LABORATOR 10 – Ceas prin întreruperi

Prezentare

Programul „ceas” din laboratorul 9 îndeplineşte 2 sarcini:

1. Afişează tasta apăsată pe linia 1.
2. Calculează timpul şi îl afişează pe linia 2

În mod normal orice program are de făcut mai multe prelucrări ca urmarea a unui eveniment. Un astfel de eveniment este apăsarea unei taste. După apăsarea unei taste programul „ceas” din laboratorul 9 face foarte puţine prelucrări, mai exact doar afişează caracterul corespunzător tastei. Ar trebui să adăugăm alte funcţiuni la acest program pentru a creşte timpul de execuţie. Pentru a nu complica prea mult programul vom simula un timp mare de execuţie cu funcţia *wait*.

După ce adăugăm *wait*(…) observăm ca ceasul nu mai funcţionează corect. Pe durata aşteptării cu *wait* ceasul va îngheța şi va rămâne în urmă. Acestea se întâmplă deoarece timpul de execuție a buclei principale **a devenit prea mare**. Soluţia constă în rezolvarea sarcinii 2 prin întreruperi.

Pentru sarcina 2 – Calculează şi afişează timpul – vom muta prelucrările executate la ciclarea timerului 2 din bucla `while(1)` într-o o rutină apelabilă pe întrerupere. Ciclarea timerului 2 va genera un apel către această rutina de întrerupere iar aici vom face actualizarea şi afişarea timpului.

Pentru acest laborator parcurgeţi obligatoriu prelegerile 6 şi 5 (în această ordine).

Desfăşurarea lucrării

Pasul 1: Crearea proiectului

Se va crea un proiect nou cu numele **kbcint** (tastatură şi ceas prin întreruperi) care va conţine aceleaşi surse ca proiectul „ceas” din laboratorul 9, adică va fi o copie a proiectului „ceas”. Pentru crearea acestui proiect procedaţi după cum urmează:

a) **Se va crea** (obligatoriu cu NEXT) proiectul cu numele **kbcint.** La crearea proiectului căsuța **Create initial file trebuie** să fie bifată.

b) **Închideţi** AVR Studio.

c) **Copiaţi ceas.c** şi **IOfn.c** din folderul **ceas** în folderul proiectului **kbcint**.

d) În folderul **kbcint ştergeţi** fişierul **kbcint.c** (care este gol, adică are lungimea zero).

e) În folderul **kbcint redenumiţi ceas.c** ca **kbcint.c.**

f) **Deschideţi** (obligatoriu cu NEXT) proiectul kbcint.

g) Proiectul deja conţine fişierul **kbcint.c.** Trebuie adăugat şi fişierul **IOfn.c**. Pentru aceasta faceţi click dreapta pe **Source Files** şi din meniul contextual ce va apare selectaţi **Add existing Source File(s):** Din folderul kbcint **adăugaţi** la proiect fişierul **IOfn.c**.

Pasul 2: Simulare prelucrare de durata mare.

Aşa cum s-a precizat anterior, se va simula o prelucrare de durată mare prin apelarea funcției *wait*(). În secţiunea de afişare a tastei apăsate, după *putchLCD*(kbcode) adăugaţi codul de mai jos:

```
…
putchLCD(kbcode);
wait(25000); //aşteapă 0,1 secunde
gotoLC(1,1);
putchLCD(' ');
gotoLC(1,1);
…
```

Pentru testare sincronizaţi-vă fie cu ceasul calculatorului după procedura explicată în laboratorul ceas, fie cu cronometrul telefonului mobil. După sincronizare apăsaţi repede ce taste doriţi. Observaţi sincronizarea. Pentru ca efectul să fie si mai evident, puteţi schimba parametrul funcţiei *wait* la 3 secunde.

Chemaţi profesorul pentru înregistrarea progresului.

Pentru ca ceasul să nu mai rămâne în urma îl vom muta în rutina de întrerupere.

Pasul 3: Iniţializare sistemului de întreruperi

În general lucru pe întreruperi presupune o succesiune de paşi. Această succesiune trebuie parcursă pentru fiecare cerere de întrerupere (IRQ).

Primii paşi aparţin fazei de iniţializare. Pentru un IRQ oarecare, notat în continuare IRQ*i*, paşii care trebuie urmaţi în faza de iniţializare sunt:

i. Se configurează blocul care generează cererea de întrerupere IRQ*i*.

ii. Se demaschează cererea de întrerupere IRQ*i*.

iii. Se demaschează întreruperea INT. (IF=1)

Paşii i-iii se execută o singură data, înainte de while(1). În cazul timerului 2 pe baza căruia s-a implementat ceasul, aceşti paşi sunt:

i. **Se configurează blocul care generează întreruperea, adică timerul 2.**

Setările timerului 2 din laboratorul 9 au fost preluate în acest laborator şi sunt deja făcute.

ii. **Se demaschează cererea de întrerupere.** În capitolul „Mascarea cererilor de întrerupere” din curs s-a precizat că orice cerere de întrerupere poate fi mascată. Bitul care maschează cererea TOV2 se află în registrul TIMSK:

TIMSK – Timer/Counter Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Daca bitul 7 – **OCIE2: Timer/Counter2 Output Compare Match Interrupt Enable** – este ‚1’, cererea OCF2 este activă, adică poate genera întrerupere. În caz contrar OCF2 este ignorată.

►În secţiunea de iniţializări din *main*, după setările timerelor şi înainte de while(1), **configuraţi OCIE2** pentru a demasca cererea de întrerupere OCF2. Documentaţi (scrieți un comentariu) această setare.

iii. **Se demaschează întreruperea INT**. În subcapitolul „Întreruperi mascabile şi nemascabile” din curs s-a precizat că mascarea întreruperii INT se face prin intermediul bistabilului IF. La Atmega16 acest bistabil se află în registrul SREG şi se numeşte I:

SREG – AVR Status Register

Bit	7	6	5	4	3	2	1	0
	I	T	H	S	V	N	Z	C
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bistabilul IF din curs este bitul I din SREG.

►În secțiunea de inițializări din *main*, după configurarea lui OCIE2, **configuraţi I** pentru a demasca întreruperea procesorului. Documentaţi această setare. Folosiţi macro setbit(…);

Pasul 4: Rutina de întrerupere

Codul pentru ceas, cu mici modificări, va deveni rutina de întrerupere. La scrierea acestei rutine se vor executa următorii paşi. Urmăriţi textul scris cu verde:

a) Rutina de întrerupere asociată lui OCF2 se va scrie în fişierul kbcint.c ce conţine *main*(). Mai întâi includeţi headerul interrupt.h cu:

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

b) Rutina de întrerupere poate fi scrisă înainte sau după *main*. Din motive de claritate a codului o vom scrie după main(). Declaraţia rutina de întrerupere cu:

```
ISR(TIMER2_COMP_vect){
```

ISR este definit în interrupt.h iar TIMER2_COMP_vect în io.h.

c) Variabilele `cycles, sec, min` şi `hrs` declarate până acum în *main* vor deveni variabile locale în ISR. Numele ales de fiecare echipă pentru variabilele secunde, minute şi ore pot să difere. Nu redenumiţi variabilele, folosiţi numele alese iniţial. **Mutaţi declararea acestor variabile din *main* în ISR.**

Atenţie: variabilele locale nu îşi păstrează valoare între apeluri. **Modificaţi** declaraţia variabilelor locale astfel încât aceasta să-şi păstreze valoarea între apeluri.

d) **Mutaţi** codului pentru ceas din main() în ISR.

e) Codul pentru ceas este scris în corpul unui if:

```
    if( TIFR...){
        TIFR=...;
        cycles++;
        if(cycles...){
            ...
        }
    }
```

if-ul care testează OCF2 din TIFR **nu mai este necesar** deoarece setarea lui OCF2 ne duce automat în rutina de întrerupere asociată lui OCF2.

Linia următoare este pentru ştergerea lui TIFR. Nici această linie **nu mai este necesară** deoarece OCF2 este şters automat de hardware la intrarea în ISRul asociat.

f) Terminaţi rutina de întrerupere:

```
}//end ISR
```

Setaţi parametrul funcţiei *wait*(…) la 5 secunde. Testaţi!

Dacă pe durata afișării tastei ceasul se afișează corect, chemați profesorul pentru validare. Dacă ați ajuns aici aveți nota 5.

Partea opțională

Pasul 5: Evidențiere eroare

Chiar dacă ceasul nu mai rămâne în urmă, aplicația nu funcționează corect. Eroarea apare însă foarte rar.

Pentru a evidenția funcționarea eronată vom modifica ce se afișează la apăsarea unei taste: în loc să afișam *k* vrem să apară mesajul „Tasta *k*”, unde *k* este caracterul corespunzător tastei apăsate. Acest mesaj se va afișa din linia 1, coloana 1. Suplimentar, mesajul se va afișa cu pauză de 0,1 secunde între caractere. În principiu afișarea se va face după cum urmează:

```
...
gotoLC(1, 1);
putsLCD("            ");
gotoLC(1, 1);
for(i=0; ...; i++){
   //caracterul i al mesajului "Tasta k"
   putchLCD(caracterul i al mesajului);
   wait(25000);
}
...
```

Dacă aplicația ar funcționa corect, după apăsarea tastei 5 afișorul LCD ar trebui să arate așa:

C L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	T	a	s	t	a		5									
2	1	0	:	1	5	:	0	7								

O variantă de afișare greșită este următoarea:

C L	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	T	a	s													
2	1	0	:	1	5	:	0	7	t	a		5				

Aplicația poate greși în multe moduri, dar întotdeauna prima parte a mesajului se va scrie pe linia 1 din coloana 1 iar restul pe linia 2, după valoarea curentă a timpului.

Dacă apare funcționarea eronată, chemați profesorul pentru validare.

Dacă ați ajuns aici aveți 2 puncte.

Pasul 6: Cauza erorii și soluție.

Explicați din ce cauză apare eroarea și găsiți o soluție. NU implementați soluția.

Când aveți o explicație și o soluție chemați profesorul.

Dacă ați ajuns aici aveți 1 punct.

Pasul 7: Afişarea corectă

Implementați soluţia de la pasul 7.

Când funcţionează chemaţi profesorul.

Dacă aţi ajuns aici aveţi 1 sau 2 puncte, în funcţie de calitatea soluţiei.