The Universal Boot Loader ("Das U-Boot")

Marek Vašut <marex@denx.de>

1. března 2014

Úvod

- U-Boot je program typu 'monitor'
- Licence GPLv2
- Nasazen v mnoha produktech
- Snadno přenositelný a laditelný
- Podporuje mnoho architektur
  ARM, BlackFin, m68k, MIPS, NIOS, PowerPC, Sparc, x86...
- 1000+ zařízení v hlavní řadě
- Poskytuje mnoho funkcí

Historie

- 1999 – 8xx-boot
- 2000 – PPCBoot
- 2002 – ARMBoot
- Listopad 2002 – PPCBoot 2.0.0 aka. U-Boot 0.1.0
- 2002 – podpora pro x86
- 2003 – podpora pro MIPS32, MIPS64, NIOS, m68k
- ...

Uživatelské rozhraní

- CLI, většinou přes seriový port
- Bourne-kompatibilní shell (HUSH z busybox)
- Podporuje skriptování
- Environment obsahuje proměnné a skripty

Uživatelské rozhraní – DEMO

```
U-Boot 2014.01 (Feb 15 2014 - 15:57:04)
CPU:   Freescale i.MX53 rev2.1 at 1200 MHz
Reset cause: POR
Board: DENX M53EVK
I2C:   ready
DRAM:  1 GiB
NAND:  256 MiB
MMC:   FSL_SDHC: 0
In:    serial
Out:   serial
Err:   serial
Net:   FEC
Hit any key to stop autoboot:  0
=>
```

Základní příkazy

- RTFM – `help`
- Memory – `md`, `mw`, `cmp`, `...`, `cp`, `nm`
- Environment – `env`, `run`
- Další – `reset`, `sleep`, `bdinfo`, `coninfo`

Memory

Vhodné na zkoumání registrů

Ne na každý paměťový region lze přistupovat libovolně

- `md [.b, .w, .l] address [# of objects]`
- `mw [.b, .w, .l] address value [count]`
- `cmp [.b, .w, .l] addr1 addr2 [count]`

```
=> md 0x74000000 0x2
74000000: ffffffff ffffffff                                    ........
=> md.w 0x74000000 0x2
74000000: ffff ffff                                                ....
=> mw 0x71000000 0x12345678 0x10
=> mw 0x72000000 0x12345678 0x8
=> cmp.b 0x71000000 0x72000000 0x40
byte at 0x71000020 (0x78) != byte at 0x72000020 (0xff)
Total of 32 byte(s) were the same
```

Environment

```
=> help env
env - environment handling commands

Usage:
env ask name [message] [size] - ask for environment variable
env default [-f] -a - [forcibly] reset default environment
env default [-f] var [...] - [forcibly] reset variable(s) to the
env delete [-f] var [...] - [forcibly] delete variable(s)
env edit name - edit environment variable
env export [-t | -b | -c] [-s size] addr [var ...] - export envi
env grep [-e] [-n | -v | -b] string [...] - search environment
env import [-d] [-t | -b | -c] addr [size] - import environment
env print [-a | name ...] - print environment
env run var [...] - run commands in an environment variable
env save - save environment
env set [-f] name [arg ...]
```

Environment 2

```
=> env set foo bar baz
=> env set quux 'echo $foo'
=> env print foo
foo=bar baz
=> env print quux
quux=echo $foo
=> run quux
bar baz
```

Další příkazy

```
=> bdinfo
arch_number = 0xFFFFFFFF
boot_params = 0x70000100
DRAM bank   = 0x00000000
-> start    = 0x70000000
-> size     = 0x20000000
DRAM bank   = 0x00000001
-> start    = 0xB0000000
-> size     = 0x20000000
eth0name    = FEC
ethaddr     = c0:e5:4e:XX:XX:XX
current eth = FEC
ip_addr     = 192.168.1.10
baudrate    = 115200 bps
TLB addr    = 0xAFFF0000
relocaddr   = 0xAFF26000
reloc off   = 0x3EF26000
irq_sp      = 0xAF521F28
sp start    = 0xAF521F18
```

Načtení souboru z lokálního média

Init média:

- SD/MMC: `mmc rescan`
- USB: `usb reset`
- SATA: `sata init`
- IDE: `ide reset`

Načtení souboru:

```
<command> <interface> <dev[:part]> [addr] [filename]
```

- command: `ext2load, ext4load, fatload`
- interface: `mmc, usb, ide, sata`

Načtení souboru ze síťě

Nastavení síťě:

```
=> env set ipaddr 192.168.1.310
=> env set netmask 255.255.255.0
=> env set serverip 192.168.1.301
=> env set gatewayip 192.168.1.354
```

ICMP:

```
=> ping ${serverip}
```

TFTP:

```
=> tftp ${loadaddr} ${serverip}:path/to/file
=> tftp path/to/file
```

Formát jádra

U-Boot podporuje mnoho formátů jádra:

- ELF – `bootelf`
- zImage – `bootz`
- uImage – `bootm`
- *fitImage* – `bootm`

FDT a Ramdisk:

```
bootm <kernel> <ramdisk> <dtb>
```

Výsledek – DEMO

```
=> tftp ${loadaddr} fitImage ; bootm ${loadaddr}
Using FEC device
TFTP from server 192.168.1.301; our IP address is 192.168.1.310
Filename 'fitImage'.
Load address: 0x70800000
Loading: #################################################
...
## Loading kernel from FIT Image at 72000000 ...
   Using 'conf@1' configuration
   Trying 'kernel@1' kernel subimage
     Description:  Linux kernel
...
   Verifying Hash Integrity ... sha1+ OK
   Booting using the fdt blob at 0x723275d8
   Loading Kernel Image ... OK
   Loading Device Tree to 8fff8000, end 8ffffd6d ... OK

Starting kernel ...
[    0.000000] Booting Linux on physical CPU 0x0
```

Vývoj

- GIT: `git://git.denx.de/u-boot.git`
- ML: `u-boot@lists.denx.de`

Děkuji za pozornost, Q/A?