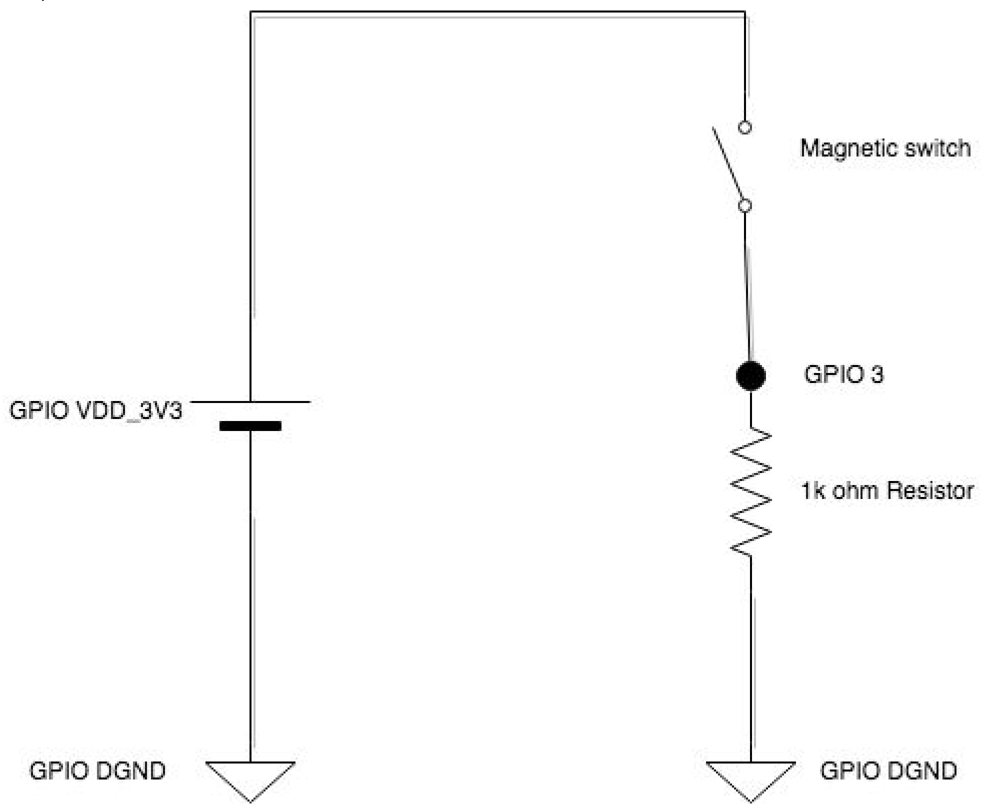
How-to connect a magnetic switch via GPIO to Beagle Bone Green

By team LMV (Leo Zhang, Min sohng, Victor Peng)
Summer 2018, CMPT 433

A magnetic switch, can be purchased from https://elmwoodelectronics.ca/products/13247
The switch will generate an value of 0 when closed and 1 when open. It can be utilize in programs that require manual trigger.

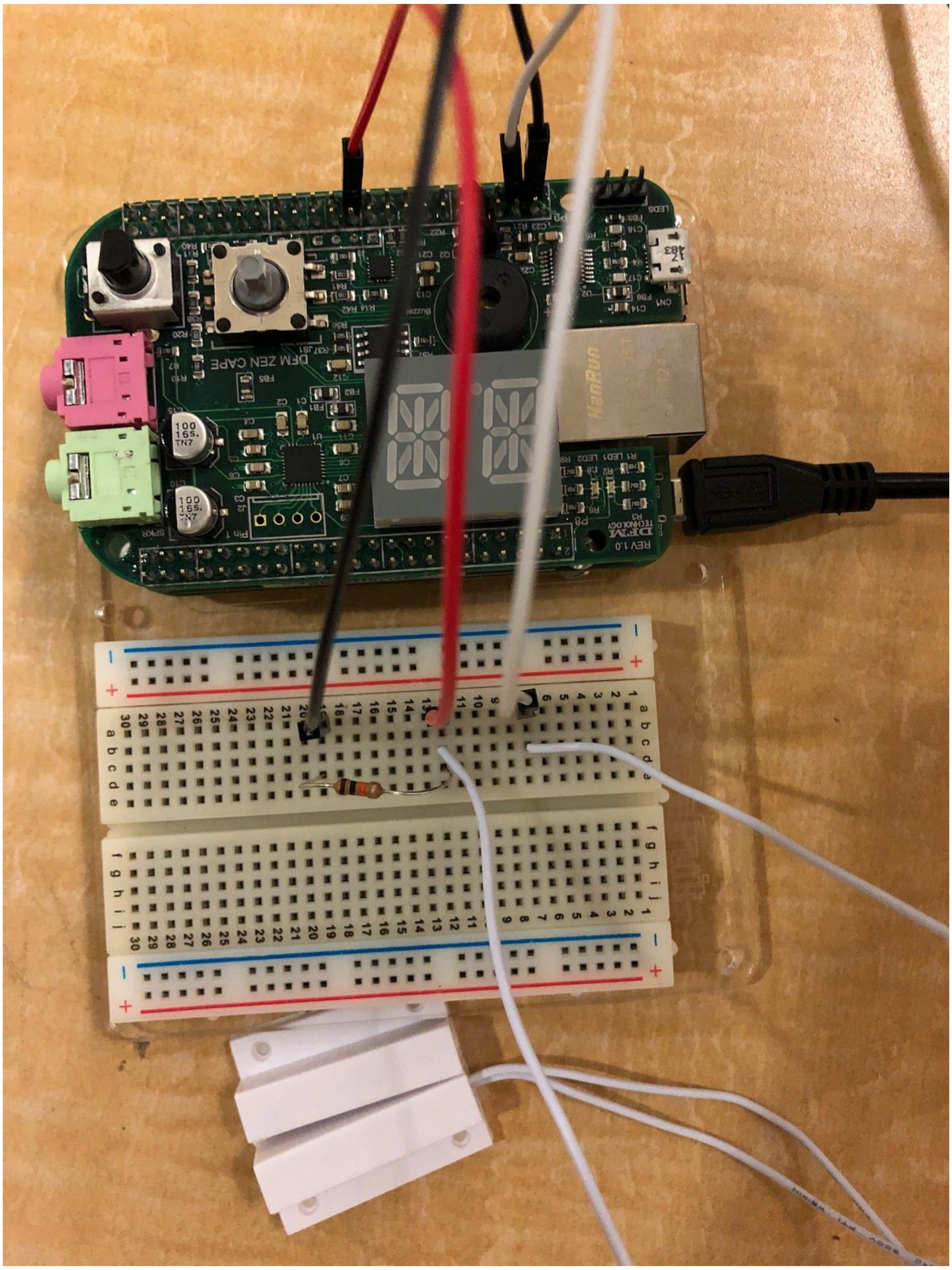
A circuit with a magnetic switch, a resistor, a GPIO DGND pin and a GPIO VDD_3V3 pin in series is constructed for setup.
I.e.)

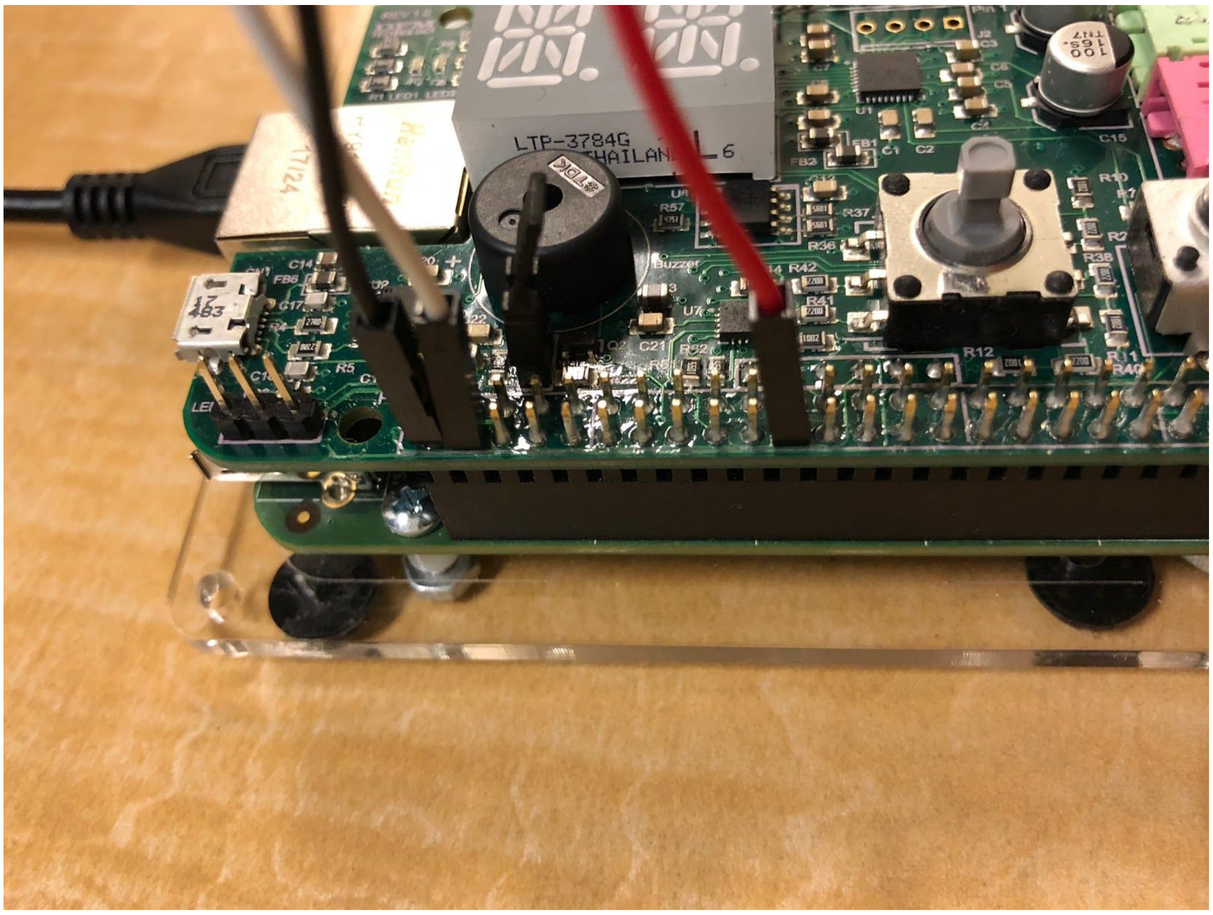


Materials required for setting up the circuit:

- A magnetic switch
- Three jumpers
- A 1k ohm resistor

Note:
Using 1k ohm resistor, so we can produce a 3.3/1k = 3.3 mA which in safe to the BBG (up to 8 mA) and the switch (up to 500mA)


DFM ZEN CAPE
HanRun
REV 1.0



Steps to read the magnetic switch value

1. Change the current working directory to /sys/class/gpio.

```
~# cd /sys/class/gpio
/sys/class/gpio# ls
export  gpiochip0  gpiochip32  gpiochip64  gpiochip96  unexport
```

2. Export the gpio pin (3 in our case)

```
# echo 3 > export
# ls
export  gpio3  gpiochip0  gpiochip32  gpiochip64  gpiochip96  unexport
```

3. Change the current working directory to gpio3.

```
# cd gpio3
# ls
active_low  device  direction  edge  power  subsystem  uevent  value
```

4. Check the switch status by reading the value file (1 means closed and 0 means open)

```
# cat value
1
# cat value
0
```

5. (Optional) Unexport the gpio pin when done

```
# cd ..
# echo 3 > unexport
# ls
export  gpiochip0  gpiochip32  gpiochip64  gpiochip96  unexport
```

C program for reading magnetic switch value:

```c
#include <stdio.h>
#include <stdlib.h>
#define GPIO_EXPORT_PATH             "/sys/class/gpio/export"
#define GPIO_PIN                     "3"
#define GPIO_PIN_VALUE_PATH          "/sys/class/gpio/gpio3/value"
#define SWITCH_CLOSED                '1'
#define SWITCH_OPEN                  '0'
int main(void)
{
  FILE *fd;
  // export GPIO pin
  fd = fopen(GPIO_EXPORT_PATH, "w");
  if (!fd) {
        fprintf(stderr, "Failed opening %s\n", GPIO_EXPORT_PATH);
        return 1;
  } else {
        int char_written = fprintf(fd, "%s", GPIO_PIN);
        fflush(NULL);
        if (char_written <= 0) {
                fprintf(stderr, "Failed writting %s to %s\n", GPIO_PIN, GPIO_EXPORT_PATH);
                return 1;
        }
        fclose(fd);
  }
  // poll on switch value
  char prev_val = ' ';
  while(1) {
        fd = fopen(GPIO_PIN_VALUE_PATH, "r");
        if (!fd) {
                fprintf(stderr, "Failed opening %s\n", GPIO_PIN_VALUE_PATH);
                return 1;
        } else {
                char val = fgetc(fd);
                if (!val) {
                        fprintf(stderr, "Failed reading %s\n", GPIO_PIN_VALUE_PATH);
                } else {
                        if (prev_val != val && val == SWITCH_OPEN)
                                printf("Switch is open\n");
                        else if (prev_val != val && val == SWITCH_CLOSED)
                                printf("Switch is closed\n");
                        prev_val = val;
                }
                fclose(fd);
        }
  }
}
```

The program above exports the GPIO pin that reads the switch then continuously poll on the status of the switch. It would generate a log message if the status changes

Troubleshooting Guide

1. Make sure the wiring are correct before connecting to power
2. Make sure the right GPIO pin is exported when reading the value file
3. The switch can only be closed if they are positioned correctly

Reference

http://www.cs.sfu.ca/CourseCentral/433/bfraser/other/GPIOGuide.pdf
https://elmwoodelectronics.ca/products/13247