EECE 375/474 Midterm Report

The iFit™ Personal Fitness Advisor

GROUP #7

Yidong Jiang
Biying Liang
Shun Zhi Zhang
Bing Liang
Ying Yin
Andrew C. H. Wong
Wen Jia Pan

Submitted:
February 14, 2007

ABSTRACT

The iFit™ Personal Fitness Advisor aims to monitor several critical human vitality factors and use this to provide a user with a meaningful interpretation of his/her current fitness status. Such factors include the user's body fat content and its rate of change, daily walking and running distance, overall daily activity, and total calories burned. The package will consist of three separate but essentially interconnected devices along with a graphical computer interface. The first component, a set of digital calipers, will be used to take skin-fold measurements required for the calculation of body fat content. The measurements will be transferred (by means of a cable) to a central wristband unit — the second component — which will store the data, perform necessary data manipulation and calculations, and display the results via an LCD screen. The third component will be a cutting-edge device, consisting of a lower-leg strap that in turn consists of an accelerometer, a microcontroller and a Bluetooth RF data transmission module. The processed data from the accelerometer will again be transferred to the wristband unit, but wirelessly. Finally, since the wristband unit can only display a limited amount of information, all accumulated data can be transferred wirelessly to any computer terminal with a Bluetooth connection and has the iFit™ software, which will perform a more in-depth analysis of the data and provide a variety of visual displays.

TABLE OF CONTENTS

1.0 INTRODUCTION

Obesity is a widespread concern in North America and also in many other parts of the world. Obesity can lead to diseases such as coronary heart disease, high blood pressure diabetes, congestive heart failure and strokes. As such, it is often desirable to have some form of guidance on our side to help us maintain a healthy lifestyle. In addition, losing weight and having a slimmer body are a popular trend because being slimmer is perceived as beautiful and desirable as depicted through media. However, losing weight is not an easy task. People usually cut down their diet, but weight can only be reduced when daily calories burned is greater than the calories intake. Hence exercising is a key component. The problem is that people usually find it difficult to keep on their exercising plan as they are discouraged by the minimal change in the results. This is because exercising may increase one's muscle content while reducing the fat percentage resulting in small weight changes. While seeking shortcuts, people may even go for plastic surgery or take slimming pills in order to lose weight, although these methods are detrimental to the body.

The goal of the project is to design and construct a set of digital devices that helps people lose weight and increase fitness in a healthier and more efficient way. The objective of the Personal Fitness Advisor (iFit™) will be to monitor the user's daily activity (walking/running distance and time) and the approximate calories burned, and to measure and record the user's body fat composition over time. All the data will be transferred to the computer to produce graphs that can give advice about the fitness level of the person.

This report will explain the background, design requirements, project objectives and our proposed approach to the project. It also provides some preliminary test results and project assessment, as well as a summary of individual team member contributions up to date.

2.0 PROJECT BACKGROUND

Various methods to measure body fat content are currently available, including hydrodensitomeny (underwater weight measurement), ultrasound, manual measurement using calipers, and bioelectrical impedance analysis (BIA).

BIA-based devices, although convenient, are not very accurate. This is because the body's bioelectrical impedance depends very much on body's water content. Hence, readings usually fluctuate greatly[1]. The ultrasound method is usually done clinically using ultrasound B-scan. There is one paper (by D. Black, J. Vora, M. Hayward and R. Marks) that describes the use of an A-scan to measure fat thickness with 15 MHz ultrasound[2]. However, the transducer for producing high frequency ultrasound (greater than 3 MHz) is very expensive, and lower frequency ultrasound, although cheaper, may not have the resolution to measure the fat depth since a certain level of sensitivity is required to differentiate between fat thicknesses. The ultrasound method is not convenient either as one still needs to measure fat thickness at three different places.

According to a fitness website, "[t]here is only one true way to get an accurate measurement of your body fat percentage, and that is by using something called body fat calipers."[3] Calipers are measurement devices used to measure thickness. The mechanical caliper is very inconvenient to use since one has to measure, record and calculate the fat percentage him/herself. There exist a few digital calipers in the market. However, they only show a single number and lack the ability to show the trend of a person's fat composition over time. Being able to monitor the body fat percentage over time will give people a better idea about their progress and can actually encourage them to keep up with their training and diet plan. Our digital caliper will be able to store data and transfer data to a computer so that a person's fat percentage can be plotted over time and monitored.

---

[1] SPARKPEOPLE. 2006. "Body Composition".
Available: http://sparkpeople.com/resource/reference_composition.asp

[2] Black, D., J Vora, M Hayward and R Marks. Dec 7, 1987. "Measurement of Subcutaneous Fat Thickness with High Frequency Pulsed Ultrasound: Comparison with a Caliper and a Radiographic Technique".

[3] The Intense Workout. 2006. "Body Fat Percentage – Why It's Important, Calipers, Calculators and MORE!". Available: http://www.intense-workout.com/body_fat.html

Another part of our package is the personal activity monitor that measures the daily personal activity in terms of distance traveled and walking/running hours. The existing similar devices like pedometers only track the number of steps. Our device will record the daily walking/running hours to calculate the approximate calories burned and transfer the data to the computer so that these can be compared with the daily fat percentage. As a result, people can have a very direct and visual observation about the relationship between their body fat content and their daily activity level. This will help to motivate them to adopt a healthier life style. The computer interface will also show some advice about the user's fitness level according to the data.

3.0 DESIGN REQUIREMENTS, GOALS, AND APPROACH PURSUED

3.1 Digital Caliper

Our caliper will feature a built-in microprocessor, which calculates and displays body fat percentage directly on its LCD display, thus eliminating the necessity to add the skin-fold readings manually and compute the body fat percentage from formulas or tables.

There will be six input buttons on the device. They are “+”, “-”, “save”, “on/off”, “reset”, and “cal” (as calculation). Once the user presses “on/off”, the caliper will turn on. The device will display some instructions requesting the user to input his/her age and initiates the input to “00” if s/he is a first time user. Then the user will press “+” or “-” to input his/her age and will press “save” to save the value.

Note that once the user’s age has been saved, it will be the default age value used later on to calculate the fat rate. Now when the user turns on the device, s/he will see some instructions displaying default age. To reset the value, the user can press “reset” after the device is on.

When the device is on, the user will place the caliper to wherever s/he wants to measure the fat. The user will place the skin-fold in between the two legs of the caliper and press “cal” to ask the device to measure the thickness of the skin-fold. The value of the thickness of the skin-fold will be sent back to the microprocessor and used to calculate the fat rate. Finally, the fat rate will be displayed on the LCD.

3.1.1 Look and Design

We purchased a caliper model as illustrated below. We will make our modification based on this model. The joint of the two legs will be cut and extended and connected to a stepper motor (this will be discussed in more detail in the motor section).

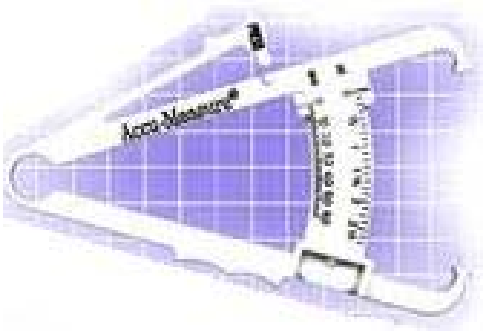


Figure 1. Caliper Model

A spring connection will be set up in between the two legs of the caliper. A pressure sensor will be placed at the upper red spot, as illustrated below. This pressure sensor will be treated as the input sensor. Another pressure sensor will be placed at the tip of one of the legs and this will be the feedback sensor.

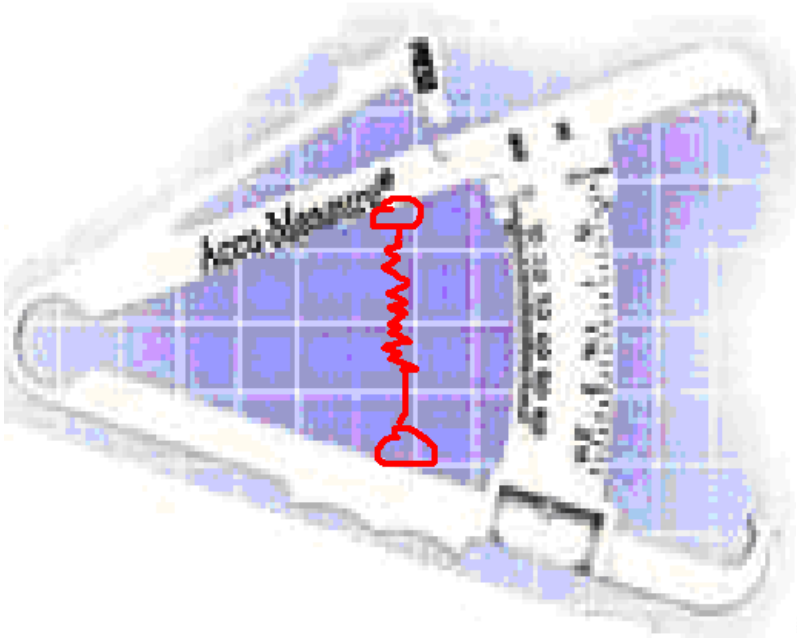


Figure 2a. Pressure Sensor Placement

Figure 2b. FlexiForce Pressure Sensor

We will use FlexiForce® A201- 1lb sensors for both the input and the feedback sensors. As illustrated in Figure 2b, the sensor is located at the tip of the connector. The three pins at the end are used to connect to a read-out circuit.

3.1.2 Sensor Circuit Design

The pressure sensors will be connected to a read-out circuit, which will convert the change in force (in resistance) to the change in voltage. We will use the recommended circuit for the FlexiForce® sensor, as illustrated in Figure 3.

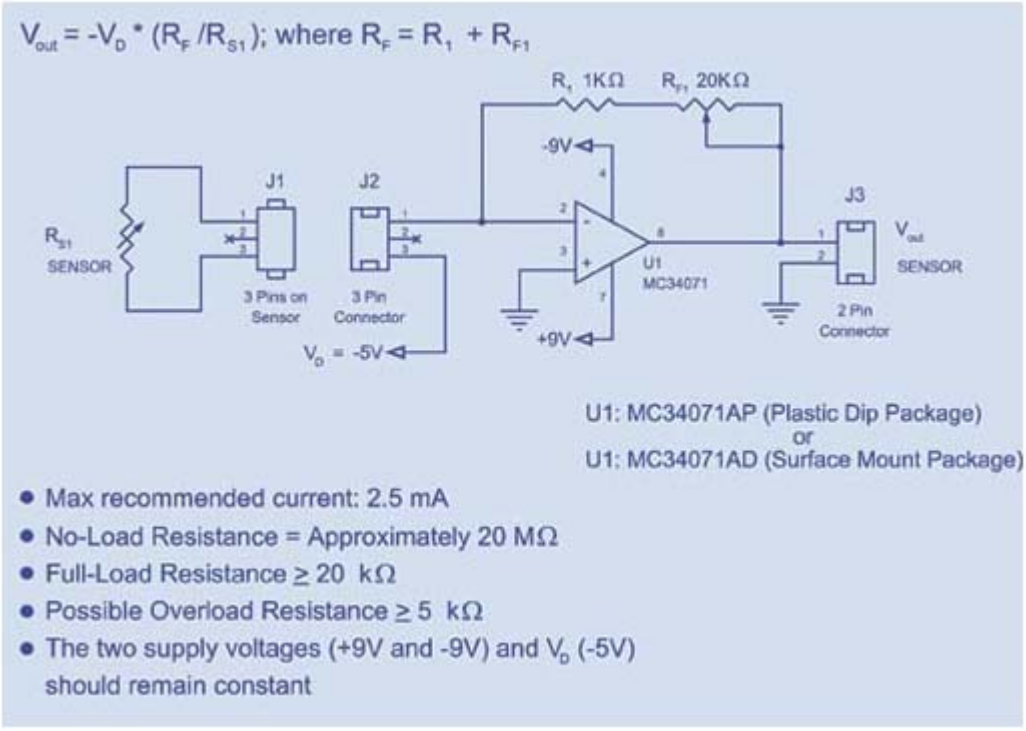


Figure 3. Force-Voltage Circuit

The above circuit is primarily used for reading force input, which will be used later to calculate the %body fat. The feedback circuit will be based on this circuit with some modifications. The feedback is used to control the adjustable legs of the caliper. Once the feedback indicates that appropriate force has been applied, the MCU will stop the motor.

3.1.3 Assumptions and Fault Toleration

We have not built the circuit yet. Hence, we are not sure about the sensitivity for a 1lb sensor. However, there is a sensitivity graph on the FlexiForce® website, which indicates the sensitivity of a 100lb sensor, as illustrated in Figure 4. We will make the assumption that the 1lb sensor will behave in a similar way.

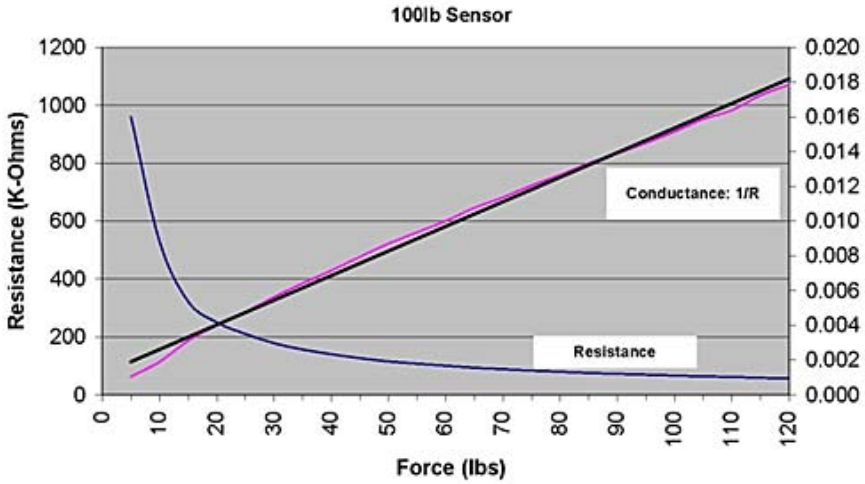


Figure 4. Sensitivity graph for a 100lb force sensor

A force-voltage diagram is derived based on the sensitivity graph of the 100 lb force sensor. We will assume for best performance that our MCU will be able to read a max 75% change and 25% voltage change for every unit change in force.

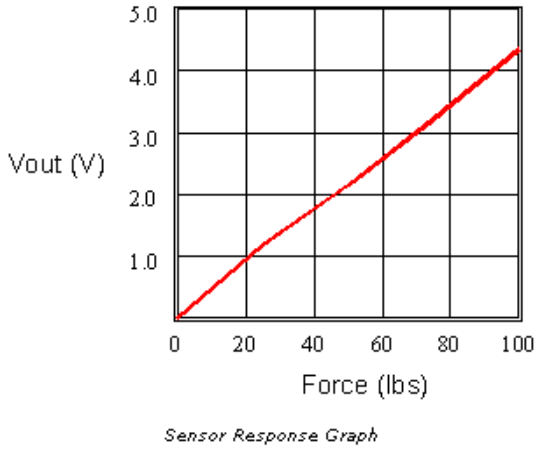


Figure 5. Force-Voltage diagram

3.1.4 The Stepper Motor

The main reason that we chose a stepper motor is that we need some method to measure how much the motor turns. There are two coils in a stepper motor. If both coils are driven with square waves which have a 1/4 phase relationship between them, the motor will rotate.[4] A transition of either square waves can cause rotor to move a small mount as a "step" as shown below.

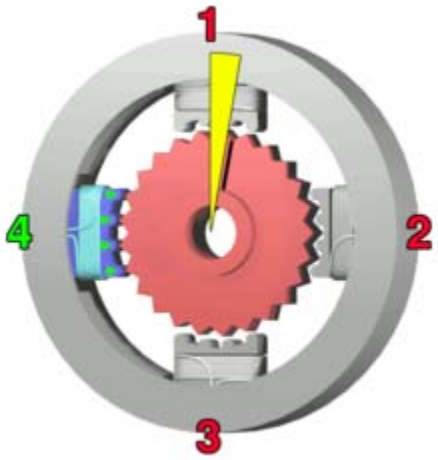


Figure 6. Stepper Motor[5]

There are two basic operation modes of a stepper motor. The commonest way to move the rotor of a step motor is called "full step". In the mode, the rotor rotates 90 degree in each step. The stepper motor may also be operated with half steps, which occurs when the rotor is moved 45 degree.

To control the motor using microcontroller, we require a circuit to transfer the microcontroller output signal also to protect the microcontroller. The method to accomplish this includes an H-bridge circuit. The complete schematic for the bridge is shown below.

---

[4] Freescale Semiconductor. 2007. "Stepper Motor".
Available: http://www.freescale.com/webapp/sps/site/overview.jsp?nodeId=02nQXGrrlPZh3C
[5] Wikipedia. 2007. "Stepper Motor". Available: http://en.wikipedia.org/wiki/Stepper_motor

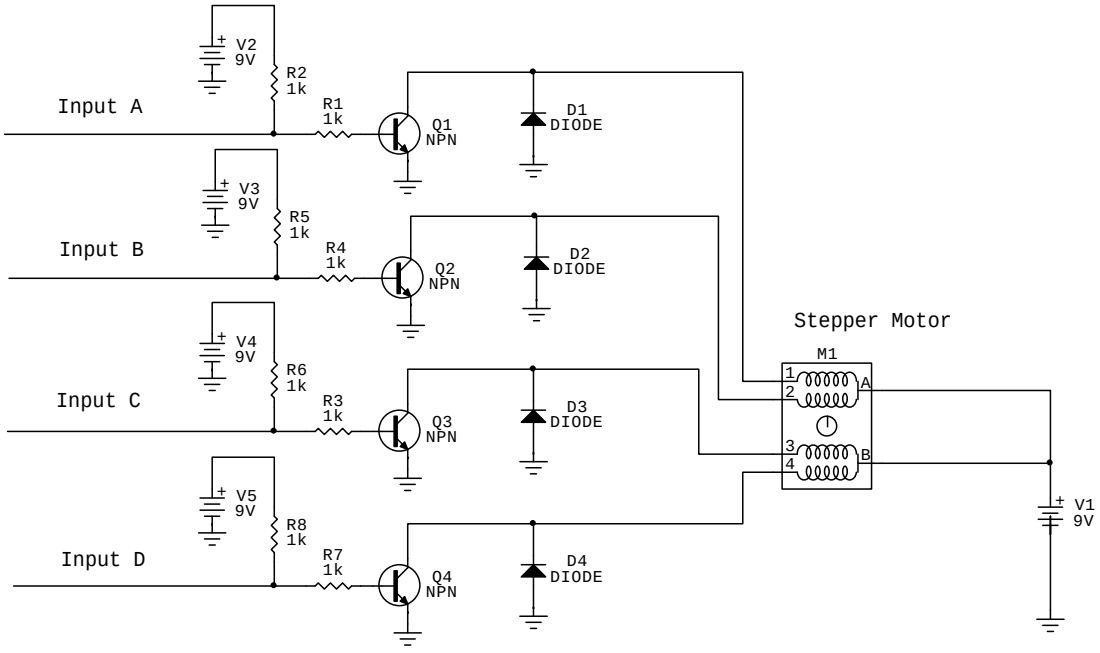


Figure 7. H-Bridge Driver

It consists of four NPN transistors which are used as digital switches to control the current direction of the motor. Different voltage on input A, B, C, and D can turn on different pair of transistors to drive the motor in different function modes. The sequences for clockwise rotation are shown in the following tables.

Step	A	B	C	D	Value
1	off	on	off	on	0101
2	on	off	off	on	1001
3	on	off	on	off	1010
4	off	on	on	off	0110
1	off	on	off	on	0101

Table 1. Full-Step Sequence for Clockwise rotation

Step	A	B	C	D	Value
1	off	on	off	on	0101
2	off	off	off	on	0001
3	on	off	off	on	1001
4	on	off	off	off	1000

5	on	off	on	off	1010
6	0ff	off	on	off	0010
7	0ff	on	on	off	0110
8	0ff	on	off	off	0100
1	0ff	on	off	on	0101

Table 2. Half-Step Sequence for Clockwise rotation

By reversing the order of the sequence, we can switch the motor into counter-clockwise rotation mode. However, to realize both rotation modes requires that the order of the sequence needs to be preserved if the motor stops for a while. Therefore, the restart sequence must be the next sequential step following the last step used. In addition, four diodes are introduced to catch the back voltage generated by motor's coil during the switching on and off. Otherwise, it is possible for this flyback voltage to be higher than the supply voltage and it results in burning the transistors.

3.2 Wristband / Data Processing Unit

The data processing unit consists of four main components: a microcontroller, an LCD display, a wireless communication unit and a user control panel. When the caliper is plugged into the processing unit, data is collected from the caliper and stored into the built-in flash memory of the microcontroller. When the accelerometer module's data transmitter is switched on, it will send its data to the processing unit using wireless communication. This data is transferred to the microprocessor and stored. The microcontroller is also responsible for displaying numeric values upon the user's request.

3.2.1 Microcontroller

We have decided to use Atmega644V MCU in data processing unit, instead of the Atmega32 which was mentioned in the design document. The main reason is that Atmega644V features a 64 KB in-system programmable flash memory, which is twice large than the Atmega32. Since the data processing unit is the central unit that will be responsible for manipulating user interface, data processing and

storage, a larger programmable flash memory will allow us to be more flexible in the programming. Moreover, this MUC features 2K bytes EEPROM, 4K bytes internal SRAM, a JTAG programming interface, low power consumption (240 μA @ 1.8V, when active at 1MHz), a on-board 10-bit ADC and six sleep modes for power conservation. The advantages of selecting Atmega644V over other chips such as the Motorola 68HC11 or the Intel 8051 become clear.

3.2.2 LCD

The LCD that we will use is the Varitronix/Part#: MDLS16265-SS-LV, instead of the Optrex/Part#:73-1048-ND which was proposed in the design document. The reason is that this Varitronix's LCD uses a Hitachi controller and CodeVision is only compatible with the Hitachi LCD HD44780 controller. According to the report of previous team, this LCD works well with CodeVision AVR code generator.

3.3 Activity Monitor

The iFit™ activity monitoring module will be comprised of three main components:

- An accelerometer for motion tracking
- A microcontroller for data collection and processing
- A Bluetooth RF module for wireless data transmission

These components will be combined into compact, light-weight packet that will take the form of an ankle band, to be worn just above the ankle of the user.

3.3.1 Accelerometer

A variety of accelerometers are currently in production, with different sensitivities, acceleration detection ranges, power consumption and temperature tolerance ranges. Each accelerometer is designed to measure acceleration in one, two, or three axes of motion. In general, there is a tradeoff between accuracy and power

consumption for these devices, something that must be taken into consideration when designing portable devices.

Initially, we decided to use the ADXL330 accelerometer, a "small, low power (180µA at 1.8V) 3-axis iMEMS® accelerometer complete with signal conditioned voltage outputs on a single monolithic IC."[6] Although this accelerometer has greater sensitivity (300mV/g), its package (LFCSP) turns out to be very difficult to be soldered. Hence, we ordered another accelerometer MMA3201 which is in a SOIC-20 package and is easier to solder. This accelerometer has a sensitivity of 50mV/g and a range of 40g in both X- and Y- axis. The first one can still be a backup.

In order to test the accelerometer, the surface mount chip needs to be soldered to the PCB. We designed a very small PCB (Figure 1) just for the accelerometer with header pins extending out for easy testing. As the PCB is ready yet, we have not had a chance to test the accelerometer. However, we did as much preparation as possible so that when the PCB comes and the accelerometer is soldered, we can test it immediately. Figure 2 shows the accelerometer connection diagram.

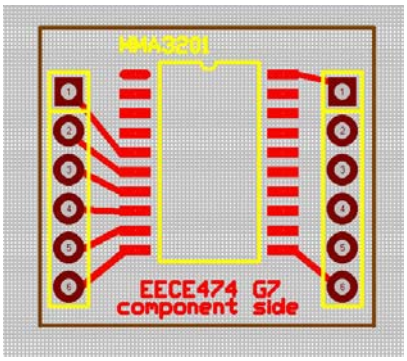


Figure 8. PCB for MMA3201 Accelerometer

[6] Analog Devices. 2007. "ADXL330".
Available: http://www.analog.com/en/prod/0%2C2877%2CADXL330%2C00.html. Accessed : 19 Jan 2007.

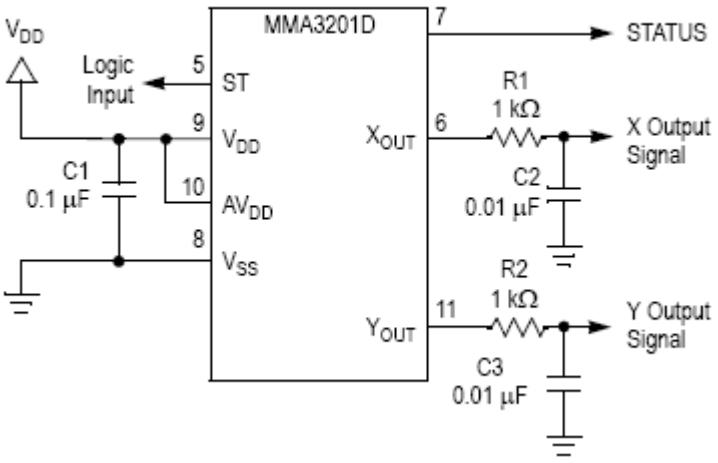


Figure 9. Accelerometer Connection Diagram

V_{DD} for MMA3201 is typically 5.0V. We use the LM340T5 voltage regulator which can deliver over 1.0A output current to provide the constant voltage.

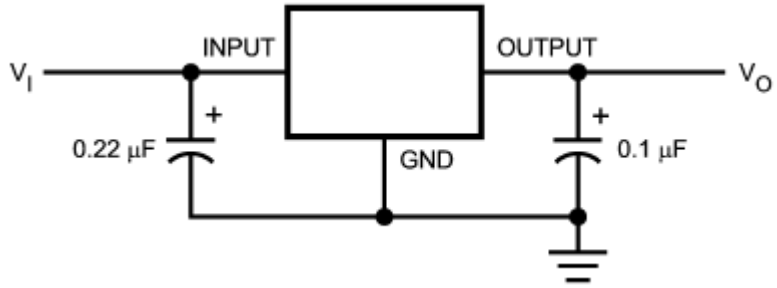


Figure 10. Circuit for Voltage Regulator

Acceleration is the time rate of change of velocity (Equation (1)). It is the slope of

$$a = \frac{\partial v}{\partial t} = \frac{\partial^2 x}{\partial t^2} \qquad (1)$$

velocity versus time graph. The accelerometer measures the accelerations of the object it is attached to in X-axis and Y-axis in terms of g (the acceleration due to gravity which is approximately 9.81m/s^2. The output from the accelerometer is linear which means the output voltage is directly proportional to the acceleration. This makes calibration and calculation much easier.

After the analog voltages output from X_{OUT} and Y_{OUT} are converted to digital values in the microcontroller, the acceleration can be calculated by Equation (2)

$$\text{Acceleration} = (V_{OUT} - V_{OFF})*g/0.05, \qquad (2)$$

Where V_{OFF} is the output signal at zero g which is $0.50V_{DD,}$ and 0.05V/g is the sensitivity.

In order to measure daily activity of a person and calories burned, we need to determine how long the person walked or run in terms of time in a day. We found a website called "caloriesperhour.com" (http://www.caloriesperhour.com/index_burn.html) that provides statistics for the amount of calories burned for different activities. Since the amount of calories burned is proportional to the speed of walking or running and time, we need to calculate speed from acceleration. As velocity is the time integration of acceleration, we can find velocity using trapezoidal method of numerical integration. Two velocities in X and Y directions will be obtained effectively giving a vector. The speed is the magnitude of velocity which is calculated as

$$\text{Speed} = \text{sqrt}\,(V_X^2+V_Y^2). \qquad (3)$$

Depending on the level of accuracy we want in determining the amount of calories burned, we can category daily activities in a number of different levels. At the first stage, we will only consider walking and running. Later, we can fine tune this and add more categories.

One major consideration we need to take care of in measuring acceleration, and hence velocity is the effect of tilt on DC accelerometer. When a person is walking, the accelerometer at the ankle will not always be in the perfect orientation such that one axis is perpendicular to the gravitational force and the other is parallel to the gravitational force. By using integration and averaging we will try to reduce

the noise in the acceleration by finding the average acceleration for each step instead of the instantaneous acceleration.

Another problem needs to be considered is that we need to detect whether the acceleration detected is due to the movement of the person or due to the movement the vehicle if the person is inside it. We can detect this by checking the pattern of the acceleration variation because the two motions will have different acceleration profiles. Another method is to add a pressure sensor near the toe of the person to detect impact. Since we have extra pressure sensors, this should be a good alternative.

3.3.2 Microcontroller

This will be the second of two microcontrollers used in our package. The first, as explained earlier, will be an integral part of the wristband unit, responsible for receiving, storing, processing and transmitting various bits of data. The second MCU that we plan to use is the Atmel Corporation's ATmega32. This MCU features a 32 KB in-system programmable flash memory, 1KB of EEPROM, 2KB of SRAM, 10 MHz maximum operating speed, a 10-bit analog-to-digital converter and six sleep modes for power conservation. These features together comprise a favourable choice of an MCU for our purposes.

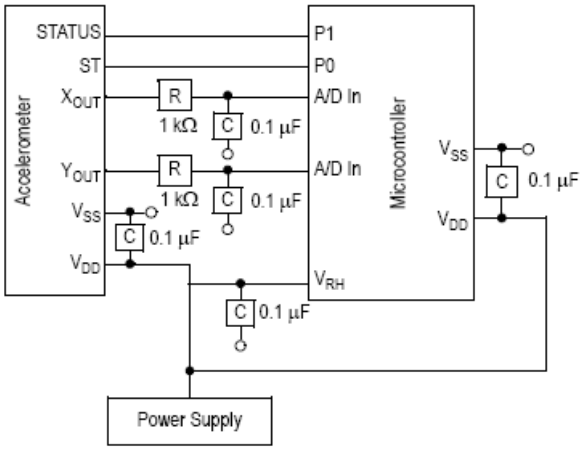


Figure 11. Interface between Accelerometer and Microcontroller

The two outputs from the accelerometer will be fed in the AD conversion input pins of the microcontroller. As the output voltages are in the range of 0-5V, no amplification is needed.

PA0 and PA1 pins are used for the input of the analog signals. AVCC and AREF pins are connected to Vcc. The ADC converts an analog input voltage to a 10-bit digital value through successive approximation. The minimum value represents GND and the maximum value represents the voltage on the AREF pin minus 1 LSB.

ADC is enabled by setting the ADC Enable bit, ADEN in ADCSRA. Voltage reference and input channel selections will not go into effect until ADEN is set. The ADC does not consume power when ADEN is cleared. So ADC should be switched off when entering power saving sleep modes.

The ADC generates a 10-bit result which is presented in the ADC Data Registers, ADCH and ADCL. By default, the result is presented right adjusted. The ADC is set to be in the Free Running mode, constantly sampling and updating the ADC Data Register. Using the ADC Interrupt Flag as a trigger source makes the ADC start a new conversion as soon as the ongoing conversion has finished. The first conversion is started by writing a logical one to the ADSC bit in ADCSRA. In this mode the ADC will perform successive conversions independently of whether the ADC Interrupt Flag, ADIF is cleared or not.

After the conversion is complete (ADIF is high), the conversion result can be found in the ADC Result Register (ADCL, ADCH). For single ended conversion, the result is

$$ADC = \frac{V_{IN} \cdot 1024}{V_{REF}}, \qquad (4)$$

where V_{IN} is the voltage on the selected input pin and V_{REF} the selected voltage reference. Hence, V_{IN} can be calculated from the ADC results using equation (4), and then acceleration can be calculated. Figure 5 shows the overall circuit diagram of the accelerometer and microcontroller.

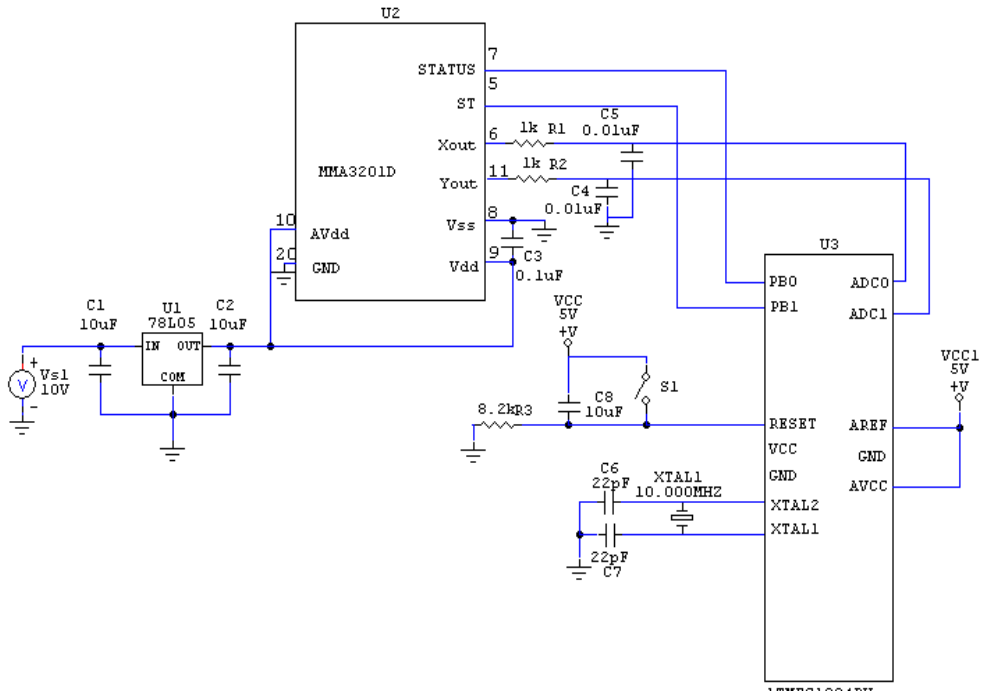


Figure 12. Overall Circuit for Accelerometer and Microcontroller

3.4 Bluetooth Data Transmission

Considering the expandability to interface with a computer, we decided that Bluetooth is an appropriate means for wireless data transfer. As there are many different types of Bluetooth chips/modules available, we narrowed down our Bluetooth module selection to the following criteria:

- Transmission range: There are three classes of Bluetooth chips with different transmission distance, class 1 – 100m, class 2 – 30m, and class 3 – 1m. Since we have one device on user's arm and the other device on user's leg, we need either class 1 or class 2 chip.

- Power consumption: our two devices will be powered normal batteries, and they are designed to work about 10 hours a day, depending on the user's activity, so we need to minimize Bluetooth power consumption. According class 1 specification, it consumes 100mW (20dBm) of power, and class 2 consumes 2.5mW (4dBm) of power. Thus, class 2 chip is appropriate for our application.
- Portability: again our iFit™ is designed to be carried every day, so we do not want any external antenna. The Bluetooth module should have a built-in antenna.
- Testability: we need to initialize and test the module on a breadboard before we solder to a PCB, so DIP package is the best one for us.
- Cost: Since we have only $400 budget, and we need two Bluetooth modules, it cannot be too expensive. We estimated the cost to be $50 per module.

After taking all the above criteria in consideration, we finally decided Parani-ESD200, manufactured by Sena, is an appropriate Bluetooth module. Figure 1 below illustrates the actual module that we will employ.

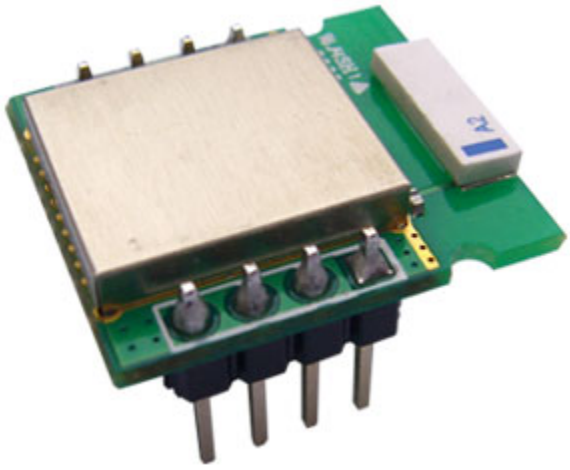


Figure 13. Parani-ESD200 Bluetooth Module[7]

In order to initialize and fully test the Bluetooth module, we need a jig board to connect to a computer. However, the jig board will cost us more than $100. We found reference on Sena.com website for how to make a RS232 interfaced jig board.

[7] SENA. 2007. Available: http://www.sena.com/images/product/industrial_bluetooth/esd200_l1.jpg.

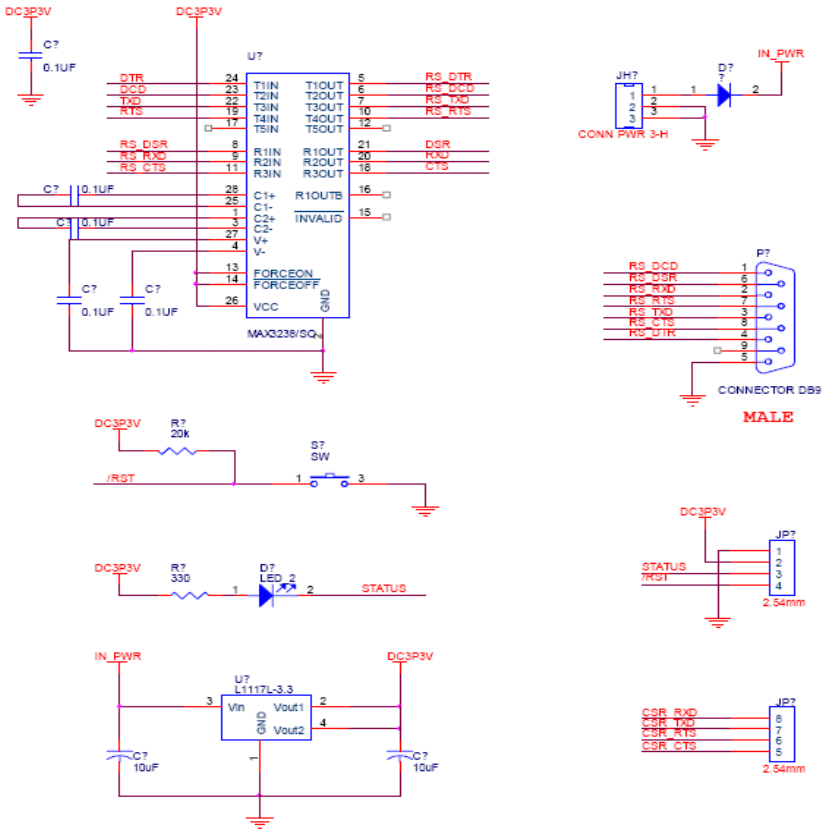


Figure 14. Jig board circuit diagram[8]

Thus, we will use the above circuit diagram as our reference jig board.

In order to connect Parani-ESD200 and our microcontroller, we will use UART (universal asynchronous receiver/transmitter) interface. UART has four pins, which are CTS (clear to send), RTS (ready to send), TxD (data output), and RxD (data input). Also, Parani-ESD200 has four other pins which are RST (reset), DCD (Bluetooth connection detect), VDD (3.3V) and GND. The connection between these pins and microcontroller is shown in the figure below.

[8] Initium. 2007. Available: www.initium.co.kr.

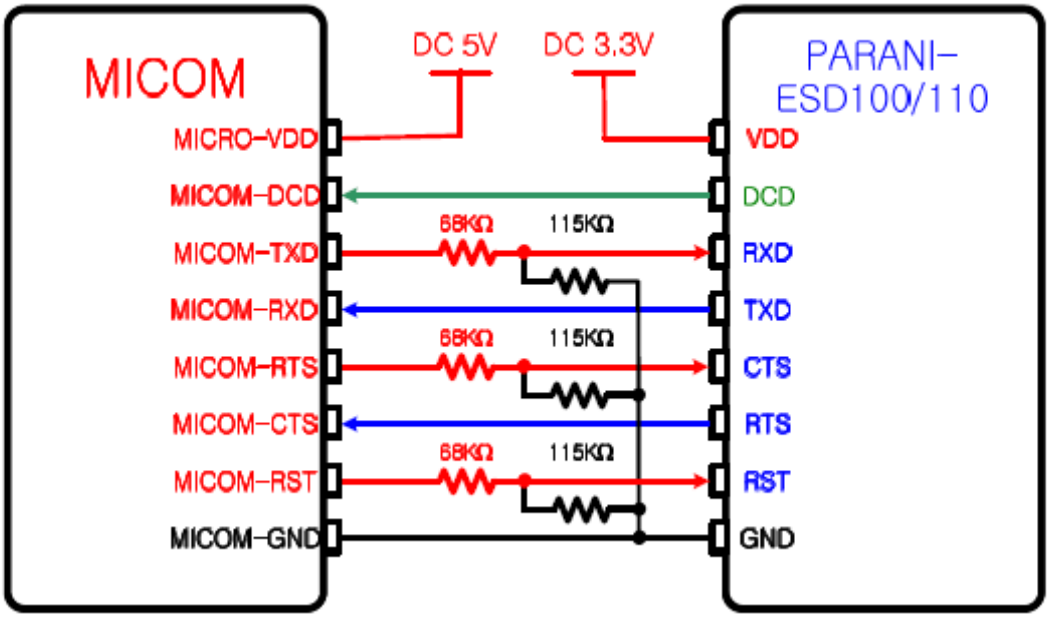


Figure 15. Connection between Bluetooth and microcontroller[9]

When choosing the programming language for our Bluetooth module, we prefer developing Bluetooth application in C instead of other high level languages which might not fit on our target device. Once we develop the code according to the desired functionality of our design, it will be uploaded to the microcontroller through serial port connection.

To initialize/setup Parani-ESD200 through jig board on a computer, we will use either standard AT command through Windows Hyper Terminal (Figure 4), or ParaniWIN provided by Sena, which is available for downloading on Sena.com. Probably, we will use ParaniWIN to configure Parani-ESD200 since it has GUI interface.

[9] SENA. 2007. "Parani-ESD".
Available: http://www.sena.com/products/industrial_bluetooth/oem_bluetooth_serial/parani_esd.

```
at

OK
at+btinfo?

000195000002,ESD100v1.0.0-000002,MODE0,STANDBY,0,0,NoFC

OK
at+btinq?

000B5320070E,Promi-MSP_20070E,020300

0009DD500027,LEECOM,1E010C

OK
atd000b5320070e

OK

CONNECT 000B5320070E
```

Figure 16. Example of AT Commands

In addition to the connection and configuration between one Bluetooth module and PC, communicating two Bluetooth modules is also required to meet our design specification. To achieve connection between two modules, one of the modules needs to be configured as a master and the other as a slave. One possible way to test this 2-module configuration would be done on two separate jig boards, each of which is connected to different COM ports on PC. We will then use ParaniWIN software to configure and connect two modules.

Since our design requires constant data transfer, we will configure the two modules such that they will automatically connect to each other when they are both powered up and in range.

3.5 Computer Connectivity and GUI

When the basic fitness data available on the iFit™ wristband is simply not enough, the user can turn to the iFit™ tracking software on his/her computer. The computer will receive a stream of raw data from the iFit™ wristband unit provided that the computer has Bluetooth connectivity (using an internal or add-on Bluetooth adapter), the iFit™

tracking software has been installed on the computer, and the "transmit" function has been activated on the wristband module.

3.5.1 The User Interface

Initially, we had chosen to design the iFit™ software as a C++ .NET application. This has certain benefits. For instance, a large part of the graphical user interface could be created using the .NET drag-and-drop libraries, and therefore we would only need to code the functionality behind all the interface components. In addition, the graphics library includes a handful of pre-defined objects such as pie charts that would be useful for our project.

However, .NET has its shortcomings, and for that reason we decided to switch to Java. The main shortcomings include the following:

- .NET is specific to the Windows operating system, so this would prevent us from expanding support for other platforms. This point may not be very important, since many Bluetooth dongles only support Windows as well.
- .NET includes huge blocks of computer-generated code, a lot of which is highly-unreadable. Modifications of such code often results in error. User-written code may only be inserted in specific areas, as we discovered, so although we successfully integrated the custom graphics functionality into the computer-generated GUI code, it was a lot more difficult than it would have been in other languages.
- Connecting to a Bluetooth device would require us to make use of the Microsoft WinSocks API. This functionality is embedded in the huge Microsoft Platform SDK package, which is poorly documented on the web since few people have attempted to take this path, probably due to similar reasoning.

The software will process the information in a variety of ways — exactly what will be calculated will be determined through further research — and present a series of meaningful graphic displays that can reveal a clear trend in the user's overall fitness. At the point, we have created a flexible foundation for our program, written in Java, which can be easily expanded upon and altered during the course of the project. The program interface will be made up of a number of components from the javax.swing package and managed by a CardLayout[10] manager so that new screens can be inserted into the program with a single "add" command. The following is the empty prototype program containing a single blank button.

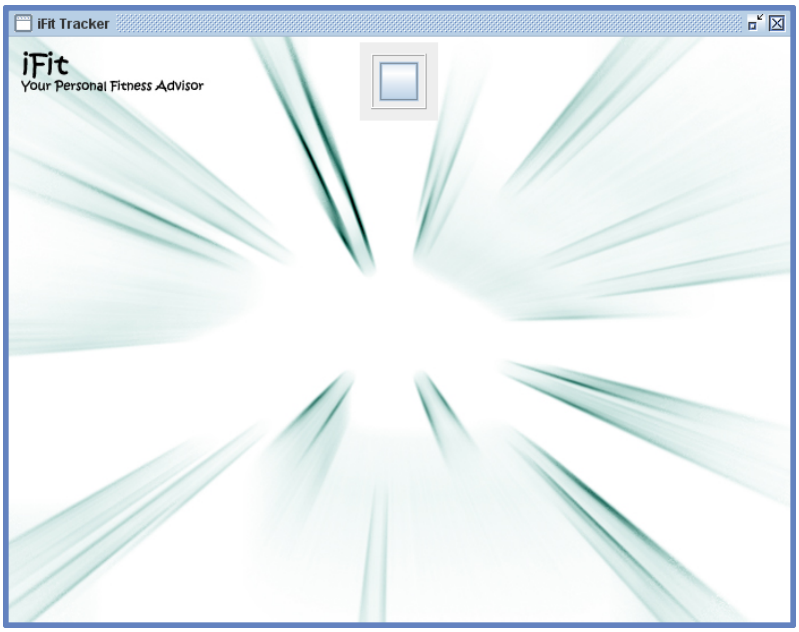


Figure 17. Empty Java Prototype GUI

The following is a mock-up of the iFit™ Tracker GUI created using C++ .NET, Photoshop, MS Excel and Paint. The final product, as mentioned above, will be written in Java and will contain more useful and detailed information in each section.

[10] Sun Microsystems. 2004. "Java 2 Platform Standard Ed. 5.0 API".
Available: http://java.sun.com/j2se/1.5.0/docs/api/.

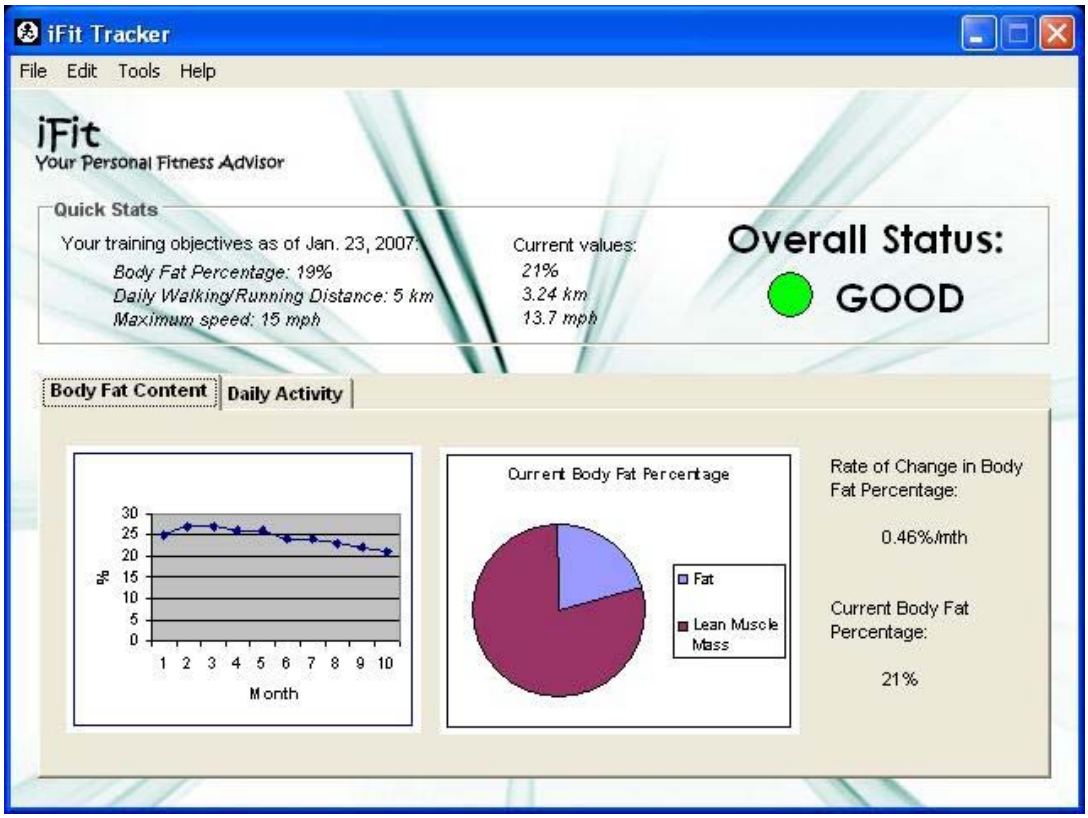


Figure 18. C++ .NET Prototype GUI resembling the final product

3.5.2 Computer Bluetooth Connectivity

The iFit™ analysis software will obtain its raw or pre-processed data through a Bluetooth connection between the computer and the wristband microcontroller/Bluetooth module. The procedure is simplified as follows:

1. Client (the microcontroller and Bluetooth module on the wristband unit) initiates transaction.
2. Receiver (the Bluetooth adapter or dongle on the computer) will detect and accept the request, and establish a Serial Port Profile (SPP).
3. iFit™ application will find and communicate with the appropriate serial port to obtain data.

The main idea is that the wireless serial Bluetooth connection can first be configured by existing software, such as BlueSoleil, so that it may be used by

applications as though a physical serial cable connected the devices at each end.[11] This software comes free with many Bluetooth dongles currently available in the market. The iFit™ application can then simply communicate with this serial port to receive any incoming data.

To communicate with serial ports in Java, we make use of the Java Communications (javax.comm) API. A high-level diagram of the API interface layers shows that this API provides a clean separation between the Java application and the physical ports.

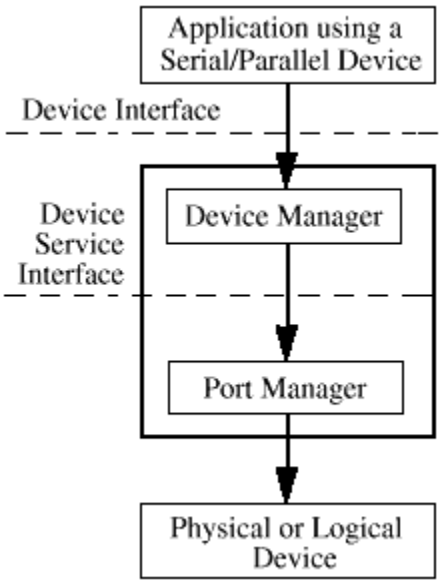


Figure 19. Java Communications API[12]

The API, shown as the black box in the preceding figure, "is a proposed standard extension that enables authors of communications applications to write Java software that accesses communications ports in a platform-independent way"[12].

[11] Mobile Fun. 2007. "USB Bluetooth Dongle".
Available: http://www.mobilefun.co.uk/download/manuals/BluetoothUSBAdapter.pdf.
[12] Java World. 2007. "Java gets serial support with the new javax.comm package".
Available: http://www.javaworld.com/javaworld/jw-05-1998/jw-05-javadev_p.html.

4.0 PROJECT FLOWCHART

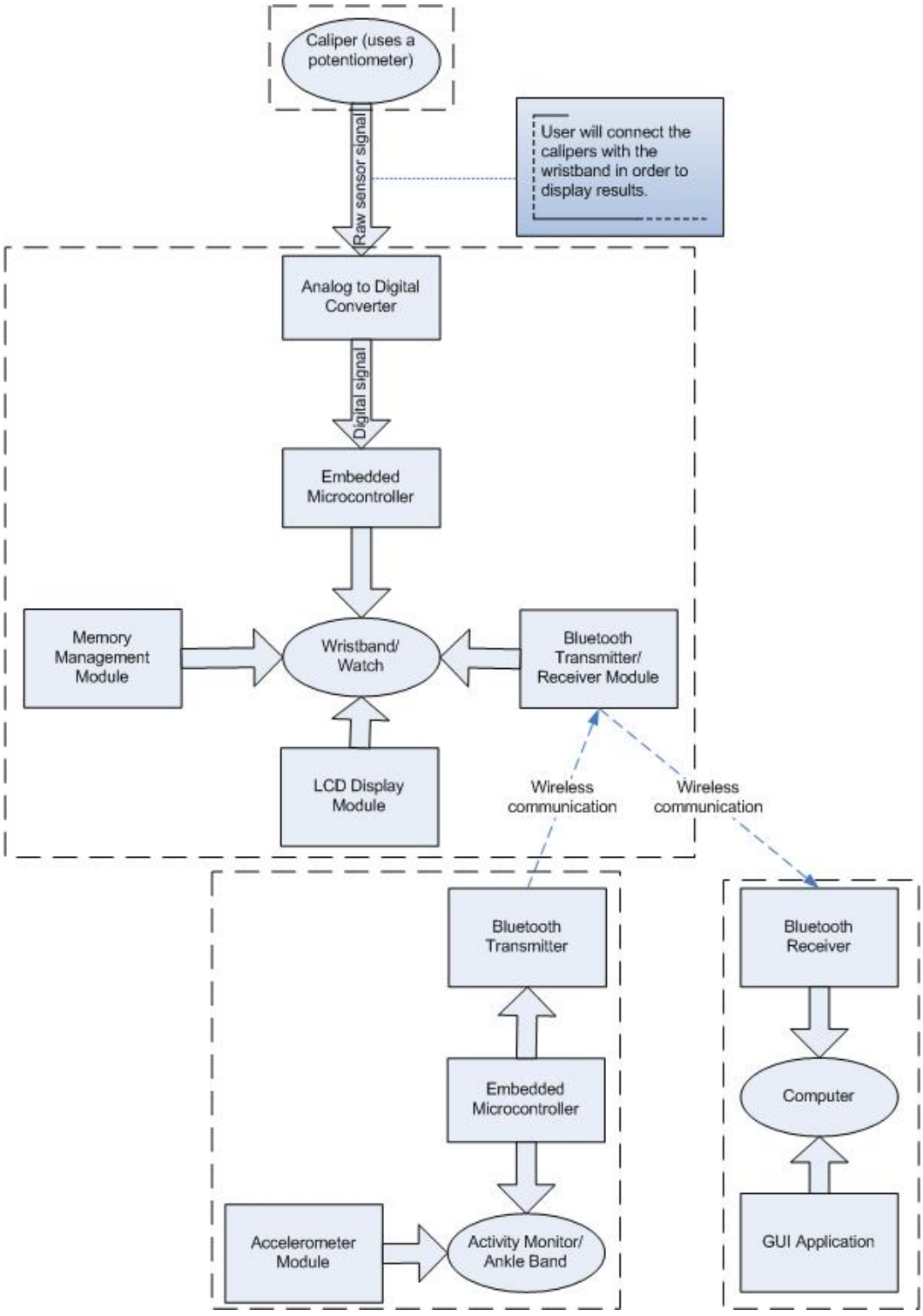


Figure 20. Modular Project Flowchart

5.0 SCALABILITY AND BUDGET OVERVIEW

5.1 Scalability

The iFit™ Personal Fitness Advisor will be a full-scale, independent, and fully functional portable device. Also, we are quite confident the project schedule and budget will fit within the given constraints. Therefore, scalability is not an issue in this project.

As for further upgrades, if time allows, we would like to use wireless communication to transfer data between the caliper and the wristband unit.

5.2 Budget

Here is a preliminary budget overview for the iFit™ Personal Health Advisor, considering only the parts we have ordered up to date.

Part Name	Supplier Name	Unit Cost	Quantity	Total Cost
Atmel Mega32 MPU	Digi-Key ATMEGA32-16PC-ND	10.14	1	10.14
Parani-ESD200-01 Bluetooth Module	SENA	55.00	2	110.00
Flexiforce Sensors A201-1 Sensor (4-Pack)	Tekson ZFLEXA201-1-4PK	77.00	1	77.00
Optrex LCD MODULE 16X2 STANDARD	Digi-Key 153-1078-ND	16.51	1	16.51
Analog Devices IC ACCELEROMETER 3-AXIS 16LFCSP	Digi-Key ADXL330KCPZ-RLCT-ND	14.37	1	14.37
Freescale Semiconductor SENS ACCEL XY AXIS +-38G 20-SOIC	Digi-Key MMA3201D-ND	19.04	1	19.04
IC REG LDO 2.85V 1A TO-263	Digi-Key RC1117M285-ND	1.19	1	1.19
IC AVR MCU FLASH 64K 40DIP	Digi-Key ATMEGA644V-10PU-ND	9.09	1	9.09
CRYSTAL 10.000 MHZ HC49/US	Digi-Key Part #: 300-8487-ND	0.87	1	0.87
PCBs	Alberta Printed Circuit Boards	-	-	-
			Total	258.21

6.0 PROJECT TIMELINE AND TASK ASSIGNMENTS

6.1 Project Timeline

TASK	DATE									
	January		*February*				*March*			
	21	28	4	11	18	25	4	11	18	25
Order the parts	▬▬▬▬									
Draw the PCB		▬▬▬▬▬▬								
Submit the first PCB			▬▬							
Submit midterm report				▬▬						
Integrate the parts					▬▬▬					
Test the device					▬▬▬▬					
Solve encountered problems							▬▬▬▬			
Prepare final report								▬▬▬▬▬▬		
Demonstrate the device										▬▬

Figure 21. Research and Writing Schedule for the Formal Report

6.2 Task Assignments

Team Members	Task Assignment
Bing Liang Biying Liang Shunzi Zhang	• Bluetooth Implementation • PCB Design and Layout
Wenjia Pan Yidong Jiang	• Caliper Design • LCD Programming • Microprocessor Integration
Ying Yin Andrew C.H.Wong	• Accelerometer module design and data interpretation • iFit™ computer software, including computer Bluetooth connectivity

7.0 RISK ASSESSMENT

Of many factors that may result in total failure of our design, the Bluetooth connection that enables data transfer between the accelerometer and the data processing unit seems most crucial and sensitive. Aside from the hardware integration with two different circuit boards, establishing the "handshaking" between the accelerometer and the data processing unit would also bring challenges to our design. Using serial-port cable connection for data transfer may back up our design in case the Bluetooth connection fails, since the circuit boards that we use support serial-port connection.

In this project, the accuracy of the caliper is very important. Therefore, we have two alternative plans to measure the thickness of skin-fold. The first one is using variable resistance method. We connect these two adjustable legs using variable resistance. The relationship between the resistance and the length of the metal will not be strictly linear. We will try our best to optimize the linearization. The second method is using a pressure sensor and a spring to measure the force exerted on the two legs. We will try the both methods to find out the more accurate one.

Processing the signals of accelerometer is difficult since our device is designed to monitor the user's physical activities; for instance, walking, jogging, and running. However, there are also activities that will cause the accelerometer to produce output even though it's not part of physical activities; especially, driving. Therefore, it is necessary to find a method to distinguish the activities.

8.0 RESULTS AND ANALYSIS

8.1 Digital Caliper

We have tried to plot an I-R^{-1} graph, in which current varies as the resistance changes. We feed 9 volts into the mock-up circuit and we vary the resistance using this constant voltage supply. It turns out that at some large resistance values, the current and R inverse has a linear relationship.

8.2 Activities and Calories Burned

By using the calculator on the caloriesperhour.com website, we obtained the following statistics.

Calories burned when walking where weight is 50kg:

149 calories in 1 hr (at 4 km/h)
170 calories in 1 hr (at 5 km/h)
217 calories in 1 hr (at 6 km/h)
295 calories in 1 hr (at 7 km/h)
395 calories in 1 hr (at 8 km/h)

Calories burned when running where weight is 50kg:

400 calories in 1 hr (at 8 km/h)
475 calories in 1 hr (at 9 km/h)
515 calories in 1 hr (at 10 km/h)
561 calories in 1 hr (at 11 km/h)
621 calories in 1 hr (at 12 km/h)

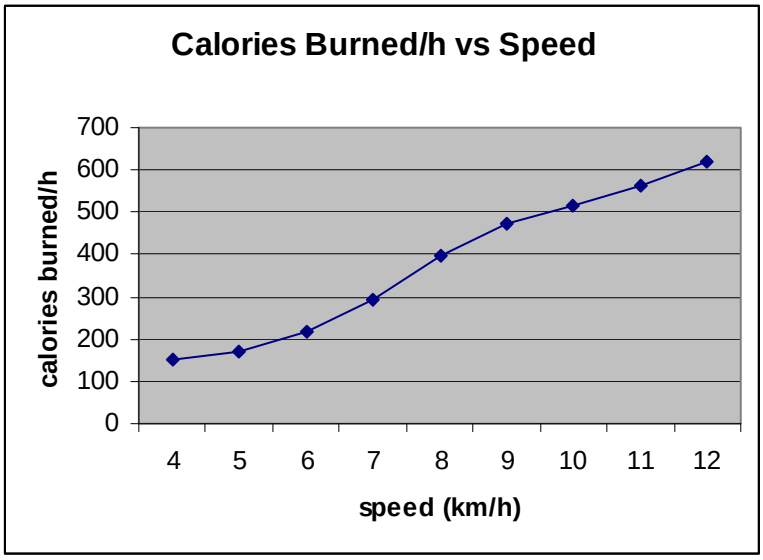


Figure 22. Calories Burned Per Hour as a function of Speed

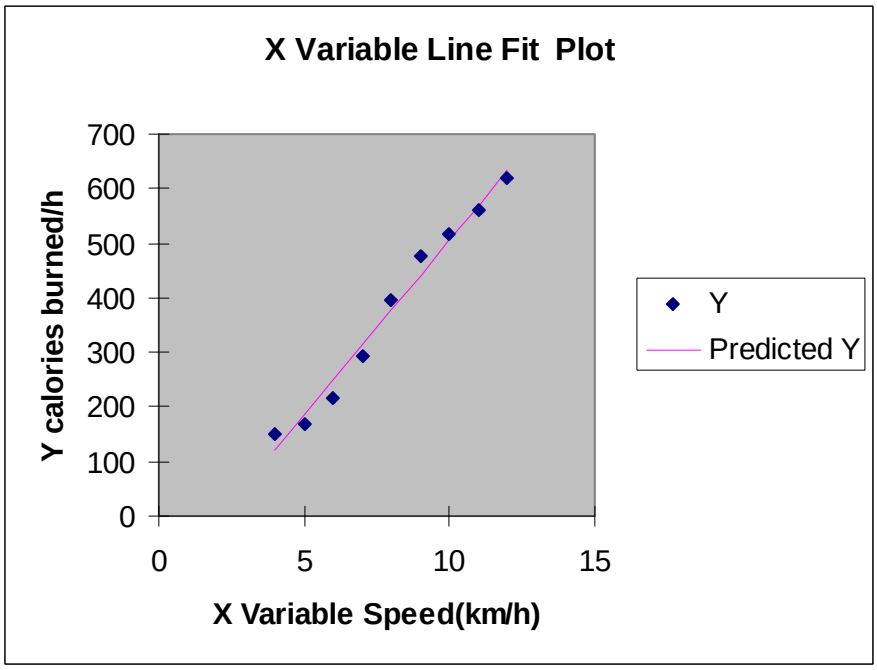
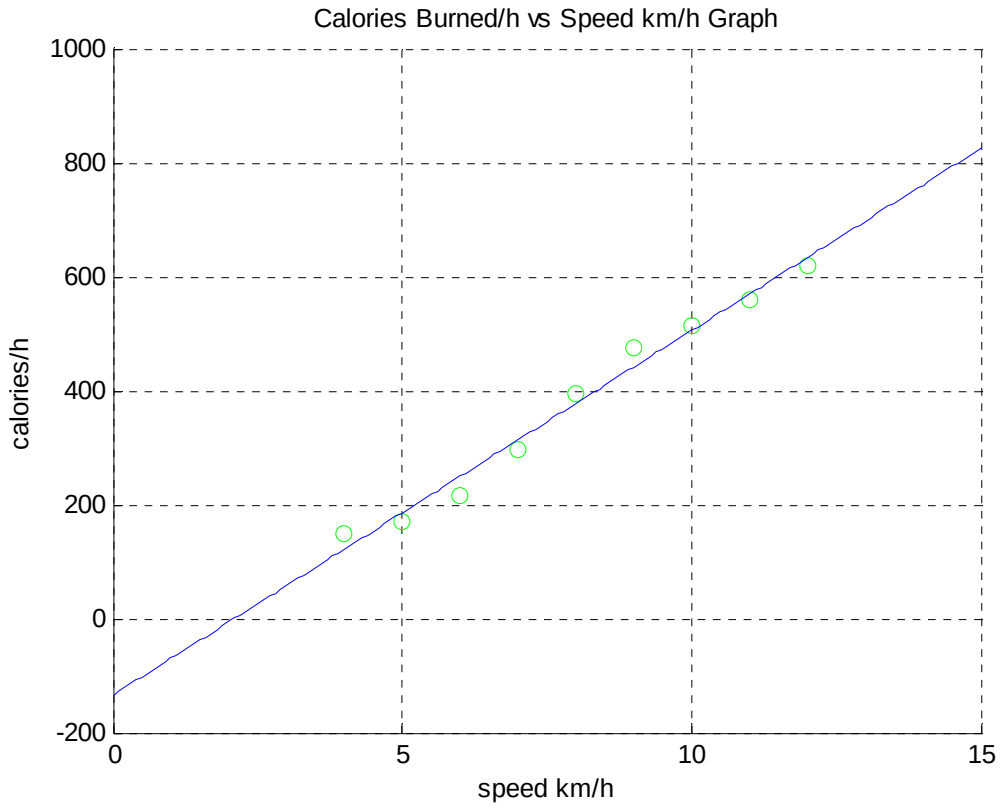
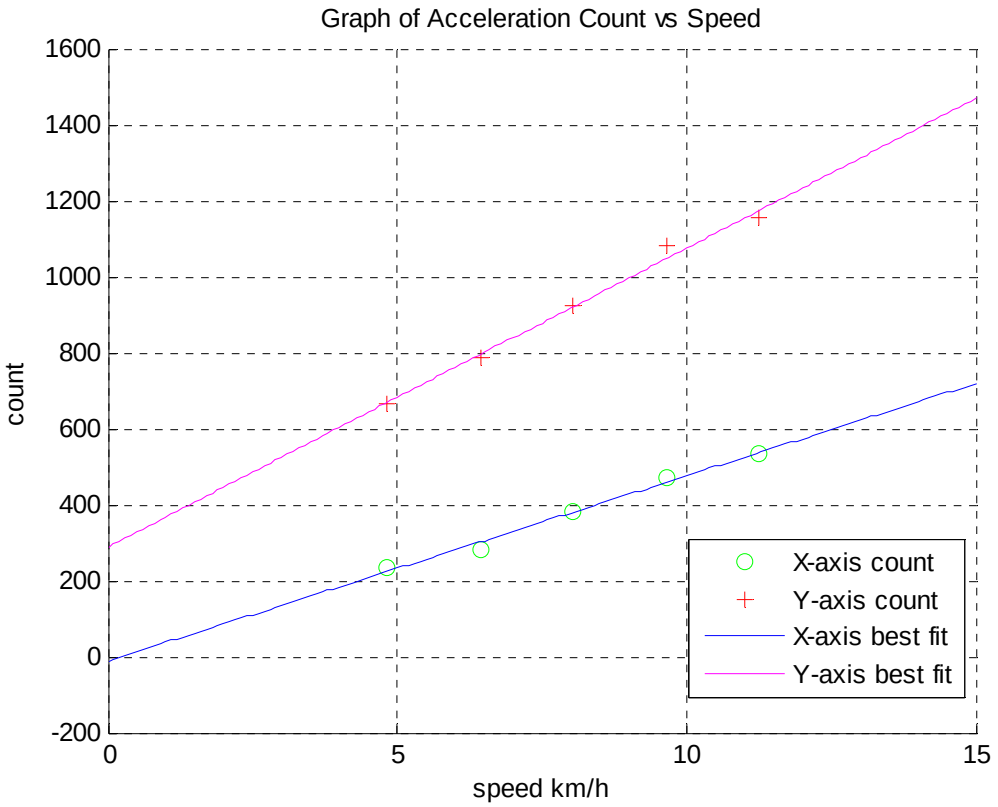


Figure 23. Linear Regression Plot of Calories Burned vs. Speed



gradient = 63.9500 y_intercept = -134.0444

Using linear regression, it is found that the relation between calories burned and the speed is very close to linear. Hence we can use this relation to calculate the daily calories burned while measuring the speed of the user throughout the day.



gradientx = 41.4721

y_interceptx = 54.6571

gradienty =78.6656

y_intercepty = 288.8000

8.3 Microcontroller and LCD

We have retrieved Atmega644V and Varitronix's LCD, Part#: MDLS16265-SS-LV. Before submitting the PCB and actually wiring the LCD, we planned to build a simple circuit to connect the microcontroller and LCD for testing. The following is the pin configuration of the LCD.

14 PIN CONNECTION	1	2	3	4	5	6	7	8	9	10	11	12	13	14	A	K
	VSS	VDD	V0	RS	R/W	E	DB0	DB1	DB2	DB3	DB4	DB5	DB6	DB7	LED(+)	LED(-)

Figure 24. Internal Pin Configuration of Varitronix LCD, Part#: MDLS16265-SS-LV

As shown, VDD is power supply for logic, V0 is power supply for driver and VSS is ground. Pin 7 to 14 are for data input/output. RS is register select input: high for data register, low for instruction register. R/W is read/write signal: high for reading mode and low for writing mode.

For the microcontroller, port B (PB7:PB0) and port C (PC7:PC0) are chosen for LCD. In detail, port D is used to provide 8-bit data and 3 bits of port C is used to control the LCD.

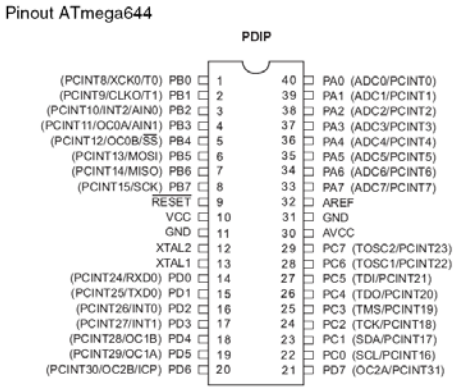


Figure 25. Pin Configuration of ATmega644

The corresponding pin connections are listed is Figure 26.

LCD		ATmega644
Pin# 1		
Pin# 2		
Pin# 3		
Pin# 4	<======>	Pin# 29
Pin# 5	<======>	Pin# 28
Pin# 6	<======>	Pin# 27
Pin# 7	<======>	Pin# 1
Pin# 8	<======>	Pin# 2
Pin# 9	<======>	Pin# 3
Pin# 10	<======>	Pin# 4
Pin# 11	<======>	Pin# 5
Pin# 12	<======>	Pin# 6
Pin# 13	<======>	Pin# 7
Pin# 14	<======>	Pin# 8

Figure 26. Connection between LCD and Microcontroller

In order to display some strings for testing, we tried the CodeVision AVR code generator to generate some LCD code on Port B of the AVR using the following code:

```
#include "mega64.h"

// the LCD is connected to port C and port B outputs
// 0×15 is the address of port B
#asm
.equ __lcd_port=0×15 ; port B
#endasm

// include the LCD driver routines
#include "lcd.h"
void main(void)
{
        // initialize the LCD for 2 lines & 16 columns
        lcd_init(20);

        // go to the first line of the LCD (x,y)
        lcd_gotoxy(0,0);

        // display the message
        lcd_putsf("Hi, Group 7");

        // go to the second line of the LCD (x,y)
        lcd_gotoxy(0,1);

        // display the message
        lcd_putsf("Hello World!  ");

        // stop here, loop now
        while (1);
}
```

9.0 INDIVIDUAL CONTRIBUTIONS

Andrew Wong

- Researched software connectivity for Bluetooth USB device using .NET and Java
- Researched and implemented simple Java RS-232 serial port detection program
- Coded iFit™ software GUIs and foundation classes in .NET and then in Java
- Carried out administrative tasks, such as organizing meetings, verifying PCB designs, coordinating parts orders and assembling reports.

Biying Liang

- Focussed mainly on Bluetooth module — learned how Bluetooth works, searched for various types of Bluetooth modules, read their datasheets and compared them
- Narrowed out Bluetooth module selection based on our design specifications and made the purchase for two Parani-ESD200 Bluetooth modules

Bing Liang

- Worked in conjunction with Biying in doing Bluetooth research, expanding our Bluetooth knowledge base and making a selection

Shun Zhi Zhang

- Researched motors to be used for automation of the digital caliper
- Created H-bridge driver circuit for the stepper motor

Wen Jia Pan

- Designed digital caliper
- Researched and purchased pressure sensors for the caliper, as well as components required for the recommended sensor circuit

Yidong Jiang

- Researched ATmega644 microcontroller and Varitronix's LCD, Part#: MDLS16265-SS
- Designed the testing circuit for the LCD and microcontroller.
- Wrote testing code to display some strings on the LCD
- Researched CodeVisionAVR C compiler and AVR Studio 4 compiler

Ying Yin

- Designed PCB for the surface mount accelerometers
- Designed and drew the circuit for the accelerometer and microcontroller, and tested part of the circuit
- Researched on calculations relating to accelerometer and AD conversions
- Installed C compiler for AVR microcontroller and wrote simple testing programs
- Researched on activities and calories burning relationships and found a website that provides useful information for this