گزارش تکلیف

آریا بنائی زاده 9431029

.1

الف)

در مورد الف این سوال برای برقراری وقفه خارجی int1 باید مراحل زیر را انجام دهیم

- در آدرس مربوط به وقفه پرش به روتین خودمان انجام دهیم
- بیت int1 در ثبات GICR را یک کنیم
- نوع حساسیت وقفه را با یک کردن بیت های ISC11,ISC10 تعیین کنیم
- بیت وقفه سراری را فعال کنیم

کد سوال به شرح زیر است. حالت های وقفه با حساسیت های مختلف قابل تعیین است.

در فایل ضمیمه شبیه سازی در نرم افزار پروتئوس نیز انجام شده

```
;
; hw-sol1.asm
;
; Created: 5/2/2018 3:14:45 PM
; Author : Arya
;
.def toggle=r22

.org 0x000
       rjmp SP_INIT
.org 0x0004
       rjmp int1_subroutine
SP_INIT:
       LDI    R16, low(RAMEND)
       OUT    SPL, R16
       LDI    R16, high(RAMEND)
       OUT    SPH, R16

start:
       ; make PD5 output
       ldi r16,(1<<DDD5)
       out ddrd,r16
       ;make portd.3 active high
       ldi r16,(1<<PD3)
       out portd,r16
       ; enable int1 interrupt
       ldi r16,(1<<INT1)
       out GICR,r16
       ;control how the interrupt is enabled
       ldi r16,(0<<ISC11)|(0<<ISC10)
       ;ldi r16,(0<<ISC11)|(1<<ISC10)
       ;ldi r16,(1<<ISC11)|(0<<ISC10)
       ;ldi r16,(1<<ISC11)|(1<<ISC10)
       out MCUCR,r16
       sei
wait:
    rjmp wait
```

```
int1_subroutine:
       inc toggle
       sbrc toggle,0; if bit is zero skip
       rjmp on
       sbrs toggle,0; if bit is one skip
       rjmp off
       rjmp wait

on:
       ldi r16,(1<<PD5)|(1<<PD3)
       out portd,r16
       reti
off:
    ldi r16,(0<<PD5)|(1<<PD3)
       out portd,r16
       reti
```

ب)

با گرفتن سیگنال ورودی از کیبورد و چک کردن هر ستون مقدار عدد مد نظر بدست می آید که با استفاده از این امر هر عدد را در رجیستر ثبت می کنیم. کد به همراه کد بخش بعدی در انتهای سوال آمده است.

پ)

دستگاه نمایشگر عدد را به پورت b وصل میکنیم. بیت ها به این صورت در سوال نگارش شده اند.

portB0 a

portB1 b

portB2 c

portB3 d

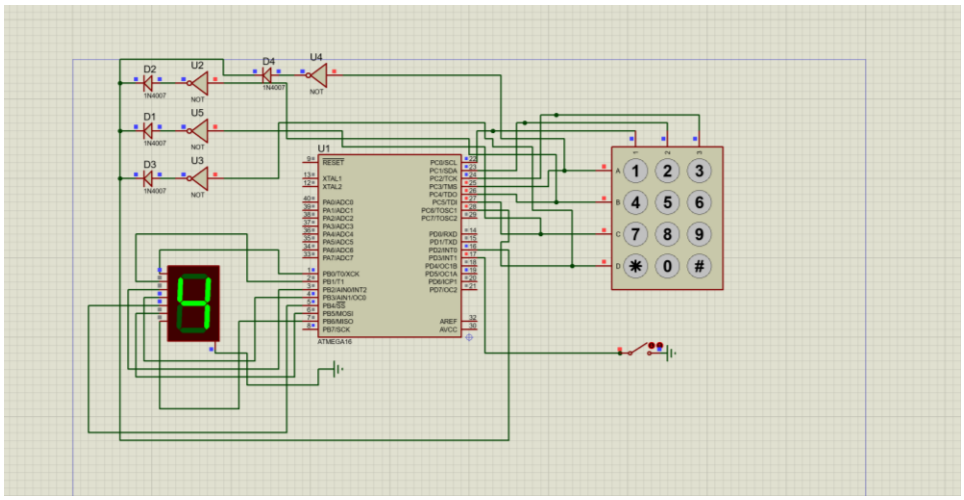
portB4 e

portB5 f

portB6 g

تنها کاری که برای نمایش لازم است انجام دهیم اضافه کردن مقادیر مربوط هر عدد به کد و قرار دادن آن روی portb است.

پس از شبیه سازی برنامه به شکل زیر در می آید.



کد سوال به شرح زیر است.

```
;
; hw-sol1.asm
;
; Created: 5/2/2018 3:14:45 PM
; Author : Arya
;
.def toggle=r22

.org 0x000
       rjmp SP_INIT
.org 0x0002
       rjmp KeyFind
.org 0x0004
       rjmp int1_subroutine
SP_INIT:
       LDI    R16, low(RAMEND)
       OUT    SPL, R16
       LDI    R16, high(RAMEND)
       OUT    SPH, R16

start:

       ; make PD5 output (1.1)
       ldi r16,(1<<DDD5)
       out ddrd,r16
       ;make portd.3 active high (1.1) and portd.2 active low (1.2)
       ldi r16,(1<<PD3)|(0<<PD2)
       out portd,r16
       ;make portc.3-6 output
       ldi r16,(1<<DDC0)|(1<<DDC1)|(1<<DDC2)
       out ddrc,r16
       ldi r16,(1<<PC3)|(1<<PC4)|(1<<PC5)|(1<<PC6)
       out portc,r16
       ;make portB output
       ldi r16,0xff
       out ddrb,r16
       ; enable int1 interrupt
       ldi r16,(1<<INT1)|(1<<INT0)
```

```
        out GICR,r16
        ;control how the interrupt is enabled
        ldi r16,(0<<ISC11)|(0<<ISC10)|(1<<ISC01)|(1<<ISC00)
        ;ldi r16,(0<<ISC11)|(1<<ISC10)
        ;ldi r16,(1<<ISC11)|(0<<ISC10)
        ;ldi r16,(1<<ISC11)|(1<<ISC10)
        out MCUCR,r16
        sei
wait:
    rjmp wait
int1_subroutine:
        inc toggle
        sbrc toggle,0; if bit is zero skip
        rjmp on
        sbrs toggle,0; if bit is one skip
        rjmp off
        rjmp wait

on:
        ldi r16,(1<<PD5)|(1<<PD3)
        out portd,r16
        reti
off:
    ldi r16,(0<<PD5)|(1<<PD3)
        out portd,r16
        reti
KeyFind:
        sbis pinc,3 ;A if zero
        rjmp keyA
        sbis pinc,4 ;B if zero
        rjmp keyB
        sbis pinc,5 ;C if zero
        rjmp keyC
        sbis pinc,6 ;D if zero
        rjmp keyD
        reti
keyA:
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,0
        sbis pinc,3
        call one
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,1
        sbis pinc,3
        call two
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,2
        sbis pinc,3
        call three
        cbi portc,0
        cbi portc,1
        cbi portc,2
```

```
        rjmp write
keyB:
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,0
        sbis pinc,4
        call four
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,1
        sbis pinc,4
        call five
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,2
        sbis pinc,4
        call six
        cbi portc,0
        cbi portc,1
        cbi portc,2
        rjmp write
keyC:
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,0
        sbis pinc,5
        call seven
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,1
        sbis pinc,5
        call eight
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,2
        sbis pinc,5
        call nine
        cbi portc,0
        cbi portc,1
        cbi portc,2
        rjmp write
keyD:
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,0
        sbis pinc,6
        call star
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,1
```

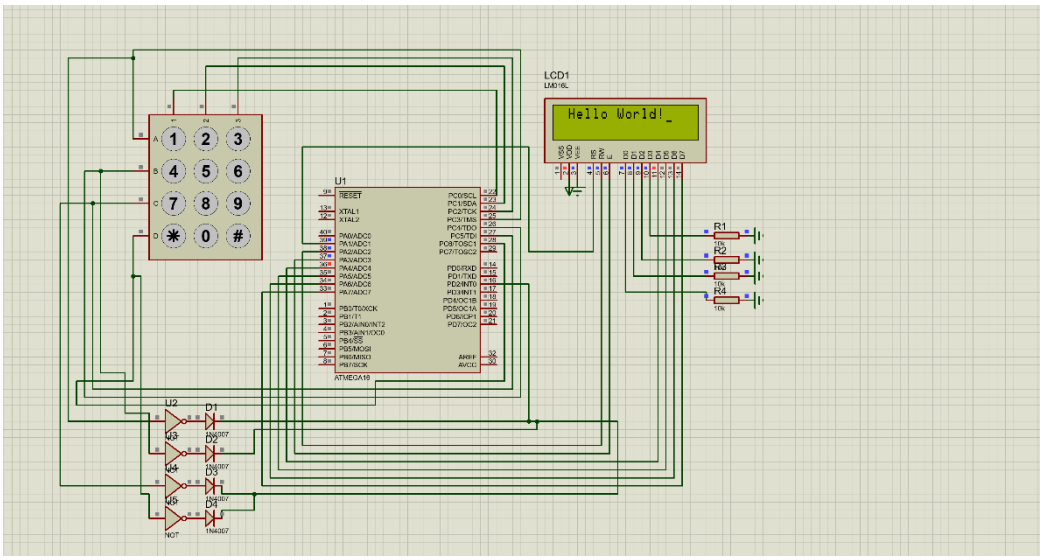
```
        sbis pinc,6
        call zero
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,2
        sbis pinc,6
        call square
        cbi portc,0
        cbi portc,1
        cbi portc,2
        rjmp write
write:
        out portb,r19
        reti
one:
        ldi r18,1
        ldi r19,0b00000110
        ret
two:
        ldi r18,2
        ldi r19,0b01011011
        ret
three:
        ldi r18,3
        ldi r19,0b01001111
        ret
four:
        ldi r18,4
        ldi r19,0b01100110
        ret
five:
        ldi r18,5
        ldi r19,0b01101101
        ret
six:
        ldi r18,6
        ldi r19,0b01111101
        ret
seven:
        ldi r18,7
        ldi r19,0b00000111
        ret
eight:
        ldi r18,8
        ldi r19,0b01111111
        ret
nine:
        ldi r18,9
        ldi r19,0b01101111
        ret
star:
        ldi r18,10
        ldi r19,0b01110111
        ret
zero:
        ldi r18,0
        ldi r19,0b00111111
```

```
        ret
square:
        ldi r18,11
        ldi r19,0b01110011
        ret
```

.2

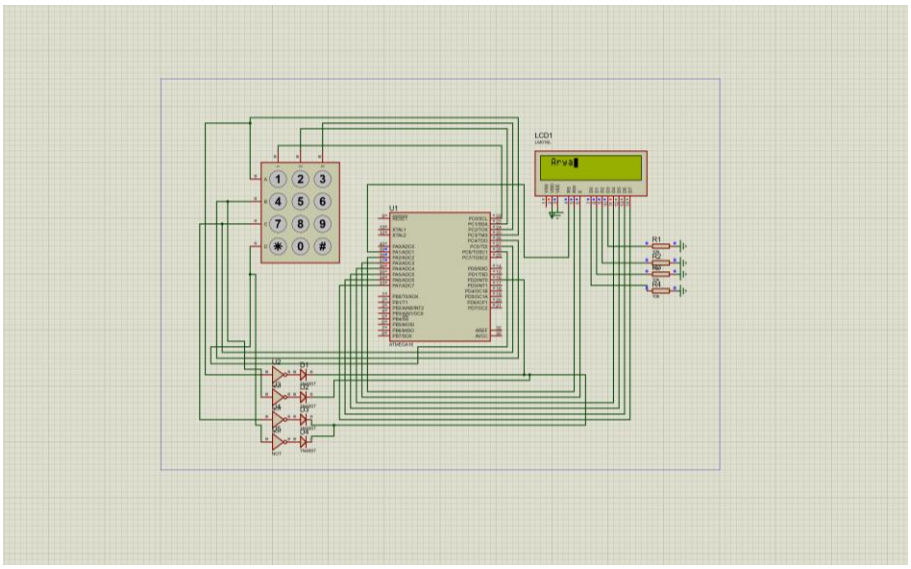
الف)

با استفاده از کد های موجود lcd میخواهیم عبارت hello world را چاپ کنیم. در این قسمت با استفاده از argument و lcd_putchar عبارت hello World را چاپ میکنیم.



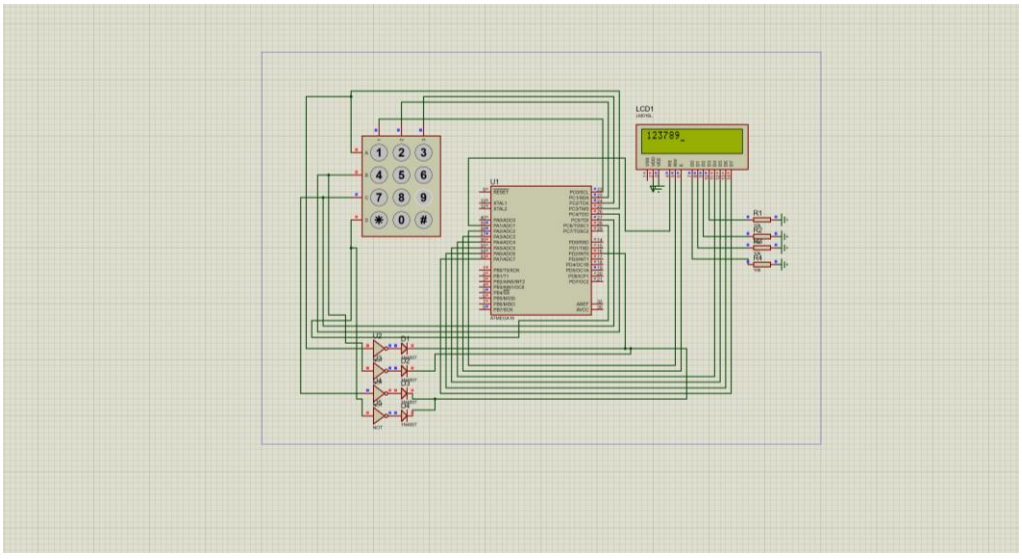
ب)

مشابه قسمت الف با این تفاوت که ابتدا مقادیر مورد نظرمان را در بخشی از حافظه ذخیره کرده و سپس آن ها را در یک حلقه نشان میدهیم.



پ)

در قسمت مربوط به ورودی گرفتن هر عدد مقدار مربوط را د lcd نشان میدهیم



کد سوال در قسمت زیر آمده است. لازم به ذکر است موارد سوال به ترتیب در روتین های , write_hello_world write_form_memory, write_from_keypad پیاده سازی شده اند.

```
;
; hw-sol1.asm
;
; Created: 5/2/2018 3:14:45 PM
; Author : Arya
```

```
;
.def toggle=r22

.org 0x000
       rjmp SP_INIT
.org 0x0002
       rjmp KeyFind
.org 0x0004
       rjmp int1_subroutine
SP_INIT:
       LDI    R16, low(RAMEND)
       OUT    SPL, R16
       LDI    R16, high(RAMEND)
       OUT    SPH, R16

start:

       ; make PD5 output (1.1)
       ldi r16,(1<<DDD5)
       out ddrd,r16
       ;make portd.3 active high (1.1) and portd.2 active low (1.2)
       ldi r16,(1<<PD3)|(0<<PD2)
       out portd,r16
       ;make portc.3-6 output
       ldi r16,(1<<DDC0)|(1<<DDC1)|(1<<DDC2)
       out ddrc,r16
       ldi r16,(1<<PC3)|(1<<PC4)|(1<<PC5)|(1<<PC6)
       out portc,r16
       ;make portB output
       ldi r16,0xff
       out ddrb,r16
       ; enable int1 interrupt
       ldi r16,(1<<INT1)|(1<<INT0)
       out GICR,r16
       ;control how the interrupt is enabled
       ldi r16,(0<<ISC11)|(0<<ISC10)|(1<<ISC01)|(1<<ISC00)
       ;ldi r16,(0<<ISC11)|(1<<ISC10)
       ;ldi r16,(1<<ISC11)|(0<<ISC10)
       ;ldi r16,(1<<ISC11)|(1<<ISC10)
       out MCUCR,r16
       sei
wait:
    rjmp wait
int1_subroutine:
       inc toggle
       sbrc toggle,0; if bit is zero skip
       rjmp on
       sbrs toggle,0; if bit is one skip
       rjmp off
       rjmp wait

on:
       ldi r16,(1<<PD5)|(1<<PD3)
       out portd,r16
       reti
off:
    ldi r16,(0<<PD5)|(1<<PD3)
       out portd,r16
```

```
	reti
KeyFind:
	sbis pinc,3 ;A if zero
	rjmp keyA
	sbis pinc,4 ;B if zero
	rjmp keyB
	sbis pinc,5 ;C if zero
	rjmp keyC
	sbis pinc,6 ;D if zero
	rjmp keyD
	reti
keyA:
	sbi portc,0
	sbi portc,1
	sbi portc,2
	cbi portc,0
	sbis pinc,3
	call one
	sbi portc,0
	sbi portc,1
	sbi portc,2
	cbi portc,1
	sbis pinc,3
	call two
	sbi portc,0
	sbi portc,1
	sbi portc,2
	cbi portc,2
	sbis pinc,3
	call three
	cbi portc,0
	cbi portc,1
	cbi portc,2
	rjmp write
keyB:
	sbi portc,0
	sbi portc,1
	sbi portc,2
	cbi portc,0
	sbis pinc,4
	call four
	sbi portc,0
	sbi portc,1
	sbi portc,2
	cbi portc,1
	sbis pinc,4
	call five
	sbi portc,0
	sbi portc,1
	sbi portc,2
	cbi portc,2
	sbis pinc,4
	call six
	cbi portc,0
	cbi portc,1
	cbi portc,2
	rjmp write
keyC:
```

```
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,0
        sbis pinc,5
        call seven
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,1
        sbis pinc,5
        call eight
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,2
        sbis pinc,5
        call nine
        cbi portc,0
        cbi portc,1
        cbi portc,2
        rjmp write
keyD:
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,0
        sbis pinc,6
        call star
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,1
        sbis pinc,6
        call zero
        sbi portc,0
        sbi portc,1
        sbi portc,2
        cbi portc,2
        sbis pinc,6
        call square
        cbi portc,0
        cbi portc,1
        cbi portc,2
        rjmp write
write:
        out portb,r19
        reti
one:
        ldi r18,1
        ldi r19,0b00000110
        ret
two:
        ldi r18,2
        ldi r19,0b01011011
        ret
three:
        ldi r18,3
```

```
        ldi r19,0b01001111
        ret
four:
        ldi r18,4
        ldi r19,0b01100110
        ret
five:
        ldi r18,5
        ldi r19,0b01101101
        ret
six:
        ldi r18,6
        ldi r19,0b01111101
        ret
seven:
        ldi r18,7
        ldi r19,0b00000111
        ret
eight:
        ldi r18,8
        ldi r19,0b01111111
        ret
nine:
        ldi r18,9
        ldi r19,0b01101111
        ret
star:
        ldi r18,10
        ldi r19,0b01110111
        ret
zero:
        ldi r18,0
        ldi r19,0b00111111
        ret
square:
        ldi r18,11
        ldi r19,0b01110011
        ret
```