Laboratory Electronic Parts Management System

By

Chao Cao

Chengcheng Huang

Final Report

Prof. Paul Scott Carney

TA: Justine Fortier

ECE445 Senior Design, Spring 2013 Project #31

April 27, 2013

Abstract

A lab manager usually needs to spend a lot of time on sorting and counting electronic components, optical equipments or expensive chips. A conventional method is to check the amount of usage everyday during the maintenance and record the change of amount on the notebook or Excel. So this project is to design a laboratory management system to help the lab manager automatically monitor the usage of these laboratory parts.

The design actualizes the following functions:

1. Eight auto induction Drawers, each has a unique ID and can be detected by the system when it is open.
2. User can enter the commands by using a numeric keypad.
3. An online MySQL Database that is able to background process all changes user make.

The entire system worked fine and it was tested based on the Design Review of this project testing procedure. The only problem this group encountered is during the check out item process, the least significant bit of the check out amount is lost. This issue is fixed by modifying the original operation C source code of the PIC.

Contents

1. Introduction

1.1 Statement of Purpose

The project goal is to design a management system can be used by teaching assistants or laboratory managers to collect how many electronic parts such as chips, I/O boxes, Arduino panels left in the drawers. This is a both hardware and software based laboratory management system that offers a set of key features that support a modern electrical laboratory operation. Most of current methods to record and modify the inventory in the laboratory are finished manually by lab managers. However, this project can finish these jobs automatically. Costumers only need to open a drawer and enter the number of items changed. This project will save lab manager a lot of time in counting the numbers of items in the laboratory.

Figure.1.1 Drawers in Senior Design Lab

1.2 Objectives

1.2.1 Goals:

The project goal is to develop an electric parts drawer system to help the TAs or lab faculty manage the parts easier and faster. The final product should be a drawer system with a display LCD screen and numeric keypad with input buttons. Each time when student/faculty borrow or return part from the a drawer, the system will automatic detect which drawer is opened and the display will show a message to tell which electronic part has been chosen and ask to enter the number to borrow or return. And it will calculate the parts left in each drawer and transmit the data to the inventory of lab managers.

1.2.2 Functions:

• System starts to work when the drawer is open

• Users can enter the amount taken or return through numeric keypad

• Brief instruction of operations will be showed on LCD screen

• The changed amount will be reflected on the database inventory

1.2.3 Benefits:

- New method to manage laboratory parts
- Save time in counting every item in laboratory
- Enables lab manager know which item is out of stock
- Recognize automatically if the drawer is open

1.2.4 Features:

- MySQL database inventory
- User friendly Interface with LCD display
- Easy operation with numeric keypad
- Wireless data transfer

Figure 1.2 a sample keypad

2 Design

2.1 PCB overview

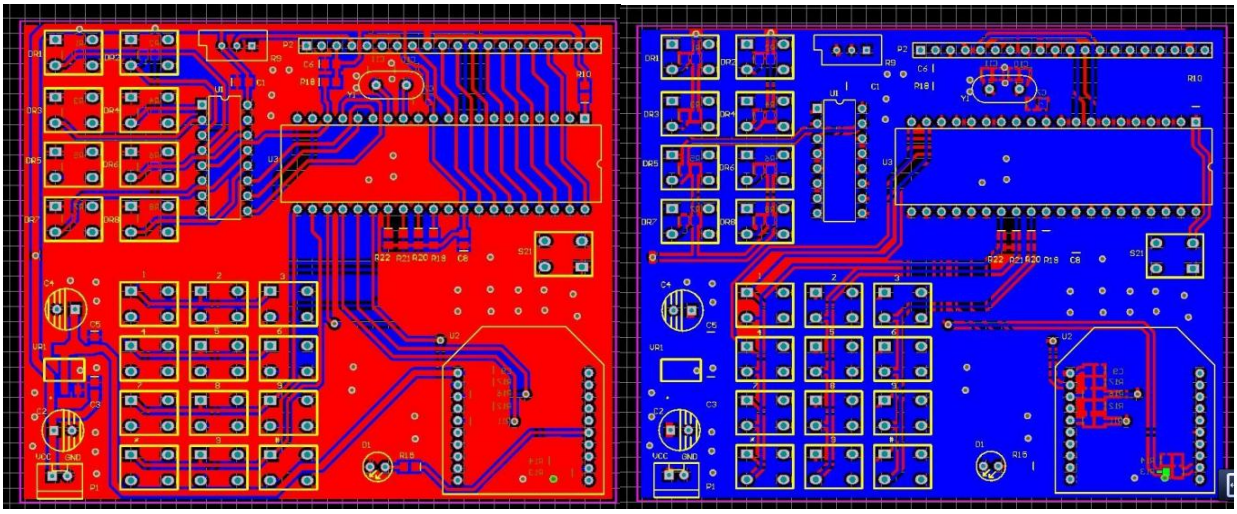


Figure 2.1 Two Layers for PCB

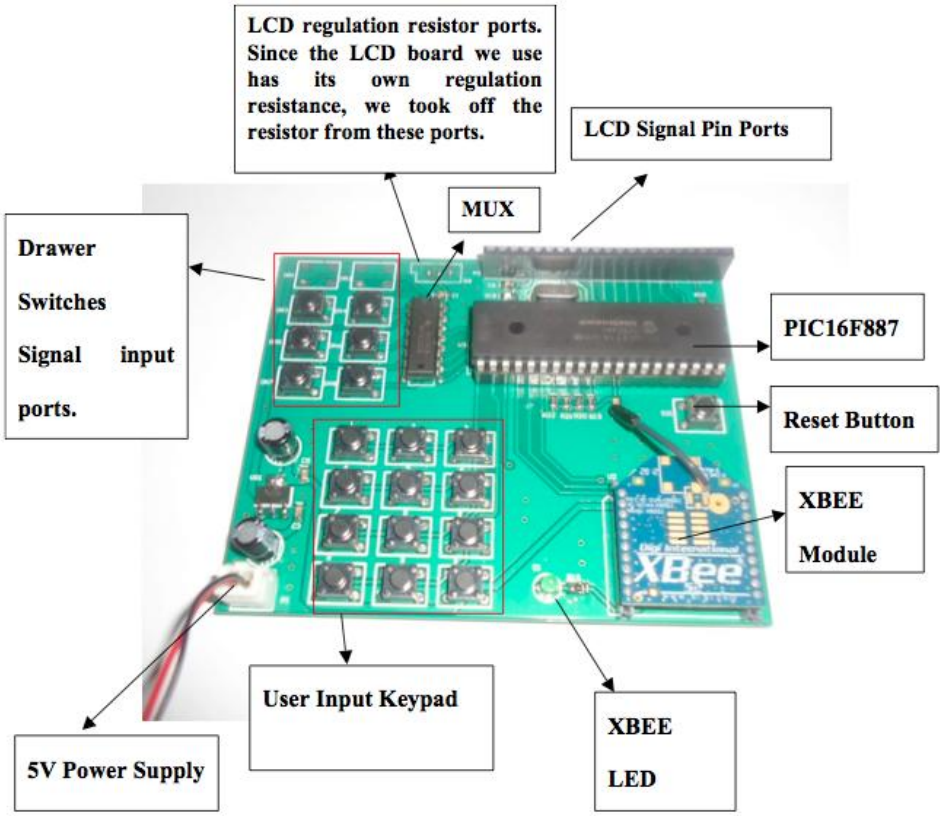


Figure 2.2 PCB for the project

2.2 Hardware Design Overview

2.2.1 Block Diagrams

2.2.1.1 Main (Top Level) Block Diagram

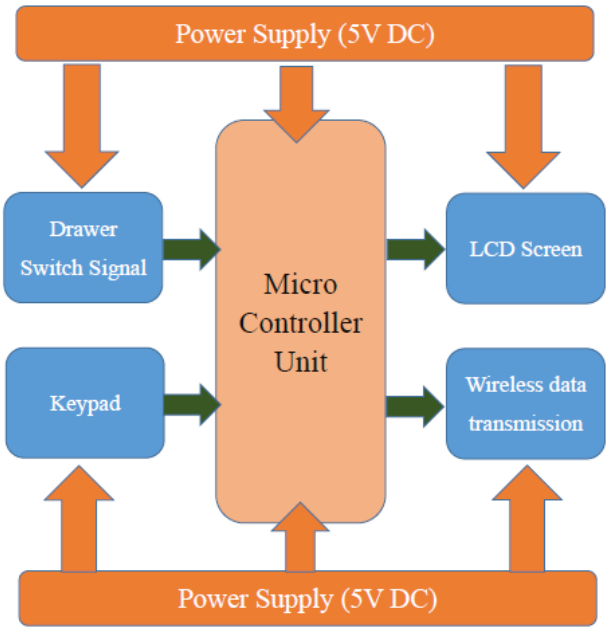


Figure 2.3 Main Block Diagram

2.2.1.2 Drawer Switch Design Diagram

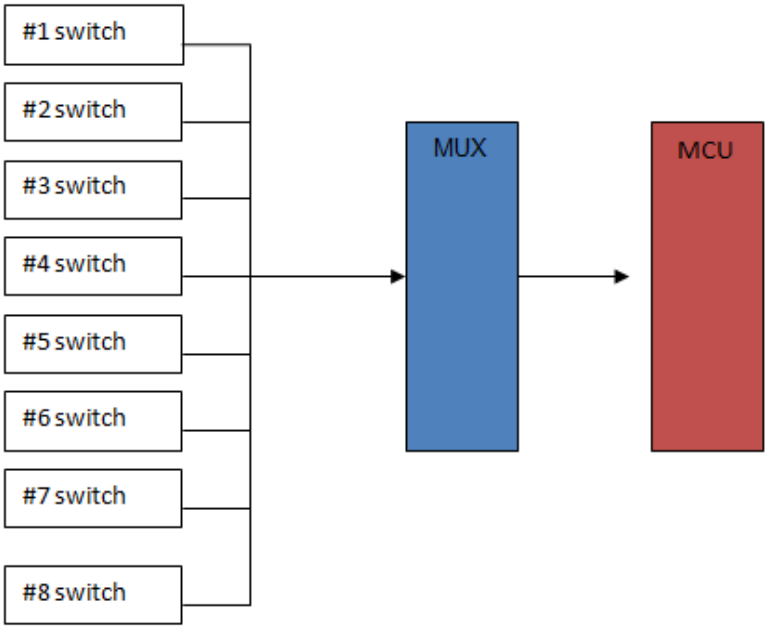


Figure 2.4 Drawer Switch Design Diagram

2.2.1.3 Keypad Design Diagram

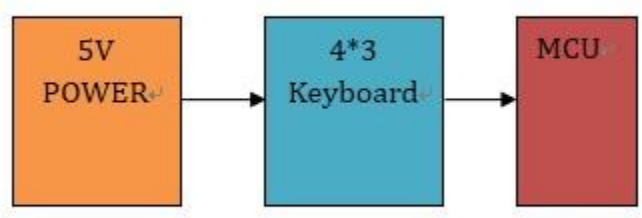


Figure 2.5 Keypad Design Diagram

2.2.1.4 Micro Control Unit (MCU) Design Diagram

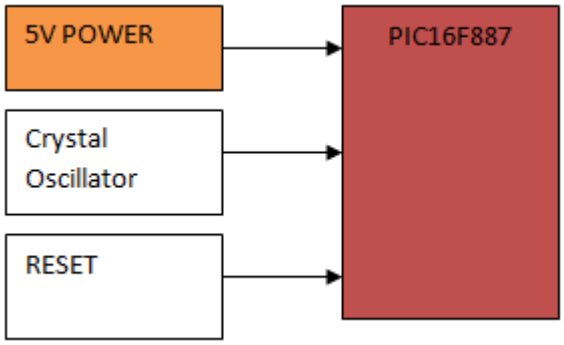


Figure 2.6 MCU Design Diagram

2.2.1.5 Power Supply Design Diagram

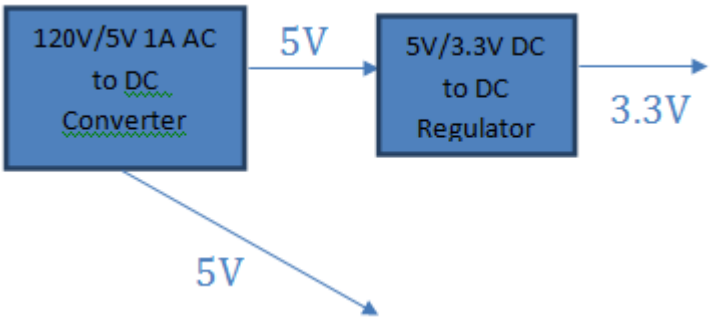


Figure 2.7 Power Supply Design Diagram

2.2.2 Block Descriptions

2.2.2.1 Drawer Switch and MUX Design

The structural design in this project is to make eight drawers be recognized when one of them is opened by users. For the drawers, this project assembles two 12.6 x 10.3 x 3.5 inches 4 drawers packs bought from Amazon.com to make an eight drawers pack. Two iron wires in parallel are set on each ceiling of the drawer. There is an area with insulated tape, which is used to block the contact with the copper strip. Therefore, when the drawer is closed, the copper strip is insulated from the iron wires. The loop is open. Once the drawer is open, the copper strip moves out of the tape area, makes the contact with iron wires and makes a closed loop.

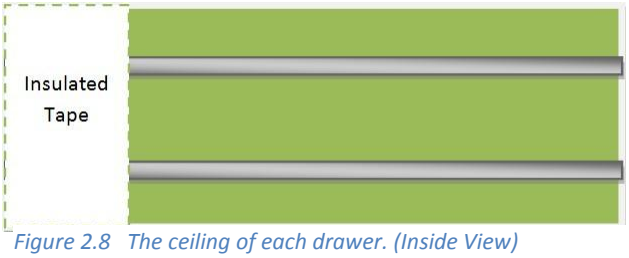


Figure 2.8 The ceiling of each drawer. (Inside View)

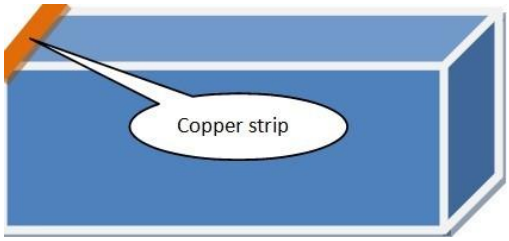


Figure 2.9 The copper strip on the drawer

Those two iron wires will be connected to the two ports from PCB, one of the ports is a constant 5V voltage signal and the other one is the signal recognizer. When the user open a drawer, the close loop will send the high voltage signal to the signal recognizer port, and the PIC will notice this drawer is open by operate with a MUX.

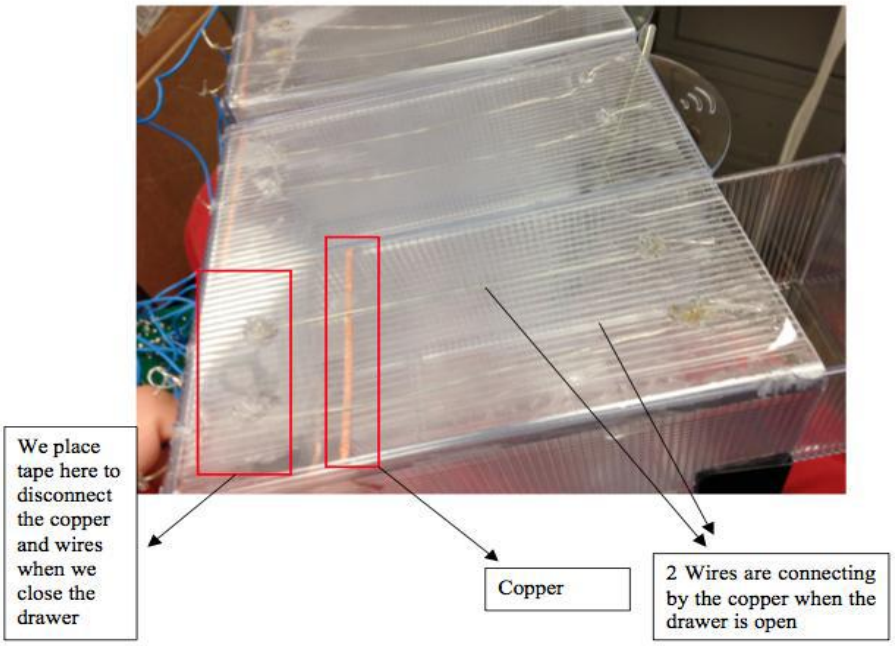


Figure 2.10 The actual drawers

On the PCB, a 8 to 1 MUX (74LS151) is using to tell the PIC which drawer is open, the basic function is working as following: each of the drawer signal recognizer port is connecting to the input signal pin of the MUX, at same time, the PIC will send out the 3-bit selecting signal (A0, A1, A2) incrementally (000,001…111) on a high frequency to the MUX. And it will also receive an output signal from MUX, which working as drawer status detector. When the output of the MUX is high, the PIC will read the selecting signal it is sending out and notice which drawer is opened. The PIC will also notice there are more than one drawer is opened if it receive high voltage signal send from MUX at two or more selecting signal status.

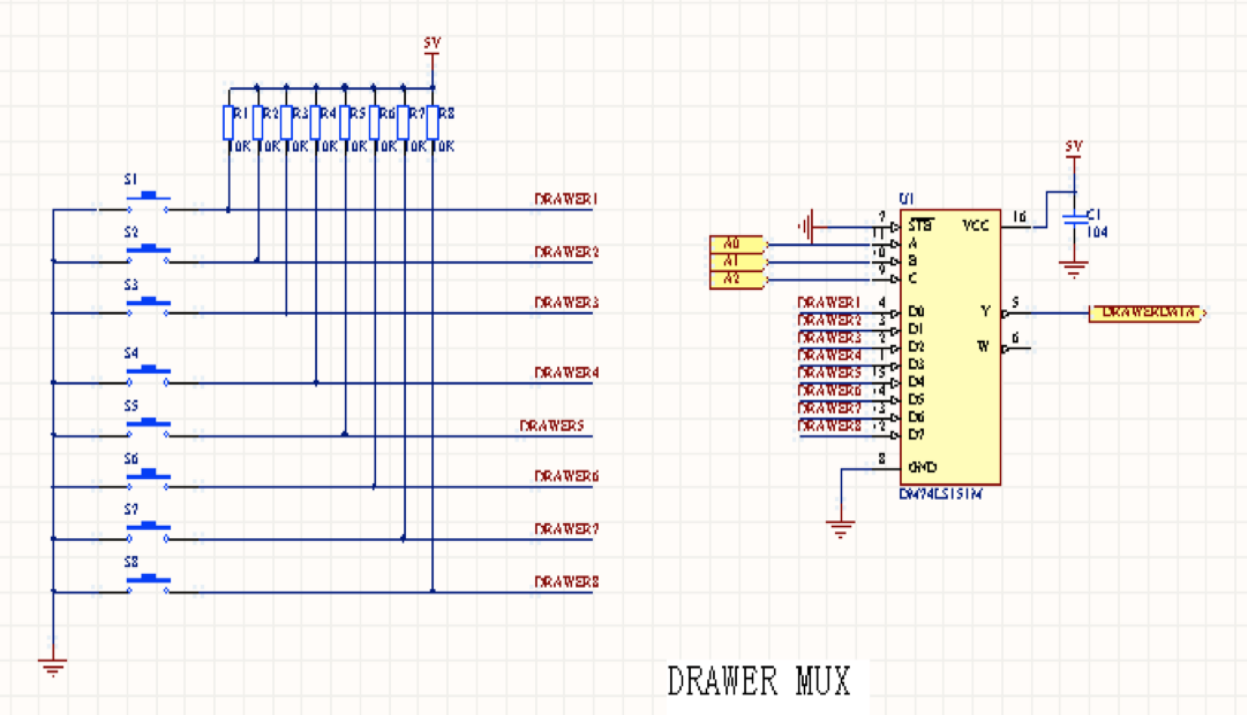


Figure 2.11 Schematic for drawer MUX

2.2.2.2 User Input Keypad

The keypad module is the second input module of the system. It is a 4*3 keypad, which contains 12 buttons working as same function as I/O switches. 10 of them are the numerical buttons (0 to 9) to enter the electrical parts to check in/out, and other two buttons are “Check In” and “Check Out” buttons. It has 3+4=7 pins connect to the PIC showing as the schematic below. By the scanning program on the PIC (similar as the scanning function of drawer-swatch part), the system will receive the signal from the button and do the next step operation.

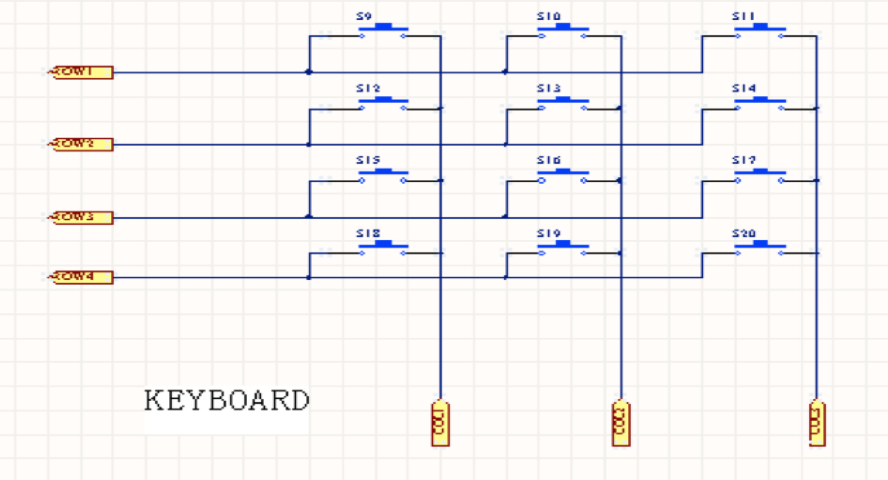


Figure 2.12 Schematic for Keypad

Here is the main part of programing code for get value from keypad

```
/*******************************************************************************
* Function Name : getkey()
* Description : get the value from keypad
* Input : i
* Output : None
* Return : None
*******************************************************************************/
uchar getkey(void)
{ uchar getch;
getch=keyscan();
switch(getch)
{case 0x28: return '0'; break;
case 0x11: return '1'; break;
case 0x21: return '2'; break;
case 0x41: return '3'; break;
case 0x12: return '4'; break;
case 0x22: return '5'; break;
case 0x42: return '6'; break;
case 0x14: return '7'; break;
case 0x24: return '8'; break;
case 0x44: return '9'; break;
case 0x18: return '*'; break;
case 0x48: return '#'; break;
default: return 0xff;
break;
}

}
```

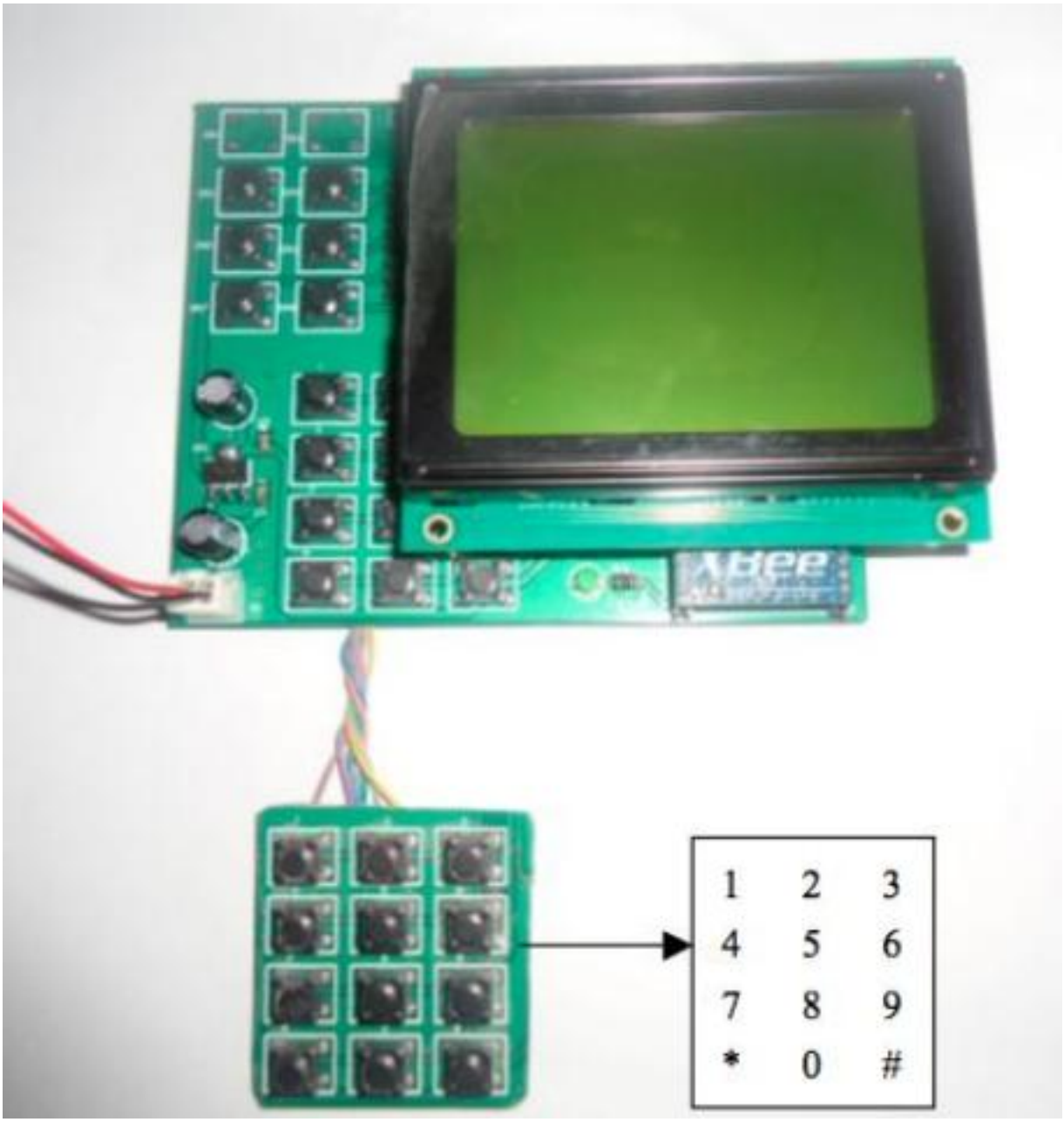


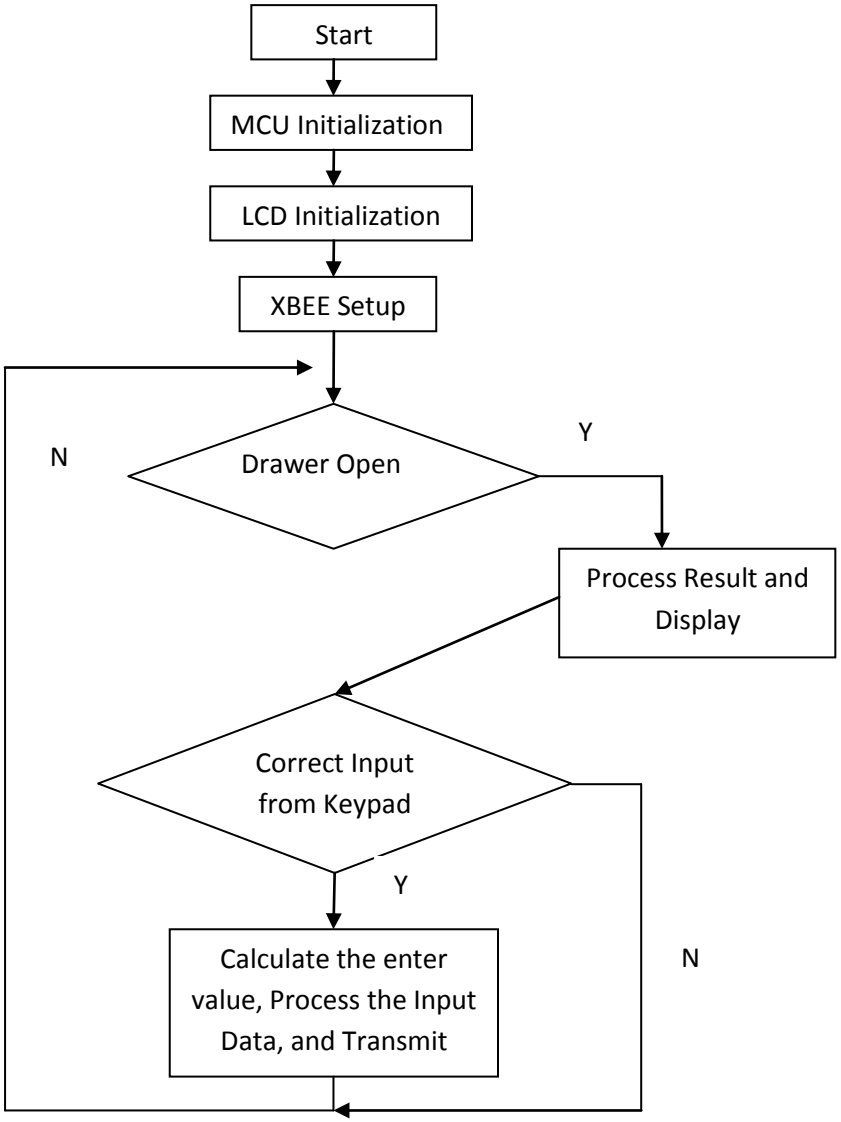
Figure 2.13 Schematic for Keypad

2.2.2.3 Micro Controller Unit (MCU)

For this project, the microcontroller, PIC16F887 will be used for the main control unit. It will connect with crystal oscillator and reset circuit (these are the basic component of MCU) showing in the fig.2.5 below. This module is the core of the system. All the input signals will be process in the MCU. And after do the operations by the program in the MCU, the MCU will sent out the signal trough the UART to LCD to display the message and sent the data out through the XBEE wireless transmission to the computer. The idea of the MCU working function this like this: after

connect the MCU with power supply (5V VCC), it will first initialize the UART and do the initialize set up with LCD and XBEE. After initialize, MCU will enter to the main program cycle; it will automatically and constantly scan the signal from the drawer switches and keypad. When the MCU detect there is drawer open, it will sent signal to LCD to show the message and wait to receive the signal from keypad. When the MCU receive the correct input signal from the keypad buttons, it will send the data through the XBEE to the computer.

The Flowchart of the MCU working process:



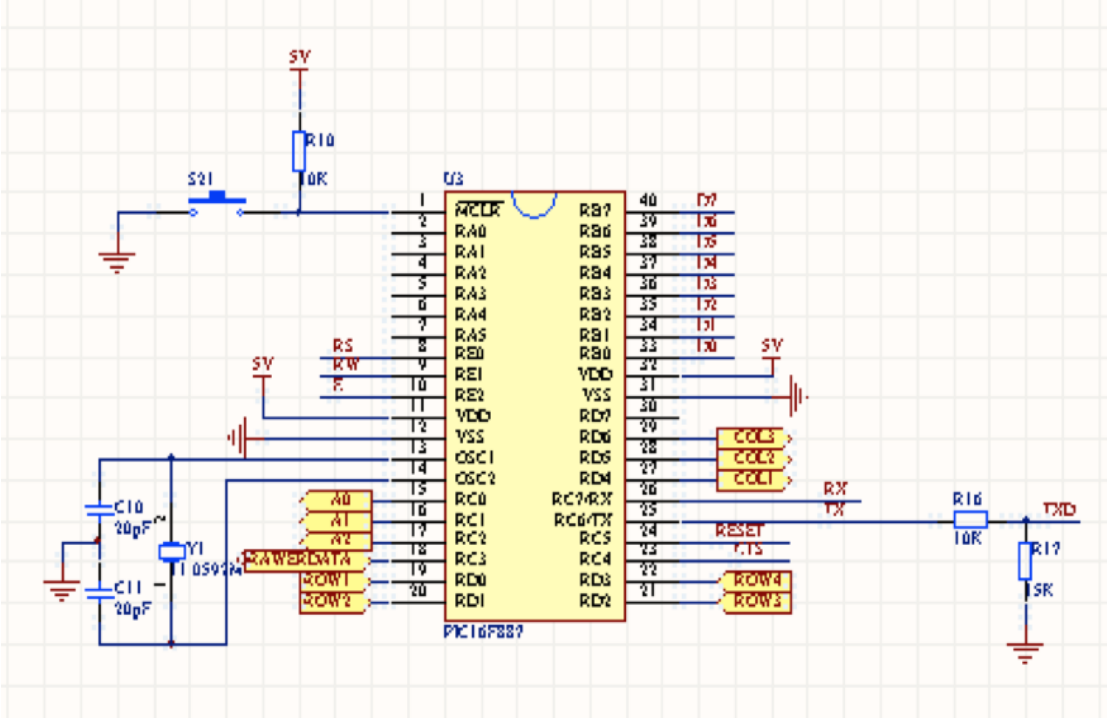


Figure 2.14 Schematic for PIC

2.2.2.4 Liquid Crystal Display (LCD)

For this project, a 128*64 LCD monitor (LCM12864, see Fig.2.4) will be used to display the message on the user interface. It connects with the MCU with 8 data bus lines. The LCM has its own control unit and character library, so the MCU can drive LCD by communicate with the LCD control unit directly. There will be 10 different messages showing on this LCD in different status: if all the drawers are closed, the message display on LCD will be "No drawer open, please open one to get start."; if one of the drawers is opened, the message display will be "Drawer X has been selected, please enter the number you wish to check in or check out." (In this message, X is the number of the open drawer, since there will be 8 drawer-switches in this project, so the LCD have 8 drawer open message); if more than one drawers open at the same time, the message display will be "Please open one drawer each time, thanks."

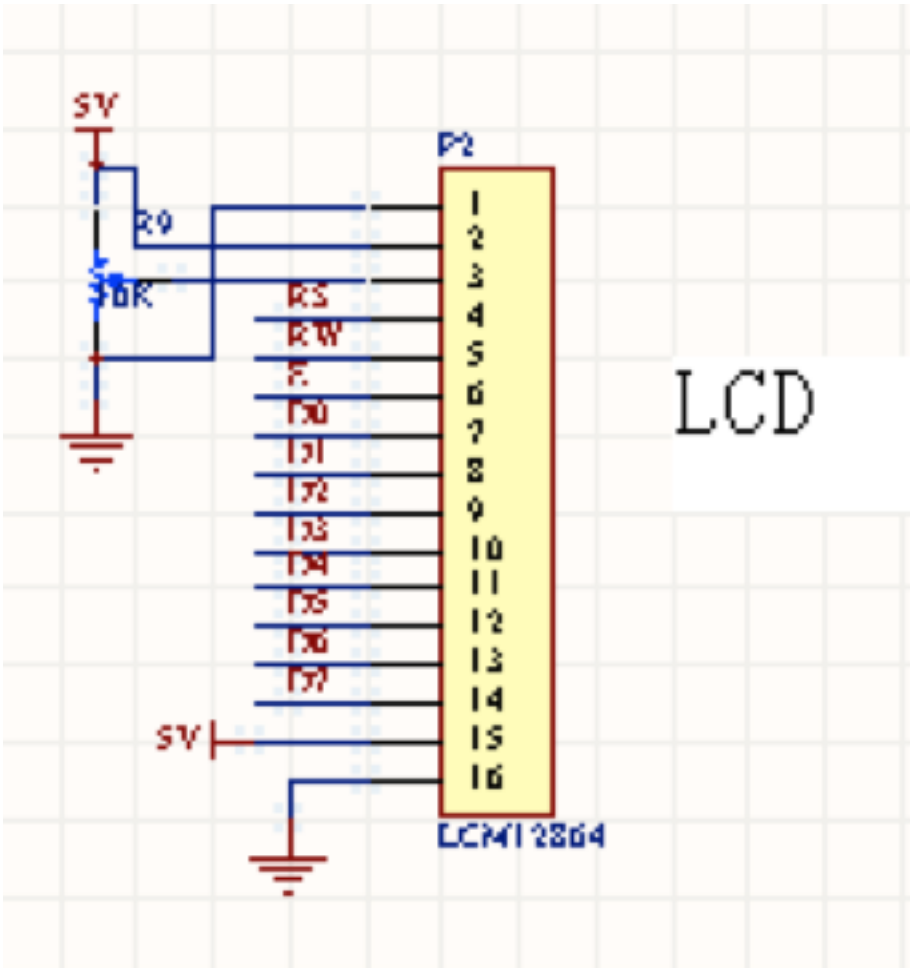


Figure 2.15 Schematic for LCD

Here is the programing code to setup the display message on LCD

```
const uchar NONED[]="No drawer open,please open one to start!";
const uchar ONED[]="DRAWER #3 open, enter the check in/out
number.(*:in,#:out):___";
const uchar CLOSEOUT[]=" item has been check out,please close the drawer.";
const uchar CLOSEIN[]=" item has been check in,please close the drawer.";
const uchar TWOD[]="Please open one drawer each time.";
const uchar senddat1[]="DRAWER #";
const uchar senddat2[]="open!";
const uchar senddat3[]="check in number:";
const uchar senddat4[]="check out number:";
```

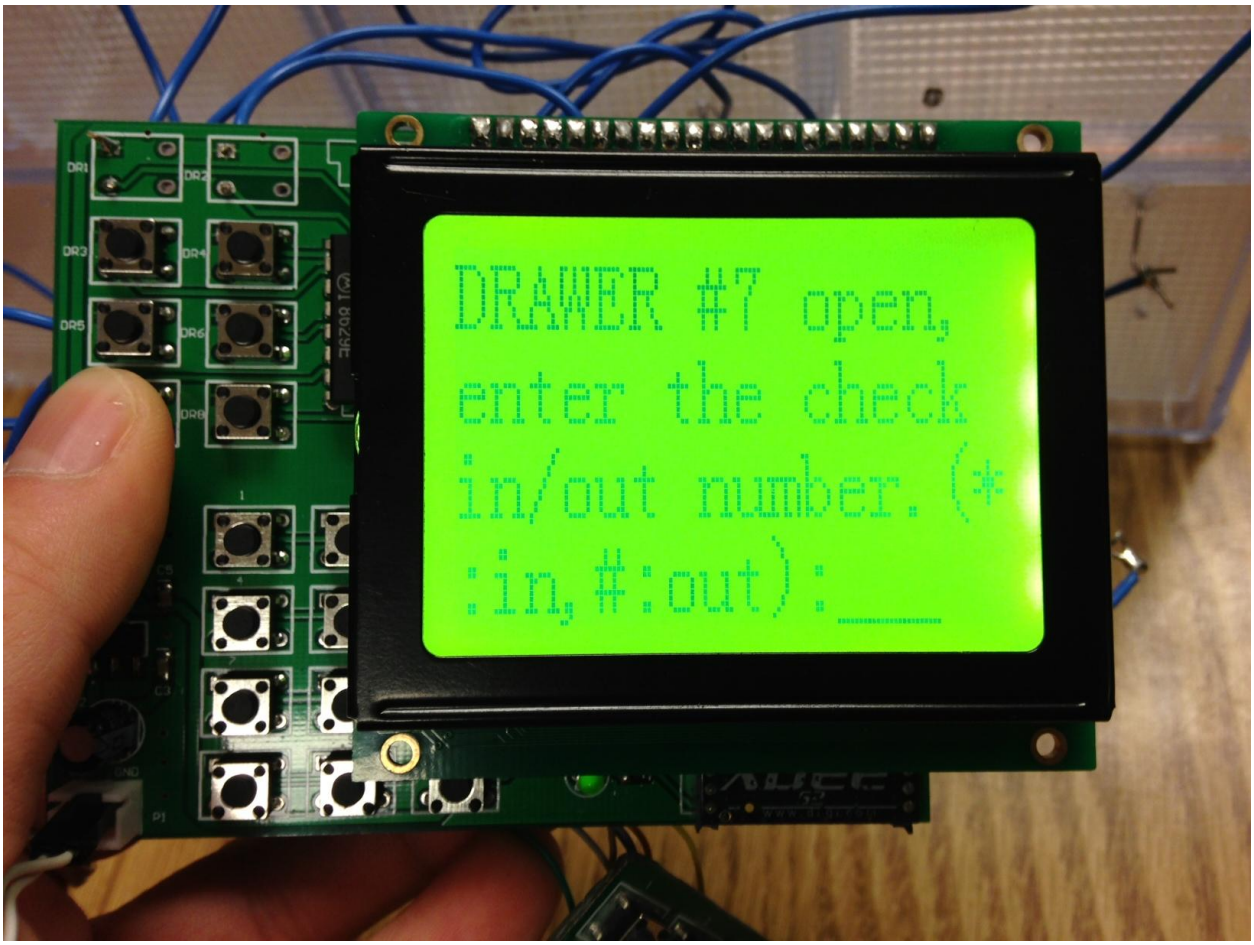


Figure 2.16 An example for LCD display

2.2.2.5 Power Supply and Voltage Regulation

There are two different standards for the power supply in this project. One is 5V, 1A DC and the other is 3.3V DC. The only part of this project that needs 3.3V DC power supply is the XBee PRO Module. This problem can be solved by using a 5V to 3.3V DC regulator. For the other parts of this project, 5V DC power supply is offered through a 120V AC to 5V, 1A DC converter as the following figure:

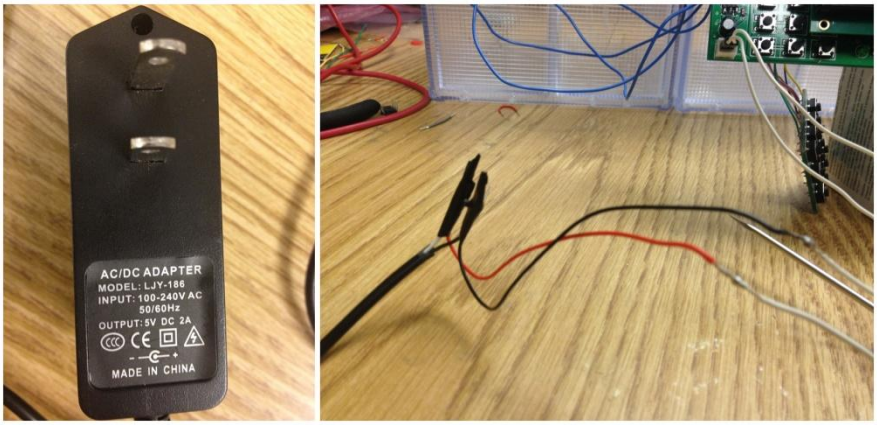


Figure 2.17 Power Supply and Voltage Regulation

On the right side of Figure XXX shows how this group changed the configuration of the adaptor. The original wires are cut off and reconnected with one red wire and one black wire for safety reason. Red represents the Vcc and Black represents the ground. The end of these two wires was soldered with a female header, in order to connect the male header of the PCB. For the 5V to 3.3V DC regulator, it is accomplished by several capacitors.

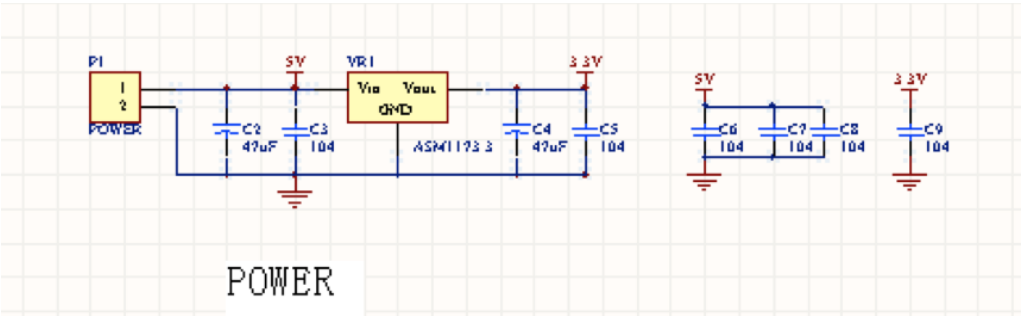


Figure 2.18 The schematic for Regulator

2.2.2.6 Wireless Data Transmission

Considering the situation that there are a lot of wires distributed in the modern laboratory, this project applies a wireless signal transfer technology through XBee Modules. Moreover, in order to get a stronger signal and transfer a longer distance, this project chooses to use the XBee PRO model. There will be totally two XBee PROs applied in the project, one works as a transmitter and the other works as a receiver. The following is the block diagram for the wireless transmission part:

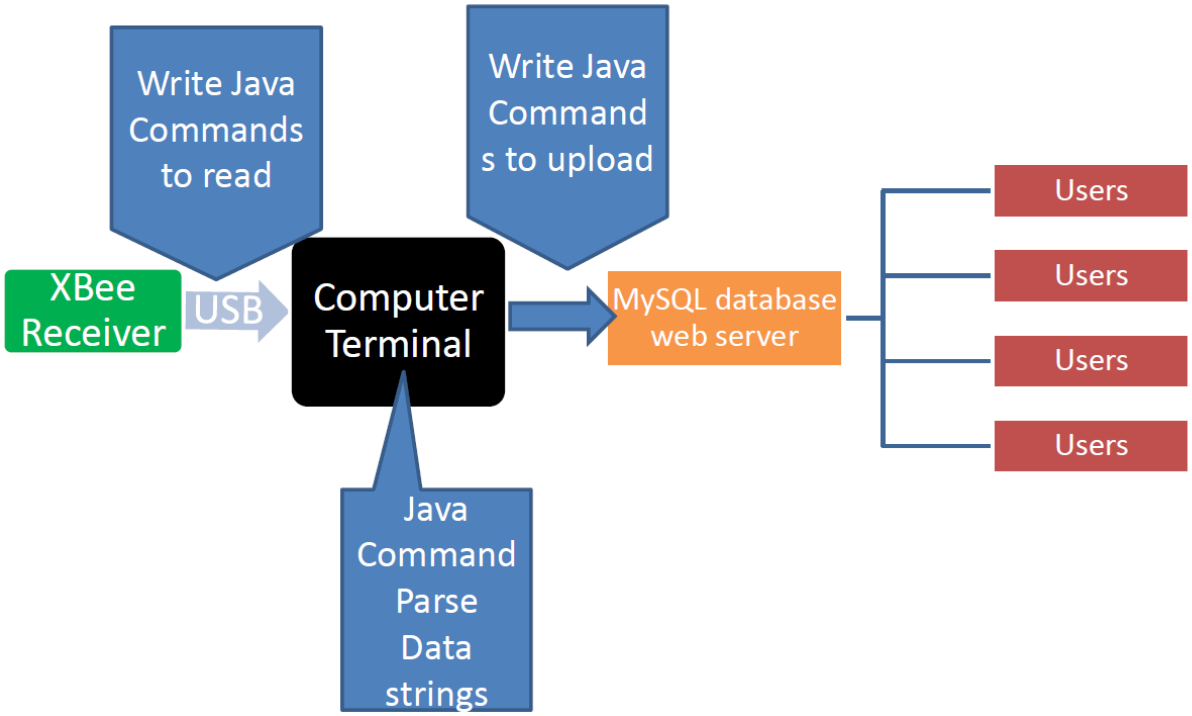


Figure 2.19 Block Diagram of Wireless data transmission

The XBee receiver gets the data from the PCB board and pushes the data through the USB connection to the computer terminal. By the Java application this group writes, the computer will read the data from the serial ports, parse the data and update them to the database. Every user can use their netid and password to view this online database. This project uses MySQL database to record the item amounts in each drawer. In the database, a table is created as the following figure shows:

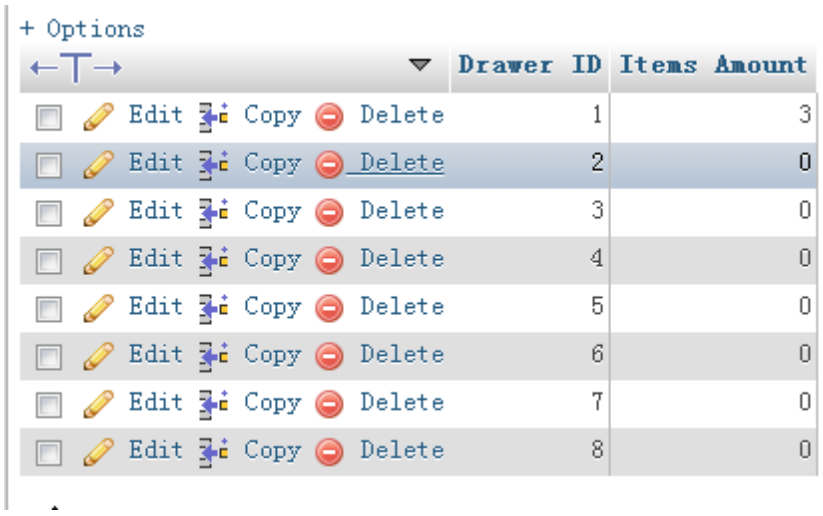


Figure 2.20 Database Inventory (In beginning)

There are totally eight drawers in the table. There's three items already have been check into drawer 1. Once this system is turned on, the inventory table will react to every change of item amounts in the drawer. For instance, a user checks in three items in drawer 1. After s/he refresh the online page of the inventory table, it will be like the following figure:

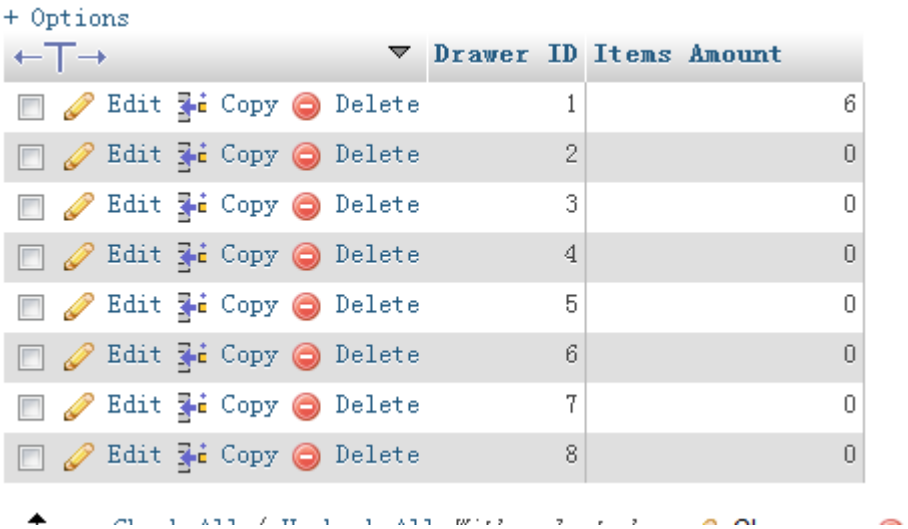


Figure 2.21 Database Inventory (In operation)

3. Design Verification

3.1 The Switch Signal

We test the drawer switch signal on breadboard in EL246 lab, after we finish design the drawer, we connect one of the iron wires from the drawer to a 5V voltage and connect the other wire to the multimeter to measure if the voltage signal of each drawer sending correctly. By doing this test, we found the initial design is unstable because the physical connecting problem between each wires, so we change a flat copper stripe and thicker wire on our final design. We connect two iron wires to the PCB and using the multimeter to testing the voltage signal send to the signal recognizer port on PCB for each drawer, we found that when the drawer is closed, the voltage is at range 0.075V to 0.246V; and when the drawer is opened. The voltage is at range 4.68V to 4.96V, which is in our expect value.

3.2 The MUX

For the MUX, we first test it by using I/O board to make sure it is working correct. Then we soldered the MUX chip On the PCB, one the PCB board we also designed 6 switch buttons as same function as the drawers to check the (since the drawer switch for our initial design disconnect sometimes, we redesign it after testing) if the MUX is operate in right function before we connect the real drawer switch on the PCB board.

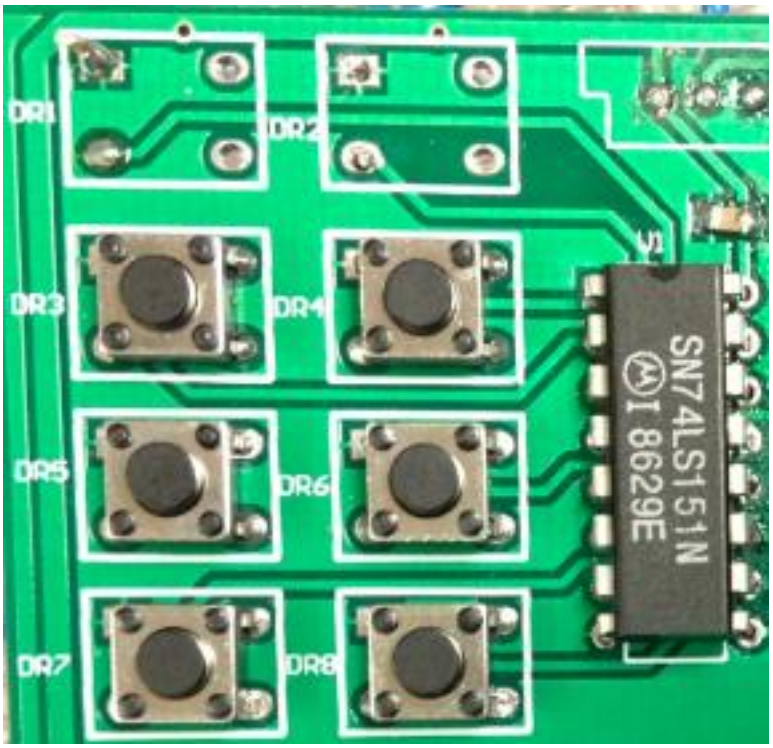


Figure 3.1 Testing button switches and MUX on PCB

3.3 The Keypad

For each keypad we use, we test each button on breadboard and I/O board one by one to make sure they are all working correctly before we solder them on the PCB. We used the LED on the I/O board to check each button if it send the correct signal when we press on it.

3.4 The PIC

Before we soldered the PIC on to PCB, we tested each function of the PIC on the breadboard, we using the I/O board LED to check if the output signal to the LCD is correct and we also using the oscilloscope to check if the selecting signal sent to MUX is correct. Since there are only 2 input for each oscilloscope and we have 3 selecting signal from PIC, so we test 2 of them each time and combine the 3 signal wave form together to check the signal, and the result is same as we set in programing, 1kHZ, 5V peak to peak.

3.5 The LCD Display

We test the LCD display directly on the PCB board, and each message shows correctly on it.

3.6 XBee Module

Since we use XBee to communicate wirelessly, the correct operation of the receiver and transmitter of XBee is crucial in our project. We test the functionality of XBee by the software called X-CTU downloaded from www.digi.com. During the test, we send sample signals from transimitter on PCB and see if the receiver can correctly get the signal. Here is what we got from the X-CTU:

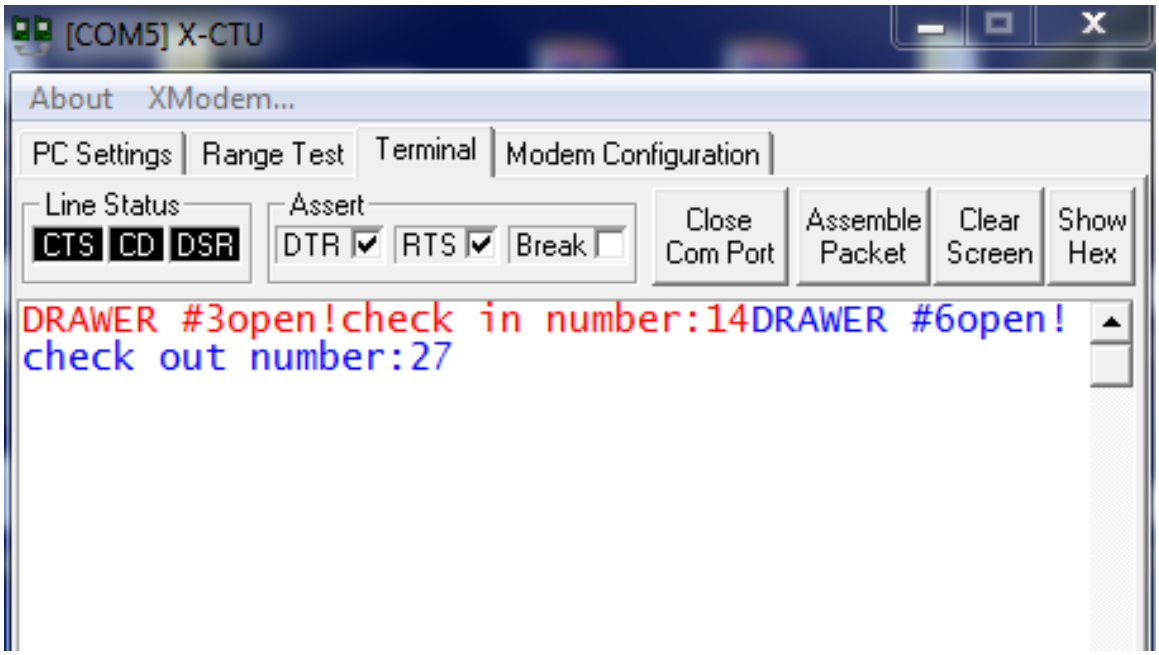


Figure 3.2 Test the XBee module

4. Costs

4.1 Parts

Part Name	Manufacture	Unit Cost	Quantity	Total Cost
Resistors, Capacitors, Inductors	Unknown	$0.05	20	$1
Keypad buttons	Unknown	$0.1	20	$2
XBee PRO 2mW Wire Antenna	Digi	$42.9	2	$85.8
Microcontroller(PIC16F877)	Motorola	$2	1	$2
PCB(Order from other company	Unknown	$20	2	$40
120AC to 5V DC converter	LG	$5.96	1	$5.96
MUX 74LS151	Motorola	$1.95	2	$3.9
Voltage Regulator LD1117	Unknown	$5.86	1	$5.86
Character LCD Module Display Screen	Qingyun Hi-Tech	$19.99	1	$19.99
4 Plastic Drawers	Amazon	$13	2	$26

4.2 Labor

Name	Rate	Hours	Total = Hourly Rate x 2.5 x Total Hours Invested
Chengcheng Huang	$35.00	150	$13,125
Chao Cao	$35.00	150	$13,125
Total		300	$26,250

5.1.3 Grand Total

Labor	$26,250
Parts	$191.5
Grand Total	**$26441.5**

5. Conclusion

For this project, all the function we expect is accomplishment, the drawer switches send the signal well, and the operations of PIC and other module are also working perfect. However, there are still some uncertainties for this project. If user pulls the whole drawer out of the drawer base, the circuit will also disconnect and the message will show "No drawer open". And there is a little error we found out for our final project, because the transmitted bit number from PIC to Computer is different between check in and check out, we could not get the right number value of check out. To fix this problem, we just need change the number of bits of the checkout message.

The initial idea for this system is use it in warehouse or supermarket. However, after talk to professor, he told us there is already exist scanning system using in supermarket, but it's seen useful to manager the small parts in our lab, so we choice this project. Moreover, we found this project is really useful on drug drawers in pharmacy store or hospital, which can help user easily to manager the drugs in storage.

6. Ethical and Safety Considerations

6.1 Ethical Issues

This project is strictly adhered to the statements of the IEEE Code of Ethics [12] that pertain to our project as follows:

- *"1. to accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment;"*

This project is designed to make the laboratory management easier to handle. It bring the lab users conveniences through automatically recording the number of pieces in drawer taken or increased. The whole design is under strict safety requirements, which promises both users and designers the maximum secure operations.

- " 3. *to be honest and realistic in stating claims or estimates based on available data;* "

This project will be experimented in practical and realistic environment. The data will be collected as detailed as possible. The formulas used in the design will be calculated iteratively at least three times to make sure that all calculations are accurate. Simulations are also required at the experimental stage.

- "7. *to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others;* "

  Since this is a group project, each group member should listen to all constructive and practical suggestions from other group members. This group will gives credits to each group member strictly and fairly depends on how much contribution he/she does.

6.2 Safety

One important requirement for this project is can be built, operated and maintained in a manner that does not harm any people. Some hazards should be eliminated or contained through good engineering design solutions. The designers of this project have already considered the following possible situations which may be dangerous to designers or users.

a. For Designers

During the design process, designers should always work under a safe condition. They should be careful when soldiering all the chips together on the PCB board and when connecting to high voltages.

b. For Users

During the operation process, there's nearly no possible dangerous situation that users can encounter when they are using this system. Most of this project work in a low voltage (safe to human-beings) environment.

References

[1] PIC 16F887 Data sheet

http://ww1.microchip.com/downloads/en/DeviceDoc/41291G.pdf

[2] LCM12864 Data Sheet

http://www.aix-mrs.iufm.fr/formations/filieres/ge/data/PIC/PICC/gdm12864a.pdf

[3] UART and Bridging Solutions

http://www.exar.com/connectivity/uart-and-bridging-solutions

[4] Transmission control protocol

https://tools.ietf.org/html/rfc793

[5] XBee Module

http://www.ladyada.net/make/xbee/configure.html

Appendix A Requirement and Verification Table

3.2 Drawer Switches:

Requirements	Verifications	Verification status (Y or N)
1) Each drawer switch sends out the I/O signal correctly.	1) a. Connecting the output pin of each drawer with a multimeter. b. Record the output value of open and close status of this drawer. c. Compare with the requirement output value. (When the drawer is closed, the output voltage should between 4.3V to 5.2V, when the drawer is opened, the voltage should between -0.7V to 0.7V)	1) Y
2) Each drawer switch controls its own output pin independently.	2) a. Connecting the output pin of the testing drawer switch with a multimeter. b. Keep the testing switch in on or off status. c. Turn on or off other switches to see if the output voltage signals of testing switch changes.	2) Y
3) MUX chip (74LS151) should work correctly.	3) a. Test the chip on the breadboard with I/O board. First connecting the chip with 5V Vcc and ground. b. Connecting the input pins and select pins with testing switches on I/O board. c. Make a truth table for the output signal. d. Connecting the output pin of the MUX chip with LED on I/O board. Turn on/off each testing switches, record each output signal and compare with the truth table.	3) Y
4) The 3-bit select signal sent from MCU should change correctly with constant	4) After finishing assign the program to the 3-bit select signal to MUX on MCU, connect the 3 pins of the signal to the oscilloscope	4) Y

frequency.	to check if it's in correct value and frequency.	

3.2 Keypad:

Requirements	Verifications	Verification status (Y or N)
1) Each button of the keypad working correctly.	1) To check if each button working correctly: a. Connecting the 3 column pins with LED on I/O board. b. Then connecting 1 of 4 row pins with 5V Vcc. Push each buttons on the power connecting row and check the change of LED light to see if it's working correctly. (Only one row pin can be connected with Vcc each time to make sure there's no other row power input to affect the result.) Check all the buttons row by row.	1) Y

3.3 MCU:

Requirements	Verifications	Verification status (Y or N)
1) Get familiar with PIC working function (Complete PIC tutorial)	1) Do some easy programing on PIC, such as light up an LED within the tutorial	1) Y
2) Connect each pin correctly.	2) Make sure finish all the simulations and check all the circuits before building the PCB board.	2) Y
3) All the programs should work in correct function.	3) After setup all connection of MCU. Test the program functions step by step. a. Make sure MCU can detect each drawer-switch input correctly. b. Make sure each output to LCD can display right message. c. Each button on keypad should input right data and MCU could calculate the numerical value of input data.	3) Y

4) The data transmit to XBEE and LCD should have correct transmission control protocol.	4) After setting up the transmission control protocol. Check if the message can display on LCD and transmit to Computer through XBEE (Through UART)	4) Y
5) Right power supply. (5V (±0.5 V) 1A AC)	5) See power supply part.	5) Y

3.4 LCD:

Requirements	Verifications	Verification status (Y or N)
1) Connect each pin correctly	1) Before connect the input pins, check the interface pin connections from the data sheet and get know the role of each pin	1) Y
2) Successful connecting and recognize the data sent from MCU.	2) a. Make sure each input connects to right pin. b. Check each display on LCD after do the display programing. Each message should match right drawer-switch.	2) Y

3.4 Wireless signal Transmittion:

Requirements	Verifications	Verification status (Y or N)
1) Connect each pin in XBee correctly	1) Before connect the input pins, check the interface pin connections from the data sheet of XBee and get know the role of each pin	1) Y
2) The XBEE module should function appropriately on an adapter with 3.3V (±0.5V) VCC from voltage	2) a. Verify the input VCC produced by voltage regulator using multimeter. b. Connect output pin 1-6 on adapter header to FTDI connector, the	2) Y

regulator	XBEE should be able to be Programmed on computer.	
3) The receiver should be able to received signal efficiently 25m (±5m) away from the transmitter.	3) Make a bunch of test code and put them into the transmitter. Put the receiver about 25m away from the transmitter and look at if it does receive the test code.	3) Y

3.5 Power Supply:

Requirements	Verifications	Verification status (Y or N)
1) 120V AC to 5V, 1A DC converter will offer the most of this project power supply 5V (±0.5 V).	1) Measure the voltage across battery by multimeter.	1) Y
2) LD1117 voltage regulators need to convert 9V to 3.3V (±0.5 V), which is the voltage require for the XBee operation	2) Build voltage regulator circuit on breadboard with 5V VCC, measure the output voltage using multimeter.	2) Y

3.6 Computer Inventory:

Requirements	Verifications	Verification status (Y or N)
1) Respond to the signal received by XBee correctly	1) Use several test data to check the accuracy of the code programmed	1) Y
2) Change the inventory automatically when MySQL is open	2) Use several test data and see the simulation result to check the inventory changes correctly	2) Y
3) After each operation, store a hard copy of data on the desktop	3) Open the desktop folder and load the data saved from last time and compare to the data record in last time.	3) Y

Appendix B Java Codes for the database update and load

```
import gnu.io.CommPort;
import gnu.io.CommPortIdentifier;
import gnu.io.SerialPort;

import java.io.BufferedReader;
import java.io.FileDescriptor;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class TwoWaySerialComm
{
        private String request;

    public TwoWaySerialComm()
    {

        super();
    }

    /**
     * use serial port to get connection with XBee and send queries to mySQL
database
     * @param portName We set our default port name COM5
     * @throws Exception show exception errors
     */
    void connect ( String portName ) throws Exception
    {
        CommPortIdentifier portIdentifier =
CommPortIdentifier.getPortIdentifier(portName);
        if ( portIdentifier.isCurrentlyOwned() )
        {
            System.out.println("Error: Port is currently in use");
        }
        else
        {
            CommPort commPort =
portIdentifier.open(this.getClass().getName(),2000);

            if ( commPort instanceof SerialPort )
            {
                //read from serial port
                 SerialPort serialPort = (SerialPort) commPort;

serialPort.setSerialPortParams(9620,SerialPort.DATABITS_8,SerialPort.STOPBITS
_1,SerialPort.PARITY_NONE);
```

```
                InputStream in = serialPort.getInputStream();

                byte[] buffer = new byte[1024];
                int len = -1;
                try
                {
                    while ( ( len = in.read(buffer)) > -1 )
                    {
                       //print out serial port information
                       request = new String(buffer,0,len);
                        System.out.print(request);

                        //parse out information get from the serial port
                        if(request.length()>1){

                                int drawerNum =
Character.getNumericValue(request.charAt(7));
                                int amount = 0;
                                if(request.charAt(19) == 'i'){
                                        amount =
Integer.parseInt(request.substring(29));
                                }
                                else if(request.charAt(19) == 'o'){
                                        amount = -
Integer.parseInt(request.substring(30));
                                }

 /**                            for ( int i  = 0 ; i < 31 ; i ++){
                                        System.out.println("character number "+
i + " is: " + request.charAt(i));
                                }
*/
                                System.out.println("Drawer Number is: " +
drawerNum);
                                System.out.println("Amount change is: " +
amount);

                                //Send queries to mySQL
                                try{
                                        Connection conn =
DriverManager.getConnection("jdbc:mysql://ardenview.com:3306/ardencfg_SD","ar
dencfg_real","1q2w3e");

                                        //Download information from the
database
                                        int currentAmount = 0;
                                        String sql = "Select amount from drawer
where id= " + drawerNum;
                                        Statement st = conn.createStatement();
                                        ResultSet rst = st.executeQuery(sql);

                                        while(rst.next()){
                                                currentAmount = rst.getInt(1);
                                        }
```

```
                                        //Update the amount
                                        currentAmount += amount;

                                        //Upload the amount to the database
                                        sql = "Update drawer set amount= ?
where id= ? ";
                                        PreparedStatement preparedStmt =
conn.prepareStatement(sql);
                                    preparedStmt.setInt   (1, currentAmount);
                                    preparedStmt.setInt   (2, drawerNum);
                                    preparedStmt.executeUpdate();
                                        conn.close();
                                }
                                catch(SQLException sqle){
                                        sqle.printStackTrace();
                                        System.exit(1);
                                }

                            }
                        }

                    }
                    catch ( IOException e )
                    {
                        e.printStackTrace();
                    }

                    OutputStream out = serialPort.getOutputStream();

                    (new Thread(new SerialReader(in))).start();
                    (new Thread(new SerialWriter(out))).start();

                }
                else
                {
                    System.out.println("Error: Only serial ports are handled by
this example.");
                }
            }
        }

        /** */
        public static class SerialReader implements Runnable
        {
            InputStream in;

            public SerialReader ( InputStream in )
            {
                this.in = in;
            }

            public void run ()
            {
                byte[] buffer = new byte[1024];
                int len = -1;
                try
                {
```

```
                while ( ( len = this.in.read(buffer)) > -1 )
                {
                    System.out.print(new String(buffer,0,len));
                }
            }
            catch ( IOException e )
            {
                e.printStackTrace();
            }
        }
    }

    /** */
    public static class SerialWriter implements Runnable
    {
        OutputStream out;

        public SerialWriter ( OutputStream out )
        {
            this.out = out;
        }

        public void run ()
        {
            try
            {
                int c = 0;
                while ( ( c = System.in.read()) > -1 )
                {
                    this.out.write(c);
                }
            }
            catch ( IOException e )
            {
                e.printStackTrace();
            }
        }
    }

    public static void main ( String[] args )
    {
        try
        {
               //Set Default Port to "COM5"
            (new TwoWaySerialComm()).connect("COM5");
        }
        catch ( Exception e )
        {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```