nesdoug

15. Adding music

If you read the Nerdy Nights tutorial, it seems like you should write your own music engine. I find you can do 90% of what you need with just Famitracker and Famitone2.

Here's the links to download each…

http://famitracker.com/downloads.php (http://famitracker.com/downloads.php)

Click on Latest Version/Binary. If you really really want to import a MIDI file, you should also download the 0.4.2 version. They never fully got the MIDI import working well, so they removed that feature after 0.4.2.

You could try to use the Famitracker driver, which has every feature under the sun, but the code is huge and slow. Much better is to use Shiru's Famitone2 driver…

https://shiru.untergrund.net/code.shtml (https://shiru.untergrund.net/code.shtml)

It has some restrictions, but we can get around most of them. No volume column. But I find you can simulate a volume column by creating extra instruments. Some will have a peak volume envelope less than F(15). You can alternate between the instruments to get different volumes out of your notes.

No fx (except Tempo changes, Loop, and End pattern). But you can simulate many of the effects with pitch and volume envelopes.

Arpeggio – use an Arpeggio sequence with a loop, like this…

| 0 4 11

Tremolo – volume envelope with a loop…

| f e d c b c d e

Vibrato – Pitch envelope with a loop…

| -2 -1 1 2 2 1 -1 -2

Pitch bends – Pitch envelope with a loop…

| 0 -1

Now, you can't go crazy, there is a limited amount of instruments you can add.

The main limitation that concerns me is the slightly smaller note range (C-1..D-6). You can get around this by editing the pitch table at the end of the Famitone2 ASM code, and utilizing an unused pitch. I haven't tried this, but it seems feasible. Here's a link to the complete NTSC pitch table, if you want to try

this idea…

http://wiki.nesdev.com/w/index.php/APU_period_table
(http://wiki.nesdev.com/w/index.php/APU_period_table)

It looks like Shiru removed 3 notes off the low end, and 14 notes off the high end.

Note – Triangle Channel notes will play one octave lower, that is just a quirk of the NES sound chip.

You should watch this tutorial, if you don't know how to use Famitracker…

The basics are…you need to create an instrument with a Volume Envelope and a Duty cycle. Then, press spacebar to enable editing. Use the keyboard to start entering notes. Z-M row = main octave, Q-P row = one octave up. Whatever instrument is highlighted will be inserted into the song.

To bump things down, highlight the part you want to bump, press 'insert' and everything below that point goes down 1. Press 'backspace' and everything below that point goes up 1 (deleting the line above the highlight).

Also, you can quickly switch octaves by pressing / or * on the keypad. 'Enter' will start the song (or stop).

Since we're using Famitone2, don't add any effects or volume column. It will probably take you a while to get used to this.

You can also use an external MIDI keyboard to enter notes, but you will still have to move the notes around (with 'insert' and 'backspace' because of the way they are inserted back to back).

We're going to put all the songs into 1 single Famitracker file. To add another song, go to Module/Module Properties/ then click Add to add a song. At the end of each song, I added a Bxx loop effect at the last line (it has to be on a unique pattern, so it doesn't loop too early).

So, I wrote 2 songs. They're not great, but just to test out the Famitone2 engine. We export the songs from Famitracker as a text file. Then bring it over to Famitone2/tools/ folder. Open a command prompt here File/Open Command Prompt. Since we are using cc65 we need to add a directive here. I typed…

text2data TestMusic.txt -ca65

…and it generated a .s ASM file. In our reset.s module, I included it like this…

.include “MUSIC/famitone2.s”

music_data:
.include “MUSIC/TestMusic.s”

and added this stuff to the startup code “detectNTSC:” and some other stuff to the nmi code to make that work. It will be able to detect whether it’s on a NTSC or PAL (European) NES.

And, all this junk was added, to make the Famitone IF statements work…

FT_BASE_ADR =$0100 ;actually the stack, but don’t worry, the stack will never grow big enough to reach the music variables

.define FT_THREAD 1
.define FT_PAL_SUPPORT 1
.define FT_NTSC_SUPPORT 1

FT_DPCM_OFF = $c000
FT_SFX_STREAMS = 1

.define FT_DPCM_ENABLE 0
.define FT_SFX_ENABLE 0 ;I have DPCM and SFx off, currently

I edited the Famitone2.s file a bit, specifically adding a few underscore labels…

_Reset_Music:
lda NTSC_MODE
ldx #<music_data ;low byte
ldy #>music_data ;high byte
FamiToneInit:

and…

_Play_Music:
FamiToneMusicPlay:

and…

_Music_Update:
FamiToneUpdate:

and, of course the export line at the top…

.export _Reset_Music, _Play_Music, _Music_Update

Now, we can call these functions from the C code, by defining their prototype…

void Reset_Music(void);

void __fastcall__ Play_Music(unsigned char song); // fastcall puts the parameter into the A register, not the stack

void Music_Update(void);

So, anytime I want to start a new song, you call Reset_Music(); and pass the song number to Play_Music(1);. Then, you have to call Music_Update(); once per frame.

I could have also included the other functions of Famitone, like Pause and Stop, but I find that all you have to do to pause the music is store zero at Register $4015, and just stop calling MusicUpdate.

((unsigned char)0x4015) = 0;

~~Then, to turn music back on later, store 0x0f at $4015, and start calling Music_Update again, every frame.~~ Wrong! *noise channel will not function, due to the way famitone only writes to 400f only during song initialization. Sorry I missed this bug. We must do this…

((unsigned char)0x4015) = 0x0f; // music channels on
((unsigned char)0x400c) = 0x30; // constant noise note, volume zero
((unsigned char)0x400f) = 0x00; // trigger noise channel note to start

Or, you could ignore me and just use the appropriate famitone Stop and Play functions.

We'll try to add some sound fx next time.

Here's the simple platformer with some music. Press 'Start' to switch between the songs.

http://dl.dropboxusercontent.com/s/4cl6dqvrzuyq2eq/lesson12.zip (http://dl.dropboxusercontent.com/s/4cl6dqvrzuyq2eq/lesson12.zip)

December 2, 2015April 15, 2017 dougfraker