

LABORATOR 9 – Ceas digital. Utilizarea timerelor

Se cere implementarea unui **ceas digital** care afișează timpul cu o precizie ce depinde numai de stabilitatea frecvenței semnalului de ceas al microcontrolerul ATmega.

Se reamintește că unitatea de comandă a oricărui procesor trece dintr-o stare în altă la apariția frontului activ al semnalului de ceas. În cazul microcontrolerelor din familia ATmega sunt disponibile trei opțiuni privitor la generarea ceasului:

1. semnalul de ceas este generat **extern**
2. semnalul de ceas este generat **intern** de un oscilator RC.
3. semnalul de ceas este generat **intern** de un amplificator, dar sunt necesare **componente externe** adiționale. Amplificatorul intern împreună cu componentele externe formează un oscilator.

La laboratoarele anterioare s-a folosit numai oscilatorul RC intern, acesta fiind setat să genereze o frecvență de 8 MHz. Această variantă este foarte ieftină și simplă deoarece nu necesită nici o componentă externă, dar are dezavantajul că frecvența ceasului generat are o precizie mică și în plus variază mult cu temperatura. Acest fapt poate fi acceptat în multe aplicații dar este **intolerabil** în aplicația **ceas digital**. Din acest motiv în acest laborator vom folosi varianta 3, adică generare internă cu componente externe suplimentare.

Componenta externă suplimentară folosită în acest laborator va fi un **cristal de cuarț** deoarece oscilatoarele cu cristal de cuarț oferă performanțe foarte bune la un preț de cost mic. Aproape toate ceasurile digitale folosesc această variantă de oscilator. Pentru a înțelege ce înseamnă performanțe bune, vom trece în revistă doi parametri ai oscilatoarelor cu cristal de cuarț.

Pentru oscilatoarele cu cuarț se definesc 4 parametri, dintre care doi sunt mai importanți:

Precizia frecvenței: $P = \text{frecvența măsurată}^{1)} / \text{frecvența înscrisă pe cuarț}$

¹⁾ măsurarea se face la 25°C, presiune de 756mmHg

Pentru oscilatoarele cu cuarț precizia este de ordinul 20 ppm (părți per milion) sau mai bună. Frecvența generată de cuarț depinde de dimensiunile sale fizice obținute prin tăiere, în procesul de fabricație. Diferența între frecvența reală și cea înscrisă se datorează nerepetitivității procesului de fabricație. De exemplu, dacă un cristal este folosit pentru ceas, 20ppm înseamnă într-o lună o deviație de:

$$60 \frac{s}{\min} \times 60 \frac{\min}{h} \times 24 \frac{h}{zi} \times 30 \frac{zile}{luna} \times \left(\pm \frac{20}{1000000} \right) = \pm 51.84 \frac{s}{luna}$$

Stabilitatea frecvenței

Orice dispozitiv electronic este afectat de temperatură. Și frecvența generată de oscilatoarele cu cuarț va varia cu temperatura:

$$S = \text{frecvența generată la temperatura } t^{\circ}\text{C} / \text{frecvența generată la } (t-1)^{\circ}\text{C}$$

Tipic, S are valoarea de 1 ppm/ °C. Pentru a corecta deviația frecvenței cu temperatura în aplicațiile de mare precizie, oscilatoarele se închid în incinte termostate.

Prezentarea Metodei

Varianta soft.

În prelegerea 5, capitolul 1, s-a arătat că este aproape imposibilă măsurarea timpului prin mijloace pur software. Dacă cineva crede că este totuși posibil, să citească acest subcapitol. În caz contrar treceți la subcapitolul următor!

Teoretic, o primă variantă pentru a determina când a trecut o secundă se bazează pe faptul că **orice instrucțiune este executată de procesor într-un număr fix de perioade de ceas**. Această soluție este implementată în software. O variantă a acestei metode conține o buclă de întârziere soft, de tipul

```
for (i=0; i<n; i++) { }
```

, întârzierea depinzând de valoarea lui n :

Sec=0;

La nesfârșit

P1. for(i=0; i<n; i++) { }

P2. Sec++;

P3. Dacă sec==60 sec=0;

P4. Afișează sec;

Repetă.

Pentru ca descrierea de mai sus să fie cât mai simplă s-au luat în considerare numai calculul și afișarea secundelor. Afișarea precisă a secundelor presupune ca buclă *La nesfârșitRepetă* să se execute în exact o secundă, indiferent dacă $sec==0$ sau nu. Pentru ca metoda să funcționeze trebuie ca suma timpilor de execuție pentru toate instrucțiunile cod mașină din care este alcătuită buclă să fie exact o secundă.

Un prim aspect de care trebuie de care trebuie ținut seama este ca **timpul de execuție să nu se schimbe de la o iterație la alta a buclei**. De obicei această condiție nu este îndeplinită. De exemplu pasul *P3: Dacă sec==60 sec=0;* implică timpi de execuție diferiți, după cum variabila sec este sau nu egală cu 60. Dacă $sec==60$ se execută $sec=0$ iar dacă nu se trece direct la pasul următor. Astfel de structuri trebuie echilibrate astfel:

P3: Dacă sec==60 sec=0; altfel așteaptă un timp egal cu timpul necesar să execuți sec=0;

În cazul în care afișarea timpului se face pe LCD, va apare o problemă suplimentară: timpul de execuție al pasului P4 depinde de controlerul LCD. Acesta are propriul generator intern de ceas. Dacă acesta nu se bazează pe cristal de cuarț, orice variație de frecvență a acestuia se traduce prin timpi diferiți de execuție de la o iterație la alta (la execuția `ckbf()`). **Într-o astfel de situație metoda soft NU funcționează!** LCD-ul cu care ați lucrat în laboratoarele anterioare are propriul oscilator și acesta nu se bazează pe cuarț.

Un alt aspect care îngreunează calculul timpului de execuție al buclei este necesitatea de a ști timpii de execuție pentru toate funcțiile din bibliotecă apelate. De exemplu, dacă în buclă se apelează `sprintf(...)` pentru a genera stringul de afișat trebuie să știm timpul său de execuție. Acest timp este aproape imposibil de aflat.

În final trebuie remarcat că dacă procesorul execută buclă de mai sus, nu mai are timp să facă și ceva util.

CONCLUZIE: în general este imposibil sau extrem de greu și cu siguranță ineficient să controlăm riguros timpul prin intermediul buclelor soft.

Varianta hard/soft (CO-DESIGN)

Problema ar putea fi rezolvată mult mai ușor dacă se folosește un **numărător hardware**. În momentul în care numărătorul ciclează (overflows) se setează un bistabil. Fie numele acestui bistabil TF (timer flag). Trebuie subliniat că:

- Numărătorul nu se oprește niciodată. Acesta numără modulo N adică 0, 1, 2,...N-1, 0, 1, 2,... N-1, 0, 1, 2,... N-1.... **Valoarea lui N se va discuta ulterior.**
- Atingerea valorii N-1 setează bistabilul TF (Timer Flag).
- Software-ul poate citi, seta sau reseta TF.

Bazându-ne pe numărător și pe bistabilul TF „Ceas digital” se poate implementa parte în hardware, parte în software astfel:

HARWARE	SOFTWARE
Numărător	TF
0	0 { Așteaptă până când TF devine ,1'
1	0 .
2	0 .
.	.
N-1	0 .
0	1 }
1	1 { Resetează TF.
2	1 .
3	0 }
4	0 { Calculează timpul
5	0 .
6	0 .
.	.
1245	0 }
1246	0 { Dacă este cazul, afișează timpul
1247	0 .
.	.
31236	0 }
31237	0 { Așteaptă până când TF devine ,1'
31238	0 .
31239	0 .
.	.
N-1	0 .
0	1 }
1	1 { Resetează TF.
2	1 .
3	0 }
4	0 { Calculează timpul
5	0 .
6	0 .
.	.

În modelul de mai sus valorile numărătorului au rol de exemplificare.

Această modalitate de implementarea aplicațiilor, în care o parte din funcțiuni sunt implementate software, iar cealaltă parte hardware, se numește CO-DESIGN. Relația dintre partea de hardware și software a aplicației este prezentată în figura următoare:

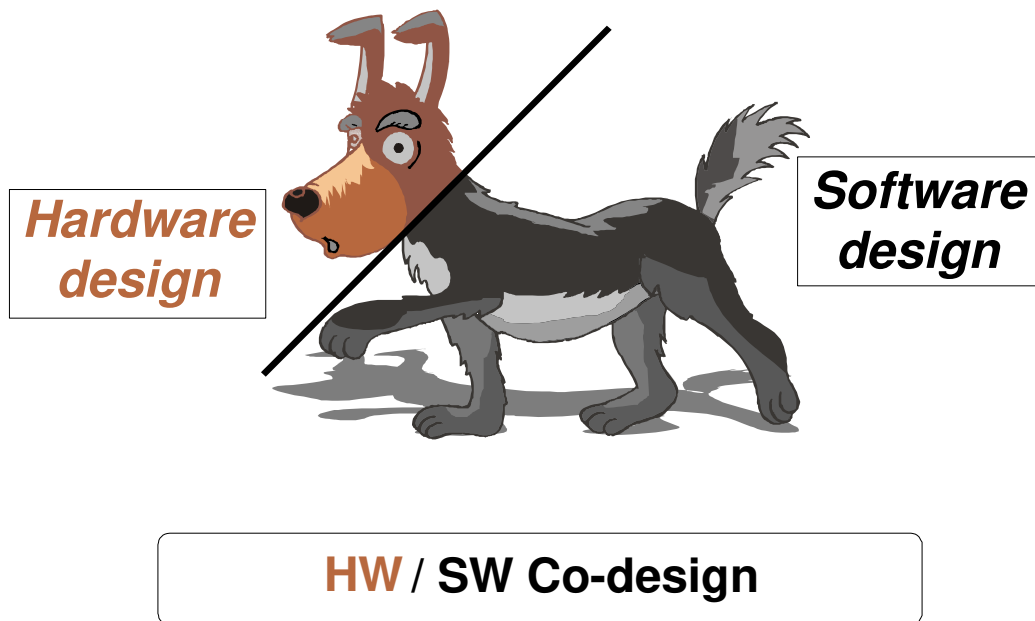


figura 1

Atenție: Metoda hard/soft, expusă mai sus, este de tip polling. Pentru ca polling-ul să funcționeze este necesar ca operațiile software:

1. Resetează TF.
2. Calculează timpul.
3. Afișează timpul.

să se execute într-un timp mai scurt decât un ciclu de numărare. Dacă aceste acțiuni durează mai mult decât un ciclu de numărare, setarea TF nu va fi sesizată și ceasul va rămâne în urmă. În acest caz trebuie folosită metoda întreruperii.

Particularități de implementare

Pentru a trece la implementare este necesar să parcurgeți documentația necesară pentru înțelegerea modului de funcționare a timerelor 0, 1 și 2. Informațiile necesare se află în prelegerea 5 și în fișierul **ATMega16A Data Sheet** (<http://www.cs.ucv.ro/~lemen/Downloads/ATmega16A.pdf#>) începând de la pagina 71 până la pagina 135. Din prelegerea 5 parcurgeți obligatoriu capitolele 2 și 3.1-3.4.

Atenție: în documentație termenul folosit pentru numărător este timer/counter.

SCOPUL LUCRĂRII

Se cere afișarea pe LCD a timpului în formatul **hh:mm:ss**. Orele se vor afișa în format 24 fără zerourile ne semnificative. Exemplu: se va afișa 5:03:40, nu 05:03:40..

Pentru implementarea ceasului digital vom folosi timerul 2. Acest timer va fi folosit pentru implementarea ceasului digital conform modelului din prelegere 5, capitolul 3.4.

Se dă $T_r=1s$ și $f_{CLK_CPU} = 8\text{ MHz}$.

După modelul din curs **faceți calculele pentru determinarea lui p , N și k pentru a obține un T_{cycle} cât mai mare.**

După ce ați determinat p , N , k și T_{cycle} chemați profesorul pentru validare.

Desfășurarea lucrării

Pasul 1: Conectarea cristalului

Modul de conectare al cristalului se face conform specificațiilor din datele de catalog. Acest mod este prezentat în figura 8.2 din **ATMega16A Data Sheet** la pagina 26 și este reluată mai jos:

Figure 8-2. Crystal Oscillator Connections

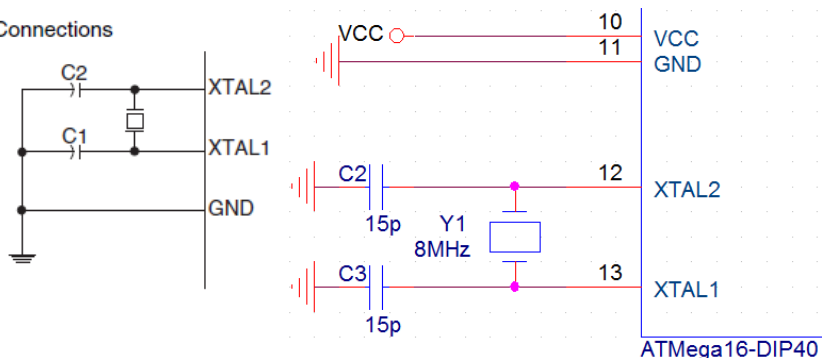


figura 2

Conform indicațiilor din datele de catalog, conectați cristalul ca în figura 2.

Pasul 2: Verificarea montajului

Înainte de a alimenta, verificați că montajul este implementat conform schemei următoare:

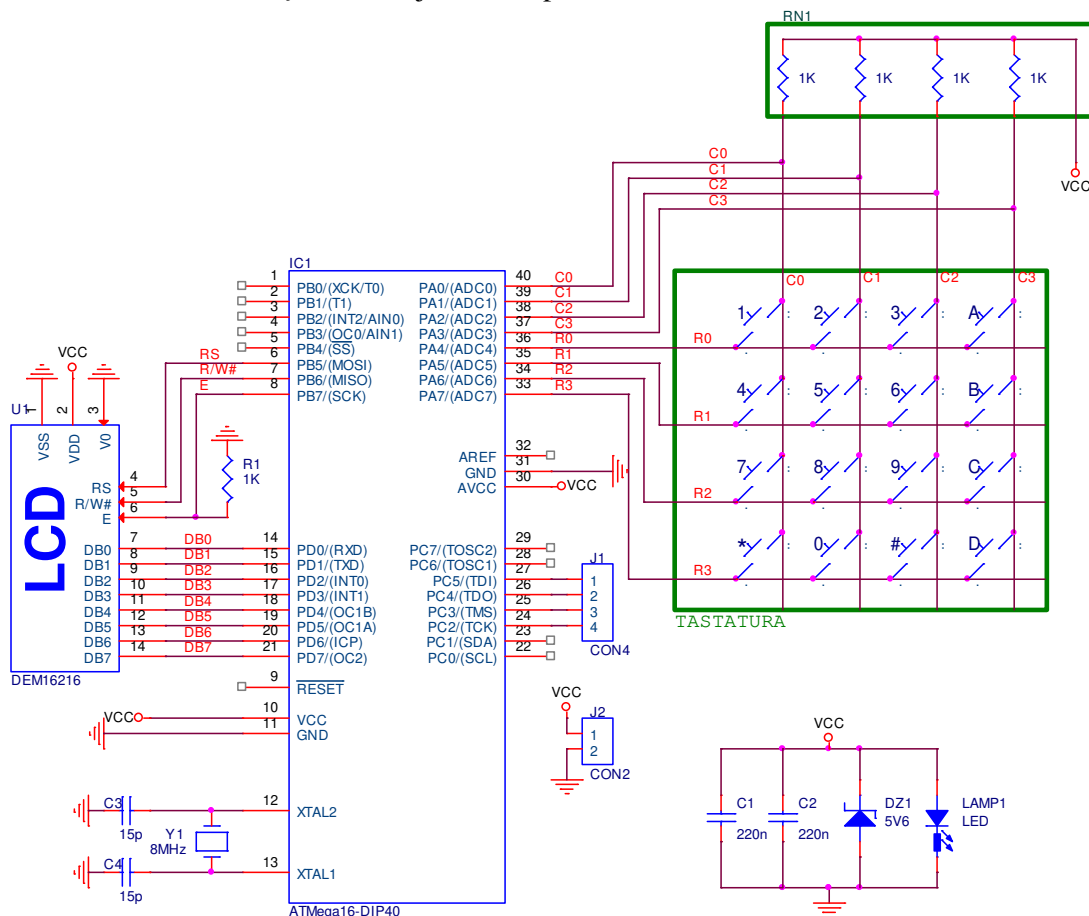


figura 3

Verificați cu atenție conexiunile de alimentare. **Dacă există o greșeală în zona alimentării, sunt mari șanse ca ceva să se ardă!** Rareori greșeli de altă natură duc la distrugerea componentelor.

Alimentați montajul **SINGURI! Dacă se arde ceva, veți plăti componentele distruse!**

Pasul 3: Crearea proiectului

Se va crea un proiect nou cu numele **ceas** care va conține aceleași surse ca proiectul „kbd” (fără LED sau shift) din laboratorul 7. Pentru crearea acestui proiect procedați după cum urmează:

- Se va crea (obligatoriu cu NEXT)** proiectul cu numele **ceas**. La crearea proiectului **obligatoriu bifați** căsuța **Create initial file**.
- Închideți** AVR Studio.
- Copiați** **kbd.c**, **IOfn.c** și **defs.h** din folderul **kbd** în folderul proiectului **ceas**.
- În folderul **ceas** **ștergeți fișierul** **ceas.c** (care este gol).
- În folderul **ceas** **redenumiți** **kbd.c** ca **ceas.c**.
- Deschideți (obligatoriu cu NEXT)** proiectul **ceas**.
- Faceți click dreapta pe **Source Files** și din meniul contextual ce va apare selectați **Add existing Source File(s): Adăugați** la proiect fișierul **IOfn.c** din folderul **ceas**.
- Faceți click dreapta pe **Header Files** și din meniul contextual ce va apare selectați **Add existing Header File(s): Adăugați** la proiect fișierul **defs.h** din folderul **ceas**.

Pasul 4: Setarea modului și a valorii p

În secțiunea de inițializare a programului trebuie să setăm timerul 2 astfel încât să numere modulo N . Modul în care orice numărător ATmega numără modulo N este modul CTC. Valoarea N se înscrie în registrul OCR. De asemenea trebuie programat factorul de scalare p . N și p au fost calculate anterior.

Modul de funcționare al numărătorului 2 și factorul de scalare p se setează prin intermediul registrului TCCR2. Detalii despre acest registru cât și despre celelalte registre care controlează numărătorul 0 se găsesc în fișierul **ATmega16A Data Sheet** de la pagina 130 și în prelegerea 5.

Mai întâi vom seta biții WGM21 și WGM20 pentru ca numărătorul 2 să fie în mod CTC. Setările se fac conform tabelului 17-2, reprodus în continuare:

Bit	7	6	5	4	3	2	1	0	
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
Read/Write	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	

Table 17-2. Waveform Generation Mode Bit Description⁽¹⁾

Mode	WGM21 (CTC2)	WGM20 (PWM2)	Timer/Counter Mode of Operation	TOP	Update of OCR2	TOV2 Flag Set on
0	0	0	Normal	0xFF	Immediate	MAX
1	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	1	0	CTC	OCR2	Immediate	MAX
3	1	1	Fast PWM	0xFF	BOTTOM	MAX

Atenție: biții WGM21 și WGM20 apar în TCCR2 în ordine inversă față de tabelul 17-2.

În continuare se programează valoarea lui p calculată anterior. Setările se fac conform tabelului 17-6, reprodus în continuare:

7	6	5	4	3	2	1	0	
FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
0	0	0	0	0	0	0	0	

Table 17-6. Clock Select Bit Description

CS22	CS21	CS20	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$\text{clk}_{T2S}/(\text{No prescaling})$
0	1	0	$\text{clk}_{T2S}/8$ (From prescaler)
0	1	1	$\text{clk}_{T2S}/32$ (From prescaler)
1	0	0	$\text{clk}_{T2S}/64$ (From prescaler)
1	0	1	$\text{clk}_{T2S}/128$ (From prescaler)
1	1	0	$\text{clk}_{T2S}/256$ (From prescaler)
1	1	1	$\text{clk}_{T2S}/1024$ (From prescaler)

Biții FOC2, COM21 și COM20 deocamdată nu se programează și vor rămâne la valorile default (zero).

Orice numărator ATmega provoacă setarea indicatorul TF. **Indicatorul TF nu există ca atare**, el este un nume generic. Numărătorul 2 are asociați doi indicatori: TOV2 și OCF2. TOV2 - Timer Overflow Flag – este setat imediat după ce numărătorul a trecut din starea 0xFF în starea 0x00 iar OCF2 – este setat imediat după ce starea numărătorului a fost egală cu conținutul registrului OCR2. Aceștia doi indicatori sunt disponibili prin intermediul registrului TIFR, al cărui conținut este reprodus mai jos.

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
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	

Mai multe informații despre acest registru se găsesc în datele de catalog, la pagina 85.

Care dintre cei doi indicator indică ciclarea Timer/counterului 2 configurat ca numărator modulo N?

Pasul 5: Scrierea programului principal

Timerul 2 va fi folosit în mod **asemănător** cu timerul 0 din implementarea ceasului digital din prelegerea 5, capitolul 3.4 „Ceas Digital cu timerul 0”. Pentru implementare aveți de nevoie de:

- p , N , k determinat anterior.
- Să știți care din indicatorii TOV2 sau OCF2 este indicatorul TF.

La ceas.c (o copie a lui kbd.c) **se va adăuga** codul pentru ceas.

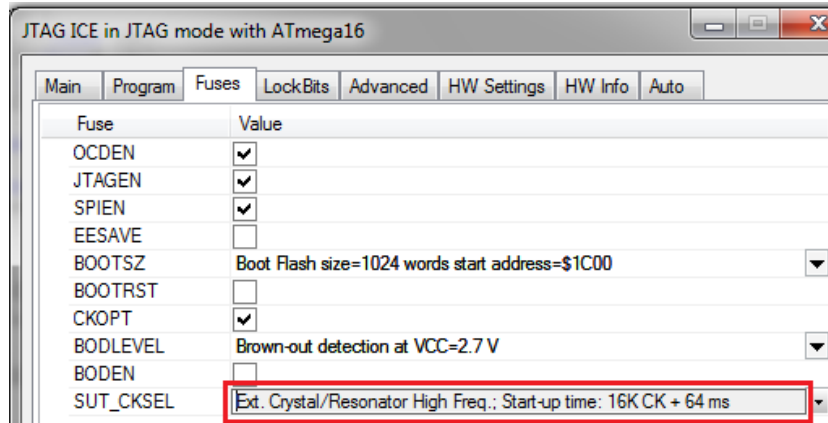
Programul principal va avea structura prezentată în curs și detaliată în continuare:

```
//include și prototipuri
int main() {
    //declarații variabile pentru tastatură - deja există

    unsigned int cycles=0;
    // alte declarații variabile pentru ceas
```


Pasul 6: Setarea tipul de oscilator și testare

Conectați-vă cu **Con** și alegeți cristalul extern ca tip al ceasului procesor. Pentru aceasta faceți selecția marcată cu roșu în figura următoare și apoi **programați** această setare:



Testați conform următoarei proceduri:

- Tastatura trebuie să funcționeze.
- Setati timpul la 23:59:50 și rulați. Verificați că tranziția 23:59:59 → 0:00:00 se face corect.
- Faceți ca timpul afișat pe LCD să fie egal cu ceasul din Windows. Mai întâi setați timpul inițial în avans cu aproximativ un min față de timpul calculatorului. Timpul inițial trebuie să aibă secunde egale cu zero. Dați comanda Run când timpul setat în program și ceasul Windows devin egali. Se admite o diferență de o secundă. De exemplu dacă ceasul Windows este 16:43:13 setați timpul inițial la 16: 44:00. Imediat după ce ați dat comanda **Run**, **chemați profesorul**.
- Lăsați să ruleze cel puțin 5 minute.
- Dacă după cele 5 minute nu există diferențe între timpul afișat pe LCD și timpul PC, **chemați profesorul pentru validare. Dacă ați ajuns aici aveți nota 5.**

Partea opțională

Pasul 7: Clipire separator

Creați un nou proiect. Acest proiect este o copie a proiectului ceas din acest laborator.

Setați controlerul de LCD să nu mai afișeze cursorul.

Faceți ca separatorii **,:** să clipească ½ sec stins și ½ sec aprins.

Pentru a verifica dacă timpul este calculat și afișat corect folosiți procedura descrisă anterior la obținerea notei 5.

Dacă funcționează, chemați profesorul pentru validare.

Dacă ați ajuns aici aveți între 1 și 2 puncte (max 7).

Pasul 8: Ajustare

Se dorește ajustarea timpului prin intermediul tastelor **,A'** și **,B'**. Rolul acestor taste este următorul:

1. **La prima apăsarea lui ,A'** măsurarea timpului se oprește și secunde devin zero. Dacă în această fază se apasă **,B'**, la fiecare apăsare minutele se vor incrementa circular, adică după 59 urmează 00. Tranziția 59 → 00 nu va incrementa orele.

2. **La a doua apăsarea lui ,A’** începe setarea orelor. Măsurarea timpului este în continuare oprită. Dacă în această fază se apasă ,B’, la fiecare apăsare orele se vor incrementa circular, adică după 23 urmează 00.
3. **La a treia apăsarea lui ,A’** se reia măsurarea timpului. Următoarea apăsare a lui ,A’ va fi tratată ca prima apăsare.

Funcționalitatea implementată anterior trebuie să se păstreze. Se recomandă implementarea acestei funcțiuni cu SFSM, conform modelului din laboratorul 8.

Testați dacă timpul este măsurat corect conform procedurii pentru nota 5.

Dacă setarea + măsurarea timpului funcționează, chemați profesorul pentru validare.

Dacă ați ajuns aici aveți între 1 și 3 puncte.

Pasul 9: Deconectare

La terminare executați următoarele operații:

1. Se oprește execuția programului cu **Break**,
2. Se oprește depanatorul cu **Stop**
3. Se apasă **Con** și apoi se șterge memoria de program cu **Erase**
4. Se închide AVR Studio.
5. Se oprește alimentarea JTAG ICE.
6. Se oprește sursa Hameg.
7. Se desface montajul, numai componentele adăugate în acest laborator.