Lab #8 Intro U-Boot

Leonardo Fusser (1946995)

Objectives:

- Learn basic U-Boot commands

Hardware: BBB, usb key, FTDI TTL-232R-3V3,

To hand in:

- Answers to all questions
- An introduction
- A conclusion
- Appropriate screenshots

Theory: Manual: Beagle_board_black_BBB_SRM.pdf. https://cdn-shop.adafruit.com/datasheets/BBB_SRM.pdf

Reference: Beaglebone Essentials p.46

Lab work:

All questions must be answered in lab in a legible manner.

Part 1: U-boot basic commands:

Plug in both cables (usb and USB to serial)
Connect on both.
Reboot (shutdown –r now from the SSH connection) go the serial window and press a key (this must be done quickly.

If U-Boot is stopped in time, it displays a command line like this: =>

Built-in commands

Type the following command and describe the result: if you discover a command is not complete… give the complete command.

Help:

Help command. Lists all commands you can use.

COM6 - PuTTY

```
=> help
?       - alias for 'help'
askenv  - get environment variables from stdin
base    - print or set address offset
bdinfo  - print Board Info structure
boot    - boot default, i.e., run 'bootcmd'
bootd   - boot default, i.e., run 'bootcmd'
bootm   - boot application image from memory
bootp   - boot image via network using BOOTP/TFTP protocol
bootz   - boot Linux zImage image from memory
cmp     - memory compare
coninfo - print console devices and information
cp      - memory copy
crc32   - checksum calculation
dfu     - Device Firmware Upgrade
dhcp    - boot image via network using DHCP/TFTP protocol
dm      - Driver model low level access
echo    - echo args to console
editenv - edit environment variable
eeprom  - EEPROM sub-system
env     - environment handling commands
exit    - exit script
ext2load- load binary file from a Ext2 filesystem
ext2ls  - list files in a directory (default /)
```

env:

Environment handling command. Shows all the options that can be used with the “env” command.

```
=> 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 their default value
s
env delete [-f] var [...] - [forcibly] delete variable(s)
env edit name - edit environment variable
env exists name - tests for existence of variable
env export [-t | -b | -c] [-s size] addr [var ...] - export environment
env import [-d] [-t [-r] | -b | -c] addr [size] - import environment
env print [-a | name ...] - print environment
env run var [...] - run commands in an environment variable
env set [-f] name [arg ...]

=>
```

`printenv (or print):`

Prints all the environmental variables on the system.

```
=> printenv
arch=arm
args_mmc=run finduuid;setenv bootargs console=${console} ${optargs} ${cape_disab
le} ${cape_enable} root=PARTUUID=${uuid} ro rootfstype=${mmcrootfstype} ${cmdlin
e}
args_mmc_old=setenv bootargs console=${console} ${optargs} ${cape_disable} ${cap
e_enable} root=${oldroot} ro rootfstype=${mmcrootfstype} ${cmdline}
args_mmc_uuid=setenv bootargs console=${console} ${optargs} ${cape_disable} ${ca
pe_enable} root=UUID=${uuid} ro rootfstype=${mmcrootfstype} ${cmdline}
args_netinstall=setenv bootargs ${netinstall_bootargs} ${optargs} ${cape_disable
} ${cape_enable} root=/dev/ram rw ${cmdline}
args_uenv_root=setenv bootargs console=${console} ${optargs} ${cape_disable} ${c
ape_enable} root=${uenv_root} ro rootfstype=${mmcrootfstype} ${cmdline}
autoconf=off
baudrate=115200
board=am335x
board_name=A335BNLT
board_rev=000C
boot=${interface} dev ${mmcdev}; if ${interface} rescan; then gpio set 54;setenv
 bootpart ${mmcdev}:1; if test -e ${interface} ${bootpart} /etc/fstab; then sete
nv mmcpart 1;fi; echo Checking for: /uEnv.txt ...;if test -e ${interface} ${boot
part} /uEnv.txt; then if run loadbootenv; then gpio set 55;echo Loaded environme
nt from ${bootenv};run importbootenv;fi;if test -n ${cape}; then if test -e ${in
```

`sleep:`

Sleep command. Used to add certain amount of delay for execution.

```
=> sleep
sleep - delay execution for some time

Usage:
sleep N
    - delay execution for N seconds (N is _decimal_ !!!)
=> sleep 5
=>
```

Part 2: Managing storing devices:

`mmc part`: Give the part number

Displays partition layout and scheme for MMC device 1 (eMMC).

```
=> mmc part

Partition Map for MMC device 1  --   Partition Type: DOS

Part    Start Sector    Num Sectors     UUID            Type
  1     2048            196608          00000000-01     0e Boot
  2     198656          7272448         00000000-02     83
=>
```

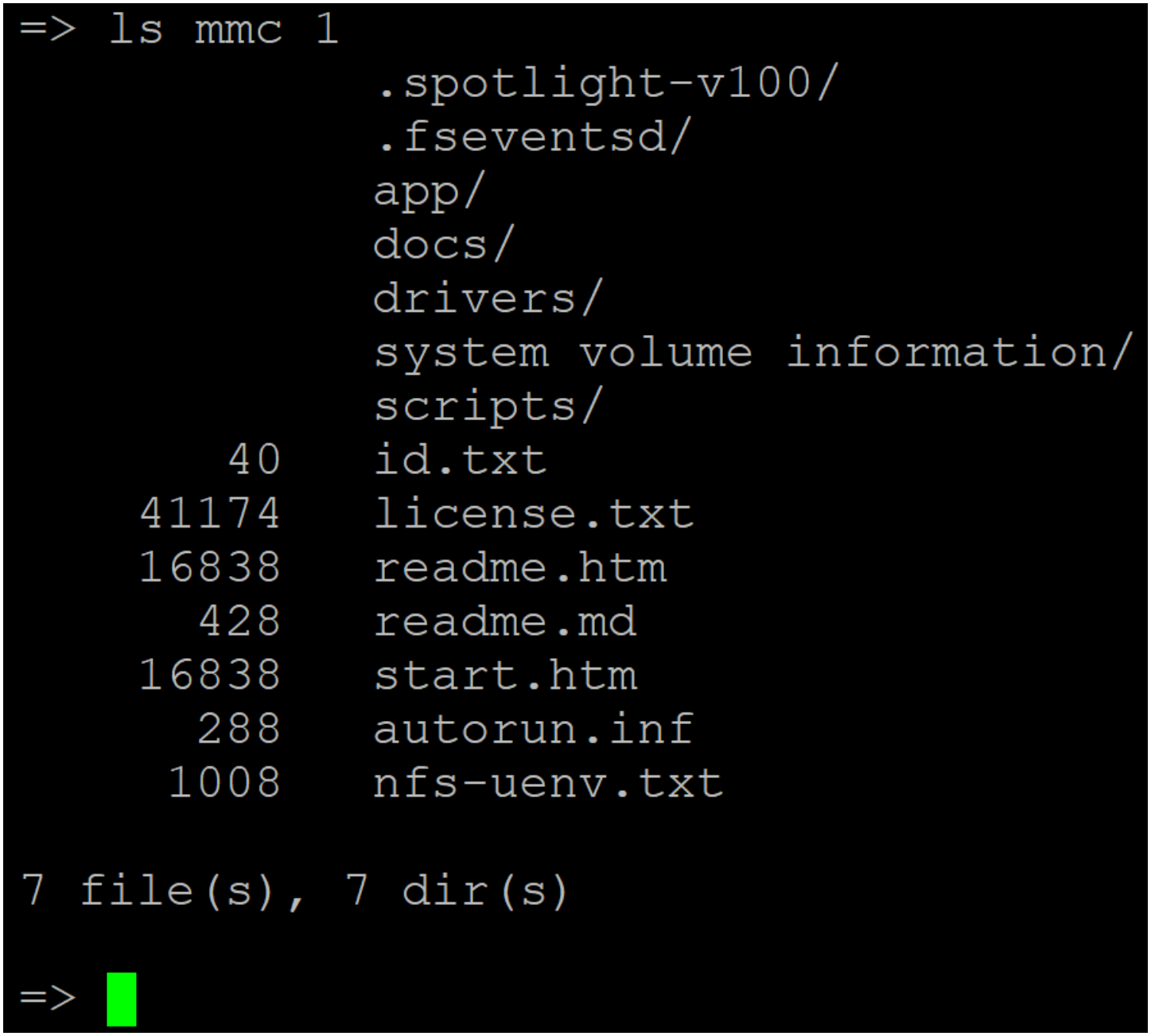
`mmc info`. Give the capacity

Displays detailed information about MMC device 1 (eMMC).

```
=> mmc info
Device: OMAP SD/MMC
Manufacturer ID: 70
OEM: 100
Name: M6270
Tran Speed: 52000000
Rd Block Len: 512
MMC version 4.5
High Capacity: Yes
Capacity: 3.6 GiB
Bus Width: 4-bit
Erase Group Size: 512 KiB
HC WP Group Size: 4 MiB
User Capacity: 3.6 GiB
Boot Capacity: 2 MiB ENH
RPMB Capacity: 512 KiB ENH
=>
```

`ls mmc 1`

Displays the content stored on the MMC device 1 (eMMC).



Insert a USB key:

```
usb start
usb dev 0
```

usb start -> starts the USB process and scans for USB devices plugged in on the system.
usb dev 0 -> displays detailed information about USB device 0.

```
=> usb start
starting USB...
USB0:   scanning bus 0 for devices... 1 USB Device(s) found
       scanning usb for storage devices... 1 Storage Device(s) found
=> usb dev 0

USB device 0:
    Device 0: Vendor: Verbatim Rev: 8.07 Prod: STORE N GO
            Type: Removable Hard Disk
            Capacity: 14860.0 MB = 14.5 GB (30433280 x 512)
... is now current device
=>
```

Read the content by typing:

```
ls usb 0
```

Displays the content stored on USB device 0.

```
=> ls usb 0
                system volume information/
     22625      scope_2.png
     22594      scope_3.png
     20384      scope_5.png
     20316      scope_6.png

4 file(s), 1 dir(s)

=>
```

Part 3: Kernel command lines:

- Write a command to turn gpio 55 on.

Use `gpio 55.` Hint: `help gpio`

gpio set 55 -> turns on USRLED2 on BBB.

```
=> help gpio
gpio - query and control gpio pins

Usage:
gpio <input|set|clear|toggle> <pin>
    - input/set/clear/toggle the specified pin
gpio status [-a] [<bank> | <pin>]  - show [all/claimed] GPIOs
=> gpio set 55
gpio: pin 55 (gpio 55) value is 1
=>
```

Set and run an environmental variable that will turn LED gpio 55 on for 2 seconds.

Turns USRLED2 on for 2 seconds then turns off.

```
=> setenv delay 2
=> setenv USR2ONdelay 'gpio set 55; sleep ${delay}; gpio clear 55 ; sleep ${dela
y}'
=> run USR2ONdelay
gpio: pin 55 (gpio 55) value is 1
gpio: pin 55 (gpio 55) value is 0
=>
```

- Set and run a script that to toggle two LEDs on the board with the frequency of 1Hz, such that if one is on the other one is off.

Toggles 2 LEDs (USRLED2 & USRLED1) on and off.

```
=> setenv LEDscript 'while sleep 1 ; do gpio toggle 55 ; gpio toggle 54 ; done'
=> run LEDscript
gpio: pin 55 (gpio 55) value is 1
gpio: pin 54 (gpio 54) value is 0
gpio: pin 55 (gpio 55) value is 0
gpio: pin 54 (gpio 54) value is 1
gpio: pin 55 (gpio 55) value is 1
gpio: pin 54 (gpio 54) value is 0
gpio: pin 55 (gpio 55) value is 0
gpio: pin 54 (gpio 54) value is 1
gpio: pin 55 (gpio 55) value is 1
```

Give a demo to the teacher.

- Boot:

Exits u-boot and starts up BBB for normal operation.