بسم الله الرحمن الرحیم

امیرحسین عباسی

گزارش تکلیف پنجم ریزپردازنده

۹۳۳۱۰۷۰

دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

1-A)

```
      ; Reset Vector
      rjmp  Start

;================================================================
===
; CODE SEGMENT
;================================================================
===
.def count=r20
.org 0x00
     jmp rst_isr
.org 0x04
     jmp int1_isr

int1_isr:
     cli
     inc count
     ldi r21, 0x01
     and r21, count ;check if the SW is pressed
     brne turn_on
     ldi r22 ,PORTD & 0b11011111
     out PORTD ,r22 ;turning off
     rjmp end
turn_on:
     ldi r22 ,PORTD | 0b00100000
     out PORTD ,r22 ;turning on
end:
     sei
     ret

rst_isr:
     CLI
     ldi r16, (0 << PD3)
     out DDRD, r16 ;define sw1 as input
     ldi r16, (1 << PD3)
     out PIND, r16 ;pulling up
     ldi r16, (1 << INT1) //enable INT1
     out GICR, r16 ;enable INT1 interupt flag in general interupt
control register
     ldi r16, (1 << PD5)
     out DDRD, r16 ;define the LED1 as output
     sei


start:

     jmp start
```

1-B)

```
     ; Reset Vector
      rjmp  Start

;==================================================================
===
; CODE SEGMENT
;==================================================================
===
.def count=r20
.org 0x00
     jmp rst_isr
.org 0x04
     jmp ext_int0_isr

ext_int0_isr:
     cli
     call keyFind
     rjmp end
keyFind :
      ldi r16, PIND & 0b11111011
      out PORTD , r16 ;pulling down the 3rd column
      jmp third_col
      ldi r16, PIND & 0b11111101
      out PORTD , r16 ;pulling down the 2rd column
      call second_col
      ldi r16, PIND & 0b11111110
      out PORTD , r16 ;pulling down the 1rd column
      call first_col
      rjmp end

third_col :
      ldi r16, PIND & 0b10000000 ;check the 4th row
      cpi r16,0b00000000
      breq sharp ;#

      ldi r16, PIND & 0b01000000 ;check the 3rd row
      cpi r16,0b00000000
      breq nine ;9

      ldi r16, PIND & 0b00100000 ;check the 2rd row
      cpi r16,0b00000000
      breq six ;6

      ldi r16, PIND & 0b00010000 ;check the 1rd row
      cpi r16,0b00000000
      breq three ;3
```

```
second_col :
      ldi r16, PIND & 0b10000000 ;check the 4th row
      cpi r16,0b00000000
      breq zero ;0
      ldi r16, PIND & 0b01000000 ;check the 3rd row
      cpi r16,0b00000000
      breq eight ;8
      ldi r16, PIND & 0b00100000 ;check the 2rd row
      cpi r16,0b00000000
      breq five ;5
      ldi r16, PIND & 0b00010000 ;check the 1rd row
      cpi r16,0b00000000
      breq two ;2
      ret
first_col :
      ldi r16, PIND & 0b10000000 ;check the 4th row
      cpi r16,0b00000000
      breq star ;*
      ldi r16, PIND & 0b01000000 ;check the 3rd row
      cpi r16,0b00000000
      breq seven ;7
      ldi r16, PIND & 0b00100000 ;check the 2rd row
      cpi r16,0b00000000
      breq four ;4
      ldi r16, PIND & 0b00010000 ;check the 1rd row
      cpi r16,0b00000000
      breq one ;1
      ret
sharp:
     ldi r22 ,0xC0
     out PORTB ,r22 ;turning on #
     ret
zero:
     ldi r22 ,0xC0
     out PORTB ,r22 ;turning on #
     ret
one:
     ldi r22 ,0xF9
     out PORTB ,r22 ;turning on #
     ret
two:
     ldi r22 ,0xA4
     out PORTB ,r22 ;turning on #
     ret
three:
     ldi r22 ,0xB0
     out PORTB ,r22 ;turning on #
     ret
four:
     ldi r22 ,0x99
     out PORTB ,r22 ;turning on #
     ret
```

```
five:
     ldi r22 ,0x92
     out PORTB ,r22 ;turning on #
     ret
six:
     ldi r22 ,0x82
     out PORTB ,r22 ;turning on #
     ret
seven:
     ldi r22 ,0xF8
     out PORTB ,r22 ;turning on #
     ret
eight:
     ldi r22 ,0x80
     out PORTB ,r22 ;turning on #
     ret
nine:
     ldi r22 ,0x90
     out PORTB ,r22 ;turning on #
     ret
star:
     ldi r22 ,0xC0
     out PORTB ,r22 ;turning on #
     ret

end:
     sei
     ret

rst_isr:
     CLI
     ldi r16,0b11110000
     out DDRC, r16 ;define columns as input and rows as output
     ldi r16,0b00000111
     out PIND, r16 ;pulling up the rows
     ldi r16, (1 << INT0)
     out GICR, r16 ;enable INT1 interupt flag in general interupt
control register
     sei


start:

     jmp start
```

2-A)

```
rjmp  Start

;=================================================================
===
; CODE SEGMENT
;=================================================================
===

.equ LCD_RS    = 1
.equ LCD_RW    = 2
.equ LCD_E     = 3

.def temp = r16
.def argument= r17       ;argument for calling subroutines
.def return    = r18          ;return value from subroutines

.org 0x00
jmp reset

reset:
     ldi  temp, low(RAMEND)
     out  SPL, temp
     ldi  temp, high(RAMEND)
     out  SPH, temp



Start:
      ; Write your code here
      call LCD_init
      ldi r17, 0x48
      call LCD_putchar
      ldi r17, 0x65
      call LCD_putchar
      ldi r17, 0x6C
      call LCD_putchar
      ldi r17, 0x6C
      call LCD_putchar
      ldi r17, 0x6F
      call LCD_putchar
      ldi r17, 0x20
      call LCD_putchar
      ldi r17, 0x57
      call LCD_putchar
      ldi r17, 0x6F
      call LCD_putchar
      ldi r17, 0x72
      call LCD_putchar
      ldi r17, 0x6C
      call LCD_putchar
      ldi r17, 0x64
      call LCD_putchar
Loop:
      rjmp  Loop
```

2-B)

```
.org LCDTABLE
.db  10,'a','m','i','r','h','o','s','e','i','n'

.org 0x00
jmp reset

LCD:
   ldi zl, 0x70     ; z= 2*184
   ldi zh, 0x01
   lpm r20, z+ ;size
repeat:
   lpm r21, z+ ;character
   mov argument, r21
   call lcd_putchar
   dec r20
   brne repeat
   ret

reset:
     ldi  temp, low(RAMEND)
     out  SPL, temp
     ldi  temp, high(RAMEND)
     out  SPH, temp


Start:

   call lcd_init
   call LCD


Loop:
     rjmp  Loop
```

2-C)

```
.org 0x00
jmp reset

.org 0x02
     jmp ext_int0_isr

ext_int0_isr:
     cli
     call keyFind
     call lcd_wait
     mov argument, r21
     call lcd_putchar
     ldi r20, (1 << PC0 | 1 << PC1 | 1 << PC2 | 1 << PC3 | 0 <<
PC4 | 0 << PC5 | 0 << PC6 | 0 << PC7 )
     out PORTC, r20
     sei
     ret

keyFind :
      ldi r16, PIND & 0b11111011
      out PORTD , r16 ;pulling down the 3rd column
      jmp third_col
      ldi r16, PIND & 0b11111101
      out PORTD , r16 ;pulling down the 2rd column
      call second_col
      ldi r16, PIND & 0b11111110
      out PORTD , r16 ;pulling down the 1rd column
      call first_col
      rjmp end

third_col :
      ldi r16, PIND & 0b10000000 ;check the 4th row
      cpi r16,0b00000000
      breq sharp ;#

      ldi r16, PIND & 0b01000000 ;check the 3rd row
      cpi r16,0b00000000
      breq nine ;9

      ldi r16, PIND & 0b00100000 ;check the 2rd row
      cpi r16,0b00000000
      breq six ;6

      ldi r16, PIND & 0b00010000 ;check the 1rd row
      cpi r16,0b00000000
      breq three ;3
```

```
second_col :
      ldi r16, PIND & 0b10000000 ;check the 4th row
      cpi r16,0b00000000
      breq zero ;0
      ldi r16, PIND & 0b01000000 ;check the 3rd row
      cpi r16,0b00000000
      breq eight ;8
      ldi r16, PIND & 0b00100000 ;check the 2rd row
      cpi r16,0b00000000
      breq five ;5
      ldi r16, PIND & 0b00010000 ;check the 1rd row
      cpi r16,0b00000000
      breq two ;2
      ret
first_col :
      ldi r16, PIND & 0b10000000 ;check the 4th row
      cpi r16,0b00000000
      breq star ;*
      ldi r16, PIND & 0b01000000 ;check the 3rd row
      cpi r16,0b00000000
      breq seven ;7
      ldi r16, PIND & 0b00100000 ;check the 2rd row
      cpi r16,0b00000000
      breq four ;4
      ldi r16, PIND & 0b00010000 ;check the 1rd row
      cpi r16,0b00000000
      breq one ;1
      ret
sharp:
     ldi r21, '#'
     ret
zero:
     ldi r21, '0'
     ret
one:
     ldi r21, '1'
     ret
two:
     ldi r21, '2'
     ret
three:
     ldi r21, '3'
     ret
four:
     ldi r21, '4'
     ret
five:
     ldi r21, '5'
     ret
six:
     ldi r21, '6'
     ret
```

```
seven:
     ldi r21, '7'
     ret
eight:
     ldi r21, '8'
     ret
nine:
     ldi r21, '9'
     ret
star:
     ldi r21, '*'
     ret

 reset:
     cli
     ldi temp, low(RAMEND)
     out SPL, temp
     ldi temp, high(RAMEND)
     out SPH, temp
     call lcd_init

     ldi r20, (1 << isc11)|(0<< isc10)|(1 << isc01)|(0 << isc00)
     out MCUCR, r20
     ldi r20, (1 << INT0) //enable INT0
     out GICR, r20
     ldi r20, (0 << PD2)
     out DDRD, r20
     ldi r20, (1 << PD2)
     out PORTD, r20

     ldi r20, (0 << PC0 | 0 << PC1 | 0 << PC2 | 0 << PC3 | 1 <<
PC4 | 1 << PC5 | 1 << PC6 | 1 << PC7)
     out DDRC, r20
     ldi r20, (1 << PC0 | 1 << PC1 | 1 << PC2 | 1 << PC3 )
     out PORTC, r20
     sei


start:
      ; Write your code here
loop:
      jmp loop
```