**ΕΡΓΑΣΤΗΡΙΑΚΗ ΆΣΚΗΣΗ 1**

**ΜΈΛΟΙ ΟMAΔΑΣ:**

**Σίδερης Στυλιανός:ΑΜ:1075217.έτος: 4ο**

**Ψυρρή Διονυσία: AM:1080424, έτος: 4ο**

Στην αρχή θα θέλαμε να σας επισημάνουμε κάποια βασικά σημεία της εργασίας. Για την παραγωγή των PWM χρησιμοποιήσαμε τους μετρητές TCAO και TCB0.Δεν κάναμε split διότι σε καμιά άσκηση μέχρι τώρα δεν είχαμε κάνει γνωριμία με τον μετρητή TCB0.

Για την έξοδο των PWM συνδέσαμε τις κατάλληλες εξόδους στους κατάλληλους πολυπλέκτες που συνδέονται απευθείας με τους τους καταχωρητές CMP.Αυτό έγινε με τις εντολές:

PORTMUX.TCAROUTEA |=PORTMUX\_TCA0\_PORTC\_gc;

PORTMUX.TCBROUTEA |=0x0;

Κατεβάσαμε το ρολόι του συστήματος στα 32kHz και δώσαμε στον TCB0 ρολόι από το ΜΑΙΝ CLOCK του συστήματος με τις εντολές.

CPU\_CCP = CCP\_IOREG\_gc; -> για να μπορέσουμε να αλλάξουμε τιμές σε καταχωρητές σαν τον παρακάτω για θέματα ασφαλείας.

CLKCTRL.MCLKCTRLA=CLKCTRL\_CLKSEL\_OSCULP32K\_gc;-> ρολόι στα 32ΚHz

**Μέρος πρώτο(ενεργοποίηση ανεμηστήρα με switch)**

Στην αρχή περιμένουμε σε μια while να σταθεροποιηθεί το νέο ρολόι.

Μετά κάνουμε αρχικοποίηση στους δύο timer για την παραγωγή των PWM(δεν ξεκινάνε όμως ακόμα). Για αυτή την ενέργεια έχουμε φτιάξει συναρτήσεις. Αρχικοποιούμε και το switch για λειτουργεία on off.

Ξεκινάει το πρόγραμμα και είμαστε στην while μέσα στην main.Εφόσον δεν έχουμε ενεργοποιήσει το switch on off δεν θα φύγουμε ποτέ από εκεί και δεν θα έχουμ επαραφωφή PWM και ο ανεμιστήρας θα είναι σε κατάσταση αναμονής.

Όταν πατηθεί το switch θα έχουμε interrupt.Μέσα στην ISR θα ενεργοποιηθούν και οι δύο μετρητές θα γίνουν δηλαδή enable με κατάλληλο **CMP** έτσι ώστε να έχουμε διαφορά φάσης στους δύο μετρητές και να μην έχουμε ταυτόχρονα interrupt (overflow)και από τους δύο ταυτόχρονα .Τώρα λοιπόν έχουμε παραγωγή PWM και ο ανεμιστήρας λειτουργεί. Στην έξοδο **PORTC** στο PIN \_\_ παίρνουμε το σήμα **PWM** για το μοτέρ της βάσης και στο **PORTA PIN \_\_** για το μοτέρ των λεπίδων.

Τα interrupt για overflow για τους μετρητές τα θέλουμε μόνο για να ανάβουμε και να σβήνουμε τα led που μας ζητάτε γιατί όπως αναφέραμε και πιο πάνω τα **PWM** τα παίρνουμε απευθείας από τις καθορισμένες εξόδους.

Αν πατήσουμε το **switch on off** και ο ανεμιστήρας είναι σε λειτουργεία τότε ο ανεμιστήρας θα κλείσει.

Για το αν ανοίξει ή αν κλείσει το καταλαβαίνουμε με μία βοηθητική μεταβλητή **fun\_on\_off** μέσα στην ISR.Το άναμμα και το σβήσιμο των κατάλληλων led έχει γίνει σύμφωνα με τις οδηγείες.

Διάγραμμα ροής:

Κώδικας:

#include <avr/io.h>

#include <avr/interrupt.h>

#define FREQ\_BASE 300

#define DUTY\_BASE FREQ\_BASE\*0.40

#define FREQ\_LEPIDES 255

#define DUTY\_LEPIDES FREQ\_LEPIDES/2

int fun\_on\_off=0;

//-------------------sinarthseis----------------------------------

void init\_timer\_base(void){

    TCA0.SINGLE.CTRLA=TCA\_SINGLE\_CLKSEL\_DIV1\_gc;//xoris prescaler

    TCA0.SINGLE.PER=FREQ\_BASE;

    TCA0.SINGLE.CMP0=DUTY\_BASE;

    //RITHMISI MODE

    TCA0.SINGLE.CTRLB=TCA\_SINGLE\_CMP0EN\_bm|TCA\_SINGLE\_WGMODE\_SINGLESLOPE\_gc;

    //ENABLE INTERRUPR

    TCA0.SINGLE.INTCTRL=TCA\_SINGLE\_OVF\_bm;

    TCA0.SINGLE.INTCTRL|=TCA\_SINGLE\_CMP0\_bm;

    TCA0.SINGLE.CNT=0;

    //H EXODOS PWM THA BGENEI STO PORTC

    PORTMUX.TCAROUTEA |=PORTMUX\_TCA0\_PORTC\_gc;

}

void init\_timer\_lepides(void){

    //ENERGOPIHSI GIA PWM

    TCB0.CTRLA = (TCB0.CTRLA & ~TCB\_CLKSEL\_gm) | TCB\_CLKSEL\_CLKDIV1\_gc;

    TCB0.CTRLB |= TCB\_CCMPEN\_bm;

    TCB0.CTRLB |= TCB\_CNTMODE\_PWM8\_gc;

    TCB0\_CCMPL= FREQ\_LEPIDES;

    TCB0\_CCMPH= DUTY\_LEPIDES;

    TCB0.CNT=15;

    //H EXODOS NA BGENEI STO PORTA

    PORTMUX.TCBROUTEA |=0x0;

    TCB0.INTCTRL|=TCB\_CAPT\_bm;

}

int main(void)

{

    //epitrepsi prosvasis se eidikoys kataxorites

    CPU\_CCP = CCP\_IOREG\_gc;

    //syxnothtia sta 32 KHh

    CLKCTRL.MCLKCTRLA=CLKCTRL\_CLKSEL\_OSCULP32K\_gc;

    //CLKCTRL.MCLKCTRLB = CLKCTRL\_PDIV\_64X\_gc | CLKCTRL\_PEN\_bm;

    //ELENXOS AN TO ROLOI EINAI STATHERO

    while (CLKCTRL.MCLKSTATUS & CLKCTRL\_SOSC\_bm)

    {

        ;

    }

     //Led gia sixnothta

     PORTD.DIR  |= PIN0\_bm|PIN1\_bm;

     //Exodoi gia pwm palmo

     PORTC.DIR  |= PIN0\_bm;

     PORTA.DIR  |= PIN2\_bm;

    init\_timer\_base();

    init\_timer\_lepides();

    //on off switch mono se kathodoki akmi

    PORTF.PIN5CTRL |=PORT\_PULLUPEN\_bm | PORT\_ISC\_FALLING\_gc;

    sei();

    while (1)

    {

        PORTE.OUT|= 0b00000000;

    }

    cli();

}

//----------------------ISR------------------------------

ISR(PORTF\_PORT\_vect){

    int y = PORTF.INTFLAGS;

    PORTF.INTFLAGS=y;

    if(!fun\_on\_off){

       TCA0.SINGLE.CNT=0;

       TCA0.SINGLE.CTRLA |=TCA\_SINGLE\_ENABLE\_bm;

       TCB0\_CNT=15;

       TCB0.CTRLA |=TCB\_ENABLE\_bm;

       fun\_on\_off=1;

    }

    else{

        TCA0.SINGLE.CTRLA &=0x0;

        TCB0.CTRLA &=0x0;

        //sbino ta led

        PORTD.OUTCLR |=0x0F;

        fun\_on\_off=0;

    }

}

ISR(TCA0\_OVF\_vect){//gia otan exoyme ena pliri kyklo

    int intflags = TCA0.SINGLE.INTFLAGS;

    TCA0.SINGLE.INTFLAGS = intflags;

    //led on off

     PORTD.OUTTGL=PIN1\_bm;

}

ISR(TCA0\_CMP0\_vect){//otan teleivsei to duty cycle

    int intflags = TCA0.SINGLE.INTFLAGS;

    TCA0.SINGLE.INTFLAGS = intflags;

}

ISR(TCB0\_INT\_vect)//lepides otan exoyme ena pliri kyklo

{

        int intflags = TCB0.INTFLAGS;

        TCB0.INTFLAGS = intflags;

        //led on off

        PORTD.OUTTGL =PIN0\_bm;

}

**Μέρος δεύτερο(διακοπή λειτουργείας με ADC)**