

Többcsatornás hanggeneráló programcsomag

VIDEOTON TV COMPUTER-re

játékfejlesztőknek

© 2023-24 by Béla Szalontai (Fast & Force)
Release 1.0 (2024.02.21)

License: assembly forráskódokra: MIT (<https://opensource.org/license/mit/>)
SIDDump.exe: <https://github.com/cadaver/siddump/blob/master/readme.txt>
SIDDumpTVCGUI.exe: © 2024 by Károly Kiss

Köszönet: Kiss Károly (SIDDumpTVCGUI fejlesztése, tesztelés, zenék, inspiráció)
Noel Persa (assembly kód optimalizálás, ötletek)
Bertók Zsolt (assembly kód hegyek a zenelejátszók demózásához)
Orosz Olivér (A Sötét Kastély játék publikus forráskódjáért)
Jarl Frey (A Treasure Island játék publikus forráskódjáért)
Sáránszky Mihály (Knight Rider főcímkép)
Major Tamás (Tvc Developer Tool)

Valódi 2 és 3 csatornás hanggenerátorok és lejátszók játékok főcímmzenéjéhez, játékok alatti többcsatornás zenéléshez, hangeffektek kiadásához, akár CRTC szinkronizálással. Játék íráshoz szabadon felhasználható Z80 assembly forráskódok.

Tartalomjegyzék

Tartalom

Tartalomjegyzék	2
Az eddig bemutatott demók.....	3
Demó 1	3
Demó 2	3
Demó 3	3
Demó 4	3
Zenei formátum:.....	3
Az alábbi hangok használhatók	3
Vezérlő kódok	4
Vezérlő kódok listája.....	5
Bináris formátum	6
ZX7	6
SID konverter	7
SIDDumpTVC.exe	8
Az ADSR számítás algoritmus.....	8
SIDDumpTVC.exe konverter használata	9
A legfontosabb paraméterek.....	9
TVC specifikus paraméterek	9
Parancssori példák	10
SIDDumpTVCGUI.exe.....	10
Lejátszó programok	11
Lehetőségek és limitációk.....	11
Általános limitáció, ha fut a zenelejátszó megszakítás	11
2 csatornás lejátszó 4340Hz-es, CRCT szinkronos.....	11
3 csatornás lejátszó 5008 Hz-es, CRTC szinkronos	11
3 csatornás lejátszó 8491 Hz-es, csak négyszögjel, ADSR	12
3 csatornás lejátszó 8491 Hz-es, különböző hullámformák	12
Integráció.....	13
3ch_adsr.dt80asm	13
Beállítások	16
3ch_wavetable.dt80asm.....	17
2ch-fx-crtc.dt80asm.....	19
A lejátszó felépítése	19

Az eddig bemutatott demók

Demó 1

Shoot em up Demo (2 csatornás zene játék közben + nem blokkoló hangeffektek, CRTC szinkronizálással, 25FPS)

<https://www.facebook.com/groups/videotontvcomputer/posts/7231654600249510/>

Demó 2

Treasure Island (3 csatornás főcímmzene + többcsatornás hangeffektek játék közben)

<https://www.facebook.com/groups/videotontvcomputer/posts/7251559191592384/>

Demó 3

Knight Rider (3 csatornás főcímmzene különböző BPM-ekkel + ZX7 képkítömörítés 3 csatornás zenelejátszás közben)

<https://www.facebook.com/groups/videotontvcomputer/posts/7263235263758110/>

<https://www.facebook.com/groups/videotontvcomputer/posts/7263305987084371/>

Demó 4

Sötét Kastély 2024 (3 csatornás zene lejátszás játék közben)

<https://www.facebook.com/groups/videotontvcomputer/posts/7288343484580621/>

Zenei formátum:

Mindegyik lejátszóhoz ugyanaz a zenei formátum tartozik, de nem mindegyik lejátszó ismeri az összes vezérlő kódot

A zene formátum univerzális, bárki akár kézzel is írhat zenét, mivel a zenei hangok neveit lehet használni

Az alábbi hangok használhatók

C1, CISZ1, D1, DISZ1, E1, F1, FISZ1, G1, GISZ1, A1, AISZ1, B1

C2, CISZ2, D2, DISZ2, E2, F2, FISZ2, G2, GISZ2, A2, AISZ2, B2

C3, CISZ3, D3, DISZ3, E3, F3, FISZ3, G3, GISZ3, A3, AISZ3, B3

C4, CISZ4, D4, DISZ4, E4, F4, FISZ4, G4, GISZ4, A4, AISZ4, B4

C5, CISZ5, D5, DISZ5, E5, F5, FISZ5, G5, GISZ5, A5, AISZ5, B5

C6, CISZ6, D6, DISZ6, E6, F6, FISZ6, G6, GISZ6, A6, AISZ6, B6

C7, CISZ7, D7, DISZ7, E7, F7, FISZ7, G7, GISZ7, A7, AISZ7, B7

C8*

(*Bizonyos lejátszók csak C7-ig tudják a hangokat lásd az adott lejátszónál feltüntetve.)

Az assembly programban szövegesen DB definíciókkal lehet leírni a zenét.

Példa:

2 csatornás lejátszóknál:

```
DB C2,DISZ3
```

```
DB C3,DISZ4
```

3 csatornás lejátszóknál:

```
DB C2,DISZ3,C3
```

```
DB C3,DISZ4,C4
```

Vezérlő kódok

Nemcsak zenei hangok, hanem vezérlő kódok is szövegesen írhatók a zenébe. Az egyes lejátszókhöz használható vezérlő kódokat a lejátszónál részletezzük.

A **MUTE**, **SKIP** ÉS **FREQINCxx** valamint **FREQDECxx** vezérlő kódok a zenei hangok **HELYÉN** állnak

Példa:

```
DB C2,DISZ3,C3
```

```
DB SKIP,DISZ4,MUTE
```

Az **ADSRxx** és **WAVExx** vezérlő kódok a zenei hangok **ELŐTT** áll(hat)nak.

Példa:

```
DB ADSR0F,D1,SKIP,SKIP
```

```
DB DISZ5,SKIP,SKIP
```

A zene végén kötelezően egy **LOOP** kódnak kell állnia!

Példa:

```
DB D1,SKIP,SKIP
```

```
DB DISZ5,SKIP,SKIP
```

```
DB LOOP
```

Vezérlő kódok listája

- **MUTE** Csatorna némítása
- **SKIP** Csatorna kihagyása (előző hang szól)
- **FRQINC00 .. FRQINC0F** Frekvencia csúsztatás/vibrato fel 1-15 (előző hang szól, annak a frekvenciája változik)
- **FRQDEC00 .. FRQDEC0F** Frekvencia csúsztatás/vibrato le 1-15 (előző hang szól, annak a frekvenciája változik)
- **ADSR00** Hang lecsengése ~ 7 ms alatt
- **ADSR01** Hang lecsengése ~24 ms alatt
- **ADSR02** Hang lecsengése ~49 ms alatt
- **ADSR03** Hang lecsengése ~74 ms alatt
- **ADSR04** Hang lecsengése ~116 ms alatt
- **ADSR05** Hang lecsengése ~169 ms alatt
- **ADSR06** Hang lecsengése ~204 ms alatt
- **ADSR07** Hang lecsengése ~240 ms alatt
- **ADSR08** Hang lecsengése ~300 ms alatt
- **ADSR09** Hang lecsengése ~752 ms alatt
- **ADSR0A** Hang lecsengése ~1501 ms alatt
- **ADSR0B** Hang lecsengése ~2402 ms alatt
- **ADSR0C** Hang lecsengése ~3003 ms alatt
- **ADSR0D** Hang lecsengése ~9001 ms alatt
- **ADSR0E** Hang lecsengése ~15000 ms alatt
- **ADSR0F** Hang lecsengése ~24001 ms alatt
- **WAVE01** Négyszög hullámforma (csak a megfelelő lejátszóval)
- **WAVE02** Háromszög hullámforma (csak a megfelelő lejátszóval)
- **WAVE03** Fűrész hullámforma (csak a megfelelő lejátszóval)
- **WAVE04** Zaj hullámforma (csak a megfelelő lejátszóval)
- **LOOP** Zene ismétlése (kötelező a zene végére)

Példa: (Knight Rider Dynamic első néhány sora)

```
DB  ADSR0F,D1,SKIP,SKIP
DB  DISZ5,SKIP,SKIP
DB  SKIP,SKIP,SKIP
DB  FISZ1,SKIP,DISZ4
DB  FISZ2,SKIP,CISZ4
DB  FRQINC01,SKIP,FRQDEC01
DB  SKIP,SKIP,FRQDEC01
DB  SKIP,SKIP,FRQDEC01
DB  SKIP,SKIP,FRQINC02
DB  FISZ1,SKIP,FRQINC02
...
```

Bináris formátum

természetesen a zene fájl azaz a DB definíciók számok, amiket bináris formátumba is átkonvertálhatunk és az assembly kódba belinkelhetjük.

Példa:

```
MUSIC_DATA  INCBIN    "music.bin"
```

ZX7

A ZX7 tömörítő igen hatékonyan be tudja tömöríteni a zenei bináris fájlunkat. Természetesen lejátszás előtt ki kell tömöríteni egy adott memória területre

Példa:

```
LD      HL,BIN_MUSIC_DATA    ; HL = tömörített zene címe
LD      DE,MUSIC_DATA        ; DE = kitömörítés címe (U3)
CALL    ZX7_DECOMPRESS       ; zene kitömörítése
...
...
MUSIC_DATA      equ      $C000
BIN_MUSIC_DATA  incbin    "music.bin.zx7"
```

SID konverter

A Zenei formátumhoz írtam egy SID konvertert (**siddumptvc.exe**). Lásd későbbi fejezetben a részletes bemutatását.

Miért SID?

- több mint **58000** SID zene elérhető azonnal
 - <https://deepsid.chordian.net/>
 - <https://www.hvsc.c64.org/downloads>
- egylépéses konvertálás parancssorból a TVC-s formátumra 2 vagy 3 csatornával, extrákkal (siddumptvc.exe)
- SID tracker zeneszerkesztők elérhetőek windows-hoz is
 - <https://chordian.net/c64editors.htm>
 - <https://blog.chordian.net/sf2/>
 - windows-hoz ajánlott: GoatTracker 2.76
(<https://sourceforge.net/projects/goattracker2/files/GoatTracker%202/2.76/>)

Akkor ez egy SID formátum/lejátszó?

NEM SID LEJÁTSZÓ! Ez egy univerzális formátum/lejátszó, ami SID-ből konvertált zenét **is** SID hangzáshoz közel hasonlóan lejátszik.

Miben különbözik a már létező TVC-s szoftveres SID lejátszótól?

- a TVC-s SID lejátszó közel 100%-ban adja vissza a SID zenék hangzását
- a TVC-s SID lejátszó program és a generált hullámforma táblák együtt közel 16 Kbyte méretet foglalnak + a zene adat, együtt kb. 20Kbyte
- ez a lejátszó 1-2 kbyte méretű + a zenei adatok, így alkalmas minőségi főcím vagy játék közbeni zenék lejátszására.
- ez a lejátszó nem ugyanúgy játssza le a SID zenéket, ahogy azok eredetileg megszólalnak. Vagy csak hullámformák vannak, vagy csak ADSR, és az ADSR-ből is csak a **RELEASE** lett megvalósítva. Nincsenek filterek stb..

SIDDumpTVC.exe

Az eredeti SIDDump1.08 programot írtam át.

Forrás: <https://github.com/cadaver/siddump>

License: <https://github.com/cadaver/siddump/blob/master/readme.txt>

Az átírt program .exe-ként futtatható windows-os gépeken, de a C nyelvű forrást is mellékelem.

Segítségével parancssorból, egy lépésben konvertálhatjuk a C64-es SID fájlokat a zenelejátszó assembly kódba beilleszthető TEXT fájlba.

Lehetőség van a kezdő FRAME megadására, illetve a hosszt mindig érdemes megadni.

Alapértelmezetten a kezdő FRAME a zene eleje, a hossz pedig 60 másodperc.

Mivel a SID nem egy zenei formátum, hanem valójában egy C64-es gépi kódú program, így a konverter szimulálja a C64-et, annak gépi kódját futtatja, abból kinyeri a zenei adatokat és így állítja elő a szöveges kimenetet. Olyan SID fájlknál, amelyek digitalizált hangokat tartalmaznak, előfordulhat, hogy hibaüzenetet kapunk. Ilyen esetben próbálkozzunk átugrani FRAME-eket.

A létrehozott szövegfájl tartalmazza a zenei hangokat, a frekvencia csúsztatásokat és az ADSR információk változását. Nem minden hang mellé teszi oda az ADSR információt, csak akkor, ha változik. Ha egy ütemben nem változik a hang, akkor oda **SKIP**-et ír a szövegfájlba.

A Frekvencia csúszások dump-olása nem okoz többlet memória használatot, mert a vezérlő kódok az amúgy is meglévő **SKIP**-ek helyére kerülnek

Az ADSR számítás algoritmusa

Összeadja az Attack a Decay és a Release SID-es idő értékeit és abból keresi ki a megfelelő Release értéket 00.0F-ig. Csak lecsengés van, de a felfutás és kitartás ideje is beszámításra kerül.

Az ADSR vezérlő kód nem minden rekordba kerül bele, csak akkor íródik ki, ha amúgy is változik, így nem okoz jelentős többletet a hangfájl hosszában. Ha minden hanghoz hozzáírnánk, akkor duplázódna a zene adat mérete, így csak pár százalékkal nő meg.

Ha kézzel írjuk a zenét, vagy máshonnan konvertáltuk, akkor az ADSR értékeket csatornánként általában elég a hangfájl elején egyszer megadni. Ez összesen 3 bájjal növeli a zene adat méretét. Ha nem adunk meg ADSR-t, akkor nem lesz lecsengés, addig szól a hang, amíg a következő hang nem jön, vagy egy **MUTE** is elnémítja a csatornát.

SIDDumpTVC.exe konverter használata

Ha parancssorból kiadjuk a siddumptvc.exe parancsot akkor egy help-et kapunk:

```
>siddumptvc.exe
```

```
Usage: SIDDUMP <sidfile> [options]
```

```
Warning: CPU emulation may be buggy/inaccurate, illegals support very limited
```

Options:

```
-a<value> Accumulator value on init (subtune number) default = 0
```

```
-c<value> Frequency recalibration. Give note frequency in hex
```

```
-d<value> Select calibration note (absnotation 80-DF). Default middle-C (B0)
```

```
-f<value> First frame to display, default 0
```

```
-l Low-resolution mode (only display 1 row per note)
```

```
-n<value> Note spacing, default 0 (none)
```

```
-o<value> Oldnote-sticky factor. Default 1, increase for better vibrato display  
(when increased, requires well calibrated frequencies)
```

```
-p<value> Pattern spacing, default 0 (none)
```

```
-s Display time in minutes:seconds:frame format
```

```
-t<value> !!! Playback time in seconds, default 60 !!!
```

```
-z Include CPU cycles+rastertime (PAL)+rastertime, badline corrected
```

TVC specific options

```
-1 channel 1 is disabled (default: enabled)
```

```
-2 channel 2 is disabled (default: enabled)
```

```
-3 channel 3 is disabled (default: enabled)
```

```
-r ADSR dump disabled (default: enabled)
```

```
-q Frequency slip dump disabled (default: enabled)
```

A legfontosabb paraméterek

- **-f** amivel az első frame-et adhatjuk meg. Ezzel állíthatjuk be, hogy honnan kezdődjön a lejátszás a zenén belül. Alapértelmezetten ez 0, tehát az elején kezdődik.
- **-t** a konvertálandó zene hossza másodpercben. Alapértelmezett: 60 (1 perc). Azaz ha egy SID zene 2 perc 31 másodperc hosszú, akkor -t151 paramétert kell megadni

TVC specifikus paraméterek

- **-r** ADSR dump kikapcsolása (default: bekapcsolva)
- **-q** Frekvencia csúszás dump kikapcsolása (default: bekapcsolva)
- **-1** 1-es csatorna kikapcsolása
- **-2** 2-es csatorna kikapcsolása
- **-3** 3-as csatorna kikapcsolása

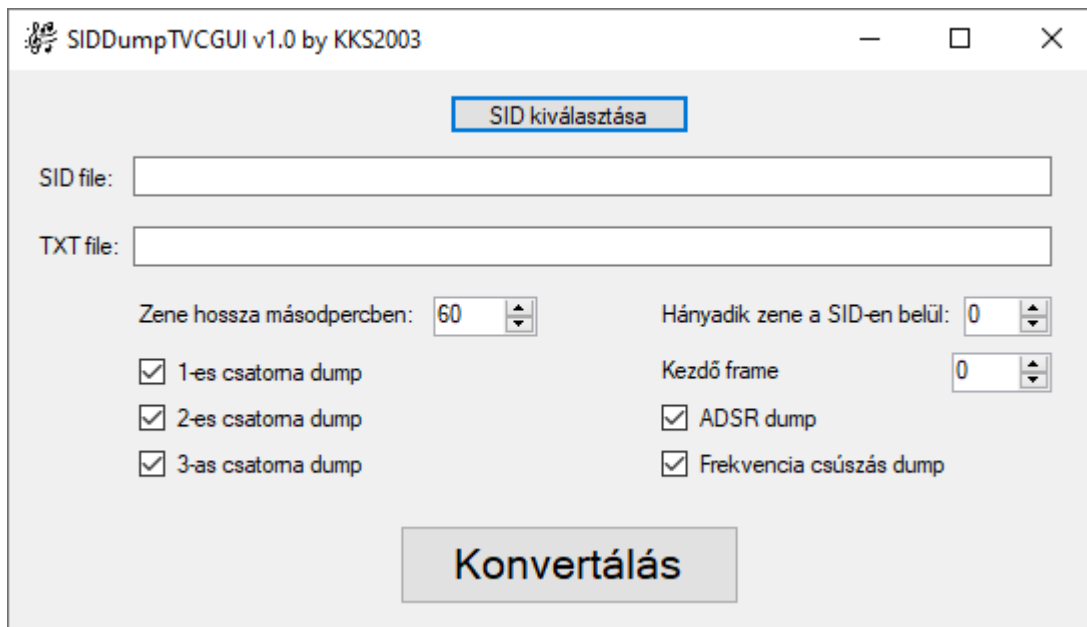
Parancssori példák

- Minden extra nélkül, csak a hangok kiírása:
`siddumptvc.exe -t39 -q -r Dune_Buggy.sid > music.txt`
- Minden bekapcsolva:
`siddumptvc.exe -t39 Dune_Buggy.sid > music.txt`
- Két csatornás zene, csak az 1. és 3. csatorna:
`siddumptvc.exe -t39 -2 Dune_Buggy.sid > music.txt`

Mivel a siddump a konzolra írja ki a kimenetét, ezért azt célszerű fájlba irányítani. Erre szolgál a parancs végén található **> music.txt** ami a konzol kimenetet beirányítja egy **music.txt** nevű fájlba.

SIDDumpTVCGUI.exe

Kiss Károly készített egy Windows alatt futtatható grafikus felhasználói interfészt, hogy még egyszerűbben konvertálhassuk a SID zenét.



A **SIDDumpTVCGUI.exe** könyvtárában kell, hogy legyen a **siddumptvc.exe** futtatható fájl is. A **SID kiválasztása** gombra kattintva keressünk és válasszunk egy **.sid** kiterjesztésű C64-es zene programot a számítógépünkön, majd a paraméterek beállítása után a **Konvertálás** gombra kattintva el is készül a zenelejátszó assembly kódba beilleszthető TXT fájlunk.

A paraméterek elnevezései magukért beszélnek.

Lejátszó programok

Miután ismertettem a lehetőségeket, a zenei formátumot és a SID konvertert, vágjunk bele a tényleges lejátszó programok ismertetésébe.

A programcsomag több különböző lehetőségekkel és funkciókkal rendelkező programból áll.

Minden program assembly forráskódja a csomag része. A programok 100% Z80 assembly-ben készültek TVC Developer Tool 4.2.0 segítségével. Abba betölthetők és F5-el azonnal futtathatók.

Lehetőségek és limitációk

Általános limitáció, ha fut a zenelejátszó megszakítás

- megszakítás nem tiltható
- stack pointer (SP) direktbe nem manipulálható (csak CALL, PUSH és POP utasításokra használható)
- RST \$38 hívások nem kezdeményezhetők
- általában ROM rutinok nem, vagy csak körültekintően használhatók
- minden hardver elemet (képernyő, billentyűzet stb. direktben portokon keresztül kell elérni)

2 csatornás lejátszó 4340Hz-es, CRCT szinkronos

- **2ch-fx-crtc.dt80asm**
- zenei hangok C1-C7 (32,7 - 2093 Hz)
- pontos CRCT szinkronnal (opcionális)
- négyszög, háromszög, fűrész és zaj hullámformák
- a 2.csatorna felhasználható hangeffektek keltésére
- Limitáció: AF', BC', DE', HL' árnyékrejiszterek (EXX) nem használhatók a főprogramban, ha megy a zene megszakítás
- hanggenerátor CPU használat: ~29%

3 csatornás lejátszó 5008 Hz-es, CRCT szinkronos

- **3ch-crtc.dt80asm**
- zenei hangok C1-C7 (32,7 - 2093 Hz)
- sebesség: 4-255-ig állítható (például a 100 -> 5008 Hz /100 = 50,08 Hz -> 3004 BPM)
- pontos CRCT szinkronnal (opcionális)
- csak négyszögjel
- Limitáció: IX, IY, rejiszterek és AF', BC', DE', HL' árnyékrejiszterek nem használhatók a főprogramban, ha megy a zene megszakítás
- hanggenerátor CPU használat: ~43%

3 csatornás lejátszó 8491 Hz-es, csak négyszögjel, ADSR

- **3ch_adsr.dt80asm**
- BPM 20-1200 (SID-hez általában 500)
- C1-C8 zenei hangok (32,7 - 4186 Hz)
- csak négyszögjel
- Hangerő lecsengés vezérelhető (ADSR00-0F kódokkal)
- frekvencia csúsztatás le-fel vezérelhetőek (FRQINC00-0F és FRQDEC00-0F kódokkal)
- teljes hossz: 1104 bájt
- a teljes lejátszó megszakításból működik, a főprogram a CPU maradék kb. 25% idejében bármit csinálhat
- Limitáció: IX, IY, regiszterek és AF', BC', DE', HL' árnyékregiszterek nem használhatók a főprogramban, ha megy a zene megszakítás

3 csatornás lejátszó 8491 Hz-es, különböző hullámformák

- **3ch_wavetable.dt80asm**
- BPM 20-1200 (SID-hez általában 500)
- C1-C8 zenei hangok (32,7 - 4186 Hz)
- négyszög, háromszög, fűrészfűrés és zaj hullámformák vezérelhetőek (WAVE01-04 kódokkal)
- nincs hangerő állítási lehetőség
- frekvencia csúsztatás le-fel vezérelhető (FRQINC00-0F és FRQDEC00-0F kódokkal)
- teljes hossz: 2210 bájt
- a teljes lejátszó megszakításból működik, a főprogram a CPU maradék kb. 25% idejében bármit csinálhat
- Limitáció: IX, IY, regiszterek és AF', BC', DE', HL' árnyékregiszterek nem használhatók a főprogramban, ha megy a zene megszakítás

Miért nincs egyszerre hullámforma is és hangerő (ADSR)?

Mert a legfontosabb, hogy ezek a lejátszók kevés memóriát foglaljanak el. Minden egyes hullámtábla 256 bájtot foglal el. Ebből van 4 (négyszög, háromszög, fűrészfűrés és zaj). (SID esetén még több kevert hullámforma tábla is.) Minden hangerőhöz le kellene generálni mind a 4 db 256 bájtos hullámtáblát, ami összesen $5 * 4 * 256 = 5120$ Kbyte lenne. Természetesen megvalósítható, de inkább azt javaslom, hogy akinek ADSR is és hullámforma is kell egyszerre, az használja a már ismert TVC-s SID playert!


Integráció

3ch_adsr.dt80asm

Kezdjük mondjuk a **Sötét Kastély 2024-el**. A Sötét Kastélyba a **3ch_adsr.dt80asm** 3 csatornás lecsengést is kezelő lejátszó lett integrálva. A hanggenerátor és a lejátszó is megszakításból fut, azaz a főprogramban csak arról kell gondoskodni, hogy meghívjuk a START_MUSIC_MIXER szubrutint, ami elmenti a meglévő megszakítást majd felülírja és inicializálja, végül elindítja. A megszakítás innentől teszi a dolgát, generálja a hangokat, lejátssza a zenét, ha **LOOP** kóddal találkozik, akkor pedig az elejéről kezd.

A lejátszó kódja az **.dt80asm** szöveges fájl 14.-től 466. soráig tart. Ezt kell bemásolni a programunk kódjába bárhova, de természetesen ki lehet indulni a lejátszó kódjából is, és abba fejleszteni a játék további részeit.

FONTOS, hogy a zenelejátszó forrásában található 484-722-es sorok is meglegyenek, azok tartalmazzák a hangfrekvencia táblát. A frekvencia tábla 256-al osztható címre kell, hogy kerüljön, ezt egy ORG assembly direktívával biztosítjuk.



```
org      $1d00
; Freq divider table for musical sounds (address must be divisible by 256)
SOUND_TABLE
dw      0           ; 0.00 Hz
dw      252         ; 32.7 Hz
dw      267         ; 34.65 Hz
dw      283         ; 36.71 Hz
dw      300         ; 38.89 Hz
dw      318         ; 41.2 Hz
dw      337         ; 43.65 Hz
dw      357         ; 46.25 Hz
dw      378         ; 49 Hz
dw      401         ; 51.91 Hz
-
```

A fenti \$1d00 cím változhat, de ennek a címnek a felső bájtját **KÖTELEZŐ** beírni a forráskód 142.sorába a MUSICPOS címke alatti sorba!

```
; parse record
MUSICPOS      ld      HL,MUSIC_DATA
|             ld      d,$1d           ; high address of sound frequency constant lookup table
MP_CH1        inc     hl
              ld      a,(hl)         ; load first byte of music record
; check control codes for channel 1
              cp      LOOP          ; first byte can be LOOP control
```

Ezután már csak arról kell gondoskodni, hogy a MUSIC_DATA címke mellett ott legyen a zene szöveges formátumban vagy binárisan.

Ha mindez ez megvan, akkor már csak két utasítást kell ismernünk:

`call START_MUSIC_MIXER`

és

`call STOP_MUSIC_MIXER`

A kettő között futhat a főcímképernyőnk, vagy akár a játékunk is.

Érdemes azért a regisztereket lementeni, lásd a forráskódban:

```
MAIN          di                ; IT disabled
              exx               ; save registers
              push HL
              push BC
              push DE
              exx
              push IX
              push IY

              call START_MUSIC_MIXER ; starts SOUND GENERATOR and MUSIC PLAYER INTERRUPT

;-----
RUNLOOP       halt                ; wait for next interrupt 0,70656ms
;-----
KEYBOARD_HANDLER ld A,7           ; check the ESC button
              out (3),A           ; select 7th row of keyboard matrix
              in A,(88)           ; read keyboard matrix row
              bit 3,A             ; check ESC key is pressed
              jr NZ,RUNLOOP

;-----
END_OF_GAME   call STOP_MUSIC_MIXER ; stops SOUND GENERATOR INTERRUPT

              pop IY              ; restore registers
              pop IX
              exx
              pop DE
              pop BC
              pop HL
              exx
              ret                ; end of game
```

A fenti kódban a **MAIN** címkénél kezdődik a program. Egyből elment minden használt regisztert a verembe, majd meghívja a **START_MUSIC_MIXER** szubrutint.

A tényleges főprogram itt a **RUNLOOP** címke után álló egyetlen halt utasítás, és az utána lévő pár sor, ami mindössze itt most annyit csinál, hogy vizsgálja a billentyűzet mátrixot, hogy az **ESC** billentyű le lett-e nyomva.

Ha igen, akkor meghívjuk a **STOP_MUSIC_MIXER** szubrutint, majd gondoskodunk az elmentett regiszterek visszatöltéséről és az utolsó **RET**-el vége is a programnak.

A főprogramban, ha fut a lejátszó, akkor **TILOS**:

- letiltani a megszakítást, mert arra az időre a zene elhallgat
- a stack pointert (SP) direktbe címezéshez használni. (csak normál CALL, PUSH és POP utasításokra használható)
- RST \$38 hívásokat kezdeményezni. Amit ROM rutinokkal akartunk megoldani, azt most le kell programozni!
- a főprogramban IX, IY, regisztereket, valamint az árnyék AF', BC', DE', HL' regisztereket használni.

HALT utasítás használható, de nem 20,096ms-ot fog várni, hanem csak 0,11776 ms-ot, mivel a megszakítás 8491 Hz-en működik.

Lássunk néhány részletet a Sötét Kastély 2024 játékba integrált zenelejátszóból:

```

|
org     6639                                ; TVC programok kezdőcíme
;(BASIC Header
db      $0F,$0A,$0,$DD                     ; "10 PRINT" - basic token
db      'USR'                               ; "USR"
db      $96                                 ; "("
db      '6659'                             ; "6659"
db      $95,$FF                             ; ")"
db      0,0,0,0,0                           ; így 20 byte a BASIC header, a kód pedig 6659-nél ($1A03) kezdődik
;)

jp      MAIN                                ; <- 6659 ($1A03) memóriacím (a BASIC header ezt hívja meg a "10 PRINT USR(6659)" utasítással)

;*****
; INTERRUPT HANDLER
; Execution time: 264 cycle
; Interrupt Frequency: ~ 8491.847826086956 Hz
; CPU load by IRQ + PLAYER: ~ 80%
;
; EXX IX = channel 1 phase accumulator
; EXX IY = channel 2 phase accumulator
; EXX HL = channel 3 phase accumulator
;
; EXX BC = local variable for calculation
; EXX DE = local variable for mixing
;*****
IT_HANDLER      ex      af,af'
                exx     out      (7),a                ; ack interrupt

```

A megszakítás és a lejátszó maradt a PRINT memória legelején, így nem kellett belenyúlni a frekvencia tábla címébe sem.

```

#include      "motor1_drawimage1.dt80asm"
#include      "motor1_alap_veg.dt80asm"
#include      "motor1_MAP.dt80asm"
#include      "motor1_grafikal.dt80asm"

; MUSIC DATA RECORDS

MUSIC_DATA      equ      $C000
BIN_MUSIC_DATA  incbin   "music.bin.zx7"          ; Main title music

nop
end

```

A zenét pedig a teljes forráskód végére tettem és ZX7-el van betömörítve.

A JP MAIN ide ugrik:

```

MAIN      exx     ; save registers
          push    HL
          push    BC
          push    DE
          exx
          push    IX
          push    IY

          di
          ld      A,$B0                ; lapozási kód: U0, U1, U2, U3
          out     ($2),A
          ld      HL,BIN_MUSIC_DATA    ; HL = tömörített zene címe
          ld      DE,MUSIC_DATA        ; DE = kitömörítés címe (U3)
          call    ZX7_DECOMPRESS       ; zene kitömörítése

          ld      A,BPM_DIVIDER/BPM
          ld      (BPM_COUNTER+1),A
          ld      A,DURATION_IRQ_CONST
          ld      (DURATION_COUNTER+1),A
          ld      HL,MUSIC_DATA-1      ; -1 is needed because parsing is started with inc hl
          ld      (MUSICPOS+1),HL
          ld      (MUSIC_RESTART+1),HL

          -----
          RUNLOOP 3 call    START        ; wait for next interrupt 0,70656ms
          -----
          END_OF_GAME call    STOP_MUSIC_MIXER    ; stops SOUND GENERATOR INTERRUPT
                  pop     IY                ; restore registers
                  pop     IX
                  exx
                  pop     DE
                  pop     BC
                  pop     HL
                  exx
                  ret                        ; end of game

```

Az 1-es részben mentem a regisztereket, A 2-esben kitömöríttem a zene adatokat ZX7 formátumból bináris zene adatokká, majd a 3-as részben indítom el az eredeti Sötét Kastély programot.

A START címkén elindul a Sötét Kastély kódja, majd valahol benne ez található:

```

;
call    RST30DB2
szoveg kiirasa "   OROSZ OLIVER "
ld      BC,$1317          ; X,Y karakter pozíció
call    RST30DB3
ld      DE,szoveg_6        ;szoveg címe
call    RST30DB2
;
szoveg kiirasa "   "
ld      BC,$0218          ; X,Y karakter pozíció
call    RST30DB3
ld      DE,szoveg_7        ;szoveg címe
call    RST30DB2

call    START_MUSIC_MIXER ; starts SOUND GENERATOR and MUSIC PLAYER INTERRUPT

WAIT_FOR_SPACE ld      A,7          ; check the SPACE button
out      (3),A          ; select 7th row of keyboard matrix
in      A,(88)          ; read keyboard matrix row
bit      5,A            ; check SPACE key is pressed
jr      NZ,WAIT_FOR_SPACE
```

Itt, amikor már felépült a kezdőképernyő, egyszerűen indítom a zenelejátszót és kész is vagyunk, működik.

Persze a limitációk miatt bele kellett nyúlnom még a kódba is, valamint az eredeti hangokat is kiesztem belőle, de ez nem tárgya ennek az integrációs leírásnak.

Beállítások

- A három csatorna master hangereje külön-külön is állítható az ADSR lecsengéstől függetlenül. Azonban a lecsengés sebessége a master hangerővel változik, csökken a lecsengési idő.

FIGYELEM! Ide szigorúan csak \$14, \$10, \$0C, \$08, \$04 vagy \$00 hangerő érték írható! \$00 esetén természetesen az adott csatorna nem fog szólni.

```

;*****
; MAIN PROGRAM
;*****
; developer could set this constants

BPM equ 500 ; <- set the Beat per Minutes (BPM) HERE

MASTER_VOLUME_CH1 equ $14 ; can be: $14,$10,$0C,$08,$04,$00
MASTER_VOLUME_CH2 equ $14 ; can be: $14,$10,$0C,$08,$04,$00
MASTER_VOLUME_CH3 equ $14 ; can be: $14,$10,$0C,$08,$04,$00
```

- Ugyanitt a BPM érték is megadható. SID-ből konvertált zenéknél ez általában 500, más kézzel írt, vagy máshonnan konvertált zenéknél ez túl gyors lenne, használjuk kisebb értéket. Minimum: 20 (leglassabb), maximum: 1200 (szupergyors)

- A videó mód beállítása is **NAGYON FONTOS!** Itt lehet megadni:

```
CH3_VOLUME      --
                and    MASTER_VOLUME_CH3
                add    A,B          ; MIX CH1 + CH2 + CH3
                or     1          ; must have in 4 color graphics mode (2 for 16 color mode)
                out    (6),A       ; write to sound port
DURATION_COUNTER ld    A,DURATION_IRQ_CONST ; 06h -> 1 ms, 55h -> 10 ms
                dec    A
```

2 színű módban írjunk ide **or 0**-t

4 színű módban írjunk ide **or 1**-t

16 színű módban írjunk ide **or** 2-t

3ch_wavetable.dt80asm

A 3 csatornás, különböző hullámformákat használó lejátszó integrálása nagyon hasonló az előzőekben leírtakhoz. A lejátszó az **ADSRxx** vezérlő kódokat nem tudja értelmezni, azonban a **FREQINCxx** és **FREQDECxx** parancsokat igen, valamint a **WAVExx** parancsokat is az alábbiak szerint:

- A **WAVExx** vezérlő kódok a csatornák hangjai előtt áll(hat)nak, bárhol a zenei adatokban. Tehát ezek nem önálló kódok, hanem egy-egy csatorna kiegészítő kódjai.
- Például: **DB WAVE01,C4,WAVE02,FISZ4,WAVE03,SKIP**

WAVE01 négyszögjel hullámforma

WAVE02 háromszög hullámforma

WAVE03 fűrészfog hullámforma

WAVE04 zaj hullámforma

A programhoz mellékelt példa zene nem tartalmaz **WAVExx** parancsokat, azonban a zene adatok előtti sorban (az asm fájl 711-es sorában) Mindhárom csatornára egyszeri beállítással megadtuk a hullámformát.

```
db WAVE01,SKIP,WAVE02,SKIP,WAVE03,SKIP
```

azaz 1-es csatorna négyszögjel, 2-es csatorna háromszög, hármás fűrész

Az integráció során figyelni kell az alábbi címkék, 256-al osztható címen való elhelyezésére:

SOUND_TABLE – ez a példa kódban \$1D00 címen kezdődik. Ha ez a cím változik, akkor a már ismertetett módon az új cím felső 8 bitjét az assembly kód 93-as sorában lévő utasításhoz kell beírni.

```

; parse record
MUSICPOS      ld    HL,MUSIC_DATA
               ld    d,$1d           ; high address of sound frequency constant lookup table
MP_CH1        inc    hl
               ld    a,(hl)         ; load first byte of music record
; check control codes for channel 1
               cp     LOOP          ; first byte can be LOOP control
               jr     NZ,MP_SKIP

```

A program 366-os sorában a BPM sebessége állítható az előzőekben ismertetett módon.

A program 368-371 soraiban pedig a 4 hullámformának megfelelő táblák címeinek felső bájtját kell megadni.

```

;*****
; MAIN PROGRAM
;*****
; developer could set this constants

BPM           equ     500           ; <- set the Beat per Minutes (BPM) HERE
|
WAVEFORM01    equ     $1e           ; address high of square table
WAVEFORM02    equ     $1f           ; address high of triangle table
WAVEFORM03    equ     $20           ; address high of sawtooth table
WAVEFORM04    equ     $21           ; address high of noise table

```

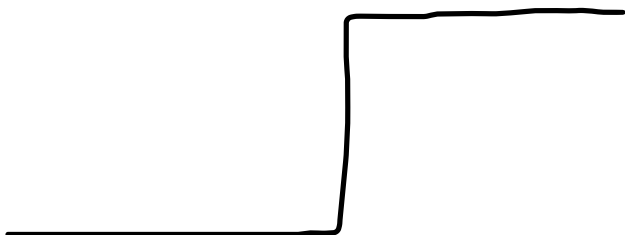
A fentiek szerint a négyszögjel tábla \$1E00 címen kezdődik a kódban, így a 368-as sorban a \$1e érték van. Ugyanígy a másik három hullámformára a \$1F00, \$2000 és a \$2100 címek vannak kijelölve. Ezeket a táblákat a kódban a

square_table
triangle_table
sawtooth_table
noise_table

címek alatt találjuk. Mindegyik hullámforma tábla 256 bájtos, ami azt jelenti, hogy a teljes hullámképe az elejétől a végéig ki kell, hogy töltsen a 256 bájtot.

Vagyis például egy négyszögjel úgy néz ki, hogy 128 db bájt nulla hangerő és 128 db bájt maximális hangerő.

Lerajzolva:



FONTOS, hogy nemcsak ez a négy, hanem bármilyen hullámforma kialakítható ezen 256 bájtos táblázatok előállításával. Az egyedi hullámformák előállításánál lényeges, hogy csak az alábbi hangerő értékeket használhatjuk: \$1C, \$18, \$14, \$10, \$0C, \$08, \$04, \$00

(maximális hangerő: \$1C, elnémítva: \$00)

A hullámforma lehet akár digitalizált is, de legyen normalizálva a fenti értékekre.

2ch-fx-crtc.dt80asm

Ez a lejátszó eltér a többitől, mert itt csak a hanggenerátor fut megszakításból, a zene lejátszása és a hangeffektek keltése a főprogram feladata. Ez azért van, mert a hangeffektek kezelése a programozó dolga, hiszen minden játékhoz más és más hangeffektre van szükség. A példa kódban a SPACE lenyomására egy robbanás szerű lecsengő és frekvenciájában lefutó zaj hullámforma szolgáltatja a robbanás effektet.

A lejátszó felépítése

A 14.-től a 104. sorig a lejátszó felépítése hasonló. Itt a megszakítás és a START_MUSIC_MIXER, valamint a STOP_MUSIC_MIXER szubrutinok.

A 42.-es és 47.-es sorokban lehet megadni a két csatorna hullámformáját.

```
; convert signal to waveform
ch1_waveform_tableH    ld    b,$1E    < ; address of wave table for channel 1
                        ld    c,d
                        ld    a,(bc)    ; lookup wave table for channel 1
                        ld    c,h      ; save h
ch2_waveform_tableH    ld    b,$1D    < ; address of wave table for channel 2
                        ld    a,(bc)    ; lookup wave table for channel 2
                        add    a,h      ; mix two channels
                        ld    h,c      ; restore h
```

Az előző 3 csatornás hullámtáblás lejátszóhoz hasonlóan itt is 4 hullámtáblánk van, de itt a lejátszó nem a zene adatokból szedi fel a hullámtábla vezérlőkódokat, hanem nekünk kell direktbe megadni a két csatornára.

FIGYELEM! A lejátszó semmilyen vezérlőkódot nem ismer a **SKIP** és a **LOOP**-on kívül.

A 109-113-as soroknál van a robbanás effekt indításának kódja. Nincs benne nagy titok, a 2. csatornát használjuk effekt csatornának, azaz a zene lejátszás közben elvesszük a 2. csatornát és effektnek használjuk.

A 128-as sorban kezdődik a főprogram RUNLOOP, ahová a játékunknak mindig vissza kell térni a CRTC szinkronizálás miatt.

Tehát a 128-145-ös sorok lefutnak, és biztosítják, hogy egy CRTC által kiváltott megszakítás után legyen a program. Azaz innentől van kb. 20ms-unk, hogy frissítsük a pályát, a játéklogikát, beolvassuk a billentyűzetet stb. Mivel eközben a megszakítás is fut, és a CPU idő 29%-át elveszi, így kb. 14ms raszter időnk van a játékra. Mindez persze csak akkor igaz, ha 50 FPS-el (Frame per second) szeretnénk frissíteni a játékkeret. Ha 25 FPS-el is megelégszünk, akkor már $40\text{ms} \cdot 0,7 = 28\text{ms}$ tiszta CPU idő áll a rendelkezésünkre, hogy a játékot levezéreljük.

A ShootEmUp demó is ezt használja ki, 25FPS-el megy a scroll és a 14 db sprite kirakása.

Mindeközben a hanggenerátor megszakításból fut, de a zenét és az effekteket etetni kell, amiért a 173-as sorban kezdődő MUSIC_PROCESSOR a felelős. Itt található a robbanás effekt lekezelése is a 173-181-es sorok közt.

ÖSSZEFOGLALVA a játékunkat CRTC szinkronizáltan a 150-168-as sorok között valósíthatjuk meg. Ügyeljünk arra, hogy a zene lejátszónak is maradjon egy nagyon pici ideje, mielőtt letelne a 20 vagy 40 msec.

FIGYELEM! 50 FPS-hez (20ms) az 59-es sorban lévő SYNC_COUNT címkehez **85**-öt kell írunk. 25 FPS-hez (40ms), ugyanez **171**-et kell írni. 25 FPS-nél ügyelni kell még arra is, hogy a képernyő a 40ms alatt kétszer frissül, azaz, ha éppen akkor rajzolunk a video memóriába, amikor ott jár a CRTC rasterszámlálója, akkor az adott sprite / grafika villoghat!

A zene lejátszás sebességét a 183 és 187-es sorokba írt azonos számmal állíthatjuk be:

```
MP_NEXT      ld      A, 1      ; count duration
              dec      A
              ld      (MP_NEXT+1), A
              jr      NZ, RUNLOOP
              ld      a, 1
              ld      (MP_NEXT+1), A
```

1 a leggyorsabb (20ms = 3000 BPM) 255 a leglassabb = 14 BPM (minden egyes növekmény 20ms-al növeli a periódusidőt, azaz például 100-as értéknél $100 \cdot 20 = 2s$, ami 30-as BPM-nek felel meg)

Ahogy a 150-154-es sorokban látszik a HALT Z80-as utasítás használható, de nem a képfrissítést várja be, hanem a hanggenerátor megszakítást, ami másodpercenként 4340-szer (4340 Hz) fog bekövetkezni.