002 – Play music with YM2413 and SN76489 sound chips (smps driver)

Prerequisite : 000 – Create a new game project

This tutorial describes :

- How to setup a DefleMask project
- How to convert music from VGM file format
- How to play the music with the game engine

Introduction

The game engine provides multiple ways to play music on TO8.

This tutorial will explain how to use the smps driver of the game engine.

The smps is a format commonly used in games for the Sega MegaDrive/Genesis.

Specification can be found here :

https://hiddenpalace.org/News/Sega_of_Japan_Sound_Documents_and_Source_Code

Setup a DefleMask project

While it's possible to write directly to smps format with a text editor, it more convenient to use a software dedicated to music programming.

As our game engine smps driver was primary targeted to play files from YM2612 with YM2413 chip, the tool chain will use tools that handle YM2612 music data.

First, you will need to buy and install DefleMask.

Next setup a new project with "Genesis" system of type "Standard"

Use those files :

- a set of YM2612 instruments that matches YM2413 preset instruments

  YM2612 settings TO YM2413 instruments

- a set of drum samples from YM2413

  YM2413 drum samples.rar

Samples should be used on FM6 track only, and should be mapped to those notes :

- `C-3 : BD`
- `C#3 : SD`
- `D-3 : HH`
- `D#3 : CYM`
- `E-3 : TOM`

If you want to know how instruments where matched from YM2413 to YM2612, use this file :

YM2612 - YM2413 intstruments.xlsx

You may also use this information to use any software that will produce VGM with YM2612 and SN76489 data.

Now make your own music … once it's done, export your project to VGM file format.

Here is a sample file (by adnz) if you are lazy :

zmusic.vgm

Convert VGM file to SMPS

To convert VGM file to SMPS, we will be using an external tool : https://github.com/Ivan-YO/vgm2smps

This tools is provided as a binary in the following directory : tools\<operating system>\music

Go to this directory and run this command (replace <yourfile> with your file name):

```
vgm2smps <yourfile>.vgm
```

If DefleMask was set to NTSC (60Hz) instead of PAL (50Hz), you will need to add a parameter that will tell the tool to convert from 60Hz to 50Hz. Use this command instead (replace <yourfile> with your file name) :

```
vgm2smps <yourfile>.vgm -fps=50
```

Now you should have a file with the same name, but with the .bin extension

This bin file is a SMPS file compliant to Sonic the Hedgehog format.

To be able to play the file on TO8 with the smps driver, we need to change some data :

- YM2612 Instrument data should be replaced with YM2413 instrument id
- Tempo should be converted
- Offsets should be converted

All those changes are handled by a java converter included in the game engine.

run this command (replace <yourfile> with your file name) from the tools\<operating system>\music directory :

```
java -cp ../../../java-generator/target/game-engine-0.0.1-SNAPSHOT-jar-with-dependencies.jar fr.bento8.to8.audio.SmpsRelocate <yourfile>.bin -0 -m68
```

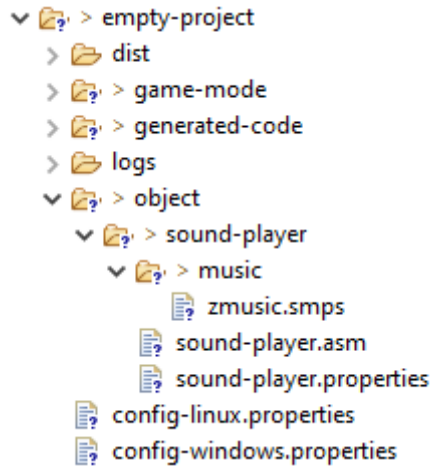
Now you should have a file with the same name, but with the .bin.smp extension

Rename this file with the .smps extension instead of .bin.smp

Play the music

File structure

Enter the infinite-loop directory and create the object directory structure (object -> sound-player -> music) :



- Put the .smps file in the music directory.
- Create a sound-player.asm
- Create a sound-player.properties

An object is a routine with a set of variables and data (images, sound, music, ..)

As opposed to the main routine of a game-mode that runs from the RAM Page 1 ($6100-$9fff), an object is executed from RAM pages (4-31) mounted in the $0000-$3fff space.

Game Mode definition

infinite-loop.properties

To define a game mode object, add a line to the **infinite-loop.properties** :

```
# Objects
object.sound_player=./object/sound-player/sound-player.properties
```

Object properties

Set those lines in the sound-player.properties :

```
code=./object/sound-player/sound-player.asm

# Sound
sound.Smps_zmusic=./object/sound-player/music/zmusic.smps
```

Object source code

Set this code into the object source code **sound-player.asm** :

```
; ------------------------------------------------------------------------
```

```
; Object - sound player
;
; input REG : [u] pointer to Object Status Table (OST)
; ---------
;
; ------------------------------------------------------------------------------

        jsr   YM2413_DrumModeOn
        jsr   IrqSet50Hz
        ldx   #Smps_zmusic
        jsr   PlayMusic
        jmp   ClearObj
```

This code is very simple for an object code, in following tutorials, you will see a much more complex approach. However, here we are just writing a code that will :

- init the sound driver to use drum instruments
- set an IRQ at 50Hz for the smps driver to send data at a regular speed to the sound cards
- load a register with the address of the music data (note that the label is the one used in the .properties file)
- start the playing of music
- self-destroy the object

Game mode source code

The game mode will be responsible of the object instantiation. To do so, we need to add the object id in the Object RAM location (the location of each object variables).

Note : this object will run once and clean this id, thus the object will be only run once.

Replace the main loop of the game mode source code **main.asm** with this one :

```
LevelMainLoop
        jsr   WaitVBL
        jsr   UpdatePalette
        jsr   RunObjects
        bra   LevelMainLoop

Object_RAM
                            fcb   ObjID_sound_player
                            fill  0,object_size-1
Object_RAM_End
```

Add also those lines of INCLUDE:

- At the beginning of the file (if those lines are not already there) :

```
        INCLUDE "./Engine/Macros.asm"
        INCLUDE "./Engine/Constants.asm"
```

- At the end of the file (if those lines are not already there) :

```
        INCLUDE "./Engine/ObjectManagement/RunObjects.asm"
        INCLUDE "./Engine/ObjectManagement/ClearObj.asm"
        INCLUDE "./Engine/Irq/IrqSmps.asm"
        INCLUDE "./Engine/Sound/Smps.asm"
```

You are ready to build and run your program.

Details in 000 tutorial