eYSIP2016

Robot State Collector

Amanpreet Singh
Amit Raushan
Shubham Gupta
Duration of Internship: 10/06/2016 – 24/07/2016

Contents

Robot State Collector

1.1 Abstract

This project aims at developing a State Collection Code which is easy to merge with any code and doesn't interfere with the code already present in the Robot. Our code should be capable of sending the sensor values collected periodically to the GUI developed for this specific purpose which would be able to encrypt the collected data and send it to the e-Yantra servers.

This would be helpful in re-creating runs ,using the code and the data collected by us, virtually.

The main tasks involved in the project are :

1. Collecting timed data about the state of the robot(sensors etc.).
2. Finding a way to efficiently storing data on the robot for a single run of the robot.
3. Developing a GUI that picks up the data from the robot,encrypts it and send it to e-yantra servers.

1.2 Completion status

1. We have been able to collect the data on Firebird V after a particular time period (currently set as 0.5s).

2. To avoid storing the data on the volatile memory on any robot we are using a X-Bee module which directly sends the data to the GUI wirelessly and by doing so we even save space on the robot's memory.

3. We have developed a GUI to read the incoming data from the Robot , encrypt the data and send it back to the server. The GUI is also capable of detecting is any changes were made to the file before sending it to the server.

4. We have also made a Robot using the TIVA board so as to check that our code and GUI are not restricted to only Firebird V.

5. We have been able to perform state collection on the newly developed robot.

1.3 Hardware parts used

- Hardware used with FireBird V :

Figure 1.1: Firebird V with Xbee module

1. X-Bee Module.
   Datasheet

Figure 1.2: xbee module

2. X-Bee Module and adapter to receive data on a Laptop/PC.

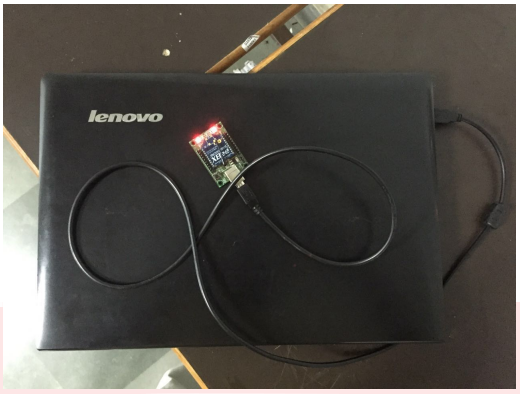

Figure 1.3: Xbee with adapter connected to laptop

- Hardware used to build a Robot using TIVA board (TM4C123GXL)
  Datasheet
  Pheripheral Driver Library
  User's Guide

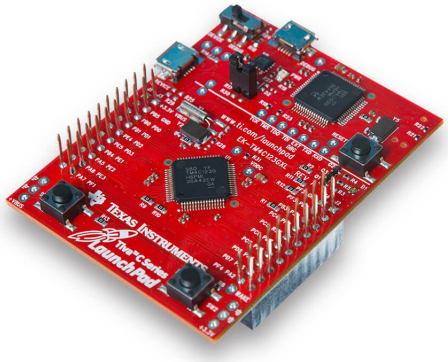

Figure 1.4: Tiva C Series TM4C123G LaunchPad

1. 2x DG02S Mini DC Gear motor.
   Datasheet

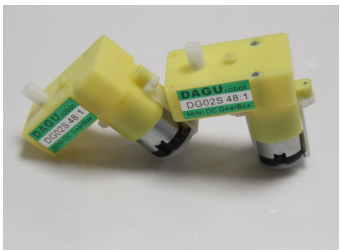

Figure 1.5: DC Geared motor

2. 2x Wheel - 65mm.

Figure 1.6: Wheel

3. L293D - Motor Driver.
   Datasheet

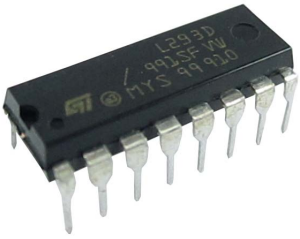

Figure 1.7: Motor Driver IC

4. 16x2 LCD.
   Datasheet

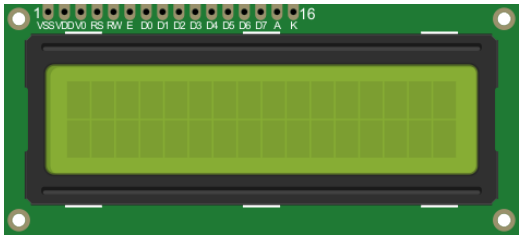

Figure 1.8: LCD 16x2 Display

5. White Line Sensor Module.
   Manual

Figure 1.9: 3 Channel Whiteline Sensors

6. SHARP 0A41SK sensor.
   Datasheet

Figure 1.10: SHARP Sensor

7. Caster Wheel.

Figure 1.11: Caster Wheel

8. LM3237 - Voltage Regulator.
   Datasheet

Figure 1.12: LM3237

9. Heat Sink.

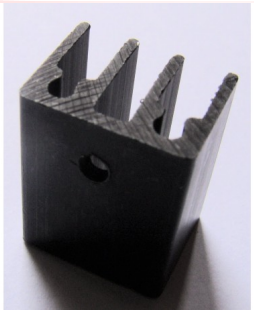

Figure 1.13: Heat Sink

10. 10 micro farad Electrolyte Capacitor.

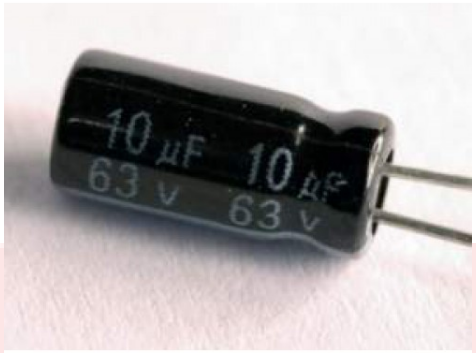

Figure 1.14: Capacitor

11. Multi-purpose PCB board.

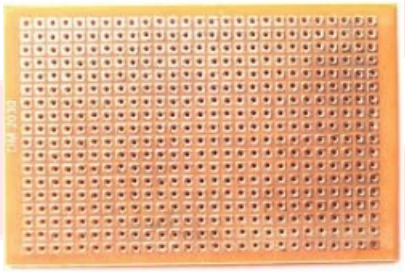

Figure 1.15: PCB Board

12. 12V Rechargeable Battery.

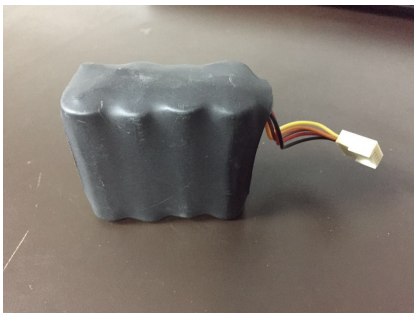

Figure 1.16: 12V Rechargeable Battery

13. 20 Pin Planar Cable.

Figure 1.17: 20 pin Connector wire

14. Female bug strip.

Figure 1.18: Female bug strip

15. Male bug strip.

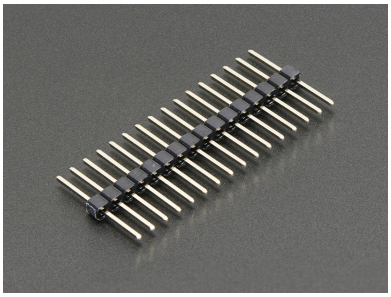

Figure 1.19: Male bug strip

16. Male to Female Jumper Wires.

Figure 1.20: Jumper Wires

17. Plastic Chassis.

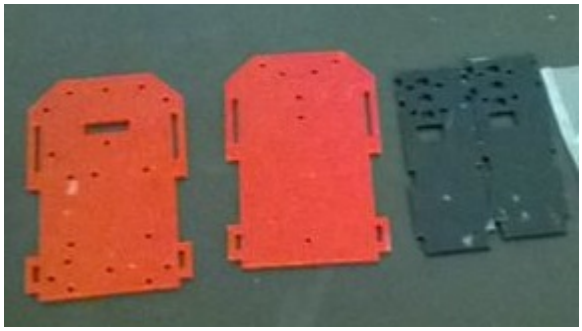

Figure 1.21: Chassis

- Hardware used with TIVA board Robot:

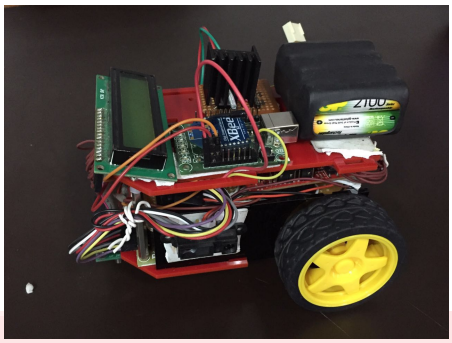

Figure 1.22: Robot

1. X-Bee Module.

Figure 1.23: xbee module

2. X-Bee Module and adapter to receive data on a Laptop/PC.

Figure 1.24: Xbee with adapter connected to laptop

1.4 Software used

- Atmel Studio 6.0
  Download Here
- XCTU-NG
  Download link
- Serial Terminal
  Download link
- Code Composer Studio v6.1.3
  Download link
- NetBeans IDE 8.1
  Download link
- AVR bootloader
  Download link

1.5 Assembly of hardware

1.5.1 Firebird V Robot

Xbee module is attached with Firebird V robot to collect it's state.

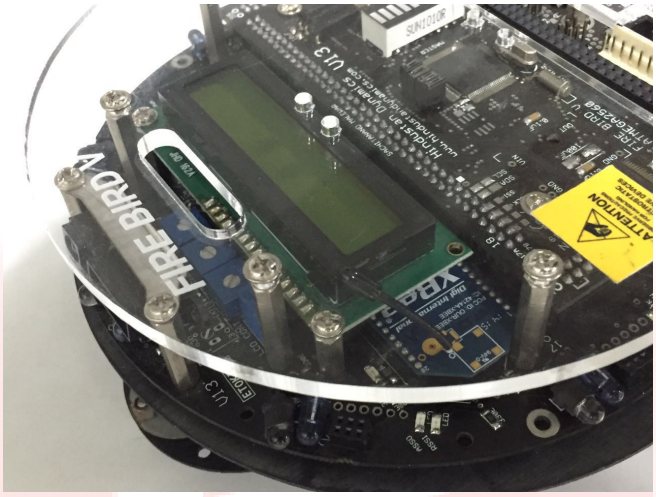

Figure 1.25: Xbee connected with firebird V

1.5.2 Tiva robot

Connections of PORT pins of Tiva board with different components:

1. Interfacing of white line sensor
    - PE1 - LEFT SENSOR
    - PE2 - CENTER SENSOR
    - PE3 - RIGHT SENSOR
    - GND - GND
    - VCC - VCC
2. Interfacing of Sharp sensor
    - PE0 - Sharp SENSOR
    - GND - GND
    - VCC - VCC

3. Interfacing of Motor Driver IC
    - PE4 - PWM1
    - PA2 - IN1
    - PA3 - IN2
    - GND - GND
    - VCC - VCC
    - PE5 - PWM2
    - PA6 - IN3
    - PA7 - IN4
4. Interfacing of X-Bee module
    - PC4 - Rx
    - PC5 - Tx
    - GND - GND
    - VCC - VCC
5. Interfacing of Buzzer
    - PF0 - Signal
    - GND - GND
6. Interfacing of LCD
    - PB0 - DATA 0
    - PB1 - DATA 1
    - PB2 - DATA 2
    - PB3 - DATA 3
    - PB4 - DATA 4
    - PB5 - DATA 5
    - PB6 - DATA 6
    - PB7 - DATA 7
    - GND - GND
    - VCC - VCC
    - VEE - GND

- VCC - Led+
- GND - Led-
- PA4 - RS
- PA5 - RW
- PC6 - EN

Block Diagram

Basic connections different components of Tiva robot.

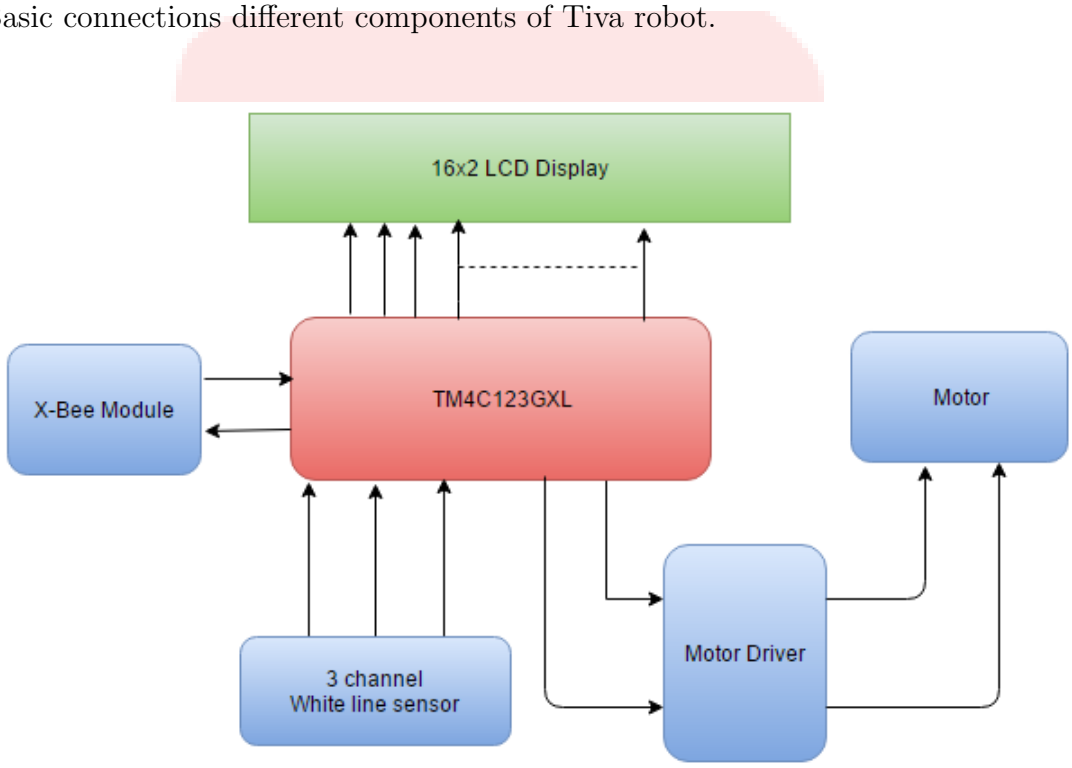

Figure 1.26: Block diagram of Tiva robot

1.5.3 Steps for assembling Tiva robot

1. All the components.

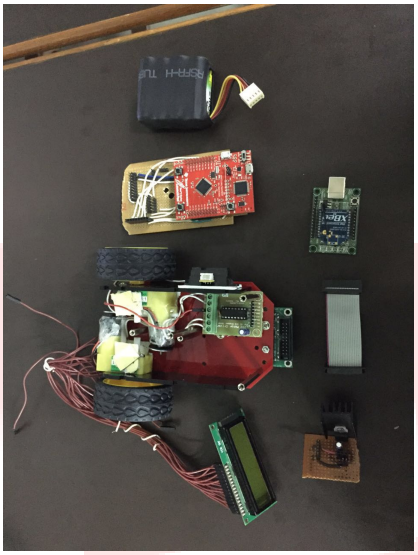

Figure 1.27: All the components used to build the bot.

2. Chassis,two dc geared motors & two wheels are assembled and the wheels are connected with the motor driver IC.

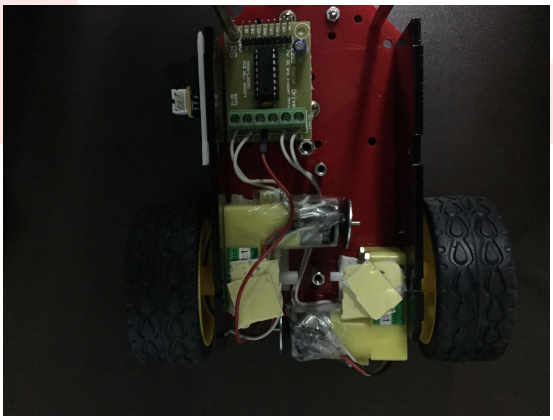

Figure 1.28: Motors and wheels attached to chassis

3. Whiteline sensors and Caster wheel is connected to the chassis.

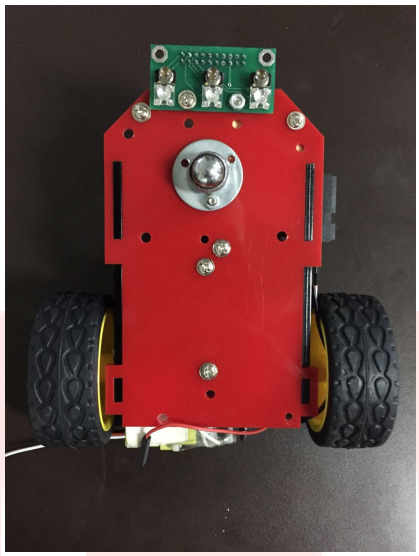

Figure 1.29: Caster Wheel and Whiteline Sensors attached to chassis

4. Sharp Sensor is attached to the chassis.

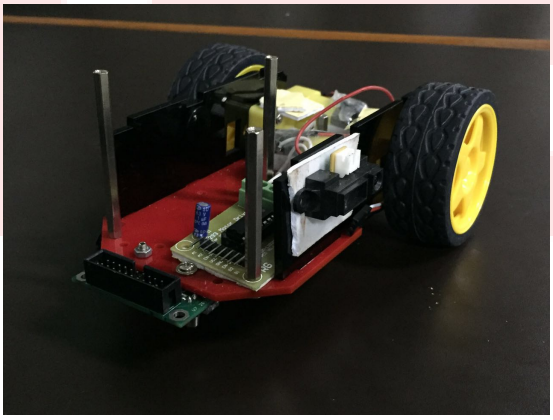

Figure 1.30: SHARP Sensor attached to chassis

5. Buzzer is attached to the PCB board designed on which TIVA board would fit and attached to the chassis.

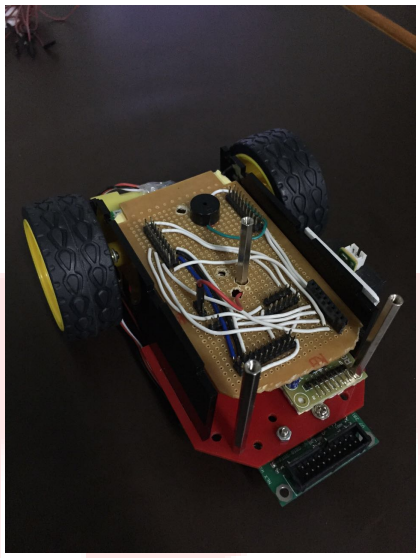

Figure 1.31: Buzzer which would be beneath the TIVA board

6. TIVA board (TM4C123GXL) is attached on the PCB board.

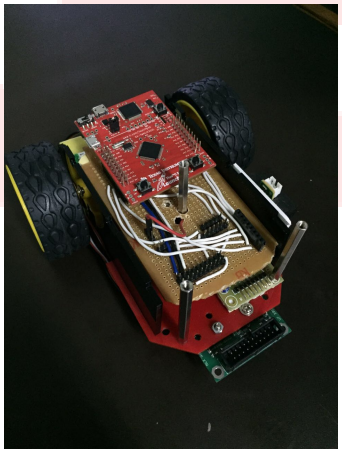

Figure 1.32: TIVA board fitted

7. Whiteline Sensors are connected using the 20 pin connector.

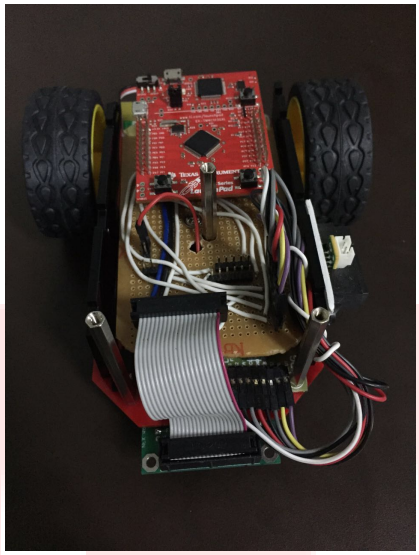

Figure 1.33: Whiteline Sensors are now connected to the TIVA board.

8. SHARP Sensor and Motor Driver IC are connected to the TIVA board.

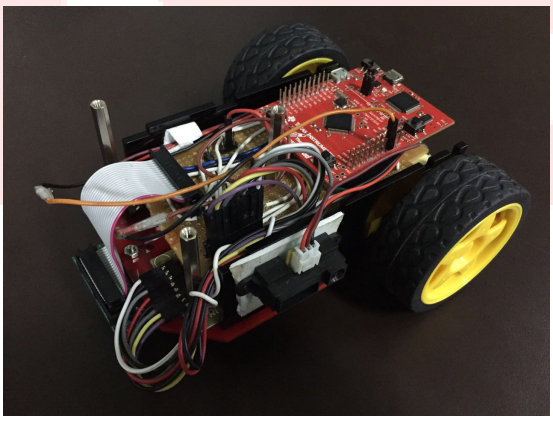

Figure 1.34: SHARP Sensor and Motor Driver IC are now connected to the TIVA board.

9. LCD is now connected to the TIVA board.

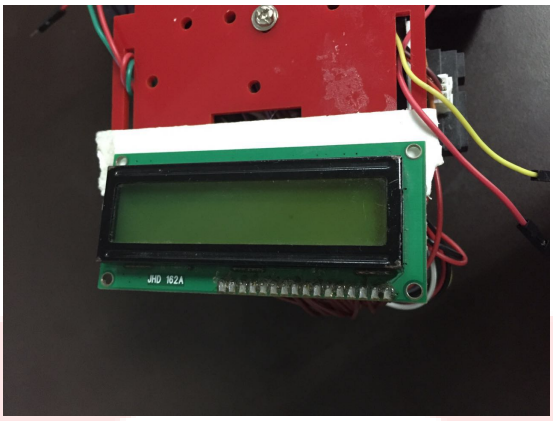

Figure 1.35: LCD after being attached to the bot.

10. X-Bee is now connected to the TIVA board.

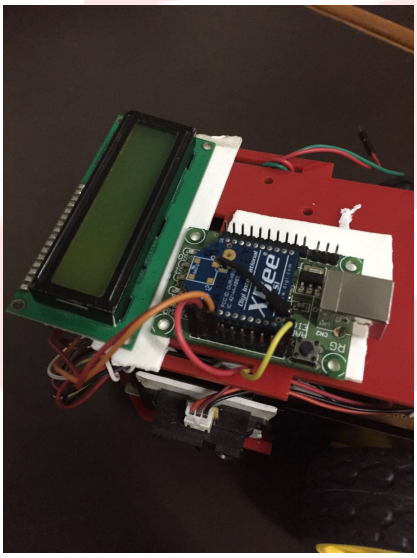

Figure 1.36: X-bee after being attached to the bot.

11. Voltade Regulator Circuit and Battery are now connected to the TIVA board.

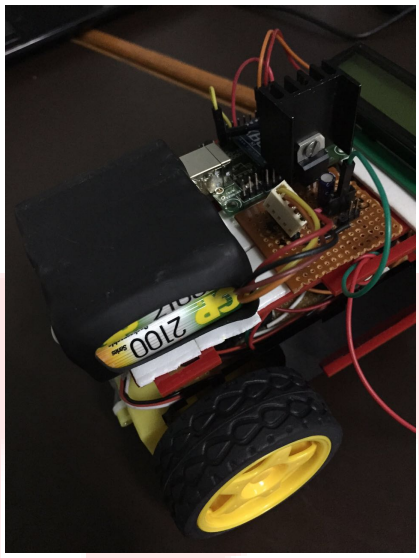

Figure 1.37: Power can now be supplied to the bot.

12. Finally the Bot is Ready for use.

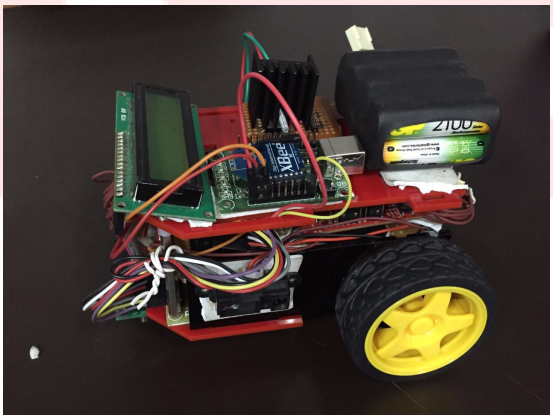

Figure 1.38: TIVA BOT.

1.6 Software and Code

Github link for the repository of code

1.6.1 Sate Collection code For Firebird V Robot

We are using Timer4 to generate an interrupt after 0.5s. Using this interrupt we can send the present values of the sensors to the GUI.
Our code is entirely inside the provided header file. A simple function call starts the State Collection.

1.6.2 Sate Collection code For TIVA Robot

We are using Timer0A to generate an interrupt after 0.5s. Using this interrupt we can send the present values of the sensors to the GUI.
Our code is entirely inside the provided header file. A simple function call starts the State Collection.

1.6.3 GUI

The GUI is capable of reading the incoming data on the X-Bee module. The data is stored and on a press of a button it gets encrypted using a symmetric key algorithm , the key of this symmetric key algorithm is encrypted using the public key provided to the user. At the end the encrypted data can be sent to the server if the user wishes.

1.7 Use and Demo

1.7.1 Steps for merging the State Collection Code with User's Code for Firebird V

1. include "common_header.h" in User' Code
2. include "state_collect.h" in User' Code
3. In main after initialising all the devices call the function "_init_devices()"

State Collection will now take place with the User's Code.

NOTE : The files included above should be present inside the same folder as that of the User's code.

1.7.2 Steps for merging the State Collection Code with User's Code for TIVA bot

1. include "common_header.h" in User' Code
2. include "collect.c in User' Code
3. include "sensor.c in User' Code
4. In main call the function "start_collection()" and "sensor_pin_config".

State Collection will now take place with the User's Code.

NOTE : The files included above should be present inside the same folder as that of the User's code.

1.7.3 Steps for using the GUI

1. Login with the credentials provided.
2. Make sure that the X-Bee adapter is connected to the laptop.
3. Click on the "Search for available COM ports " button.
4. From the Drop Down Menu select the COM port on which the X-Bee adapter is connected. Selecting the COM port would establish a connection with the X-Bee and the "Not connected" would change to "Connected".

5. After the run is complete Click on "End of Run" button to indicate the end of the robot's task.
6. To send the data to the server click on "Connect to the Server" button.
7. To disconnect from the COM port press the "Disconnect" button.

1.8 Future Work

1. LCD data can be collected.
2. Server Side Code can be improved such that it shouldn't have to be restarted after each run.
3. Position encoders can be configured with the TIVA Bot.

1.9 Bug report and Challenges

1.9.1 Bugs:

1. X-Bee doesn't run at baud rate of 115200 on Firebird V.
2. X-Bee doesn't run at baud rate of 9600 on TIVA bot.
3. If we change the frequency of the system clock on TIVA bot the X-Bee doesn't work.
4. Java code at server side stops after each run.
5. If we have to collect data of the position encoders then it has to be done with the interrupt used by the user and hence we have to change the user's code a bit.
6. The server side code hasn't been tested with multiple clients.
7. The server would give an error if the user logs in and closes the GUI without sending the data.
8. We don't know what would happen if connection between the client and the server is lost while transmitting the data.
9. There would be a loss of data if the user presses the Disconnect button while the GUI is collecting the State data.
10. On TIVA board no formula for conversion of digital values of SHARP sensor to actual distance has been used as TIVA board has a 12 bit ADC channel and all the formulas we found were meant for 8 bit ADC channels.

1.9.2 Challenges:

1. Storing the Data efficiently on the bot's memory – This was avoided by sending the data using a X-Bee module.
2. We were first transmitting the data as soon as it was read. This caused a problem as the earlier data was not sent and the new data arrived which caused them to overlap and hence resulted in loss of information – This was solved by waiting for the previous data to be sent first and then sending the new data.

3. Encryption without using any padding as the data size was not a multiple of 16 bytes and hence the AES encryption algorithm would fail – This was solved by using a padding which ensures that the data is always a multiple of 16 bytes.

4. Sending the data in a integer format using X-Bee module – This was solved by digitising the integer and then sending it.

5. Sending the data in a string form over Java Sockets – This was solved by converting the strings to byte arrays and then sending them.

6. Power when provided directly at VBUS of the TM4C123GXL it would cause the robot to restart – This problem disappeared after 2 days.

7. Motor Driver IC would not run from the 5V coming from the TIVA board – This problem disappeared during testing.

8. The PIN PF0 was locked and we were unaware of this fact and tried using this port like any other pin – After a lot of research on the web we found how to unlock the PINS.

Bibliography

yet to be done. :P