

## 07-1 Bootloaders

## Bootloader Challenges

```
#include <stdio.h>

int main(int argc, char **argv) {
    printf("Hello, World!\n");
    return 0;
}
```

## Challenges

- To do
  - DRAM Controller needs initialization
  - May need to copy from Flash to RAM
  - There is no stack
  - Libraries may be needed
  - A context needs to be established
- To where does the processor branch on power up?

u-boot/arch/arm/cpu/armv7/u-boot.lds

```
OUTPUT_FORMAT("elf32-littlearm", "elf32-
littlearm", "elf32-littlearm")
OUTPUT_ARCH(arm)
ENTRY(_start)
SECTIONS
{
    . = 0x00000000;

    . = ALIGN(4);
    .text :
    {
        arch/arm/cpu/armv7/start.o    (.text)
        *(.text)
    }
}
```

u-boot/arch/arm/cpu/armv7/start.S

```
.globl _start
_start:
    b        reset
    ldr      pc, _undefined_instruction
    ldr      pc, _software_interrupt
    ldr      pc, _prefetch_abort
    ldr      pc, _data_abort
    ldr      pc, _not_used
    ldr      pc, _irq
    ldr      pc, _fiq

_undefined_instruction: .word undefined_instruction
_software_interrupt:   .word software_interrupt
...
_pad:                  .word 0x12345678 /* now 16*4=64 */
.global _end_vect
_end_vect:
```

Figure 26-6. 32KB ROM Memory Map

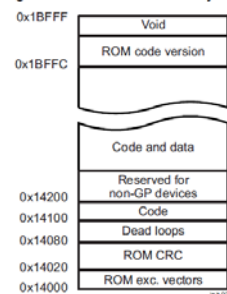


Figure 26-7. 64KB RAM Memory Map of GP Devices

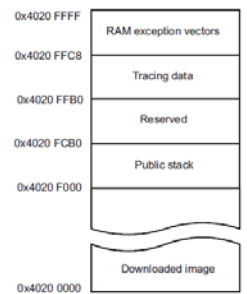


Table 26-7. ROM Exception Vectors

Address	Exception	Content
14000h	Reset	Branch to the public ROM code startup
14004h	Undefined	PC = 4020FFC8h
14008h	Software interrupt (SWI)	PC = 4020FFCCh
1400Ch	Prefetch abort	PC = 4020FFD0h
14010h	Data abort	PC = 4020FFD4h
14014h	Unused	PC = 4020FFD8h
14018h	IRQ	PC = 4020FFDCh
1401Ch	FIQ	PC = 4020FFE0h

Figure 26-3. ROM Memory Map

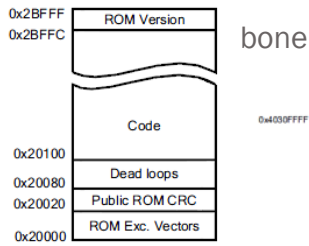


Figure 26-4. Public RAM Memory Map

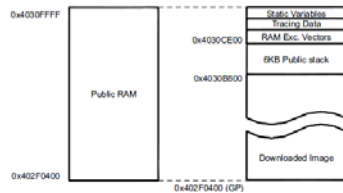
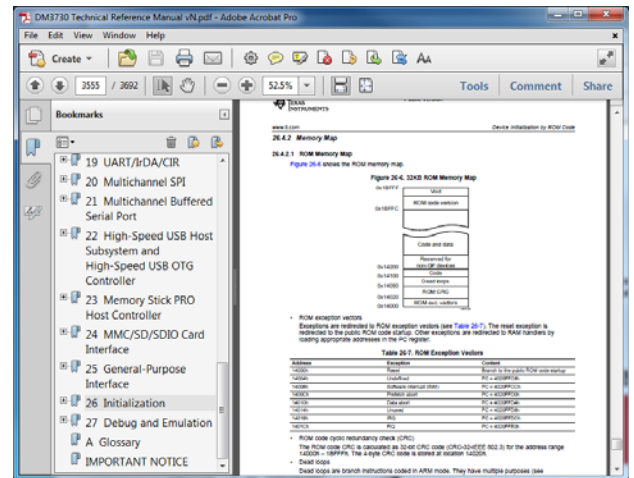


Table 26-1. ROM Exception Vectors

Address	Exception	Content
20000h	Reset	Branch to the Public ROM Code startup
20004h	Undefined	PC = 4030CE04h
20008h	SWI	PC = 4030CE08h
2000Ch	Pre-fetch abort	PC = 4030CE0Ch
20010h	Data abort	PC = 4030CE10h
20014h	Unused	PC = 4030CE14h
20018h	IRQ	PC = 4030CE18h
2001Ch	FIQ	PC = 4030CE1Ch



## u-boot/System.map - xM

```
80e80000 T _start
80e80020 t _undefined_instruction
80e80024 t _software_interrupt
80e80028 t _prefetch_abort
80e8002c t _data_abort
80e80030 t _not_used
80e80034 t _irq
80e80038 t _fiq
80e8003c t _pad
80e80040 T _end_vect
80e80040 t _TEXT_BASE
```

## u-boot/System.map - bone

```
80100000 T _start
80100020 t _undefined_instruction
80100024 t _software_interrupt
80100028 t _prefetch_abort
8010002c t _data_abort
80100030 t _not_used
80100034 t _irq
80100038 t _fiq
8010003c t _pad
80100040 T _TEXT_BASE
```

## The Stack (u-boot/arch/arm/cpu/armv7/start.S)

```
/* Set stackpointer in internal RAM to call board_init_f */
call board_init_f:
    ldr sp, =(CONFIG_SYS_INIT_SP_ADDR)
    bic sp, sp, #7 /* 8-byte alignment for ABI compliance */
    ldr r0,=0x00000000
    bl board_init_f
```

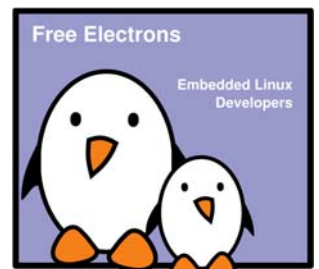
- board\_init\_f is defined in u-boot-arch/arm/lib/board.c

```
• (From include/configs/omap3_beagle.h)
#define CONFIG_SYS_INIT_RAM_ADDR 0x4020f800
#define CONFIG_SYS_INIT_RAM_SIZE 0x800
#define CONFIG_SYS_INIT_SP_ADDR (CONFIG_SYS_INIT_RAM_ADDR + \
    CONFIG_SYS_INIT_RAM_SIZE - \
    GENERATED_GBL_DATA_SIZE)
```

## The U-boot bootloader

### The U-boot bootloader

Michael Odenacker  
Thomas Petazzoni  
Free Electrons



© Copyright 2004-2009, Free Electrons.  
Creative Commons BY-SA 3.0 license  
Latest update: 10/15/2012,  
Document sources, updates and translations:  
<http://www.free-electrons.com/doc/u-boot/>  
Corrections, suggestions, contributions and translations are welcome!

## U-Boot

U-Boot is a typical free software project

- ▶ Freely available at <http://www.denx.de/wiki/U-Boot>
- ▶ Documentation available at <http://www.denx.de/wiki/U-Boot/Documentation>
- ▶ The latest development source code is available in a Git repository: <http://git.denx.de/cgi-bin/gitweb.cgi?p=u-boot.git;a=summary>
- ▶ Development and discussions happen around an open mailing-list <http://lists.denx.de/pipermail/u-boot/>
- ▶ Since the end of 2008, it follows a fixed-interval release schedule. Every two months, a new version is released. Versions are named YYYY.MM.

## Compiling/Installing U-Boot

- ▶ See [http://elinux.org/EBC\\_Exercise\\_22\\_Cross-Compiling\\_and\\_Finding\\_the\\_Right\\_Kernel#Installing\\_a\\_New\\_U-boot](http://elinux.org/EBC_Exercise_22_Cross-Compiling_and_Finding_the_Right_Kernel#Installing_a_New_U-boot)

- ▶ On Host

```
host$ scp u-boot.bin root@beagle:.
```

## Compiling/Installing U-Boot

▶ On the Beagle

```
beagle$ cd /media/
beagle$ mkdir mmcblk0p1
beagle$ mount /dev/mmcblk0p1 mmcblk0p1/
beagle$ ls -ls mmcblk0p1/
total 354
 2 drwxr-xr-x 4 root root 2048 May 16 15:29 Docs
 2 drwxr-xr-x 5 root root 2048 May 16 15:29 Drivers
 6 -rwxr-xr-x 1 root root 5829 Aug 14 10:10 LICENSE.txt
84 -rwxr-xr-x 1 root root 85058 Aug 14 08:19 MLO
14 -rwxr-xr-x 1 root root 13976 Aug 14 10:10 README.htm
 2 -rwxr-xr-x 1 root root 27 Aug 14 10:23 Uenv.txt
 2 -rwxr-xr-x 1 root root 178 Aug 14 10:10 autorun.inf
 2 -rwxr-xr-x 1 root root 171 Aug 14 10:10 info.txt
238 -rwxr-xr-x 1 root root 241948 Aug 14 08:19 u-boot.img
 2 -rwxr-xr-x 1 root root 33 Aug 14 10:23 uEnv.txt.orig
beagle$ cp u-boot.bin /media/mmcblk0p1
beagle$ shutdown -r now
```

## U-boot prompt

- ▶ Power-up the board.

```
U-Boot SPL 2011.09-00053-gb423c52 (Aug 10 2012 - 11:26:55)
Texas Instruments Revision detection unimplemented
No daughter card present
No AC power, disabling frequency switch
OMAP SD/MMC: 0
reading u-boot.img
reading u-boot.img
```

```
U-Boot 2011.09-00053-gb423c52 (Aug 10 2012 - 11:26:55)
I2C: ready
DRAM: 256 MiB
WARNING: Caches not enabled
No daughter card present
NAND: HW ECC Hamming Code selected
No NAND device found!!!
0 MiB
MMC: OMAP SD/MMC: 0, OMAP SD/MMC: 1
*** Warning - readenv() failed, using default environment
```

```
Net: cpsw
Hit any key to stop autoboot: 0
U-Boot#
```

## U-boot prompt

- The U-Boot shell offers a set of commands. We will study the most important ones, see the documentation for a complete reference or the help command.

## Information commands

```
Board information
Flash information
OMAP3 beagleboard.org # bdiinfo
arch_number = 0x0000060A
boot_params = 0x80000100
DRAM bank = 0x00000000
-> start = 0x80000000
-> size = 0x10000000
DRAM bank = 0x00000001
-> start = 0x90000000
-> size = 0x10000000
baudrate = 115200 bps
TLB addr = 0x9FFF0000
relocaddr = 0x9FFF7E000
reloc off = 0x1FF76000
irq_sp = 0x9FF1DF60
sp start = 0x9FF1DF60
FB base = 0x00000000
```

```
u-boot # flinfo
Bank # 1: AM3517 (0x42000000) 16384 (0x4000) 32768 (0x8000)
Size: 256K (0x100000)
Start: 0x42000000
End: 0x42000000
Size: 16K (0x4000)
Start: 0x42000000
End: 0x42000000
```

Can't find

## Environment variables (1)

- U-Boot can be configured through environment variables, which affect the behavior of the different commands
- See the documentation for the complete list of environment variables
- The `printenv` command also to display all variables or one :

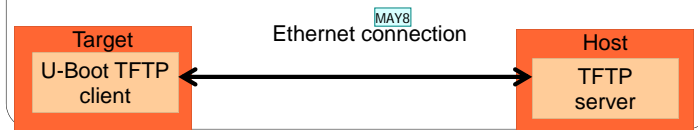
```
autoload=yes
baudrate=115200
bootargs_defaults=setenv bootargs console=${console} ${optargs}
bootcmd=if mmc rescan; then echo SD/MMC found on device ${mmc_dev};if run
loadbootenv; then echo Loaded
environment from ${bootenv};run importbootenv;fi;if test -n $uenvcmd; then echo
Running uenvcmd ...;run
uenvcmd;fi;if run mmc_load_uimage_ext4; then run mmc_args;bootm
${kloadaddr};fi;fi;run nand_boot;
bootdelay=1
bootenv=uEnv.txt
bootfile=uImage
console=ttty0,115200n8
ethact=cpsw
ethaddr=d4:94:a1:39:ed:0c
```

## Environment variables

```
nor_src_addr=0x08080000
rootpath=/export/rootfs
script_addr=0x81900000
spi_args=run bootargs_defaults;setenv bootargs ${bootargs} root=${spi_root}
rootfstype=${spi_root_fs_type}
ip=${ip_method}
spi_boot=echo Booting from spi ...; run spi_args; sf probe ${spi_bus_no}:0; sf
read ${kloadaddr}
${spi_src_addr} ${spi_img_siz}; bootm ${kloadaddr}
spi_bus_no=0
spi_img_siz=0x380000
spi_root=/dev/mtdblock4 rw
spi_root_fs_type=jffs2
spi_src_addr=0x62000
static_ip=${ipaddr}:${serverip}:${gatewayip}:${netmask}:${hostname}::off
stderr=serial
stdin=serial
stdout=serial
Environment size: 2755/8188 bytes
```

## Transferring files to the target

- U-Boot is mostly used to load and boot a kernel image, but it also allows to change the kernel image and the root filesystem stored in flash.
- Files must be exchanged between the target and the development workstation. This is possible:
  - Through the network if the target has an Ethernet connection, and U-Boot contains a driver for the Ethernet chip. If so, the TFTP protocol can be used to exchange files
  - Through the serial line if no Ethernet connection is available.



## U-boot mkimage

```
host$ cd u-boot
host$ file u-boot u-boot.bin
u-boot:      ELF 32-bit LSB executable,
             ARM, version 1 (SYSV),
             statically linked, not
             stripped

u-boot.bin: data

host$ ls -sh u-boot u-boot.bin
1.4M u-boot
332K u-boot.bin
```

## U-boot mkimage

- ▶ The kernel image that U-Boot loads and boots must be prepared, so that an U-Boot specific header is added in front of the image
- ▶ This is done with a tool that comes in U-Boot, **mkimage**
- ▶ Debian / Ubuntu: just install the **uboot-mkimage** package
- ▶ Or, compile it by yourself: simply configure U-Boot for any board of any architecture and compile it. Then install **mkimage**:  
host\$ **cp uboot/tools/mkimage /usr/local/bin/**
- ▶ The special target **ulmage** of the kernel Makefile can then be used to generate a kernel image suitable for U-Boot.

## u-boot/include/configs/omap3\_beagle.h

```
/*
 * High Level Configuration Options
 */

/* This is an ARM V7 CPU core */
#define CONFIG_OMAP 1 /* in a TI OMAP core */
#define CONFIG_OMAP34XX 1 /* which is a 34XX */
#define CONFIG_OMAP3430 1 /* which is in a 3430 */
#define CONFIG_OMAP3_BEAGLE 1 /* working with BEAGLE */

#include <asm/arch/cpu.h> /* get chip and board defs */
#include <asm/arch/omap3.h>

/*
 * Display CPU and Board information
 */
#define CONFIG_DISPLAY_CPUINFO 1
#define CONFIG_DISPLAY_BOARDINFO 1
```

## .../include/configs/omap3\_beagle.h

```
* commands to include */
#include <config_cmd_default.h>

#define CONFIG_CMD_CACHE
#define CONFIG_CMD_EXT2 /* EXT2 Support */
#define CONFIG_CMD_FAT /* FAT support */
#define CONFIG_CMD_JFFS2 /* JFFS2 Support */
...
#undef CONFIG_CMD_FLASH /* flinfo, erase, protect */
#undef CONFIG_CMD_FPGA /* FPGA configuration Support */
#undef CONFIG_CMD_IMI /* iminfo */
#undef CONFIG_CMD_IMLS /* List all found images */
```

## .../include/configs/omap3\_beagle.h

```
#define CONFIG_SYS_NO_FLASH
#define CONFIG_HARD_I2C 1
#define CONFIG_SYS_I2C_SPEED 100000
#define CONFIG_SYS_I2C_SLAVE 1
#define CONFIG_SYS_I2C_BUS 0
#define CONFIG_SYS_I2C_BUS_SELECT 1
#define CONFIG_I2C_MULTI_BUS 1
#define CONFIG_DRIVER_OMAP34XX_I2C 1
#define CONFIG_VIDEO_OMAP3 /* DSS Support */
```

## U-Boot Monitor Commands

- U-Boot supports >70 standard command sets
- More than 150 unique commands
- Enable with CONFIG\_CMD\_\* macros.

Command Set	Commands
CONFIG_CMD_FLASH	Flash memory commands
CONFIG_CMD_MEMORY	Memory dump, fill, copy, compare, and so on
CONFIG_CMD_DHCP	DHCP Support
CONFIG_CMD_PING	Ping support
CONFIG_CMD_EXT2	EXT2 File system support

## U-Boot Monitor Commands

- To enable a specific command, define the macro
- Macros are defined in your board-specific configuration file
- Instead of typing out each individual macro start from the full set of commands defined in

**u-boot/include/config\_cmd\_all.h.**

- List of useful default commands sets

**u-boot/include/config\_cmd\_default.h**

```
$ wc config_cmd_*
 92  567 4181 config_cmd_all.h
 43  237 1673 config_cmd_default.h
 18   45  366 config_cmd_defaults.h
153  849 6220 total
```