/\*Begining of Auto generated code by Atmel studio \*/

#include <Arduino.h>

/\*End of auto generated code by Atmel studio \*/

#include <MPU9250.h>

#include <quaternionFilters.h>

//Beginning of Auto generated function prototypes by Atmel Studio

//End of Auto generated function prototypes by Atmel Studio

#define debugYaw false

#define imuDebugMode false

#define IMU1\_ADDRESS 0x68

MPU9250 IMU1; // Name IMU (on main board)

int intPin = 12;

int myLed = 13;

String receivedString = "";

const int rowLength = 2;

static int receivedValue = 0;

static int receivedArray[rowLength] = {};

static int checksum = 0;

void setup() {

Wire.begin();

Serial.begin(9600);

//Serial.print("started\n");

// Defining imu id

byte imu1\_id = IMU1.readByte(IMU1\_ADDRESS, WHO\_AM\_I\_MPU9250);

// Perform IMU Self Test for QA of factory compliance

IMU1.MPU9250SelfTest(IMU1.SelfTest, IMU1\_ADDRESS);

//printSelfTestResults(IMU1, IMU1\_ADDRESS, imuDebugMode);

IMU1.calibrateMPU9250(IMU1.gyroBias, IMU1.accelBias, IMU1\_ADDRESS);

//printCalibrationResults(IMU1, IMU1\_ADDRESS, imuDebugMode);

}

void loop() {

while (Serial.available() > 0) {

int idx = 0;

while (idx < rowLength) {

//Serial.print("Column Index: "); debugging

//Serial.println(idx);

int CharReceived = Serial.read();

//if(isDigit(CharReceived)){

if (1) {

receivedString += (char)CharReceived;

}

if (CharReceived == '|') {

Serial.println("Starting to read row.");

receivedString = "";

delay(20);

}

else if (CharReceived == ',') {

Serial.print("received: ");

Serial.println(receivedString);

receivedValue = receivedString.toInt();

Serial.print("value: ");

Serial.println(receivedValue);

receivedString = "";

receivedArray[idx] = receivedValue;

checksum += receivedValue;

idx++;

delay(20);

}

else if (CharReceived == ';') {

receivedValue = receivedString.toInt();

Serial.print("value: ");

Serial.println(receivedValue);

receivedString = "";

receivedArray[idx] = receivedValue;

if (checksum != receivedValue) {

Serial.println("PANIC NOW CHKSUM IS DIFFERENT YA DINGUS!");

}

Serial.print("Row received: ");

for (int printidx = 0; printidx < rowLength; printidx++) {

Serial.print(receivedArray[printidx]);

Serial.print(" ");

}

Serial.println("Proceeding to next row!");

idx++;

delay(20);

}

delay(50);

}

}

}

void printSelfTestResults(MPU9250 imu, byte imu\_address, bool debugMode) {

if (debugMode) {

Serial.print("For IMU located at device address "); Serial.print(imu\_address, HEX); Serial.println(":");

Serial.print("x-axis self test: acceleration trim within : ");

Serial.print(imu.SelfTest[0], 1); Serial.println("% of factory value");

Serial.print("y-axis self test: acceleration trim within : ");

Serial.print(imu.SelfTest[1], 1); Serial.println("% of factory value");

Serial.print("z-axis self test: acceleration trim within : ");

Serial.print(imu.SelfTest[2], 1); Serial.println("% of factory value");

Serial.print("x-axis self test: gyration trim within : ");

Serial.print(imu.SelfTest[3], 1); Serial.println("% of factory value");

Serial.print("y-axis self test: gyration trim within : ");

Serial.print(imu.SelfTest[4], 1); Serial.println("% of factory value");

Serial.print("z-axis self test: gyration trim within : ");

Serial.print(imu.SelfTest[5], 1); Serial.println("% of factory value");

}

}

void printCalibrationResults(MPU9250 imu, byte imu\_address, bool debugMode) {

if (debugMode) {

Serial.print("For IMU located at device address "); Serial.print(imu\_address, HEX); Serial.println(":");

Serial.println("Gyro biases:");

Serial.print("X: "); Serial.println(imu.gyroBias[1]);

Serial.print("Y: "); Serial.println(imu.gyroBias[2]);

Serial.print("Z: "); Serial.println(imu.gyroBias[3]);

Serial.println("Accel. biases:");

Serial.print("X: "); Serial.println(imu.accelBias[1]);

Serial.print("Y: "); Serial.println(imu.accelBias[2]);

Serial.print("Z: "); Serial.println(imu.accelBias[3]);

}

}

void plotAngles(float yaw, float pitch){

Serial.print(pitch);

Serial.print("\t");

Serial.println(yaw);

}

/\*

void imuRead(MPU9250 imu, byte imu\_address, bool debugMode, bool debugY) {

if (imu.readByte(imu\_address, INT\_STATUS) & 0x01)

{

imu.readAccelData(imu.accelCount, imu\_address);

imu.getAres();

imu.ax = (float)imu.accelCount[0] \* imu.aRes; // - accelBias[0];

imu.ay = (float)imu.accelCount[1] \* imu.aRes; // - accelBias[1];

imu.az = (float)imu.accelCount[2] \* imu.aRes; // - accelBias[2];

imu.readGyroData(imu.gyroCount, imu\_address);

imu.getGres();

imu.gx = (float)imu.gyroCount[0] \* imu.gRes;

imu.gy = (float)imu.gyroCount[1] \* imu.gRes;

imu.gz = (float)imu.gyroCount[2] \* imu.gRes;

imu.readMagData(imu.magCount); // Read the x/y/z adc values

imu.getMres();

imu.magbias[0] = +470.;

// User environmental x-axis correction in milliGauss TODO axis??

imu.magbias[1] = +120.;

// User environmental x-axis correction in milliGauss

imu.magbias[2] = +125.;

imu.mx = (float)imu.magCount[0] \* imu.mRes \* imu.magCalibration[0] -

imu.magbias[0];

imu.my = (float)imu.magCount[1] \* imu.mRes \* imu.magCalibration[1] -

imu.magbias[1];

imu.mz = (float)imu.magCount[2] \* imu.mRes \* imu.magCalibration[2] -

imu.magbias[2];

imu.updateTime();

// Update Quaternions

MahonyQuaternionUpdate(imu.ax, imu.ay, imu.az, imu.gx \* DEG\_TO\_RAD,

imu.gy \* DEG\_TO\_RAD, imu.gz \* DEG\_TO\_RAD, imu.my,

imu.mx, imu.mz, imu.deltat);

//Tait Bryan Angle Calculation

imu.yaw = atan2(2.0f \* (\*(getQ() + 1) \* \*(getQ() + 2) + \*getQ() \*

\*(getQ() + 3)), \*getQ() \* \*getQ() + \* (getQ() + 1) \* \*(getQ() + 1)

- \* (getQ() + 2) \* \*(getQ() + 2) - \* (getQ() + 3) \* \*(getQ() + 3));

imu.pitch = -asin(2.0f \* (\*(getQ() + 1) \* \*(getQ() + 3) - \*getQ() \*

\*(getQ() + 2)));

imu.roll = atan2(2.0f \* (\*getQ() \* \*(getQ() + 1) + \* (getQ() + 2) \*

\*(getQ() + 3)), \*getQ() \* \*getQ() - \* (getQ() + 1) \* \*(getQ() + 1)

- \* (getQ() + 2) \* \*(getQ() + 2) + \* (getQ() + 3) \* \*(getQ() + 3));

imu.pitch \*= RAD\_TO\_DEG;

imu.yaw \*= RAD\_TO\_DEG;

// Declination of San Luis Obispo (35°17'37"N 120°40'05"W) is

// 12° 36' E ± 0° 20' (or 12.5°) on 2018-12-01

// - http://www.ngdc.noaa.gov/geomag-web/#declination

imu.yaw -= 12.5;

imu.roll \*= RAD\_TO\_DEG;

printAnglesIMU(imu, imu\_address, debugMode);

imu.count = millis();

imu.sumCount = 0;

imu.sum = 0;

if(debugY){

printYaw(imu.yaw);

}

}

}

\*/

void printAnglesIMU(MPU9250 imu, byte imu\_address, bool debugMode) {

if (debugMode)

{

Serial.print("IMU Address 0x"); Serial.print(imu\_address, HEX); Serial.println(":");

Serial.print("Yaw, Pitch, Roll: ");

Serial.print(imu.yaw, 2);

Serial.print(", ");

Serial.print(imu.pitch, 2);

Serial.print(", ");

Serial.println(imu.roll, 2);

Serial.print("rate = ");

Serial.print((float)imu.sumCount / imu.sum, 2);

Serial.println(" Hz");

}

}

void printYaw(float yaw){

Serial.println(yaw,2);

}

void plotQuats(){

Serial.print(\*(getQ()));Serial.print("\t");

Serial.print(\*(getQ()+1));Serial.print("\t");

Serial.print(\*(getQ()+2));Serial.print("\t");

Serial.println(\*(getQ()+3));

}