CS684 Documentation

CS684 – 2010 Project

Project: Controlling Firebird V using an Android based phone via Bluetooth

Project Team number: 7

Team members (Name and roll numbers):

Jatin Kanzaria	*09307919*
Rohan Shah	*09307050*
Jagbandhu	*09307603*
K. L. Srinivas	*09307051*

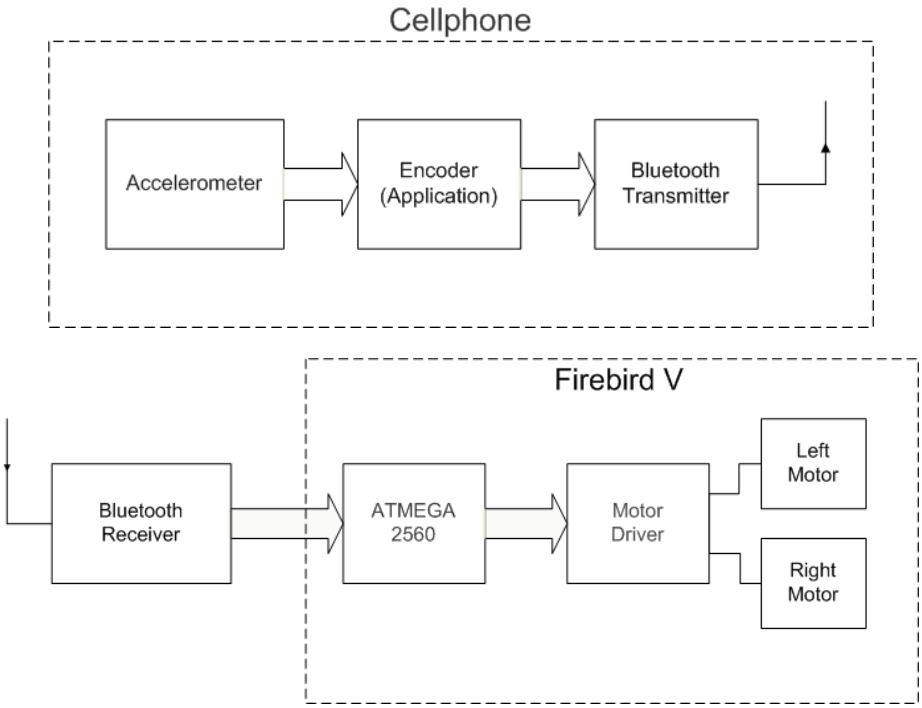
Controlling Firebird V using an Android based phone via Bluetooth

Project Title: Controlling Firebird V using an Android based phone via Bluetooth.
Project Team number: 7
Team members (Name and roll numbers):
Jatin Kanzaria 09307919
Rohan Shah 09307050
Jagbandhu 09307603
K. L. Srinivas 09307051

1. Introduction

1.1 Problem statement

To develop an application on the Android phone which will sense the tilt of the device and send this data over Bluetooth to FB5. Depending on the direction of the tilt the FB5 will move in the corresponding direction. Speed of the FB5 will be determined by the extent of tilt in given direction. FB5 currently does not have a Bluetooth support, so adding it will further enhance FB5 capabilities.



Above diagram shows the general block diagram of the present application. Whole problem can be divided into two major components.

User Interface on Smart Phone:

A Graphic User Interface is developed on a smart phone which helps user control the robot movement. This interface runs a backhand program which connects and disconnects with the Bluetooth module of FB5. During connected mode, application continuously reads the accelerometer reading to identify the direction of tilt of smart phone. When change in accelerometer reading crosses the threshold, direction of the tilt is transmitted via

Bluetooth transmitter. Bluetooth transmitter works in raw data mode wherein data characters are sent without any encoding. Data string is terminated with line feed and carriage return characters "#0D #0A".

Bluetooth Module on FB5:

An extension on FB5 is developed containing Bluetooth to serial converter and additional supporting hardware. Present logic of 5v is converted to 3v logic using resistor divider network. Supply voltage is generated using LM 317 regulator IC. On receiving data from Bluetooth to serial converter, direction and speed values are determined. Upon determining the direction and speed, necessary change in settings of PWM width and motor direction are done. Direction of motion is also displayed on LCD for convenience.

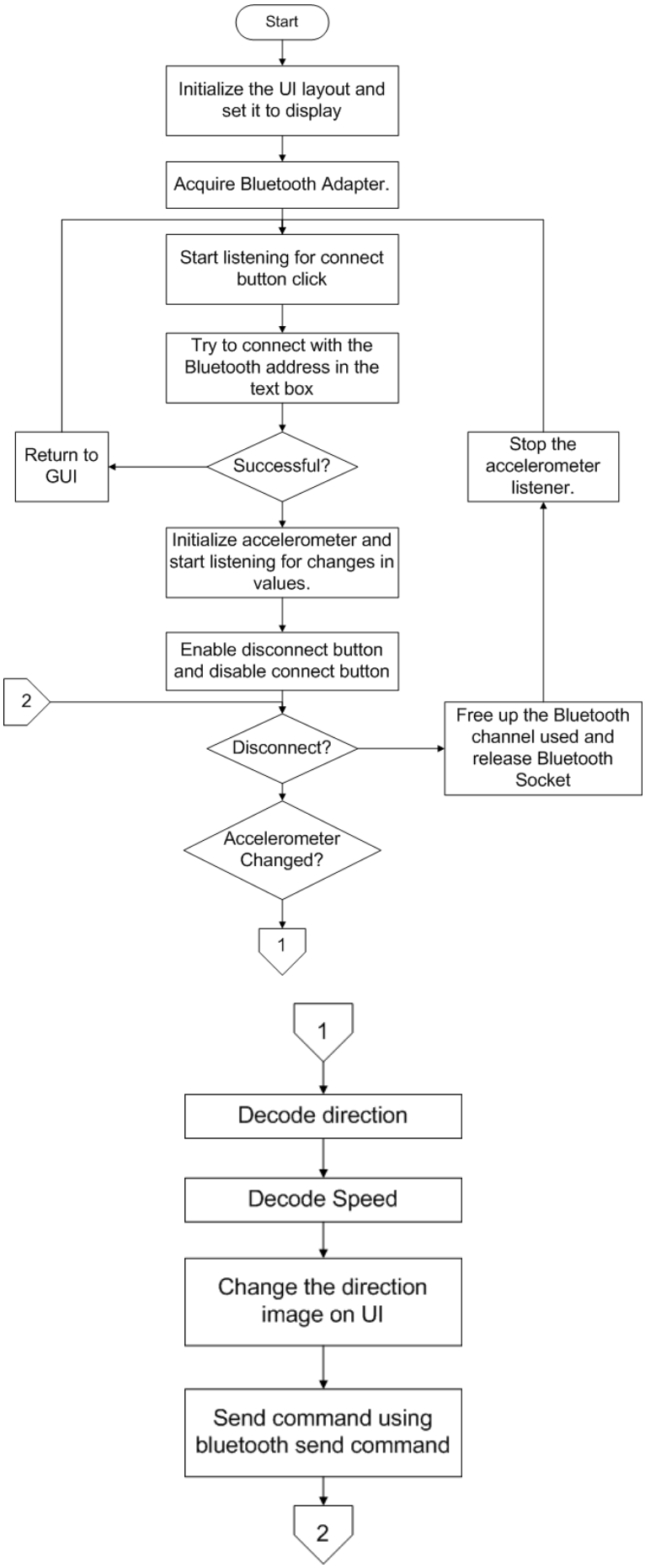
1.2 Requirements Specification

1.2.1 Interface accelerometer and Bluetooth module on Android phone.
Interface accelerometer to detect the tilt of mobile phone and decide the direction of motion. Send this command to FB5 through Bluetooth.

1.2.2 Interface Bluetooth module on FB5.
Bluetooth module works on 3V supply. Obtained 3V supply from 5V supply with minimal power wastage. Also convert 5V Logic to 3V connection with Bluetooth module.

1.3 System Flowchart

Below figure shows the flow diagram of application developed on android smart phone.


Start
Initialize the UI layout and set it to display
Acquire Bluetooth Adapter.
Start listening for connect button click
Try to connect with the Bluetooth address in the text box
Successful?
Return to GUI
Initialize accelerometer and start listening for changes in values.
Enable disconnect button and disable connect button
2
Disconnect?
Free up the Bluetooth channel used and release Bluetooth Socket
Stop the accelerometer listener.
Accelerometer Changed?
1
1
Decode direction
Decode Speed
Change the direction image on UI
Send command using bluetooth send command
2

1.4 Assumptions and limitations

1.4.1 Range of FB5 is limited by range of Bluetooth modules on mobile as well as Bluetooth to serial converter.

1.4.2 Only one FB5 can be controlled at a time.

1.4.3 Gestures other than tilt are not incorporated in the present application.

1.4.4 Tilt in more than one direction at a time is not considered. Priorities to left and right turns is higher than forward and backward motion.

1.4.5 A sharp jerk in x/y direction causes acceleration to increase and hence it considers as a tilt in corresponding direction causing false trigger.

1.5 Setup and any extensions implemented on the robot

1.5.1 No extra mechanical setup or extension is required.

1.6 Additional hardware used

1.6.1 Module to convert 5V logic to 3V logic. Supply voltage of 3.3V is obtained by using LM317 (cost=Rs. 40) powered by 5V supply on external POD connection terminal. Logic conversion is obtained using resistor divider network.

1.6.2 Bluetooth module is connected directly to the logic converter module mentioned above. This module costs around Rs. 1500.

1.6.3 Android 2.1 OS based smart phone with Bluetooth 2.0 version and 3-axis accelerometer. We selected Motorola Milestone which cost us Rs. 23,000. Other option includes Samsung galaxy i9000 & HTC desire.

1.7 Any other important aspect/issues, specific to your project

1.7.1 One to one pairing between mobile and FB5 ensures no false commands.

1.7.2 Bluetooth module needs to be configured using PC for baud rate of 9600 and no flow control mode before it can communicate with FB5.

2. Present Status

2.1 A timeline based picture of project stating current status + requirements completed

All objectives that were aimed to be developed in this course project are achieved. FB5 moves as per the tilt of smart phone (Motorola Milestone).

2.2 If there are any delays, why they occurred? How you have overcome issues?

- Many ways of Bluetooth communication are possible. Bluetooth module used on FB5 supports raw data transfer (RFCOMM). Figuring out which method of Bluetooth communication was suitable and implementing that was time consuming.
- Also Android SDK literature talks about Bluetooth communication using SDP server only, which is not supported on Bluetooth module to be mounted on FB5.
- Due to lack of proper documentation and support, AT commands which we initially got were not suitable for this firmware version and we were not able to communicate with Bluetooth to serial module. After an exhaustive search on the internet, we got the reference to AT command supported by this firmware.
- Delay occurred due to malfunctioning of Bluetooth to serial converter module. This occurred

when we connected Bluetooth module directly to power supply for initial testing. After connecting Bluetooth module to its terminal, when we turn ON the power supply, initial surge damaged the Bluetooth module and it started drawing excess current. Issue was resolved only by replacing the module.

2.3 Critical steps in your projects: hardware, interfacing, algorithmic complexity, etc.

2.3.1 Interpretation of accelerometer data:

Built in commands in API for accelerometer can provide roll, pitch, yaw angles. But these are with respect to earth's north and not from user's view point. Hence these values would change depending on the direction in which user is standing. Hence raw accelerometer data has to be used for finding tilt. A sharp jerk in any direction causes it to interpret as a tilt, hence delay has been introduced so as to sample accelerometer at a smaller speed. But this decreases responsivity. Hence a tradeoff between two situations.

2.3.2 Bluetooth Communication channel:

Android SDK API talks mainly about a Bluetooth channel using client-server architecture using a Service Discovery Protocol (SDP) on server end. But in our case, Bluetooth module on FB5 is server end which does not provide SDP service. Hence a raw data transfer mechanism using RFCOMM had to be implemented (not mentioned anywhere in reference literature).

2.3.3 3V supply from 5V supply with minimal power wastage:

LM400 works on 3.3V supply & Firebird V has 5V supply so for LM400 power supply is designed using LM317 voltage regulator IC. LM 317 is used as it uses minimum current among the available voltage converter IC. 78L03 which uses current in the order of micro ampere cannot be used as it requires minimum of 3.5V drop across its input and output terminal. Circuit diagram of power supply is given below. Vout in LM317 is given by:

$$V_{out} = 1.25 * \left(1 + \frac{R2}{R1}\right)$$

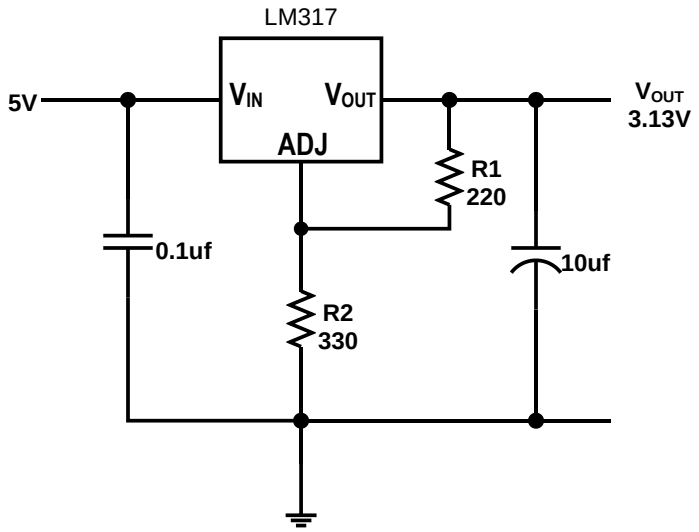


Figure 1. 3.1V regulatot circuit

2.3.4 Logic conversion from 5V to 3V:

To convert transmit and receive logic of AVR 2560 to 3v logic, we use a potential divider circuit. From transmit pin of AVR, two resistors 220Ω and 330Ω are connected in series. Voltage at the junction of two resistors corresponds to 3v logic. Ideally, we need a buffer which converts the 3v output of Bluetooth module to 5v. but since in TTL voltage range for logic high is 2.4v to 5v, we have just connected a resistor

to avoid a direct short between 3v terminal and 5v terminal. This resistor acts as a protection resistor incase AVR receive pin gets short circuited to 5v.

2.4 Test plan

2.4.1 Test criteria and description:

- Movement of FB5 in desired direction with speed confirming to extent of tilt.
- Tilting in simultaneously two directions should give preference to rotation (right or left).

2.4.2 Results:

- Speed variation from 10% to100% in 8 steps was achieved.

2.5 Individual roles and contributions

- Mobile application: Jatin Kanzaria and K.L.Srinivas.
- Bluetooth module interfacing on FB5: Rohan Shah and Jagbandhu.

2.6 How much time devoted to project so far - man-days

Our group consists of four members and each one devoted about 6 hours per week for about two months. This include overall time required for project development starting from background research, setup, implementation, testing and documentation.

3 Final Roadmap of Project

3.1 State deadlines for each activity – Project completion, Documentation, code documentation etc.

Task	Person	Expected Deadline	Completion Date
Reading accelerometer values	Srinivas	5/10/10	4/10/10
Communication using Bluetooth(in mobile)	Jatin	15/10/10	15/10/10
Integration of accelerometer and bluetooth modules(in mobile)	Srinivas and Jatin	20/10/10	18/10/10
Bluetooth-Serial module testing & setup	Jagbandhu	27/10/10	27/10/10
Integrating bluetooth module onto FB5	Rohan and Jagbandhu	01/11/10	02/11/10
Integrating mobile app with FB5	Jatin and Rohan	04/11/10	04/11/10
Documentation & Report making	All	08/11/10	09/11/10

4 Innovation, Creativity and Reusability Index of your Project

4.1 Innovations in project

In this present advanced technology world, many electronic devices has Bluetooth support and developing controlling application based on this technology will enhance the capability of FB5.

4.2 How you have enhanced reusability in project

Accelerometer reading code is modular and can be reused by any other application by suitably initializing and calling it.

Similarly, for Bluetooth after initialization, a send command with "byte array" as argument, transmits the array over Bluetooth.

5 Help us in improving the process

5.1 What you think can be improved in terms of project activities

Students are well motivated to get indulged in challenging projects and this requires some time for understanding and implementing. We think that number of classes and class assignments should be reduced in second half of semester so that students can devote more time on projects and come up with some innovative ideas.

5.2 Any comments on the current schedule of events

Everything is fine with project schedule but we think there should be sufficient time between prototype demo and final demo.

5.3 Are you satisfied with the way the course activities have gone – specially the project?

Yes, we are satisfied.

6 Bug Report

6.1 All known bugs, uniquely numbered and pointers to their resolution

- Response coming from Bluetooth module is also interpreted as data coming from paired device.

7 Future Scope

- On the mobile platform, instead of just tilt, other gestures such as touch can be sensed to determine the direction and speed for motor control.
- Developing an application on mobile which senses the actual acceleration of mobile in 2 dimension and convert it to full path which FB5 has to move.
- Wireless surveillance camera can be mounted onto FB5 which will stream back the video captured through WIFI to Android based phone.

8 Learning's

- A good understanding of how mobile applications are developed on android platform.
- Bluetooth protocols.
- Bluetooth module (LM400) interfaces using AT command.
- Hardware for 5V to 3V logic conversion.

Appendices-A: Readme

CS684 – 2010 Project

Project: Controlling Firebird V using an Android based phone via Bluetooth.

Students:

Name	Roll No.	Email
Jatin Kanzaria	**09307919**	**jatin@ee.iitb.ac.in**
Rohan Shah	**09307050**	**rohanshah@ee.iitb.ac.in**
Jagbandhu	**09307603**	**jagbandhu@ee.iitb.ac.in**
K.L.Srinivas	**09307051**	**k.l.srinivas@ee.iitb.ac.in**

Project Objective

To develop an application on the Android phone which will sense the tilt of the device and send this data over Bluetooth to FB5. Depending on the direction of the tilt the FB5 will move in the corresponding direction. Speed of the FB5 will be determined by the extent of tilt in given direction. FB5 currently does not have a Bluetooth support, so adding it will further enhance FB5 capabilities.

Hardware Platform

1. Android 2.1 OS based mobile with Bluetooth and Accelerometer Sensor.
2. Firebird V ATMEGA2560
3. LM400 bluetooth module
4. LM317 voltage regulator
5. Serial communication through UART2 of ATMEGA2560.

Software

1. Android SDK with Android 2.1 platform for Linux.
2. Eclipse IDE 3.5 for Linux with the Android Development Tool – ADT 0.9.9.
3. WinAVR.
4. AVR Studio 4.

Code Description

Code Files:

Filename	Purpose	Executes on
Android_FB5.java	Main program	Android phone
BluetoothComm.java	Contains Bluetooth communication modules	Android phone
AccelerometerReader.java	Contains modules to read accelerometer values and take appropriate actions.	Android phone
Firebird_BT.c	Process the data from Bluetooth module and corresponding actions are taken on FB5	FB5

Deliverables

Filename	Contains
Android_FB5.tar.gz	Source codes for mobile application to read accelerometer and send commands to FB5 via Bluetooth.
Firebird_BT.tar.gz	Source codes for controlling FB5 robot. Contains header files and main executable programs.

Execution Instructions

Instructions for installing Android_FB5 application:

1. Attach the phone with PC using the USB cable.
2. Copy the Android_FB5.apk file found in /Android_FB5/bin/ and paste in some location on the memory card of the phone.
3. Go to that location and open the Android_FB5 file.
4. It will ask the option to “install”, click on “Install”.
5. The application icon appears along with other software.
6. Open the application and feed in the Bluetooth MAC address of the Bluetooth module connected to FB5 in the text box provided. Now click “Connect” to establish connection between phone and FB5.
7. Now tilting the phone in different directions will cause FB5 to move in corresponding direction.
8. The arrow on the UI indicates the direction of movement and colour indicates the speed. For higher speed colour starts turning reddish.

Instructions for editing the Android_FB5 source code:

1. Install Eclipse IDE 3.5 (for Linux). Instructions on how to install can be found here.. http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/helios/SR1/eclipse-java-helios-SR1-linux-gtk.tar.gz
2. Install Android SDK with Android Platform 2.1. Instructions on how to install can be found here.. http://developer.android.com/sdk/installing.html
3. To view source files, open the required file from /Android_FB5/src/com/iitb/android_fb5/
4. To edit the source file and recompile them, make a new project and copy following files to the working project directory:
    a. /Android_FB5/src/com/iitb/android_fb5/Android_FB5.java
    b. /Android_FB5/src/com/iitb/android_fb5/BluetoothComm.java
    c. /Android_FB5/src/com/iitb/android_fb5/AccelerometerReader.java
    d. /Android_FB5/res/main
    e. /Android_FB5/AndroidManifest
    f. Copy paste all icon files from /Android_FB5/res/drawable-hdpi/ to the corresponding location in working project folder.
5. Now edit the code as required and build.
6. Install the application as described earlier.

Configuring and Interfacing Bluetooth module (LM400) with FB5

For reception of the command via Bluetooth ,we have used Bluetooth module LM400. The interface of LM400 to host system is UART.

1. LM400 works on 3.3V supply & Firebird V has 5V supply so for LM400 power supply is designed using LM317 voltage regulator IC.
2. LM400 has to be first configured before connecting it to Firebird V .For that LM400 UART pins are connected to PC-serial port through MAX232 IC and LM400 UART reciever's pin is given input through potential divider circuit as shown in below figure so as to make TTL logic level 5V to 3.3V.
3. LM400 is configured using the TERMINAL software of PC through which PC's serial port can be accessed. By default LM400 has baud rate of 19200 and hardware flow control. For communicating with FB5 it has to first set at the baud rate of 9600 with no hardware control. For this from the terminal of PC ATL1 is sent to the module through serial port of PC for making its baud rate 9600 and ATC0 for disabling flow control. Now the module is ready for interfacing with FB5.

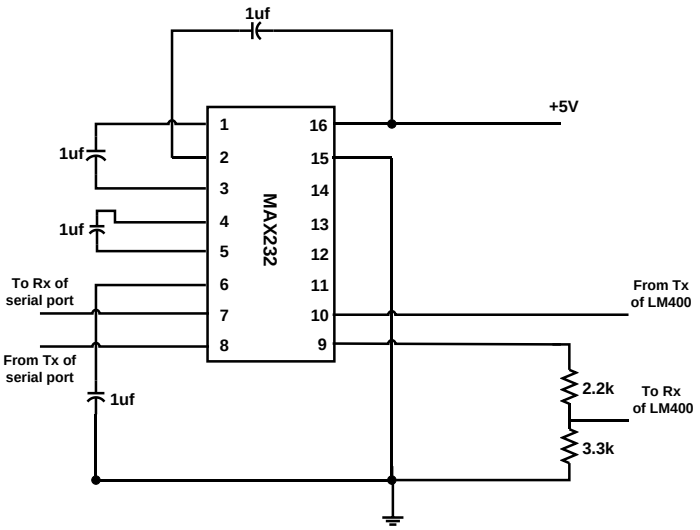


Fig: Interfacing ckt. of LM400 with PC-serial port

4. Bluetooth module is interfaced with FB5 through expansion slot present in FB5.UART2 of ATMEGA2560 is used for serial communication with module. Rx & Tx pin of LM400 is connected to pin 1 & 2 of expansion slot respectively. Power supply is taken from pin 21 & 23 of expansion slot. Also jumper J1 of FB5 is to be changed so that UART2 of ATMEGA2560 gets connected with expansion port of FB5 (refer FB5 hardware manual).

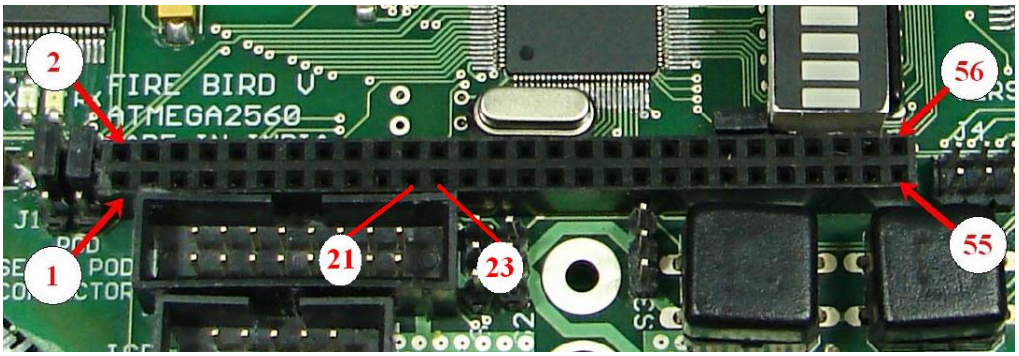


Fig: Expansion slot of FB5

Now FB5 is ready to get instruction from Bluetooth module.

Appendices-B: Source Code

FB5 code : Global.h

```
/*****************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in              -*- c -*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

  Software released under Creative Commence cc by-nc-sa licence.
  For legal information refer to:
  http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

*****************************************************************************/

#ifndef _GLOBAL
#define _GLOBAL

#ifndef F_CPU
#define F_CPU 11059200UL
#endif

unsigned char main_buf[100],maincnt;
unsigned char ser_buf[100],ser_trax_buf[100];
unsigned int count,trx_count,trx_curr_cnt;
unsigned char decode,res_wait;

      unsigned char flag;
      unsigned int unit;
      unsigned int tens;
      unsigned int hundred;
```

```
      unsigned int thousand;
      unsigned int million;
      unsigned int temp;

#endif
```

FB5 code : FireBird.c

```
/**@mainpage package FireBird-5 Control Through Bluetooth
 @author Group 7:             Jagbandhu         09307603
                              Kanzaria Jatin    09307919
                              K. L. Srinivas    09307051
                              Shah Rohan        09307050

 AVR Studio Version 4.17, Build 666

 Date: 8th November 2010

 This experiment demonstrates FireBird 5 control through Bluetooth Interface


 Bluetooth Connections:
            Bluetooth     POD Extension Connector Pin
                    Tx  --> 1  (Txd)
                    Rx  --> 2  (Rxd)
                    VCC --> 21 (+5v)
                    GND --> 23 (GND)

 Note:

 1. Make sure that in the Bluetooth Module is configured to work at baud rate of 9600
      bps, No Parity and No Flow Control.

 2.   Make sure that in the configuration options following settings are
      done for proper operation of the code

      Microcontroller: atmega2560
      Frequency: 11059200
      Optimization: -O0 (For more information read section: Selecting proper optimization
options
                                    below figure 4.22 in the hardware manual)

 3. Jumper for USART2 is connected in POD direction and not in USB.

*****************************************************************************/

/*****************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in              -*- c -*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.
```

```
*****************************************************************************/

#define F_CPU 11059200ul

#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

#include "BlueTooth.h"
#include "MotorControl.h"
#include "GlobalVar.h"
#include "LCD.h"

/*Function to Initialize the Peripheral Devices*/
void Init_Devices(void)
{
      Init_USART2();
      Init_Motor_Peri();
      LCD_Init();
}

/*Main Function*/
void main(void)
{
      cli();
      Init_Devices();
      sei();
      LCD_WR_Command(0x01);
      LCD_Cursor(1,1);
      LCD_String("ANDROID_FB5");
      while (1)
      {
            _delay_ms(100);
            if (decode == 1)
            {
                  unsigned char speed;
                  decode = 0;
                  /*Speed on the scale of 1 to 8.
                        1 => Minimum
                        8 => Maximum*/
                  switch (main_buf[1])
                  {
                        case '1':
                              speed = 0x1F;
                              break;
                        case '2':
                              speed = 0x3F;
```

```
                break;
        case '3':
                speed = 0x5F;
                break;
        case '4':
                speed = 0x7F;
                break;
        case '5':
                speed = 0x9F;
                break;
        case '6':
                speed = 0xBF;
                break;
        case '7':
                speed = 0xDF;
                break;
        case '8':-
                speed = 0xFF;
                break;
        default :
                speed = 0x00;
                break;
}
/* Direction:
        F => Forward
        B => Reverse
        L => Left Turn
        R => Right Turn
        S => Stop*/
switch (main_buf[0])
{
        case 'F' :
                Forward(speed);
                LCD_WR_Command(0x01);
                LCD_Cursor(1,1);
                LCD_String("FORWARD:");
                LCD_Print(1,10,main_buf[1]-0x30,1);
                break;
        case 'B' :
                Reverse(speed);
                LCD_WR_Command(0x01);
                LCD_Cursor(1,1);
                LCD_String("REVERSE:");
                LCD_Print(1,10,main_buf[1]-0x30,1);
                break;
        case 'L' :
                Left_Turn(speed);
                LCD_WR_Command(0x01);
                LCD_Cursor(1,1);
                LCD_String("LEFT TURN:");
                LCD_Print(1,12,main_buf[1]-0x30,1);
                break;
        case 'R' :
                Right_Turn(speed);
                LCD_WR_Command(0x01);
                LCD_Cursor(1,1);
                LCD_String("RIGHT TURN:");
                LCD_Print(1,13,main_buf[1]-0x30,1);
                break;
        case 'S' :
                Stop();
                LCD_WR_Command(0x01);
                LCD_Cursor(1,1);
```

```
                        LCD_String("STOP");
                        break;
                  default :
                        Stop();
                        LCD_WR_Command(0x01);
                        LCD_Cursor(1,1);
                        LCD_String("ANDROID_FB5");
                        break;
            }
      }
   }
}
```

FB5 code : Bluetooth.h

```
/*******************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in              -*- c -*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

  Software released under Creative Commence cc by-nc-sa licence.
  For legal information refer to:
  http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

*******************************************************************************/

#ifndef F_CPU
#define F_CPU 11059200UL
#endif
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

void Init_USART2();
SIGNAL(SIG_USART2_RECV);
SIGNAL(SIG_USART2_TRANS);
void BlueTooth_Add();
```

FB5 code : Bluetooth.c

```
/**@mainpage package Bluetooth Communication
 @author Group 7:             Jagbandhu         09307603
                              Kanzaria Jatin    09307919
                              K. L. Srinivas    09307051
                              Shah Rohan        09307050

 AVR Studio Version 4.17, Build 666

 Date: 8th November 2010

 Bluetooth Connections:
            Bluetooth     POD Extension Connector Pin
                    Tx  --> 1  (Txd)
                    Rx  --> 2  (Rxd)
                    VCC --> 21 (+5v)
                    GND --> 23 (GND)

 Note:

 1. Make sure that in the Bluetooth Module is configured to work at baud rate
     of 9600bps, No Parity and No Flow Control

 2. Make sure that in the configuration options following settings are
      done for proper operation of the code

      Microcontroller: atmega2560
      Frequency: 11059200
      Optimization: -O0 (For more information read section: Selecting proper opti-
                     mization options below figure 4.22 in the hardware manual)

 3. Jumper for USART2 is connected in POD direction and not in USB.

*****************************************************************************/

/*****************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in        -*- c -*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
```

```
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

  Software released under Creative Commence cc by-nc-sa licence.
  For legal information refer to:
  http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

****************************************************************************/
#ifndef F_CPU
#define F_CPU 11059200UL
#endif

#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>

#include "BlueTooth.h"
#include "GlobalVar.h"

/*Function to Initialize USART2 for Communication with Bluetooth
      Baud Rate = 9600
      8 bit
      No Parity
      Transmit and Recieve interrupt Enable*/
void Init_USART2()
{
      UCSR2B = 0x00;
      UCSR2A = 0x00;
      UCSR2C  = 0x06;
      UBRR2L = 0x47;
      UBRR2H = 0x00;
      UCSR2B = 0xd8;
}

/*Recieve Interrupt Handler
      Returns Character String Arrived in main_buf array
      Indicates Main Routine after Whole String Has Arrived*/
SIGNAL(SIG_USART2_RECV)
{
      unsigned char data = UDR2;
      if (data != 0x0a)
      {
            ser_buf[count] = data;
            count++;
      }
      else
      {
            int i;
            if ((ser_buf[0] == '\'') || (ser_buf[0] == 'C') || (ser_buf[0] == 'D'))
                  count = 0;
            else
            {
                  decode = 1;
                  for (i=0;i<count;i++)
                        main_buf[i] = ser_buf[i];
                  maincnt = count;
                  count = 0;
            }
```

```
	}
}

/*Transmit Interrupt Subroutine
	Transmit String in ser_trans_buf array
	No. of characters to be transmitted in trx_count*/
SIGNAL(SIG_USART2_TRANS)
{
	if(trx_count >= trx_curr_cnt)
		UDR2 = ser_trax_buf[trx_curr_cnt];
		trx_curr_cnt++;
}

/*Function to Inquire MAC Address of Bluetooth Module Connected on FB5*/
void BlueTooth_Add()
{
	ser_trax_buf[0] = 'A';
	ser_trax_buf[1] = 'T';
	ser_trax_buf[2] = 'B';
	ser_trax_buf[3] = '?';
	ser_trax_buf[4] = 0x0D;
	trx_count = 4;
	trx_curr_cnt = 1;
	UDR2 = ser_trax_buf[0];
	res_wait = 1;
}
```

FB5 code : Motorcontrol.h

```
/*****************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in              -*- c -*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

  Software released under Creative Commence cc by-nc-sa licence.
  For legal information refer to:
  http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

*****************************************************************************/

#ifndef F_CPU
#define F_CPU 11059200UL
#endif
#include <avr/io.h>
#include <util/delay.h>

void Init_Motor_Peri();
void Init_Motor_Port();
void Init_PWM_Ports();
void Init_Motor_Timer();
void Set_Velocity(unsigned char lm,unsigned char rm);
void Forward(unsigned char lm);
void Reverse(unsigned char lm);
void Left_Turn(unsigned char lm);
void Right_Turn(unsigned char lm);
void Stop();
```

FB5 code : Motorcontrol.c

```
/**@mainpage package Motor Control
 @author Group 7:             Jagbandhu 09307603
                              Shah Rohan 09307050
                              Kanzaria Jatin    09307919
                              K. L. Srinivas    09307051


 AVR Studio Version 4.17, Build 666

 Date: 13th January 2010

*****************************************************************************/

/*****************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in               -*- c -*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

  Software released under Creative Commence cc by-nc-sa licence.
  For legal information refer to:
  http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

*****************************************************************************/

#ifndef F_CPU
#define F_CPU 11059200UL
#endif
#include <avr/io.h>
#include <util/delay.h>

#include "MotorControl.h"
```

```
/*Functio to initialize Peripherals required for Motor Speed Control*/
void Init_Motor_Peri()
{
      Init_PWM_Ports();
      Init_Motor_Timer();
      Init_Motor_Port();
}

/*Function to initialize Ports For PWM*/
void Init_PWM_Ports()
{
      DDRL=0x18;
      PORTL = 0x18;
}

/*Function to initialize Ports For Motor*/
void Init_Motor_Port()
{
      DDRA=0x0F;
      PORTA = 0x00;
}

/*Function to initialize Timer for PWM*/
void Init_Motor_Timer()
{
      TCCR5B = 0x00;
      TCCR5A = 0xA9;
      TCCR5B = 0x0B;
}

/*Function to Set the Velocity for Motor Speed
      Left Motor Speed in lm
      Right Motor Speed in rm*/
void Set_Velocity(unsigned char lm,unsigned char rm)
{
      OCR5AL = (unsigned char)lm;
      OCR5BL = (unsigned char)rm;
}

/*Function to Set the Motion of Motor in Forward Direction
      Speed of Motion in lm*/
void Forward(unsigned char lm)
{
      Set_Velocity(lm, lm);
      PORTA = 0x06;
}

/*Function to Set the Motion of Motor in Reverse Direction
      Speed of Motion in lm*/
void Reverse(unsigned char lm)
{
      Set_Velocity(lm, lm);
      PORTA = 0x09;
}

/*Function to Rotate FB5 in Left Direction
      Speed of Motion in lm*/
void Left_Turn(unsigned char lm)
{
      Set_Velocity(lm, lm);
      PORTA = 0x05;
}
```

```
/*Function to Rotate FB5 in Right Direction
      Speed of Motion in lm*/
void Right_Turn(unsigned char lm)
{
      Set_Velocity(lm, lm);
      PORTA = 0x0A;
}

/*Function to Stop Motor*/
void Stop()
{
      PORTA = 0x00;
}
```

FB5 code : LCD.h

```
/*****************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in              -*- c -*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

  Software released under Creative Commence cc by-nc-sa licence.
  For legal information refer to:
  http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

*****************************************************************************/

#include <avr/io.h>
#include <util/delay.h>

#ifndef F_CPU
#define F_CPU 11059200UL
#endif
#define RS 0
#define RW 1
#define EN 2
#define LCD_PORT PORTC

#define sbit(reg,bit)   reg |= (1<<bit)
#define cbit(reg,bit)   reg &= ~(1<<bit)

void LCD_Init_Ports();
void LCD_Reset_4bit();
void LCD_Init();
void LCD_WR_Command(unsigned char);
void LCD_WR_Char(char);
void LCD_Home();
```

```
void LCD_Cursor(char, char);
void LCD_Print(char, char, unsigned int, int);
void LCD_String(char*);
```

FB5 code : LCD.c

```
/**@mainpage package lcd_interface
 @author Group 7:             Jagbandhu         09307603
                              Kanzaria Jatin    09307919
                              K. L. Srinivas    09307051
                              Shah Rohan        09307050

 AVR Studio Version 4.17, Build 666

 Date: 13th January 2010

 LCD Connections:
                   LCD    Microcontroller Pins
                    RS  --> PC0
                    RW  --> PC1
                    EN  --> PC2
                    DB7 --> PC7
                    DB6 --> PC6
                    DB5 --> PC5
                    DB4 --> PC4

 Note:

 1. Make sure that in the configuration options following settings are
      done for proper operation of the code

      Microcontroller: atmega2560
      Frequency: 11059200
      Optimization: -O0 (For more information read section: Selecting proper optimization
options
                                    below figure 4.22 in the hardware manual)

*****************************************************************************/

/*****************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in              -*- c -*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
```

```
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

  Software released under Creative Commence cc by-nc-sa licence.
  For legal information refer to:
  http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

********************************************************************************/

#ifndef F_CPU
#define F_CPU 11059200UL
#endif
#include <avr/io.h>
#include <util/delay.h>

#include "LCD.h"
#include "GlobalVar.h"

/*****Function to Reset LCD*****/
void LCD_Reset_4bit()
{
	_delay_ms(1);

	cbit(LCD_PORT,RS);				//RS=0 --- Command Input
	cbit(LCD_PORT,RW);				//RW=0 --- Writing to LCD
	LCD_PORT = 0x30;			//Sending 3
	sbit(LCD_PORT,EN);				//Set Enable Pin
	_delay_ms(5);					//Delay
	cbit(LCD_PORT,EN);				//Clear Enable Pin

	_delay_ms(1);

	cbit(LCD_PORT,RS);				//RS=0 --- Command Input
	cbit(LCD_PORT,RW);				//RW=0 --- Writing to LCD
	LCD_PORT = 0x30;			//Sending 3
	sbit(LCD_PORT,EN);				//Set Enable Pin
	_delay_ms(5);					//Delay
	cbit(LCD_PORT,EN);				//Clear Enable Pin

	_delay_ms(1);

	cbit(LCD_PORT,RS);				//RS=0 --- Command Input
	cbit(LCD_PORT,RW);				//RW=0 --- Writing to LCD
	LCD_PORT = 0x30;			//Sending 3
	sbit(LCD_PORT,EN);				//Set Enable Pin
	_delay_ms(5);					//Delay
	cbit(LCD_PORT,EN);				//Clear Enable Pin

	_delay_ms(1);

	cbit(LCD_PORT,RS);				//RS=0 --- Command Input
	cbit(LCD_PORT,RW);				//RW=0 --- Writing to LCD
	LCD_PORT = 0x20;			//Sending 2 to initialise LCD 4-bit mode
	sbit(LCD_PORT,EN);				//Set Enable Pin
	_delay_ms(5);					//Delay
	cbit(LCD_PORT,EN);				//Clear Enable Pin


}
```

```
/*Function to Initialize Port For LCD connection*/
void LCD_Init_Ports (void)
{
      DDRC = DDRC | 0xF7;     //all the LCD pin's direction set as output
      PORTC = PORTC & 0x80;   // all the LCD pins are set to logic 0 except PORTC 7
}


/*****Function to Initialize LCD*****/
void LCD_Init()
{
      LCD_Init_Ports();
      LCD_Reset_4bit();
      _delay_ms(1);
      LCD_WR_Command(0x28);               //LCD 4-bit mode and 2 lines.
      LCD_WR_Command(0x01);
      LCD_WR_Command(0x06);
      LCD_WR_Command(0x0E);
      LCD_WR_Command(0x80);
}


/*****Function to Write Command on LCD*****/
void LCD_WR_Command(unsigned char cmd)
{
      unsigned char temp;
      temp = cmd;
      temp = temp & 0xF0;
      LCD_PORT &= 0x0F;
      LCD_PORT |= temp;
      cbit(LCD_PORT,RS);
      cbit(LCD_PORT,RW);
      sbit(LCD_PORT,EN);
      _delay_ms(5);
      cbit(LCD_PORT,EN);

      cmd = cmd & 0x0F;
      cmd = cmd<<4;
      LCD_PORT &= 0x0F;
      LCD_PORT |= cmd;
      cbit(LCD_PORT,RS);
      cbit(LCD_PORT,RW);
      sbit(LCD_PORT,EN);
      _delay_ms(5);
      cbit(LCD_PORT,EN);
}

/*****Function to Write Data on LCD*****/
void LCD_WR_Char(char letter)
{
      char temp;
      temp = letter;
      temp = (temp & 0xF0);
      LCD_PORT &= 0x0F;
      LCD_PORT |= temp;
      sbit(LCD_PORT,RS);
      cbit(LCD_PORT,RW);
      sbit(LCD_PORT,EN);
      _delay_ms(5);
      cbit(LCD_PORT,EN);

      letter = letter & 0x0F;
```

```
	letter = letter<<4;
	LCD_PORT &= 0x0F;
	LCD_PORT |= letter;
	sbit(LCD_PORT,RS);
	cbit(LCD_PORT,RW);
	sbit(LCD_PORT,EN);
	_delay_ms(5);
	cbit(LCD_PORT,EN);
}


void LCD_Home()
{
	LCD_WR_Command(0x80);
}


/*****Function to Print String on LCD*****/
void LCD_String(char *str)
{
	while(*str != '\0')
	{
		LCD_WR_Char(*str);
		str++;
	}
}

/*** Position the LCD cursor at "row", "column". ***/

void LCD_Cursor (char row, char column)
{
	switch (row) {
		case 1: LCD_WR_Command (0x80 + column - 1); break;
		case 2: LCD_WR_Command (0xc0 + column - 1); break;
		case 3: LCD_WR_Command (0x94 + column - 1); break;
		case 4: LCD_WR_Command (0xd4 + column - 1); break;
		default: break;
	}
}

/***** Function To Print Any input value upto the desired digit on LCD *****/
void LCD_Print (char row, char coloumn, unsigned int value, int digits)
{
	flag = 0;
	if(row==0||coloumn==0)
	{
		LCD_Home();
	}
	else
	{
		LCD_Cursor(row,coloumn);
	}
	if(digits==5 || flag==1)
	{
		million=value/10000+48;
		LCD_WR_Char(million);
		flag=1;
	}
	if(digits==4 || flag==1)
	{
		temp = value/1000;
		thousand = temp%10 + 48;
		LCD_WR_Char(thousand);
```

```
            flag=1;
      }
      if(digits==3 || flag==1)
      {
            temp = value/100;
            hundred = temp%10 + 48;
            LCD_WR_Char(hundred);
            flag=1;
      }
      if(digits==2 || flag==1)
      {
            temp = value/10;
            tens = temp%10 + 48;
            LCD_WR_Char(tens);
            flag=1;
      }
      if(digits==1 || flag==1)
      {
            unit = value%10 + 48;
            LCD_WR_Char(unit);
      }
      if(digits>5)
      {
            LCD_WR_Char('E');
      }

}
```

Android Phone code :

```
///**********************************Android_FB5.java
file*******************************////


/**
 * Project Name: Android_FB5
 * Author:       Jatin Kanzaria.
 *               K.L.Srinivas.
 *               Rohan Shah.
 *               Jagbandhu.
 * Date:   8/11/2010
 */
/*****************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in               -*- c
-*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

  Software released under Creative Commence cc by-nc-sa licence.
  For legal information refer to:
  http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

*****************************************************************************/

package com.iitb.android_fb5;
```

```
import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Toast;

/** Main activity which starts when the application is first Run.
 * Task: (1)Initialise the User Interface.
 *          (2)Initialise button click listeners.
 *          (3)Implements all button click listeners.
 *          (4)Call subroutines to connect to Bluetooth and start Accelerometer on
'Connect' button click.
 *          (5)Call subroutines to disconnect from bluetooth and stop accelerometer
on close of application.
 */
public class Android_FB5 extends Activity {

     final String tag = "Android_FB5";

     /** Bluetooth related objects. */
     private AccelerometerReader mAccelerometerReader = null;
     private BluetoothComm mBluetoothComm = null;
     private static final int REQUEST_ENABLE_BT = 2;
     private BluetoothAdapter mAdapter = null;

     /** UI related objects. */
     private Button mSendButton;
     private Button mConnectButton;
     private Button mDisconnectButton;
     private ImageView mImageView;

     /** Called when the activity is first created.
      *  Starts all the button click listeners.*/
     @Override
     public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);

          setContentView(R.layout.main); /** Set the layout of UI from
"/res/layout/main". */

          Log.d(tag,"Android FB5 started..");

          mAdapter = BluetoothAdapter.getDefaultAdapter(); /** Get the Bluetooth
hardware and create a handle for it. */
          if (mAdapter == null) {
               Toast.makeText(this, "Bluetooth is not available. Closing
Application", Toast.LENGTH_LONG).show();
               finish();
               return;
          }
```

```
            mSendButton = (Button) findViewById(R.id.button_send); /** Start
listeners for button click.*/
            mSendButton.setOnClickListener(SendListener);

            mConnectButton = (Button)this.findViewById(R.id.connect);
            mConnectButton.setOnClickListener(ConnectListener);

            mDisconnectButton = (Button)this.findViewById(R.id.disconnect);
            mDisconnectButton.setOnClickListener(DisconnectListener);

            mDisconnectButton.setEnabled(false); /** Enable only 'Connect' button
initially. */
            mConnectButton.setEnabled(true);
            mSendButton.setEnabled(false);

            mImageView = (ImageView) findViewById(R.id.ImageView);
            mImageView.setImageBitmap(null);
      }


      /** Called when 'Connect' button is clicked. Starts the connection procedure over
BT*/
      private OnClickListener ConnectListener = new OnClickListener()
      {
            public void onClick(View v)
            {
                  Log.d(tag,"Connect Requested");
                  startup();  /** Start the BT connection process and accelerometer.*/

            }
      };

      /** Called when 'Disconnect' button is pressed. Frees the BT channel and stop
accelerometer listener. */
      private OnClickListener DisconnectListener = new OnClickListener()
      {
            public void onClick(View v)
            {
                  Log.d(tag,"Disonnect Requested");
                  if(mBluetoothComm != null)mBluetoothComm.free_channel(); /**Free up
the BT channel. */
                  if(mAccelerometerReader != null)
mAccelerometerReader.unregisterListener();  /** Stop listening to Accelerometer
changes. */
                  mConnectButton.setEnabled(true);
                  mDisconnectButton.setEnabled(false);
                  mSendButton.setEnabled(false);
            }
      };

      /** Only used for testing purpose. */
      private OnClickListener SendListener = new OnClickListener() {
            public void onClick(View v) {

                  // sending test string...
                  byte[] write_buffer = new byte[6];
                  write_buffer[0] = 'J';
```

```
                write_buffer[1] = 'A';
                write_buffer[2] = 'T';
                write_buffer[3] = 'I';
                write_buffer[4] = 'N';

                try {mBluetoothComm.BluetoothSend(write_buffer);
                }
                catch (Exception e){e.printStackTrace();
                }
                Log.d(tag, "Write on button press successful");

          }
     };

     /** Called when the activity starts. Gives a request to turn ON the Bluetooth
id OFF*/
     @Override
     public void onStart() {
          super.onStart();
          Log.d(tag, "++ ON START ++");
          /** If bluetooth is not enabled, ask for user permission to turn on
bluetooth. */
          if (!mAdapter.isEnabled()) {
                Intent enableIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                startActivityForResult(enableIntent, REQUEST_ENABLE_BT);
          } else {
                //if(mAccelerometerReader == null) startup();
          }
     }



     /** Called when the activity resumes. */
     @Override
     public synchronized void onResume() {
          super.onResume();
          if(mAccelerometerReader != null){
                mAccelerometerReader.registerListener();
                }
     }


     /** Called when the activity is aborted.
      * Stops the BT channel and stops accelerometer listener. */
     @Override
     public void onDestroy() {
          super.onDestroy();
          if(mAccelerometerReader != null){
                mAccelerometerReader.unregisterListener();}
          if(mBluetoothComm != null){
                mBluetoothComm.free_channel();}
          Log.e(tag, "--- ON DESTROY ---");
     }

     /** Initialisation function.
      * Called from : 'Connect' button click listener.
```

```
       * Task: (1)Establish connection between phone and bluetooth module on FB5.
       *          (2)Start a listener for changes in value of Accelerometer sensor.
       * Arguments : Null
       * Return : Null
       */
      private void startup()
      {
            mBluetoothComm = new BluetoothComm(this);
            Toast.makeText(this, "Connecting...", Toast.LENGTH_LONG).show();
            try {
                  Log.d(tag, "Initialisation Started...");

                  /** Bluetooth initialise function returns true if connection is
succesful, else false. */
                  if(mBluetoothComm.Initialise() == false)
                  {
                        Toast.makeText(this, " No connection established ",
Toast.LENGTH_SHORT).show();
                        return;
                  }
                  else
                  {
                        Toast.makeText(this, " Connection established ",
Toast.LENGTH_SHORT).show();
                  }
                  Log.d(tag, "Initialisation Successful");
            } catch (Exception e) {
                  e.printStackTrace();
                  Log.e(tag, "Initialisation Failed");
            }
            /** Accelerometer initialisation. */
            mAccelerometerReader = new
AccelerometerReader(getApplicationContext(),this,mBluetoothComm);
            /** Enable 'Disconnect' button. */
            mDisconnectButton.setEnabled(true);
            mConnectButton.setEnabled(false);
            mSendButton.setEnabled(true);
      }

      /** Called when the activity resumes after prompting user to turn ON the
bluetooth.
       * If turned ON, goes ahead with application, else closes the connection and stops
application.
       */
      public void onActivityResult(int requestCode, int resultCode, Intent data)
      {
            Log.d(tag, "onActivityResult " + resultCode);
            if (requestCode == REQUEST_ENABLE_BT)
            {
                  /** When the request to enable Bluetooth returns. */
                  if (resultCode == Activity.RESULT_OK) {
                        Log.d(tag,"BT Enabled");
                        // Bluetooth is now enabled
                  } else {
                        // User did not enable Bluetooth or an error occured
                        Log.d(tag, "BT not enabled");
```

```
                        Toast.makeText(this, "Bluetooth was not enabled. Closing
application..", Toast.LENGTH_LONG).show();
                        finish();  /** Terminate the activity and close application.
*/
                        return;
                    }
                }
            }

}
```

Android Phone code :

```
////*****************************AccelerometerReader.java
file*******************************/////

/**
 * Project Name: Android_FB5
 * Author:       Jatin Kanzaria.
 *               K.L.Srinivas.
 *               Rohan Shah.
 *               Jagbandhu.
 * Date:   8/11/2010
 */
/*****************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in               -*- c
-*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

  Software released under Creative Commence cc by-nc-sa licence.
  For legal information refer to:
  http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

*****************************************************************************/


package com.iitb.android_fb5;
```

```
import android.app.Activity;
import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorListener;
import android.hardware.SensorManager;
import android.util.Log;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

/** Class for implementing Accelerometer sensor reading.
 * Task: (1) Start listener for change in accelerometer values.
 *          (2) Accelerometer change listener, sending appropriate commands over BT
after decoding
 *          (3) Stop accelerometer listener.
 */
@SuppressWarnings("deprecation")
public class AccelerometerReader{

      final String tag = "Android_FB5";

      private static final int STOP = 0;
      private static final int FRONT = 1;
      private static final int RIGHT = 2;
      private static final int BACK = 3;
      private static final int LEFT = 4;

      private SensorManager mSensorManager;
      private float mAccelX = 0;
      private float mAccelY = 0;
      public float mAccelZ = 0;

      private int prev_state = -1;
      private int curr_state = -1;
      private byte cur_speed = -1;
      private byte prev_speed = -1;

      Activity mactivity;
      TextView xViewA = null;
      TextView yViewA = null;
      TextView zViewA = null;
      TextView cmdView = null;
      ImageView mImageView = null;

      private BluetoothComm mmBluetoothComm;
      private Context mcontext;

      private int count = 0;
      private final int DELAY = 20;

      private final SensorListener mSensorAccelerometer = new SensorListener()
      {
            /** Called when there is a change in accelerometer value. Takes the
acceleration values, decodes it
```

```
             * and sends corresponding command strings over bluetooth.
             */
            public void onSensorChanged(int sensor, float [] values)
            {
                  //Log.d(tag, "onSensorChanged: " + sensor + ", x: " + values[0] + ",
y: " + values[1] + ", z: " + values[2]);
                  /** Insert delay, so as to filter out the change in acceleration caused
due to sudden jerks. */
                  if(count<DELAY)
                  {
                        count = count + 1;
                        return;
                  }

                  count = 0;

                  /** Copy the values of acceleration in 3 directions. */
                  mAccelX = values[0];
                  mAccelY = values[1];
                  mAccelZ = values[2];

                  /** Display the acceleration value in text box. */
                  xViewA.setText("Acceleration in X:  " + mAccelX);
                  yViewA.setText("Acceleration in Y:  " + mAccelY);
                  zViewA.setText("Acceleration in Z:  " + mAccelZ);

                  byte[] send_buffer = new byte[4];

                  /** Start decoding accelerometer values. */
                  /** With phone held in upright condition, +ve X-axis goes to right,
                   *  +ve Y-axis goes front and +ve Z-axis points towards sky.
                   *
                   *  Hence  +ve x-value--> Right
                   *         -ve x-value--> Left
                   *         +ve y-value--> Front
                   *         -ve y-value--> Back
                   */
                  if (mAccelX>2.0 && mAccelZ>-9.7)
                  {
                        curr_state = RIGHT;
                        cur_speed = find_speed(mAccelX);
                  }
                  else if (mAccelX< -2.0 && mAccelZ>-9.7)
                  {
                        curr_state = LEFT;
                        cur_speed = find_speed(mAccelX);
                  }
                  else //(mAccelX<2 && mAccelX>-2)
                  {
                        if (mAccelY>2 && mAccelZ>-9.7)
                        {
                              curr_state = FRONT;
                              cur_speed = find_speed(mAccelY);
                        }
                        else if (mAccelY<-2 && mAccelZ>-9.7)
                        {
                              curr_state = BACK;
```

```
                        cur_speed = find_speed(mAccelY);
                    }
                    else
                    {
                        curr_state = STOP;
                        cur_speed = 0;
                    }
                }

                /** Update the text box and send BT command only if direction or speed
is changed. */
                if(prev_state != curr_state || prev_speed !=cur_speed)
                {
                    switch (curr_state)
                    {
                    case STOP:
                    {
                        cmdView.setText("Command: Stop"); //Update text box.
                        send_buffer[0] = 'S'; // Set buffer with string to
indicate Stop command.
                        setImage(curr_state,cur_speed);  // set 'Stop' image on
UI
                        break;
                    }
                    case FRONT:
                    {
                        cmdView.setText("Command: Front. " + "Speed: "
+cur_speed);
                        setImage(curr_state,cur_speed);
                        send_buffer[0] = 'F';
                        break;
                    }
                    case RIGHT:
                    {
                        cmdView.setText("Command: Right. " + "Speed: "
+cur_speed);
                        setImage(curr_state,cur_speed);
                        send_buffer[0] = 'R';
                        break;
                    }
                    case BACK:
                    {
                        cmdView.setText("Command: Back. " + "Speed: "
+cur_speed);
                        setImage(curr_state,cur_speed);
                        send_buffer[0] = 'B';
                        break;
                    }
                    case LEFT:
                    {
                        cmdView.setText("Command: Left. " + "Speed: "
+cur_speed);
                        setImage(curr_state,cur_speed);
                        send_buffer[0] = 'L';
                        break;
                    }
                    }
```

```
                    int tmp = cur_speed + (byte)48;

                    send_buffer[1] = Byte.valueOf((byte)tmp); //Set buffer to
indicate the speed

                    send_buffer[2] = 0x0d; //Add 'CR' 'LF' at the end of command
string
                    send_buffer[3] = 0x0a;


                    mmBluetoothComm.BluetoothSend(send_buffer); //Bluetooth send
function.
                    Log.d(tag, "Transmitted: "+ send_buffer[0] + cur_speed);
               }

               prev_state = curr_state; //Update the state and speed
               prev_speed = cur_speed;
          }

          @Override
          public void onAccuracyChanged(int sensor, int accuracy) {
               // not used
          }
     };

     /** Function to find the speed (from 1 to 8) based on the amount of tilt.
      * Task: (1)Based on the acceleration values it finds the speed(irrespective of
direction) to be sent over BT.
      * Arguments: Acceleration value.
      * Return : Encoded speed on a scale of 1 to 8.
      */
     private byte find_speed(float acc_value)
     {
          byte speed = 0;
          if ((acc_value>2 && acc_value<=3) || (acc_value<-2 && acc_value>=-3))
          {
               speed = 1;
          }
          else if ((acc_value>3 && acc_value<=4) || (acc_value<-3 && acc_value>=-4))
          {
               speed = 2;
          }
          else if ((acc_value>4 && acc_value<=5) || (acc_value<-4 && acc_value>=-5))
          {
               speed = 3;
          }
          else if ((acc_value>5 && acc_value<=6) || (acc_value<-5 && acc_value>=-6))
          {
               speed = 4;
          }
          else if ((acc_value>6 && acc_value<=7) || (acc_value<-6 && acc_value>=-7))
          {
               speed = 5;
          }
          else if ((acc_value>7 && acc_value<=8) || (acc_value<-7 && acc_value>=-8))
          {
               speed = 6;
```

```
		}
		else if ((acc_value>8 && acc_value<=9) || (acc_value<-8 && acc_value>=-9))
		{
			speed = 7;
		}
		else
		{
			speed = 8;
		}

		return speed;

	}


	/** Function to set an appropriate image on UI
	 * Task: Based on the direction and the speed, an appropriate image is displayed on the UI.
	 * Arguments: Direction(state) and speed.
	 * Return : Null
	 */
	private void setImage(int state, int speed)
	{
		switch(state)
		{
		case STOP:
		{
			mImageView.setImageResource(R.drawable.stop);
			break;
		}
		case FRONT:
		{
			switch (speed)
			{
			case 1:
			{
				mImageView.setImageResource(R.drawable.front1);
				break;
			}
			case 2:
			{
				mImageView.setImageResource(R.drawable.front3);
				break;
			}
			case 3:
			{
				mImageView.setImageResource(R.drawable.front4);
				break;
			}
			case 4:
			{
				mImageView.setImageResource(R.drawable.front5);
				break;
			}
			case 5:
			{
				mImageView.setImageResource(R.drawable.front6);
```

```
            break;
      }
      case 6:
      {
            mImageView.setImageResource(R.drawable.front7);
            break;
      }
      case 7:
      {
            mImageView.setImageResource(R.drawable.front8);
            break;
      }
      case 8:
      {
            mImageView.setImageResource(R.drawable.front9);
            break;
      }

      }
      break;
}
case BACK:
{
      switch (speed)
      {
      case 1:
      {
            mImageView.setImageResource(R.drawable.back1);
            break;
      }
      case 2:
      {
            mImageView.setImageResource(R.drawable.back3);
            break;
      }
      case 3:
      {
            mImageView.setImageResource(R.drawable.back4);
            break;
      }
      case 4:
      {
            mImageView.setImageResource(R.drawable.back5);
            break;
      }
      case 5:
      {
            mImageView.setImageResource(R.drawable.back6);
            break;
      }
      case 6:
      {
            mImageView.setImageResource(R.drawable.back7);
            break;
      }
      case 7:
      {
```

```
                mImageView.setImageResource(R.drawable.back8);
                break;
        }
        case 8:
        {
                mImageView.setImageResource(R.drawable.back9);
                break;
        }

        }
        break;
}
case RIGHT:
{
        switch (speed)
        {
        case 1:
        {
                mImageView.setImageResource(R.drawable.right1);
                break;
        }
        case 2:
        {
                mImageView.setImageResource(R.drawable.right3);
                break;
        }
        case 3:
        {
                mImageView.setImageResource(R.drawable.right4);
                break;
        }
        case 4:
        {
                mImageView.setImageResource(R.drawable.right5);
                break;
        }
        case 5:
        {
                mImageView.setImageResource(R.drawable.right6);
                break;
        }
        case 6:
        {
                mImageView.setImageResource(R.drawable.right7);
                break;
        }
        case 7:
        {
                mImageView.setImageResource(R.drawable.right8);
                break;
        }
        case 8:
        {
                mImageView.setImageResource(R.drawable.right9);
                break;
        }
```

```
            }
            break;
      }
      case LEFT:
      {
            switch (speed)
            {
            case 1:
            {
                  mImageView.setImageResource(R.drawable.left1);
                  break;
            }
            case 2:
            {
                  mImageView.setImageResource(R.drawable.left3);
                  break;
            }
            case 3:
            {
                  mImageView.setImageResource(R.drawable.left4);
                  break;
            }
            case 4:
            {
                  mImageView.setImageResource(R.drawable.left5);
                  break;
            }
            case 5:
            {
                  mImageView.setImageResource(R.drawable.left6);
                  break;
            }
            case 6:
            {
                  mImageView.setImageResource(R.drawable.left7);
                  break;
            }
            case 7:
            {
                  mImageView.setImageResource(R.drawable.left8);
                  break;
            }
            case 8:
            {
                  mImageView.setImageResource(R.drawable.left9);
                  break;
            }

            }
            break;
      }
      }


}

/** Constructor for the class.Starts acceleration listener.
```

```
	 * Acquires handels on the text box and image view.
	 */
	public AccelerometerReader(Context context, Activity activity, BluetoothComm
mBluetoothComm)
	{
		/** Register Sensor listener as soon as the class is instanciated. */

		mSensorManager =
(SensorManager)activity.getSystemService(Context.SENSOR_SERVICE);

	mSensorManager.registerListener(mSensorAccelerometer,SensorManager.SENSOR_AC
CELEROMETER,SensorManager.SENSOR_DELAY_GAME);

		mactivity = activity;
		mmBluetoothComm = mBluetoothComm;
		mcontext = context;

		/** Acquire handels on the text box and image view. */
		xViewA = (TextView) mactivity.findViewById(R.id.xbox);
		yViewA = (TextView) mactivity.findViewById(R.id.ybox);
		zViewA = (TextView) mactivity.findViewById(R.id.zbox);
		cmdView    = (TextView)mactivity.findViewById(R.id.cmdbox);
		mImageView = (ImageView) mactivity.findViewById(R.id.ImageView);

	}

	/** Function to start the accelerometer listener. */
	public void registerListener()
	{

	mSensorManager.registerListener(mSensorAccelerometer,SensorManager.SENSOR_AC
CELEROMETER,SensorManager.SENSOR_DELAY_GAME);
	}

	/** Function to stop the accelerometer listener.
	 * Also sets the acceleration text box and image viewer blank.
	 */
	public void unregisterListener()
	{
		mSensorManager.unregisterListener(mSensorAccelerometer);
		xViewA.setText("Acceleration in X:  - - - - ");
		yViewA.setText("Acceleration in Y:  - - - - ");
		zViewA.setText("Acceleration in Z:  - - - - ");
		cmdView.setText("");
		mImageView.setImageBitmap(null);
		Toast.makeText(mcontext, "Disconnected", 0).show();
	}

	/** Function to access x-acceleration values from outside the class. */
	public float getXvalue()
	{
		return mAccelX;
	}

	/** Function to access y-acceleration values from outside the class. */
	public float getYvalue()
	{
```

```
            return mAccelY;
      }

      /** Function to access z-acceleration values from outside the class. */
      public float getZvalue()
      {
            return mAccelZ;
      }

}
```

Android Phone code :

```
///***************************BluetoothComm.java
file*********************************////


/**
 * Project Name: Android_FB5
 * Author:       Jatin Kanzaria.
 *               K.L.Srinivas.
 *               Rohan Shah.
 *               Jagbandhu.
 * Date:   8/11/2010
 */
/*****************************************************************************

   Copyright (c) 2010, ERTS Lab IIT Bombay erts@cse.iitb.ac.in               -*- c
-*-
   All rights reserved.

   Redistribution and use in source and binary forms, with or without
   modification, are permitted provided that the following conditions are met:

   * Redistributions of source code must retain the above copyright
     notice, this list of conditions and the following disclaimer.

   * Redistributions in binary form must reproduce the above copyright
     notice, this list of conditions and the following disclaimer in
     the documentation and/or other materials provided with the
     distribution.

   * Neither the name of the copyright holders nor the names of
     contributors may be used to endorse or promote products derived
     from this software without specific prior written permission.

   * Source code can be used for academic purpose.
       For commercial use permission form the author needs to be taken.

  THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
  AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
  IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
  ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE
  LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
  CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
  SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
  INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
  CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
  ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
  POSSIBILITY OF SUCH DAMAGE.

  Software released under Creative Commence cc by-nc-sa licence.
  For legal information refer to:
  http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

*****************************************************************************/

package com.iitb.android_fb5;
```

```
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.lang.reflect.Method;
import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.util.Log;
import android.widget.EditText;

/** Class to implement all routines related to bluetooth communication.
 *    Task: Initialise bluetooth and establish connection.
 *          Send a byte array over the BT channel.
 *          Disconnect from the BT device and free the BT channel.
 */
public class BluetoothComm{

     final String tag = "Android_FB5";

     /** BT related objects. */
     private BluetoothSocket mBluetoothSocket = null;
    private InputStream mInputStream = null;
    private OutputStream mOutputStream = null;
    private BluetoothDevice mBluetoothDevice = null;

    /** UI related objects. */
    private EditText mAddressText = null;
    private final Activity mactivity;

    /** Constructor for the class. Copies the 'activity' object for its use.*/
    public BluetoothComm(Activity activity)
    {
     mactivity = activity;
    }

     /** Class for all Bluetooth related functions.
      *  Task: (1)Acquire a BT socket and connect over that socket.
      *      (2)Establish input and output streams over the socket for data transfer
      *  Arguments: Null
      *  Return: True is initialisation was successful, else False.
      * @throws Exception
      */
    public boolean Initialise() throws Exception
     {
           /** Get a handle to the BT hardware. */
           BluetoothAdapter mBluetoothAdapter =
BluetoothAdapter.getDefaultAdapter();

           String add_string;
           /** Get the Address of BT device to be connected with, from the text box
on UI. */
           mAddressText = (EditText) mactivity.findViewById(R.id.btaddress);
           add_string = mAddressText.getText().toString();
           try {
                 /** Link the taget BT address to be connected. */
```

```
                mBluetoothDevice = mBluetoothAdapter.getRemoteDevice(add_string);
            }catch (IllegalArgumentException e)
            {
                /** Exception is thrown if BT address is not valid. Then return false*/
                return false;
            }

            //mBluetoothDevice =
mBluetoothAdapter.getRemoteDevice("00:25:56:DF:76:85");//jatin pc
        //mBluetoothDevice =
mBluetoothAdapter.getRemoteDevice("00:1F:DF:D6:71:A1");//jatin phone
        //mBluetoothDevice =
mBluetoothAdapter.getRemoteDevice("80:50:1B:60:E6:D0");//k.l.
        //mBluetoothDevice =
mBluetoothAdapter.getRemoteDevice("00:24:2C:C2:C8:66");//k.l.pc
        //mBluetoothDevice =
mBluetoothAdapter.getRemoteDevice("00:12:6F:03:72:48");//serial adapter

        Method m;
            m= mBluetoothDevice.getClass().getMethod("createRfcommSocket",new
Class[] { int.class });
            mBluetoothSocket = (BluetoothSocket)m.invoke(mBluetoothDevice,
Integer.valueOf(1));
        Log.d(tag, "Connecting...");

        try {
            /** This is a blocking call and will only return on a successful connection
or an exception. */
             mBluetoothSocket.connect();
        } catch (IOException e) {
            /** If target BT device not found or connection refused then return false.
*/
             try {
                 mBluetoothSocket.close();
             } catch (IOException e2) {
                 Log.e(tag, "unable to close() socket during connection failure", e2);
             }
             Log.e(tag,"returning false");
             return false;
        }

        Log.d(tag, "Connected");
        /** Get input and output stream handles for data transfer. */
            mInputStream = mBluetoothSocket.getInputStream();
            mOutputStream = mBluetoothSocket.getOutputStream();
            return true;
     }

     /** Function to send data over BT.
      * Task: (1)To send the byte array over Bluetooth Channel.
      * Arguments: An array of bytes to be sent.
      * Return: Null
      */
    public void BluetoothSend(byte[] write_buffer)
     {
          try {
          mOutputStream.write(write_buffer);
```

```
         }catch (IOException e){Log.e(tag, "Writing on command error");}
         Log.d(tag, "Writing on command successful");
      }

     /** Function to close BT connection.
      * Task: (1)Close input and output streams
      *           (2)Close Bluetooth socket.
      * Arguments: Null
      * Return: Null
      */
      public void free_channel()
      {
            try {
             if (mInputStream != null) {
                 mInputStream.close();
             }
             if (mOutputStream != null) {
                 mOutputStream.close();
             }
             if (mBluetoothSocket != null) {
                 mBluetoothSocket.close();
             }
             Log.d(tag, "BT Channel free");

        } catch (IOException e) {
             e.printStackTrace();
        }
      }

}


////*****************************Main.xml layout file***********************/////



<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" android:orientation="vertical"
android:background="#4e7eb2">
<TextView
      android:text="@string/hello"
    android:layout_height="wrap_content" android:layout_width="fill_parent"
android:textColor="#ad2628"/>


<TextView
    android:id="@+id/xbox"
    android:layout_width="fill_parent" android:layout_height="wrap_content"
android:textColor="#ad4805" android:text="Acceleration in X:  - - - -"/>
<TextView
    android:id="@+id/ybox"
    android:layout_height="wrap_content" android:layout_width="fill_parent"
android:textColor="#ad4805" android:text="Acceleration in Y:  - - - -"/>
<TextView
    android:id="@+id/zbox"
```

```
    android:layout_height="wrap_content" android:layout_width="fill_parent"
android:textColor="#ad4805" android:text="Acceleration in Z:  - - - -"/>

<TextView
    android:id="@+id/cmdbox"
    android:layout_height="wrap_content" android:layout_width="fill_parent"
android:textColor="#ffb71d4d"/><ImageView android:layout_height="wrap_content"
android:id="@+id/ImageView" android:layout_weight="1"
android:layout_width="fill_parent" android:layout_gravity="center"></ImageView>
<Button android:id="@+id/button_send"
            android:layout_height="wrap_content" android:textColor="#b71d4d"
android:hapticFeedbackEnabled="true" android:text="Test Send"
android:layout_width="wrap_content" android:layout_gravity="right"/>



<LinearLayout android:id="@+id/LinearLayout01"
android:layout_height="wrap_content" android:orientation="horizontal"
android:layout_width="fill_parent" android:gravity="bottom"
android:background="#ad4805"><TextView android:id="@+id/TextView04"
android:layout_width="wrap_content" android:layout_height="wrap_content"
android:text="Bluetooth Address" android:textColor="#ffffffff"
android:gravity="center" android:width="95px"
android:layout_gravity="center"></TextView><EditText
android:layout_width="wrap_content" android:layout_height="wrap_content"
android:layout_weight="1" android:layout_gravity="center_vertical"
android:id="@+id/btaddress" android:cursorVisible="false"
android:layout_marginTop="3px" android:width="250px"
android:text="00:12:6F:03:72:48"></EditText><Button
android:layout_width="wrap_content" android:layout_height="wrap_content"
android:text="Connect" android:id="@+id/connect" android:layout_weight="1"
android:layout_gravity="center_vertical" android:layout_marginTop="3px"></Button>

<Button android:layout_width="wrap_content" android:layout_height="wrap_content"
android:text="Disconnect" android:layout_weight="1"
android:layout_gravity="center_vertical" android:layout_marginTop="3px"
android:id="@+id/disconnect"></Button>


</LinearLayout>




</LinearLayout>



///************************AndroidManifest.xml file*****************************////


<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.iitb.android_fb5"
      android:versionCode="1"
      android:versionName="1.0">
      <uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

```
    <uses-permission android:name="android.permission.BLUETOOTH" />
    <application android:icon="@drawable/icon" android:label="@string/app_name">
        <activity android:name=".Android_FB5"
                  android:label="@string/app_name" android:icon="@drawable/icon"
android:screenOrientation="landscape">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

    </application>


</manifest>
```

Appendices-C: Final Presentation

Controlling FB5 using Android Phone over Bluetooth

Group No. 7
Jatin Kanzaria
Rohan Shah
Jagbandhu
K.L.Srinivas

Problem statement:

- Controlling FB5 using mobile via Bluetooth
- User Interface developed on Android Platform
- FB5 to sense user gestures (tilting of phone in different directions)
- Two major components of project :
  1. Developing user interface.
  2. Building Bluetooth module on FB5.

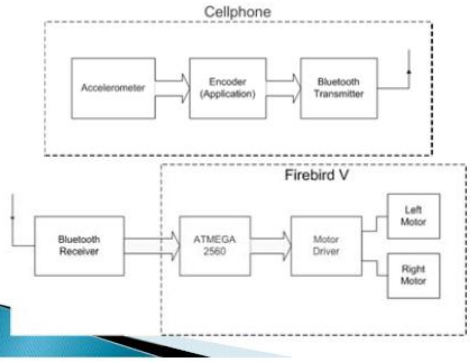
Requirements/Task specifications:

- Interface accelerometer on Android phone
  - Built in commands in API for accelerometer can provide acceleration values, used for decoding tilt values.
- Interface Bluetooth module on Android phone.
  - Raw data transfer mechanism using RFCOMM is implemented
- Interface Bluetooth module on FB5.
  - 3V supply from 5V supply with minimal power wastage using LM317.
  - Logic conversion from 5V to 3V using potential divider.

Project plan:

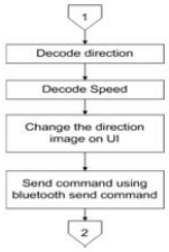
Task	Person	Expected Deadline	Completion Date
Reading accelerometer values	Srinivas	5/10/10	4/10/10
Communication using Bluetooth(in mobile)	Jatin	15/10/10	15/10/10
Integration of accelerometer and bluetooth modules(in mobile)	Srinivas and Jatin	20/10/10	18/10/10
Bluetooth-Serial module testing & setup	Jagbandhu	27/10/10	27/10/10
Integrating bluetooth module onto FB5	Rohan and Jagbandhu	01/11/10	02/11/10
Integrating mobile app with FB5	Jatin and Rohan	04/11/10	04/11/10

Block Diagram



Project flowchart:

Project flowchart (cont.):



Innovation and Challenges:

- Innovations:
  - Bluetooth connectivity on FB5
- Challenges:
  - Developing mobile application on Android platform
  - Interfacing Bluetooth module with FB5

Task completed:

- Reading accelerometer values
- Communication using Bluetooth in mobile
  - Initial testing using SDP
  - Implementing RFCOMM communication
- Integration of accelerometer and Bluetooth modules in mobile
- Bluetooth-Serial module testing and setup
  - 3V power supply
  - 5V to 3V logic conversion
- Integration Bluetooth module onto FB5
  - Configuration settings
- Integration mobile application with FB5

Test plan:

- Test criteria and description:
  - Movement of FB5 in desired direction with speed confirming to extent of tilt.
  - Tilting in simultaneously two directions should give preference to rotation (right or left).
- Results:
  - Speed variation from 10% to100% in 8 steps was achieved.

Reusability:

- Accelerometer reading code is modular and can be reused by any other application by suitably initializing and calling it.
- Similarly, for Bluetooth after initialization, a send command with "byte array" as argument, transmits the array over Bluetooth.
- Application layer of FB5 and Bluetooth layer works independently.

Future Enhancements:

- Wireless surveillance camera can be mounted onto FB5 which will stream back the video captured through WIFI to Android based phone.
- On the mobile platform, instead of just tilt, other gestures such as touch can be sensed to determine the direction and speed for motor control.
- Developing an application on mobile which senses the actual acceleration of mobile in 2 dimension and convert it to full path which FB5 has to move.