 ethernet，启动 u-boot；
g) 使用 TFTP 相关命令（如 tftpboot、bootp、dhcp 等），把 image 文件从 tftp 服务器下载到目标板的 memory 中；
h) 使用 bootm 命令，启动 memory 中的 image 文件；

5. STDIO 常用命令说明（部分）

5.1. Help 操作命令

- help [command ...]
  - 显示对“command”的提示信息；如果“command”为空，则显示所有支持的命令列表；

5.2. SPI Nor 操作命令

- sf probe 2
  - 探测 spi

- sf erase offset len
  - erase len' bytes from offset' ，len 必须是 sector 的倍数，offset 是 spi nor 内的偏移地址；
- sf read addr offset len
  - read len' bytes starting at offset' to memory at `addr'，注意 addr 和 len 要 4 字节对齐；
- sf write addr offset len
  - write len' bytes from memory at addr' to flash at `offset'，注意 addr 和 len 要 4 字节对齐；

5.3. Environment 操作命令

- setenv env value
  - 有的环境变量由几个命令组成如 bootcmd ，则其值前要加 ‘ ’；
- saveenv
  - 保存到 Flash 上；
- printenv
  - 打印 environment；

5.4. Boot 操作命令

- bootelf [address]
  - load 内存地址 address 处的 ELF 文件；
- bootm [addr [arg ...]]
  - 启动在内存 addr 处的 image 文件；如果 addr 为空，则从默认的 loader address 处尝试启动；

5.5. SD/MMC 操作命令

- mmcinfo <dev num>
  - 显示 SD/SDHC/MMC 卡的相关信息；
- mmc read <device num> addr blk# cnt
  - 将 SD/SDHC/MMC 卡设备<device num>从 blk#开始的 cnt 个 block 的内容，读取到内存地址 addr 处；
- mmc write <device num> addr blk# cnt
  - 将内存地址 addr 处开始的内容，写到 SD/SDHC/MMC 卡设备<device num>从 blk#开始的 cnt 个 block 区域；
- mmc rescan <device num>
  - 重新初始化 SD/SDHC/MMC 卡设备<device num>，该命令目前在 amlogic uboot 上被暂时屏蔽；
- mmc list
  - 罗列 SD/SDHC/MMC 类所有链接设备；

5.6. FAT 操作命令

- fatload <interface> <dev[:part]> <addr> <filename> [bytes]
  - 从接口<interface>的设备<dev[:part]>load 二进制文件<filename>到地址<addr>，长度为[bytes]；
- fatls <interface> <dev[:part]> [directory]
  - 显示接口<interface>的设备<dev[:part]>目录[directory]下的文件列表；
- fatinfo <interface> <dev[:part]>
  - 显示接口<interface>的设备<dev[:part]>的 fat 文件系统的信息；

5.7. 网络操作命令

- ping <pingAddress>
  - 向 IP 地址为 pingAddress 的网络节点发送 ping 命令；
- tftpboot [loadAddress] [[hostIPaddr:]bootfilename]
  - 从 IP 为 hostIPaddr 的 TFTP 服务器，下载 bootfilename 文件到目标板的 memory 地址 loadAddress 处；bootfilename 是服务器端 Base Directory 目录的相对路径名；
- bootp [loadAddress] [[hostIPaddr:]bootfilename]
  - 目标板 IP 地址从 BOOTP 服务器动态获取，再从 IP 为 hostIPaddr 的 TFTP 服务器，下载 bootfilename 文件到目标板；
- dhcp [loadAddress] [[hostIPaddr:]bootfilename]
  - 目标板 IP 地址从 DHCP 服务器动态获取，再从 IP 为 hostIPaddr 的 TFTP 服务器，下载 bootfilename 文件到目标板；