Embedded Systems

Coursework 1, Part 1: Raspberry Pi, Python and I^2C

> Do not try to power up the Raspberry Pi until you have followed these instructions in order. It won't do anything at all until it has been configured properly and you could damage it.

1. Find the required parts in your Lab-in-a-Box

- Raspberry Pi Zero WH
- FTDI TTL-234X-3V3 USB-UART cable
- Breadboard
- Raspberry Pi breakout PCB
- microSD card and USB adapter
- Sensor kit
- Wire and wire strippers

2. Set up Raspbian OS

a. Download and install Raspberry Pi Imager to your computer from here: https://www.raspberrypi.org/software/

b. Connect the microSD card to your computer. Use the USB adapter if your computer doesn't have a microSD socket.

c. Run Raspberry Pi Imager. It requires admin privileges because it needs to rewrite the file system on the microSD card.

   i. Select the operating system Raspberry Pi OS Lite (32-bit). It's under Raspberry Pi OS (other) in the list of options.

   ii. Select the SD Card. Be careful – you could erase your data if you choose the wrong card.

   iii. Select 'WRITE' to download and write the operating system image.

d. You need to enable the serial console to enter commands on the Raspberry Pi. This is done by editing the options that are passed to the Linux kernel when it boots up. You need to make this edit on the microSD card now because you have no other way to log in to the Raspberry Pi.

   i. The image you have written contains a partition called 'boot', which you should be able to access as a removable drive through the file system of your computer. Remove and reinsert the microSD card if you can't find it.

   ii. Open the file `config.txt` with a text editor and add the following line to the end:

```
enable_uart=1
```

   iii. Save the file and unmount (eject) the microSD card. Remove it from your computer and insert it into the Raspberry Pi.

3. Establish communication with your Raspberry Pi

In this guide:

```
raspberrypi:~$ command     #Linux prompt command on the Pi
host:~$ command            #Linux/macOS prompt command on your laptop
>>> command                #Python prompt command
```

a. Connect your FTDI USB to Serial cable to your laptop and find the port name. Don't connect the cable to the Raspberry Pi yet
   i. In Windows:
      1. Open Device Manager and look under 'Ports' for an entry like 'USB Serial Port (COMn)'
      2. The port name is the part COMn
   ii. In MAC or Linux:
      1. Open a command prompt and list the available serial devices

         ```
         host:~$ ls /dev/{tty,cu}.*
         ```

      2. Try the command with and without the cable connected and look for a line which only appears when it is connected. That is the name of your serial port.
   iii. If the system does not recognise the USB device, install the serial port driver from here: https://www.ftdichip.com/Drivers/VCP.htm

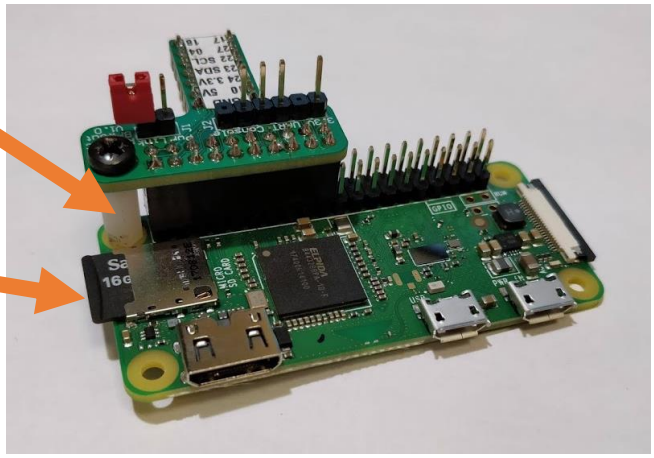
b. Open a terminal over the serial port
   i. In Windows, install a terminal client like PuTTY
      - Check the box 'Serial'
      - Enter the serial port name under 'Serial Line'
      - Set the speed to 115200
   ii. In MAC or Linux, you can use `screen` from the command line. This example is where `/dev/ttyS0` is the serial port:

      ```
      host:~$ screen /dev/ttyS0 115200
      ```

c. Connect and boot the Raspberry Pi
   i. Connect the breakout PCB to the Raspberry Pi.

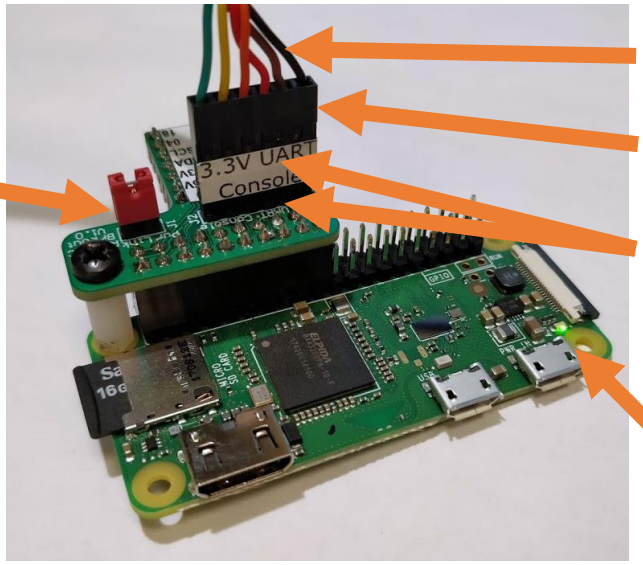
Post aligns with hole in Raspberry Pi

microSD Card



   ii. Connect the USB-UART cable to the breakout PCB in the correct orientation

iii. Connect the power link to power up the Raspberry Pi. This supplies power from the USB-UART cable.



> Take care with connections to the Raspberry Pi header. The USB cable provides 5V and the Raspberry Pi I/O pins can only tolerate 3.3V. Always check the positioning and orientation of the breakout board and USB-UART connector before connecting the power link.
>
> You can also supply power to the 'PWR IN' connector on the Raspberry Pi with a USB power supply. Leave the power link from the USB-UART cable disconnected if you do this; the serial terminal will still work.

d. Log in to the Raspberry Pi
   i. Watch your serial terminal to see boot messages from the Raspberry Pi. It may be blank for a little while on the first boot because it resizes the SD Card partition. You should also see the green activity LED blink.
   ii. Wait for the login prompt to appear. Log in with the username `pi` and the password `raspberry`

4. Set up the Raspberry Pi

a. Connect to WiFi with WPA2-PSK networks like home WiFi or mobile hotspot
   i. Run the Raspberry Pi configuration tool:

```
raspberrypi:~$ sudo raspi-config
```

   ii. Use the arrow keys and enter to select 'System Options', then 'Wireless LAN'
   iii. Enter the SSID and password of your WiFi network when prompted
   iv. Accept the option to reboot the device
   v. Log in and check you have a network connection:

```
raspberrypi:~$ ping google.com
```

b. Connect to WiFi with WPA2-Enterprise networks like Imperial College
   i. On the Raspberry Pi console, create a hash of your College password so it can't be easily read from the SD Card. The hash is a 32 digit hexadecimal number:

```
raspberrypi:~$ echo -n plaintext_password_here | iconv -t utf16le |
      openssl md4
```

ii. Clear the command history to remove your password from it. Make sure you close the serial console when you are finished so it can't be read by scrolling back:

```
raspberrypi:~$ history -c
```

iii. Edit the `wpa_supplicant` file to add the network details:

```
raspberrypi:~$ sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

iv. Add the following lines to the end of the file, replacing *uuu* with your username and *ppp* with password hash you just calculated. Save the file with Ctrl+o and exit with Ctrl+x.

```
network={
      ssid="Imperial-WPA"
      scan_ssid=1
      key_mgmt=WPA-EAP
      identity="uuu"
      password=hash:ppp
      eap=PEAP
      phase1="peaplabel=0"
      phase2="auth=MSCHAPV2"
 }
```

> You can add additional `network={}` stanzas to the file to connect to other networks. The device will connect to whichever is available. Search online to find the correct syntax for other types of WiFi network

v. Restart networking:

```
raspberrypi:~$ sudo reboot
```

vi. Log in and check you have a network connection:

```
raspberrypi:~$ ping google.com
```

> Hashing the password is not completely secure but it does prevent someone from reading your password by viewing the wpa_supplicant.conf file. Two problems remain:
>
> 1. The hash gives anyone the ability to connect to a WiFi network that requires your password.
> 2. The hash could be broken to find your password – though this would require a large amount of computer power.
>
> Be careful with access to the SD card and change your College password if you lose it. Erase it before you return the kit at the end of the term.

c. Enable the SSH server so you can log in and transfer files via the network

i. Change the Raspberry Pi password from the default to prevent anyone else from logging in

```
raspberrypi:~$ passwd
```

ii. Run the configuration utility and use the menu system to enable SSH under 'Interfacing Options'

```
raspberrypi:~$ sudo raspi-config
```

iii. Find the IP address for your Raspberry Pi. Look in data in the wlan0 section

```
raspberrypi:~$ ip addr
```

iv. Log in to your Raspberry Pi from your computer over the network using SSH
- In Windows, use PuTTY, or another SSH client
- In Mac, Linux or Windows with openSSH installed, use the command line (example IP address shown):

```
host:~$ ssh pi@146.169.152.34
```

> You should be able to connect to the Raspberry Pi from any computer on the same network, depending on the structure of the network. In general, you won't be able to reach it via the internet.
>
> The IP address is likely to change if you restart or reconnect the Raspberry Pi. You may be able to fix it if you have control over your DHCP server, but you can't on the College network.

v. Copy a file (just a text file to test) from your computer to the Raspberry Pi
- In Windows, use WinSCP or a similar client
- In Mac, Linux or Windows with openSSH installed, use the command line:

```
host:~$ scp test.txt pi@146.169.152.34:~/test.txt
```

5. Select and research a sensor

Choose an I^2C sensor from the selection available. It doesn't need to be one you will use in your coursework.

- What does it do?
- What is the power supply voltage? If it uses a breakout module then check documentation for the module
- What is the control flow for the sensor?
  - What is it's I^2C bus address?
  - Does it need enabling or configuring before use?
  - Does it measure automatically or on demand?
  - How do you configure it (if applicable)?
  - How do you request a measurement (if applicable)?
  - How do you read back the result?
  - What conversion is needed to convert the result into something meaningful?

6. Establish communication with your sensor

a. Power down the Raspberry Pi

i. Shut down the operating system. This reduces the chance of file corruption and it should be done before any power down.

```
raspberrypi:~$ sudo halt
```

ii. Remove the power link

b. Plug the Raspberry Pi breakout adapter and sensor module into different locations on the breadboard

c. Wire up the sensor module
   i. Check the documentation for the sensor module (the breakout, not the chip itself) to find the power requirements.
   ii. Use 3.3V if possible to reduce the chance of damage
   iii. Wire up the power supply pins
   iv. Wire up the I2C data and clock lines (SDA and SCL)
      - I2C pullup resistors are not required because they are included on the Raspberry Pi.

d. Power the Raspberry Pi again and log in via serial or SSH

e. Install the I2C tools package:

```
raspberrypi:~$ sudo apt-get install i2c-tools
```

f. Enable I2C (Interface Options) and reboot

```
raspberrypi:~$ sudo raspi-config
raspberrypi:~$ sudo reboot
```

g. Search the I2C bus for your sensor module.

```
raspberrypi:~$ sudo i2cdetect -y 1
```

The command shows a map of all the I2C addresses. You should see an entry at the address you expect from reading the sensor documentation. Note that some devices have an address that is configurable by making connections to certain pins.

7. Set up Python

a. Install pip, the python package manager, and the smbus2 and gpiozero modules

```
raspberrypi:~$ sudo apt-get install python3-pip
raspberrypi:~$ sudo pip3 install smbus2 gpiozero
```

b. Use the interactive python terminal to get data from your sensor by referring to the sensor datasheet and using the smbus library functions shown in the lecture

```
raspberrypi:~$ python3
```

c. Convert the bytes of data returned from the sensor into useful numerical values

d. Write the functions into a python script for future use.
   i. On the Raspberry Pi, use the `nano` text editor, or install and learn to use `vim`

ii. On your computer, use your favourite text editor and transfer the files with `scp`

This guide covers the simplest methods of developing on the Raspberry Pi. But it is not very efficient to edit files directly on the Pi and copying from your computer is slow and can cause mistakes with file versions.

Alternatives you can investigate include:

- Set up VNC to give you a graphical interface to the Pi
- Use `rsync` to synchronise files between the Pi and your development machine
- Use `git` to synchronise files between the Pi and your development machine and manage contributions from multiple team members
- Install and run a Jupyter notebook server on the Raspberry Pi, which can be accessed on a computer via a web browser

8. Sensor list

Part Number	Function	Analogue output
FXOS870 + FXAS21002	9 Degree of freedom motion sensor	No
SEN13329	Load Cell (force sensor)	Yes (differential, use max. gain)
TMP006	Thermopile (non-contact temperature sensor)	No
VL53L0X	Time of flight distance sensor	No
LIS3DH	Accelerometer	No
MLX90393	Magnetometer (compass)	No
AS7262	Spectral light sensor	No
BMP280	Atmospheric pressure sensor	No
Si7021	Air temperature and humidity sensor	No
CCS811	Air quality sensor	No
D6F-V03A1	Air velocity sensor	Yes (0-5V output)
P182A	Flex sensor	Yes (resistive)
ADS1115	ADC for analogue sensors	
LV-EZ1	Ultrasonic rangefinder	Yes
MPRLS	Air pressure sensor with tube	No