Writeup 3

CS 444
November 10, 2018

Anton Synytsia, Eytan Brodsky, David Jansen

Contents

I. Required Technologies

The following technologies are required to perform this lab:

1) Raspberry Pi, preferably model 3B+,
2) Micro SD card and card reader, with at least 4GB space.
3) 3.3V TTL UART to USB converter for establishing serial console.
4) Power adapter or charger for Raspberry Pi.
5) Access to `OS2` server.

II. Setup

The following sections describe how to set up Raspbian Stretch Lite on Raspberry Pi. The first section describes how to setup Raspbian on SD card, the second section focuses on testing Raspbian with serial console, the third section describes how to build Raspberry Pi Linux kernel on `OS2`, and the fourth section describes how to setup a built Linux kernel image on SD card.

A. Downloading and Setting Up Raspbian

Refer to the following steps for setting up Raspbian Stretch Lite on SD card:

1) Download and extract `2018-10-09-raspbian-stretch-lite.zip` from https://www.raspberrypi.org/downloads/raspbian/.
2) Download and install Etcher from https://www.balena.io/etcher/.
3) Mount the micro SD card to your laptop, via the SD card reader.
4) Start Etcher and do the following:
   a) Set image to `2018-10-09-raspbian-stretch-lite.img` (or alike).
   b) Set drive to SD card.
   c) Click textitFlash!
5) Once the setup is complete, navigate to the SD card drive and use text editor to append the following to `config.txt`:

```
kernel=kernel8.img
enable_uart=1
```

B. Running Raspbian with Serial Console

The following steps, heavily based on Adafruit guide, describe how to initiate a Raspbian serial console session:

1) Install Prolific Chipset and SiLabs CP210X drivers for the TTL serial cable [1].
2) Connect black, white, and green wires to the outer pins 3, 4, and 5 respectively [2].
3) Leave red wire unpinned, as the a separate power adapter is used instead [2]. It is important that only one power source is used as the board can get damaged [2].
4) Insert the micro SD card into Raspberry Pi.
5) Insert the TTL serial cable USB into your laptop.

6) Start Putty and do the following:
   a) Set *Connection type* to serial mode.
   b) Set *Serial line* to `COM6`; COM6 here refers to the port of our TTL serial cable. To determine the port of your TTL serial cable, on Windows platform, access *Device Manager* and check for the available ports; for other platforms, refer to Adafruit guide [2].
   c) Set *Speed* to 115200 [2].
   d) Click *Open*.
7) Connect the power adapter to Raspberry Pi. It is important that this step is performed after initiating the serial console session.
8) (Optional) Within the serial console, press *RETURN* key to activate communications [2].
9) After loading, use `pi` as user name and `raspberry` as password.

C. Raspberry Pi Linux Kernel

1) Setting Up: Perform the following steps for downloading and setting up version 4.14.y Raspberry Pi Linux kernel on `OS2`:

```
cd /scratch/fall2018/group1
git clone git@github.com:raspberrypi/linux.git
cd linux
# checkout v4.14.y
git checkout tags/raspberrypi-kernel_1.20180417-1
```

2) Compiling: Execute the following set of commands to build Raspberry Pi Linux Kernel on `OS2`:

```
cd linux
KERNEL=kernel8
make -j4 ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- bcmrpi3_defconfig
make -j4 ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- all
```

Once built, refer to the next section for setting up the built image on SD card.

3) Uploading to SD Card: Refer to the following steps for setting up `kernel8.img` on SD card:

1) Download `Image` from `arch/arm64/boot/Image` to your local file system, either using WinCP or another file transfer protocol.
2) Rename `Image` to `kernel8.img`.
3) Mount SD card to your laptop.
4) Copy `kernel8.img` to SD card, so that it is located at the same path as `kernel7.img`.

III. Morse Code LED Trigger

The following sections describe our solution to Raspberry Pi Morse code LED trigger, as well as, instructions for compiling, setting up, and running the blinker.

A. *Solution*

B. *Compiling*

Perform the following steps to compile Raspberry Pi with our Morse code LED trigger:

1) Provided that Raspberry Pi Linux Kernel is cloned and checked out to the correct version at your local space, on `OS2`, copy `linux` folder, shipped with this repository, to your `linux` folder. This will overwrite and add the following files to `linux/drivers/leds/trigger/`:

   ledtrig-morse.c Our Morse code LED trigger.

   Kconfig Configures our Morse code LED trigger.

   Makefile Registers our Morse code LED trigger.

2) Executed the following set of commands to rebuild the kernel:

```
cd linux
make clean
KERNEL=kernel8
make -j4 ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- bcmrpi3_defconfig
make -j4 ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- all
```

C. *Setting Up*

Upload the new, built kernel image to SD card, described in II-C3 section. You may have to delete the original `kernel8.img` from the SD card first though.

D. *Running*

Start a new serial console session, described in II-B section. Then run the following set of commands to activate the Morse code LED trigger:

```
sudo su
echo none > /sys/class/leds/led0/trigger # optional
echo morse > /sys/class/leds/led1/trigger
```

IV. Version Control

acronym	**meaning**
V	`version`
tag	`git tag`
MF	Number of `modified files`.
AL	Number of `added lines`.
DL	Number of `deleted lines`.

V	tag	date	commit message	MF	AL	DL
1		2018-10-08	Initial commit	1	1	0
2		2018-10-08	Create .gitignore	1	6	0
3		2018-10-08	Update state	1	0	1
4		2018-10-08	Setup readme	1	5	0
5		2018-10-08	Update README.md	1	2	0
6		2018-10-08	Uploaded concurrency	2	225	0
7		2018-10-08	changed permissions	4	0	0
8		2018-10-09	Setup writeup TeX	2	126	0
9		2018-10-09	Added pygments	1	98	0
10		2018-10-09	included pygements to preamble	1	8	6
11		2018-10-09	First sucessful compile of writeup1	6	2008	2
12		2018-10-09	Updated gitignore	5	3	2005
13		2018-10-09	Setup Writeup	1	10	20
14		2018-10-09	Added git attributes to enforce EOL	1	7	0
15		2018-10-09	EOL deal	2	1	2
16		2018-10-09	Trying to fix TEX compiling	2	79	1
17		2018-10-09	Added sections to writeup	3	8	81
18		2018-10-09	Added the writeup for Command Logs and for the Qemu flags	11	97	10
19		2018-10-10	Work on concurrency writeup	8	30	56
20		2018-10-10	More work on Concurrency writeup	2	12	7
21		2018-10-10	Made Concurrency use circular queue	1	41	34
22		2018-10-10	Fixed concurrency print type	1	9	6
23		2018-10-10	Renamed Assignment1 folder to Concurency and finished my writeup	7	251	244
24		2018-10-11	Disabled BIB command in makefile	2	3	3
25		2018-10-11	Added compile instructions	1	26	3
26		2018-10-11	README adjustments	1	3	3
27		2018-10-13	corrected name	1	1	1
28		2018-10-13	explain -localtime and -append root=/dev/vda rw console=ttyS0 debug	1	2	2
29		2018-10-13	add work log.	1	6	0
30		2018-10-13	Added the version control table	1	31	1
31		2018-10-13	Changed up how the lists were made	1	48	31

Please continue at the next page

V	tag	date	commit message	MF	AL	DL
32		2018-10-15	Refactored rdrand to use X86 intrinsic and created a general rand function	1	88	69
33		2018-10-15	Updated concurrency writeup to reflect the refactored code.	1	7	10
34		2018-10-15	Updated work log	5	477	77
35		2018-10-15	Removed thirdparty	1	0	362
36		2018-10-22	Added concurrency2	2	219	0
37		2018-10-23	Setup writeup2 and update comments in concurrency	5	233	4
38		2018-10-27	Unignored Linux Yocto with our sched	7	2	1
39		2018-10-27	Unignoring yocto	1	0	2
40		2018-10-27	yocto...	1	1	0
41		2018-10-27	Added clook scheduler	4	582	2
42		2018-10-27	Setup Writeup2	11	9310	501
43		2018-10-27	Updated makefiles	3	8	9
44		2018-10-27	Making refs work	1	1	1
45		2018-10-27	Added Eyton's scheduler tester	2	16	0
46		2018-10-27	Added David's makefile	2	27	1
47		2018-10-28	Fixed compile errors in CLOOK and modifed Kconfig for inclusion	4	84	6
48		2018-10-28	Added command for starting VM	1	5	0
49		2018-10-29	Made clook print if unexpected access order	2	8	0
50		2018-10-29	Clook Scheduler Writeup	11	165	8
51		2018-10-29	Fixes to README	1	7	7
52		2018-10-29	Version control log	1	79	0
53		2018-10-29	More work on Writeup2	18	101	91
54		2018-10-29	Updated makefiles and writeup for assign 1 and 2	13	69	9021
55		2018-10-29	Modified assignment 1 make	2	9	3
56		2018-10-29	Added the Concurrency Section 2	1	35	0
57		2018-10-29	Changed code styling	1	2	2
58		2018-10-29	Update writeup2.tex	1	21	0
59		2018-10-29	Minted Python Testing Script	1	6	7
60		2018-10-29	Update writeup2.tex	1	1	14
61		2018-10-30	Added raspberry linux to ignore file	10	1	1
62		2018-10-30	Started Writeup3	5	175	0
63		2018-11-07	Update README.md	1	4	0

Please continue at the next page

V	tag	date	commit message	MF	AL	DL
64		2018-11-07	Created writeup for Downloading Raspberry Pi Linux Kernel, Preparing SD Card, and Testing Raspbian.	4	74	10
65		2018-11-07	Updated makefiles so that PDFs are copied to resulting folder	7	4	1
66		2018-11-08	Completed Part 1 for Writeup	3	52	21
67		2018-11-08	updated compress / extract commands in README	1	3	8
68		2018-11-10	Work on Writeup3	2	16	4
69		2018-11-10	Uploading modifid Linux files	7	712	0
70		2018-11-10	Finalized morse code trigger	2	38	34
71		2018-11-10	Writeup for compiling and running Morse code LED trigger	3	61	35
72		2018-11-10	Recompiled writeup3	1	0	0
73		2018-11-10	Fixed references	1	2	2
74		2018-11-10	Fixed formatting	2	11	17
75		2018-11-10	Updated PDF	1	0	0

References

[1] S. Monk, “Software installation,” https://learn.adafruit.com/adafruits-raspberry-pi-lesson-5-using-a-console-cable/software-installation-mac, Nov 2018.

[2] ——, “Adafruit raspberry pi lesson 5. using a console cable,” https://learn.adafruit.com/adafruits-raspberry-pi-lesson-5-using-a-console-cable/overvi
Nov 2018.