3.Bucle de întârziere şi citirea întrerupătoarelor

Cuprins

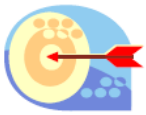
Cuprins Laborator 3

Cunoştinţe preliminare

- Înţelegerea conceptelor programării în limbaj de asamblare;
- Cunoaşterea programului SID;
- Capacitatea de a scrie şi de a verifica un program simplu.

Introducere

În Laboratorul 3 studenţii rulează programele la viteza procesorului, prin introducerea unor bucle de întârziere. În partea a doua a laboratorului se citesc întrerupătoarele şi se identifică linia pe care este cuplat fiecare întrerupător.



Obiective

După parcurgerea acestui laborator studenţii vor şti să scrie un program care apelează bucle de întârziere şi vor înţelege importanţa întârzierilor controlate în comanda unor procese. În partea a doua a laboratorului studenţii învaţă să citească un port de intrare şi să identifice poziţia unei linii de intrare.

Durata medie de studiu individual

Durata medie de studiu individual este de 2 ore.

3.1.Bucle de întârziere

Se reia discuția din ședința de laborator precedentă legată de faptul că rularea programului care comandă o secvență dinamică de aprindere a LED-urilor la viteza procesorului nu produce o clipire vizibilă. Explicația este evidentă, viteza cu care procesorul execută trimiterile la port este foarte mare și ochiul nu percepe clipirea.

Pentru a faca ca secvența dinamică de aprindere să fie vizibilă, după fiecare aprindere trebuie introdusă o buclă de întârziere sau o apelarea unei rutine de întârziere din sistemul de operare. La laborator se utilizează metoda introducerii unei bucle de întârziere. Au fost folosiți regiştrii DX pentru adresarea portului şi AX pentru date, deci pentru bucla de întârziere rămân disponibili regiştrii BX şi CX. În principiu se încarcă un registru şi se decrementează până ajunge la zero. Pentru a nu scrie secvența de întârziere de mai multe ori se poate realiza o subrutină de întârziere. În programul principal se introduce instrucțiunea

CALL <*adresaÎntârziere*>

şi la *adresaîntârziere* se scrie secvența de program care se termină cu instrucțiunea RET.

După ce s-a făcut **CALL** la o rutină de întârziere programul nu se mai poate rula pas cu pas (de fapt se poate, dar ar dura nepractic de mult). Se poate scrie programul fără instrucțiunile **CALL** care se înlocuiesc cu instrucțiuni **NOP** (câte 3 instrucțiuni succesive pentru fiecare instrucțiune **CALL**). Se rulează programul pas cu pas şi se verifică funcționalitatea. Dacă totul este în ordine se înlocuieşte prima instrucțiune **NOP** cu **CALL**. Se notează adresele la care s-a pus primul **NOP** şi scrieți instrucțiunea **CALL** cu **A<adresa>**. La sfârşit se mai verifică odată programul cu **L<adresa>**.

Programul principal va conține instrucțiunile:

```
MOV DX, 0378 ; DX <= adresa portului de ieşire
MOV AL, FF   ; AL <= cuvântul de trimis
OUT DX, AL   ; scrie AL la portul DX
CALL <adresaÎntârziere>  ; apel procedură de întârziere
```

Subrutina de întârziere, la adresa <adresaÎntârziere>, va conține instrucțiunile:

```
<adresaÎntârziere>:  MOV BX,FFFF   ; BX <= valoarea întârzierii
bucla:               DEC BX
                     CMP BX, 0000
                     JNZ bucla
```

Dacă este nevoie de o întârziere mai mare trebuie se pot folosi două registre şi două bucle.

O confuzie frecventă este făcută de studenți între bucla după buclă şi bucla în buclă. Diferența între cele două variante este prezentată în figura 3.1.

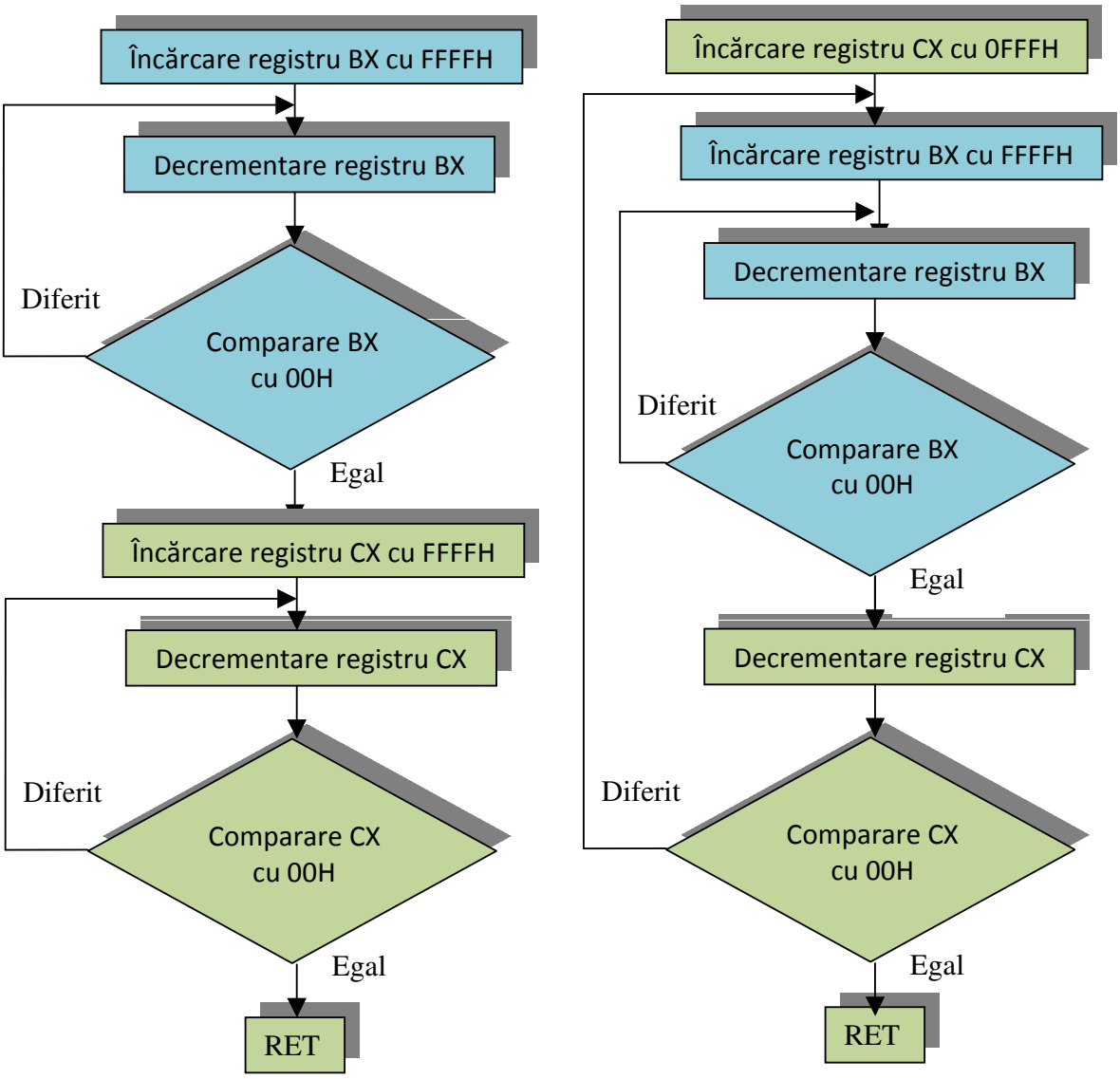


Figura 3.1. Schema logică a întârzierii buclă după buclă (stânga) şi buclă în buclă (dreapta)

Comparând structura celor două programe se vede că bucla după buclă asigură o dublare a întârzierii, ceea ce este insuficient. Bucla în buclă asigură o rulare a buclei din interior de FFFFH ori, ceea ce depăşeşte întârzierea necesară. Întârzierea optimă depinde de viteza calculatorului şi poate fi stabilită prin încercări de fiecare student.

3.2.Citirea întrerupătoarelor

Citirea întrerupătoarelor se poate face cu un program foarte simplu rulat pas cu pas.

```
Start:  MOV DX,0379
        IN AL,DX
        JMP start
```

Programul este rulat după fiecare apăsare a unui întrerupător şi se observă valoarea hexazecimală citită în AL. Prin conversia în binar se poate afla linia pe care este cuplat fiecare întrerupător.

Rularea pas cu pas a programului din figura 3.2. arată valorile citite în AL. Dacă nu este apăsat niciun întrerupător se citeşte 78H, la apăsarea primului întrerupător se citeşte F8H, la al doilea 38H, la al treilea 58H etc.

Următoarea secvență simplă care poate fi realizată de studenți afişează pe LED-uri informația citită de la întrerupătoare.

```
Start:  MOV DX,0379
        IN AL,DX
        MOV DX,0378
        OUT DX,AL
        JMP start
```

Se constată că apăsarea unor întrerupătoare înseamnă trecerea de la nivel logic zero la unu, dar există şi excepții, apăsarea însemnând trecerea de la nivel unu la zero.

```
#a0100
0B9A:0100 MOV DX,0379
0B9A:0103 IN AL,DX
0B9A:0104 JMP 0100
0B9A:0107 .
#XIP
IP 0100
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 0000 0000 0000 0000 0000 0000 0000 0000 0100 MOV    DX,0379
*0B9A:0103
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 0000 0000 0000 0379 0000 0000 0000 0000 0103 IN     AL,DX
*0B9A:0104
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 0078 0000 0000 0379 0000 0000 0000 0000 0104 JMP    0100
*0B9A:0100
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 0078 0000 0000 0379 0000 0000 0000 0000 0100 MOV    DX,0379
*0B9A:0103
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 0078 0000 0000 0379 0000 0000 0000 0000 0103 IN     AL,DX
*0B9A:0104
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 00F8 0000 0000 0379 0000 0000 0000 0000 0104 JMP    0100
*0B9A:0100
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 00F8 0000 0000 0379 0000 0000 0000 0000 0100 MOV    DX,0379
*0B9A:0103
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 00F8 0000 0000 0379 0000 0000 0000 0000 0103 IN     AL,DX
*0B9A:0104
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 0038 0000 0000 0379 0000 0000 0000 0000 0104 JMP    0100
*0B9A:0100
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 0038 0000 0000 0379 0000 0000 0000 0000 0100 MOV    DX,0379
*0B9A:0103
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 0038 0000 0000 0379 0000 0000 0000 0000 0103 IN     AL,DX
*0B9A:0104
#T
            AX   BX   CX   DX   SP   BP   SI   DI   IP
--I------ 0058 0000 0000 0379 0000 0000 0000 0000 0104 JMP    0100
*0B9A:0100
#
```

Figura 3.2. Rularea programului de citire a întrerupătoarelor

Rezumat

În această lucrare studenții care nu au ştiut înțeleg de ce nu s-a văzut clipirea LED-urilor la viteza procesorului şi învață să facă un program de întârziere. Se discută o greşeală frecventă pe care o fac unii studenți care la scrierea buclei de întârziere. În a doua parte a şedinței de laborator se scrie un program de citire a întrerupătoarelor şi se identifică apoi linia pe care este cuplat fiecare întrerupător.

Bibliografie

1. Petre Ogrutan, Carmen Gerigan, *Memorii, interfete si periferice*, Indrumar de laborator, Reprografia Universitatii 1998, online la: http://vega.unitbv.ro/~ogrutan/lab/index.html
2. P. A. Carter, *PC AssemblyLanguage, 2003* http://pdos.csail.mit.edu/6.828/2012/readings/pcasm-book.pdf