به نام خدا

پروژه ی دوم میکروپروسسور

بازی کوچکMAZE

رضاکریم زاده

9333084

تابستان96

در مرحله ی اول برای انجام پروژه باید سعی کنیم روی lcd ردیف های موانع را ایجاد کنیم که یکی از آنها به صورت رندم خالی است حال ببینیم در میکرو چگونه رندم تولید می کنند:

درمیکرو با تابع ()rand یک عدد رندم دروغین بین0تا32767 به ما می دهد حال ما عدد رندمی بین0و3 می خواهیم که برای این کار به شکل زیر عمل می کنیم:

```
(int)((rand()/(RAND_MAX/4)))
```

به این ترتیب و با گرد کردن عدد توسط تابعint یک عدد صحیح نسبتا رندم بین 0و3 بدست می آوریم.

پس برای آنکه این ردیف از"|" ها را که یک خانه ی رندم خالی دارد روی lcd بیندازیم به شکل تابع زیر عمل می کنیم:

```
void chap(void)
{
        time=difficulty();
        t=m*3;
        for(j=0;j<=t;j=j+3)
        {
      for(i=0;i<4;i++)
      {
        if(i!=jump[j/3])
        {
          lcd_gotoxy(j,i);
          lcd_puts("|");
          lcd_gotoxy(13,x);
          lcd_puts("*");
        }
      }
      }
}
```

تابع چاپ به این شکل کار می کند که در هر بار فراخوانی تابع difficulty که میزان سرعت بازی را مشخص می کند و بعدا کامل توضیح داده خواهد شد را فراخوانی کرده تا فاصله ی افتادن بلوک ها آپدیت شود سپس متغییرt را محاسبه می کند که 3برابر m است. m برای آن تعریف شده است که در ابتدای بازی کهm صفر است فقط یک ردیف یعنی ردیف صفرم چاپ شود و با افزایشm تعداد ردیف ها افزایش یابد تا اینکه کل ردیف ها که 6تا هستند چاپ شود و مقدار m برابر عدد ثابت5 شود به شکل زیر:

```
m++;
if(m>5)
{
 m=5;
}
```

حال برای چاپ برروی lcd دو حلقه ی for تشکیل داده ایم که در for اول با توجه به m تعداد ردیف هایی که باید با فاصله ی 3کاراکتر از هم چاپ شوند تعیین می شود سپس در for داخلی هر ردیف چاپ می شود به صورتی که با توجه با آرایه ی jump قسمت خالی هر ردیف مشخص می شود به این صورت که اگر مخالف آن عدد رندم درون آرایه ی jump بود چاپ می شود همچنین نشانگر"*"هر بار با توجه به مختصات x که در اینتراپت های خارجی تعیین می شود آپدیت می شود. حال به بررسی آرایه ی jump می پردازیم:

این آرایه به صورت6تایی یعنی jump[6] تعریف شده است چون در lcd هربار باید6 ردیف نشان داده شود ودر هر ردیف یک عدد رندم باشد که با پایین آمدن ردیف عدد رندم نیز پایین بیاید برای این کار هر بار آرایهی jump را به سمت راست شیفت می دهیم تا با پایین آمدن ردیف عدد رندم مربوط به آن ثابت بماند و هر بار برای تولید رندم جدید آنرا در jump[0] قرار می دهیم به شکل زیر:

```
 for(k=0;k<5;k=k+1){
jump[5-k]=jump[4-k];
}
jump[0]=(int)((rand()/(RAND_MAX/4)));
chap();
```

بعد از شیفت دادن و قرار دادن رندم جدید در jump[0] تابع چاپ فراخوانی می شود تا lcd به روز شود.

درمرحله ی دوم پروژه اینتراپت های خارجی0و1و2 را فعال میکنیم تا استفاده های جانبی از هرکدام صورت گیرد:

```
GICR|=(1<<INT1) | (1<<INT0) | (1<<INT2);
MCUCR=(1<<ISC11) | (0<<ISC10) | (1<<ISC01) | (0<<ISC00);
MCUCSR=(0<<ISC2);
GIFR=(1<<INTF1) | (1<<INTF0) | (1<<INTF2);
```

در وقفه ی مربوط به int2 راه اندازی بازی شکل می گیرد یعنی تا وقتی این وقفه اعمال نشده باشد و start=0 باشد همه ی قسمت های بازی به جز قسمت تعیین سختی بازی غیر فعال خواهد بود که بایک شرط if در هر قسمت این کار صورت گرفته است:

```
interrupt [EXT_INT2] void ext_int2_isr(void)
{
  start=1;
}
```

در وقفه ی مربوط به int0 می خواهیم نشانگر"*"به چپ حرکت کند بنابراین بایدx که در تابع چاپ هربار آپدیت می شود تعیین شود برای رفتن به سمت چپ باید هربار تا زمانیکه x برابر 3 نشده یکی به آن اضافه کنیم. به صورت زیر سپس برای آپدیت lcd آنرا پاک میکنیم و تابع چاپ را فراخوانی میکنیم:

```
 interrupt [EXT_INT0] void ext_int0_isr(void)
{
if(start!=0){
if(x!=3){
x=x+1;
lcd_clear();
chap();
}
}
}
```

در وقفه ی مربوط به int1 می خواهیم نشانگر"*"به راست حرکت کند بنابراین بایدx که در تابع چاپ هربار آپدیت می شود تعیین شود برای رفتن به سمت چپ باید هربار تا زمانیکه x برابر 0 نشده یکی از آن کم کنیم. به صورت زیر سپس برای آپدیت lcd آنرا پاک میکنیم و تابع چاپ را فراخوانی میکنیم:

```
interrupt [EXT_INT1] void ext_int1_isr(void)
{
if(start!=0){
if(x!=0){
x=x-1;
lcd_clear();
chap();
}
}
}
```

در مرحله ی سوم باید شرط باخت بازی را مورد بررسی قرار دهیم به این صورت که تا زمانی که کل ردیف ها وارد صفحه نشده باشند یعنیm=5 نشده باشد وارد چک کردن شرط باخت نمی شود سپس هربار که ردیف ها از نشانگر * گذشتند و مختصات x مخالف مقدار موجود در jump[5] بود تابع gameover() فراخوانی خواهد شد.

```
if(m==5){
if(x!=jump[5])
{
gameover();
}
 w=1+(1.5-time)/.6+w;
}
```

پس از آنکه شرط باخت بازی بررسی شد و طلاقی شکل نگرفته بود امتیاز آن قسمت با توجه به دستور داده شده محاسبه خواهد شد و به مقدار قبلی اضافه می شود.

حال به بررسی تابعgameover() می پردازیم:

دراین تابع با توجه به دستور پروژه باید 4 ثانیه به صورت چشمک زن عبارت"GAME OVER" نشان داده شود سپس امتیاز نهایی چاپ شود همچنین از طریقusart امتیاز نهایی ارسال شود :

```
void gameover(void)
{
  lcd_clear();
  printf("final score:");
  putchar(13);
  ftoa(w,1,str);
  sprintf(lcd,"%5s",str);
  puts(lcd);
  putchar(13);

  for(i=0;i<4;i++){
  lcd_clear();
  lcd_gotoxy(3,1);
  lcd_puts("GAME OVER");
  delay_ms(500);
  lcd_clear();
  delay_ms(500);
  }


  ftoa(w,1,str);
  sprintf(lcd,"YOUR SCORE IS:%5s",str);
  lcd_clear();
  lcd_gotoxy(0,1);
  lcd_puts(lcd);
  delay_ms(2000);
  while(1){start=0;}
}
```

در آخر این تابع می خواهیم از آن خارج نشود پس یک لوپ بی نهایت تعریف می کنیم و در آن هر بارstart=0 را قرار می دهیم تا دیگر قسمت های بازی هم از کار بیفتد.

در قسمت دیگر بازی هر بار تغییری در امتیاز رخ داد از طریق usart باید این تغییر ارسال شود برای این کار بعد از اینکه از شرط باخت رد شدیم و امتیاز بازی اضافه شد این امتیاز جدید را ارسال می کنیم:

```
ftoa(w,1,str);
sprintf(lcd,"%5s",str);
puts(lcd);
putchar(13);
```

در مرحله ی چهارم زمانبندی بازی انجام می شود که باید با استفاده از اینتراپت تایمر صفر نوشته شود قبل از پرداختن به این قسمت باید تابع difficulty() مورد بررسی قرار گیرد:

```
float difficulty(void)
{
  time1=read_adc(0);
  t1=(time1*1.2)/1023+.3;
  time2=(int)(t1*10);
  time=time2/10+(time2%10)*.1;
  return time;
}
```

در این تابع با استفاده از adc مقدار ولتاژ روی پتانسیومتر خوانده شده و بااستفاده از روابط فوق آنرا ابتدا به عددی بین0.3تا1.5 تبدیل کرده وسپس در10 ضرب کرده و رند میکنیم و با رابطه ی آخر فقط یک رقم اعشار آنرا استخراج می کنیم و این مقدار را برمی گردانیم تا برای زمان بندی و امتیاز دهی استفاده شود.

حال ببینیم زمانبندی افتادن ردیف ها چگونه رخ می دهد:

بعد از اینکه از شرط باخت بیرون آمدیم باید زمانی صبر کنیم تا آپدیت بعدی شکل گیرد که این زمان از طریق adc توسط کاربر مشخص می شود و تابع difficulty() آنرا به ما می دهد پس برای این تاخیر تابعdelay() را می نویسیم:

برای این کار ابتدا تایمر صفر را فعال می کنیم و اینتراپت سرریز آنرا روی25ms تنظیم می کنیم و با توجه به زمانی که کاربر تنظیم کرده تعداد دفعات اینتراپت که برابر می شود با زمان تعیین شده ضرب در40 را در یک حلقه ی while صبر میکنیم.

```
void delay(void)
{
    loop=time*40;
    timing=0;
    while(timing<loop)
    {
    }
}
```

```
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
TCNT0=0x3D;
timing++;
}
```

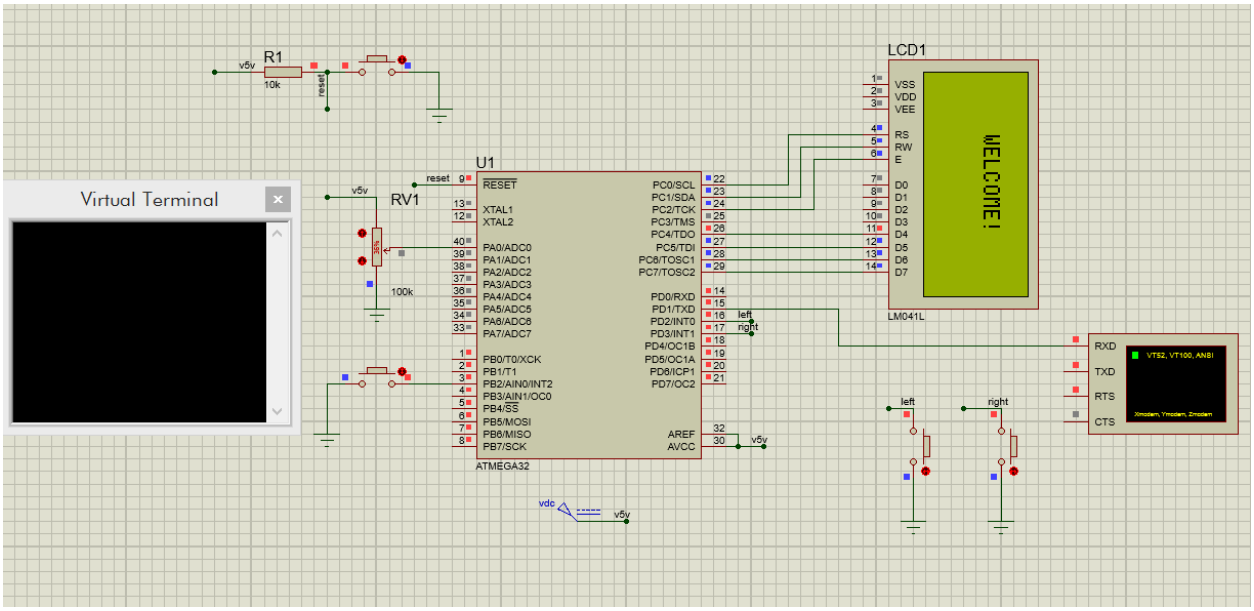
همانطور که مشاهده می شود آنقدر در حلقه ی while می ایستد تا تاخیر خواسته شده حاصل شود.

در مرحله ی آخر میخواهیم وقتی start=0 وارد تنظیم سختی بازی شویم و تا لحظه ی فشردن دکمه ی start در آن باقی بمانیم برای این کار از یک if استفاده می کنیم و درون آن به تنظیم شرایط خواسته شده می پردازیم.

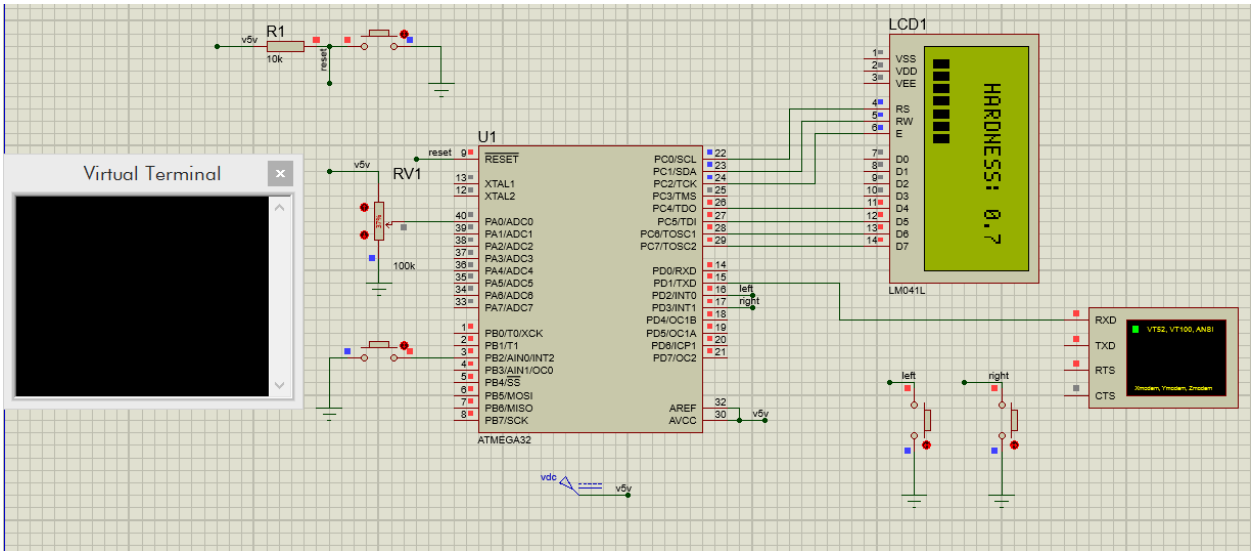
```
if(start==0)
{
  time=difficulty();
  ftoa(time,1,str);
  sprintf(lcd,"HARDNESS:%4s",str);
  lcd_clear();
  lcd_gotoxy(2,1);
  lcd_puts(lcd);
  for(i=0;i<time2;i++)
{
  lcd_gotoxy(i,3);
  lcd_putchar(0xff);
}
delay_ms(200);
}
```

همانطور که ملاحظه می شود ابتدا تابع difficulty() فراخوانی می شود و زمان تنظیم شده توسط کاربر تعیین می شود سپس این عدد ابتدا روی lcd نمایش داده میشود سپس با یک حلقه ی for با توجه با آسانی یا سختی بازی تعداد بلوک های پر شده کمترو بیشتر می شود . این نمونه برداری هر200میلی ثانیه انجام می شود.

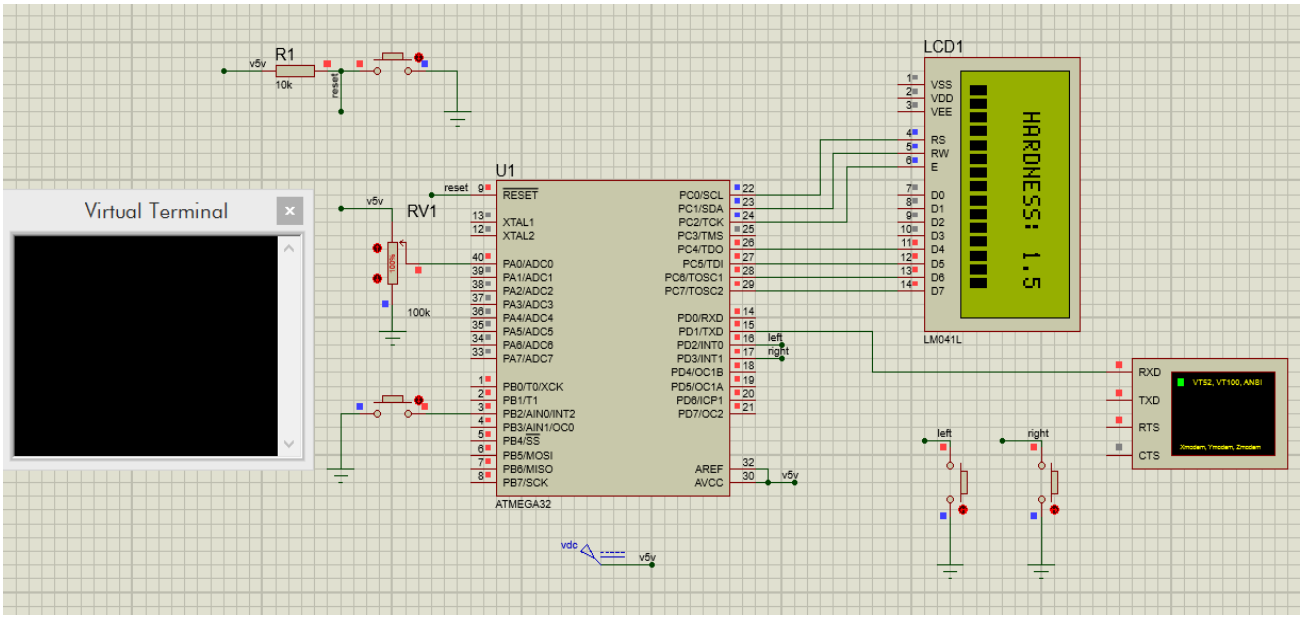
با راه اندازی بازی وارد صفحه ی خوش آمد گویی می شویم :



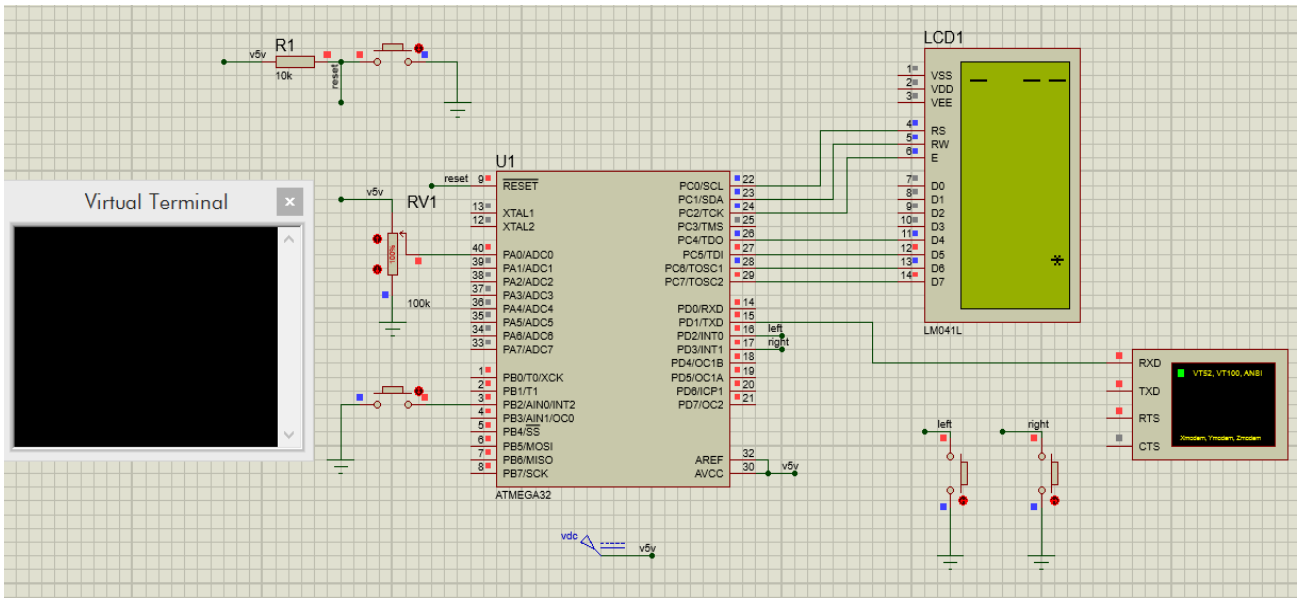
بعد از آن وارد صفحه ی تنظیم سختی بازی می شویم:



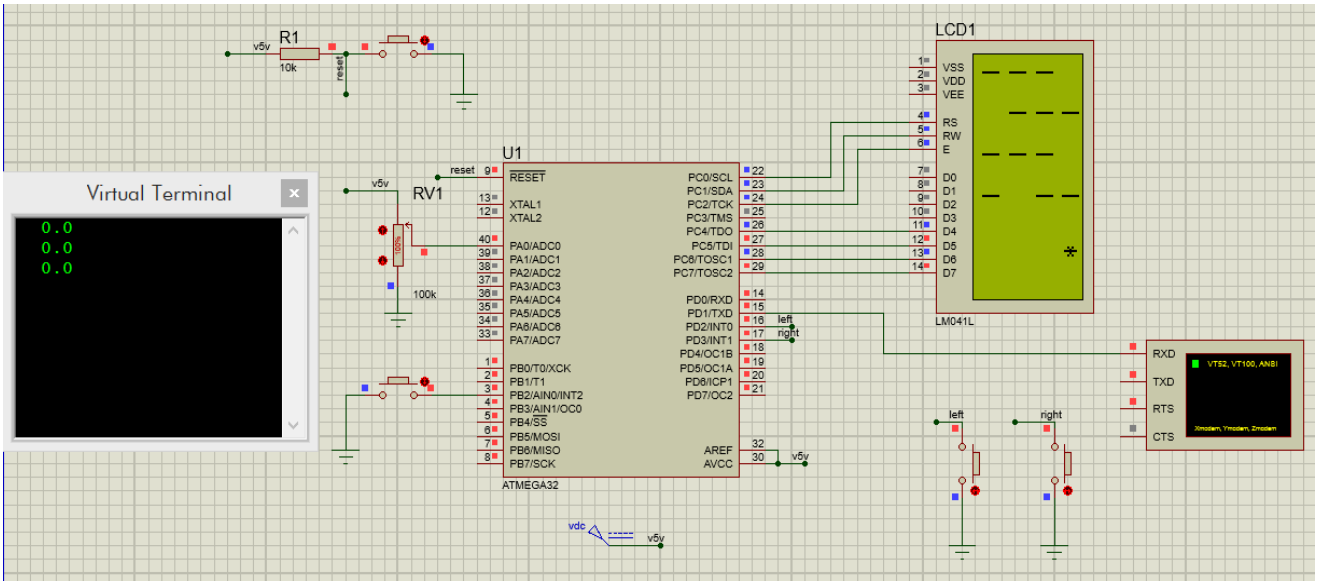
همانطوز که ملاحظه می شود یا سخت یا آسان شدن بازی تعداد بلوک های توپر تغییر می کند:

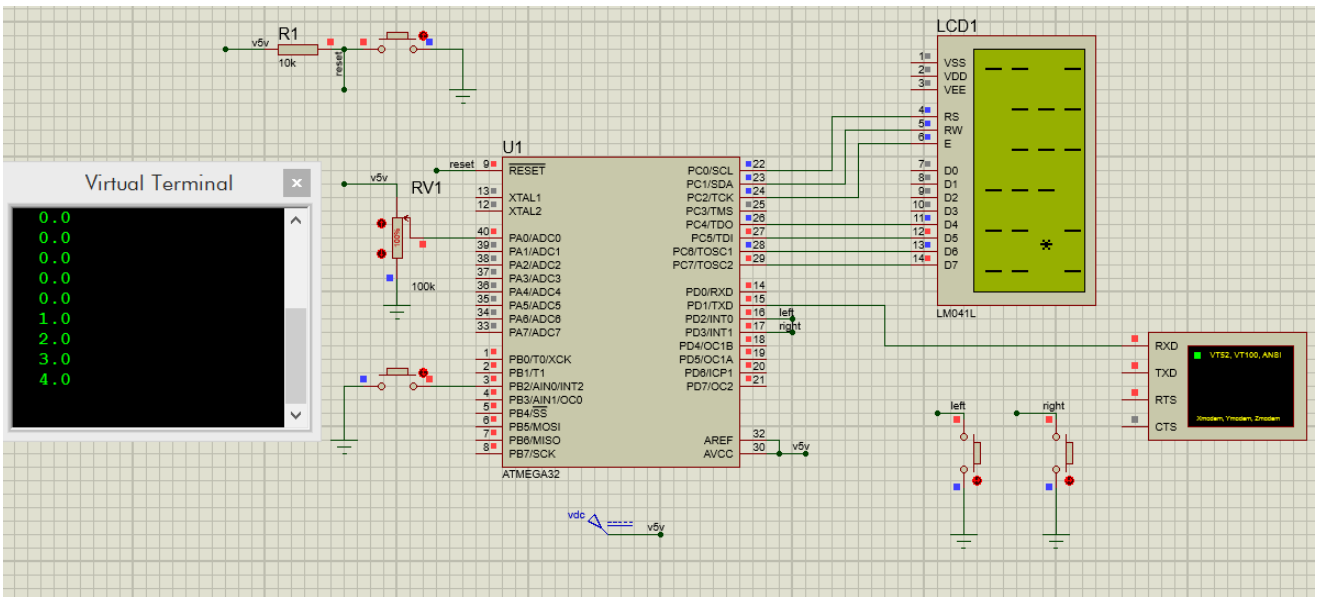


بعد از فشردن کلید start وارد بازی خواهیم شد:

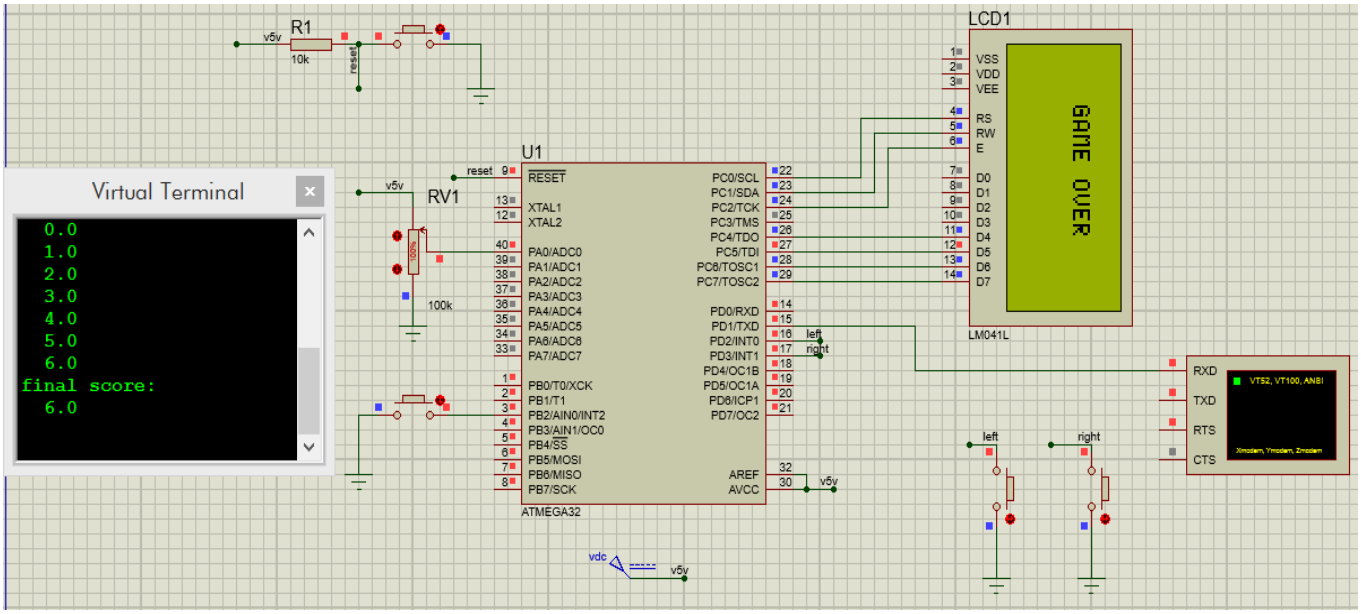


همانطور که مشاهده می شود در ابتدا یک ردیف وارد شده و سپس یکی یکی وارد صفحه می شوند تا صفحه پرشود همچنین هر لحظه امتیاز از طریق یوزارت اعلام می شود:

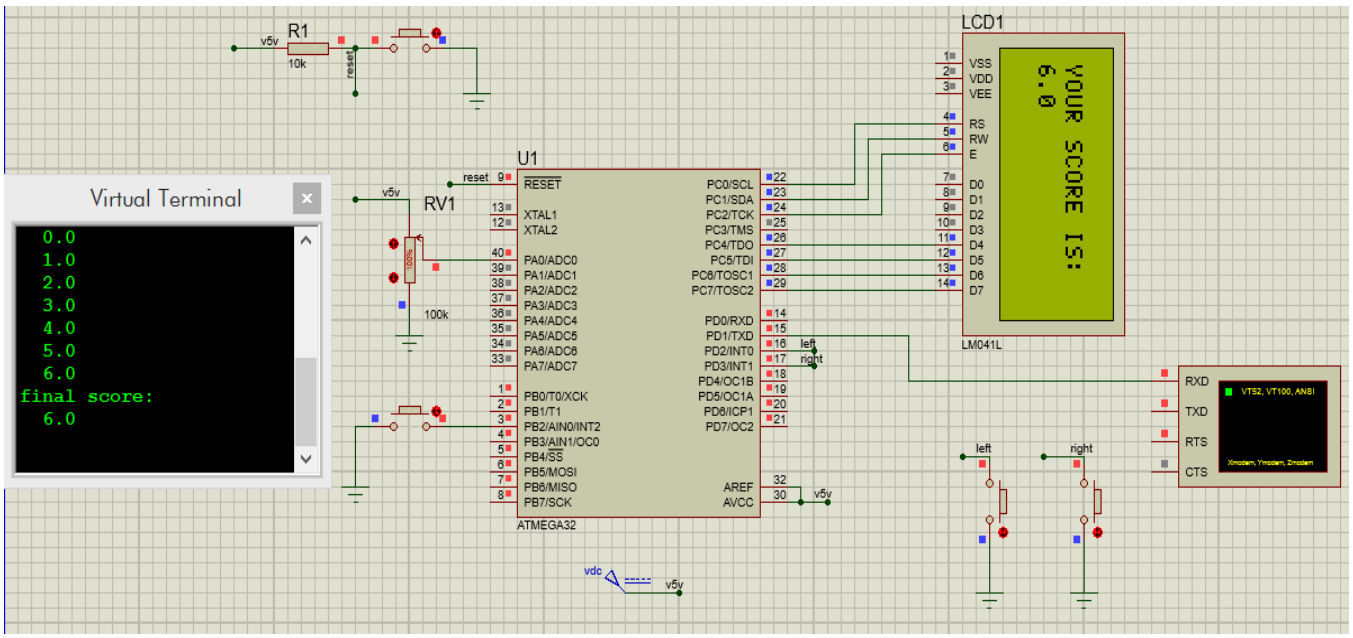




در صورت برخورد نشانگربا ردیف ها وارد تابع()gameover می شویم:



بعد از 4 ثانیه نمایش game over به صورت چشمک زن امتیاز نهایی نشان داده خواهدشد:



این امتیاز نهایی در virtual terminal نیز قابل ملاحظه است .

سپس بازی تا فشردن مجدد دکمه ی reset در همین وضع می ماند و در صورت فشردن این دکمه دوبار شاهد خوشامد گویی و ورود به تنظیم سختی بازی خواهیم بود.