DropBone IMU

From Team4774

Contents

Outline

DropBone IMU is Team 4774's software interface to the MPU 6050. It runs on a BeagleBone Black (though it can run on any Linux computer with access to an MPU 6050 though I^2C) and broadcasts packets to the rest of the attached network through a UDP broadcast server.

Building and Running

To build, you must be running the GNU C compiler, and use the make command. The output executable file is dropboneimu.

To execute it, simply run the dropboneimu file (if not root you will need to run with sudo; make sure your device has I^2C enabled).

dropboneimu has several options, which are available through the -h or -? flags:

- -i
  - Disable interrupt pin. Will make code run if not wired up with interrupt pin
- -v
  - Verbose mode. Print out yaw, pitch and roll as they are received
- -s
  - Silent mode. Do not print anything
- -b
  - No broadcasts. Stops the UDP server from broadcasting information received from the MPU over UDP
- -h, -?
  - Display usage message, then exit

Architecture

Based on a Beaglebone Black with a GY-521 IMU breakout board.

The GY-521 has an MPU-6050 accelerometer/gyroscope chip on board.

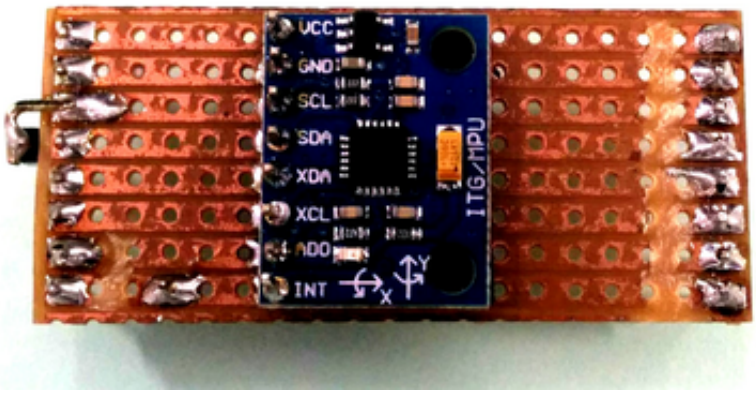
Communication

BBB to MPU-6050 is conducted over the I2C bus.

Information from the MPU-6050 is then sent out to the rest of the robot network via UDP.

Circuit

The GY-521 is wired into a custom veroboard to connect to the BBB I2C bus. It connects to the primary BBB I2C bus (/dev/i2c-1).



GY-521	BBB	BBB pin number	BBB GPIO number
VCC	GPIO_50	14	50
GND	GPIO_51	16	51
SCL	I2C2_SCL	19	N/A
SDA	I2C2_SDA	20	N/A
XDA	-	-	-
XCL	-	-	-
AD0	-	-	-
INT	GPIO_14	26	14

The circuit diagram is shown below:

GPIO

In order to provide power and interrupts for the MPU-6050 the following GPIO pins need to be set.

GPIO 14 (to in), GPIO 50 (to out) and GPIO 51 (to out).

To do this, the following shell commands can be issued on the BBB:

```
# gpio(1) 28 and 29 for power - 1*32+28 = 50
echo 50 > /sys/class/gpio/export
echo 51 > /sys/class/gpio/export
# gpio(0) 14 for interrupt - 0*32+14 = 14
echo 14 > /sys/class/gpio/export

echo 'out' > /sys/class/gpio/gpio50/direction
echo 'out' > /sys/class/gpio/gpio51/direction
echo 'in' > /sys/class/gpio/gpio14/direction

# create interrupts on falling edge of gpio14
echo 'falling' > /sys/class/gpio/gpio14/edge

echo 0 > /sys/class/gpio/gpio51/value
echo 1 > /sys/class/gpio/gpio50/value
```

IMU

A quaternion, gyroscope, and accelerometer values are retrieved from the IMU by the BBB as longs. They are converted to floats prior to broadcast using a scale factor.

drop_bone_imu.c waits until the values being received from the IMU have stabilized (indicating the end of calibration) before it captures a quaternion to be the zero point for the quaternions coming off the IMU and starts broadcasting over UDP.

The non-zero values received from the IMU after calibration are corrected for roll, pitch, yaw and the quaternion by finding the offset after calibration and multiplying the offset by the quaternion with the function q_multiply().

Yaw, Pitch and Roll values are converted from a quaternion using the euler() function.

Data Broadcast

Data packets are broadcast by the BBB to the rest of the network using UDP.

They are broadcast on port 4774. (No way!)

The form of the packets is:

Yaw, Pitch, Roll, Gyro X, Gyro Y, Gyro Z, Accel X, Accel Y, Accel Z, Quaternion W, Quaternion X, Quaternion Y, Quaternion Z

X axis is towards ethernet port, Y axis is towards 5V power jack, and Z axis is up. Essentially orientate the BBB so you can read the main writing.

To broadcast a packet, call the send_udp() function, as defined in udp.h.

int udp_send(float *data, unsigned int length) ==> returns number of bytes broadcast, and -1 if the broadcast fails.

udp_send() will automatically initialise a static socket if one does not already exist.

It additionally loops through and broadcasts to all interfaces on the BBB.

Each float in data is put into a string and is given 12 characters width maximum. Fields are comma delimited, and packets from the IMU are broadcast in SI units.

Source Code

Code to run the DropBone IMU is available at the Team 4774 github repo:
https://github.com/thedropbears/DropBoneImu

Retrieved from "http://thedropbears.org.au/wiki/index.php?title=DropBone_IMU&oldid=167"

- This page was last modified on 23 November 2014, at 23:30.
- This page has been accessed 1,520 times.