ATMEL: MICROCONTROLERE AVR RISC

Marin Pripa. Alexandru Gaidei.

Universitatea Tehnică a Moldovei

Abstract: *Microcontrolerul este o configuraţie minimală de sistem de calcul, capabil să execute la o viteză foarte mare instrucţiunile unui program care este o secvenţă logică de operaţii. Microcontrolerul are integrat pe acelaşi chip: oscilatorul, memoria (RAM, ROM, EEPROM), numărătoare, blocuri analogice ş.a.*

Cuvinte-cheie: *microcontroler, avr, risc, programare, c, asamblare, programatoare.*

1. Introducere

Limbajele la nivel înalt devin metoda de programare standard pentru microcontrolere. Limbajul "C" este cel mai utilizat limbaj de nivel înalt folosit in microcontrolere. Un producător important de microcontrolere este Corporaţia ATMEL, care a adaptat AVR-ul la limbajul "C". În figura 1 sunt prezentate microcontrolere de tip AVR.

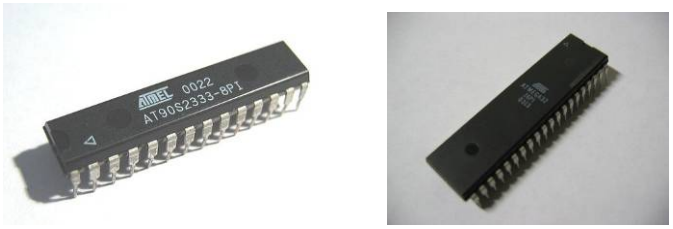

Fig. 1 AVR AT90S şi AVR ATMEGA32

Microcontrolere AVR se împart în trei subfamilii: ATtiny - AVR în 8 pini, AT90Sxxxx - AVR de uz general şi ATmega - AVR de înaltă performanţă. AVR-urile sunt dotate cu memorie program şi de date reprogramabile. Toate au consum redus de putere, executând totuşi 1 MIPS (Mega Instructions Per Second). Frecvenţele maxime a generatorului de impuls sunt de 1 ÷ 4MHz la circuitele ce se alimentează la 2,7V (LV) şi 8 ÷ 12 MHz la celelalte.

2. Arhitectură RISC superioară

Pentru a executa o anumite funcţii se utilizează arhitecturi concrete. Arhitecturile RISC sunt frecvent alese acolo unde este necesară o viteză mare. Arhitectura AVR este concepută ca o arhitectură RISC, cu un număr mare de instrucţiuni pentru a reduce dimensiunea de cod păstrând viteza mare de execuţie. O îmbunătăţire esenţială este optimizarea limbajului de asamblare şi a arhitecturii pentru limbajul "C".

Multe arhitecturi de microcontrolere au un număr scăzut de registre generale sau de registre de serviciu (acumulatoare) - de obicei 1 ÷ 8 registre. AVR-ul are 32 de registre de uz general, pe care compilatorul "C" le foloseşte la maximum la discifrarea programului. Microcontrolerul AVR ca şi alte microcontrolere are o construcţie structurală similară ca a unui calculator modern (Fig.2).

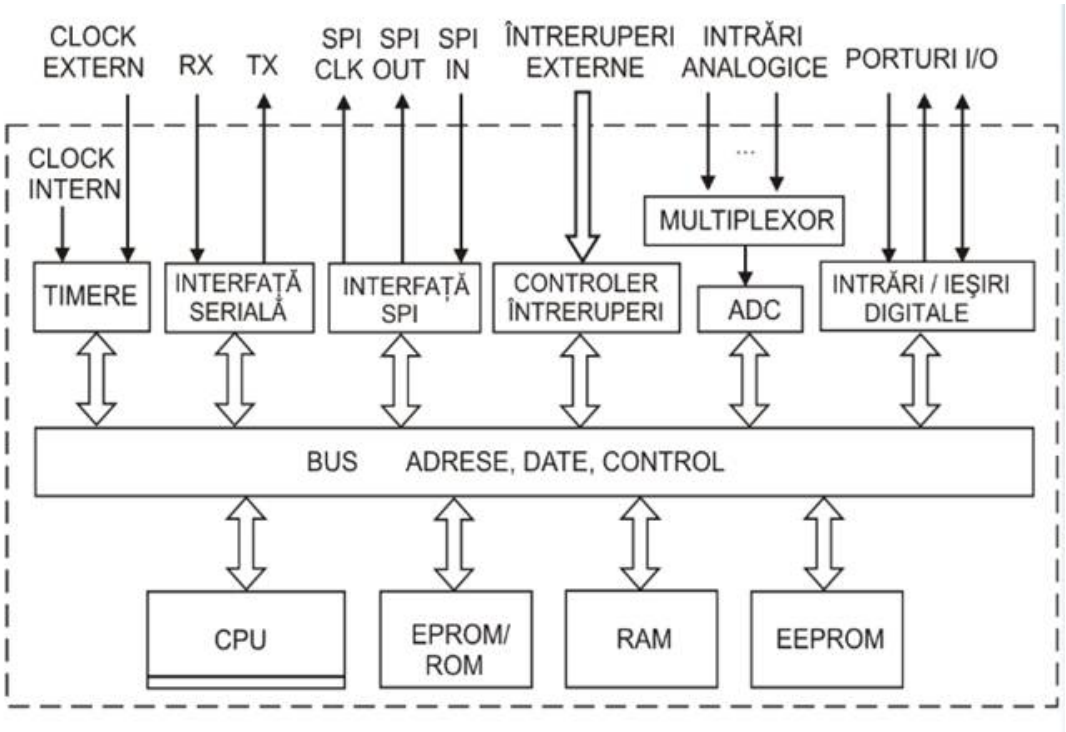

Fig. 2Structura microcontrolerului AVR

3. Adaptat la limbajul "C"

Pentru a îndeplini funcţii concrete, microcontrolerul trebuie să fie programat. Programul software conferă microcontrolerului, abilitatea de a realiza funcţii diferite cu aceeaşi configuraţie hardware . Scrierea programului se realizează de obicei intr-un editor ce permite salvarea liniilor de comandă introduse.

Există mai multe opţiuni pentru scrierea programului de control al aplicatiei şi anume:

- cod masină (cod hexadecimal)
- limbaj de asamblare,
- limbaj de nivel inalt (C, Basic,Pascal, etc)

Limbajul "**C**" este cel mai folosit HLL (**H**igh **L**evel **L**anguage) pentru microcontrolere. Deoarece majoritatea arhitecturilor de microcontrolere sunt proiectate pentru a fi programate în limbaj de asamblare, susţinerea pentru instrucţiunile de "**C**" este deficitară. ATMEL a dorit dezvoltarea unei arhitecturi care să fie eficientă atât pentru limbajul "**C**", cât şi pentru asamblare. Cooptarea în echipă a mai multor experţi de "**C**" a făcut posibilă realizarea unui cod foarte eficient pentru microcontrolerele de 8 biţi cu suport de 16 biţi. În programarea în "**C**", este necesar de a folosi variabile definite în proceduri locale.Variabilele locale vor aloca memorie RAM doar atunci când se execută instrucţiuni. Pentru a utiliza variabilele locale AVR-ul are 32 registre de uz general, toate legate la ALU (**A**ritmetic **L**ogic **U**nit).

```
Function: A = ((A .and. 84h) + (B.eor.C).or.80h

AVR code          CISC code
EOR  B,C          MOV  ACC,C
ANDI A,#84h       EOR  ACC,B
ADD  A,B          MOV  TMP,ACC
ORI  B,#80h       MOV  ACC,A
                  AND  ACC,#84h
                  ADD  ACC,TMP
                  OR   ACC,#80h
                  MOV  A,ACC

8 bytes           12-16 bytes
4 clocks          48-96 clocks
```

Fig. 3Instrucțiuni in codul AVR RISC şi CISC.

Observăm că în codul AVR RISC sunt 4 instrucțiuni, pe când în CISC sunt de 2 ori mai multe.

Când se utilizează limbajul "**C**", numere mari din tipurile "integer" (16 biți), "long" (32 biți), "float" (32 biți) sunt frecvent folosite. De regulă procesarea cu asemenea numere generează dimensiuni de cod foarte mari, dar din moment ce AVR-ul este proiectat pentru a-i face față, codul AVR generat este foarte eficient.

```
Example 1:
void routine(void)
      {
      long n1, n2;
      int n3;
      ...
      if (n1 != n2) n3 +=5;
      ...
      }
```

```
...
CP    R0,R4       ; n1-n2 (byte 0)
CPC   R1,R5       ; n1-n2-C (byte 1)
CPC   R2,R6       ; n1-n2-C (byte 2)
CPC   R3,R7       ; n1-n2-C (byte 3)
BREQ  EQUAL       ; Branch if equal
SUBI  R16,LOW(0xfffb)  ;n3+5 low byte
SBCI  R17,HIGH(0xfffb);n3+C high byte
EQUAL:
...
```

Fig. 4

O parte a programului scrisă in „C" şi aceeaşi parte tradusă în codul de asamblare.

4. Programatoare AVR

Pentru a transfera codul hexadecimal rezultat în urma compilării, în memoria ROM (memoria program) a microcontrolerului este nevoie de un programator. Programatorul este compus dintr-un modul electronic care asigură interfațarea între aplicația ce conține microcontrolerul şi calculator (PC), şi un program software ce rulează pe PC.

În fig.5 sunt prezentate programatoare simple fabricate în condiţii casnice.

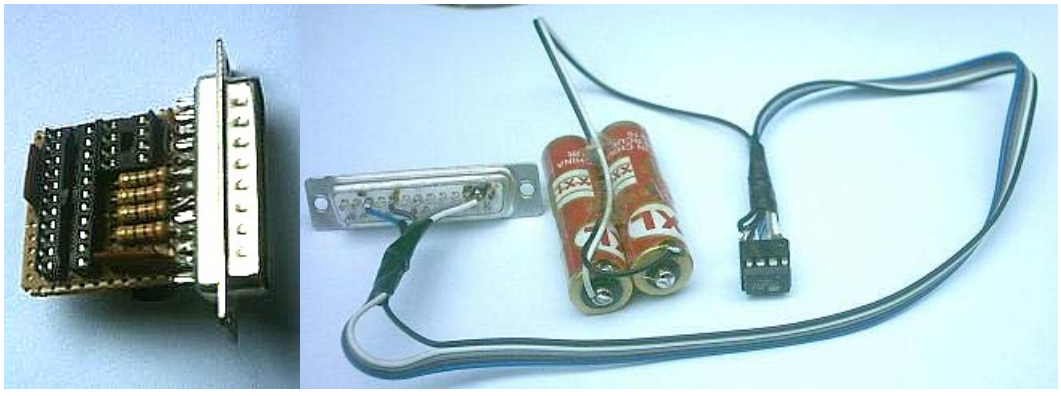

Fig. 5Programatoare simple.

Mai performante sunt programatoarele prezentate în fig.6:

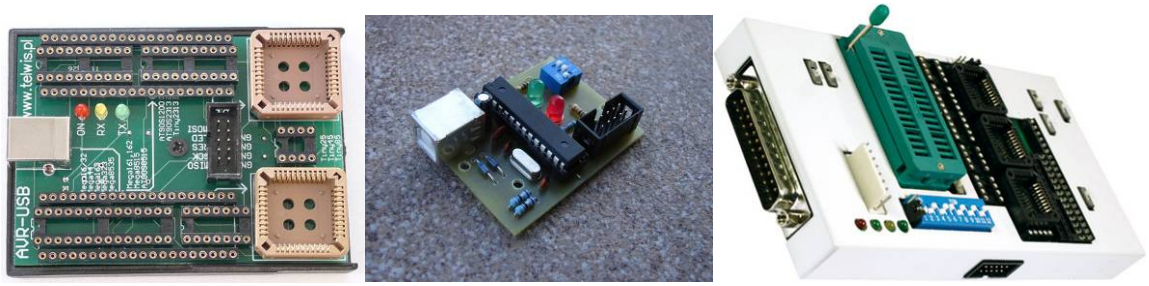

Fig. 6Programatore complexe.

Conectarea la PC se efectuiază prin intermediul cablului ISP la portul Paralel, Serial sau în ultimul timp se utilizează cablu de conecatre ISP – USB. În fig.7este prezentat cablul Paralel – ISP.

Fig. 7 Cablu Paralel – ISP.

Alimentarea microcontrolerului în timpul programarii se poate efectua atît de la o sursa externă, cît şi direct de la portul PC, în dependenţă de tipul Programatorului utilizat.

5. Domeniul de utilizare

Printre multele domenii unde utilizarea microcontrolerelor este practic un standard industrial se pot menţiona: în industria de automobile (controlul aprinderii/motorului, climatizare, diagnoză, sisteme de alarmă, etc.), în electronică de consum (sisteme audio, televizoare, camere video şi videocasetofoane, telefonie mobilă, GPS-uri, jocuri electronice, etc.), în aparatura electrocasnică, în controlul mediului şi climatizare, în industria aerospaţială, în mijloacele moderne de măsurare - instrumentaţie (aparate de măsură, senzori şi traductoare inteligente), la realizarea de periferice pentru calculatoare, în medicină.

Ca un exemplu din industria de automobile (automotive industry), unde numai la nivelul anului 1999, un BMW seria 7 utiliza 65 de microcontrolere, iar un Mercedes din clasa S utiliza 63 de microcontrolere. La momentul actual, practic este foarte greu de găsit un domeniu de aplicaţii în care să nu se utilizeze microcontrolerele.