SCROLLING LED MATRIX DISPLAY

SYSTEM USING PIC

AND

DIGITAL CLOCK USING PIC

Submitted by:

KEVIN A. MENDI

DONEIL V. ROSAURO

MARK KEVIN C. BALBUENA

June 14, 2017

**SCROLLING LED MATRIX DISPLAY**

**SYSTEM USING PIC**

**BACKGROUND**

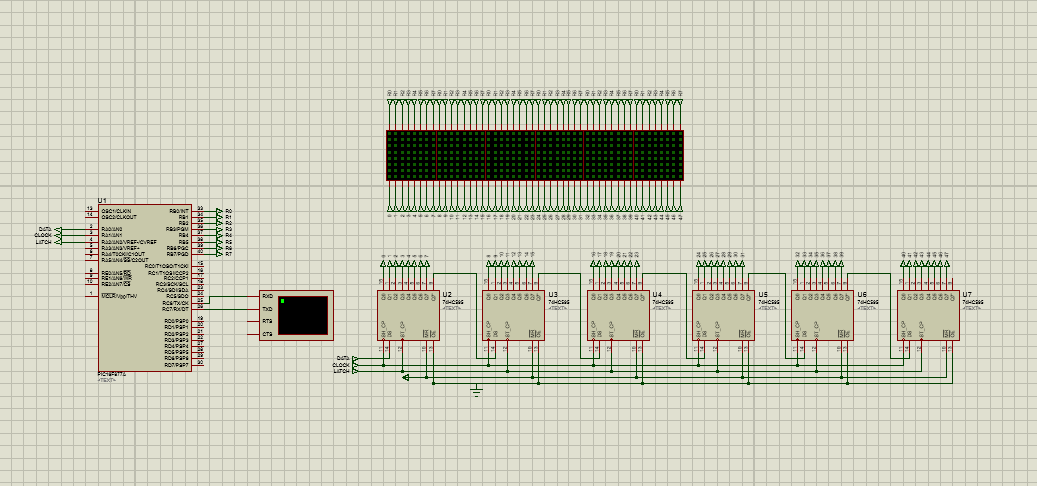
In this modern world full with vast growing technology, information is being transfer and convey into many form. Each of the information were conveyed by its specific need, like a computer server that store massive data that can be excess throughout the world in. GSM or Global System for Mobile Communication is the ability of a machine or program to receive and interpret strings, or to understand and carry out text message commands. As for this device, the temperature and chance of raining sensors will gather data every 15 minutes and the information is being visually transmitted from the dot matrix display straight to the viewers by using GSM.

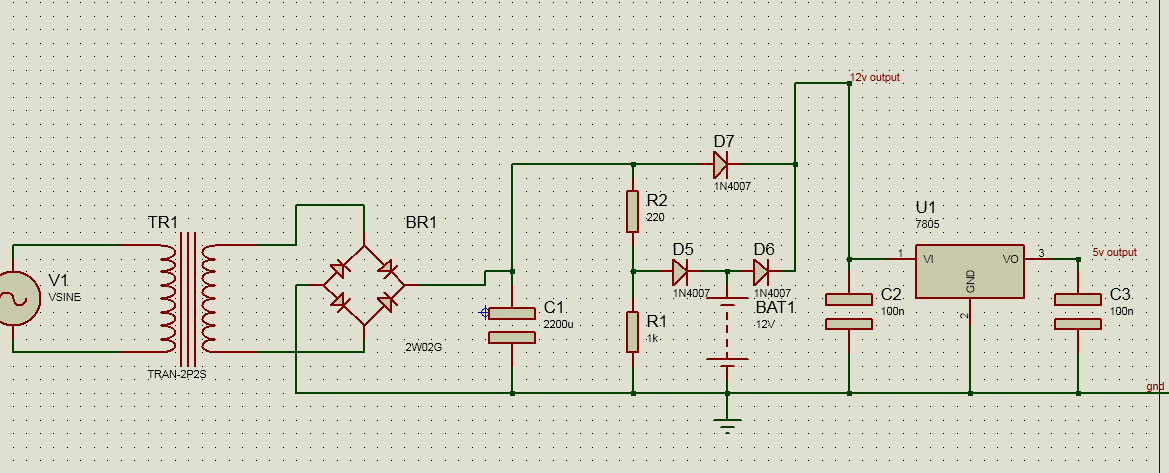
The System can easily be expanded to handle more characters. The control of this LED Matrix is based on PIC16f877a. This microcontroller will be program using mikroC. The design of the project was categorically or partly done and simulated with Proteus software which includes the following components: PIC16f877a, 74HC595, 8x8 Dot Matrix display, 20 Mhz Crystal, ULN2803 transistor, GSM module, Capacitor, 220V – 5V Transformer/power source.

**OBJECTIVES**

1. To design efficient and safe circuit for Dot Matrix Display.
2. Develop a code for microcontroller.

**CUIRCUIT DESIGN**

****

****

**CODES**

//#use delay(clock=20000000)

//#define LETTER 56

//#define LETTER 368 //448-64 4dm //456-80 5dm //464-96 6dm //472-111 7dmatrix

#define DATA PIN\_A0

#define CLOCK PIN\_A1

#define LATCH PIN\_A2

#define LETTER1 83

#define LETTER2 99

#define LETTER3 126

int i,j,k;

const char \*message = "Hello";

void f74595\_init(){

/\*

output\_bit(DATA,1);

output\_bit(CLOCK,1);

output\_bit(DATA,0);

\*/

RA0\_bit = 0xFF;

RA1\_bit = 0xFF;

RA0\_bit = 0x00;

}

void f74595\_clock(){

/\*

output\_bit(CLOCK,1);

output\_bit(CLOCK,0);

output\_bit(LATCH,1);

output\_bit(LATCH,0);

\*/

RA1\_bit = 0xff;

RA1\_bit = 0x00;

RA2\_bit = 0xff;

RA2\_bit = 0x00;

}

/\*

const char table[LETTER+96]={ //80=5 //96=6 // 111=7

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, // //58 x 8 = 464 6dm //59x8=472 7dmatrix

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

//0,0,0,0,0,0,0,0, //

//0,0,0,0,0,0,0,0, //

0x00,0xfc,0xfe,0x12,0x12,0xfe,0xfc,0x00, //A

0x00,0xfe,0xfe,0x92,0x92,0xfe,0x6c,0x00, //B

0x00,0x7c,0xfe,0x82,0x82,0xc6,0x44,0x00, //C

0x00,0xfe,0xfe,0x82,0x82,0xfe,0x7c,0x00, //D

0x00,0xfe,0xfe,0x92,0x92,0x92,0x82,0x00, //E

0x00,0xfe,0xfe,0x12,0x12,0x12,0x02,0x00, //F

0x00,0x7c,0xfe,0x82,0xa2,0xe6,0x64,0x00, //G

0x00,0xfe,0xfe,0x10,0x10,0xfe,0xfe,0x00, //H

0x00,0x00,0x82,0xfe,0xfe,0x82,0x00,0x00, //I

0x00,0x60,0xe0,0x82,0xfe,0x7e,0x02,0x00, //J

0x00,0xfe,0xfe,0x38,0x6c,0xc6,0x82,0x00, // K

0x00,0xfe,0xfe,0x80,0x80,0x80,0x80,0x00, // L

0x00,0xfe,0xfe,0x0c,0x18,0x0c,0xfe,0xfe, // M

0x00,0xfe,0xfe,0x0c,0x18,0x30,0xfe,0xfe, //N

0x00,0x7c,0xfe,0x82,0x82,0xfe,0x7c,0x00, //O

0x00,0xfe,0xfe,0x22,0x22,0x3e,0x1c,0x00, //P

0x00,0x3c,0x7e,0x42,0x62,0xfe,0xbc,0x00, //Q

0x00,0xfe,0xfe,0x32,0x72,0xde,0x8c,0x00, //R

0x00,0x4c,0xde,0x92,0x92,0xf6,0x64,0x00, //S

0x00,0x06,0x02,0xfe,0xfe,0x02,0x06,0x00, //T

0x00,0x7e,0xfe,0x80,0x80,0xfe,0xfe,0x00, //U

0x00,0x3e,0x7e,0xc0,0xc0,0x7e,0x3e,0x00, //V

0x00,0xfe,0xfe,0x60,0x30,0x60,0xfe,0xfe, //W

0x00,0xc6,0xee,0x38,0x10,0x38,0xee,0xc6, //X

0x00,0x0e,0x1e,0xf0,0xf0,0x1e,0x0e,0x00, //Y

0x00,0xc2,0xe2,0xb2,0x9a,0x8e,0x86,0x00, //Z

0x00,0x7c,0xfe,0x92,0x8a,0xfe,0x7c,0x00, //0

0x00,0x80,0x88,0xfe,0xfe,0x80,0x80,0x00, //1

0x00,0xc4,0xe6,0xa2,0x92,0x9e,0x8c,0x00, //2

0x00,0x44,0xc6,0x92,0x92,0xfe,0x6c,0x00, //3

0x00,0x30,0x28,0x24,0xfe,0xfe,0x20,0x00, //4

0x00,0x4e,0xce,0x8a,0x8a,0xfa,0x72,0x00, //5

0x00,0x7c,0xfe,0x92,0x92,0xf6,0x64,0x00, //6

0x00,0x06,0x06,0xe2,0xfa,0x1e,0x06,0x00, //7

0x00,0x6c,0xfe,0x92,0x92,0xfe,0x6c,0x00, //8

0x00,0x4c,0xde,0x92,0x92,0xfe,0x7c,0x00, //9

0x00,0x00,0x00,0x6c,0x6c,0x00,0x00,0x00, //:

0x00,0x00,0x80,0xec,0x6c,0x00,0x00,0x00,//;

0x08,0x1c,0x36,0x63,0x41,0x00,0x00,0x00, //<

0x00,0x24,0x24,0x24,0x24,0x24,0x24,0x00, //=

0x00,0x41,0x63,0x36,0x1c,0x08,0x00,0x00, // >

0x02,0x03,0x51,0x59,0x0f,0x06,0x00,0x00, //?

0x08,0x0c,0x06,0x03,0x06,0x0c,0x08,0x00, //^

0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80, //\_

0x00,0x00,0x06,0x5f,0x5f,0x06,0x00,0x00, //!

0x30,0x7a,0x4f,0x5d,0x37,0x7a,0x48,0x00, //&

0x46,0x66,0x30,0x18,0x0c,0x66,0x62,0x00, //%

0x00,0x00,0x0e,0x1b,0x1b,0x0e,0x00,0x00, //degree

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0 //8 espacios

};

\*/

///////////////////message 1//////////

int const table1[LETTER1+96]={

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, // //58 x 8 = 464 6dm //59x8=472 7dmatrix

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0x00,0x7c,0xfe,0x82,0xa2,0xe6,0x64,0x00, //G

0x00,0x7c,0xfe,0x82,0x82,0xfe,0x7c,0x00, //O

0x00,0x7c,0xfe,0x82,0x82,0xfe,0x7c,0x00, //O

0x00,0xfe,0xfe,0x82,0x82,0xfe,0x7c,0x00, //D

0,0,0,

0x00,0xfe,0xfe,0x0c,0x18,0x0c,0xfe,0xfe, // M

0x00,0x7c,0xfe,0x82,0x82,0xfe,0x7c,0x00, //O

0x00,0xfe,0xfe,0x32,0x72,0xde,0x8c,0x00, //R

0x00,0xfe,0xfe,0x0c,0x18,0x30,0xfe,0xfe, //N

0x00,0x00,0x82,0xfe,0xfe,0x82,0x00,0x00, //I

0x00,0xfe,0xfe,0x0c,0x18,0x30,0xfe,0xfe, //N

0x00,0x7c,0xfe,0x82,0xa2,0xe6,0x64,0x00, //G

0x00,0x00,0x06,0x5f,0x5f,0x06,0x00,0x00, //!

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0 //8 espacios

};

///////////////////////////message 2 ////////////////

int const table2[LETTER2+96]={

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, // //58 x 8 = 464 6dm //59x8=472 7dmatrix

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0x00,0xfe,0xfe,0x10,0x10,0xfe,0xfe,0x00, //H

0x00,0xfc,0xfe,0x12,0x12,0xfe,0xfc,0x00, //A

0x00,0xfe,0xfe,0x22,0x22,0x3e,0x1c,0x00, //P

0x00,0xfe,0xfe,0x22,0x22,0x3e,0x1c,0x00, //P

0x00,0x0e,0x1e,0xf0,0xf0,0x1e,0x0e,0x00, //Y

0,0,0,

0x00,0xfe,0xfe,0x10,0x10,0xfe,0xfe,0x00, //H

0x00,0x7c,0xfe,0x82,0x82,0xfe,0x7c,0x00, //O

0x00,0xfe,0xfe,0x80,0x80,0x80,0x80,0x00, // L

0x00,0xfe,0xfe,0x80,0x80,0x80,0x80,0x00, // L

0x00,0x00,0x82,0xfe,0xfe,0x82,0x00,0x00, //I

0x00,0xfe,0xfe,0x82,0x82,0xfe,0x7c,0x00, //D

0x00,0xfc,0xfe,0x12,0x12,0xfe,0xfc,0x00, //A

0x00,0x0e,0x1e,0xf0,0xf0,0x1e,0x0e,0x00, //Y

0x00,0x00,0x06,0x5f,0x5f,0x06,0x00,0x00, //!

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0 //8 espacios

};

//////////////////////////message 3///////////////

int const table3[LETTER3+96]={

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, // //58 x 8 = 464 6dm //59x8=472 7dmatrix

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0,0,0,0,0,0,0,0, //

0x00,0xfe,0xfe,0x38,0x6c,0xc6,0x82,0x00, // K

0x00,0xfe,0xfe,0x92,0x92,0x92,0x82,0x00, //E

0x00,0xfe,0xfe,0x92,0x92,0x92,0x82,0x00, //E

0x00,0xfe,0xfe,0x22,0x22,0x3e,0x1c,0x00, //P

0,0,0,

0x00,0x4c,0xde,0x92,0x92,0xf6,0x64,0x00, //S

0x00,0xfc,0xfe,0x12,0x12,0xfe,0xfc,0x00, //A

0x00,0xfe,0xfe,0x12,0x12,0x12,0x02,0x00, //F

0x00,0xfe,0xfe,0x92,0x92,0x92,0x82,0x00, //E

0,0,0,

0x00,0xfe,0xfe,0x92,0x92,0x92,0x82,0x00, //E

0x00,0x3e,0x7e,0xc0,0xc0,0x7e,0x3e,0x00, //V

0x00,0xfe,0xfe,0x92,0x92,0x92,0x82,0x00, //E

0x00,0xfe,0xfe,0x32,0x72,0xde,0x8c,0x00, //R

0x00,0x0e,0x1e,0xf0,0xf0,0x1e,0x0e,0x00, //Y

0x00,0x7c,0xfe,0x82,0x82,0xfe,0x7c,0x00, //O

0x00,0xfe,0xfe,0x0c,0x18,0x30,0xfe,0xfe, //N

0x00,0xfe,0xfe,0x92,0x92,0x92,0x82,0x00, //E

0x00,0x00,0x06,0x5f,0x5f,0x06,0x00,0x00, //!

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0, //8 espacios

0,0,0,0,0,0,0,0 //8 espacios

};

char mess;

char menu;

int count=0;

int length=5;

int x=0,y=0,lo=0;

int LETTER=0;

int scroll(int let)

{

LETTER = let;

for(i=0;i<LETTER+48;i++){ //ALPHA column //48=6dm

for(k=0;k<10;k++){ //SPEED

f74595\_init();

for(j=0;j<48;j++){ //Column

f74595\_clock();

//output\_b(~tabla[j+i]);

if(LETTER == 83)

{

PORTB=(~table1[j+i]); //display

///delay\_us(615);

delay\_us(497); //680=5 //615=6dm

}

else if (LETTER == 99)

{

PORTB=(~table2[j+i]); //display

///delay\_us(615);

delay\_us(500);

}

else if (LETTER == 126)

{

PORTB=(~table3[j+i]); //display

///delay\_us(615);

delay\_us(500);

}

}

}

}

return LETTER;

}

void main(){

UART1\_Init(9600);

Lcd\_Init();

TRISB = 0;

TRISA = 0;

ADCON1=6;

PORTA=0;

PORTB=0;

TRISD =0;

PORTD = 0;

i=0;j=0;k=0;

Lcd\_Cmd(\_LCD\_CLEAR); // Clear display

Lcd\_Cmd(\_LCD\_CURSOR\_OFF); // Cursor off

start:

UART1\_Write\_Text("--Menu--");

UART1\_Write(13);

UART1\_Write\_Text("(1) Good Morning !");

UART1\_Write(13);

UART1\_Write\_Text("(2) Happy Holiday !");

UART1\_Write(13);

UART1\_Write\_Text("(3) Keep Safe Everyone !");

UART1\_Write(13);

UART1\_Write\_Text("Enter your choice: ");

while(1)

{

if (UART1\_Data\_Ready())

{

menu = UART1\_Read(); //detecting keypress

if(menu == '1')

{

UART1\_Write(13);

UART1\_Write\_Text("Good Morning !"); // Virtual Terminal Reply

UART1\_Write(13);

scroll(83);

goto start;

}

else if(menu == '2')

{

UART1\_Write(13);

UART1\_Write\_Text("Happy Holiday !"); // Virtual Terminal Reply

UART1\_Write(13);

scroll(99);

goto start;

}

else if(menu == '3')

{

UART1\_Write(13);

UART1\_Write\_Text("Keep Safe Everyone !"); // Virtual Terminal Reply

UART1\_Write(13);

scroll(126);

goto start;

}

} //if

} //while

} //main

**BLOCK DIAGRAM**



MR. RECIEVER

Hello ! this is Temperature and Chance of Raining Sensor

THE ARTIST

Oh Gosh ! It’s me

Dot Matrix

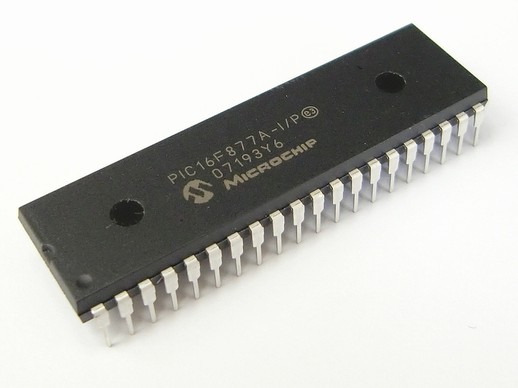
MR. BUILDER

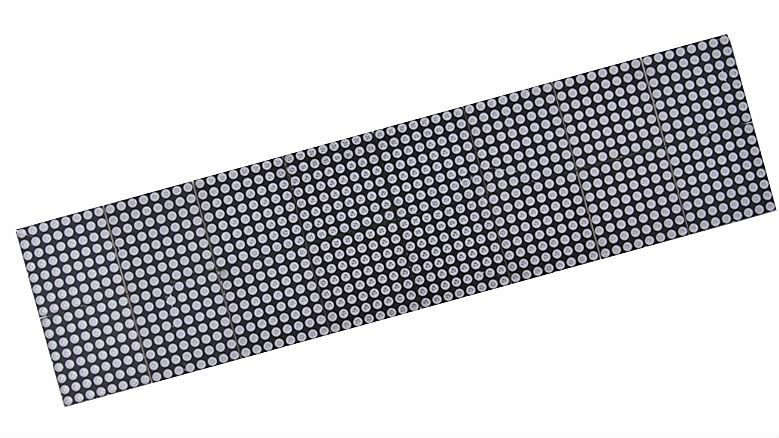
Here we go again..

I’m PIC16f877a

MR. SENDER

Oh Hi ! .. My Name is GSM





**Proposed Budget**

1. - 16f877a Microcontroller 309.71 each Php.309.71

12 - 74HC595 20 each Php.240

12 - Led Dot Matrix 5mm 8x8 150 each Php.1800

1 - Crystal 20Mhz 11 each Php.11

2 - Capacitor 22nf 5 each Php.10

1 - Uln2803 25 each Php.25

1 - GSM module 1500 each Php.1500

1 - 220v to 5v power source 250 each Php.250

**Total: Php.4145.71**

**DIGITAL CLOCK**

**Software**

1. Programming language

-Micro C

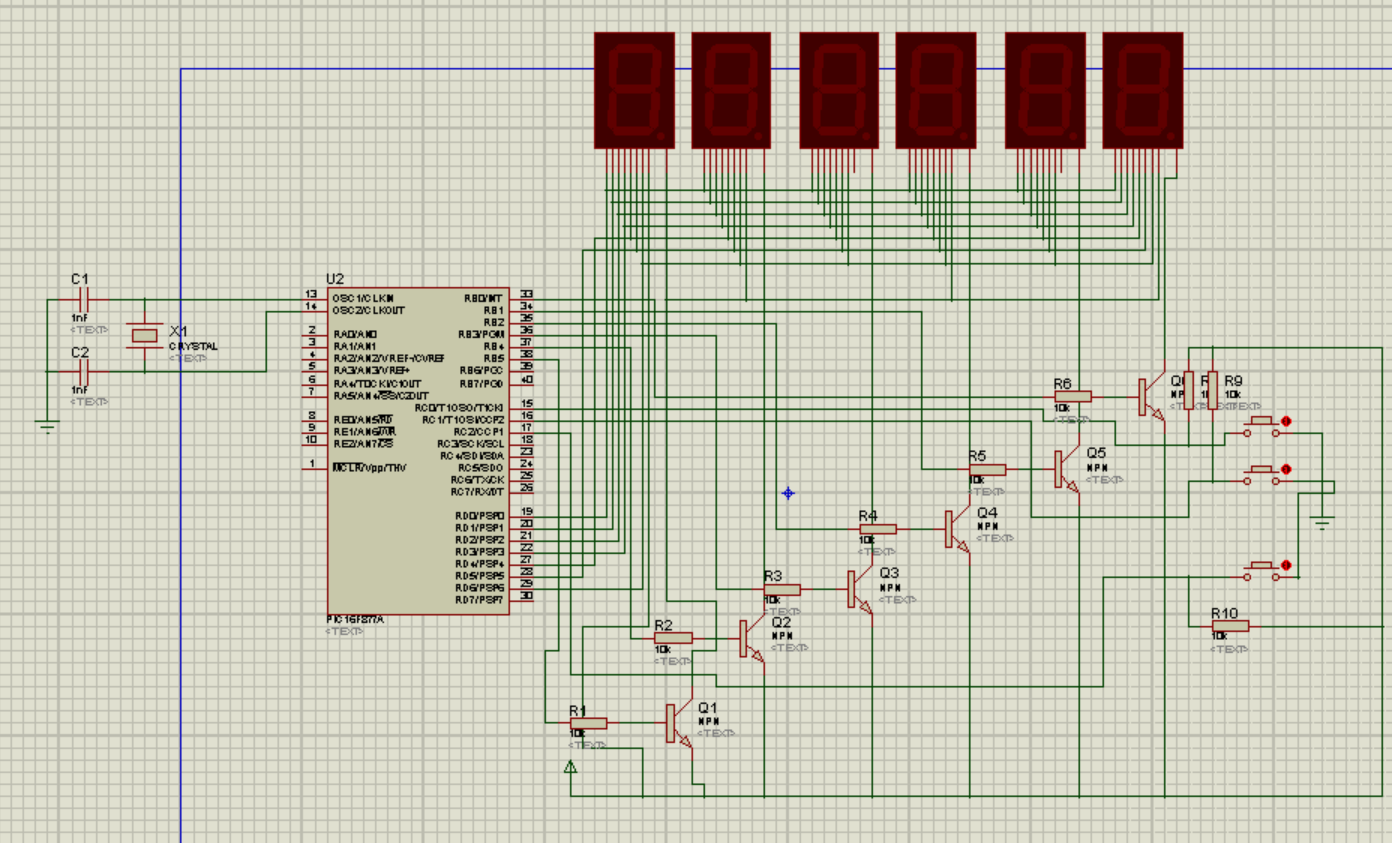
2. Simulation Software

-Proteus

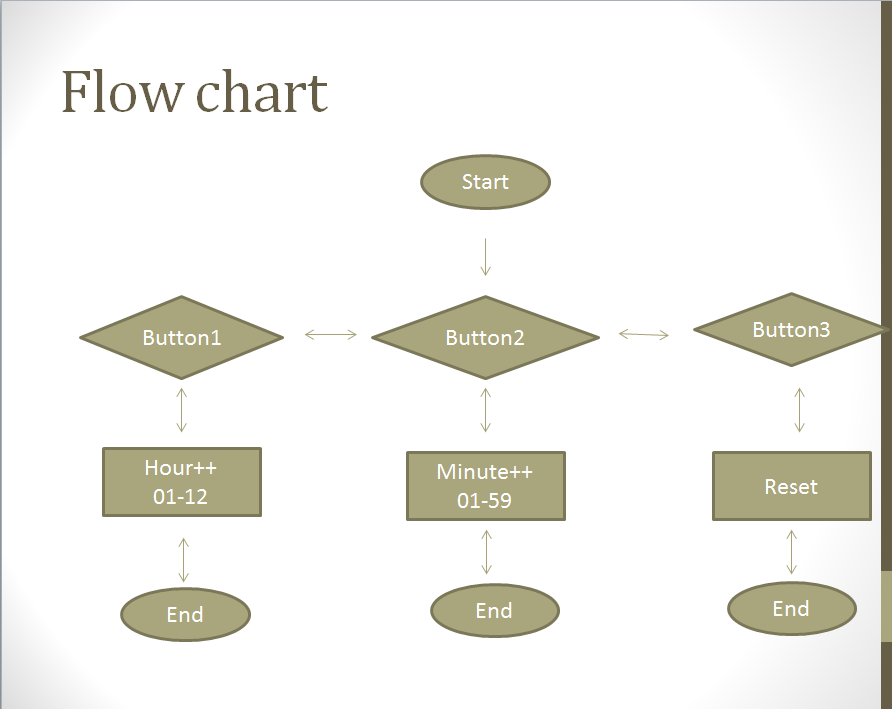
**Components**

* Pic16f877a
* 7 segments
* Transistor
* Resistor
* Button
* Capacitor
* Crystal oscillator

**CIRCUIT DIAGRAM**

****

**FLOW CHART**

****

**CODE**

unsigned short mask(unsigned short num);

unsigned char x=0,s1=0,s2=0,m1=0,m2=0,h1=0,h2=0;

unsigned int y=0;

void interrupt(){

if (x==0){portb=1;portd=mask(s1);}

if (x==1){portb=2;portd=mask(s2);}

if (x==2){portb=4;portd=mask(m1);}

if (x==3){portb=8;portd=mask(m2);}

if (x==4){portb=16;portd=mask(h1);}

if (x==5){portb=32;portd=mask(h2);}

x++;

y++;

if(y==401)

{

s1++;

y=0;

}

if(x==6) x=0;

INTCON =0b00100000;

TMR0=217;

}

void main() {

OPTION\_REG=0b010000101;

INTCON= 0b10100000;

TMR0=217;

trisb=0;

portb=0;

trisd=0;

portd=0;

trisc=255;

for(;;){

if (s1==10){s2++;s1=0;}

if (s2==6){m1++;s2=0;}

if (m1==10){m2++;m1=0;}

if (m2==6){h1++;m2=0;}

if (h1==10){h2++;h1=0;}

if ((h2==1)&&(h1==3)){s1=0;s2=0;m1=0;m2=0;h1=1;h2=0;}

if (rc2\_bit==0){

s1=0;s2=0;m1=0;m2=0;h1=0;h2=0;

for(;rc2\_bit==0;){}

}

if (rc0\_bit==0){

m1++;

for(;rc0\_bit==0;){}

}

if(rc1\_bit==0) {

h1++;

for(;rc1\_bit==0;) {}

}

}

}

unsigned short mask (unsigned short num){

switch(num){

case 0:return 0xC0;

case 1:return 0xF9;

case 2:return 0xA4;

case 3:return 0xB0;

case 4:return 0x99;

case 5:return 0x92;

case 6:return 0x82;

case 7:return 0xf8;

case 8:return 0x80;

case 9:return 0x90;

}

}

**Proposed Budget**

1. - Pic16f877a Php 270
2. - 7 segments Php 150
3. - Transistor Php 18
4. - Resistor Php 10
5. - Button Php 15
6. - Capacitor Php 6
7. - Crystal oscillator Php 20

**Php 489**