NAZARBAYEV UNIVERSITY
SCHOOL OF SCIENCE AND TECHNOLOGY

Project Assignment #3

CSCI502 – HARDWARE/SOFTWARE CO-DESIGN

Spring Semester 2019

IMU SENSOR INTERFACING WITH BEAGLEBONE BLACK

DUE TIME AND DATE

- **Class presentation and report uploaded to Moodle on Wednesday 6 March**

LEVEL OF COLLABORATION ALLOWED

You will be working in groups of two students

DELIVERABLES REQUIRED

You group is required to prepare and submit to Moodle system an MS Word report with Linux terminal screenshots for all major tasks and steps from this assignment. In addition you will present your assignment in class.

REFERENCES

Derek Molloy,"Exploring BeagleBone. Tools and Techniques for Building with Embedded Linux. 2nd edition", Wiley, 2018 (available in Moodle)
Derek Molloy,"Exploring BeagleBone. Tools and Techniques for Building with Embedded Linux. 1nd edition", Wiley, 2015 (available in library)

INTRODUCTION

Do you want to detect the collision of a robot or to build a self-balancing robot? Or you have plans to build a drone? All these robots need sensors such as accelerometers, gyroscopes, magnetometers and IMUs. These small components are embedded into the robot to generate information about the different mechanical phenomenon such as acceleration, vibration, tilt, orientation in space, angular velocity, pitch or rotation.

These types of sensors with capabilities to measure the acceleration, tilt, angular velocity, and other mechanical phenomena are used in different devices including smartphones, gaming consoles or toys.

If an **accelerometer sensor** is designed to measure the acceleration and tilt, the gyroscopic sensor measures the angular velocity and orientation. The IMU sensor is a special one designed to combine the features of an accelerometer and gyroscope in order to display complete information about the acceleration, position, orientation, speed, etc. for a robot.

The accelerometer sensor measure acceleration in two different units including meters per second squared, or when the acceleration felt like a weight, in G-forces. The advantages of the accelerometer sensor include a high accuracy in applications with noises, as well the acceleration measurement down to zero Hertz. The biggest disadvantage of this sensor is the limited high frequency where the sensor works.

The **gyroscope sensor** is inexpensive and measures in degrees per second or revolutions per second the angular velocity. It's frequently used in robotic applications to measure the balancing and send corrections to motors or drones to stabilize the flight.

The **magnetometer sensor** are finding increasing use as compasses in consumer devices such as mobile phones and tablet computers

The **IMU or Inertial Measurement Unit** is a sensor that hosts the above three types of sensors.

ASSIGNMENT DETAILS

Read Chapter 1 of the "Exploring BeagleBone" textbook (available in Moodle)
GET FAMILIAR WITH THE SAFETY RULES ON PAGE 27 OF THE TEXTBOOK (PAGE 21 IN 1ST EDITION OF THE TEXTBOOK)

TASK 1. IMU INTERFACING (30% OF THE TOTAL GRADE)

Your group will interface an inertial measurement unit (IMU) to the BBB board using I^2C synchronous communication bus.

The Pololu MinIMU-9v3 www.pololu.com/product/2468 is an inertial measurement unit (IMU) that packs an L3GD20H 3-axis gyro and an LSM303D 3-axis accelerometer and 3-axis magnetometer onto a tiny 0.8″ × 0.5″ board. An I²C interface accesses nine independent rotation, acceleration, and magnetic measurements that can be used to calculate the sensor's absolute 3D orientation. The MinIMU-9 v3 board includes a voltage regulator and a level-shifting circuit that allows operation from 2.5 to 5.5 V. Some groups will receive the Pololu AltIMU-10v4 unit (https://www.pololu.com/product/2470) that is exactly the same as the MinIMU-9v3 but contains an additional LPS331AP digital barometer sensor for altitude measurements (is not used in this assignment).

Some groups may receive the Pololu MinIMU-9 v5 IMU (https://www.pololu.com/product/2738) that is an updated version of the MinIMU-9 v3 but with LSM6DS33 3-axis gyro and 3-axis accelerometer and an LIS3MDL 3-axis magnetometer. The datasheets of these sensors will be provided. In this case the control and data register addresses and settings used for IMU programming may/will differ. However, you can still use the instructions bellow as guidelines.

I²C Communication

The L3GD20H's gyro and the LSM303D's accelerometer and magnetometer can be queried and configured through the I²C bus. A detailed explanation of the protocols used by each device can be found in the L3GD20H and the LSM303D datasheets (available in Moodle).

The L3GD20H and LSM303D each have separate slave addresses on the I²C bus. The following table shows the slave addresses of the sensors:

Sensor	Slave Address (default)
L3GD20H (gyro)	1101011b (0x6B)
LSM303D (accelerometer and magnetometer)	0011101b (0x1D)

To connect the sensor to the BBB:

1. Study principles of general purpose input/output (GPIO) interfacing to BBB in **Chapter 6 of the textbook (pages 247- 250)** (1st edition: pages 201-203)
2. Study relevant section of **Chapter 8 of the textbook (pages 341 - 360)** (1st edition: pages 275 - 290) and connect the IMU sensor to the BBB board similar way as in Figure 8-2 (p. 349 using the following connections:

   IMU pin / BBB input/output

   SCL – P9_19
   SDA – P9_20
   GND – P9_1
   VDD – P9_3
   VIN – disconnected.
3. Follow the textbook and test i2c-tools: **i2cdetect**, **i2cdump** and **12cget**, in the BBB terminal.
4. Observe that the IMU gyro, accelerometer and magnetometer sensors are all off by default and do not provide orientation measurements. Use sensor datasheets to learn correct data register addresses, e.g. 8-bit registers **OUT_X_L (0x28)** and **OUT_X_H (0x29)** provide 16-bit X-axis gyro measurement data (L3GD20H sensor). The register reading commands examples are **i2cget -y 2 0x6B 0x28** and **i2cget -y 2 0x6B 0x29**. Answer in the report why the value **2** is used in the above commands?
5. In order to turn the IMU on, the sensor configuration registers have to be set to the normal mode of operation. This can be done by sending the following control worlds to the corresponding configuration registers as given in the table below using **i2cset** command, e.g. **i2cset -y 2 0x6b 0x20 0x0F**

L3GD20H		**LSM303D**	
Register (address)	Control word	Register	Control word
CTRL1 (0x20)	0x5F	CTRL1	0x57
CTRL2 (0x21)	0x20	CTRL2	0x00
		CTRL5	0x64
		CTRL6	0x20
		CTRL7	0x00

6. Learn the meaning of the control words using sensor datasheets and modify if necessary the corresponding accelerometer, magnetometer and gyroscope sensor control registers and set the sensors to following settings:

	L3GD20H	**LSM303D**	
	Gyroscope	**Accelerometer**	**Magnetometer**
Data output frequency	50Hz	50Hz	6.25Hz
Range of output frequency	± 245 deg/sec	$\pm 2g\ (2\ x\ 9.81 m/s^2)$	$\pm 4\ gauss\ (2x10^{-4} Tesla)$

Provide your modified control words in the report.

7. Use the program example similar to Listing 8-1 on page 356 of the textbook (for more details please follow the 1st edition of the textbook: page 286) (the code is available in Moodle) and write a program for setting the configuration registers and reading measurement data from x, y and z axes of all of the IMU onboard sensors. Note that the 16-bit data word for each measurement axis is obtained by combining readings from two 8-bit data registers (low and high).
8. Using the IMU sensor datasheets you are required to prepare a register map containing information about the IMU configuration and data registers. Please be able to explain the meaning of the sensor control words during the class presentation.

TASK 2. IMU SIGNAL PROCESSING (25% OF THE TOTAL GRADE)

1. Study, download, and implement an open-source IMU sensor measurement fusion algorithm in C code developed by Mr. Sebastian Madgwick from http://www.x-io.co.uk/open-source-imu-and-ahrs-algorithms/
   The algorithm outputs orientation estimations in the quaternion form. Study the self-study lecture on quaternions in Moodle. More information about quaternions can be found in Internet, for example at http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-17-quaternions/
2. Study the Magwick IMU signal fusion algorithm carefully and define in what physical values the algorithm accepts the input sensor data. In your IMU sensor read programs convert the sensor raw measurements to physical values using scaling factors defined based on the maximum measurements values and the IMU sensor settings set above. Write your calculations in the report.
3. Interface your IMU sensor measurement data program with the algorithm code and obtain orientation estimations in the terminal window when running on the BBB.

TASK 3. IMU GUI DEVELOPMENT (35% OF THE TOTAL GRADE)

In this task you will develop a client/server application to visualize an IMU sensor orientation on your laptop/PC. The BBB with connected IMU sensor acts as a server and sends processed orientation data to your laptop/PC (client). The client runs a Qt GUI for visualizing the IMU sensor 3D orientation. Please implement the following:

1. Study the principles of Internet of Things and use of BeagleBone board as a web server in **Chapter 11 of the textbook** (1st edition: chapter 10) and focus on the **C++ Client/Server section on pages 545-548 of the textbook** (1st edition: pages 412-415).
2. Study the Qt GUI application development in Chapter 13 of the textbook (1st edition: chapter 11)**.** In particular, focus on the **Remote UI Application Development section (Fat-Client Qt GUI Application subsection) on pages 625 – 641 of the textbook** (1st edition: pages 655-462).
   Examples program codes from Chapters 11 and 13 can be downloaded from the textbook Github page.
3. **On the server side (BBB)** please implement a multithreaded application using PThreads as follows:
   - Thread 1 reads the IMU raw sensor measurement data, converts it to physical values as explained in Steps 1 and 2, and passes it to Thread 2;
   - Thread 2 runs the open-source Magwick IMU sensor measurement fusion algorithm and sends the quaternion estimates to Thread 3; The open-source Magwick IMU sensor measurement fusion algorithm, that accepts the IMU data from Thread 1, and passes the quaternion estimates to Thread 3, the server socket communication, for reading by the client side. **Please make the quaternion component normalization before sending the data.**
   - Thread 3 implements the server socket communication and sends the data to the socket for reading by the client side.

   Use shared variables (protected by mutexes if needed) to pass the data from one thread to another. The textbook describes the use of the Apache web server embedded into the BBB Debian image. You may use alternative socket communication servers such as NodeJS, etc. if you prefer.
4. On the **client side (your laptop/PC)** you will implement GUI application in the Qt Creator which handles:
   - the client socket communication. It reads the data.
   - runs the data processing and visualizes IMU orientation.

   Please develop the IMU sensor orientation visualization in the Qt GUI using the provided in Moodle **test-opengl** framework based on OpenGL and QTthread classes. You may use alternative visualization tools, e.g. QtCanvas3D or others, if preferable.

Note that the provided programs may differ from the descriptions in the textbook and you also may encounter problems with running them properly. In this case try to cange the project settings for a different versions of Qt for compilation or use the example codes as a template for your own project programs.

GRADING CRITERIA

Demonstration and grading of your working BBB&IMU setups and program solutions (with questions/answers) will be conducted during the class lab session on Wednesday 6 March. The total grade for all implemented assignments as described will constitute 90%.

The bonus 10% of the grade will be awarded to the groups showing outstanding coding experience in the form of modifications/added complexity of Task 3, project report completeness and quality, etc.

Please prepare a detailed report with program code and submit it to the project folder in Moodle by the end of Wednesday 6 March.

This project evaluation will be done using individual grading depending on the level of participation and understanding of the project assignments.

Late submission penalty – 10% per day