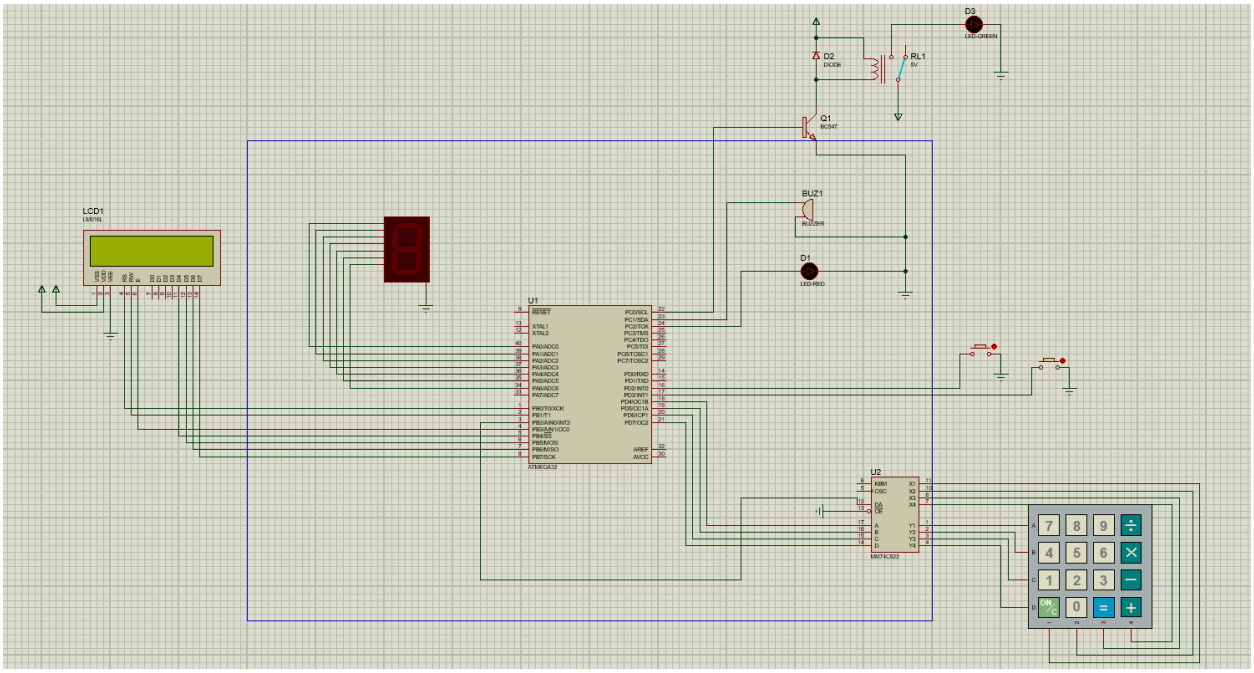
تمرین اجباری شماره3(میکروکنترلر پیشرفته)
احمدرضا عافی 40223052
درس مدار منطقی دکتر پورفرد

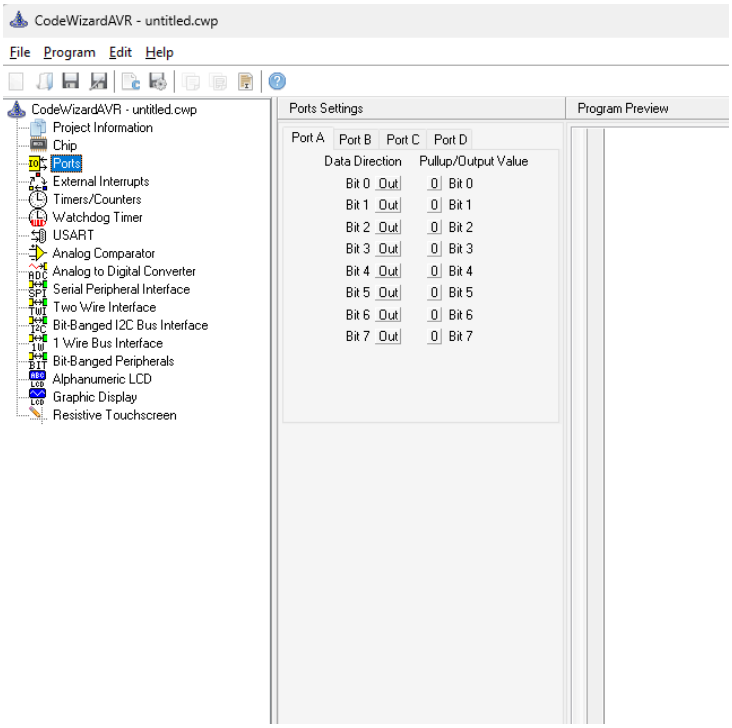
راهنمای فایل ها:

V2_1.c: نسخه اولیه نوشته شده
V2_2.c: نسخه بهبود یافته، بهینه شده و ماژولار
V2_3.c: نسخه کامپرکت شده سنتکسی
V2_1_t.c: نسخه برای محاسبه مدت هر بار ماندن در حلقه وایل میکرو برای نسخه اول نشوته شده
V2_2_t.c: نسخه برای محاسبه مدت هر بار ماندن در حلقه وایل میکرو برای نسخه دوم نشوته شده
V2_4.c: نسخه اضافه شده و کامل ترشده نرم افزار

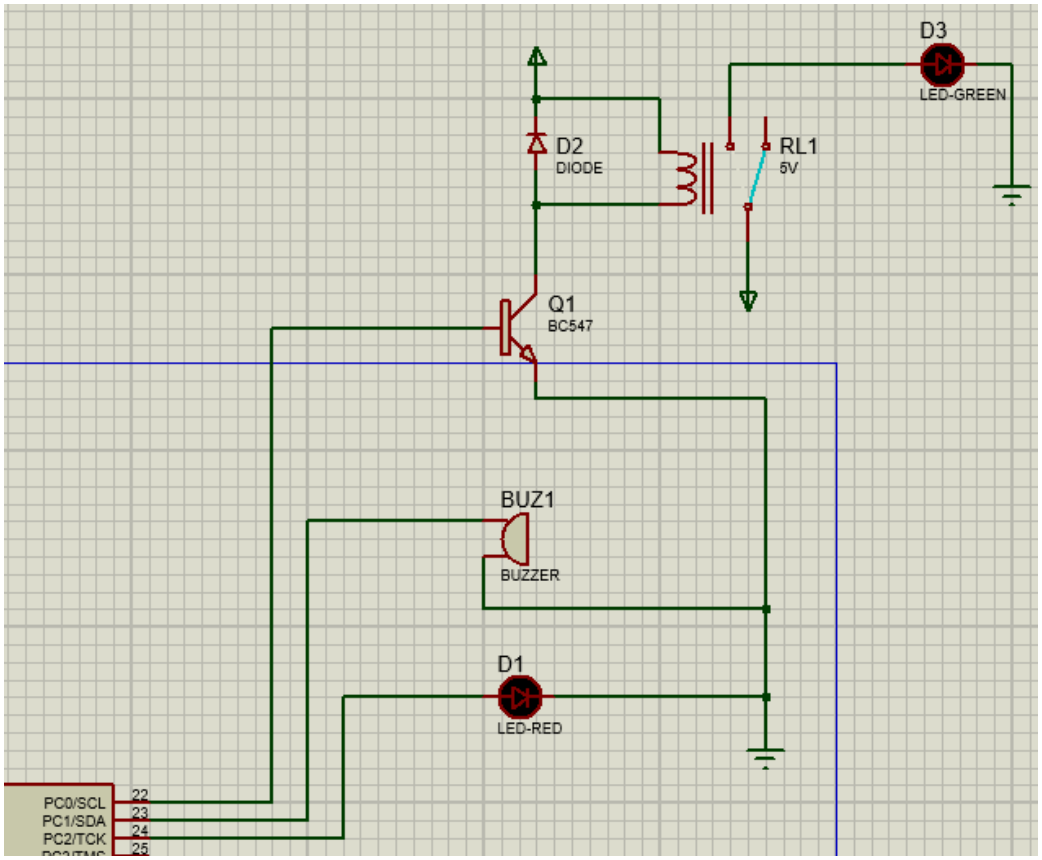
ابتدا مدار را در پروتیوس به صورت زیر ترسیم میکنیم:



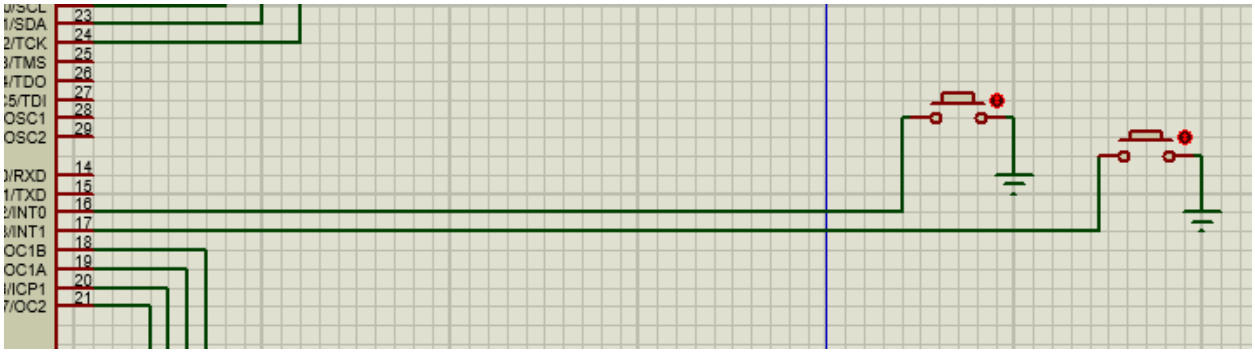
6 پورت seven segment را به ترتیب به پورت های پایه A میکرو کنترلر متصل میکنیم و در تنظیمات کد ویژن نیز این پورت ها را روی حالت خروجی میگذاریم:



3 پورت از پایه C میکرو کنترلر را به ترتیب به رله، بازر و LED وصل میکنیم و در تنظیمات کدویژن نیز به طور مشابه روی خروجی قرار میدهیم. دقت شود که در شبیه ساز از رله و بازر با ولتاژ کمتر از 5 ولت استفاده شوند که با تغییر ولتاژ پایه میکرو کار کنند. برای LED و باز پایه را مستقیم وصل میکنیم و برای رله پایه میکرو را به IC BC547 transistor وصل میکنیم و مدار را به صورت روبرو طراحی میکنیم تا با 1 شدن خروجی میکرو رله وصل و LED روشن شود:

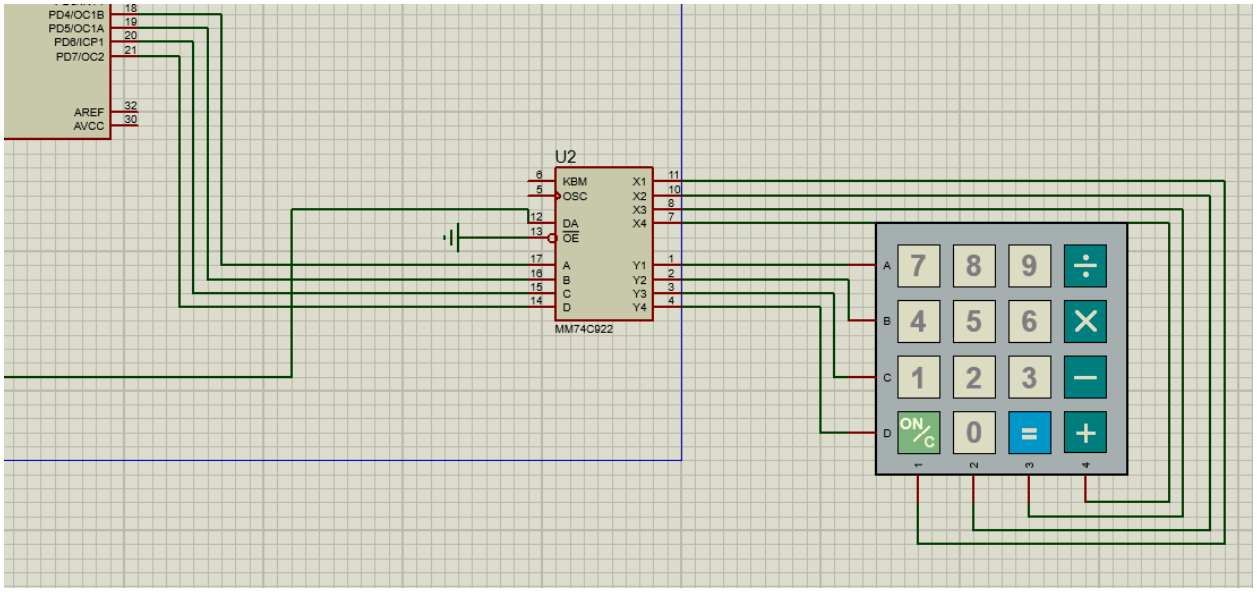


دو پورت اینتراپت در پایه D را نیز به دو کلید متصل میکنیم. کلید متصل به اینتراپت صفر برای ورود و خروج به منو با روش بانزگیری دوم (که در این روش بعد از برداشتن دست از روی کلید عملیات انجام میشود) و کلید متصل به اینتراپت یک برای حرکت در منو که به روش اول بانزگیری میشود(با نگه داشته شدن منو را اسکرول میکند).

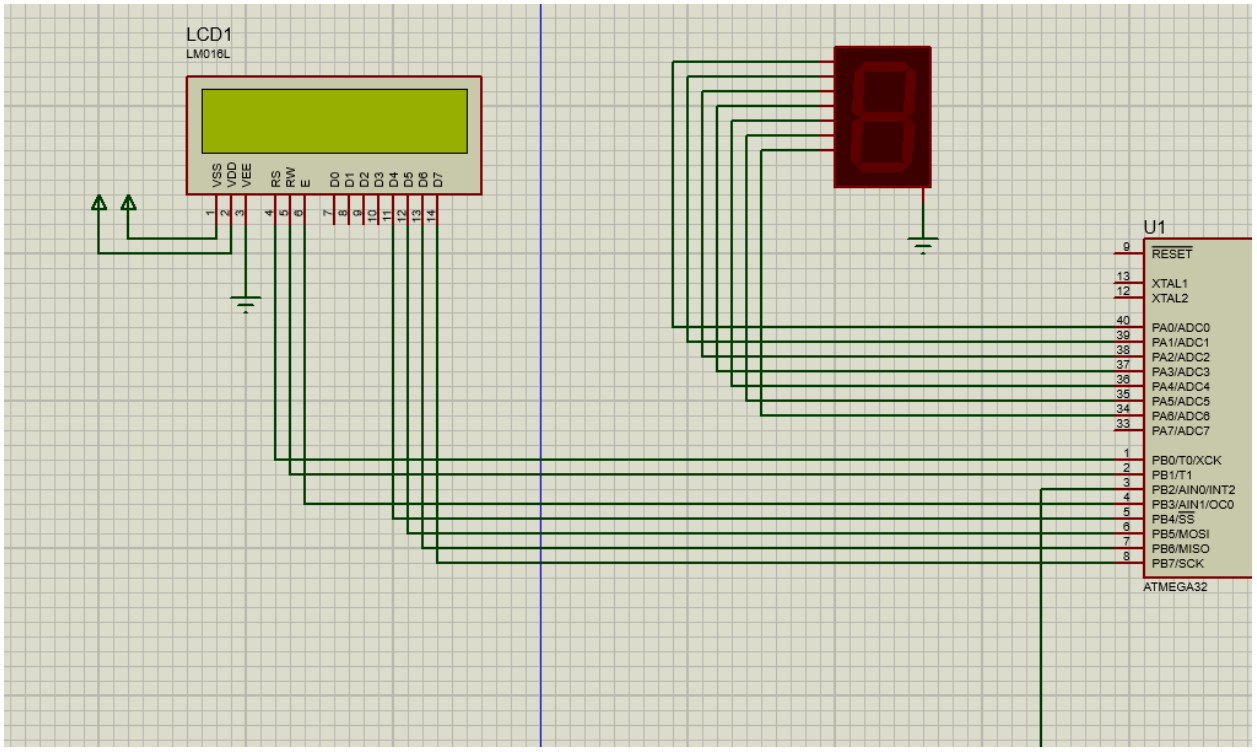


و پورت اینتراپت دوم در پایه B به پایه DA در IC74C922 متصل شده است.

4پین آخر پایه C نیز شامل 4و5و6و7 نیز به پورت های A تا D این IC متصل شده اند که خروجی بانزگیری شده کیبور را به میکرو ارسال میکنند. لازم است در تنظیمات میکرو پین های 4تا7 پایه C را در حالت ورودی قرار دهیم.
پایه های کیبورد را نیز در جهت X, Y به پایه های IC74C922 متصل کنیم.

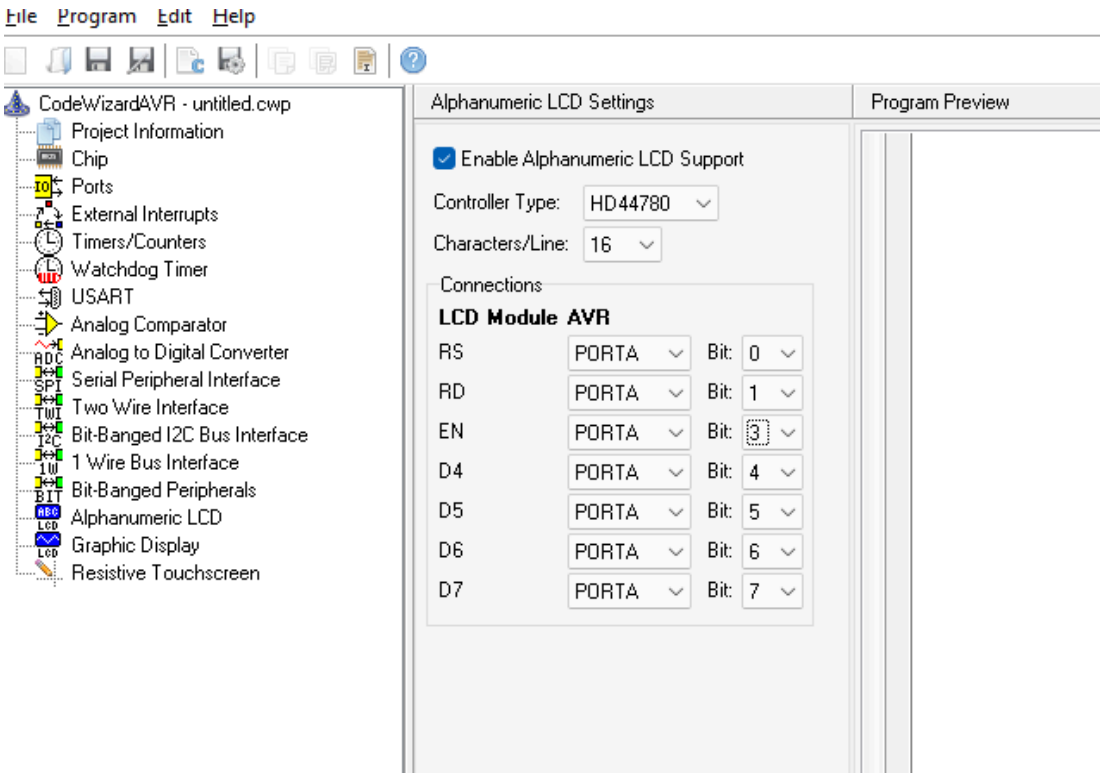


ال سی دی نیز به صورت زیر به میکرو وصل شده:



2پایه مربوط به روشن شدن ال سی دی و شدت روشنایی به منبع وصل شده اند. 3 پایه تنظیمات نیز به ترتیب به پین های 0و1و3 پایه B وصل شده اند. 4 پایه ارسال اطلاعات نیز به پین های 4 تا 7 پایه B متصل اند.

برای نمایش خروجی روی ال سی دی از کتاب خانه Buil_in استفاده شده که با تنظیمات اولیه LCD در کدویژن فعال میشود:



ولی نمایش خروجی روی سون سگمنت به کمک مپ کردن حالات مختلف روشن شدن و اعداد استفاده شده که به صورت زیر است:

```
unsigned char seven_seg[16] = {
    0x3F, // 0  ->  0011 1111
    0x06, // 1  ->  0000 0110
    0x5B, // 2  ->  0101 1011
    0x4F, // 3  ->  0100 1111
    0x66, // 4  ->  0110 0110
    0x6D, // 5  ->  0110 1101
    0x7D, // 6  ->  0111 1101
    0x07, // 7  ->  0000 0111
    0x7F, // 8  ->  0111 1111
    0x6F, // 9  ->  0110 1111
    0x77, // A  ->  0111 0111
    0x7C, // B  ->  0111 1100
    0x39, // C  ->  0011 1001
    0x5E, // D  ->  0101 1110
    0x79, // E  ->  0111 1001
    0x71  // F  ->  0111 0001
};
```

با مساوی قرار دادن پین های پایه A یک سری از سگمنت های مشخص در seven-segment روشن میشوند که عدد مورد نظر یا حرف مورد نظر را نشان میدهند. به دست آوردن بیت های روشن و خاموش هر حالت با ترتیب عددی سگمنت ها راحت قابل محاسبه است.

و اما خواندن کیبورد نیز به کمک مپ کردن خروجی 4 بیتی IC که پورت C ارسال شده ممکن است به این صورت که در صورت فشرده شدن کلید اینتراپت دوم که مخصوص کلید است فعال شده و مقدار 4 پین آخر پایه C را به کمک کد زیر میخواند:

```
key = (PIND >> 4) & 0x0F
```

در مرحله بعد از این کی استفاده شده و کاراکتر استخراج میشود به کمک آرایه راهنمای زیر:

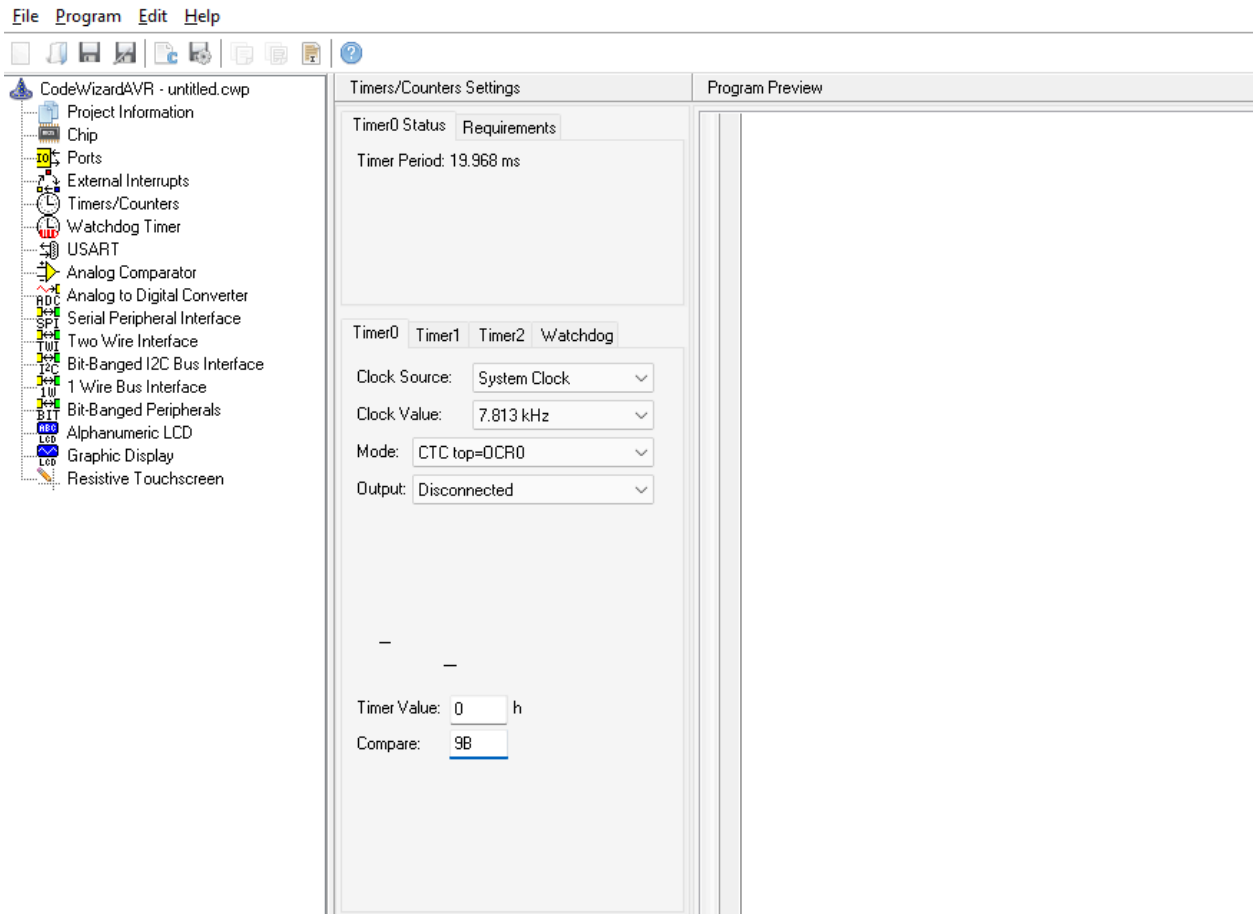
```
unsigned char keypad[16] = {'7','8','9','/','4','5','6','*','1','2','3','-',' ','0','=','+'}
```

واضح است که:

```
Character = keypad[key]
```

در مرحله بعد نیاز به تنظیم تایمر میباشد که برای بهینه سازی صرفای از یک تایمر 20ms استفاده میکنم که برای گرفتن بانس هروقت متغییر بزرگتر از 10 شد (یعنی 200ms) عمل کند و برای محاسبه 1 ثانیه هر وقت بزرگتر از 50 شد عمل کند.
برای تایمر 20ms به صورت زیر روی حالت کامپر مچ و فرکانس 7813 که درواقع 8000000/1024 میباشد قرار میدهیم و مقدار کامپر را روی 9B=155 که نشان میدهد:

```
19.968ms = (1/7813)*156
```



پس از انجام این تنظیمات وارد کد میشنویم.

در ابتدا متغییر های مورد نظر را تعریف میکنیم:

```
// Declare your global variables here
int keypad_touch=0, start=0, t1=0, t2=0, t3=0, counter=0, menu=1, option=1, clc=0, D1=1, D2=1, list=0;
unsigned char keypad[16] = {'7','8','9','/','4','5','6','*','1','2','3','-',' ','0','=','+'};
unsigned char key;
unsigned char password[5];
unsigned char correct_password[5] = "*14/";
unsigned char seven_seg[16] = {
    0x3F, // 0  ->  0011 1111
    0x06, // 1  ->  0000 0110
    0x5B, // 2  ->  0101 1011
    0x4F, // 3  ->  0100 1111
    0x66, // 4  ->  0110 0110
    0x6D, // 5  ->  0110 1101
    0x7D, // 6  ->  0111 1101
    0x07, // 7  ->  0000 0111
    0x7F, // 8  ->  0111 1111
    0x6F, // 9  ->  0110 1111
    0x77, // A  ->  0111 0111
    0x7C, // B  ->  0111 1100
    0x39, // C  ->  0011 1001
    0x5E, // D  ->  0101 1110
    0x79, // E  ->  0111 1001
    0x71  // F  ->  0111 0001
};
```

در قسمت بعد کدهای داخل اینتراپت ها و تایمر را مینویسیم:

```
55
56  // External Interrupt 0 service routine
57  interrupt [EXT_INT0] void ext_int0_isr(void)
58  {
59  // Place your code here
60  D2=0;
61
62  }
63
64  // External Interrupt 1 service routine
65  interrupt [EXT_INT1] void ext_int1_isr(void)
66  {
67  // Place your code here
68  // to change the menu
69  D1=0;
70  clc=0;
71  }
72
73  // External Interrupt 2 service routine
74  interrupt [EXT_INT2] void ext_int2_isr(void)
75  {
76  keypad_touch++;
77  key = (PIND >> 4) & 0x0F;
78
79  }
80
81  // Timer 0 output compare interrupt service routine
82  interrupt [TIM0_COMP] void timer0_comp_isr(void)
83  {
84  // Place your code here
85  t1++;
86  t2++;
87  t3++;
88
89  }
```

یک اینتراپت برای کلید 1 که وظیفه سلکت و خروج از گزینه انتخاب شده را به عهده دارد با متغییر D2 و اینتراپت دیگر برای کلید 2 که وظیف تغییر گزینه لیست را به عهده دارد با متغییر D1، اینتراپت سوم هم برای صفحه کلید و خواندن وضعیت پین های پایه D به که به آی سی بانز گیر صفحه کلید متصل اند.

در تایمرهم 3 متغییر تایم (یکی برای محاسبه یک ثانیه و گردش سون سگمنت، یکی برای بانز کلیدها، و یکی برای محاسبه یک ثانیه بازر) تعریف شده اند.

کد با یک سویچ کیس (switch case) آغاز میشود که در ابتدا متغییر آن یعنی menu روی حالت 1 قرار دارد. منو 1 صفحه درخواست پسورد را نشان میدهد و شمارش سون سگمنت را آغاز میکند:

```
while (1)
      {
      // Place your code here
      switch (menu){
      case 1:
      if(t1>49){
        t1=0;
        counter++;
        PORTA=seven_seg[counter];
        if(counter==15){
            counter=0;
        }
      }
      lcd_gotoxy(0,0);
      if(keypad_touch==0){
        lcd_putsf("Please Enter the password");
      }
```

در ادامه برای خواندن پسور و انجام مقایسه با پسور درست نیز یک سویچ کیس روی تعداد لمس کیبور تعبیه شده است.
در صورت یک بار لمس شدن کیبورد مقدار وردی داخل خانه اول پسور ریخته میشود در صورت دوبار خانه دوم و در صورت سه بار در خانه سوم و اگر کیبورد برای بار چهارم لمس شود مقدار ورودی کیبور را داخل متغییر آخر ریخته و با متغییر درست مقایسه میکند:

```
switch (keypad_touch){
      case 1:
          if(start==0){
              lcd_clear();  // Clear LCD line
              start++;
          }
          password[0]=keypad[key];
          lcd_puts(password);
          break;
      case 2:
          password[1]=keypad[key];
          lcd_puts(password);
          break;
      case 3:
          password[2]=keypad[key];
          lcd_puts(password);
          break;
      case 4:
          password[3]=keypad[key];
          lcd_puts(password);
          if(strcmp(password, correct_password) == 0){
              menu=3;
              PORTA=0x00;
              counter=0;
          }
          else{
              menu=2;
              t2=0;
              lcd_clear();
              lcd_putsf("wrong password");
          }
          keypad_touch=0;
          start=0;
          memset(password, 0, sizeof(password));  // Fills with null characters
          break;
}
break;
```

در مرحله بعدی پس از مقایسه دو حالت ممکن است رخ بدهد اگر پسورد درست با پسورد ورودی برابر نباشد. نمایشگر wrong password را نشان میدهد و تایمر t2 را صفر میکند و منو را به حالت دو میبورد. (مقدار ریخته شد در پسور ورودی را نیز با دستور memset صفر میکند.) در منو دو اگر 1 ثانیه زمان بگذرد دوباره منو را به حالت اول و ورودی پسور برمیگرداند:

```
274         case 2:
275           if(t2>49){
276               menu=1;
277           }
278           break;
```

از آنجا که اینتراپت روی 20ms تعریف شده و داخل اینتراپت مقدار متغییر هربار یکی عوض میشود بزرگتر شدن از 49 نشان دهنده گذشتن یک ثانیه است.

ولی اما اگر پسور به درستی وارد شده باشد سون سگمنت خاموش شده و منو به حالت 3 میرود که نشان دهنده منو رله، بازر و LED است:

در این منو ابتدا بانز دو کلید گرفته میشود:

```
279         case 3:
280           //taking the switch1 bounce method 1
281           if(D1==0){
282               if(PIND.3==0){
283                   t3=0;
284                   D1=1;
285                   option++;
286                   clc=0;
287               }
288           }
289           else{
290               if(PIND.3==0){
291                   if(t3>10){
292                       D1=0;
293                   }
294               }
295           }
296           //taking the switch2 bounce method 2
297           if(D2==0){
298               if(PIND.2==0){
299                   t3=0;
300               }
301               if(t3>10){
302                   if(list==0){
303                       list=option;
304                       clc=0;
305                   }
306                   else{
307                       list=0;
308                       PORTC=0x00;
309                       PORTA=0x00;
310                       clc=0;
311                       counter=0;
312                   }
313                   D2=1;
314               }
315           }
```

در اینجا بانز گیری کلید 1 و 2 به دو روش مختلف انجام شده کلید اول که برای تغییر منو است به این صورت است که محض فشرده شدن منو تغییر میکند و چک میشود اگر بعد از مدت تقریبی 220ms هنوز کلید فشرده شده باشد یک بار دیگر عملیات را انجام میدهد و منو عوض میشد.

اما کلید دو به این صورت است که بعد از فشرده شدن اگر به مدت حداقل 220ms فشرده شود سپس برداشته شود عمل میکند.

```
317     if(list==0){
318     if (option>3){
319         option=1;
320     }
321     switch (option){
322         case 1:
323             if(clc==0){
324                 lcd_clear();
325                 clc++;
326             }
327             lcd_gotoxy(0,0);
328             lcd_putsf("Relay <=");
329             lcd_gotoxy(0,1);
330             lcd_putsf("Buzzer");
331             break;
332         case 2:
333             if(clc==0){
334                 lcd_clear();
335                 clc++;
336             }
337             lcd_gotoxy(0,0);
338             lcd_putsf("Buzzer <=");
339             lcd_gotoxy(0,1);
340             lcd_putsf("LED");
341             break;
342         case 3:
343             if(clc==0){
344                 lcd_clear();
345                 clc++;
346             }
347             lcd_gotoxy(0,0);
348             lcd_putsf("LED <=");
349             break;
350     }
351     }
352     else{
```

متغییر option نشان دهنده این است که منو رله، بازر، LED روی کدام حالت است، و متغییر لیست نشان میدهد که اگر دکمه ورود به یه آپشن لیست فشرده شده آن آپشن کدام بوده و درصورت فشرده نشدن هیچکدام صفر است:

```
357                 lcd_gotoxy(0,0);
358                 switch (list){
359                     case 1:
360                         PORTC.0=1;
361                         PORTA=seven_seg[12];
362                         lcd_putsf("Relay On");
363                         break;
364                     case 2:
365                         if(counter==0){
366                             t1=0;
367                             PORTC.1=1;
368                             counter++;
369                         }
370                         else{
371                             if(t1>49){
372                                 PORTC.1=0;
373                             }
374                         }
375
376                         PORTA=seven_seg[11];
377                         lcd_putsf("Buzzer On");
378                         break;
379                     case 3:
380                         PORTC.2=1;
381                         PORTA=seven_seg[10];
382                         lcd_putsf("LED On");
383                         break;
384                 }
385             }
386
387
388
389         }
390         }
391 }
```

اگر وارد منو رله بشود، پین متصل به آن روشن شده که موجب اتصال رله و روشن شدن LED میشود.و سون سگمت C را نشان میدهد، LCD نیز روشن شده رله را نمایش میدهد.

اگر وارد منو بازر شود، پین متصل به بازر برای 1 ثانیه روشن میشود.و سون سگمنت B را نشان میدهد، LCD نیز روشن شدن بازر را نمایش میدهد.

اگر وارد منو LED شود، پین متصل به LED روشن میشود. و سون سگمنت A را نشان میدهد، LCD نیز روشن شدن LED را نمایش میدهد.

در مرحله بعد سعی در بهینه کردن کد و نوشتن کد به صورت ماژولار شده است.
ابتدا لازم است به این نکته اشاره کنیم که دستور switch case از مرتبه O(1) میباشد و حلقه if else نیز اگر شرط به صورت == تعریف شود به مرتبه O(1) توسط کامپایلر آپتیمایز میشود ولی اگر دستور به صورت < > باشد همانطور که در سطح گیت بررسی کردیم مدار آن پیچیده تر و مرتبه آن بالاتر است پس برای بهینه کردن این کد سعی شده که از حداقل متغییرها، حلقه استفاده شود و در صورت نیاز از شروط به صورت == و switch case استفاده شود.

```
// Global variables
int keypad_touch = 0, counter=0, t1=0, t2 = 0, menu = 1, option = 1, clc = 0, D1 = 1, D2 = 1, list = 0;
unsigned char keypad[16] = {'7', '8', '9', '/', '4', '5', '6', '*', '1', '2', '3', '-', ' ', '0', '=', '+'};
unsigned char key;
unsigned char password[5] = {0};
unsigned char correct_password[] = "*14/";
const unsigned char seven_seg[16] = {
    0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07,
    0x7F, 0x6F, 0x77, 0x7C, 0x39, 0x5E, 0x79, 0x71
};

// External Interrupt 0 ISR
interrupt [EXT_INT0] void ext_int0_isr(void) {
    D2 = 0;
}

// External Interrupt 1 ISR
interrupt [EXT_INT1] void ext_int1_isr(void) {
    D1 = 0;
    clc = 0;
}

// External Interrupt 2 ISR (Keypad input)
interrupt [EXT_INT2] void ext_int2_isr(void) {
    keypad_touch++;
    key = (PIND >> 4) & 0x0F;
}

// Timer 0 ISR
interrupt [TIM0_COMP] void timer0_comp_isr(void) {
    t1++;
    t2++;
}

```

تعریف متغییرها و اینتراپتها و تایمر به صورت قبل انجام شده فقط یک متغییر t3 کم شده زیرا بعد از ورود به منو سون سگمنت غیرفعال میشود از متغییر تایمرهمان برای محاسبه 1ثانیه صدا دادن بازر میتوان استفاده کرد.

```
void check_password() {
    if (strcmp(password, correct_password) == 0) {
        menu = 3;
        counter=0;
        PORTA = 0x00;
    } else {
        menu = 2;
        lcd_clear();
        lcd_putsf("Wrong password");
    }
    keypad_touch = 0;
    clc=0;
    t2=0;
    memset(password, 0, sizeof(password));
}
void seg_counter(){
    if(t1>49){
        t1=0;
        counter++;
        PORTA=seven_seg[counter];
        if(counter==15){
            counter=0;
        }
    }
}
void login(){
    lcd_gotoxy(0, 0);
    if (keypad_touch == 0) {
        lcd_putsf("Enter password:");
    } else {
    if (clc == 0) { lcd_clear(); clc = 1; }
    password[keypad_touch - 1] = keypad[key];
    lcd_puts(password);
    if (keypad_touch == 4){
        password[keypad_touch - 1] = keypad[key];
        check_password();
        }
    }
}
```

فانکشن check_passweod برای صحت سنجی پسورد وارد شده با مقدار درست استفاده میشود که پس از وارد شدن پسورد صدا زده میشود. و مشابه قبل در صورت درست بودن منو را به حالت 3 و در صورت اشتباه بودن منو را به حالت دو منتقل میکند.
فانکشن seg_counter فرآیند شمارش توسط سون سگمنت را انجام میدهد.
فانکشن login وظیفه گرفتن ورودی از کاربر توسط کیبورد را به عهده دارد که در اینجا بجای سوییچ کیس به صورت تجمیعی تر با یک حلقه شرط if تعریف شده است.

```
void bounce1(){
    if(D1==0){
        if(PIND.3==0){
            t2=0;
            D1=1;
            option++;
            if(option==4){option=1;}
            clc=0;
        }
    }
    else{
        if(PIND.3==0){
            if(t2>10){
                D1=0;
            }
        }
    }
}
void bounce2(){
    if(D2==0){
        if(PIND.2==0){
            t2=0;
        }
        if(t2>10){
            D2=1;
            if(list==0){
                list=option;
                clc=0;
            }else{
                list=0;
                PORTC=0x00;
                PORTA=0x00;
                counter=0;
                clc=0;
                t2=0;
            }
        }
    }
}
```

در ادامه دو تابع مشابه کد مرحله قبل برای گرفتن بانز کلیدها استفاده شده که در هنگام ورد به منو هربار صدا زده میشوند.

```
void handle_options(){
    lcd_gotoxy(0, 0);
    if (clc == 0) { lcd_clear(); clc = 1; }
    switch(option){
        case 1:
            lcd_putsf("Relay <=\nBuzzer");
            break;
        case 2:
            lcd_putsf("Buzzer <=\nLED");
            break;
        case 3:
            lcd_putsf("LED <=");
            break;
    }
}
```

تابع نشان دهنده منو به صورت بالا تعریف شده

```
void handle_list(){
    if (clc == 0) { lcd_clear(); clc = 1; }
    lcd_gotoxy(0, 0);
    switch(list){
        case 1:
            PORTC.0 = 1;
            PORTA = seven_seg[12];
            lcd_putsf("Relay On");
            break;
        case 2:
            PORTA = seven_seg[11];
            lcd_putsf("Buzzer On");
            if(counter==0){
                t1=0;
                PORTC.1=1;
                counter++;
            }else{
                if(t1>49){
                    PORTC.1=0;
                    }
                }
            break;
        case 3:
            PORTC.2 = 1;
            PORTA = seven_seg[10];
            lcd_putsf("LED On");
            break;
    }
}
```

تابع مدیریت هریک از گزینه های منو به صورت بالا تعریف شده.
و در نهایت تابع مدیریت فراخوانی هرکدام از حالات و مدیریت حالات مختلف به صورت پایین تعریف میشود:

```
void handle_menu() {
    switch (menu) {
        case 1:
            seg_counter();
            login();
            break;
        case 2:
            if (t2 > 49) menu = 1;
            break;
        case 3:
            //taking the switch1 bounce method 1
            bounce1();
            //taking the switch2 bounce method 2
            bounce2();
            if (list == 0) {
                handle_options();
            } else {
                handle_list();
            }
            break;
    }
}
```

که خود این تابع در حلقه while(1) صدا زده میشود.

خب حال قصد دارم با مکانیزم ساده ای که پیاده کرده ام مقدار زمان ماندن در هر مرتبه حلقه وایل را به طور تقریبی اندازه گیری کنم: برای این کار به ابتدا و انتهای حلقه این مورد را اضافه کردم:

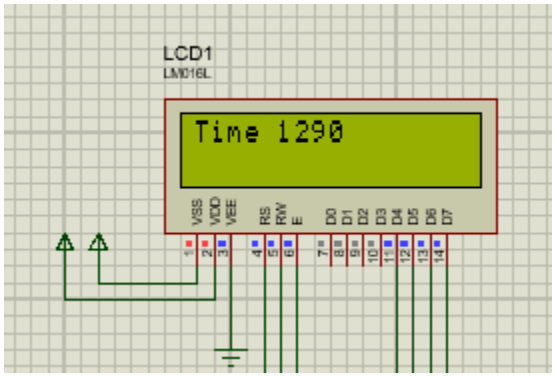
```
while (1) {
    counter3++;
    loop_start_time = t3;
    handle_menu();
    loop_end_time = t3;
    loop_duration = loop_end_time - loop_start_time;
    time+=loop_duration;
    if(counter3==30000){
        lcd_clear();
        sprintf(str, "Time %d", time);
        lcd_puts(str);
        delay_ms(10000);
    }
}
```

در اینجا من ابتدا یک تایمر تعریف کردم که در اینتراپت تایمر هر مرتبه یکی اضافه میشود سپس تایم را قبل و بعد از اجرای کد اندازه میگیرم. در هر مرتبه اجرای کد و ورود به حلقه نیز مقدار counter3 را یک عدد زیاد میکنم در نهایت بعد از 30000 مرتبه ورود به حلقه وایل مجموع همه duration ها را روی LCD نمایش میدهم که با رابطه ساده:

Each time in the loop = (time/30000)*20ms

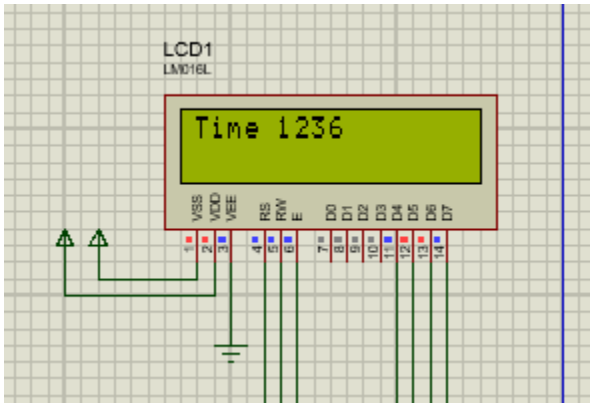
زمان هربار در حلقه while ماندن و انجام عملیات ها محاسبه میشود:

برای کد اول که به صورت غیرماژولار و کمتر بهینه نوشته شده مقادیر به صورت زیر است:



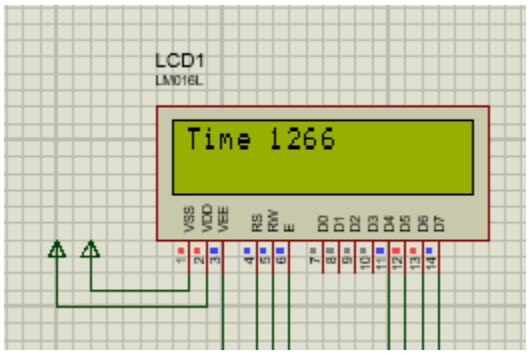
با انجام محاسبات مدت زمان ماندن در حلقه while برابر 0.86ms است که این مقدار تقریبی است و وابسته به وارد شدن در بخش های مختلف کد است ولی میانگین و دید خوبی به ما میدهد.

برای کد دوم که به صورت بهینه شده و ماژولار نوشته شده مقادیر به صورت زیر است:



با انجام محاسبات مدت زمان ماندن در حلقه while برابر 0.824ms است که این مقدار تقریبی است و وابسته به وارد شدن در بخش های مختلف کد است ولی میانگین و دید خوبی به ما میدهد.

برای ورژن سوم که کامپکت شده و بهینه ترین ورژن است:



با انجام محاسبات مدت زمان ماندن در حلقه while برابر 0.844ms است که این مقدار تقریبی است و وابسته به وارد شدن در بخش های مختلف کد است ولی میانگین و دید خوبی به ما میدهد.

من برحسب علاقه ورژن دیگری از برنامه را طراحی کردم که بنظرم کاربری تر می آمد به اینصورت که بجای رمز عبور فیکس شده قابلیت Register و Login را به اپلیکیشن اضافه کردم.
به این صورت که در ابتدا منو Register و Login ظاهر میشود که با انتخاب Register یوزرنیم و پسور از کاربر خواسته میشود. یوزرنیم حداکثر 9 کاراکتری و پسور حداکثر 4 کاراکتری است. با فشاردن دکمه = نشان دهنده انتهای ورودی است پس میتوان یوزرنیم و پسور با کاراکتر کمتر نیز وارد کرد.
سپاس با ورود در منو Login با وارد کردن یوزرنیم و پسور در صورت صحت و از قبل ریجستر شده بودن به منو رله، بازر، LED وارد میشود که در اینجا گزینه Logout نیز قرار گرفته شده. توضیحات بیشتر در ویدیو و کد ورژن چهارم قرار گرفته شده است.