VDL programozási nyelv

és

felhasználói kézikönyv

TARTALOM

Bevezetés

A Sinclair ZX Spectrum 48k személyi számitógéphez készült VDL programozási nyelv implementáció kényelmes video-grafikai lehetőségeket nyujt a felhasználó számára.

A VDL segitségével látványos szöveges jellegü mozgó ábrák készithetők reklám, oktatási, tudományos és egyéb célra. Az elkészült müsorok egyrészt képmagnetofonon, másrészt a ZX Spectrum háttértárolóján (ZX Microdrive vagy kazettás magnetofon) tárolhatók.

A jó képminőség érdekében érdemes a megjelenitéshez szines video monitort használni.

A feliratok készitésénél 12 különböző jellegü és méretü betükészletből választhatunk. Egyes betütipusok "arányosak", vagyis az **I** keskenyebb, az **M** és **W** szélesebb az átlagosnál.

A VDL lényeges jellemzője a téglalap alaku ablakok definiálása. Egyidejüleg tetszőleges számu és méretü ablakot helyezhetünk a 22x32 mezőből álló képernyő bármely területére, egymást átlapolva, vagy egymás mellett. Nyolc keret közül választhatunk, melyekkel szegélyezhetjük az ablakokat. Az ablakon belüli területet négy irányban mozgathatjuk különböző sebességgel. A kiirandó szöveg szerkesztett formában kerül az ablakba, ez azt jelenti, hogy ha egy szó már nem fér el a sorban, automatikusan a következő sor elejére kerül.

A VDL programnyelv felépitése logikus és egyszerü, elsajátitása nem okoz különösebb nehézséget. A Spectrum BASIC ismerete előnyt jelent.

Programozás VDL nyelven

A VDL segitségével készült müsorok két file-t tartalmaznak: a kazettán található **cVDL** nevü gépi kódu részt és a felhasználó által VDL nyelven megirt programot. Programirás előtt töltsük be Microdrive kazettáról a **"run"** nevü file-t:

```
PRINT USR 0
RUN
```

Ha a beolvasás sikerült, rövid **BASIC** programot láthatunk; egyben a VDL implementáció is a számitógépben van:

```
9990 RANDOMIZE USR 63103: STOP
9991 CLEAR 40000:LOAD *"m";1;"cVDL"CODE : RUN
9992 SAVE *"m";1;"run" LINE 9991
9993 VERIFY *"m";1;"run"
```

A VDL nyelv utasitásait **BASIC REM** sorokba kell beirni. Egy sorba több utasitást is irhatunk, ekkor az egyes utasitásokat pontosvesszővel kell elválasztani. A VDL nyelv a strukturált programirást támogatja szubrutinok kényelmes használatával. Az utasitások egy betűből, vagy jelből állnak, amit adott esetben argumentumként további betük, vagy számok egészithetnek ki. Az igy felépitett utasitások elé számot irhatunk - az un. ismétlési tényezőt -, ami meghatározza, hogy az utasitást hányszor kell egymás után végrehajtani. Irjunk egy rövid programot:

```
10 REM TB;2=ZX Spectrum;5SU
```

A programot (ami mindig kell, hogy tartalmazza a 9990-9993 sorokat is!) **RUN** paranccsal futtathatjuk. A képernyő alján két sorban megjelenik a **'ZX Spectrum'** felirat, ami ezután **5** sorral feltolódik. Ez a ciklus a végnélkül ismétlődik. Leállitásához nyomjuk meg az **X** billentyüt.

A megirt programot Microdrive kazettán tárolhatjuk **GO TO 9992** utasitással. A **'run'** helyett tetszőleges file-nevet adhatunk, legfeljebb **10** karakter hosszuságig. Az igy tárolt program **LOAD *"m";1;"file-név"** parancsra betöltődik és azonnal elindul. A program futása közben az **X** billentyü megnyomására a program leáll, és visszatérünk szerkesztő üzemmódba. A listán a kurzor azt a **REM**-sort jelzi, ahol félbeszakitotuk a programot. A futó programot az **S** billentyü megnyomásával megállithatjuk, majd - az **S** és az **X** kivételével - bármelyik billentyüre tovább futtathatjuk. Az **S** és az **X** hatása csak az aktuális utasitás befejeztével érvényesül.

Ha a programunkba szintaktikai hiba (olyan utasitás, amit a VDL nem ismer fel) kerül, nem történik semmi végzetes, a VDL az ilyen utasitást egyszerüen figyelmen kivül hagyja, vagy helyes értékkel helyettesiti be. A program futtatáskor automatikusan tizesével ujrasorszámozódik. Igy nem jelent gondot programirás közben ujabb sorok beszurása. Ha nem akarunk élni ezzel a lehetőséggel, akkor **9990** sort helyettesitsük a következővel:

```
9990 RANDOMIZE USR 63149: STOP
```

Utasításkészlet

Az utasitások leirásánál az alábbi jelölöéseket használjuk:

b	betü az argumentumban
s	szám az argumentumban
(..)	a zárójeles argumentum elhagyható

Az utasitásokat kis és nagybetüvel egyaránt irhatjuk, jelentésük ugyanaz.

Ahol annak értelme van, az utasitás előtt szereplő szám meghatározza, hogy az utasitás hányszor kerüljön végrehajtásra. Ez a szám az ismétlési tényező, értéke **2** és **255** közötti lehet.

W(b)(ssss) - ablak definiálása

Pl. **Wa06120816**
W00240032
Wa

Az ablak helyzetét és méretét az *ssss* argumentum határozza meg. Az ablak azonositója a *b* argumentum, ami tetszőleges betü lehet, a kis és nagy betük önálló értelmezésével. Igy a programban létrehozhatunk egy legfeljebb **46** tételből álló ablak definiciós blokkot:

```
100 REM Wa1006160B
110 REM Wb05161216
120 REM Wc07150228
```

Az itt szereplő ablakokra a programban bárhol hivatkozhatunk rövid formában:

```
200 REM Wb;C17;F*;Wa;C73;F ;E7;3P
```

Atmeneti célra azonositó nélkül is definiálhatunk ablakot, erre azonban később nem tudunk hivatkozni:

```
300 REM W06120422;E1;C37;E6;F;3P
```

Az ablak méretet és helyzetét négy számmal kell meghatározni:

Waabbccdd ahol:

aa- az ablak legfelső sorának sorszáma a képernyőn
bb- az ablak mélysége
cc- az ablak legelső oszlopának sorszáma a képernyőn
dd- az ablak szélessége

Minden méretet jelölő szám kétjegyü, ha kisebb tiznél, nullával kell kiegésziteni.

A képernyő 24x32 karaktermező méretü, sorainak és oszlopainak számozása az alábbi:

```
0 ..................... 31
.............................
.............................
.............................
.............................
.............................
23 ..................... 31
```

A VDL aktuális ablàka betöltés után:

00240032

Ez a teljes képernyőt jelenti, mert a **0.** sor **0.** oszlopától **24** karaktermező mélységben és **32** karaktermező szélességben definiál ablakot.

Az aktuális ablak mindig a legutóbb definiált ablak. Az utasitások nagy része mindig az aktuális ablakra vonatkozik.

E(s) - ablak keretének meghatározása

Mindig az aktuális ablakra vonatkozik.

Az *s* a keret kódja, értéke **0** és **7** közötti lehet. A használható keretek a **2.** melléklet ábráin láthatók.

Definiáljunk egy ablakot:

```
10 REM Wa06120816
```

Az ablakba rajzoljunk másodpercenként uj keretet:

```
20 REM Wa;E1;P
30 REM Wa;E2;P
40 REM Wa;E3;P
50 REM Wa;E4;P
60 REM Wa;E5;P
70 REM Wa;E6;P
80 REM Wa;E7;P
90 REM Wa;E8;P
100 REM Wa;E9;P
```

A keret az ablak hasznos területét egy-egy mező szélességü sávval csökkenti. Ezalól kivétel, amikor az ablak olyan kicsi, hogy a keret már nem férne el; ekkor az utasitás nem kerül végrehajtásra.

Ha az E utasitást többször hajtjuk végre, több keretet is rajzolhatunk az ablakba, pl.:

```
3E4  vagy  E4;E2;E5
```

F(b) - ablak átszinezése, vagy kitöltése karakterrel

Az aktuális ablakra vonatkozik. A *b* paraméter tetszőleges karakter lehet, beleértve a grafikus jeleket is.

Az *F* utasitás a teljes ablakot átszinezi a legutóbb *C* utasitással megadott háttér és előtér szinre:

```
W00120016;C16;F;5P
```

Ha a *b* paraméter szerepel, akkor az átszinezéssel együtt a megadott karakterrel is kitölti az ablakot. Irjuk tele csillaggal a definiált ablakot:

```
W00120016;F*;5P
```

Az ablak gyors törlésének legegyszerübb módja, ha **SPACE** karakterrel töltjük ki:

```
W00120016;TB;=VDL;P;C66;F ;P
```

Az *F* utasitással olyan villogást is előállithatunk, ahol nem az előtér és a háttér szine váltakozik, hanem az előtér periódusosan beleolvad a háttérbe, mivel azonos szinü lesz:

```
10 REM W10031016;T9;=villogás;20 A
20 REM *A;C77;F;C70;F;R
```

Cs(s) - háttér és előtér szinének beállitása

Az utasitás az aktuális ablakra vonatkozik. Hatása nem közvetlen, csak az azt követő kiirás vagy kitöltés szinét állitja be. Lásd még: *=; >; F; C*

Az első paraméter a háttér szinét jelöli, a második - ha van - az előtér szinét. Ha az előtér szinét nem határozzuk meg, akkor az nem változik.

A szinek kódjai megegyeznek a Spectrum **BASIC** jelölésével:

0 - fekete
1 - kék
2 - vörös
3 - bibor
4 - zöld
5 - világoskék
6 - sárga
7 - fehér

Állisunk be vörös alapon fehér ablakot:

```
10 REM Wa00240032;C27;F*;3P
```

Most szinezzük át a hátteŕt zöldre:

```
20 REM Wa;C4;F;3P
```

A VDL betöltéskor fehér előtér és fekete háttér szint állit be.

Ab(b) - attributum beállitás

Az utasitás az aktuális ablakra vonatkozik. Hatása nem közvetlen, csak az azt követő kiirás vagy kitöltés attributumát állitja be. Lásd még: *=; >; F; C*

Vegyük sorra a lehetséges paramétereket:

AB - nagy fényerő (BASIC BRIGHT 1)
AD - kis fényerő (BASIC BRIGHT 0)
AF - villogás be (BASIC FLASH 1)
AS - villogás ki (BASIC FLASH 0)

Egy utasitásban egyszerre két jellemzőt is beállithatunk.

A VDL betöltésekor az *ABS* állapot áll be.

Bs - BORDER szin meghatározás

A Spectrum **BORDER** területére vonatkozik.

Az *s* a szinek kódja, értéke **0 - 7** közötti. A szinek kódjai megegyeznek a Spectrum **BASIC** jelölésével.

Villogtassuk a **BORDER**-t három szinnel egy másodpercenként:

```
10 REM B2;P;B7;P;B4;P
```

A VDL betöltéskor bibor **BORDER** szint állit be.

P - egy másodperces várakozás

A program futását egy másodpercre leállitja. Hosszabb késleltetés eléréséhez az utasiást többször kell végrehajtani, pl. 5 másodpercig tartó várakozást igy valósithatunk meg:

```
10 REM B7;5P;B0;5P
```

A *P* utasitásnak nincs argumentuma.

=(szöveg) - szöveg kiíratása

Az aktuális ablak legalsó sorába írja az = jel utáni szöveget. Ha szöveg nem fér el egy sorban, akkor az első sor felcsuszik az ablakban és a következő sor alá kerül. Ha szöveg hosszabb, mint amennyi az ablakban elfér, a szöveg eleje kiuszik az ablakból. Uj sorba kerül minden olyan szó, amely már nem fér el az adott sor végén. A kiirt szöveg az ablak bal széléhez igazodik. Ha a szövegben egymás után kettő vagy több **SPACE** karakter szerepel, azt a VDL automatikusan egy szóközre csökkenti. Amennyiben mégis nagyobb szóközt szeretnénk hagyni, tegyünk a szövegbe **TRUE VIDEO**, vagy **INV.VIDEO** karaktereket.

Minden = jel uj sorban kezdi a kiiratás. A szöveg az aktuális betütipussal jelenik meg. (Lásd a *T* utasitást)

```
10 REM W10030016;C17;F ;
20 REM =VDL
30 REM =feliratozó program;5P
```

A kiratő utasitás elé ismétlési tényezőt is tehetünk. Ekkor a szöveg többször irodik az ablakba, midig egymás alá:

```
10 REM W00240032
20 REM 10=ZX Spectrum
```

A kiirandó szöveg ékezetes betüket is tartalmazhat. Ezeket a ZX Spectrum grafikus üzemmódjában érhetjük el, tehát **CAPS SHIFT** és 9 lenyomása után. A grafikus üzemmódból szintén a **CAPS SHIFT + 9** lenyomásával térhetünk vissza.

Az ékezetes karakterek elhelyezkedése a billentyüzeten az alábbi:

```
A S  E R  D  O P  K L  N M  U I  H J  C B   F G
Á á  É é  i  O ö  O ő  O ó  U ü  U ü  U u   : ;
```

Szintén grafikus üzemmódban irható a **:** és a **;** irásjel, mivel ezek eredetije **BASIC**, illetve **VDL** utasitást elválasztó jel.

A grafikus üzemmódban irt T betü apró sakktábla minta formájában jelenik meg, lehetőséget nyujtva arra, hogy a nyolc alapszintől eltérő szineket hozzunk létre:

```
10 REM W00240032;C14;FT;5P
```

<(szöveg) - szöveg kiiratása jobbról beusztatva

Az aktuális ablak legalsó sorába irja a < jel utáni szöveget, ugy, hogy az jobbról uszik be az ablakba. A kiirt szövegek az ablak jobb széléhez igazodnak. Ha szöveg hosszabb, mint amennyi egy sorban elfér, a szöveg eleje kiuszik az ablak bal oldalán. Igy mozgó fényujságszerü feliratozást készithetünk. Ha a szövegben egymás után kettő vagy több **SPACE** karakter szerepel, azt a VDL automatikusan egy szóközre csökkenti. Amennyiben mégis nagyobb szóközt szeretnénk hagyni, tegyünk a szövegbe **TRUE VIDEO,** vagy **INV.VIDEO** karaktereket (**CAPS SHIFT 3** vagy **4**).

Minden < jel uj sorban kezdi a kiiratást. A szöveg az aktuális betütipussal jelenik meg. (Lásd a *T* utasitást)

```
10 W10030016;C17;F ;
20 REM <VDL
30 REM <feliratozó program;5P
```

A kirató utasitás elé ismétlési tényezőt is tehetünk. Ekkor a szöveg többször iródik az ablakba, midig egymás alá:

```
10 REM W00240032
20 REM 10<ZX Spectrum
```

A kiirandó szöveg ékezetes betüket is tartalmazhat. Ezeket a ZX Spectrum grafikus üzemmódjában érhetjük el, tehát **CAPS SHIFT** és **9** lenyomása után. A grafikus üzemmódól szintén a **CAPS SHIFT + 9** lenyomásával térhetünk vissza.

Szintén grafikus üzemmódban irható a **:** és a **;** irásjel, mivel ezek eredetije **BASIC,** illetve **VDL** utasitást elválasztó jel.

A grafikus üzemmódban irt **T** betü apró sakktábla minta formájában jelenik meg.

S(b) - ablak tartalmának mozgatása

Az utasitás az aktuális ablak tartalmát mozditja el a lehetséges négy irány valmelyikében egy karakternezőnyi távolságra.

Ha a *b* paraméter nincs megadva, az elmozdulás felfelé történik, egyébként az alábbiak szerint:

SR - jobbra
SL - balra
SD - le
SU - fel

Az ismétlési tényezővel több karaktermezőnyi elmozditásra adhatunk utasitást:

```
10 REM W10101010
20 REM 2=elmozditás
30 REM 4SU;5SR;2SD;3SL;5S
```

T(s) - betütipus kiválasztás

A VDL **12** betütipusból álló karakterkészlettel rendelkezik, ezeket használhatjuk az **=** és a **<** kiirató utasitások esetében. Egyes betütipusok csak nagy-, más tipusok csak kisbetüket tartalmaznak. Ha a programban ilyen betütipus használatakor a kis-, vagy nagybetü helyett a nem létezővel irjuk a kiiratandó szöveget, a VDL automatikusan behelyettesiti a megfelelőt. Ez a szabály nem vonatkozik az ékezetes betükre. Az egyes betütipusok karaktereinek mérete eltérő, létezik **1x1, 2x1, 2x2** és **3x2** karaktermező méretü. Egy-két betütipus un. arányos méretü, azaz nem egyforma széles helyet foglalnak a sovány és széles karakterek.

A 12 betütipus az **1.** melléklet ábráin látható.

A betütipus *s* paramétere **0** - **9** közötti szám lehet, illetve megengedett az **A** és **B** is. Ha a paraméter ettől eltérő, vagy nincs megadva, a VDL a Spectrum alapkészletét használja.

Szintén a Spectrum betükészletének karaktere jelenik meg a kiiratáskor, ha a kiirandó karakter nem szerepel az aktuális készletben. A VDL betöltéskor a **T3** betükészletet állitja be.

Vegyük sorra a betütipusokat:

T0 - a ZX Spectrum karakterkészlete kiegészitve az ékezetes betükkel. Mérete: **1x1**

T1 - **1x1** méretü Serif, csak nagybetü.

T2 - **2x2** méretü Serif, arányos, csak nagybetü.

T3 - **2x2** méretü dekorativ, modern tipus, csak nagybetü.

T4 - **1x1** méretü Magnetic Ink tipus, csak nagybetü.

T5 - **1x1** méretü Bold tipus, csak nagybetü.

T6 - **1x2** méretü Sans Serif, teljes készlet.

T7 - **1x2** méretü Elegant, teljes készlet.

T8 - **1x1** méretü Bold tipus, csak nagybetü.

T9 - **1x3** méretü Sans Serif, arányos, csak kisbetü.

TA - **3x2** méretü Serif, arányos, csak nagybetü.

TB - **3x2** méretü árnyékolt, arányos, csak nagybetü.

= - végrehajtási sebesség beállitása

Az egyes utasitások végrehajtása közötti időkésleltetés állitható be vele. értéke **20 msec,** ennek többszöröse állitható be az ismétlési tényezővel. Az utasitás hatása folyamatosan tart, mig ujabb sebesség beállitó utasitással azt felül nem irjuk. A VDL betöltéskor **10=** értéket, azaz **200 msec**-ot állit be.

b - szubrutinhivás

Ez az utasitás a BASIC **GOSUB** megfelelője. Hatására a VDL megkeresi a programban a ***b*** kezdetü szubrutint, azt végrehajtja, majd folytatja a programot ott, ahol félbeszakitotta a szubrutinhivás miatt.

A *b* paraméter - szubrutinazonositó - egy betü lehet, a kis- és nagybetüs azonositók önálló jelentésüek.

Ha a hivatkozott azonositóju szubrutin nincs a programban, a VDL a szubrutinhivást figyelmen kivül hagyja.

Ha a megnevezett azonositóval több szubrutin is található a programban - ez programozási hiba! - akkor a sorszám szerinti első szubrutin az érvényes.

A meghivott szubrutinok további szubrutinokat is hivhatnak, legfeljebb **10**-szeres hivási mélységig.

**b* - szubrutincimke

Ez az utasitás azonositja az egyes szubrutinokat - ezek alapján lehet hivni azokat. A cimke több betübői is állhat, ám a VDL csak az első betüt veszi figyelembe.

R -szubrutin vége

Ez az utasitás a BASIC **RETURN** megfelelője, paramétere nincs.

Hatására a VDL befejezi a szubrutin futtatását és folytatja a programot a szubrutinhivást követő utasitástól.

Kivételes esetben előfordulhat, hogy az *R* utasitásra nem szubrutinnivás következtében fut a program. Ennek a programozási megoldásnak egy esetben van értelme: ha a legfelső szubrutinhivási szinten használjuk. Ebben az esetben az *R* utasitásnak nincs visszatérési cime, ezért a program az első sortól ujraindul. A szemléltető programokban megfigyelnetjük, hogy a programok legmagasabb szintü moduljai ezzel a módszerrel készültek.

Feltétlenül ügyeljünk arra, hogy az *R* utasitást ne felejtsük le a szubrutinok végéről, mert igy a program a szubrutinból ráfut a következő programrészre, felboritva a program futását.

Programozási tanácsok

A VDL nyelv felépitése támogatja a moduláris programozást. A program első sora a fő vezérlő legyen:

```
10 REM  A;R;*A;.....;R
```

A szubrutinhivás nélkül használt *R* hatására a ujra és ujra ismétlődik, végtelen ciklusban. A legmagasabb szintü modul az **A** cimkéjü. Ez a kipontozott részen további alacsonyabb szintü modulokat - szubrutinokat - hiv, azok pedig továbbiakat, stb.

Az ilyen módszerü programozás előnye, hogy az egyes szubrutinokat menetközben könnyen ellenőrizni tudjuk. A **Z** cimkéjü szubrutint a **10.** sor ilyen - átmeneti - kijavitásával tudjuk ellenőrizni:

```
10 REM  Z;R;*A;.....;R
```

A program elején alakitsunk ki egy ablak-definiáló modult. Az itt definiált ablakokra bárnol hivatkozhatunk a programban. Ennek több előnye is van:

- A hivatkozás rövidebb, mint a teljes definiálálás.
- Az ablak változtatása esetén a módositást csak egy helyen kell elvégezni.
- Mivel a definiáló modult könnyü áttekinteni, elkerülhető a téves, azonos nevü definiálálás.

A müsor egyes képeiről készithetünk nyomtatást papirra. Javitsuk ki a **9990** sort:

```
9990 RANDOMIZE USR 63103:COPY
```

A müsort az **X** billentyüvel leállitva a nyomtatás elkészül.

VDL utasitáskészlet

Utasitás	Hatása	Paraméterek
A	Attributum beállitás	B világos D normál F villogó S álló
B	BORDER beállitás	0-7 szinkód
C	szin beállitás	0-7 háttérszin 0-7 előtérszin
E	keret beállitás	1-9 keretkód
F	ablak kitöltése, átszinezése	a kitöltendő karakter vagy semmi
P	1 másodpercces szünet	nincs paramétere
R	visszatérés szubrutinból	nincs paramétere
S	ablak tartalmának mozgatása	U fel D le R jobbra L balra ha nincs paraméter, akkor fel
T	betütipus beállitása	0-9 és A-B Ha nincs paraméter, akkor a Spectrum karakterkészletét állitja be
W	ablak meghatározása	Wraabbccdd, ahol: r : egybetüs ablak-azonositó aa: kezdő sor bb: sorok száma cc: kezdő oszlop dd: oszlopok száma
=	szöveg kiiratása balra igazitással	kiirandó szöveg
<	szöveg kiiratása jobbra igazitással	kiirandó szöveg
↑ b	szubrutinhivás	szubrutinazonositó
*	szubrutincimke	szubrutinazonositó
=	késleltetés beállitás	nincs paramétere, de az ismétlési tényezővel hatása állitható

Irodalom

1. Tarján György: ZX Spectrum Bevezető és BASIC pogramozási kézikönyv, 1984 Ipari Informatikai Központ

2. Vidákovics Attila: ZX Interface 1 és ZX Microdrive, 1984 Ipari Informatikai Központ