CHAPTER 5

SYSTEM IMPLEMENTATION

6.1 System Specification

6.1.1 Hardware

The minimum hardware specification to run an Arduino sketches on Arduino Development Environment (Arduino 1.0.1) and upload it to Arduino UNO board is as follows:

- Have USB 2.0 port

The detector sketches are built on the following system:

- Intel Core i7-2620M 2.70GHz
- 8 GB RAM

The minimum hardware specification to run X-CTU (software for updating firmware and configuring parameters) to connect to XBee:

- Have USB 2.0 port

6.1.2 Software

The software specification to run an Arduino code on Arduino Development Environment (Arduino 1.0.1 windows edition) is as follows:

- Windows (32 or 64-bit)
- Java Runtime Environment

The software specification to run X-CTU software for updating firmware and configuring parameters for XBee:

- Windows 2000 until Windows 7 (32 or 64-bit)

6.2 Operational Procedure

In order to develop and run the detector, there are some operational procedures needed to be followed.

To develop the detector, follow the following steps:

1. Download Arduino Development Environment (choose the version that suits your operating system) from http://arduino.cc/en/Main/Software
2. Download X-CTU (only available for windows) from http://www.digi.com/support/productdetail?pid=3352 and install the X-CTU
3. Configure the XBee module (one as transmitter and one as receiver) through X-CTU
    a. Start the X-CTU program
    b. Select the COM port (USB Serial Port)
    c. Set the baud rate into 9600 (default)
    d. Set the flow control = NONE, Data Bits = 8, Parity = NONE, Stop Bits = 1
    e. To verify, click Test/Query
    f. Go to “Modem Configuration”
    g. Click on **Modem Parameters** -> **"Read"** to read in the current version and settings

h. Scroll down in the settings pane until you find the **Serial Interfacing -> Interface Data Rate** setting. Click on the setting and select **3** for 9600 baud.

i. Do a to h for both XBee

4. Set up the XBee hardware for the one connected through XBee Shield v1.1 to Arduino

   a. Very simple connections from Arduino to XBee with **4 wires**:

Arduino	**<--->**	**XBee**
+5V	<--->	+5V
GND	<--->	GND
Digital Pin 2	<--->	TxD
Digital Pin 3	<--->	RxD

5. Open the Arduino Development Environment
6. Open the Emotion Detector project (File -> Open -> EmotionDetector.ino)
7. After finish making changes to the code and make sure that the analog pin has been set correctly with the baud rate, click verify
8. Connect the Arduino microcontroller to PC using USB Cable
9. Set the board by selecting Tools -> Board -> Arduino UNO
10. Set the serial port used, Tools -> Serial Port -> (depends on the COM# where Arduino UNO is connected)
11. Click “upload” and open up the serial monitor

To use the detector, follow the following steps:

1. Set up the X-CTU
    a. Open X-CTU
    b. Select the USB Serial Port where XBee is connected to PC through XBee USB Adapter (Usually COM19)
    c. Make sure the baud rate is set to 9600
    d. Click on Test/Query to make sure the XBee module is connected
    e. Go to Terminal and wait for the input sent from XBee on the Arduino board
    2. Set up the Emotion Detector
    a. Connect a 9V battery to Arduino board DC input (Can also use an adaptor)
    b. Press the reset button to start the Arduino
    c. Follow the instruction on the X-CTU Terminal
    d. Make contact to fingertip or earlobe with the Pulse Sensor and put any fingertip on the LM35 Temperature Sensor

6.3 Implementation Strategy

Since the author's scope is only to build a prototype, implementation will not take place during the courses of this writing. However, if in the future the system is going to be implemented in real life, then the detector must be converted into a form of devices that is more fashionable, elegant and more user-friendly. This will be further discussed in Recommendation and Future Works section.

6.4 Prototype Details

The following picture shows all the components in the prototype, both hardware and software:

Before using the Arduino Emotion Detector, we need to first setup the hardware. Connect the XBee module to the XBee USB Adapter and use USB to mini USB – male to male cable to connect the XBee USB Adapter to PC (Shown below).

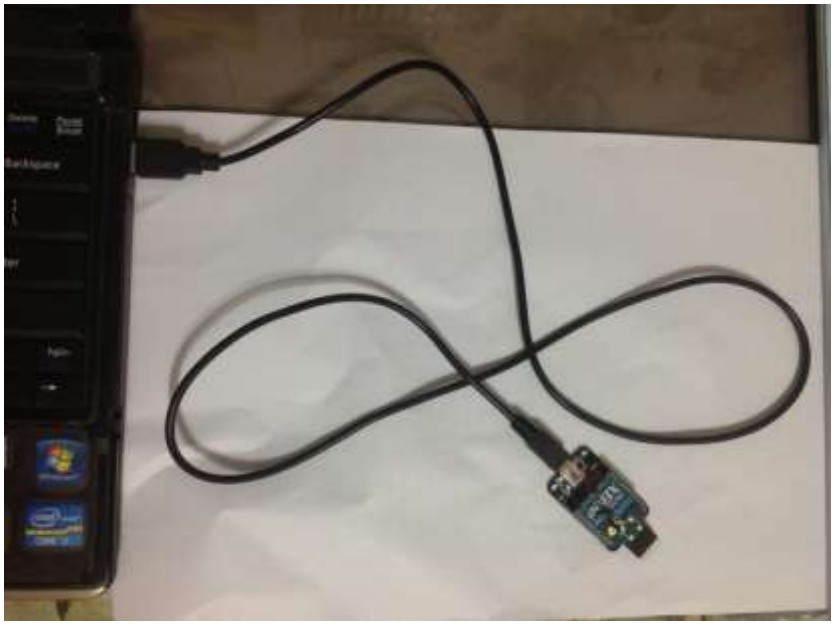

Figure 5.1: Connection of XBee to PC with USB Cable

When the XBee has been connected properly to PC, there will be a red light flashing from the XBee USB Adapter (Shown below).

Figure 5.2: Red light showing XBee has been connected

Open the X-CTU program. Select the right COM# used for plugging the XBee to PC. Then set baud rate to 9600 and click on Test/Query (Figure 5.3). When a pop up window shown (Figure 5.4). Then XBee has been connected to the program.

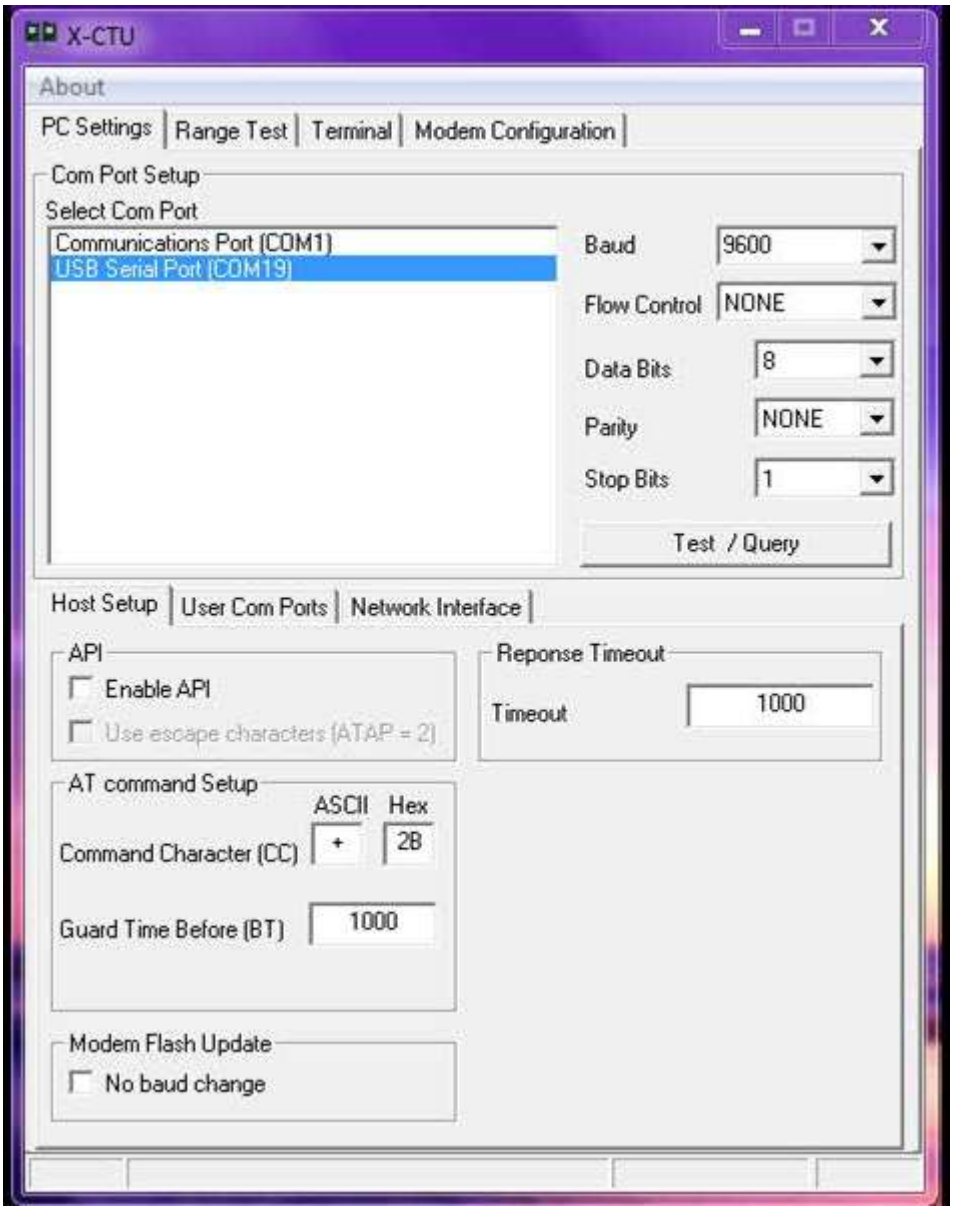

Figure 5.3: X-CTU Program

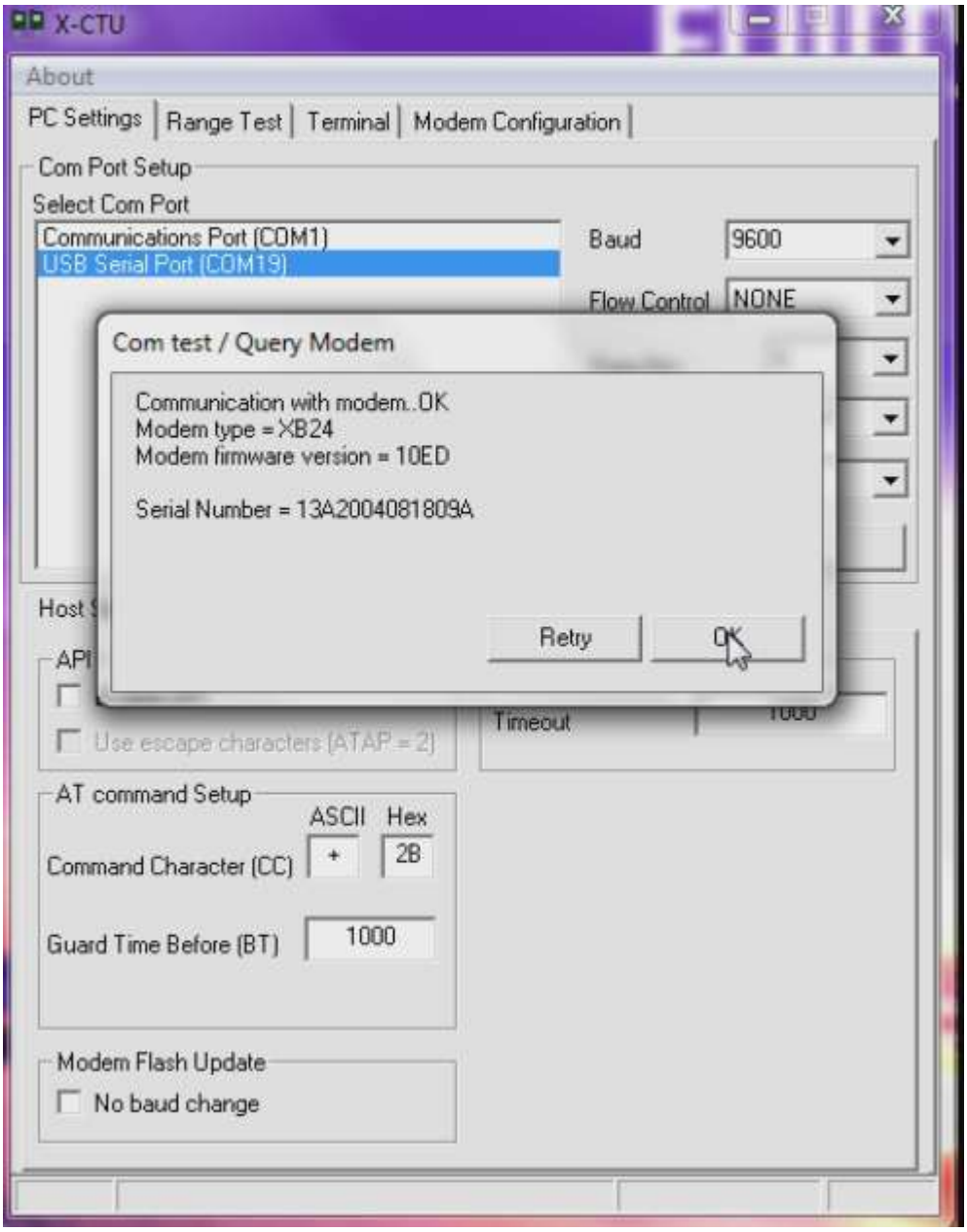

Figure 5.4: Test/Query Pop Up Window

Now go to the terminal tab (Figure 5.5). This will be the window that show the data sent from other XBee (Label 1) connected to Arduino UNO (Label 2) through XBee Shield v1.1(Label 3) where the pulse sensor (Label 4) and skin temperature sensor (Label 5) is connected (Figure 5.6). The Arduino UNO board is powered by a 9V battery (Label 6) (Figure 5.6).

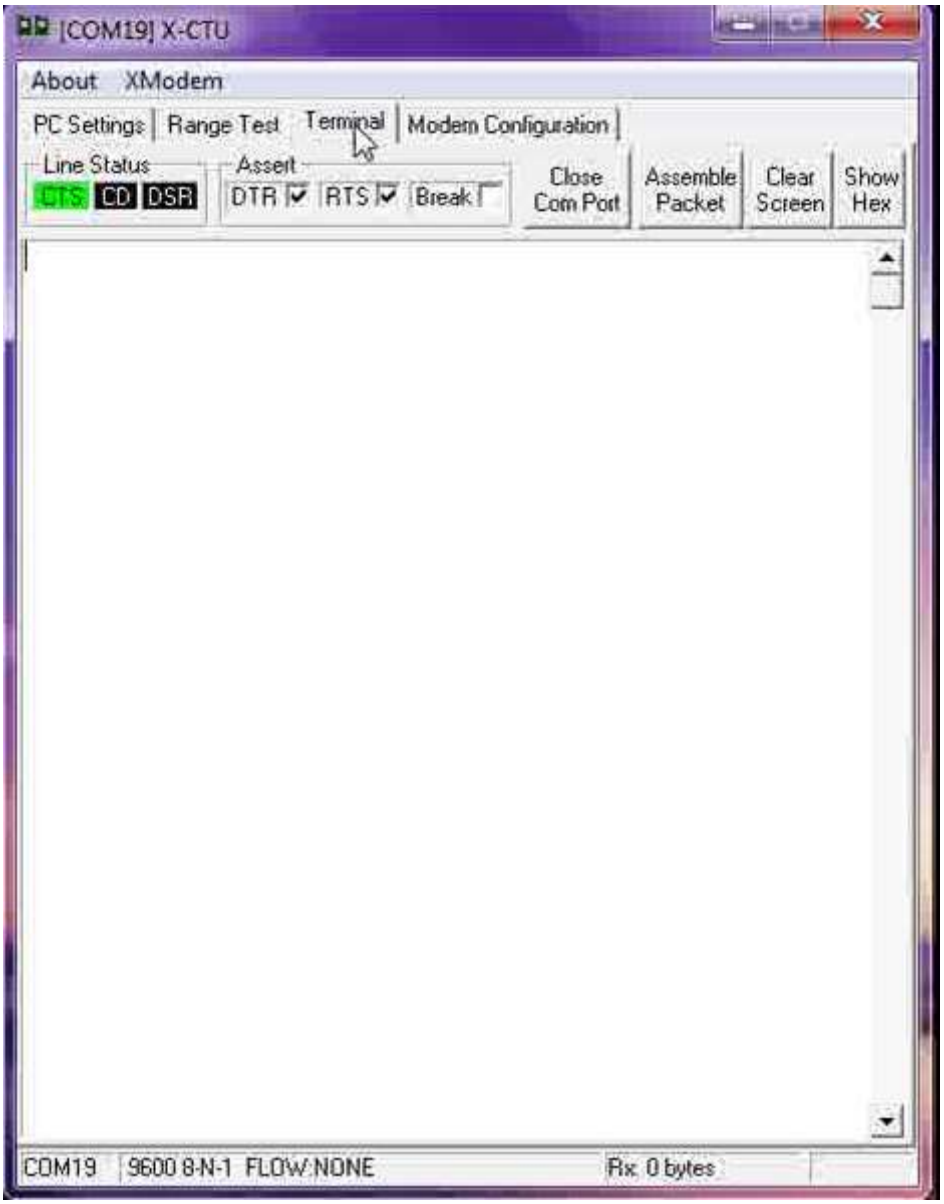

Figure 5.5: X-CTU Terminal

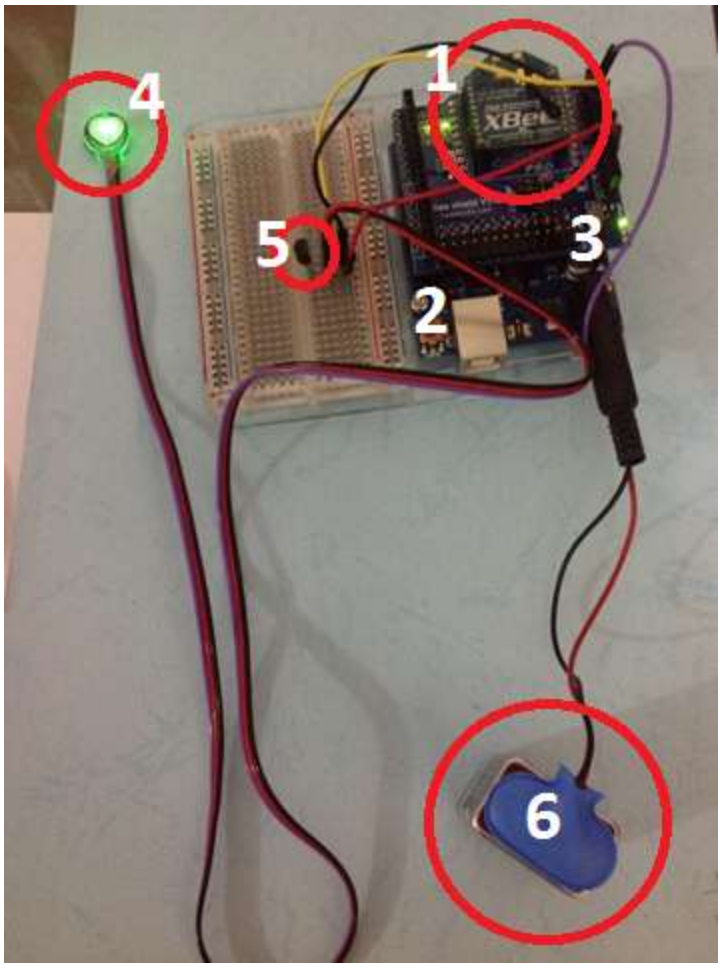

Figure 5.6: Arduino Emotion Detector powered by 9V battery

Start the Emotion Detector by pressing the red reset button (Figure 5.7). This will start the Emotion Detector and send data to X-CTU terminal (Figure 5.8). The green light on XBee connected to PC will light up, saying that another XBee is communicating to it (Figure 5.9).

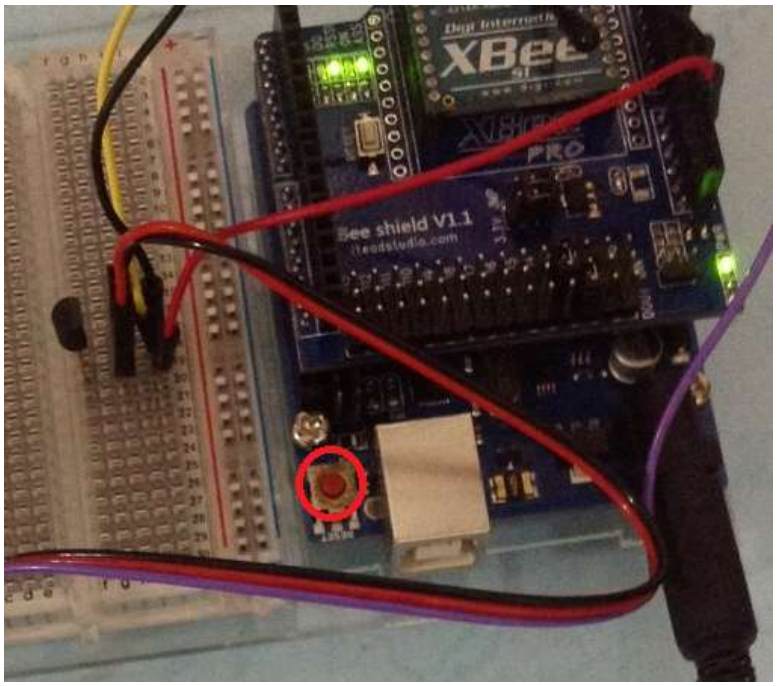

Figure 5.7: Emotion Detector Reset Button

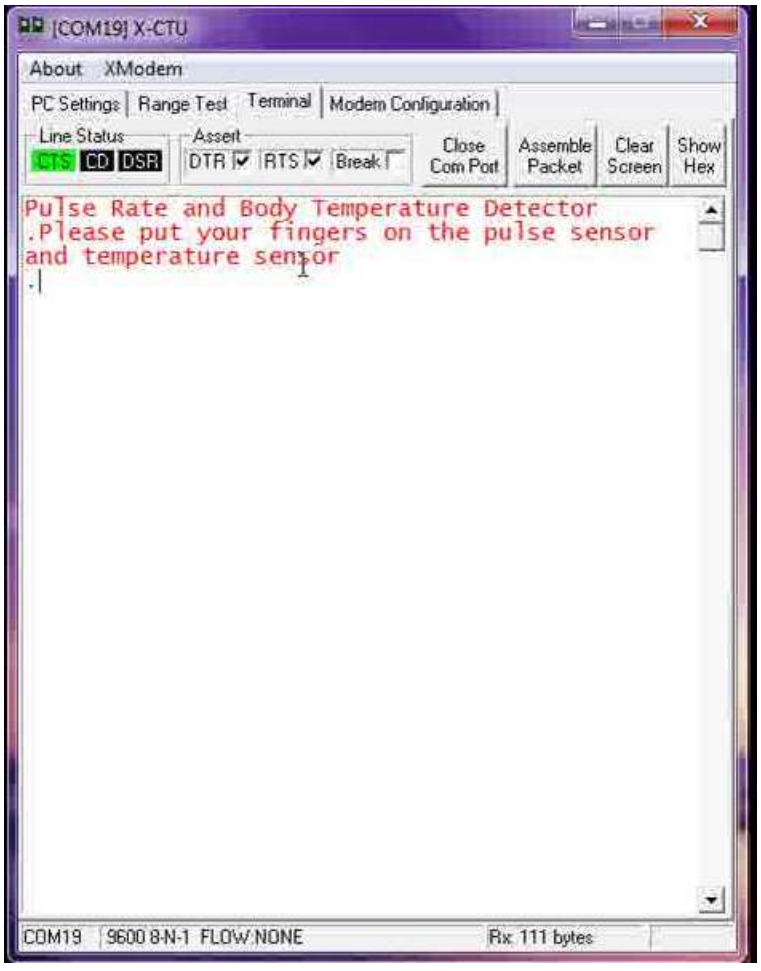

Figure 5.8: Instruction shown on X-CTU Terminal after reset

Figure 5.9: Green Light on XBee USB Adapter

User will follow instruction on X-CTU terminal by first putting fingers on the skin temperature sensors (Figure 5.10) and then on the pulse sensors (Figure 5.11).

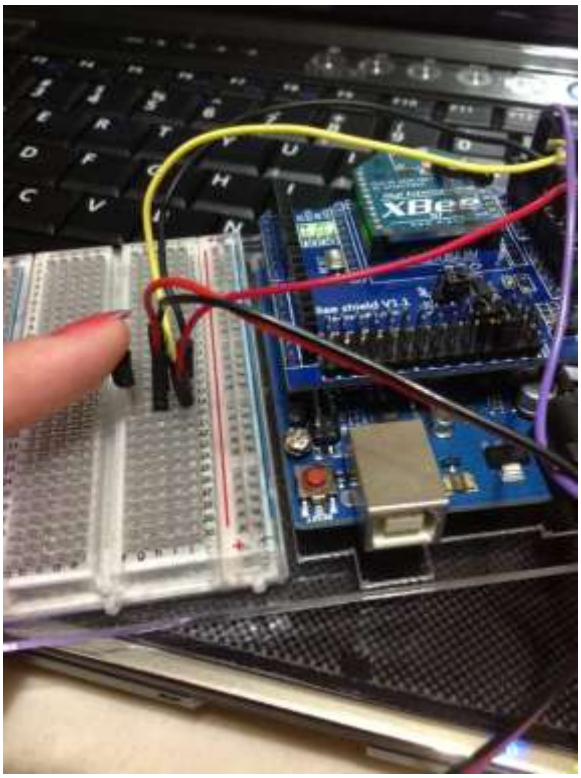

Figure 5.10: Fingers on the temperature sensors

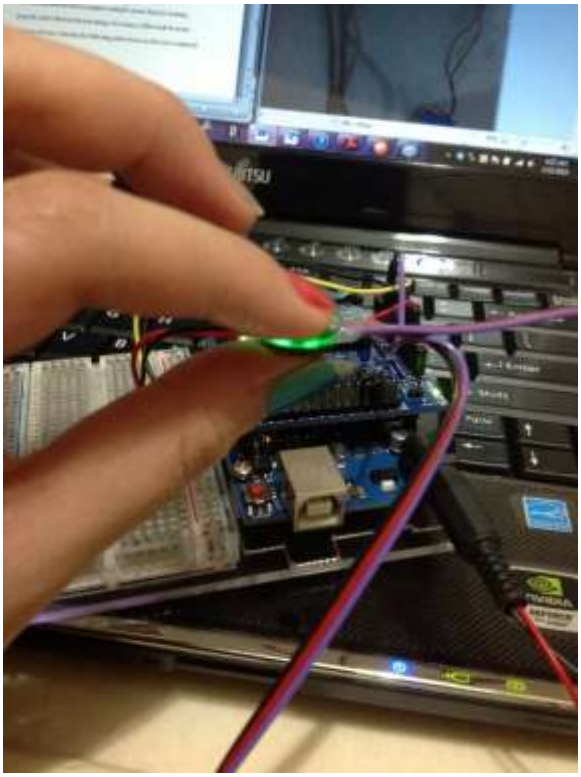

Figure 5.11: Hold the pulse sensors

Data will keep being sent to the X-CTU terminal. Make sure the fingers on both sensors are not removed until the emotion analysis is completed and seen on the X-CTU terminal (Shown below).

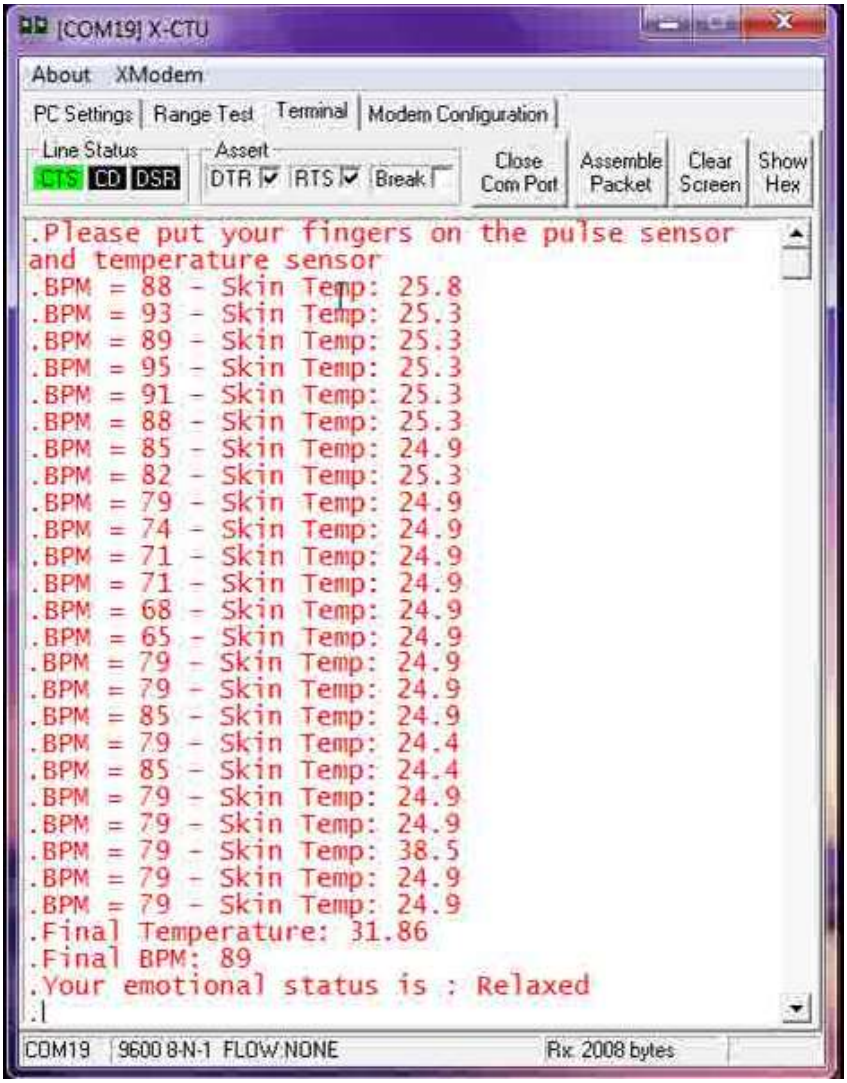

Figure 5.12: Complete result on X-CTU Terminal

6.5 Test Plan and Result

The emotion detector device requires testing to ensure that it is working properly and to find out the percentage of accuracy of the result from the emotion detector. Listed in the following subsections are the tests conducted along with their results.

6.5.1 Module Testing

Individual test cases are created for each unit to be tested.

6.5.1.1 Temperature Sensor Able To Detect Changes In The Temperature

Test Case ID	EDMT-01
Test Case Description	Test to determine whether the LM35 temperature sensor is reacting to the changes in temperature
Test Case Objective	Test whether the temperature is rising when being heat up with hair dryer
Pre-Conditions	The LM35 is properly connected and power up
Input Criteria	None
Expected Output	Temperature printed in the X-CTU Terminal will show increment in its temperature value
Status	Pass
Remarks	None

Table 5.1: Test Case EDMT-01

6.5.1.2 Pulse Sensor Detect Heart Rate And Not Some Random Data

Test Case ID	EDMT-02
Test Case Description	Test to determine whether the Pulse Sensor really detect heart rate value through the sensor
Test Case Objective	Test whether the pulse sensor only works when having contact with body parts and value is not a random data
Pre-Conditions	The Pulse Sensor is properly connected and power up
Input Criteria	None
Expected Output	BPM printed in the X-CTU Terminal will only works when user have contact with the Pulse Sensor and the result has an accuracy ±10 BPM
Status	Pass
Remarks	None

Table 5.2: Test Case EDMT-02

6.5.1.3 Arduino UNO Board Processed the Correct Information According To the Logic Assign

Test Case ID	EDMT-03
Test Case Description	Test to determine whether the Arduino UNO board processed the correct raw data inputted from the sensors correctly
Test Case Objective	Test whether the temperature shown and heart rate shown on serial monitor matched with the data gathered from the analog input
Pre-Conditions	The Arduino UNO is powered and connected correctly
Input Criteria	None
Expected Output	The manual data matched the data shown on the serial monitor
Status	Pass
Remarks	None

Table 5.3: Test Case EDMT-03

6.5.1.4 Both Xbee Module Is Talking To Each Other Sending and Receiving the Correct Data

Test Case ID	EDMT-04
Test Case Description	Test to determine whether the data send through XBee is not lost, corrupted or exchanged with other signals
Test Case Objective	To make sure data send is same with data shown on X-CTU Terminal
Pre-Conditions	Both wireless XBee is connected correctly and power up
Input Criteria	None
Expected Output	Data printed on serial monitor will be same with data printed on X-CTU Terminal
Status	Pass
Remarks	None

Table 5.4: Test Case EDMT-04

6.5.2 Integration Testing

6.5.2.1 Touch the Pulse Sensor – Show the BPM detected

Test Case ID	EDIT-01
Test Case Description	Test that after each module is combine into one detector, the Pulse Sensor still working and showing the BPM number
Test Case Objective	To make sure the BPM is shown on X-CTU Terminal
Pre-Conditions	The Pulse Sensor is connected correctly, no collision of analog pin used with other sensors
Input Criteria	User must hold the Pulse Sensor and see whether the LED 13 is blinking
Expected Output	Data printed on X-CTU Terminal
Status	Pass
Remarks	None

Table 5.5: Test Case EDIT-01

6.5.2.2 Touch the temperature sensor – Show the skin temperature detected

Test Case ID	EDIT-02
Test Case Description	Test that after each module is combine into one detector, the temperature sensor still working and showing the temperature number
Test Case Objective	To make sure the temperature is shown on X-CTU Terminal
Pre-Conditions	The temperature is connected correctly, no collision of analog pin used with other sensors
Input Criteria	User must touch the temperature sensor
Expected Output	Data printed on X-CTU Terminal
Status	Pass
Remarks	None

Table 5.6: Test Case EDIT-02

6.5.2.3 Send the data processed from Arduino UNO to PC through wireless XBee

Test Case ID	EDIT-03
Test Case Description	Test that after each module is combine into one detector, the Wireless XBee is still working as it is and not interfere by the signals of other sensors
Test Case Objective	To make sure the data send between Arduino UNO and PC is the same and has no difference
Pre-Conditions	The XBee is connected correctly, no collision with other sensors
Input Criteria	None
Expected Output	Data printed on X-CTU Terminal correctly
Status	Pass
Remarks	None

Table 5.7: Test Case EDIT-03

6.5.3 System Testing

System testing tests the system to ensure that it is working as expected.

6.5.3.1 System Functional Test

Test Case ID	EDST-01
Test Case Description	Test to check the complete system
Test Case Objective	To check whether the system fulfills the functional requirement stated in Section 4.8.4
Pre-Conditions	The module, wires and components have been connected correctly
Input Criteria	User performs all action needed by both sensors in generating input to the Arduino UNO
Expected Output	The system should perform as expected in the functional requirements without any error
Status	Pass
Remarks	None

Table 5.8: Test Case EDST-01

6.5.3.2 System Performance/Responsiveness Test

Test Case ID	EDST-02
Test Case Description	Test to check responsiveness of the whole system
Test Case Objective	To test how fast the system responds to sensors input when both sensors have be in contact with user body part or skin
Pre-Conditions	The module, wires and components have been connected correctly
Input Criteria	User performs all action needed by both sensors in generating input to the Arduino UNO
Expected Output	The system should only take reasonable amount of time (approximately 1000mS) to read input after the previous input
Status	Pass
Remarks	None

Table 5.9: Test Case EDST-02

6.5.4 User Testing

Although the application is still in prototype phase, it is beneficial to gain feedbacks from users in order to further improve the program in the future.

6.5.4.1 User Testing Design

The user testing will include a number of participants first trying out the detector, using all the features, and give a feedback accordingly (ranging from 1 to 5) for the following categories:

- Responsiveness – Refers to how quick/fast the detector triggers an event when the user uses the sensors available in the detector.
- Intuitiveness – Refers to how comfortable is the users in using the detectors.
- Accurate result– Refers to the closest mood to the actual user mood.

6.5.4.2 User Testing Result

Due to limited time, 10 samples picked randomly from BINUS INTERNATIONAL students participate in the user testing. The following snapshot shows the participants testing the detector:

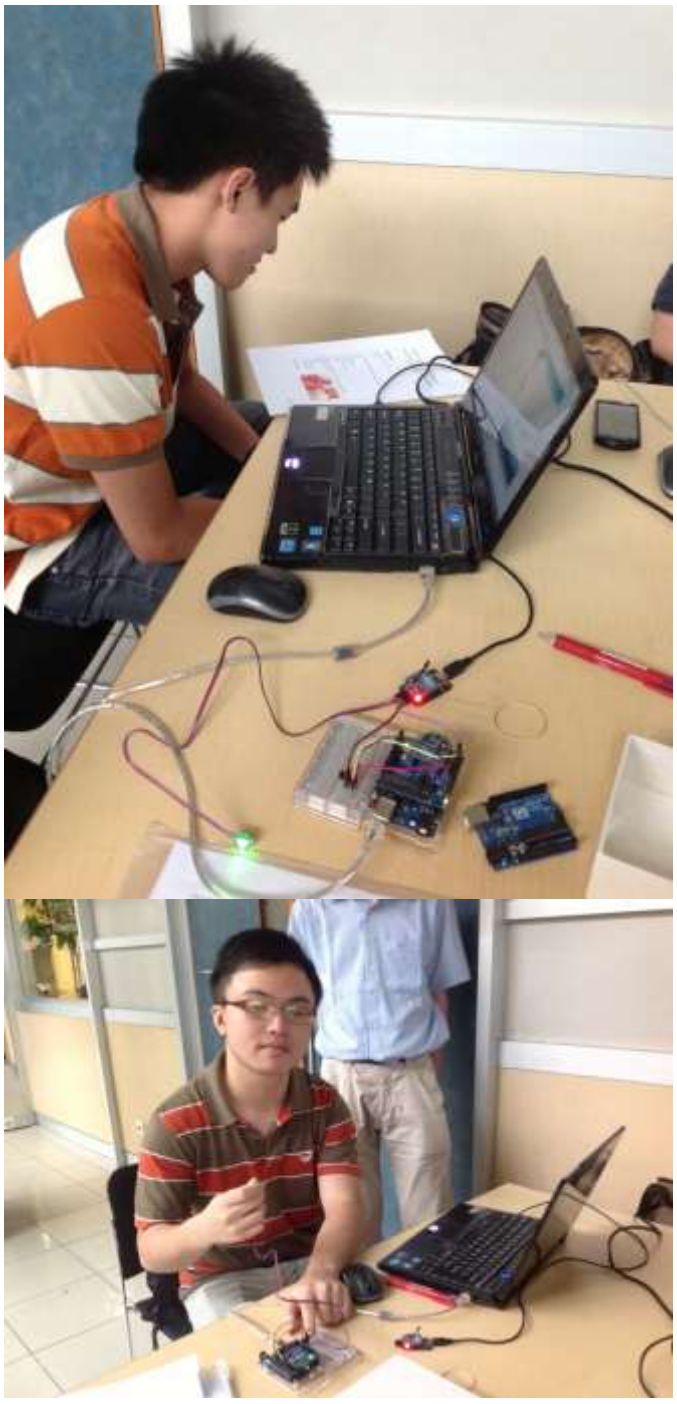

Figure 5.13: User Testing by Yonathan and Albert

Figure 5.14: User Testing by Febndy and Vishal

During the testing, every tester is given questionnaire form (attached in APPENDICES). User will need to compare the current emotion that they predict and manual counted BPM with the BPM and emotion analyze by the emotion detector. Then they will give score to the level of responsiveness, intuitiveness and accuracy of the result. Below is the detail of the testing results:

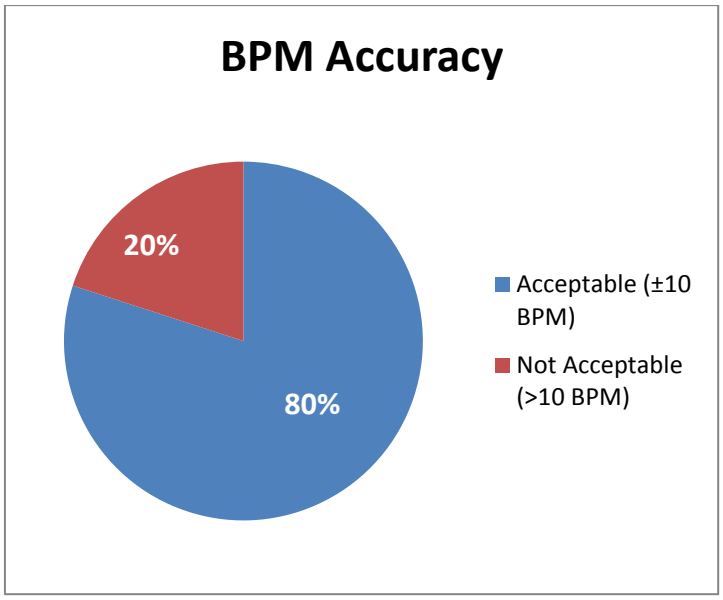

Figure 5.15: BPM Accuracy

As we can see in the pie chart above, by comparing the manual BPM with the BPM generated by the system (Arduino heart rate emotion detector), there is 80% accuracy rate which means that the BPM generated by the system could be consider as acceptable from the average rate of 50%.

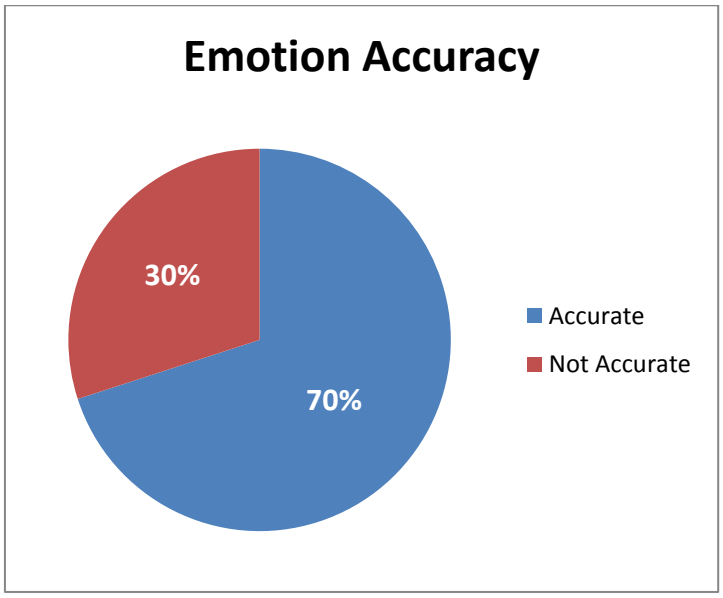

Figure 5.16: Emotion Accuracy

While the pie chart above shows the percentage of accuracy of the emotion by comparing the emotion that tester feel with the emotion detected by the system (Arduino heart rate emotion detector). We could see that the system has 70% accuracy level, which means that the system is reliable and the emotion correlation table could be accepted for usage in the future, though the emotion correlation table still need to be consulted to an expert.

The following shows the summary of the results:

Criteria	**Average Score (out of 5)**	**Additional Comments**
Responsiveness	4.5	Participants are overall satisfied with the responsiveness as the sensors are sensitive enough to even slight holding.
Intuitiveness	3.7	A few participants complain regarding the detector, since they need some time to know how to use it and the build are not so user-friendly.
Accuracy	3.9	Most of the participants are satisfied with the emotion

		result from the EmotionDetector but the BPM is slightly difference from the manual BPM count.

Table 5.10: User Testing Result

From the testing result, we could conclude that from responsiveness point of view, the small sensors are doing quite well and has no problem with difficulty of finding the perfect spot to use it. While from the intuitiveness feedback, the detector need to be modify into a more user friendly form so that it could be easily used. Lastly is the accuracy problem. Though the testing compares the emotion result from the detector with the real emotion that they feel at that time, the accuracy of the emotion state by the user cannot be guarantee. This is because people naturally tend to be bad predictors of their own emotional reactions [45].