Chapter 1
Meet the Raspberry Pi
YOUR RASPBERRY PI board is a miniature marvel, packing considerable computing power into a footprint no larger than a credit card. It’s capable of some amazing things, but there are a few things you’re going to need to know before you plunge head-first into the bramble patch.

TIP
If you’re eager to get started, skip ahead a couple of pages to find out how to connect your Raspberry Pi to a display, keyboard and mouse.

ARM vs. x86

The processor at the heart of the Raspberry Pi system is a Broadcom BCM2835 system-on-chip (SoC) multimedia processor. This means that the vast majority of the system’s components, including its central and graphics processing units along with the audio and communications hardware, are built onto that single component hidden beneath the 256 MB memory chip at the centre of the board (see Figure 1-1).

It’s not just this SoC design that makes the BCM2835 different to the processor found in your desktop or laptop, however. It also uses a different instruction set architecture (ISA), known as ARM.
Developed by Acorn Computers back in the late 1980s, the ARM architecture is a relatively uncommon sight in the desktop world. Where it excels, however, is in mobile devices: the phone in your pocket almost certainly has at least one ARM-based processing core hidden away inside. Its combination of a simple reduced instruction set (RISC) architecture and low power draw make it the perfect choice over desktop chips with high power demands and complex instruction set (CISC) architectures.

The ARM-based BCM2835 is the secret of how the Raspberry Pi is able to operate on just the 5V 1A power supply provided by the onboard micro-USB port. It’s also the reason why you won’t find any heat-sinks on the device: the chip’s low power draw directly translates into very little waste heat, even during complicated processing tasks.

It does, however, mean that the Raspberry Pi isn’t compatible with traditional PC software. The majority of software for desktops and laptops is built with the x86 instruction set architecture in mind, as found in processors from the likes of AMD, Intel and VIA. As a result, it won’t run on the ARM-based Raspberry Pi.

The BCM2835 uses a generation of ARM’s processor design known as ARM11, which in turn is designed around a version of the instruction set architecture known as ARMv6. This is worth remembering: ARMv6 is a lightweight and powerful architecture, but has a rival in the more advanced ARMv7 architecture used by the ARM Cortex family of processors. Software developed for ARMv7, like software developed for x86, is sadly not compatible with the Raspberry Pi’s BCM2835—although developers can usually convert the software to make it suitable.

That’s not to say you’re going to be restricted in your choices. As you’ll discover later in the book, there is plenty of software available for the ARMv6 instruction set, and as the Raspberry Pi’s popularity continues to grow, that will only increase. In this book, you’ll also learn how to create your own software for the Pi even if you have no experience with programming.

Windows vs. Linux

Another important difference between the Raspberry Pi and your desktop or laptop, other than the size and price, is the operating system—the software that allows you to control the computer.

The majority of desktop and laptop computers available today run one of two operating systems: Microsoft Windows or Apple OS X. Both platforms are closed source, created in a secretive environment using proprietary techniques.

These operating systems are known as closed source for the nature of their source code, the computer-language recipe that tells the system what to do. In closed-source software, this recipe is kept a closely-guarded secret. Users are able to obtain the finished software, but never to see how it’s made.
The Raspberry Pi, by contrast, is designed to run an operating system called GNU/Linux—hereafter referred to simply as Linux. Unlike Windows or OS X, Linux is open source: it’s possible to download the source code for the entire operating system and make whatever changes you desire. Nothing is hidden, and all changes are made in full view of the public. This open source development ethos has allowed Linux to be quickly altered to run on the Raspberry Pi, a process known as porting. At the time of this writing, several versions of Linux—known as distributions—have been ported to the Raspberry Pi’s BCM2835 chip, including Debian, Fedora Remix and Arch Linux.

The different distributions cater to different needs, but they all have something in common: they’re all open source. They’re also all, by and large, compatible with each other: software written on a Debian system will operate perfectly well on Arch Linux and vice versa.

Linux isn’t exclusive to the Raspberry Pi. Hundreds of different distributions are available for desktops, laptops and even mobile devices; and Google’s popular Android platform is developed on top of a Linux core. If you find that you enjoy the experience of using Linux on the Raspberry Pi, you could consider adding it to other computing devices you use as well. It will happily coexist with your current operating system, allowing you to enjoy the benefits of both while giving you a familiar environment when your Pi is unavailable.

As with the difference between ARM and x86, there’s a key point to make about the practical difference between Windows, OS X and Linux: software written for Windows or OS X won’t run on Linux. Thankfully, there are plenty of compatible alternatives for the overwhelming majority of common software products—better still, the majority are free to use and as open source as the operating system itself.

Getting Started with the Raspberry Pi
Now that you have a basic understanding of how the Pi differs from other computing devices, it’s time to get started. If you’ve just received your Pi, take it out of its protective anti-static bag and place it on a flat, non-conductive surface before continuing with this chapter.

Connecting a Display
Before you can start using your Raspberry Pi, you’re going to need to connect a display. The Pi supports three different video outputs: composite video, HDMI video and DSI video. Composite video and HDMI video are readily accessible to the end user, as described in this section, while DSI video requires some specialised hardware.

Composite Video
Composite video, available via the yellow-and-silver port at the top of the Pi known as an RCA phono connector (see Figure 1-2), is designed for connecting the Raspberry Pi to older display devices. As the name suggests, the connector creates a composite of the colours
found within an image—red, green and blue—and sends it down a single wire to the display device, typically an old cathode-ray tube (CRT) TV.

When no other display device is available, a composite video connection will get you started with the Pi. The quality, however, isn’t great. Composite video connections are significantly more prone to interference, lack clarity and run at a limited resolution, meaning that you can fit fewer icons and lines of text on the screen at once.

**HDMI Video**

A better-quality picture can be obtained using the HDMI (High Definition Multimedia Interface) connector, the only port found on the bottom of the Pi (see Figure 1-3). Unlike the analogue composite connection, the HDMI port provides a high-speed digital connection for pixel-perfect pictures on both computer monitors and high-definition TV sets. Using the HDMI port, a Pi can display images at the Full HD 1920x1080 resolution of most modern HDTV sets. At this resolution, significantly more detail is available on the screen.

If you’re hoping to use the Pi with an existing computer monitor, you may find that your display doesn’t have an HDMI input. That’s not a disaster: the digital signals present on the HDMI cable map to a common computer monitor standard called DVI (Digital Video Interconnect). By purchasing an HDMI-to-DVI cable, you’ll be able to connect the Pi’s HDMI port to a monitor with DVI-D connectivity.
If your monitor has a VGA input—a D-shaped connector with 15 pins, typically coloured silver and blue—the Raspberry Pi can’t connect to it. Adapters are available that will take in a digital DVI signal and convert it to an analogue VGA signal, but these are expensive and bulky. The best option here is simply to buy a more-modern monitor with a DVI or HDMI input.

**DSI Video**

The final video output on the Pi can be found above the SD card slot on the top of the printed circuit board—it’s a small ribbon connector protected by a layer of plastic. This is for a video standard known as Display Serial Interface (DSI), which is used in the flat-panel displays of tablets and smartphones. Displays with a DSI connector are rarely available for retail purchase, and are typically reserved for engineers looking to create a compact, self-contained system. A DSI display can be connected by inserting a ribbon cable into the matched connector on the Pi, but for beginners, the use of a composite or HDMI display is recommended.

**Connecting Audio**

If you’re using the Raspberry Pi’s HDMI port, audio is simple: when properly configured, the HDMI port carries both the video signal and a digital audio signal. This means that you can connect a single cable to your display device to enjoy both sound and pictures.
Assuming you’re connecting the Pi to a standard HDMI display, there’s very little to do at this point. For now, it’s enough to simply connect the cable.

If you’re using the Pi with a DVI-D monitor via an adapter or cable, audio will not be included. This highlights the main difference between HDMI and DVI: while HDMI can carry audio signals, DVI cannot.

For those with DVI-D monitors, or those using the composite video output, a black 3.5 mm audio jack located on the top edge of the Pi next to the yellow phono connector provides analogue audio (see Figure 1-2). This is the same connector used for headphones and microphones on consumer audio equipment, and it’s wired in exactly the same way. If you want, you can simply connect a pair of headphones to this port for quick access to audio.

While headphones can be connected directly to the Raspberry Pi, you may find the volume a little lacking. If possible, connect a pair of powered speakers instead. The amplifier inside will help boost the signal to a more audible level.

If you’re looking for something more permanent, you can either use standard PC speakers that have a 3.5 mm connector or you can buy some adapter cables. For composite video users, a 3.5 mm to RCA phono cable is useful. This provides the two white-and-red RCA phono connections that sit alongside the video connection, each carrying a channel of the stereo audio signal to the TV.

For those connecting the Pi to an amplifier or stereo system, you’ll either need a 3.5 mm to RCA phono cable or a 3.5 mm to 3.5 mm cable, depending on what spare connections you have on your system. Both cable types are readily and cheaply available at consumer electronics shops, or can be purchased even cheaper at online retailers such as Amazon.

**Connecting a Keyboard and Mouse**

Now that you’ve got your Raspberry Pi’s output devices sorted, it’s time to think about input. As a bare minimum, you’re going to need a keyboard, and for the majority of users, a mouse or trackball is a necessity too.

First, some bad news: if you’ve got a keyboard and mouse with a PS/2 connector—a round plug with a horseshoe-shaped array of pins—then you’re going to have to go out and buy a replacement. The old PS/2 connection has been superseded, and the Pi expects your peripherals to be connected over the **Universal Serial Bus (USB) port**.

Depending on whether you purchased the Model A or Model B, you’ll have either one or two USB ports available on the right side of the Pi (see Figure 1-4). If you’re using Model B, you
can connect the keyboard and mouse directly to these ports. If you’re using Model A, you’ll need to purchase a USB hub in order to connect two USB devices simultaneously.

A USB hub is a good investment for any Pi user: even if you've got a Model B, you’ll use up both your available ports just connecting your keyboard and mouse, leaving nothing free for additional devices such as an external optical drive, storage device or joystick. Make sure you buy a powered USB hub: passive models are cheaper and smaller, but lack the ability to run current-hungry devices like CD drives and external hard drives.

**TIP**

If you want to reduce the number of power sockets in use, connect the Raspberry Pi’s USB power lead to your powered USB hub. This way, the Pi can draw its power directly from the hub, rather than needing its own dedicated power socket and mains adapter. This will only work on hubs with a power supply capable of providing 700mA to the Pi’s USB port, along with whatever power is required by other peripherals.

Connecting the keyboard and mouse is as simple as plugging them in to the USB ports, either directly in the case of a Model B or via a USB hub in the case of a Model A.
A Note on Storage

As you’ve probably noticed, the Raspberry Pi doesn’t have a traditional hard drive. Instead it uses a Secure Digital (SD) memory card, a solid-state storage system typically used in digital cameras. Almost any SD card will work with the Raspberry Pi, but because it holds the entire operating system, it is necessary for the card to be at least 2 GB in capacity to store all the required files.

SD cards with the operating system preloaded are available from the official Raspberry Pi Store along with numerous other sites on the Internet. If you’ve purchased one of these, or received it in a bundle with your Pi, you can simply plug it in to the SD card slot on the bottom side of the left-hand edge. If not, you’ll need to install an operating system—known as flashing—onto the card before it’s ready to go.

Some SD cards work better than others, with some models refusing to work at all with the Raspberry Pi. For an up-to-date list of SD card models known to work with the Pi, visit the eLinux Wiki page: http://www.elinux.org/RPi_VerifiedPeripherals#SD_cards

Flashing the SD Card

To prepare a blank SD card for use with the Raspberry Pi, you’ll need to flash an operating system onto the card. While this is slightly more complicated than simply dragging and dropping files onto the card, it shouldn’t take more than a few minutes to complete.

Firstly, you’ll need to decide which Linux distribution you would like to use with your Raspberry Pi. Each has its advantages and disadvantages. Don’t worry if you change your mind later and want to try a different version of Linux: an SD card can be flashed again with a new operating system at any point.

The most up-to-date list of Linux releases compatible with the Pi is available from the Raspberry Pi website at http://www.raspberrypi.org/downloads.

The Foundation provides BitTorrent links for each distribution. These are small files that can be used with BitTorrent software to download the files from other users. Using these links is an efficient and fast way to distribute large files, and keeps the Foundation’s download servers from becoming overloaded.

To use a BitTorrent link, you’ll need to have a compatible client installed. If you don’t already have a BitTorrent client installed, download one and install it before trying to download the Raspberry Pi Linux distribution. One client for Windows, OS X and Linux is µTorrent, available from http://www.utorrent.com/downloads.
Which distribution you choose to download is up to you. Instructions in the rest of the book will be based on the Debian Raspberry Pi distribution, a good choice for beginners. Where possible, we’ll give you instructions for other distributions as well.

Linux distributions for the Raspberry Pi are provided as a single image file, compressed to make it faster to download. Once you’ve downloaded the Zip archive (a compressed file, which takes less time to download than the uncompressed files would) for your chosen distribution, you’ll need to decompress it somewhere on your system. In most operating systems, you can simply double-click the file to open it, and then choose Extract or Unzip to retrieve the contents.

After you’ve decompressed the archive, you’ll end up with two separate files. The file ending in sha1 is a hash, which can be used to verify that the download hasn’t been corrupted in transit. The file ending in img contains an exact copy of an SD card set up by the distribution’s creators in a way that the Raspberry Pi understands. This is the file that needs to be flashed to the SD card.

**WARNING**
During the following, you’ll be using a software utility called `dd`. Used incorrectly `dd` will happily write the image to your main hard drive, erasing your operating system and all your stored data. Make sure you read the instructions in each section thoroughly and note the device address of your SD card carefully. Read twice, write once!

**Flashing from Linux**

If your current PC is running a variant of Linux already, you can use the `dd` command to write the contents of the image file out to the SD card. This is a text-interface program operated from the command prompt, known as a *terminal* in Linux parlance. Follow these steps to flash the SD card:

1. Open a terminal from your distribution’s applications menu.
2. Plug your blank SD card into a card reader connected to the PC.
3. Type `sudo fdisk -l` to see a list of disks. Find the SD card by its size, and note the device address (`/dev/sdX`, where `X` is a letter identifying the storage device. Some systems with integrated SD card readers may use the alternative format `/dev/mmcblkX`—if this is the case, remember to change the target in the following instructions accordingly).
4. Use `cd` to change to the directory with the `.img` file you extracted from the Zip archive.
5. Type `sudo dd if=imagefilename.img of=/dev/sdX bs=2M` to write the file `imagefilename.img` to the SD card connected to the device address from step 3. Replace `imagefilename.img` with the actual name of the file extracted from the Zip archive. This step takes a while, so be patient! During flashing, nothing will be shown on the screen until the process is fully complete (see Figure 1-5).
Flashing from OS X

If your current PC is a Mac running Apple OS X, you’ll be pleased to hear that things are as simple as with Linux. Thanks to a similar ancestry, OS X and Linux both contain the `dd` utility, which you can use to flash the system image to your blank SD card as follows:

1. Select Utilities from the Application menu, and then click on the Terminal application.
2. Plug your blank SD card into a card reader connected to the Mac.
3. Type `diskutil list` to see a list of disks. Find the SD card by its size, and note the device address (/dev/diskX where X is a letter identifying the storage device).
4. If the SD card has been automatically mounted and is displayed on the desktop, type `diskutil unmountdisk /dev/diskX` to unmount it before proceeding.
5. Use `cd` to change to the directory with the `.img` file you extracted from the Zip archive.
6. Type `dd if=imagefilename.img of=/dev/diskX bs=2M` to write the file `imagefilename.img` to the SD card connected to the device address from step 3. Replace `imagefilename.img` with the actual name of the file extracted from the Zip archive. This step takes a while, so be patient!

Flashing from Windows

If your current PC is running Windows, things are slightly trickier than with Linux or OS X. Windows does not have a utility like `dd`, so some third-party software is required to get the
image file flashed onto the SD card. Although it’s possible to install a Windows-compatible version of dd, there is an easier way: the Image Writer for Windows. Designed specifically for creating USB or SD card images of Linux distributions, it features a simple graphical user interface that makes the creation of a Raspberry Pi SD card straightforward.

The latest version of Image Writer for Windows can be found at the official website: https://launchpad.net/win32-image-writer. Follow these steps to download, install and use the Image Writer for Windows software to prepare the SD card for the Pi:

1. Download the binary (not source) Image Writer for Windows Zip file, and extract it to a folder on your computer.
2. Plug your blank SD card into a card reader connected to the PC.
3. Double-click the Win32DiskImager.exe file to open the program, and click the blue folder icon to open a file browse dialogue box.
4. Browse to the imagefilename.img file you extracted from the distribution archive, replacing imagefilename.img with the actual name of the file extracted from the Zip archive, and then click the Open button.
5. Select the drive letter corresponding to the SD card from the Device drop-down dialogue box. If you’re unsure which drive letter to choose, open My Computer or Windows Explorer to check.
6. Click the Write button to flash the image file to the SD card. This process takes a while, so be patient!

WARNING
No matter which operating system you’re writing from, it’s important to ensure you leave the SD card connected until the image has been completely written. If you don’t, you may find that Pi doesn’t boot when the SD card is connected. If this happens, start the process again.

When the image has been flashed onto the SD card, remove it from the computer and insert it into the Raspberry Pi’s SD card slot, located underneath the circuit board. The SD card should be inserted with the label facing away from the board and pushed fully home to ensure a good connection.

Connecting External Storage
While the Raspberry Pi uses an SD card for its main storage device—known as a boot device—you may find that you run into space limitations quite quickly. Although large SD cards holding 32 GB, 64 GB or more are available, they are often prohibitively expensive.

Thankfully, there are devices that provide an additional hard drive to any computer when connected via a USB cable. Known as USB Mass Storage (UMS) devices, these can be physical hard drives, solid-state drives (SSDs) or even portable pocket-sized flash drives (see Figure 1-6).
The majority of USB Mass Storage devices can be read by the Pi, whether or not they have existing content. In order for the Pi to be able to access these devices, their drives must be mounted—a process you will learn in Chapter 2, “Linux System Administration”. For now, it’s enough to connect the drives to the Pi in readiness.

**Connecting the Network**

While the majority of these setup instructions are equally applicable to both the Raspberry Pi Model A and the Model B, networking is a special exception. To keep the component count—and therefore the cost—as low as possible, the Model A doesn’t feature any onboard networking. Thankfully, that doesn’t mean you can’t network the Model A; only that you’ll need some additional equipment to do so.

**Networking the Model A**

To give the Model A the same networking capabilities as its more expensive Model B counterpart, you’ll need a USB-connected Ethernet adapter. This connects to a free USB port on the Raspberry Pi or a connected hub and provides a wired Ethernet connection with an RJ45 connector, the same as is available on the Model B.

A 10/100 USB Ethernet adapter—with the numbers referring to its two-speed mode, 10 Mb/s and 100 Mb/s—can be purchased from online retailers for very little money. When buying an Ethernet adapter, be sure to check that Linux is listed as a supported operating system. A few models only work with Microsoft Windows, and are incompatible with the Raspberry Pi.

Don’t be tempted to go for a gigabit-class adapter, which will be referred to as a 10/100/1000 USB Ethernet adapter. Standard USB ports, as used on the Raspberry Pi, can’t cope with the speed of a gigabit Ethernet connection, and you’ll see no benefit to the more expensive adapter.
**Wired Networking**

To get your Raspberry Pi on the network, you’ll need to connect an *RJ45 Ethernet patch cable* between the Pi and a switch, router or hub. If you don’t have a router or hub, you can get your desktop or laptop talking to the Pi by connecting the two directly together with a patch cable.

Usually, connecting two network clients together in this way requires a special cable, known as a *crossover cable*. In a crossover cable, the receive and transmit pairs are swapped so that the two devices are prevented from talking over each other—a task usually handled by a network switch or hub.

The Raspberry Pi is cleverer than that, however. The RJ45 port on the side of the Pi (see Figure 1-7) includes a feature known as *auto-MDI*, which allows it to reconfigure itself automatically. As a result, you can use any RJ45 cable—crossover or not—to connect the Pi to the network, and it will adjust its configuration accordingly.

*Figure 1-7*: The Raspberry Pi Model B's Ethernet port

If you do connect the Pi directly to a PC or laptop, you won’t be able to connect out onto the Internet by default. To do so, you’ll need to configure your PC to *bridge* the wired Ethernet...
port and another (typically wireless) connection. Doing so is outside the scope of this book, but if you are completely unable to connect the Pi to the Internet in any other way, you can try searching your operating system’s help file for “bridge network” to find more guidance.

With a cable connected, the Pi will automatically receive the details it needs to access the Internet when it loads its operating system through the Dynamic Host Configuration Protocol (DHCP). This assigns the Pi an Internet Protocol (IP) address on your network, and tells it the gateway it needs to use to access the Internet (typically the IP address of your router or modem).

For some networks, there is no DHCP server to provide the Pi with an IP address. When connected to such a network, the Pi will need manual configuration. You’ll learn more about this in Chapter 4, “Network Configuration”.

**Wireless Networking**

Current Raspberry Pi models don’t feature any form of wireless network capability onboard, but—as with adding wired Ethernet to the Model A—it’s possible to add Wi-Fi support to any Pi using a USB wireless adapter (see Figure 1-8).

![Figure 1-8: Two USB wireless adapters, suitable for use with the Raspberry Pi](image)

Using such a device, the Pi can connect to a wide range of wireless networks, including those running on the latest 802.11n high-speed standard. Before purchasing a USB wireless adapter, check the following:
Ensure that Linux is listed as a supported operating system. Some wireless adapters are provided with drivers for Windows and OS X only, making them incompatible with the Raspberry Pi. A list of Wi-Fi adapters known to work with the Raspberry Pi can be found on the following website: http://elinux.org/RPi_VerifiedPeripherals#USB_WiFi_Adapters

Ensure that your Wi-Fi network type is supported by the USB wireless adapter. The network type will be listed in the specifications as a number followed by a letter. If your network type is 802.11a, for example, an 802.11g wireless adapter won’t work.

Check the frequencies supported by the card. Some wireless network standards, like 802.11a, support more than one frequency. If a USB wireless adapter is designed to work on a 2.4GHz network, it won’t connect to a 5GHz network.

Check the encryption type used by your wireless network. Most modern USB wireless adapters support all forms of encryption, but if you’re buying a second-hand or older model, you may find it won’t connect to your network. Common encryption types include the outdated WEP and more modern WPA and WPA2.

Configuration of the wireless connection is done within Linux, so for now it’s enough to simply connect the adapter to the Pi (ideally through a powered USB hub.) You’ll learn how to configure the connection in Chapter 4, “Network Configuration”.

Connecting Power

The Raspberry Pi is powered by the small micro-USB connector found on the lower left side of the circuit board. This connector is the same as found on the majority of smartphones and some tablet devices.

Many chargers designed for smartphones will work with the Raspberry Pi, but not all. The Pi is more power-hungry than most micro-USB devices, and requires up to 700mA in order to operate. Some chargers can only supply up to 500mA, causing intermittent problems in the Pi’s operation (see Chapter 3, “Troubleshooting”).

Connecting the Pi to the USB port on a desktop or laptop computer is possible, but not recommended. As with smaller chargers, the USB ports on a computer can’t provide the power required for the Pi to work properly.

Only connect the micro-USB power supply when you are ready to start using the Pi. With no power button on the device, it will start working the instant power is connected and can only be turned off again by physically removing the power cable.