EET 2812

Activity 7

Single Board Computer (RPi)

LED Sensor

Cuyahoga Community College

Youth Technology Academy

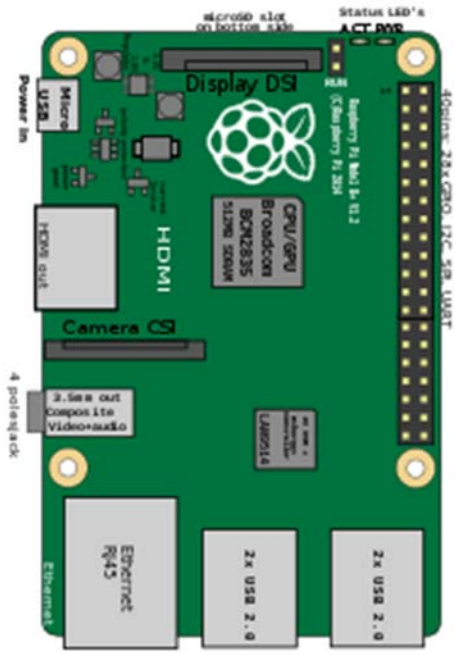
ACTIVITY

RPi Sensors Programming – LED

Overview: Learn about the GPIO pins on the RPi. Create a circuit with the RPi and an LED then write a python program to light an LED.

Vocabulary: RPi, GPIO Pins, LED, resistor, breadboard, ground, function library.

GPIO Pins on the RPi



Raspberry Pi B+
B+ J8 GPIO Header

	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7
DNC	27	28	DNC
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21

3.3V and 5V: Power pins

GPIO: General Purpose Input Output pins

GND(Ground): Electrical grounding is important because it provides a reference voltage level (called zero potential or ground potential) against which all other voltages in a system are established and measured. An effective electrical ground connection also minimizes the susceptibility of equipment to interference, reduces the risk of equipment damage due to lightning, eliminates electrostatic buildup that can damage system components, and helps protect personnel who service and repair electrical, electronic, and computer systems. In effect, an electrical ground drains away any unwanted buildup of electrical charge. When a point is connected to a good ground, that point tends to stay at a constant voltage, regardless of what happens elsewhere in the circuit or system. The earth, which forms the ultimate ground, has the ability to absorb or dissipate an unlimited amount of electrical charge.

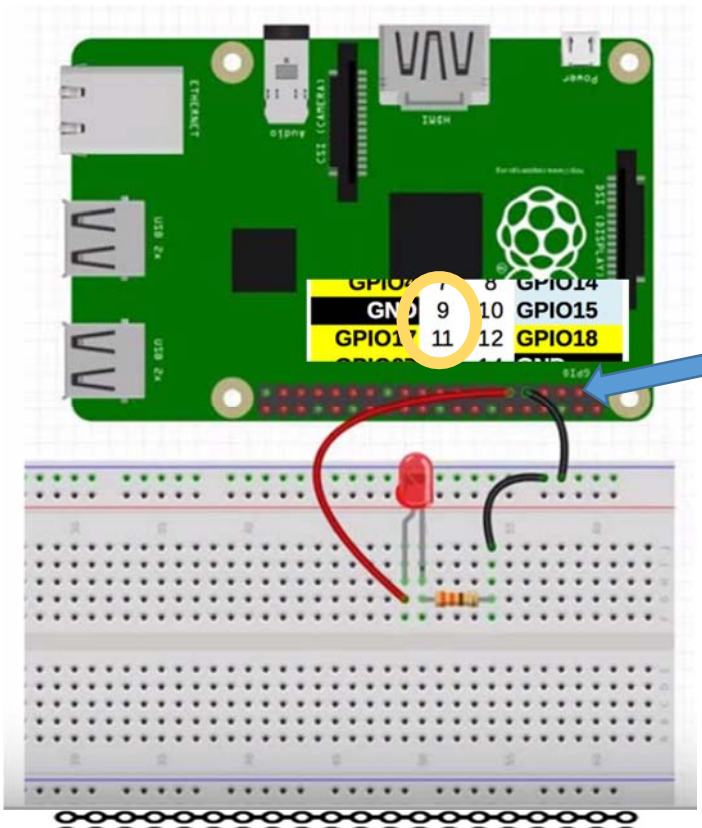
A chassis ground is a connection to the main chassis of a piece of electronic or electrical equipment. In older appliances and in desktop computers, this is a metal plate, usually copper or aluminum. In some modern equipment, it is a foil run on the main printed circuit board, usually running around the periphery. Chassis ground is sometimes called common ground. It provides a point that can be considered to have zero voltage. All other circuit voltages (positive or negative) are measured or defined with respect to it.
(from http://whatis.techtarget.com/definition/ground)

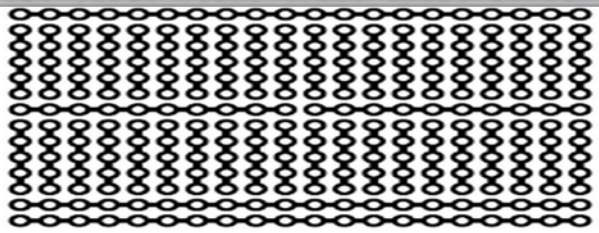
DNC: Do Not Connect

Resistors: If we allow too much current through the LED, it will burn very bright for a very short period of time before it burns out, so we need a resistor to limit the current. Calculating the resistor value is not difficult but for now, just use anything from 270Ω to 330Ω. Anything higher will make the LED dimmer.
(from https://projects.drogon.net/raspberry-pi/gpio-examples/tux-crossing/gpio-examples-1-a-single-led/)

LED: LEDs are Light Emitting Diodes and the diode part is important for us – they only pass electricity one way, so we need to make sure we put them in the circuit the right way round. They have a long leg and a slightly shorter leg. The long leg goes to the plus side (to the power pin) and the shorter leg to the negative (or 0v) side.

Blink the LED





Breadboard wiring layout –
how holes are connected with wires

Materials:

RPi, Breadboard, LED, Resistor and wires (2 male and 2 female), laptop computer running Putty.

Instructions:

Connect the circuit as shown using pins GND9 and 11

LED – DIRECTION MATTERS! Long leg always connects to power on the board

- at the command prompt make sure you are at the home directory by typing **$cd**

```
pi@raspberrypi ~ $ mkdir my_python
```

- make a directory called my_python to store our programs by typing **$mkdir my_python**
- you can change to the my_python directory by typing **$cd my_python**

We're going to write our first program using the python shell. To get here type **$sudo python** from the home directory (type **$cd** if you are in another directory). You should see the following:

```
pi@raspberrypi ~ $ sudo python
Python 2.7.3 (default, Mar 18 2014, 05:13:23)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more info
rmation.
>>>
```

Type in each command as follows. If you make a mistake the shell will let you know right away.

>>>import RPi.GPIO as GPIO	import the GPIO library – Lets the RPi know what the pins do
>>> GPIO.setmode(GPIO.BOARD)	number scheme we are going to use – we will use the BOARD numbers (the physical pin numbers)
>>>GPIO.setup(11,GPIO.OUT)	tell the RPi which pin you are using and whether the pin is an input or an output pin.
>>>GPIO.output(11,True)	Turn on or send power to the pin
>>>GPIO.output(11,False)	Turn off or stop sending power to the pin
>>>GPIO.output(11,1)	True=1
>>>GPIO.output(11,0)	False =0
>>>GPIO.cleanup()	Reset all the pins – **ALWAYS END YOUR PROGRAMS WITH THIS COMMAND!**

Press "Ctrl Z" to exit the python shell.

Turn on the LED by writing a program in python

Change directory to my_python **$cd my_python**

We're going to use the text editor on the RPi called nano to write a python program called blink. Type: **$nano blink.py**

```
pi@raspberrypi ~ $ ls
Desktop  my_python  python_games
pi@raspberrypi ~ $ cd my_python
pi@raspberrypi ~/my_python $ nano blink.py
```

The screen should look like this:

We're going to type in almost the same thing we did earlier to turn on the LED. This time however, we will use a variable to store the pin number we are going to turn on and off.

Notice that some words change color. Why do you think this happens?

Type in the following:

```
import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
red=11
GPIO.setup(red,GPIO.OUT)
GPIO.output(red,True)
time.sleep(1)
GPIO.output(red,False)
time.sleep(1)
GPIO.output(red,True)
time.sleep(1)
GPIO.output(red,False)
GPIO.cleanup()
```

```
File Name to Write: blink.py
^G Get Help    M-D DOS Format  M-A Append
^C Cancel      M-M Mac Format  M-P Prepend
```

import time is a command to import the time function library which allow us to turn the LED on and off.

Then press **CTRL O** to write (save) the file, **ENTER** and then **CTRL X** to exit.

When finished type in the command **$sudo python blink.py** to run the program. The LED in your circuit should blink off and on twice.

```
pi@raspberrypi ~/my_python $ sudo python blink.py
```

Open the blink program again to edit. **$nano blink.py**

This time we will ask the user how many times they want the light to blink using variables and use a "**for loop"** to turn the light on and off multiple times.

```
 GNU nano 2.2.6              File: blink.py

import time
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BOARD)
red=11
GPIO.setup(red,GPIO.OUT)
blink_num=input('How Many TImes Do You Want to Blink?')
for i in range(0,blink_num):
        GPIO.output(red,True)
        time.sleep(1)
        GPIO.output(red,False)
        time.sleep(1)

GPIO.cleanup()
```

Exit the editor and Run the program.

information adapted from www.toptechboy.com