SEGÉDLET
"Mikrokontroller programozás" laboratóriumi gyakorlataihoz

A gyakorlatok eredményes elvégzéséhez szükséges ismeretek:
- IBM PC kezelése
- Szövegszerkesztés, Windows Commander
- A SAB80C515 mikrokontroller felépítése, utasításai
- IAR 8051 Assembler
- HyperTerminál (a PC-n futó kommunikációs program)
- A mikrokontroller oktatórendszer monitorprogramja

A HyperTerminal

A **HyperTerminal** egy a PC-n futtatható kommunikációs program. Segítségével a PC-t a mikrokontroller terminájaként használhatjuk. A rendszer vezérlését ilyenkor a mikrokontroller oktatórendszer **monitorprogramja** végzi.

Ha a Windows Commander parancssorába begépelünk egy parancsot, az operációs rendszer megkísérli a paranccsal megegyező nevű futtatható programfájl elindítását az éppen használt meghajtó aktuális alkönyvtárában. Ha itt nem találja, akkor sorra veszi a PATH környezeti változóban megadott alkönyvtárakat.

A laborban található számítógépek környezeti változóit úgy állítottuk be, hogy valamennyi futtatandó program megfelelő paraméterekkel történő elindításához szükséges BATCH programok elérési útját tartalmazza. Ezért elegendő egy saját alkönyvtár létrehozása után ebben az egy alkönyvtárban dolgozni. A szükséges programok a nevük begépelésével futtathatók. A parancsokat a Windows Commander parancssorába kell begépelni.

A HyperTerminal indítása a Windows XP képernyőn (asztal) található **Micro1** nevű piros-sárga telefonokat ábrázoló ikonra kattintással történik. A HyperTerminal az adott munkahely megfelelő soros portján (COM1 vagy COM2), 9600 bit/s sebességgel indul el. Ha ettől eltérő paraméterek megadása szükséges, kérje a mérésvezető segítségét.

A mikrokontroller oktatórendszer monitorprogramja

Egy mikroprocesszoros rendszeren **mindig fut** valamilyen program. Ahhoz, hogy bekapcsolás után a rendszer működőképes legyen, valamilyen nem felejtő memóriában kell elhelyezni a programját.

A mikrokontroller oktatókészlet használata során a saját futtatandó programjainkat a PC merevlemezén tároljuk és innét szeretnénk **letölteni** az oktatórendszer memóriájába. Ezért az oktatórendszeren bekapcsolás után el kell indulni egy olyan működtető programnak, ami ezt képes fogadni és futtatni. A programfejlesztés során szükségünk lesz még arra is, hogy a programunk által módosított memóriarekeszek és/vagy regiszterek tartalmát megvizsgáljuk, a program hibáit megkeressük.

Azt a programot, ami az oktatórendszer és a programot fejlesztő felhasználó közötti kommunikációt lehetővé teszi **monitorprogramnak** hívják.

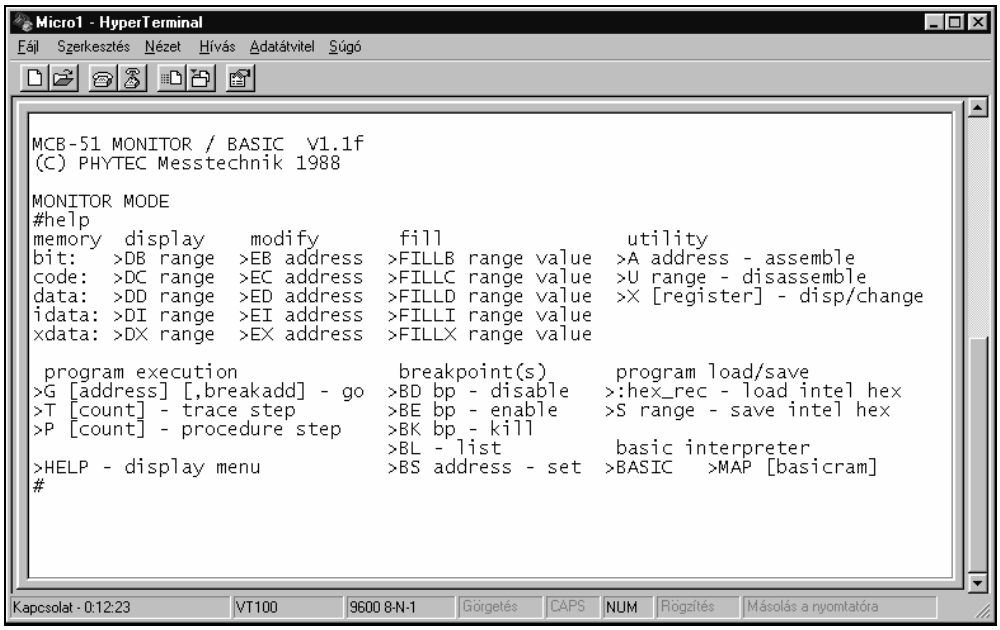
A monitorprogram a külső programmemóriában helyezkedik el a 8000H címen. Hívása saját programból az

```
LJMP        8000H
```

utasítással lehetséges. Ekkor a monitorprogram bejelentkezik és egy # prompttal jelzi, hogy parancsot vár.

Ha a PC-n elindítjuk a HyperTerminalt, a PC és az oktatórendszer soros csatornán csatlakoztatva van egymáshoz és az oktatórendszer be van kapcsolva, a monitorprogram **bejelentkezik** a képernyőn. Ezt minden egyes esetben megteszi, ha az oktatórendszer piros RESET

gombját megnyomjuk. Egy # prompt jelzi, hogy a monitorprogram **parancsot vár**. Az egyik igen hasznos parancs a **help**. Begépelése után az 1. ábrán látható képernyőt kapjuk.



1. ábra

A monitorprogram vezérlőbillentyűi és billentyűkombinációi

Ctrl D, vagy **Delete**	A kurzor aktuális pozícióján található karakter törlése.
Ctrl H, vagy **Backspace**	A kurzor előtti karakter törlése.
Ctrl L, vagy ←	A kurzor mozgatása balra.
Ctrl R, vagy →	A kurzor mozgatása jobbra.
Ctrl X	A sor törlése.
Enter	A sor értelmezésének indítása.
Esc	Ha a sor elején használjuk, akkor az előző parancsot visszaírja, tetszés szerint javíthatjuk. Sor közben hasonló az Enter-hez.
Ctrl O	A kurzort a sor elejére ill. végére mozgatja.

Amikor a monitorprogram valamilyen listát jelenít meg a képernyőn:

Ctrl S	A listázás ideiglenes felfüggesztése.
Ctrl Q	A felfüggesztett listázás folytatása.
Ctrl C	A parancsvégrehajtás megszakítása.

A monitorprogram parancsaiban használt változók

A monitorprogram minden számot (címet, adatot) hexadecimálisan értelmez, ezért a számtípus jelzésére szolgáló H betűt nem kell (**nem szabad!**) kitenni. Az egyes monitorparancsok lehetnek paraméter nélküliek, vagy tartalmazhatnak 1, 2 vagy 3 paramétert. Az egyes monitorparancsok ismertetése során a zárójelbe tett paraméterek nem kötelezőek (opcionálisak), de egy második vagy harmadik opcionális paramétert csak akkor használhatunk, ha az előtte lévőt (lévőket) is használtuk.

cím A cím az alábbi tartományokba eshet:

bitcím:	0...FF (bit cím)
programkód:	0...FFFF (code cím)
adat (direkt cím):	0...FF (data cím)
adat (indirekt cím):	0...FF (idata cím)
Külső adat cím:	0...FFFF (xdata cím)

tartomány Címtartomány olyan monitorparancsok estén, amelyeknél megadható a kezdő és a végcím. Elválasztásukra a szóköz vagy a vessző használható.

érték Memóriarekeszbe, regiszterbe vagy bitcímre beírandó adat értéke.
bit: 0 vagy 1
bájt: 0...FF

lépészám A végrehajtandó programlépések száma: 0...FFFF

töréspont A töréspont sorszáma: 0...9

töréscím A töréspont címe: 0...FFFF

regiszternév A regiszternevek az alábbiak:
RA Akkumulátor regiszter (0E0H)
RB B segédakkumulátor (0F0H)
R0...R7 általános regiszterek az aktuális regiszterbankban
PSW Program állapotszó (0D0H)
DPTR 16 bites adatcím-mutató (DPL 82H, DPH 83H)
SP Veremmutató (81H)
PC Utasítás címmutató (utasításszámláló)

A monitorprogram parancsai

A parancsok beírhatók kisbetűkkel vagy nagybetűkkel. A parancsok egyes változóit elválaszthatjuk vesszővel vagy szóközzel. A parancs betűjele után nem kötelező elválasztó karaktert írni. A parancs lezárására, végrehajtásának elindítására az <Enter> billentyű szolgál.

HELP A monitorparancsok listázása

; tetszőleges szöveg Megjegyzés beírása a monitorprogram nem értelmezi

BASIC A Basic interpreter elindítása a mikrokontrolleren. Kilépés EXIT paranccsal.

DISPLAY parancsok

A memóriatartalomnak a terminálképernyőre listázására szolgálnak. A parancsok **D** betűvel kezdődnek, második karakterük jelzi a listázandó memória típusát. Ha nem adunk meg kezdőcímet, a listázást annak a memóriarekesznek a tartalmával kezdi, aminek a címét a PC (utasításszámláló, **P**rogram **C**ounter) tartalmazza, és négy képernyősornyi listát kapunk. Ha megadjuk az első listázandó rekesz címét, de a végcímet nem, akkor négy képernyősornyi listát kapunk a megadott kezdőcímtől. Ha a végcímet is megadjuk, akkor folyamatos listát jelenít meg a megadott tartományról. A listázást a 2. oldalon megadott billentyűkombinációkkal vezérelhetjük

DC (kezdőcím (végcím)) **D**isplay **C**ommand memory: a programmemória tartalmának listázása

Példák:

```
dc 8000
C:8000: 02 80 3F 73 79 73 74 65 6D 20 A0 02 C3 D0 80 02 : ..?system  .CP..
C:8010: 00 00 FF 02 A9 CB D2 DF 43 87 80 75 98 5A 22 FF : ....)KR_C..u.Z".
C:8020: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF : ................
C:8030: 02 C4 19 11 36 E0 CA C5 83 CA C8 C5 82 C8 22 90 : .D..6`JE.JHE.H".
#dc 8000 800f
C:8000: 02 80 3F 73 79 73 74 65 6D 20 A0 02 C3 D0 80 02 : ..?system  .CP..
#dc8000 805f
C:8000: 02 80 3F 73 79 73 74 65 6D 20 A0 02 C3 D0 80 02 : ..?system  .CP..
C:8010: 00 00 FF 02 A9 CB D2 DF 43 87 80 75 98 5A 22 FF : ....)KR_C..u.Z".
C:8020: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF : ................
C:8030: 02 C4 19 11 36 E0 CA C5 83 CA C8 C5 82 C8 22 90 : .D..6`JE.JHE.H".
C:8040: 7E 2C E4 F5 23 F0 90 7F 42 F0 12 88 46 90 00 00 : ~,du#p..Bp..F...
C:8050: 93 F9 F4 F0 E4 93 F4 B5 01 B5 F0 75 81 07 11 16 : .ytpd.t5.5pu....
#_
```

DD (kezdőcím (végcím)) **D**isplay **D**ata memory: a belső direkt címezhető adatmemória tartalmának listázása.

DI (kezdőcím (végcím)) **D**isplay **I**ndirect data memory: a belső indirekt címezhető adatmemória tartalmának listázása.

DX (kezdőcím (végcím)) **D**isplay e**X**ternal data memory: a külső adatmemória tartalmának listázása.

(A DD, DI és DX parancsok a DC-hez hasonlóan működnek, ezért ezekhez példákat nem adunk meg.)

DB (kezdőcím (végcím)) **D**isplay **B**it addressable data memory: a belső bitenként címezhető adatmemória tartalmának listázása.

Példák:

```
C:8000: 02 80 3F 73 79 73 74 65 6D 20 A0 02 C3 D0 80 02 : ..?system  .CP..
C:8010: 00 00 FF 02 A9 CB D2 DF 43 87 80 75 98 5A 22 FF : ....)KR_C..u.Z".
C:8020: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF : ................
C:8030: 02 C4 19 11 36 E0 CA C5 83 CA C8 C5 82 C8 22 90 : .D..6`JE.JHE.H".
C:8040: 7E 2C E4 F5 23 F0 90 7F 42 F0 12 88 46 90 00 00 : ~,du#p..Bp..F...
C:8050: 93 F9 F4 F0 E4 93 F4 B5 01 B5 F0 75 81 07 11 16 : .ytpd.t5.5pu....
#
#
#db 90
B:90H.0 (90): 1 1 1 1 1 1 1 1
B:98H.0 (98): 0 0 1 1 1 0 1 0
B:A0H.0 (A0): 1 1 1 1 1 1 1 0
B:A8H.0 (A8): 0 0 0 0 0 0 0 0
#db 90 97
B:90H.0 (90): 1 1 1 1 1 1 1 1
#db 90 c7
B:90H.0 (90): 1 1 1 1 1 1 1 1
B:98H.0 (98): 0 0 1 1 1 0 1 0
B:A0H.0 (A0): 1 1 1 1 1 1 1 0
B:A8H.0 (A8): 0 0 0 0 0 0 0 0
B:B0H.0 (B0): 1 0 1 1 1 1 1 1
B:B8H.0 (B8): 0 0 0 0 0 0 0 0
B:C0H.0 (C0): 0 0 0 0 0 0 0 0
#
```

EDIT Editáló, azaz memóriatartalom szerkesztő parancsok

A memóriatartalom megváltoztatására szolgálnak. A parancsok **E** betűvel kezdődnek, második karakterük jelzi a memória típusát. A kezdőcím megadása kötelező, a parancs a képernyőn megjeleníti a megadott rekesz tartalmát. Ha szükséges a megváltoztatása begépeljük az új értéket majd megnyomjuk az Enter-t. (Amennyiben nem kívánjuk módosítani, nyomjuk meg az <Enter> billentyűt.) A megadott érték bekerült a memóriába és folytathatjuk a szerkesztést a következő rekesszel. Ha befejeztük a szerkesztést egy pont begépelésével térhetünk vissza a monitorprogram parancsfogadó üzemmódjához.

EC cím

Edit **C**ommand memory: a programmemória tartalmának folyamatos módosítása a megadott kezdőcímtől. A programmemória 8000H cím fölötti rekeszei csak olvashatóak, módosítani csak az ez alatti rekeszek tartalmát lehet, ami viszont megegyezik a külső adatmemória azonos című rekeszeivel. (l. jegyzet 13. oldal.)

Példák:

```
#ec 8000
C:8000: 02 33
ERROR *** NO CODE MEMORY AT ADDRESS: 8000
#dc 1000 100f
C:1000: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
#ec 1000
C:1000: 00 11
C:1001: 00 22
C:1002: 00 33
C:1003: 00 44
C:1004: 00 .
#dc 1000 100f
C:1000: 11 22 33 44 00 00 00 00 00 00 00 00 00 00 00 00 : ."3D............
#
```

ED cím

Edit **D**ata memory: a belső direkt címezhető adatmemória tartalmának folyamatos módosítása a megadott kezdőcímtől.

EI cím

Edit **I**ndirect data memory: a belső indirekt címezhető adatmemória tartalmának folyamatos módosítása a megadott kezdőcímtől.

EX cím

Edit e**X**ternal data memory: a külső adatmemória tartalmának folyamatos módosítása a megadott kezdőcímtől.

(Az ED, EI és EX parancsok a EC-hez hasonlóan működnek, ezért ezekhez példákat nem adunk meg.)

EB cím

Edit **B**it addressable data memory: a belső bitenként címezhető adatmemória tartalmának folyamatos módosítása a megadott kezdőcímtől.

Példa:

```
#db 90 97
B:90H.0 (90): 1 1 1 1 1 1 1 1
#eb 90
B:90H.0 (90): 1
B:90H.1 (91): 1 0
B:90H.2 (92): 1 0
B:90H.3 (93): 1
B:90H.4 (94): 1 0
B:90H.5 (95): 1 .
#db 90 97
B:90H.0 (90): 1 0 0 1 0 1 1 1
#_
```

FILL parancsok

Egy megadott memóriatartomány valamennyi rekeszének ugyanazzal az értékkel történő feltöltésére szolgálnak. A FILL után még be kell gépelni ötödik karakterként a memória típusát.

FILLC kezdőcím végcím érték — **F**ill **C**ommand memory: a programmemória valamennyi rekeszébe ugyanazt az értéket tölti.
Tartomány: cím 0...7FFF, érték 0...FF.

Példák:

```
#dc 1000
C:1000: 11 22 33 44 00 00 00 00 00 00 00 00 00 00 00 00 : ."3D............
C:1010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
C:1020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
C:1030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
#fillc 1000 101f 33
#dc1000
C:1000: 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 : 3333333333333333
C:1010: 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 : 3333333333333333
C:1020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
C:1030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
#fillc 1005 100b AA
#dc 1000
C:1000: 33 33 33 33 33 AA AA AA AA AA AA AA 33 33 33 33 : 33333*******3333
C:1010: 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 33 : 3333333333333333
C:1020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
C:1030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : ................
#
```

FILLD kezdőcím végcím érték — **F**ill **D**ata memory: a belső direkt címezhető adatmemória valamennyi rekeszébe ugyanazt az értéket tölti.
Tartomány: cím 0...FF, érték 0...FF.

FILLI kezdőcím végcím érték — **F**ill **I**ndirect data memory: a belső indirekt címezhető adat-memória valamennyi rekeszébe ugyanazt az értéket tölti.
Tartomány: cím 0...FF, érték 0...FF.

FILLX kezdőcím végcím érték — **F**ill e**X**ternal data memory: a külső adatmemória valamennyi rekeszébe ugyanazt az értéket tölti.
Tartomány: cím 0...FF, érték 0...FF.

(A FILLD, FILLI és FILLX parancsok a FILLC-hez hasonlóan működnek, ezért ezekhez példákat nem adunk meg.)

FILLB kezdőcím végcím érték — **F**ill **B**it adressable data memory: a belső bitenként címezhető adatmemória valamennyi rekeszébe ugyanazt az értéket tölti.
Tartomány: cím 0...FF, érték 0 vagy 1.

Példák:

```
#db f8 ff
B:F8H.0 (F8): 1 1 1 1 1 1 1 1
#fillb f8 ff 0
#db f8 ff
B:F8H.0 (F8): 0 0 0 0 0 0 0 0
#fillb fa fc 1
#db f8 ff
B:F8H.0 (F8): 0 0 1 1 1 0 0 0
#
```

X	A regiszterek tartalmának kiíratása
X regiszternév	A megadott nevű regiszter tartalmának módosítása

Példák:

```
#x
RA RB R0 R1 R2 R3 R4 R5 R6 R7    PSW     DPTR  SP   PC
00 00 00 00 0C 31 4F 01 00 8C  ---R0---  0000  07  0000
#x ra
D:E0: 00 22
#x
RA RB R0 R1 R2 R3 R4 R5 R6 R7    PSW     DPTR  SP   PC
22 00 00 00 0C 31 4F 01 00 8C  ---R0---  0000  07  0000
#x r7
D:07: 8C f4
#x
RA RB R0 R1 R2 R3 R4 R5 R6 R7    PSW     DPTR  SP   PC
22 00 00 00 0C 31 4F 01 00 F4  ---R0---  0000  07  0000
#x dptr
DPTR = 0000 33f7
#x
RA RB R0 R1 R2 R3 R4 R5 R6 R7    PSW     DPTR  SP   PC
22 00 00 00 0C 31 4F 01 00 F4  ---R0---  33F7  07  0000
#xpc
PC = 0000 100
#x
RA RB R0 R1 R2 R3 R4 R5 R6 R7    PSW     DPTR  SP   PC
22 00 00 00 0C 31 4F 01 00 F4  ---R0---  33F7  07  0100
#
```

ASSEMBLÁLÁS

Ez egy különleges memóriaszerkesztő parancs. Segítségével mnemonikus utasításokat vihetünk be a külső programmemóriába, de címkéket nem használhatunk.

A (cím)	**A**ssemble: programbevitel/módosítás mnemonikus alakban az egylépéses assembler használatával. Befejezése a pont begépelésével lehetséges.

Példa:

```
#a 100
0100: CLR    T2(97)              inc a
0101: SUBB   A,@R1               nop
0102: JB     F8,0102             nop
0103: MOV    R0,A                sjmp 100
0105: JNB    F8,0102             .
#_
```

DISASSEMBLÁLÁS

A memóriatartalmat mnemonikus utasításokra fordítja vissza.

U (start-cím (stop-cím)) — A memóriatartalom visszafordítása és listázása assembly szimbólumokkal (disassemblálás)

Példa:

```
#u 100
0100: INC    A
0101: NOP
0102: NOP
0103: SJMP   0100
0105: JNB    F8,0102
0108: CPL    T2(97)
010A: SJMP   0102
010C: MOV    R7,A
010D: MOV    R7,A
010E: MOV    R7,A
#u100 104
0100: INC    A
0101: NOP
0102: NOP
0103: SJMP   0100
#_
```

TÖRÉSPONTKEZELÉS

Töréspontos programfuttatással tesztelhetjük, hogy a programunk egy bizonyos utasításáig megfelelően működött-e. Így eldönthetjük, hogy a hiba előtte vagy utána van a programban. A töréspontkezelés részletes ismertetésére példán keresztül a mérési utasításban kerül sor.

BS cím — **B**reakpoint **S**et: töréspont előjegyzése a megadott címre. A monitor-program egy sorszámot rendel hozzá (0...9), amivel a későbbiekben hivatkozhatunk rá. Töréspontón a programfutás megszakad, a memória és a regiszterek tartalma megvizsgálható. (ld. 2. ábra)

BL — **B**reakpoint **L**ist: az előjegyzett töréspontok listázása. A monitor-program megadja a töréspont sorszámát, címét és azt, hogy engedélyezett vagy tiltott. (ld. 2. ábra)

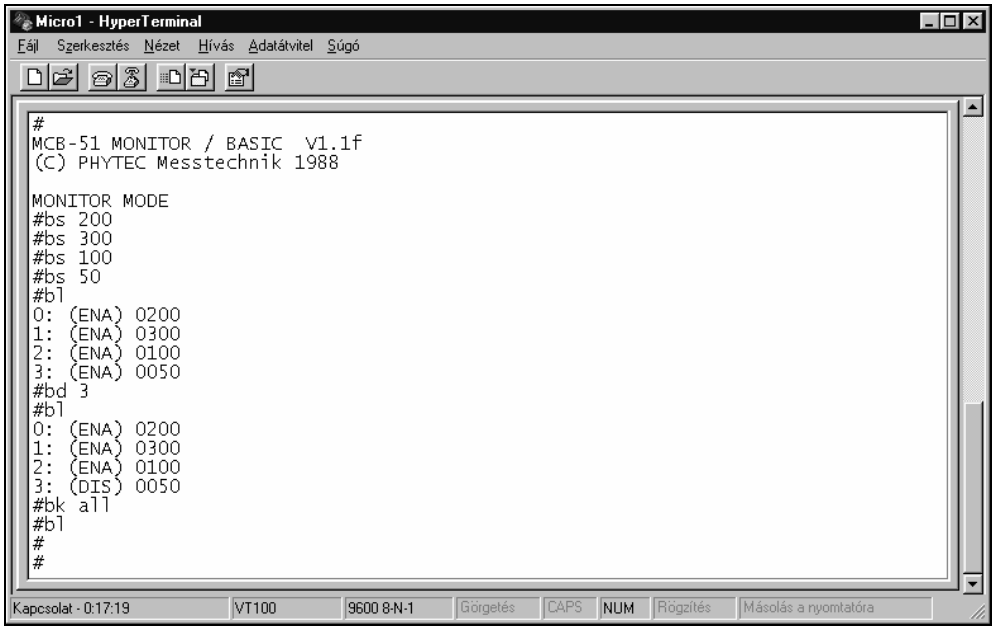
BK ALL — **B**reakpoint **K**ill ALL: valamennyi töréspontot törli az előjegyzésből.

BK bp (bp (...)) — **B**reakpoint **K**ill: a megadott sorszámú töréspontó(ka)t törli az előjegyzésből. (ld. 2. ábra)

BD bp (bp (...)) — **B**reakpoint **D**isable: a megadott sorszámú töréspontó(ka)t tiltja.

BE bp (bp (...)) — **B**reakpoint **E**nable: a megadott sorszámú töréspontó(ka)t engedélyezi.

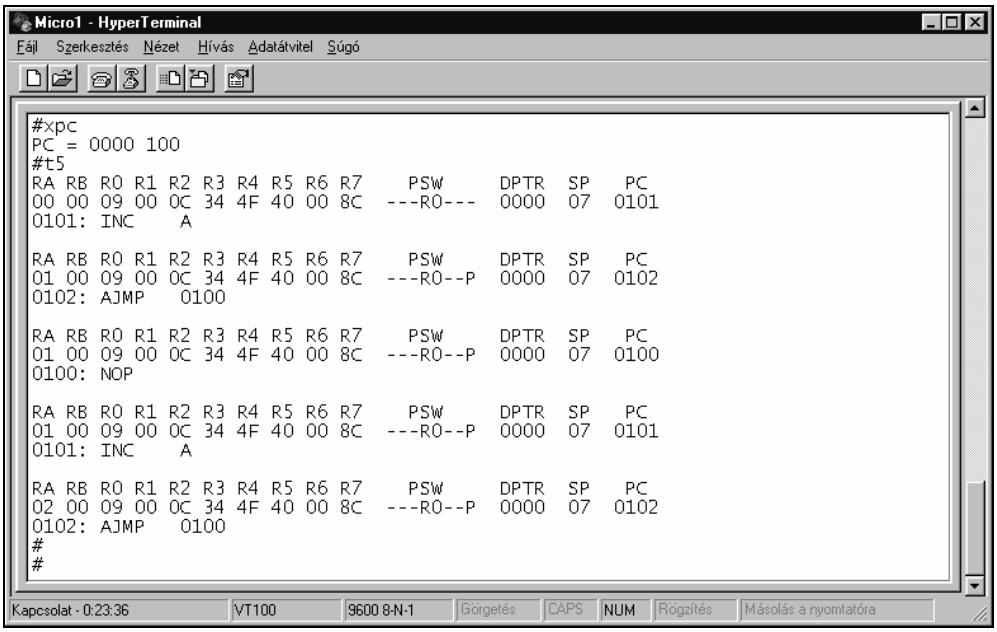
BE ALL — **B**reakpoint **E**nable ALL: valamennyi töréspontot engedélyezi.



2. ábra

G (kezdőcím (töréscím)) — **G**o: programfuttatás a megadott kezdőcímtől vagy a PC pillanatnyi értékétől. Ha törésponti címet is megadunk, a program futása az adott címen megszakad. Ez a töréspont nem kerül be az előjegyzésbe!

T (lépésszám) — **T**race: programvégrehajtás lépésenként nyomkövetéssel a PC pillanatnyi értékétől. Kiadása előtt a PC-t az xpc paranccsal be kell állítani. Megadhatjuk, hogy hány utasítást hajtson végre. Minden utasítás végrehajtása után listázza a regiszterek tartalmát. (ld. 3. ábra)



3. ábra

P (lépésszám) — **P**rocedure trace: ugyanaz mint az előző, de az eljárásokat (szubrutinokat) egy lépésnek tekinti.

S kezdőcím végcím	Memóriatartalom listázása Intel-hexa formátumban. A HyperTerminal segítségével fájlba másolható.

(A HyperTerminál segítségével bármilyen más a terminálképernyőn megjelenő információ is fájlba irányítható.)

Az IAR rendszer használata

A programfejlesztés menetét a jegyzet is tartalmazza. A szövegszerkesztővel (Windows XP Notepad, Windows Commander, vagy a fejlesztőrendszer E nevű editora) megírt assembly forrásprogramot a saját könyvtárunkban tároljuk tetszőleges (max. 8 karakter) programnévvel, de kötelezően **.S03** kiterjesztéssel, pl. **prog1.s03**. A fordítás, vagyis a 8051 Assembler futtatása az **ALL prog1** begépelésével lehetséges. A kiterjesztés (.s03) kiírása **tilos**! Ha a fordítás hibátlan, akkor a **prog1.r03**, **prog1.lst** és a **prog1.hex** fordítási termékek jönnek létre a könyvtárunkban. A **.hex** kiterjesztésű fájl letölthető a mikrokontroller memóriájába és futtatható.

Fordítási hiba esetén elindul a rendszer **E** nevű saját editora. Betöltődik a forrásprogram és külön ablakba a hibalista. A sorrendben legelső hiba pirossal kiemelve jelenik meg. A programhibák kijavítása után a programot elmentjük, majd Windows Commander parancssorában ismételten megkíséreljük a lefordítását.