

MICROCONTROLERE – Lucrarea de laborator 1

Scopul lucrării:

- descrierea modului de lucru cu compilatorul Code Vision AVR si cu programul de depanare Astudio, specifice microcontrolerelor Atmel
- descrierea arhitecturii microcontrolerelor AVR – Atmel
- realizarea unui program, in limbaj C, cu programarea porturilor de intrare – iesire si a timerului din structura microcontrolerului Atmel AT90S8515.
- prezentarea placii de evaluare cu microcontroler AVR-Atmel, STK 500
- modul de executie cu simulare (depanare) si pe placa reala

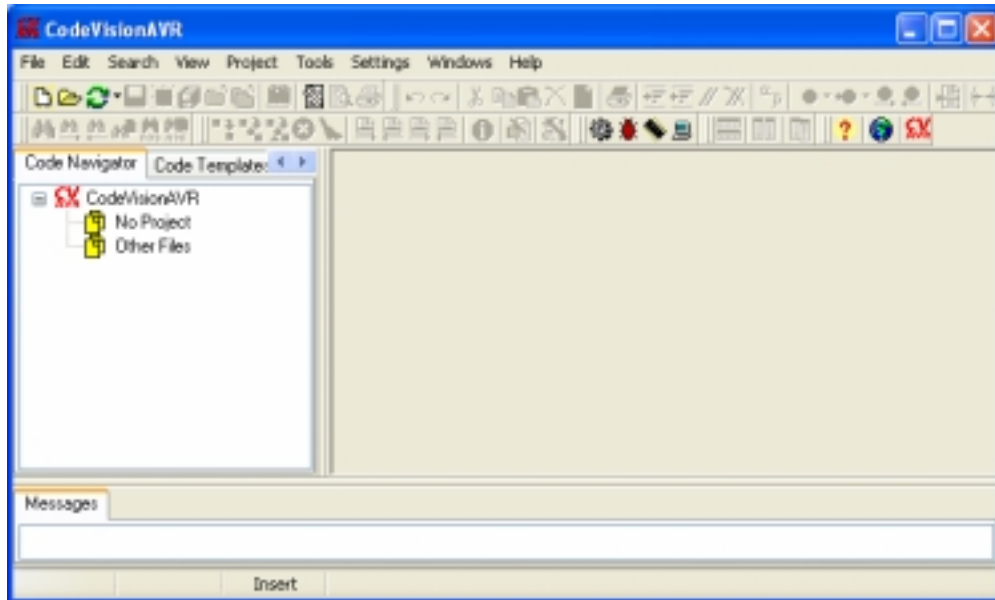
Desfasurarea lucrării

1. Se va studia arhitectura microcontrolerului AVR – Atmel AT90S8515
2. Se va studia modul de operare al programului CVAVR
3. Se va studia arhitectura placii de evaluare STK 500
4. Se va contrui un proiect nou care sa realizeze programarea porturilor in conformitate cu structura STK 500 si a timerului care va genera intreruperi periodice cu perioada de 20 ms.
5. Se vor realiza urmatoarele programe (a-d):
 - a) se va aprinde LED-ul corespunzator butonului SW apasat
 - b) la apasarea butonului SW0 se vor aprinde LED-urile pare, iar la apasarea butonului SW7 se vor aprinde LED-urile impare
 - c) la apasarea butonului SW1, LED-ul LED0 se va aprinde/stinge cu o cadenta de 1 sec.
 - d) se vor aprinde LED-urile LED0-7 pe rind cite 0.5 secunde incepind cu LED-ul 0, in mod cyclic

Toate programele vor fi simulate cu Astudio si rulate in timp real pe STK 500.

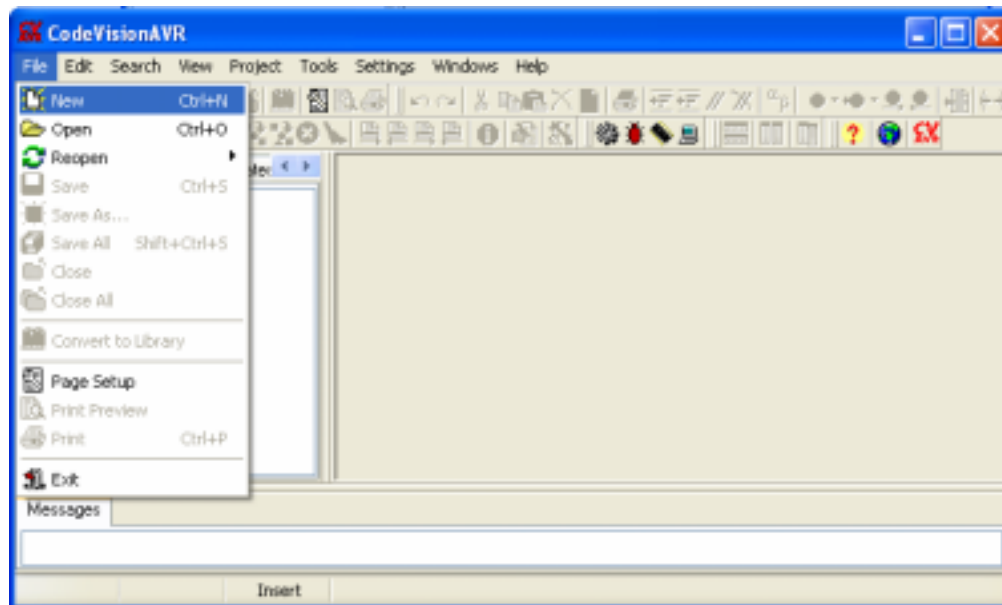
Descrierea modului de lucru cu compilatorul CVAVR si a depanatorului ASTUDIO

Se lanseaza in executie programul CVAVR (Code Vision AVR).

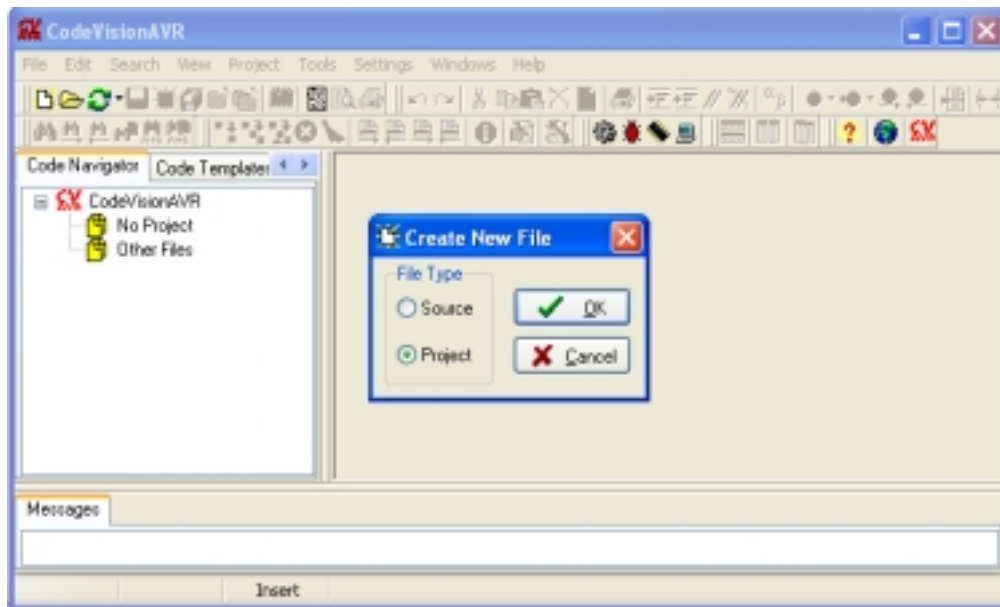


Crearea unui proiect nou

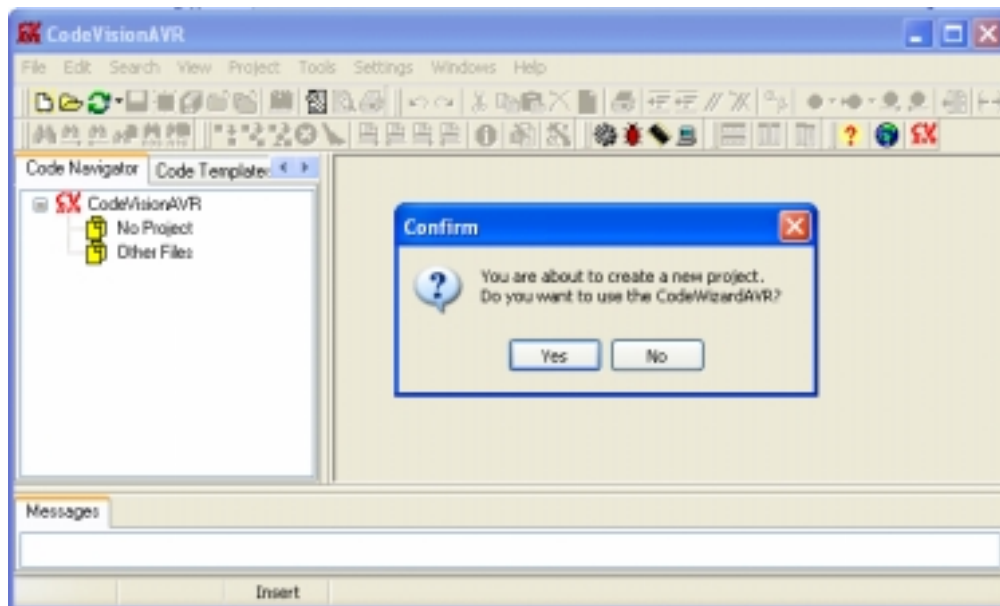
Se selecteaza din bara de meniu – File->New



In fereastra “Create new file” se selecteaza ‘Project’



Se apasa “OK” :

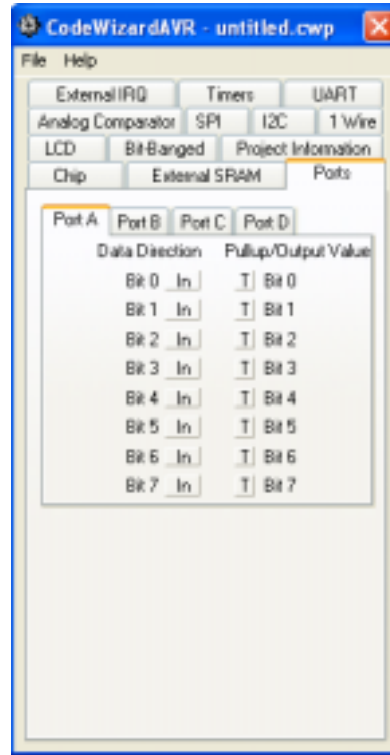
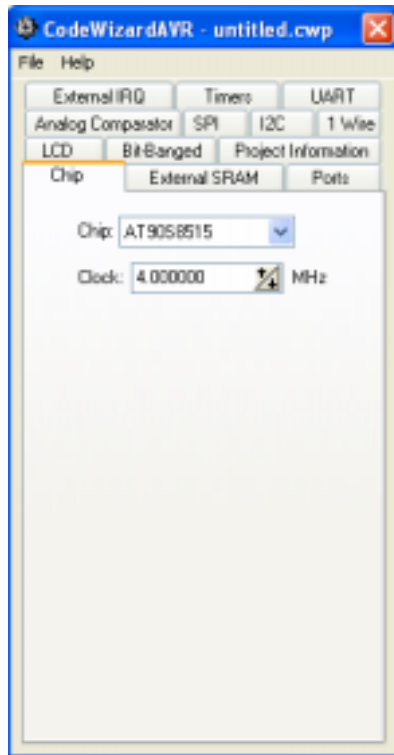


Se opteaza pentru utilizarea “ Code Wizard AVR” care va asista crearea proiectului. Programul “Code Wizard AVR” va genera un program sursa care va contine toate programarile resurselor dorite. Programarea acestor resurse se va realize in mod graphic.

Apare o fereastra de dialog pentru selectarea resurselor din microcontroler:

Se vor programa doua tipuri de resurse: porturile de intrare – iesire (PORTD – intrare, PORTB – iesire) si timer-ul TIMER0.

Se va selecta “Ports” :



Se selecteaza “Port B” ca iesire cu totii bitii in “1” si “Port D” ca intrare cu totii bitii “Pull up” (P):

Celelalte optiuni snt : “0” – pentru iesire si starea de inalta impedanta (“three state” – T) pentru intrari. Fiecare bit al unui port poate fi programat individual ca intrare sau ca iesire.

Un port I/O este constituit din 3 registre : PORTx – contine iesirile (daca portul este de iesire), PINx – contine intrarile (daca portul e de intrare) si DDRx – Direction Data Register – care indica pentru fiecare bit daca acesta e intrare (“0”) sau iesire (“1”). Sufixul “x” indica portul din microcontroler – x = { A,B, C, D, E ... }.

In programele scrise in limbajul C se pot accesa porturile de intrare – iesire astfel:

- porturile de intrare :

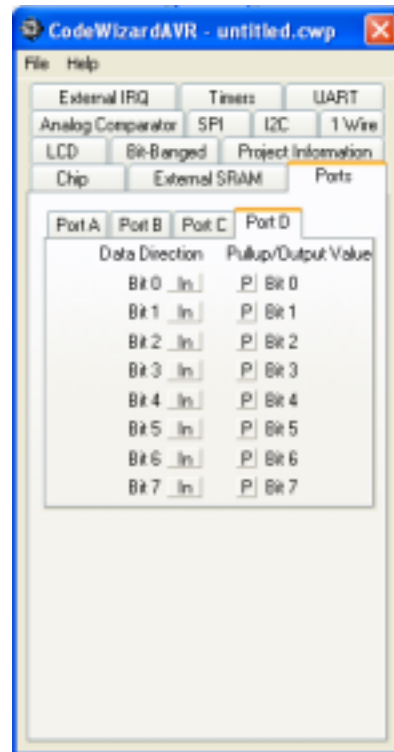
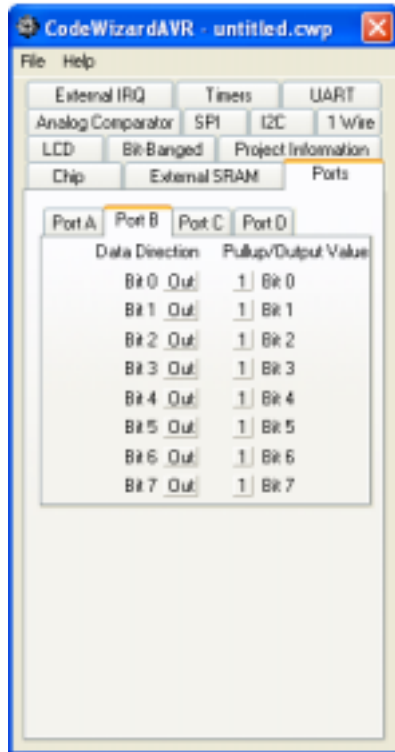
a= PINx; // se citeste in variabila a valoarea portului de intrare x

a= PINx.y; // se citeste in variabila a valoarea bitului y al portului de intrare x

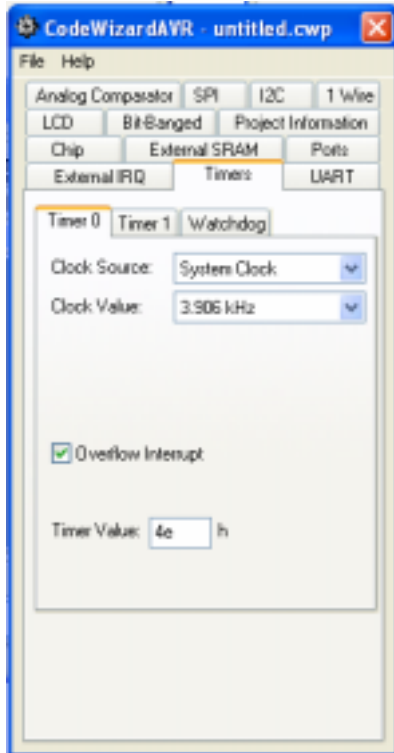
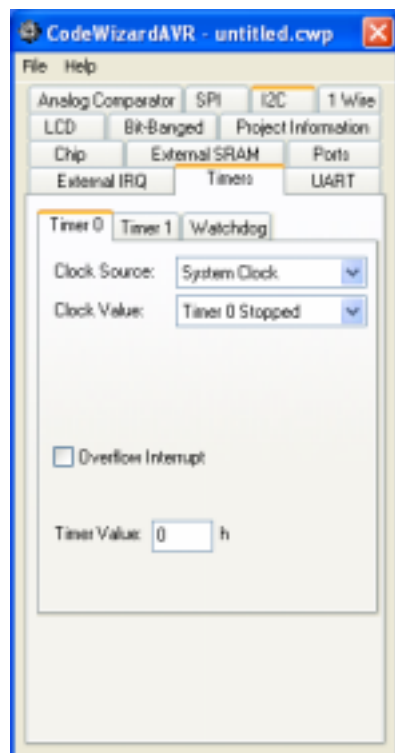
- porturile de iesire :

PORTx=a; // se scrie in portul de iesire x valoarea variabilei a

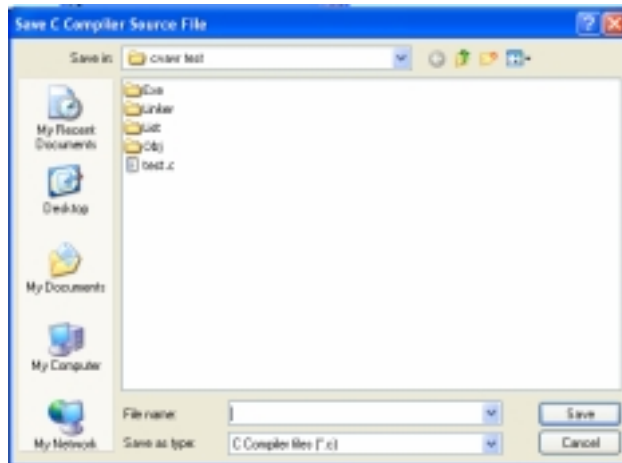
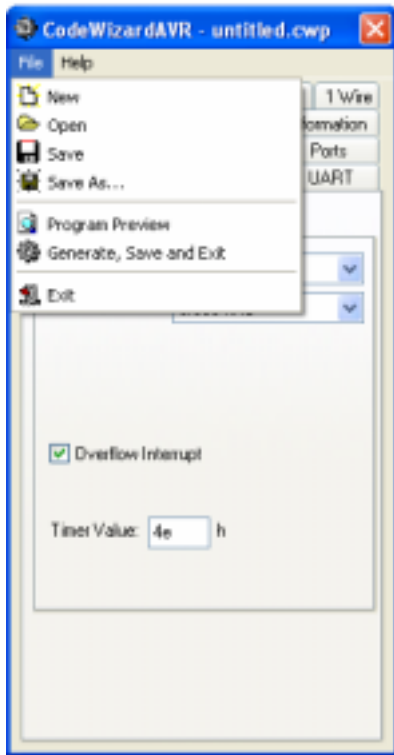
PORTx.y=a; // se scrie in portul de iesire x bitul y valoarea variabilei a



Se selecteaza "Timers" -> Timer0; Clock source - "System Clock", "Clock Value" 3.906 kHz, se bifeaza "Overflow Interrupt" si se inscrie in "Timer Value" valoarea "4e". Timerul 0 este programat sa genereze o intrerupere la fiecare 20ms.



Dupa terminarea setarilor dorite pentru toate resursele se selecteaza din meniu “File” si “Generate, Save and Exit”. Se salveaza proiectul.



Programul generat automat este urmatorul:

```

Automatic Program Generator
© Copyright 1998-2008 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date   : 8/1/2008
Author : Freeware, for evaluation and non-commercial use only
Company :
Comments:

Chip type       : AT90S8515
Clock frequency : 4.000000 MHz
Memory model    : Small
External RAM size : 0
Data Stack size : 128
*****/

#include <90s8515.h>

char a;
    
```

```

// Timer 0 overflow interrupt service routine
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
// Reinitialize Timer 0 value
TCNT0=0x4E;
// Place your code here

a=PORTD;
PORTB=a;
}

// Declare your global variables here

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=1 State6=1 State5=1 State4=1 State3=1 State2=1 State1=1 State0=1
PORTB=0xFF;
DDRB=0xFF;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=P State6=P State5=P State4=P State3=P State2=P State1=P State0=P
PORTD=0xFF;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: 3.906 kHz
TCCR0=0x05;
TCNT0=0x4E;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge

```

```

// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
GIMSK=0x00;
MCUCR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x02;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;

// Global enable interrupts
#asm("sei")

while (1)
{
// Place your code here

};
}

```

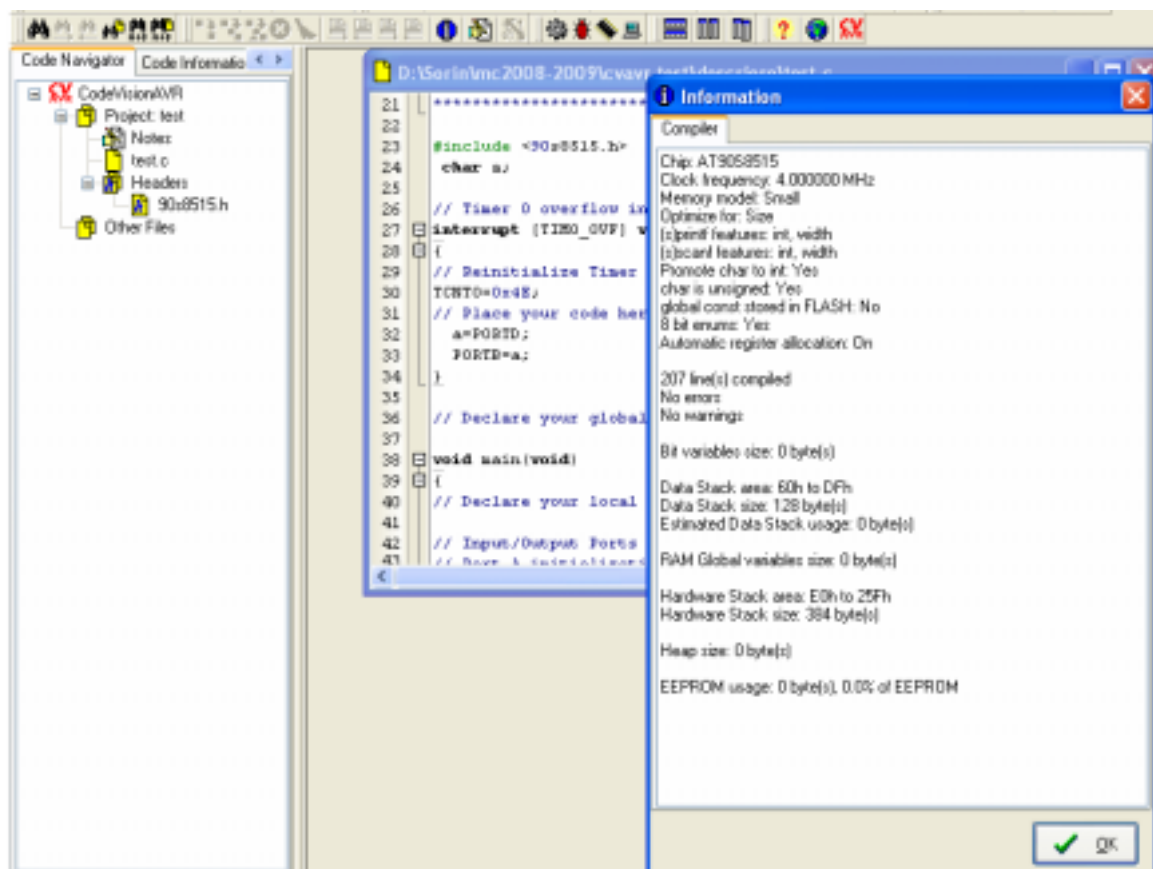
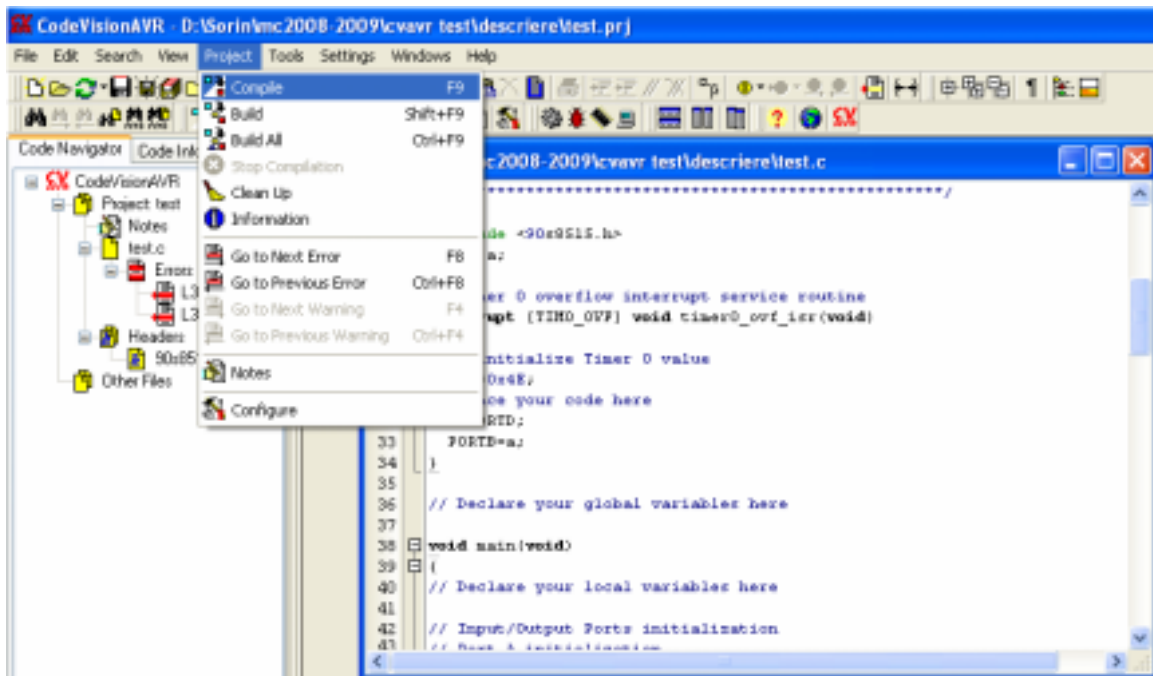
Codul utilizatorului poate fi scris in rutina de servire a intreruperilor sau in programul principal. De asemenea pot fi declarate variabile globale sau locale.

Vom introduce urmatoarele modificari in programul generat automat:

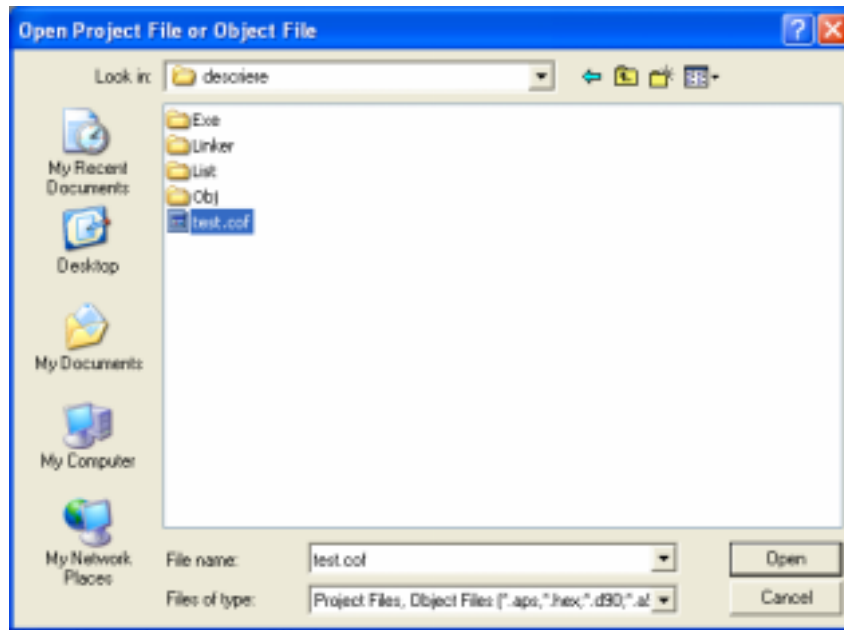
- se va declara o variabila globala a de tip caracter
- in variabila a se va citi portul PORTD
- valoarea lui a se va transmite in portul PORTB

Modificarile sint marcate in programul anterior cu litere ingrasate pe fond colorat.

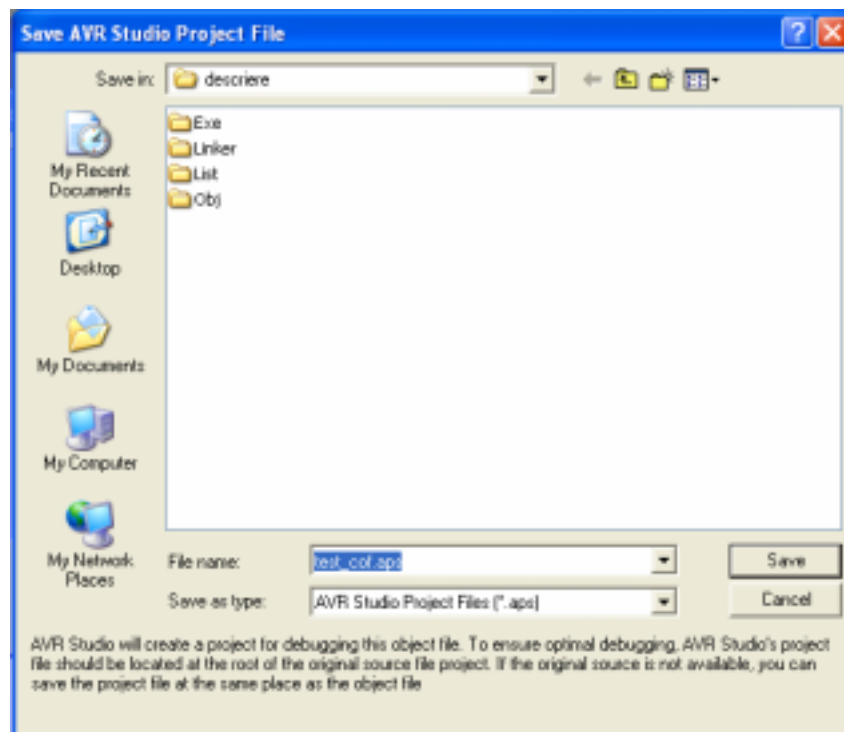
Programul va fi compilat si apoi rulat cu ajutorul depanatorului ASTUDIO.



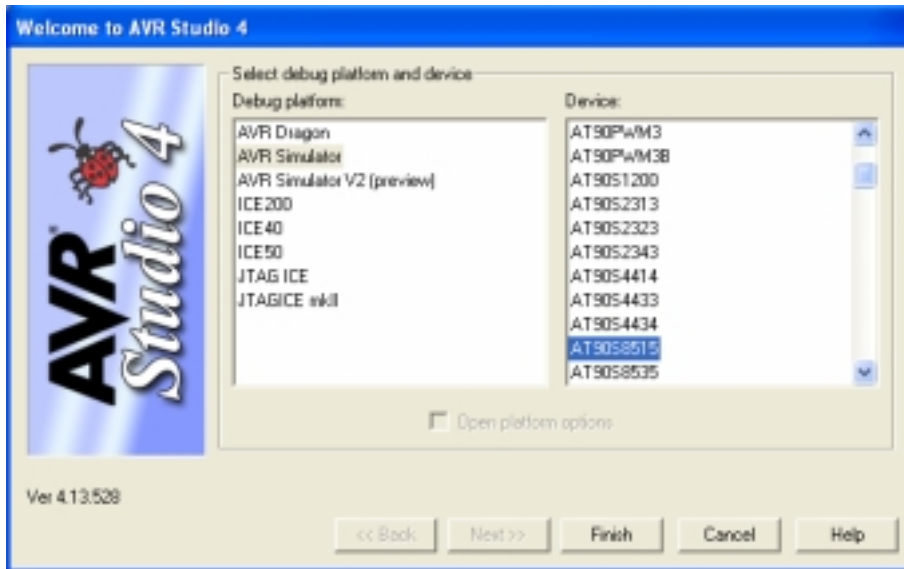
Daca nu au aparut erori se genereaza fisierul executabil cu comanda “Build”:



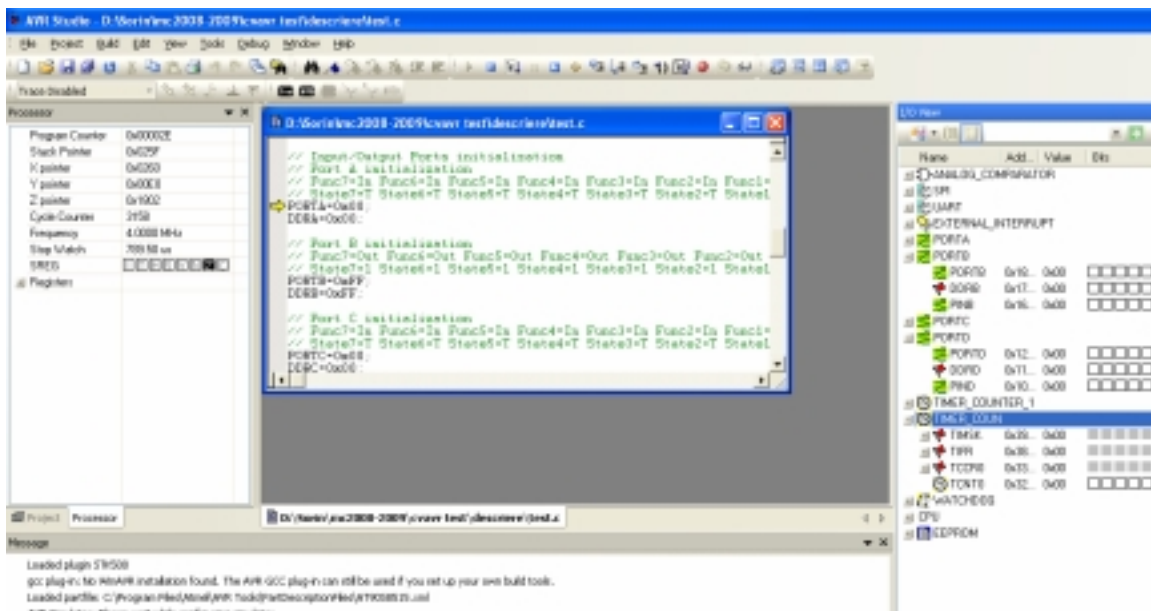
Se salveaza proiectul in format ASTUDIO:



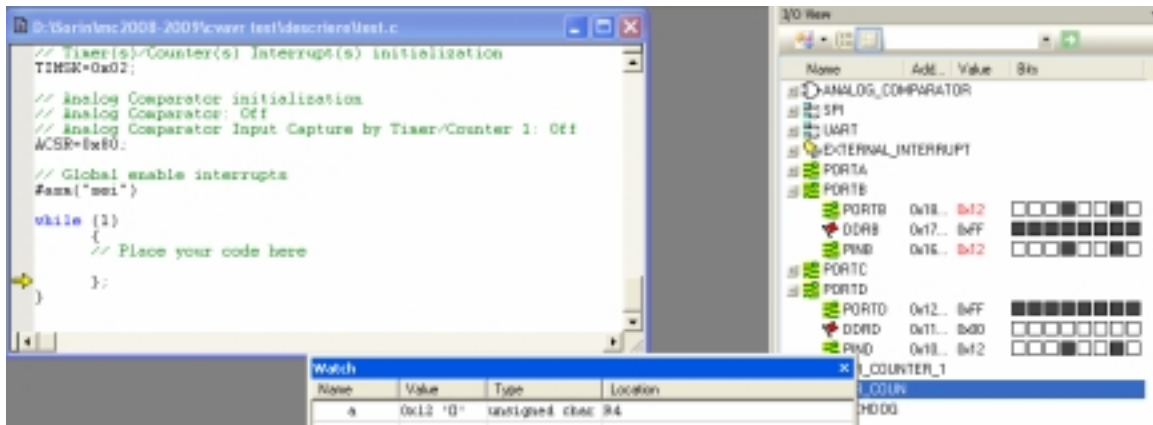
Se selecteaza platforma "AVR Simulator" si microcontrolerul "AT90S8515"




Se apasa butonul “Finish” si se expandeaza PORTB, PORTD si Timer0:

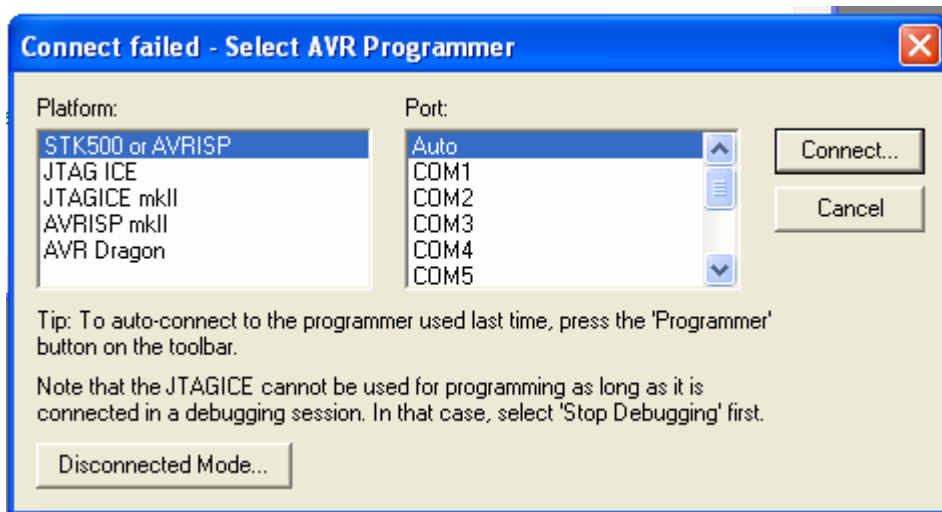


Programul poate fi simulat pas cu pas (cu tasta F11) sau complet (cu tasta F5). De asemenea se pot introduce puncte de intrerupere a programului - *breakpoints* (cu tasta F9). Se pot vizualiza resursele microcontrolerului (registre, memorie, porturi I/O) variabile ale programului.



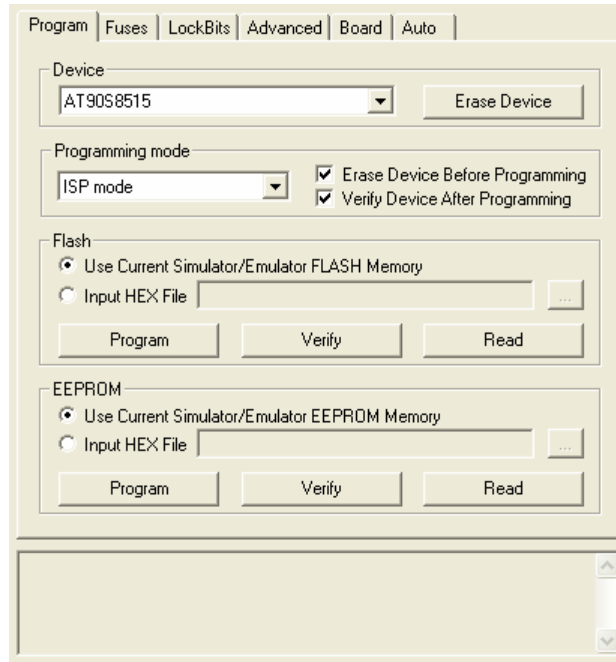
Executia in timp real a programului (pe placa de evaluare STK500)

Programul poate fi incarcat in placa de evaluare STK500 pentru a fi executat in timp real cu butonul :



Se selecteaza placa de evaluare STK500 si portul Auto. Conectarea se va face cu butonul "Connect".

Se va efectua programarea memoriei flash interne a microcontrolerului:



Descrierea functionala a porturilor de intrare – iesire (microcontroler AVR)

In figura 1 este ilustrata implementarea tipica a unui port de intrare iesire ai carui biti pot fi configurati individual. Sint necesare 3 registre:

- 1 registru de iesire (PORT sau PORT_x, x = A,B,C,D,...)
- 1 registru de intrare (PIN sau PIN_x, x=A,B,C,D,...)
- 1 registru de directie (DDR sau DDR_x, x=A,B,C,D,...)

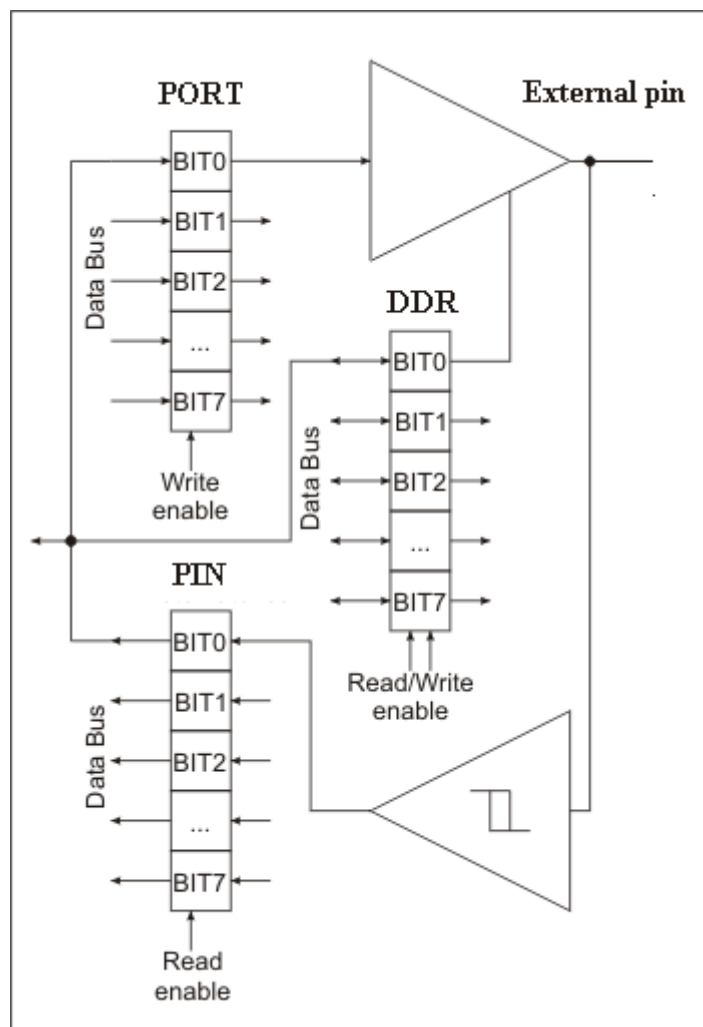


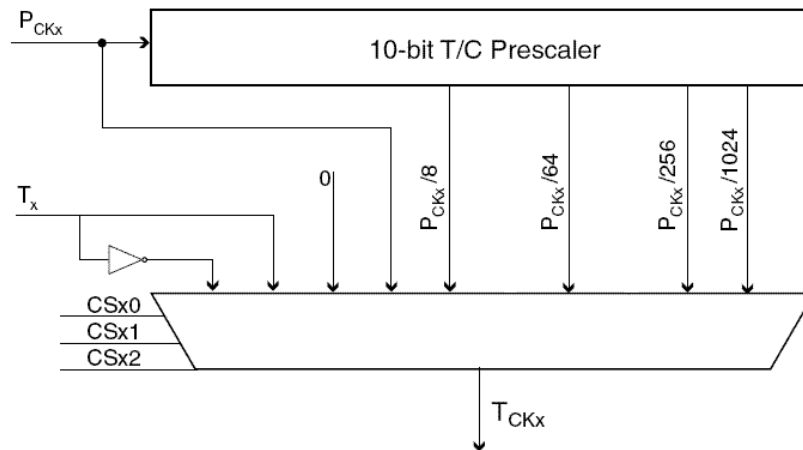
Figura 1. Port de intrare – iesire configurabil

Scierea registrului DDR cu 0 sau cu 1 valideaza sau nu bufferele ce au ca iesire pinii extern ai portului. O scriere in port se va efectua la adresa registrului PORT, iar o citire a portului se va realiza printr-o citire de la adresa registrului PIN.

Descrierea functionala a Timer-ului (microcontroler AVR)

Circuitul de timp programabil (Timer) este format din doua blocuri: un circuit de prescalare (ce va genera un tact T_{CKx} , utilizat in cel de al doilea bloc) si un registru-numerator ce va fi decrementat la fiecare front activ al semnalului de ceas generat de blocul de prescalare.

In figura 2 este ilustrat blocul de prescalare:



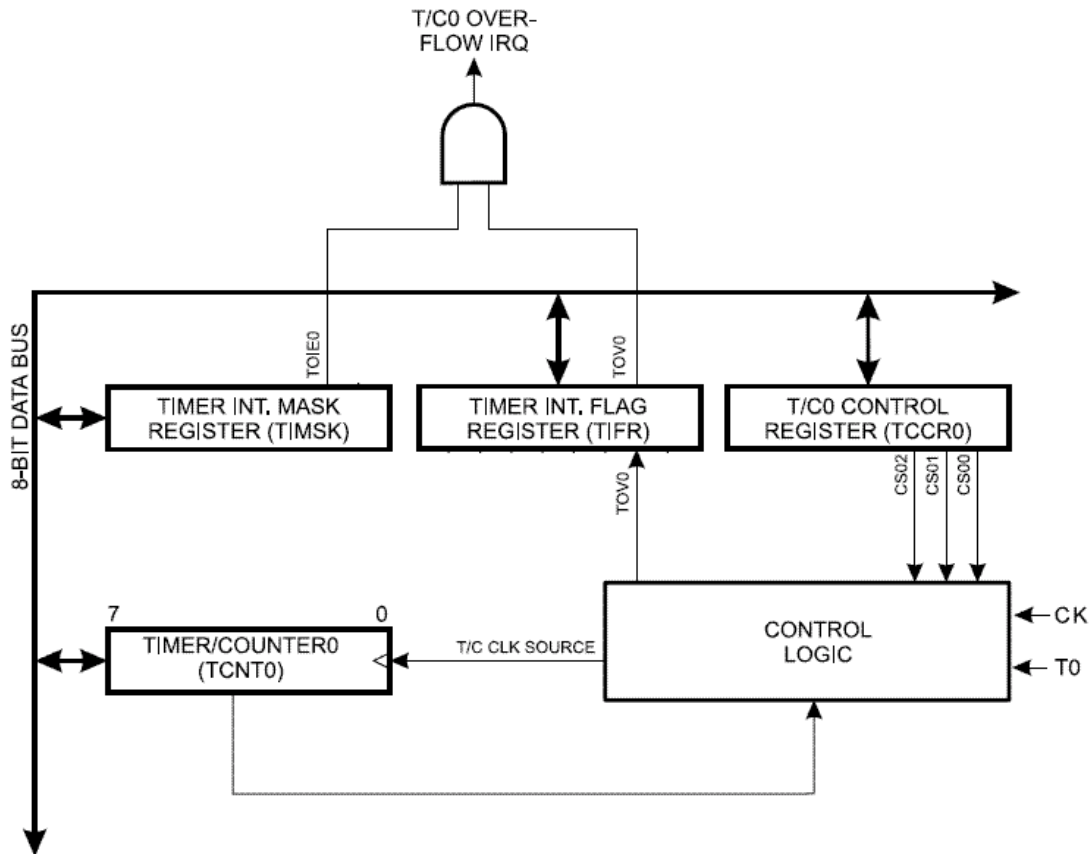
TCCR _x			Synchronous Timer0 & Timer1 $P_{CK} = CK$
Bit 2	Bit 1	Bit 0	
CS _{x2}	CS _{x1}	CS _{x0}	$T_{CK0,1}$
0	0	0	0 (Timer Stopped)
0	0	1	P_{CK} (System Clock)
0	1	0	$P_{CK}/8$
0	1	1	$P_{CK}/64$
1	0	0	$P_C/256$
1	0	1	$P_{CK}/1024$
1	1	0	External Pin Tx falling edge
1	1	1	External Pin Tx rising edge

Figura 2 Blocul de prescalare

P_{CKx} reprezinta semnalul de tact la intrarea in blocul de prescalare (ceasul de system). Sufixul $x = 0,1$ este numarul timeru-lui.

Exista un registru de comanda al timer-ului, TCCR_x; in acest registru bitii CS_{x2}, CS_{x1} si CS_{x0} sint utilizati ca intrari de selectie pentru multiplexorul care genereaza semnalul de ceas catre blocul registru-numarator ce va fi decrementat.

Ca semnal de ceas pentru decrementare poate fi utilizat si un semnal extern, Tx.
Structura detaliata a timer-ului 0 este ilustrata in figura 3.



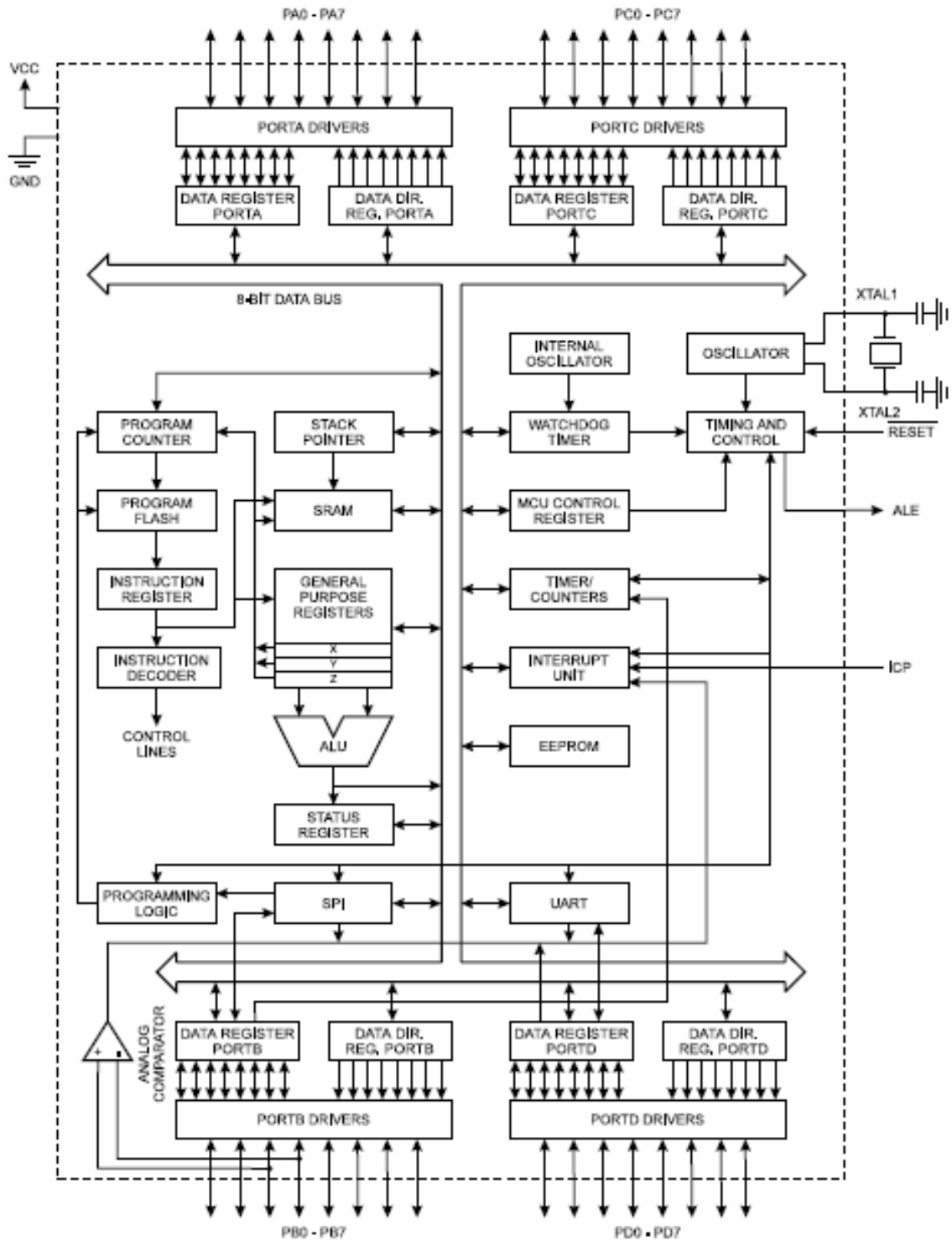
TOV0 – Timer Overflow: este 1 cind registrul numarator (TCNT0) prin valoarea 0
TOIE0 – Timer Overflow Interrupt Enable: este 1 cind intreruperile la trecerea prin 0 a registrului numarator (TCNT0) sint validate.

Figura 3. Structura timer-ului 0

Se observa ca exista 2 registre TIFR (Timer Interrupt Flag Register) si TIMSK (Timer Interrupt Mask Register), utilizate pentru a genera intreruperi la fiecare trecere prin zero a registrului numarator. Perioada intreruperilor este calculata prin impartirea perioadei ceasului generat de blocul de prescalare la o constanta prestabilita.

Anexa

Figure 1. The AT90S8515 Block Diagram



PLACA DE EVALUARE AVR- STK 500

Placa de evaluare AVR-STK500 permite studierea modului de dezvoltare a aplicatiilor cu microcontrolere Atmel-AVR.

Structura STK500 este prezentata in figura 5.

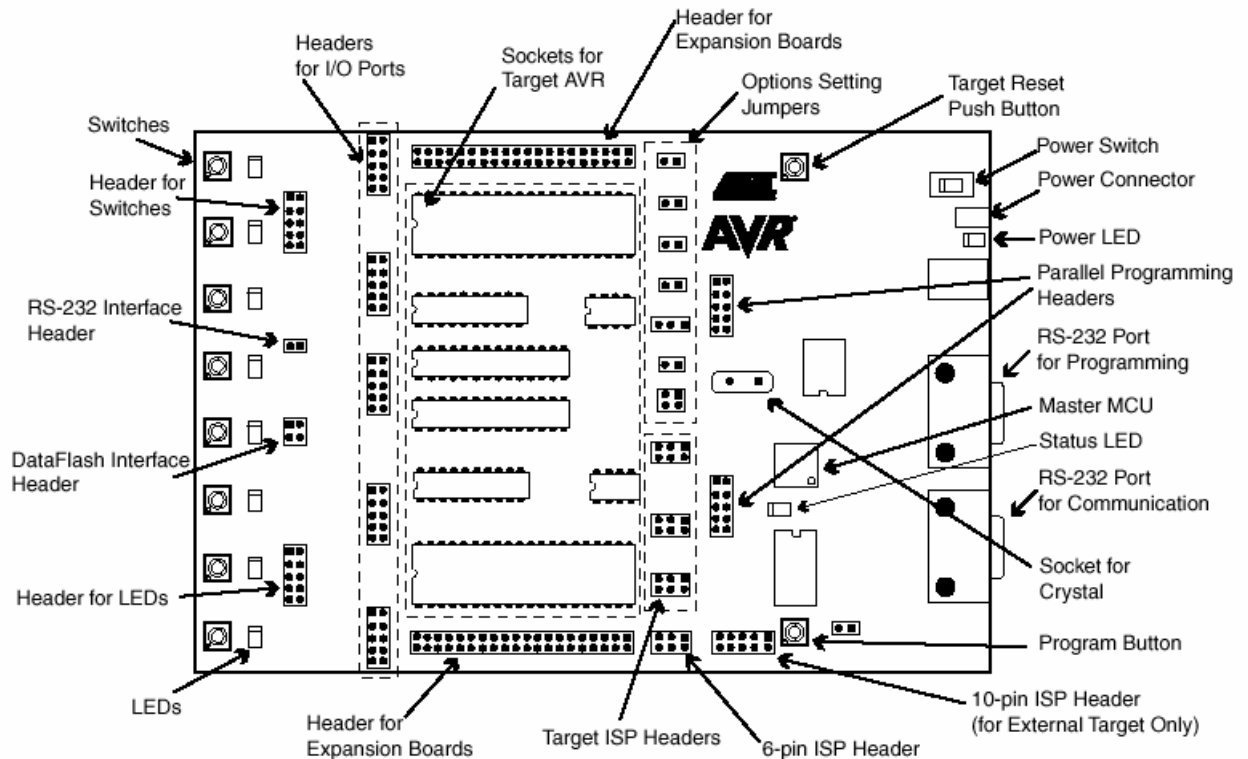


Figura 5. Structura STK500

Placa are in componenta urmatoarele blocuri:

- bloc de comunicare cu un calculator PC (pe linie seriala RS232)
- microcontroler AVR
- bloc de intrare – iesire (8 butoane SW0-SW7 si 8 LED-uri LED0-LED7)
- bloc de alimentare
- bloc de programare in circuit a memoriilor interne ale microcontrolerului AVR
- port de comunicatie seriala suplimentar

Arhitectura microcontrolerului AVR este prezentata in figura 6:

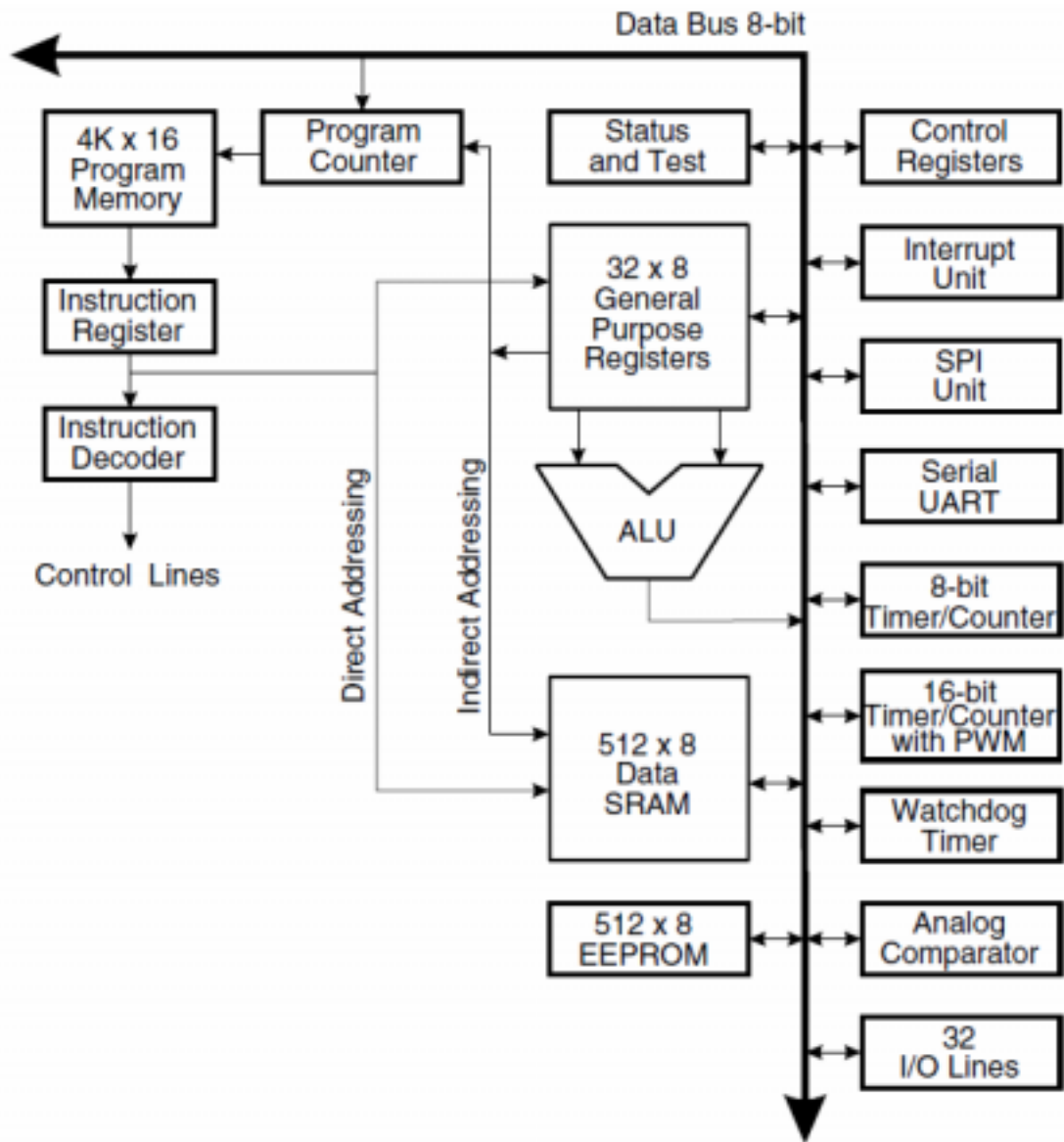


Figura 6 Arhitectura microcontrolerului AVR (AT90S8515)

De asemenea placa permite conectarea butoanelor SW si a LED-urilor pe diferite porturi ale microcontrolerului AVR.

Conectarea placii STK500 cu un calculator se realizeaza ca in figura 7.

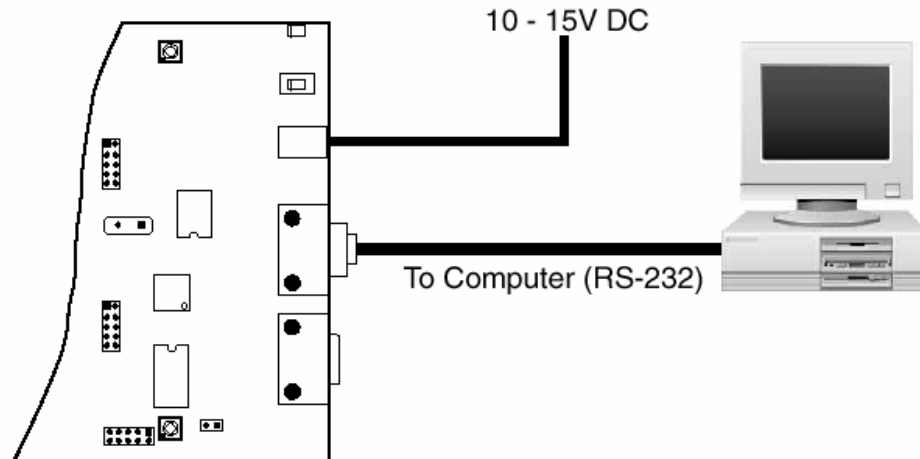


Figura 7. Conectarea STK500 cu un calculator

Comunicatia intre calculator si STK500 precum si programarea memoriei interne se realizeaza prin intermediul programului Astudio.

Structura butoanelor si a LED-urilor este urmatoarea:

Butoanele SW sunt realizate ca in figura 8 (starea de repaus este 1 logic). Toate butoanele SW0-7 sunt conectate cu portul PORTD care trebuie configurat ca intrare.

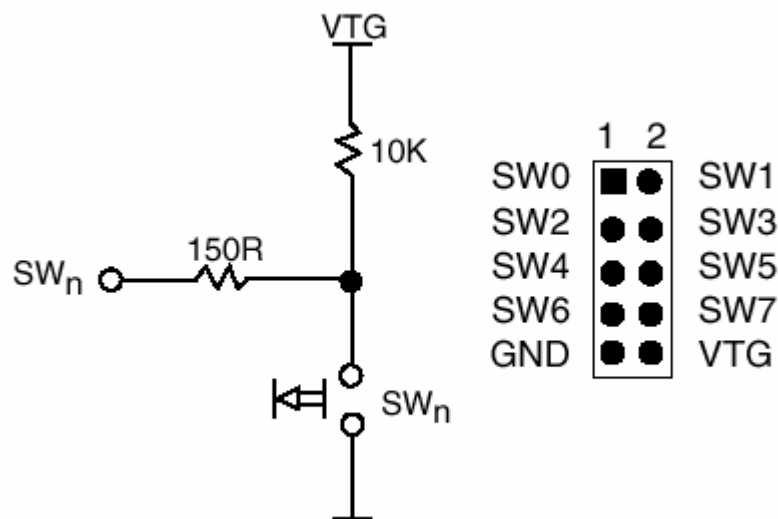


Figura 8. Butoanele SW0-SW7

Apasarea unui buton SW are ca effect transmiterea, pe pinul corespunzator al portului PORTD, a semnlului 0 logic.

LED-urile sunt conectate pe portul PORTB, care trebuie configurat ca iesire. Modul de conectare al unui LED este ilustrat in figura 9.

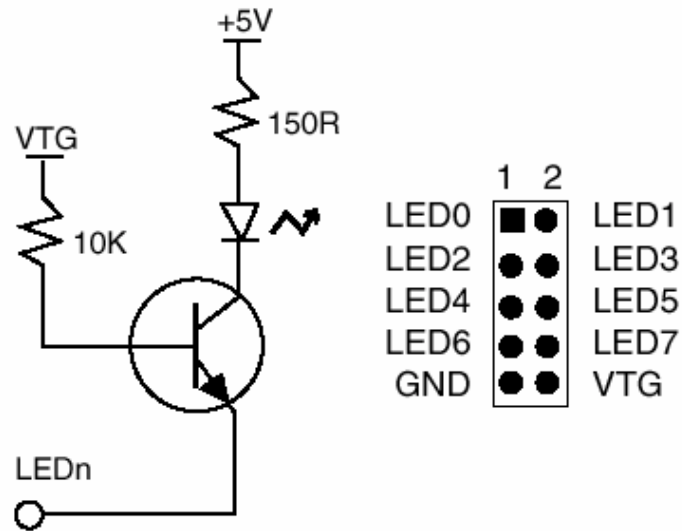


Figura 9. Conectarea LED-urilor

Pentru aprinderea unui LED se va scrie, pe pinul corespunzator al portului PORTB, valoarea 0 logic.