**Raspberry Pi Zero W**

The Raspberry Pi is a popular Single Board Computer (SBC) in that it is a full computer packed into a single board. Many may already familiar with the [Raspberry Pi 3](https://www.sparkfun.com/products/13825) and its predecessors, which comes in a form factor that has become as highly recognizable. The Raspberry Pi comes in an even smaller form factor. The introduction of the Raspberry Pi Zero allowed one to embed an entire computer in even smaller projects. This guide will cover the latest version of the Zero product line, the [Raspberry Pi Zero - Wireless](https://www.sparkfun.com/products/14277), which has an onboard WiFi module. While these directions should work for most any version and form factor of the Raspberry Pi, it will revolve around the Pi Zero W.

**Hardware Specifications**

1. **Mini HDMI**

The Zero uses a mini HDMI connector to save space. To connect the Zero to a monitor or television, you will need a mini HDMI to HDMI adapter or [cable](https://www.sparkfun.com/products/14274).

1. **USB On-the-Go(OTG)**

To save space, the Zero has opted for a [USB On-the-Go (OTG)](https://en.wikipedia.org/wiki/USB_On-The-Go) connection. The Pi Zero uses the same Broadcom IC that powered the original Raspberry Pi A and A+ models.

This IC connects directly to the USB port allowing for OTG functionality, unlike the Pi B, B+, 2 and 3 models, which use an onboard USB hub to allow for multiple USB connections.

The Raspberry Pi 3 and other models have traditionally had 2-4 standard size female USB connectors, which allowed for all variety of devices to connect including mice, keyboards, and WiFi dongles.

To connect a device with a standard male USB connection, you will need a [USB OTG cable](https://www.sparkfun.com/products/14276). Plug the microUSB end into the Pi Zero, and plug your USB device into the standard female USB end. For use with other standard USB devices, it is recommended that you use a powered USB hub. Wireless keyboard and mouse combos work best as they have one USB dongle for both devices.

1. **Power**

Like other Pis, power is provided through a microUSB connector. Voltage supplied to the power USB should be in the range of **5-5.25V**.

1. **Micro SD card**

Another familiar interface is the microSD card slot. Insert your microSD cards that contains your Raspberry Pi image file here.

1. **WiFi & Bluetooth**

As with the Raspberry PI 3, the Zero W offers both 802.11n wireless LAN and Bluetooth 4.0 connectivity. This frees up many of the connections that would have been made over USB, such as a WiFi dongle and a USB keyboard and mouse if substituting a Bluetooth keyboard/mouse.

1. **Camera Connector**

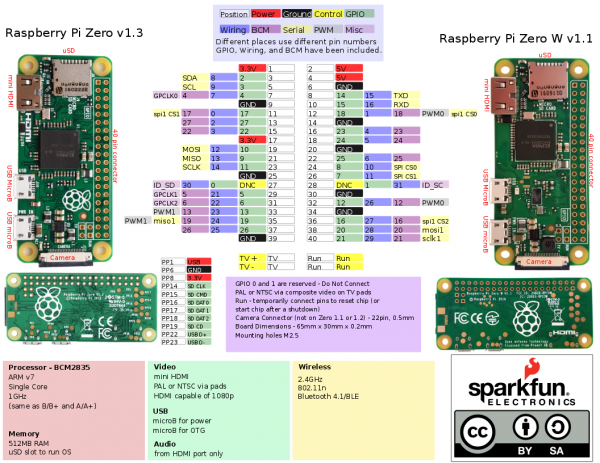
The Raspberry Pi Zero V1.3+ and all Zero W’s have an onboard camera connector. This can be used to attach the [Raspberry Pi Camera module](https://www.sparkfun.com/products/14028). However, the connector is a 22pin 0.5mm and different than the standard Pi. You will need a different [cable](https://www.sparkfun.com/products/14272) to connect the camera to the Pi Zero W.

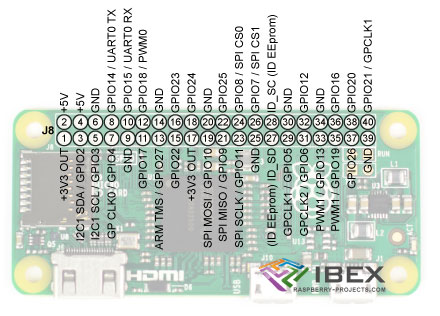
1. **GPIO**

As with all other models of the Raspberry Pi, there are a plethora of GPIO pins broken out, many of which other functionality such as I2C. If you are using the GPIO header, you may want to consider soldering [headers](https://www.sparkfun.com/products/14275) to it.

1. **Additional connections**

Two sets of thruhole pads labeled TV and Run. The TV pads allow you to connect an RCA jack to the board instead of using the HDMI out. The Run pins connect to the chips reset pin and will either turn the board off or turn it back on once it has been shut down. Connecting a button here is a good way to power cycle your board.



****

**Install Raspbian OS into SD card**

This resource explains how to install a Raspberry Pi operating system image on an SD card. You will need another computer with an SD card reader to install the image.

We recommend most users download [NOOBS](https://www.raspberrypi.org/documentation/installation/noobs.md), which is designed to be very easy to use. However, more advanced users looking to install a particular image should use this guide.

**Download the image**

Official images for recommended operating systems are available to download from the Raspberry Pi website <https://www.raspberrypi.org/downloads/>

The raspbian with Raspberry Pi Desktop and Raspberry Pi Lite. Desktop version is used for GUI related and the image contained in the ZIP archive over 4 GB in size.

**Writing an image to the SD card**

You will need to use an image writing tool to install the image you have downloaded on your SD card.

**Etcher** is a graphical SD card writing tool that works on Mac OS, Linux and Windows, and is the easiest option for most users. Etcher also supports writing images directly from the zip file, without any unzipping required. To write your image with Etcher:

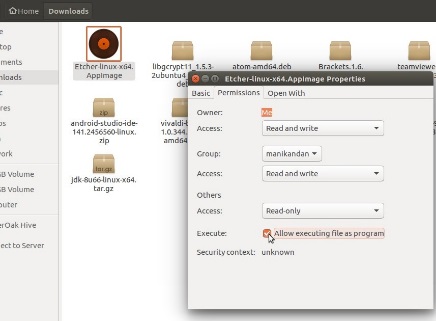
1. Download [Etcher](https://etcher.io/) and install it.
2. Connect an SD card reader with the SD card inside.
3. Open Etcher and select from your hard drive the Raspberry Pi .img or .zip file you wish to write to the SD card.
4. Select the SD card you wish to write your image to.
5. Review your selections and click 'Flash!' to begin writing data to the SD card.

**Install Etcher into Linux**

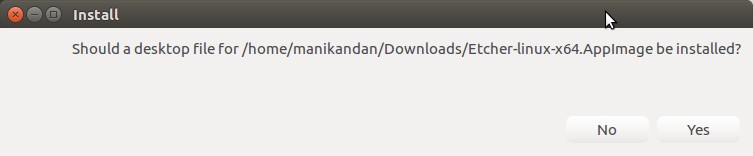
There are two ways to install Etcher image into Linux system.

**First method:**

1. Download etcher software from this path <https://etcher.io/> for your appropriate installed OS.
2. By default the downloaded files are stored in **/Home/Downloads/.**
3. Right click on App image file **etcher-electron-1.4.4-linux-x64.AppImage** and select properties. Go to permissions tab and check **Allow executing files as program option**.



1. Open a terminal and go to downloaded App image path.
2. Execute this command “sudo ./etcher-electron-1.4.4-linux-x64.AppImage”.



1. Select OS image to flash into SD card after Etcher installed successfully.

**Note:** Some may got issue like **“syntax error “(” unexpected”** while implementing step 4. In such scenario try an alternate way in second method as mentioned below.

**Second method:**

1. Select **Software & Updates** and go to **Other Software** tab.
2. Select **add** and place this line “**deb** <https://dl.bintray.com/resin-io/debian> **stable etcher**” into APT line.
3. Add this source into other software by authenticate with login password. It will be added in the Other Software and close it.
4. Open a terminal and execute this command “**sudo apt-key adv - -keyserver hkp://pgp.mit.edu:80 - -recv-keys 379CE192D401AB61**”.
5. Open a terminal and execute this command “**sudo apt update && sudo apt install etcher-electron**”.
6. Open Etcher and select OS image to flash into SD card.

If you want to install OS on Linux using command line tools then follow this link <https://www.raspberrypi.org/documentation/installation/installing-images/linux.md>

**WiFi headless setup**

Follow the link which is given in chapter references.

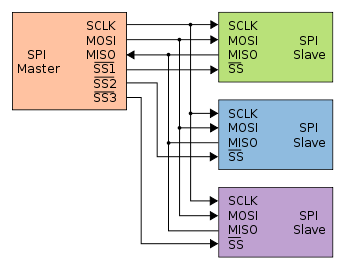
**Interfaces**

1. **SPI**

Serial peripheral Interface bus is a synchronous serial communication interface, used for short distance communication primarily in an embedded systems. Typical applications include **[Secure Digital](https://en.wikipedia.org/wiki/Secure_Digital" \o "Secure Digital)** (**SD**) cards and **[Liquid Crystal Display](https://en.wikipedia.org/wiki/Liquid_crystal_display" \o "Liquid crystal display)** (**LCD**). [Serial Peripheral Interface (SPI)](https://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus" \t "_blank) is a synchronous serial bus commonly used to send data between micro controllers and small peripherals such as **shift registers, sensors, and SD cards**.

The **Shift Register** is another type of sequential logic circuit that can be used for the storage or the transfer of binary data. This sequential device loads the data present on its inputs and then moves or “shifts” it to its output once every clock cycle.

SPI devices communicate in [full duplex](https://en.wikipedia.org/wiki/Full_duplex" \o "Full duplex) mode using a [master-slave](https://en.wikipedia.org/wiki/Master-slave_(technology)" \o "Master-slave (technology)) architecture with a single master. The master device originates the [frame](https://en.wikipedia.org/wiki/Frame_(networking)" \o "Frame (networking)) for reading and writing. Multiple slave devices are supported through selection with individual [slave select](https://en.wikipedia.org/wiki/Slave_select" \o "Slave select) (SS) lines as given in below image.



The SPI bus can operate with a single master device and with one or more slave devices.

**Data transmission**

To begin communication, the bus master configures the clock, using a frequency supported by the slave device, typically up to a few MHz. The master then selects the slave device with a logic level 0 on the select line. If a waiting period is required, such as for an analog-to-digital conversion, the master must wait for at least that period of time before issuing clock cycles.

During each SPI clock cycle, a full duplex data transmission occurs. The master sends a bit on the MOSI line and the slave reads it, while the slave sends a bit on the MISO line and the master reads it. This sequence is maintained even when only one-directional data transfer is intended.

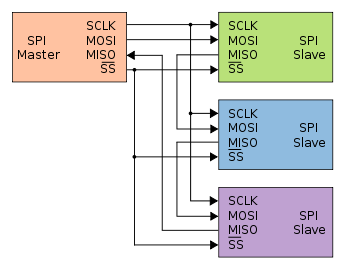
**Independent slave configuration**

In the independent slave configuration, there is an independent chip select line for each slave. A pull-up resistor between power source and chip select line is highly recommended for each independent device to reduce cross-talk between devices.[[3]](https://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus" \l "cite_note-dorkbotSPI-3) This is the way SPI is normally used. Since the MISO pins of the slaves are connected together, they are required to be tri-state pins (high, low or high-impedance).

**Daisy chain configuration**

Some products that implement SPI may be connected in a [daisy chain](https://en.wikipedia.org/wiki/Daisy_chain_(electrical_engineering)" \o "Daisy chain (electrical engineering)) configuration, the first slave output being connected to the second slave input, etc. The SPI port of each slave is designed to send out during the second group of clock pulses an exact copy of the data it received during the first group of clock pulses.

The whole chain acts as a communication [shift register](https://en.wikipedia.org/wiki/Shift_register" \o "Shift register); daisy chaining is often done with shift registers to provide a bank of inputs or outputs through SPI. Each slave copies input to output in the next clock cycle until active low SS line goes high. Such a feature only requires a single SS line from the master, rather than a separate SS line for each slave.



**Transfer Mode**

SPI devices sometimes use another signal line to send an interrupt signal to a host CPU. Examples include pen-down interrupts from touchscreen sensors, thermal limit alerts from temperature sensors, alarms issued by real time clock chips, [SDIO](https://en.wikipedia.org/wiki/Secure_Digital" \l "SDIO" \o "Secure Digital),[[6]](https://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus" \l "cite_note-3wireSDI-6) and headset jack insertions from the sound codec in a cell phone. Interrupts are not covered by the SPI standard; their usage is neither forbidden nor specified by the standard.

**LoopBack Testing**

This can be used to test SPI send and receive. Connect a wire between MOSI and MISO. It does not test CE0 CE1. Download source code from this path “<https://raw.githubusercontent.com/raspberrypi/linux/rpi-3.10.y/Documentation/spi/spidev_test.c>”.

Compile the source code as mentioned below by using GCC,

gcc –o spidev\_test spidev\_test.c

./spidev\_test -D /dev/spidev0.0

spi mode: 0

bits per word: 8

max speed: 500000 Hz (500 KHz)

FF FF FF FF FF FF

40 00 00 00 00 95

FF FF FF FF FF FF

FF FF FF FF FF FF

FF FF FF FF FF FF

DE AD BE EF BA AD

F0 0D

Please follow the below mentioned paths for interfacing an SPI ADC (Analog to Digital Converter) chip to Raspberry Pi using C++ and Python respectively.

<http://www.hertaville.com/interfacing-an-spi-adc-mcp3008-chip-to-the-raspberry-pi-using-c.html>

<https://raspberrypi-aa.github.io/session3/spi.html>

**Using an SPI interface with C**

<http://www.raspberry-projects.com/pi/programming-in-c/spi/using-the-spi-interface>

**References**

<https://learn.sparkfun.com/tutorials/getting-started-with-the-raspberry-pi-zero-wireless> <https://github.com/initialstate/pi-zero-w-motion-sensor/wiki/Part-1.-Setting-Up-the-Pi-Zero-W>

**How to solder GPIO pins**

<http://dbakevlar.com/2016/03/adding-gpio-pins-to-the-raspberry-pi-zero/>

**How to use GPIO pins without soldering**

<http://www.instructables.com/id/Testing-Pi-Zero-Without-Soldering-Header/>

**Install Etcher**

<https://connectwww.com/how-to-install-and-run-etcher-image-burner-on-ubuntu/4676/>

<https://www.omgubuntu.co.uk/2017/05/how-to-install-etcher-on-ubuntu>

<https://www.youtube.com/watch?v=4hJsTjYib2w>

**WiFi setup in Raspberry Pi zero without display (headless)**

<https://core-electronics.com.au/tutorials/raspberry-pi-zerow-headless-wifi-setup.html>

<https://medium.com/@DavidMaitland/raspberry-pi-zero-headless-setup-92fb72daf88d>

**SPI interfaces**

<https://www.raspberrypi.org/documentation/hardware/raspberrypi/spi/README.md>

<https://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus>

**How to enable SPI and I2C**

<https://www.raspberrypi-spy.co.uk/2014/08/enabling-the-spi-interface-on-the-raspberry-pi/>

<https://learn.sparkfun.com/tutorials/raspberry-pi-spi-and-i2c-tutorial>

**Controlling Raspberry Pi zero over UART**

<https://elementztechblog.wordpress.com/2016/05/02/controlling-raspberry-pi-zero-over-uart/>

**Configuring the GPIO serial port on Raspbian**

<https://spellfoundry.com/2016/05/29/configuring-gpio-serial-port-raspbian-jessie-including-pi-3/>

**Basics of UART communication circuit**

<http://www.circuitbasics.com/basics-uart-communication/>