

Author: Shauing
Version: 3.1
Originally written as ''How the SNES PoP Prologue Music Is Written In The ROM''.
1.0 version written from January 20th, 2019, 2:40 am, to January 25th, 2019, 2:11 am
Re-writes/corrections done on December 7th, 2019, starting at 6:43 pm and ending at 7:46 pm.
More additions and re-writes done on July 3rd, 2021, from 2:21 pm to 3:06 pm
Additions done on July 7th, 2021 at 3:13 am.
Re-writes/Additions done on July 19th, 2021 at 5:53 pm.
More additions done on August 17th, 2021 at 3:26 am and 4:35 pm.
Extra additions/re-writes done on August 20th, 2021 at 9:50 pm.
Simplified version done on August 22nd, 2021 from 6:58 pm to 7:57 pm.
Additions/Re-writes done on September 9th, 2021, 1:43 pm.
More re-writes/additions done on September 22nd, 2021, 8:40 pm and on October 30th, 2021, 2:14 pm.

This is my documentation on how the music is written in the SNES Prince of Persia music engine, trying to compile and simplify the information as clear as I possibly can.
Some things are still not 100% clear, but nothing major that requires the music engine to work.

(Mostly Using Title/Prologue Music as an example)

Part I. Define the length of data of the music track, where the instrument data is located and how many channels will be enabled and their sequence data start locations:

```
91 OD E1 0C FF 13 00 40 00 58 00 6D 00 82 00 A0 00 BE 00 D9 00
     91 0D = Data Length (Writing music space, starting from
          next byte)
     E1 0C = Offset to instrument data (where they are located)
     FF = Flag to enable channels; FF enables all channels.
          Original game only uses FF.
          00-FE Activates specific channels.
     13 00 to D9 00 = Offset to sequence channels 1 to 8 (Where
                      each channel sequences starts - 2 bytes
                      per channel).
                      8 channels must be defined even if
                      previous flag is not FF.
```


Part II. Where the music data for each channel starts, starting with echo commands, where each sequence starts, and whether the channel will loop back to a certain previous sequence or just end:

Channel 1 Sequences:
01 FD 00 02 0A 01 03 55 01 01 69 01 03 55 01 01 69 01 03 55 01 01 69 01 03 55 01 01 69 01 03 55 01 01 69 01 01 82 01 10 A3 01 FE 16 00

First three bytes:
01 = ? Original game only uses 01.
FD 00 = Where the start point for Channel 1's music data is located.

Sequence Programming:
Every sequence is programmed with three bytes, for example these three numbers (02 0A 01):
02 = Number of times sequence will loop
0A 01 = Where the first note of the sequence is located.

Close Channel 1 Sequence:
FE = Repeat to first note of sequence (FF = Stop)
(00 = Channel 1 ends)
16 (if FE is used) = From which sequence will repeat
(otherwise it should be 00)
00 = Song ends.

Repeat for all remaining 7 channels.

Part III. Music Data for each channel:

Thanks to David here for the following documentation regarding writing track data from his music_format.txt, and KungFuFurby for additional information and command duplicates (not documented here in order to keep it simple and clear).
Some corrections/extra comments added by me in italics, based on the SNES format document on princed.org, KungFuFurby findings and my experimentations:

n=number value

Tn = set tempo (in what units) (bigger=longer/slower)
Vn = set volume (0-15) (initial=13)
^ = increase volume
_ = decrease volume
@n = set instrument
On = set octave (0-7) (initial=4)
+ = up one octave (wraps around)
- = down one octave (wraps around)
C,D,E,F,G,A,B(n) = play note (n = set length of this sound and
next ones, relative to Tn)*(n=0-255)*
# before CDEFGAB = sharp note
R(n) = rest/pause (n = same as for CDEFGAB)
=n = set balance (0-127) (initial/default=64 center)
< = same as =127 (balance to left)
> = same as =0 (balance to right)
[= disable Decay(?)/start merge(?)
& = disable Attack(?)/separate merge(?)
] = enable Attack/Delay? /end merge(?)
[x&y] = merge two sounds
)n = set pitch up?/set detune? (signed) /
Set cents (upwards only)
$n = set filter? set all echo coefficients to a preset /
(0-3) FIR Filter presets
$c,n = set filter of channel? set an echo coefficient (signed) /
Manually set FIR Filter for each channel (0-7),(0-255)
Qn = set early stop for each sound? (1-16) set fill ratio: note
will fill n/16 part of the time given for it (0 means 16)
Pn = merge sounds for given length? fade out? modulation? (flags |= $04)
disable Decay? /
Pitch bending, disables attack and merges notes for the length of the instrument decay, 0 deactivates it, 1-255 means how fast the pitch will bend to the next note
*Hn = echo volume
*In = echo feedback volume
*Jn = echo delay
*Kn = echo *panning*
Ln = move the balance left-right in a wave

! = *Flag the game checks to resume movement after the last Jaffar and four-armed boss intro ''cutscenes''*

% = skip to next % (not used *in original game*)/
If there's no other % (I believe) it searches through the entire ROM as it stops everything (including game) before going to the next sequence (if there's any, otherwise the game will most likely freeze)

Mn = ? pitch modulation enable ? (not used) /
Activates pitch modulation flag

*Nn = stop? noise? (not used) (default=255)/
It generates noise on the channel is written depending of the value (0-31), activating the N flag, but it also activates the E flag (Echo). Adding n255 afterwards (i.e. N31N255), cancels the noise but the echo flag remains active but only goes up to five channels.

** No echo output occurs due to a bug in the code seemingly in the N command section where the accumulator containing the EON (Echo enable) bits was overwritten, therefore no echo is enabled.*
KungFuFurby from SNESLab's Discord provided a fix for this:
Replace: E4 CF 8D 6C 3F B1 09 8D 4D 3F B1 09
With: 2D EB CF E8 6C DA F2 EE E8 4D DA F2
Explanation for the fix: The main thing I did was to save the old accumulator containing the EON flags upon arrival, then retrieved them after writing to the FLG DSP register so that they would be written correctly.

Adding some extra information regarding track data:
- Tempo (T) command for this tune, is located at the beginning of the music data space for Channel 8.
- Here's why I determined to be the best note values, but maybe depending of the song you might want, for example, the whole note have a value of 64 rather than 128, but the rest shall be adapted to accomodate for this change:

2 = 64th
4 = 32nd
8 = Sixteenth
12 = Dotted Sixteenth (Sixteenth + 32nd)
16 = Eighth
24 = Dotted Eighth (Eighth + Sixteenth)
32 = Quarter
48 = Dotted Quarter (Quarter + Eighth)
64 = Half
96 = Dotted Half (Half + Quarter)
128 = Whole
192 = Dotted Whole (Whole + Half)
255+1*= Double Whole (*Note values cap at 255)('&' command can be used to merge both note lengths)

Part IV. Instrument Configuration. Choosing an instrument, define Attack/Decay/Sustain/Release values, pitch bend or vibrato options, when to start the chosen option, cents and octave values.

First Instrument Sample Settings:
00 7F ED 00 00 0A 05 00 15 00 00

00 = Instrument Sample
7F ED = Attack & Decay Volume
00 00 = Sustain & Release Volume (?)
0A 05 = Pitch envelope: 1st byte: bending distance between two pitches
2nd byte: speed of bending
00 = Effect of pitch envelope:
00 = Up & Down Vibrato
01 = Pitch Bend Up Only
02 = Up & Down No Bending
03 = ? (Different effect from the above three)
15 = Time To Activate Pitch envelope (It also affects the speed)
00 00 = Cents (00-7F upwards)(80-FF downwards) & Octave (00 or FF used in original game)

Instrument Samples List:
00: Brass (Transposing Instrument)
01: Trumpet
02: Ney (Flute-like Instrument)
03: Strings (Transposing Instrument)
04: Marimba
05: Sitar (Transposing Instrument)
06: Zurna (Oboe-like instrument)
07: Bass
08: Bass Drum
09: Snare Drum
0A: Hi-Hat
0B: Tom
0C: Crash Cymbal
0D: Shaker
0E: Timpani
0F: Ney II (?)(Sounds identical to Ney)
10: Xylophone/Chimes (Konami logo instrument)(Only used there/Not in Japanese version)

Sound Effects Samples List (can be used in the music data using the values in the range 40-54):
40: Prince Steps
41: Door Sound 1?

42: Door Sound 2?
43: Door Sound 3? (All three sound not quite like percussion)
44: Synth-like (High frequencies only)
45: Organ-ish/Finger-bass-ish instrument(?) (synth-like)
46: Noisy-instrument/sound (part of gate closing?)
47: Synth-y instrument(?)
48: Low Guitar-ish(?)
49: Door closing(?)
4A: Less-bell-ish sound (Perhaps part of gate opening/closing)
4B: Bell-ish sound (Part of gate opening/closing)
4C: Mid-low noisy sound
4D: Jaffar's Magic Charge
4E: Jaffar's Magic Charge 2
4F: Swords Clashing
50: Guard Getting Hit
51: Gate Opening/Closing
52: More Jaffar's Magic Charge(?)
53: Amazon Getting Hit
54: Prince Getting Hit

To-do (possibly): Add a section for inserting custom instrument/sound samples.
