Project Report
Sensor Module Interfacing

Task: Interfacing Accelerometer MMA7361 with ATmega2560 in Firebird V Robot

Team members

Chayatan Mukilan A Shantanu

July 9, 2014

Under the guidance of
Prof. Kavi Arya
and
Parin Chedda

Embedded and Real-Time Systems Laboratory
Department of Computer Science and Engineering
Indian Institute of Technology
Bombay

Contents

Abstract

The project aims at interfacing an accelerometer sensor with Fire Bird V educational robot. This additional module can be used for detection of the degree of tilt of the module. In this paper we will see about the basic MMA7361 Accelerometer sensor module interfacing with Atmega 2560 in Fire Bird V robot. This will include the working principle, basic interfacing circuit, programming and applications of the MMA7361 Accelerometer.

1 Introduction

As the name indicates this sensor is used to measure acceleration. Acceleration here does not mean rate of change in velocity in a particular axis. This sensor instead provides the g force that is acting on the test mass located inside the sensor. The measure of force that each coordinate experiences due to the action of gravity is what the sensor actually measures. For example, an accelerometer at rest on the surface of the earth will measure an acceleration g= 9.81 m/s2 straight upwards, due to its weight. By contrast, accelerometers in free fall or at rest in outer space will measure zero. The term for the type of acceleration that accelerometers can measure is g-force acceleration.

2 How an Accelerometer Works

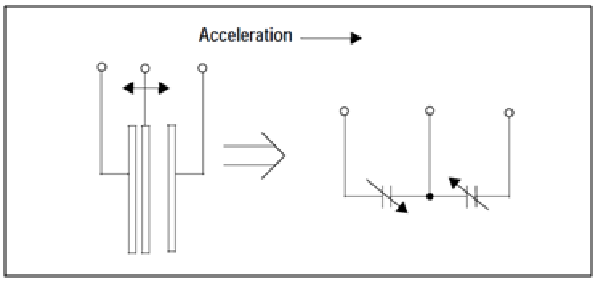


Figure 1: Structure of a g-Cell

The Freescale accelerometer is a surface-micromachined integrated-circuit accelerometer. The device consists of a surface micromachined capacitive sensing cell (g-cell) and a signal conditioning ASIC contained in a single package.

The g-cell is a mechanical structure, as shown in figure 1, formed from semiconductor materials (polysilicon) using semiconductor processes (masking and etching). It can be modeled as a set of beams attached to a movable central mass that move between fixed beams. The movable beams can be deflected from their rest position by subjecting the system to an acceleration. As the beams attached to the central mass move, the distance from them to the fixed beams on one side will increase by the same amount that the distance to the fixed beams on the other side decreases. The change in distance is a measure of acceleration. The g-cell beams form two back-to-back capacitors. As the center beam moves with acceleration, the distance between the beams changes and each capacitor's value will change, (C = A/D). Where A is the area of the beam, is the dielectric constant, and D is the distance between the beams.

3 Pin Connections of MMA7361 Accelerometer

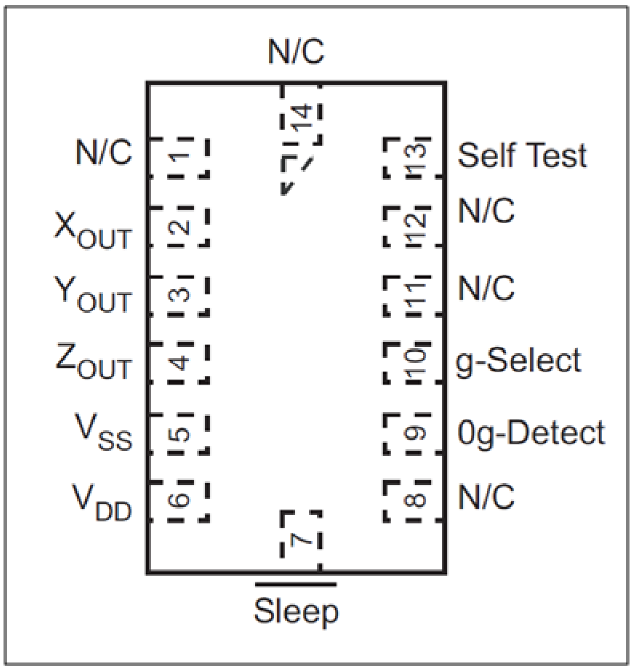


Figure 2: Pin Diagram

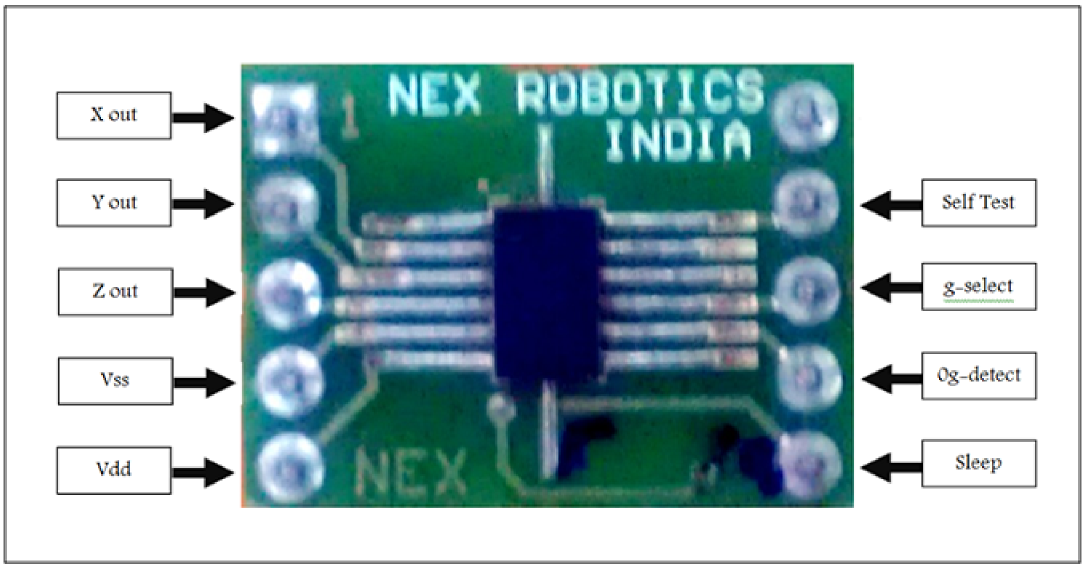
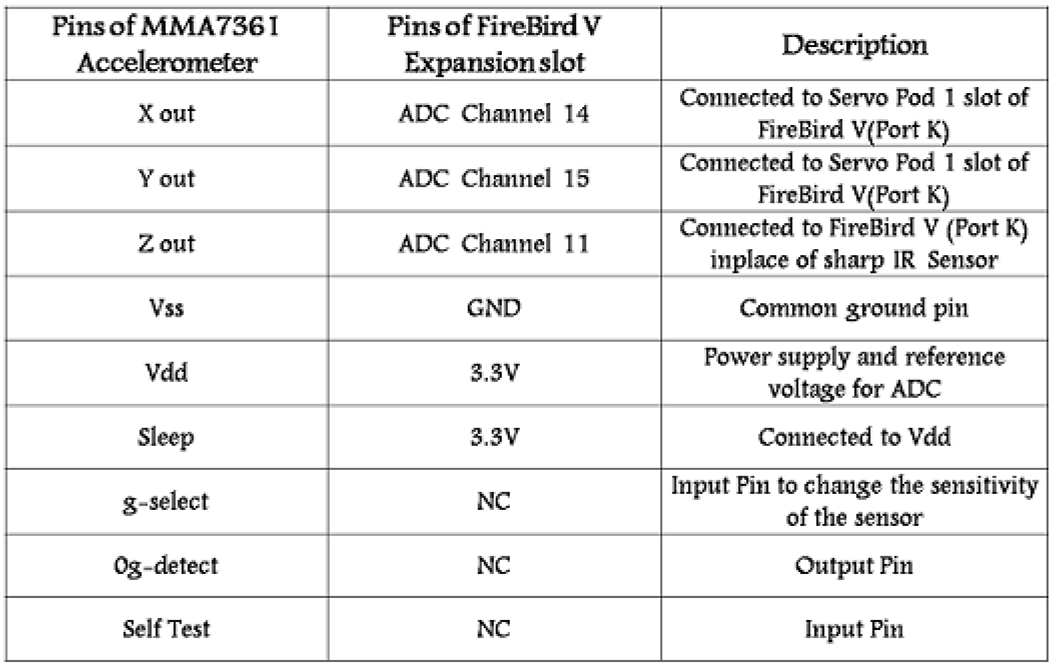


Figure 3: Pin Connections of the Accelerometer

4 Connections of MMA7361 Accelerometer with the ATmega2560

1. Connect the Xout, Yout, and Zout pins to the PORT K pins (analog channels) of the ATmega2560 processor to convert the analog output values of the Accelerometer to digital values that can be used by the Microcontroller.
2. Connect the Vss Pin of the accelerometer to the Ground pin of the Microcontroller.
3. Connect the Vdd Pin of the accelerometer to the 3.3V of the Microcontroller.
4. The Sleep pin is shorted to the Vdd pin of the Accelerometer, in order to provide significant reduction in the operating current.
5. The remaining pins of the accelerometer i.e. Self Test, G select, 0g detect pins are left unconnected because
   - 0g detect pin is an output pin which gives high output if all the x, y, z axes are at 0g.
   - Self Test pin allows the verification of the mechanical and electrical integrity of the accelerometer at any time before or after installation.
   - G select pins are left open because the operation of the accelerometer is in the 0g mode, where the sensitivity is 800mv/g.



Pins of MMA7361 Accelerometer	Pins of FireBird V Expansion slot	Description
X out	ADC Channel 14	Connected to Servo Pod 1 slot of FireBird V(Port K)
Y out	ADC Channel 15	Connected to Servo Pod 1 slot of FireBird V(Port K)
Z out	ADC Channel 11	Connected to FireBird V (Port K) inplace of sharp IR Sensor
Vss	GND	Common ground pin
Vdd	3.3V	Power supply and reference voltage for ADC
Sleep	3.3V	Connected to Vdd
g-select	NC	Input Pin to change the sensitivity of the sensor
0g-detect	NC	Output Pin
Self Test	NC	Input Pin

Figure 4: Connections of MMA7361 Accelerometer with the ATmega2560

5 Orientation of the Accelerometer along the Various Axes

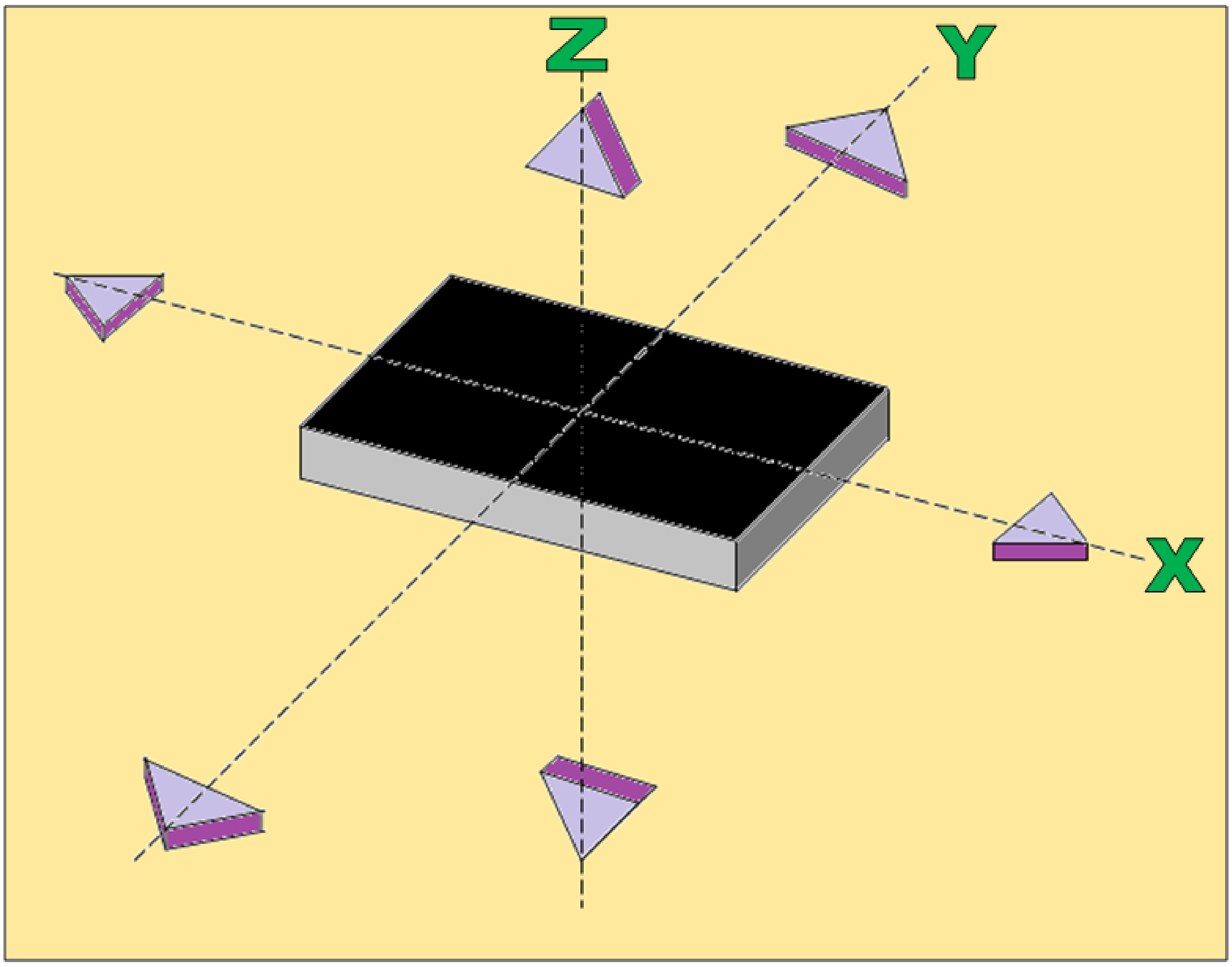
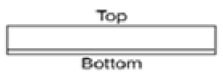
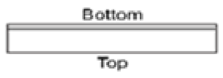
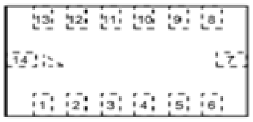


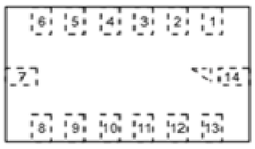
Figure 5: Orientation of the Accelerometer along the Various Axes

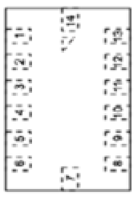
6 Table showing the values of the Accelerometer along the Various Axes

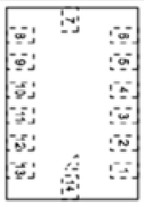
Parameters	G Value	Analog(Theoretical) Volts	Analog(Practical) Volts	ADC(Practical)	Orientation
x	0	1.65	1.63	85	
y	0	1.65	1.73	89	
z	1	2.45	2.13	109	

x	0	1.65	1.63	84	
y	0	1.65	1.74	87	
z	-1	0.85	0.55	28	

x	0	1.65	1.63	84	
y	-1	0.85	0.93	48	
z	0	1.65	1.34	65	

Parameters	G Value	Analog(Theoretical) Volts	Analog(Practical) Volts	ADC(Practical)	Orientation
x	0	1.65	1.6	83	
y	1	2.45	2.5	128	
z	0	1.65	1.35	67	

x	1	2.45	2.39	123	
y	0	1.65	1.72	88	
z	0	1.65	1.4	70	

x	-1	0.85	0.86	43	
y	0	1.65	1.72	88	
z	0	1.65	1.27	64	

7 C Code

7.1 Using the header file

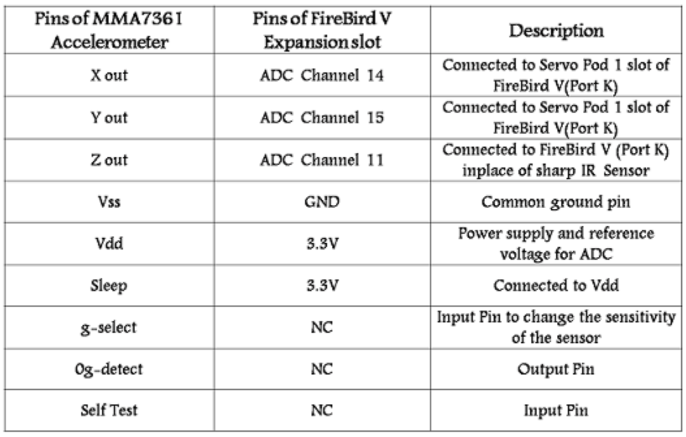
As explained earlier, this sensor, senses orientation of the particular axis and accordingly gives the analog output on the Xout, Yout and Zout pins.
Therefore we have defined a header file called as **accelerometer.h** where we call the following functions that are used in interfacing the Accelerometer to the Firebird V Robot. This header file can be found in the **Headers** folder.
The following functions are used in this header file:

- `acc_init_devices() function : This function initializes all the ADC ports and configures the respective ADC registers.`
- `acc_process() : This function is used to compare the ADC values with the threshold, make the oriental decision and updates the left, right, forward and backward flags.`
- `acc_get_x() : This function will get the ADC values of the X co-ordinate of the Accelerometer when connected to the ADC`
- `acc_get_y() : This function will get the ADC values of the Y co-ordinate of the Accelerometer when connected to the ADC`
- `acc_get_z() : This function will get the ADC values of the Z co-ordinate of the Accelerometer when connected to the ADC`

These functions will be explained in detailed in the later sections.

7.2 Sample code calling the accelerometer.h header file

Now we will see the Sample Program calling the scan pir.h header. Prior to the C Code, The following connections should be made:



Pins of MMA7361 Accelerometer	Pins of FireBird V Expansion slot	Description
X out	ADC Channel 14	Connected to Servo Pod 1 slot of FireBird V(Port K)
Y out	ADC Channel 15	Connected to Servo Pod 1 slot of FireBird V(Port K)
Z out	ADC Channel 11	Connected to FireBird V (Port K) inplace of sharp IR Sensor
Vss	GND	Common ground pin
Vdd	3.3V	Power supply and reference voltage for ADC
Sleep	3.3V	Connected to Vdd
g-select	NC	Input Pin to change the sensitivity of the sensor
0g-detect	NC	Output Pin
Self Test	NC	Input Pin

Figure 6: Connections of MMA7361 Accelerometer with the ATmega2560

SAMPLE CODE

```
#define F_CPU 14745600
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <math.h>
#include "lcd.h"
#include "accelerometer.h"

int main(void)
{

unsigned char x,y,z;     //Initialise variables
acc_init_devices();      //Initialise the Ports for LCD & ADC
lcd_cursor(2,8);

while(1)
{
acc_process();    //Read the ADC values of the
                  //Accelerometer connected to the ADC pins
x=acc_get_x();             //Get X Co-ordinate values
y=acc_get_y();             //Get Y Co-ordinate values
z=acc_get_z();             //Get Z Co-ordinate values
lcd_print(2,1,x,3);        //Display the X value in LCD
lcd_print(2,6,y,3);        //Display the Y value in LCD
lcd_print(2,11,z,3);       //Display the Z value in LCD

}
return 0;
}
```

7.3 Functions used in the accelerometer.h header file

1. **acc_init_devices() :**

```
void acc_init_devices()
{
cli();           //Clear Interrupt
port_init();     //Initialise the ADC & LCD Port pins
lcd_init();      //Initialise LCD
adc_init();      //Initialise ADC
sei();           //Clear Interrupt
}
```

2. **acc_init_devices() :**

```
void acc_process(void)
{
x= ADC_Conversion(14);  //ADC Conversion of X out
y= ADC_Conversion(15);  //ADC Conversion of Y out
z= ADC_Conversion(11);  //ADC Conversion of Z out

//The code below can be uncommented to check if the
//values obtained by the ADC for the Accerometer are
// accurate or not

/*
lcd_cursor(1,1);
lcd_string("X = ");
lcd_print(1,4,x,3);
lcd_cursor(1,7);
lcd_string(" y = ");
lcd_print(1,11,y,3);
lcd_cursor(2,1);
lcd_string("z = ");
lcd_print(2,4,z,3)
;*/

}
```

3. **acc_get_x() :**

```
unsigned char acc_get_x(void)
{
return(x);     //return the value of x
}
```

4. **acc_get_y() :**

```
unsigned char acc_get_y(void)
{
return(y);     //return the value of y
}
```

5. **acc_get_z() :**

```
unsigned char acc_get_z(void)
{
return(z);     //return the value of z
}
```

8 Sample Output as seen on LCD

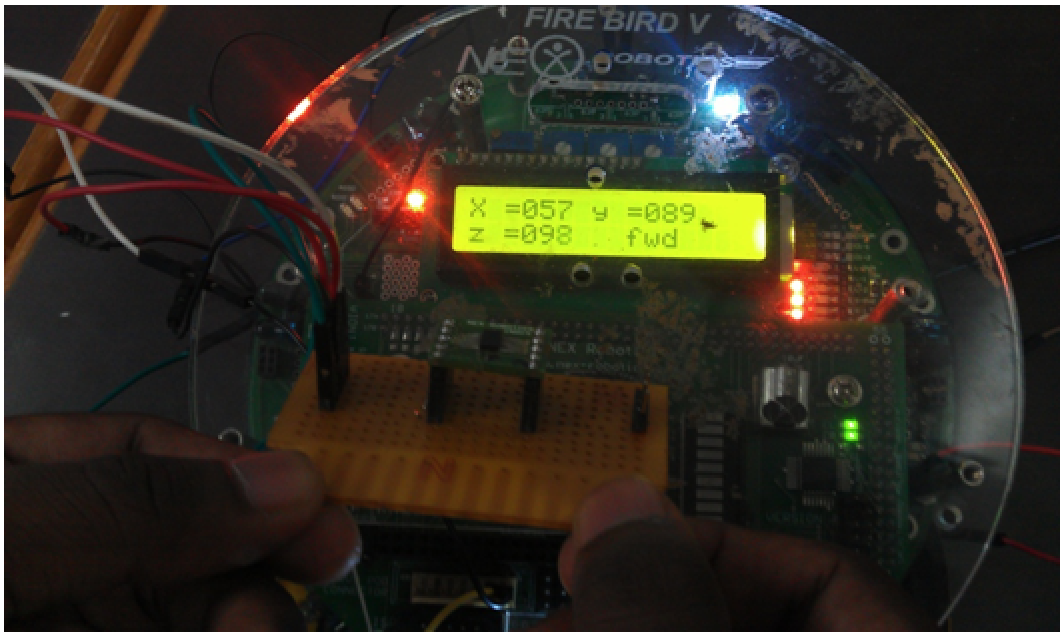


Figure 7: LCD Sample Output : Forward

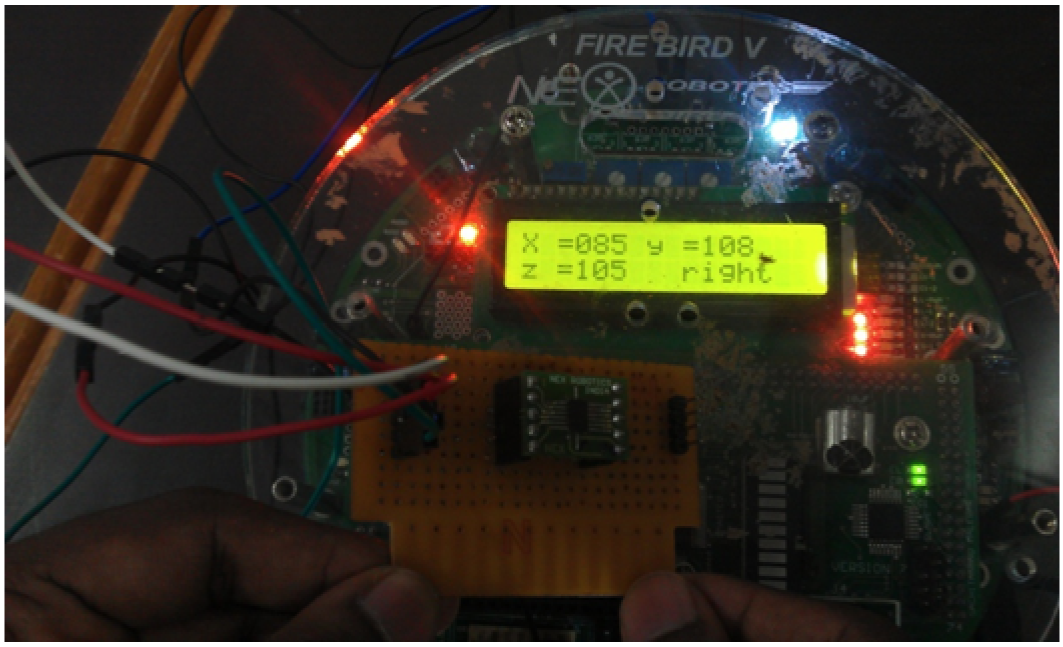


Figure 8: LCD Sample Output : Right

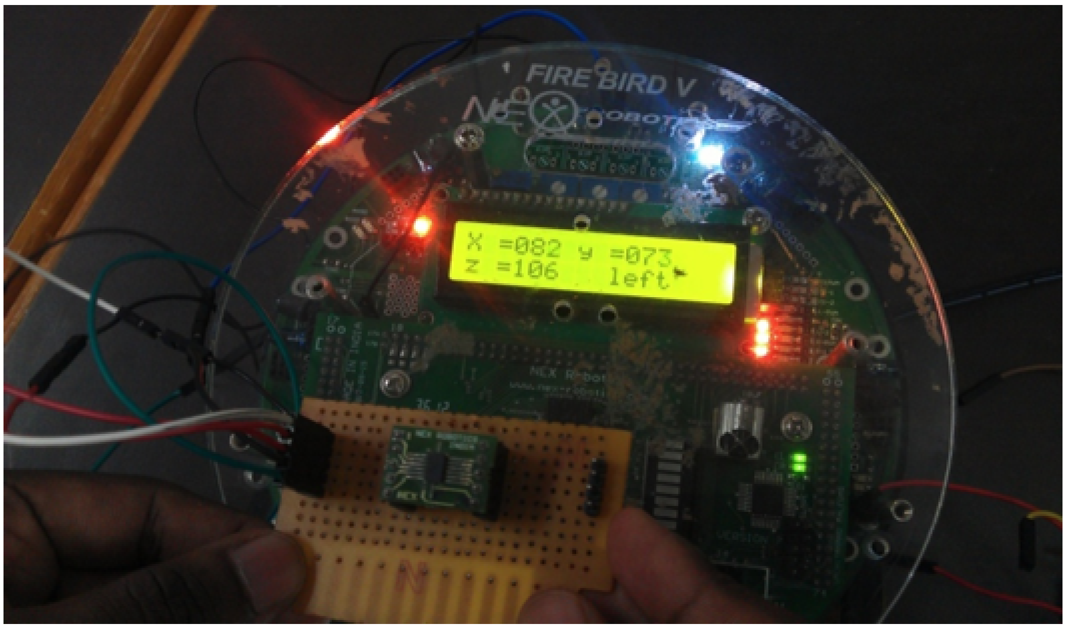


Figure 9: LCD Sample Output : Left

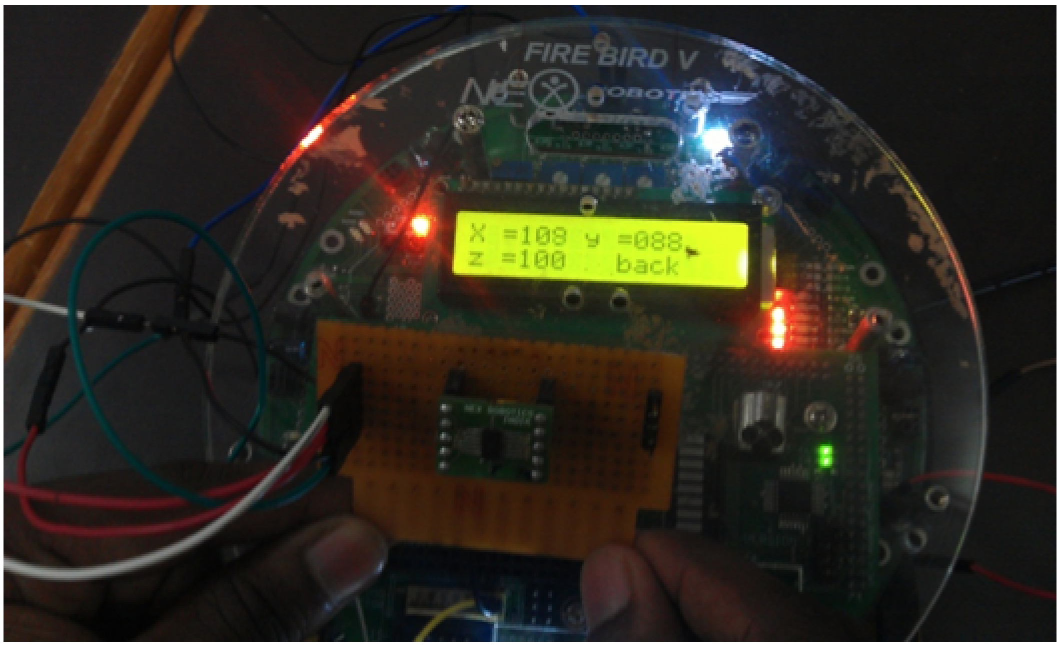
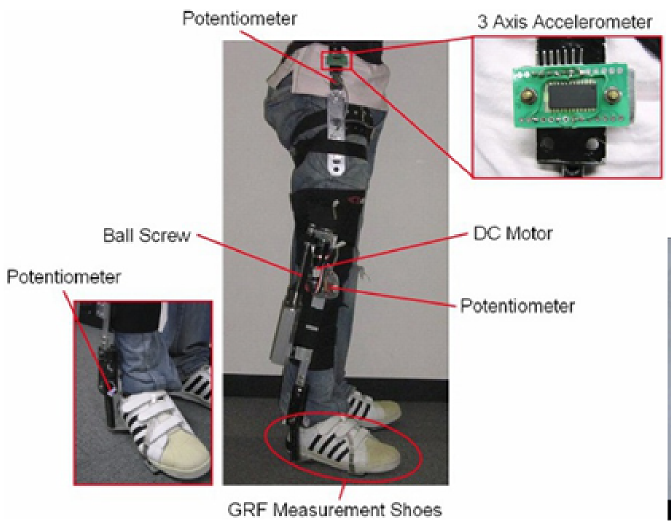


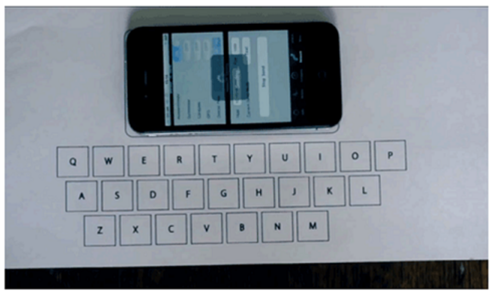
Figure 10: LCD Sample Output : Back

9 Applications

- Currently used in :
  - Mobiles and iPods for changing orientation from Landscape to Portrait mode or vice versa.
  - Gaming for motion sensing
  - Protect laptops and mobiles when under free fall
  - Vehicles to detect collision and deploy airbags.
  - As an alternative to spirit level for balancing.
- Future Uses :
  - Can be used for simulating driver training, in which a steering wheel including an accelerometer will turn the vehicle on the screen according to the tilt provided. This would enable safe driving practice with any dangers of collision.
  - For Robot Movement similar to the walking support system as shown in the picture figure 11a
  - Accelerometers measuring dynamic forces such as vibrations can be used for designing Virtual Keyboards as shown in 10b



(a) Walking Support System



(b) Virtual Keyboard Concept