Raspberry Pi Computer Architecture Essentials

With the release of the Raspberry Pi 2, a new series of the popular compact computer is available for you to build cheap, exciting projects and learn about programming.

In this book, we explore Raspberry Pi 2’s hardware through a number of projects in a variety of programming languages. We will start by exploring the various hardware components, which will provide a base for the programming projects and guide you through setting up the tools for Assembler, C/C++, and Python. We will then learn how to write multithreaded applications and Raspberry Pi 2’s multi-core processor. Moving on, you’ll get hands-on by expanding the storage options of the Raspberry Pi beyond the SD card and interacting with the graphics hardware. Furthermore, you will be introduced to the basics of sound programming while expanding upon your knowledge of Python while building a web server. Finally, you will learn to interact with third-party microcontrollers.

From writing your first Assembly language application to programming graphics, this title guides you through the essentials.

Who this book is written for
Raspberry Pi Computer Architecture Essentials is for those who are new to and those who are familiar with the Raspberry Pi. Each topic builds upon earlier ones to provide you with a guide to Raspberry Pi’s architecture. From the novice to the expert, there is something for everyone.

What you will learn from this book
- Set up your Raspberry Pi 2 and learn about its hardware
- Write basic programs in Assembly language to learn about the ARM architecture
- Use C and C++ to interact with electronic components
- Find out about the Python language and how to use it to build web applications
- Interact with third-party microcontrollers
- Experiment with graphics and audio programming
- Expand Raspberry Pi 2’s storage mechanism by using external devices
- Discover Raspberry Pi 2’s GPIO pins and how to interact with them

In this package, you will find:

- The author biography
- A preview chapter from the book, Chapter 1 'Introduction to the Raspberry Pi's Architecture and Setup'
- A synopsis of the book’s content
- More information on *Raspberry Pi Computer Architecture Essentials*
Andrew K. Dennis is the manager of professional services software development at Prometheus Research. Prometheus Research is a leading provider of integrated data management for research and is the home of HTSQL, an open source navigational query language for RDBMS.

Andrew has a diploma in computing, a BSc in software engineering, and is currently studying for a second BSc in creative computing in his spare time.

He has over 12 years of experience working in the software industry in the UK, Canada, and the USA. This experience includes e-learning courseware development, custom CMS and LMS development, SCORM consultancy, web development in a variety of languages, open source application development, blogging about the integration of web technologies with electronics for home automation, and punching lots of Cat5 cables.

His interests include web development, e-learning, 3D printing, Linux, the Raspberry Pi and Arduino, open source projects, home automation and the use of web technology in this sphere, amateur electronics, home networking, and software engineering.
Preface

Are you interested in the myriad features of your Raspberry Pi 2? From the hardware to the software, do you wish to understand how you can interact with these features?

Then this is the book for you!

The Raspberry Pi 2 is one of the latest hardware offerings in the Raspberry Pi family. With many new and improved features than previous versions, there is so much more an enthusiast can do.

This book will walk you through how you can get the most out of your device.

You will learn about how to program on the Raspberry Pi using the Assembly language, Python, and C/C++. This will include building a web server in Python and saving data to an SQLite database. Ever wondered what threads are? These are covered here too.

In addition to this, you will explore the various types of GPIO pins and how these can be used to interact with third party microcontrollers and electronic circuits.

The sound and graphics capabilities of the Raspberry Pi 2 are also experimented with through a number of projects. And to expand the Raspberry Pi’s storage option, we will also set up an external HDD via USB.

Finally, the book concludes with a project that brings together many of the technologies explained throughout the chapters.

By the time you finish reading this book, you’ll have a firm knowledge of the Raspberry Pi 2 and how you can devise your own projects that use its capabilities.
What this book covers

Chapter 1, Introduction to the Raspberry Pi's Architecture and Setup, provides an introduction to the Raspberry Pi and its hardware architecture. We will explore the various hardware components in detail, and this will provide a basis for the programming projects in future chapters. A quick guide to getting Raspbian installed and SSH enabled is also provided.

Chapter 2, Programming on Raspbian, provides an introduction to the programming languages used in this book. An explanation will be provided of which language is used and why. This chapter will also guide you through setting up the tools for Assembler, C/C++, and Python. Three introduction programs will then be walked through to give you the opportunity to test that your setup works.

Chapter 3, Low-Level Development with Assembly Language, explores programming in the Raspbian operating system using the Assembler programming language.

Chapter 4, Multithreaded Applications with C/C++, having looked at Assembler, we move up the programming hierarchy to C/C++. We learn how to write multithreaded applications and understand their usefulness. Through these applications, we learn more about the multi-core CPU of the Raspberry Pi 2.

Chapter 5, Expanding on Storage Options, offers a guide to expanding the storage options of the Raspberry Pi beyond the SD card.

Chapter 6, Low-Level Graphics Programming, shows you how to interact with the graphics hardware on the Raspberry Pi 2. Here you will learn how to draw to the screen via the frame buffer.

Chapter 7, Exploring the Raspberry Pi's GPIO Pins, shows you how to interact with electronic components using the Raspberry Pi's GPIO pins. Here we look at how Python libraries can be used to simplify the process.

Chapter 8, Exploring Sound with the Raspberry Pi 2, gives an introduction to the basics of sound programming using the Raspberry Pi's hardware. Learn about live coding via the Sonic-Pi IDE to generate your own algorithmic music.

Chapter 9, Building a Web Server, expands upon your knowledge of Python to build a web server via Flask. This chapter explores the Ethernet and Wi-Fi capabilities of the Raspberry Pi for delivering web-based applications. In this chapter, you will also learn about using SQLite to store data and display it via a web page. Topics covered also include Apache and NGINX.

Chapter 10, Integrating with Third-Party Microcontrollers, in this chapter we learn how to interact with third-party microcontrollers such as the Arduino. These devices can form the basis of robotics projects and augment the abilities of the Raspberry Pi.
Chapter 11, Final Project, will conclude the book with a final project that brings together many of the topics explored throughout previous chapters.
Introduction to the Raspberry Pi's Architecture and Setup

This chapter provides a brief introduction to the Raspberry Pi 2 Model B including both its history and its hardware architecture.

As well as discussing its system architecture, we will also look at some time saving methods for installing the Raspbian operating system.

Finally, we will wrap up with a number of tips and tricks, including how to monitor the voltage, overclock the CPU, and check the device's temperature. These quick tips should get you started exploring the operating system, installing software, and investigating the hardware.

We will mainly focus on the following topics:

- Raspberry Pi hardware architecture and components
- Installing Raspbian via a boot loader and enabling and testing SSH with RSA keys

History and background of the Raspberry Pi

The Raspberry Pi is a credit card-sized computer designed and manufactured in the UK with the initial intention of providing a cheap computing device for education. Since its release, however, it has grown far beyond the sphere of academia.
Its origins can be found in the University of Cambridge's Computer Laboratory in 2006. Computer scientist Eben Upton, along with Rob Mullins, Jack Lang and Alan Mycroft, were concerned that incoming computing undergraduate students had grown divorced from the technical aspects of computing. This was largely due to school syllabuses that placed an emphasis on using computers rather than understanding them.

Off the back of this initial concern, the Raspberry Pi foundation was formed. Over the next six years the team worked on developing a cheap and accessible device that would help schools to teach concepts such as programming, thus bringing students closer to understanding how computing works.

The Raspberry Pi's initial commercial release was in February 2012. Since then, the board has gone through a number of revisions and has been available in two models, those being Model A and Model B.

The Model A device is the cheaper and simpler of the two computers and the Model B the more powerful, including support for Ethernet connectivity.

In February 2015, the Raspberry Pi 2 Model B was released, and this is the device discussed in this book.

The new Raspberry Pi 2 is significantly more powerful than previous versions, opening us up to many new possibilities.

We will now look at the hardware of the device to get a basic understanding of what it is capable of doing. Future chapters will build upon the basics presented here.

**Raspberry Pi hardware specifications**

The new Raspberry Pi is built on the back of the Broadcom BCM2836. The BCM2836 is a system-on-a-chip processor containing four ARM cores and Broadcom's VideoCore® IV graphics stack.

In contrast to this, previous Raspberry Pi A and B models only contained a single core.

On top of this, several other components make up the device, including USB, RCA, and microSD card storage. The previous Raspberry Pi Model B only contained two USB drives and a microUSB compared to the four USB drives and microUSB of the second version.

You can read a good breakdown of how the two boards standup to each other by visiting the following website: [http://www.alphr.com/raspberry-pi-2/1000353/raspberry-pi-2-vs-raspberry-pi-b-a-raspberry-pi-comparison](http://www.alphr.com/raspberry-pi-2/1000353/raspberry-pi-2-vs-raspberry-pi-b-a-raspberry-pi-comparison).
So, compared to earlier models, version 2 is a far more capable computer, yet still remains at the same price. The added benefit of having multiple cores allows us to explore different programming techniques for utilizing them.

Next, we shall cover the core components of the Raspberry Pi board in more detail. The following is an image of the board with a description of each component:

![Raspberry Pi Board Image](Image courtesy of Wikipedia)

**Dimensions**

The Raspberry Pi 2 is a small machine measuring only 85.60 mm x 56 mm x 21 mm and weighing approximately 45g. This small size makes it suitable for embedded projects, home automation devices, arcade machines, or building small multi-device clusters.

**System on Chip**

The System on Chip (SoC) architecture that the Raspberry Pi 2 implements is the Broadcom BCM2836, which we touched upon earlier in this chapter. This contains a CPU, GPU, SDRAM, and single USB port. Each of these items is discussed in more detail under the appropriate heading.
Introduction to the Raspberry Pi's Architecture and Setup

**CPU**
A central processing unit is the brain of your Raspberry Pi. It is responsible for processing machine instructions, which are the result of your compiled programs.

The BCM2836 implements a 900 MHz quad-core ARM Cortex-A7 processor. This runs on the ARMv7 instruction set.


**GPU**
The graphics processing unit (GPU) is a specialist chip designed to handle the complex mathematics required to render graphics.

The Broadcom VideoCore Iv 250 MHz supports OpenGL ES 2.0 (24 GFLOPS) Mpeg-2 and VC-1 (with license). It also includes a 1080p30 H.264/MPEG-4 AVC decoded/encoder.

The documentation for the GPU can be found on Broadcom's website at https://www.broadcom.com/docs/support videocore/VideoCoreIV-AG100-R.pdf.

**SDRAM**
The Raspberry Pi 2 comes equipped with 1 GB of SDRAM, which is shared between the GPU and CPU.

**4 USB 2.0 ports and 1 SoC on-board USB**
The previous version of the Raspberry Pi Model B contained only a single microUSB port and a two standard USB ports. The Raspberry Pi 2 has been expanded to include an onboard 5-port USB hub.

This allows you to connect four standard USB cables to the device and a single microUSB cable. The micro USB port can be used to power your Raspberry Pi 2.

**MicroSD card port**
The microSD card is the main boot and storage mechanism of the Raspberry Pi. It is upon the microSD card that you will load your operating system and store data. Later in this book we will look at using the microSD purely for booting the Raspberry Pi, and then using a USB hard drive as a storage mechanism. In this chapter, we will delve into how we can setup the SD card with the Raspbian operating system.
Ethernet port

One of the benefits of the Raspberry Pi 2 Model B is that it contains an Ethernet port. Many Raspberry Pi packages available on Amazon and similar stores include a wireless USB dongle; however, this results in you having to use up a USB port. If you plan to place your Raspberry Pi near a router or switch or have enough Ethernet cable, then you can connect your Raspberry Pi directly with the Ethernet jack.

The Raspberry Pi 2 supports 10/100 Mbps Ethernet, and the USB adapter in the third/fourth port of USB hub can also be used for Ethernet via a USB to Ethernet adapter.

Ethernet to USB adapters can be purchased from most good electronics stores and you can read more about the technology at https://en.wikipedia.org/wiki/Ethernet_over_USB.

Audio

The Raspberry Pi 2 implements the Inter-IC Sound (I2S) serial bus for audio input and output. This allows the device to connect multiple digital audio devices together. A 3.5mm TRRS jack is available and shared with the analog video output. The HDMI component also provides digital audio output.


GPIO pins

The main method for interacting with electronic components and expansion boards is through the general purpose input/output (GPIO) pins on the Raspberry Pi.

The Raspberry Pi 2 Model B contains 40 pins in total. Future chapters will also look at how we can program these to control electronic devices.

As the acronym suggests the GPIO pins can accept both input and output commands and can be controlled by programs in a variety of languages running on the Raspberry Pi.

The input for example could be readings from a temperature sensor, and the output a command to another device to switch an LED on or off.

The Raspberry Pinout project provides an interactive guide to each GPIO pin and can be found at http://pinout.xyz/.
Video – analog TV out
As well as providing a digital method for hooking up to a TV or monitor, the Raspberry Pi 2 also comes with analog support. The method of connection is commonly known as a composite or RCA port and earlier models of the Raspberry Pi came specifically with an RCA jack. RCA cables typically come with three connectors, two for audio and one (often yellow) for video.

With the release of the Raspberry Pi 2 the composite video (RCA) and 3.5 mm audio jacks functionality has been merged into a single TRRS hardware component. Therefore, if you wish to use video through this port, you may need to get a 3.5mm Mini AV TRRS to RCA cable instead. These can be found at any good electronics stores or on Amazon.

The Raspberry Pi 2 supports both PAL and NTSC standards.

Video – HDMI port
Also included is a High-Definition Multimedia Interface (HDMI) port. This allows the Raspberry Pi 2 to be hooked up to high definition devices such as televisions and monitors. This port provides a digital alternative to the TRRS jack.

The HDMI port is ideal for streaming video and audio to your TV or monitor.

Basic hardware needed
In order to get up and running with your Raspberry Pi 2 you will need the following additional hardware components:

• MicroSD card
• Micro USB power cable
• Monitor—preferably HDMI
• HDMI cable or 3.5mm to RCA AV cable
• USB keyboard
• USB mouse
• Protective case—optional
• Wi-Fi dongle or Ethernet cable

Many websites offer starter kits that include some of these components, and an existing monitor can be reused.
Before we can power up and start using our Raspberry Pi, however, we need to install an operating system on a microSD card.

The microSD card – the main storage and boot device of the Raspberry Pi 2

A micro secure digital (microSD) card is a portable high performance storage medium used in a variety of electronic devices including cameras, phones and computers. You may already be familiar with them if you use one of the devices we have just listed.

Our Raspberry Pi 2 comes equipped with a microSD slot, which lets us use a microSD card as our main storage and boot mechanism. The card is therefore used in a similar manner to a hard drive on a traditional computer or portable device.

The previous Raspberry Pi models used a standard SD card, which was much larger. Therefore, the microSD card saves space on the circuit board and does not poke out as far, reducing the risk of it being broken.

When choosing a microSD card for your projects, there are a variety of brands on the market, and they come in a range of storage sizes running into the tens of gigabytes.

For the projects in this book we recommend using a card with a large amount of storage and you should look at choosing a card that is at least 8 GB in size. The NOOBS application, for example, requires a card of at least this size.

The official Raspberry Pi website provides a guide to microSD cards at http://www.raspberrypi.org/documentation/installation/sd-cards.md and is a good place to start.

We will now discuss the option of purchasing a microSD card preinstalled with the Raspbian operating system or New Out Of the Box Software (NOOBS) versus formatting and installing the operating system ourselves.

For those who really wish to understand the Raspberry Pi 2 in detail, installing the operating system from scratch may be a more rewarding experience.
Introduction to the Raspberry Pi’s Architecture and Setup

Preinstalled microSD card versus creating your own

A number of websites offer microSD cards preloaded with one of the operating systems that are available for the Raspberry Pi 2. An example can be found at the Allied Electronics website at http://www.alliedelec.com/raspberry-pi-8gb-sd-card-raspberry-pi-noobs-1-4/70470344/.

These are a good solution for anybody looking to get up and running quickly or who are not comfortable installing an operating system by themselves from scratch. They are also useful for those who do not have a second computer to work with in order to format a new microSD card.

The official Raspberry Pi distributions Element 14 also offer a preinstalled microSD card equipped with NOOBS, a Raspberry Pi 2 operating system boot loader. It can be found at https://www.element14.com/community/community/raspberry-pi.

The second option is to purchase a new blank microSD card and follow the instructions contained in this chapter.

It should be noted that if you do not have a home Mac or PC accessible to format a new blank microSD card, then we would recommend acquiring a preformatted card. This should come loaded with either Debian Jessie Raspbian, or the NOOBS boot loader application.

The NOOBS operating system installation manager

This book assumes that the reader will be installing the Raspbian operating system himself or herself. The simplest method for doing this is to install the NOOBS operating system installation manager onto your microSD card.

NOOBS makes the setup of your Raspberry Pi 2 easy and also provides you with a mechanism for choosing other operating systems that are compatible with the Raspberry Pi.

The official Raspberry Pi website contains an introduction and guide to NOOBS and can be found at http://www.raspberrypi.org/help/noobs-setup/.

If you already have a blank microSD card, you can download NOOBS from https://www.raspberrypi.org/downloads/noobs/.
When installing Raspbian for the first time via NOOBS you will also be presented with the raspi-config screen. This provides some handy shortcuts that allow you to do the following:

- Expand the file system
- Change the user password
- Enable boot to desktop
- Change language
- Enable the camera if you have purchased the peripheral
- Add to Rastrack Raspberry Pi Map
- Overclock your Raspberry Pi
- Explore some advanced configuration options

If you choose not to install Raspbian via NOOBS, then the following section will guide you through the process. If you are using NOOBS you can skip to the Raspbian installation wrap-up section.

**Downloading the latest version of Raspbian**

Your first task will be to download the Raspbian operating system from the official Raspberry Pi website at https://www.raspberrypi.org/downloads/raspbian/.

There are several options for downloading Raspbian including an older version of the OS based on Debian Wheezy. We recommend grabbing the latest version, and it can be obtained over either BitTorrent or via a ZIP file.

The latest version as of September 2015 is Raspbian Jessie

Once you have obtained a copy of the operating system you can move onto formatting your microSD card and installing the image.

**Setting up your microSD card and installing the Raspbian operating system**

The Raspbian installation process involves two steps:

- Formatting the microSD card to the FAT file system
- Copying the Raspbian image to the card
It is important that we quickly look at what **File Allocation Table (FAT)** is and why we need it.

FAT is a method for defining which sectors of a disk or microSD card files are stored in and which sectors on the disk are free to have new data written to them.

The standard has its origins in the 1970s for use on floppy disks and was developed by Bill Gates and Marc McDonald.

You can read more about FAT here: [https://en.wikipedia.org/wiki/File_Allocation_Table](https://en.wikipedia.org/wiki/File_Allocation_Table).

Due to its simplicity of implementation and robustness, this standard is still used on SD and microSD cards today. Therefore, it is the format you will need in order to install the Raspberry Pi’s operating system onto your microSD card.

Due to its widespread adoption you may find and microSD card you purchase is already formatted to FAT.

We recommend, however, formatting any new cards you purchase to ensure you do not encounter any problems.

The official Raspberry Pi website provides handy how-to guides for the three major operating systems on how to format and install the Raspbian image.

You can read an up-to-date overview of the installation procedure at [https://www.raspberypi.org/documentation/installation/installing-images/README.md](https://www.raspberypi.org/documentation/installation/installing-images/README.md).

The following are guides to formatting an SD card for your particular operating system:

- **Mac OS X** ([https://www.raspberypi.org/documentation/installation/installing-images/mac.md](https://www.raspberypi.org/documentation/installation/installing-images/mac.md))

Having completed installing the operating system we can now look at some final configuration before exploring some interesting features of Raspbian.
Raspbian installation wrap-up

The following section assumes you have your Raspberry Pi connected to a monitor and with a keyboard and mouse available. It also assumes you have your configuration set to boot to desktop and have powered up and logged into your device.

You should at this point connect your device to your home router. If you are planning on using Wi-Fi, read on.

Now that you have successfully installed Raspbian you should see the Linux desktop.

If you do not see the desktop, but the command line instead, you can type `startx` to start the GUI.

This desktop contains icons in the top menu linking to a number of programs installed by default with the operating system.

One important icon is the link to LXTerminal. This icon launches the Linux terminal window. Click on this icon and you should see the command line load.

The following tasks in this section can all be performed in this window.

As a handy shortcut you can also load the raspi-config application at any time by typing the following command:

```
sudo raspi-config
```

You can read about the `sudo` command here: https://www.sudo.ws/.

If you update settings in this manner you may need to reboot the Raspberry Pi for them to take affect.

Check SSH is running

In order to connect to our Raspberry Pi 2 from another device via a terminal window we need to ensure that the Secure Shell (SSH) server is up and running. SSH is the default mechanism for secure communication between our Linux machines. If you used NOOBS to install the OS you may have configured the SSH server at this point via the advanced options. We can check that the SSH service is running successfully as follows.
Open up a terminal window from the Raspbian desktop and type the following command:

```bash
ps aux | grep sshd
```

The following `sshd` process should be displayed. This tells us the services are up and running:

```
root    2017  0.0  0.3   6228  2892 ?        Ss   15:13   0:00 /usr/sbin/sshd
```

If the SSH process does not appear, it is simple to start it. Enter the following command into the terminal:

```bash
sudo /etc/init.d/ssh start
```

After you have executed this command try running the following again and check that the `sshd` process is now running:

```bash
ps aux | grep sshd
```

By default, to login to the Raspberry Pi 2 over SSH you will be prompted for a username and password. If you have not changed this the username is `pi` and the password is `raspberry`.

In addition to the username and password method, we can also use an RSA key to authenticate and gain access to the Raspberry Pi over a network. We discuss this process next.

**RSA key generation for SSH**

RSA keys are a useful method to login to the Raspberry Pi. They remove the need to enter a username and password and lock down access to a handful of hardware devices.

A RSA key consists of two parts: a public and private key. A public key can be shared with anyone and any machine with that key on and can in theory let you have access. Therefore, if you purchase more Raspberry Pis, you can place your public key on each, and negate the need to remember multiple passwords.

The private key portion should be kept secret and is located on the machine you will use to access your Raspberry Pi from. For security reasons it is best to keep the private key on a single device.

You can read more about RSA keys and the cryptographic theory behind them here: [https://en.wikipedia.org/wiki/RSA_(cryptosystem)](https://en.wikipedia.org/wiki/RSA_(cryptosystem)).
One important piece of information you will need is the IP address or hostname of your Raspberry Pi.

If Wi-Fi is disabled/not connected, you can enable it via the Menu | Preferences | WiFi Configuration link on the desktop.

You can obtain this by looking at your local home router, or by running the following commands in the terminal window of Raspbian.

For the IP address, run this command:

```
sudo ip addr show
```

Where to look depends on whether you are using a wireless or wired connection. Ethernet can be found usually at `eth0` and wireless at `wlan0`.

If you would prefer to see the hostname you can run this command:

```
sudo hostname
```

Make a note of this information, as you will need it to connect to the Raspberry Pi 2 from your second device.

If you are using a Mac or Linux to SSH into the Raspberry Pi you can generate the RSA key via the terminal window using the following steps. If you are using Windows, skip to the relevant section further on in this chapter.

**Linux and Mac RSA key generation**

Start by opening up your Mac or Linux terminal. From the command line run the following command:

```
ssh-keygen -t rsa -b 4096 -C "username"
```

You should replace the username with your own. A message similar to the following will be displayed:

**Generating public/private rsa key pair.**

Following this you will see a prompt:

**Enter file in which to save the key (/Users/username/.ssh/id_rsa):**

You can press enter here and the key will be saved to the path listed in the prompt. Note that it may look slightly different to the preceding example depending on your username and operating system.
Introduction to the Raspberry Pi's Architecture and Setup

Following this, you have the option of adding a password to the RSA private key. These prevent unauthorized users of your second computer from accessing the Raspberry Pi.

**Enter passphrase (empty for no passphrase):**

**Enter same passphrase again:**

Once you have added a passphrase, the key generation process is complete. You should now see your key's fingerprint:

Your identification has been saved in /Users/username/.ssh/id_rsa.
Your public key has been saved in /Users/username/.ssh/id_rsa.pub.
The key fingerprint is:

Now we have our key, we need to load it into ssh-agent and then copy it onto the Raspberry Pi 2.

Adding the key to the agent can be done with the following command:

```bash
ssh-add ~/.ssh/id_rsa
```

This now allows the SSH command to use your key when trying to authenticate.

If your ssh-agent isn't running, you can use the following command to start it:

```bash
eval "$(ssh-agent -s)"
```

Before we can SSH into the Raspberry Pi we need to add the public key you created to it.

Thankfully, we can do this in a single command using the Raspberry Pi's default username and password, or if you changed it, that username and password. Run the following command from your terminal. Remember to swap the IP address in the command below with the IP address or hostname you recorded earlier:

```bash
cat ~/.ssh/id_rsa.pub | ssh user@ip'cat >> .ssh/authorized_keys'
```

With the public key now located on the Raspberry Pi you can attempt to SSH in.

If the .ssh directory and authorized_keys file do not exist on your Raspberry Pi you can create them under the pi user. To create the directory type `mkdir .ssh` and to create an empty file in this directory type `touch authorized_keys`.
This can be achieved using the SSH command along with your username and IP address:

```bash
ssh -A username@ip
```

You will notice that you can now login to the computer remotely and will be presented with a terminal window prompt.

**Windows RSA key generation**

In order to connect to the Raspberry Pi from a Windows device you will need three pieces of software, Pageant, PuTTY, and PuTTYgen.

Start by downloading PuTTYgen from the following URL: http://the.earth.li/~sgtatham/putty/latest/x86/puttygen.exe.

Next download PuTTY from this URL: http://the.earth.li/~sgtatham/putty/latest/x86/putty.exe. Like PuTTYgen, this is also an executable you can run from your desktop.

Then finally grab Pageant from this URL: http://the.earth.li/~sgtatham/putty/latest/x86/pageant.exe.

We will start by generating our public and private key pair. Open up the PuTTYgen executable.

Once you have this open, run through the following steps:

1. Set the key type as **SSH-2 RSA**.
2. Click on the **Generate** button.
3. You’ll now be asked to move your mouse around to generate some random data.
4. Give your key a passphrase.
5. Next, use the **Save private key** button to save the generated private key.
6. Finally, click the **Save public key** button.

Our next task is going to be to add the key to the Raspberry Pi 2's **authorized_keys** file.

You will need the Raspberry Pi's login details and IP for the next steps.
Open up the PuTTY executable.

From the **Category** list on the left, select **Session** if this is not already open. Now add the following details:

1. In the **Host Name** field, enter the Raspberry Pi’s IP address.
2. Set the **Port** to 22.
3. Select the **SSH** radio button.
4. You can now optionally save these details for future connections.
5. Click the **Open** button.

If this is your first connection, you should now see a popup appear called **PuTTY Security Alert**. Click the **Yes** button to move on.

If you entered the connection details successfully, the PuTTY terminal window will now present you with a login prompt for the Raspberry Pi.

You will need to enter in the login name here. By default, this is set as **pi**; however, you may have changed it if you setup the device via NOOBS.

Following this you will be prompted for the password. This will be **raspberry** by default, or whichever password you set if you configured Raspbian via NOOBS.

If your login was successful, you should now see the Raspbian command line prompt.

We now want to edit the `authorized_key` file. We can use the default text editor installed by Raspbian to edit this file. If the file does not exist, you can create it and the `.ssh` directory.

It is located under the **pi** user account:

`.ssh/authorized_keys`

Copy and paste the public key you saved from PuTTYgen into this file.

Save the file and exit it.

You can now logout of the Raspberry Pi.

The final tool we need to test is the Pageant application. This is our windows SSH authentication agent.

Open up the Pageant application. It should be available in the system tray in Windows.
Next, follow these steps:

1. Right click on the **Pageant** icon.
2. Select **Add Key** from the menu.
3. A pop-up will display listing any keys you have.
4. Select the **Add Key** button.
5. From the pop-up window, select the private key you generated with PuTTYgen.
6. Click the **Open** button.
7. Next, you should be prompted to enter the passphrase for your key.
8. Fill this in and click **OK**.
9. You should now see it listed in the **Pageant Key List** window.
10. You can now close the key list.

Whenever you try and access the Raspberry Pi via PuTTY, all you have to enter is the username and host.

Open up PuTTY and connect to the Raspberry Pi again. You should now see you are logged in without a password prompt appearing. This is because you have authenticated your private key against your public key, which was added to the authorized_keys list.

On the first time logging in over SSH you may see a security alert/prompt. You can select **Yes** to this.

This completes setting up access to the Raspberry Pi remotely. We can now add a static IP address and run some diagnostic tests on our device.

We shall now move onto running some tests on the Raspberry Pi via Raspbian. These can be performed either directly on the Pi via the terminal window, or over the SSH connection you just created.

**Assign a static IP to your Raspberry Pi 2**

Assigning a static IP address to your Raspberry Pi means that when you switch it on or off a new IP will not be assigned. Instead, it will always contain the same IP address, meaning you do not have to hunt down the value assigned to it by the DHCP server each time you reboot.
To start with, check the IP address range on your router and find a free IP address. You will also need the subnet mask and the default gateway.

Next, we need to assign the free address to the Raspberry Pi. Editing the interfaces file can do this:

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

In the open file you will need to locate the line that specifies eth0 or wlan0 depending on whether you are wired or wireless. For example:

```
iface eth0 inet dhcp
```

Change the value dhcp to static:

```
iface eth0 inet static
```

Once this is done we need to add three lines directly below it specifying the IP address we wish to assign, netmask, and gateway. You should have these values from checking your router earlier.

```
address 192.168.1.132
netmask 255.255.255.0
gateway 192.168.1.1
```

Save the file using Ctrl + X and press Y to save.

We do not need to reboot the Raspberry Pi to apply these changes, but can stop and start the network interface using the following commands:

```
sudo /etc/init.d/networking stop
sudo /etc/init.d/networking start
```

If you now run the command from earlier to check the IP address, you should see it is the new value you assigned:

```
sudo ip addr show
```

In the preceding example we used the nano text editing tool. We shall now look at some other options for editing files.
Installing Screen and Vim

Two useful tools to install on your Raspberry Pi 2 are Screen—a terminal multiplexor—and Vim—a text editor.

We will be installing these via a package management tool called apt-get. A package management tool is used for installing extra software onto your operating system. It makes the process easy by keeping track of and downloading any libraries or dependencies needed by the software. It also makes upgrades and removal quite simple.

You can read more about apt-get at http://linux.die.net/man/8/apt-get.

Before installing Screen and Vim you should update the cache of the apt-get repository. This can be done by running the following command:

```bash
sudo apt-get update
```

We are now ready to install our terminal multiplexor.

We will start by installing Screen. This will allow you to keep multiple bash shells open when you login and out of your Raspberry Pi, so you can leave applications running while you are not directly connected to the device.

The Bourne Again Shell (bash) is the shell used in Raspbian by default. You can read more about it here: https://www.gnu.org/software/bash/

To install Screen you can use the apt-get package manager:

```bash
sudo apt-get install screen
```

Once installation is complete, to run Screen you simply type the following command:

`screen`

The Screen application will now load, allowing you to create multiple windows containing bash sessions. To create a new window in the screen session type the following command:

`Ctrl + a then c`

If you want to remove a window you can kill it. The command to do this is as follows:

`Ctrl + a then k`
When you have multiple windows open you will want to navigate between them. To move between each open window use the following command:

`Ctrl + a then num #where num is the screen number, for example 1 or 3`

To give the screens window a user friendly name type this command:

`Ctrl + a then Shift + a.`

This will give you a prompt where you can label the window for ease of use. To detach from a screen session type the following command:

`Ctrl + a then d`

To re-attach you can then type this command:

`screen -x`

If more than one screen session is open, type the ID in after the `-x`, for example:

`screen -x 1234`


By default, the Screen application is very plain looking. However, its look and feel can be modified through a `.screenrc` file.

To learn more about this process, check out the gnu.org site's section on customizing `screen` at https://www.gnu.org/software/screen/manual/html_node/Customization.html#Customization.

**Vim – an optional handy text editor**

In addition to the text editors installed by default with Raspbian, you may also wish to install Vim, a powerful text-editing tool. You will see this tool referenced later in this book, so you may find it easier to follow along if you install this.

To install it via our package manager run the following command:

`sudo apt-get install vim`

Vim is a complex tool but if you persist with it, you will find it rewarding. A guide can be found here: http://vimhelp.appspot.com/.

Finally, there are a number of other text editors worth exploring if you wish. You can find a list at the official Raspberry Pi website here: https://www.raspberrypi.org/documentation/linux/usage/text-editors.md.
Running tests on the OS and configuration changes

There are ranges of hardware tests we can run on the Raspberry Pi to learn more about it. These include checking voltage readings, the temperature of the device, and testing that the GPIO pins work correctly.

You can run these tests by either connecting to the Pi over SSH or loading up the LXTerminal from the desktop.

Diagnostic tests

The following diagnostic tests provide basic information on your Raspberry Pi. This just provides a taster and many more are available. A more comprehensive list of commands is available via the links at the end of this section.

You should, however, run these tests to get a basic idea of what is possible.

The system information of your Raspberry Pi can be run via the following command:

```
cat /proc/cpuinfo
```

Version information can be seen via the following command:

```
cat /proc/version
```

Memory information can be accessed using the following command:

```
cat /proc/meminfo
```

The microSD cards partitions via the following command:

```
cat /proc/partitions
```

To check the temperature of the device we can use the `vcgencmd` command:

```
vcgencmd measure_temp
```

We can also use this command with a different parameter to see the voltages. The basic command is as follows:

```
vcgencmd measure_volts id
```
Introduction to the Raspberry Pi's Architecture and Setup

In this command, `id` is one of the following items:

- `core` for the core voltage
- `sdram_c` for the sdram Core voltage
- `sdram_i` for the sdram I/O voltage
- `sdram_p` for the sdram PHY voltage

You can find more commands at elinux.org where a guide to `vcgencmd` can be found (http://elinux.org/RPI_vcgencmd_usage).

Over and underclocking the Raspberry Pi

You may want to tweak the performance of your Raspberry Pi 2. This can be achieved by overclocking the device.

Overclocking is the process of forcing the CPU or other component, for example the GPU, to operate faster than its advertised or OS configured clock frequency. In the process of overclocking it is also possible to change the operating voltage to increase the device's speed.

There is a risk associated with overclocking a device, such as instability of its operation or faster degradation of components.

The `raspi-config` menu provides a set of screens to guide you through this process.

You can access `raspi-config` from the command line by typing this command:

```
sudo raspi-config
```

Then you select an overclock option from the menu that is presented.

Alternatively, you can modify the boot configuration file directly from inside the terminal window.

You will need to edit the `/boot/config.txt` file.

Once you have this open you will see a number of commented out values, for example, `#arm_freq=800`.

In the case of our Raspberry Pi 2 the processor runs at 700MHz. We could uncomment this line and up the speed of the processor to 800MHz.

An in-depth guide to overclocking the Raspberry Pi 2 can be found at Hayden James' website: http://haydenjames.io/raspberry-pi-2-overclock/. 
Going further – testing the GPIO pins

For those interested in exploring diagnostic tools further there is the option of downloading the pigpio GPIO pin test. As you start to work more with the pins this test will come in handy for debugging problems, and allow you to check if you have accidentally damaged a pin.

You can download the pigpio library directly to your Raspberry Pi from http://abyz.co.uk/rpi/pigpio/download.html.

An overview and instructions on use can be found at http://abyz.co.uk/rpi/pigpio/index.html.

Example applications and tests can be found at http://abyz.co.uk/rpi/pigpio/examples.html.

For those looking for a digital waveform view for the Raspberry Pi's GPIO pins you can install piscope from http://abyz.co.uk/rpi/pigpio/piscope.html.

Some handy Linux commands

The following Linux commands are very useful and you will find yourself using them often. Remember you can type man command where command is the command you are interested in at any time to learn more about it and the parameters it accepts.

Most commands also contain a more concise set of documentation under help as well.

The cd command allows you to change directories:

cd /home/pi

The touch command creates an empty file:

touch test.txt

The cp command can be used to copy files:

    cp /home/pi/test.txt /home/newuser/test.txt

The mv command can be used to move a file or rename a file:

    mv /home/pi/test.txt /home/pi/test2.txt

The rm command will remove a file or directory depending on the flag used:

    rm text.txt
To find out which directory you are in you can use the `pwd` (present working directory) command:

`pwd`

If you wish to list the contents of a directory you can use the `ls` command:

`ls`

To learn more about basic Linux commands, the Debian operating system website provides a in-depth guide at [https://www.debian.org/doc/manuals/user/ch6.html](https://www.debian.org/doc/manuals/user/ch6.html).

**Troubleshooting**

You may from time to time hit problems in your setup or when running software or hardware with your Raspberry Pi 2. A good first place to check for help is the eLinux Raspberry Pi troubleshooting page at [http://elinux.org/R-Pi_Troubleshooting](http://elinux.org/R-Pi_Troubleshooting).

As well as this site, the official Raspberry Pi website has an active and friendly forum where you can post questions for help at [https://www.raspberrypi.org/forums/](https://www.raspberrypi.org/forums/).

Finally, the Raspberry Pi section of Stack Exchange is another great resource, at [http://raspberrypi.stackexchange.com/](http://raspberrypi.stackexchange.com/).

**Summary**

In this chapter we learned about the basics of the Raspberry Pi 2's hardware.

We discovered how to setup a microSD card and install the Raspbian operating system on it.

Following this, we created RSA keys to access our RPI over SSH and installed a number of useful tools via a package manager and experimented with some diagnostic tests.

With our Raspberry Pi 2 setup and ready to go we can now move onto writing applications that work on the device.
Where to buy this book

You can buy Raspberry Pi Computer Architecture Essentials from the Packt Publishing website.

Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.

Click here for ordering and shipping details.