Group Name: U1F609

Group Members:

Raymon Gulati, rgulati@sfu.ca
Suhua Qin, aprilq@sfu.ca
Victor Yuan, vqyuan@sfu.ca
Agnes Utomo, automo@sfu.ca

This How To Guide contains the following two guides:

1. Grove PIR Motion Sensor Guide
2. BBB Live Video Streaming to Display on HTML Guide

Grove PIR Motion Sensor Guide

This document guides the user through:

1. Wiring up and setting up the Motion Sensor
2. Verifying setup via Command Line
3. A program sample in C which interacts with the Motion Sensor

Table of Contents

Formatting:

- Commands starting with `$` are Linux console commands on the host PC:

  ```
  $ echo "Hello world!"
  ```

- Commands starting with `#` are Linux commands on the target (BeagleBone):

  ```
  # echo "Hello embedded world!"
  ```

- Almost all commands are case sensitive.

1. Introduction

PIR Sensors or Passive InfraRed Sensors work by detecting the difference in infrared amounts between two infrared-sensitive slots. When a person or animal passes by, it radiates infrareds, which will be noticed by the first slot. The difference between the new infrared level in slot one and the ambient (environment level) infrared in the second slot is what cause the sensor to detect motion.

Grove PIR Motion Sensor is a digital sensor which will output either 1 (HIGH) or 0 (LOW) to its signal pin. The sensor has a detecting distance of 3 metres and detecting angle of 120 degrees. Up to the time this guide is written, the sensor does not support BeagleBone and we can’t use the Grove I2C connector on BeagleBone Green.

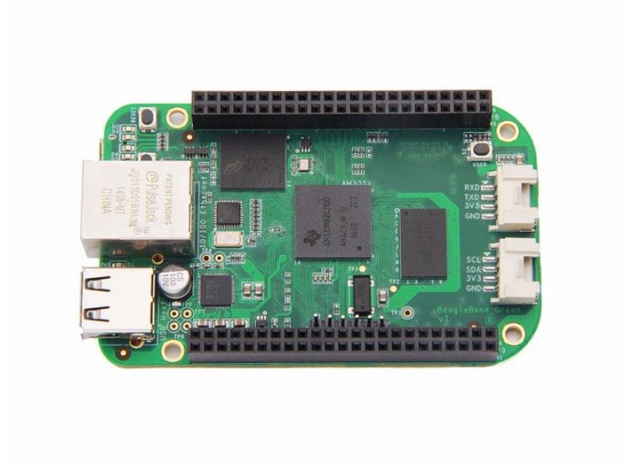
2. Setup

Grove PIR Motion Sensor uses a specific Grove cable which consists of four cables: red, black, white, and yellow cable. The red cable is for power, black cable is for ground, white cable is for secondary digital I/O, and yellow cable is for primary digital I/O.

2.1 Required Parts

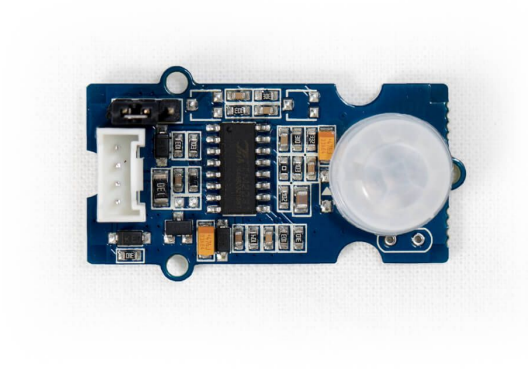
The following parts are needed to follow this guide:

- 1x BeagleBone Green



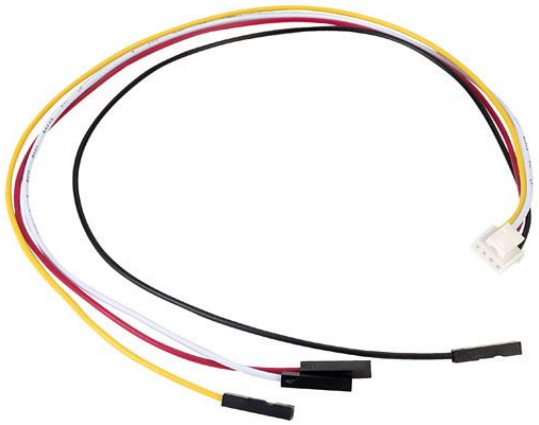
You can get it here.

- 1x Grove PIR Motion Sensor



You can get it here.

- 1x Grove 4 pin Female Jumper to Grove 4 pin Conversion Cable

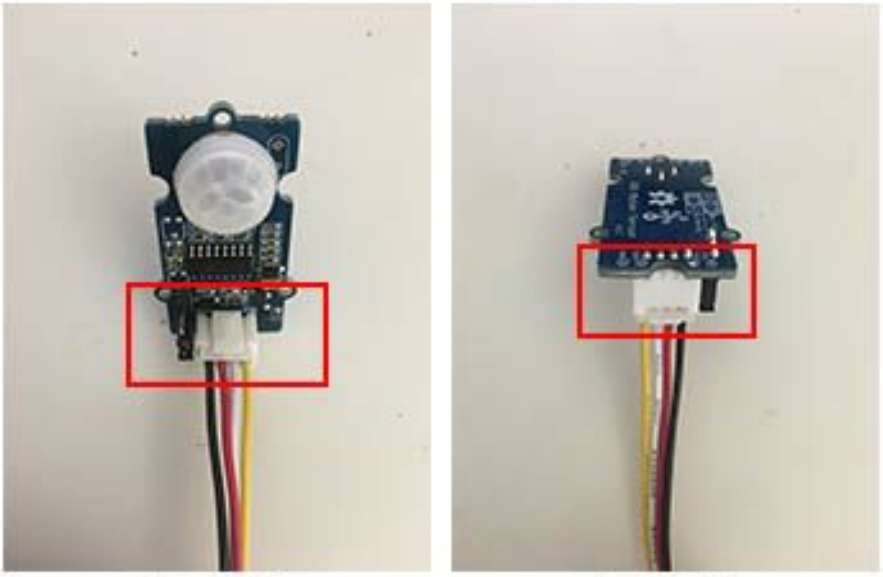


You can get it here.

2.2 Wiring the sensor

1. Do the following before you start wiring:
    - To ensure safety, turn off your BeagleBone before executing the following steps.
    - To avoid static, wear an antistatic wrist strap or ground yourself by touching an unpainted metal on your computer, such as the USB port.
2. Plug in the Grove 4 pin cable to the Grove connector on the Grove PIR Motion Sensor.

- It should look like this:

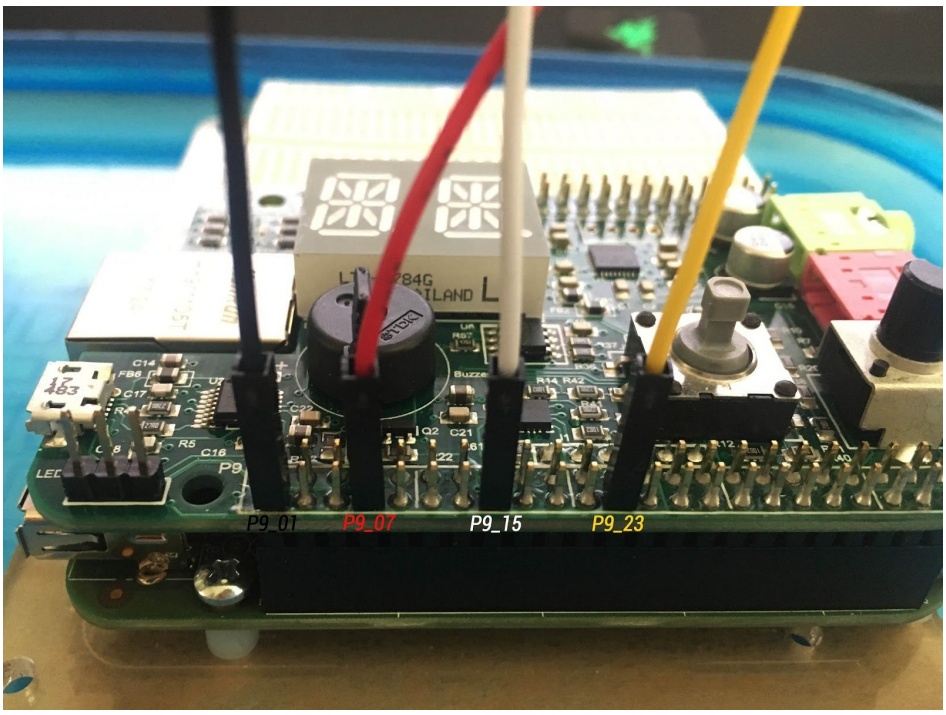


Front Side Back Side

- Ensure that the cable is fully plugged into the Grove connector.

3. Wire up the four female jumper cables:
   - **Black** cable to *GROUND* pin (**P9_01 or P9_02**)
   - **Red** cable to *SYS_5V* pin (**P9_07 or P9_08**)
   - **White** cable to a *free* GPIO pin (**P9_12, P9_15, P9_23,** or **P9_27**)
   - **Yellow** cable to a *free* GPIO pin (**P9_12, P9_15, P9_23,** or **P9_27**)

In the following diagram, I use P9_01, P9_07, P9_15, and P9_23:



Notes:

- Make sure that each female jumper cable is tightly connected to the pin. Otherwise, it would be easily plugged out from the pins.
- P9_12 corresponds to GPIO #60.
- P9_15 corresponds to GPIO #48.
- P9_23 corresponds to GPIO #49.
- P9_27 corresponds to GPIO #115.
- You can check the header mappings here.

2.3 Setting up via Command Line

This section assumes that you have successfully completed Brian Fraser's Quick-Start guide and have access to your BeagleBone.

1. Connect your BeagleBone to your host PC.
2. SSH to your target device:

```
$ ssh root@192.168.7.2
```

- Change the IP address accordingly. You can check the IP assigned by running `ifconfig` on your host.

3. Go to `sysfs` directory for GPIO:
   ```
   # cd /sys/class/gpio
   ```
4. Export the pin which is connected to you primary I/O (*yellow cable*) in order to tell Linux to handle is as GPIO.
   ```
   # echo 49 > export
   ```
   - You will get the following error if the pin has been exported:
     `write error: Device or resource busy`
5. Verify the export:
   ```
   # ls /sys/class/gpio
   ```
   - It should show `gpio49`. See the screenshot below:

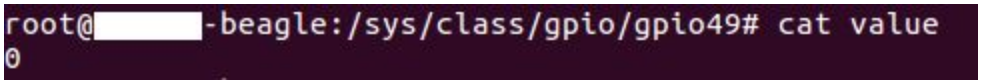
6. Troubleshooting
   - If you receive `write error: Device or resource busy` while trying to export the pin, it may mean that the pin has already been exported or loaded at startup by a virtual cape.
     - Try rewiring the yellow cable to a different pin and export the corresponding GPIO pin
     - You can find the list of free pins here:
       - P8 Header Table
       - P9 Header Table

3. Verifying Setup

In this section, we will verify the setup by reading the value of GPIO pin 49 (P9_23). Change the pin based on your wiring.

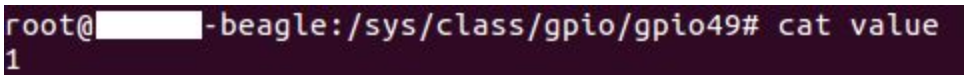
1. To reduce as much noise, it is recommended to set the PIR Motion Sensor in an area with no animals or people.
2. Set up the sensor to face a wall / ceiling.
3. Check the reading on the GPIO pin:
   - `# cd /sys/class/gpio/gpio49`
   - `# echo in > direction`
   - `# cat value`

   You should get a 0 since there's no motion in front of the sensor:

4. Put your hand in front of the sensor and read the pin again:
   - `# cat value`

   You should get a 1 now:

5. Troubleshooting:
   - If the value is switched (1 for step 3 and 0 for step 4), try checking the value for `active_low`:

     `# cd /sys/class/gpio/gpio49`

     `# cat active_low`
     - If it returns 1, set it to 0

       `# echo 0 > active_low`
   - If the value returned is 0 on step 4:
     - Check if each of the female jumper cable is properly connected to the pins.
     - The PIR Motion Sensor requires 5V to power it up, so it won't work if you connect it to VDD_5V or DC_3.3V.
     - Check if you're connecting the yellow cable to the correct pin and the corresponding GPIO number.

- Try connecting the yellow cable to a different pin.
- Try re-exporting the pin:

```
# echo 49 > unexport
# echo 49 > export
```

- Try rebooting the board

4. C Code

4.1 Setting up sensor

To set up the sensor, we simply write the `GPIO_NUM` (49 in the code) to the `/sys/class/gpio/export` file:

```
void initPIRSensor() {
    FILE *pFile = fopen(“/sys/class/gpio/export”, “w”);
    if (pFile == NULL) {
        printf("ERROR: Unable to open file.\n");
        exit(1);
    }

    int written = fprintf(pFile, "%d", 49);
    if (charWritten <= 0) {
        printf("ERROR: Unable to write file.\n");
    }

    fclose(pFile);

    FILE *pFile = fopen(“/sys/class/gpio/gpio49/direction”,
“w”);
    if (pFile == NULL) {
        printf("ERROR: Unable to open file.\n");
        exit(1);
    }

    int written = fprintf(pFile, ”in”);
    if (charWritten <= 0) {
        printf("ERROR: Unable to write file.\n");
    }

    fclose(pFile);
}
```

4.2 Reading the sensor

The following code assumes that the sensor's signal pin is wired to GPIO #49:

```
void readSensor() {
    FILE *pFile = fopen("/sys/class/gpio/gpio49/value", "r");
    if (pFile == NULL) {
        printf("ERROR: Unable to open file.\n");
        exit(1);
    }

    int buff_size = 1024;
    char buff[buff_size];
    fgets(buff, (buff_size-1), pFile);
    int val;
    sscanf(buff, "%d", &val);

    fclose(pFile);

    if(val == 1) {
        printf("Motion Detected!\n");
    } else {
        printf("No Motion Detected.\n");
    }
}
```

5. Useful References

1. Explanation on how Passive InfraRed Sensor works. https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/how-pirs-work
2. BeagleBone Green's Basic Information and Mappings. http://wiki.seeedstudio.com/BeagleBone_Green/
3. Introduction to Grove Connectors. http://www.switchdoc.com/2016/02/tutorial-intro-to-grove-connectors-for-arduinoraspberry-pi-projects/
4. Documentation on Grove PIR Motion Sensor. http://wiki.seeedstudio.com/Grove-PIR_Motion_Sensor/
5. BeagleBone Black P9 Header Table. http://www.cs.sfu.ca/CourseCentral/433/bfraser/solutions/zen/BeagleboneBlackP9HeaderTable.pdf
6. BeagleBone Black P8 Header Table. http://www.cs.sfu.ca/CourseCentral/433/bfraser/solutions/zen/BeagleboneBlackP8HeaderTable.pdf

BBB Live Video Streaming to Display on HTML Guide

1. Wiring

Plug the camera USB-A side to the beaglebone board.

2. Video Streaming

Follow the Capturing and Streaming Webcam Video guide here: (https://www.cs.sfu.ca/CourseCentral/433/bfraser/other/2017-student-howtos/CapturingAndStreamingWebcamVideoOnBBG.pdf)

Add-on & Changes to the above guide:

1. Make changes to this line of code

```
/* Timeout. */
tv.tv_sec = 2;
tv.tv_usec = 0;
```

```
/* Timeout. */
tv.tv_sec = 5;
tv.tv_usec = 0;
```

2. Installing ffmepg

   # cd /etc/apt/

   # nano sources.list

   Add the following line to the end of the file

   deb http://ftp.debian.org/debian jessie-backports main

   # apt-get -y install ffmpeg

3. Always recompile the capture.c file after downloaded it to a new computer, otherwise will get “pipe 0” error

3. Living Streaming to A Webpage

1. Downdload nodeStream package from this link: https://github.com/geraldo/nodeStream

2. Open server.js file and change the parameters in NXServer to your own configuration (2nd line of the file). For example, we downloaded the package to '/home/april/cmpt433/public/v4l2_lib/', and we stream ffmpeg output to 192.168.7.1 at port 1234. Our changes will look like this:

```
//--
var nxserver = new NXServer('/home/april/cmpt433/public/v4l2_lib/nodeStream', 8080, '192.168.7.1', 1234);
```

3. Open a browser and enter the following url to view the live streaming video:

   localhost:8080/client

Troubleshoot:

- if see the error "cannot spawn ffmepg", make sure you install ffmpeg correctly (follow the above page)