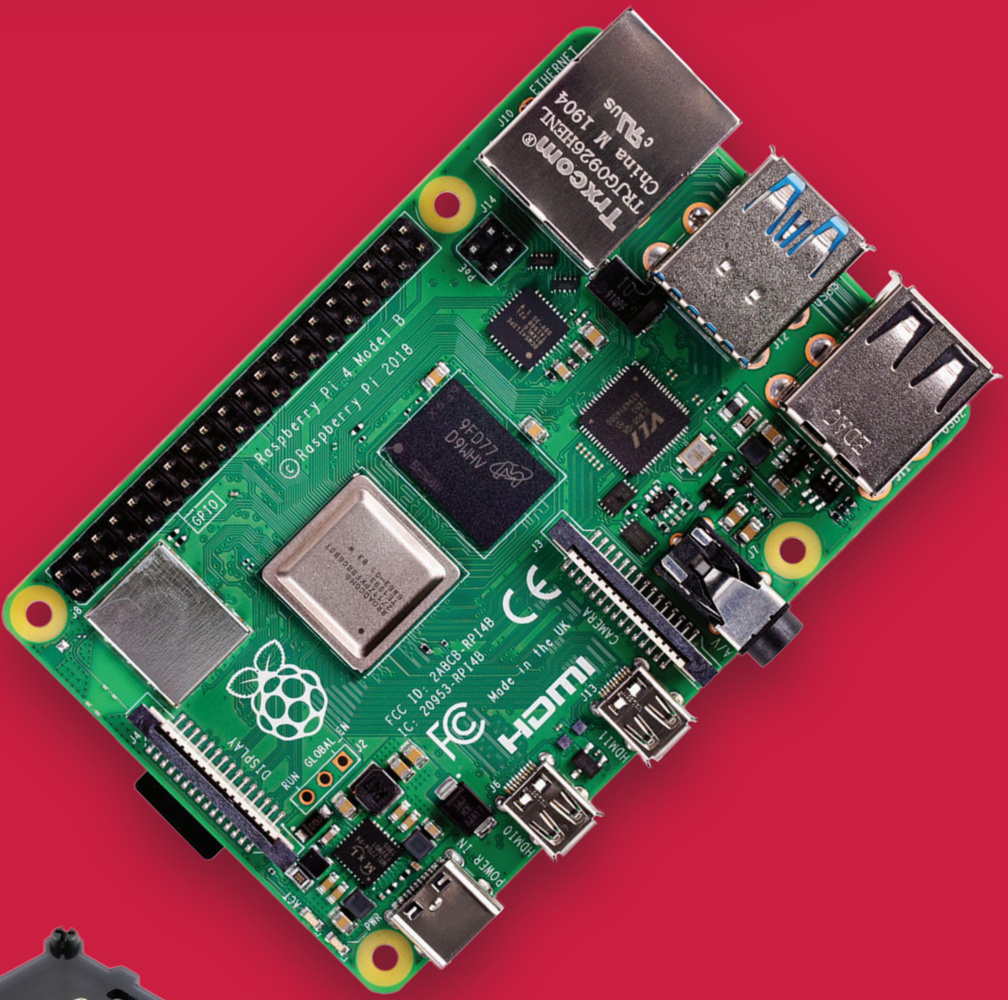
ÉLECTRONIQUES • ELECTONICS CORP.

Raspberry Pi 4 Starter Kit

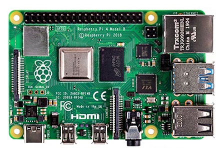


INTRODUCTION to The Raspberry Pi

The family of Raspberry Pi boards was created by the UK's *Raspberry Pi Foundation* back in 2012. A Raspberry Pi runs a *microprocessor* and not a *microcontroller*. This simply means that it is more closely related to your computer than it is to your Arduino. Read on if you are curious about their differences. The hardware of the Raspberry Pi can be connected to an Ethernet or Wi-Fi network and can interface with multitudes of different sensors and peripheral modules using its many connection ports and protocols (USB, HDMI, RCA, UART, I2C and GPIO). The brain of the Raspberry Pi is controlled by its operating system (OS) which can be changed and loaded for different applications in the same way you change outfits for different occasions.

Alternatively, a user can write and run their own scripts in Bash or Python to develop their project even more.

This kit includes:

Raspberry Pi 4 - Model B ABRA Part Numbers: PI4B-1GB, or PI4B-2GB, or PI4B-4GB	**See specifications on page 4.**	
Raspberry Pi Case ABRA Part Number: WAVE-16931	Acrylic case for protection with access to all the ports and a cooling fan.	
Power Adapter ABRA Part Number: PI4B-PS-SW	AC to DC 5V 3A USB Type-C wall adapter with an inline ON/OFF switch	
NOOBS microSD ABRA Part Number: PI4B-NOOBS	Operating System stored on 32GB class-10 microSD card with a MicroSD to SD card adapter	

What you'll need to have:

USB Keyboard	Interface with command line	
USB Mouse	Interface with desktop	
HDMI Cable	Connect to display	
Monitor	HDMI Display	

Raspberry Pi vs. Arduino:

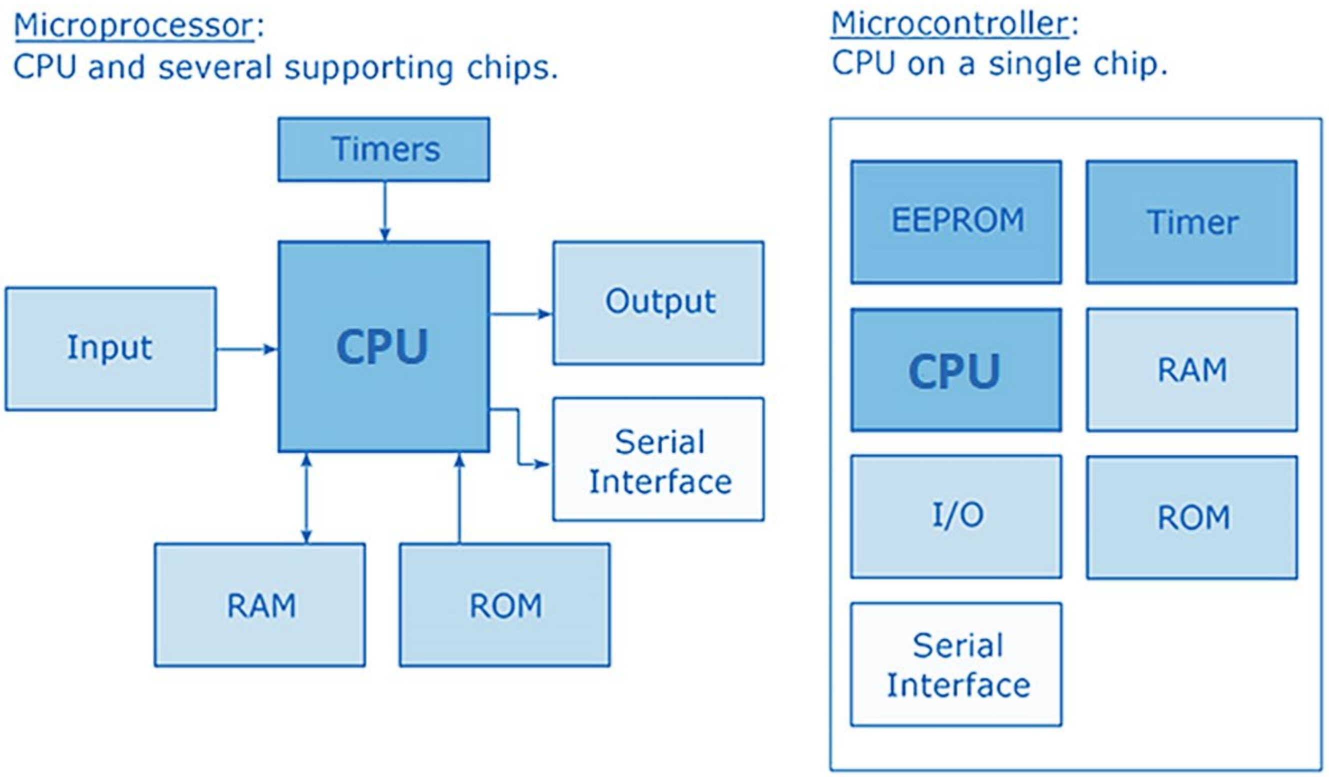
Many users ask themselves the question *"What is the difference between the Raspberry Pi and the Arduino*?" and *"Which is best for my project?"* This comparison will hopefully clarify some of these uncertainties.

Arduino is a microcontroller board. A microcontroller is a self-contained computer that can execute one program at a time, over and over again. It is easy to use if you know C or C++ programming language.

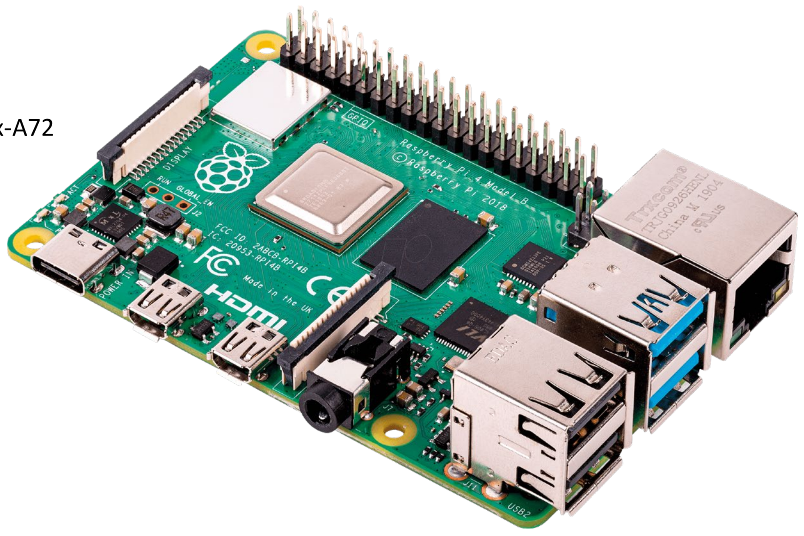
Raspberry Pi is a microprocessor which is a general-purpose computer with many devices placed around it to serve the purpose, it usually runs on a Linux-based operating system, and has the ability to execute multiple programs at the same time. It is more complicated to use.

Both devices can interact with peripherals and sensors and connect to networks via Bluetooth and the Internet. However, Raspberry Pi is a better multi-tasker with more memory. The Arduino is great for repetitive tasks.

A microprocessor is a collection of systems that contain a CPU with several external memories and peripherals. Without the supporting chips a CPU will not be able to achieve anything. On the other hand, a microcontroller self-contains all the components required (CPU, RAM, ROM, Oscillator, A/D converter, timer, counter, etc.) to run standalone without the need of external devices.



SPECIFICATIONS

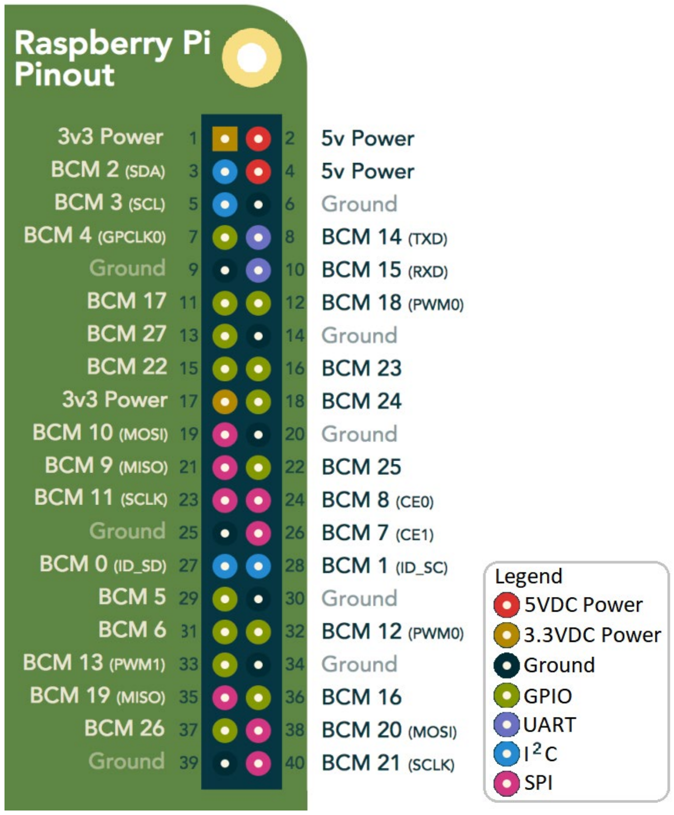


Raspberry Pi 4 – Model B

- CPU – Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- RAM – 1GB, 2GB or 4GB LPDDR4-2400 SDRAM (depending on model)
- WiFI – 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Ethernet – Gigabit
- Low-Level Peripherals:
    - 27x GPIO
    - UART
    - I2C
    - SPI w/ two chip selects
    - +3.3VDC
    - +5VDC
    - Ground
- USB – 2 USB 3.0 ports; 2 USB 2.0 ports
- GPIO header – Raspberry Pi standard 40-pin
- HDMI – 2 × micro-HDMI ports (up to 4kp60 supported)
- Display port – 2-lane MIPI DSI
- Camera port – 2-lane MIPI CSI
- Audio – 4-pole stereo audio and composite video port
- Storage – Micro-SD card slot for loading operating system and data storage
- Power Requirements –
    - 5VDC via USB-C connector (minimum 3A*)
    - 5VDC via GPIO header (minimum 3A*)
    - *A good quality 2.5A power supply can be used if downstream USB peripherals consume less than 500mA in total.
- Power over Ethernet (PoE) enabled (requires separate PoE HAT)
- Operating temperature – 0 – 50 °C ambient
- Misc. – H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode), OpenGL ES 3.0 graphics
- OS – Raspbian, Windows IoT Core, RaspBSD, Lakka, RetroPie, Ubuntu Core, Linutop, LibreELEC, OSMC, Pidora, Arch Linux, RISC OS and More!
- Dimensions – 88 x 58 x 19.5mm / 3.5 x 2.3 x 0.76”

GPIO

The Raspberry Pi are equipped with many General Purpose Input Output (GPIO) pins that also support popular communication protocols. These pins can be used to connect the Raspberry Pi to sensors, buttons and other pieces of hardware and modules by configuring them to either read inputs or write outputs. They also have a few 5V and 3.3V power and ground terminals.


Raspberry Pi Pinout
3v3 Power 1 | 2 5v Power
BCM 2 (SDA) 3 | 4 5v Power
BCM 3 (SCL) 5 | 6 Ground
BCM 4 (GPCLK0) 7 | 8 BCM 14 (TXD)
Ground 9 | 10 BCM 15 (RXD)
BCM 17 11 | 12 BCM 18 (PWM0)
BCM 27 13 | 14 Ground
BCM 22 15 | 16 BCM 23
3v3 Power 17 | 18 BCM 24
BCM 10 (MOSI) 19 | 20 Ground
BCM 9 (MISO) 21 | 22 BCM 25
BCM 11 (SCLK) 23 | 24 BCM 8 (CE0)
Ground 25 | 26 BCM 7 (CE1)
BCM 0 (ID_SD) 27 | 28 BCM 1 (ID_SC)
BCM 5 29 | 30 Ground
BCM 6 31 | 32 BCM 12 (PWM0)
BCM 13 (PWM1) 33 | 34 Ground
BCM 19 (MISO) 35 | 36 BCM 16
BCM 26 37 | 38 BCM 20 (MOSI)
Ground 39 | 40 BCM 21 (SCLK)
Legend
5VDC Power
3.3VDC Power
Ground
GPIO
UART
I²C
SPI

There are two naming schemes you should familiarize yourself with: WiringPi and Broadcom numbering. The physical pin numbering is displayed in numbers from 1 to 40. WiringPi, the GPIO interfacing library you will most likely be using, has its own hardware-independent numbering system internally. You can refer to https://pinout.xyz/ for more information regarding these pins. Remember to verify which pin you are actually writing to or from when programming your Pi.

Listed below is the description of available functions and features on the GPIO header:

DC Power and Ground Pins

The power and ground pins are used to power external circuitry. All Raspberry Pies with the standard 40-pin GPIO headers will have two 5V pins and two 3.3V pins, always in the same place. Along with the 5V and 3.3V pins, 8 ground pins are available. Power and ground pins are what let your Raspberry Pi power external components and modules in your project. However, remember that the proper HAT or external circuitry should always be installed before attempting to power anything through these pins. Powering something with too much current draw or significant voltage spikes, like a motor without the appropriate motor controller/driver, will damage the pins and could render them unusable.

I2C

I2C, or the Inter-Integrated Circuit protocol, allows your Raspberry Pi (master) to serially control multiple sensors and components, known as slaves. The communication is done through two wires, the SDA (data pin) and SCL (clock speed pin). Each slave device is created with a unique address to allow for fast communication with many devices. The ID_EEPROM pins such as GPIO pin 27 and 28 are also I2C, but are ONLY used for communicating with HATs, not slave components.

SPI

SPI, or Serial Peripheral Interface, is also used to serially control components with a master-slave relationship, though it is not as compact. It requires clock (SCLK), Master-Out Slave-In (MOSI), and Master-In Slave-out (MISO) pins to function. The pins do what their names imply, with SCLK regulating data speed, MOSI used to send orders from the Pi to external SPI peripherals, and MISO doing the opposite.

UART
If you've tinkered with Arduinos or any kind of microcontrollers before, you may have heard of UART or Serial before. Universal Asynchronous Receiver/Transmitter is used to hook up microcontrollers to the computers that program them and also for communication between other devices with the receive (Rx) and transmit (Tx) pins. These pins can be used to control your Pi through another computer if the serial console is enabled in "raspi-config" or to control an Arduino directly if you are not able to use a USB cable for your project.

PWM
Along with these functions, all pins are capable of software pulse-width modulation (PWM) while GPIO pins 12, 13, 18 and 19 are capable of hardware pulse-width modulation. This is useful if you want to drive a low power servo motor or an H-bridge.

OPERATING SYSTEMS
Think of a Raspberry Pi as a real PC running Linux. The computer needs an operating system and an interface (Desktop). This section will showcase several of the most popular Raspberry Pi Operating Systems (OS) as mentioned in the specifications section on page 4.

Raspbian

Raspbian is a Debian-based engineered especially for the Raspberry Pi and it is the perfect general-purpose OS for RPi users. It employs the *Openbox* stacking window manager and the RPi Improved *Xwindows Environment Lightweight* coupled with a number of pre-installed software which includes *Minecraft Pi*, *Java*, *Mathematica*, and *Chromium*.
Raspbian is the Raspberry foundation's official supported OS and is capable of accomplishing any task you throw at it. Stick with this OS if you want to have a hassle-free experience.
https://www.raspberrypi.org/downloads/raspbian/

Windows IoT Core

Windows IoT Core is a Windows OS built specially for the Raspberry Pi as a development platform for programmers and coders. Its aim is for programmers to use it to build prototypes of *Internet of Things (IoT)* devices using the Raspberry Pi and Windows 10.
It has an emphasis on security, connectivity, creation, and cloud integration. Unlike other titles on this list, you can't use it without running *Windows 10* on your PC as you need *Visual Studio* on a *Windows 10* setup to work with it.
https://developer.microsoft.com/en-us/windows/iot

RaspBSD

RaspBSD is a free and open-source image of *FreeBSD 11* that has been preconfigured in 2 images for Raspberry Pi computers.
If you didn't know, *FreeBSD* isn't Linux, but it works in pretty much the same way as it is a descendant of the research by the *Berkeley Software Distribution* and it is among the world's most broadly used Operating Systems today with its code existing in game consoles e.g. PlayStation 4*,* macOS, etc.
http://www.raspbsd.org/

Lakka

Lakka is a free, lightweight, and open-source distro with which you can turn even the smallest PC into a full-blown game console without the need for a keyboard or mouse.
It features a beautiful User Interface and so many customization options you might get overwhelmed. Its PS4-like UX brings style to the Raspberry Pi so pick it if you're a gamer.
https://www.lakka.tv/get/linux/rpi

RetroPie

RetroPie is an open-source Debian-based software library with which you can emulate retro games on your Raspberry Pi, PC, or ODroid C1/C2 and it currently stands as the most popular option for that task.
RetroPie used the *EmulationStation* frontend and *SBC* to offer users a pleasant retro gaming experience so you can't go wrong with it.
https://retropie.org.uk/download/

Ubuntu Core

Ubuntu Core is the version of Ubuntu designed for *Internet of Things* applications. Ubuntu is the most popular Linux-based Operating System in the world with over 20+ derivatives and given that it has an active and welcoming forum, it will be easy to get up and running with *Ubuntu Snappy Core* on your Raspberry Pi.
https://ubuntu.com/download/iot

EXPLORING YOUR RASPBERRY PI

The *ABRA Raspberry Pi 4 Model B kit* comes with a preloaded ready-to-use NOOBS/Raspbian operating system. To get started right away, jump ahead to *"STARTING RASPBERRY PI".*
The following sections will showcase how you can explore the Raspberry Pi for its full capability.

Loading The Operating System Image

If you chose to run a **different OS** than NOOBS/Raspbian , it must be loaded and booted from the SD card.

NOTE: You can choose to overwrite the current NOOBS on your SD card or simply use a new one.

Download the image (.img) file for your operating system and unzip it somewhere you'll remember.

NOTE: Simply saving the image file on the SD card does not configure the OS!

This process requires 2 steps:
1) Re-formatting the SD card (to wipe its contents).
2) Re-imaging new software on the card.

We recommend using these free software tools on your regular computer to complete the task.

SD Formatter: https://www.sdcard.org/downloads/

Etcher Disk Imager: https://etcher.io/

NOTE: Never remove the SD card from your computer or Raspberry Pi without "safely removing" it. This will most certainly corrupt the data and this whole process will need to be repeated.

Command Line

Every OS offers an easy to use user interface (UI). Behind the scenes of each OS is the command line which allows you to unleash a much more powerful set of features, all without a mouse. The command line can become quite verbose and seem overly complicated with uncertainty of which commands to use and how to use them. This section will highlight a few basic commands that are essential to have in your repertoire.

***NOTE:* Each command must be followed by "Enter" return carriage.**

Basic Commands:
Refer to these excellent guides:
https://www.raspberrypi.org/documentation/linux/usage/commands.md/
https://www.circuitbasics.com/useful-raspberry-pi-commands/

Sudo Commands:
You're logging in as the normal ***pi*** user. You can run commands as the ***root*** user with the ***sudo*** command before the program you want to run.

Some commands make permanent changes to the state of your system and require ***root*** privileges to run. The command ***sudo*** temporarily gives your account the ability to run these commands When you append ***sudo*** to the start of a command and press enter you will be asked for your password, then the command you want to run will be run using root privileges.

****NOTE:* Some commands that require *sudo* to run can irreparably damage your system!**

Nano Commands:
Nano is the Linux text editor and is often useful when you need to modify an on-board file.
e.g.

```
nano example.txt
```

Once in ***nano***, there is a menu of further commands for navigating and making changes. You have to navigate through the text with the cursor keys. To exit, press **[CTRL + x]** , you will be asked to **SAVE (Y/N)**.

****NOTE:* To find out more about a particular command, you can run the *man* command followed by the command you want to know more about** (e.g. *man ls, man rm, man apt-get, etc.*)**. The man page (or manual page) for that command will be displayed, which includes information about the flags for that program and what effect they have. Some man pages will give example of their correct usage.**

Linux Tutorial:

After all, the Raspberry Pi runs a ***Linux*** kernel, so these commands are always useful to know.
http://linuxcommand.org/lc3_learning_the_shell.php

Internet Connectivity

The simplest way to get internet connectivity is by connecting an **Ethernet (RJ45)** cable from your network router to the Raspberry Pi. Your device should become visible on your network.

Raspberry Pi 4 Model B conveniently allows for **Wi-Fi** connectivity. This of course varies with each OS and is sometime accessible on the Desktop. Here is a quick method to connect via the command line.
Open the ***wpa-supplicant*** configuration file in nano:

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

Go to the bottom of the file and add the following:

```
network={
	ssid="yourSSID"
	psk="yourPassword"
}
```

You can verify whether it has successfully connected using:

```
ifconfig wlan0
```

If the *inet addr* field has an IP address beside it, the Raspberry Pi has connected to the network. If not, check that your password and SSID are correct.

IP Address:
It is also important to know your device's IP address. Here is a quick method to retrieve it.

```
sudo ifconfig
```

Bluetooth Connectivity

Pairing a Bluetooth device to your project can certainly add a nice touch and definitely helps free up some pins. This section will showcase how you can do so.

Graphical Method: (This approach will of course vary with each OS)
Open Terminal and type:

```
sudo apt-get install bluetooth bluez blueman
```

(Internet connection required)

Once the packages are downloaded and installed, type:

```
sudo reboot
```

To access this menu, on the Raspberry Pi desktop click Menu, located in the upper left corner of the screen, scroll down to *Preferences* with your cursor and click *Bluetooth Manager*. You can now pair any nearby devices, or you can make your Raspberry Pi discoverable so you can pair your phone to it from your phone's Bluetooth settings.

Command Line:
Open a new Terminal window.

Type `sudo bluetooth ctl` and insert the administrator password.

Type `agent on`

Type `default-agent`

Type `scan on`. The unique MAC addresses of all the Bluetooth devices around the Raspberry Pi will appear and look something like an alphanumeric XX:XX:XX:XX:XX:XX.

If you make the device you want to pair discoverable, the device nickname may appear to the right of the address.

To pair the device, type `pair [device MAC address]`

The command will look something like:

```
pair CC:94:4B:25:AA:E0
```

Compiling/Updating Software

There are certain particularities with Linux based systems like Raspberry Pi that need to be understood when compiling software. There is a dedicated part of the OS called a ***package***. It links to a repository of software so it can download all the files. (Internet connection is required)

Updating Packages of OS Software

Simply enter `sudo apt-get update` to get and install the latest version of the available packages from the repository.

Compiling Software

Some packages are not available from a repository. In this case, retrieve the software from the supplier's website and save/unzip it in a folder (usually files are **.tar.gz** or **.tgz**). Unzip by typing:

```
tar zxvt <filename>
```

This creates a new directory which you must ***cd*** into. You can hopefully find an **INSTALL** file and follow those instructions (if your device dependencies are relevant).

GPIO Control

You can program the pins to interact in amazing ways with the real world. Inputs don't have to come from a physical switch; it could be input from a sensor or a signal from another computer or device from a network.

****NOTE:* Avoid connecting motors without appropriate driver circuits.**

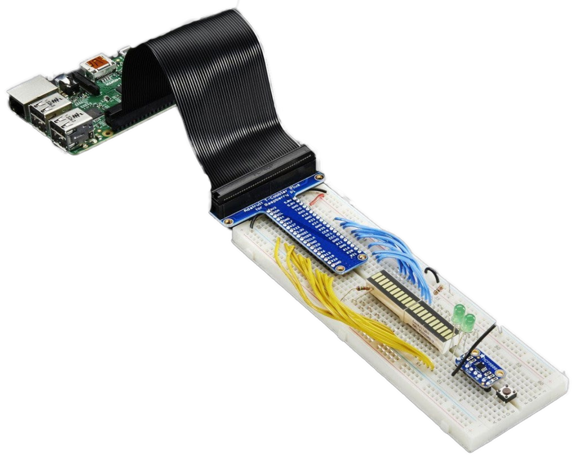
The pins can be used as Inputs or Outputs. The circuit is connected appropriately, and the next step is to write a Python program or Bash script to tell the pin to go HIGH (3.3V) or LOW (0V). You can also directly control a pin from the command line.

****NOTE:* Sensors that run at 5V require a 3.3-5V Logic Level Shift circuit.**

RASPBERRY PI 4 MODEL B RELATED PRODUCTS

PI Cobbler

It is often much easier to prototype a circuit using a breadboard. The Pi Cobbler helps make that task much easier by extending the GPIO pins to your breadboard.



PI HAT

Much like Arduino offers a wide range of Shields, the Raspberry Pi equivalent is called a *HAT*. The HAT fits snuggly on top of the GPIO pins to perform a speciality task.

RPi GPIO LIBRARY

There exists a wonderful pre-made library for configuring GPIO. Here is how to use it:
If your OS repository is equipped (i.e. You have Raspbian as your OS), enter the following:

```
sudo apt-get install rpi.gpio
```

If not, do it manually:

```
wget https://pypi.python.org/packages/source/R/RPi.GPIO/RPi.GPIO-0.5.11.tar.gz
tar -xvf RPi.GPIO-0.5.11.tar.gz cd RPi.GPIO-0.5.11 sudo python setup.py install
```

PYTHON CODING

Raspberry PI runs Python/Scratch code for the most part. Python is entirely text based and therefore can be written on any text editor. **Avoid** using a text editor that changes the format, as the program depends significantly on the text format (.py).

The RPI comes with a default development environment called *IDLE* where you can input commands. We know we can enter commands into the line directly. But what if we want to run several commands in a row? This is called a *script*.

First, install Python 3:

```
sudo apt-get install python3
sudo apt-get install python
```

Scripts can be written in Nano by creating a new script file:

```
sudo nano hello-world.py
```

Once your script is completed, you can compile and run it:

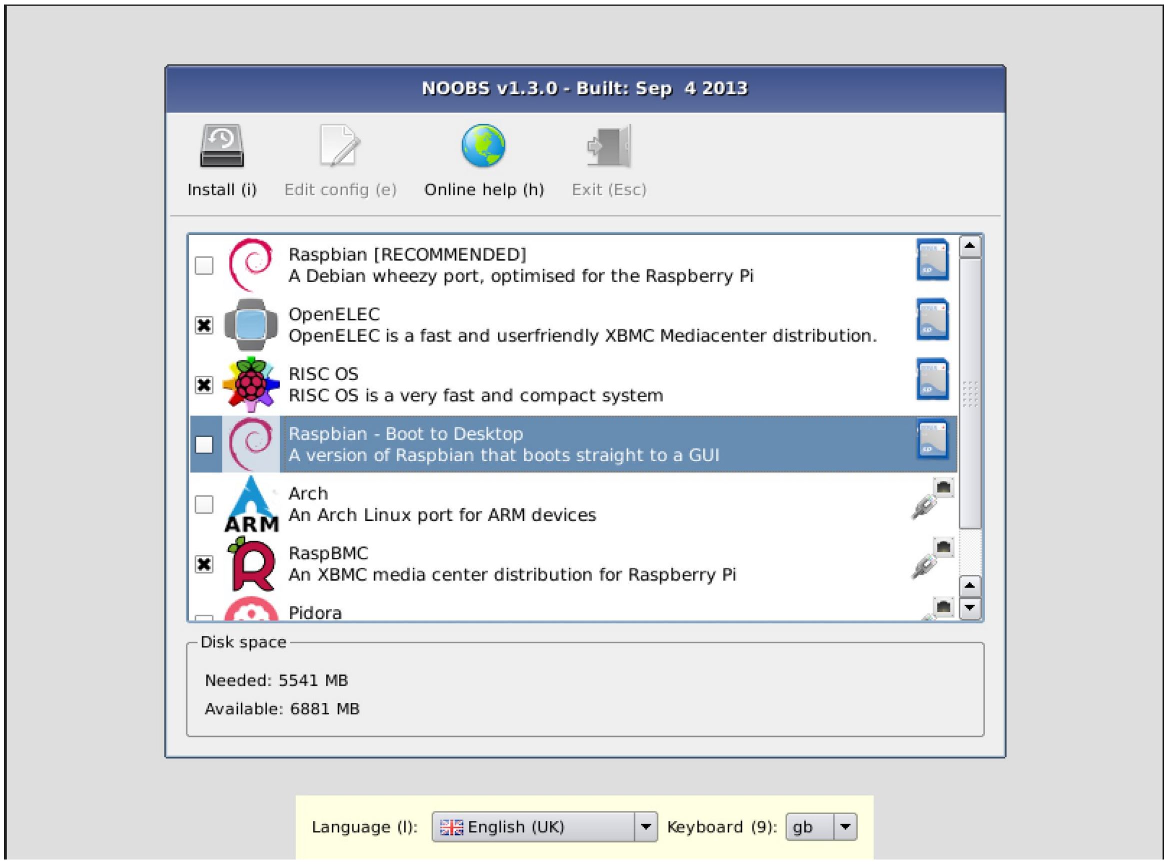
```
python hello-world.py
```

Refer to the following link for more information:
http://www.circuitbasics.com/how-to-write-and-run-a-python-program-on-the-raspberry-pi/

STARTING YOUR RASPBERRY PI

Please follow along to see how to easily you can load and begin using the Raspberry Pi computer. Loading by the image on the microSD memory card referred to as *NOOBS* is a way of making Raspberry Pi setup much easier. You won't need network access, and you won't need to download any special imaging software.

Loading the OS Using NOOBS



- [ ] Raspbian [RECOMMENDED] — A Debian wheezy port, optimised for the Raspberry Pi
- [x] OpenELEC — OpenELEC is a fast and userfriendly XBMC Mediacenter distribution.
- [x] RISC OS — RISC OS is a very fast and compact system
- [ ] Raspbian - Boot to Desktop — A version of Raspbian that boots straight to a GUI
- [ ] Arch — An Arch Linux port for ARM devices
- [x] RaspBMC — An XBMC media center distribution for Raspberry Pi
- Pidora

Disk space
Needed: 5541 MB
Available: 6881 MB

Language (l): English (UK) Keyboard (9): gb

Insert the microSD card in the memory slot underneath the board. Plug your USB C power connector into the port next to the micro-HDMI video ports. You'll need a USB keyboard plugged in. When you boot up for the first time, you'll see a flashing light and a menu prompting you to install one of the several available operating systems into the free space on the card. The choice means you can boot the Pi with a regular operating system like Raspbian, or any other that was mentioned above.

Once you've selected the OS you like, use dialogue box at the bottom of the page to select the language you prefer. The installer will build the OS that you selected. Now is a good time for a snack or a coffee break.

Once you've installed an operating system, your Pi will boot as normal. However, **NOOBS will remain on your card.**

You now have a user-friendly **desktop to configure settings and connect to the web**.

Command Line Troubleshooting

Holding the **Shift** key down during boot will open the recovery interface. This allows you to switch to a different operating system or overwrite a corrupted card with a fresh install of the current one. It also lets you edit the **config.txt** configuration file for the currently installed operating system directly.

Once the installation is complete your Pi will reboot, and a bunch of strange looking text will appear on your screen. This is called the “command line terminal”. At the bottom of the page, the last line reads:

```
        raspberrypi login:
```

The default username and password are:

```
        username: pi
        password: raspberry
```

When you hit the enter key you will see the following line:

```
        Pi@raspberrypi ~ $
```

Type the following to launch the desktop:

```
        startx
```

Once it’s up and running, it’s a good idea to grab the latest version of all the packages by connecting your Pi to the internet, opening a terminal and running:

```
        sudo apt-get update
        sudo apt-get upgrade
```

The Raspberry Pi configuration menu will start automatically the first time you boot, or can be run at any time by typing:

```
        sudo raspi-config
```

Congratulations! You now have started up your Pi!