Mikrokontrollerek programozása BASIC nyelven

Az alapok

Már olyan sokszor megkaptam a kérdést:

szeretném beleásni magam a mikroprocesszorok programozásába, mi kell nekem ehhez?

És ilyenkor elkezdem, hogy:
kell választani egy mikrokontroller típust, kell egy hardver, amin tudunk próbálkozni, kell egy fordító program, ami a mikrokontroller számára emészthetővé teszi a programunkat, kell egy égető, kell egy ...

De inkább most megpróbálom a hosszas telefonos magyarázkodások helyett leírni most mindezt.

Kell egy proci

Először is választanunk kell egy mikrokontroller típust. Na lássuk. Gyárt mikrokontrollert az INTEL, a Mitsubishi, a TEXAS, a Zilog, a ... Szóval sokan. Nekünk persze olyan kell, ami kis darabszámban is, könnyen, és olcsón beszerezhető, nem kell hozzá drága fejlesztő rendszer, égető. Ja, és bőven legyen hozzá felhasználási példa, doksi a neten.

Na így már drasztikusan csökken a szóba jöhető típus családok száma, tulajdonképp háromra.

Az „MCS51-es" mikrokontrollerek

A mikrokontroller elnevezést, de magát a mikrokontrollert is az INTEL találta fel. Az MCS51-es típuscsaládja pedig egy „szabványt" teremtett, nagyon sokáig ezekből adtak el legtöbbet, messze a legnépszerűbbek voltak. (A 8031, 8032 még külső epromban tárolta a működtető programot, később megjelentek a belső epromos 8751, 8752-esek, amikkel már valóban 1 chipes mikroszámítógépeket lehetett csinálni.) Aztán a 90-es évektől kezdve az INTEL inkább a „nagy" processzorokra fókuszál, az MCS51-es utasításkészlettel, felépítéssel eladott mikrokontrollerek többségét ma már a másodgyártók adják el, a DALLAS, SIEMENS, a PHILIPS, a WINBOND, ATMEL, stb.

Ezek közül talán az ATMEL adja a legnagyobb választékot, a 20 lábú 89C2051, 89C4051, és a nagyobb, 40 lábú – plcc tokban 44 – 89C51/52 típusok a legismertebbek. És megjelent ezeknek a típusoknak a sorosan, akár áramkörben is programozható változatai, a 89S-esek. (89S2051, 89S51, stb.)

Miért választhatnánk ezt a „családot" ?

Viszonylag olcsón van hozzá programozó, legalábbis az ATMEL gyártmányokhoz, és ezek könnyen be is szerezhetőek.

Mivel több gyártó is csinál „kompatibilis” mikrokontrollereket, nem vagyunk egyetlen gyártóra utalva.

Talán ennek a leglogikusabb a belső felépítése, ha assembly nyelven szeretnénk programozni, akkor ezt ajánlanám.

És akkor pár dolog, ami ellene szól:

Bár sokan gyártanak ilyen felépítéssel mikrokontrollert, de igazából csak az „ATMEL-es” alap típusokat egyszerű beszerezni, szóval látszólag sok típusból tudunk választani, valójában mégsem.

Hasonló a helyzet az égetéssel is, a nagyobb tudású ATMEL-eket, és a többi gyártó típusait csak a komolyabb – drágább – égetők tudják programozni.

A MICROCHIP PIC-jei:

A MICROCHIP a 90-es évek elején robbant be a mikrokontrollerek piacára, és hamarosan elképesztő sikeres lett. De miért? Hiszen nem volt nagy múltja a mikrokontroller gyártásban. Kicsi se. Ráadásul az első típusok tudása, utasításkészlete messze alatta volt az akkor favoritnak számító MCS51-es családba tartozó típusoknak.

Az ötlet a következő volt: a MICROCHIP ingyen adta a mikrokontroller programozásához szükséges szoftvereket, csinált egy újra programozható, eeprom memóriás típust, a 16C84-et, és ehhez publikált egy egyszerű, után építhető programozó áramkört. (Ma talán nehéz megérteni, hogy mi volt ebben a nagy durranás, a jelenleg kapható mikrokontrollerek többsége újraírható, és egy csomó szoftver letölthető a gyártók honlapjáról. De annak idején ez nem volt így.) Az előbbiek miatt hamar rákattantak PIC-ekre a lelkes amatőrök, és a diákok, akik aztán a munkahelyi fejlesztéseknél is a már megismert típusokat részesítették előnyben.

A MICROCHIP abban is úttörő volt, hogy rengeteg alkalmazási példát adott közre, amit tovább bővített a rengeteg közzé tett „hobbysta” áramkör, program. Ma már ott tartunk, hogy sokan keverik a mikrokontroller, és a „pic” megnevezést.

Közben a MICROCHIP kijavította az induló szériák gyerekbetegségeit. Ma már az összes típus újraírható, flash memóriával rendelkezik, elképesztő típus választékban, a 6 lábútól a „százlábúkig”, 8, 16, vagy akár 32 bites maggal kaphatóak, olyan speckó beépített funkciókkal, mint pl. a touch – érintő – billentyű, vagy képernyő kezelés, usb, can, lin, vagy épp infra kommunikáció, ethernet csatoló, és még rengeteg minden.

Foglaljuk össze az előnyöket:

Nagyon sok, olcsón, könnyen beszerezhető típus. (Vannak persze olyan típusok, ahol az 1-2 darabos beszerzés nehézkes, de a „maradék” is bőven elég.)

A MICROCHIP megtartotta azt a jó tulajdonságát, hogy ingyen adja a például az assembly fordító programot, olcsó, sőt, után építhető programozót biztosít, a net pedig tele van alkalmazási példákkal.

A PIC-ek népszerűsége miatt aztán egy sor cég gyárt hozzá fejlesztési segédeszközöket, fordító programokat, egyebeket.

Ami nekem nem tetszik a PIC-ekben:

Az assembly programozás utasítás készlete, de még a mikrokontrollerben használatos megnevezések is mások, mint a többieknél. (Pl. minden programozással foglalkozó könyvben accumulator a „fő" regiszter neve. Na ezt pl. átnevezték W regiszternek. A fix értéket mindenhol „konstans"-nak nevezik, na PIC-nél ez „literal". Meg ilyenek. Szörnyű. De ez persze csak akkor gond, ha assemblyben szeretnénk programozni.)

Talán túl sok is a PIC típusok száma. A MICROCHIP nap mint nap dob piacra új, és újabb típusokat, egyben vezet ki régebbieket a gyártásból. Csakhogy a nagyobb tudású utódok nem mindig tudják 100 százalékosan helyettesíteni a kifutott változatot. Lehet, egy kicsit módosítani kell a programon, stb. Tipikusan a „kevesebb több lenne" esete.

Az AVR mikrokontrollerek:

Az MCS51-es mikrokontrollereknél már találkoztunk az ATMEL nevével. A gyártó nem elégedett meg az INTEL kompatibilis procik gyártásával, csinált egy saját „családot", amit AVR néven vezetett be. Ezek egy része láb kompatibilis az MCS51-es család legnépszerűbb tagjaival, de teljesen más belső felépítéssel, utasítás készlettel rendelkeznek. (Egyedül – valami idióta módon – a reset láb polaritását változtatták meg, amire mindig figyelni kell, ha egy MCS51-es mikrokontrollert AVR-re cserélünk.)

Az ATMEL is átvette a MICROCHIP ötletét, azaz ingyen szoftverekkel, alkalmazási példákkal csábítja magához a felhasználókat. Nem is eredménytelenül, a PIC-ek után az AVR-eknek a legnagyobb a rajongótábora.

De van pár dolog, ami miatt már nem tervezem be új áramkörökbe az AVR-eket. Az egyik, hogy a mikrokontrollerbe épített adat memória tartalmát – amibe olyan adatokat rakhatunk el, amikre egy áramszünet után is szükségünk van – hajlamos „elfelejteni". Pl. egy kapcsoló óra áramkörnél szembesültünk azzal, hogy a ki/bekapcsolás után a beprogramozott kapcsolási idők közül néhány eltűnt, vagy értelmetlen értéket vett fel. Ezt még a kezdő széria hibájának is fel lehetne fogni, de pl. a 90s2313, 90s8515-öt felváltó tiny2313, atmega8515 típusoknál ugyanúgy előjött.

Egy másik probléma az IC-k beprogramozásakor jön elő. A mai mikrokontrollerekbe többségébe a működtető programon kívül, egy csomó beállító bitet is be kell „égetni" a működéshez. (Itt lehet megadni, pl. milyen oszcillátorral biztosítjuk az óra jelet, a mikrokontroller reset-jével, vagy automatikus resetjével – watchdog – kapcsolatos beállításokat, meg még rengeteg dolgot.)

Ezek a beállító bitek a PIC-ek esetében a működtető programban is megadhatóak, a beégetendő tartalom ezeket már egyértelműen tartalmazza. Az AVR-eknél nem! Itt bizony a programozáskor kell egyenként beállítani ezeket, ami nem csak kényelmetlen, de elképesztően meg is nehezítheti a mikrokontroller működésre bírását, főleg ha nem saját magunk írtuk a

programot, és nem tudjuk, hogy pl. milyen órajel osztásokkal – mert ez is beállítható – számolt a működtető program. Ilyenkor lehet próbálkozni a különböző beállításokkal... És még egy órási baklövés, a program beírás kapcsán. Az AVR-ekbe – is – két féle módon vihetjük be a működtető programot. Vagy sorosan, amikor bitenként léptetjük be az adatokat, vagy párhuzamosan, amikor bájtonként töltjük fel a program memóriát.

A soros programozás előnye, hogy mindössze 3 kivezetésen keresztül programozható az IC – a mikrokontroller a panelon programozható, és egyszerű, olcsó lehet a programozó áramkör - cserébe egy kicsit több idő kell a beíráshoz, mint a párhuzamos programozás esetén. Néhány beállító bit – már megint ezek a beállító bitek – a soros programozási módban csak beírható, de törölni már nem lehet. Tehát pl. könnyedén letilthatjuk a soros programozás lehetőségét, és akkor gyakorlatilag ki is zártuk magunkat a mikrokontrollerből, mert innentől kezdve se törölni, se újraírni nem lehet már azt a soros programozónkkal. (!!!)

Kell egy hardver:

Ahhoz hogy egy mikrokontrollerre programot írjunk, természetesen kell egy mikrokontrolleres áramkör.

Az előbbiek alapján akkor választanunk kell egy mikrokontrollert. Én jelenleg a MICROCHIP PIC-ekre szavazok, a mostanában fejlesztésre kerülő áramköreinkbe ezek kerülnek. És ezen persze rögtön el is lehetne vitatkozni - feleslegesen. Olyan ez, mint ha arra kéne válaszolni, hogy a német, az olasz, vagy mondjuk a japán kocsik-e a jobbak. Nincs jó válasz.

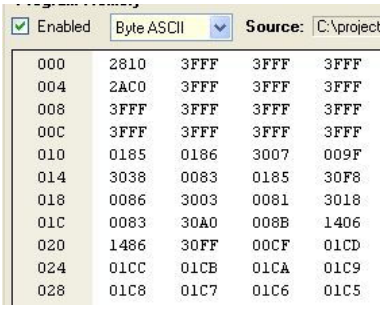
Szóval kell egy panel, amin a mikrokontrolleren kívül persze legyen egy 5 voltos stabilizátor a tápfeszültség előállításhoz, néhány LED mindenképp, de jó ha van LCD csatlakozó is.

Abból a választékból, amit a kísérletekhez tudok ajánlani, a legegyszerűbb a PIC8 panel, ami egy 8 lábú procit, 5 ledet, meg a tápot tartalmazza. De a tanuláshoz inkább megfelel a PICDEMO panel, ez már 40 kábú PICet tud fogadni, jut láb az LCD-nek, 8 LEDnek, rs232 csatlakozásnak, ha PC-s kapcsolatot is szeretnénk.

Ha pedig valamilyen vezérlési feladatot akarunk megoldani, akkor jó ha van pár relés kimenetünk, a bemenő jeleket pedig sorkapocsokba tudjuk kötni. A PICPLC1 panelen 4 bemenet, és 1 relés kimenet, a PICPLC8-on 8 bemenet, és 8 relés kimenet, a PICPLC16-on pedig 16 relés kimenet van kiépítve.

Kell egy fordító

Amikor egy mikrokontrolleren fut egy program, akkor gyakorlatilag az történik, hogy a memóriájába beprogramozott adatokat veszi elő szép sorban, és annak megfelelően csinál valamit. Mondjuk magas szintre állítja egy kivezetését – portját – ami aztán pl. egy LED világítását okozza. Ha tehát „belenézünk" egy mikrokontroller program memóriájába, akkor egy csomó adatot látunk:



Elvileg úgy is lehetne programozni a procit, hogy megnézzük a leírásában, hogy melyik utasításhoz milyen kód tartozik, aztán azt beleírjuk a memóriába, majd a következőt, és így tovább. Ez a „gépi kódú programozás”. Hihetetlenül macerás, már rég nem divat.

Nyilván valami „emberi” nyelven szeretnénk megírni a programot, amit aztán valahogy át kell alakítani a proci számára is érthető – végrehajtható – kódokká. A probléma megoldása egy „fordító program”. Van sok fajta. Annyi, ahány számítógép programozási nyelv van. Azon belül pedig minden mikropocesszor családhoz van külön, több változatban.

A mikrokontrollerekre általában assembly, C, és BASIC nyelvű fordítókat csinálnak. Amúgy még létezik pl. PASCAl fordító, és ide sorolhatóak talán a PLC-k egyedi programozási nyelve is. De maradjunk a három legelterjedtebbnél. Melyiket válasszuk?

Az assembly programozáskor közvetlenül adunk utasításokat a mikroprocesszornak. Ezen a nyelven lehet tökéletesen kihasználni a lehetőségeket, bizonyos feladatok nem is oldhatóak meg más nyelven. (Pl. mondjuk hogy van egy touch billentyű kezelő funkció egy mikrokontrollerben, de erre még nem csináltak BASIC nyelvű utasítást.) De aki assemblyben akar programot írni, annak tökéletesen kell ismerni az adott mikrokontroller belső felépítését. Talán egy példa erre. Mondjuk hogy be akarunk gyújtani egy LED-et, ami a mikrokontroller RB6 kivezetésére van kötve, azaz ezt a lábat magas szintre kell állítanunk. Az assembly programnál ez így néz ki:

```
movlw     .0
movwf     trisb
bsf       portb,6
```

Azaz az első két utasítással beállítjuk, hogy a mikrokontroller portb kivezetései legyenek most kimenetek – mert azokat lehetne épp bemenetként is használni – majd kiadjuk az „állítsd a B port 6. kivezetését 1-be” utasítást.

Hát nem túl barátságos ez a nyelv...

Nézzük akkor ezt BASIC-ben:

```
HIGH PORTB.6
```

Ez azért könnyebben emészthető, ugye? (Ja igen, a mikrokontrollernek most is tudnia kell, hogy kimenetként használjuk most a B portot, de ezt BASIC fordító rendezi majd, azaz beszerkeszti a végrehajtandó kódok közé, az ehhez szükséges utasításokat.)

A harmadik szóba jöhető lehetőség, a C nyelvű programozás. Ezen a nyelven lehet a legáttekinthetőbb programokat írni, a „nyelvezete” azonban nem olyan könnyen érthető mint a BASIC-é.

Szóval ha gyorsan akarunk eredményeket elérni – és most ez a cél – akkor szerintem a BASIC nyelv kell nekünk. A C meg az assembly egyenlőre maradjon a profiknak. (Azaz majd nekünk, de később.)

Jó, akkor tehát kell egy BASIC fordító program. (angolul compiler)

Több cég is csinál ilyent.

Itt van pl a „PICBASIC PRO” program, amit a Microengeniering Labs árul. Leírások, példaprogramok, egyebek a www.melabs.com címen.

Amikor ezt a leírást írom, ingyen beszerezhető a PIVBASIC PRO 3 „student” verziója. De erről van egy önálló leírás.

És meg van még a régi verzió demó programja is. A DEMO verziónak van pár korlátozása. Az egyik, hogy maximum 32 soros lehet a programunk, a másik hogy erősen korlátozva van a választható picek választéka.
Az ingyenes, demo verzió megtalálható a "mikroklub CD"-n is a MIKROKLB\PICBASIC\picbasicpro2-demo könyvtárban. Telepíthetjük. De erről egy külön leírásban, a MIKROPROG-2.PDF-ben bővebben. (De jelen pillanatban a 3-as verzió „studentje” sokkal jobb választás)
De a kezdőlépéseknél ezek még nem igazán problémák, amúgy meg meglepően sok minden belefér abba a 32 sorba...

Kell egy mikrokontroller programozó

Mondjuk hogy elkészült az első programunk. Már csak az kell, hogy ez valahogy belekerüljön a mikrokontrollerbe. Egy olyan eszköz, ami bele programozza – égeti, írja, tölti, ahogy tetszik – a működtető kódokat a mikrokontrollerbe, esetünkben egy PIC-be.
Rengeteg IC programozó eszköz kapható, de ha csak PIC-hez akarjuk azt használni, akkor jelenleg a legjobb, mellesleg a legolcsóbb választás, a PICKIT2. Gyors, USB-s, és rengeteg PIC típust ismer, lévén hogy a MICROCHIP fejlesztette. Ez egy úgynevezett soros programozó, amivel akár egy áramkörön belül beprogramozható a mikrokontroller.
De erről egy külön leírás szól, a részletekbe itt most nem megyek bele.

Kapcsolódó dokumentáció:

picbasic-A-fordito-progi.pdf : a PICBASIC PRO 3 basic fordítóprogramról

picbasic-Forditas-betoltes.pdf : egy BASIC program lefordítása, betöltése a mikrokontrollerbe

picbasic-Mintaprogik.pdf : röviden a BASIC mintaprogikról

A PIC BASIC PRO régi, demó verziójáról olvashatunk a MIKROPROG-2.PDF leírásban. Ehhez egy konkrét példa, egy BASIC program letöltésére a PICDEMO panelba a PICBASICDEMO.PDF leírásban.

Az előbbi leírások, programok letölthetőek a lenti honlapcímről, vagy megtalálhatóak a „mikroklub cd”-n.

Végül nincs más hátra, mint hogy sok sikert kívánjak a használathoz. Viszontlátásra: Torkos Csaba 8100 Várpalota Táncsics u. 7. Telefon: napközben: 88/473-784, egész nap: 06/30/9472-294, email: mikroklub@vnet.hu Internet: http://www.mikroklub.hu, http://www.eprom.hu