
بسم الله الرحمن الرحيم

گزارش تکمیلی پایان نامه ی

پروژه ی اسیلوسکوپ

افراد پروژه :

علی ناعمی رضا رنجبر

حسین عادلی محمود مردی

فهرست

تشریح عملکرد پایه های میکرو اصلی :

در میکروی اصلی پورت ها به شکل زیر عمل می کنند :

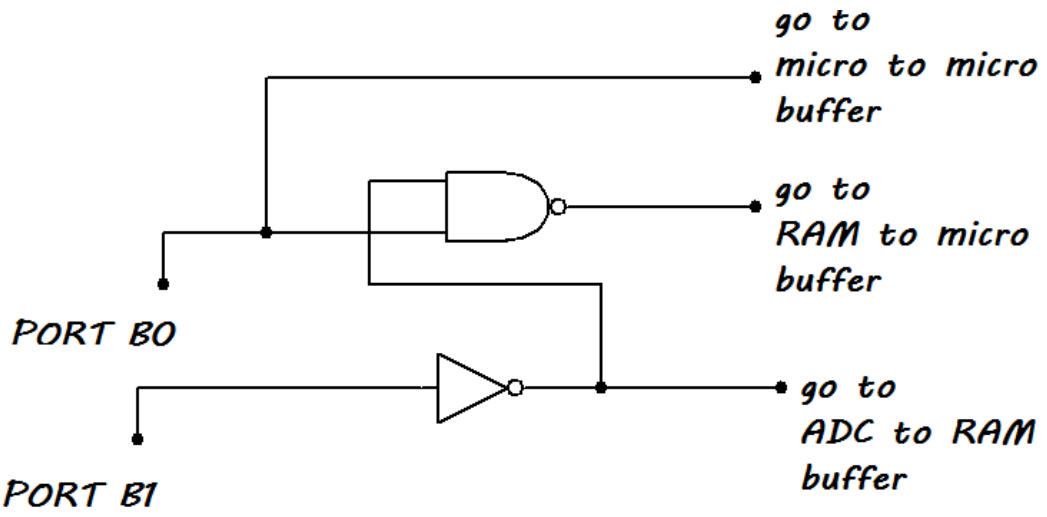
پورت A : به عنوان ورودی استفاده شده و دو کد را دریافت می کند:

۱) کد رنج های ولتاژ و زمان .

۲) کد خروجی ADC (که همان موج ورودی است که به صورت کد دیجیتالی در آمده) .

پورت B : این پورت پر کار ترین پورت میکرو است که فعالیت های کنترلی ، در یافت فرمان شروع نمونه گیری و در یافت اینتراپت را به عهده دارد .

پین های B.0 و B.1 : این دو پین به کمک مدار زیر سه بافر مدار را کنترل می کنند .

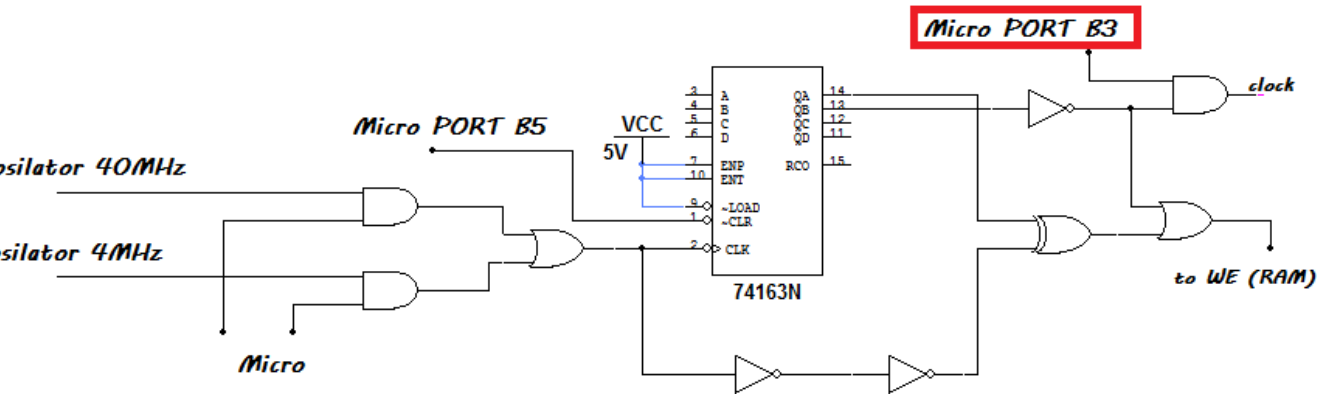


جدول عملکرد بافر ها به شرح زیر است .

بافر فعال	پورت B.0	پورتB.1
میکرو به میکرو	0	0
رم به میکرو	1	0
میکرو به میکرو و ADC به رم	0	1
ADC به RAM	1	1

پین B.2 : این پین به عنوان اینتراپت به کار رفته و برای ۴ کلید ورودی و پایه ی QD آخرین کانتر رم عمل می کند که در حالت عادی ۱ و در لبه ی پایین رونده میکرو وارد اینتراپت می شود.

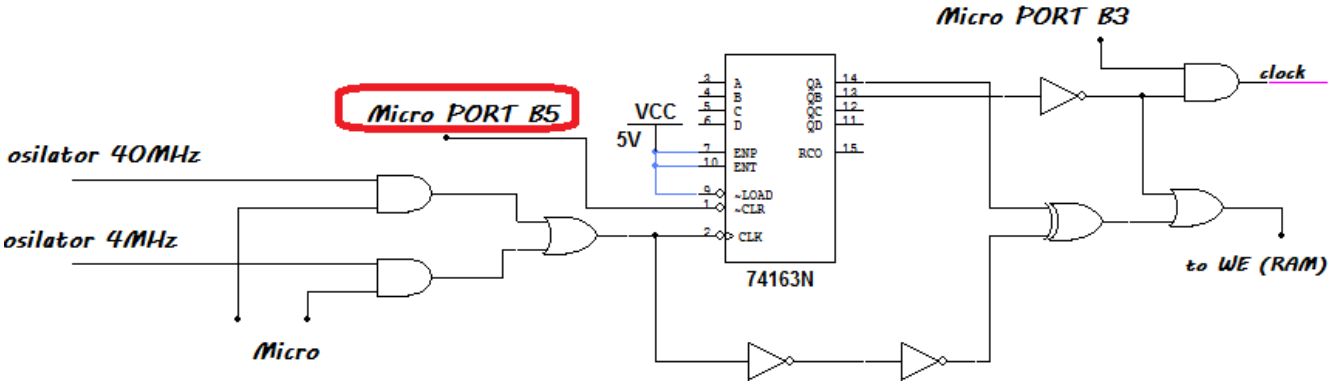
پین B.3 : این پایه در حالت نمونه برداری یک است و به قسمت آماده ساز کلاک رفته که در مدار زیر مشهود است . اما در حالتی که میکرو می خواهد موج را رسم کند (زمانی که میکرو شروع به خواندن رم و رسم موج روی LCD می کند) این پایه کلاک CE رم و ۷۴۱۶۳ ها را که آدرس دهی رم را به عهده دارند تأمین می کند . موج خروجی این پایه در زیر آمده است .



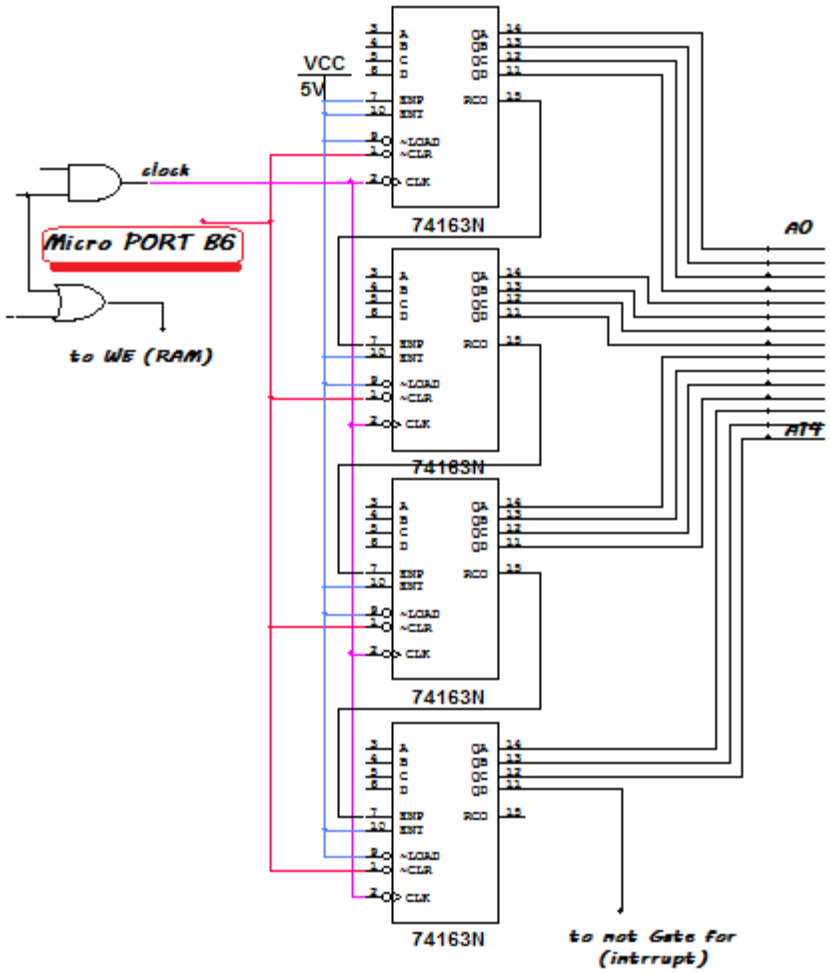
پین B.4 : این پین که به OE وصل است در حالت نمونه گیری یک است چون که نیازی به فعال کردن OE رم نیست ولی در حالت خواندن شکل موج از رم این پایه با پایه ی قبلی که بیان گردید نقش آماده ساز موج ورودی را ایفا می کنند . که شکل آن در زیر آمده است .

(شکل سیاه مربوط به پین B.4 و شکل صورتی مربوط به پین B.3 است.)

پین B.5 : این پین به clear اولین 74163 رفته است که در هنگام نمونه گیری ۱ و در هنگام نمونه برداری و رسم موج فعال (صفر) می باشد ، تا کلاک ها را میکرو بسازد .



پین B.6 : این پین نیز به clear کانتر های مرتبط با آدرس دهی رم رفته تا در زمانی که نیاز به صفر کردن آدرس رم است ، این پایه فعال (صفر) شود .



پین B.7 : این پین به عنوان ورودی است ، که با فشار کلید متصل به این پین نمونه گیری از ورودی آغاز می گردد .

پورت C : این پورت به صورت کامل در اختیار LCD می باشد .

پورت D : ۵ بیت کم ارزش این پورت نیز متصل به LCD گرافیکی می باشد ولی ۳ بیت پر ارزش آن به عنوان ورودی استفاده شده و به کلید های فشاری متصل اند . این سه کلید برای بزرگنمایی و کوچک نمایی و ریست بزرگنمایی و کوچک نمایی به حالت اولیه مورد استفاده واقع شده اند .

طریقه بزرگنمایی به این شکل است که ما بجای خواندن ۱۰۰ آدرس پشت سر هم ، با فشردن کلید مربوطه ۲۰۰ بیت را یک در میان می خوانیم و اگر بار دیگر کلید بزرگ نمایی فشرده شود ، ۴۰۰ بیت را به شکل ۱بیت به ازای ۴ بیت می خوانیم و این روند تا ۳۰۰۰۰ بیت به شکل ۱بیت به ازای ۳۰۰ بیت ادامه دارد و کلید کوچک نمایی برعکس کلید بزرگنمایی عمل می کند و کلید ریست هم ۱۰۰ بیت پشت سر هم را برای نمونه برداری انتخاب می کند .

برنامه میکرو :

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.03.5 Evaluation
Automatic Program Generator
© Copyright 1998-2008 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project : oscop
Version :
Date    : 5/15/2010
Author  : Freeware, for evaluation and non-commercial use only
Company :
```

```
Comments:



Chip type           : ATmega32
Program type        : Application
Clock frequency     : 1.000000 MHz
Memory model        : Small
External RAM size   : 0
Data Stack size     : 512
*****************************************************/
#include <mega32.h>
#include <glcd.h>
#include <delay.h>
#define s1 PORTB.0 .
#define s2 PORTB.1
#define s4 PORTB.3
#define s5 PORTB.4
#define s6 PORTB.5
#define s7 PORTB.6
#define s8 PINB.7
#define zo PIND.7
#define zi PIND.6
#define zr PIND.5
unsigned char f,top2,b,c,ch,voptop,in[100];
unsigned int top1,a,aa,zn=1;
bit n=1;
unsigned char a11,a1,a2=0,d,soton=27,i,j;
// External Interrupt 2 service routine
interrupt [EXT_INT2] void ext_int2_isr(void)
{
if(s8==0&&ch==0){ch=1; s1=0; s2=1; s7=0; s5=1; s4=1;
```

```
delay_us(10); s7=1; s6=1; goto end;}
if(zo==0){
if(zn<128){zn*=2; delay_us(10);}
if(zn>=128){zn+=50; if(zn>=300){zn=300;} delay_us(10);}
goto tanzim;}
if(zi==0){
if(zn>128){if(zn==300){zn=328;} zn-=50; delay_us(10);}
if(zn<=128){if(zn==1){zn=1; delay_us(10); goto tanzim;}
 zn/=2; delay_us(10);}
goto tanzim;}
if(zr==0){zn=1; goto tanzim;}
delay_us(50);
tanzim:
if(s8!=0||ch==1){
s7=0; s6=0; s2=0; s1=0; s5=1; s4=1; delay_us(50);
voptop=PINA; delay_us(50);
s1=1; s2=1;
s6=0; s7=0; s4=1; s5=1; delay_us(4); s4=0; delay_us(4);
s4=1; s7=1;
for(c=0;c<=100;c++){
s4=1; delay_us(5); s4=0; delay_us(5);}
s4=1; b=0; aa=0;
for(b=0;b<100;b++){  if(b==50){s6=0; s7=0; s4=1; s5=1;
delay_us(4); s4=0; delay_us(4); s4=1; s7=1;}
for(a=0;a<=zn;a++){
s4=1; delay_us(2); s4=0; delay_us(2); }
s5=0; delay_us(2); in[b]=PINA; delay_us(2); s5=1;
delay_us(2); s4=1; delay_us(2);}
}
end:
}
```

```
flash char name[]={
0xFC,0x06,0x03,0x01,0x01,0x81,0xC1,0x41,0x41,0x41,
0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,
0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,0xC1,0x41,
0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,
0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,0x41,
0x41,0xC1,0x81,0x01,0x01,0x01,0x01,0xE1,0x11,0x11,
0x21,0xFD,0x01,0xE1,0x51,0x51,0x51,0x61,0x01,0x21,
0x51,0x51,0x91,0x21,0x01,0xF5,0x01,0xE1,0x11,0x11,
0x21,0xF1,0x01,0xF1,0x21,0x11,0x11,0xE1,0x01,0xE1,
0x51,0x51,0x51,0x61,0x01,0xF1,0x21,0x11,0x01,0x01,
0x11,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,
0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,
0x01,0x01,0x01,0x01,0x01,0x03,0x06,0xFC,
0xFF,0x00,0x00,0x00,0x00,0xFF,0x00,0x00,0x00,0x00,
0x84,0x00,0x00,0x00,0x80,0x40,0xA4,0x10,0x08,0x08,
0x08,0x10,0x20,0xC4,0x80,0x00,0x00,0x00,0xFF,0x00,
0x00,0x00,0x00,0x84,0x00,0x00,0x00,0x00,0x00,0x00,
0x84,0x00,0x00,0x00,0x00,0x00,0x00,0x84,0x00,0x00,
0x00,0x00,0xFF,0x00,0x00,0x00,0x00,0x01,0x02,0x02,
0x01,0x03,0x00,0x01,0x02,0x02,0x02,0x01,0x00,0x01,
0x02,0x02,0x02,0x01,0x00,0x03,0x00,0x09,0x0A,0x0A,
0x09,0x07,0xC0,0x43,0x40,0x40,0x40,0x83,0x00,0x01,
0x02,0x02,0x02,0x01,0x00,0x03,0x00,0x00,0x00,0x00,
0x02,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x40,0x00,
0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,
0xFF,0x00,0x00,0x00,0x00,0xFF,0x08,0x08,0x08,0x08,
0x88,0x0C,0x0A,0x09,0x08,0x08,0x88,0x08,0x08,0x08,
0x08,0x08,0x08,0x88,0x08,0x09,0x0A,0x04,0xEB,0x10,
0x28,0x48,0x88,0x88,0x08,0x08,0x08,0x08,0x08,0x08,
```

0x88,0x08,0x88,0x48,0x28,0x18,0x08,0x88,0x08,0x08,

0x08,0x08,0xFF,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x3F,0x04,0x04,0x04,0x0C,0x13,0x20,0x00,

0x20,0x00,0x3F,0x02,0x01,0x00,0x1A,0x25,0x25,0x15,

0x3E,0x00,0x3F,0x02,0x01,0x01,0x3E,0x80,0x7F,0x00,

0x3F,0x12,0x21,0x21,0x1E,0x00,0x1A,0x25,0x25,0x15,

0x3E,0x00,0x3F,0x02,0x01,0x00,0x00,0xFF,

0xFF,0x00,0x00,0x00,0x00,0x7F,0xC0,0x80,0x80,0x80,

0x90,0x80,0x80,0x80,0x80,0x80,0x90,0x80,0x80,0x80,

0x80,0x80,0x80,0x90,0x80,0x80,0x80,0x80,0xFF,0x80,

0x80,0x80,0x80,0x91,0x82,0x84,0x88,0x88,0x88,0x84,

0x92,0x81,0x80,0x80,0x80,0x80,0x80,0x90,0x80,0x80,

0x80,0xC0,0x7F,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x80,0x60,0x5C,0x42,0x5C,0x60,0x80,0x00,

0x00,0x00,0xF8,0x10,0x08,0x08,0xF0,0x00,0xD0,0x28,

0x28,0xA8,0xF0,0x00,0xF0,0x28,0x28,0x28,0xB0,0x00,

0xF8,0x08,0x08,0xF0,0x08,0x08,0xF0,0x00,0xFA,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,

0xFF,0x00,0x00,0x00,0xC0,0xC0,0x40,0x40,0xC0,0x80,

0x00,0x00,0x00,0x00,0x00,0x00,0x40,0x40,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x40,0x40,0x00,0x00,0x00,0x00,

0x00,0xC0,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0xC0,0xC0,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0xF9,0x40,0x40,0x40,0x40,0xF8,0x01,0x00,

0x01,0x00,0x41,0xA0,0xA0,0xA0,0xC1,0x00,0xC0,0x21,

0x21,0x40,0xF9,0x00,0xC0,0xA1,0xA1,0xA1,0xC0,0x00,

0xF9,0x00,0xE8,0x01,0x00,0x00,0x01,0x00,0x01,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,

0xFF,0x00,0x00,0x00,0x3F,0x3F,0x20,0x20,0x30,0x1F,

0x0F,0x00,0x00,0x00,0x00,0x00,0x3F,0x3F,0x00,0x00,

0x00,0x00,0x00,0x9E,0xBF,0xA1,0xB3,0xFF,0x7F,0x00,

0x00,0x00,0x00,0x00,0x3F,0x3F,0x00,0x00,0x00,0x00,

0x01,0x1F,0x3F,0x21,0x00,0x00,0x00,0x00,0x00,0x1A,

0x3F,0x25,0x35,0x3F,0x3E,0x00,0x00,0x00,0x00,0x00,

0x3F,0x3F,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0xE7,0xC0,0x00,0x00,0x00,0xC7,0xE0,0x00,

0x04,0x00,0x03,0x84,0x84,0x82,0x07,0x80,0x83,0x04,

0x04,0x02,0x87,0x80,0x83,0x04,0x04,0x84,0x02,0x80,

0x07,0x00,0x87,0x80,0x00,0xE0,0x00,0xA0,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,

0xFF,0x00,0x00,0x00,0xE0,0xF0,0x10,0x10,0xF0,0xE0,

0x00,0x20,0x70,0x50,0xD0,0xB0,0x20,0x00,0xE0,0xF0,

0x10,0x10,0x30,0x20,0x00,0xF4,0xF4,0x00,0xFC,0xFC,

0x00,0xFC,0xFC,0x00,0xE0,0xF0,0x10,0x10,0xF0,0xE0,

0x00,0x20,0x70,0x50,0xD0,0xB0,0x20,0x00,0xE0,0xF0,

0x10,0x10,0x30,0x20,0x00,0xE0,0xF0,0x10,0x10,0xF0,

0xE0,0x00,0xF0,0xF0,0x30,0x10,0xF0,0xE0,0x00,0xE0,

0xF0,0x50,0x50,0x70,0x60,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x1F,0x00,0x07,0x18,0x07,0x00,0x1F,0x00,

0x00,0x10,0x00,0x1F,0x00,0x00,0x1F,0x00,0x00,0x1F,

0x00,0x0D,0x12,0x12,0x0A,0x1F,0x00,0x1F,0x01,0x00,

0x00,0x0F,0x10,0x10,0x09,0x1F,0x00,0x1F,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0xFF,

0x3F,0x60,0xC0,0x80,0x81,0x83,0x82,0x82,0x83,0x81,

0x80,0x81,0x83,0x82,0x82,0x83,0x81,0x80,0x81,0x83,

0x82,0x82,0x83,0x81,0x80,0x83,0x83,0x80,0x83,0x83,

0x80,0x83,0x83,0x80,0x81,0x83,0x82,0x82,0x83,0x81,

0x80,0x81,0x83,0x82,0x82,0x83,0x81,0x80,0x81,0x83,

0x82,0x82,0x83,0x81,0x80,0x81,0x83,0x82,0x82,0x83,

0x81,0x80,0x8F,0x8F,0x83,0x82,0x83,0x81,0x80,0x81,

0x83,0x82,0x82,0x83,0x81,0x80,0x80,0x80,0x80,0x80,

0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,

0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,

0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,

0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,

0x80,0x80,0x80,0x80,0x80,0xC0,0x60,0x3F

};

flash char oscop[1024]={

0x1C,0x20,0x40,0x20,0x1C,0x00,0x60,0x10,0x0C,0x00,

0x7C,0x14,0x14,0x08,0x00,0x00,0x48,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0xFE,0x03,0x01,0x01,

0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,

0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,

0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,

0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,

0x01,0x01,0x01,0x01,0x01,0x01,0x81,0xFE,0x81,0x01,

0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,

0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,

0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,

0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,0x01,

0x01,0x01,0x01,0x01,0x01,0x01,0x03,0xFE,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0x00,0x00,0x00,

0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,

0x00,0x00,0x00,0x00,0x00,0x00,0x80,0xFF,0x80,0x00,

0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,

0x00,0x00,0x00,0x00,0x01,0x00,0x00,0xFF,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0x00,0x00,0x00,

0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,

0x00,0x00,0x00,0x00,0x00,0x00,0x80,0xFF,0x80,0x00,

0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x01,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0x00,0x00,

0x00,0x00,0x00,0x00,0x01,0x00,0x00,0xFF,

0x04,0x3E,0x44,0x04,0x00,0x30,0x08,0x06,0x00,0x3E,

0x0A,0x0A,0x04,0x00,0x00,0x00,0x24,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0x80,0x80,0x80,

0xC1,0x80,0x80,0x80,0x80,0x80,0x80,0xC1,0x80,0x80,

0x80,0x80,0x80,0x80,0x80,0xC1,0x80,0x80,0x80,0x80,

0x80,0x80,0x80,0xC1,0x80,0x80,0x80,0x80,0x80,0x80,

0x80,0xC1,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0xC1,

0x80,0x80,0x80,0x80,0x80,0x80,0x00,0xBF,0x00,0x80,

0x80,0x80,0x80,0x80,0x80,0xC1,0x80,0x80,0x80,0x80,

0x80,0x80,0x80,0xC1,0x80,0x80,0x80,0x80,0x80,0x80,

0x80,0xC1,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0xC1,

0x80,0x80,0x80,0x80,0x80,0x80,0x80,0xC1,0x80,0x80,

0x80,0x80,0x80,0x80,0xC1,0x80,0x80,0x7F,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0x00,0x00,0x00,

0x81,0x00,0x00,0x00,0x00,0x00,0x00,0x81,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x81,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x81,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x81,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x81,

0x00,0x00,0x00,0x00,0x00,0x00,0x80,0xFE,0x80,0x00,

0x00,0x00,0x00,0x00,0x00,0x81,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x81,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x81,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x81,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x81,0x00,0x00,

0x00,0x00,0x00,0x00,0x81,0x00,0x00,0xFF,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0x00,0x00,0x00,

0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x80,0x00,0x00,

0x00,0x00,0x00,0x00,0x00,0x80,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x80,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,

0x00,0x00,0x00,0x00,0x00,0x00,0x80,0xFF,0x80,0x00,

0x00,0x00,0x00,0x00,0x00,0x80,0x00,0x00,0x00,0x00,

0x00,0x00,0x00,0x80,0x00,0x00,0x00,0x00,0x00,0x00,

0x00,0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,

0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,0x00,0x00,

0x00,0x00,0x00,0x00,0x80,0x00,0x00,0xFF,

```
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0xFF,0x00,0x00,0x00,
0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x80,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x80,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x80,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,
0x00,0x00,0x00,0x00,0x00,0x00,0x80,0xFF,0x80,0x00,
0x00,0x00,0x00,0x00,0x00,0x80,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x80,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x80,0x00,0x00,
0x00,0x00,0x00,0x00,0x80,0x00,0x00,0xFF,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
0x00,0x00,0x00,0x00,0x00,0x00,0x7F,0xC0,0x80,0x80,
0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,
0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,
0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,
0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,
0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x7F,0x80,0x80,
0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,
0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,
0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,
0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,0x80,
0x80,0x80,0x80,0x80,0x80,0x80,0xC0,0x7F

};
void t_um()
{
 write_char(13,1,86);  // write 'v'
```

```
 if(f==1)
  {
    write_char(0,5,97);//write u
        write_char(6,5,83); // write s
  }
 if(f==2){
  {
    write_char(0,5,77);//write m
    write_char(6,5,83); // write s
  }



}
// Declare your global variables here

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=In Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=T State1=0 State0=0
PORTB=0x00;
DDRB=0x7B;
```

```
// Port C initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTC=0x00;
DDRC=0xFF;

// Port D initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTD=0xe0;
DDRD=0x1F;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
```

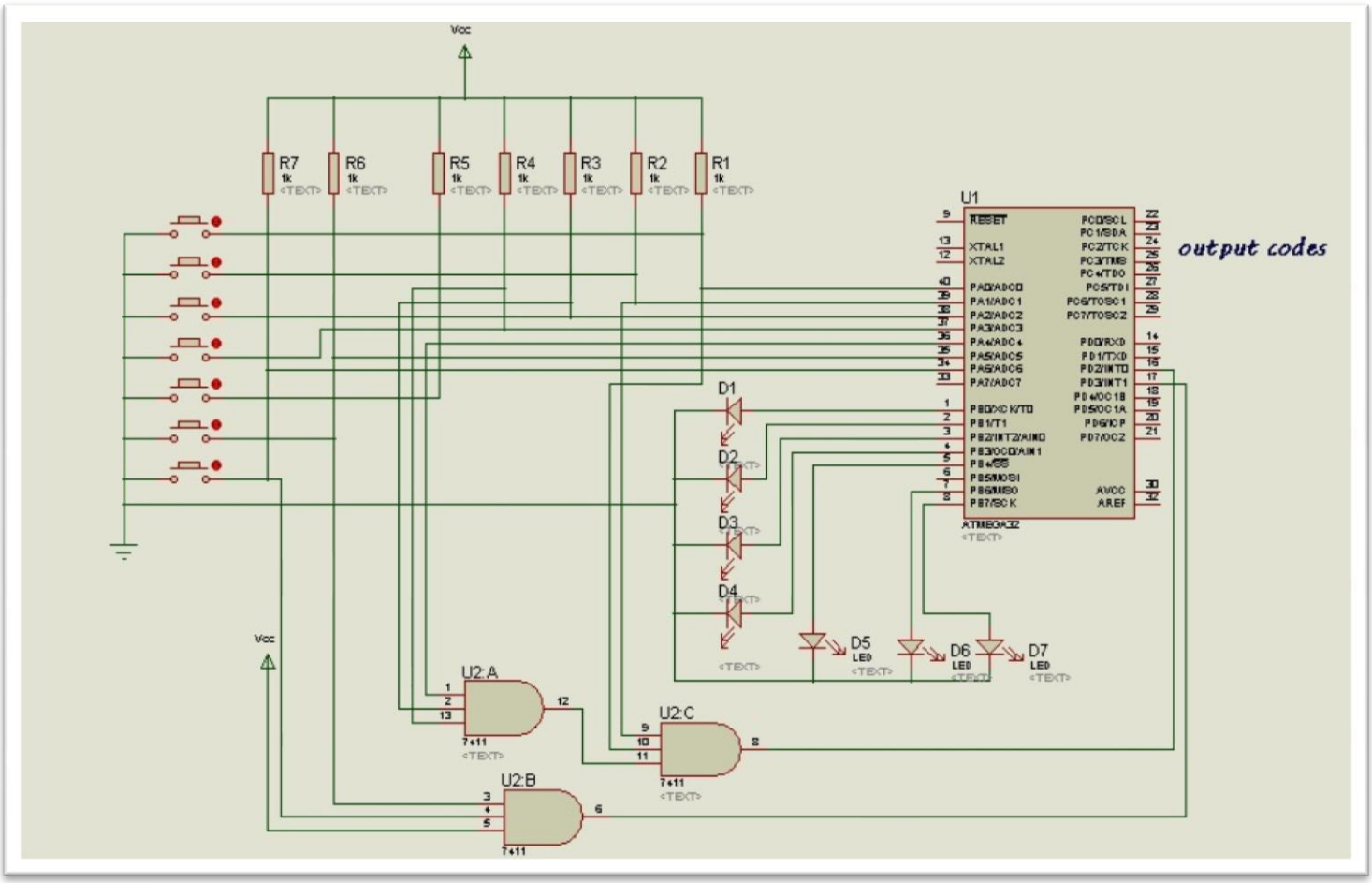
```
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: On
// INT2 Mode: falling Edge
GICR|=0x20;
MCUCR=0x00;
```

```
MCUCSR=0x00;
GIFR=0x20;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// Global enable interrupts
#asm("sei")

GLCD_init();
GLCD_clear();
dot_reset();
image(name);
delay_ms(2000);
image(oscop);
for(b=0;b<100;b++){
in[b]=b*2; }
s7=0; s6=0; s2=0; s1=0; s4=1; s5=1; delay_us(10); s4=0;
while (1){
      switch(voptop){
       case 0b10100100: write_num(0,1,20,2); top1=100; top1*=zn; if(top1>=10000){top2=2; top1/=1000; f=2;
t_um();}
 else{top2=4; f=0; t_um();} write_num(0,4,top1,top2); break;
       case 0b10011011: write_num(0,1,10,2); top1=100; top1*=zn; if(top1>=10000){top2=2; top1/=1000; f=2;
t_um();}
 else{top2=4; f=0; t_um();} write_num(0,4,top1,top2); break;
```

```
        case 0b10010010: write_num(0,1,4,2);  top1=100; top1*=zn; if(top1>=10000){top2=2; top1/=1000; f=2;
t_um();}

else{top2=4; f=0; t_um();} write_num(0,4,top1,top2); break;

        case 0b10001001: write_num(0,1,2,2);  top1=100; top1*=zn; if(top1>=10000){top2=2; top1/=1000; f=2;
t_um();}

 else{top2=4; f=0; t_um();} write_num(0,4,top1,top2); break;

        case 0b10000000: write_num(0,1,1,2);  top1=100; top1*=zn; if(top1>=10000){top2=2; top1/=1000; f=2;
t_um();}

 else{top2=4; f=0; t_um();} write_num(0,4,top1,top2); break;

        case 0b01100100: write_num(0,1,20,2); top1=10;  top1*=zn; if(top1>=10000){top2=2; top1/=1000; f=2;
t_um();}

 else{top2=4; f=0; t_um();} write_num(0,4,top1,top2); break;

        case 0b01011011: write_num(0,1,10,2); top1=10;  top1*=zn; if(top1>=10000){top2=2; top1/=1000; f=2;
t_um();}

 else{top2=4; f=0; t_um();} write_num(0,4,top1,top2); break;

        case 0b01010010: write_num(0,1,4,2);  top1=10;  top1*=zn; if(top1>=10000){top2=2; top1/=1000; f=2;
t_um();}

 else{top2=4; f=0; t_um();} write_num(0,4,top1,top2); break;

        case 0b01001000: write_num(0,1,2,2);  top1=10;  top1*=zn; if(top1>=10000){top2=2; top1/=1000; f=2;
t_um();}

 else{top2=4; f=0; t_um();} write_num(0,4,top1,top2); break;

        case 0b01000000: write_num(0,1,1,2);  top1=10;  top1*=zn; if(top1>=10000){top2=2; top1/=1000; f=2;
t_um();}

 else{top2=4; f=0; t_um();} write_num(0,4,top1,top2); break;  }

       for(j=0;j<100;j++)

     {

        a1=in[j];  a1=a1/4;

        a1=63-a1;

        write_dot(soton,a1,oscop);

        soton++;

        if(ch==1){delay_ms(100) ch=0;}}


         soton=27;

         delay_ms(1000);

         image(oscop);
```

```
dot_reset() ;   };   }
```

تشریح عملکرد میکروی تنظیم کننده رنج ولتاژ و زمان :



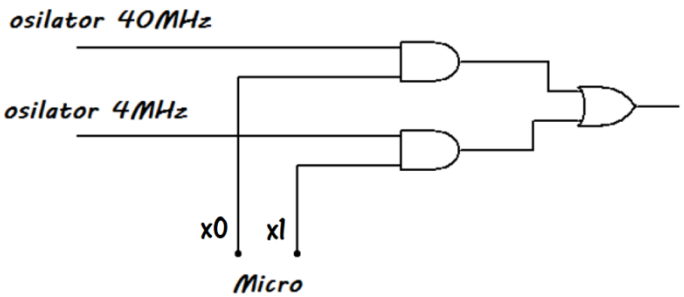
پورت A : نقش این پورت تشخیص کلید (رنج) انتخاب شده است . که بیت ۵و۶ آن دریافت ورودی برای رنج زمان و بیت ۰ و ۱ و ۲ و ۳ و ۴ دریافت ورودی برای رنج ولتاژ می باشد .

پورت B : نقش این پورت روشن کردن LED مرتبط با کلید فشرده شده در رنج زمان و رنج ولتاژ است .

پورت C : این پورت یک کد تولید می کند که شیوه ی تولیدش در زیر بیان شده است .

Code of Port C : X1,X0,Y2,Y1,Y0,Z2,Z1,Z0

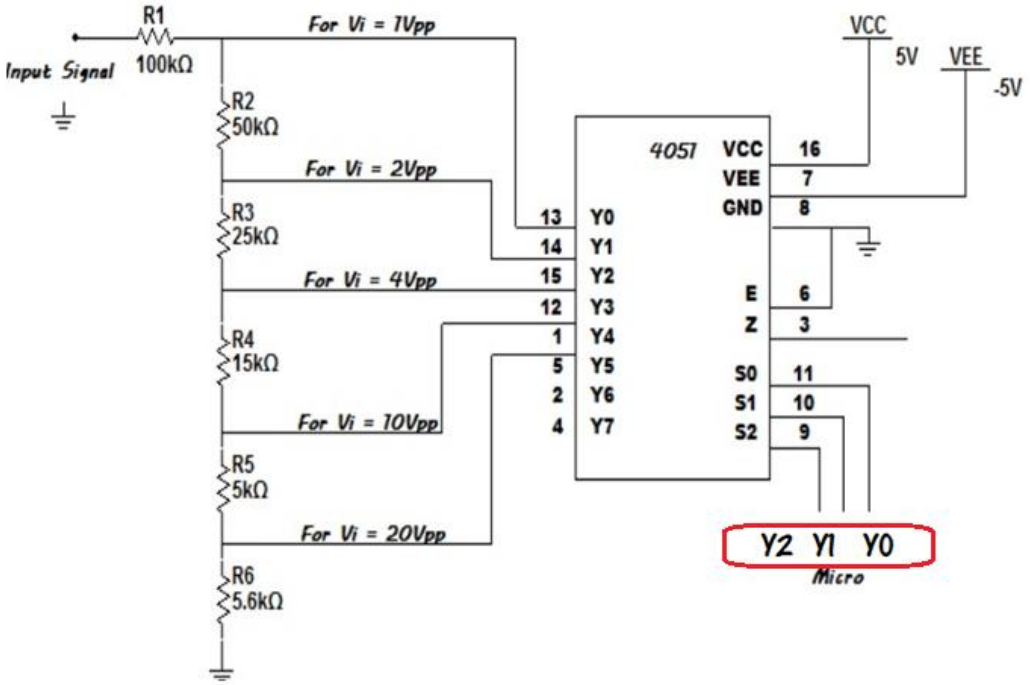
X1,X0 : این دو بیت که پر ارزش ترین بیت های خروجی پورت C هستند ، مربوط به رنج زمان اند که به مدار زیر می روند و عملکردشان در جدول زیر آمده است .



X1	X0	عملکرد
1	0	خروجی فعال 4 MHz
0	1	خروجی فعال

		40MHz

Y2,Y1,Y0 : این سه بیت دقیقا مشابه Z2,Z1,Z0 هستند ، که اعداد ۰ و ۱ و ۲ و ۳ و ۴ را می سازند و به ۴۰۵۱ اعمال می گردند تا رنج ولتاژ را انتخاب گردد .



قابل ذکر است که پورت C از طریق بافر به پورت A میکروی اصلی نیز اعمال می گردد .

پورت D : این پورت تنها برای اینتراپت مورد استفاده قرار گرفته است ، که اینتراپت صفر برای کلید ها و کدسازی رنج ولتاژ و اینتراپت یک برای دو کلید و کدسازی رنج زمان مورد استفاده قرار گرفته است .

برنامه میکرو :

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.03.5 Evaluation
Automatic Program Generator
© Copyright 1998-2008 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com
```

```
Project :
Version :
Date    : 5/8/2010
Author  : Freeware, for evaluation and non-commercial use only
Company :
Comments:


Chip type           : ATmega16
Program type        : Application
Clock frequency     : 1.000000 MHz
Memory model        : Small
External RAM size   : 0
Data Stack size     : 256
*****************************************************/

#include <mega16.h>
#define k1 PINA.0
#define k2 PINA.1
#define k3 PINA.2
#define k4 PINA.3
#define k5 PINA.4
#define k6 PINA.5
#define k7 PINA.6
// External Interrupt 0 service routine
interrupt [EXT_INT0] void ext_int0_isr(void)
```

```
{
// Place your code here
if(k1==0)
{
  PORTB&=0b11000000;
  PORTC&=0b11100000;
  PORTC.0=1;
}
if(k2==0)
{
  PORTB&=0b11000000; PORTB.0=1; PORTB.3=1;
  PORTC&=0b11100000;
  PORTC.1=1;
}
if(k3==0)
{
  PORTB&=0b11000000; PORTB.1=1; PORTB.4=1;
  PORTC&=0b11100000;
  PORTC.2=1;
}
if(k4==0)
{
  PORTB&=0b11000000; PORTB.0=1; PORTB.1=1; PORTB.3=1; PORTB.4=1;
  PORTC&=0b11100000;
  PORTC.3=1;
}
```

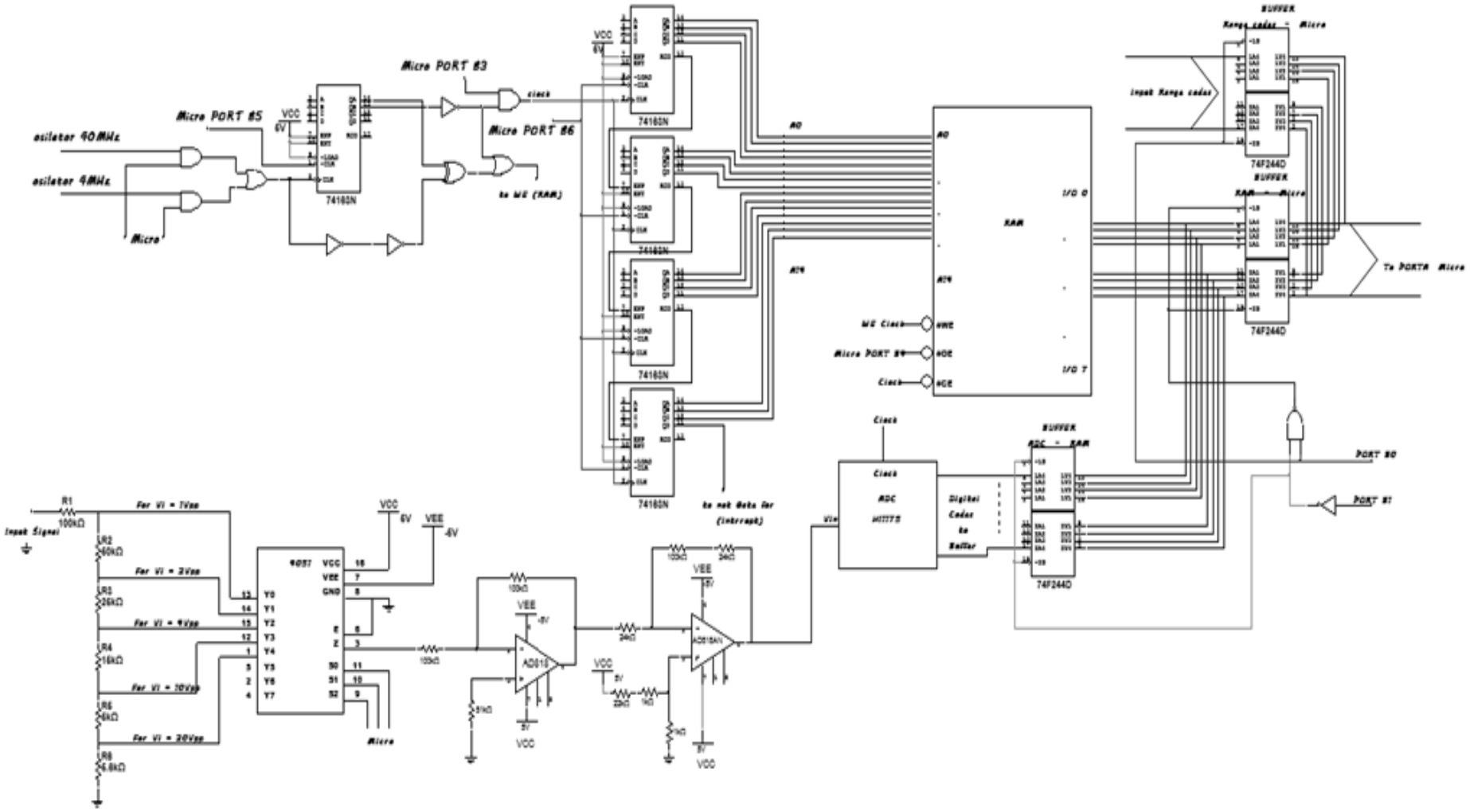
```
if(k5==0)
{
  PORTB&=0b11100100; PORTB.2=1; PORTB.5=1;
  PORTC&=0b11100000;
  PORTC.4=1;
}
}

// External Interrupt 1 service routine
interrupt [EXT_INT1] void ext_int1_isr(void)
{
 if(k6==0)
 {
  PORTC.6=1; PORTC.7=0;
  PORTB.7=0; PORTB.6=1;
 }
 if(k7==0){
  PORTB.7=1; PORTB.6=0;
  PORTC.6=0; PORTC.7=1;
  }
  }

// Declare your global variables here

void main(void)
{
```

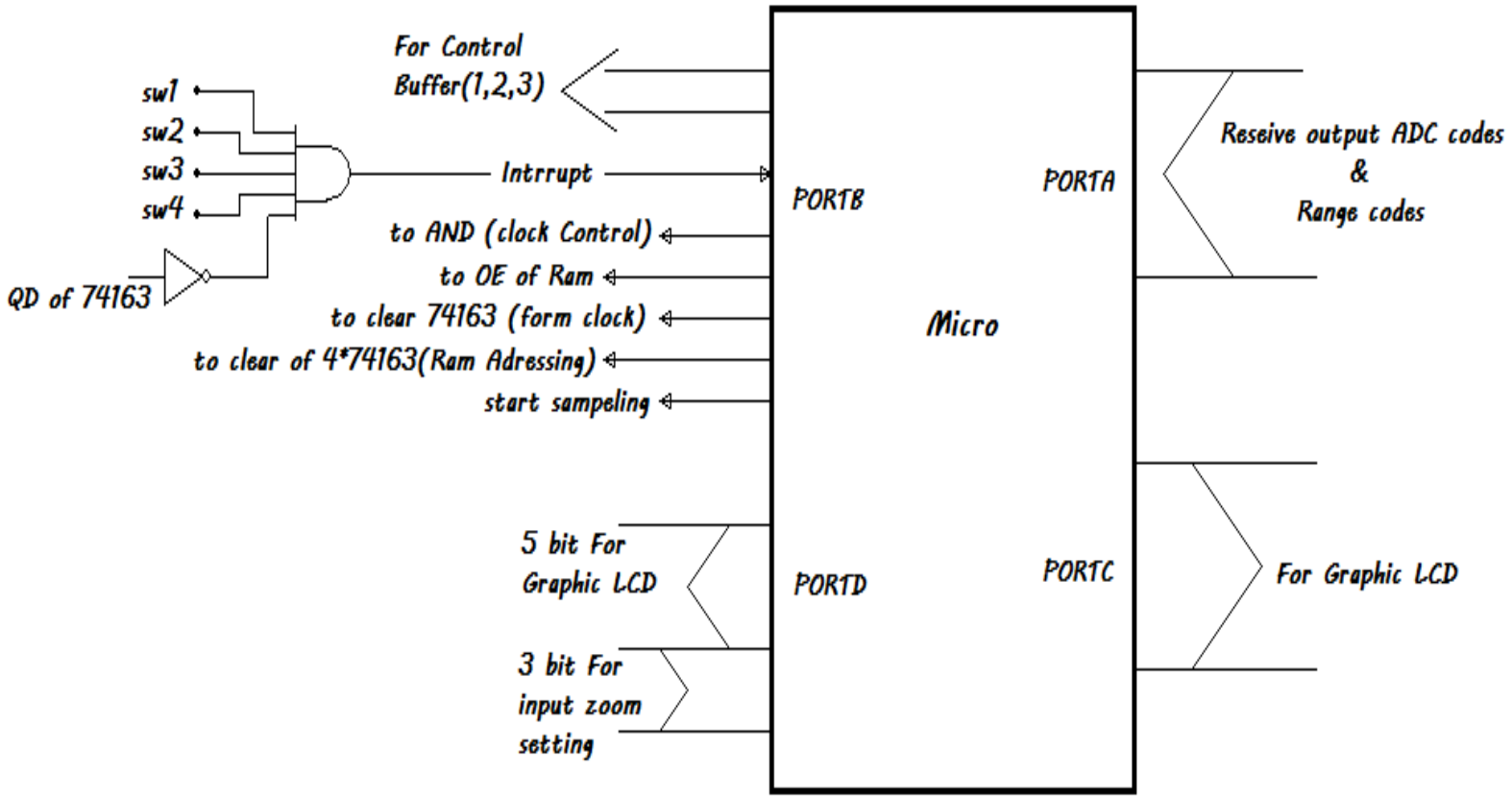
```
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0xff;
DDRA=0x00;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out
Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTB=0x00;
DDRB=0xFF;

// Port C initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out
Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTC=0x00;
DDRC=0xFF;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
```

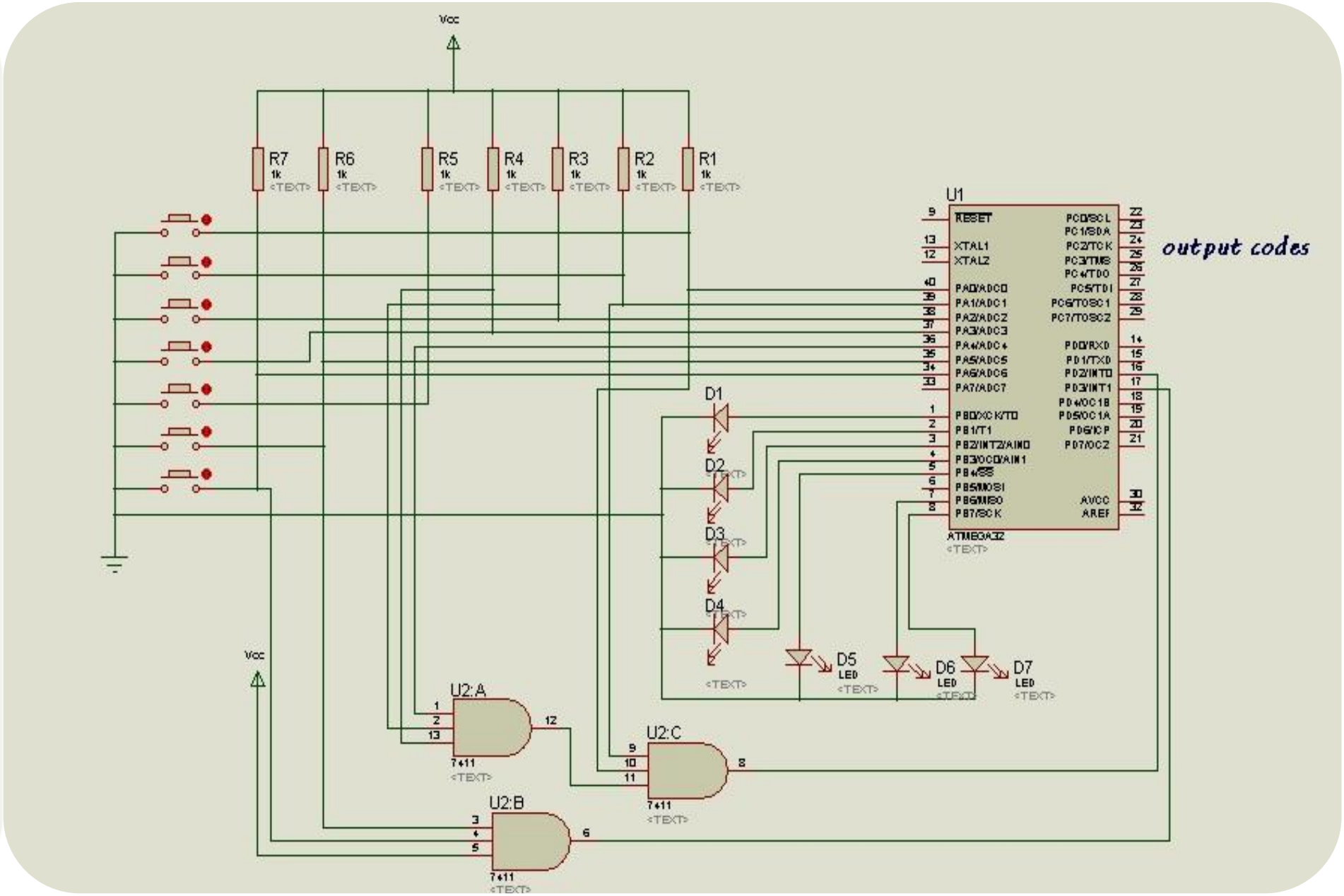
```
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
```

```
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: On
// INT0 Mode: Falling Edge
// INT1: On
// INT1 Mode: Falling Edge
// INT2: Off
GICR|=0xC0;
MCUCR=0x0A;
```

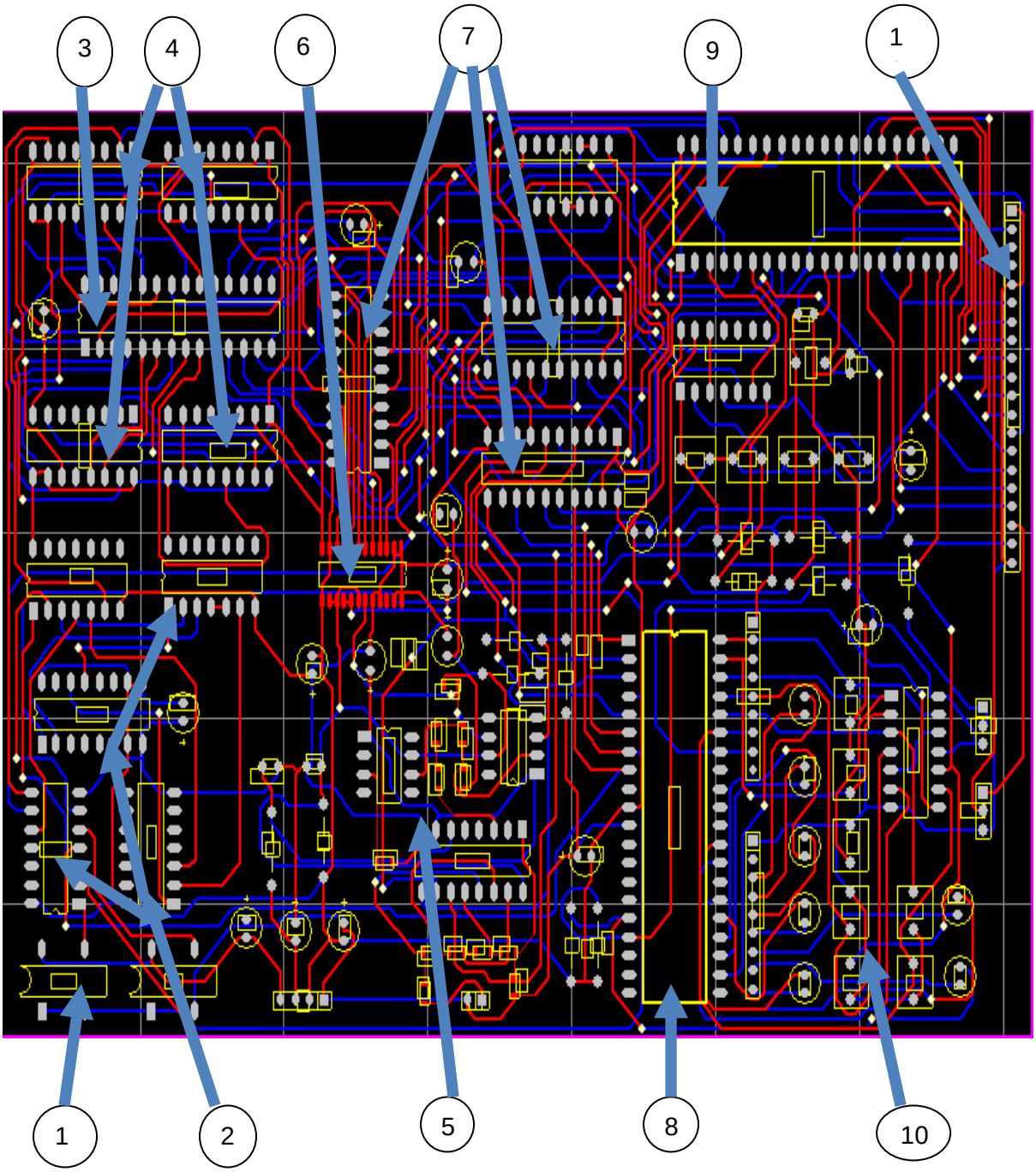
```
MCUCSR=0x00;
GIFR=0xC0;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// Global enable interrupts
#asm("sei")
PORTC=0x90;
PORTB=0xa4;
while (1)
    {
    // Place your code here


    };
}
```


Micro PORT B3
Micro PORT B5
Micro PORT B6
oscilator 40MHz
oscilator 4MHz
Micro
74160N
to WE (RAM)
to not Data for (interrupt)
RAM
I/O 0
I/O 7
WE Clock
Micro PORT B4
Clock
Input Range codes
74F244D
To PORTA Micro
PORT B0
PORT B1
Clock
ADC
Digital Codes to Buffer
4051
AD818
AD818AN
Input Signal
For Vi = 1Vpp
For Vi = 2Vpp
For Vi = 4Vpp
For Vi = 10Vpp
For Vi = 20Vpp


For Control
Buffer(1,2,3)
sw1
sw2
sw3
sw4
Intrrupt
QD of 74163
to AND (clock Control)
to OE of Ram
to clear 74163 (form clock)
to clear of 4*74163(Ram Adressing)
start sampeling
PORTB
PORTA
Micro
Reseive output ADC codes
&
Range codes
5 bit For
Graphic LCD
3 bit For
input zoom
setting
PORTD
PORTC
For Graphic LCD

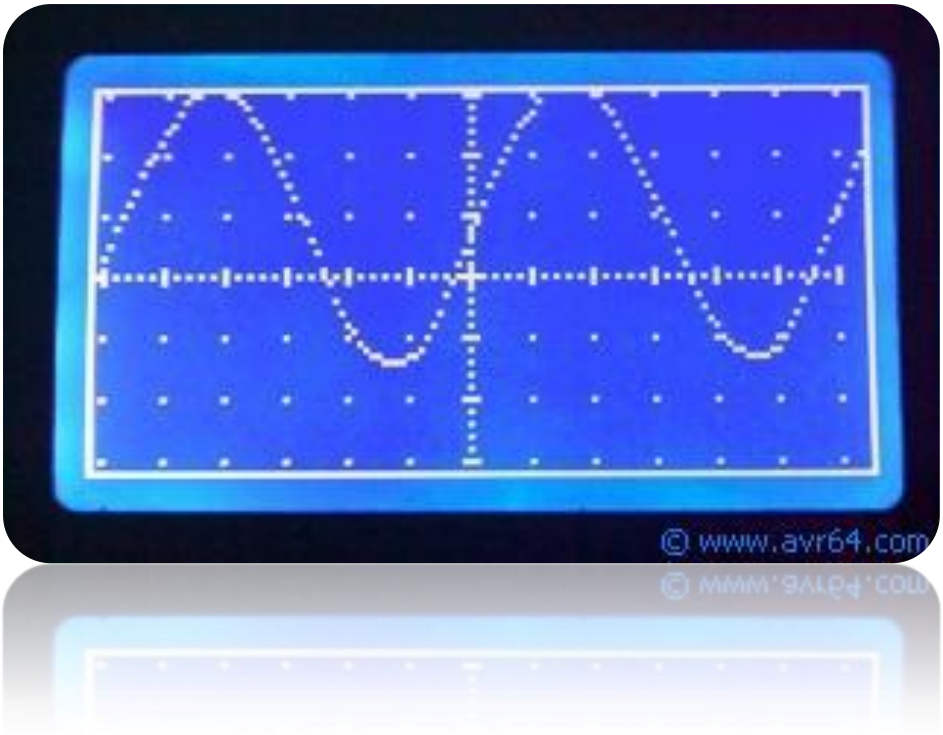

Vcc
R7
1k
R6
1k
R5
1k
R4
1k
R3
1k
R2
1k
R1
1k
U1
RESET
XTAL1
XTAL2
PA0/ADC0
PA1/ADC1
PA2/ADC2
PA3/ADC3
PA4/ADC4
PA5/ADC5
PA6/ADC6
PA7/ADC7
PB0/XCK/T0
PB1/T1
PB2/INT2/AIN0
PB3/OC0/AIN1
PB4/SS
PB5/MOSI
PB6/MISO
PB7/SCK
PC0/SCL
PC1/SDA
PC2/TCK
PC3/TMS
PC4/TDO
PC5/TDI
PC6/TOSC1
PC7/TOSC2
PD0/RXD
PD1/TXD
PD2/INT0
PD3/INT1
PD4/OC1B
PD5/OC1A
PD6/ICP
PD7/OC2
AVCC
AREF
ATMEGA32
output codes
D1
D2
D3
D4
D5
LED
D6
LED
D7
LED
U2:A
U2:B
U2:C
7411

برد مدار چاپی:



لازم به ذکر است که توضیح زیر به صورت اجمالی میباشد و کامل آن در قسمت های بعد میباشد.

1. **در این قسمت مدار دو اسیلاتور اصلی مدار که وظیفه ی نوسان سازی را دارند قرار دارند.**
2. **در این قست گیت ها و شمارنده ای که وظیفه ی ایجاد کلاک ADC و RAM را دارند قرار دارند.**
3. **Ram که وظیفه ی ذخیره ی کد های خروجی ADC را دارد.**
4. **شمارنده هایی که وظیفه ی ایجاد پالس های آدرس دهی RAM را دارد در این قسمت جاگذاری شده اند.**
5. **قسمت ورودی مدر که شامل ماتیلکسر و تقویت کننده میباشد در این قسمت میباشد.**
6. **ADC مدر که به صورت SMD میباشد و بر روی برد قرار میکیرد.**
7. **بافر های مدار که وظیفه ی ارتباط دهی و قطع ارتباط پردازنده و ADC و RAM با هم را دارا میباشند و در هر لحظه با توجه به عملکرد مدار آن ها را به هم وصل میکنند.**
8. **پردازنده جانبی که وظیفه ی تغییر تایم و ولتاژ ورودی برای نمایش بر روی GLCD را بر عهده دارد.**
9. **میکرو اصلی که هم خواندن کد از RAM و هم نمایش بر روی LCD و هم خواندن مقادیر تایم و ولتاژ از میکرو دوم را بر عهدا دارد.**
10. **سوکت برای اتصال GLCD به برد.**
11. **سوییچ های مربوط به تغییر T/D و V/D مدار.**



از قابلیت های این نوع LCD توانایی

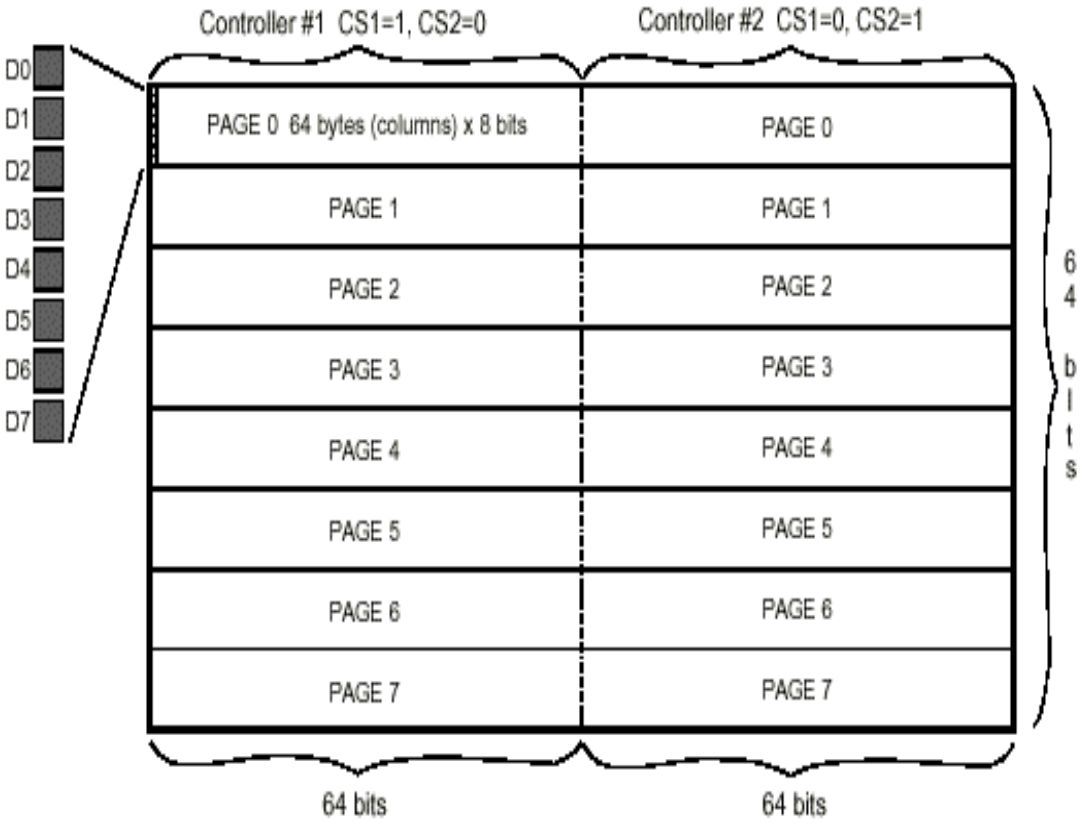
روشن کردن پسکسل به پیکسل میباشد.

که در این صورت میتوان انواع افکت ها و اشکال مختلف را بر روی آن به نمایش در آورد.

برای نشان دادن شکل موج در این پروژه از LCD گرافیکی استفاده کرده ایم که لازم است به طرز کار آن بپردازیم.

این LCD از نوع GDM128*64 میباشد و داری ۲۰ پایه برای کنترول و ارتباط دیتا. از مشخصات آن میتوان گفت که در ساختار به دو قسمت تقسیم شده است که برای فعال سازی هر کدام باید پایه ی مربو ط به آن قسمت را فعال نمود و هر صفحه نمایش به ۸ صفحه که هر کدام دارای ۸ سطر و ۶۴ ستون میباشد.

در شکل زیر چگونگی صفحه بندی وترکیب پایه ها را مشاهده یکنید

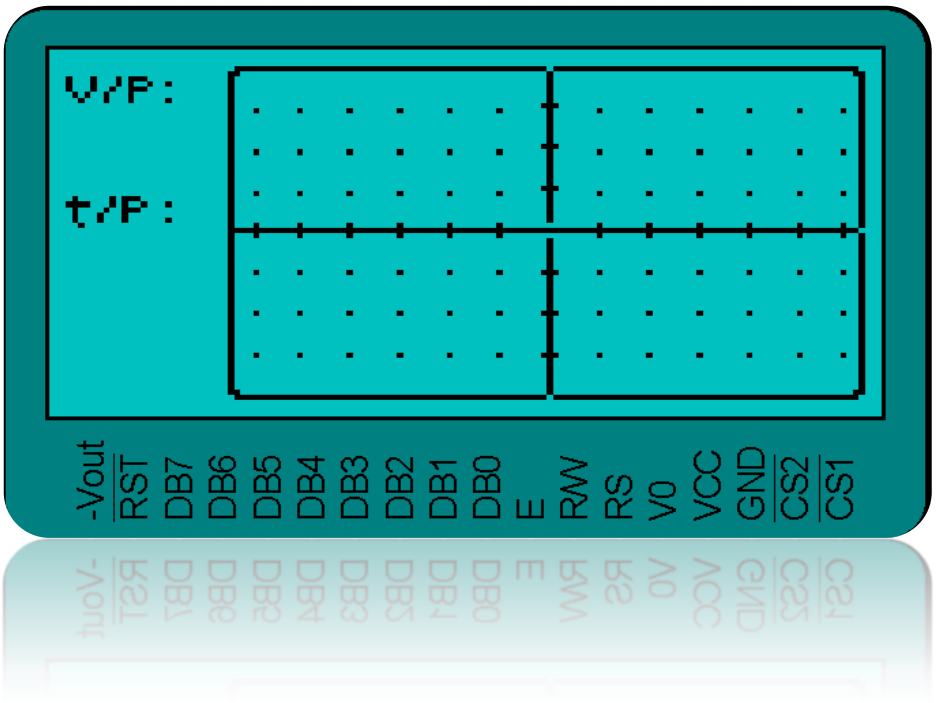


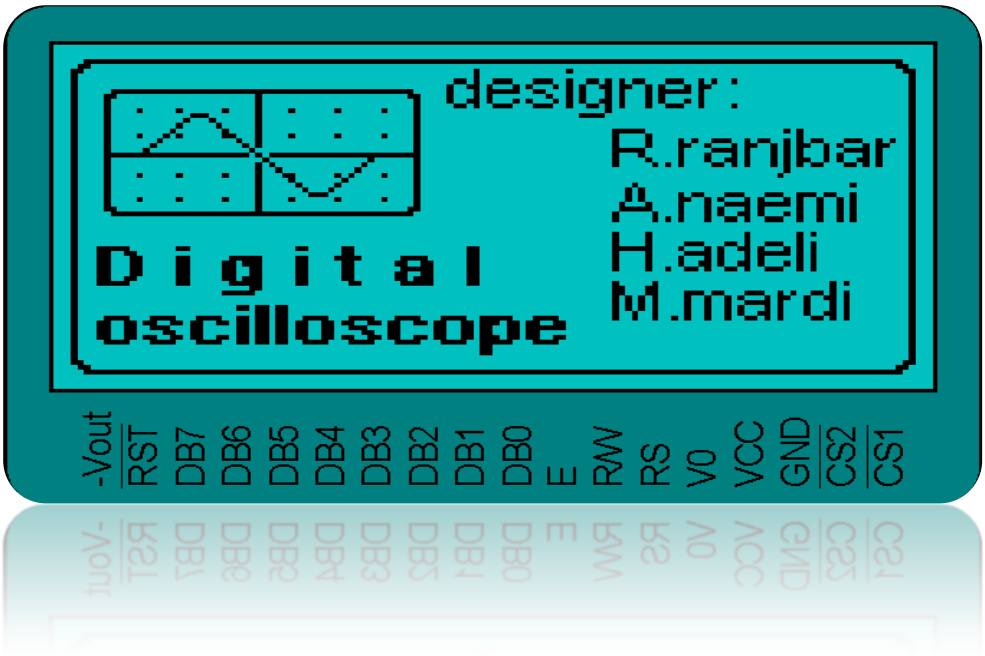
همان طور که در شکل بالا مشاهده میکنید صفحه ی LCD به دو نیم چپ و راست تقسیم شده و هر قسمت دارای ۷ صفحه و هر صفحه دارای ۶۴ ستون که هر ستون ۸ بیت را داراست.

شماره پایه	نام پایه	وظیفه هر پایه
۱	VSS	پایه ی زمین lcd که بهید به زمین ۵ ولت وصل شود.
۲	VDD	باید به ۵+ وصل شود.
۳	VO	برای تنظیم وضوح lcd که باید توسط ولتاز منفی که به وسیله ی پایه ی VEE ایجا میشودبا یک پتانسیومتر تنظیم شود .
۴	RS	اگر صفر شود رجیستر دستور فعالمیشود و اطلاعات بر روی DB0 تا DB7 بعنوان دستور و اگر یک باشد رجیستر داده برا نمایش یر روی lcd فعال میشود
۵	R/W	اگر صفر باشد اطلاعات بر روی lcd نوشته میشود و اگر یک باشد اطلاعات از روی پایه ها توسط میکرو خوانده میشود.
۶	E	برای ذخیره ی اطلاعات روی lcd باید یک پالس حداقل 450ns

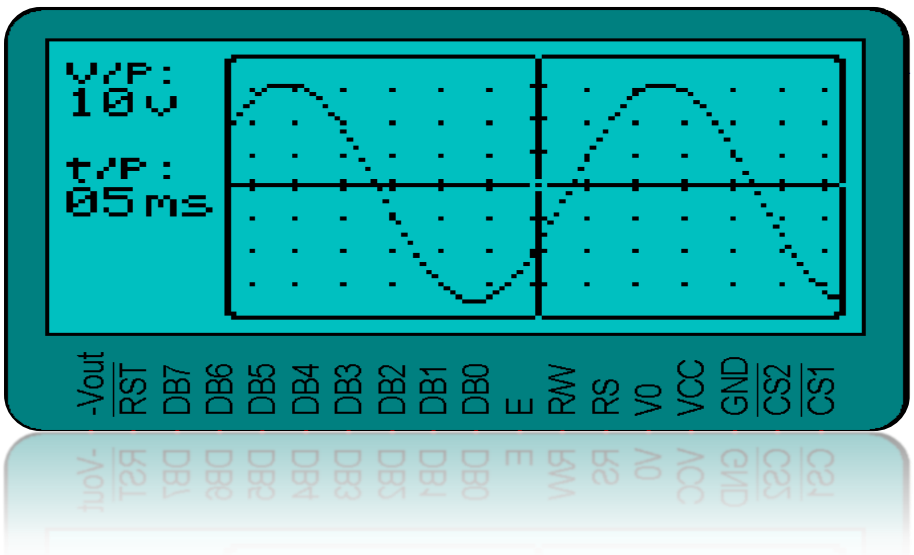
		روی آن با لبه ی پایین رونده ایجاد شود.
۱۴-۷	DB0-DB7	برای انتقال داده به lcd و یا خواندن از آن استفاده میشود.
۱۵	CSI	برای انتخاب نیمه چپ lcd
۱۶	CS2	برای انتخاب نیمه راست lcd
۱۷	RES	اگر صفر شود lcd در حالت غیر فعال قرار میگیرد.
۱۸	VEE	یک ولتاژ منفی در حدود ۹.۵ برای پایه ی vo ایجاد میکند.
۱۹	A	پایه ی اند روشن کننده چراغ پشت lcd میباشد و به ولتاژ حدود ۵ ولت نیاز دارد.
۲۰	K	پایه کاتد چراغ پشت lcd که باید به زمین وصل شود.

در این پروژه ما ابتدا یک افکت ثابت که متناظر صفحه اسیلوسکوپ است را بر روی صفحه ی LCD نمایش داده هایم و سپس با روشن کردن پیکسل ها ی متناظر با شکل موج ورودی ،شکل موج مورد نظر را بر روی آن به نمایش در آورده ایم. سپس با خواندن مقادیر زمان و ولتاژ و نمایش آن در قسمت V/P و T/P کار را به پایان رساندیم.





همان طور که در شکل زیر مشاهده میشود با روشن کردن یک نقطه در سطر و اضافه کردن یک واحد در ستون LCD میتوان شکل موج را بروی LCD نمایش داد.



با این کار اگر ما کد ها را درست خوانده باشیم شمای شکل موج ورودی بر روی صفحه GLCD نمایش داده میشود. همچنین چون مقدار موج ورودی بعد از ۰ تا ۲۵۵ متغییر است و تعداد سطر های LCD ۰ تا ۶۳ میباشد بنابر این باید مقدار ورودی را بر ۴ تقسیم کنیمو همچنین چون LCD در ربع چهارم قرار دارد و مقدار ها رو به پایین افزایش میابد بنابر این باید عدد حاصا از تقسیم را از ۶۳ کم کنیم.

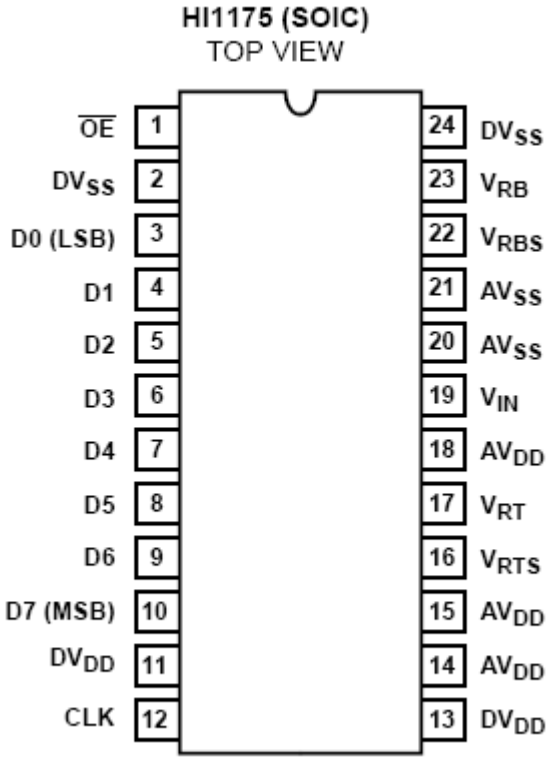
ADC :

برای تبدیل سیگنال ورودی به دیجیتال ما به یک مبدل آنالوگ به دیجیتال نیاز داریم که از IC ، HI1175 استفاده کرده ایم. این ADC با حداکثر فرکانس کاری 20MHZ در هر لحظه با توجه به کلاک ورودی ، از سیگنال ورودی نمونه گرفته و آن را به ۸ بیت دیجیتال تبدیل میکند. از مزیت این IC این این که خود یک ولتاژ در حدود ۲.۶ ولت ایجاد میکند که به عنوان ولتاژ مرجع از آن استفاده میشود . در این صرت به ازای هر 10mv یک تغییر در بیت خروجی داریم . لازم به ذکر ایت که این IC با کلاک کمتر از 500KHZ کار نمیکند. لازم به ذکر است که این مبدل در لبه ی پایین رونده کلاک ورودی نمونه را گرفته و با تاخیر ۲.۵ سیکل نمونه را در خروجی نمایش میدهد.

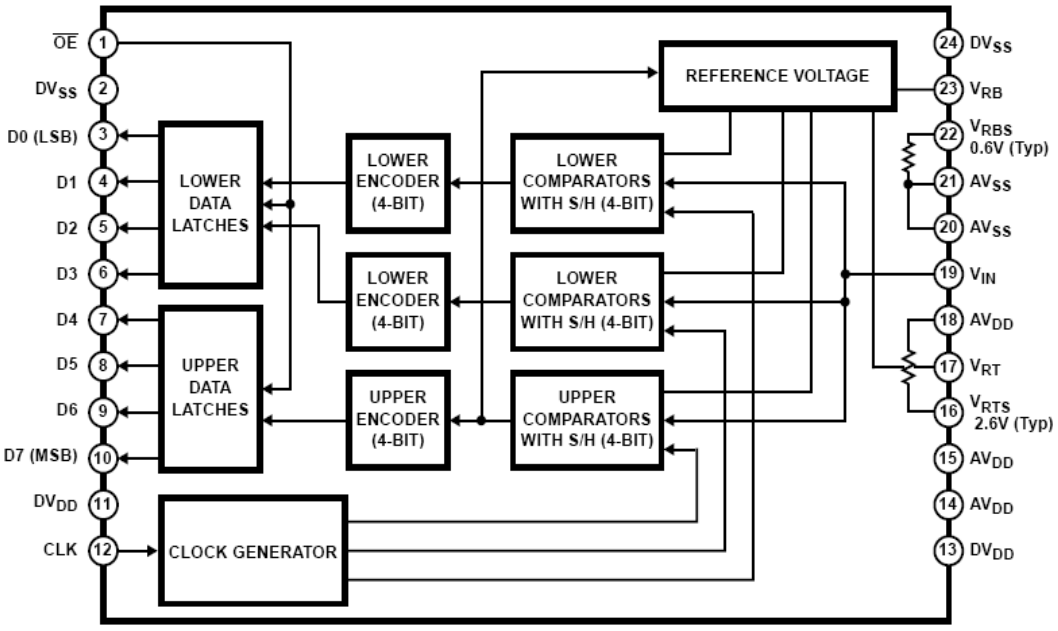
معرفی پایه ها:

شماره پایه	نام پایه	شرح پایه ها
۱	OE	پایه ی کنترولی میباشد که در صورت صفر بودن IC فعال و در صورت یک بودت خروجی غیر فعال است.
۲و۲۴	DVSS	زمین دیجیتال میباشد.
۳-۱۰	D0 t0 D7	دیتای دیجیتال خجی میباشند.
۱۱و۱۳	DVDD	باید به ۵ ولت دیجیتال وصل شوند.
۱۲	CLK	کلاک ورودی.
۱۶	VRTS	ولتاژ مرجع بالا را تولید میکند(2.56v)
۱۷	VRT	ورودی ولتاژ مرجع بالا.
۲۳	VRB	ورودی ولتاژ مرجع پایین.
۱۴و۱۵و۱۸	AVDD	باید به ۵ ولت آنالوگ وصل شود.
۱۹	VIN	ولتاژ ورودی
۲۰و۲۱	AVSS	زمین آنالوگ میباشد.
۲۲	VRBS	ولتاژ مرجع پایین را تولید میکند(0.6)

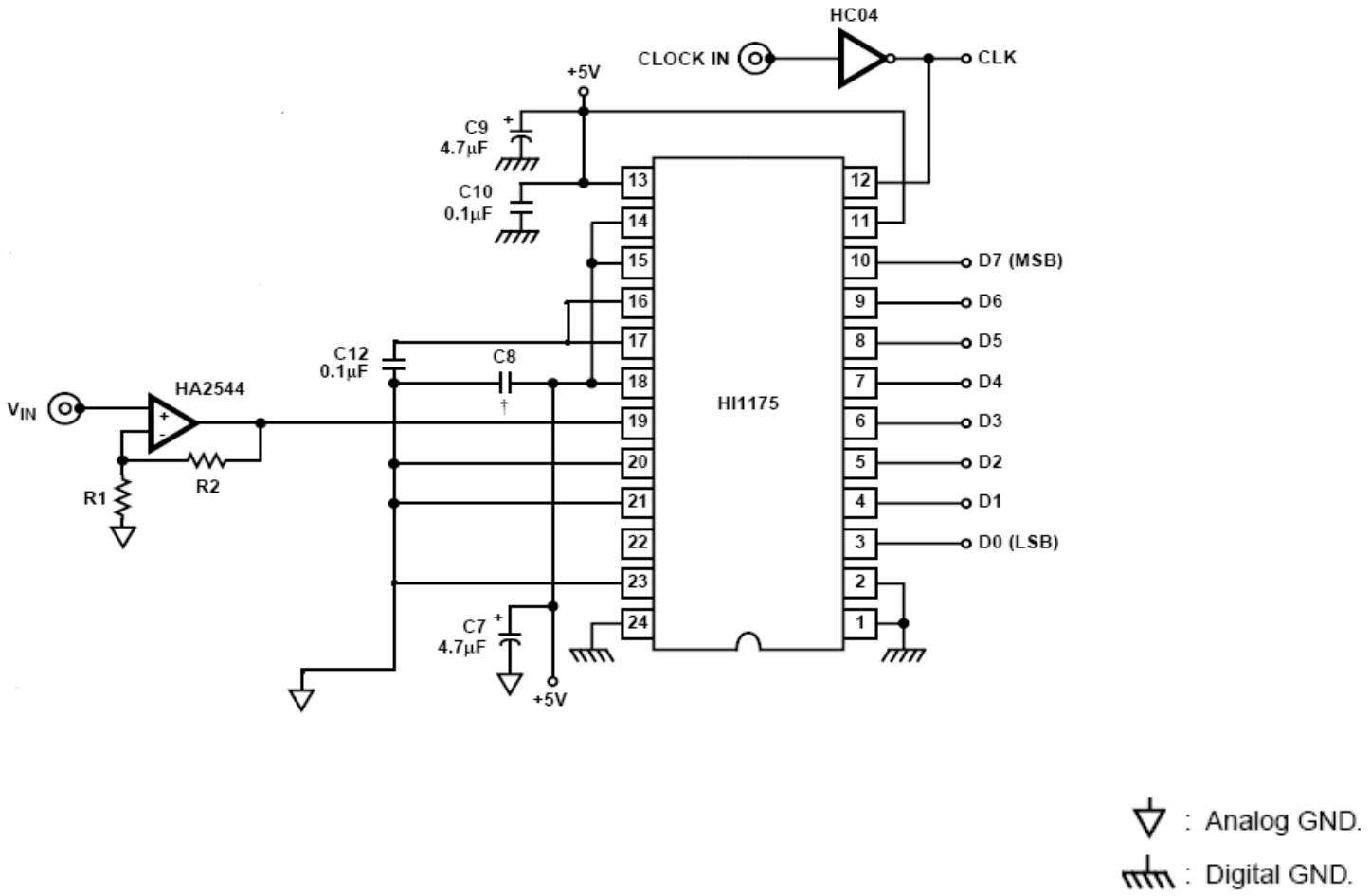
شمای IC:



مدار داخلی IC :

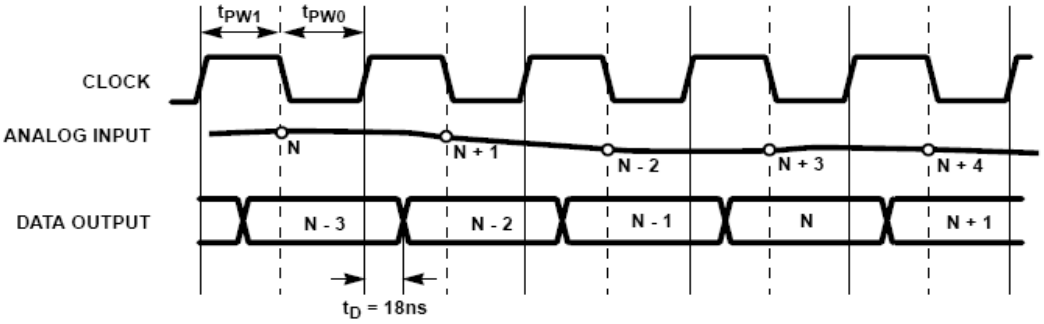


مدار عملی:



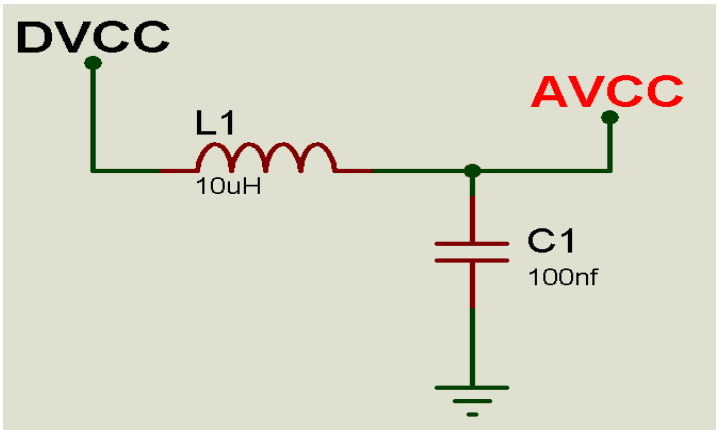
لازم به ذکر است که سیگنال ورودی باید در محدوده ولتاژ مرجع باشد که برای این کار میتوان از OP-amp **استفاده کرد. همچنبن وجود خازن های نویز گیر ضروری است. برای این کار در قسمت تقویت ، کننده ولتاژ ورودی را با یک مقدار** DC **جمع میکنیم.**

رابطه ی سیگنال ورودی با کلاک و خروجی:

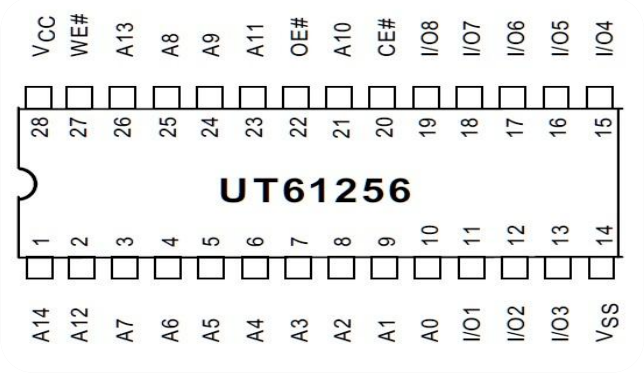


نکته ای که در اینجا قابل ذکر است این است ک نمونه ها ۲.۵ سیکل بعد در خروجی ظاهر میشوند.

در این پروژه ما برای ایجاد ولتاژ مرجع ورودی از خروجی های پایه های VRBS و VRTS استفاده کرده ایم به این صورت که این پایه ها را به صورت مستقیم به VRB و VRT وصل کرده ایم. همچنین برای عدم قرار گرفتن نویز بر روی مدار از یک مدار LC برای ایجاد ولتاژ AVCC (ولتاژ تغذیه ی آنالوگ) استفاده کرده ایم.



آشنایی با RAM :



رم وسیله ای است که اطلاعات را در آن نوشته و می خوانیم . البته با قطع برق رم نیز پاک می شود . پایه های رم را می توان به ۴ دسته ی زیر تقسیم کرد :

۱) پایه های تغذیه (VCC , VSS)

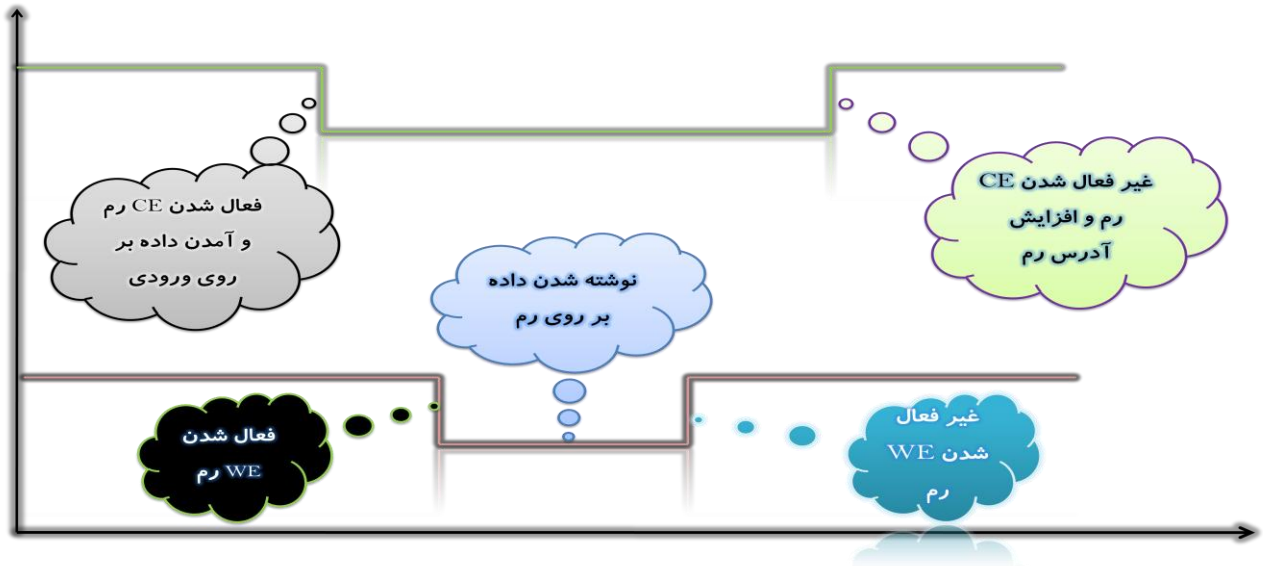
۲) پایه های آدرس (A0,...,A14)

۳) پایه های ورودی و خروجی (I/O0,...,I/O8)

۴) پایه های کنترلی (OE,WE,CE)

شیوه ی نوشتن اطلاعات بر روی رم :

برای نوشتن اطلاعات روی رم ابتدا آدرس مورد نظر را به رم اعمال می کنیم و بعد از آن پایه ی CE فعال می گردد سپس اطلاعات بر روی ورودی قرار گرفته و پس از آن WE فعال می شود و ورودی بر روی رم نوشته می شود و WE غیر فعال و CE نیز غیرفعال می گردد و در تمام این مدت OE غیر فعال می ماند .



شیوه ی خواندن اطلاعات از روی رم :

برای خواندن اطلاعات از روی رم ابتدا آدرس مورد نظر را به رم اعمال می کنیم و بعد از آن پایه ی CE فعال می گردد و پس از آن OE فعال می شود و داده از روی رم خوانده می شود و سپس OE غیر فعال و بعد از آن CE غیر فعال می گردد و در تمام این مدت WE غیر فعال می باشد .