A

Wireless LED Display

EECS 129B COMP SENIOR PROJECT, Prof. Raymond Klefstad

Brandon Stark
Trinh Nguyen
Seok Choi

A wireless LED display made by DOT matrices LEDs. It can display pre-set messages chosen by the user. The User interface will consist of a LCD and button selection that will communicate wirelessly with the LED display.

A table of contents

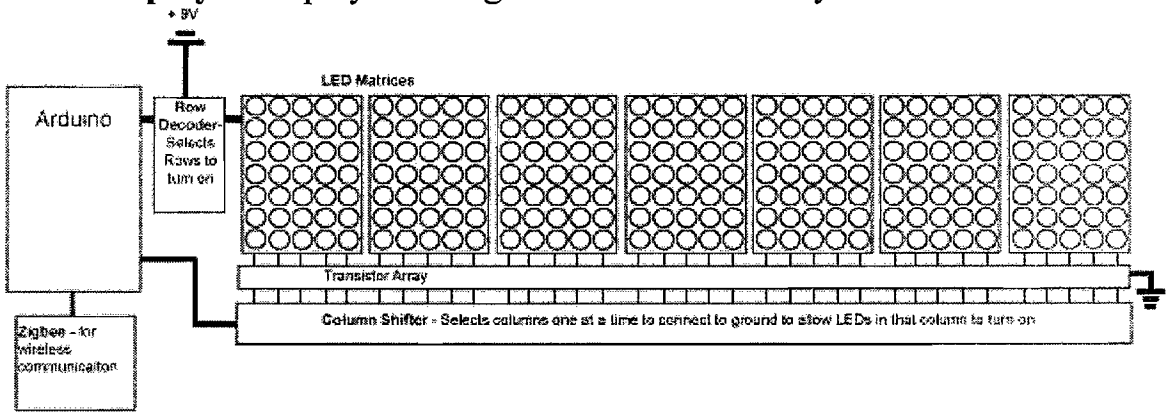
Project description

Wireless dot matrix light emitting diode (LED) display system allows you to express yourself effectively. It comes with a remote control so you can easily switch between messages, which are 5 pre-composed messages (up to 7 characters in length each). You are able to select any one of them to show. You can broadcast your own custom message and express yourself or use it to advertise your web site or business. The display is great for attracting attention and works great as an advertising tool! Using the bright LEDs, your message is able to be seen day or night. The possibilities of this system are endless! Here are some examples:

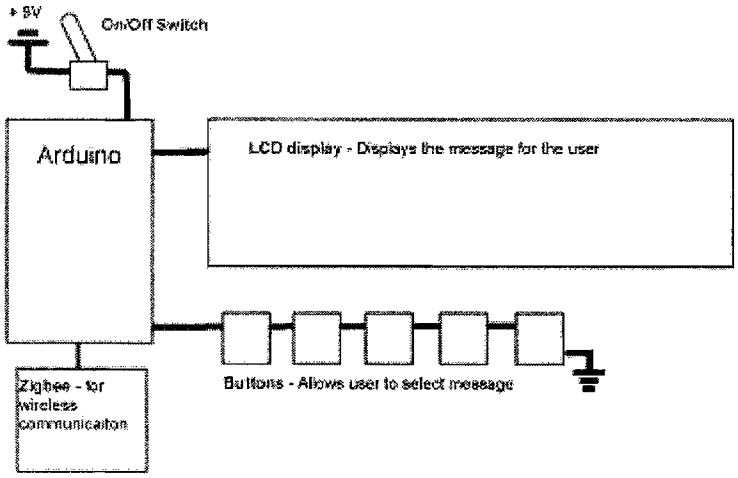
- You can express your feelings or thoughts
- You can promote your business
- You can promote your web site
- You Can mount on a car to warn surrounding drivers
- You can advertise your favorite products

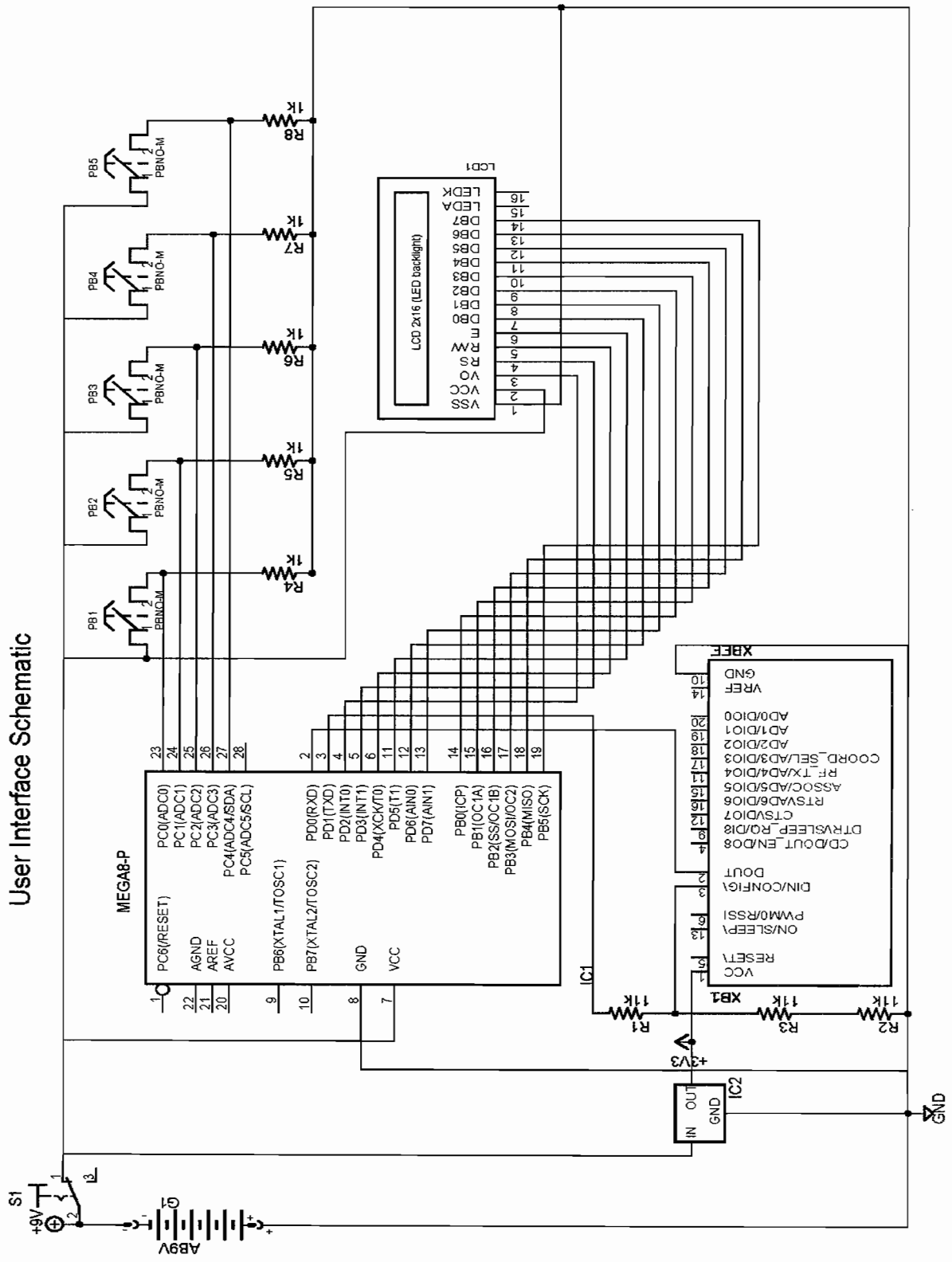
System Level Block Diagram

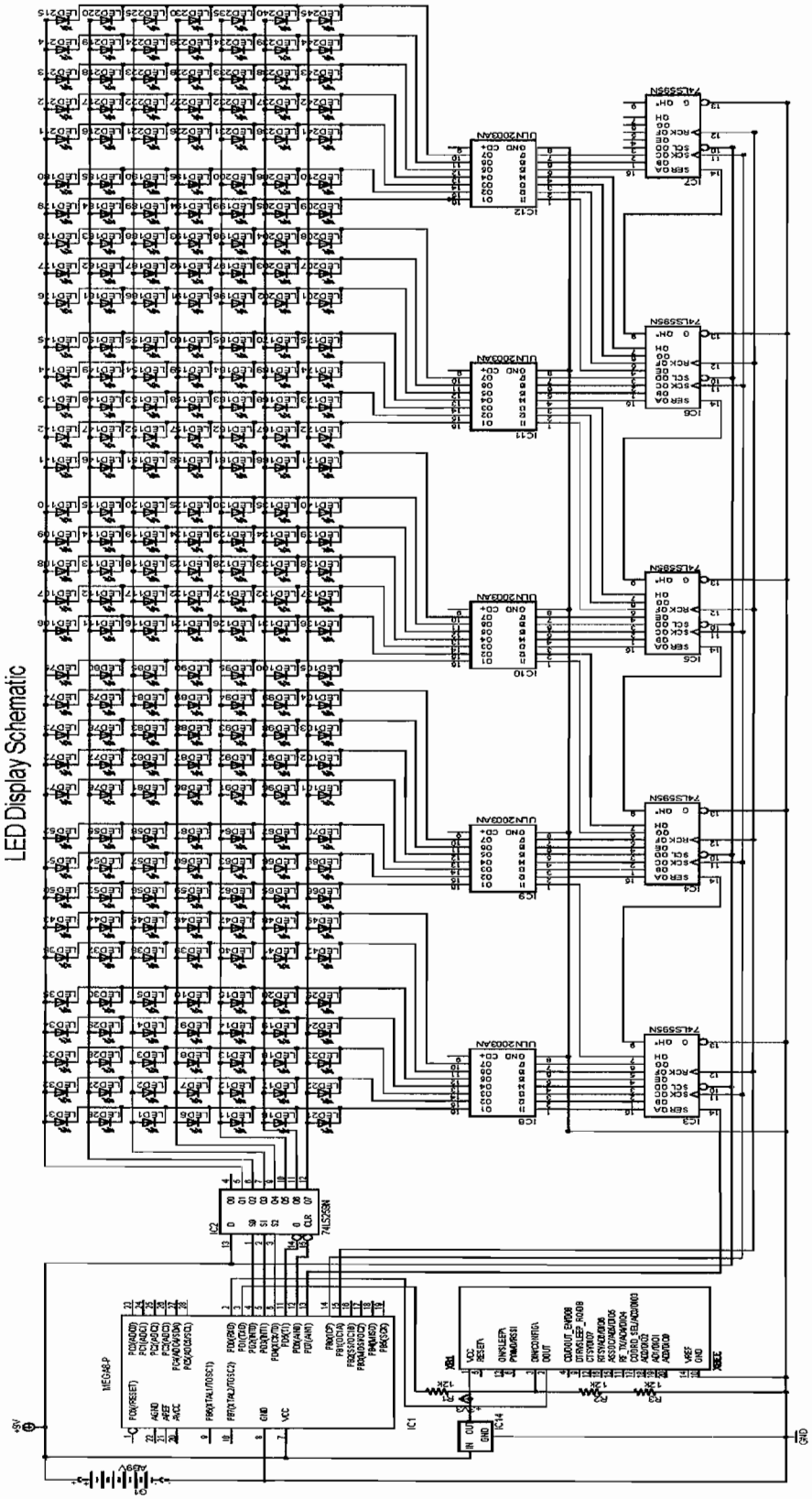
LED Display – Displays messages received wirelessly.



LCD Display and User Interface - Allows the user to select a message and displays it back using an LCD display




User Interface Schematic
MEGA8-P
LCD1
LCD 2x16 (LED backlight)
XBEE
XB1
IC1
IC2
S1
G1
+9V
AB9V
+3V3
GND
R1
R2
R3
R4
R5
R6
R7
R8
1k
11k
PB1
PB2
PB3
PB4
PB5
PBNO-M


LED Display Schematic

Software Description

Transmitter –

For the transmitter as well as the receiver, we coded the Arduino boards using the arduino.exe program which closely resembles C code. For the transmitter unit, in order to use the 16x2 LCD, we include the library for LiquidCrystal which allows us to initialize the LCD. We create an instance of the LiquidCrystal lcd in order to print to the screens. We also create character arrays that store string messages that will be displayed onto the LCD. Since the buttons on our transmitter are connected to the Arduino through the analog pins, we created variables to hold the button identifier values as well as the analog values. At first in the setup, we set the baud rate as well as initialize the LCD. For debugging purposes we also set the ledPin, pin 13, to an output. Inside the loop of the main program, we send out a serial print in order to put the Xbee into command mode. We do this by sending out the string "+++" after which we change the ID to 'AT4321'. To exit this command mode we then send out a serial print string of 'ATCN'. We only wanted to do this once, so we created a boolean variable that would turn false after this was done once.*

Once the Xbee is ready, we read in the values from the buttons with the analogRead() function and store them in variables called 'tempx' where x is the button number. Since the analog pins are receiving no voltage because they are grounded, the values of these 'temp' values are equal to zero. When buttons are pressed the values are raised, and therefore we created if statements to check to see if a button was pressed. If a button is noted as being pressed, a variable known as 'var' is changed. In a switch case, the 'var' variable is looked at and depending on that value, corresponding to the which button was pressed, a message is printed to the LCD as well as a serial print is printed so that it can be transmitted through the Xbee to receiver connected to the LED display. During each of these steps, the lcd is first cleared as well as delayed each time through the loop.

*See Appendix A-1

Software Description (cont.)

Receiver –

For the receiver the Arduino code used puts the receiver Xbee into command mode and then changes to ATID4321. Also the baud rate is set to 9600. Inside of the main loop, the code checks to see if there is something being sent using the Serial.available( ) function. If this value is greater than zero then we assign the value to the variable 'var' with the Serial.read( ) command. Depending on the value of 'var' we turn on a corresponding test LED as well as call upon string methods stringX( ) where 'x' denotes the message corresponding to the value received from the transmitter. The stringX methods then output the corresponding hard coded message to the DOT matrix LED's by cycling through the columns and rows. For the stringX( ) methods, we used our own character generation in order to print each letter to each corresponding DOT matrix LED. There are 5 pre-defined messages each corresponding to the characters "a, b, c, d, e" that are sent out from the transmitter as a byte. When the byte is read by the receiver then we start to deal with the timing for our LED display.*

The most critical portion of the LED display software is the timing. Because there are not enough pins to light up each LED individually, a combination of a shift registers and an 8bit decoder are used to light up a single column at a time. In order to maintain a stable image, the entire message has to be displayed under 25ms. In this set up, each of the 35 columns are lit up for less than 600 μs.

Once a message has been selected by receiving a byte from the transmitter, a display function is called. The first time the display function is called, the first column is lit up by sending 1 to the shift register. At the beginning of each column in the letter, the decoder is cleared and then the shift register is clocked, which sets the column to be active. The next step is to set to high the row LEDs that need to be lit up. The Arduino sends a message for the decoder to enter a memory mode, which allows multiple outputs to be turned on. Next, the row is selected and the decoder leaves memory mode. This is repeated until all the required rows are selected. After a delay of 600 μs, the Arduino clocks the shift register, setting the next column active. The decoder is then cleared and new rows are selected. This process continues for all 35 columns until the end is reached. The process is looped until the next message is selected.**

*See Appendix A-2
**See Appendix B-1

System test plan

This system can be easily tested by anyone using the controller and LED display.

i) Controller part

-Turning on and off: Use the switch on the controller. You can test by watching LCD screen on the controller. If it is ready, you will see the message such as "select a message" on the LCD screen.

-Choosing pre-programmed message using buttons: Push any button on the controller, and verify that the message, which belongs to the button, shows up on the LCD screen.

-Selecting different message: Select different button on the controller, and confirm that the message has been changed.

ii) LED display part

-After choosing the message by using the controller, verify that same text message with the one on the controller displays on LED and check whether every letter is easily recognizable.

Cost Analysis

Quantity	Supplier	Part Number	Description	Price	Total
2	www.sparkfun.com	Arduino-USB	Arduino USB Board	$31.95	$63.90
1	www.sparkfun.com	LCD-16x2-Green	Basic 16x2 Character LCD STN - Black on Green	$11.95	$11.95
2	www.sparkfun.com	Proc-AM8	Atmega8	$3.66	$7.32
1	www.digikey.com	3009Y-105-ND	Variable Resistor POT 1.0M OHM	$2.72	$2.72
2	www.digikey.com	XB24-AWI-001-ND	MODULE ZIGBEE 1MW W/WIRE ANT	$23.20	$46.40
2	www.digikey.com	497-1491-5-ND	IC LDO REGULATOR +3.3V TO-220	$0.77	$1.54
7	www.jameco.com	206560PS	Dot Matrix LED Display	$1.65	$11.55
2	www.jameco.com	216451	9V Battery connector	$0.34	$0.68
5	www.jameco.com	149948	SWITCH,PB,TACT,SPST,OFF-(ON)	$0.23	$1.17
1	www.jameco.com	72160	SWITCH,TOG,SPST,ON-OF(SMTS101)	$1.09	$1.09
5	www.jameco.com	34278	IC,ULN2003A	$0.39	$1.95
5	www.jameco.com	461105	IC, 74HC595	$0.62	$3.10
1	www.jameco.com	915817	8-bit Addressable Latch 74HC259	$0.25	$0.25
6	www.jameco.com	691121	12kOhm resistors CF1/4W123JRC	$0.01	$0.06
2	Fry's Electronics		9V Battery	$1.50	$3.00
				Total	**$153.68**

Summary

While our group recognizes this senior design project as a success, we understand that there were many problems that we ran into along the way in both the design and implementation process. The system tests that we created were all met as well as satisfactory fabrication, but we will now outline problems that arose during this quarter.

One of the first problems that we ran into was not the late ordering of parts form last quarter, but the re-ordering of parts based on changing designs. This was a particular issue when we ordered shift registers that were too small for our needs and were not able to be soldered. We then had to re-implement with new shift registers. After we received the new shift registers about 2 weeks into the quarter, we were able to move on with the receiver portion of the project.

Another problem that we encountered was getting the LED's to cycle through faster than 40Hz. We needed the LED's to cycle through this fast because it allows for a constant image to be displayed. We solved this by displaying the LEDs by columns instead of individually cycling through each individual LED. This allowed us to meet the 40Hz requisite. We also ran into problems with the brightness of the LEDs. The simple solution was to use the 9V battery to power the Arduino board and the LEDs however, cycling at 40Hz needed more voltage and more current. To drive the LEDs to be on for about half of a microsecond, using a single 9V battery will last about 30 min. A solution that wasn't implemented but could be used is 8 AA batteries that will allow for approx. 3 hours of run time because of the added current.

The biggest problem we ran into was having the two Arduino boards communicate with one another. We ran into many problems getting the serial ports to work either transmitting or receiving the correct message without added garbage from the wireless signal. We even tried to connect directly between the pins as well and still received problems. We corrected this by testing the serial ports one by one on the receiver and the transmitter to make sure that it was in receive and transmit modes and that the information being sent was correct. Solving all of these problems we feel that we were successful in our plans for this quarter.

Appendix. A-1 Program Code for Transmitter

```
/* Transmitter Code
 */
#include <LiquidCrystal.h> //include LiquidCrystal library

LiquidCrystal lcd = LiquidCrystal(); //create a LiquidCrystal object to control an LCD
char string1[] = "Hello"; //variable to store the string "Hello!"
char string2[] = "EECS129";
char string3[] = "Warning";
char string4[] = "Danger";
char string5[] = "Help Me";
char intro[] = "Select a Message";
boolean flag = false;

int button1 = 0;
int button2 = 1;
int button3 = 2;
int button4 = 3;
int button5 = 4;
int temp1 = 0;
int temp2 = 0;
int temp3 = 0;
int temp4 = 0;
int temp5 = 0;
int var = 0;

void setup() {
  lcd.init(); //initialize the LCD
  Serial.begin(9600);
  //pinMode(ledPin, OUTPUT);
}

void loop()
{
  if (flag == false){
     Serial.print("+++");
     delay(100);
     Serial.print("ATID4321\n");
     //Serial.print("ATCN\n");
    flag=true;
  }
   lcd.clear();
  temp1 = (analogRead(button1));
  temp2 = (analogRead(button2));
  temp3 = (analogRead(button3));
  temp4 = (analogRead(button4));
  temp5 = (analogRead(button5));

  if(temp1 > 1000)
  {
     var = 1;
    // digitalWrite(ledPin, HIGH);
  }
  if(temp2 > 1000)
  {
```

```
    var = 2;
  }
  if(temp3 > 1000)
  {
    var = 3;
  }
  if(temp4 > 1000)
  {
    var = 4;
  }
  if(temp5 > 1000)
  {
    var = 5;
  }
  switch(var){
   case 1:
    lcd.clear(); //clear the display
    lcd.printIn(string1); //send the string to the LCD
    Serial.print("a");
    //delay(200);
    break;

   case 2:
    lcd.clear(); //clear the display
    lcd.printIn(string2); //send the string to the LCD
    Serial.print("b");
    //delay(200);
    break;

   case 3:
    lcd.clear(); //clear the display
    lcd.printIn(string3); //send the string to the LCD
    Serial.print("c");
    //delay(200);
    break;

   case 4:
     lcd.clear(); //clear the display
     lcd.printIn(string4); //send the string to the LCD
     Serial.print("d");
     //delay(200);
     break;

   case 5:
     lcd.clear(); //clear the display
     lcd.printIn(string5); //send the string to the LCD
     Serial.print("e");
     //delay(200);
     break;
   default:
      lcd.printIn(intro); //send the string to the LCD
  }
  delay(200);
}
```

Appendix. A-2 Program Code for Receiver

```
/* Receiver
*/
#define pinS0 2
#define pinS1 3
#define pinS2 4
#define pinClr 5
#define pinG 6
#define pinDin 7
#define pinRCK 8
#define pinSCK 9
#define delayM 300

boolean flag = false;
byte var;
int LEDpin =13;
int LEDpin2 =12;
int LEDpin3 =11;
int LEDpin4 =10;
int message = 0;

void setup()
{
 //Outputs
 //74HC259
 pinMode(pinS0, OUTPUT);
 pinMode(pinS1, OUTPUT);
 pinMode(pinS2, OUTPUT);
 pinMode(pinClr, OUTPUT);
 pinMode(pinG, OUTPUT);
 Serial.begin(9600);

 //74HC595
 pinMode(pinDin, OUTPUT);
 pinMode(pinRCK, OUTPUT); //STcp Register Clock
 pinMode(pinSCK, OUTPUT); //SHcp Shift Clock

 pinMode(LEDpin, OUTPUT);
 pinMode(LEDpin2, OUTPUT);
 pinMode(LEDpin3, OUTPUT);
 pinMode(LEDpin4, OUTPUT);
}

void loop()
{
 if (flag == false){
```

```
  Serial.print("+++");
  delay(100);
  Serial.print("ATID4321\n");
  //Serial.print("ATCN\n");
 flag=true;
}
if (Serial.available() > 0) {
                // read the incoming byte:
                var = Serial.read();
        //digitalWrite(LEDpin, HIGH);
        Serial.print(var);

        if(var =='a')
          {
           digitalWrite(LEDpin, HIGH);
           message = 1;
          }
        if(var =='b')
          {
           digitalWrite(LEDpin2, HIGH);
           message = 2;
          }
        if(var =='c')
          {
           digitalWrite(LEDpin3, HIGH);
           message = 3;
          }
        if(var =='d')
          {
          digitalWrite(LEDpin4, HIGH);
          message = 4;
          }
         if(var =='e')
          {
          digitalWrite(LEDpin4, HIGH);
          message = 5;
          }
}

switch(message) {
 case 1:
    display1();
  break;
case 2:
    display2();
  break;
case 3:
```

```
      display3();
      break;
   case 4:
      display4();
       break;
   case 5:
      display5();
      break;
  }
}
void display1(){
  //Start Din
  digitalWrite(pinDin, HIGH);

  H();
  El();
  L();
  L();
  O();
  space();
  space();
}


//*************************************************************************************************
void display2(){
  //Start Din
  digitalWrite(pinDin, HIGH);

  E();
  E();
  C();
  S();

  //***********************************************
  Clear(); //clear row select
   clock();
  //Set Rows

  Row0();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
```

```
Clear();
clock();

Row2();
Row7();

delayMicroseconds(delayM);
//*******************************
//Column 3
Clear();
clock();

fullColumn();

delayMicroseconds(delayM);
//*******************************
//Column 4
Clear();
clock();

Row7();

delayMicroseconds(delayM);
//*******************************
//Column 5
Clear();
clock();

Row0();

delayMicroseconds(delayM);

//*********************************************
Clear(); //clear row select
 clock();
//Set Rows

Row2();
Row5();
Row6();
Row7();

delayMicroseconds(delayM);

//Only keep Din high for first clock
digitalWrite(pinDin,LOW);
//*******************************
```

```
//Column 2
Clear();
clock();

onefourseven();

delayMicroseconds(delayM);
//*******************************
//Column 3
Clear();
clock();

onefourseven();


delayMicroseconds(delayM);
//*******************************
//Column 4
Clear();
clock();

onefourseven();

delayMicroseconds(delayM);
//*******************************
//Column 5
Clear();
clock();

Row2();
Row3();
Row7();

delayMicroseconds(delayM);

//*********************************************
Clear(); //clear row select
 clock();
//Set Rows

Row2();
Row3();

delayMicroseconds(delayM);

//Only keep Din high for first clock
digitalWrite(pinDin,LOW);
```

```
//*********************************
//Column 2
Clear();
clock();

onefourseven();

delayMicroseconds(delayM);
//*********************************
//Column 3
Clear();
clock();

onefourseven();

delayMicroseconds(delayM);
//*********************************
//Column 4
Clear();
clock();

Row1();
Row4();
Row6();

delayMicroseconds(delayM);
//*********************************
//Column 5
Clear();
clock();

Row2();
Row3();
Row4();
Row5();


delayMicroseconds(delayM);


}

//*****************************************************************************************************
void display3(){
 //Start Din
 digitalWrite(pinDin, HIGH);
```

```
  W();
  A();
  R();
  N();
  I();
  N();
  G();

}
//*************************************************************************************************
void display4(){
 //Start Din
 digitalWrite(pinDin, HIGH);

 D();
 A();
 N();
 G();
 El();
 R();
 space();

}
//*************************************************************************************************
void display5(){
 //Start Din
 digitalWrite(pinDin, HIGH);

 H();
 El();
 L();
 P();
 space();
 M();
 El();

}
//*************************************************************************************************




void clock(){
 digitalWrite(pinSCK, HIGH);
 digitalWrite(pinRCK, HIGH);
 digitalWrite(pinSCK, LOW);
 digitalWrite(pinRCK, LOW);
```

```
}
void clock1(){
}

void Row1(){
 digitalWrite(pinG, HIGH);

 digitalWrite(pinS0, HIGH);
 digitalWrite(pinS1, LOW);
 digitalWrite(pinS2, LOW);

 digitalWrite(pinG, LOW);
}

void Row2(){
 digitalWrite(pinG, HIGH);

 digitalWrite(pinS0, LOW);
 digitalWrite(pinS1, HIGH);
 digitalWrite(pinS2, LOW);

 digitalWrite(pinG, LOW);
}

void Row3(){
 digitalWrite(pinG, HIGH);

 digitalWrite(pinS0, HIGH);
 digitalWrite(pinS1, HIGH);
 digitalWrite(pinS2, LOW);

 digitalWrite(pinG, LOW);
}

void Row4(){
 digitalWrite(pinG,HIGH);

 digitalWrite(pinS0, LOW);
 digitalWrite(pinS1, LOW);
 digitalWrite(pinS2, HIGH);

 digitalWrite(pinG,LOW);
}

void Row5(){
 digitalWrite(pinG,HIGH);
```

```
 digitalWrite(pinS0, HIGH);
 digitalWrite(pinS1, LOW);
 digitalWrite(pinS2, HIGH);

 digitalWrite(pinG,LOW);
}

void Row6(){
 digitalWrite(pinG,HIGH);

 digitalWrite(pinS0, LOW);
 digitalWrite(pinS1, HIGH);
 digitalWrite(pinS2, HIGH);

 digitalWrite(pinG,LOW);
}
void Row7(){
 digitalWrite(pinG,HIGH);

 digitalWrite(pinS0, HIGH);
 digitalWrite(pinS1, HIGH);
 digitalWrite(pinS2, HIGH);

 digitalWrite(pinG,LOW);
}
void Row0(){
 digitalWrite(pinG,HIGH);

 digitalWrite(pinS0, LOW);
 digitalWrite(pinS1, LOW);
 digitalWrite(pinS2, LOW);

 digitalWrite(pinG,LOW);
}

void fullColumn(){
 Row1();
 Row2();
 Row3();
 Row4();
 Row5();
 Row6();
 Row7();
}

void lfullColumn(){
 Row3();
```

```
 Row4();
 Row5();
 Row6();
 Row7();
}

void onefourseven(){
 Row1();
 Row4();
 Row7();
}
void threefiveseven(){
 Row3();
 Row5();
 Row7();
}

void A(){
 //***********************************************
 Clear(); //clear row select
  clock();
 //Set Rows

 Row6();


 delayMicroseconds(delayM);

 //Only keep Din high for first clock
 digitalWrite(pinDin,LOW);
 //*******************************
 //Column 2
 Clear();
 clock();

 threefiveseven();

 delayMicroseconds(delayM);
 //*******************************
 //Column 3
 Clear();
 clock();

 threefiveseven();

 delayMicroseconds(delayM);
 //*******************************
```

```
  //Column 4
  Clear();
  clock();

  threefiveseven();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();


  Row4();
  Row5();
  Row6();


  delayMicroseconds(delayM);
}

void C(){
  //************************************************
  Clear(); //clear row select
   clock();
  //Set Rows

  Row2();
  Row3();
  Row4();
  Row5();
  Row6();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  Row1();
  Row7();

  delayMicroseconds(delayM);
  //*******************************
```

```
  //Column 3
  Clear();
  clock();

  Row1();
  Row7();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
  Clear();
  clock();

  Row1();
  Row7();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();

  Row2();
  Row6();

  delayMicroseconds(delayM);
}


void H(){
  //*********************************************
  Clear(); //clear row select
   clock();
  //Set Rows
  fullColumn();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  Row4();
```

```
  delayMicroseconds(delayM);
  //*******************************
  //Column 3
  Clear();
  clock();

  Row4();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
  Clear();
  clock();

  Row4();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();

  fullColumn();

  delayMicroseconds(delayM);
}
void I(){
  //************************************************
  Clear(); //clear row select
   clock();
  //Set Rows
  Row0();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  Row3();
  Row7();

  delayMicroseconds(delayM);
  //*******************************
```

```
  //Column 3
  Clear();
  clock();

  Row1();
  IfullColumn();

  delayMicroseconds(delayM);
  //*********************************
  //Column 4
  Clear();
  clock();


  Row7();

  delayMicroseconds(delayM);
  //*********************************
  //Column 5
  Clear();
  clock();

  Row0();

  delayMicroseconds(delayM);
}
void W(){
  //************************************************
  Clear(); //clear row select
   clock();
  //Set Rows
  fullColumn();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*********************************
  //Column 2
  Clear();
  clock();

  Row6();

  delayMicroseconds(delayM);
  //*********************************
  //Column 3
```

```
  Clear();
  clock();

  Row4();
  Row5();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
  Clear();
  clock();

  Row6();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();

  fullColumn();

  delayMicroseconds(delayM);
}

void M(){
  //*********************************************
  Clear(); //clear row select
   clock();
  //Set Rows
  fullColumn();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  Row2();

  delayMicroseconds(delayM);
  //*******************************
  //Column 3
  Clear();
```

```
  clock();

  Row3();
  Row4();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
  Clear();
  clock();

  Row2();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();

  fullColumn();

  delayMicroseconds(delayM);
}

void D(){
  //***********************************************
  Clear(); //clear row select
   clock();
  //Set Rows
  Row1();
  Row7();


  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  fullColumn();

  delayMicroseconds(delayM);
  //*******************************
  //Column 3
```

```
  Clear();
  clock();

  Row1();
  Row7();

  delayMicroseconds(delayM);
  //*********************************
  //Column 4
  Clear();
  clock();

  Row1();
  Row7();

  delayMicroseconds(delayM);
  //*********************************
  //Column 5
  Clear();
  clock();

  Row2();
  Row4();
  Row5();
  Row6();

  delayMicroseconds(delayM);
}

void G(){
  //*********************************************
  Clear(); //clear row select
   clock();
  //Set Rows

  Row4();


  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*********************************
  //Column 2
  Clear();
  clock();
```

```
  threefiveseven();


  delayMicroseconds(delayM);
  //*******************************
  //Column 3
  Clear();
  clock();

  threefiveseven();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
  Clear();
  clock();

  threefiveseven();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();

  lfullColumn();

  delayMicroseconds(delayM);
}

void E(){
  //***********************************************
  Clear(); //clear row select
   clock();
  //Set Rows
  fullColumn();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  onefourseven();
```

```
delayMicroseconds(delayM);
//*********************************
//Column 3
Clear();
clock();

onefourseven();

delayMicroseconds(delayM);
//*********************************
//Column 4
Clear();
clock();

Row1();
Row7();

delayMicroseconds(delayM);
//*********************************
//Column 5
Clear();
clock();

Row0();


delayMicroseconds(delayM);
}
void El(){
//************************************************
Clear(); //clear row select
 clock();
//Set Rows
Row4();
Row5();
Row6();

delayMicroseconds(delayM);

//Only keep Din high for first clock
digitalWrite(pinDin,LOW);
//*********************************
//Column 2
Clear();
clock();
```

```
  threefiveseven();

  delayMicroseconds(delayM);
  //*******************************
  //Column 3
  Clear();
  clock();

  threefiveseven();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
  Clear();
  clock();

  threefiveseven();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();

  Row4();
  Row5();

  delayMicroseconds(delayM);
}
void L(){
  //*********************************************
  Clear(); //clear row select
   clock();
  //Set Rows
  Row0();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  Row7();
```

```
  delayMicroseconds(delayM);
  //*******************************
  //Column 3
  Clear();
  clock();

  fullColumn();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
  Clear();
  clock();

  Row7();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();

  Row0();

  delayMicroseconds(delayM);
}
void O(){
  //*********************************************
  Clear(); //clear row select
   clock();
  //Set Rows

  Row4();
  Row5();
  Row6();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  Row3();
  Row7();
```

```
 delayMicroseconds(delayM);
 //*******************************
 //Column 3
 Clear();
 clock();

 Row3();
 Row7();

 delayMicroseconds(delayM);
 //*******************************
 //Column 4
 Clear();
 clock();

 Row3();
 Row7();

 delayMicroseconds(delayM);
 //*******************************
 //Column 5
 Clear();
 clock();


 Row4();
 Row5();
 Row6();

 delayMicroseconds(delayM);
}
void space(){
 //*******************************
 //Column 11
 Clear();
 clock();
 Row0();
 delayMicroseconds(delayM);
 //*******************************
 //Column 12
 Clear();
 clock();
 Row0();
 delayMicroseconds(delayM);
 //*******************************
 //Column 13
```

```
  Clear();
  clock();
  Row0();
  delayMicroseconds(delayM);
  //*******************************
  //Column 14
  Clear();
  clock();
  Row0();
  delayMicroseconds(delayM);
  //*******************************
  //Column 15
  Clear();
  clock();
  Row0();
  delayMicroseconds(delayM);
}
void P(){
  Clear(); //clear row select
   clock();

  lfullColumn();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  Row3();
  Row5();

  delayMicroseconds(delayM);
  //*******************************
  //Column 3
  Clear();
  clock();

  Row3();
  Row5();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
```

```
  Clear();
  clock();

  Row3();
  Row5();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();


  Row4();

  delayMicroseconds(delayM);
}
void T(){
  //*********************************************
  Clear(); //clear row select
   clock();
  //Set Rows
  Row1();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  Row1();

  delayMicroseconds(delayM);
  //*******************************
  //Column 3
  Clear();
  clock();

  fullColumn();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
  Clear();
```

```
  clock();

  Row1();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();

  Row1();

  delayMicroseconds(delayM);
}
void N(){
  //***********************************************
  Clear(); //clear row select
   clock();
  //Set Rows
  IfullColumn();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  Row4();

  delayMicroseconds(delayM);
  //*******************************
  //Column 3
  Clear();
  clock();

  Row3();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
  Clear();
  clock();

  Row3();
```

```
 delayMicroseconds(delayM);
 //*******************************
 //Column 5
 Clear();
 clock();

 Row4();
 Row5();
 Row6();
 Row7();

 delayMicroseconds(delayM);
}
void R(){
 //***********************************************
 Clear(); //clear row select
  clock();

 lfullColumn();

 delayMicroseconds(delayM);

 //Only keep Din high for first clock
 digitalWrite(pinDin,LOW);
 //*******************************
 //Column 2
 Clear();
 clock();

 Row4();

 delayMicroseconds(delayM);
 //*******************************
 //Column 3
 Clear();
 clock();

 Row3();


 delayMicroseconds(delayM);
 //*******************************
 //Column 4
 Clear();
 clock();
```

```
  Row3();

  delayMicroseconds(delayM);
  //*******************************
  //Column 5
  Clear();
  clock();

  Row4();

  delayMicroseconds(delayM);
}
void S(){
  //***********************************************
  Clear(); //clear row select
   clock();
  //Set Rows
  Row2();
  Row3();
  Row6();

  delayMicroseconds(delayM);

  //Only keep Din high for first clock
  digitalWrite(pinDin,LOW);
  //*******************************
  //Column 2
  Clear();
  clock();

  onefourseven();

  delayMicroseconds(delayM);
  //*******************************
  //Column 3
  Clear();
  clock();

  onefourseven();

  delayMicroseconds(delayM);
  //*******************************
  //Column 4
  Clear();
  clock();

  onefourseven();
```

```
 delayMicroseconds(delayM);
 //*****************************
 //Column 5
 Clear();
 clock();

 Row2();
 Row5();
 Row6();

 delayMicroseconds(delayM);
}
void Clear(){
 //pre clear
 digitalWrite(pinG,LOW);
 digitalWrite(pinClr,LOW);
 //clear
 digitalWrite(pinG,HIGH);
 //start memory mode
 digitalWrite(pinClr,HIGH);
}
```

Appendix. B-1 Timing Diagram

Timing Diagram

	Column 1																				
Row Select	Clear					Clock				Select Row1		Select Row2		Delay 600 µs							
S0	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0
S1	x	x	x	x	x	x	x	x	x	0	0	1	0	0	0	0	0	0	0	0	0
S2	x	x	x	x	x	x	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0
Row Decoder In																					
Clr	x	x	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G	x	0	0	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0
Shift Register In																					
Din	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sck	x	x	x	x	x	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Rck	x	x	x	x	x	x	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Row Decoder Out																					
RD0	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RD1	x	x	x	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1
RD2	x	x	x	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
RD3	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RD4	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RD5	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RD6	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RD7	x	x	x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Shift Register Out																					
SR0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
SR1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	Column 2																				
Row Select	Clear					Clock				Select Row3		Select Row5		Delay 600 µs							
S0	x	x	x	x	x	x	x	x	x	1	0	1	0	0	0	0	0	0	0	0	0
S1	x	x	x	x	x	x	x	x	x	1	0	0	0	0	0	0	0	0	0	0	0
S2	x	x	x	x	x	x	x	x	x	0	0	1	0	0	0	0	0	0	0	0	0
Row Decoder In																					
Clr	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
G	1	0	0	1	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0
Shift Register In																					
Din	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sck	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Rck	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Row Decoder Out																					
RD0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RD1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RD2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RD3	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1
RD4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RD5	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
RD6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
RD7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Shift Register Out																					
SR0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR1	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
SR2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SR9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0