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

ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ

$4^{Η}$ ΑΣΚΗΣΗ

21/05/2021

Αλέξανδρος Νατσολλάρη ΑΜ: 1057769

Παναγιώτης Μπαρμπούνης ΑΜ: 1054382

Παραδοχές :

- Η τιμή που διαβάστηκε από την μνήμη βρίσκεται στα LED 0 έως 3 του PORTD με το LED0 να είναι το λιγότερο σημαντικό δυαδικό ψηφίο και το LED3 το πιο σημαντικό δυαδικό ψηφίο. Δηλαδή έστω ότι διαβάζεται ο αριθμός 3(δεκαδικό)-0011(δυαδικό) τότε LED3=0(άρα κλειστό) LED2=0(άρα κλειστό) LED1=1(άρα ανοιχτό) LED0=1(άρα ανοιχτό).
- Ο παλμός CMP0 είναι στο PIN6 του PORTD και χρησιμοποιείται για την λειτουργία write και ο παλμός CMP1 είναι στο PIN7 του PORTD και χρησιμοποιείται για την λειτουργία read.
- Για interrupt χρησιμοποιήσαμε για την λειτουργία write το SWITCH5 του PORTF και το SWITCH6 του PORTC για την λειτουργία read.
- Για να κάνουμε μετατροπή τον δεκαδικό αριθμό από δυαδικό, φτιάξαμε έναν πίνακα αλήθειας όπως φαίνεται και στον κώδικα όπου ανάλογα την τιμή που διαβάζεται στο δεκαδικό από την μνήμη μετατρέπεται στο δυαδικό και βγαίνει στην έξοδο των LED3 LED2 LED1 LED0.
- Εξαιτίας του ότι η τιμή του CMP1 είναι μικρότερη του CMP0 εκτελείται πρώτα το ISR του CMP0 και μετά του CMP1.

Πείραμα

Ο κώδικας της άσκησης είναι ο ακόλουθος:

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


int sw5=0;
int sw6=0;
int TyxaiaTimh=0;
int memory[32];
int n=0;
int ArithmosPoudiavastike=0;
int main(){
	//LED palmwn
	PORTD.DIR |= PIN6_bm; //palmos CMPO
	PORTD.DIR |= PIN7_bm;//palmos CMP1
	//LED exodou
	PORTD.DIR |= PIN0_bm;//least significant bit
	PORTD.DIR |= PIN1_bm;
	PORTD.DIR |= PIN2_bm;
	PORTD.DIR |= PIN3_bm;//most significant bit

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

	//pullup enable and Interrupt enabled with sense on both edges
	//interrapt gia thn leitoyrgia READ
	PORTC.PIN6CTRL |= PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
```

```
	//prescaler=1024
	TCA0.SINGLE.CTRLA=TCA_SINGLE_CLKSEL_DIV1024_gc;
	TCA0.SINGLE.PER = 254; //select the resolution
	TCA0.SINGLE.CMP0 = 127; //select the duty cycle
	TCA0.SINGLE.CMP1 = 90; //select the duty cycle

	//select Single_Slope_PWM
	TCA0.SINGLE.CTRLB |= TCA_SINGLE_WGMODE_SINGLESLOPE_gc;
	//enable interrupt Overflow
	TCA0.SINGLE.INTCTRL = TCA_SINGLE_OVF_bm;
	//enable interrupt COMP0
	TCA0.SINGLE.INTCTRL |= TCA_SINGLE_CMP0_bm;
	TCA0.SINGLE.INTCTRL |= TCA_SINGLE_CMP1_bm;

	TCA0.SINGLE.CTRLA |= TCA_SINGLE_ENABLE_bm; //Enable



	while (1){ }
}


ISR(TCA0_OVF_vect){
	//clear the interrupt flag
	int intflags = TCA0.SINGLE.INTFLAGS;
	TCA0.SINGLE.INTFLAGS = intflags;
	if(sw5==1){


		//vazoume if gia na mhn exoume memory overflow
		if(n<32){
			memory[n]=TyxaiaTimh;
			printf("memomry cell %d: %d",n,memory[n]);
			n++; //counter pou deixnei kathe fora se poio keli na apothikeusoume thn teleutaia
timh
			//wste na mhn grafoyme panw se alles
		}
		else if(n>=32){
			printf("mememory is full\n");
		}

		sw5=0;
	}

	PORTD.OUT |= PIN6_bm; //Otan ektelestei h entolh anapse LED6
	PORTD.OUT |= PIN7_bm; //Otan ektelestei h entolh anapse LED7
}


ISR(TCA0_CMP0_vect){
	//clear the interrupt flag
	int intflags = TCA0.SINGLE.INTFLAGS;
	TCA0.SINGLE.INTFLAGS = intflags;
	if(sw5==1){


		//vazoume if gia na mhn exoume memory overflow
		if(n<32){
			memory[n]=TyxaiaTimh;
			printf("memomry cell %d: %d",n,memory[n]);
			n++; //counter pou deixnei kathe fora se poio keli na apothikeusoume thn teleutaia
timh
```

```
                        //wste na mhn grafoyme panw se alles
                }
                else if(n>=32){
                        printf("mememory is full\n");

                }
                sw5=0;
        }

        PORTD.OUTCLR = PIN6_bm; //Otan ektelestei h entolh kleise to LED6
}

ISR(TCA0_CMP1_vect){
        //clear the interrupt flag
        int intflags = TCA0.SINGLE.INTFLAGS;
        TCA0.SINGLE.INTFLAGS = intflags;
        //sbhnoume ta LED exodoy meta apo mia periodo
        PORTD.OUTCLR = PIN0_bm;//0
        PORTD.OUTCLR = PIN1_bm;//0
        PORTD.OUTCLR = PIN2_bm;//0
        PORTD.OUTCLR = PIN3_bm;//0
        if(sw6==1){
                ArithmosPoudiavastike=memory[n-1];
                printf("%d",ArithmosPoudiavastike);
                sw6=0;
                //ftiaxame ton pinaka alhtheias gia ta 4Bit ths RAM
                //wste na anaboun ta katalhla LED pou anaparhstoun to READ
                if(ArithmosPoudiavastike==0){
                        PORTD.OUTCLR = PIN0_bm;//0 least significant bit
                        PORTD.OUTCLR = PIN1_bm;//0
                        PORTD.OUTCLR = PIN2_bm;//0
                        PORTD.OUTCLR = PIN3_bm;//0 most significant bit
                }
                else if (ArithmosPoudiavastike==1){
                        PORTD.OUT |= PIN0_bm;  //1 least significant bit
                        PORTD.OUTCLR = PIN1_bm;//0
                        PORTD.OUTCLR = PIN2_bm;//0
                        PORTD.OUTCLR = PIN3_bm;//0 most significant bit
                }
                else if (ArithmosPoudiavastike==2){
                        PORTD.OUTCLR = PIN0_bm;//0 least significant bit
                        PORTD.OUT |= PIN1_bm;  //1
                        PORTD.OUTCLR = PIN2_bm;//0
                        PORTD.OUTCLR = PIN3_bm;//0 most significant bit
                }
                else if (ArithmosPoudiavastike==3){
                        PORTD.OUT |= PIN0_bm;  //1 least significant bit
                        PORTD.OUT |= PIN1_bm;  //1
                        PORTD.OUTCLR = PIN2_bm;//0
                        PORTD.OUTCLR = PIN3_bm;//0 most    significant bit
                }
                else if (ArithmosPoudiavastike==4){
                        PORTD.OUTCLR = PIN0_bm;//0 least significant bit
                        PORTD.OUTCLR = PIN1_bm;//0
                        PORTD.OUT |= PIN2_bm;  //1
                        PORTD.OUTCLR = PIN3_bm;//0 most significant bit
                }
                else if (ArithmosPoudiavastike==5){
                        PORTD.OUT |= PIN0_bm;  //1 least significant bit
                        PORTD.OUTCLR = PIN1_bm;//0
                        PORTD.OUT |= PIN2_bm;  //1
                        PORTD.OUTCLR = PIN3_bm;//0 most significant bit
```

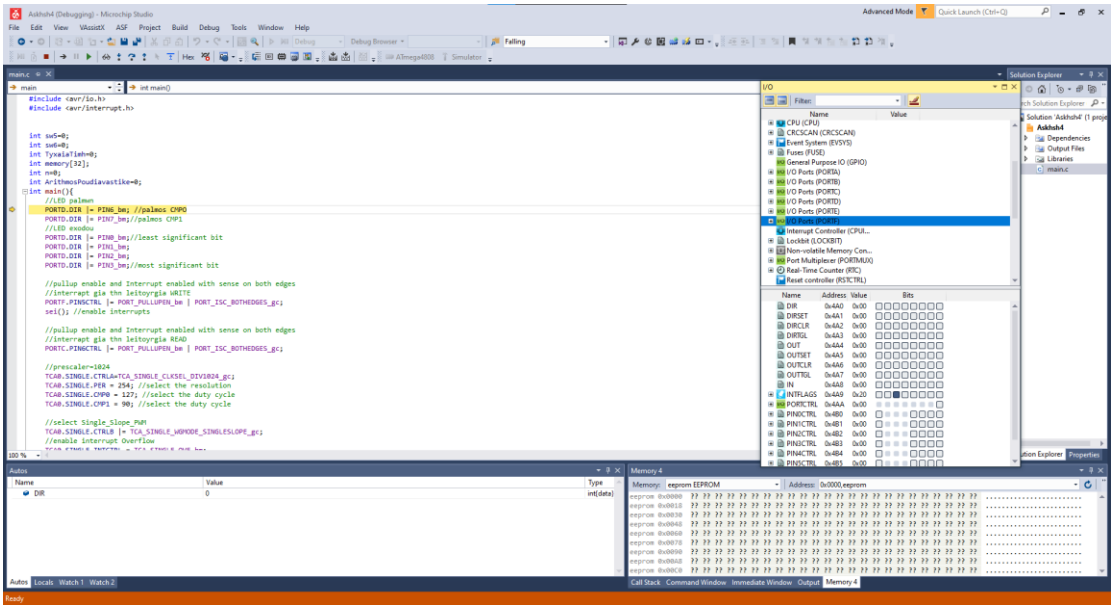
```
        }
        else if (ArithmosPoudiavastike==6){
               PORTD.OUTCLR = PIN0_bm;//0 least significant bit
               PORTD.OUT |= PIN1_bm;  //1
               PORTD.OUT |= PIN2_bm;  //1
               PORTD.OUTCLR = PIN3_bm;//0 most significant bit
        }
        else if (ArithmosPoudiavastike==7){
               PORTD.OUT |= PIN0_bm;  //1 least significant bit
               PORTD.OUT |= PIN1_bm;  //1
               PORTD.OUT |= PIN2_bm;  //1
               PORTD.OUTCLR = PIN3_bm;//0 most significant bit
        }
        else if (ArithmosPoudiavastike==8){
               PORTD.OUTCLR = PIN0_bm;//0 least significant bit
               PORTD.OUTCLR = PIN1_bm;//0
               PORTD.OUTCLR = PIN2_bm;//0
               PORTD.OUT |= PIN3_bm;  //1 most significant bit
        }
        else if (ArithmosPoudiavastike==9){
               PORTD.OUT |= PIN0_bm;  //1 least significant bit
               PORTD.OUTCLR = PIN1_bm;//0
               PORTD.OUTCLR = PIN2_bm;//0
               PORTD.OUT |= PIN3_bm;  //1 most significant bit
        }
        else if (ArithmosPoudiavastike==10){
               PORTD.OUTCLR = PIN0_bm;//0 least significant bit
               PORTD.OUT |= PIN1_bm;  //1
               PORTD.OUTCLR = PIN2_bm;//0
               PORTD.OUT |= PIN3_bm;  //1 most significant bit
        }
        else if (ArithmosPoudiavastike==11){
               PORTD.OUT |= PIN0_bm;  //1 least significant bit
               PORTD.OUT |= PIN1_bm;  //1
               PORTD.OUTCLR = PIN2_bm;//0
               PORTD.OUT |= PIN3_bm;  //1 most significant bit
        }
        else if (ArithmosPoudiavastike==12){
               PORTD.OUTCLR = PIN0_bm;//0 least significant bit
               PORTD.OUTCLR = PIN1_bm;//0
               PORTD.OUT |= PIN2_bm;  //1
               PORTD.OUT |= PIN3_bm;  //1 most significant bit
        }
        else if (ArithmosPoudiavastike==13){
               PORTD.OUT |= PIN0_bm;  //1 least significant bit
               PORTD.OUTCLR = PIN1_bm;//0
               PORTD.OUT |= PIN2_bm;  //1
               PORTD.OUT |= PIN3_bm;  //1 most significant bit
        }
        else if (ArithmosPoudiavastike==14){
               PORTD.OUTCLR = PIN0_bm;//0 least significant bit
               PORTD.OUT |= PIN1_bm;  //1
               PORTD.OUT |= PIN2_bm;  //1
               PORTD.OUT |= PIN3_bm;  //1 most significant bit
        }
        else if (ArithmosPoudiavastike==15){
               PORTD.OUT |= PIN0_bm;  //1 least significant bit
               PORTD.OUT |= PIN1_bm;  //1
               PORTD.OUT |= PIN2_bm;  //1
               PORTD.OUT |= PIN3_bm;  //1 most significant bit
        }
```

```
	}

	PORTD.OUTCLR = PIN7_bm; //Otan ektelestei h entolh kleise to LED7
}

//interrapt gia thn leitoyrgia WRITE
//energopoihte me to PIN5 tou PORTF
ISR(PORTF_PORT_vect){
	int intflags = PORTF.INTFLAGS;
	PORTF.INTFLAGS=intflags;
	TyxaiaTimh=rand() % (15 + 1 - 0) + 0;


	sw5=1;//timh poy dhlwnei oti energopoihthike o SWITCH5
}

//interrapt gia thn leitoyrgia READ
//energopoihte me to PIN6 tou PORTC
ISR(PORTC_PORT_vect){
	int intflags = PORTC.INTFLAGS;
	PORTC.INTFLAGS=intflags;

	sw6=1;//timh poy dhlwnei oti energopoihthike o SWITCH6
}
```

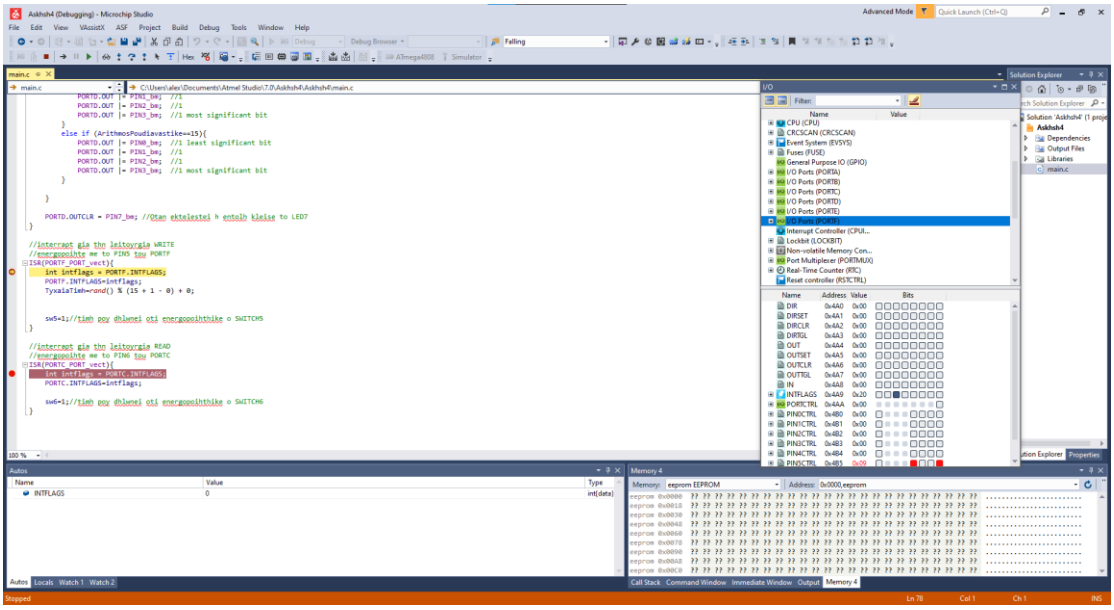
Αποτελέσματα Εκτέλεσης

Ξεκινώντας βάζουμε breakpoint στα εξής σημεία για να ελέγχουμε τα ενδιάμεσα αποτελέσματα(να σημειώσουμε εδώ ότι δεν στέλνουμε το zip με τα breakpoint ήδη μέσα γιατί κατά το testing παρατηρήσαμε ένα bug, καθώς αφού έστειλε ο ένας το zip στον άλλο τα breakpoint δεν δούλευαν κατά την εκτέλεση του κώδικα και υπήρχε πρόβλημα): γραμμή 54, 62, 67, 80, 88, 93, 104, 113, 216, 222, 233.

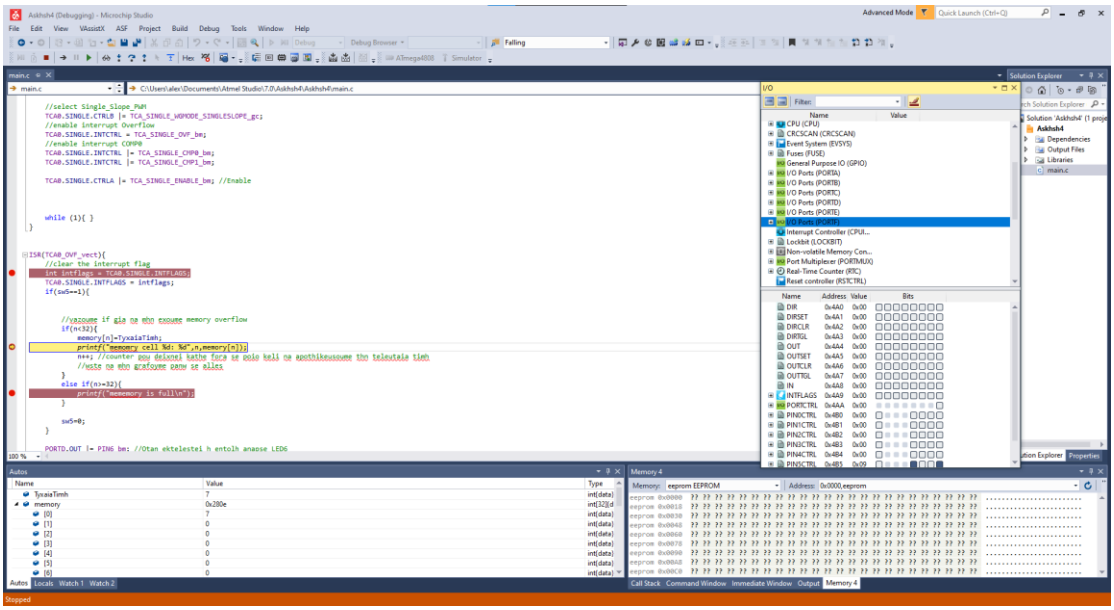
Ξεκινώντας πατάμε SWITCH5 του PORTF για να εκτελέσουμε την λειτουργία write :



Έπειτα πατάμε run και ενεργοποιείται το interrupt `ISR(PORTF_PORT_vect)` όπου αποθηκεύεται μια τυχαία τιμή από την συνάρτηση rand() στην μεταβλητή `TyxaiaTimh` και η μεταβλητή sw5 γίνεται ίση με 1 ώστε να ενεργοποιηθεί η διαδικασία write σε rising/falling edge και μόνο τότε.



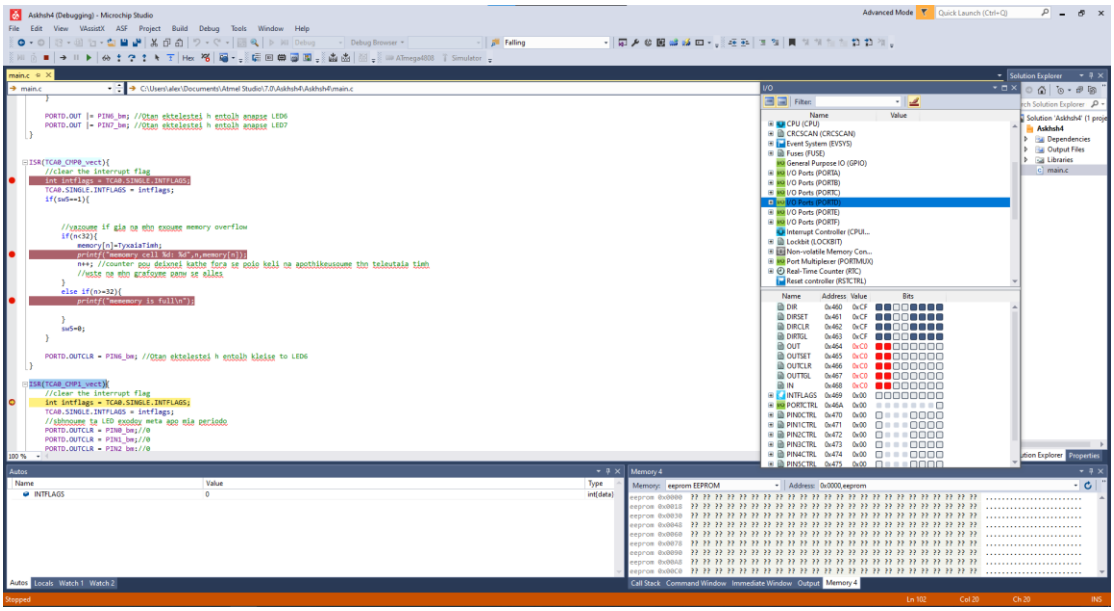
Μόλις ενεργοποιηθεί το write γράφεται η τυχαία τιμή της μεταβλητής TyxaiaTimh που παίρνει τιμές από την rand() στο πρώτο κελί του πίνακα memory[].

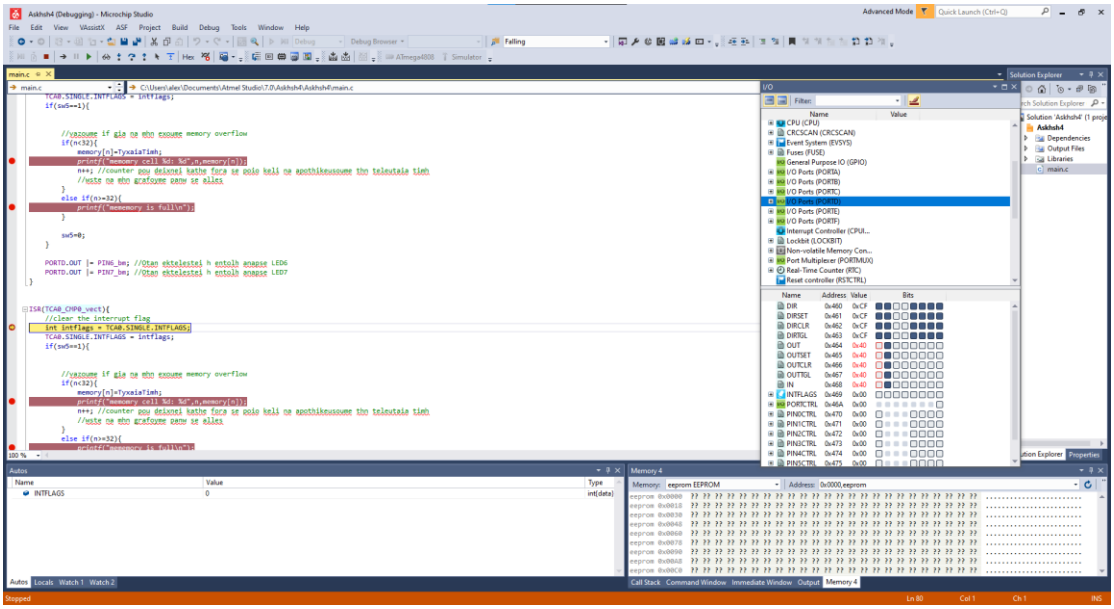


Στο ISR(TCA0_OVF_vect) κάθε φορά που εκτελείται και μόλις εκτελεστούν οι εντολές

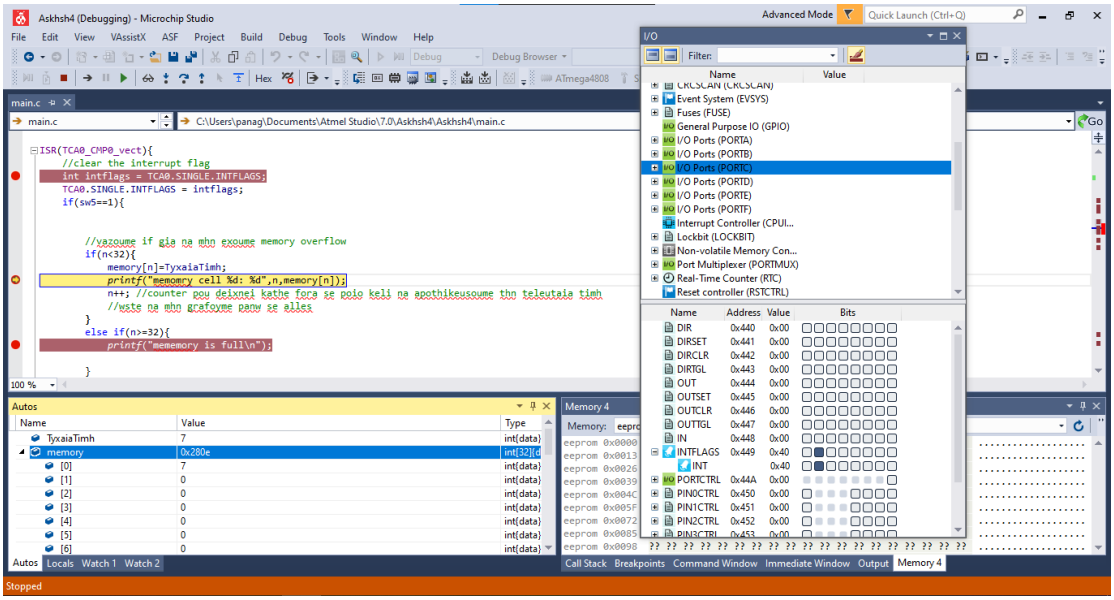
```
PORTD.OUT |= PIN6_bm; //Otan ektelestei h entolh anapse LED6
PORTD.OUT |= PIN7_bm; //Otan ektelestei h entolh anapse LED7
```

ανάβουν τα led6 και led7 τα οποία δηλώνουν ότι οι παλμοί είναι σε υψηλή στάθμη και κλείνουν στα ISR(TCA0_CMP0_vect) και ISR(TCA0_CMP1_vect) αντίστοιχα, για να δείξουν ότι οι παλμοί πάνε σε χαμηλή στάθμη.



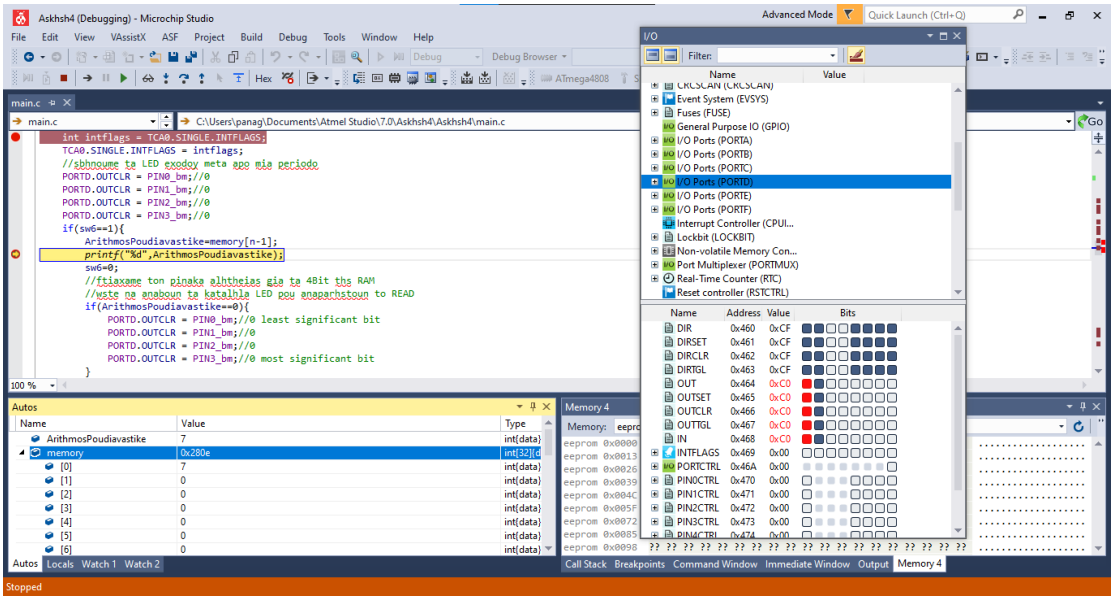


Αφού αποθηκευτεί η τιμή της rand() στην μνήμη(για εμάς ο πίνακας memory[] που αντιπροσωπεύει την μνήμη) πατάμε το SWITCH6 του PORTC ώστε να ξεκινήσει η διαδικασία του read του περιεχομένου της μνήμης. Άρα η ροή του κώδικα πάει στο `ISR(PORTC_PORT_vect)` όπου η τιμή της μεταβλητής sw6 γίνεται ίση με ένα για να ενεργοποιηθεί η διαδικασία read στο `ISR(TCA0_CMP1_vect)`. Αν δεν πατηθεί το SWITCH6 του PORTC μεταφερόμαστε στην εκκίνηση.



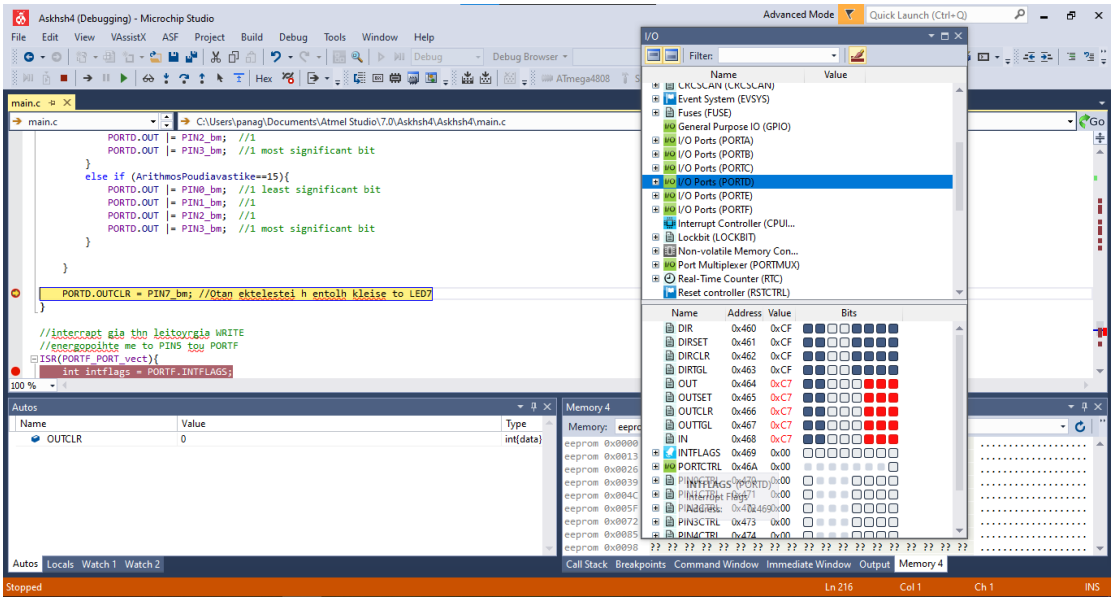
Στην συνέχεια η ροή του κώδικα πηγαίνει στο `ISR(TCA0_OVF_vect)` και έπειτα στο `ISR(TCA0_CMP1_vect)` όπου ξεκινά η διαδικασία read.

Όπως φαίνεται εδώ διαβάζεται η τελευταία τιμή που γράφτηκε στην μνήμη (πίνακας memory[]).



Στην συνέχεια αφού διαβάσει την τιμή και έχουμε Falling edge η τιμή εμφανίζεται στα

LED3 LED2 LED1 LED0. Εδώ αφού διαβάστηκε η τιμή 7 (0111 δυαδικό) τα LED θα γίνουν LED3=0(άρα κλειστό) LED2=1(άρα ανοιχτό) LED1=1(άρα ανοιχτό) LED0=1(άρα ανοιχτό).



Στα LED3 LED2 LED1 LED0 παραμένει η τιμή που διαβάστηκε για μια περίοδο του παλμού CMP1 όπου σβήνουν.

Αν αφού εμφανιστεί η τιμή στα LEDs και πατηθεί το SWITCH5 του PORTF κατά την διάρκεια της λειτουργίας read μεταφερόμαστε πάλι στην λειτουργία write. Αν δεν πατηθεί το SWITCH5 του PORTF κατά την διάρκεια της λειτουργίας read μεταφερόμαστε στην εκκίνηση.