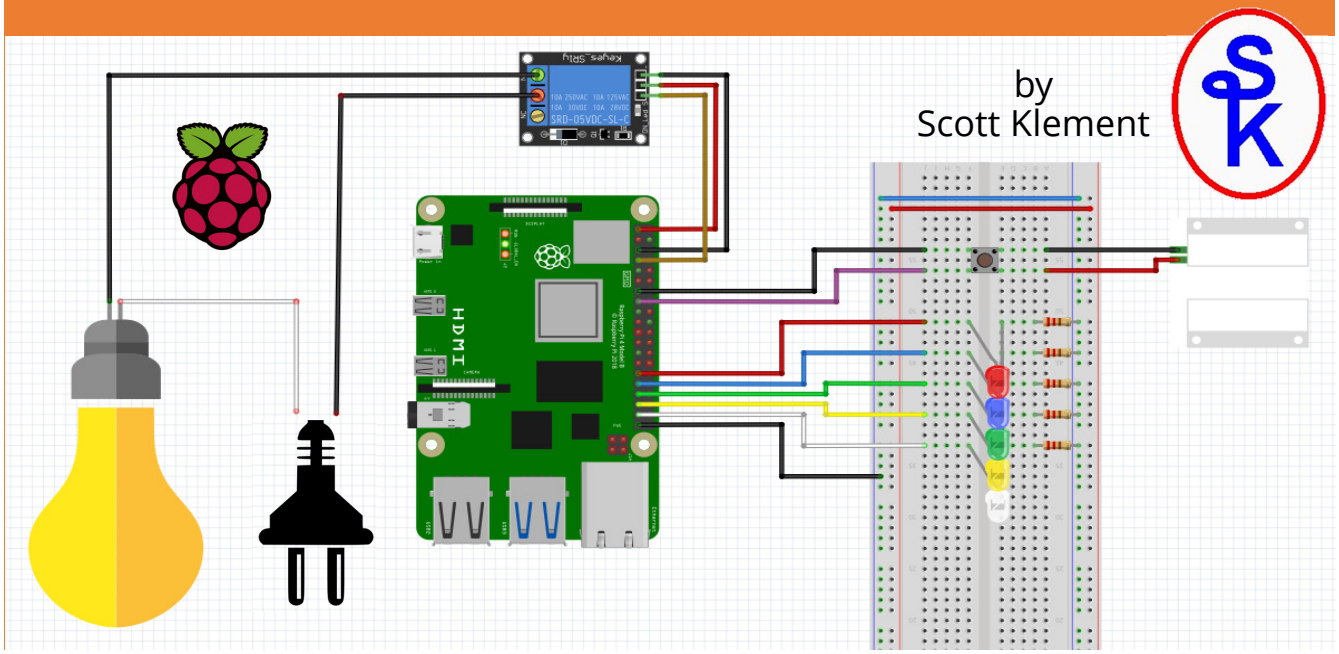
A First Look at Physical Computing With Raspberry Pi



by
Scott Klement

Two Raspberry Pi computers (with my hand for comparison)

- Zero-W (approx. $10)
- 4B 4GB (approx. $60)

What?

The Raspberry Pi is a small, inexpensive, energy efficient computer.

- Storage on microSD cards
- HDMI video/audio
- USB ports for keyboard/mouse
- Wired & Wifi Networking
- ARM-based CPU
- GPIO pins

Why?

Small, efficient, and inexpensive.

- Stick it under a table
- Back of a monitor
- Inside/above/under a cabinet
- Inside a vehicle
- ...really, anywhere...

Raspberry Pi

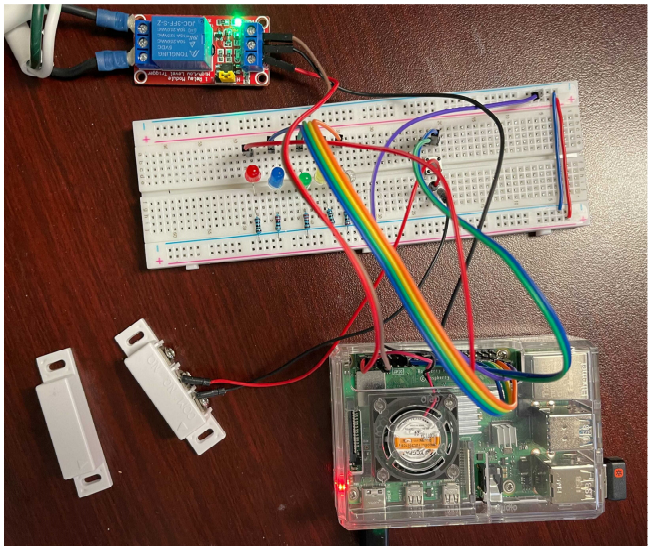
Use it to control electronics, and interface with a bigger computer.

- Such as your PC
- Power Systems running IBM i

Write your own custom application that controls the GPIO pins, and make it do whatever you want.

- Turn stuff on or off.
- Read sensors (temperature, pressure, "eyes"/infrared, cameras)
- Make conveyors move.
- Interface printers, scanners, scales, industrial terminals...
- The possibilities are endless!
- Motors/Servos
- Ultrasonic sensors
- Solenoid valves

Show Me!



Simple demonstration of

- Turn LEDs on/off.
- Turn Power (Lamp) on/off.
- Read a door sensor.

Imagine...

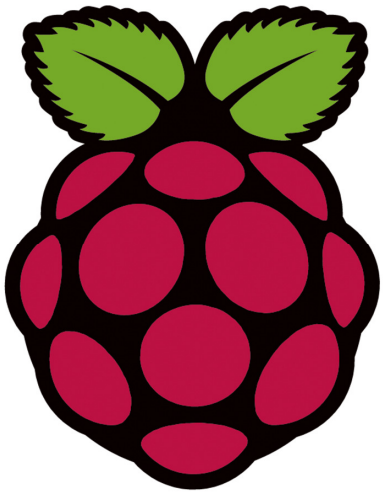
I demonstrated turning LEDs and a light on/off.

Think of all of the other things you could turn off!

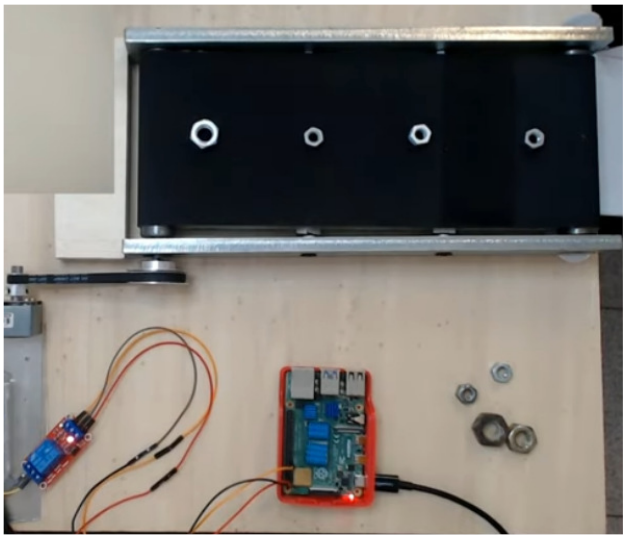
And it's done by program logic, so could be done under any logic you can imagine.

Likewise, I demonstrated reading from a switch.

Think of all of the other sensors and devices you can read from.



Conveyor

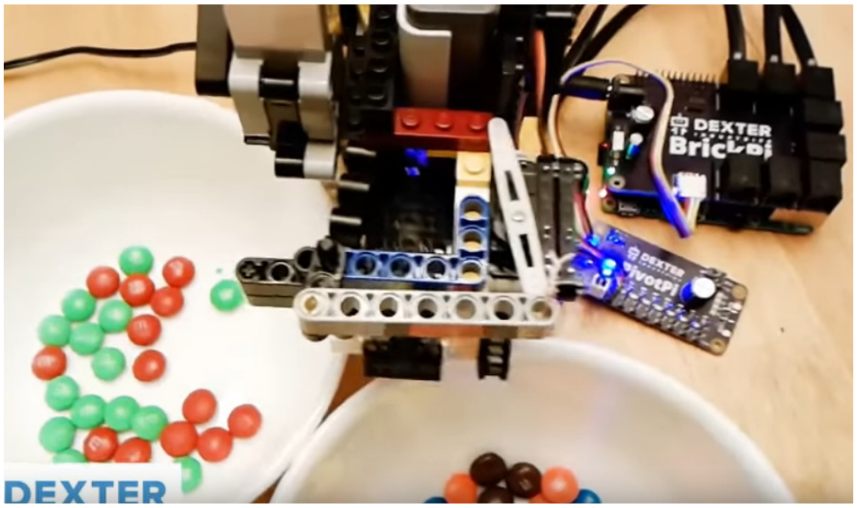


There are cameras available that work well with image detection software (such as OpenCV)

This project stops the conveyor belt when a nut over a given size is detected at the end of the conveyor. The worker could then remove the one that's too large.

With servos you could build a robotic arm (or buy a prebuilt one) that removes the nut. (Or any other type of item.) -- not shown.

Sorting M&Ms



Sorts M&Ms by color. Pretty cool, to play with -- but there are also industrial uses for it.

Programming GPIO

In Node.js, I use the open source onoff module to interact with the gpio pins.

To turn one on, you simply write 1 to it. To turn it off, write 0.

```
var Gpio = require("onoff").Gpio;

var led = new Gpio(5, 'out');

// Turn on
led.writeSync(1);

// Turn off after 5 seconds
setTimeout(() => led.writeSync(0), 3000);
```

GPIO Toggle

In Raspberry Pi, you can read from pin (even if in 'out' mode)

This makes it easy to toggle.

```
var Gpio = require("onoff").Gpio;

var relay = new Gpio(4, 'out');

function toggle() {
  var currentValue = relay.readSync();
  relay.writeSync(currentValue ^ 1);
}

toggle();
setTimeout(toggle, 3000);
```

Async Button

Naturally, you can read the same way you do a write. But, how do you know when to read?

onoff provides a watch event that can fire a function when the state of a button changes. It can be 'falling' (switch closing), 'rising' (switch opening) or 'both'.

There's also a debounceTimeout to avoid the situation where a button might open/close more than once rapidly.

```
var Gpio = require("onoff").Gpio;

var button = new Gpio(23, 'in', 'both',
                      {debounceTimeout: 10});

button.watch((err, value) => {
  console.log((value===1) ? 'up':'down');
});
```

The End.

The code to write the REST API and communicate with RPG is too much to show you here.

But it is similar to any other REST API you might code.

The entire code is on GitHub, if you want to see it.

https://github.com/ScottKlement/rpg-raspi-demo

Likewise, working with cameras, servos, ultrasonics, and more is possible, but too much for a 20 minute session. There's tons of examples and sample projects online, though -- just google it!

Before you know it, you'll be interacting with the physical world, doing physical computing.