Αναφορά εργαστηρίου ΑΥΓΕΡΗΣ ΝΙΚΟΛΑΟΣ 1067508

Ερώτημα 1

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
int array[3]={0,0,0,0};
int x=0;
int interr=0;
int arraycode[4]=(6,5,5,6); //right code
void init_timer_TCA0 ();



int main() {
       PORTD.DIR |= PIN0_bm;
       PORTD.OUTCLR |= PIN0_bm; // led is off
       PORTF.PIN5CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
       PORTF.PIN6CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
          while (  array[0]== arraycode[0] && array[1]== arraycode[1] && array[2]== arraycode[2] && array[3]== arraycode[3] )
{  // this while function ends when code is right
                                    while(x!=4) //4 digits
                                   {
                                      sei(); //enable interrupts

                                      while (interr==0) {};  //waiting for an interrupt to occure – so go to isr button
                                      cli();



                                   }
}
       void init_timer_TCA0 ();

};
      ISR(PORTF_PORT_vect) //first function isr
       {
              int intflags=PORTF.INTFLAGS;
              PORTF.INTFLAGS=intflags;
              interr=1;
              x=x+1;
              if (PORTF.OUT == 01000000) //SW6
              {    array[x]=6;}

              else if (PORTF.OUT == 00100000)   //SW5
              {    array[x]=5;}

              else {array[x]=0; }

       }


void init_timer_TCA0 (void);
{

       TCA0.SINGLE.CNT=0; //clear counter
       TCA0.SINGLE.CTRLB=0; // normal mode
```

```
if (x==4) // code given , make them zero again
 {
   x=0;
   for (int i=0; i==3; i++)
 array[i]=0;
 }
```

```
	TCA0.SINGLE.CMP0=ped;// when reaches that value interrupt
	TCA0.SINGLE.CTRLA = TCA_SINGLE_CLKSEL_CIV1024_gc;
	TCA0.SINGLE.CTRLA |=1; // enable
	TCA0.SINGLE.INTCTRL = TCA_SINGLE_CMP0bm; // interrupt enable
	sei(); // accept interrupts
	while (interr==0) {}
	cli();



}

ISR(TCA_CMP0_vect) //isr for initial tca
{
	TCA0.SINGLE.CTRLA=0; // disable clear fl
	int intflags=TCA0.SINGLE.INTFLAGS;
	TCA0.SINGLE.INTFLAGS=intflags;
	interr=1;

}
```

Ερώτημα 2

Στο ερώτημα 2, βάζουμε μια διαφορετική μεταβλητή ιντερ να διαχειριζεται το ιντεραπτ του τιμερ διότι πρέπει να ξεχωρίζουμε πότε τελειώνει ο τιμερ ή πότε κάποιος πατάει τον κωδικό.

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
int array[3]={0,0,0,0};
int x=0;
int interr=0;
int interr1=0;

int arraycode[4]=(6,5,5,6); //right code
void init_timer_TCA0 ();
void ADC_init (void);
```

```
int main() {
	PORTD.DIR |= PIN0_bm;
	PORTD.OUTCLR |= PIN0_bm; // led is off
	PORTF.PIN5CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
	PORTF.PIN6CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
while (  array[0]!= arraycode[0] && array[1]!= arraycode[1] && array[2]!= arraycode[2] && array[3]!= arraycode[3] ){  // this while function ends when code is right
                                  while(x!=4) //4 digits
                                 {
                                    sei(); //enable interrupts

                                    while (interr==0) {};  //waiting for an interrupt to occure - so go to isr button

                                     cli();
                                                                        if (x==4) // code given , make them zero again
                                                                         {
                                                                           x=0;
                                                                           for (int i=0; i==3; i++)
                                                                         array[i]=0;
                                                                         }
                                 }
}
	void init_timer_TCA0 ();

	void ADC_init ();



};



  ISR(PORTF_PORT_vect) //first function isr
	{
		int intflags=PORTF.INTFLAGS;
		PORTF.INTFLAGS=intflags;
		interr=1;
		x=x+1;
		if (PORTF.OUT == 01000000) //SW6
		{    array[x]=6;}

		else if (PORTF.OUT == 00100000)   //SW5
		{    array[x]=5;}

		 else {array[x]=0; }

	}









void init_timer_TCA0 (void);
{

	TCA0.SINGLE.CNT=0; //clear counter
```

```
	TCA0.SINGLE.CTRLB=0; // normal mode
	TCA0.SINGLE.CMP0=ped;// when reaches that value interrupt
	TCA0.SINGLE.CTRLA = TCA_SINGLE_CLKSEL_CIV1024_gc;
	TCA0.SINGLE.CTRLA |=1; // enable
	TCA0.SINGLE.INTCTRL = TCA_SINGLE_CMP0bm; // interrupt enable
	sei(); // accept interrupts
	while (interr==0) {}
	cli();



}

ISR(TCA_CMP0_vect) //isr for initial tca
{
	TCA0.SINGLE.CTRLA=0; // disable clear fl
	int intflags=TCA0.SINGLE.INTFLAGS;
	TCA0.SINGLE.INTFLAGS=intflags;
	Interr1=1;

}









void ADC_init (void){
		PORT.DIR|= PIN0_bm; //PIN IS OUTPUT
		//INITIALLIZE THA ADC FOR FREE RUNNING MODE
		ADC0.CTRLA |= ADC_RESSEL_10BIT_gc; //10BIT RESOLUTION
		ADC0.CTRLA |= ADC_FREERUN_bm; //FREE RUNNING MODE ENABLED
		ADC0.CTRLA |= ADC_ENABLE_bm; //ENABLE ADC
		ADC0.MUXPOS |= ADC_MUXPOS_AIN7_gc; // THE BIT ENABLE DEBUG MODE
		ADC0.DBGCTRL |= ADC_DBGRUN_bm; // WINDOW COMPARATOR MODE
		ADC0.WINLT |= 10; // SET THRESHOLD
		ADC0.INTCTRL |= ADC_WINCM0_bm; // ENABLE INTERRUPTS FOR WCM
		ADC0.CTRLE |= ADC_WINCM0_bm; // INTERRUPT WHEN RESULT<WINLT
		sei();
		ADC0.COMMAND |= ADC_STCONV_bm; // START CONVERSION
		while(interr==0){}


	}
ISR(ADC0_WCMP_vect) { // adc
		int intflags=ADC0.INTFLAGS;
		ADC0.INTFLAGS = intflags;
		PORTD.OUTCLR=PIN0_bm; // led is on

		void init_timer_TCA0 ();
		 while ( interr1 || array[0]!= arraycode[0] && array[1]!= arraycode[1] && array[2]!= arraycode[2] && array[3]!= arraycode[3] ){  // this while function ends when code is right
		                      while(x!=4) //4 digits
		                     {
		                        sei(); //enable interrupts

		                        while (interr==0) {};  //waiting for an interrupt to occure - so go to isr button

		                         cli();
		                                                 if (x==4) // code given , make them zero again
		                                                  {
		                                                    x=0;
		                                                    for (int i=0; i==3; i++)
		                                                  array[i]=0;
		                                                  }
```

```
                                    }
}
PORTD.OUT |= PIN0_bm; //led on
        }
```

Ερώτημα 3

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
int array[3]={0,0,0,0};
int x=0;
int interr=0;
int interr1=0;

int arraycode[4]=(6,5,5,6); //right code
void init_timer_TCA0 ();
void ADC_init (void);
```

```
int main() {
       PORTD.DIR |= PIN0_bm;
       PORTD.OUTCLR |= PIN0_bm; // led is off
       PORTF.PIN5CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
       PORTF.PIN6CTRL|=PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc; //pullup enabled and interrupt enabled with sense on both edges
while (  array[0]!= arraycode[0] && array[1]!= arraycode[1] && array[2]!= arraycode[2] && array[3]!= arraycode[3] )
{  // this while function ends when code is right
                                  while(x!=4) //4 digits
                                 {
                                    sei(); //enable interrupts

                                    while (interr==0) {};  //waiting for an interrupt to occure – so go to isr button

                                     cli();
                                                                  if (x==4) // code given , make them zero again
                                                                   {
                                                                     x=0;
                                                                     for (int i=0; i==3; i++)
                                                                   array[i]=0;
                                                                   }
                                 }
}

void init_timer_TCA0 ();
//here starts the adc function ----------------------------------------------------------------------------------------------------------------------------------

void ADC_init ();
while ( array[0]!= arraycode[0] && array[1]!= arraycode[1] && array[2]!= arraycode[2] && array[3]!= arraycode[3]){
PORTD.OUT |= PIN0_bm; //led on
TCA0.SINGLE.CTRLA = TCA_SINGLE_CLKSEL_CIV1024_gc;
TCA0.SINGLE.PER=254;
TCA0.SINGLE.CMP0=127;
TCA0.SINGLE.CTRLB|= TCA_SINGLESLOPE_gc;
TCA0.SINGLE.INTCTRL=TCA_SINGLE_OVF_bm;
TCA0.SINGLE.INTCTRL=TCA_SINGLE_CMP0_bm;
TCA0.SINGLE.CTRLA |= TCA_SINGLE_ENABLE_bm;
sei();
 while(x!=4) //4 digits  here we do again the same thing as in the previous function, we just let the alarm on and waiting for password----------------------------
                                 {
                                    sei(); //enable interrupts

                                    while (interr==0) {};  //waiting for an interrupt to occure – so go to isr button

                                     cli();
                                                                  if (x==4) // code given , make them zero again
                                                                   {
                                                                     x=0;
                                                                     for (int i=0; i==3; i++)
                                                                   array[i]=0;
                                                                   }
                                 }



}
TCA0.SINGLE.CTRLA |= TCA_SINGLE_DISENABLE_bm; //when code is correct disable the alarm sound

};
```

```
ISR(TCA0_OVF_vect){
//clear the interrupt flag
int intflags=TCA0.SINGLE.INTFLAGS;
TCA0.SINGLE.INTFLAGS=intlfags;
PORTD.OUT|=PIN0_bm;
}
ISR(TCA0_CMP0_vect){
//clear the interrupt flag
int intflags=TCA0.SINGLE.INTFLAGS;
TCA0.SINGLE.INTFLAGS=intflags;
PORTD.OUT|=PIN0_bm;
}
```

```
ISR(PORTF_PORT_vect) //first function isr
    {
           int intflags=PORTF.INTFLAGS;
           PORTF.INTFLAGS=intflags;
           interr=1;
           x=x+1;
           if (PORTF.OUT == 01000000) //SW6
           {    array[x]=6;}
           else if (PORTF.OUT == 00100000)   //SW5
           {    array[x]=5;}

           else {array[x]=0; }

      }
```

```
void init_timer_TCA0 (void);
{

       TCA0.SINGLE.CNT=0; //clear counter
       TCA0.SINGLE.CTRLB=0; // normal mode
       TCA0.SINGLE.CMP0=ped;// when reaches that value interrupt
       TCA0.SINGLE.CTRLA = TCA_SINGLE_CLKSEL_CIV1024_gc;
       TCA0.SINGLE.CTRLA |=1; // enable
       TCA0.SINGLE.INTCTRL = TCA_SINGLE_CMP0bm; // interrupt enable
       sei(); // accept interrupts
       while (interr==0) {}
       cli();



}

ISR(TCA_CMP0_vect) //isr for initial tca
{
       TCA0.SINGLE.CTRLA=0; // disable clear fl
       int intflags=TCA0.SINGLE.INTFLAGS;
       TCA0.SINGLE.INTFLAGS=intflags;
```

```
	Interr1=1;
}


void ADC_init (void){
		PORT.DIR|= PIN0_bm; //PIN IS OUTPUT
		//INITIALLIZE THA ADC FOR FREE RUNNING MODE
		ADC0.CTRLA |= ADC_RESSEL_10BIT_gc; //10BIT RESOLUTION
		ADC0.CTRLA |= ADC_FREERUN_bm; //FREE RUNNING MODE ENABLED
		ADC0.CTRLA |= ADC_ENABLE_bm; //ENABLE ADC
		ADC0.MUXPOS |= ADC_MUXPOS_AIN7_gc; // THE BIT ENABLE DEBUG MODE
		ADC0.DBGCTRL |= ADC_DBGRUN_bm; // WINDOW COMPARATOR MODE
		ADC0.WINLT |= 10; // SET THRESHOLD
		ADC0.INTCTRL |= ADC_WINCM0_bm; // ENABLE INTERRUPTS FOR WCM
		ADC0.CTRLE |= ADC_WINCM0_bm; // INTERRUPT WHEN RESULT<WINLT
		sei();
		ADC0.COMMAND |= ADC_STCONV_bm; // START CONVERSION
		while(interr==0){}


	}


ISR(ADC0_WCMP_vect) { // adc  edo o timer exei jekinhsh na metraei kai perimenoyme na teleiosei. Tote jekinaei o adc kai an dei pali anthropo energopoieitai timer.
 // o neos timer me interr1 perimenoyme na teleiosei h perimenoyme na dothei 4 fores lathos kodikos h allios 12 lathos chfia
		int intflags=ADC0.INTFLAGS;
		ADC0.INTFLAGS = intflags;
		PORTD.OUTCLR=PIN0_bm; // led is on

		 void init_timer_TCA0 ();
 while ( interr1==0  || x==12 ){  // this while function ends, if interr=1 so timer over or x==12 then code given 3 times
                                  while(x!=4) //4 digits
                                 {
                                  sei(); //enable interrupts

                                  while (interr==0) {};  //waiting for an interrupt to occure - so go to isr button

                                   cli();
                                                        if (x==4) // code given , make them zero again
                                                         {
                                                           x=0;
                                                           for (int i=0; i==3; i++)
                                                         array[i]=0;
                                                         }
                        }
                }

}
```