Raspberry PI (RPI) 3 Model B

EEE3096S RPi LAB MANUAL

CHAPTER 1

Author: Do Yeou Ku

University of Cape Town

Revised for EEE3096S by S. Winberg and O. Konan, June 2018

Overview of Practicals

Welcome to the EEE3096S RPi lab manual. This document provides the collection of practicals related to the Raspberry Pi that we will be working though in the course. This first part *outlines* what you are expected to do for each pratical assignment, in particular how to submit or demonstrate your results at completion of the assignment. Please note that you should always get into the habit of having a readme.txt file to indicate who submitted the files, e.g. if it is a group assignment make sure all the student numbers for the group are clearly indicated in this file.

Contents

Practical 1: Headless RPI-3B

The objective of the practical 1 is to make your RPI-3B headless and make a new folder called <*student_number*> remotely via SSH.

1. Take screenshots of your command prompt (SSH) after executing following commands:
   a. `ifconfig` (etho0 must be visible)
   b. `lscpu` (number of core must be visible)
   c. `ls` (must show the new file you created)
2. Answer the following questions:
   a. Define a headless RPI (2)
   b. Explain the meaning of `iface`, `eth0`, `inet`, and `static` in detail. (4)
   c. Explain the difference between the subnet mask `255.255.255.0` and the IP address `192.168.1.15` (4)

At the end of the practical 1, you are to submit one .pdf file including your screenshots and answers on Vula. The pdf file must have the following format
`prac_<prac_number>_<std_number>.pdf`

Chapter 1 Introduction to RPI

1.1. Background

The Raspberry Pi (or RPI as we might refer to it more concisely in this document) is a complete "credit-card sized computer"[1]; it's a single board computer (SBC) meaning that the whole computer system is on one PCB. The fundamental enabling component of the RPi is its system-on-chip (SOC) around which it is built. A SOC is an Integrated Circuit (IC) that combines the various essential parts of a computer system into one chip – these are various types of SOCs, not necessarily all implement a computer system, but the RPi uses the Broadcom that integrates most of the pieces needed in a GPU-enabled computer. The SOC cannot operate along of course, it at least needs the physical ports and power connections that will link it to other peripherals (such as the network) and to power. Although the SOC is the main component that enables the RPI to be a low-cost computer system, there is inherent difference between a SOC and a normal PC type computer platform. To put it simply, the RPi can be thought of more as a microcontroller development (or evaluation) board than a computer you might use for doing Word Processing or doing scientific calculations – but nevertheless the RPi can still do a pretty good job of these more PC-type tasks.

The RPi, just like any other computer, requires an operating system (OS) to make it useful. An OS (as you should know from Comp Sci) is a type of system software that manages the hardware and software resources of your computer to provide common services that computer programs need in order to run. The Raspbian OS is the one commonly used with the RPi, and is the one we will use – and abuse (by hacking the kernel and doing other fun things) – in our pracs. Raspbian is a version of the more commonly used (for PCs) Debian Linux OS; Raspbian is provided and maintained by the Raspberry PI Foundation as the primary OS for their family of RPis.

The next section will dive into some of the hardware specs of the RPi 3.

[1] https://www.raspberrypi.org/

1.2. RPi-3B Specification

The Raspberry Pi 3Model B has the following main components and is depicted in Figure 1:

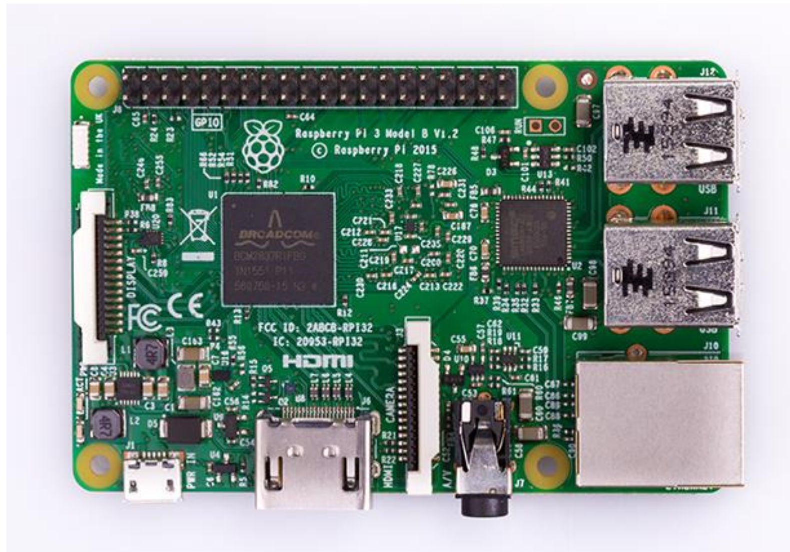


Figure 1 RPI-3B

- 1.2GHz 64-bit quad-core ARMv8 CPU
- 1GB RAM
- 802.11n Wireless LAN
- Bluetooth 4.1
- Bluetooth Low Energy (BLE)
- 4 USB ports
- Full HDMI port
- Ethernet port
- Combined 3.5mm audio jack and composite video
- Camera interface (CSI)
- Display Interface (DSI)
- Video Core IV 3D graphics core
- Micro SD card slot
- 40 GPIO pins

The RPi3-B has four independent cores which read and execute 32-bit and/or 64-bit instructions in its central processing unit, which is clocked at 1.2GHz and whose architecture is that of the ARMv8-A architecture. It has 1GB of random-access memory. It supports wireless communication such as IEEE's WLAN 802.11n, as well as Bluetooth 4.1 (that in turn is compatible with Bluetooth 4.0 and BLE). It has four USB ports, a full HDMI port and an Ethernet port on board, in addition to 40 GPIO pins that can be control from software.

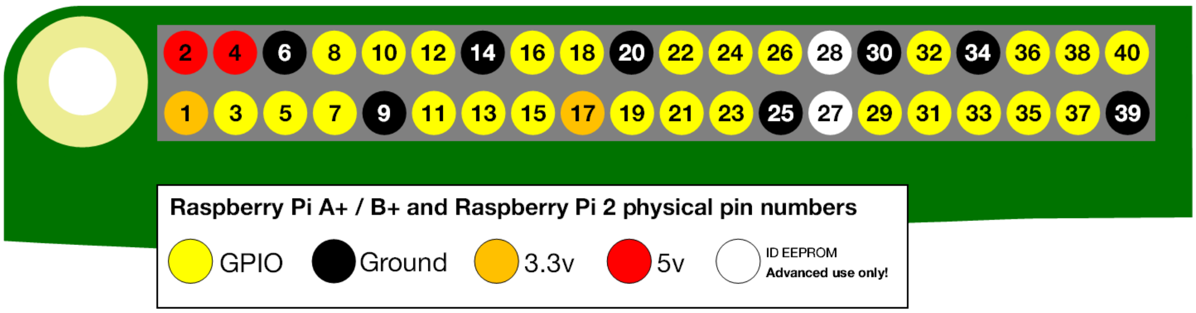
When programming the GPIO pins, there are two methods to refer to them: physical numbering and GPIO numbering. GPIO numbering refers to the pin numbering from the RPi board's perspective, whereas physical numbering refers to the physical hardware pin number on the board (by counting pins on the header).

As some programming language support only one of the two methods to refer to the GPIO pins, it is important to check which of the method you are using or the sample code that you might be looking at is using. For this course, *GPIO numbering* will be used.

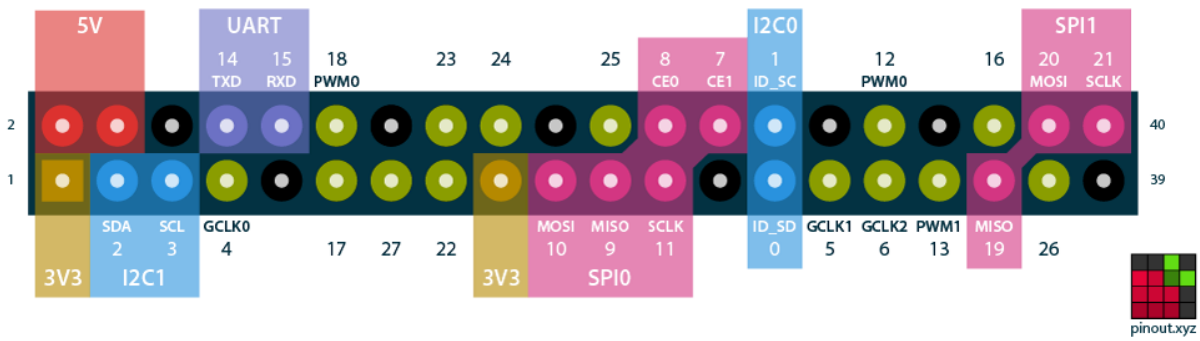
Note

Note: C supports only GPIO numbering whereas Python provides a choice between the two.

Figure 2 describe the pin numbering of the RPi's main "extended GPIO" header. This is the header shown on the top of the board in Figure 1. Note that not all the pins are GPIO (i.e. software programmable pins – some of these pins are linked to power or ground and can thus be used to provide power to peripherals that might be connected to theRPi (and hence the reason why the designers called the extended GPIO header in the product manual). The physical pin numbering is shown at the top of the Figure 2 and the GPIO software numbering is shown at the bottom of the figure.



(a)



(b)

Figure 2 RPI GPIO pin numbering (a) physical numbering (b) GPIO numbering

From the diagram, the pin functions of the RPi's 40 pin header are summarised as follows:

- 2x 5V voltage pins
- 2x 3V3 voltage pins
- 8x Ground pins
- 26x GPIO pins (D/A input and D output) of which 14x support alternate functions such as:
    - 3x GCLK
    - 3x PWM
    - 1x UART: TX and RX
    - 2x I2C: SDA and SCL
    - 2x SPI: SCLK, MOSI, MISO

The RPi board is typically powered through a micro Universal Serial Bus (micro USB) which provides a 5V regulated supply of up to 500 mA. I hope you remember the most basic law in electrical engineering: Kirchhoff's voltage and current law. It is bad practice to disregard the current rating of any device or shield that you will connect to RPi board! If your current draw is above the supplied current rating, it will kill your RPi board! And may also kill the USB port on your PC as well if you are powering the RPi from your PC's USB port.

1.3. The Raspbian image

The RPi by default tries to boot its OS from a microSD card, if one is inserted in the SD card slot. Usually you need an SD card of at least 4GB to be useful. But how does one get the image set up on the SD card? No worries, we're going to go through doing so in this section.

There are two typical ways to write the Raspbian image onto a microSD card, namely: 1) utilizing NOOBS or 2) compiling Raspbian source code. As this section of the manual is introduction to Raspberry Pi, we will start by using the easier option which is NOOBS.

"NOOBS" is an easy operating system installer which contains Raspbian. It also provides a selection of alternative operating systems which are then downloaded from the internet and installed" – Raspberry Foundation

The Raspberry Pi (RPI) is a complete system on chip (SOC). A SOC is an integrated circuit (IC) that include crucial parts of a computer. Although SOC enables RPI to perform a low-cost computer system, there is inherent difference between SOC and a computer processor. In this cases, the SOC essentially contains the microprocessor and w a while lot more, which you will find out more in the lectures.

The are two versions of this part of the lab : starting 'headless' with just ssh and no monitor, use that approach if in the lab, using the NOOBS_UCT.img (see link to be announced on Vula). Or follow the headed approach (which is the first principles approach using the original files from the RaspberryPi website).

Headless Approach

Start by getting a copy of the NOOBS_UCT.img file on your PC. Then use the Win32 Diskimager to install this image into the SD Card for your Raspberry Pi. Then boot the RPi using this SD card. Connect the lab PC's second network card to the RPi using the network cable provided.

Plugin and power up your RPi. Wait for a few seconds (about 30s to be sure) for it to boot. From your Windows machine start Putty, or if in Linux use ssh. Connect via ssh to the following IP address:

192.168.0.10

You may need to set up the network card on your Windows/Linux machine to have a static IP, in which case please apply the following settings to your PC network card:

Static IP: 192.168.0.1

Netmark: 255.255.255.0

You may need to turn off (disable) and turn on (enable) your network card to make sure the new address is in effect. Not retry the SSH connection to the RPi.

Note that the default username and passwords are:

Username: pi

Password: raspberry

Not continue to Section 1.4 to continue the prac in headless mode.

Headed Approach

- Download NOOBS.zip from https://www.raspberrypi.org/downloads/noobs/ (or get from tutor, it is a bit big to fit on Vula)
- Unzip the downloaded zip file.
- Prepare microSD card.
- Copy the content of NOOBS.zip to microSD card.
- Boot Raspberry Pi with the microSD card with NOOBS on.
- Connect up the Ethernet capable to the second network port on your PC
- Connect the Pi to HDMI-enabled screen, keyboard and a mouse. (you may need to borrow an HDMI->VGA adaptor, the tutors have each been allocated one.)

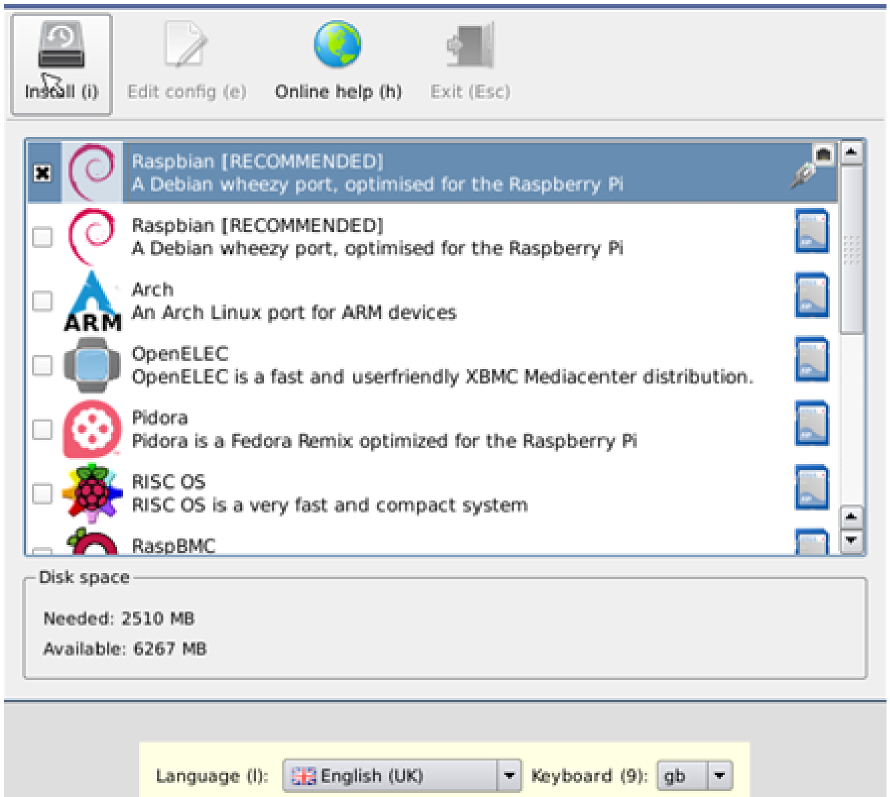


Figure 3 NOOBS screen

Select Raspbian [RECOMMENDED] and English(US) and click Install. This will automatically install Raspbian OS onto your microSD card.

There are two indicators on Raspberry Pi: red power led and green activity led. Whilst the red power led is always on, the green activity led flickers when there is data transmission to the operating systems microSD card. If the green activity led is not flickering but solid, it means there is a problem with the microSD card.

1.4. Basic Linux commands

As mentioned before, Raspbian is based on Debian (Linux), therefore commands used on the terminal of Raspbian is almost identical to the commands of Linux. Below table contains few basic commands or command prefixes that are essential to use Raspberry Pi terminal.

`$ sudo`	Command run as root (admin) user
`$ sudo raspi-config`	Open Raspberry Pi configuration tool
`$ sudo halt`	End all operation
`$ sudo reboot`	Reboot Raspbian
`$ ifconfig`	Display network configuration
`$ ls`	List all folders under the directory
`$ ls -l`	List all folders in detail under the directory
`$ lscpu`	List machine CPU specification/properties
`$ cd`	Change directory
`$ cd ..`	Navigate up one file level
`$ cd ~`	Navigate to home
`$ pwd`	View current working directory
`$ mkdir /$<folder_name>$`	Make a new folder called <folder_name> in the current working directory
`$ mv /$<file_name>$ /$<new_directory>$`	Move the file <file_name> to <new_directory>
`$ cp /$<file_name>$ /$<destination_directory>$/$<copy_name>$`	Copy the file <file_name> to <destination_directory> with file name <copy_name>

1.5. Headless RPI

A headless device refers to a device which is accessed remotely over the local network (or internet) instead of being controlled by direct hardware input. Configuring the Raspberry Pi as a headless device allows user to control it via a terminal. To access a headless RPi, the Secure Socket Shell (SSH) tool is used to achieve an encrypted network connection that provides administrators with a secure way to access the remote device.

Connect HDMI-enabled screen, mouse and keyboard to Raspberry Pi with Raspbian installed. Boot Raspberry Pi and open a terminal on Raspberry Pi.

First thing to do is to provide static IP for Raspberry Pi: 192.168.1.15

```
$ sudo nano /etc/network/interfaces
```

Modify the file by adding the following highlighted code to it:

```
Auto lo

Iface lo inet loopback

Auto eth0

Iface eth0 inet static

      Netmask 255.255.255.0

      Address 192.168.1.15

Allow-hotplug wlan0

Iface wlan0 inet manual

      Wpa-conf /etc/wpa/supplicant/wpa_supplicant/.conf

Allow-hotplug wlan1

Iface wlan1 inet manual

      Wpa-conf /etc/wpa/supplicant/wpa_supplicant/.conf
```

Use the command below to enter configuration menu.

```
sudo raspi-config
```

Go to advanced options > SSH > Yes, to enable SSH on Raspberry PI.

Once, SSH is enabled on Raspberry PI, it is now time to access Raspberry PI on controlling machine. If the controlling machine is Windows, a software is required to access Raspberry Pi remotely. Putty is recommended. If the machine is Linux based such as Ubuntu or MAC, no additional software is required.

Now connect Raspberry PI to the control machine via LAN, Ethernet port.

On Linux-based machine, use the command blew to access Raspberry Pi terminal.

```
$ ssh pi@192.168.1.15
```

On Windows, execute PuTTy software and provide host IP address, Port: 22, SSH then open remote terminal.

Putty needs an IP address and a Port to open the SSH tunnel to your device. Secondly the device you are trying to remotely access via SSH needs to have an SSH server configured on it, fortunately by default the Raspberry Pi operating system, Raspbian, that we use configures an SSH server on Port 22 for us.

Connect your Pi to your local network via a LAN cable connected directly to your router (we will look at Wi-Fi a bit later) and once it has a good connection then the activity lights will illuminate on the Ethernet port. This means the PI's network card has communicated with the router and has correctly been allocated an IP address by the router.

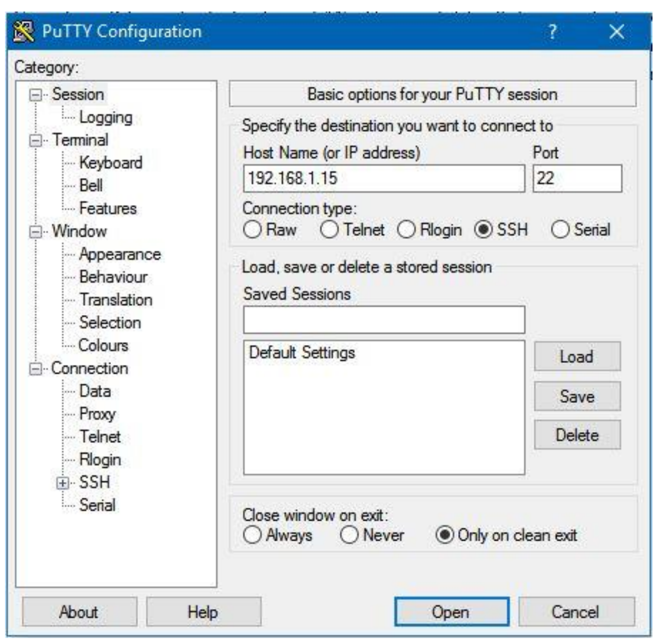
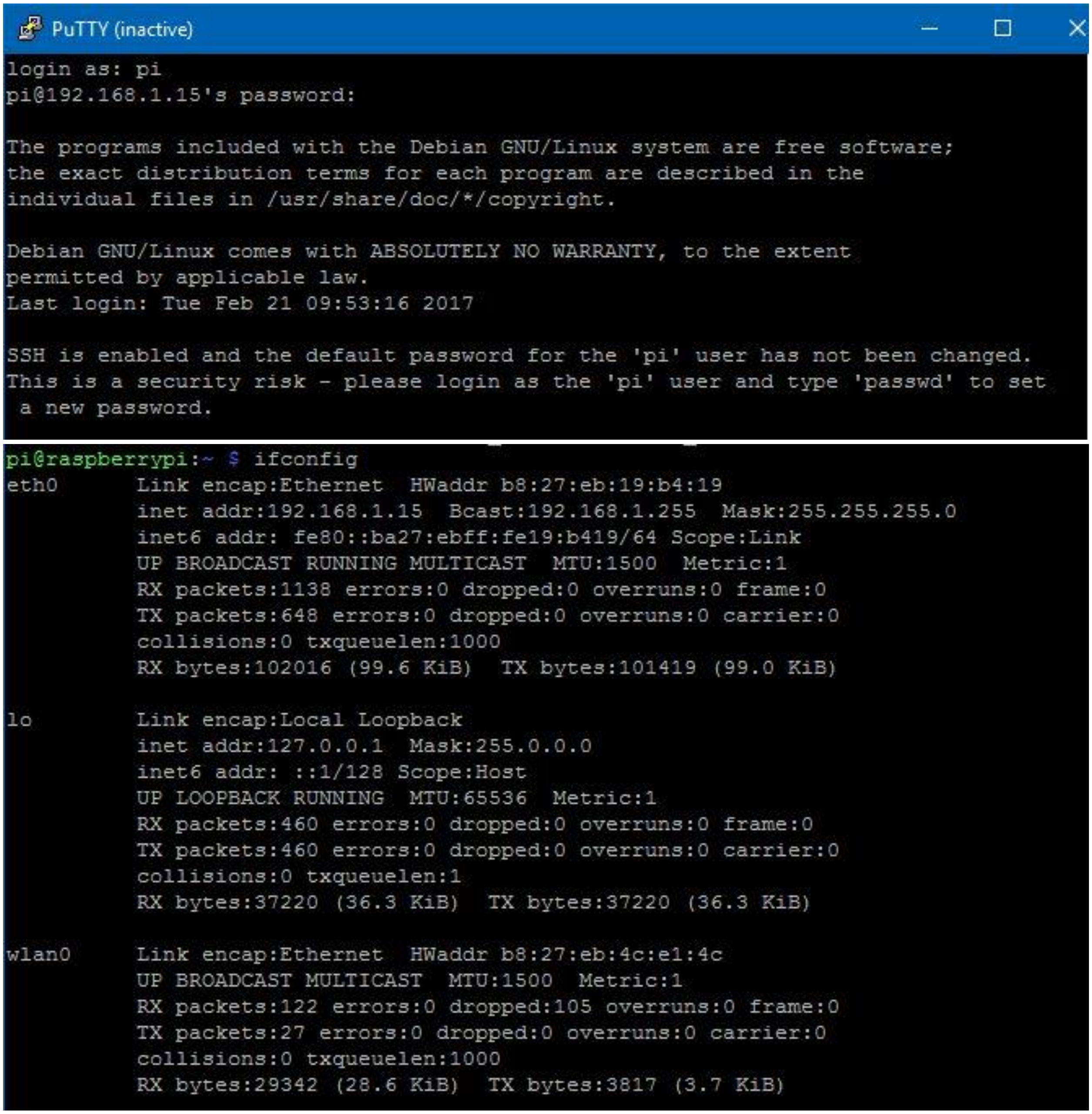


Figure 4 PuTTy Configuration



```
PuTTY (inactive)
login as: pi
pi@192.168.1.15's password:

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 Feb 21 09:53:16 2017

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set
 a new password.

pi@raspberrypi:~ $ ifconfig
eth0      Link encap:Ethernet  HWaddr b8:27:eb:19:b4:19
          inet addr:192.168.1.15  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::ba27:ebff:fe19:b419/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1138 errors:0 dropped:0 overruns:0 frame:0
          TX packets:648 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:102016 (99.6 KiB)  TX bytes:101419 (99.0 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:460 errors:0 dropped:0 overruns:0 frame:0
          TX packets:460 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:37220 (36.3 KiB)  TX bytes:37220 (36.3 KiB)

wlan0     Link encap:Ethernet  HWaddr b8:27:eb:4c:e1:4c
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:122 errors:0 dropped:105 overruns:0 frame:0
          TX packets:27 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:29342 (28.6 KiB)  TX bytes:3817 (3.7 KiB)
```

Figure 5 ifconfig output on headless RPI