The Embedded Gyrometer
“The Need for Speed”

Objective:

Embedded system design focuses on gathering useful data, processing that data, and providing a useful representation of information. In the past decade we have seen an explosion of wearable health devices ranging from heart rate sensors to step counters to distance trackers. These devices are designed to help us meet our fitness goals and help us keep in good physical shape. The objective of this semester’s embedded challenge is to build a wearable speedometer which can calculate velocity by measuring angular velocities available from our built-in gyroscope (L3GD20) - without a GPS. Our gyroscope is able to measure 3-axis angular velocity. Strategically placing the sensor on the legs or feet can capture the angular velocities and with a bit of processing can convert those angular velocities to linear forward velocity and calculate distance traveled (using only a gyro!)

Required Parts:

STM32F429 Discovery Board with built in gyroscope.

Description of method:

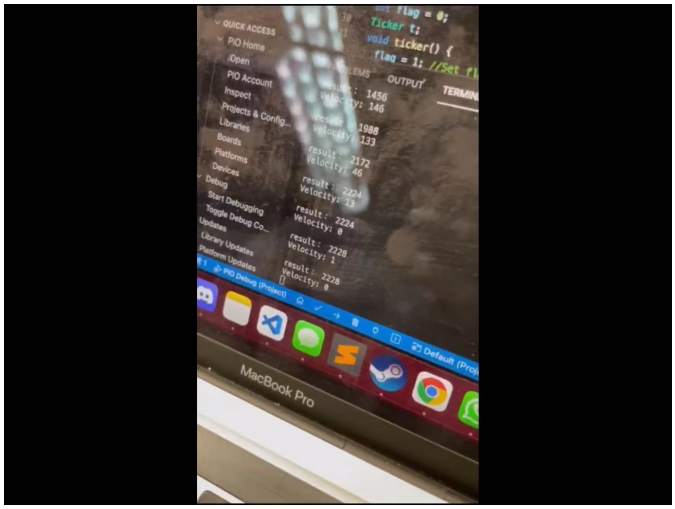
Workflow & Math:

First do chip select to pull down so that we can enable the gyro.
Inside the while loop, get the most significant bit and least significant bit of value gained from the gyroscope, and store them into the buffer.
Then combine the significant bit and least significant bit together to form our actual data, and store them into an array.
Then implement the ticker, calculate the angular rate by doing data of **x-axis * pi / 180**.
Then calculate absolute velocity based on the **angular * stride**.
Store the velocity to the FIFO buffer.

Get the velocity from the FIFO buffer.
Get an average velocity of every 4 samples.
Now use **velocity * the time**, and then keep accumulating this result to get distance. **Distance += velocity * time * 2**. 2 is the factor found by running experiments.
Now we get the final distance.

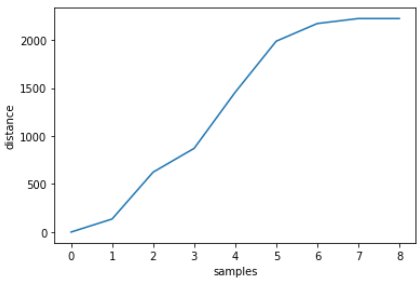
Data Storage:
Store the data into a FIFO buffer, use a locker to avoid read/write hazard.

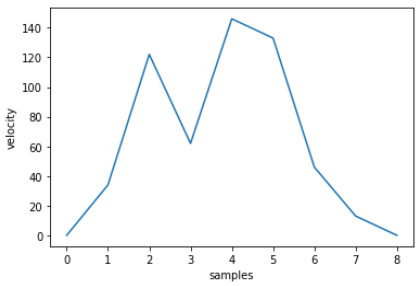
Test Result:



It is one of the tests. The plot of the test is shown below. I traveled 73 feet.
The result I got from my code is 2224 centimeters, which is 72.9 feet.

Plots:





The velocity dropped dramatically at some point, that is because I turned around

Code:

```
#include <mbed.h>
#include <math.h>

#define L3GD20_WHO_AM_I      0x0F
#define L3GD20_CTRL_REG1     0x20
#define L3GD20_REFERENCE     0x25
#define L3GD20_CTRL_REG6     0x40
#define L3GD20_OUT_X_L       0x28
#define fifoSize             40
#define ERROR_FULL           0Xff


SPI spi(PF_9, PF_8, PF_7); // mosi, miso, sclk
DigitalOut cs(PC_1); // use pin 14 as chip select

// variables to calculate the distance
int mag = 0;          // w
int distance = 0;     // initialize distence
float stride = 0.6;  //stride length
int counter = 0;      // counter for averaging the velocity
int v = 0;
int vSum = 0;         // sum of velocity for averaging the velocity

// variabels to store the data gained from gyrosope
float x, y, z;
char buffer[6];
signed short data[3];

// fifo buffer to store data
int fifoBuffer[fifoSize] = {};
int head = 0;
int tail = 0;
volatile int fifoLock = 0;

// sampling rate
float sampling_rate = 0.55;
float timeX = 20;

PwmOut Pout(PB_0);
```

```
// ticker
int flag = 0;
Ticker t;
void ticker() {
 flag = 1; //Set flag
}

// fifo write
void fifoWriter() {
 if (fifoLock == 1) while(1);
 else {
   fifoLock = 1; // lock
   head = (head + 1) % fifoSize;
   fifoBuffer[head] = v;
   fifoLock = 0; // release the lock
 }
 }
// fifo read
int fifoRead() {
  if (fifoLock == 1) while(1);
  else {
   fifoLock = 1; // lock
   tail = (tail + 1) % fifoSize;
   fifoLock = 0; // release the lock
   return fifoBuffer[tail];
  }
}



int main() {
 /* polling
 *the data line is active when SCK goes to low
 *and the data values are read when SCK goes to low
 */
 cs=1; // chip select
 spi.format(8,3);
 spi.frequency(1000000); // set up clock rate
 cs=0;
 spi.write(L3GD20_REFERENCE); // L3GD20_REFERENCE     0x25
 cs=1;
 cs=0;
 spi.write(L3GD20_CTRL_REG1); //L3GD20_CTRL_REG1     0x20
```

```
 spi.write(L3GD20_WHO_AM_I); // L3GD20_WHO_AM_I      0x0F
 cs=1;
 Pout.period(1);
 Pout.write(0.2);

 signed distance = 0;

while (1) { // infinite loop
 wait_us(1000000);
 cs=0;
 // RW bit, MB, output address
 spi.write(0x80|L3GD20_CTRL_REG6|L3GD20_OUT_X_L); ;
 for (int i = 0;i<=5;i++) {
 buffer[i]=spi.write(0x00); // put data read from gyro to buffer
 }
 // put RW and MB together
 cs=1;
 data[0] = buffer[1]<<8 | buffer[0]; //x
 data[1] = buffer[3]<<8 | buffer[2]; //y
 data[2] = buffer[5]<<8 | buffer[4]; //z
  // implement ticker
 t.attach(&ticker, 500ms);
 if (flag == 1){
   for (int i = 0; i < timeX/sampling_rate; i++) {
     mag = data[0]*3.14/180;
   }
   flag = 0;
 }


 v = mag*stride;
 fifoWriter();

 counter++; // increment the counter
 //vSum += abs(v);
 vSum += abs(fifoRead());
 int avg = 0;
 /** get the average velocity everytime counter equals 4 **/
   if (counter % 4 == 0) {
     avg = vSum/4;
     distance += avg*2;
   int result = distance;
```

```
   // flush the buffer
   printf("\n\r");
   printf("result :  %d\n\r", result*2);
   printf("Velocity: %d\n\r",avg);
   vSum = 0;
   }

 }
}
```