SECURITY DOOR SYSTEM USING CARD READER

By

WAN HESMAN BIN WAN RUSLI

FINAL PROJECT REPORT

Submitted to the Electrical & Electronics Engineering Programme
in Partial Fulfillment of the Requirements
for the Degree
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Universiti Teknologi Petronas
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

SECURITY DOOR SYSTEM USING CARD READER

by

Wan Hesman bin Wan Rusli

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

Approved:

Mr. Patrick Sebastian
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

June 2009

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.

Wan Hesman bin Wan Rusli

ACKNOWLEDGEMENT

In the name of Allah, the most gracious and the most merciful.

First of all, the author would like to express lots of thanks to Allah S.W.T for the blessing and his lovely parents, Wan Rusli Bin Wan Yusof and Wan Nasirah binti Mohamed, also family members for their priceless support, encouragement, constant love and valuable advice.

The author would like to express his most gratitude to his supervisor, Mr. Patrick Sebastian for his guidance and readiness to assist the author throughout the process of completing this project. Mr. Patrick has given the author splendid opportunity to work with him as well as giving valuable opinions in making this project a success.

Special thanks go to Dr. Yap Vooi Voon, Lecturer from Electrical and Electronic Department for guiding the author in completing this project. His idea of setting up the project in modules helps the author a lot in developing the prototype.

Last but not least, the author would like to express his special thanks to all lecturers from Electrical and Electronic Department, Universiti Teknologi PETRONAS and all his colleagues for all the supports during the development of the project. The support and assistance coming from all parties involved in this project are really appreciated. The author sincerely would like to apologize for any mistakes that he made during this project.

Thank you to all.

ABSTRACT

This report describes about the research done about the title selected for the Final Year Project which named as Security Door System using Card Reader under Mr. Patrick Sebastian supervision. There are a lot of security door systems sold worldwide. This project is aimed to produce a low-cost security system that is able to use card reader as the interface. The card reader will read the information from the user's card and open the door. It based on PIC Microcontroller to control the authentication and access logging.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background of Study

This project emphasizes the idea of having a cost-effective security door system, using Wiegand card reader as the medium recognition. This project leads to the development of a system that uses PIC Microcontroller as the controller of the system. The system will be a robust system and comply with IEEE standards. In addition, the system will able be upgradable for further improvement in the future.

1.2 Problem Statement

Security system is a huge market with multi-million market. It has been projected that many people would buy this system and equipped it to their house or office. There has been a lot type of security system in the market, from small start-up company to big well-funded companies such as Mercury Security. Mostly of the products are expensive and majority of people could not afford to buy one. On the other hand, the security should be able to hold many data for office use. The system developed by the companies has limited expandability due to safety reason.

So, based on the problems mentioned before, a cost-effective security system has been build to overcome the problems.

1.3 Objectives

The objective of this project is to build a security door system by using a Wiegand card reader that is low-cost. In this project, PIC microcontroller is used as the heart of the system, and a Wiegand card reader as the intermediate between the user and the system.

Another qualification the card readers must meet was huge amount of capacity. It should be able to support from 100 to 1000 users which is usual number of employees for a small and medium business (SME). This project is going to use 8K Non-Volatile Memory (EEPROM) to save all the data.

Another pre-requisite for the card reader is that it must be sturdy enough to handle frequent use while still being reasonably compact. The system is simple to use and effective, hence its use in stores.

1.4 Scope of Study

The study on the security system design and implementation is to be completed within approximately one year timeframe (two semesters). The scopes for phase 1 are the design of core electronic circuit, software implementation, and also prototype testing.

When the hardware implementation is completed, the testing will be done on at one of the lab doors.

CHAPTER 2

LITERATURE REVIEW

Security Door System using Card Reader basic function is to store and display numbers which are the input on the keypad. The microchip has been programmed to scan the keypad and identify which key has been pressed. Once the data has been programmed into the PIC microchip we can use the card to read the data and enter the door.

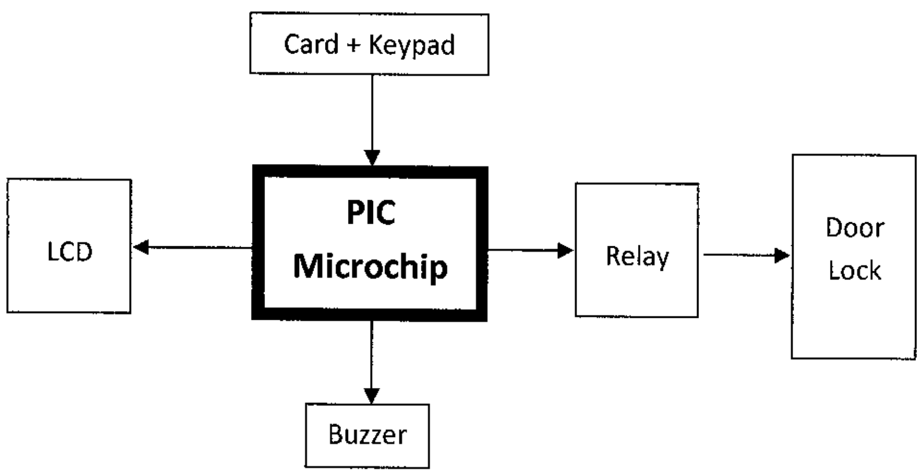

Figure 1: Overview of Security Door System

From the above figure, there are 7 main items used in this project, which are card reader, the keypad, LCD display, PIC Microchip, buzzer, relay and also door lock. Card reader and the keypad works as the input, as for the LCD display, buzzer and the relay are works as the output from the microchip. The system works by receiving input from the card and the PIN key in by using keypad. If both of the inputs are correct, the system will open the door lock which controlled by the relay. The buzzer works when the user failed to key in the correct PIN. The details for each item are discussed as below.

2.1 Microcontroller

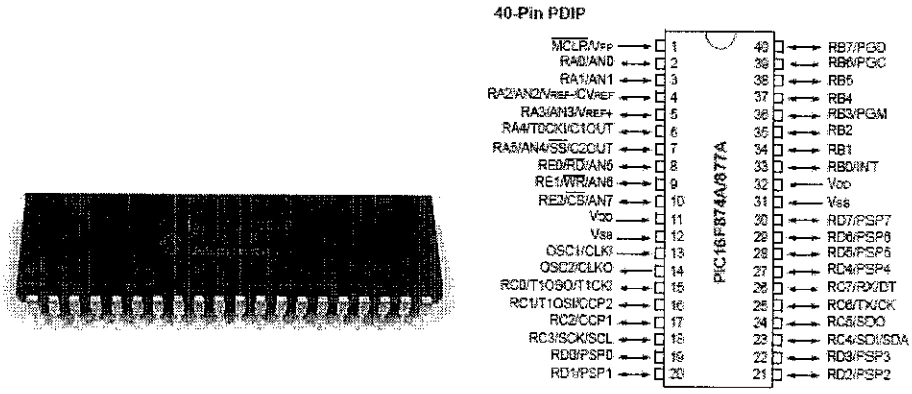

Figure 2: Physical chip and pin diagram of PIC16F877A

PIC Microcontroller is one of the popular microchip among the hobbyist. This powerful yet easy-to-program CMOS Flash-based 8-bit microcontroller packs Microchip's powerful PIC architecture. It has 35 single word instructions which are straightforward for newcomers. The features of the device:

- 5 channels of 10-bit Analog-to-Digital (A/D) converter
- 256 bytes of EEPROM data memory
- ICD (In-Circuit Debugging) function

A. Internal EEPROM

The data EEPROM is readable and writable during normal operation. This memory is indirectly addressed through the Special Function Registers (SFR). There are six SFRs used to read and write this memory:

- EECON1
- EECON2
- EEDATA

- EEDATH
- EEADR
- EEADRH

When interfacing to the data memory block, EEDATA holds the 8-bit data for read/write and EEADR holds the address of the EEPROM location being accessed. This device has 256 bytes of data with an address range from 00h to FFh.

When interfacing the program memory block, the EEDATA and EEDATH registers form a two-byte word that holds the 14-bit data for read/write and EEADR and EEADRH registers form a two-byte word that holds the 13-bit address of the program memory location being accessed.

The EEPROM data memory allows single-byte reads and writes. A byte write in data EEPROM memory automatically erases the location and writes the new data. When the device is code-protected, the CPU may continue to read and write the data EEPROM memory.

Even though the PIC Microcontroller has its own EEPROM data memory, 256 bytes is not enough to store huge amount of data as one of the purpose of the project. In order to store these data, external EEPROM has been used.

2.2 Non-Volatile Memory (EEPROM)

In order to save data without erasing it during turned off; EEPROM is needed to do the task. EEPROM stands for Electrically Erasable Programmable Read-Only Memory. Selective programming can be done to an EEPROM chip. The value of certain cells can be altered without using needing to erase the programming on the other cells. Sections of data can be erased and replaced without affecting the rest of the chip's programming.

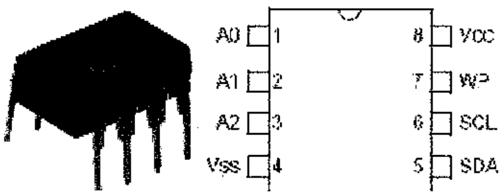

Figure 3: Microchip EEPROM 24AA64 and pin diagram of EEPROM

2.3 Wiegand card reader

Wiegand card reader contain a digital buffering circuit in which the signals generated in the read head by the Wiegand wires are separated into discrete digital '1' and '0' data pulses. Normally, the data is buffered and the output on two separate electrical signal lines with a common signal ground line.

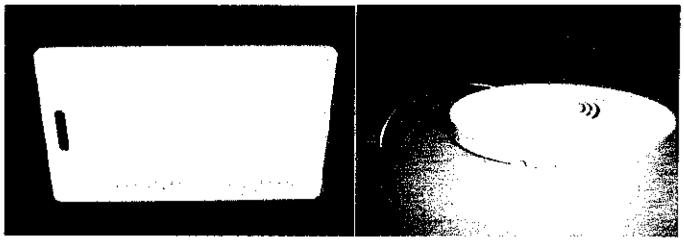

Figure 4: Wiegand card and the card reader

a. Signaling

For an electrical standpoint, every peripheral device within a security system must be electrically compatible with its controller. Many controllers currently in use were designed to accept the Wiegand standard for data signaling. This standard is significantly different than the other major signaling standard, in term of clock and data. If another reader technology wanted to be connected to a Wiegand type system, it needs to be electrically compatible with the controller.

Many controllers in use today support Wiegand signaling. The Wiegand signaling standard utilizes two data lines to carry card data to the controller. They are named data1 and data0. As the names impart, the data1 line carries the "1" bits of the data stream to the controller, and the data0 line carries the "0" bits. The picture below is a graphical representation of a Wiegand data stream for the binary value "01101". Each dip in the line represents a change from 5V to 0V, thus communicating the bit.

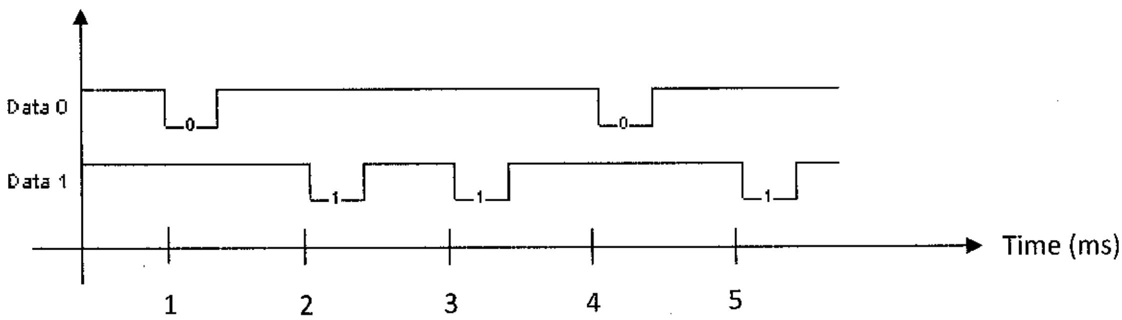

Figure 5: Data stream for binary value of 01101

b. Data Format

Wiegand data format is characterized by the total bit count and the distribution of data fields on a card. The figure below illustrates the use of 26-bit Wiegand, the most commonly used Wiegand data format.

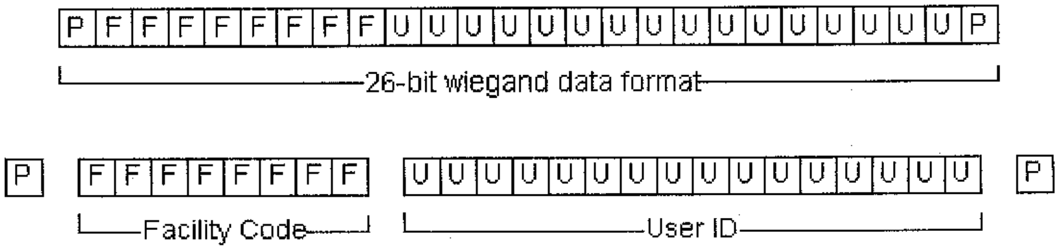

Figure 6: Wiegand data format

This Wiegand format consists of two parity bit, 8-bit facility code, and 16-bit user ID, for a total of 26 (1+8+16+1=26) bits.

2.4 Keypad

In this project, a 4x3 keypad is used to key in the data. The keypad is consists of 7 pins and the layout is as shown in. The board is a series of pushbutton switches that provide structured input for measuring user input. Output pins are 1-7, where pin 1 corresponds to the pin closest to the * key.

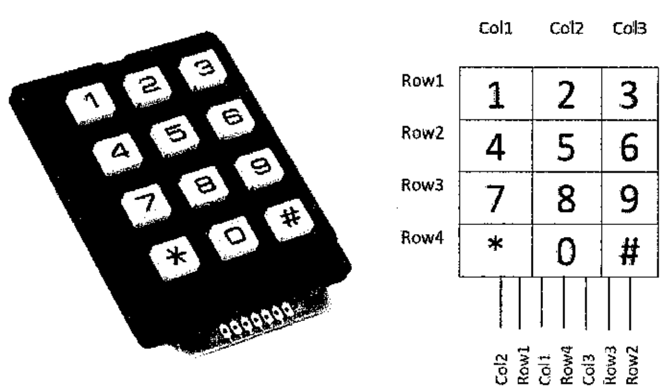

Figure 7: Physical layout of 4x3 keypads and list of output pins respected to each number on the keypad

2.5 LCD Display

As for LCD display, there are several types of it in the market, such as 2x16, 1x20, and 2x20. As for this project, the author chooses 2x16 as the display as it can display 2 lines of information and the length of the display is 16 characters for each line.

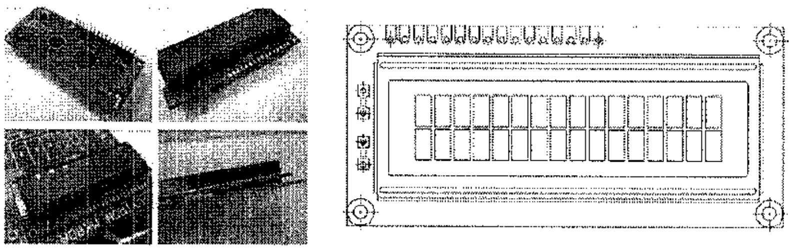

Figure 8: 2x16 LCD Display and schematic layout of LCD Display

The LCD display has 16 pins on it. The function for each pin are describe as below:

PIN NUMBER	SYMBOL	FUNCTION
1	Vss	GND
2	Vdd	+5V
3	Vo	Contrast Adjustment
4	RS	H/L Register Select Signal
5	RW	H/L Read/Write Signal
6	E	H-> L Enable Signal
7	DB0	H/L Data Bus Line
8	DB1	H/L Data Bus Line
9	DB2	H/L Data Bus Line
10	DB3	H/L Data Bus Line
11	DB4	H/L Data Bus Line
12	DB5	H/L Data Bus Line
13	DB6	H/L Data Bus Line
14	DB7	H/L Data Bus Line
15	Vss	-ve Voltage Output
16	Vdd	Power Supply for B/L

Table 1: Datasheet of LCD display

CHAPTER 3

METHODOLOGY

3.1 Project Flow

This project will go through the following flowchart which is the basic steps to ensure the continuity of the project:

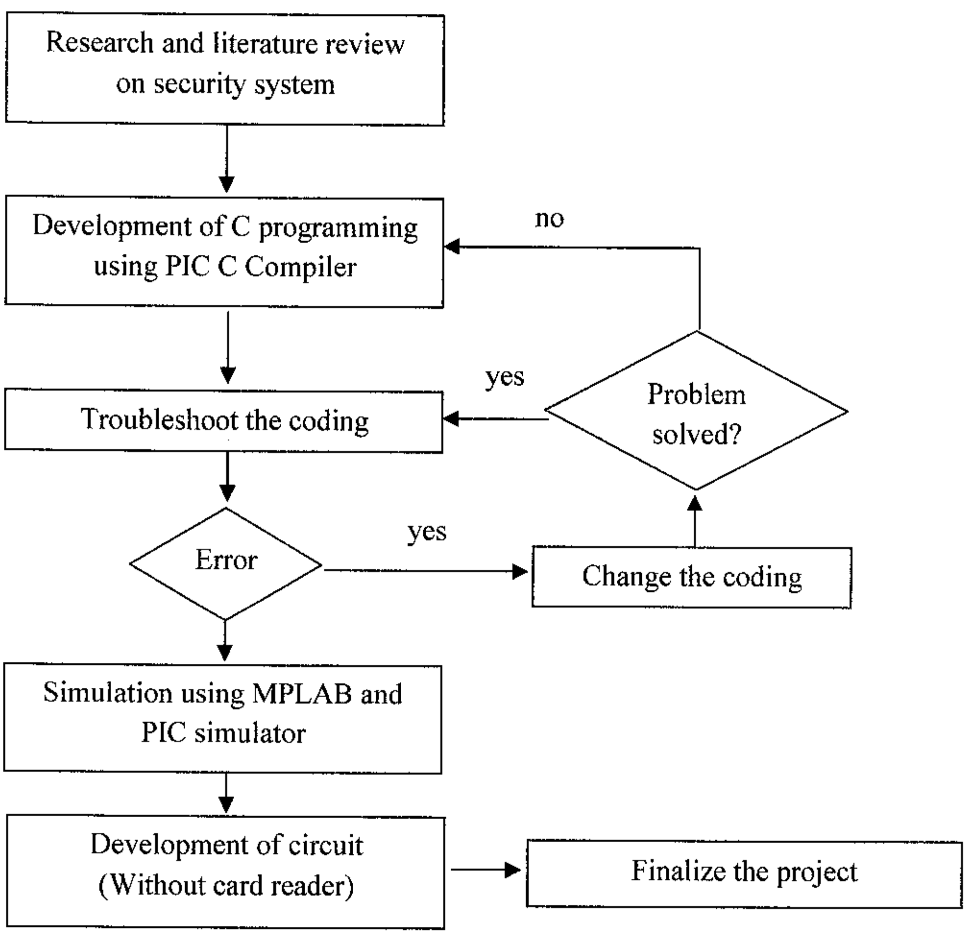

Figure 9: Project's Flowchart

3.2 Mode of operation

I-Security system is a standalone device that allows access to registered users identified by the magnetic cards. The system includes:

- Add user card/reader and add PIN by keypad.
- Delete user card by keypad/reader and delete PIN by keypad
- A 'lock down' of the system when someone tries to gain access by repeatedly swiping incorrect cards.
- Lock release time: 1-99 seconds
- Set the permanent lock release
- Output point – door lock
- Input point – door sensor, exit push button
- A 16x2 LCD display which will display the user feedback on the current state (such as 'Access Allowed')

3.2.1 Enter into Programming Mode

1. Press *9999# within 5 seconds, both **Ready** (green LED) and **accept** LED (yellow LED) will illuminated, Open LED will flashing for a few seconds.

3.2.2 In Programming Mode

1. Press 0 [new master code] # [6 digit] # (master code must be in 4-8 digit numbers)
2. Press * after changing the master code, otherwise unit will disregard the new code and revert back to the factory default code. (123456)

There are 2 ways of adding user cards, which are by reader or keypad. Multiple cards can be added at a time.

A. Adding User Cards by using reader

1. Press 1 01 [user identification read card] #
2. The user identification must be in 3 digit numbers and first card start with 000 up to 999.
3. The system will auto added a user PIN with 1234 for every card.

Here is an example of how the data would be:

No	Name	Card Number	User Identification	User PIN
1	Ahmad	003243	000	1234
2	Allen	034658	001	1234
3	Zaki	010102	002	1234
4	Muthusamy	054992	003	1234

Table 2: Sample of list name with details

B. Add multiple cards at the same time by reader

1. Press 1 01 user identification [read card] [read card] [read card] ... [read card] #
2. Each 3-digit number only corresponds to one card. When adding multiple cards, the 3-digit cards for the second card will add 1 automatically, and the number ranges from 000 to 999.

C. Add User Cards by Keypad

In the condition of not having reader, the card can be added only using keypad. The problem with this system is that the card number needs to be inserted manually into the system.

The user identification must be 3 digit numbers and the first card will start from 000 up to 999. The system will automatically add a user PIN of 1234 for each card.

D. Add multiple cards at the same time by keypad

1. Press 1 02 [user identification key] in 6 digits [card number key] in 6 digits [card number] ... key in 6 digits card number #

2. The process are the same with adding multiple card by reader, only the difference is the card number need to be inserted manually for each card.

E. Delete user card(s)

There are 4 options to delete a user card(s). It is either by deleting all records at one go, reading the card using reader, keying in card number or keying in the user identification number. The process for each step is as below:

1. Press: 2 0000 # to delete all user cards.
2. Press: 2 01 Read card # to delete individual user card.
3. Press: 2 02 key in 6 digits card number # to delete individual user card
4. Press: 2 03 user identification number # to delete individual user card

F. Setting Operation Mode

There are 3 different options for user operation mode, card only, card and PIN and valid code (valid card or PIN). The optioned used is common to all users.

1. Press: 3 00 # valid card only (default setting)
2. Press: 3 01 # valid card and PIN (user need to change the default PIN before using this card and PIN together)
3. Press 3 02 # valid card or PIN

G. Setting Lock Release Time

1. Press 4 [new time] from 00-99 seconds #
2. Default door relay strike time is 5 seconds. In order for this feature to work, door contacts must be connected. There are 2 programming functions that work together in this mode:

    a. if the door is not closed after the door release time, keypad buzzer sounds.

    b. if the door forced open, keypad buzzer sounds.

H. Changing User PINs

The default setting for each user PIN code is 1234; this can be modified so that each user has unique individual 4 digit code.

1. Press: 6 01 read [user card] [user PIN] # [new PIN] #
2. Press: 6 02 key in [6 digits card number] [user PIN] # [new PIN] #

I. Setting Keypad Lockout

There are 2 levels of keypad security available for the system.

1. Press: 7 00 # to disable feature (factory default setting)
2. Press: 7 01# to set keypad locked for 10 minutes when enter 5 times wrong user Pin in succession

3.2.3 Normal Running Mode

The blue LED will flash to indicated the control unit is running

1. Using card to release door, wave card in front of the controller
2. Using PIN to release the door, press: [user PIN] #

3.2.4 Resetting to Factory Default Setting

In order to reset all settings to the factory default values, the Non Volatile Memory (EEPROM) must be reset. Reset Non Volatile Memory by switching off the power and press key 1 & 3. After switching on and remove the key 1 & 3, key in new [Master PIN] and #. The system will give a beep to indicate the controller reset to factory default setting.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Results

4.1.1 Circuit Diagram

Before a prototype can be built, a circuit diagram has to be produced first. The purposes of having this circuit diagram are:

a. to verify the voltage and current needed for the circuit
b. and to check whether there are enough connections in order for the circuit to work properly

A circuit diagram based on the requirement of the project has been developed. The components inside the circuit diagram are as below:

a.	*PIC16F877A x 1*	*j.*	*4 x 3 Keypad x 1*
b.	*Magnetic Door Lock x 1*	*k.*	*LED 3mm Green x 1*
c.	*Crystal 20MHz x 1*	*l.*	*LED 3mm Yellow x 1*
d.	*Voltage Regulater-LM7805*	*m.*	*2 x 16 Character LCD x 1*
e.	*Diode 1N4007 x 1*	*n.*	*Resistor 1/4W 1K x 5*
f.	*Diode 1N4148 x 1*	*o.*	*Resistor 1/4W 10K x 2*
g.	*E-Cap 100uF 25V x 2*	*p.*	*Resistor 1/4W 220R 3*
h.	*C-Cap 104 x 2*	*q.*	*Preset 4K7 x 1*
i.	*C-Cap 30pF x 2*	*r.*	*SRD Relay 12V x 1*

Table 3: Components of the circuit diagram

Circuit diagram is as below:

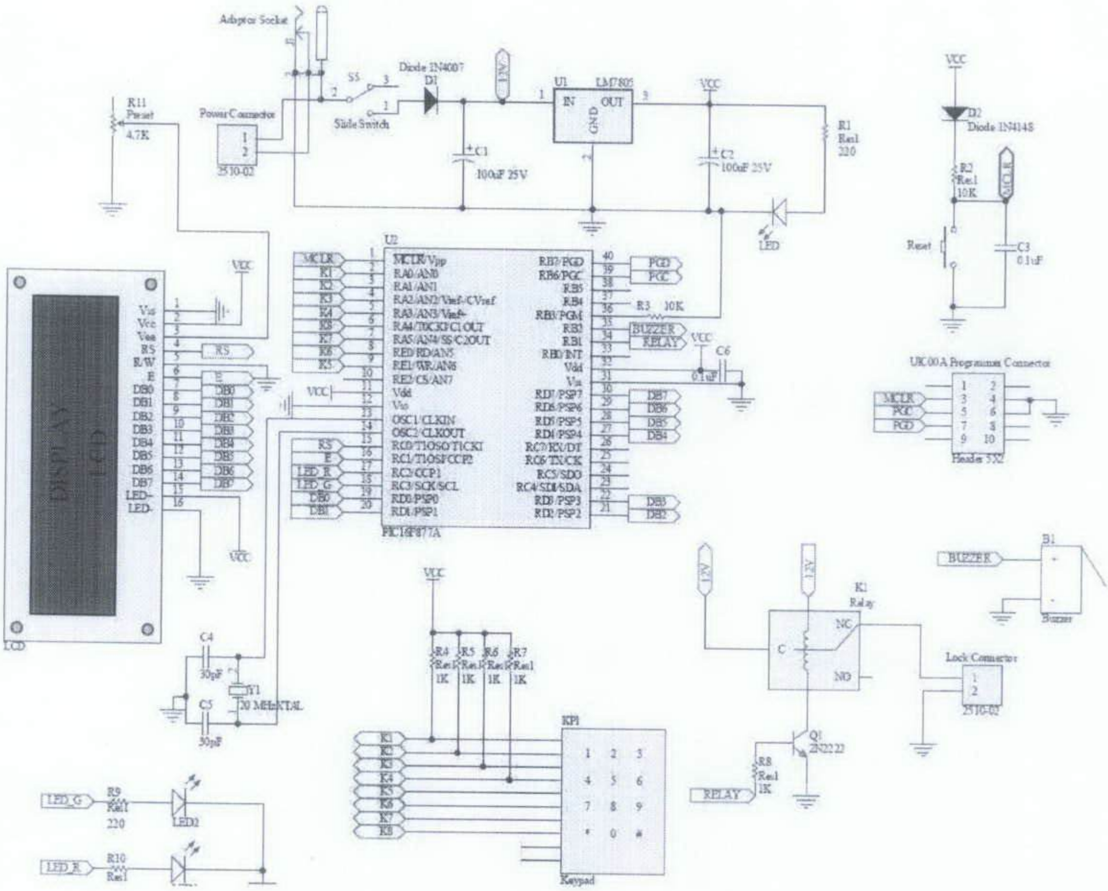

Figure 10: Circuit diagram without Wiegand Reader

4.1.2 Circuit Simulation

After the components to be used in the project have been chosen, here comes the part of the programming. There are two ways of programming, C-Language Programming or Assembly Language. As for this project, C Language has been chosen for its simplification. The coding for the project can be assessed in the *Appendix A*.

The coding is tested using PIC Simulator. The software can be used to simulate many components, such as LCD display, keypad and also the PIC itself. Here is an example of using the simulator to simulate the LCD display.

Figure 11: Simulation of LCD Display using PIC Simulator

The output of the LCD display in the actual hardware is as below:

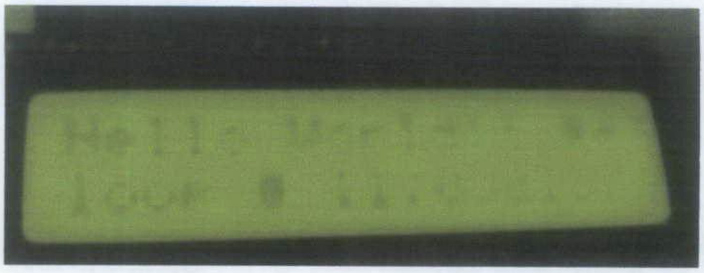

Figure 12: Output on LCD display

4.1.3 Circuit with Keypad and LCD Display

In order to monitor the progress of the project, the prototype has been divided into smaller modules according each function. The modules are the RFID module, LCD display module, keypad module and door-locking module.

All modules have been able to work independently. As part of the final progress of the project, all the modules need to combines together to build a single prototype. The figure below is the prototype after combining all the modules.

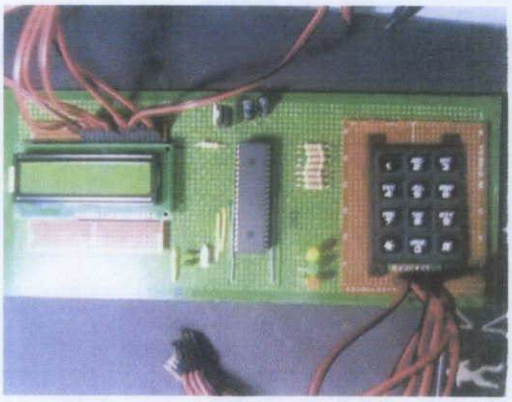

Figure 13: The prototype on a circuit board

a. The circuit has been able to display all the numbers on the keypad onto the LCD. It took few weeks to verify the problem as the connections for the keypad differs from one manufacturer to another.

4.1.2 Cost

As one of the objective of the project, the cost building the prototype is monitored and it is listed as below:

Item	Price per item (RM)	Quantity	Price (RM)
Wiegand RFID Reader	150.00	1	150.00
2x16 LCD Display	30.00	1	30.00
4x3 Keypad	22.00	1	22.00
PIC 16F877A	22.00	1	22.00
Magnetic Switch	6.00	1	6.00
Multi-turn Preset	3.50	1	3.50
Crystal Oscillator (20Mhz)	2.00	1	2.00
Howay IC Socket	0.80	1	0.80
RFID Card	0.00	3	0.00
		TOTAL	**236.30**

Table 4: List of purchased items

As a part of the requirement of the project, the cost of building the prototype need to be as low as possible without sacrificing the function as a security door system. From the table above, the total cost so far is RM 236.30, which is quite economical considering the performance and potential of it. Wiegand RFID Reader acquires almost 70% of the total cost.

4.2 Problems Encountered

A. Inability to Connect with EEPROM

The storage capacity of internal EEPROM inside the Microchip PIC 16F877A is limited to 256 bytes of data only. In order to tremendously improve the capacity, external EEPROM is needed. The connection for the EEPROM has been done but unable to read/write data from it.

4.3 Discussions

It is important to troubleshoot the coding from time to time and there are still a lot of improvements need to be done to tackle the problems faced.

This project is workable and able to adapt into real world application and might be used by University Technology PETRONAS or other companies for security purposes, if the following objectives are achieved:

(a) Able to store data up to 1000 users
(b) Fast and robust system
(c) Effortlessly expandable depending on needs of the industry

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

In Malaysia, security door system research and development has reaches a maturity stages where there are a lot of companies providing the services throughout the country. Most of the systems are being outsourced from other countries such as United States and Japan. HID security door system is one of the most popular in the market nowadays.

The current system centralized the entire database into one server which connects to each card reader. The disadvantage of having centralized system is when there is a power failure, the door cannot be access and it would get worse if emergency happens such as fire. By having this security system, it would act independently and does not depend on a centralized server for the database. All the data are secured inside the module itself.

As for the conclusion, this system can overcome the problem that occurs in the existing system. The tag is hard to falsify by intruders because there is no connection between the panel and server.

5.2 Recommendation

There are few recommendations for future work that can be done to the system, so that it can be more reliable and practical for daily use.

a. Wiegand Data Encryption

A Wiegand Converter for generating a bi-directional data includes a Terminal Data Separator, a Switch Controller, a Bi-directional Switch to control the Input or Output Wiegand Data send through a Bi-directional Wiegand Data Line. The Received Data sent to Converter and Transmitted Data received From Converter are in recognized computer form such as ASCII. The unsecured Wiegand protocols can then be manipulated by a terminal or transmitted it via a secured form or different data communication protocols and reversed it back to an acceptable Wiegand form for controlling the devices to be located at remote sites or in separate compartments.

REFERENCE

[1] Datasheet PIC16F87XA, 28/40/44-Pin Enhanced Flash Microcontroller, 2003 Microchip Technology Inc

[2] Martin Bates, 'PIC Microcontrollers, An Introduction to Microelectronics', 2004 Elsevier Ltd

[3] C. H. Lim, Y. H. Dan, K. T. Lau and K. Y. Choo, 'Smart Card Reader', School of Electrical and Electronic Engineering, Nanyang Technological University Singapore

[4] B Moreau, 'Smart Card: Concept, Market and Trends', Proceedings Smart Card Technology and Applications '89 Seminar, Singapore.

[5] C Y Lim and Y H Dan, 'PCbased Smart Card Reader', NTU FYP Report, Feb 1992, Singapore.

[6] International Standard, IS0 7816, Parts 1 - 3, 1987. IEEE Transactions on Consumer Electronics, Vol. 39, No. 1, FEBRUARY 1993 Biographies

[7] Allen L Wyatt, 'Using Assembly Language', Micro-Tech Publications, 1989.

[8] Wiegand Card - http://www.mercury-security.com/technology/whenyou.htm

[9] Cheng, Lee Ming (Hong Kong, CN), Chan, Chi Kwong (Hong Kong, CN), Hsie, Wei Cheng (Northborough, MA, US), 'Wiegand Converter and method of generating a bi-directional data', http://www.freepatentsonline.com/7293698.html

APPENDIX A

Main.c

```
// Author       : Wan Hesman
// Project      : Security Door System using card reader
// Description : PIC16F877A + 4x3 keypad + LCD + RFID Card Reader
// Security system that will activate the realy and buzzer after a preset
6-digit password
// is entered. LCD will display ****** when the keypad is pressed.
// The preset password is 123456

// RFID Wiegand Reader

// PIN B0,C4 - D0, green
// PIN B1,C5 - D1, white
// PIN B2,C7 - Buzz, orange

//==========================================================================
========================

#include <16F877a.h>
#fuses HS,NOWDT,NOPROTECT,NOLVP
#use delay(clock=20000000)                    // using 20Mhz clock
#use I2C(Master,sda=PIN_C0,scl=PIN_C1)        // setting for EEPROM


//==========================================================================
========================

//global variable
int8 seconds;
int8 int_count;

#define INTS_PER_SECOND 62          //(10000000/(4*16*250*10))


//==========================================================================
========================

#include <lcd.c>
#include <keypad.c>
//#include <string.h>
//#include <prog_mode.c>
//#include <display_i.c>
```

```
//===========================================================================
=======================

//#byte PORTA=5 //define PORT A address
//#byte PORTB=6 //define PORT B address
//#byte PORTC=7 //define PORT C address

//===========================================================================
=======================

#INT_TIMER2

void clock_isr()
{
   if(--int_count==0) {
      ++seconds;
      int_count = INTS_PER_SECOND;
   }
}

//===========================================================================
=======================

int buffer[3];
int i;

void main()
{
   char input[10];
   char k;                  // variable name test
   char  char_num=0;                  //  initialize  to  indicate  character
location
   char pass;

   lcd_init();                 // lcd initialization
   kbd_init();                 // keypad initialization

   set_tris_a(0b00000000);
   set_tris_b(0b00000000);
   set_tris_c(0b00000000);

   lcd_putc("\f:::welcome:::\n");
   delay_ms(1000);
   lcd_getc(1,1);
   lcd_putc("Press * to Tag");

   output_low(PIN_C7);      //On buzzer and disable reader
   delay_ms(1000);
   output_high(PIN_C7);     //Off buzzer and enable reader

   while (TRUE)
   {  pulang:
      k=get_key();
      if(k!=0)
```

```
        if (k=='*')      //enter tag mode
        {
        lcd_getc(1,1);
        lcd_putc("\fTag it\n");

        do{
           buffer[0]=0;
           buffer[1]=0;
           buffer[2]=0;
           for(i=0; i<26;i++){  //Wiegand Format (26bit binary)
              while(input(PIN_C4)==1 && input(PIN_C5)==1);
              if(input(PIN_C4)==0 ){
                 if(i!=0 && i != 25){         //Not start bit and end
bit

                    shift_left(buffer,3,0);
                 }
              }
              if(input(PIN_C5)==0){
                 if(i!=0 && i != 25){
                    shift_left(buffer,3,1);
                 }
              }
           delay_us(500);
           }
           printf(lcd_putc,"%X%X%X\n",buffer[2],buffer[1],buffer[0]);
        delay_ms(20000);
        goto pulang;
        }
   while(1);
   }
}
}
```

APPENDIX B

Lcd.c

```
// As defined in the following structure the pin connection is as
follows:
//      D0  enable
//      D1  rs
//      D2  rw
//      D4  D4
//      D5  D5
//      D6  D6
//      D7  D7
//
//   LCD pins D0-D3 are not used and PIC D3 is not used.

// Un-comment the following define to use port B
// #define use_portb_lcd TRUE


struct lcd_pin_map {                 // This structure is overlayed
           BOOLEAN enable;           // on to an I/O port to gain
           BOOLEAN rs;               // access to the LCD pins.
           BOOLEAN rw;               // The bits are allocated from
           BOOLEAN unused;           // low order up.  ENABLE will
           int     data : 4;         // be pin B0.
        } lcd;


#if defined(__PCH__)
#if defined use_portb_lcd
   #byte lcd = 0xF81                   // This puts the entire structure
#else
   #byte lcd = 0xF83                   // This puts the entire structure
#endif
#else
#if defined use_portb_lcd
   #byte lcd = 6                  // on to port B (at address 6)
#else
   #byte lcd = 8                 // on to port D (at address 8)
#endif
#endif

#if defined use_portb_lcd
   #define set_tris_lcd(x) set_tris_b(x)
#else
   #define set_tris_lcd(x) set_tris_d(x)
```

```
#endif


#define lcd_type 2            // 0=5x7, 1=5x10, 2=2 lines
#define lcd_line_two 0x40     // LCD RAM address for the second line


BYTE const LCD_INIT_STRING[4] = {0x20 | (lcd_type << 2), 0xc, 1, 6};
                              // These bytes need to be sent to the LCD
                              // to start it up.


                              // The following are used for setting
                              // the I/O port direction register.

struct lcd_pin_map const LCD_WRITE = {0,0,0,0,0}; // For write mode all
pins are out
struct lcd_pin_map const LCD_READ = {0,0,0,0,15}; // For read mode data
pins are in



BYTE lcd_read_byte() {
      BYTE low,high;
      set_tris_lcd(LCD_READ);
      lcd.rw = 1;
      delay_cycles(1);
      lcd.enable = 1;
      delay_cycles(1);
      high = lcd.data;
      lcd.enable = 0;
      delay_cycles(1);
      lcd.enable = 1;
      delay_us(1);
      low = lcd.data;
      lcd.enable = 0;
      set_tris_lcd(LCD_WRITE);
      return( (high<<4) | low);
}


void lcd_send_nibble( BYTE n ) {
      lcd.data = n;
      delay_cycles(1);
      lcd.enable = 1;
      delay_us(2);
      lcd.enable = 0;
}


void lcd_send_byte( BYTE address, BYTE n ) {

      lcd.rs = 0;
      while ( bit_test(lcd_read_byte(),7) ) ;
      lcd.rs = address;
```

```
      delay_cycles(1);
      lcd.rw = 0;
      delay_cycles(1);
      lcd.enable = 0;
      lcd_send_nibble(n >> 4);
      lcd_send_nibble(n & 0xf);
}


void lcd_init() {
    BYTE i;
    set_tris_lcd(LCD_WRITE);
    lcd.rs = 0;
    lcd.rw = 0;
    lcd.enable = 0;
    delay_ms(15);
    for(i=1;i<=3;++i) {
       lcd_send_nibble(3);
       delay_ms(5);
    }
    lcd_send_nibble(2);
    for(i=0;i<=3;++i)
       lcd_send_byte(0,LCD_INIT_STRING[i]);
}


void lcd_gotoxy( BYTE x, BYTE y) {
   BYTE address;

   if(y!=1)
     address=lcd_line_two;
   else
     address=0;
   address+=x-1;
   lcd_send_byte(0,0x80|address);
}

void lcd_putc( char c) {
   switch (c) {
     case '\f'   : lcd_send_byte(0,1);
                   delay_ms(2);
                                           break;
     case '\n'   : lcd_gotoxy(1,2);        break;
     case '\b'   : lcd_send_byte(0,0x10);  break;
     default     : lcd_send_byte(1,c);     break;
   }
}

char lcd_getc( BYTE x, BYTE y) {
   char value;

    lcd_gotoxy(x,y);
    while ( bit_test(lcd_read_byte(),7) ); // wait until busy flag is
low
    lcd.rs=1;
```

APPENDIX C

Keypad.c

```
// Keypad is to be connected to Port B
// Port B pull up resistors to be used and set in the main program
// Connections of PortB to 3x4 keypad
// PortB Connection    Keypad
// Pin_B0              x1
// Pin_B1              x2
// Pin_B2              x3
// Pin_B3              Not Used
// Pin_A0              Y1
// Pin_A1              Y2
// Pin_A2              Y3
// Pin_A3              Y4

char get_key(void)
{
      // char t;
      // while (1);

            output_b(input_b()| 0xFF);
            output_bit(PIN_A0,0);

            if (input(PIN_B0) ==0)
            {
                  while (input(PIN_B0) ==0);
                  delay_ms(100);
                  return '1';
            }
            else if (input(PIN_B1) ==0)
            {
                  while (input(PIN_B1) ==0);
                  delay_ms (100);
                  return '2';

            else if (input(PIN_B2) ==0)
            {
                  while (input(PIN_B21) ==0);
                  delay_ms (100);
                  return '3';
            }

            output_b(input_b()| 0xFF);
            output_bit(PIN_A1,0);

            if (input(PIN_B0) ==0)
            {
                  while (input(PIN_B0) ==0);
                  delay_ms(100);
                  return '4';
```

```
}
else if (input(PIN_B1) ==0)
{
      while (input(PIN_B1) ==0);
      delay_ms (100);
      return '5';

else if (input(PIN_B2) ==0)
{
      while (input(PIN_B21) ==0);
      delay_ms (100);
      return '6';
}

output_b(input_b()| 0xFF);
output_bit(PIN_A2,0);

if (input(PIN_B0) ==0)
{
      while (input(PIN_B0) ==0);
      delay_ms(100);
      return '7';
}
else if (input(PIN_B1) ==0)
{
      while (input(PIN_B1) ==0);
      delay_ms (100);
      return '8';

else if (input(PIN_B2) ==0)
{
      while (input(PIN_B21) ==0);
      delay_ms (100);
      return '9';
}

output_b(input_b()| 0xFF);
output_bit(PIN_A3,0);

if (input(PIN_B0) ==0)
{
      while (input(PIN_B0) ==0);
      delay_ms(100);
      return '*';
}
else if (input(PIN_B1) ==0)
{
      while (input(PIN_B1) ==0);
      delay_ms (100);
      return '0';

else if (input(PIN_B2) ==0)
{
      while (input(PIN_B21) ==0);
      delay_ms (100);
```

APPENDIX D

Rfid.c

```
///////////////////////////////////////////////////////
//   RFID Wiegand Reader                             //
//   revised by: Wan Hesman b. Wan Rusli              //
///////////////////////////////////////////////////////
//    PIN B0 - D0                                    //
//    PIN B1 - D1                                    //
//    PIN B2 - Buzz                                  //
///////////////////////////////////////////////////////

#include <16f877A.h>
#use delay(clock=20000000)
#use rs232(baud=9600,xmit=PIN_C6,rcv=PIN_C7,PARITY=N)
#fuses hs,noprotect,nowdt,nolvp

#byte PORTA=5 //define PORT A address
#byte PORTB=6 //define PORT B address
#byte PORTC=7 //define PORT C address

int buffer[3];
int i;

main()
{
   set_tris_a(0b00000000);
   set_tris_b(0b00000000);
   set_tris_c(0b00000000);
//   set_tris_d(0b00000000);

   output_low(PIN_B2);       //On buzzer and disable reader
   delay_ms(1000);
   output_high(PIN_B2);      //Off buzzer and enable reader

   do{
      buffer[0]=0;
      buffer[1]=0;
      buffer[2]=0;
      for(i=0; i<26;i++){   //Wiegand Format (26bit binary)
         while(input(PIN_B0)==1 && input(PIN_B1)==1);
         if(input(PIN_B0)==0 ){
            if(i!=0 && i != 25){           //Not start bit and end bit
               shift_left(buffer,3,0);
            }
         }
         if(input(PIN_B1)==0){
            if(i!=0 && i != 25){
               shift_left(buffer,3,1);
            }
         }
```