University of Cape Town

EEE3096S

Embedded Systems II

Lab Handbook

Keegan Crankshaw

28 July 2019

Introduction

This lab handbook serves as a side-by-side reference manual for you to use in your laboratory sessions. Not all information is included here (we expect you do to some research of your own) but it is a good starting point. It also includes some additional information you may find useful as you use the Raspberry Pi for your own projects.

This handbook covers a range of topics from installation, to cross compilation, to some more advanced topics such as networking. There are links to useful learning resources, as well as some quick reference guides for you to refer back to as you work through all your practicals.

The hope is that once the course is complete, you can hold on to this manual and use it for any other Raspberry Pi related projects you may choose to undertake.

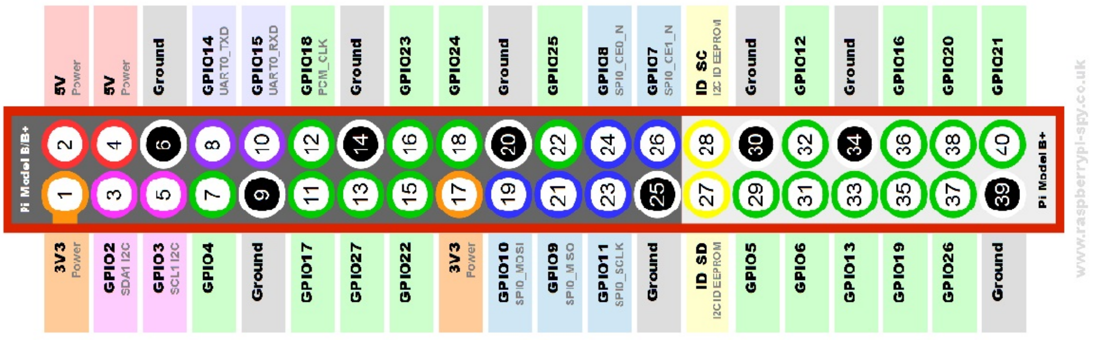
While all efforts were made to ensure that this manual is correct and accurate, it is quite possible that errors have been made. Please email any errors you find to the TA.

Contents

1 Raspberry Pi 3B+

This course uses the Raspberry Pi 3B+. The pinout for this device is shown below. Note that some pins have special uses. This pins may not be able to be used as GPIO. Note that the header has a notch to indicate pin 1. It's also important to note the difference board numbering and BCM numbering. Board numbering is the pin count on the headers, i.e. the number shown in the circle in the image. BCM stands for Broadcom SOC channel - and refers to the GPIO number in the descriptions next to number in the circle. For example, board number 3 is GPIO 2, and board number 33 is GPIO number 13. Some pins have special functions. You'll learn about these functions as the course progresses.

You can only use one board numbering system (BOARD or GPIO) per project, and you usually need to configure this using a method made available in the library.



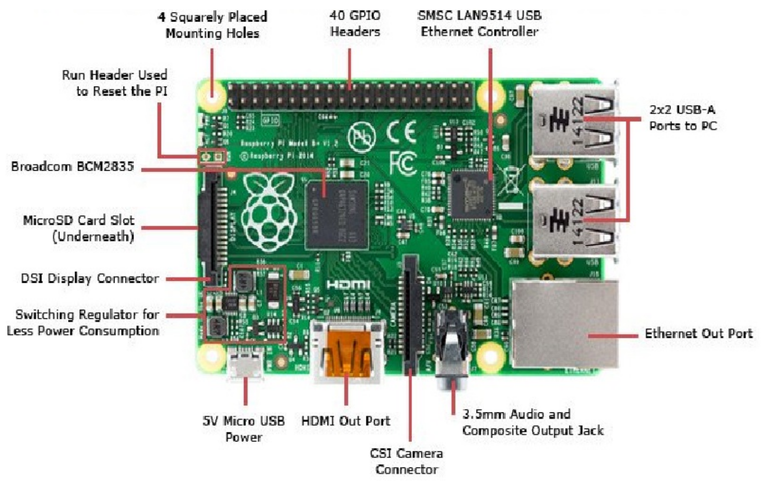

4 Squarely Placed Mounting Holes
40 GPIO Headers
SMSC LAN9514 USB Ethernet Controller
Run Header Used to Reset the PI
2x2 USB-A Ports to PC
Broadcom BCM2835
MicroSD Card Slot (Underneath)
DSI Display Connector
Switching Regulator for Less Power Consumption
Ethernet Out Port
5V Micro USB Power
HDMI Out Port
CSI Camera Connector
3.5mm Audio and Composite Output Jack

Figure 1: GPIO Header of the B+ (source)

2 Setting up your Pi

In order to use the Pi you need to install an operating system on it and set up networking. The following will lead you through installing Raspbian as an operating system. Raspbian is an operating system created for the Raspberry Pi, built on Debian. You will then configure the networking settings in Raspbian to allow you to access the Pi remotely using SSH.

While there are no requirements for this course on which operating system to use, it is strongly recommended you become familiar with a Linux-based operating system. Recommendations include Ubuntu and Mint.

2.1 Prerequisites

Hardware Requirements	Software Requirements
• Raspberry Pi	• Rasbian (Desktop & Recommended Software)
• Ethernet Cable	• PuTTY (the full suite, Windows only). [1]
• Power source for Raspberry Pi	• SDFormatter
• Means of writing to the Micro SD Card	• Etcher, an image writing tool

2.2 Prepare the SD Card

The SD Card needs to be formatted before it can be used. To make your life simple, there is a tool called "SD Card Formatter" - a link is available in the Software Requirements box of the prerequisites section.

2.3 Install and configure Raspbian

This process will follow headless installation, as we are going to assume most students do not have access to spare screens, keyboards and mouses[2].

Headless mode on the Raspberry Pi refers to using it without direct user input and output (essentially no screen, mouse or keyboard connected directly to it). This is how all of the practicals will be conducted, as it is how most IoT devices are configured (over a network).

1. Insert the SD card into the computer. If you do not have a reader available, speak to a friend or tutor who does.

2. Format the Micro SD card
   If you're on Windows, this is where you would use SDFormatter. If you're on Ubuntu, you can use GParted.

[2] Yes, "mouses" is entirely valid is and used to distinguish a computer device from a rodent: https://en.wikipedia.org/wiki/Computer_mouse#Naming

3. Write image to SD card
   Open Etcher. Select the downloaded zip image, and the SD card, and format. At the end of format, it may read that it failed, but don't worry. Upon completion, Windows will try to mount partitions on the SD card that it can't read. Just press "Cancel" and then "OK" to the dialog boxes that pop up. The boot partition is the only partition we will be dealing with.

4. Enable SSH
   SSH is covered in detail in Section 10, but for now just enable SSH by creating a file called "ssh" (no file extension) on the boot partition of the SD card.

5. Configure the Networking
   This section can get complicated, so we've created Section 8 to help explain it. For now, just follow the steps in Section 8.2 and Section 8.3.

6. Boot
   Insert the SD Card into the Pi. Connect an Ethernet cable between your PC and the Pi. Connect the Pi to a power source.

7. Ensure connectivity
   Ensure you can "see" your Pi. Open a command window or terminal, and run

   ```
   $ ping 192.168.137.15
   ```

   You should see a response similar to

   ```
   $ 64 bytes from 192.168.137.15: icmp_seq=1 ttl=64 time=0.557 ms
   ```

   or

   ```
   $ Reply from 192.168.137.15 bytes=32 time=0.5ms TTL=52
   ```

8. SSH in to the Pi
   Section 10 covers all the details on SSH. For now, open a command prompt or terminal on your machine, and enter in the following:

   ```
   $ ssh pi@192.168.137.15
   ```

   You will be asked to add the machine to known hosts. Select yes.
   The password required to log in is simply "raspberry"

9. Configure
   For practicals you will need to enable a few services on the Pi.

   - Open a terminal. You can either click the icon or press Ctrl+Alt+T
   - Run

     ```
     $ sudo raspi-config
     ```

   - Use the arrow keys and scroll down to "Interfacing Options"
   - Enable SSH, VNC, SPI, I2C and Serial

- Save and exit raspi-config by using the left and right arrow keys to jump to the bottom buttons. Your Pi will reboot. Wait about 30 seconds, and SSH in again.

10. Expand the Filesystem The Rasberry Pi Installation may not make use of the full SD card. In order to give yourself as much file space as you have access to, you need to expand the filesystem.

    - SSH into the Pi
    - Run

    ```
    $ sudo raspi-config
    ```

    - Use the arrow keys and scroll down to "Advanced Options"
    - Select "Expand Filesystem"
    - You will be shown a message saying that the root partition has been resized. Select "Ok", then "Finish" then "Yes" to reboot your Pi.

11. Creating another user
    It isn't good practice to run everything as root, so we create another user with sudo privileges. The first command adds the user, the next command adds the user to the group `sudo`, which gives them sudo execution rights. On the Pi, run the following:

    ```
    $ sudo adduser <username>
    $ sudo adduser <username> sudo
    ```

    Now that you have created a new user, you can log into the Pi via SSH using that username as follows:

    ```
    $ ssh <username>@192.168.137.15
    ```

    To switch users while you are SSH'd in run

    ```
    $ sudo su - <username>
    ```

    To switch to root (not recommended), run

    ```
    $ sudo su
    ```

    To change your password, you can simply run

    ```
    $ passwd
    ```

    To change the password of another user, drop down to root and run

    ```
    $ passwd <username>
    ```

    To delete a user, switch to another user with root access, and run

    ```
    $ sudo deluser -remove-all-files -f <username>
    ```

3 The Unix Shell

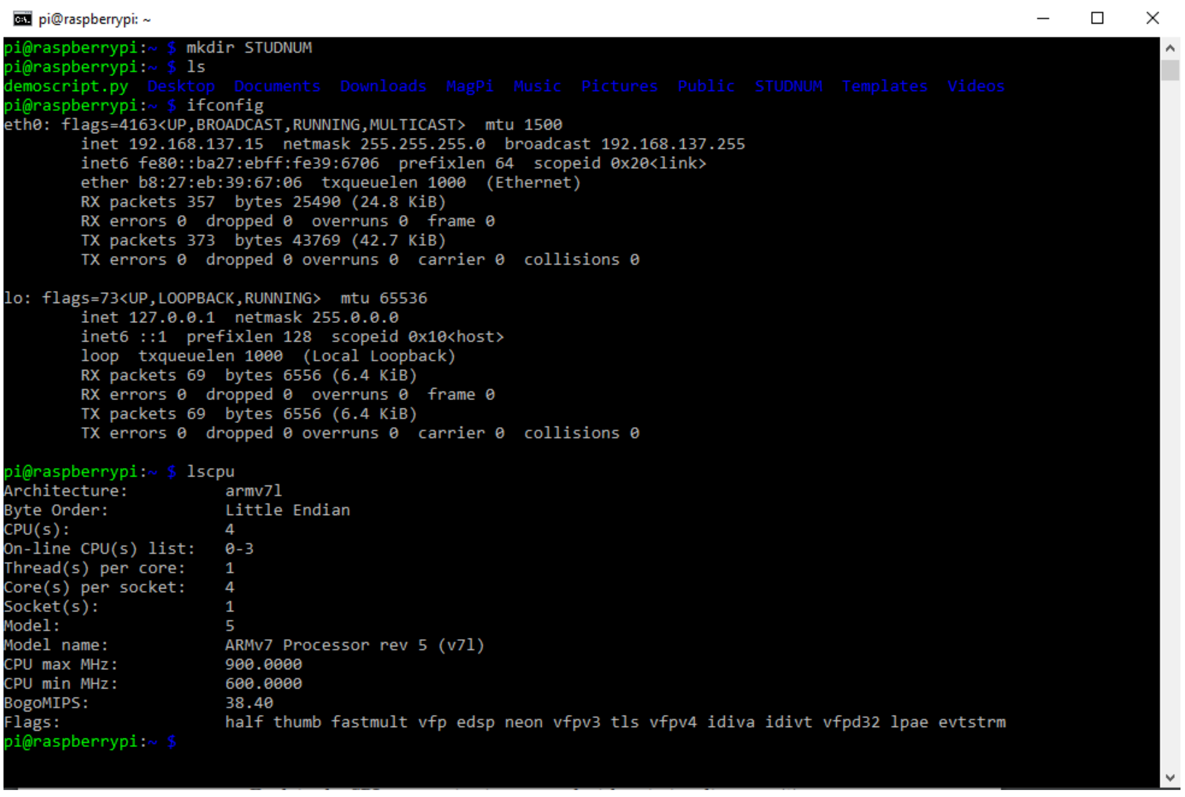
You worked with a few Unix commands in the shell in the installation section. This section does a bit of a deeper dive into commands that you may find useful.

Being able to use the Unix Shell and terminal commands is an invaluable skill, and a requirement for this course. Follow this guide (https://swcarpentry.github.io/shell-novice/) for more learning resources.

Some useful commands are listed below. There is also space to add your own commands if you find any that are useful.

Table I: Some useful shell commands

Command	**Use**
ls	List current files and folders in directory. ls -al is useful to list everything
pwd	Prints the current working directory
cd <directory>	Change to a specified directory. eg "cd .. " will take you one level up
ifconfig	Shows details on current network interfaces
touch <file>	Create <file>
nano <file>	Opens <file>in the nano text editor
vim <file>	Opens <file>in the vim text editor
mkdir <dir>	Creates the folder specified in <dir>
sudo <cmd>	executes the command <cmd>with sudo privileges
raspi-config	Must be run as administrator Opens up the Raspberry Pi Configuration Tool
man <command>	Opens the manual for a particular command
ssh	Creates an SSH session. See Section 10
scp	Secure copy. See Section 12



```
pi@raspberrypi: ~
pi@raspberrypi:~ $ mkdir STUDNUM
pi@raspberrypi:~ $ ls
demoscript.py  Desktop  Documents  Downloads  MagPi  Music  Pictures  Public  STUDNUM  Templates  Videos
pi@raspberrypi:~ $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.137.15  netmask 255.255.255.0  broadcast 192.168.137.255
        inet6 fe80::ba27:ebff:fe39:6706  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:39:67:06  txqueuelen 1000  (Ethernet)
        RX packets 357  bytes 25490 (24.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 373  bytes 43769 (42.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 69  bytes 6556 (6.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 69  bytes 6556 (6.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pi@raspberrypi:~ $ lscpu
Architecture:          armv7l
Byte Order:            Little Endian
CPU(s):                4
On-line CPU(s) list:   0-3
Thread(s) per core:    1
Core(s) per socket:    4
Socket(s):             1
Model:                 5
Model name:            ARMv7 Processor rev 5 (v7l)
CPU max MHz:           900.0000
CPU min MHz:           600.0000
BogoMIPS:              38.40
Flags:                 half thumb fastmult vfp edsp neon vfpv3 tls vfpv4 idiva idivt vfpd32 lpae evtstrm
pi@raspberrypi:~ $
```

Figure 2: An example output of running some commands in the shell

4 Text Editors

When editing files on the Raspberry Pi there are multiple options, depending on whether you choose to edit the file in a specific program, or edit it within the command shell.

4.1 GUI Based Text Editors

4.1.1 Notepad++

Notepad++ is a text editor for Windows. It can be downloaded here. Language can be set to enable syntax highlighting. To ensure you use spaces instead of tabs, go to Settings - Preferences - Tab Settings - tick the box "Replace by Space"..

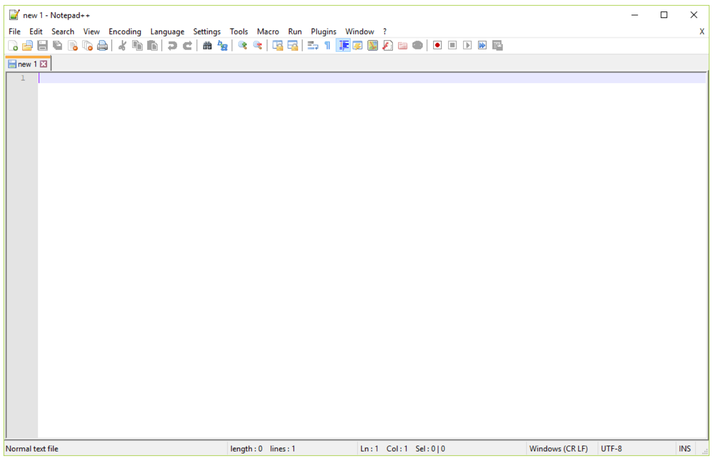
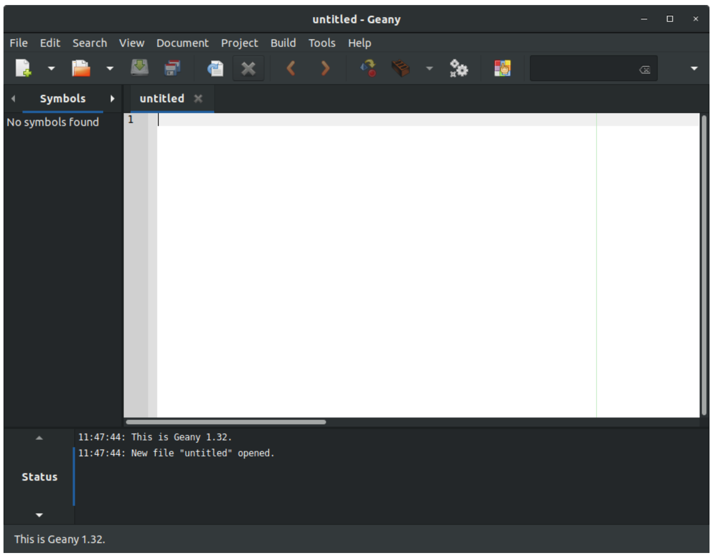

new 1 - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
new 1
Normal text file
length : 0 lines : 1
Ln : 1 Col : 1 Sel : 0 | 0
Windows (CR LF) UTF-8 INS

Figure 3: The Notepad++ text Editor

4.1.2 Geany

Geany is a GUI based text editor for Linux. It is very similar to Notepad++ on Windows and supports essentially the same features. The programming language can be set for syntax highlighting. Take note of whether you are using tabs or spaces (4 spaces is recommended for indentation). This can be set through preferences - editor - indentation - type - spaces. After changing the setting, close and reopen the file.



```
11:47:44: This is Geany 1.32.
11:47:44: New file "untitled" opened.
```

This is Geany 1.32.

Figure 4: The Geany Text Editor

4.2 Terminal Based Text Editors

4.2.1 Nano

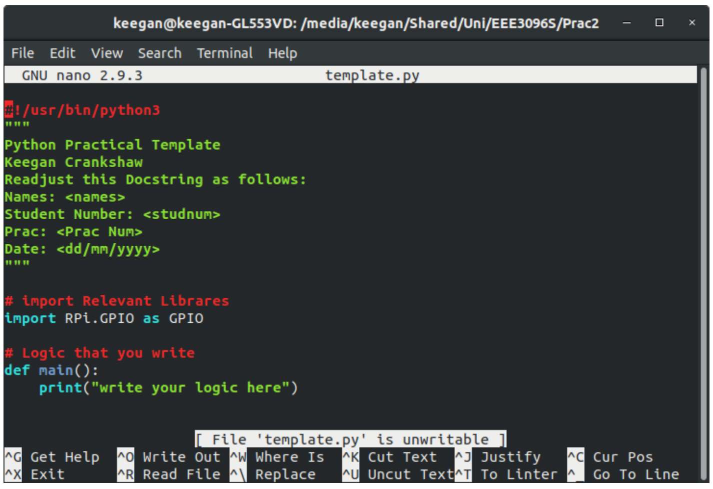
Nano is a very easy to use file editor that runs within the command shell. Nano comes installed on Ubuntu and Raspbian by default, but if you do find yourself needing to install it, you can run

```
$ sudo apt-get install nano
```

Once it is installed, you can use nano to edit text files by typing

```
$ nano filename
```

You will be presented with an interface like this (currently the user is editing the python template):


keegan@keegan-GL553VD: /media/keegan/Shared/Uni/EEE3096S/Prac2
File Edit View Search Terminal Help

```
  GNU nano 2.9.3                         template.py

#!/usr/bin/python3
"""
Python Practical Template
Keegan Crankshaw
Readjust this Docstring as follows:
Names: <names>
Student Number: <studnum>
Prac: <Prac Num>
Date: <dd/mm/yyyy>
"""

# import Relevant Librares
import RPi.GPIO as GPIO

# Logic that you write
def main():
    print("write your logic here")

                      [ File 'template.py' is unwritable ]
^G Get Help   ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit       ^R Read File ^\ Replace   ^U Uncut Text^T To Linter ^_ Go To Line
```

Figure 5: The python template opened in nano

Commands for nano are displayed on the bottom. A quick reference is as follows:

- Arrow Keys to move around as expected
- Ctrl-W to find (Where is)
- Ctrl-O to save (Write Out)
- Ctrl-X to Exit (will be prompted to save on exit, press 'y' to save, press 'n' to exit without saving)
- On Windows using Putty SSH client, 'paste' is performed by right-clicking with the mouse anywhere in the window, the contents of the clipboard will be pasted at the current cursor location

Note: By default, nano inserts a tab when the tab key is pressed. If you'd like to change this to insert a number of spaces, for example, four (as is recommended by PEP 8), do the following:

- Edit your /.nanorc file (or create it)

```
$ sudo nano ~/.nanorc
```

- Edit it to contain the following:

```
set tabsize 4
set tabstospaces
```

- Reboot your Pi to enable the changes

4.2.2 Other Terminal Editors

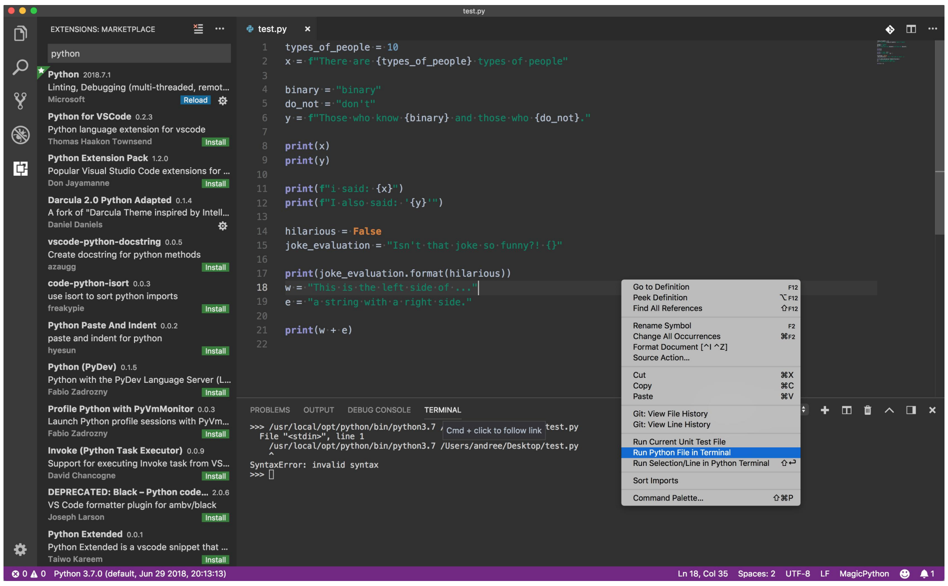
There are other terminal based text editors, such as emacs and vim. Feel free to use whichever you are comfortable with.

5 Programming IDEs

There are a few specific IDEs which you may find useful in this course. Specifically, they are CLion and PyCharm, both of which are created by JetBrains. You can get access to licences for JetBrains IDEs by signing up for the Github Education Pack.

5.1 Visual Studio Code

Visual Studio Code is a powerful IDE that can be configured to run various programming languages. It has an array of plugins from CC++ to Latex, as well as Python and Raspberry Pi Plug-ins. You can learn more about Visual Studio Code here.


EXTENSIONS: MARKETPLACE
python
Python 2018.7.1
Linting, Debugging (multi-threaded, remot...
Microsoft
Reload
Python for VSCode 0.2.3
Python language extension for vscode
Thomas Haakon Townsend
Install
Python Extension Pack 1.2.0
Popular Visual Studio Code extensions for ...
Don Jayamanne
Darcula 2.0 Python Adapted 0.1.4
A fork of "Darcula Theme inspired by Intell...
Daniel Daniels
vscode-python-docstring 0.0.5
Create docstring for python methods
azaugg
code-python-isort 0.0.3
use isort to sort python imports
freakypie
Python Paste And Indent 0.0.2
paste and indent for python
hyesun
Python (PyDev) 0.1.5
Python with the PyDev Language Server (L...
Fabio Zadrozny
Profile Python with PyVmMonitor 0.0.3
Launch Python profile sessions with PyVm...
Fabio Zadrozny
Invoke (Python Task Executor) 0.0.9
Support for executing Invoke task from VS...
David Chancogne
DEPRECATED: Black – Python code... 2.0.6
VS Code formatter plugin for ambv/black
Joseph Larson
Python Extended 0.0.1
Python Extended is a vscode snippet that ...
Taiwo Kareem
test.py
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Cmd + click to follow link
Go to Definition
Peek Definition
Find All References
Rename Symbol
Change All Occurrences
Format Document
Source Action...
Cut
Copy
Paste
Git: View File History
Git: View Line History
Run Current Unit Test File
Run Python File in Terminal
Run Selection/Line in Python Terminal
Sort Imports
Command Palette...
Python 3.7.0 (default, Jun 29 2018, 20:13:13)
Ln 18, Col 35
Spaces: 2
UTF-8
LF
MagicPython

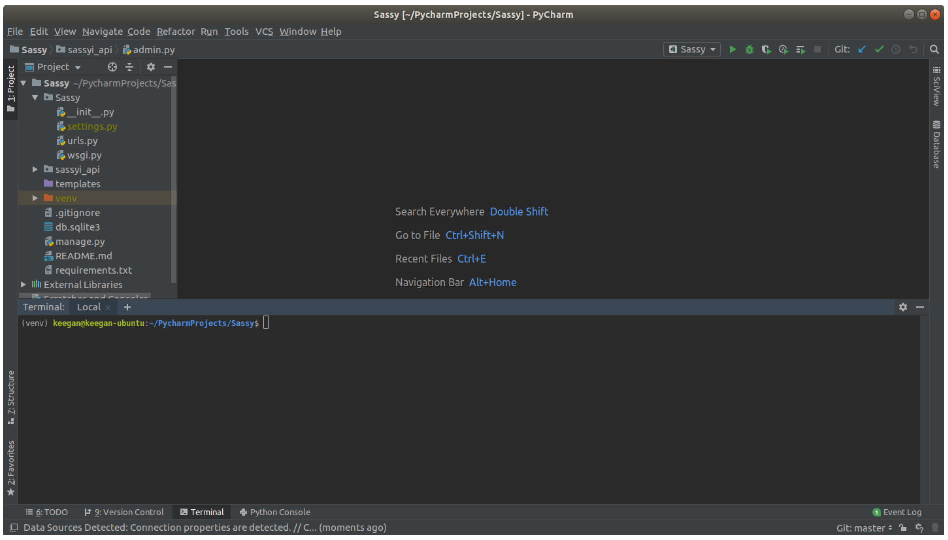
```
types_of_people = 10
x = f"There are {types_of_people} types of people"

binary = "binary"
do_not = "don't"
y = f"Those who know {binary} and those who {do_not}."

print(x)
print(y)

print(f"i said: {x}")
print(f"I also said: '{y}'")

hilarious = False
joke_evaluation = "Isn't that joke so funny?! {}"

print(joke_evaluation.format(hilarious))
w = "This is the left side of ..."
e = "a string with a right side."

print(w + e)
```

```
>>> /usr/local/opt/python/bin/python3.7 /Users/andree/Desktop/test.py
  File "<stdin>", line 1
    /usr/local/opt/python/bin/python3.7 /Users/andree/Desktop/test.py
    ^
SyntaxError: invalid syntax
>>> 
```

Figure 6: VSCode configured for Python (image source)

5.2 PyCharm

PyCharm is the Python Editor created by JetBrains. You can read more about PyCharm here. Learn about setting up a virtual environment (good practice in Python) here.



```
(venv) keegan@keegan-ubuntu:~/PycharmProjects/Sassy$ 
```

Figure 7: The PyCharm IDE

5.3 CLion

CLion is a cross compilation tool developed by JetBrains that will be used in this course.

This section covers installation and configuration of JetBrains CLion on Windows and Ubuntu. It assumes you have a JetBrains Education account (possible through the Student GitHub Pack - https://education.github.com/pack).

5.3.1 CLion on Windows

CLion on Windows for cross compilation requires use of the Ubuntu subsystem. At this point, it's likely better to dual boot the two operating systems. However, if you're unable to do this, you can follow through the guide below. A reminder, you don't need to use CLion - comiplation on Windows is available as pointed out in Section 15.4.

In this guide, we're going to be working with Ubuntu 18.04 as the distribution. To install WSL, follow this tutorial. Be sure to select Ubuntu 18.04 as your distribution.

Once you've installed it, you can start the instance by pressing the start button and searching for "Ubuntu". Upon first run, you will be asked to create a username and password. Remember these details, as they will be used later. Once you have created your username and password, run the following commands:

```
$ sudo apt-get update
$ sudo apt-get ugrade
$ sudo apt-get install libc6-armel-cross libc6-dev-armel-cross
$ sudo apt-get install binutils-arm-linux-gnueabi libncurses5-dev lib32z1
$ sudo apt-get install gcc-arm-linux-gnueabi g++-arm-linux-gnueabi
```

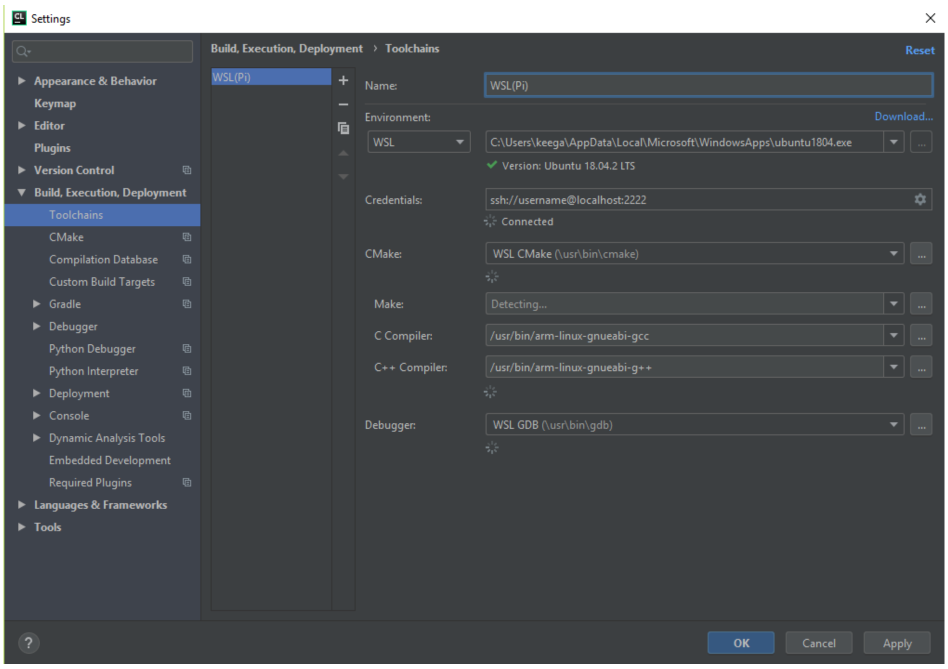
Then, configure ssh and relevant/related items using the JetBrains script. Still on the Ubuntu sub-system, run:

```
$ wget https://raw.githubusercontent.com/JetBrains/clion-wsl/master/ubuntu_
    ↪ setup_env.sh
$ bash ubuntu_setup_env.sh
```

This script configures an SSH connection on port 2222. Test is by running the following on the Ubuntu Subsystem

```
$ ssh username@127.0.0.1 -p 2222
```

Now, open CLion. Navigate to File - Settings - Build, Execution, Deployment - Toolchains. Under environment, Select "WSL". Click the gear next to Credentials, and enter in your username and password, ensuring that the correct port number (2222) is used. You will need to change C Compiler and C++ Compilers to use the Raspberry Pi Compilers we installed in the earlier steps. Change the C compiler to use "/usr/bin/arm-linux-gnueabi-gcc" and the C++ Compiler to use "/usr/bin/arm-linux-gnueabi-g++". The configuration should now look as follows:



Name: WSL(Pi)
Environment: WSL C:\Users\keega\AppData\Local\Microsoft\WindowsApps\ubuntu1804.exe
✓ Version: Ubuntu 18.04.2 LTS
Credentials: ssh://username@localhost:2222
Connected
CMake: WSL CMake (\usr\bin\cmake)
Make: Detecting...
C Compiler: /usr/bin/arm-linux-gnueabi-gcc
C++ Compiler: /usr/bin/arm-linux-gnueabi-g++
Debugger: WSL GDB (\usr\bin\gdb)

Figure 8: CLion WSL Configuration on Windows

Once you create and compile a project, you should be able to move it to the Pi using SCP (or PSCP if SCP is disabled), adding the executable flag, and running it. Do not be surprised if there is an error once you've built it on the Pi

5.3.2 CLion on Ubuntu

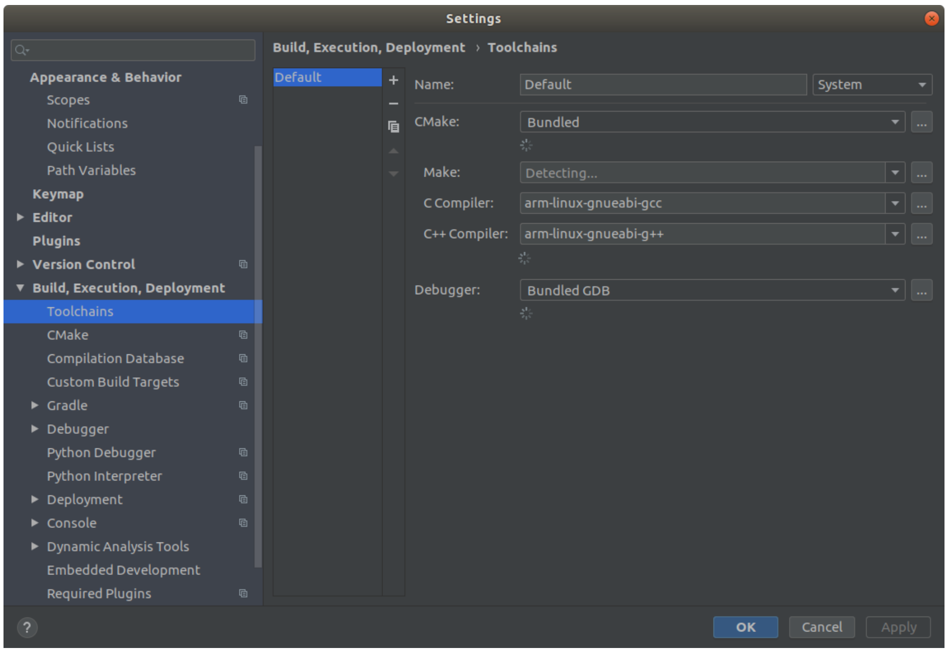
CLion on Ubuntu is considerably easier to configure. Install CLion by running

```
$ snap install clion --classic
```

RUn the following commands to install required packages:

```
$ sudo apt-get update
$ sudo apt-get ugrade
$ sudo apt-get install libc6-armel-cross libc6-dev-armel-cross
$ sudo apt-get install binutils-arm-linux-gnueabi libncurses5-dev lib32z1
$ sudo apt-get install gcc-arm-linux-gnueabi g++-arm-linux-gnueabi
```

Your configuration file should be as follows:



Name: Default System
CMake: Bundled
Make: Detecting...
C Compiler: arm-linux-gnueabi-gcc
C++ Compiler: arm-linux-gnueabi-g++
Debugger: Bundled GDB

Figure 9: CLion Ubuntu Configuration

6 Git

Throughout the course you will be required to write and submit code and other files to an online git repository, as well as submitting it through Vula. Git is a popular version control application that allows you to keep a record of changes you have made to files over time. Additionally, by using a version controlled repository that is accessible to other collaborators, it provides a powerful way for many people to collectively build a larger system. Almost all open source projects use a version control system to enable them to work with anyone else around the world in a structured and organised manner. There are many platforms that offer free hosting services where you are able to share repositories, Github, and Gitlab are two such options. Making an account on either of these (or a similar alternative) allows you to: (1) keep track of your own projects and the changes you make to them, (2) work collaboratively with teammates if you choose to share your repositories with them, and (3) is an increasingly common way of sharing a portfolio of your prior work with potential employers.

Git is a local tool. To use online backups, it's recommended to use a remote (online) repository management tool. GitHub is recommended because, as a student, there are many benefits which you can access. See https://education.github.com/pack to sign up.

Git can be intimidating in the beginning, but it becomes invaluable as you progress in software development. This page has a great guide you can follow to get well acquainted.

Before arriving at Prac 1, make sure you have created a demo repository on a computer and pushed it to the cloud using the instructions found below.

6.1 A Quick Git Get Go

This sub section has a bunch of useful ”recipes” to follow that will be common throughout your experience with git.

6.1.1 Creating a GitHub Account and Configuring your Computer

- Start by creating a GitHub account
- Install git on your computer
  Lab computers already have git installed.
  If you’re using a Linux based system, this can be as simple as `sudo apt-get install git`
  If you’re using Windows, you need to download and use an installer.
- Once git is installed, run the following commands:
  Note: Do not do this on the lab computers. If you are on a lab computer, rather set the user.name and user.email parameters from within the created git repository, and do not use the --global flag. Only use the global flag if you’re on your own system, such as your own PC or your Pi.

```
$ git config --global user.name "Your Name"
$ git config --global user.email "github email address"
```

- Git is now configured

6.1.2 Creating a New Project

Git consists of three primary stages: Untracked, staged and committed. Untracked files are not tracked by the repository. Staged files are files staged for commit but not yet committed. Committed files are ”saved” to git. On your local system:

- Create a folder and enter into it
- Run `git init`
- Create a new text file, for example ”test.txt”
- Run `git status`
- You will see there is an untracked file. Add it to git by running `git add test.txt`
- It is possible to add all untracked files by running `git add .`
- If you run `git status` again, you will see that ”test.txt” has been staged, but not yet committed. Commit test.txt by running `git commit -m "Created test.txt"`. The `-m` flag is to include a git commit message. It’s useful to use these messages to explain what has changed in this commit.

6.1.3 Linking GitHub and your Local Project

On GitHub, create a new repository and give it a meaningful name and description. Take note of the link (something like https://github.com/<username>/<project-name>.git)

GitHub gives instructions on how to push an existing repository from the command line, but for completeness sake the commands are included here:

```
$ git remote add origin https://github.com/<username>/<project-name>.git
$ git push -u origin master
```

If you refresh the GitHub page, you should now see your files and commits.

6.1.4 Understanding .gitignore

Related SW Carpentry link: Ignoring Things

You may have seen the option to add a .gitginore when creating a new repository on GitHub. This is used to get Git to ignore certain files, such as interim or raw data files.

6.1.5 Fetching an Existing Git Repository

In order to do the practicals, you will need to fetch initial files and templates that have been created for you. These are available from this git repository. While you could navigate to GitHub, download a compressed folder, move it to the Pi, unzip it, and then work with the files, it's much easier to run the following command:

```
$ git clone https://github.com/kcranky/EEE3096S.git
```

7 LaTeX

7.1 Overview

LaTeXis a great way to write documents, and is required for use in all your documents to be submitted for this course. LaTeXis better than basic text editors such as Microsoft Word or Google Docs due to the following[3]:

1. Dealing with mathematical notation.
   Layout and entry are generally easier using LaTeX than some other sort of equation editor. Online tools such as Detexify make it very simple to find the symbol you need.

[3]Adapted from this stack exchange question

2. Consistent handling of intra-document references and bibliography.
   As of a couple of years ago the major editors still had problems with re-numbering cross-references and bibliography items. This is never a problem with BibTeX or LaTeX.

3. Separation of content and style.
   In principle this means that you can write your document without caring how it is formatted, and at the end of the day wrap it in the style-file provided by the journal publisher before submission to conform to the house style. In practice some of the journal publishers demand special formatting commands that partially moots this process. Furthermore recent versions of Word and LibreOffice Writer, when properly used, should be able to keep track of various levels of section heading separate from the body text, and apply uniform styling to each level. The gap is somewhat closing.

4. Tables and illustrations.
   With online tools such as Tables Generator, creating tables in LaTeXis as simple as copy-pasting data from excel. Images can be inserted exactly where you specify them without worrying about justification or overlay.

There are a few difficulties with LaTeX. These include:

1. Difficulties with collaborative editing (consider the convenience of Google Docs)
2. Spell check (Microsoft Word has a much more advanced spell and grammar check)
3. Ease of use (LaTeXis technically a ”document preparation system” as opposed to a text editor)

However, many if not all of these issues are mitigated by the use of an online tool known as Overleaf. Overleaf provides you with templates, the ability to collaborate, and (thankfully), a spell check function. It runs in browser and doesn't require any installation.

If you would like to run an offline version, there are various options, but Visual Studio Code with the Latex Workshop Plugin is suggested.

7.2 Using Overleaf

Once you have created an account on Overleaf, you need a template to work from. The IEEE conference paper template is available on Download the zip file.

- In Overleaf, click ”New Project” and select ”Upload Project”
- Select or drag the report template zip file you downloaded from Vula
- The template will load and you will be able to edit it.
- Note: you will need to change the ”Main document” in the Menu when editing your document to be ”Report.tex”

- Note: You will not be able to compile the source into a pdf if you are viewing *Preamble.tex*

7.3 Useful Latex Tools

7.3.1 Overleaf

Overleaf is an online collaborative Latex editor. It is recommended you use Overleaf for this course, as it has plenty templates which you can draw from.

7.3.2 Detexity

Detexify is a web app you can use to find Latex codes for specific symbols.

7.3.3 Tablesgenerator

Tablesgnerator is a website for easily creating Latex tables. You can copy-paste tables from other applications such as Excel.

7.3.4 Citation Machine

Citation machine can be used to easily generate BibTex.

8 Networking on the Pi

There are many ways to interface with the Pi. This section will cover types of network connectivity.

8.1 A Brief Overview of Networks

It is useful to have a basic idea of how networks, IP addresses, and subnets work. For this, it it suggested you read this article from Microsoft: https://tinyurl.com/y2z8x9za

8.2 Assigning a Static IP to the Pi

This section will assign a static Ethernet address to the Raspberry Pi. This is useful for your first configuration. If you have not done so, it is recommended you follow the instructions in

Section2 to configure your Raspberry Pi.

1. Insert the SD card into your computer and navigate to the BOOT partition
2. Open ”cmdline.txt” and append the following to the line (don’t create a new line)

   ```
   ip=192.168.137.15
   ```

   This tells the Raspberry Pi to configure the Ethernet port to use the IP address 192.168.137.15
3. Enable SSH as per Section 10.
4. You need to configure your PC to use the same subnet as the Pi. To do so, see the information below in Section 8.3

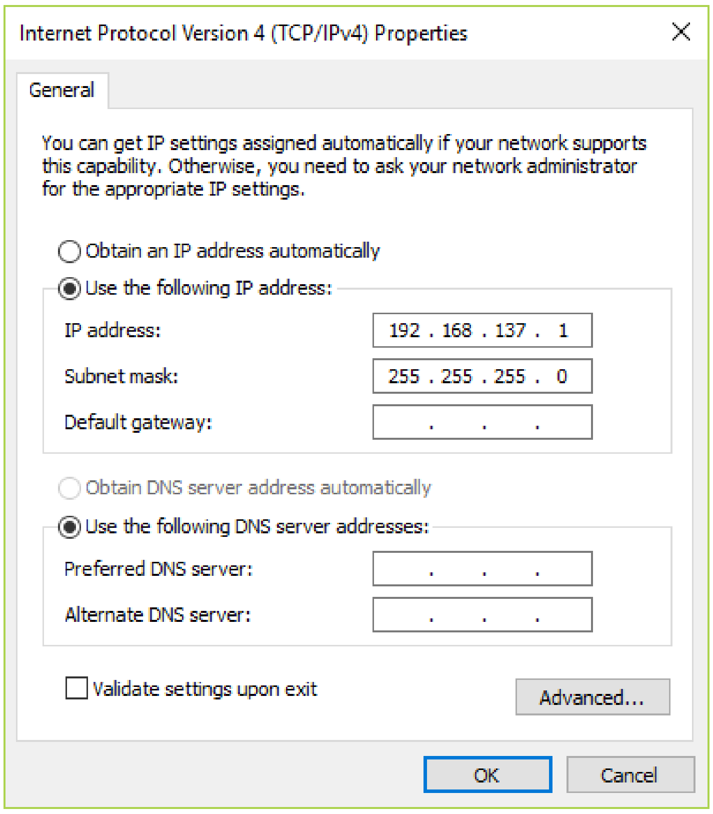
8.3 Assigning a Static IP to your Computer

In order to use Ethernet for SSH, VNC, etc, it is required that the Pi and your computer all be on the same subnet. This section details how to do it.

8.3.1 Windows

To change the IP of your Ethernet port on Windows 10, complete the following steps:

- Right click on your network option in Windows taskbar
- Select”Open Network & Internet Settings”, on the lower right hand side of the screen.
- Select ”Change Adapted Options”
- Right click on the Ethernet Connection and select ”Properties”
- Select ”Internet Protocol Version 4 (TCP/IPv4) and click ”Properties”
- Select ”Use the following IP address:” and enter in the following options:
  - IP Address: 192.168.137.1
  - Subnet Mask: 255.255.255.0
- You have successfully changed the IP of the Ethernet card on your computer. It is suggested that you now ensure connectivity by attempting to ping the Pi.


Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

( ) Obtain an IP address automatically
(● Use the following IP address:)

IP address:	192 . 168 . 137 . 1
Subnet mask:	255 . 255 . 255 . 0
Default gateway:	. . .

( ) Obtain DNS server address automatically
(● Use the following DNS server addresses:)

Preferred DNS server:	. . .
Alternate DNS server:	. . .

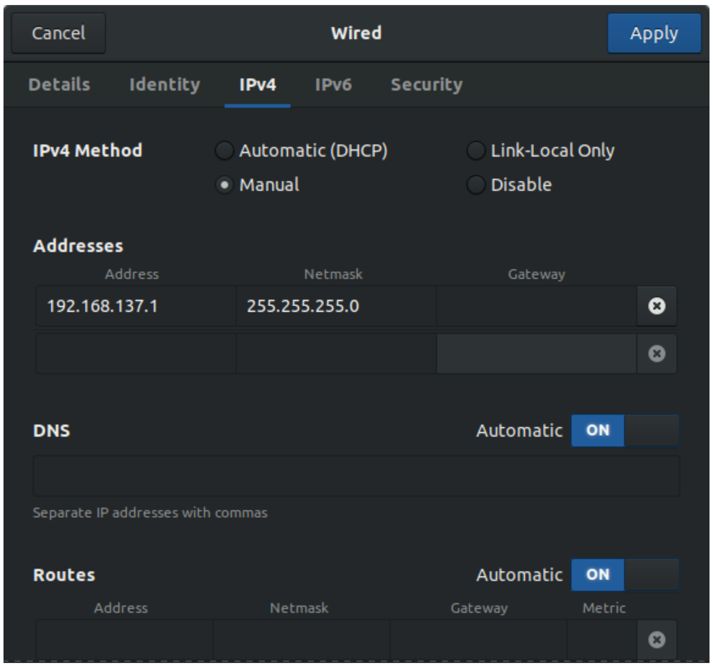
☐ Validate settings upon exit

Figure 10: The IPv4 configuration screen in Windows 10

8.3.2 Ubuntu

To change the IP of your Ethernet port on Ubuntu, complete the following steps:

- Click the network interface icon on the status bar and select Wired Settings
- Click the gear button of the interface you'd like to change
- Select the IPv4 Tab, and change the IPv4 method to Manual
- Under "Addresses" enter in the following:
  - IP Address: 192.168.137.1
  - Subnet Mask: 255.255.255.0
- You can leave Gateway and DNS blank



IPv4 Method: Automatic (DHCP) [✓ Manual] Link-Local Only Disable

Addresses

Address	Netmask	Gateway
192.168.137.1	255.255.255.0	

DNS — Automatic: ON

Separate IP addresses with commas

Routes — Automatic: ON

Address	Netmask	Gateway	Metric

Figure 11: The IPv4 configuration screen in Ubuntu 18.10

8.4 Ensuring connectivity

Sometimes you may want to debug your connection to the Pi. A fast way to do this is via the *ping* command. *Ping* sends a packet to a particular host (in this case the Pi), and measures the time taken for a response from that host.

To use the ping command, open a command prompt window or terminal and type the following:

```
$ ping 192.168.137.15
```

If that host is unreachable (the Pi hasn't booted yet or is incorrectly configured), a message will show that the host is unreachable. If everything was correctly configured, you should get

```
Reply from 192.168.137.15: bytes=32 time<1ns TTL=64
```

This means your Pi and computer are both correctly configured. See section 10 for configuring your Pi for SSH access.

NB: Don't be surprised if you can't ping a Windows machine from your Pi. Windows blocks the specific type of packet required for a ping in the firewall.

9 Advanced Connectivity Options

9.1 Setting a static IP Through Config

Once you've successfully SSH's into your Pi, it's a good idea to configure the networking options in the config files directly.

Use a text editor such as nano to open /etc/dhcpcd.conf as sudo user, and edit it to the following:

```
# Static IP profile for eth0
profile static_eth0
static ip_address=192.168.137.15/24
static routers=192.168.137.1
static domain_name_servers=192.168.137.1 8.8.8.8

# Ethernet interface configuration
interface eth0
fallback static_eth0

# Wireless configuration
interface wlan0
metric 200
```

9.2 Providing your Pi with wireless Internet Access

There are two possible methods of this that will be presented, each with it's own advantages and disadvantages.

The first is using Ethernet passthrough from your computer to the Raspberry Pi. This leaves the WiFi free to host your own access point, and you can host services such as a Node-Red server or media center on the Pi.

The second involves connecting to a wireless network. In the example we give you, we're only going to add a connection to Eduroam. While you could host other services on the Pi when using WiFi connectivity, it would require some access to port forwarding and other things that ICTS unfortunately won't allow.

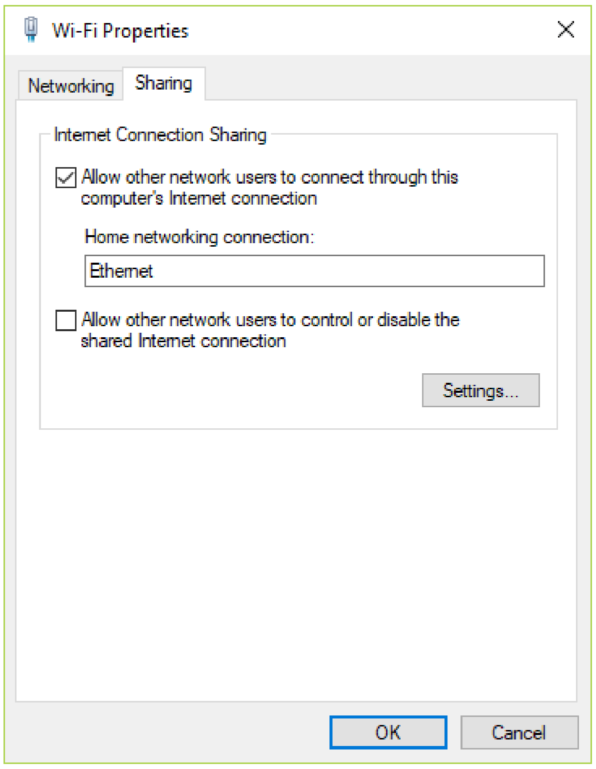
It is recommended that you use Ethernet pass through for the practicals.

9.2.1 Using WiFi to Ethernet passthrough to give your Pi internet access

There may be a situation in which you want your Pi to work as an access point rather than using the WiFi interface to provide the Pi with internet access. In this situation, you need to get internet access through the ethernet port. If you're connected to Windows, you can use network sharing. Complete the following to enable network sharing:

Windows

1. Ensure the Pi is unplugged
2. Right click on your network option in Windows taskbar
3. Select "Open Network & Internet Settings", on the lower right hand side of the screen.
4. Select "Change Adapted Options"
5. Right click on your WiFi network and select "Properties"
6. Click the "Sharing" tab, and enable the first checkbox [4]
7. Select the Ethernet connection that your Pi will be using in the drop down box (Usually just "Ethernet", but may be different if there are multiple Ethernet posts on your system)


Wi-Fi Properties

Networking | Sharing

Internet Connection Sharing

✓ Allow other network users to connect through this computer's Internet connection

Home networking connection: Ethernet

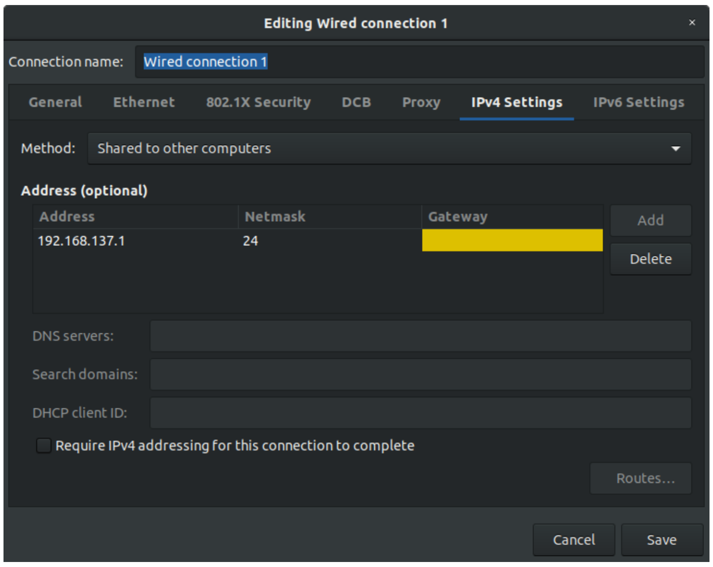
☐ Allow other network users to control or disable the shared Internet connection

Figure 12: Using WiFi to Ethernet passthrough in Windows

[4]This setting is what forces us to have to use the subnet 192.168.137.x.

Ubuntu

1. Ensure the Pi is unplugged and turned off.
2. Open a terminal and run `nm-connection-editor`
3. Select the wired connection you'd like to share your WiFi to
4. Select the IPv4 settings tab
5. Under Method, select "Shared to other computers"
6. under the IP addresses, click "Add" and enter in an address of 192.168.137.1, and a netmask of 24
7. Select "Save", and close the windows
8. Plug in the Pi, start an SSH session and see if you can ping google.co.za



Connection name: Wired connection 1

Method: Shared to other computers

Address (optional)

Address	Netmask	Gateway
192.168.137.1	24	

DNS servers:

Search domains:

DHCP client ID:

☐ Require IPv4 addressing for this connection to complete

Figure 13: nm-connection-editor in Ubuntu 18.10

9.2.2 Connecting to Eduroam - Raspbian Buster (Raspbian Site)

These instructions come from here.

Unfortunately, when you use bleeding edge technology, you may end up cutting yourself. Raspbian Buster updates `wpa_supplicant` to a version that doesn't have support for the authentication method used by Eduroam. So we need to roll back `wpa_supplicant` to an older version. Ensure your Pi has internet access through a method such as Ethernet passthrough, and run the following:

```
$ sudo apt-get remove wpasupplicant
```

Edit /etc/apt/sources.list :

```
$ sudo nano /etc/apt/sources.list
```

And change

```
deb http://raspbian.raspberrypi.org/raspbian/ buster main contrib non-free
    ↪ rpi
```

to

```
deb http://raspbian.raspberrypi.org/raspbian/ stretch main contrib non-free
    ↪ rpi
```

Run

```
$ sudo apt-get update
$ sudo apt-get install wpasupplicant
```

This will install the correct version of `wpa_supplicant`. Check that **version 2.4** is installed by running

```
$ wpa_supplicant -v
```

Now, change the sources file back.

```
$ sudo nano /etc/apt/sources.list
```

And edit the contents to contain

```
deb http://raspbian.raspberrypi.org/raspbian/ buster main contrib non-free
    ↪ rpi
```

Finally, run

```
$ sudo apt-get update
```

To update sources The correct version of `wpa_supplicant` should now be installed, and you can configure your Pi for WiFi as you would if you were using Raspbian Stretch as explained in below.

9.2.3 Connecting to Eduroam - Raspbian Stretch (Vula)

This section provides the instructions on how to configure the Pi to use Eduroam. It is possible to do using the GUI through VNC, but we will not cover that here.

1. SSH into your Pi.
2. Generate a hash for your password. Take note of it as it is needed in a later step

   ```
   $ echo -n your_uct_password_here | iconv -t utf16le | openssl md4
   ```

3. Open /etc/wpa_supplicant/wpa_supplicant.conf

   ```
   $ sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
   ```

4. Edit it so it looks as follows:

   ```
   ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
   update_config=1
   country=ZA

   network={
   ssid="eduroam"
   key_mgmt=WPA-EAP
   identity="studentnumber@wf.uct.ac.za"
   password=hash:generated_hash_from_earlier
   }
   ```

5. Save the file
6. Open `/etc/dhcpcd.conf`

   ```
   $ sudo nano /etc/dhcpcd.conf
   ```

7. Make sure the following lines are in the document:

   ```
   interface wlan0
   metric 200
   ```

8. Reboot your Pi
9. WiFi Generally takes a little longer to initialize than Ethernet, so give it time. You can see if it's ready by running

   ```
   $ ifconfig
   ```

   And seeing if an IPv4 address (”inet”) has been given.

10. Test your configuration by pinging from the WiFi interface:

    ```
    $ ping google.com -I wlan0
    ```

9.3 Debugging Connections

This section is a work in progress. As more common issues are reported by students, this section will be expanded.

9.3.1 Note for Windows Users

If you are having trouble accessing the internet from your Pi using the Ethernet Passthrough technique, it's likely the Windows bridge needs a reset. Change the IPv4 address of your Ethernet port to "Obtain an IP address automatically", then on your WiFi connection, disable sharing, click okay, and then re-enable sharing. Go back to the IP configuration of your Ethernet post, and double check the IP to see if it's been assigned 192.168.137.1. If not, you will need to change it using the "Advanced settings" button.

9.3.2 WiFi

See if you can see wireless networks.

```
$ iwlist wlan0 scan
```

If you cannot connect via WiFi, enter into a shell on the Pi and run:

```
$ journalctl -u wpa_supplicant | grep wlan0
$ journalctl -u dhcpcd.service | grep wlan0
```

This will output the log files and notify you of any incorrect configurations in wpa_supplicant.

The following command will force the interface to be up (if it can be):

```
sudo ifconfig wlan0 up
```

If all else fails, reboot and try again. Some services can be restarted without restarting the Pi, for example:

```
sudo systemctl restart dhcpcd
```

9.3.3 "Failure resolving URLs or "unknown host"

If you try to ping a website and it fails, but pinging a URL works as expected, it is likely an issue with DNS configuration. Open the resolv.conf file:

```
$ sudo nano /etc/resolv.conf
```

And edit it to read the following:

```
nameserver 8.8.8.8
nameserver 192.168.137.1
```

8.8.8.8 is the IP of Google's DNS server. A DNS (Domain Name Service) server is responsible for converting human-readable addressed (for example google.co.za) to something the network architecture can understand (172.217.170.67, in this example).

If you still get this error, try running the following command:

```
$ sudo route add default gw 192.168.137.1
```

Where the IP supplied is the IP of the computer or router you are connected to.

9.4 Configuring the Pi to Act as an Access Point

If you are hosting a server on the Raspberry Pi, or perhaps want to create a WiFi network for guests to connect to, the Pi can act as an access point. This guide comes from https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md and https://frillip.com/using-your-raspberry-pi-3-as-a-wifi-access-point-with-hostapd/.

Note that this WiFi connection will not provide internet access by bridging the Ethernet port (that's something else entirely), but it works well for hosting services on the Pi, such as a Node-Red server.

1. SSH into the Pi, update, and reboot to ensure updates have taken place

   ```
   $ sudo apt-get update
   $ sudo apt-get dist-upgrade
   $ sudo reboot
   ```

2. Wait for the Pi to reboot, and reconnect via SSH. Install hostapd and dnsmasq

   ```
   $ sudo apt-get install hostapd dnsmasq
   ```

3. Configure a static IP in dhcpcd

   ```
   $ sudo nano /etc/dhcpcd.conf
   ```

   Adjust the contents so that the wireless interface is described as follows:

   ```
   interface wlan0
   static ip_address=192.168.4.1/24
   nohook wpa_supplicant
   ```

   Save and close that file, and restart the dhcp service

```
$ sudo systemctl restart dhcpcd
```

4. Run the following commands to create save the original dnsmasq, and create a new file which will be edited:

   ```
   $ sudo mv /etc/dnsmasq.conf /etc/dnsmasq.conf.orig
   $ sudo nano /etc/dnsmasq.conf
   ```

   Place the following in the now open file:

   ```
   interface=wlan0
   listen-address=192.168.4.1
   dhcp-range=192.168.4.2,192.168.4.180,255.255.255.0,24h
   server=8.8.8.8
   domain-needed
   bogus-priv
   ```

   Save and close the file

5. Restart the dnsmasq service

   ```
   $ sudo systemctl reload dnsmasq
   ```

6. Configure hostapd. Open up the configuration file:

   ```
   sudo nano /etc/hostapd/hostapd.conf
   ```

   Place the following configuration in the file. Some assumptions are made about the technical aspects of it, but these are beyond the scope of this course. Note that network name and password **do not** have quotes around them.

   ```
   interface=wlan0
   driver=nl80211
   ssid=TestNetwork
   hw_mode=g
   channel=7
   ieee80211n=1
   wmm_enabled=1
   macaddr_acl=0
   auth_algs=1
   ignore_broadcast_ssid=0
   wpa=2
   wpa_passphrase=TestNetwork
   wpa_key_mgmt=WPA-PSK
   wpa_pairwise=TKIP
   rsn_pairwise=CCMP
   ```

   Save and close the file

7. Next, the system needs to know where to find this configuration file. Open the configuration file:

   ```
   $ sudo nano /etc/default/hostapd
   ```

   Find the line with "#DAEMON_CONF" and replace it with:

   ```
   DAEMON_CONF="/etc/hostapd/hostapd.conf"
   ```

   Save and close this file.

8. Now enable and start hostapd:

   ```
   $ sudo systemctl unmask hostapd
   $ sudo systemctl enable hostapd
   $ sudo systemctl start hostapd
   ```

9. Add routing and masquerade by opening sysctl:

   ```
   $ sudo nano /etc/sysctl.conf
   ```

   And uncomment this line by removing the preceding # symbol:

   ```
   net.ipv4.ip_forward=1
   ```

   Save and close the file

10. Add a masquerade, and save the iptables rule:

    ```
    $ sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
    $ sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
    ```

11. Edit /etc/rc.local and add this just above "exit 0" to install these rules on boot.

    ```
    $ sudo nano /etc/rc.local
    ```

    Place the following just above "exit 0" to install these rules on boot

    ```
    iptables-restore < /etc/iptables.ipv4.nat
    ```

9.5 TTL over USB

Note: The Pi uses 3.3V logic levels. Using a 5V logic module will permanently damage the pins.

This option allows you to use a USB to UART converter such as a FT232R or CP2102. Begin by removing the SD card, and insert it into a computer. Make the following changes on the boot partition:

cmdline.txt: add the following (on the same line)

```
console=serial0,115200
```

config.txt

```
uart_enable=1
dtoverlay=pi3-disable-bt
```

10 SSH

10.1 Enabling SSH

If you have not connected to your Pi and configured it for SSH, you need to do so. SSH is disabled by default on new installations of Raspbian.

To enable SSH, add it as an enabled service in the `raspi-config` menu. If you do not have access to the Pi as yet, do the following:

1. Insert the SD card into a computer
2. Navigate to the BOOT partition
3. Create a file called ”ssh”
4. Your Pi will enable SSH upon next boot

10.2 Using SSH

To use SSH on your Pi, you need to connect to the computer to a network. See Section 8 on various ways that can be done (it is suggested to use Ethernet upon first connection).

Once your Pi is connected to the computer and you have ensured connection (see Section 8.4), use can log in to your Pi via SSH. If you are on a Linux-based system, such as Ubuntu or Mac, you should be able to run the following. Note that the default username is ”pi” and the default password is ”raspberry”.

```
$ ssh <username>@192.168.137.15
```

If you are on Windows, you may need to use PuTTY. Some instances of windows have SSH in the command line, and you can run the command shown above. But if not, you will need to do the following:

- Open PuTTY

- In the ”Hostname” field, enter in ”192.168.137.15”
- Click ”Open”. A terminal window will be opened. If it is the first time you’re SSH’ing into your Pi on this particular computer, you will be asked about the server fingerprint. Click ”Yes” to continue.
- You will be asked for a username and password. The default username is ”pi” and the password is ”raspberry”.
- You should now successfully connected to your Raspberry Pi via SSH

11 VNC

In the previous section, control via SSH was introduced. As previously mentioned, the Raspberry Pi can be used as a standalone desktop computer. However, it is a little impractical to carry around a screen and all the other required peripherals when you’re working with your Pi. This is where VNC comes in.

In computing, Virtual Network Computing (VNC) is a graphical desktop-sharing system that uses the Remote Frame Buffer protocol (RFB) to remotely control another computer.[5]

There are various options for VNC servers. Raspbian comes installed with Real VNC but it needs to be enabled. Other options, such as tightVNC and ultraVNC also exist and can also be used.

1. Activate Real VNC

   - Start by connecting to the Pi via SSH, and opening up raspi-config

     ```
     $ sudo raspi-config
     ```

   - Scroll down using the arrow keys to 5 - Interfacing Options
   - Scroll down to VNC, and select ”Yes” when asked to enable it
   - Select ”Finish”

2. Adjust resolution
   This can be done in two ways:

   - Setting through /boot/config.txt
     Edit config.txt and uncomment these lines:

     ```
     framebuffer_width=1280
     framebuffer_height=720
     ```

[5]Thanks Wikipedia!

- On the Pi desktop in VNC
  Do this if you have already connected to VNC. This is a little more difficult as it required you to play with windows in order to see the buttons you need.
  - Connect to the Pi through VNC.
  - In the desktop menu, go to Preferences - Raspberry Pi Configuration and click the "Set Resolution" button.
  - Select a more appropriate resolution (1280*720 suggested)
  - Select "Okay" and then "Okay". You will be asked to reboot your Pi, do so.

3. Download a viewer
   VNC Viewer is available at this URL:
   https://www.realvnc.com/en/connect/download/viewer/
   Download your choice of app (For example the standalone installer or the Chrome App)

4. Set up the connection
   - Open up VNC viewer
   - Enter the IP of your Pi
   - Click connect
   - You will need log in

5. Configure the Pi
   Upon first boot to desktop, you may be asked to configure some options on the Raspberry Pi. Simply hit next/skip through all of them as they will be configured at a later stage.

12 SCP

SCP or "Secure Copy" is a protocol that allows you to transfer files. To read all the details relating to scp, run `man scp` . The basic format of the command is as follows:

```
$ scp <local_file> <username>@<destination_ip>:<source_directory_location>
```

Some example are as follows (assuming your Pi is located at 192.186.137.15):

- Sending a single file from your computer to the Pi
  ```
  $ scp <file_name> <username>@192.168.1.15:
  ```
- Sending a folder from your computer to the Pi
  ```
  $ scp -r <folder_name> <username>@192.168.1.15:
  ```
- Sending a single file from your computer to a specific folder on the Pi:
  ```
  $ scp <file_name> <username>@192.168.1.15:<destination_directory>
  ```

12.1 Using SCP on Windows

On Windows, SCP may not be enabled. You can get around it by using the `pscp` command, included in the full PuTTY suite (see Section 2.1). It operates in the exact same way, but instead of running `scp`, you need to run `pscp`.

13 FTP

Occasionally you may want to use a GUI to browse and transfer files. The File Transfer Protocol (FTP) is a standard network protocol used for the transfer of computer files between a client and server on a computer network. FTP is built on a client-server model architecture using separate control and data connections between the client and the server.[6]

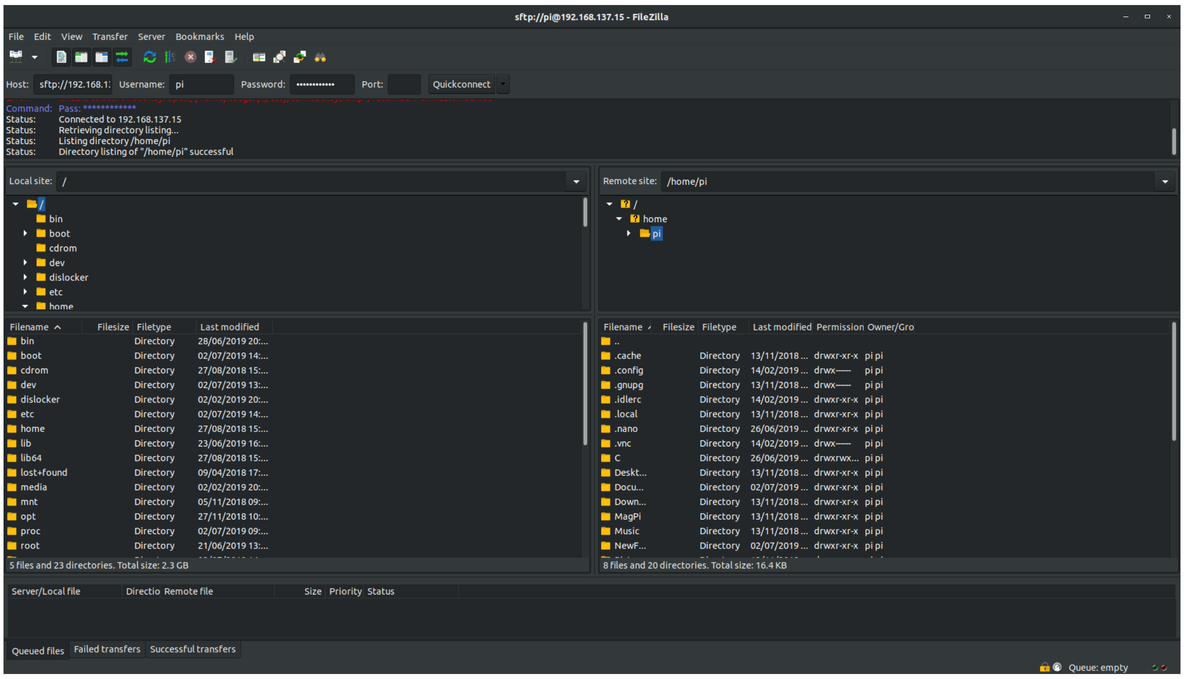
FileZilla is a free to use FTP program available for Windows and Linux. Download and install it.

When you launch FileZilla, a GUI with a few options across the top will be shown. In ”Host”, enter in ”sftp://192.168.137.15”. In ”Username” and ”Password” enter in the username and password you use to SSH in to the Pi. Click ”Quickconnect”

When connecting for the first time, you can choose to save the password or set a master password. Neither of these are necessary. You will be asked to add the server’s host key. Click ”always trust this host, add this key to the cache” and select ”OK”.

You will now be able to browse the files on the Raspberry Pi, and drag and drop from the Pi to your computer, or from the computer to your Pi.

[6]Off Wikipedia


sftp://pi@192.168.137.15 - FileZilla
File Edit View Transfer Server Bookmarks Help
Quickconnect
Queued files
Failed transfers
Successful transfers
Queue: empty

Host: sftp://192.168.1: Username: pi Password: ••••••••••••• Port:

```
Command:  Pass: *************
Status:   Connected to 192.168.137.15
Status:   Retrieving directory listing...
Status:   Listing directory /home/pi
Status:   Directory listing of "/home/pi" successful
```

Local site: /

```
/
  bin
  boot
  cdrom
  dev
  dislocker
  etc
  home
```

Filename	Filesize	Filetype	Last modified
bin		Directory	28/06/2019 20:...
boot		Directory	02/07/2019 14:...
cdrom		Directory	27/08/2018 15:...
dev		Directory	02/07/2019 13:...
dislocker		Directory	02/02/2019 20:...
etc		Directory	02/07/2019 14:...
home		Directory	27/08/2018 15:...
lib		Directory	23/06/2019 16:...
lib64		Directory	27/08/2018 15:...
lost+found		Directory	09/04/2018 17:...
media		Directory	02/02/2019 20:...
mnt		Directory	05/11/2018 09:...
opt		Directory	27/11/2018 10:...
proc		Directory	02/07/2019 09:...
root		Directory	21/06/2019 13:...

5 files and 23 directories. Total size: 2.3 GB

Remote site: /home/pi

```
/
  home
    pi
```

Filename	Filesize	Filetype	Last modified	Permission	Owner/Gro
..					
.cache		Directory	13/11/2018 ...	drwxr-xr-x	pi pi
.config		Directory	14/02/2019 ...	drwx------	pi pi
.gnupg		Directory	13/11/2018 ...	drwx------	pi pi
.idlerc		Directory	14/02/2019 ...	drwxr-xr-x	pi pi
.local		Directory	13/11/2018 ...	drwxr-xr-x	pi pi
.nano		Directory	26/06/2019 ...	drwxr-xr-x	pi pi
.vnc		Directory	14/02/2019 ...	drwx------	pi pi
C		Directory	26/06/2019 ...	drwxrwx...	pi pi
Deskt...		Directory	13/11/2018 ...	drwxr-xr-x	pi pi
Docu...		Directory	02/07/2019 ...	drwxr-xr-x	pi pi
Down...		Directory	13/11/2018 ...	drwxr-xr-x	pi pi
MagPi		Directory	13/11/2018 ...	drwxr-xr-x	pi pi
Music		Directory	13/11/2018 ...	drwxr-xr-x	pi pi
NewF...		Directory	02/07/2019 ...	drwxr-xr-x	pi pi

8 files and 20 directories. Total size: 16.4 KB

Server/Local file	Directio	Remote file	Size	Priority	Status

Figure 14: The FileZilla GUI, connected to the Pi from Ubuntu 18.10

14 Python Tips and Tricks

This chapter contains some useful tips and tricks to writing good Python for embedded systems. You will also find the template you are required to use for your practicals, as well as how to make use of good practices, such as debouncing, or making use of the Raspberry Pi's multicore architecture by implementing threading.

Python, while not as powerful as C, is quickly becoming a common choice for embedded systems developers due to its ease of use[7].

14.1 The RPI.GPIO Library

The RPi.GPIO library is library used on the Raspberry Pi. Documentation for the library can be found here:
https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/

It is included in the environment variables by default, so, in order to use it, you can simply just import it:

```
include RPi.GPIO as GPIO
```

[7]See https://spectrum.ieee.org/at-work/innovation/the-2018-top-programming-languages

14.2 Python Programming Template

This is available on the GitHub Repository.

```
#!/usr/bin/python3
"""
Python Practical Template
Keegan Crankshaw
Readjust this Docstring as follows:
Names: <names>
Student Number: <studnum>
Prac: <Prac Num>
Date: <dd/mm/yyyy>
"""

# import Relevant Librares
import RPi.GPIO as GPIO

# Logic that you write
def main():
    print("write your logic here")


# Only run the functions if
if __name__ == "__main__":
       # Make sure the GPIO is stopped correctly
       try:
           while True:
                  main()
       except KeyboardInterrupt:
              print("Exiting gracefully")
              # Turn off your GPIOs here
              GPIO.cleanup()
       except:
              print("Some other error occurred")
```

14.3 Interrupts

Adding an Interrupt in Python is as simple as:

```
GPIO.add_event_detect(BTN_B, GPIO.RISING, method_on_interrupt)
GPIO.add_event_detect(BTN_PIN, GPIO.FALLING, callback=callback_method(),
                      bouncetime=300)
```

15 Toolchains Compilation and MakeFiles

15.1 Toolchains

A toolchain is a collection of tools that, in this context enables you to write code for an embedded system. For C-based development, a toolchain may consist of the following:

- A text editor or IDE
  This is used to write the code that you plan to run on your embedded system.
- Make
  An automation tool for compiling, linking, and executing files. More on this later.
- Compiler
  Turns the C code you've written into assembly
- Assembler
  Turns assembly code into binary object files
- Linker
  A linker takes one or more object files and converts them into an executable which can run on the target system.

Usually the compiler, assembler and linker are all integrated into one single command which can be run. The most common of these is GCC (GNU Compiler Compiler Collection) which is what will be used in this course.

15.2 Compilation

If you are on the Pi and you wish to compile something, you can run:

```
$ g++ <file>.c -o <compiled_file_name>
```

15.3 Make Files

https://www.gnu.org/software/make/manual/make.html
Make files are a way of simplifying the compilation and build process.

Here's a simplified makefile for Prac 2:

```
.RECIPEPREFIX +=
CC = arm-linux-gnueabihf-g++
CFLAGS = -lm -lrt

```

```
 5 PROG = bin/*
 6 OBJS = obj/*
 7
 8
 9 default:
10     $(CC) $(INCLUDE) $(CFLAGS) -c src/Prac2.c -o obj/Prac2.o
11     $(CC) $(INCLUDE) $(CFLAGS) -c Tools/Timer.cpp -o obj/Timer.o
12     $(CC) -o bin/Prac2 obj/Prac2.o obj/Timer.o $(CFLAGS)
13
14 run:
15     bin/Prac2
16
17 clean:
18     rm -rf $(PROG) $(OBJS)
```

Breaking it down line by line, we have the following:

1. Tells make that we are using spaces instead of tabs
2. CC sets the compiler we're using
3. Set compiler flags
5. Directory containing binaries to run
6. Directory containing object files
9. Define the default rule, called when simply running `$ make`
10. Compile object files
11. Compile library files
12. Link object files into binary
14. Define a new rule to be called when running `$ make run`
15. Run the Prac 2 binary
17. Define a new rule to be called when running `$ make clean`
18. Remove the compiled binaries and object files

15.4 Cross Compilation

When it comes to large programs, or programs that you may need to test with multiple parameters, it is useful to use a more powerful system to compile the program for the Raspberry Pi as opposed to the Raspberry Pi itself. This can save you time and effort.

15.4.1 Requirements

On Windows, download and install the cross compilation framework:
http://gnutoolchains.com/raspberry/
When installing, make sure you select "Add to Path".

On a Linux/Ubuntu-based system, run

```
$ sudo apt-get install libc6-armel-cross libc6-dev-armel-cross
$ sudo apt-get install binutils-arm-linux-gnueabi libncurses5-dev lib32z1
$ sudo apt-get install gcc-arm-linux-gnueabi g++-arm-linux-gnueabi
```

15.4.2 Using cross compilation

Cross compilation is as simple as setting a different compiler in your make file or compilation script. For example, instead of

```
$ g++ <file>.cpp -o <compiled_file_name>
```

You would run

```
$ arm-linux-gnueabihf-g++ <file>.cpp -o <compiled_file_name>
```

15.4.3 Moving the files to the Pi

To move the compiled files to the Pi, SCP (See section 12) is quick and painless solution. Once the compiled file has been copied across, add the execution flag and run the file by running the following commands on the Raspberry Pi:

```
$ chmod +x <compiled_file_name>
$ ./<compiled_file_name>
```

15.5 JetBrains CLion

Thankfully, cross-compilation is a common task and companies know this, so they develop tools to make our lives easier (and make themselves money). In the interest of your education (and the hopes that you spend money on their tools at a later stage), they make these tools accessible to you. For instructions on how to install, configure and use CLion, see Section 5.3.