PM / Laboratorul Nr. 1
Introducere in Microcontrollere, Porturi I/O

Va trebui sa scrieti un program in assembler care sa aprinda cele opt leduri de pe placa de dezvoltare intr-o succesiune predeterminata. Dupa scrierea si simularea programului in „Proteus" acesta va trebui „transferat" pe microcontroller.

Pornind de la codul sursa dat de noi in suportul de laborator, realizati urmatoarele 2 lucruri:

1. Aprindeti intr-un mod coerent cele 8 leduri dupa o secventa data de tasta care a fost apasata, executand pentru fiecare un ciclu complet.
2. Faceti ca fiecare LED sa stea aprins aproximativ 0.5-1 secunde. (deci un ciclu complet sa dureze 4-8 secunde).

Pentru realizarea simularii:

1. Downloadaţi schema folosita ca punct de plecare in cadrul acestui laborator de pe siteul oficial PM.
2. Deschideti schema laboratorului 1 (fisierul lab1.dsn) cu aplicatia Proteus.
3. Observati in cadrul schemei microcontrollerul ATmega16 si cele opt leduri cuplate le pinii PORTA 0-7 precum si cele patru butoane legate la PORTD. Cand microcontrollerul comanda HIGH pe unul din pini, ledul cuplat la pinul respectiv se aprinde. Similar, cand pe unul din pini este LOW, ledul ramane stins.
4. Microcontrollerul executa un program indicat de voi. Pentru a specifica acest program, trebuie sa indicati un fisier sursa Proteusului. In meniul Source dati click pe Add/Remove Source file, alegeti New si selectati fisierul sursa al laboratorului 1 (lab1.asm), pe care in prealabil il downloadati de pe site. La Code Genenration Tool, alegeti avrasm32 (este compilatorul surselor noastre). Urmatoarea etapa este sa compilati acest fisier – selectati din meniul Source comanda Build All. Acum dati click drepta pe microcontroller, apoi click stanga. In meniul care va apare, selectati la Program File fisierul lab1.hex.
5. Pentru a executa programul, apasati F12 sau Debug à Execute. Microcontrollerul vostru va aprinde becul verde la un semafor si pe cel rosu la celalalt. Puteti executa pas cu pas programul, pornind cu CTRL+F12, apoi avansand cu F10 (salt peste apeluri de rutine) sau F11 – executie instructiune cu instructiune, inclusiv apeluri de rutine. Cu F9 puteti pune breakpoint.

Pentru programarea microcontroller-ului:

1. Conectati placuta alimentata la portul serial de programare. Deschideti din maniul Start al Windowsului programul PonyProg2000 din gupul PonyProg. Setati tipul microcontrollerului AVR micro ATmega16. Din meniul Setup alegeti Interface Setup, selectati acolo programatorul Serial, pe portul COM1 (sau portul pe care ati legat cablul de programare la calculator) si API-ul SI Prog API. Salvati setarile.
2. In meniul File, selectati Open Program File, selectati lab1.hex.
3. In meniul Command, alegeti comanda Write All. Daca totul a decurs Ok, dupa verificare va va aparea mesajul Write Successful.

Sfaturi:

1. Trebuie doar sa extindeti codul dat in mod similar cu ce am facut noi in suportul de laborator.
2. Trebuie ca intre trecerea de la un timp la altul sa inserati un apel la o rutina ca sa dureze 0.5-1 secunde. Cum faceti ca rutina sa dureze 0.5-1 secunde? In corpul ei incrememntati un registru de la 0 la 255, apoi cand ajunge valoarea la 255 resetati acest registru si incrementati un al doilea registru. Cand valorea celui de al doilea registru ajunge la 255, il resetati pe si acesta, si incrementati un al treilea registru. In fine, cand valoarea acestui de-al treilea

registru ajunge la aproximativ 10-25 (alegeti voi o valoare intre 10 si 25), il resetati si iesiri din rutina de asteptat – adica treceti la urmatorul timp al semaforului.

Materiale Suplimentare

1. http://www.avr-asm-tutorial.net/avr_en/beginner/REGISTER.html
   despre registrele Atmega16

2. http://www.avr-asm-tutorial.net/avr_en/beginner/PORTS.html
   despre porturi si I/O cu Atmega16