CERTIFICATION OF APPROVAL

DEVELOPMENT OF THE ISOLATION WARD DISPENSER PLATFORM

by

Jaayaprakash Narianan

A project dissertation submitted to the
Department of Electrical & Electronic Engineering
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronic Engineering)

Approved:

____

Patrick Sebastian
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

May 2012

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

JAAYAPRAKASH A/L NARIANAN

14044

ABSTRACT

The applications of robotics are basically the heart of various industries, namely manufacturing, production, agriculture, or the educational line. There is also a notable growth in the usage of robotics in the field of medicine. Robotics is usually applied to perform tasks in order to ease a man's daily routine. This development aims to develop an useful method to transport needs to patients in an isolation ward during any viral outbreaks, such as the H1N1, SARS, ANTHRAX or the bird flu via the usage of a robot. The aim is to assist both patients and medical staff, in terms of safety and time management efficiency. The mobilization concept of the robot is done by using the line tracking system, with the aid of other minor modules in order to assist the platform in completing its assigned task. The functionality and effectiveness of the Security Modules, Line Tracking Modules, Intercom Modules, Compartment Modules, Motion Detection Modules as well as the Emergency Call Modules were studied, compared, selected and then implemented onto the Isolation Ward Dispenser Platform.

ACKNOWLEDGEMENT

A heartfelt gratitude to Mr. Patrick Sebastian, the project supervisor for the Development of the Isolation Ward Dispenser Platform, for showering continuous assistance in the form of technical support, advice and valuable suggestions. Without his guidance, the Isolation Ward Dispenser Platform would not become reality.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Project Background

Various robots have been designed for a variety of applications domestically as well as for the industries. One of the most notable application of these robots are those used in hospitals, used to dispense medications stored in an automated storage and retrieval system based inventory. This eases a pharmacist's task in distributing medications to patients. However, these robots are normally stationary and only a part of the structure performs the pick and place task. The Isolation Ward Dispenser Platform would assist the medical staff in transporting needs to the patients directly. This is vital in cases where the patients are quarantined.

The usage of this Isolation Ward Dispenser Platform contributes in time management efficiency as frequent checks do not have to be performed on the quarantined patients frequently. This allows the medical staff to focus on the critical patients instead. Implementation of this robot may be considered as a onetime investment as it helps reduce the risk of medical staff getting infected by any contagious diseases in an event of a confirmed case within the isolation ward.

1.2 Problem Statement

The occurrences of viral outbreaks shock the whole world. There was no immediate cure, and the diagnosed patients were just isolated to prevent the virus from spreading further. Since the patients basic needs are usually delivered to them by the medical staff, the risk of them being infected as well are equally high. The platform assists in preventing the medical staff that saves lives from being in high risk instead.

There are two categories of quarantined patients. The first category is where the patient is only a suspect of a viral disease and is to be quarantined temporarily, and the other is where a patient is confirmed with one of the mentioned viral diseases, and requires full medical attention. The Isolation Ward Dispenser Platform was designed for the use of dispensing needs only to patients that fall under the first category.

During a viral outbreak, medical staff would have to split their attention to both the critical patients, and the quarantined patients. Critical patients, such as those admitted in an ICU due to chronic diseases, or accidents should be given priority at all cost. Thus, having a platform to perform part of the medical staffs’ task would help in terms of time management.

1.3 Objectives

The aim of this project is to develop a platform which transport needs to the patients in an isolation ward, during a viral outbreak. Operating on a line tracking system, it is able to mobilize to patients by itself.

The second objective is to develop a platform which saves both time and cost for the medical staff. They can focus on critical patients and frequent routine checks on the non-critical patients can be reduced. Besides, the risk of them getting infected by these viral diseases could be lowered as well. Multiple patients could be attended in one go as the platform can carry more needs compared to a single medical staff, thus, saving precious time which could be allocated on the critical patients instead.

1.4 Scope of Study

This project is prototype based which utilizes the knowledge of robotics in the field of medicine. It is a combination of various minor modules, forming a major system which performs the assigned task. The system integration concept was applied for this project.

A brief study was conducted on the types of line tracking algorithms, intercom systems, compartment management systems, the integration of keypad and LCD display for password input interface, which acts as a security feature, emergency call function as well as the motion detection feature.

The functionality of this project is demonstrated based on a downscaled prototype of an actual isolation ward. The dimension of the prototype ward platform was designed to be about 1.0m x 1.6m. Six beds, in the size of a standard tissue box will be placed evenly on the prototype ward. The lines will be drawn onto the prototype ward using a black insulation tape.

The functionality of this robot will be limited to six patients only, in a single isolation ward. The prototype dimension is limited to 35cm x 35cm x 35cm. Two independent DC motors was used to power the wheels. A 9.6V rechargeable battery pack acts as the power supply. The robot will consist of three levels, where the first containing the motors and the batteries, the second level consists of the circuitry, and the top level would consist of the compartments, LCD and keypad.

1.5 Relevancy of the Project

The Isolation Ward Dispenser Platform will be a downscaled prototype of a system which could be implemented in the hospitals to perform the above mentioned tasks. The line tracking module acts as the robots mobilization technique. The intercom feature would act as a medium of communication between patients and the medical staff. The security feature using the keypad and LCD prevents patients from retrieving items from the wrong compartments. Besides, the emergency call feature functions exactly like the call button in the hospitals, which is linked to the main helpdesk.

The aim is to demonstrate a scenario of patients in an isolation ward, being attended by the Isolation Ward Dispenser Platform. By demonstrating via a fully functioning downscaled prototype, the author would ensure that the project objectives, goals and implementations are clearly understood and appreciated by everyone.

However, the concept of the Isolation Ward Dispenser Platform can also be utilized in many other fields, such as in a manufacturing factory. This platform could be used to transport tools or components from one workstation to another. This platform can also be used in an office, to transport letters or bills, from one cubicle to another. The scope was narrowed down to hospitals to help ease the understanding of this principle of operation.

1.6 Feasibility of the Project within Scope and Time Frame

In order to complete the prototype within 14 weeks, a schedule was drafted out, tasks to be done for this project was divided on the first week itself. Comparisons based on the project budget, and the time frame was made as well. The aim is to develop a prototype with the minimal cost, in the shortest period of time, with the most optimal functionality and reliability. While doing this, the implementation is ensured to be within the scope defined in the project objectives earlier. In order to keep track of time, cost and scope, a suggested Gantt Chart was followed to maintain a positive pace, collected all receipts of purchase and performing cost evaluation, and also frequently referring to the problem statement and project objectives before implementing a new module onto the project.

CHAPTER 2

LITERATURE REVIEW

Robotics is the branch of technology that deals with the design, construction, operation, structural disposition, manufacture and application in various tasks, either domestically or in industries. [1]. A simple robot consists of four main elements, which are the controller, the actuators, the sensors, and the power supply itself. The Isolation Ward Dispenser Platform is a robot that is designed to operate strictly in hospitals only, but it would still consist of the mentioned elements, integrated with other minor modules which enable it to achieve the stated objectives above.

2.1 Line Tracking Module

The platform could either operate with this principle, where the light dependent resistor (LDR) sensors were used to detect and follow a line which forms the path in the prototype ward, or via the usage of IR LEDs, which transmits an infrared beam, which is then reflected on the surface and captured by the receiver LED. A comparative study was carried out to determine which is better for the platform.

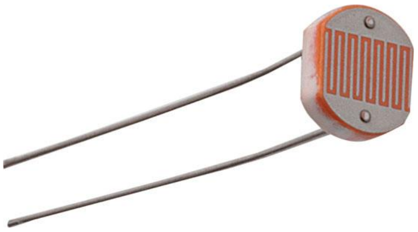

Figure 1: LDR

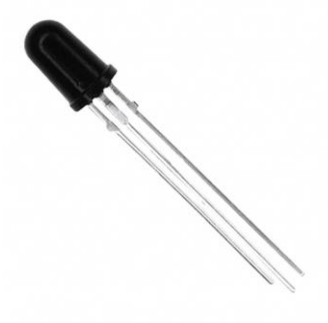

Figure 2: IR LED

The circuitry would consist of two PIC microcontrollers, and C algorithms were used for both. Microcontrollers are preferred as it utilizes a single wire controller to receive data and power, and it allows expansion of the input and output ports anytime if necessary, compared to the conventional microprocessors which are usually fixed and only operable via assembly language. [2]

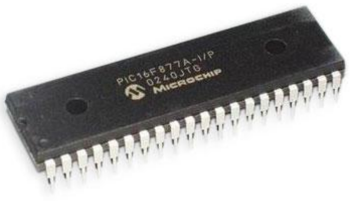

Figure 3: 16F877A PIC

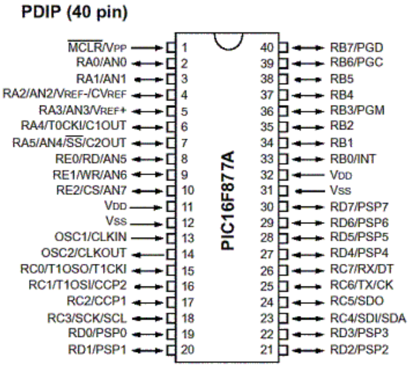

Figure 4: 16F877A PIC pins

2.2 Intercom Module

The intercom circuit could either be retrieved from a ‘Walkie Talkie’ set, where one transmits signal in transmitting mode and the other receives signal in receiving mode, or vice versa. [3] Before transmission, the signal is to be amplified by the source device before being received by the receiver device. The amplifier and filter circuit was has to be constructed and added separately. However, with time and budget, a more effective intercom system could be developed using the obtained schematics shown in Figure 6 below.

Figure 5: Intercom Circuit from Walkie Talkie

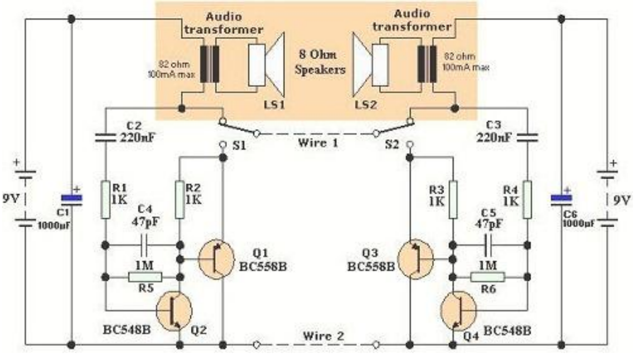

Figure 6: Sample Two Way Intercom Circuit

2.3 Compartments Module

LEDs and limit switches is to be used on the compartments. The LEDs act as indicators of whether or not the compartment is locked or unlocked. The product limit switch determines the presence of the item in the compartment, and the door limit switch indicates if the door is opened or not. All these are then linked to the buzzer which acts as an alarm. If the wrong compartment was opened, the alarm would be triggered. Only the unlocked compartment would have a green indicator where the rest will remain red.

However, the limit switches could be eliminated by using the electromagnetic lock concept, which can be activated by the means of a PIC signal. These locks have different holding force, which had to be researched before being implemented on the platform.

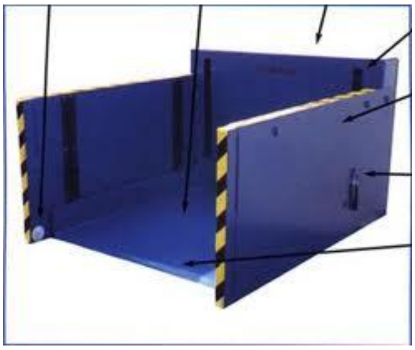

Figure 7: Compartment With Limit Switches

Figure 8: Electromagnetic Lock

2.4 Motion Detection Module

In order to detect movements so that the robot does not collide with any object or person, PIR sensor module will be mounted onto the front of the Isolation Ward Dispenser Platform. Different types of motion sensors performs the detection differently, which is why a thorough study was done on the detection distance, accuracy and other parameters before selecting the appropriate motion sensor.[4] For instance, an analogue distance sensor could be used as well. An ultrasonic sensor would also perform this task effectively.

For this project, the distance of the detection is required to be within 30cm from the platform, since this is a downscaled prototype. In reality, the detection range should be at least 1m. This is vital to ensure the platform is not damaged due to impact, and also to make sure the platform does not injure anyone in the ward.

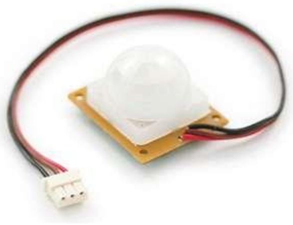

Figure 9: PIR Sensor Module

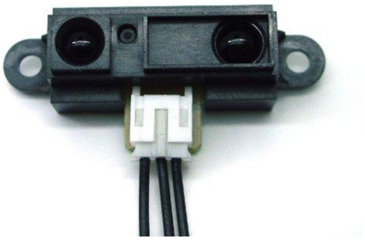

Figure 10: Analogue Distance Sensor

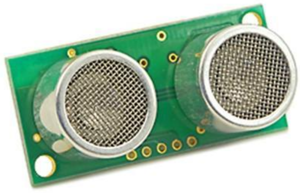

Figure 11: Ultrasonic Sensor Module

2.5 Security Module

Patients would be required to input their patient ID in order to access their needs from the respective compartments. A modern password interface would include successive arrays of images for display on a client device. [6] This password system was also integrated to the LCD display, which displays the patients name while being attended. The passwords received from the patients are compared with the calculated password, which is the preset password, and access is granted to the respective compartment if it is a match. [7] The LCD will be integrated to the PIC and keypad. This feature acts as a security precaution, preventing patients from opening the wrong compartment. However, another effective method will be by using RFID concept. An RFID scanner can be placed onto the platform, where the patients tag acts as the ID tags. With this method, the patients won’t have to input password manually.

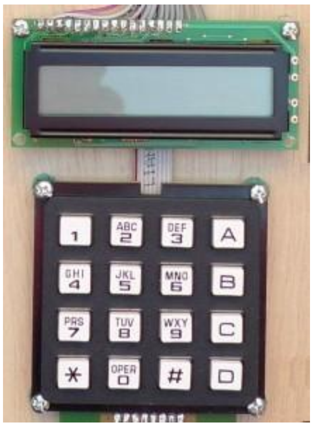

Figure 12: Keypad and LCD Integration

Figure 13: Simple RFID Concept

2.6 Emergency Call Module

The Isolation Ward Dispenser Platform should be able to distribute medications to a specific patient when required, using the 315MHz transmitter and receiver. The transmitter which will be used as an emergency buttons here sends an interrupt signal to the receiver, linked to the PIC circuit on the platform, which then skips the next patient to attend the patient with priority. To do this, the transmitter inputs must be linked to every patient’s bed.

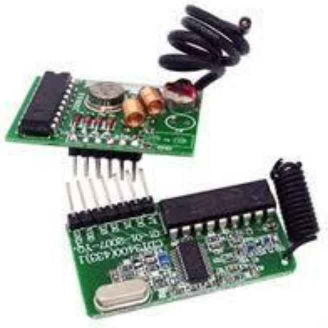

Figure 14: Transmitter and Receiver as Emergency Buttons

The Isolation Ward Dispenser Platform will also be able to skip a current patient if required. The sequence is completed, and then it would return to the unattended patient.

If the patient does not respond even after the alarm triggers, the robot would proceed to the end position of the track and wait for the next instruction.

CHAPTER 3

METHODOLOGY/PROJECT WORK

3.1 Project work

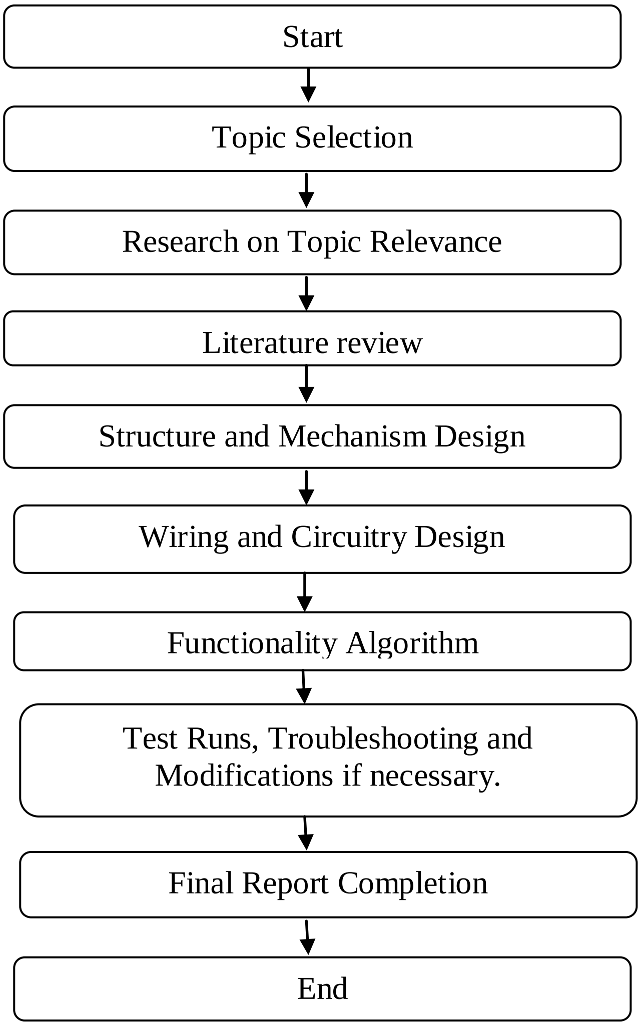
Start
Topic Selection
Research on Topic Relevance
Literature review
Structure and Mechanism Design
Wiring and Circuitry Design
Functionality Algorithm
Test Runs, Troubleshooting and Modifications if necessary.
Final Report Completion
End

Table 1: Project Activities Flow

This is basically a prototype based project. Specifically, it focuses on the application of the line tracking movement system in hospitals, designed to attend patients in a quarantined ward. Firstly, research was conducted on several modules mentioned above.

With the obtained information, the literature review on the application of robotics in the field of medicine was done. After completing the literature review, the type of line tracking algorithm used was confirmed. The usage and the placement of the appropriate sensor play a major part in the robot movement's reliability.

Then, the intercom system circuit was added on both the robot and the reception desk. A 'Walkie Talkie' set was used for this. As for the compartments, LEDs and limit switches were integrated together and linked to the PIC and the buzzer. The PIR sensor on the robot was linked to the buzzer, which alerts a person if he or she is obstructing the robots path.

The design and fabrication of the Isolation Ward Dispenser Platform is to be completed in two weeks, and the wiring and circuitry requires another two weeks. More time was allocated for the programming and troubleshooting phase, in order to construct a consistent and reliable code for the robots functionality, thus a minimal of 100 test runs are to be performed, with a maximum allowable failure of 2 runs.

3.2 Research Methodology

Research was a method taken in order to gain information regarding the major scope of the project. The sources of the research cover the online sites regarding line tracking systems, intercom systems, movement sensors, compartments, e-journals, e-thesis and several other trusted links.

Implemented research steps:

1. Gained information on the functionality of the line tracking system, the concept, codes and application of the system.

2. Drafted down operation principle for the Isolation Ward Dispenser Platform operation based on a prototype scale ward.

3. Study and research on intercom systems, compartment locks, and movement sensors.

4. Designed the circuitry of the Isolation Ward Dispenser Platform based on above mentioned features.

5. Structure and mechanical design of the Isolation Ward Dispenser Platform.

6. Construction of a functional PIC code for the robot operations.

7. Simulation and testing of the robots functionality and reliability.

CHAPTER 4

RESULTS

4.1 Prototype Base And Wheels

The base plate, two driving wheels and a multidirectional wheel was successfully mounted together. Two 7.2V DC motors were attached to the driving wheels. The multidirectional wheel, mounted at the front of the base acts as a support and guide for the robot movement. The motors were linked to separate 10A motor drivers, which was then connected to the PIC. Robot speed was controlled by the PIC PWM function.

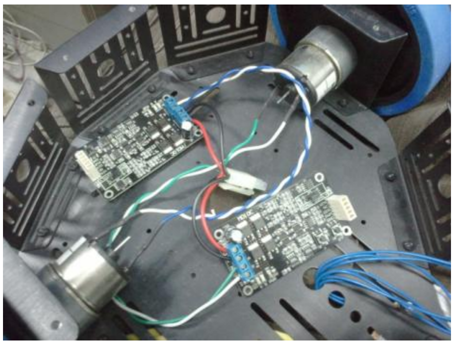

Figure 15: DC Motors and 10A Motor Drivers

As a backup, a motor driver circuit is designed using the L298N motor driver IC. In an event of a breakdown, this driver will be used instead. It has six inputs to the PIC, which is the PWM1, PWM2, CW1, CW2, CCW1, and CCW2. However, in order to reduce the inputs to the PIC, the 10A motor drivers are a better choice since it only has four inputs to the PIC, which is the PWM1, PWM2, DIR1 and DIR2. In order to control the direction of the motors using this, DIR is either set to 0 (clockwise) or 1 (anti-clockwise).

The following diagram illustrates the schematics of the motor driver using the L298N IC.

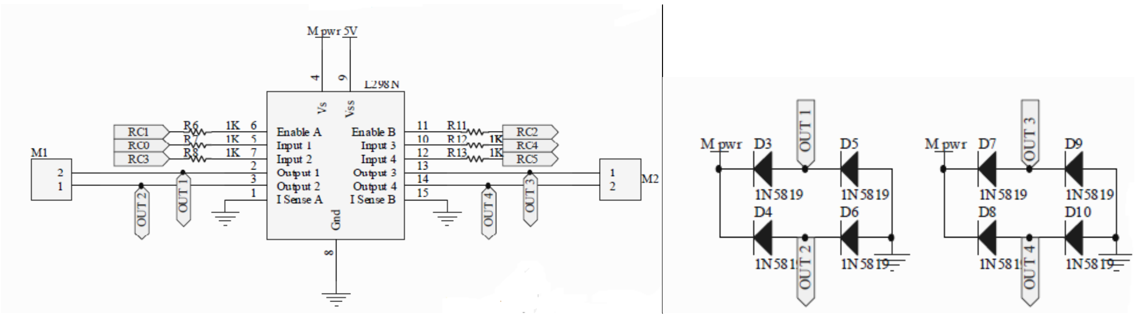

Figure 16: Motor Driver Schematics using L298N

4.2 Study On Selecting An Effective Communication Method

After making comparisons between constructing an intercom circuit from scratch, or via the usage of a readily available ‘walkie talkie’ device, the latter was utilized as the communication tool for the Isolation Ward Dispenser Platform in order to save cost and time. A part of the ‘walkie talkie’ will be mounted onto the robot itself, whereas the other part will be placed onto the medical staffs’ helpdesk. The ‘walkie talkie’ will be powered by an independent battery source. After several range testing, the maximum achievable range was about 5m, which is sufficient for this project. However, the output speaker volume was low and contained noise. In future, additional filter and amplifier circuits are to be added to improve the output audio quality.

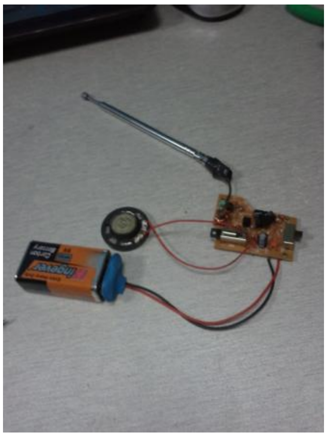

Figure 17: Intercom Circuit obtained from Walkie Talkie

4.3 Comparison Between LDR Sensors And IR Sensors

LDR sensors are basically more reliable, and not light sensitive. It operates on 5V DC. The IR sensors, operating on 5V DC as well, have a lower reliability under sunlight. The author used LDR sensors as they are cheaper and smaller in size, which assists in reducing the robots weight and cost. After performing a test on the LDR sensor detection logic, a truth table was constructed for the sensor inputs, as shown in the table below:-

Sensor Conditions	**Actions Performed**
11111	No Line Is Detected(Keep Turning Anti-Clockwise)
11000	Right Turn
10001	Move Forward
10000	Slight Right Turn
00011	Left Turn
00100	N/A (Move Forward)
00001	Slight Left Turn
00000	Junction

Table 4: Possible LDR Sensor Conditions for Line Tracking

Remarks:

1- White surface (Floor)

0- Black surface (Line)

These are some of the possible condition for five sets of LDR sensors. Even though it was more time consuming to construct a functioning algorithm, the usage of five LDR sensors increases the line tracking reliability.

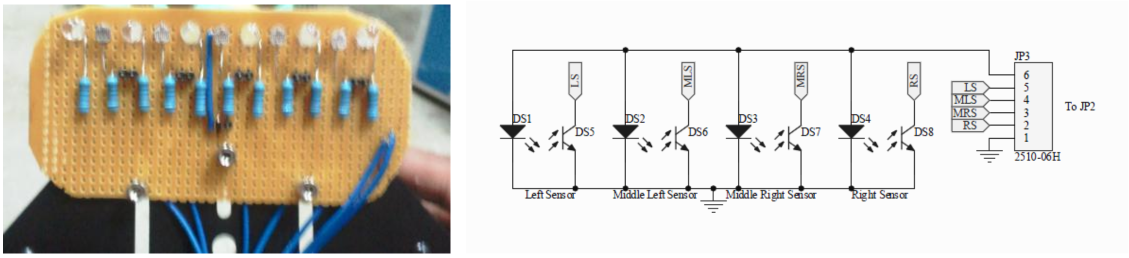

Figure 18: LDR Circuit coupled with Superbright LEDs and schematics

4.4 Compartment Module

Plastic compartment boxes were used as the platform compartments. Since the platform was designed to cater six patients at one go, a total of six compartments were used. Two limit switches, and two LEDs were used on each compartment. The following shows the functions of the mentioned switches and LEDs on the compartments:-

Component	Function
Green LED	When it is ON, it shows that the compartment had been unlocked. Buzzer would not be triggered when this compartment is opened.
Red LED	When it is ON, it shows that the compartment is locked. Buzzer will be triggered when this compartment is opened.
Both LEDs OFF	This is when the compartment is empty and not used. The platform would be programmed to skip the unattended patient.
Door Limit Switch	This indicates if the compartment door is opened or closed.
Compartment Limit Switch	This indicates if the items in the compartment had been retrieved or not.

Table 5: Compartment Switches and LEDs

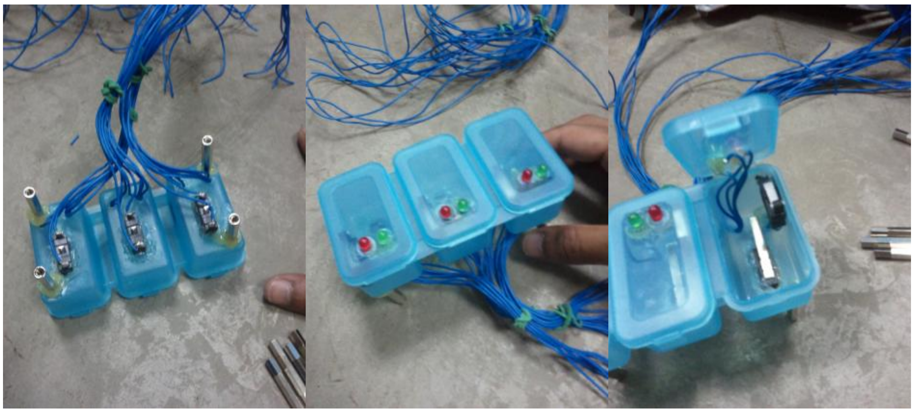

Figure 19: The Designed Compartment Module

4.5 Motion Detection Module

A PIR sensor module was used for the motion detection. Since the desired range of detection is 30cm, the distance of detection had to be reduced via trial and error, using the trimmer resistor on the sensor module itself. The obtained data is tabulated as below:-

Resistance (kOhm)	Detection Range (m)
5	4.6
4	3.8
3	2.8
2	1.7
1	0.8
0.5	0.3

Table 6: PIR Resistance VS Range Data

Figure 20: PIR Sensor Module

However, since the detection angle is about 120^o, the PIR sensor also detects the prototype beds, which are actually out of the platform movement path, and stationary. To overcome this, a ring of cardboard is fabricated and placed around the sensor body. This reduces the detection angle to about 80^o. The aim is to detect any obstacles directly in front of the platform. The only problem faced is that the PIR sensor only detects object with motion. Thus, A static object placed in front of the object would not be detected. This simply means that in real life, the platform would be able to detect a moving person or object, but it might not respond to a static person or object. This can be overcomed by using the ultrasonic sensor, which senses objects based on the principle of wave reflection.

4.6 Emergency Call Module using Transmitter and Receiver

After thorough research and colsultations from the project supervisor, the 315MHz transmitter and receiver were used to complete this module. This works exactly like the car remote control, which unlocks the car from a distance. For the Isolation Ward Dispenser Platform, the emergency call button is basically a push button, which sends signal to the platform itself. When a patient pushes the emergency button, it simply means that priority will be given to that patient. The platform will skip the next patient, and attend the patient with priority first. Since a single transmitter and receiver module is used, a total of four inputs can be used for this project. So, only four push buttons were connected.

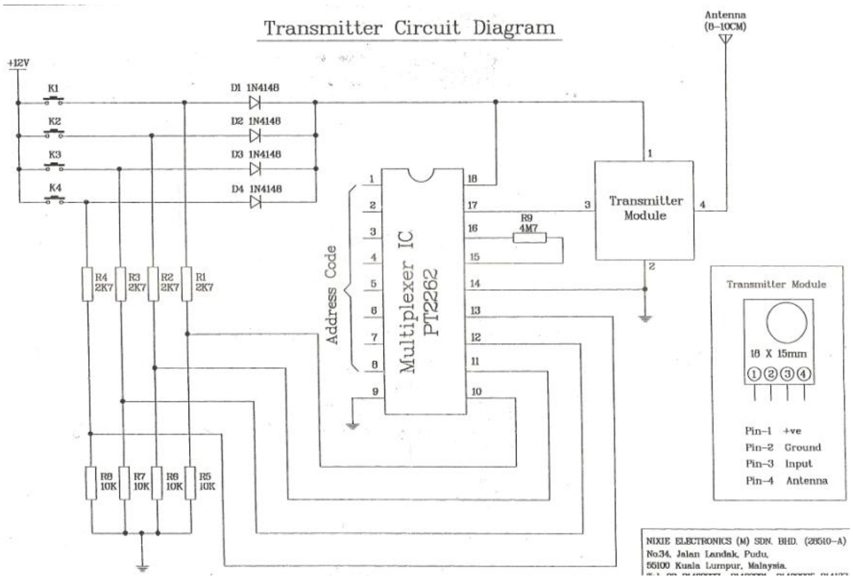

Figure 21: A 315MHz Transmitter Schematic

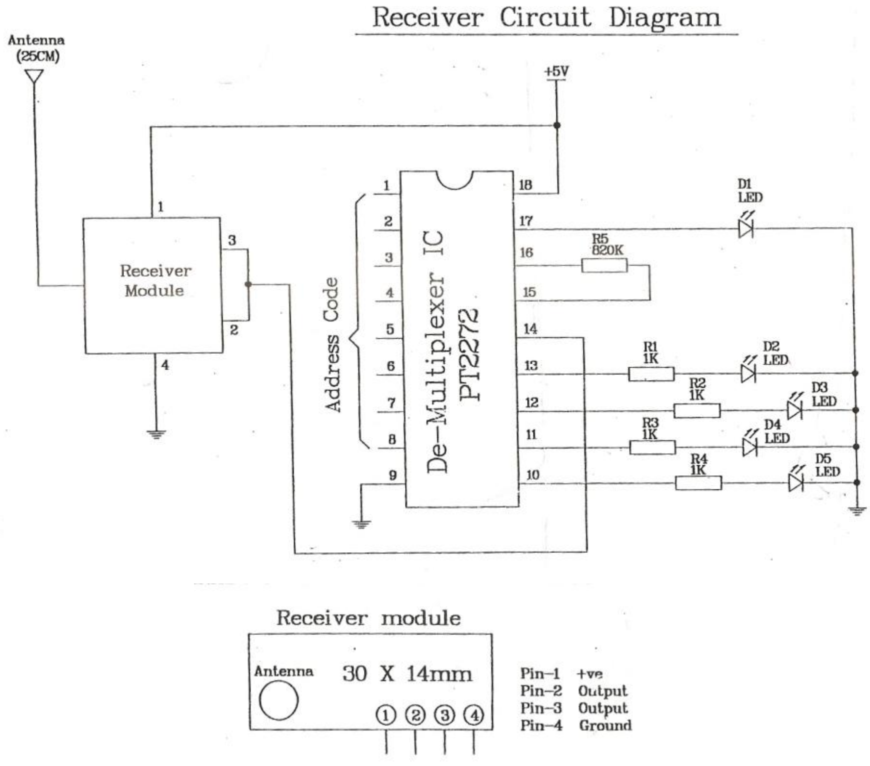

Figure 22: A 315MHz Receiver Schematic

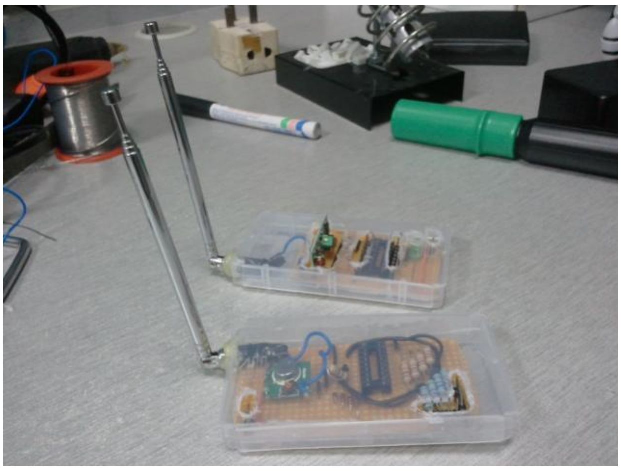

Figure 23: The Designed 315MHz Transmitter and Receiver Circuit

4.7 Selection Of The Most Effective Prototype Platform

The platform could either be a black surface with white lines, or vice versa. However, after thorough studies and research, it was found that the LDR sensors are more reliable with a white surface with black lines instead. A low voltage (about 0.76V) will be obtained for a black surface, and a high voltage (about 4.22V) will be obtained for a white surface. Using this, the sensor outputs could be used by the PIC to control the movement algorithm. Another main factor of the platform was the formation of beds and the track. The robot would stop in front of every patients bed, which is the junction marked on the line. Thus, there will be six junctions in this platform, with a separate START and END position for the Isolation Ward Dispenser Platform.

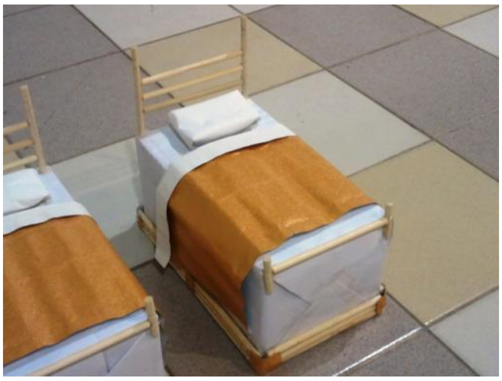

Figure 24: Isolation Ward Patients Bed Prototype

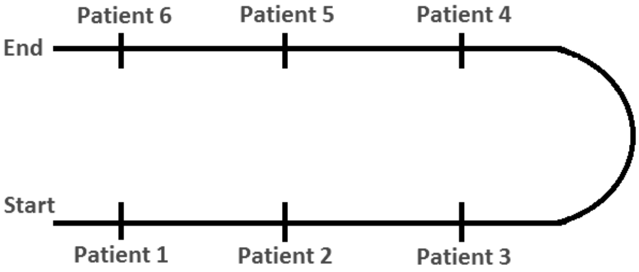

Figure 25: Isolation Ward Platform Formation

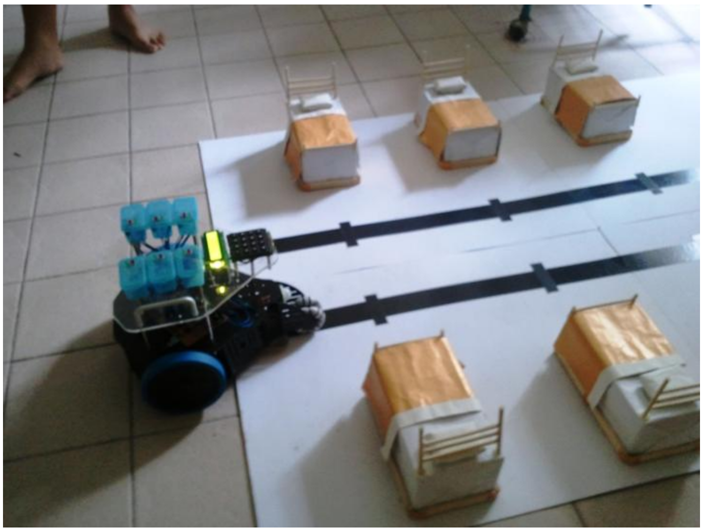

Figure 26: The Prototype Platform and Ward

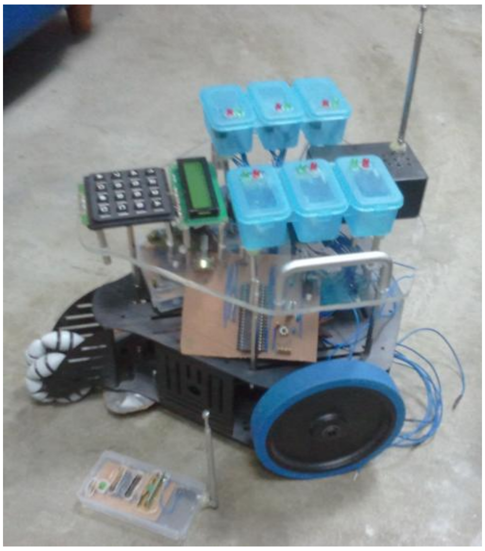

Figure 27: The Isolation Ward Dispenser Platform

4.8 PIC pin configuration

A total of two PICs were used for the Isolation Ward Dispenser Platform. The first PIC will be integrated with the Security Module, Compartments Module and the Emergency Call Module. On the other hand, the second PIC will be integrated with the Line Tracking Module, Movement Module and the Motion Detection Module.

The Intercom Module is an independent circuit where it is not linked to any of the PICs mentioned above.

The following are the PIC pin configurations for both the PICs above.

Pin Number	Pin Function	Connected to
1	MCLR	RESET BUTTON
2	RA0	K1
3	RA1	K2
4	RA2	K3
5	RA3	K4
6	RA4	K8
7	RA5	K7
8	RE0	K6
9	RE1	K5
10	RE2	Green LED 1
11	Vdd	5V
12	Vss	GND
13	OSC1	CRYSTAL
14	OSC2	CRYSTAL
15	RC0	RS
16	RC1/CCP2	E
17	RC2/CCP1	Red LED 1
18	RC3	Compartment Switch 1
19	RD0	DB0
20	RD1	DB1
21	RD2	DB2
22	RD3	DB3
23	RC4	Compartment Door Switch 1
24	RC5	Green LED 2
25	RC6	Red LED 2
26	RC7	Compartment Switch 2
27	RD4	DB4
28	RD5	DB5
29	RD6	DB6
30	RD7	DB7
31	Vss	GND
32	Vdd	5V
33	RB0	Compartment Door Switch 2
34	RB1	Green LED 6
35	RB2	BUZZER
36	RB3	-

37	RB4	Red LED 6
38	RB5	Compartment Switch 6
39	RB6	Compartment Door Switch 6
40	RB7	-

Table 7: PIC 1 Configuration

Pin Number	Pin Function	Connected to
1	MCLR	RESET
2	RA0	-
3	RA1	-
4	RA2	-
5	RA3	-
6	RA4	-
7	RA5	-
8	RE0	-
9	RE1	-
10	RE2	-
11	Vdd	5V
12	Vss	GND
13	OSC1	CRYSTAL
14	OSC2	CRYSTAL
15	RC0	5
16	RC1/CCP2	6
17	RC2/CCP1	11
18	RC3	7
19	RD0	-
20	RD1	-
21	RD2	-
22	RD3	-
23	RC4	10
24	RC5	12
25	RC6	PIR SIGNAL
26	RC7	BUZZER
27	RD4	-
28	RD5	-
29	RD6	-
30	RD7	-
31	Vss	GND
32	Vdd	5V
33	RB0	OUT1
34	RB1	OUT2
35	RB2	OUT3

36	RB3	OUT4
37	RB4	OUT5
38	RB5	-
39	RB6	-
40	RB7	-

Table 8: PIC 2 Configuration

4.9 Isolation Ward Dispenser Platform Operation Principle

When the start button is pushed, the robot will immediately follow the line and stops at junction 1, which is located behind patient 1's bed. Once the robot stops, the buzzer will beep once to alert the patient of the robots presence. Assuming that the patient is aware of the robots presence, the LCD would display the password prompt interface.

The patient would have to input their patient ID, and then press 'A' to unlock their compartment. Once the compartment is unlocked, the LED turns green, and the alarm is disabled, where the buzzer would not beep when the first compartment door is opened. Once the product is removed, the limit switch becomes open, and after a delay of three seconds, the robot would proceed to junction 2.

In an event if the patient is not on the respective bed, the buzzer will again beep, but twice this time. If there are still no responses within 10 seconds, the robot would proceed to patient 2's bed. In another scenario, if the person who attends the robot is not patient 1, and assuming that the wrong password is input, the buzzer will beep continuously. The system will prompt for the password again.

And, if patient 1 wrongly opens the wrong compartment, or remove the product from the wrong compartment, the buzzer will beep until the product is replaced into it, and the door is closed. If one of the emergency buttons is pushed, the robot skips the immediate next patient and attends the patient that pushed the button first.

Once done, it would continue the cycle from the previous area. In case the robot senses motion within 30cm from the robots front, it would stop, and the buzzer would beep until the obstruction is cleared from the path. After three seconds of delay, the robot would move again.

CHAPTER 5

5.1 CONCLUSION

As a conclusion, this project was a comprehensive research study and design of the Isolation Ward Dispenser Platform, which aids both the patients and hospital staff during the outbreak of any contagious disease outbreak. The Isolation Ward Dispenser Platform prevents the medical staff from being infected by any viral diseases, and also assists them in terms of time management. More patients could be attended in one go, and in a shorter period of time. The project was related to the application of the line tracking robot movement which is widely used in the industries. The study on appropriate compartments, movement sensors, intercom system, LCD and keypad also played a major part in turning the Isolation Ward Dispenser Platform into reality. The line tracking module, motion detection module, emergency call module, security module, compartments module and the intercom module was successfully incorporated onto a single platform. However, due to time and cost constraint, the RFID system was replaced with the keypad and LCD, and the electromagnetic locks were replaced with compartments with limit switches instead.

5.2 RECOMMENDATION

The Isolation Ward Dispenser Platform was an useful development in order to ease the medical staffs job. However, it still could be improved in numerous ways. Firstly, the security module which prevents patients from retrieving the wrong medications could be replaced with the RFID module. With this, the patient would only have to scan his or her tag onto the module on the platform, and thus unlocking the respective compartment.

Next, the compartment module is to be improved using the electromagnetic lock. Thus, when the patient verification system is complete, this releases the electromagnetic lock, which unlocks the respective compartment. With this, the patients would not have to manually open the compartments.

In order for the platform to function and mobilize using LDR sensors, a grid of black lines are needed on the floor surface. Thus, hospitals which intend to implement this system has to prepare a ward with black lines drawn around it. To overcome this, ultrasonic sensors could be used on all four sides of the platform, which allows it to manoeuvre intelligently without hitting any obstacles.

The intercom module used for this platform is not effective as it contains noises, and the output volume is too low. To improve this, a filter circuit and an audio amplifier circuit could be added onto both ends of the intercom circuits.

Last but not least, the platform would be a hit in the market of medical world if it could automatically retrieve the medication and dispense it to the patients via a gripper. Besides, it would be better if the platform could record medical measurements such as the pulse rate, temperature, and pressure and so on.

REFERENCES

[1] Oxford Dictionaries. 2010, Retrieved 17 February 2012. *Definition of Robotics.*

[2] Abdlmonem H. Beitelmal et al. *Microcontroller for Controlling an Actuator.* Patent number: 7902966. Issue date: 8 March 2011. Filing date: 18 June 2007. Application number: 11/764,410.

[3] Howard J. Morrison et al, *Walkie-talkie Device*, Patent number: 3939418, Filing date: 7 Oct 1974, Issue date: 17 February 1976

[4] Le Masurier, Guy C.; Lee, Sarah M.; Tudor-Locke, Catrine, Medicine & Science in Sports & Exercise:, May 2004 - Volume 36 - Issue 5 - pp 905-910, APPLIED SCIENCES: Physical Fitness and Performance Motion Sensor Accuracy under Controlled and Free-Living Conditions

[5] Mizoguchi F., Wen W., *User Selectable Authentication Interface and Universal Password Oracle*, Patent number: 0030924, Pub. Date: 12 February, 2004

[6] Patrick S. Brady, Michael J. Kenning, David A. Roberts, Robert E.V. Semos, Mark J. Stirland, Richard B. Ward, *Personal Identification Systems*, Patent number: 5606614, Patent date: 25 February 1997

[7] Elie J. Jreij, Tave P. Dunn, *Method and System for Providing Password Protection*, Patent no: 6178236, Patent date: 23 January 2001

APPENDICES

Figure 28: The Stingray Robot kit, which was used as the Isolation Ward Dispenser Robot Base

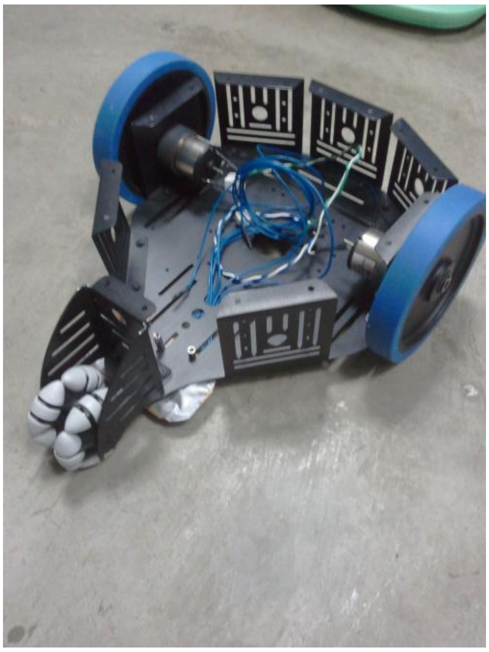

Figure 29: The Isolation Ward Dispenser Robot Base with two 7.2V DC Motors attached to the Wheels

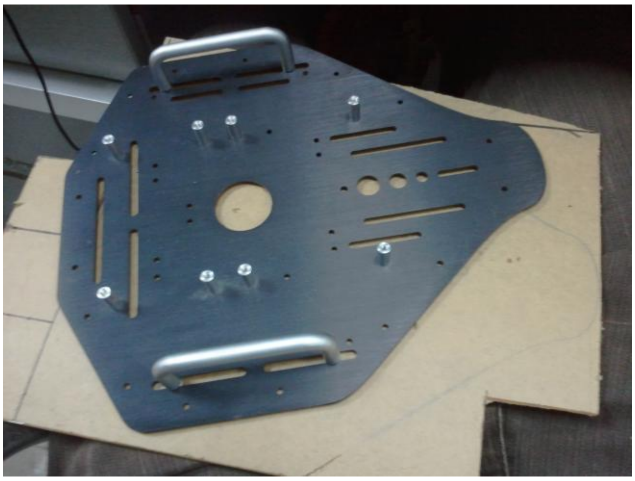

Figure 30: Acrylic Forming the Upper Level of the Isolation Ward Dispenser Robot

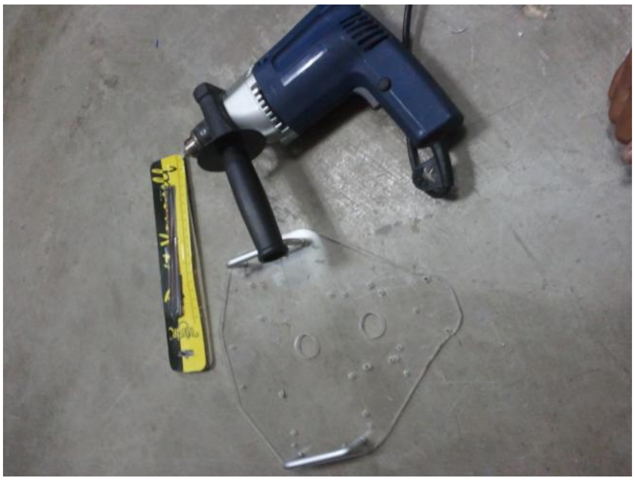

Figure 31: Acrylic Forming Upper Level of the Isolation Ward Dispenser Robot After Cutting and Drilling Process

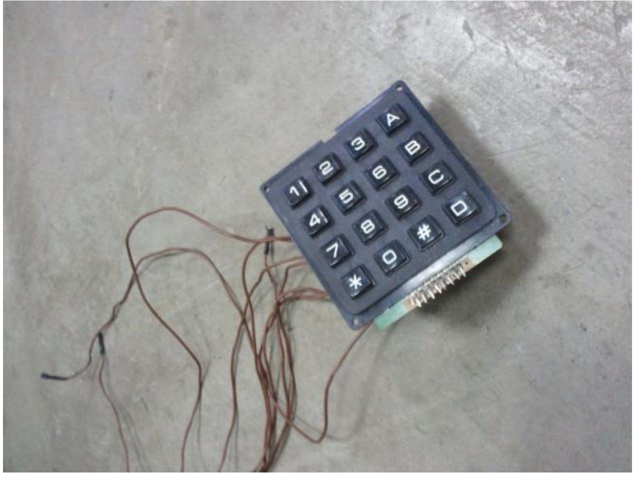

Figure 32: The 4x4 Keypad as the Password Input Device

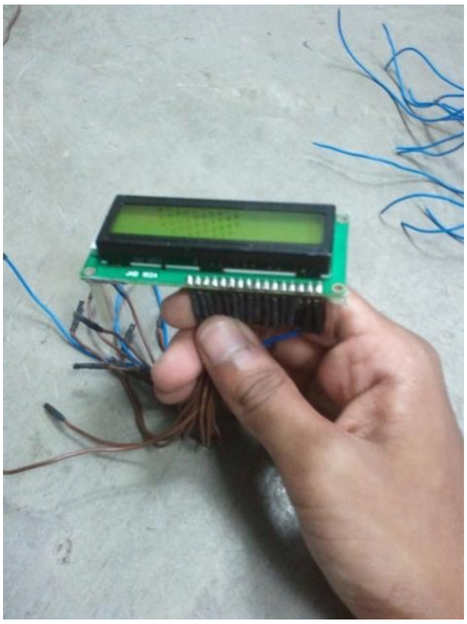

Figure 33: The 2x16 LCD Display for Patient Name Display

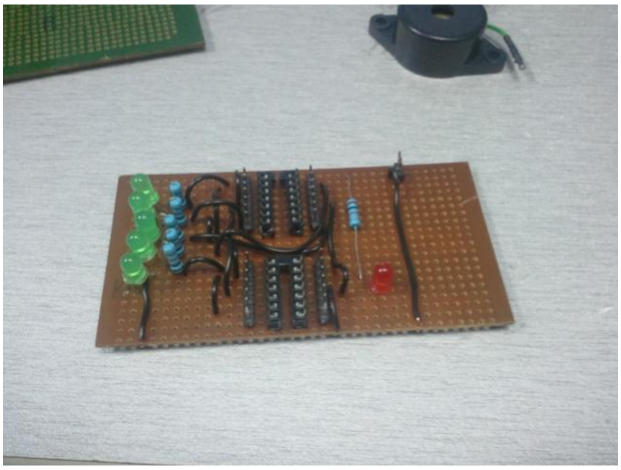

Figure 34: The LDR Sensor Comparator Circuitry

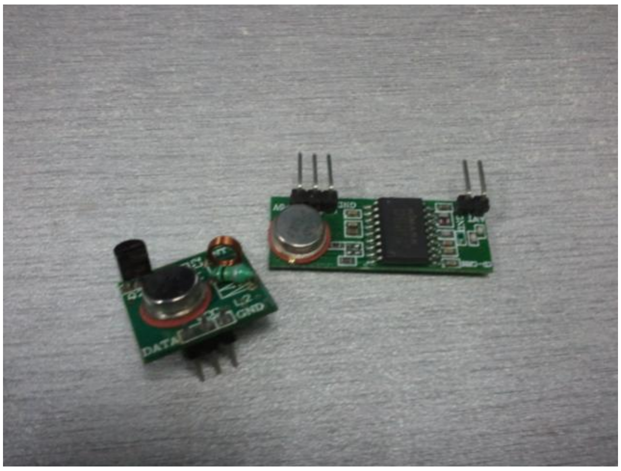

Figure 35: 315MHz Transmitter and Receiver Module for the Emergency Buttons Function

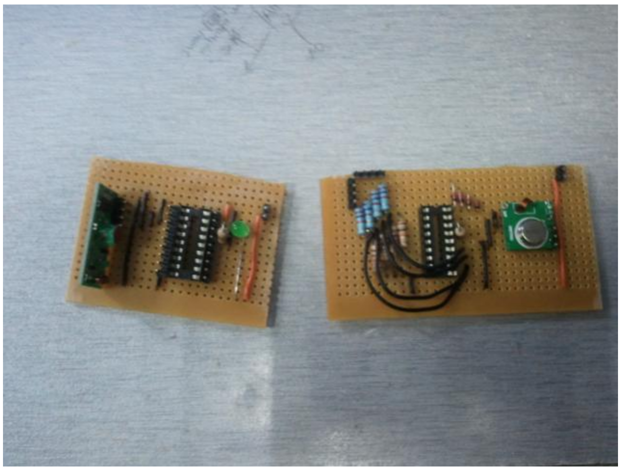

Figure 36: The Transmitter and Receiver Circuitry After Soldering

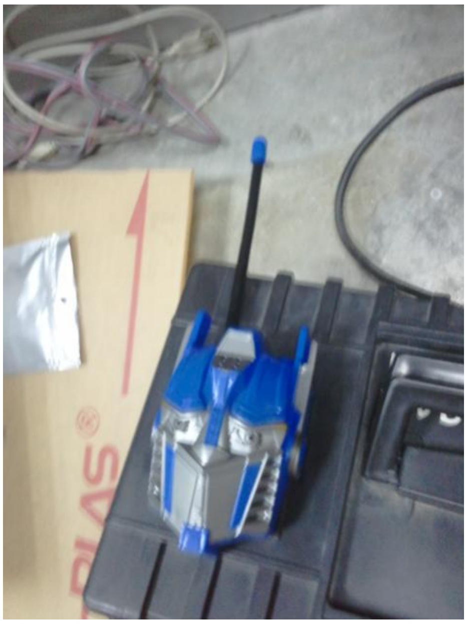

Figure 37: The Walkie Talkie Set Before Dismantling

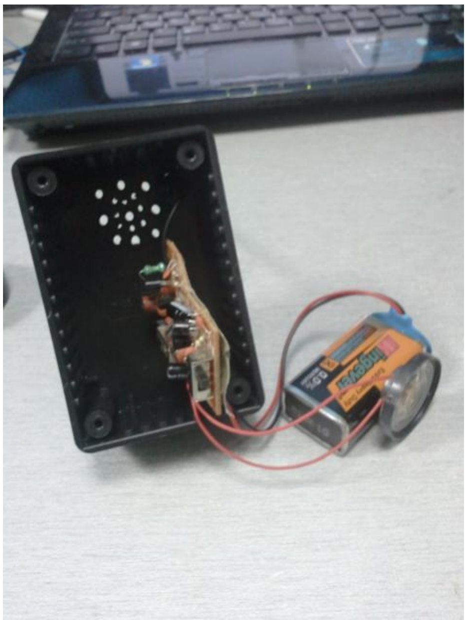

Figure 38: The Walkie Talkie Circuitry Placed in the Black Casing

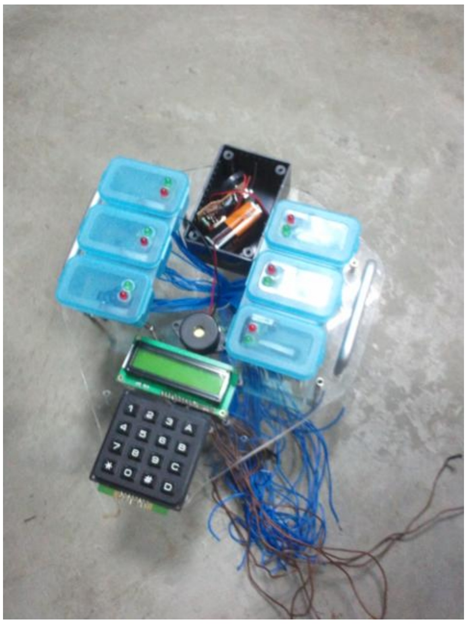

Figure 39: After Placing the Components Onto the Upper Level of the Isolation Ward Dispenser Robot

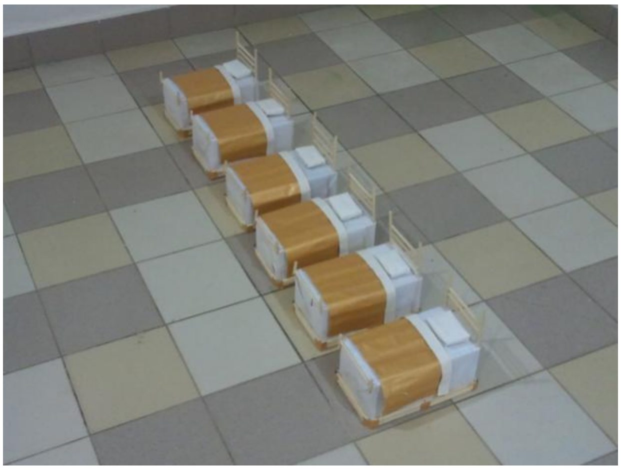

Figure 40: Six Isolation Ward Prototype Patients Bed

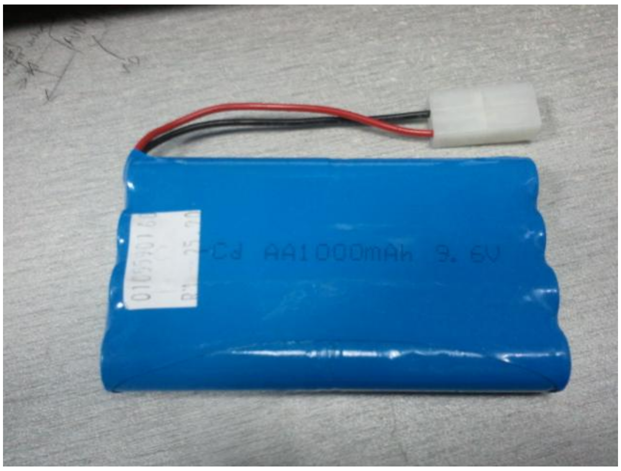

Figure 41: The 9.6V Ni Cd Rechargeable Battery Pack as the Power Supply

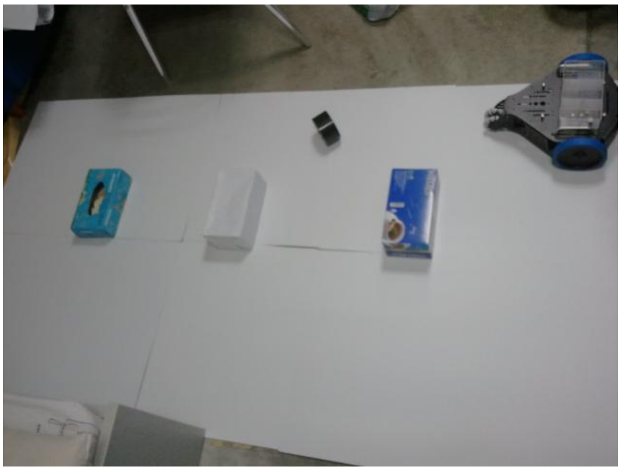

Figure 42: Construction of the Isolation Ward Platform

Isolation Ward Dispenser Platform Algorithm

The Isolation Ward Dispenser Platform operates based on C++ codes, which is programmed to the PICs via the MPLAB IDE 7.20 software. The codes were separated based on functions, and then combined as one file. However, the codes will be separated based on modules to ease the troubleshooting process in the future.

Motion Detection Module

```
//==========================================================================
//        include
//==========================================================================
#include <pic.h>
```

```
//    configuration
//==========================================================================
__CONFIG (0x3F32);

//    define
//==========================================================================
#define       alarm_set      RA0
#define       sensor         RB2
#define buzzer        RC4

//    main function
//==========================================================================
void main(void)
{
    unsigned char status;
    unsigned long temp1,temp2;

    ADCON1 = 0x06;                                              //Congifure Port A as digital I/O

    TRISA = 0b11111111;                                         //Configure Port A as Input
    TRISB = 0b11111111;                                         //Configure Port B as Input
    TRISC = 0b00000000;                                         //Configure Port C as Output

    status=0;
    buzzer=0;

    while(1)                                                    //Infinity Loop
    {
            //scan input
            if((alarm_set==0)&&(status!=3))                     //Alarm set
            {
                    while(alarm_set==0) continue;
                    status=1;
                    temp1=0;
            }

            if((sensor==1)&&(status==2))            //Motion detected
                    status=3;

            //processing output
            switch(status)
            {
                    //Permitted period mode (15 seconds)
                    case 1:

                            temp1+=1;
                            if(temp1<10000) buzzer=1;       //Sound 'Beep'
                            else if(temp1>1000000)
                            {
                                    temp1=0;
                                    status=2;
                            }
                            else buzzer=0;
                            break;

                    //Scanning mode
                    case 2:
                            temp2+=1;
                            if(temp2<2500);
                            else if(temp2<200000) ;
                            else temp2=0;
                            break;

                    //Detected mode
                    case 3:
                            temp2+=1;
                            if(temp2<40000)                     // buzzer activated
                            {
```

```
                                        buzzer=1;
                                }
                                else if(temp2<60000)
                                {

                                        buzzer=0;
                                }
                                else temp2=0;
                                break;
                }
        }
}
```

Line Tracking and Movement Module

```
//=========================================================================================
//      include
//=========================================================================================
#include <pic.h>

//      configuration
//=========================================================================================
__CONFIG ( 0x3F32 );

//      define
//=========================================================================================
#define sw1                             RE0
#define sw2                             RE1
#define motor_ra RC0
#define motor_rb RC3
#define motor_la RC4
#define motor_lb RC5

#define s_left                  RB0
#define s_mleft                 RB1
#define s_mright                RB2
#define s_right                 RB3

#define buzzer                  RE2

#define         rs                      RB7
#define         e                       RB6
#define         lcd_data                PORTD
#define b_light                  RB5
#define SPEEDL                  CCPR1L
#define SPEEDR                  CCPR2L


//      global variable
//=========================================================================================unsigned
char data[6] = {0};
const unsigned char line [] = {"1.LINE FOLLOW"};

unsigned int result;
unsigned int To=0,T=0,TH=0;
unsigned char REC;
unsigned char i=0,raw;

unsigned int us_value (unsigned char mode);

//      function prototype
//=========================================================================================
void init(void);
void delay(unsigned long data);
void send_config(unsigned char data);
void send_char(unsigned char data);
void e_pulse(void);
void lcd_goto(unsigned char data);
void lcd_clr(void);
void send_string(const char *s);
void dis_num(unsigned long data);

void line_follow(void);
void ultrasonic(void);
void wireless_xbee(void);
void analog_sen(void);
void SKPS_PScon(void);

void forward(void);
void stop (void);
void backward (void);
void reverse (void);
void left(void);
void right(void);

void uart_send(unsigned char data);
unsigned char uart_rec(void);
unsigned char skps(unsigned char data);
void skps_vibrate(unsigned char motor, unsigned char value);
void read_adc(char config);


//      interrupt prototype
//===========================================================================================static  void
interrupt isr(void)
{
        if (TMR0IF)                                     // TMR0 is overflow
        {
                TMR0IF = 0;                             // clear flag bit
                To +=0x100;                             // count number of TMR0 overflow ( make it to 16bit TMR)
        }

        if(RBIF)                                        // there is change bit on RB4-RB7
        {
                RBIF = 0;                               //                  ____
                if (RB4)                                // Rb4 is 1 mean is rising form 0  __|
                {
                        TMR0 = 0;                       // clear all counter involved, start new count for period of RB4 high
                        To = 0;
                }
                                                                                                  //
                                                                        ___
                else TH = TMR0 + To;                    // RB4 is 0 mean is falling form 1          |_____  //  save  TH,  RB4
high period
        }

        if(RCIF)
        {
                RCIF = 0;                               // clear flag bit

                if (RCREG == 'R') data[i=0]= RCREG;     // check if start byte 'R' is met
                else if (RCREG == 100) data[i=0]= RCREG;// check if start byte 'd'(decimal 100) is met
                if ((data[0] == 'R'))data [i++] = RCREG;        // save the data in data array
                if (i>4) i = 4;
                                                        // if the data array reached max, set the index to 4
        }
}


//      main function
//===========================================================================================void
main(void)
{
        unsigned char m=0,i =0;
        delay(20000);
        init();                                                                                    //
initiate cnfiguration and initial condition
        buzzer = 1;                                                                                //
inditcate the circuit is on with beep
        lcd_clr();                                                                     // clear the LCD
screen
        send_string("Select mode");                                    // display "select mode"
        lcd_goto(20);                                                                  // move to 2nd
line
        send_string(mode[m]);                                                  // display string according
to the mode
        buzzer = 0;                                                                                //
stop beep

        while(1)                                                                       // loop
        {
                if( !sw1)                                                              // if button SW1
is pressed
                {
                        while(!sw1);                                                   // wait ubtul
button is released
                        m++;
                        if ( m > 4) m = 0;                                     // if mode is added more
than three, set to zero
                        lcd_goto(20);                                                  // start display at
20
                        send_string(mode[m]);                                  // display string depend on
mode
                        send_string("       ");                        // space to overwrite long words
                }

                if (!sw2)                                                              // if button
SW2is pressed
                {
                        while(!sw2);                                                   // wait until
button is released
                        switch(m)                                                      // check what is
the current mode, execute the mode
                        {

                                case 0 :    line_follow();                     // mode 1 : line follow
                                                        break;
                                case 1 :    ultrasonic();                      // mode 2 : ultrasonic mode
                                                        break;
                                case 2 :    analog_sen();                      // mode 3 : analog sensor mode
                                                        break;
                                case 3 :    wireless_xbee();           // mode 4 : wireless xbee mode
                                                        break;
                                case 4 :  SKPS_PScon();                        // mode 5 : PS2 Controller Mode
                                                        break;
                                default :   ;
                        }
                }
        }
}

//==========================================================================================
// Initailization
// Description : Initialize the microcontroller
//==========================================================================================
void init()
{
        // ADC configuration
        ADCON1 = 0b10000100;                                   //set RA0 and RA1 as Analog Input, left justified
```

```
	// setup for capture pwm
	RBIE = 1;								// enable interrupt on change of port B

	// motor PWM configuration
	PR2 = 255;								// set period register
	T2CON =		0b00000100;				//
	CCP1CON =	0b00001100;				// config for RC1 to generate PWM		( for more detail
refer datasheet section 'capture/compare/pwm')
	CCP2CON =	0b00001100;				// config for RC2 to generate PWM

	// Tris configuration (input or output)
	TRISA = 0b00000011;						//set RA0 and RA2 pin as input,other as output
	TRISB = 0b00011111;						//set RB0-RB4 pin as input, other as output
	TRISC = 0b10000000;						//set PORTC pin as output
	TRISD = 0b00000000;						//set all PORTD pin as output
	TRISE = 0b00000011;

	// TMR 0 configuation
	T0CS = 0;
	PSA = 0;
	PS2 = 1;								// prescale 1:32
	PS1 = 1;								//
	PS0 = 1;								//
	TMR0IE = 1;								// TMR0 Interrupt
	TMR0 = 0;

	//setup UART
	SPBRG = 0x81;							//set baud rate to 9600 for 20Mhz
	BRGH = 1;									//baud rate high speed option
	TXEN = 1;									//enable transmission
	TX9 = 0;
	CREN = 1;									//enable reception
	SPEN = 1;								//enable serial port
	RX9 = 0;
	RCIE = 1;								//enable interrupt on eachdata received

	// enable all unmasked interrupt
	GIE = 1;
	PEIE = 1;

	// LCD configuration
	send_config(0b00000001);		//clear display at lcd
	send_config(0b00000010);		//Lcd Return to home
	send_config(0b00000110);		//entry mode-cursor increase 1
	send_config(0b00001100);		//diplay on, cursor off and cursor blink off
	send_config(0b00111000);		//function

	TX_PIN = 1;
	b_light = 0;
	buzzer = 0;
	stop();
}

//==========================================================================================
// Mode subroutine
//==========================================================================================
// Mode 1 : line follow subroutine
// Description: Program for the robot to follow line
// For more detail about line follow concept please refer PR5
//==========================================================================================
void line_follow()
{
	unsigned char memory;

	lcd_clr();									// clear lcd screen
	send_string("Position");				// display "position" string
```

```
     while(1)
     {
             if ((s_left==1)&&(s_mleft==0)&&(s_mright==0)&&(s_right==0))                                        //    if
only sensor left detected black line
             {       forward();
                                                         // motor forward
                     SPEEDL = 0;
                                                                 // left motor speed is 0
                     SPEEDR = 255;
                                                         // right motor speed is 255(full speed)
                     memory = PORTB&0b00001111;
                                                 // save current sensor position
                     lcd_goto(20);
                                                         // lcd go to 2nd line 1st character
                     send_string ("right  ");
                                         // display "right"mean the robot's position is on the right side of the line
             }
             else if ((s_left==1)&&(s_mleft==1)&&(s_mright==0)&&(s_right==0))                       // if only sensor
left detected black line
             {       forward();
                                                         // motor forward
                     SPEEDL = 180;
                                                         // left motor speed is 180
                     SPEEDR = 255;
                                                         // right motor speed is 255(full speed)
                     memory = PORTB&0b00001111;
                     lcd_goto(20);
                     send_string ("m_right2");
             }
             else if ((s_left==0)&&(s_mleft==1)&&(s_mright==0)&&(s_right==0))                       // if only sensor
middle left detected black line
             {       forward();
                                                         // motor forward
                     SPEEDL = 200;
                                                         // left motor speed is 200
                     SPEEDR = 255;
                                                         // right motor speed is 255(full speed)
                     memory = PORTB&0b00001111;
                     lcd_goto(20);
                     send_string ("m_right1  ");
             }
             else if ((s_left==1)&&(s_mleft==1)&&(s_mright==1)&&(s_right==0))                       //    if    sensor
middle left and sensor left detected black line
             {       forward();
                                                         // motor forward
                     SPEEDL = 200;
                                                         // left motor speed is 200
                     SPEEDR = 255;
                                                         // right motor speed is 255(full speed)
                     memory = PORTB&0b00001111;
                     lcd_goto(20);
                     send_string ("m_right1  ");
             }
             else if ((s_left==0)&&(s_mleft==1)&&(s_mright==1)&&(s_right==0))                       //    if    sensor
middle left and sensor middle right detected black line
             {       forward();
                                                         // motor forward
                     SPEEDL = 255;
                                                         // left motor speed is 255(full speed)
                     SPEEDR = 255;
                                                         // right motor speed is 255(full speed)
                     memory = PORTB&0b00001111;
                     lcd_goto(20);
                     send_string ("middle ");
             }
             else if ((s_left==0)&&(s_mleft==0)&&(s_mright==1)&&(s_right==0))                       // if only sensor
middle right detected black line
```

```
                {           forward();
                                                            // motor forward
                            SPEEDL = 255;
                                                            // left motor speed is 255(full speed)
                            SPEEDR = 200;
                                                            // right motor speed is 200
                            memory = PORTB&0b00001111;
                            lcd_goto(20);
                            send_string ("m_left1   ");
                }
                else if ((s_left==0)&&(s_mleft==1)&&(s_mright==1)&&(s_right==1))              //    if    sensor
middle left, sensor middle right and sensor right detected black line
                {           forward();
                                                            // motor forward
                            SPEEDL = 255;
                                                            // left motor speed is 255(full speed)
                            SPEEDR = 200;
                                                            // right motor speed is 200
                            memory = PORTB&0b00001111;
                            lcd_goto(20);
                            send_string ("m_left1   ");
                }
                else if ((s_left==0)&&(s_mleft==0)&&(s_mright==1)&&(s_right==1))              // if sensor right
and sensor middle right detected black line
                {           forward();
                                                            // motor forward
                            SPEEDL = 255;
                                                            // left motor speed is 255(full speed)
                            SPEEDR = 180;
                                                            // right motor speed is 180
                            memory = PORTB&0b00001111;
                            lcd_goto(20);
                            send_string ("m_left2 ");
                }
                else if ((s_left==0)&&(s_mleft==0)&&(s_mright==0)&&(s_right==1))              // if only sensor
right detected black line
                {           forward();
                                                            // motor forward
                            SPEEDL = 255;
                                                            // left motor speed is 255(full speed)
                            SPEEDR = 0;
                                                                    // right motor speed is 0
                            memory = PORTB&0b00001111;
                            lcd_goto(20);
                            send_string ("left   ");
                }
                else if ((s_left==0)&&(s_mleft==0)&&(s_mright==0)&&(s_right==0))              //   if   all   sensor
coult not detected black line
                {           forward();
                                                            // motor forward
                            if ((memory == 0b00000001)||(memory == 0b00000011)||(memory == 0b0000010)||(memory ==
0b0000111))
                            {
                                    SPEEDL = 0;
                                                                    // left motor speed is 0
                                    SPEEDR = 255;
                                                            // right motor speed is 255(full speed)
                            }
                            else if ((memory == 0b00001000)||(memory == 0b0000100)||(memory == 0b00001100)||(memory ==
0b0001110))
                            {
                                    SPEEDL = 255;
                                                            // left motor speed is 255(full speed)
                                    SPEEDR = 0;
                                                                    // right motor speed is 0
                            }
                }
```

```
			else if ((s_left==1)&&(s_mleft==1)&&(s_mright==1)&&(s_right==1))				// if all sensor
detected black line
			{
				forward();
										// motor forward
			}
		}
}

//============================================================================================
// Motor control function
// Description : subroutine to set the robot moving direction
//============================================================================================
void forward ()
{
	motor_ra = 0;
	motor_rb = 1;
	motor_la = 0;
	motor_lb = 1;
}

void backward ()
{
	motor_ra = 1;
	motor_rb = 0;
	motor_la = 1;
	motor_lb = 0;
}

void left()
{
	motor_la = 1;
	motor_lb = 0;
	motor_ra = 0;
	motor_rb = 1;
}
void right()
{
	motor_la = 0;
	motor_lb = 1;
	motor_ra = 1;
	motor_rb = 0;
}

void stop()
{
	motor_la = 1;
	motor_lb = 1;
	motor_ra = 1;
	motor_rb = 1;
}
```

Security Module

```
//============================================================================================
//	include
//============================================================================================
#include <pic.h>

//============================================================================================
//	configuration
//============================================================================================
__CONFIG ( 0x3F32 );
```

```
//==========================================================================================
//       define
//==========================================================================================
#define          rs                              RC0
#define          e                               RC1
#define          lcd_data                        PORTD
#define buzzer                                   RB2
#define green_led_1                              RE2
#define red_led_1                                RC2
#define door_limit_sw_1                          RD3
#define comp_limit_sw_1                          RC3
#define green_led_2                              RC5
#define red_led_2                                RC7
#define door_limit_sw_2                          RB0
#define comp_limit_sw_2                          RC6
#define green_led_6                              RB1
#define red_led_6                                RB4
#define door_limit_sw_6                          RB6
#define comp_limit_sw_6                          RB5
#define DB3                                      RC4
//===========================================================================================
//       function prototype
//===========================================================================================
void delay(unsigned long data);
void send_config(unsigned char data);
void send_char(unsigned char data);
void e_pulse(void);
void lcd_goto(unsigned char data);
void lcd_clr(void);
void send_string(const char *s);
void clearrow1(void);
void clearrow2(void);
void clearrow3(void);
void clearrow4(void);
void scancolumn1(void);
void scancolumn2(void);
void scancolumn3(void);
void scancolumn4(void);
void beep_once(void);
void beep_twice(void);

//============================================================================================
//       global variable
//============================================================================================
unsigned char password_count=0;
unsigned char keyin_char[3];                     // Declare an array to stall the 6-digit key in password
unsigned char stalled_char1[1]="1";              // Declare an array to stall the 6-digit desired password
unsigned char stalled_char2[1]="3";
unsigned char stalled_char3[1]="6";
//============================================================================================
//       main function
//============================================================================================
void main(void)
{
         ADCON1=0b00000110;           //set all portA pins as digital I/O
         TRISA=0b11001111;            //clear bit 4&5 portA as output and set the rest as input
         TRISB=0b01100001;            //set portB as output
         TRISD=0b00001000;            //set portD as output
         TRISC=0b01001000;            //set bit4-7 portC as input(connected to 4 row of keypad)
         TRISE=0b00000000;            //set portE as output

         PORTC=0;
         PORTD=0;


red_led_1=1;
red_led_2=1;
red_led_6=1;
```

```
green_led_1=0;
green_led_2=0;
green_led_6=0;
buzzer=0;

//send_config(0b00001001); //clear display at lcd
//      send_config(0b00000010);        //Lcd Return to home
//      send_config(0b00000110);        //entry mode-cursor increase 1
//      send_config(0b00001100);        //diplay on, cursor off and cursor blink off
//      send_config(0b00111000);        //function

//      lcd_clr();                                              //clear LCD
//      delay(1000);                                            //delay
//      lcd_goto(0);                                            //initial display
//      send_string("PLEASE ENTER");                            //Display "PLEASE ENTER" on lcd
//      lcd_goto(20);                                           //Display on 2nd line
//      send_string("6-DIGIT PASSWORD");                        //Display "6-DIGIT PASSWORD" on lcd


while(1)
{
                clearrow1();                                    //Clear 1st output pin and set the others
                scancolumn1();                                  //scan column 1-4
                clearrow2();                                    //Clear 2nd output pin and set the others
                scancolumn2();                                  //scan column
                clearrow3();                                    //Clear 3rd output pin and set the others
                scancolumn3();                                  //scan column
                clearrow4();                                    //Clear 4th output pin and set the others
                scancolumn4();                                  //scan column

        if(password_count==1)
        {
        password_count=0;
        break;
        }
        {
                if ((door_limit_sw_1==0)||(door_limit_sw_2==0)||(door_limit_sw_6==0))
                {
                buzzer=1;
                }
                else
                {
                buzzer=0;
                }

        }
}



    while(1)
                {

        while(1)
        {
                        keyin_char[0]="";

                                //keypad scanning algorithm
                clearrow1();            //Clear 1st output pin and set the others
                scancolumn1();          //scan column 1-4
                clearrow2();            //Clear 2nd output pin and set the others
                scancolumn2();          //scan column
                clearrow3();            //Clear 3rd output pin and set the others
                scancolumn3();          //scan column
                clearrow4();            //Clear 4th output pin and set the others
                scancolumn4();          //scan column

                if(password_count==1)
```

```
			{
				password_count=0;
				if((keyin_char[0]==stalled_char1[0]))		//compare the keyin value with stalled value to test 
whether password is correct
					{
					//lcd_clr();				//clear lcd
					//lcd_goto(0);
					//send_string("SUCCESS!");			//display SUCCESS
					beep_once();				//beep one time
					green_led_1=1;
					red_led_1=0;

								while(1)
								{
									if(door_limit_sw_1==0)
										{
											while(1)
											{

	if(door_limit_sw_1==1)
												{


	green_led_1=0;

	break;
												}
											}
										break;
										}
								}
				//	while(1);
					break;				//infinity loop
					}

				else
					{
				//	lcd_clr();				//clear lcd
				//	lcd_goto(0);
				//	send_string("ERROR!");			//display ERROR!
					beep_twice();
				//	while(1);				//infinity loop
					}

			}


		}
////////////////////////////////////////////////////////////////////////////////////////////////////
while(1)
{
			clearrow1();				//Clear 1st output pin and set the others
			scancolumn1();				//scan column 1-4
			clearrow2();				//Clear 2nd output pin and set the others
			scancolumn2();				//scan column
			clearrow3();				//Clear 3rd output pin and set the others
			scancolumn3();				//scan column
			clearrow4();				//Clear 4th output pin and set the others
			scancolumn4();				//scan column

		if(password_count==1)
		{
		password_count=0;
		break;
		}
		{
			if ((door_limit_sw_2==0)||(door_limit_sw_6==0))
			{
          buzzer=1;
          }
          else
          {
          buzzer=0;
          }

      }
}
/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
      while(1)
      {
          keyin_char[0]="";

          clearrow1();                    //Clear 1st output pin and set the others
          scancolumn1();                  //scan column 1-4
          clearrow2();                    //Clear 2nd output pin and set the others
          scancolumn2();                  //scan column
          clearrow3();                    //Clear 3rd output pin and set the others
          scancolumn3();                  //scan column
          clearrow4();                    //Clear 4th output pin and set the others
          scancolumn4();                  //scan column

      if(password_count==1)
          {
              password_count=0;
              if((keyin_char[0]==stalled_char2[0]))   //compare the keyin value with stalled value to test
whether password is correct
                  {
                  //lcd_clr();                                      //clear lcd
                  //lcd_goto(0);
                  //send_string("SUCCESS!");        //display SUCCESS
                  beep_once();                                    //beep one time
                  green_led_2=1;
                  red_led_2=0;
                              while(1)
                              {
                                  if(door_limit_sw_2==0)
                                      {
                                          while(1)
                                          {

  if(door_limit_sw_2==1)
                                              {


  green_led_2=0;

  break;
                                              }
                                          }
                                      break;
                                      }
                              }
              //  while(1);                                       //infinity loop
                  break;
                  }

              else
                  {
              //  lcd_clr();                                      //clear lcd
              //  lcd_goto(0);
              //  send_string("ERROR!");              //display ERROR!
                  beep_twice();
              //  while(1);                                       //infinity loop
                  }
          }
      }
```

```
//////////////////////////////////////////////////////////////////////////////
while(1)
{
                    clearrow1();                                //Clear 1st output pin and set the others
                    scancolumn1();                              //scan column 1-4
                    clearrow2();                                //Clear 2nd output pin and set the others
                    scancolumn2();                              //scan column
                    clearrow3();                                //Clear 3rd output pin and set the others
                    scancolumn3();                              //scan column
                    clearrow4();                                //Clear 4th output pin and set the others
                    scancolumn4();                              //scan column

            if(password_count==1)
            {
            password_count=0;
            break;
            }
            {
                    if (door_limit_sw_6==0)
                    {
                    buzzer=1;
                    }
                    else
                    {
                    buzzer=0;
                    }

            }
}

/////////////////////////////////////////////////////////////////////////////////
            while(1)
            {
keyin_char[0]="";

                    clearrow1();                                //Clear 1st output pin and set the others
                    scancolumn1();                              //scan column 1-4
                    clearrow2();                                //Clear 2nd output pin and set the others
                    scancolumn2();                              //scan column
                    clearrow3();                                //Clear 3rd output pin and set the others
                    scancolumn3();                              //scan column
                    clearrow4();                                //Clear 4th output pin and set the others
                    scancolumn4();                              //scan column

            if(password_count==1)
                    {
                            password_count=0;
                            if((keyin_char[0]==stalled_char3[0]))       //compare the keyin value with stalled value to test
whether password is correct
                                    {
                                    //lcd_clr();                                        //clear lcd
                                    //lcd_goto(0);
                                    //send_string("SUCCESS!");          //display SUCCESS
                                    beep_once();                                        //beep one time
                                    green_led_6=1;
                                    red_led_6=0;
                                                    while(1)
                                                        {
                                                            if(door_limit_sw_6==0)
                                                                    {
                                                                        while(1)
                                                                        {

    if(door_limit_sw_6==1)
                                                                                {
```

```
    green_led_6=0;

    break;
                                                                                            }
                                                                                    }
                                                                            break;
                                                                            }
                                                            }
                            //      while(1);                                           //infinity loop
                                    break;
                                    }
                            else
                                    {
                            //      lcd_clr();                                          //clear lcd
                            //      lcd_goto(0);
                            //      send_string("ERROR!");                      //display ERROR!
                                    beep_twice();
                            //      while(1);                                           //infinity loop
                                    }
                    }
            }



    }
}


//=========================================================================================
//  scanning functions
//=========================================================================================
void clearrow1(void)                    //clear the 1st row and set the others
{
    RE1=0;                                              //RE1,RE0, RA5 and RA4 are the output pins from PIC which connect to 4 pins of keypad
    RE0=1;
    RA5=1;
    RA4=1;
}

void clearrow2(void)                    //clear the 2nd row and set the others
{
    RE1=1;
    RE0=0;
    RA5=1;
    RA4=1;
}

void clearrow3(void)                    //clear the 3rd row and set the others
{
    RE1=1;
    RE0=1;
    RA5=0;
    RA4=1;
}

void clearrow4(void)                    //clear the 4th roe and set the others
{
    RE1=1;
    RE0=1;
    RA5=1;
    RA4=0;
}

void scancolumn1(void)
{
    if(RA0==0)                                                          //if key '1' is being pressed
```

```
        {
                while(RA0==0)continue;                      //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);                //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                          //Display the symbol '*' at LCD
                keyin_char[password_count]='1'; //Stall the '1' value at the keyin_char array
                password_count+=1;                             //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA1==0)                                                              //if key '2' is being pressed
        {
                while(RA1==0)continue;                      //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);                //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                          //Display the symbol '*' at LCD
                keyin_char[password_count]='2'; //Stall the '2' value at the keyin_char array
                password_count+=1;                             //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA2==0)                                                 //if key '3' is being pressed
        {
                while(RA2==0)continue;                      //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);                //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                          //Display the symbol '*' at LCD
                keyin_char[password_count]='3'; //Stall the '3' value at the keyin_char array
                password_count+=1;                             //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA3==0)                                                 //if key 'A' is being pressed
        {
                while(RA3==0)continue;                      //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);                //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                          //Display the symbol '*' at LCD
                keyin_char[password_count]='A';          //Stall the 'A' value at the keyin_char array
                password_count+=1;                             //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
}

void scancolumn2(void)
{
        if(RA0==0)                                                                   //if key '4' is being pressed
        {
                while(RA0==0)continue;                      //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);                //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                          //Display the symbol '*' at LCD
                keyin_char[password_count]='4'; //Stall the '4' value at the keyin_char array
                password_count+=1;                             //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA1==0)                                                              //if key '5' is being pressed
        {
                while(RA1==0)continue;                      //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);                //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                          //Display the symbol '*' at LCD
                keyin_char[password_count]='5'; //Stall the '5' value at the keyin_char array
                password_count+=1;                             //increase the Password_count variable's value by 1 and the
result stall back to the variable
```

```
        }
        else if(RA2==0)                                         //if key '6' is being pressed
        {
                while(RA2==0)continue;                  //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);               //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                 //Display the symbol '*' at LCD
                keyin_char[password_count]='6'; //Stall the '6' value at the keyin_char array
                password_count+=1;                      //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA3==0)                                         //if key 'B' is being pressed
        {
                while(RA3==0)continue;                  //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);               //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                 //Display the symbol '*' at LCD
                keyin_char[password_count]='B';//Stall the 'B' value at the keyin_char array
                password_count+=1;                      //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
}

void scancolumn3(void)
{
        if(RA0==0)                                              //if key '7' is being pressed
        {
                while(RA0==0)continue;                  //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);               //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                 //Display the symbol '*' at LCD
                keyin_char[password_count]='7'; //Stall the '7' value at the keyin_char array
                password_count+=1;                      //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA1==0)                                         //if key '8' is being pressed
        {
                while(RA1==0)continue;                  //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);               //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                 //Display the symbol '*' at LCD
                keyin_char[password_count]='8'; //Stall the '8' value at the keyin_char array
                password_count+=1;                      //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA2==0)                                         //if key '9' is being pressed
        {
                while(RA2==0)continue;                  //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);               //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                 //Display the symbol '*' at LCD
                keyin_char[password_count]='9'; //Stall the '9' value at the keyin_char array
                password_count+=1;                      //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA3==0)                                         //if key 'C' is being pressed
        {
                while(RA3==0)continue;                  //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);               //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                 //Display the symbol '*' at LCD
                keyin_char[password_count]='C';//Stall the 'C' value at the keyin_char array
                password_count+=1;                      //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
}

void scancolumn4(void)
{
        if(RA0==0)                                              //if key '*' is being pressed
        {
                while(RA0==0)continue;                  //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);               //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                 //Display the symbol '*' at LCD
                keyin_char[password_count]='*'; //Stall the '*' value at the keyin_char array
                password_count+=1;                      //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA1==0)                                         //if key '0' is being pressed
        {
                while(RA1==0)continue;                  //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);               //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                 //Display the symbol '*' at LCD
                keyin_char[password_count]='0'; //Stall the '0' value at the keyin_char array
                password_count+=1;                      //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA2==0)                                         //if key '#' is being pressed
        {
                while(RA2==0)continue;                  //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);               //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                 //Display the symbol '*' at LCD
                keyin_char[password_count]='#'; //Stall the '#' value at the keyin_char array
                password_count+=1;                      //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
        else if(RA3==0)                                         //if key 'D' is being pressed
        {
                while(RA3==0)continue;                  //waiting the key to be released
                if(password_count==0)lcd_clr();  //Clear the LCD if the key is the 1st password
                lcd_goto(password_count);               //The cursor of LCD points to the column equivalent to the value of
password_count variable
                send_char('*');                                 //Display the symbol '*' at LCD
                keyin_char[password_count]='D';         //Stall the 'D' value at the keyin_char array
                password_count+=1;                      //increase the Password_count variable's value by 1 and the
result stall back to the variable
        }
}

//==========================================================================================
//      General Purpose         functions
//==========================================================================================
void delay(unsigned long data)
{
        for( ;data>0;data-=1);
}

void beep_once(void)
{
        buzzer=1;                                       //buzzer on
        delay(8000);
        buzzer=0;                                       //buzzer off
}
```

```
void beep_twice(void)
{
        buzzer=1;                                               //buzzer on
        delay(8000);
        buzzer=0;                                               //buzzer off
        delay(13000);
        buzzer=1;                                               //buzzer on
        delay(8000);
        buzzer=0;                                               //buzzer off
}

//=====================================================================================
//      LCD         functions
//=====================================================================================
void send_config(unsigned char data)
{
        rs=0;                                                                   //clear rs into config mode
        lcd_data=data;
        delay(50);
        e_pulse();
}

void send_char(unsigned char data)
{
        rs=1;                                                                   //set rs into write mode
        lcd_data=data;
        delay(50);
        e_pulse();
}

void e_pulse(void)
{
        e=1;
        delay(50);
        e=0;
        delay(50);
}

void lcd_goto(unsigned char data)
{
        if(data<16)
        {
                send_config(0x80+data);
        }
        else
        {
                data=data-20;
                send_config(0xc0+data);
        }
}

void lcd_clr(void)
{
        send_config(0x01);
        delay(50);
}

void send_string(const char *s)
{
        unsigned char i=0;
        while (s && *s)send_char (*s++);

}
```