PROJECT #2 GUIDE

UCB ECEN 5803 FALL 2025 PROJECT #2: RASPBERRY PI 4 MODEL B QUAD ARM CORTEX A72 WITH QNX AND LINUX

TABLE OF CONTENTS

I. MODULE 1 INSTALL QNX SDP AND MOMENTICS IDE

1. Follow the directions in QNXInstallation.pdf to install the QNX Software Center, QNX SDP, and Momentics IDE on your PC. You may also want to add the QNX Toolkit for Visual Studio Code, which can also be used as your IDE. You will need to configure the installation so that it can compile code for your ARM target processor on the Raspberry Pi 4 Model B.
2. To prepare for code development for QNX, watch the first 20 minutes of each of the following videos: https://www.youtube.com/watch?v=y42V_7ZTa-s You do not need the display shown, any HDMI monitor will do, including ones in the Lab.
https://www.youtube.com/watch?v=s8_rvkSfj10 Introduction to QNX Tools.
Booting QNX on the Rpi. https://gitlab.com/qnx/quick-start-images See also
https://devblog.qnx.com/set-up-a-headless-qnx-raspberry-pi

II. MODULE 2 BOOT QNX, CREATE G.711 CODEC

1. To test QNX, you must download a target image and install it in a micro SD card. Download the image as using the QNX Software Center directed from here:
https://www.qnx.com/developers/docs/qnxeverywhere/com.qnx.doc.qnxeverywhere/topic/qsti/intro.html or get the image file .img from Canvas. Follow the instructions here
https://www.qnx.com/developers/docs/qnxeverywhere/com.qnx.doc.qnxeverywhere/topic/qsti/install.html and here
https://www.qnx.com/developers/docs/qnxeverywhere/com.qnx.doc.qnxeverywhere/topic/qsti/interacting-with-the-system.html to install the QNX starter target image for Raspberry Pi on your SD card. Please be aware that QNX for ARM is still in early stages of development, so some things may not work as you expect. Do not spend more than a couple of hours trying to make it work before reaching out for help. A

recommended SD Card is SanDisk Ultra Micro SDHC 16GB . Use the Raspberry Pi imager to flash the image to your SD Card. Once completed, edit the network file on the SD card to configure WiFi details before removing the SD Card from you SD Card writer.

Also refer to these helpful sites:
QNX on Pi Introduction: https://gitlab.com/elahav/qnx-rpi-book
Getting started with the QNX8 Raspberry Pi4 BSP: https://www.rtts.com.au/qnx/rpi4_bsp.php
https://www.raspberrypi.org/documentation/
https://www.raspberrypi.org/documentation/hardware/raspberrypi/bootmodes/bootflow.md
https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up
https://www.raspberrypi.org/documentation/installation/installing-images/windows.md
https://elinux.org/RPi_Easy_SD_Card_Setup

2. Insert your SD Card into the Raspberry Pi; connect an HDMI display to the HDMI 0 port, and a keyboard and mouse to the USB 2.0 ports, and power it on. On first boot, you should briefly see a colorful splash screen on the display, which indicates that the image successfully booted. If the image doesn't boot properly, you will get a Raspberry Pi boot message. Next, you will be greeted by the QNX Welcome screen. This indicates that Screen started successfully, and the Pi is ready to run graphical apps. When you're ready, click the icon in the center to launch the terminal application.
3. At the terminal, use **qnxuser** as the username and password. Capture a screen shot of the terminal window.
4. Connect a serial cable (available from the e-store or ITLL) between your Raspberry Pi and your PC as shown here: https://www.qnx.com/developers/docs/qnxeverywhere/com.qnx.doc.qnxeverywhere/topic/qsti/interacting-with-the-system.html . Reboot your Raspberry Pi. Upon booting up, the serial port should be sending out debug messages. Open a terminal window in PuTTY or TeraTerm to capture them. What do you see?
5. How much memory is used by the code? (What is the image size?) Look in the guides given to help find how you determine this.
6. The Ethernet port should also be active. Connect using SSH to see the GUI window. Reboot the system – what do you see?
7. Determine how to check your Raspberry Pi's thermal temperature. Capture a screenshot of what it reports.
8. Write C code for a G.711 coder/decoder. See http://www.opensource.apple.com/source/tcl/tcl-20/tcl_ext/snack/snack/generic/g711.c or for an example. Use this decoder to decode a file given to you by your instructor. You will need to use the Momentics IDE or VS Code to compile code for this application and then run it on the target board. Alternatively, you could also do this in Linux using gcc on the target board.
9. Record your observations. How is the behavior of QNX different from Linux?

III. MODULE 3 SCRIPTING WITH LINUX

Aim: Creation and running an executable script in Linux to print useful information about the Linux Kernel.

Scope: The function implemented by the script is under the discretion of the group. The complexity and number of functions implemented is also under the discretion of the group.

Setup: to create your Raspbian Linux development environment, follow the directions given in Practical Homework 3

Procedure:
Creating an executable script:
i) Use a suitable scripting language for writing a script that uses commands in the terminal for access data relevant to the terminal. Scripts can be written in bash or python or something similar. Information for bash can be obtained from http://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.html and http://www.freeos.com/guides/lsst/. An introduction to python can be found here: https://docs.python.org/2/tutorial/ and https://www.linuxjournal.com/article/1121 .

ii) Convert the script to an executable file.

iii) An additional 2 bonus points will be awarded if the script is executed at boot time.

Example of what the executable script may perform
- display the current running processes.
- display the current data and time, kernel version
- display the kernel dump.

The points on this exercise will vary on two factors:

A. Number of functions implemented by the script. (Minimum: 3)
B. Complexity of the functions implemented by the script. (A function is not complex if there is 1 exclusive command that performs that function).

IV. MODULE 4 BUILD YOUR OWN PBX WITH ASTERISK

1. For development of code in Linux, GCC and GDB are the preferred compiler and debugger to use. Go to www.asterisk.org and https://www.asterisk.org/downloads and download Asterisk. Look at https://wiki.asterisk.org/wiki/display/AST/Beginning+Asterisk and read the sections on Beginning Asterisk, Installing Asterisk, and the Hello World Project. Use either the Angstrom package repository to install the asterisk package directly or after connecting the Raspberry Pi 4 Model B to the internet under Linux. Carefully read the documentation and online guides and incorporate Asterisk, the open source PBX into one of your Linux SD cards. How much memory is used by the code? (What is the image size?)
2. Using either a SIP phone plugged into the same LAN as the Raspberry Pi 4 Model B (you may need an Ethernet switch to create the network), or with a PC running a softphone application connected to the Raspberry Pi 4 Model B, configure Asterisk to provide a voicemail message at extension 100. Configure your SIP phone or softphone and register with Asterisk. Show your Asterisk setup in a screenshot.
3. Make a call to extension 100 and record what you hear. Show your Asterisk setup in a screenshot.
4. [Optional, for 3 extra credit points] Add another SIP phone or softphone to the network, and make a phone to phone call.

V. MODULE 5 BUILD A QNX TELECOM OR IOT APPLICATION (OPTIONAL, FOR 10 POINTS EXTRA CREDIT)

1. Using QNX on the Raspberry Pi 4 Model B, or on a VM on your laptop, create an application related to some telecom or IoT function. You can start with one of the test programs included in the BSP. Some possible applications:
   Any Mobile Phone apps, like SMS messaging, voicemail, or contact manager;
   A Morse Code Receiver that decodes and displays incoming Morse Code generated by a switch on a small microcontroller board;
   An IoT sensor aggregation application that uses SPI, I2C, and/or GPIO sensors and packetizes the information for Ethernet Access;
   A Web server that provides a home webpage for your Raspberry Pi 4 Model B. At a minimum it should show the time and number of accesses of the page.
2. Build your application, and any associated test hardware or interfaces, and test your new application.
3. Document your new application by providing a zip file of the code and screenshots showing the application in operation. Be prepared to demo this application to the TAs.

VI. MODULE 6 MEASURE CORTEX-A72 DMIPS UNDER QNX AND LINUX (OPT., UP TO 5 POINTS EXTRA CREDIT)

1. Port your DMIPs benchmark code from Project 1 and compile and run it under Linux on the Raspberry Pi 4 Model B. Record you DMIPs numbers using 1, 2, or 4 cores.
2. Port your DMIPs benchmark code from Project 2 and compile and run it under QNX on the Raspberry Pi 4 Model B. Does it match your expectations? How does it compare to the Linux implementation – do you get the same performance numbers?

VII. MODULE 7 SCRIPTING WITH QNX (OPT., UP TO 5 POINTS EXTRA CREDIT)

Repeat Module 3 using QNX. The default command shell on the Raspberry Pi in QNX is bash. A python 3 interpreter is also available. Write a script as before and make it an executable file. Do not attempt to run the script at boot time, this is not required. Capture a screenshot of your script running.