

4.MÉRÉS

A terminálok használata a mikrogép programjaiból

1. A mérés célja:

A mikrogépen futó programokban a számítógép termináljai (klaviatúra, képernyő) is használhatók be-, illetve kimeneti perifériákként. A mérés során ismerkednek meg a terminálok használatának megoldásaival.

2. A szükséges ismeretek:

2.1. A mikrogép és a PC fizikai kapcsolata.

A mikrokontrolleres gyakorló készülék (mikrogép) és a személyi számítógép (PC) között az RS 232 szabvány szerinti, három vezetékes (null-modem) kommunikációs kapcsolat van.

A mikrogép **monitor** programja az inicializálást követően a **soros vonalat** „figyeli”. Amikor innen értelmezett parancs érkezik, akkor a megfelelő rutint végrehajtja, majd visszatér az alaphelyzetbe.

A PC-ben futó **terminál** program alaphelyzetben egyrészt a **soros vonalat**, másrészt a **klaviatúrát** „figyeli”. A klaviatúráról érkező jeleket a soros vonalon továbbítja (parancs a mikrogépnek.) A soros vonalon a mikrogép felül érkeznek jelek. Ezek a képernyőre jutó adatok, illetve kijelzést vezérlő kódok.

A leírtak szerint kialakított kapcsolat lehetőséget biztosít arra, hogy

- a monitor programból hardverteszteléseket végezhesünk,
- felhasználói programból pedig a mikrogép termináljaiként használhassuk a PC terminálokat.

2.2. Hardver tesztelés a monitor programból.

A monitor programban található rutinok egy része a klaviatúráról adott egy betűs parancsokkal aktiválható. A rutinok a mikrokontroller

- memória tartalmának kiírását, módosítását,
- a Portok és a további perifériák lekérdezését, esetleges módosítását

hajtják végre. A hívható parancsok a **H** (help) leütése után íródnak ki a képernyőre (lásd 2.mérés.)

2.3. A terminálok kezelése felhasználói programból.

A felhasználói, vagy saját programból a soros vonal közvetlen lekérdezésével olvashatunk klaviatúráról, illetve adatkivittel írhatunk a képernyőre.

Mind a kétirányú vonalkezelést – megfelelő státusz-bit **TI**, **RI** -) **lekérdezése** (polling) **segítségével** végezzük, mivel a soros vonal megszakítása a lépésenkénti üzemmódra van fenntartva.

A mikrokontrollerben lévő soros vonali illesztő (USART) **RS232** szabvány szerinti, és **full-duplex**. A vételnél, és az adásnál is az ún. **SBUF** soros vonali bufferbe kerül az adat (bár azonos a cím, de fizikailag két különböző SFR biztosítja az adás, és vétel egyidejű használatát.)

Az RS232 szabvány szerinti **aszinkron** adás – vétel bájtonként **önszinkronozó**. A szinkronozás azáltal valósul meg, hogy adásszünetben a vonalon **magas-szint** van, és minden bájtnak egy **0 – szintű START** bittel kezdődik. A vevőnek - a vonalon megjelenő - **1 – 0** szintváltás jelzi egy bájtnak a kezdetét. A zavarvédelmet biztosítja az, hogy a detektált jelváltás után - az **adási sebességből** adódó **bit-idő**n belül – három különböző időpontban is ellenőrzi a szintet. Amennyiben a három mért-érték közül legalább kettő 0, akkor tekinti az észlelt **1 – 0** szintváltást érvényes bájtnak a kezdetének.

Az adás-vétel sebességét **bit/sec** (baud) dimenzióval adjuk meg. Az adás-vétel szinkronozását **Baud – rate** generátor jele végzi. A gyakorló készülékben a mikrokontroller **T1** programozható **időzítő/számláló** egysége szolgáltatja a szinkronozó impulzusokat és **9600 Baud** - os sebességre van beállítva, melynek elvégzéséről minden egyes újraindítás (reset) után a gyakorlóban futó monitor-program gondoskodik.

7	6	5	4	3	2	1	0
SM0	SM1	SM2	REN	TB8	RB8	TI	RI

SM0:SM1	A soros port üzemmódját kijelölő bitek			
	<div>SM0SM1</div>			
	0	0	MODE-0	8 bites adat be- illetve kivitele (elő az LSB) az RxD ponton keresztül, A TxD ponton léptető impulzus sorozat lép ki, mely az oszcillátor frekvencia 1/12-e.
	0	1	MODE-1	START bit (0) + 8 adatbit (elő az LSB) + 1 db STOP bit kivitele a TxD ponton illetve vétele az RxD ponton. A baud-rate változtatható.
	1	0	MODE-2	START bit (0) + 8 adatbit (elő az LSB) + kiegészítő bit + 1 db STOP bit kivitele a TxD ponton illetve vétele az RxD ponton. A kiegészítő bit adáskor a TB8 bittel egyezik meg, vételkor pedig az RB8 bitbe olvasódik be. A baud-rate értéke az oszcillátor frekvencia 1/32-e vagy 1/64-e (a PCON regiszter SMOD bitjétől függően)
	1	1	MODE-3	Megegyezik a MODE-2-vel, de itt a baud-rate értéke változtatható.
SM2	Többprocesszoros kommunikációt (is) lehetővé tevő bit. MODE-0 esetén: az értéke kötelezően nulla! MODE-1 esetén: ha SM2=1, akkor az RI nem aktivizálódik, ha nem érkezett be érvényes STOP bit. MODE-2 és MODE-3 esetén: ha SM2=1, akkor az RI nem aktivizálódik, ha a vett kiegészítő bit (RB8) értéke nulla.			
REN	A soros vételt engedélyező (1) illetve tiltó (0) bit.			
TB8	MODE-2 és MODE-3 esetén a kiküldendő kiegészítő bit.			
RB8	A vett bitsorozat 9. bitje. MODE-0 esetén: nincs használva MODE-1 esetén: a vett STOP bit (ha SM2=0) MODE-2 és MODE-3 esetén: a vett kiegészítő bit			
TI	Adás megszakítás jelzőbit (programban törölendő!) MODE-0 esetén a 8 bit adásának végén 1-be billen MODE-1, MODE-2 és MODE-3 esetén a STOP bit adásának kezdetén 1-be billen.			
RI	Vétel megszakítás jelzőbit (programban törölendő!) MODE-0 esetén a 8 bit vételének kezdetén 1-be billen MODE-1, MODE-2 és MODE-3 esetén a STOP bit vételének félidejénél 1-be billen, kivéve ha SM2-vel mást írtunk elő.			

Saját berendezések fejlesztésénél (ahol nincsen monitor program) a programfejlesztő mérnöknek kell a soros port inicializálást elvégezni a felhasználói programban. A soros-port beállítása a mikrokontroller SCON (serial control, SFR cím: 89H) regiszterével lehetséges.

MODE-1 és MODE-3 esetén a baud-rate változtatható, értékét a T1 programozható időzítő/számláló 2-es (automatikus újratöltési) üzemmódjával állíthatjuk be.

$$\text{baud rate} = \frac{1}{32} \cdot \frac{1}{T1 \text{ periódusideje}} = \frac{1}{32} \cdot \frac{\text{oszcillátor frekvencia}}{12 \cdot [256 - TH1]} \quad (1)$$

A fenti képlet alapján a T1 újratöltési értéke a kívánt baud értéknek megfelelően:

$$TH1 = 256 - \frac{\text{oszcillátor frekvencia}}{12 \cdot 32 \cdot \text{baud rate}} \quad (2)$$

Számítási példa: ha az oszcillátor frekvencia 11.0592 MHz, és az óhajtott baud-rate 9600:

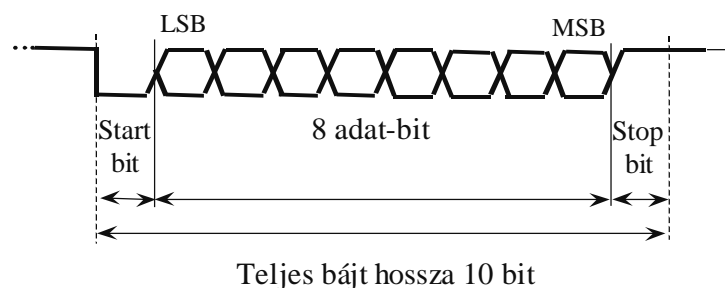
$$TH1 = 256 - \frac{11.0592 \cdot 10^6}{12 \cdot 32 \cdot 9600} = 256 - 3 = 253 = FD_H$$

Ha a fenti számítást „kerek” 12 MHz-es órajel estén is elvégezzük, akkor TH1=252.745 értéket kapunk, ami nyilvánvalóan nem írható be ténylegesen a TH1 regiszterbe. Ha viszont a TH1-be az ehhez legközelebb eső kerek értéket (253-at) töltünk be, akkor az (1) képlet szerint 10416.66 tényleges baud értéket kapjuk, ami 8.5%-os hibát jelent az óhajtott 9600 baud-hoz képest. Ez a magyarázat tehát az első látásra értelmetlen és szinte megjegyezhetetlen értékű 11.0592 MHz-es oszcillátor használatára. Csak így biztosítható a szabványos baud-rate.

A PCON regiszter 7-es bitje (SMOD bit) 1-be állításával a MODE-1, MODE-2 és MODE-3 üzemmódok esetén a beállított baud-rate megkétszereződik. Ha a fenti számítást elvégezzük 11.0592 MHz-es oszcillátor frekvenciával és 19200 óhajtott baud értékkel, akkor is tört számot kapunk a TH1-re. Azonban az SMOD bit segítségével beállítható a szabványos 19200 baud is, ha TH1-be a 9600 baud-nak megfelelő FD_H értéket töltjük be, és az SMOD bit 1-be állításával megkétszerezünk a baud értéket ($19200 = 2 \cdot 9600$).

Megjegyezzük, hogy amíg az eredeti 8051 mikrokontrollernél a változtatható baud-rate értéke csak a T1 időzítővel állítható be, addig ez más típusoknál (pl: 8052) a T2 számlálóval is megtehető. Egyes újabb variánsok saját, az időzítő/számláló egységektől független baud-rate generátorral rendelkeznek, amelyek már sok esetben kerek értékű (pl: 12 MHz-es) oszcillátor frekvencia mellett is képesek a szabványos baud-rate értékek előállítására.

A szabvány az aszinkron kommunikációnál többféle bájtfelépítést is megenged. A gyakorlatban beprogramozott bájtfelépítés az alábbi ábra szerinti.



Lássuk a konkrét programrészletet a soros port inicializálására (9600 baud, 11.0592 MHz):

```

MOV     SCON,#72H      ; MODE=1, SM2=1, REN=1, TI=1
ANL     TMOD,#0FH
ORL     TMOD,#20H      ; T1 = 2-es (automatikus újratöltési) üzemmód
MOV     TH1,#0FDH      ; T1 újratöltési értéke
; ORL    PCON,#80H      ; Ha ez a sor van, akkor 19200 baud, egyébként 9600
SETB    TR1            ; T1 indítása

```

2.3.1. Vétel (Receive) a soros vonalról

A USART vevő vonala az **RxD** jelű bemenet. Detektálja az 1-0 átmenetet, és ellenőrzi a Start-bit létét. Ezt követően egy léptetőregiszterbe lépteti a soros vonalon egymás után érkező biteket. Amikor az utolsó adatbit is a regiszterbe kerül, akkor írja 1-be az **RI** státusz bitet. Az **RI** jelzi a processzornak, hogy adat érkezett a soros vonalról, és azt ki kell olvasni az **SBUF** regiszterből. A jelzés után azonnal ki kell olvasni az adatot, mert ha a kiolvasás előtt újabb soros adat érkezik, akkor az felülírja a korábbi értékét. Az **RI**-bitet szoftverből törölni kell. A vétel programrészlete:

```

UJRA:   JNB     RI,UJRA      ; amíg RI nem 1 ismét lekérdezés
        CLR     RI          ; RI törlése
        MOV     A,SBUF      ; az új adat kiolvasása

```

2.3.2. Adás (Transmit) a soros vonalra

A USART adó-vonala a **TxD** jelű kimeneti pont. Az adó bufferbe (**SBUF**) írt adat automatikusan lép ki a **TxD** pontra a beprogramozott bájt-felépítés szerint, ésadási sebességgel. A **TI** jelű státusz-bit mindaddig **0**, amíg a teljes adat ki nem lépett a regiszterből. Az adási szándékot a programban a **TI** bit ellenőrzésével (polling) kell kezdeni. A kiviendő adatot csak akkor szabad az **SBUF**-ba írni, ha **TI** = 1. A vétel programrészlete:

```

        MOV     A,adat      ; a kiviendő adat behozatala
ISMET:  JNB     TI,ISMET    ; amíg TI nem 1 ismét lekérdezés
        CLR     TI
        MOV     SBUF,A      ; az új adat kiírása

```

Külön kell néhány szót ejteni egy szöveg képernyőre íratásáról. A forrásnyelvű fájlban a program-szegmensben, de a főprogramon kívül kell az egyes szövegeket deklarálni a következő formában:

```

SZOVEG1: DB     'Ez az első szöveg', 0
SZOVEG2: DB     'Ez a második szöveg', 0

```

A szöveg-deklarálásnál meg kell adni a szövegazonosítót (pl. **SZÖVEG1** stb.), majd a **DB** direktívát (define byte), és ezt követően a kiírandó szöveget ' –jelek között. A fordítóprogram az ' jelek közötti karakterek ASCII kódját helyezi le a memóriába. Így azok soros vonalon történő kivitele a PC képernyőjére az eredeti szöveget fogjuk látni. A sor utolsó tagja az **un.** szöveg termináló jel, amely itt a 0 szám, ami más érték is lehet. (A C nyelv a stringek után automatikusan a 0 termináló jelet teszi le).

A szöveg kiírása karakterenként történik egy olyan elől tesztelő ciklussal, amely a kiválasztott szöveg kezdetétől – a memóriából – egymás után beolvassa a karaktereket, és ha az nem egyenlő a 0 – val akkor kiírja a soros vonalra. A 0 észlelésekor kell befejezni a ciklust.

	MOV	R7,#0	; karakterszámláló törlése
	MOV	DPTR,#SZOVEG1	; a kiírandó szöveg első karakterének címe
UJRA:	MOV	A,R7	; következő karakter sorszáma
	MOVC	A,@A+DPTR	; a karakter ASCII kódjának beolvasása
	JZ	VEG	; a termináló karakter ?, ha igen akkor ;vége a kiíratásnak.
ISM:	JNB	TI,ISM	; lehet adatot kiadni a soros vonalra?
	CLR	TI	
	MOV	SBUF,A	
	INC	R7	
	JMP	UJRA	
VEG:	...		

2.3.3. A képernyő vezérlése

A fejlesztői környezetben használt TVTERM nevű terminál program biztosítja a képernyőkép ANSI szabvány szerinti beállításait.

Az un. *escape-szekvenciák* segítségével többek között mód van:

- a képernyő törlésére,
- a cursor-pozíció tetszőleges állításár
- sor törlésére,
- 16 színű szöveg, és háttér beállítására stb.

Az *escape-szekvencia* elnevezés arra utal, hogy a képernyőn történő kiíratás jellemzőit egy-egy – az **Esc** karakter kódjával (1BH, 27 dec.) kezdődő kódsorozat állítja be. A részletes leírást a terminál használatának leírása (TERM.DOC) állományból tanulható meg.

3. Házi feladatok

Írja meg a következő feladatok programjait külön forrásnyelvi fájlokba!

- 3.1. A képernyőre írja ki a saját nevét, címét és foglalkozását. Az egyes szövegrészek – soremelés, kocsni vissza - után az ENTER lenyomására jelenjenek meg.
- 3.2. Két abszolút értékű 8 bites számot olvasson be a klaviatúráról, és a számokkal végezze el a négy alapműveletet. Írja ki a képernyőre decimálisan a számokat és az eredményeket!
- 3.3. Alakítsa át az előző programot úgy, hogy az egyes alapműveletek előtt új adatbevitel legyen !

4. Mérési feladatok:

- 4.1. A házi feladatban megírt fájlokat másolja be saját alkönyvtárába. Végezze el a fordítást, szerkesztést és a hibátlan program HEX konverzióját.
- 4.2. Valós idejű környezetben ellenőrizze a feladatokat megvalósító programok működését.

5. Kérdések

- 5.1. Milyen fizikai kapcsolat van a mikrogép és a PC között?
- 5.2. Milyen módon történik az adatátvitel a mikrogép és a PC termináljai között?
- 5.3. Hogyan lehet a felhasználói programból kezelni a terminálokat?

5.4. Melyek a soros vonali illesztő státusz bitjei?

5.5. Hogyan lehet szöveget kiíratni a képernyőre a soros vonal közvetlen programozásával?