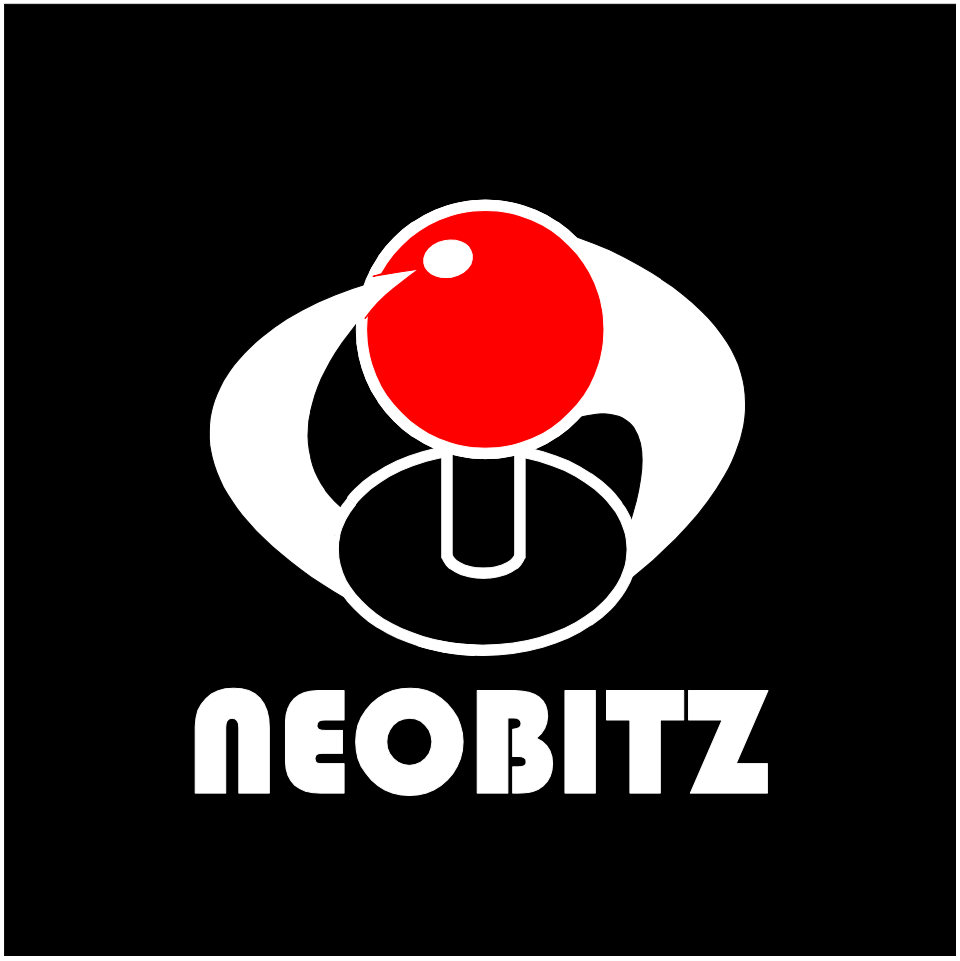
Neo Sound Builder

User's Guide



CONFIDENTIAL – DO NOT DISTRIBUTE

About Neo-Geo ADPCM-A Sample Format

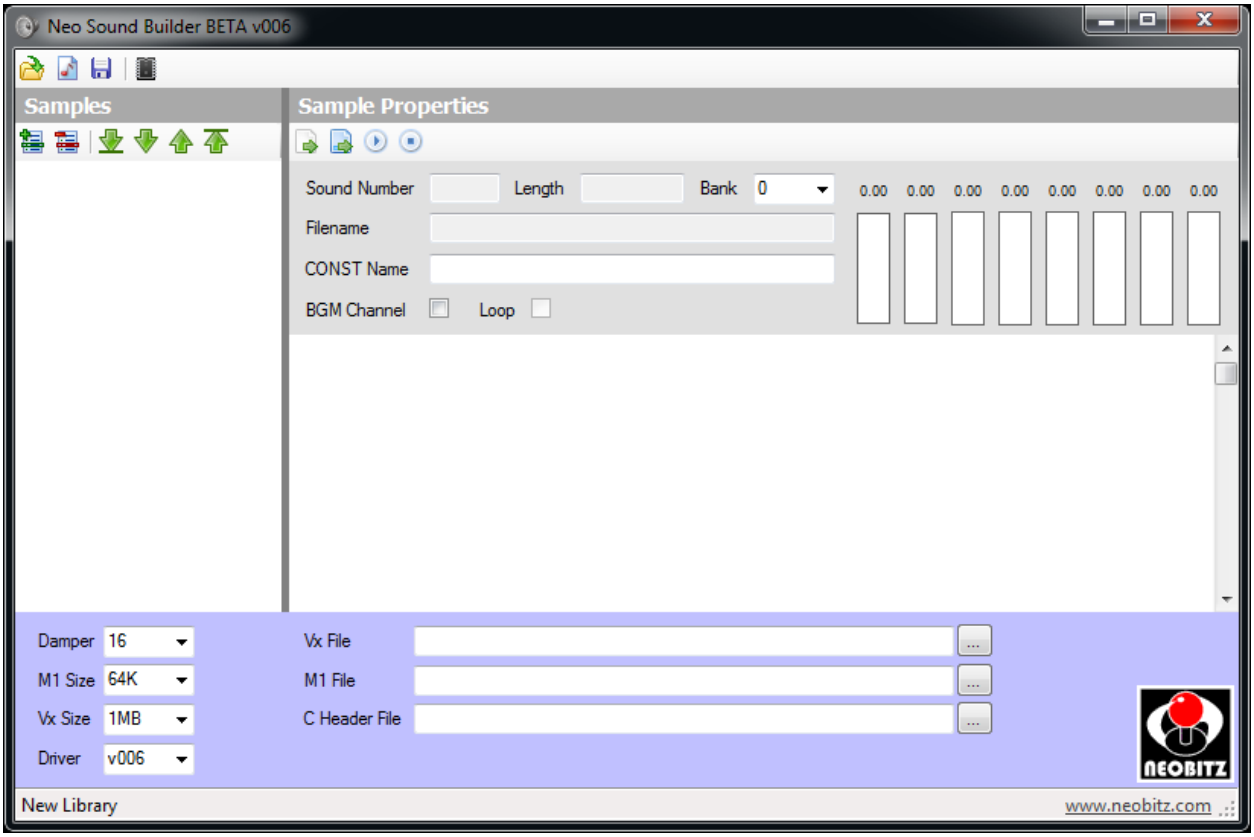
The Neo supports multiple formats of audio. Currently, the Neobitz sound driver does not support FM, PSG or ADPCM-B. The driver only supports ADPCM-A. The other types of audio may be added at a later date.

The Neo supports 7 channels of ADPCM-A samples. Neo Sound Builder dedicates one channel to background music and the remaining six channels for game sound effects. The sound driver will play the sound effects in a "round robin" sequence meaning it will start on channel 1, then 2, then 3, up to channel 6 and then start back over at channel 1.

ADPCM-A on the Neo has a fixed sample rate of 18,500 Hz. APDCM-A also has boundary limitations so you can only play a sample that is approximately 1:54 (1 minute, 54 seconds) long.

When creating sound samples for your game, save them as WAV files with the sample data set to 16 Bit PCM Signed Mono. Make sure your sample rate is 18,500 Hz. A good sound editing program to use would be Goldwave (www.goldwave.com) to re-sample and edit WAV files.

Using Neo Sound Builder



Neo Sound Builder allows you to easily manage a library of samples to use in your Neo game. Samples are imported and converted to ADPCM-A and your library is stored as a .NSL (Neo Sound Library) file.

The top tool bar icons in order are:

- Open Library – Allows you to open a library you've previously created.
- New Library – Clears all samples and starts a new library.
- Save Library – Allows you to name and save your sample library.
- Build ROMS – Build sound ROMS (Vx) and the sound driver (M1 ROM)

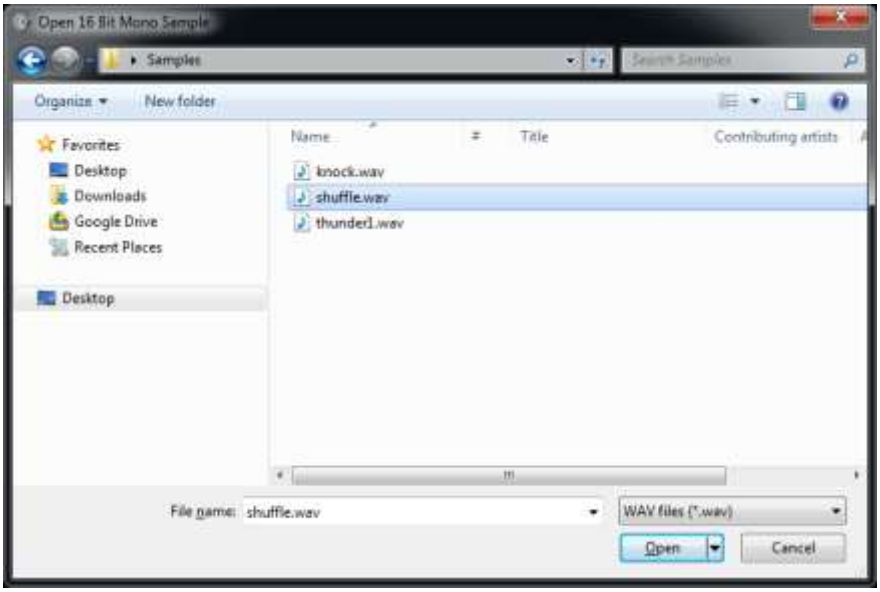
Neo Sound Builder is divided into three sections. The section on the left is the list of samples in the library. The section on the right is information about the currently selected sample and the section on the bottom is the ROM build settings.

Managing Samples

The samples toolbar is used to manage the samples in your library. The icons allow you to:

- Add Sample – Adds a sample to the library
- Remove Sample – Removes sample from the library
- Move to Bottom – Moves the sample to the bottom of the sample list
- Move Down – Moves the sample one location down in the sample list
- Move Up – Moves the sample one location up in the sample list
- Move to Top – Moves the sample to the top of the sample list

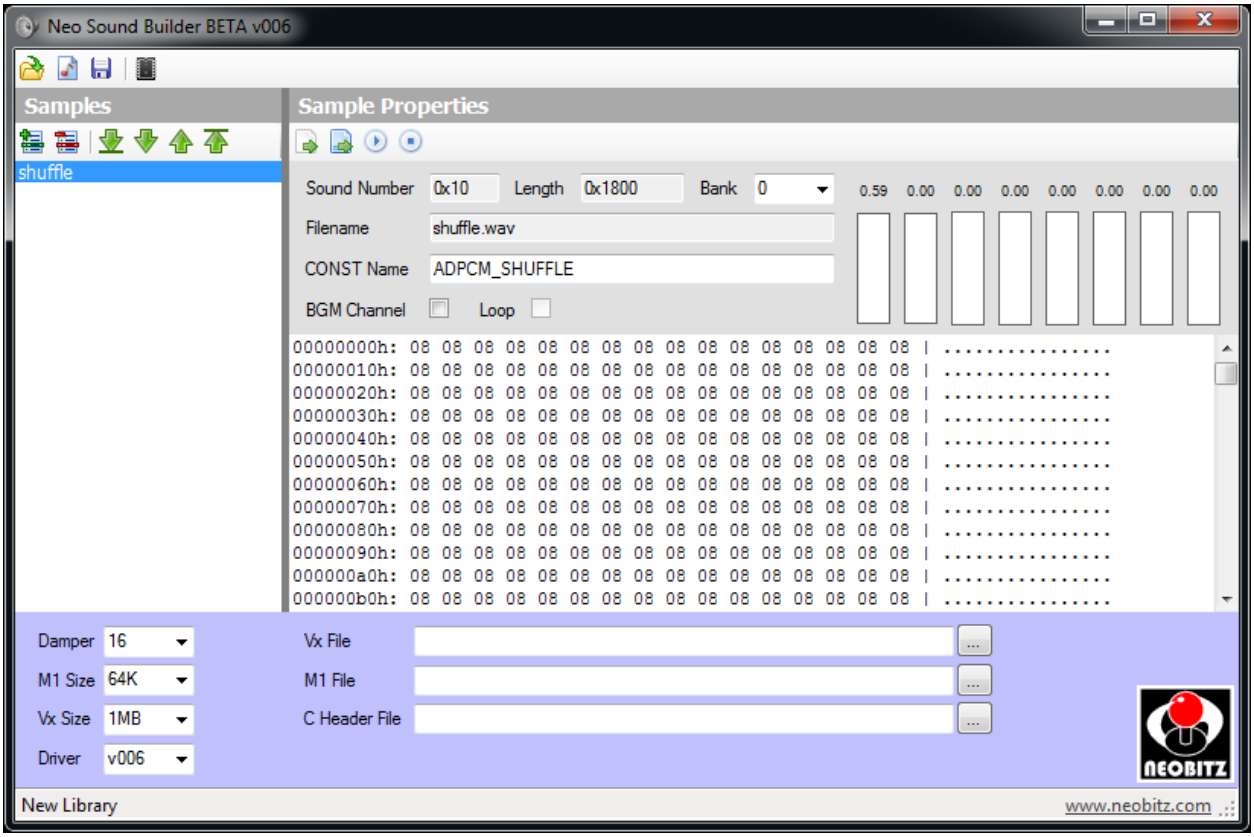
You'll start building a library by adding your audio samples to a library. You do this by clicking the Add Sample icon. This will open a file dialog to allow you to select the audio file to import.



The default file type that Neo Art Box wants to import is a WAV file, but several other formats are supported:

.WAV – 16 bit signed PCM WAV
.SND – 16 bit signed raw data
.NAS – Neo Audio Sample (raw ADPCM-A data)

Once you select the file to import, Neo Sound Builder will convert the sample to ADPCM-A format and add it to the library.



After the sample is imported into the library, it will be selected and the details about the sample will be shown in the Sample Properties section.

When a sample is imported, it is assigned a sequential Sound Number starting at 0x10. For your reference, the length of the sample and the original filename are shown. A CONST Name is also created for your sample. This is a C constant name associated with the sample number to make playing samples in your game easier. If you change the order of the samples in the sample list, the Sound Number will change but using a constant name in your code will keep you from having to change your game code.

The Neo hardware has a limit on how samples can be played. It's best to think of the audio being stored in 1MB banks. If your sample is too large and extends across a bank's boundary, the sample will not play properly. For this reason, your samples need to be smaller than 1MB when encoded or about a 4MB WAV, which is about 1:54 (1 minute, 54 seconds) of play time. Also, if a bank has a lot of smaller samples in it, make sure the last sample does not cross the bank boundary.

To make managing sound banks easier, Neo Sound Builder has visual gauges to show how full a bank is. If you need to move a sample to another bank that has more free

space, you can select a new bank to store the sample in by changing the Bank selection. If you have a bank that has too many samples, causing one of them to cross the boundary, the gauge will turn red.

The BGM Channel and Loop checkboxes control how your sample is played by the sound driver on the Neo.

The Neo as 7 ADPCM-A channels. Neo Sound Builder dedicates one channel to background music (BGM) so any time you start a sample that is flagged as a BGM sample, it will stop the currently playing background sample and start the new sample. The Loop option will automatically “loop” the BGM sample. Loop is only available for samples played on the dedicated background music channel.

The remaining 6 ADPCM-A channels are used to play all game sound effects. This should be adequate for most games. When you play a sound effect for the first time, it will be played on channel 1. When you play the next sample, it will be played on channel 2. As you continue to play samples, the next channel is used until it reaches the 7th channel, which it then starts back on channel 1. This is typically called a “round robin” sequence.

For reference, the data for the sample is displayed in hex format.

Within the Sample Properties section, you can export or play a sample.

- Export Encoded Sample (NAS - Neo Audio Sample format)
- Export WAV
- Play Sample
- Stop Sample

ROM Settings

The bottom section of Neo Sound Builder controls how the ROMS are built.

Damper – When importing the 16 bit data, it needs to be scaled down to 12 bits for the Neo before encoding. On some samples that have a large dynamic range, the samples may have static and popping. The Damper will lower the bit rate of the sample. This value must be set prior to importing a sample.

NOTE: As of beta v006, the ADPCM-A encoding has been improved and the Damper setting shouldn’t need changed unless absolutely necessary.

M1 Size – Size of the M1 file to output. The audio driver does not support banking so it can only be 64K currently. If you need a 128K file for testing in emulators, change this to 128K and the file will be padded.

Vx Size – The size of the V ROMS to generate.

Driver – Version of driver to use in your project. Currently v006 is the most stable version.

VxFile – Where to output the Vx files.

M1 File – Where to output the M1 file.

C Header File – Where to output the C header file. This header file contains everything needed to play samples in your game and defines all of the C constant values for your sounds.

Playing Sounds

After adding sounds to your library and setting up the output files, you can build the Vx, M1 and .h header file, typically named “sound.h.” You will need to include this header file in your game code.

Example main.c Code

```
#include <stdlib.h>
#include <video.h>
#include <input.h>
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <ctype.h>

#include "sound.h"

int main()
{
   wait_vbl();

   send_sound_command(ADPCM_MUSIC);
}
```

Example Generated sound.h File

```
// ADPCM HEADER FILE created by Neo Sound Builder

inline void send_sound_command(BYTE CommandNo)
{
    (*((PBYTE)0x320000)) = CommandNo;
}

// ADPCM System commands
#define ADPCM_OFF (0x7f)
```

```
// ADPCM Sample commands
#define ADPCM_SHUFFLE (0x10)
#define ADPCM_KNOCK (0x11)
#define ADPCM_THUNDER1 (0x12)
#define ADPCM_MUSIC (0x13)
```

This header file consists of a simple inline function that sends a byte value to the Neo's sound output register (0x320000). The Neo will then send the byte to the Z80 where the sound driver will accept the byte and play the associated sound number or perform the system command.

Currently there is only one system command for the Neobitz Z80 sound driver. The ADPCM_OFF command will stop playback of all samples. As the sound driver advances, new system commands may be added.