2η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ

ΓΙΑ ΤΟ ΜΑΘΗΜΑ

"Εργαστήριο Μικροϋπολογιστών"

Κυριακόπουλος Ιωάννης Α.Μ. :03117440

Παγώνης Γεώργιος Α.Μ.: 03117030

2.1 (Assembly)

```
;
; microlab2_1.asm
;
; Created: 10/27/2020 7:07:39 PM
; Author : georg
;
; Replace with your application code
.INCLUDE "m16def.inc"


.def input = r18
.def temp = r19
.def temp2=r17
.def A = r20
.def B = r21
.def C = r22
.def D = r23
.def F0 = r24
.def F1 = r25
main:
		clr temp
		out DDRC ,temp
		ser temp
		out DDRB,temp
```

```
in input , PINC ; take the input from the PINC
mov A,input
andi A,0x01 ; mask the lsb

mov B,input
andi B,0x02 ; mask the 2nd lsb
ror B ; move it to the 1 position

mov C,input
andi C,0x04 ; mask the 2nd lsb
ror C ; move it to the 1 position
ror C

mov D,input
andi D,0x08 ; mask the 4th lsb
ror D ; move it to the 1 position
ror D
ror D

mov temp2,B
com temp2 ; create the B'
andi temp2,0x01 ;temp2=B'

mov temp,A
com temp ; create the A'
andi temp,0x01; temp=A'

and temp,B ; temp=A'B
and temp2,C ; temp2=B'C
and temp2,D ; temp2= B'CD
or temp,temp2 ; temp=A'B+B'CD
mov F0,temp
mov temp,A
```

```
        and temp,C ; temp =AC

        mov temp2,B
        or temp2,D ; temp2 = B+D
        and temp,temp2 ; temp =(AC)(B+D)
        mov F1,temp
        clc
        rol F1 ; move it to the 2nd position
        com F0 ; create the F0'
        andi F0,0x01 ; mask the first position
        add F0,F1
        out PORTB,F0 ; show it to the PORTB
        rjmp main
end:
```

(C)

```
/*
 * GccApplication1.cpp
 *
 * Created: 10/28/2020 6:42:12 PM
 * Author : georg
 */
#include <avr/io.h>

unsigned char A,B,C,D,NOTA,NOTB,F0,F1;

int main(void){
        DDRC=0x00;  // C= input
        DDRB=0xff;  // B= output
        while(1){
                A = PINC &0x01; // mask the  lsb
                B = PINC &0x02; // mask the 2 position
                B = B >> 1; // move it to the 1 position
```

```
                C = PINC & 0x04; // mask the 3 position
                C = C >> 2; // move it to the 1 position
                D = PINC &0x08; //mask the 4 position
                D = D >> 3; // move it to the 1 position
                NOTA = A ^ 0x01; // xor with 1 so we came out with A' and B'
                NOTB = B ^ 0x01;
                F0 = (NOTA & B)|(NOTB & C & D);
                F0 = F0 ^ 0x01; // F0'
                F1 = (A & C) & (B | D);
                F1 = F1 << 1; // move it to the second position
                F0 = F0 + F1;
                PORTB = F0; // output to the port B
        }
}
```

2.2

```
;
; AssemblerApplication1.asm
;
; Created: 10/28/2020 4:20:49 PM
; Author : georg
;
; Replace with your application code
.include "m16def.inc"

                .org 0x0
                rjmp RESET
                .org 0x4
                rjmp ISR1

RESET:
        .def temp = r20
        .def input =r21
```

```
.def interupt_counter=r22
.def timer_counter=r26

clr interupt_counter
clr timer_counter

ldi temp,LOW(RAMEND)
out SPL,temp
ldi temp,HIGH(RAMEND)
out SPH,temp

 ldi r24,(1<<ISC11)|(1<<ISC10) ; mask the MCUCR for the positive edge of the interupt1
out MCUCR,r24
ldi r24 ,(1<<INT1)
out GICR,r24 ; mask the GICR of the interupt1
sei ; enable interupts

ser temp
out DDRC,temp ; DDRC output

loop:
out PORTC,timer_counter ; show the timer_counter
;ldi r24,low(100)
;ldi r25,high(100)
;rcall wait_msec
inc timer_counter ; increace the timer_counter
rjmp loop ; go again

ISR1:
cli
in temp, SREG ; save the System Register
push temp
clr temp
```

```
        out DDRA,temp ; A= input
        ser temp
        out DDRB,temp ; B = output
        inc interupt_counter ; increace the interupt_counter
        in input,PINA ; check A if you have to show it or not
        cpi input,0xC0
        brne dont_display_the_interupt_counter

display_the_interupt_counter:
        out PORTB,interupt_counter
        ;ldi r24,low(100)
        ;ldi r25,high(100)
        ;rcall wait_msec
        pop temp
        out SREG,temp
        reti

dont_display_the_interupt_counter:
        ;ldi r24,low(100)
        ;ldi r25,high(100)
        ;rcall wait_msec
        pop temp
        out SREG,temp
        reti
```

2.3

```
/*
 * GccApplication1.cpp
 *
 * Created: 10/28/2020 6:42:12 PM
 * Author : georg
 */
```

```
#include <avr/io.h>
#include <avr/interrupt.h>

unsigned char A,B,temp;
char x,c;
volatile int flag=1;

 ISR(INT0_vect){
        //pd2 routine
        cli();
        if((PINA & 0x04) == 0x04){
                x=0x00;
                if((PINB &0x01) == 0x01) x++;
                if((PINB &0x02) == 0x02) x++;
                if((PINB &0x04) == 0x04) x++;
                if((PINB &0x08) == 0x08) x++;
                if((PINB &0x10) == 0x10) x++;
                if((PINB &0x20) == 0x20) x++;
                if((PINB &0x40) == 0x40) x++;
                if((PINB &0x80) == 0x80) x++; //count the pins of B
                c=0x00;
                while(x>0){
                        c=c<<1;
                        c=c+1;
                        x=x-1;
                }
                PORTC=c; // open so many leds of the C starting from the lsb as the count of B


        }
        else{
```

```
		x=0x00;
		if((PINB &0x01) == 0x01) x++;
		if((PINB &0x02) == 0x02) x++;
		if((PINB &0x04) == 0x04) x++;
		if((PINB &0x08) == 0x08) x++;
		if((PINB &0x10) == 0x10) x++;
		if((PINB &0x20) == 0x20) x++;
		if((PINB &0x40) == 0x40) x++;
		if((PINB &0x80) == 0x80) x++; //count the pins of B

		PORTC=x; // output them on binary form
	}
	sei();
}

int main(){

	GICR=(1<<INT0); // be ready for the interrupt0
	MCUCR = (1<<ISC01|1<<ISC00); // wait on the positive edge of the interrupt0
	sei();

	DDRA = 0x00; // A input
	DDRB = 0x00; // B input
	DDRC = 0xff; // C output
	while(1){
		PORTC=0x00; // wait for the interrupt0
		}
	}
```