

1. Title Page:

Assingment 1 : Raspberrypi 4 Configuration

Author(s): Ritesh Tekriwal

Email: rtekri@uw.edu

Date of submission : Jan 29, 2025

2. Abstract:

This Assignment outlines the Configuration for raspberrypi. The goal is to build a complete Embedded Development Environment for Raspberry Pi 4. This environment involves setting up the Raspberry Pi hardware platform and all the associated software. The aim is to have a headless system (a system that operates without a monitor, graphical user interface (GUI) or peripheral devices, such as keyboard and mouse).

3. Introduction:

Raspberry pi is a mini portable computer that needs an Operating System to work. This work is about how we set it up such that it can be utilised to it's full potential. The first part is about getting the right equipment and install the software. The next part is about testing the installation and running some test code to gather some performance metrics for the setup.

3.1 Equipment Required

1. Raspberry Pi board
2. Power supply compatible with the Raspberry Pi model
3. microSD Card recommend at least 16GB (method of programming)
4. SD Card Reader
5. Wireless router (plus Ethernet ports) connected to the internet
6. Ethernet cable
7. HDMI monitor and Cable
8. Laptop/desktop with Ubuntu, Windows or Mac OS
9. USB Keyboard and mouse

3.2 Steps

1. Create boot/installation media drive
2. First Boot of Rpi
3. First Login
4. Update and upgrade all the software
5. Verification
6. Test Experiment

4. Methods:

4.1 Create boot/installation media drive

This step is to get a SD Card ready with an Operating System and then install it on the RaspberryPi Hardware. At the end of this step you will have a SD Card ready with a RaspberryPi OS and should be ready to turn on your Pi Hardware.

0. Insert the SD Card into the SD Card Reader and connect it to your Computer.

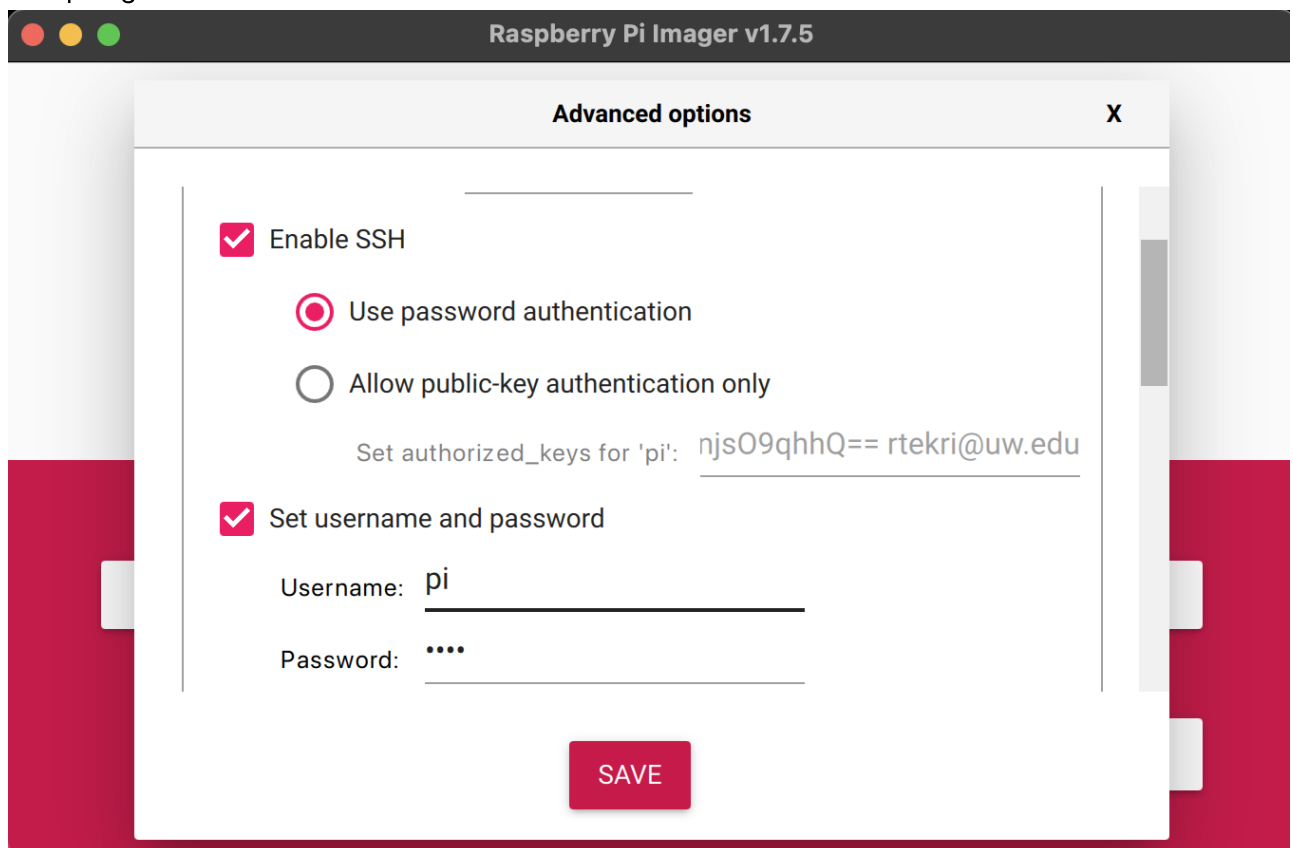
The easiest way to Write an OS image is to use [Pi Imager](#).

This tool helps with the following:

1. Pick the right OS for you ![\[OSSelection\]\(../img/Screenshot 2025-01-20 at 12.08.10.png\)](#)

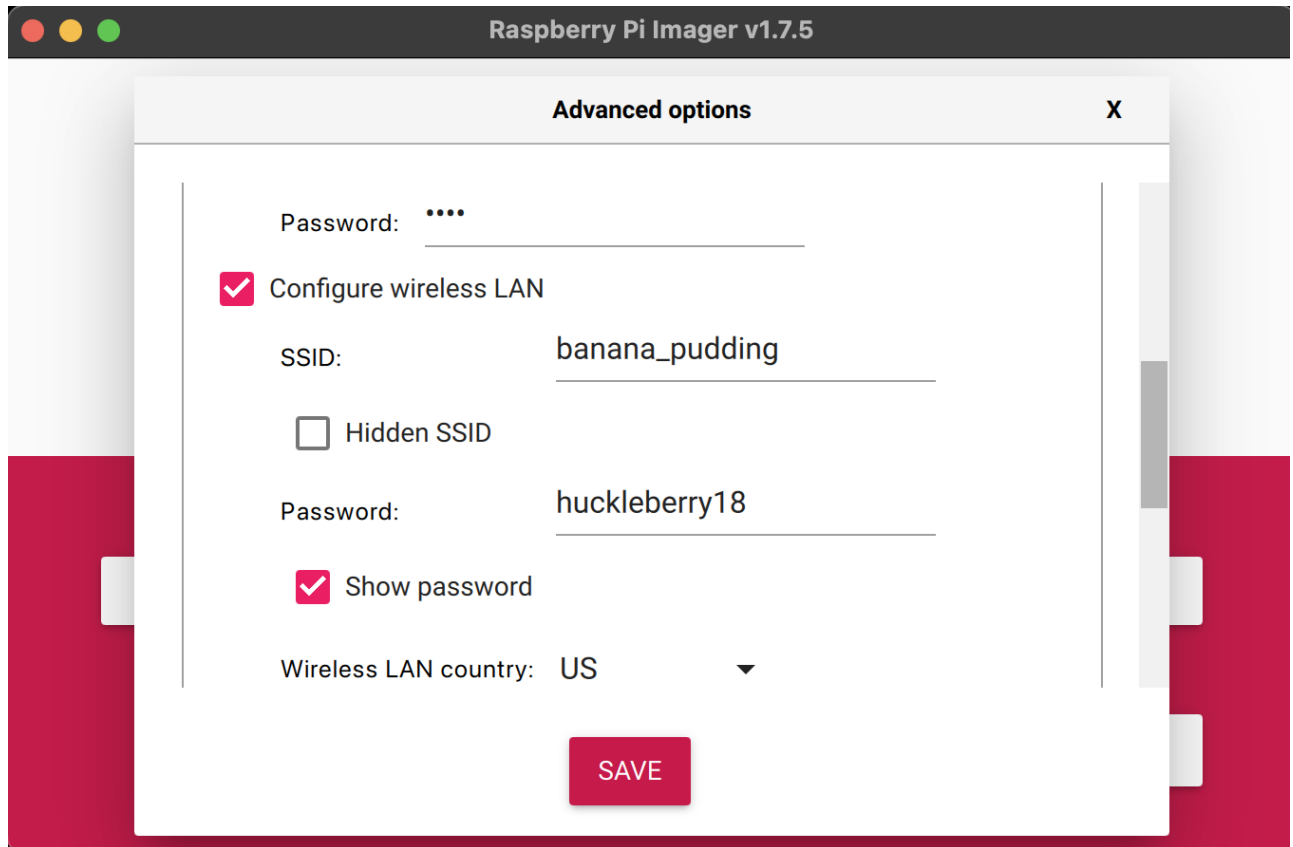
[More info on the different OS](#)

2. Setup Login Credentials and Enable SSH



[More info on SSH](#)

2. Setup Wifi



Once you have specified all the above details, you can go ahead and **Write** the OS image to the SD Card.



This should take roughly 15mins depending upon your internet connection. Once the writing is done, the tool will notify you that it is safe to eject the SD card from your Computer.

4.2 Install an Operating System


0. Ensure the power to the rpi is off and install the SD Card into the Rpi SD Card Slot
1. Power on the Rpi You should be able to see some lights blinking on your Rpi which denotes that the Rpi is trying to boot from the SD Card Image that you just installed.
- 2.

4.3 First Login

There are 2 options to do this:

4.3.1. Desktop Login

This needs you to connect a monitor, keyboard and mouse to your rpi

1. You can turn down the power to the RPi from Step 4.2.2 and connect a monitor, keyboard and mouse using the HDMI and USB connectors on the RPi.
2. Power On the Rpi
3. Once booted up, you should be able to see a screen load up and the homescreen on your newly set Rpi.  DesktopLogin

4.3.2. Headless Login

Prerequisite: Setup SSH on your PC

If you do not need the Desktop variant and are cool enough with the headless version, you can do that in two ways:

1. Using Ethernet
 1. Connect an Ethernet Cable to your PC and the RPi
 2. Open Terminal or SSH Terminal on your PC
 3. Run the following command. Replace the **raspberrypiName** with the name you had set in Step 4.1.2

```
ssh <raspberrypiName>.local
```

```
+ ~/Work/GitClones/raspi git:(main) ✗ ssh pi@raspberrypi.local
The authenticity of host 'raspberrypi.local (192.168.4.49)' can't be established.
ED25519 key fingerprint is SHA256:aly7W7///S9tQtC21Csfqmw+3v3QG3v4hSrjY12o0A.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'raspberrypi.local' (ED25519) to the list of known hosts.
pi@raspberrypi.local's password:
Linux raspberrypi 6.1.21-v8+ #1642 SMP PREEMPT Mon Apr  3 17:24:16 BST 2023 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Oct 22 14:09:52 2024
pi@raspberrypi:~$
```

4. If all goes well, you should be prompted to key in the password that you had set in 4.1.2

2. Using WiFi

1. If you configured a WiFi network in Step 4.1.2, once the RPi is powered on, it should connect to the network automatically.
2. If it is connected, you should be able to ssh into the RPi wirelessly.
3. Run the following command. Replace the **raspberrypiName** with the name you had set in Step 4.1.2

```
ssh <raspberrypiName>.local
```

```
+ ~/Work/GitClones/raspi git:(main) ✗ ssh pi@raspberrypi.local
The authenticity of host 'raspberrypi.local (192.168.4.49)' can't be established.
ED25519 key fingerprint is SHA256:aly77W7///S9tQtC21Csfqmw+3v3QG3v4h5rjY12o0A.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'raspberrypi.local' (ED25519) to the list of known hosts.
pi@raspberrypi.local's password:
Linux raspberrypi 6.1.21-v8+ #1642 SMP PREEMPT Mon Apr  3 17:24:16 BST 2023 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Oct 22 14:09:52 2024
pi@raspberrypi:~$
```

If this commands timeouts, this means that your computer doesn't identify the raspberrypi name and you will have to know the IP address of your Rpi. Repeat all the steps 4.3.1 and once you have logged in, run the following command

```
hostname -I
```

```
+ ~/Work/GitClones/raspi git:(main) ✗ ssh pi@raspberrypi.local
The authenticity of host 'raspberrypi.local (192.168.4.49)' can't be established.
ED25519 key fingerprint is SHA256:aly77W7///S9tQtC21Csfqmw+3v3QG3v4h5rjY12o0A.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'raspberrypi.local' (ED25519) to the list of known hosts.
pi@raspberrypi.local's password:
Linux raspberrypi 6.1.21-v8+ #1642 SMP PREEMPT Mon Apr  3 17:24:16 BST 2023 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Oct 22 14:09:52 2024
pi@raspberrypi:~$
pi@raspberrypi:~$
pi@raspberrypi:~$ hostname -I
169.254.92.2 192.168.4.49 fddf:b1d0:ccca:1:ed83:e1bd:d2b6:a427
pi@raspberrypi:~$
```

192.168.4.49 is the IP address you are looking for.

4. Now you are ready to SSH into your RPi wirelessly. Run the following command

```
ssh pi@192.168.4.49
```

Enter the password when prompted and you are in. You are successfully connected to the RPi wirelessly.

```
+ ~/Work/GitClones/raspi git:(main) * ssh pi@192.168.4.49
The authenticity of host '192.168.4.49 (192.168.4.49)' can't be established.
ED25519 key fingerprint is SHA256:aly77W7///S9tQtC21Csfqmw+3v3QG3v4hSrjY12o0A.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:27: raspberrypi.local
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.4.49' (ED25519) to the list of known hosts.
pi@192.168.4.49's password:
Linux raspberrypi 6.1.21-v8+ #1642 SMP PREEMPT Mon Apr  3 17:24:16 BST 2023 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jan 13 21:40:36 2025 from 192.168.4.24
pi@raspberrypi:~$
```

4.4 Update and upgrade all the software

You will need an active internet connection on your RPi to perform this step

1. Package Manager Update The package manager uses the package list to know what packages need updating and where it can download them. It polls package repositories for these lists.

Run the following command on your device to update the package list.

```
sudo apt update
```

2. Full Update Once the package list has been updated, we can run a full upgrade on any available package.

```
sudo apt full-upgrade
```

3. Optional Kernal Updates

```
sudo apt install raspberrypi-kernel-headers
```

4. Reboot

Once done with the above, you can reboot the RPi for the updates to work effectively.

```
sudo reboot
```

4.5 Verification

Verification is the process to discover the truth, accuracy and validity. Create a file on you host PC call it something like hellow.c (basically a standard hello world test program), write the following code:

```
#include <stdio.h>

int main(void){
    printf("hello_world\n");

    return 1;
}
```

From your host PC use **scp** to copy over the files to the Raspberry Pi.

Once the file is safely across, log into the Raspberry Pi and go to the directory where the file was placed.

```
+ ~/Work/GitClones/raspi git:(main) ✗ scp hellow.c pi@192.168.4.49:/home/pi/Work/.
pi@192.168.4.49's password:
Permission denied, please try again.
pi@192.168.4.49's password:
hellow.c
+ ~/Work/GitClones/raspi git:(main) ✗ ssh pi@192.168.4.49
pi@192.168.4.49's password:
Linux raspberrypi 6.1.21-v8+ #1642 SMP PREEMPT Mon Apr  3 17:24:16 BST 2023 aarch64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jan 13 21:41:46 2025 from 192.168.4.24
pi@raspberrypi:~ $ ls
Bookshelf  Work
pi@raspberrypi:~ $ cd Work/
pi@raspberrypi:~/Work $ ls -ltr
total 4
-rw-r--r-- 1 pi pi 81 Jan 13 21:46 hellow.c
pi@raspberrypi:~/Work $
```

Verify if the file looks okay and then compile hellow.c. You can run the following to view the contents of the file

```
cat hellow.c
```

If the files looks right, you can compile it by running the following

```
gcc hellow.c -o hellow
```

Next you can execute the compiled code to see the results

```
./hellow
```

```

pi@raspberrypi: ~/Work (ssh)
pi@raspberrypi:~/Work $ cat hellow.c
#include <stdio.h>

int main(void){
    printf("hello_world\n");

    return 1;
}pi@raspberrypi:~/Work $ gcc hellow.c -o hellow
pi@raspberrypi:~/Work $ ./hellow
hello_world
pi@raspberrypi:~/Work $

```

If you are successfully able to see the results, it is safe to conclude that your RPi setup is working.

Optional Method to copy files to your RPi

rsync

You can run the following command to effectively keep the files up to date between your host computer and RPi

```
rsync -a --exclude="/.*" pi@192.168.4.49:<RPi Folder Path>. <Host PC Folder>/.
```

```

~/Work/GitClones rsync -a --exclude="/.*" raspi pi@192.168.4.49:/home/pi/Work/raspi
pi@192.168.4.49's password:
~/Work/GitClones

```

This keeps the files in the folder names *raspi* on the Host PC up to date with the same folder on the RPi. The exclude arguments ignore any hidden folder such as git history or files

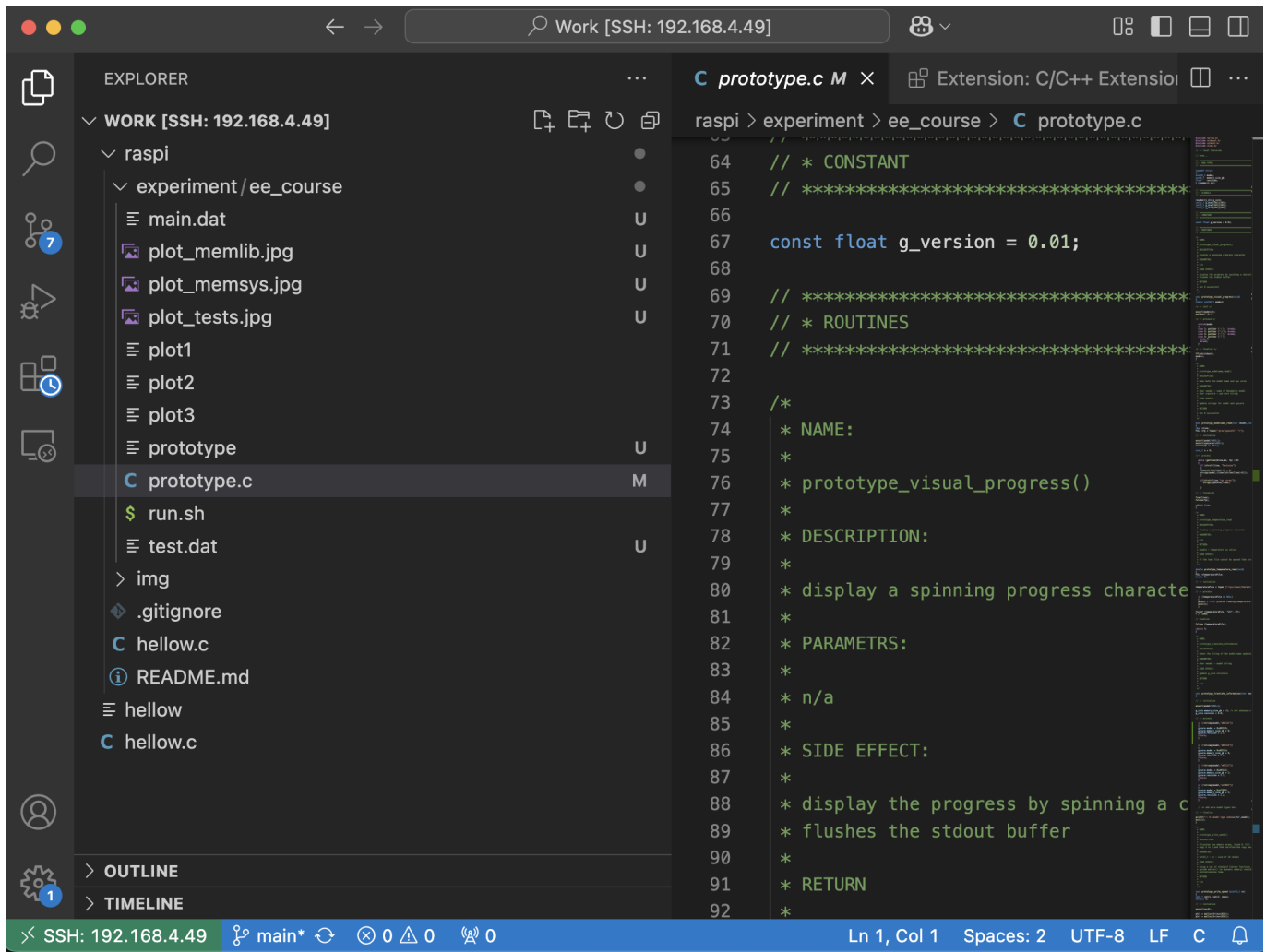
4.5 Test Experiment

1. Setup Version Control (Github) First things first, a github repository was created to keep all the code in one place.

Github Repo

<https://github.com/tekritesh/raspi>

Optional Setup VSCode SSH If you are using VSCode as your IDE for coding, you can install the [Remote SSH Client](#) which lets you edit code and access all the files from the RPi as if it were host computer.



```
64 // * CONSTANT
65 // *****
66
67 const float g_version = 0.01;
68
69 // *****
70 // * ROUTINES
71 // *****
72
73 /*
74  * NAME:
75  *
76  * prototype_visual_progress()
77  *
78  * DESCRIPTION:
79  *
80  * display a spinning progress character
81  *
82  * PARAMETRS:
83  *
84  * n/a
85  *
86  * SIDE EFFECT:
87  *
88  * display the progress by spinning a c
89  * flushes the stdout buffer
90  *
91  * RETURN
92  *
```

2. Downloaded ee course.zip from Canvas: located files/Assignments/ee course.zip was unzipped in this repo [here](#)
3. Unzip, i.e., unzip ee course.zip
4. Run **rsync** to copy the experiment code onto the RPi.

```
rsync -a --exclude="/.*" pi@192.168.4.49:/home/pi/Work/raspi/. raspi/.
```

5. Install Gnuplot on RPi

```
sudo apt install gnuplot
```

6. Execute the experiment shell script

```
./run.sh
```

This errors out since it is not able to find the version of RPi

```
pi@raspberrypi:~/Work/raspi/experiment/ee_course $ ./run.sh
**** initialize
rm: cannot remove 'main.dat': No such file or directory
rm: cannot remove 'test.dat': No such file or directory
**** compile code
**** execute test - 5 to 10 minutes
-- I: Raspberry Pi Simple Test [ver:0.01]
-- E: model type unknown d03115**** graph the data
Fontconfig warning: ignoring UTF-8: not a valid region tag
"plot1" line 19: warning: Skipping data file with no valid points
"plot1" line 19: warning: Skipping data file with no valid points

plot "main.dat" using 1:4 title 'working with the memory system' with lines,      "main.dat" using 1:5 title 'working against the memory system' with lines      ^
"plot1" line 19: x range is invalid

Fontconfig warning: ignoring UTF-8: not a valid region tag
"plot2" line 17: warning: Skipping data file with no valid points
"plot2" line 17: warning: Skipping data file with no valid points
"plot2" line 17: warning: Skipping data file with no valid points

plot "main.dat" using 1:2 smooth sbezier title 'time to completion' with lines,      "main.dat" using 1:3 smooth sbezier title 'relative temperature C' with lines,      "main.dat" using 1:($2*$3) smooth sbezier t
itle 'est. energy' with lines
^
"plot2" line 17: x range is invalid

Fontconfig warning: ignoring UTF-8: not a valid region tag
"plot3" line 21: warning: Skipping data file with no valid points

plot "test.dat" using 2:xtic(1) title 'time taken' with histograms      ^
"plot3" line 21: x range is invalid

**** complete
```

On further investigation it was found that the error was in [prototype_translate_information](#) function where the RPi used did not match any of the given models. So appropriate changes were by adding this to the [code] https://github.com/tekritesh/raspi/blob/main/experiment/ee_course/prototype.c#L257

```
if (!strcmp(model,"d03115"))
{
    g_core.model = 0xd03115;
    g_core.memory_size_gb = 8;
    g_core.revision = 1.5;
    return;
}
```

These details can be found by running the following command on the RPi Terminal

```
cat /proc/cpuinfo
```

```

pi@raspberrypi:~/Work/raspi/experiment/ee_course $ cat /proc/cpuinfo
processor       : 0
BogoMIPS      : 108.00
Features      : fp asimd evtstrm crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant   : 0x0
CPU part      : 0xd08
CPU revision  : 3

processor       : 1
BogoMIPS      : 108.00
Features      : fp asimd evtstrm crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant   : 0x0
CPU part      : 0xd08
CPU revision  : 3

processor       : 2
BogoMIPS      : 108.00
Features      : fp asimd evtstrm crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant   : 0x0
CPU part      : 0xd08
CPU revision  : 3

processor       : 3
BogoMIPS      : 108.00
Features      : fp asimd evtstrm crc32 cpuid
CPU implementer : 0x41
CPU architecture: 8
CPU variant   : 0x0
CPU part      : 0xd08
CPU revision  : 3

Hardware      : BCM2835
Revision      : d03115
Serial        : 10000000726737ba
Model         : Raspberry Pi 4 Model B Rev 1.5

```

Rerunning the shell script, the code executed successfully and generated the expected graphs.

```

pi@raspberrypi:~/Work/raspi/experiment/ee_course $ ./run.sh
**** initialize
**** compile code
**** execute test - 5 to 10 minutes
-- I: Raspberry Pi Simple Test [ver:0.01]
-- I: MEM 868
-- I: REV 1.5
-- I: TEM (baseline): 41.868 C (WARM)
-- I: TST 500
/..... [100] 5.228825 sec
/..... [200] 15.916779 sec
/..... [300] 26.762164 sec
/..... [400] 37.752824 sec
/..... [500] 48.949077 sec
-- I: TEM 11.688 C
**** graph the data
Fontconfig warning: ignoring UTF-8: not a valid region tag
Fontconfig warning: ignoring UTF-8: not a valid region tag
Fontconfig warning: ignoring UTF-8: not a valid region tag
**** complete
pi@raspberrypi:~/Work/raspi/experiment/ee_course $ ls
main.dat plot1 plot2 plot3 plot_memlib.jpg plot_memsys.jpg plot_tests.jpg prototype prototype.c run.sh test.dat
pi@raspberrypi:~/Work/raspi/experiment/ee_course $

```

5. Results:

Fig1 Total Time to Run the Tests

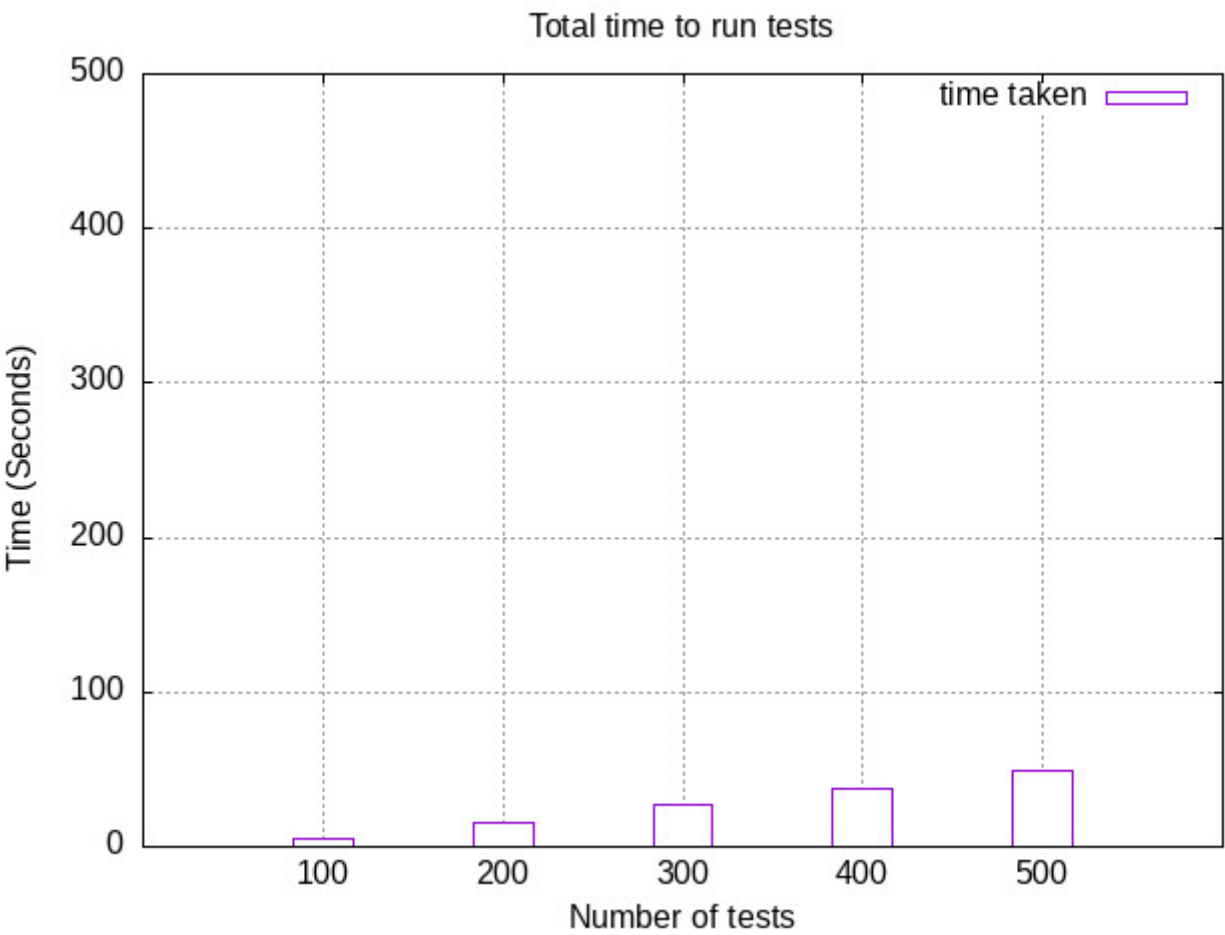


Fig2 Memory Library Tests

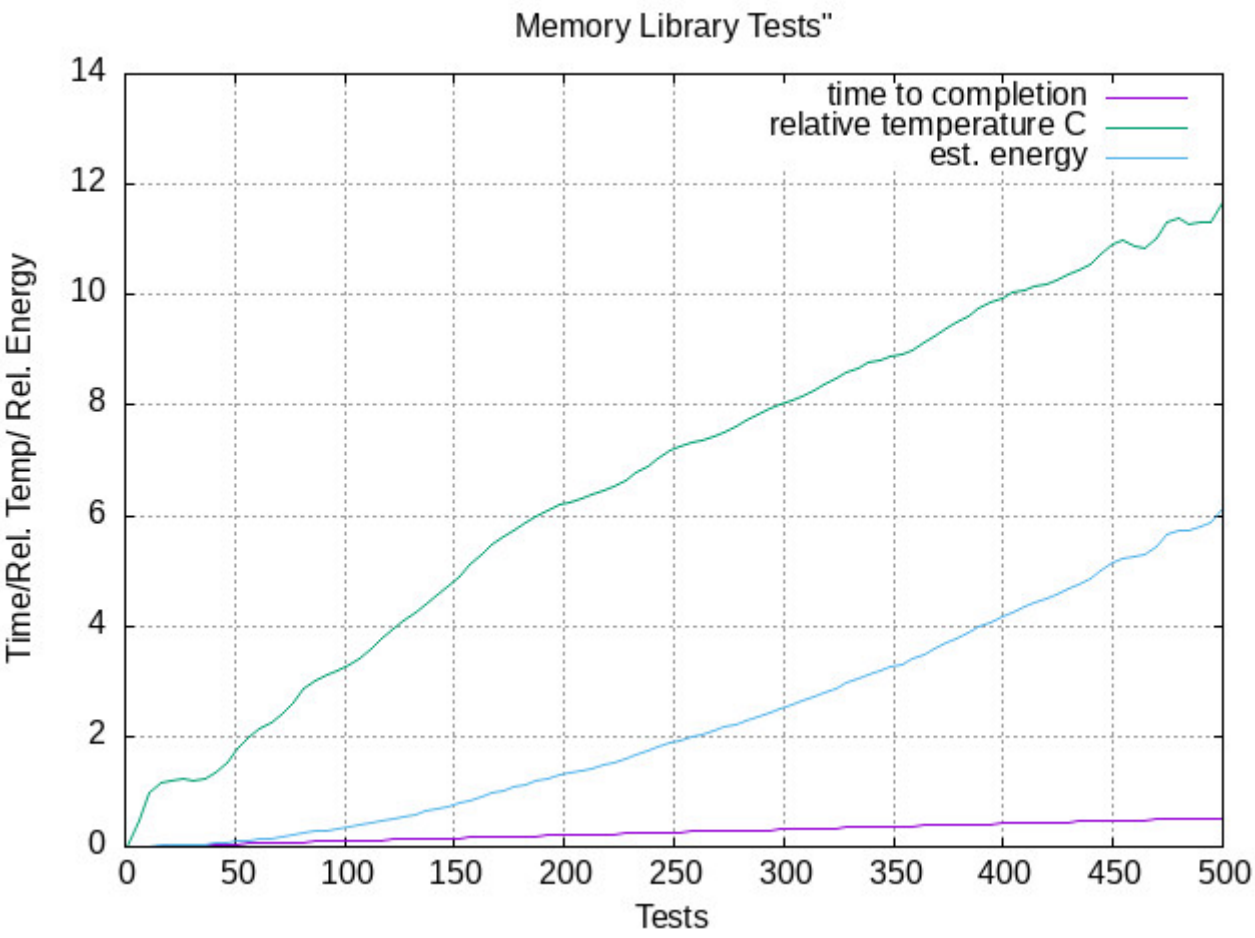
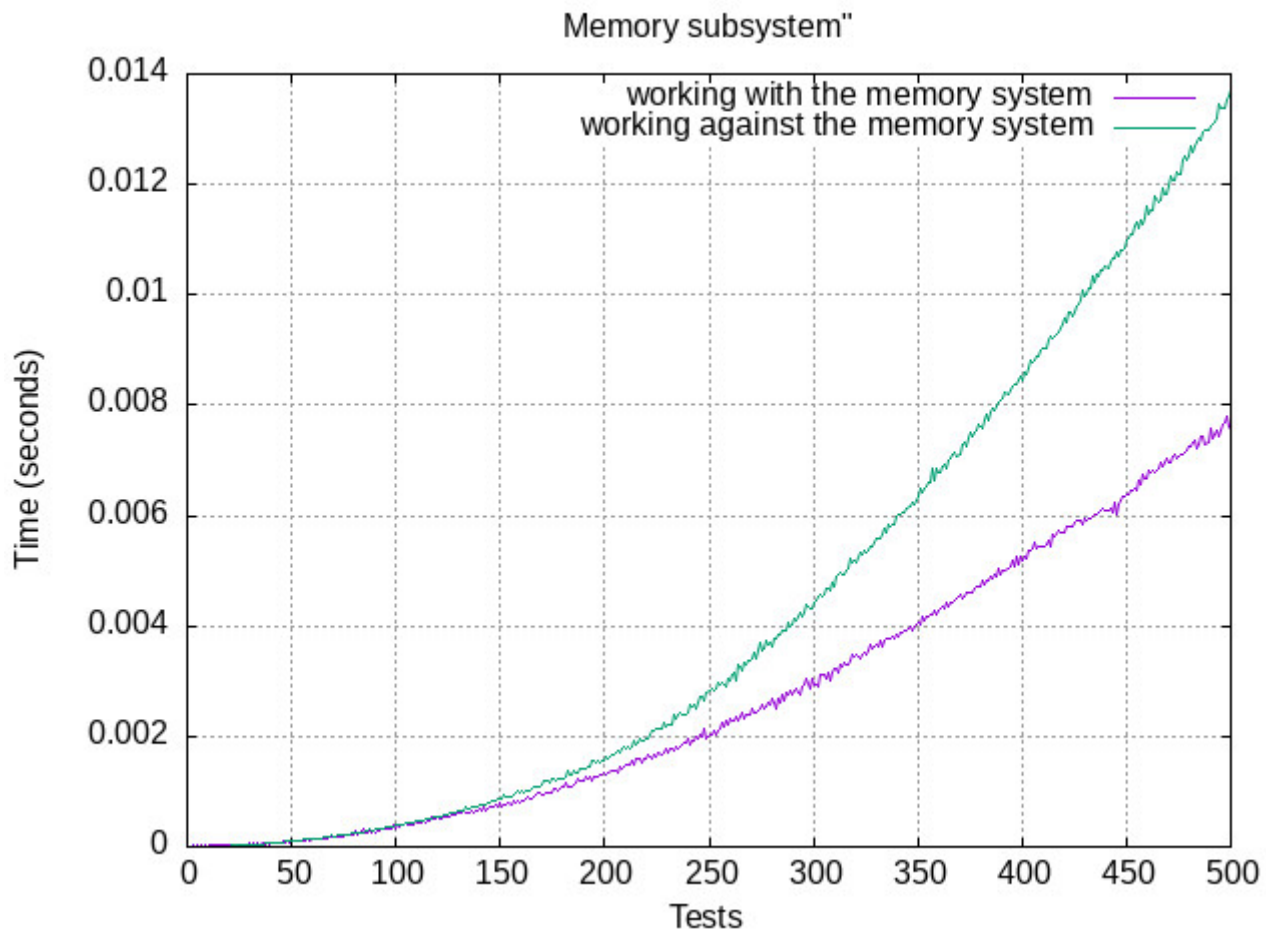


Fig3 Memory Subsystem Tests



6. Discussion:

- From Fig1 we can see as the number of tests increases, the time taken to execute them also increases.
- From Fig2 we can see at the number of tests increase on the X-axis, the temperature as well as the energy consumed by the CPU increase almost linearly with the number of tests.
- From Fig3 we see that the memory usage is significantly higher when the matrix is not loaded from cache.

7. Conclusion:

- Summarize the main findings • Emphasize the significance of the results

8. References:

- [Wikipedia](#)
- [RaspberryPi](#)
- [StackOverFlow](#)

9. Acknowledgments:

10. Appendices:

11. Figures and Tables: