MIDIbox SD Card Polyphonic Sample Player

This is a brief description of a project in progress to play back samples in real time from an SD card to an I2S based DAC on the Midibox. It will play a number of samples simultaneously, mixing the output to the DAC, the number of samples depends upon the performance of the SD card.

Hardware

You will need:

- Midibox LPC1769 core (or STM32 now, but that has limited polyphony due to core speed) - see http://www.ucapps.de/mbhp_core_lpc17.html (or make a minimal setup like I did)
- I2C DAC (ie TDA1543 or similar) - see http://www.ucapps.de/mbhp_i2s.html but MAKE SURE YOU RUN THE TDA1543 DAC FROM 5V AS 3.3V SOUNDS BAD
- SD card interface - see http://www.ucapps.de/mbhp_sdcard.html
- SD card formatted as FAT (NOT FAT32 - important as FAT32 performance currently isn't fast enough) - this limits you typically to cards ⇐2GB
- MIDI data via USB or the standard Midi in

It has now been run on both LPC1769 and STM32 cores... 🙂

Creating files on SD card

As of rev 1371 on the SVN, it now reads a file called 'player.cfg' from the SD card which is a UNIX style file (there's an example in the SVN) which contains name value pair separated by a space. At present it only supports two parameters, eg:

```
midichannel 1
lee_hw 1
```

The first one (obviously) sets the midi channel for the player. The second value is to enable my simple rotary bank switch on J10 (you probably want this set to zero if you have a real midibox core, and want to use MIDI program change to select the bank).

The system currently looks for a file called 'bank.1' on the SD card root by default (and bank.2, bank.3 etc as you send a program change). This is a text file, but must have Unix style line feeds (not DOS style) - just use your friendly programmer's file editor to create one (or make it on a Mac 🙂). It will read up to 64 lines from this file, each line mapping a MIDI note, and the format is e.g:

```
0x39 0 0031 c_a.raw
0x3a 1 0000 c_bf.raw
```

where the first number is the MIDI note number in Hex, the single digit is 0=no sample hold,

1=sample hold (ie for drums etc), the 4 digit number is a decimal value for an envelope decay (0000=no decay, and up to 1023 is supported) and after the space is the filename of the sample to play (up to 8.3 characters).

It looks for the samples in the root of the SD card.

The sample format is fixed as: monophonic (1 channel) 44.1kHz 16 bit RAW signed little endian. Quite a mouthful. Basically use Audacity, Soundforge etc to convert whatever samples you have into this format.

Currently you have to provide a note mapping for each note you want to sound, and usually therefore you will need to generate a sample of the right pitch for each note. This is a lot of samples for a full key range, but hey hopefully you only have to do this once!

Other notes

- The performance of the SD card will alter how well this works. Remembered to only use FAT format? Good.
- Every 5.8mS, it will read data from the SD card depending on which MIDI notes are on. There's a built in safety net that stops reading more notes after about 4-5mS (otherwise we risk a crash).
- It is coded to play up to 8 notes polyphony, but depending on the card speed you may get only 5-6 from time to time...
- If you watch the debug output in MIOS Studio, it reports the samples loaded on initialisation and also if the safety net is kicking in

Latest features

- Now supports a config file on the SD card
- Now supports MIDI program change to load banks from the SD card (eg for patch 1 it will load bank.1, patch 2 it will load bank.2 etc)
- Envelope filtering and voice assignment now runs as a seperate MIOS task (increasing time available for sample read/mixing, increased responsiveness and more stability)
- We can now do a decay envelope, and this is specified for each individual samples (eg staccato / legato)
- You can do sample hold per sample, eg have a bank where drum sounds are on some notes, and other non-hold samples such as bass are on other notes
- It now supports velocity sensitivity, and this is translated to volume via a look up table
- Bank switching is supported on the fly (ie without a reboot) - though program change isn't there yet
- Thanks to TK the sector positions on the SD card are cached leading to much better stability with multiple notes sounding

To do

- Support more MIDI events (hold pedal, etc)
- Make the bank file parsing more robust, and support comments, invalid lines etc

Files

- Now uploaded to the MIOS32 SVN at:
- http://svnmios.midibox.org/listing.php?repname=svn.mios32&path=%2Ftrunk%2Fapps%2Fsynthesizers%2FSD+card+sample+player%2F

From:
https://wiki.midibox.org/ - **MIDIbox**

Permanent link:
https://wiki.midibox.org/doku.php?id=midibox_sd_card_sample_player&rev=1324597413

Last update: **2011/12/22 23:43**