



ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ  
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ  
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ  
ΕΡΓΑΣΤΗΡΙΟ ΤΕΧΝΟΛΟΓΙΑΣ & ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ

---

## ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ

2<sup>η</sup> ΑΣΚΗΣΗ

16/04/2021

Αλέξανδρος Νατσολλάρη AM: 1057769

Παναγιώτης Μπαρμπούνης AM: 1054382

**Παραδοχές :** Στο PIN0 θεωρούμε το φανάρι του μεγάλου δρόμου, στο PIN1 το φανάρι των πεζών και στο PIN2 το φανάρι του μικρού δρόμου. Θεωρούμε επίσης ότι λευκό τετραγωνάκι αντιπροσωπεύει κόκκινο φανάρι ενώ κόκκινο τετραγωνάκι και αντίστοιχα μπλε αντιπροσωπεύει πράσινο φανάρι. Το ped το βάλαμε 120 γιατί σύμφωνα με τον τύπο στις διαφάνειες και μετρήσεις που κάναμε υπολογίσαμε ότι το timer λειτουργεί για περίπου 15 δευτερόλεπτα. Επίσης έχουμε προσθέσει μερικά delay για να προσομοιώνουμε την καθυστέρηση από όταν ανάβει κόκκινο σε ένα φανάρι και πράσινο στο άλλο. Τέλος θεωρούμε ως αρχική κατάσταση την κατάσταση πράσινο φανάρι μόνο στον μεγάλο δρόμο.

## Πείραμα

Ο κώδικας της άσκησης είναι ο ακόλουθος:

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <stdio.h>

#define del2 10
#define ped 120

int interr=0; //logic flag
int g=0;
int w=0;

int main(void){

    short int rnd,tyxaios_arithmos;
    //fanari megalou dromou
    PORTD.DIR |= PIN0_bm; //PIN is output
    PORTD.OUT |= PIN0_bm;

    //fanari pezwn
    PORTD.DIR |= PIN1_bm; //PIN is output

    //fanari mikrou dromou
    PORTD.DIR |= PIN2_bm; //PIN is output

    while (1) {

        //pullup enable and Interrupt enabled with sense on both edges
        PORTF.PIN5CTRL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
        sei(); //enable interrupts

        //dinoume mia random timh opws orizei h ekfwnhsh
        tyxaios_arithmos=rand();

        //kanoume extract to teleutaio psifio gia sygrishsh parakatw
        rnd=tyxaios_arithmos % 10;
        printf("%d",rnd);
    }
}
```

```

if(rnd==0 || rnd==5 || rnd==8)
{
    //anavoume to fanari toy mikrou dromou kai kleinoume tou mikrou
    PORTD.OUTCLR= PIN0_bm; //kane kokkino to fanari tou megalou dromou
    _delay_ms(del2); //mikrh kathusterhsh apo thn stigmh pou ginetai prassino to fanari
    PORTD.OUT |= PIN2_bm; //kane prassino to fanari toy mikrou dromou
    g=1;
}
else{
    PORTD.OUTCLR= PIN2_bm;//kane kokkino to fanari toy mikrou dromoy ean den einai
kokkino

    _delay_ms(del2);//kathusterhsh gia to pote tha ginei prassino to fanari
    PORTD.OUT |= PIN0_bm;//kane prasino to fanari toy megaloy dromou ean den einai
prassino
    g=0;
}

if (interr==1)
{
    PORTD.OUTCLR= PIN0_bm;//kane kokkino to fanari toy megaloy dromou
    _delay_ms(del2); // 2sec
    PORTD.OUT |= PIN1_bm;//kane prasino to fanari twn pezwn

    PORTD.OUT |= PIN2_bm;//kane prasino to fanari toy mikrou dromou

    w=0;
    TCA0.SINGLE.CNT = 0; //clear counter
    TCA0.SINGLE.CTRLB = 0; //Normal Mode (TCA_SINGLE_WGMODE_NORMAL_gc
    TCA0.SINGLE.CMP0 = ped; //When reaches this value -> interrupt CLOCK FREQUENCY/1024
    TCA0.SINGLE.CTRLA = TCA_SINGLE_CLKSEL_DIV1024_gc; // (= 0x7<<1 )
    TCA0.SINGLE.CTRLA |=1;//Enable
    TCA0.SINGLE.INTCTRL = TCA_SINGLE_CMP0_bm; //Interrupt Enable (=0x10)

    Interr=0;

    while(w==0){

    }
    PORTD.OUTCLR= PIN1_bm;//kane kokkino to fanari twn pezwn
    cli(); //disenable interrupts

    if(g==0){
        PORTD.OUTCLR= PIN2_bm;//kane kokkino to fanari toy mikrou dromou
        _delay_ms(del2);//kathusterhsh gia to pote tha ginei prassino to fanari
        PORTD.OUT |= PIN0_bm;//kane prasino to fanari toy megaloy dromou
    }
}
}

ISR(PORTF_PORT_vect){
    //clear the interrupt flag
}

```

```

int intflags = PORTF.INTFLAGS;
PORTF.INTFLAGS=intflags;
interr=1;
sei(); //enable interrupts
}

ISR(TCA0_CMP0_vect){
    TCA0.SINGLE.CTRLA = 0; //Disable
//clear flag
    int intflags = TCA0.SINGLE.INTFLAGS;
    TCA0.SINGLE.INTFLAGS=intflags;
    w=1;
    sei(); //enable interrupts
}

```

## Αποτελέσματα Εκτέλεσης

Αρχικά βάζουμε breakpoint στα σημεία που θέλουμε να δούμε τις τιμές των led που μας ενδιαφέρουν. Οι γραμμές στις οποίες βάλαμε breakpoints είναι : 43 ( για να βλέπουμε την τιμή από το τελευταίο ψηφίο του αριθμού που παράγει η rand()), 51, 55, 58, 72, 87, 93 και 112.

Έπειτα ακολουθεί εξήγηση της λειτουργίας με screenshots και περιγραφή :

Αρχική κατάσταση έχουμε όπως φαίνεται στο screenshot όπου είναι (ανοιχτό)πράσινο φανάρι μόνο του μεγάλου δρόμου(PIN0) και κόκκινα(κλειστά) τα PIN1 και PIN2 που αντιστοιχούν στο φανάρι μικρού δρόμου και φανάρι πεζών αντίστοιχα.

The screenshot shows the Atmel Studio interface during debugging. The code editor displays a C program for the ATmega408 microcontroller. The I/O register viewer shows the state of various pins and registers. The memory dump window shows the contents of EEPROM memory at address 0x0000. The solution explorer lists the project files.

```

main.c

main() {
    //fanari mikrou dromou
    PORTD.DIN |= PIN2_bm; //PIN is output

    while (1) {
        //pullup enable and interrupt enabled with sense on both edges
        PORTD.PINCTRL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
        sei(); //enable interrupts

        //diavase mia random timi apo grisei kai ekfereish
        tyxios_arithmos=rand();

        //kaneis extract to telotato seifiso gia synexish parakate
        rand=tyxios_arithmos % 16;
        printf("%d\n",rand);

        if(rand==5 || rand==8 || rand==0)
        {
            //paramevne to fanari toy mikrou dromou kai kleidoume tou mikrou
            PORTD.DINL|= PIN2_bm; //kane kokeino to fanari toy mikrou dromou
            delay_ms(10); //ekfereish taktos kai apotelesme se 10ms
            PORTD.DIN |= PIN2_bm; //kane prassingi to fanari toy mikrou dromou
            g++; //paramevne to fanari toy mikrou dromou
        }
        else{
            //paramevne to fanari toy mikrou dromou kai kleidoume tou mikrou
            PORTD.DINL|= PIN2_bm; //kane kokeino to fanari toy mikrou dromou
            delay_ms(12); //ekfereish taktos kai apotelesme se 12ms
            PORTD.DIN |= PIN2_bm; //kane prassingi to fanari toy mikrou dromou
            g++; //paramevne to fanari toy mikrou dromou
        }
    }
}

```

Πατώντας run βλέπουμε ότι η μεταβλητή rnd που αποθηκεύεται το τελευταίο ψηφίο του αριθμού που παράγει η rand() είναι διάφορο του 0 ή 5 ή 8 άρα δεν μπαίνει στην if.

Φτάνοντας στο breakpoint στην γραμμή 58 βλέπουμε στο screenshot από κάτω ότι αφού δεν υπάρχει interrupt και η rand() δεν παρήγαγε αριθμό που τελειώνει σε 0 ή 5 ή 8 τα φανάρια μένουν όπως ήταν:

The screenshot shows the Atmel Studio interface during debugging, similar to the previous one but with different register values. The I/O register viewer shows the state of various pins and registers. The memory dump window shows the contents of EEPROM memory at address 0x0000. The solution explorer lists the project files.

```

main.c

main() {
    //fanari mikrou dromou
    PORTD.DIN |= PIN2_bm; //PIN is output

    while (1) {
        //fanari mikrou dromou
        PORTD.PINCTRL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
        sei(); //enable interrupts

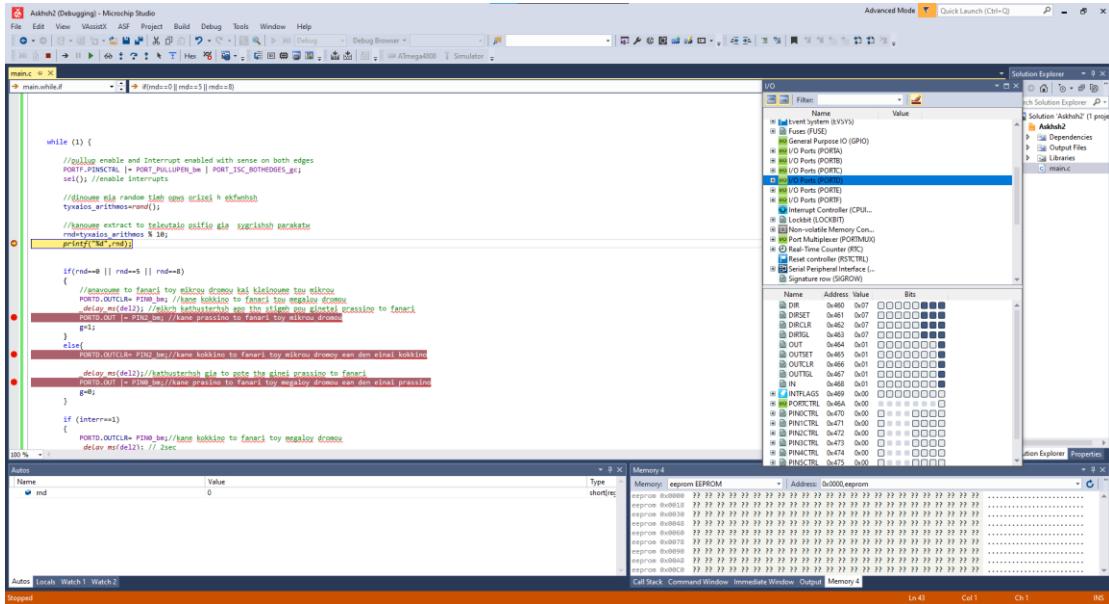
        //diavase mia random timi apo grisei kai ekfereish
        tyxios_arithmos=rand();

        //kaneis extract to telotato seifiso gia synexish parakate
        rand=tyxios_arithmos % 16;
        printf("%d\n",rand);

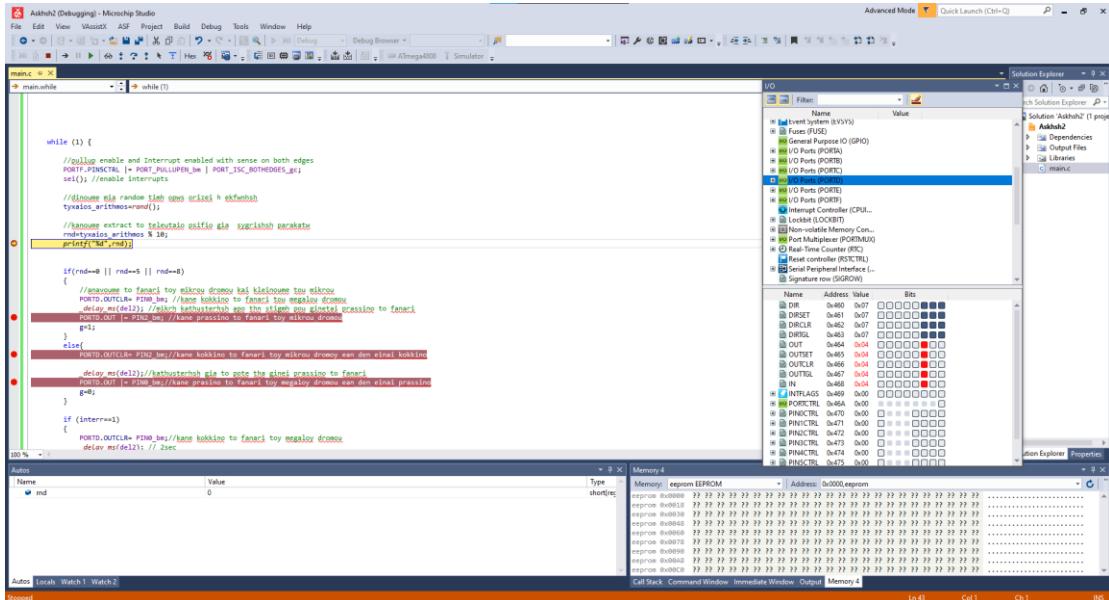
        if(rand==5 || rand==8 || rand==0)
        {
            //paramevne to fanari toy mikrou dromou kai kleidoume tou mikrou
            PORTD.DINL|= PIN2_bm; //kane kokeino to fanari toy mikrou dromou
            delay_ms(10); //ekfereish taktos kai apotelesme se 10ms
            PORTD.DIN |= PIN2_bm; //kane prassingi to fanari toy mikrou dromou
            g++; //paramevne to fanari toy mikrou dromou
        }
        else{
            //paramevne to fanari toy mikrou dromou kai kleidoume tou mikrou
            PORTD.DINL|= PIN2_bm; //kane kokeino to fanari toy mikrou dromou
            delay_ms(12); //ekfereish taktos kai apotelesme se 12ms
            PORTD.DIN |= PIN2_bm; //kane prassingi to fanari toy mikrou dromou
            g++; //paramevne to fanari toy mikrou dromou
        }
    }
}

```

Αν στην rand() έρθει αριθμός που τελειώνει σε 0 ή 5 ή 8 (δηλαδή ενεργοποιηθεί ο αισθητήρας του φαναριού του μικρού δρόμου) έχουμε τα εξής αποτελέσματα:

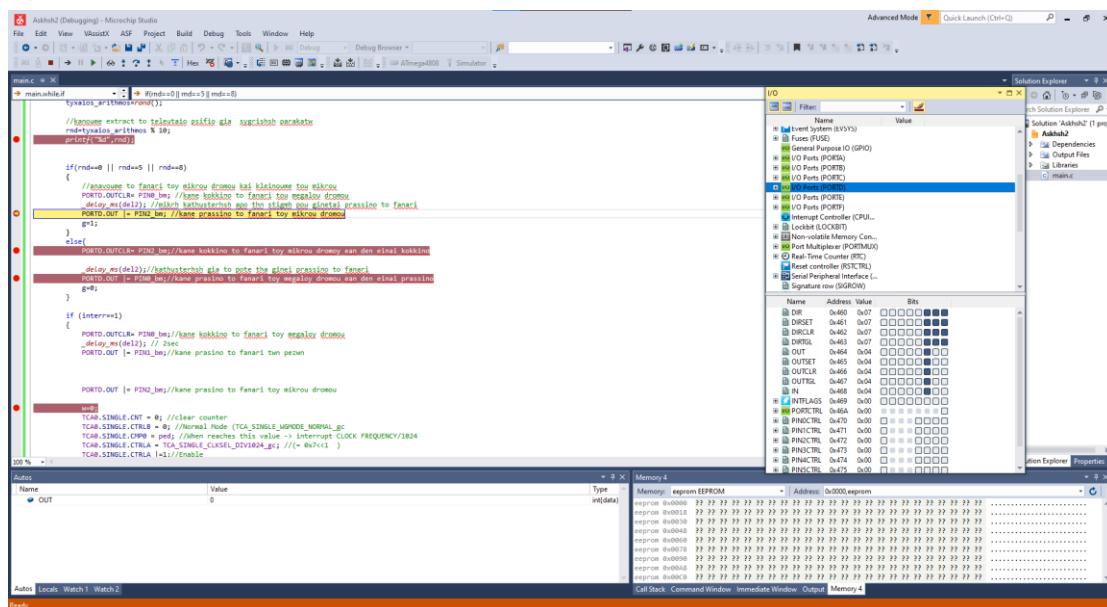
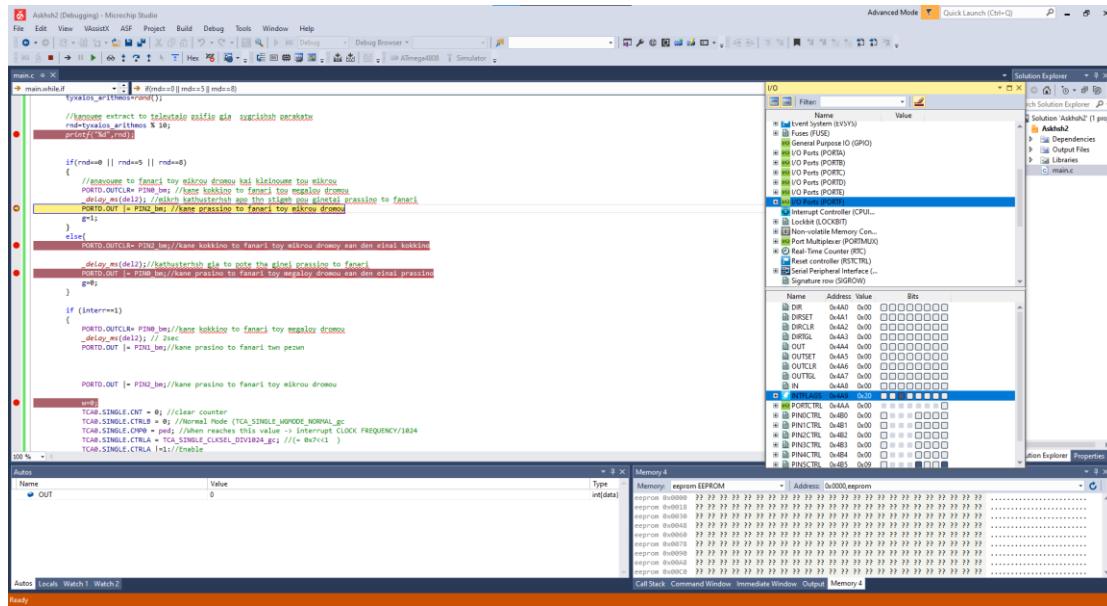


Όπως φαίνεται και στο screenshot ο κώδικας μπήκε στην if και αφού εκτελέστηκε ο κώδικας της γραμμής 51 όπου έχουμε το breakpoint το φανάρι του μικρού δρόμου έγινε πράσινο και το φανάρι του μεγάλου δρόμου έγινε κόκκινο.

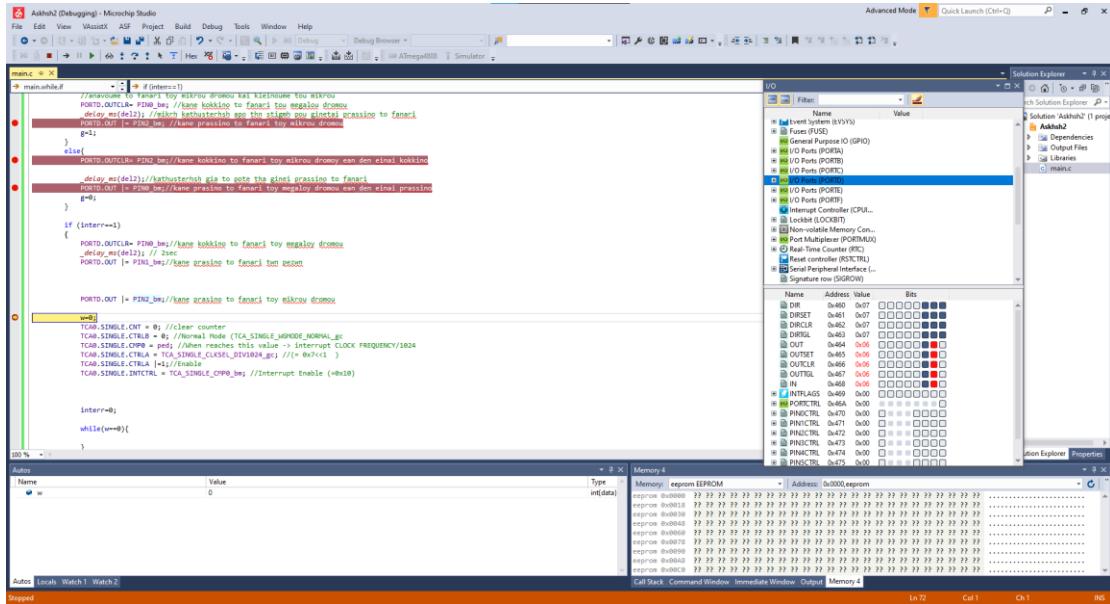


Εδώ όσες φορές και να πατήσουμε run (χωρίς interrupt από πεζό) έχουμε εναλλαγή φαναριών του μικρού δρόμου και του μεγάλου δρόμου από πράσινο σε και κόκκινο και από κόκκινο σε πράσινο αντίστοιχα. Εάν το φανάρι του μικρού δρόμου είναι πράσινο και έρθει και άλλο αυτοκίνητο παραμένει πράσινο μέχρι να μείνει κενός ο δρόμος.

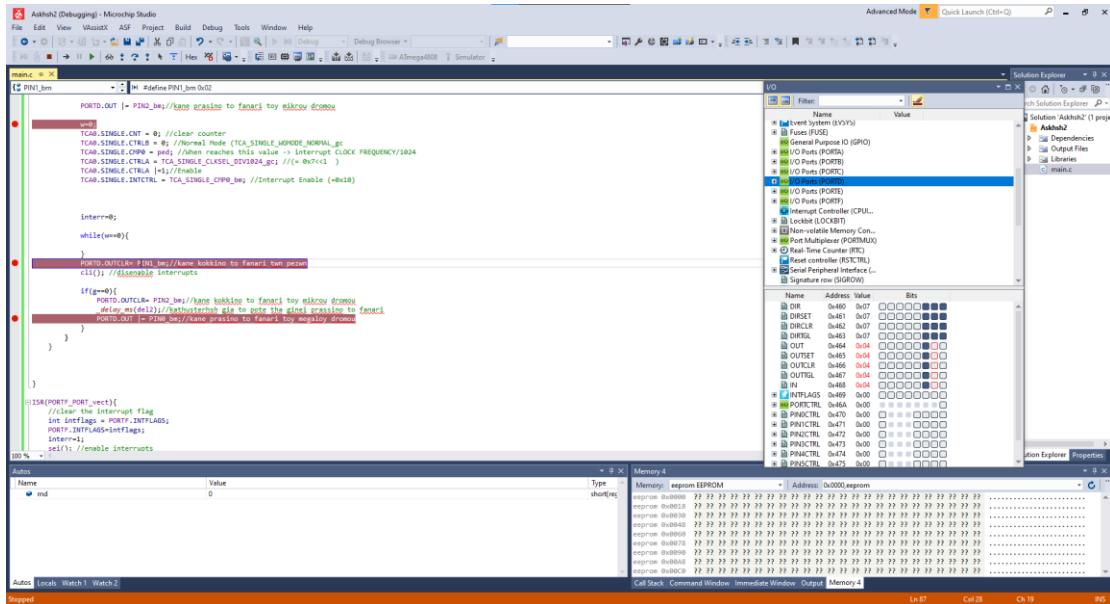
Έστω τώρα ότι έχουμε πράσινο φανάρι στον μικρό δρόμο (κόκκινο στο φανάρι του μεγάλου δρόμου) και πατιέται interrupt(PIN5 PORTF) από κάποιο πεζό :



Πατώντας run φτάνουμε στο breakpoint γραμμή 72 βλέπουμε ότι άναψε πράσινο στο φανάρι των πεζών και τα άλλα 2 φανάρια παρέμειναν στην κατάσταση που ήταν:



Μετά ενεργοποιούμε το wait όπου και μένει το πράσινο φανάρι των πεζών ανοιχτό για περίπου 15 δευτερόλεπτα και αφού εκτελεστεί και ο κώδικας της γραμμής 87 που έχουμε το breakpoint, το φανάρι των πεζών γίνεται κόκκινο και στο επόμενο βήμα επιστρέφουμε στην προηγούμενη κατάσταση που ήμασταν πριν το interrupt:



Screenshot of Microchip Studio showing the code for Askish2 (Debugging) project. The code in main.c includes comments in Greek about pin functions and port control. The I/O register viewer shows the state of various pins and ports. The memory viewer shows the EEPROM memory at address 0x2000.

```

// Askish2 (Debugging) - Microchip Studio
File Edit View VascuSTK ASF Project Build Debug Tools Window Help
File Edit View VascuSTK ASF Project Build Debug Tools Window Help Advanced Mode Quick Launch (Ctrl+Q)
main.c [PIN1.hm]
#include "PIN1.hm"
#define PIN1_bm 0x02
PORTD.DIN |= PIN1_bm; //PIN is output

//fanari mikrou dromou
PORTD.DIN |= PIN2_bm; //PIN is output

while (1) {
    //pullup enable and Interrupt enabled with sense on both edges
    PORT1_PDSCTRL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
    sei(); //enable interrupts

    //diomos mia random timi opou orizis h ekfanash
    tyxaio_arithmos=<random>;

    //kameis extract to telestasio gafris gia apokathise parakate
    rnd_tyxaio_arithmos % 10;
    printf("%d\r\n");

    if((rnd==5 || rnd==8) {
        //emoneuse to fanari toy mikrou dromou kai klimonei ton mikrou
        PORTD.DIN&=~PIN1_bm; //kane klimino to fanari toy megalo dromou
        delay_ms(100); //ekfanash katwsterish apo to stigmo pou ginetai prassingi to fanari
        PORTD.DIN |= PIN1_bm; //kane prassingi to fanari toy mikrou dromou
        delay_ms(100);
    }
    else{
        PORTD.DOUT|= PIN2_bm; //kane klimino to fanari toy mikrou dromou ean den einai klimino
        delay_ms(100); //ekfanash ria to note the gieni prassingi to fanari
        PORTD.DOUT |= PIN2_bm; //kane prassingi to fanari toy megalo dromou ean den einai prassingi
        delay_ms(100);
    }
    a++;
}

```

I/O Register View:

Name	Address	Value	Bits
DIRECT	0x460	0x07	00000000
DIRECT	0x461	0x07	00000000
DIRECT	0x462	0x07	00000000
DIRECT	0x463	0x07	00000000
OUTSET	0x464	0x01	00000000
OUTSET	0x465	0x01	00000000
OUTSET	0x466	0x01	00000000
OUTSET	0x467	0x01	00000000
INTFLAGS	0x469	0x00	00000000
PINCTRL	0x470	0x00	00000000
PINCTRL	0x471	0x00	00000000
PINCTRL	0x472	0x00	00000000
PINCTRL	0x473	0x00	00000000
PINCTRL	0x474	0x00	00000000
PINCTRL	0x475	0x00	00000000

Memory View:

Name	Address	Value	Bits
exprom EEPROM	0x2000	0x0000	00000000
exprom EEPROM	0x2001	0x0000	00000000
exprom EEPROM	0x2002	0x0000	00000000
exprom EEPROM	0x2003	0x0000	00000000
exprom EEPROM	0x2004	0x0000	00000000
exprom EEPROM	0x2005	0x0000	00000000
exprom EEPROM	0x2006	0x0000	00000000
exprom EEPROM	0x2007	0x0000	00000000
exprom EEPROM	0x2008	0x0000	00000000
exprom EEPROM	0x2009	0x0000	00000000
exprom EEPROM	0x200A	0x0000	00000000
exprom EEPROM	0x200B	0x0000	00000000
exprom EEPROM	0x200C	0x0000	00000000
exprom EEPROM	0x200D	0x0000	00000000
exprom EEPROM	0x200E	0x0000	00000000
exprom EEPROM	0x200F	0x0000	00000000
exprom EEPROM	0x2010	0x0000	00000000
exprom EEPROM	0x2011	0x0000	00000000
exprom EEPROM	0x2012	0x0000	00000000
exprom EEPROM	0x2013	0x0000	00000000
exprom EEPROM	0x2014	0x0000	00000000
exprom EEPROM	0x2015	0x0000	00000000
exprom EEPROM	0x2016	0x0000	00000000
exprom EEPROM	0x2017	0x0000	00000000
exprom EEPROM	0x2018	0x0000	00000000
exprom EEPROM	0x2019	0x0000	00000000
exprom EEPROM	0x201A	0x0000	00000000
exprom EEPROM	0x201B	0x0000	00000000
exprom EEPROM	0x201C	0x0000	00000000
exprom EEPROM	0x201D	0x0000	00000000
exprom EEPROM	0x201E	0x0000	00000000
exprom EEPROM	0x201F	0x0000	00000000

Έστω ότι το φανάρι του μεγάλου δρόμου είναι πράσινο(PIN0) και στον μικρό δρόμο το φανάρι είναι κόκκινο(PIN2) και ότι κάποιος πεζός πατάει το κουμπί για να ανοίξει πράσινο φανάρι στους πεζούς:

Screenshot of Microchip Studio showing the code for Askish2 (Debugging) project. The code in main.c includes comments in Greek about pin functions and port control. The I/O register viewer shows the state of various pins and ports. The memory viewer shows the EEPROM memory at address 0x2000.

```

// Askish2 (Debugging) - Microchip Studio
File Edit View VascuSTK ASF Project Build Debug Tools Window Help
File Edit View VascuSTK ASF Project Build Debug Tools Window Help Advanced Mode Quick Launch (Ctrl+Q)
main.c [PIN1.hm]
#include "PIN1.hm"
#define PIN1_bm 0x02
PORTD.DIN |= PIN1_bm; //PIN is output

//fanari mikrou dromou
PORTD.DIN |= PIN2_bm; //PIN is output

while (1) {
    //pullup enable and Interrupt enabled with sense on both edges
    PORT1_PDSCTRL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
    sei(); //enable interrupts

    //diomos mia random timi opou orizis h ekfanash
    tyxaio_arithmos=<random>;

    //kameis extract to telestasio gafris gia apokathise parakate
    rnd_tyxaio_arithmos % 10;
    printf("%d\r\n");

    if((rnd==5 || rnd==8) {
        //emoneuse to fanari toy mikrou dromou kai klimonei ton mikrou
        PORTD.DIN&=~PIN1_bm; //kane klimino to fanari toy megalo dromou
        delay_ms(100); //ekfanash katwsterish apo to stigmo pou ginetai prassingi to fanari
        PORTD.DIN |= PIN1_bm; //kane prassingi to fanari toy mikrou dromou
        delay_ms(100);
    }
    else{
        PORTD.DOUT|= PIN2_bm; //kane klimino to fanari toy mikrou dromou ean den einai klimino
        delay_ms(100); //ekfanash ria to note the gieni prassingi to fanari
        PORTD.DOUT |= PIN2_bm; //kane prassingi to fanari toy megalo dromou ean den einai prassingi
        delay_ms(100);
    }
    a++;
}

```

I/O Register View:

Name	Address	Value	Bits
DIRECT	0x460	0x07	00000000
DIRECT	0x461	0x07	00000000
DIRECT	0x462	0x07	00000000
DIRECT	0x463	0x07	00000000
OUTSET	0x464	0x01	00000000
OUTSET	0x465	0x01	00000000
OUTSET	0x466	0x01	00000000
OUTSET	0x467	0x01	00000000
INTFLAGS	0x469	0x00	00000000
PINCTRL	0x470	0x00	00000000
PINCTRL	0x471	0x00	00000000
PINCTRL	0x472	0x00	00000000
PINCTRL	0x473	0x00	00000000
PINCTRL	0x474	0x00	00000000
PINCTRL	0x475	0x00	00000000

Memory View:

Name	Address	Value	Bits
exprom EEPROM	0x2000	0x0000	00000000
exprom EEPROM	0x2001	0x0000	00000000
exprom EEPROM	0x2002	0x0000	00000000
exprom EEPROM	0x2003	0x0000	00000000
exprom EEPROM	0x2004	0x0000	00000000
exprom EEPROM	0x2005	0x0000	00000000
exprom EEPROM	0x2006	0x0000	00000000
exprom EEPROM	0x2007	0x0000	00000000
exprom EEPROM	0x2008	0x0000	00000000
exprom EEPROM	0x2009	0x0000	00000000
exprom EEPROM	0x200A	0x0000	00000000
exprom EEPROM	0x200B	0x0000	00000000
exprom EEPROM	0x200C	0x0000	00000000
exprom EEPROM	0x200D	0x0000	00000000
exprom EEPROM	0x200E	0x0000	00000000
exprom EEPROM	0x200F	0x0000	00000000
exprom EEPROM	0x2010	0x0000	00000000
exprom EEPROM	0x2011	0x0000	00000000
exprom EEPROM	0x2012	0x0000	00000000
exprom EEPROM	0x2013	0x0000	00000000
exprom EEPROM	0x2014	0x0000	00000000
exprom EEPROM	0x2015	0x0000	00000000
exprom EEPROM	0x2016	0x0000	00000000
exprom EEPROM	0x2017	0x0000	00000000
exprom EEPROM	0x2018	0x0000	00000000
exprom EEPROM	0x2019	0x0000	00000000
exprom EEPROM	0x201A	0x0000	00000000
exprom EEPROM	0x201B	0x0000	00000000
exprom EEPROM	0x201C	0x0000	00000000
exprom EEPROM	0x201D	0x0000	00000000
exprom EEPROM	0x201E	0x0000	00000000
exprom EEPROM	0x201F	0x0000	00000000

The screenshot shows the Microchip Studio interface with the project "Atkshh2" open. The assembly code window displays a loop that toggles pins and handles interrupts. The memory dump and I/O register views are visible on the right, showing the state of various ports and timers.

```

main.c + x
main.while:
    //PORTD.DIN |= PIN2_bm; //PIN is output
    PORTD.DIN |= PIN2_bm; //PIN is output

    //fanari mikrou dronou
    PORTD.DIN |= PIN2_bm; //PIN is output

    while (1) {
        //pullup enable and Interrupt enabled with sense on both edges
        PORTF.PINSEL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
        sei(); //enable interrupts

        //dikomeno mia random timi opou crizis h ekfwbash
        tykhanis_katathesis=&rnd();
        //kaneis extract to teleutiko seifio mia synexish parakate
        rnd=(tykhanis_katathesis%10);
        printf("rnd=%d\n",rnd);

        if(rnd==5 || rnd==8)
        {
            //fanari to fanari toy mikrou dronou kai klimene tou mikrou
            PORTD.DINL|= PIN2_bm; //kane kokkinio to fanari toy megalo dronou
            delay_ms(100); //ekfwbash katathesis apo to itigion pou ginetai prassingi to fanari
            PORTD.DINL|= PIN2_bm; //kane agkalo to fanari toy mikrou dronou
            delay_ms(100); //ekfwbash katathesis apo to itigion pou ginetai prassingi to fanari
            p++; //next pin
        }
        else
        {
            //fanari to fanari toy mikrou dronou kai klimene tou mikrou
            PORTD.DINL|= PIN2_bm; //kane kokkinio to fanari toy megalo dronou
            delay_ms(100); //ekfwbash katathesis apo to itigion pou ginetai prassingi to fanari
            PORTD.DINL|= PIN2_bm; //kane agkalo to fanari toy megalo dronou
            delay_ms(100); //ekfwbash katathesis apo to itigion pou ginetai prassingi to fanari
            p++; //next pin
        }

        if((rnd==5 || rnd==8) && (interr==1))
        {
            PORTD.DOUTL|= PIN2_bm; //kane kokkinio to fanari toy mikrou dronou
            delay_ms(100); //ekfwbash gia to pose tis gineki prassingi to fanari
            PORTD.DOUTL|= PIN2_bm; //kane agkalo to fanari toy megalo dronou
            delay_ms(100); //ekfwbash katathesis apo tis gineki prassingi to fanari
        }
    }

    while(1);
}

```

Φτάνοντας στο breakpoint της γραμμής 72 βλέπουμε ότι το φανάρι των πεζών έγινε πράσινο και το φανάρι του μεγάλου δρόμου έγινε κόκκινο:

The screenshot shows the Microchip Studio interface with the project "Atkshh2" open. The assembly code window displays a loop that toggles pins and handles interrupts. The memory dump and I/O register views are visible on the right, showing the state of various ports and timers.

```

main.c + x
main.while:
    //PORTD.DIN |= PIN2_bm; //PIN is output
    PORTD.DIN |= PIN2_bm; //PIN is output

    //fanari mikrou dronou
    PORTD.DIN |= PIN2_bm; //PIN is output

    while (1) {
        //pullup enable and Interrupt enabled with sense on both edges
        PORTF.PINSEL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
        sei(); //enable interrupts

        //dikomeno mia random timi opou crizis h ekfwbash
        tykhanis_katathesis=&rnd();
        //kaneis extract to teleutiko seifio mia synexish parakate
        rnd=(tykhanis_katathesis%10);
        printf("rnd=%d\n",rnd);

        if(rnd==5 || rnd==8)
        {
            //fanari to fanari toy mikrou dronou kai klimene tou mikrou
            PORTD.DINL|= PIN2_bm; //kane kokkinio to fanari toy megalo dronou
            delay_ms(100); //ekfwbash katathesis apo to itigion pou ginetai prassingi to fanari
            PORTD.DINL|= PIN2_bm; //kane agkalo to fanari toy mikrou dronou
            delay_ms(100); //ekfwbash katathesis apo to itigion pou ginetai prassingi to fanari
            p++; //next pin
        }
        else
        {
            //fanari to fanari toy mikrou dronou kai klimene tou mikrou
            PORTD.DINL|= PIN2_bm; //kane kokkinio to fanari toy megalo dronou
            delay_ms(100); //ekfwbash katathesis apo to itigion pou ginetai prassingi to fanari
            PORTD.DINL|= PIN2_bm; //kane agkalo to fanari toy megalo dronou
            delay_ms(100); //ekfwbash katathesis apo to itigion pou ginetai prassingi to fanari
            p++; //next pin
        }

        if((rnd==5 || rnd==8) && (interr==1))
        {
            PORTD.DOUTL|= PIN2_bm; //kane kokkinio to fanari toy mikrou dronou
            delay_ms(100); //ekfwbash gia to pose tis gineki prassingi to fanari
            PORTD.DOUTL|= PIN2_bm; //kane agkalo to fanari toy megalo dronou
            delay_ms(100); //ekfwbash katathesis apo tis gineki prassingi to fanari
        }
    }

    while(1);
}

```

Εδώ το φανάρι των πεζών μένει πράσινο για όσο έχουμε ορίσει το timer(15 δευτερόλεπτα) και έπειτα κλείνει το led ( δηλαδή γίνεται κόκκινο το φανάρι των πεζών) και γίνεται πράσινο το φανάρι του μεγάλου δρόμου. Επιστρέφουμε δηλαδή στην κατάσταση πριν το interrupt:

The screenshot shows the Atmel Studio interface with the following details:

- Solution Explorer:** Shows the project "Atskh02" with files "main.c" and "main.while.s".
- Code Editor:** Displays the assembly code for "main.while.s". The code includes instructions for setting up timer 0, enabling interrupts, and handling them. It also includes comments in German.
- Registers:** Shows the state of registers like R0-R31, RSP, and PC.
- Memory:** A dump of memory starting at address 0x00000000, showing data for variables like `intdata` and `intflags`.
- Call Stack:** Shows the current stack frame.
- Output:** Shows the build log and other output messages.

The screenshot shows the Atmel Studio interface with the following windows:

- Solution Explorer**: Shows the project structure for "Atkashh2" with files like main.c.
- Properties**: Shows the properties for the selected file.
- Memory**: Displays memory dump from address 0x000000 to 0x0000FF, showing data for EEPROM EPROM.
- Call Stack**: Shows the current call stack.
- Command Window**, **Immediate Window**, **Output**, **Memory**: Standard development command windows.
- Autos**, **Locals**, **Watch 1**, **Watch 2**: Standard debugging watch windows.

The code in the main.c file is as follows:

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

void setup() {
    // Set PORTB as output
    DDRB = _BV(PIN2);
    // Set PORTC as output
    DDRC = _BV(PIN2);
}

void loop() {
    // Turn on LED
    PORTB |= _BV(PIN2);
    // Turn off LED
    PORTB ^= _BV(PIN2);

    // Wait for 1 second
    delay_ms(1000);
}
```

## Επιπλέον γλωποίηση μόνο με delay

Ο κώδικας είναι ο ακόλουθος:

```
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <stdio.h>

#define del1 10
#define del2 3
#define ped 30

int interr=0; //logic flag
int g=0;
int main(void){

    short int rnd,tyxaios_arithmos;
    //fanari megalou dromou
    PORTD.DIR |= PIN0_bm; //PIN is output
    PORTD.OUT |= PIN0_bm;

    //fanari pezwn
    PORTD.DIR |= PIN1_bm; //PIN is output

    //fanari mikrou dromou
    PORTD.DIR |= PIN2_bm; //PIN is output

    //pullup enable and Interrupt enabled with sense on both edges
    PORTF.PIN5CTRL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
    sei(); //enable interrupts
    while (1) {
        //dinoyme mia random timh opws orizei h ekfwnhsh
        tyxaios_arithmos=rand();

        //kanoume extract to teleutaio psifio gia sygrishsh parakatw
        rnd=tyxaios_arithmos % 10;
        printf("%d",rnd);

        if(rnd==0 || rnd==5 || rnd==8)
        {
            //anavoume to fanari toy mikrou dromou kai kleinoume tou mikrou
            PORTD.OUTCLR= PIN0_bm; //kane kokkino to fanari tou megalou dromou
            _delay_ms(del2); //mikrh kathusterhsh apo thn stigmh pou ginetai prassino to fanari
            PORTD.OUT |= PIN2_bm; //kane prassino to fanari toy mikrou dromou
            g=1;
        }
        else{
            PORTD.OUTCLR= PIN2_bm;//kane kokkino to fanari toy mikrou dromoy ean den einai kokkino

            _delay_ms(del2);//kathusterhsh gia to pote tha ginei prassino to fanari
            PORTD.OUT |= PIN0_bm;//kane prasino to fanari toy megaloy dromou ean den einai prassino
            g=0;
        }
    }
}
```

```

if (interr==1)
{
    PORTD.OUTCLR= PIN0_bm;//kane kokkino to fanari toy megaloy dromou
    _delay_ms(del2);
    PORTD.OUT |= PIN1_bm;//kane prasino to fanari twn pezwn

    PORTD.OUT |= PIN2_bm;//kane prasino to fanari toy mikrou dromou

    _delay_ms(del1);
    interr=0;

    PORTD.OUTCLR= PIN1_bm;//kane kokkino to fanari twn pezwn

    if(g==0){
        PORTD.OUTCLR= PIN2_bm;//kane kokkino to fanari toy mikrou dromou
        _delay_ms(del2);//kathusterhsh gia to pote tha ginei prassino to fanari
        PORTD.OUT |= PIN0_bm;//kane prasino to fanari toy megaloy dromou
    }
}

}

ISR(PORTF_PORT_vect){
    //clear the interrupt flag
    int intflags = PORTF.INTFLAGS;
    PORTF.INTFLAGS=intflags;
    interr=1;
    sei(); //enable interrupts
}

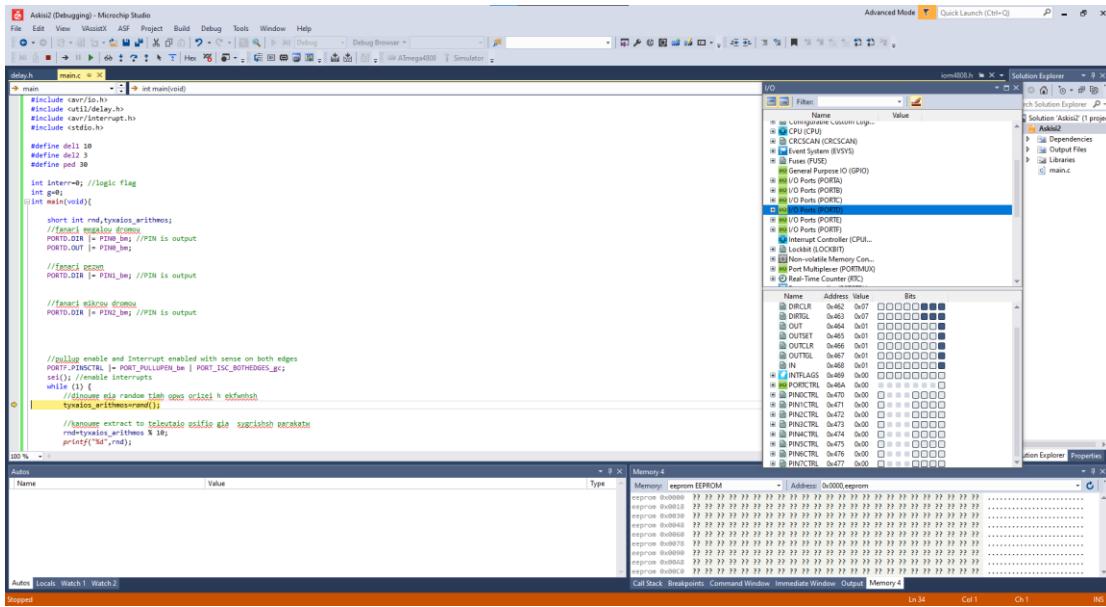
```

### Παρατηρήσεις σχετικά με τον κώδικα:

Αρχικά κάνουμε την παραδοχή πως το φανάρι του μεγάλου δρόμου μένει συνέχεια ανοιχτό έως ότου υπάρξει κάποια διακοπή όπου και γίνεται κόκκινο. Οπότε ξεκινώντας τρέχουμε τον κώδικα βήμα - βήμα με f10 .Το PIN0 αντιστοιχεί στο φανάρι του μεγάλου δρόμου, το PIN1 στο φανάρι των πεζών και το PIN2 στο φανάρι του μικρού δρόμου.

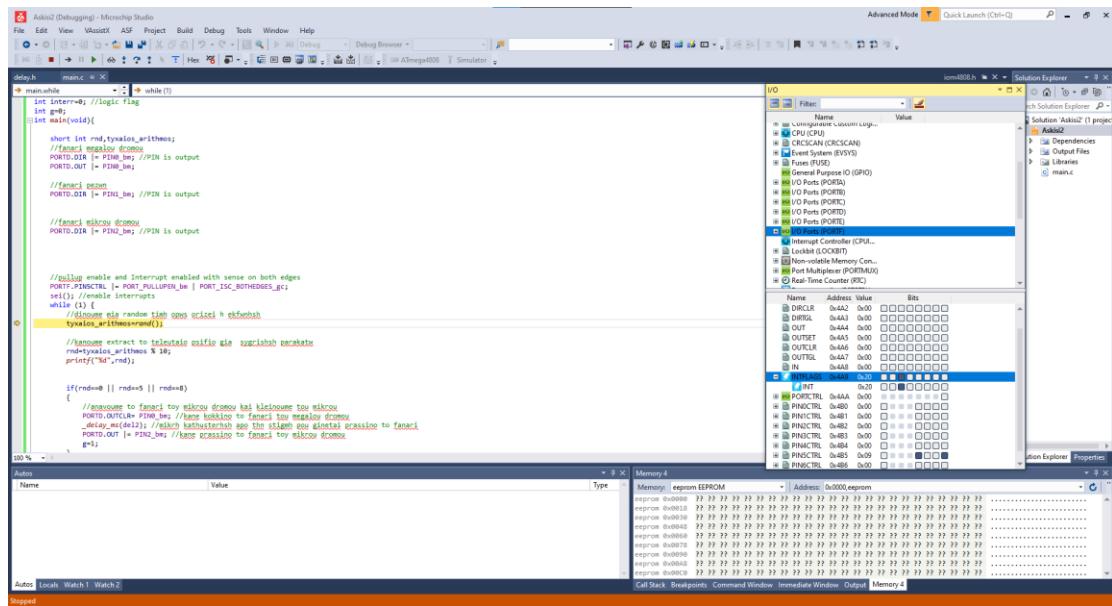
## ΑΠΟΤΕΛΕΣΜΑΤΑ ΚΩΔΙΚΑ:

1) Αρχική κατάσταση τρέχουμε μέχρι το σημείο που φαίνεται στο screenshot όπου είναι (ανοιχτό) πράσινο φανάρι μόνο του μεγάλου δρόμου (PIN0) και κόκκινα (κλειστά) τα PIN1 και PIN2 που αντιστοιχούν στο φανάρι μικρού δρόμου και φανάρι πεζών αντίστοιχα.

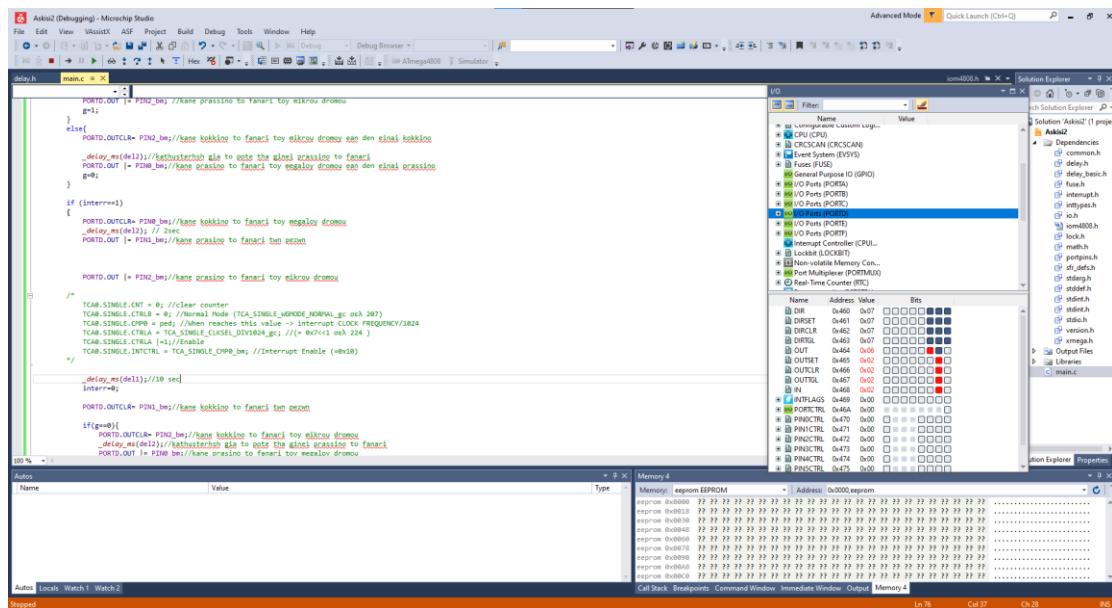


2) Κάνουμε τώρα παραδοχή ότι κάποιος πεζός πατάει το κουμπί (PIN5 PORTF για εμάς) και δημιουργείται interrupt έτσι ώστε να ανάψει κόκκινο το φανάρι του μεγάλου δρόμου και πράσινο στο φανάρι των πεζών και το φανάρι του μικρού δρόμου. Επίσης έχουμε προσθέσει delay στον κώδικα έτσι ώστε όταν ανάβει κόκκινο φανάρι στον μεγάλο δρόμο να μην ανάβει απευθείας πράσινο φανάρι στο φανάρι των πεζών και το φανάρι του μικρού δρόμου.

α)



β)



Σε αυτή την κατάσταση βάλαμε delay αντί για το timer. Το delay εδώ ουσιαστικά προσομοιώνει το timer που ορίζει πόσο θα μείνει ανοιχτό το φανάρι των πεζών. Μόλις λήξει το delay κλείνει το φανάρι των πεζών:

Aksil2 (Debugging) - Microchip Studio

File Edit View VAS5xx AS Project Build Debug Tools Window Help

File Explorer Advanced Mode Quick Launch (Ctrl+Q)

delay.h main.c

```
delay.h
main.c
```

else{  
 PORTD\_OUTCLR = PIN2\_bm; //kane kokkingo to fanari toy mikrou dromou can den elmai kokkingo  
  
 \_delay\_ms(ms); //athastherash glie to pose the ginel grassingis to fanari  
 PORTD\_OUT |= PIN0\_bm; //kane grassingis to fanari toy megaloj dromou can den elmai grassingis  
 ms;  
}  
  
if (intererrr == 1){  
 PORTD\_OUTCLR = PIN2\_bm; //kane kokkingo to fanari toy mikrou dromou  
 \_delay\_ms(ms); //pose the ginel grassingis to fanari  
 PORTD\_OUT |= PIN0\_bm; //kane grassingis to fanari toy megaloj dromou  
  
 PORTD\_OUT |= PIN2\_bm; //kane grassingis to fanari toy mikrou dromou  
  
 TCA0\_SINGLE.CTR0 = 4; //Clear counter  
 TCA0\_SINGLE.CTR0 += 8; //Normal mode (TCA\_SINGLE\_MODE\_NORMAL\_gc == 287)  
 TCA0\_SINGLE.CWP0 = ped; //When reaches this value -> interrupt CLOCK\_40MHz /1024  
 TCA0\_SINGLE.CTR0 = TCA\_SINGLE\_CTR0\_EVEN04\_gc; // 0x0<=c1 <= 224  
 TCA0\_SINGLE.CTR0 += 1; //TCA0\_SINGLE\_CTR0\_EVEN04\_gc  
 TCA0\_SINGLE.INTEN0 |= TCA\_SINGLE\_CWP0\_bm; //Interrupt enable (m=0)  
}  
  
\_delay\_ms(ms); //10 sec  
intererrr = 0;  
  
PORTD\_OUTCLR = PIN2\_bm; //kane kokkingo to fanari toy mikrou dromou  
\_delay\_ms(ms); //athastherash glie to pose the ginel grassingis to fanari  
PORTD\_OUT |= PIN0\_bm; //kane grassingis to fanari toy megaloj dromou  
}

100%

Name Value Type  
OUTCLR 0 int(data)  
9 0 int(data)

I/O Filter: Name Value  
CPU (CPU)  
CRISCAN (CRISCAN)  
Memory (EVRAM)  
Fuses (FUSE)  
General Purpose IO (GPIO)  
I2C (I2C)  
I/O Ports (PORT)  
I/O Ports (PORTC)  
I/O Ports (PORTD)  
I/O Ports (PORTF)  
Memory Controller (MC)  
Locksmith (LOCKSMITH)  
Non-volatile Memory Con.  
Port Multiplexer (PORTMU)  
Real-Time Counter (RTC)

Name Address Value Bits  
DIRA 0x460 0x00 00000000  
DIRB 0x461 0x07 00000111  
DIRC 0x462 0x07 00000111  
DIRD 0x463 0x07 00000111  
OUTA 0x464 0x04 00000100  
OUTB 0x465 0x04 00000100  
OUTC 0x466 0x04 00000100  
OUTD 0x467 0x04 00000100  
INTFLAG 0x469 0x00 00000000  
PORTC 0x46A 0x00 00000000  
PORTD 0x46B 0x00 00000000  
PINCCTRL 0x471 0x00 00000000  
PINCTRL 0x472 0x00 00000000  
PINACTRL 0x473 0x00 00000000  
PINFCCTRL 0x474 0x00 00000000  
PINICCTRL 0x475 0x00 00000000

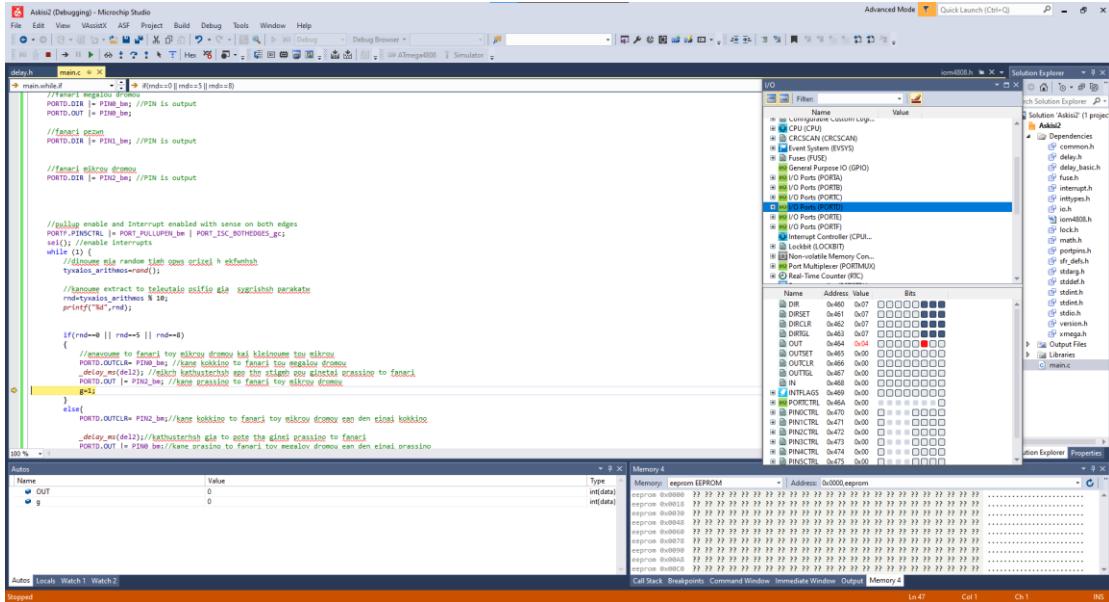
iom808.h

Advanced Mode Solution Explorer Properties

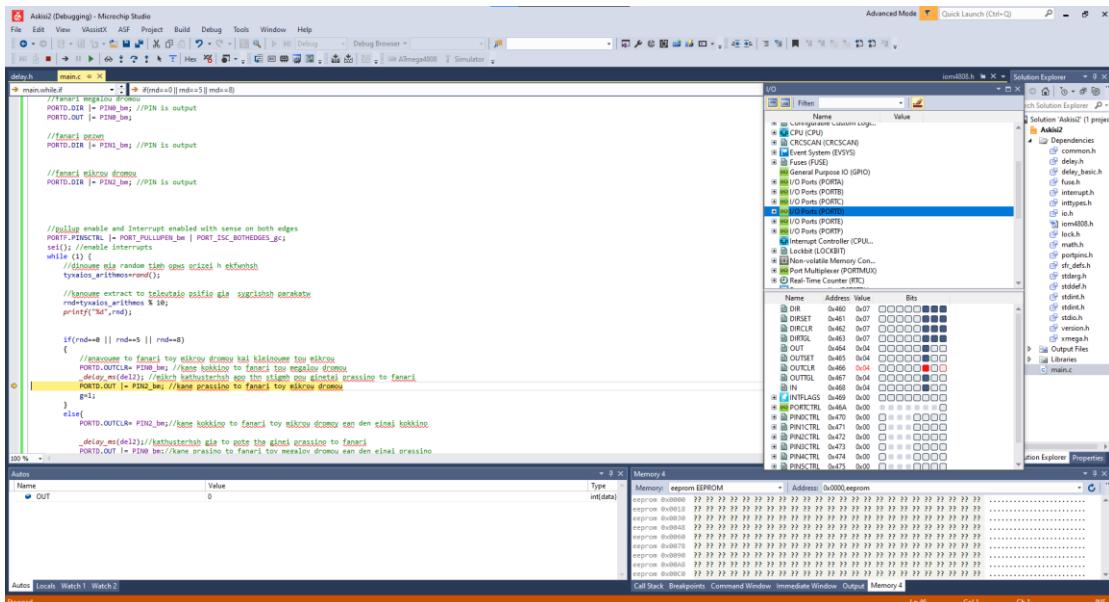
Solution Explorer: Aksil2 [1 project]  
Dependencies common.h delay.h delay\_basic.h float.h intermpt.h inttypes.h io.h math.h portpins.h stdarg.h stdint.h stdio.h stdlib.h version.h xmega.h  
Output Files Libraries main.c  
Properties

γ) Επίσης με την if ουσιαστικά επιστρέφουμε στην κατάσταση πριν κληθεί το interrupt δηλαδή πράσινο στο φανάρι του μεγάλου δρόμου(OUT PIN0 ανοιχτό, δηλαδή=1) όπως ορίζει το διάγραμμα ροής:

3) Είμαστε πάλι στην αρχική κατάσταση(φανάρι πράσινο στο μεγάλο δρόμο) και έρχεται αυτοκίνητο στο φανάρι του μικρού δρόμου (χωρίς να έχει πατηθεί κουμπί από κάποιο πεζό). Εδώ μόλις ο αισθητήρας (που εδώ έχουμε προσομοιώσει με την συνάρτηση rand()) ενεργοποιηθεί ανάβει κόκκινο στο φανάρι του μεγάλου δρόμου και έπειτα από χρόνο ίσο με αυτόν που έχουμε ορίσει στο delay ανάβει πράσινο στο φανάρι του μικρού δρόμου:



Από εδώ αν έρθει πάλι αυτοκίνητο (δηλαδή αριθμός από την rand που τελειώνει σε 0 ή 5 ή 8 ) το φανάρι του μικρού δρόμου παραμένει πράσινο:



Αν δεν έρθει άλλο αυτοκίνητο και δεν έχει πατηθεί κουμπί από κάποιον πεζό επιστρέφει στην κατάσταση πράσινο φανάρι μεγάλου δρόμου:

The screenshot shows the Atmel Studio 7 IDE interface for AVR microcontroller development. The main window displays assembly code for the ATmega4809 chip. The code includes various instructions like `ldi`, `out`, `in`, and `reti`, along with comments in Greek. The assembly window has tabs for `Labels`, `Autos`, `Locals`, `Watch 1`, and `Watch 2`. To the right of the assembly window is a memory dump window titled "Memory 4" showing memory from address `0x0000_0000` to `0x0000_00FF`. The right side of the interface features the Solution Explorer, which lists the project files: `AtmelStudio7.sln`, `Dependencies`, `common.h`, `delay.h`, `delay_basic.h`, `fuses.h`, `general.h`, `io.h`, `lock.h`, `math.h`, `portpins.h`, `rfi_afch.h`, `status.h`, `stdlib.h`, `stdint.h`, `version.h`, `xmegah.h`, `Output Files`, and `Libraries`. The Solution Explorer also shows the current file being edited: `mainwhile.s`.

**4) Εχουμε πράσινο στο φανάρι του μικρού δρόμου και κάποιος πεζός πατάει κουμπί να ανάψει πράσινο στο φανάρι των πεζών(PIN1).**

The screenshot shows the Atmel Studio 7 IDE interface. The top menu bar includes File, Edit, View, VausBox, ASF Project, Build, Debug, Tools, Window, Help, Advanced Mode, and Quick Launch (Ctrl+Q). The main window has tabs for delay.h, main.c, and main.h. The code editor displays C code for an ATmega4808 microcontroller. The Solution Explorer on the right lists projects like AtmelStudio and files such as iom4808.h, delay.h, main.c, and fuse.h. The Memory browser at the bottom shows memory starting at address 0x0000\_expon, with a hex dump of the EEPROM contents.

A screenshot of the Microsoft Visual Studio IDE interface. The top menu bar includes File, Edit, View, VASSOS, ASI, Project, Build, Debug, Tools, Window, Help, and Advanced Mode. The title bar says "Atkis2 (Debugging) - Microsoft Studio". The main window shows the code editor with C/C++ files like delay.h, main.c, and main.white.c. The Solution Explorer on the right lists the project "Atkis2" with files like delay.h, delay\_basic.h, delay.h, fuses.h, and main.c. The Registers window at the bottom shows memory addresses from 0x00000000 to 0xFFFFFFFF with various registers like R0-R31, PC, SP, LR, and CPSR.

Το φανάρι των πεζών(PIN1) αφού γίνει πράσινο μένει πράσινο όσο ορίζει το delay και όταν κλείσει και ολοκληρωθεί αυτή η διαδικασία επιστρέφουμε στην κατάσταση που ήμασταν πριν δηλαδή πράσινο στο φανάρι του μικρού δρόμου :

The screenshot shows the Microsoft Visual Studio IDE interface for an AVR project named 'Atk012'. The main window displays the C source code for 'delay.h' and 'main.c'. The code includes various port manipulation functions like PORTD\_DIR, PORTD\_OUT, and PORTD\_SetPin. It also features a timer interrupt setup using the Timer/Counter (T/C) module, specifically T/C0, with prescaler 128 and initial value 0x0000. The code uses the F\_CPU constant defined as 16MHz. The 'Solution Explorer' pane on the right lists the project files: 'Atk012' (solution), 'common' (project), 'delay.h', 'delay\_basic.h', 'fuses' (header), 'fuses.h', 'io.h', 'inttypes.h', 'iobit.h', 'lockit.h', 'math.h', 'portpin.h', 'uf\_ddef.h', 'status.h', 'stddif.h', 'stdint.h', 'version.h', 'xmega.h', 'Output Files', and 'Libraries'. The 'Memory 4' pane at the bottom shows memory dump data for the EEPROM starting at address 0x000000.

Η διαδικασία αυτή επαναλαμβάνεται επ' άπειρον .