«به نام خدا»

نام: فاطمه زهرا بخشنده استاد: دکتر میثم عبداللهی

شماره دانشجویی: 98522157

گزارش تمرین ششم:

توضیحات:

فایل مربوط به هر سوال تمرین، در این فایل زیپ قرار داده شده است. سوال اول به زبان اسمبلی، و سوال های بعدی با C نوشته شده اند.

توضیحات الگوریتم هر سوال:

سوال اول:

در این سوال می خواهیم اعداد فیبوناتچی یک و دو رقمی را روی دو 7-segment با فاصله یک ثانیه از هم نمایش دهیم.

ابتدا تعاریف اولیه را انجام می دهیم. PORTC و PORTD را به عنوان خروجی قرار می دهیم، پورت C رقم یکان را روی 7-segment نشان می دهد و پورت D دهگان.

رجیستر های های کمکی R18 و R19 و R20 را به عنوان متغیر اولیه برای محاسبه اعداد فیبوناتچی تعریف می کنیم.

```
;BAKHSHANDE
.include "m32def.inc"
        LDI     R16,HIGH(RAMEND)
        OUT     SPH,R16
        LDI     R16,LOW(RAMEND)
        OUT     SPL,R16

        LDI     R16,0xFF
        OUT     DDRC,R16        ;PORTC as output
        OUT     DDRD,R16        ;PORTD as output

        OUT     PORTA,R16       ;Enabling Pullup Resistor on PORTA
        LDI     R16,0x00
        OUT     DDRA,R16        ;PORTA as input'

INIT:   LDI     R18 , 1
        LDI     R19 , 0
        LDI     R20 , 0
```

سپس الگوریتم فیبوناتچی را پیاده سازی می کنیم. عدد مورد نظر در R16 قرار میگیرد.

```
FIB:    MOV     R16 , R18
        LDI     R17 , 0
        MOV     R20 , R18
        ADD     R18 , R19
        MOV     R19 ,R20

YEKAN:  CPI     R16, 10
        BRGE    DAHGAN
        RJMP    DISPLAY

DAHGAN: LDI     R21,10
        SUB     R16,R21
        INC     R17
        RJMP    YEKAN

CHECK:  CPI     R16,9
        BREQ    DISPLAY
        RJMP    FIB
```

در Yekan و Dahgan چک میکنیم اگر مقدار عدد بزرگ تر از 10 بود تا زمانی که مقدار R16 به پایین 10 برسد هر دفعه مقدار R17 را یکی زیاد و مقدار R16 را ده تا کم کند. به این صورت در آخر رقم دهگان در R17 و رقم یکان در R16 قرار میگیرد.

حالا باید عدد را نمایش دهیم. یکان را به پورت C منتقل کرده و دهگان را به پورت D و سپس ONESEC_DEL را صدا میزنیم تا دقیقا یک ثانیه delay ایجاد کند.

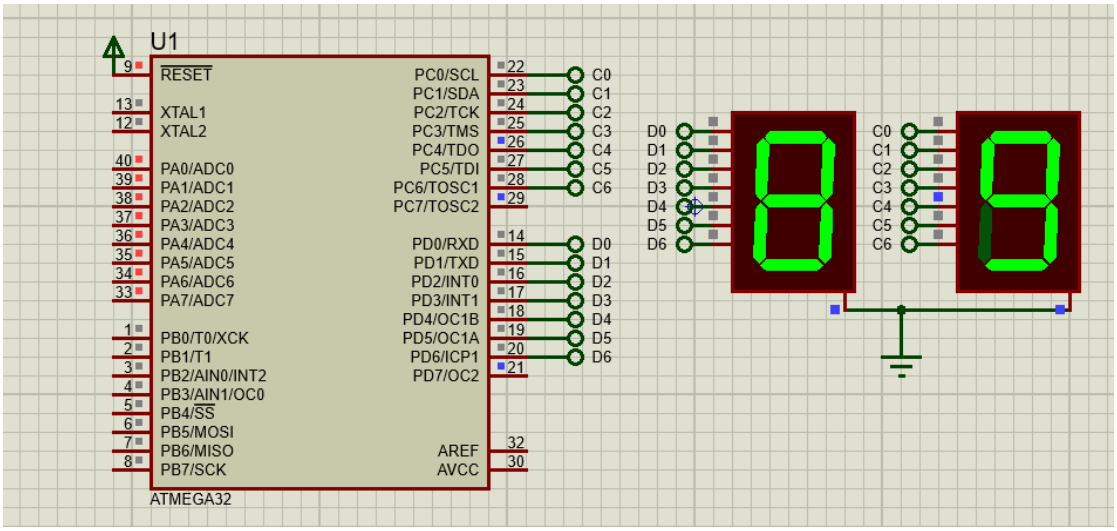
```
DISPLAY:
        MOV  R22,R16  ;Yekan
        CALL CONVERT
        OUT  PORTC,R22

        MOV  R22,R17  ;Dahgan
        CALL CONVERT
        OUT  PORTD,R22

        CALL ONESEC_DEL
        RJMP CHECK


ONESEC_DEL:
        LDI     R27, 8   ; One clock cycle;
DELAY1: LDI     R28, 125 ; One clock cycle
DELAY2: LDI     R29, 250 ; One clock cycle
DELAY3: DEC     R29      ; One clock cycle
        NOP              ; One clock cycle
        BRNE    DELAY3   ; Two clock cycles when jumping to Delay3,
                         ;1 clock when continuing to DEC
        DEC     R28      ; One clock cycle
        BRNE    DELAY2   ; Two clock cycles when jumping to Delay2,
                         ;1 clock when continuing to DEC
        DEC     R27      ; One clock Cycle
        BRNE    DELAY1   ; Two clock cycles when jumping to Delay1,
                         ;1 clock when continuing to RET
        RET
```

در ONESEC_DEL ما دقیقا به اندازه ای loop میزنیم که تعداد clock cycle های ایجاد شده 1000000 شود و یک ثانیه زمان ببرد. (با توجه به کلاک cpu)

برای نشان دادن هر رقم نیز در Convert پورت مورد نظر را معادل عدد مناسب هگز آن رقم، برای نشان دادن روی 7-segment قرار میدهیم.

```
CONVERT:
        CPI     R22,0
        BRNE    C1
        LDI     R22,0x3F
        RET
C1:     CPI     R22,1
        BRNE    C2
        LDI     R22,0x06
        RET
C2:     CPI     R22,2
        BRNE    C3
        LDI     R22,0x5B
        RET
C3:     CPI     R22,3
        BRNE    C4
        LDI     R22,0x4F
        RET
C4:     CPI     R22,4
        BRNE    C5
        LDI     R22,0x66
        RET
C5:     CPI     R22,5
        BRNE    C6
        LDI     R22,0x6D
        RET
C6:     CPI     R22,6
        BRNE    C7
        LDI     R22,0x7D
        RET
```

همین روند را تا رسیدن به عدد آخر فیبوناتچی دورقمی یعنی 89 ادامه میدهیم. و از آن به بعد 89 به صورت ثابت نمایش داده میشود و تغییر نمی کند.



سوال دوم:

در این سوال، میخواهیم یک ماشین حساب طراحی کنیم.

ابتدا اعمال ریاضی و CLEAR را در آن DEFINE میکنیم.

پورت D و C را تعریف کرده و Lcd به اندازه 16 تعریف می کنیم.

با استفاده از تابع keypad کلید های فشرده شده را دریافت میکنیم.

```
unsigned char keypad() {
    unsigned char res = 255;

    KEYPAD_R1 = 1;
    KEYPAD_R2 = 0;
    KEYPAD_R3 = 0;
    KEYPAD_R4 = 0;

    delay_ms(5);
    if (KEYPAD_C1)
        res = KEYPAD_NUM7;
    else if (KEYPAD_C2)
        res = KEYPAD_NUM8;
    else if (KEYPAD_C3)
        res = KEYPAD_NUM9;
    else if (KEYPAD_C4)
        res = DIV;

    KEYPAD_R1 = 0;
    KEYPAD_R2 = 1;
    KEYPAD_R3 = 0;
    KEYPAD_R4 = 0;
```

اگر عدد دریافتی بین 0 تا 9 باشد، آن را رقم در نظر گرفته و در آرایه میریزیم. اگر operator دریافت کردیم، آن را در آرایه مخصوص به operator میریزیم.

```
if (key != 255) {
    while (keypad() != 255);
    delay_ms(20);
    if(key >=0 && key <=9)
    {
    lcd_putchar(key + 48);
    number[count] *= 10;
    number[count] += key;
    OP = 0;
    }
```

اگر کاربر دکمه = را بزند ابتدا lcd را clear میکنیم:

```
else if(key == '=')
{
    lcd_clear();
    calculate();
    for(i=0;i<5;i++)
        number[i]=0;
    count = 0;
    OP = 1;
}
```

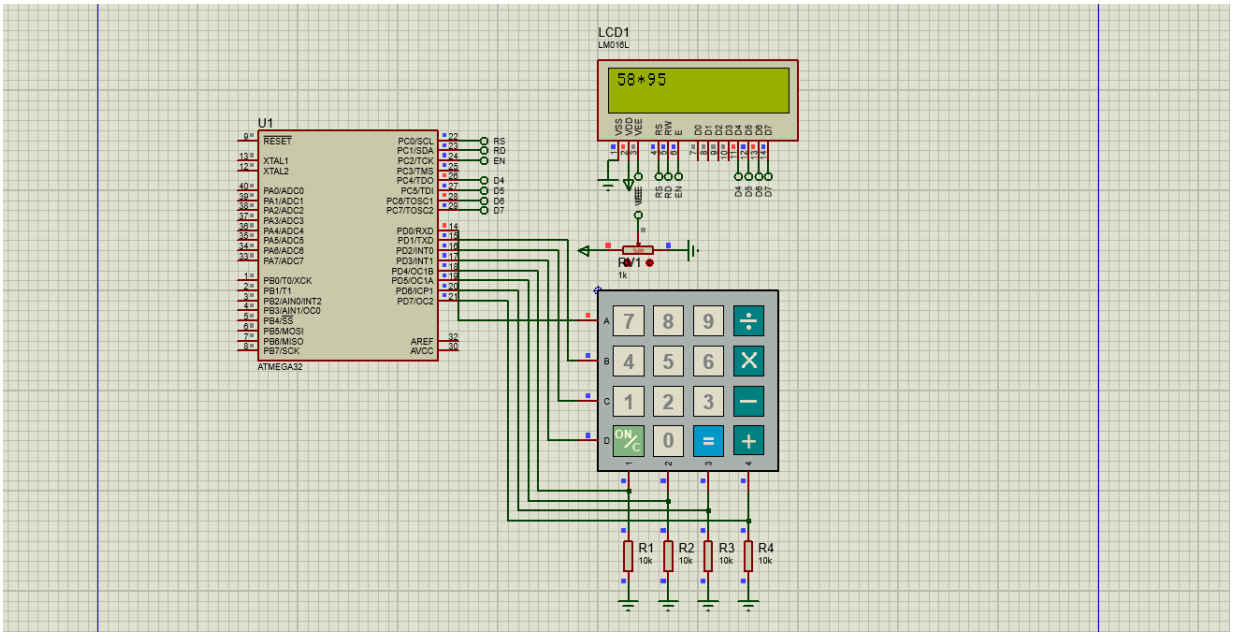
و سپس تابع زیر را صدا میزنیم تا عملیات محاسبه را انجام دهیم، محاسبات را با رعایت تقدم عملیات انجام میدهیم:

```
void calculate()
{
    char i,j,k;
    char operators[4]={'/','*','-','+'};
    unsigned char buffer[16];
    int temp;
    for(k=0;k<4;k++)
    {
        for(i=0;i<count;i++)
        {
            if(operator[i] == operators[k])
            {
                if(k==0)
                    number[i] = number[i] / number [i+1];
                else if(k==1)
                    number[i] = number[i] * number [i+1];
                else if(k==2)
                    number[i] = number[i] - number [i+1];
                else if(k==3)
                    number[i] = number[i] + number [i+1];
                for(j=i;j<count - 1;j++)
                {
```

اگر c وارد شود lcd پاک میشود و آرایه ها خالی میشوند.

```
}
else if(key == 'c')
{
    lcd_clear();
    for(i=0;i<5;i++)
        number[i]=0;
    count = 0;
    OP = 1;
}
```

خروجی:



سوال سوم:

در این سوال، باید یک counter داشته باشیم که از 0 تا 9 به فاصله یک ثانیه بشمارد و دوباره از اول شروع کند. و با یک دکمه آن را stop یا resume کنیم. یک array برای تبدیل اعداد 0 تا 9 به معادل عدد مناسب هگز آن رقم، برای نشان دادن روی 7-segment تعریف میکنیم. متغیر second برای ثانیه و متغیر button را به معنی فشرده شدن کلید برای stop تعریف میکنیم.

```
unsigned int second = 0;
unsigned int button = 0;

char convert[] = {
    0x40,
    0x79,
    0x24,
    0x30,
    0x19,
    0x12,
    0x02,
    0x78,
    0x00,
    0x10};
```

با استفاده از interrupt ها، یک تایمر تعریف می کنیم تا هر یک ثانیه، در صورتی که متغیر button صفر بود، متغیر second را زیاد کند. Second باید بین 0 تا 9 بماند پس mod آن را به 10 میگیریم.

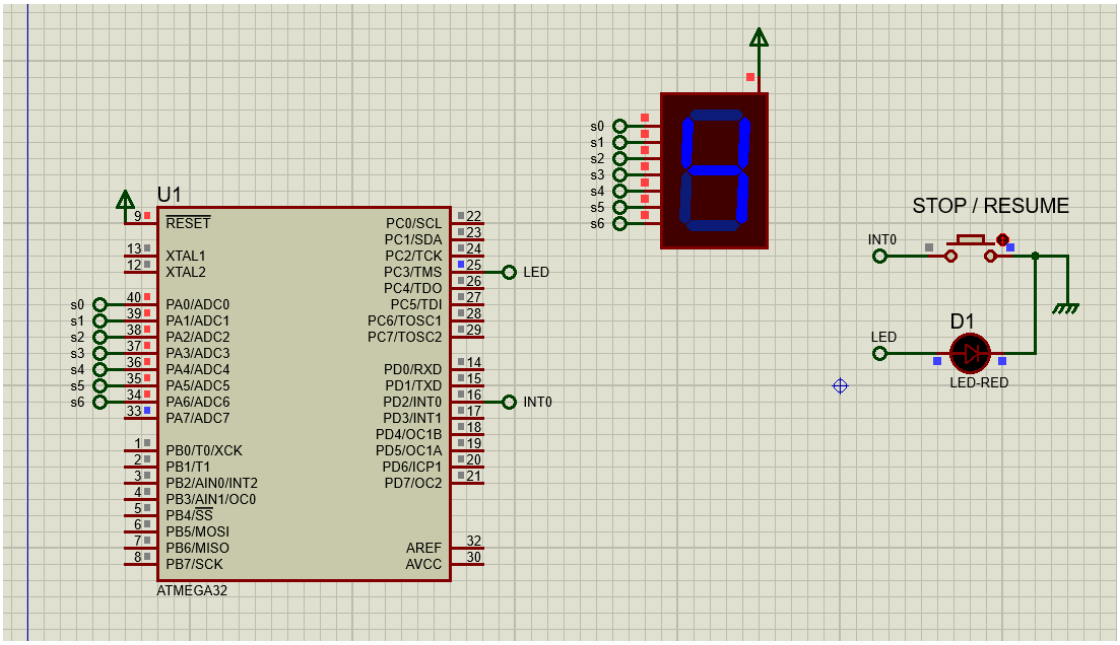
یک interrupt هم برای چک کردن فشرده شدن کلید تعریف میکنیم. کلید را به پورت INT0 متصل میکنیم. در صورت صفر بودن بیت سوم پورت C متصل به LED، یعنی کلید فشرده شده، پس متغیر button را switch میکنیم. اگر یک بود صفر، و اگر صفر بود یک میشود.

```
interrupt [EXT_INT0] void ext_int0_isr(void) {
    if (!PORTC.3)
        button = 1 - button;
}

// Timer1 output compare A interrupt service routine
interrupt [TIM1_COMPA] void timer1_compa_isr(void) {

    if (button == 0)
        second = (second + 1) % 10;
}
```

در main پس از config کردن تایمر و interrupt ، عدد مورد نظر را به پورت A میدهیم که به 7-segment متصل است.

```
    MCUCSR = 0x00;
    GIFR = 0xE0;

    // Timer 1 config
    TCCR1A = 0x00;
    TCCR1B = 0x0C;
    TCNT1H = 0x00;
    TCNT1L = 0x00;
    ICR1H = 0x00;
    ICR1L = 0x00;
    OCR1AH = 0x7A;
    OCR1AL = 0x12;
    OCR1BH = 0x00;
    OCR1BL = 0x00;
    TIMSK = 0x12;

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

    while (1)
    {
        PORTA = convert[second];
    }
}
```

خروجی:



سوال چهارم:

در این سوال، باید رطوبت هوا را از سنسور DHT11 گرفته و روی lcd نمایش دهیم. اگز کمتر از 40 درصد یا بیشتر از 60 درصد بود led روشن میشود.

ابتدا پورت B را خروجی میگذاریم. آن را به LED متصل میکنیم. یک lcd با اندازه 16 تعریف میکنیم. و Humadity را روی آن مینویسیم.

```
void main(void)
{
    DDRB = 0xFF;

    lcd_init(16);
    lcd_gotoxy(0, 0);
    lcd_puts("Humadity: ");

    while(1)
    {
        DDRA |= (1<<DHT11_PIN);
        PORTA &= ~(1<<DHT11_PIN);
        delay_ms(20);
        PORTA |= (1<<DHT11_PIN);
        DDRA &= ~(1<<DHT11_PIN);
        while(PINA & (1<<DHT11_PIN));
        while(!(PINA & (1<<DHT11_PIN)));
        while(PINA & (1<<DHT11_PIN));

        H = read_h();
```

سپس رطوبت را از سنسور DHT11 دریافت می کنیم، که به بیت یکم پورت A متصل است. آن را جلوی کلمه Humadity، مینویسم.

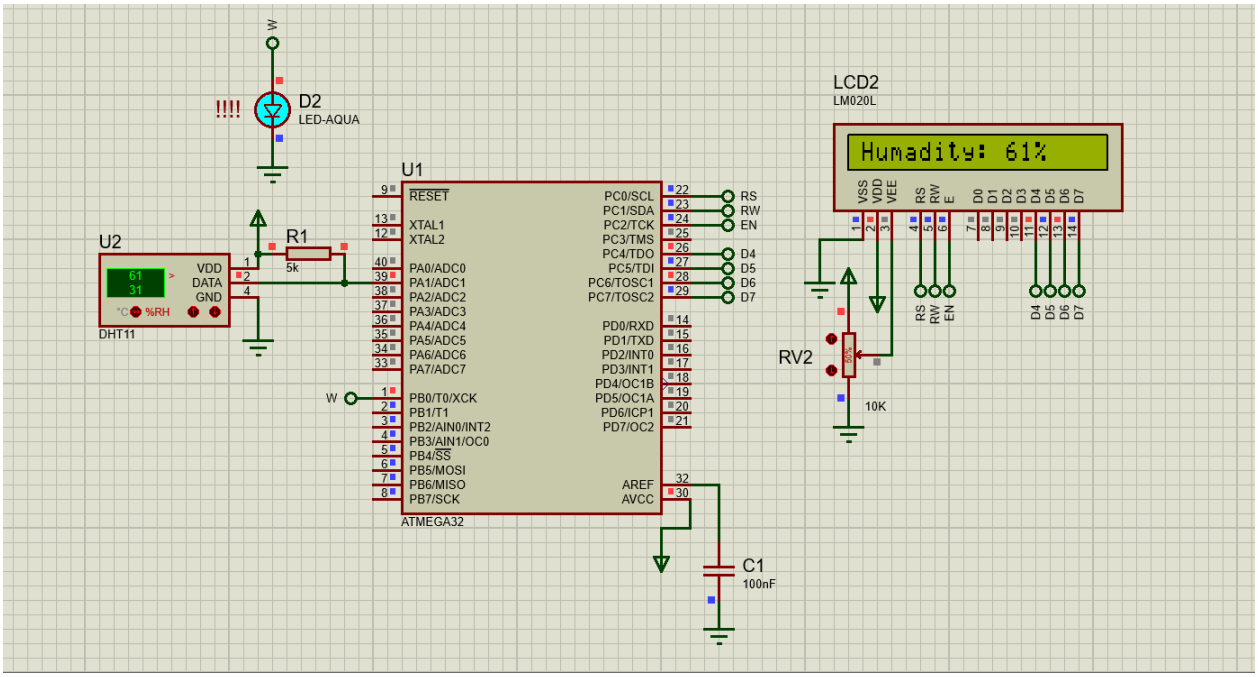
برای نوشتن عدد دورقمی باید آن را رقم به رقم چاپ کنیم که از تابع زیر استفاده میکنیم.

```
unsigned char int_to_char(int x)
{
    return x + '0';
}

void put_on_lcd(int value, int c)
{
    unsigned int r = 0;
    unsigned char tens = int_to_char(value / 10);
    unsigned char ones = int_to_char(value % 10);
    lcd_gotoxy(c, r);
    lcd_putchar(tens);
    lcd_gotoxy(c + 1, r);
    lcd_putchar(ones);
}
```

اگر درصد رطوبت دریافت شده بالای 40 یا پایین 60 بود، بیت صفرم پورت B را یک میکنیم، که به LED متصل است. در غیر این صورت آن را صفر میگذاریم.

```
        put_on_lcd(H, 10);
        lcd_gotoxy(12, 0);
        lcd_puts("%");

        if(H < 40 || H > 60)
           PORTB = 0x01;
        else PORTB = 0x00;

        delay_ms(100);
    }
```

خروجی:


W
!!!!
D2
LED-AQUA
U1
RESET
XTAL1
XTAL2
PA0/ADC0
PA1/ADC1
PA2/ADC2
PA3/ADC3
PA4/ADC4
PA5/ADC5
PA6/ADC6
PA7/ADC7
PB0/T0/XCK
PB1/T1
PB2/AIN0/INT2
PB3/AIN1/OC0
PB4/SS
PB5/MOSI
PB6/MISO
PB7/SCK
ATMEGA32
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/ICP1
PD7/OC2
AREF
AVCC
RS
RW
EN
D4
D5
D6
D7
C1
100nF
U2
VDD
DATA
GND
DHT11
R1
5k
LCD2
LM020L
Humadity: 61%
VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
RV2
10K