FluidSynth

Adding Sostenuto pedal to FluidSynth

jean-jacques ceresa

- 27/02/2015 0001-Add-sostenuto-pedal-functionality.patch - obsolete
- 11/03/2015 0002-Add-sostenuto-pedal-functionality.patch
- 22/03/2015 More détails on specification 4 (1.2.1) : 0003-Add-sostenuto-pedal-functionality.patch
- 29/03/2015 Better implementation of specification 4 (1.2.1): 0004-Add-sostenuto-pedal-functionality.patch

1. Adding sostenuto pedal

In FluidSynth (v 1.1.6) only Sustain pedal is supported. Sostenuto pedal is ignored.

If you have no idea of what sostenuto pedal is you could read chapters 1.1, 1.2, and chapter 1.3 that gives instructions on how to play Sostenuto and Sustain pedals.

Chapter 1.4 gives detailed explanations to add support code fin fluidsynth or Sostenuto.

1.1. Sustain pedal behavior.

Talking about Sustain pedal, musicians often say "the damper pedal" or "pedale forte".

When a musician play a key note on accoustic instrument, the sound release quickly when he release the key note.

Using sustain pedal, note continues when musician release key note. Depending of the instrument , sound take more or less time to release.

The advantages are:
- The sound resonates among.
- The musician can enhance easily a legato playing. This enhancement is particularly necessary on keyboard instruments that naturally aren't favorable to legato playing.

The disadvantages are:
- If musician insists too (i.e he forgets releasing the pedal), the sound may increase each time a note is sustained.
- Also , the musician must be careful with a lot of dissonances introduced very easily by sustained notes.

So, using the sustain is beneficial when used sparingly and carefully.

1.1.1. Sustain behavior

To sustain note, a musician must depress the Sustain pedal before he releases the key note. (No matter if pedal is depressed before or after this key note).
As the pedal is depressed, any note is sustained when the musician releases the key note. Any following notes will be sustained too. Alls sustained notes will be released when Sustain pedal is released.

The best way is to try this immediately.

1.1.2. FluidSynth specifications

Specification 1
- On noteOff event, note is sustained if sustain pedal is depressed.

Specification 2:
- When sustain pedal is released, alls notes sustained will be released.

1.2. Sostenuto pedal behavior.

Talking about Sostenuto pedal, musicians often say "the tonal pedal".

Sostenuto behavior is similar to sustain, so when sostenuto is depressed, any note is sustained when a key is released, however the purpose of this pedal is to choose some notes to sustain and others notes that will not be sustained.

This is the point of interest of sostenuto pedal. Only notes depressed before sostenuto pedal will be sustained, following notes will not be sustained. As a musician you can consider the notes played before sostenuto pedal as bases note of your chord (consonants notes). Following notes could be extensions of your sustained chord. That means that these following notes can be temporarly introduce dissonances with current sustained chord and the musician can control easily duration of these dissonances.

As you see, a musician have a better control on notes played after the sostenuto pedal as these will not be sustained by sostenuto pedal but can be sustained and released later by sustain pedal. This can be done independently of notes sustained by sostenuto pedal. That means that sustain pedal can be changed while the sostenuto pedal is down without affecting the notes held by the sostenuto pedal.

So the musician may think of two independant groups of notes, one sustained/released by sostenuto and the following one not sustained or sustained/released by sustain pedal !.

1.2.1. FluidSynth specifications

Specification 3

On noteOff event, if sostenuto pedal is depressed

- (1) Notes depressed before Sostenuto pedal will be sustained.
- (2) Notes depressed after Sostenuto pedal will not sustained (see exception below in spec 4).

If both sostenuto and sustain pedal are both depressed, sostenuto has precedence over sustain. That means that:
- notes depressed before sostenuto are sustained by sostenuto and not by sustain (FLUID_VOICE_HELD_BY_SOSTENUTO).
- notes depressed after sostenuto are sustained by sustain (FLUID_VOICE_SUSTAINED) (see spec 6).

Specification 4:

- When a note is currently sustained (by sustain or sostenuto), playing a same note stops the previous note and starts a new note.
  If the previous note was sustain by sostenuto, on noteOff the new note will keep a sustained state (even if Sostenuto pedal is depressed before the new note is released).

Specification 5:

- When sostenuto pedal is released, alls notes sustained by sostenuto (FLUID_VOICE_HELD_BY_SOSTENUTO) will be released.

Specification 6:

While notes are sustained by sostenuto, next notes aren't allowed to be sustained by sostenuto but can be sustained by sustain.
This means than while notes are sustained by Sostenuto , a musicien can add more sustained notes (FLUID_VOICE_SUSTAINED) using sustain pedal temporariy. Those later notes (FLUID_VOICE_SUSTAINED) will be released when sustain will be released and former notes continue to be be sustained by sostenuto. Those former notes (FLUID_VOICE_HELD_BY_SOSTENUTO) will be released when sostenuto will be released.

1.3. How to play Sustain or Sostenuto pedal easily ?.

There is a simple and unified way to play Sustain or Sostenuto pedal. For example to play a sustained chord follow the instructions below:

1) Play your chord (keys down)
2) Play your pedal down (sustain or sostenuto) and release your chord (keys up).

Now you chord is sustained (depending of the choosen soundfont instrument).

Each time you want to change chord you may need to use the pedal (sustain or sostenuto) so that next chord overlaps previous chord ending. This is called legato playing.
3) Play your next chord (keys down).

4) (a)Release the pedal (this release previous chord). (b)Depress your pedal down and (c) release your chord (key up). (Execute steps 4.a, 4.b, 4.c consecutively).

So for each chord change you need to do steps (3) and (4.x) consecutively thus:
(3,4.x)….(3,4.x)….(3,4.x)…

3) Now play following notes. These are not allowed to be sustained by sostenuto pedal but you can use sustain pedal to sustain/release these notes.

1.4. How to add Sostenuto code in FluidSynth ?

Adding code to support Sostenuto in FluidSynth is easy and easy to verify. All is based on the previous specifications decribed in previous chapter (see 1.1.2, 1.2.1).

- Chapter 1.6 explains a correction on *fluid_synth_release_voice_on_same_note_LOCAL()* function behavior. Note that this correction is not related to adding sostenuto support. It is only a chance to correct a bad behavior as we need to add code for sostenuto in this function.

- Patches are to be done on following files (version 1.1.6):
  fluid_chan_1.1.6.h, fluid_chan_1.1.6.c,
  fluid_voice_1.1.6.h, fluid_voice_1.1.6.c,
  fluid_synth_1.1.6.h, fluid_synth_1.1.6.c,

  Using following diff results:

  diff -Nur ./Fluid_1.1.6 ./Fluid_1.1.6_sost > 0003-Add-sostenuto-pedal-functionality.patch

- Chapter 1.7 explains how (step by step) to add Sostenuto code. This helps reading .diff files.
- To verify the behavior you need Sustain pedal and Sostenuto pedal. Chapter 1.5 gives you an handy way to simulate a sostenuto pedal using a sustain pedal.

1.5. Using fluidsynth router to simulate a sostenuto pedal by Sustain pedal

1.5.1. Verifying Sustain behavior

Before verifying Sostenuto you may prefer to verify Sustain behavior. Please follow specifications in 1.1

1.5.2. Verifying Sostenuto behavior

In the case of only a sustain pedal is available, you don't need to buy a sostenuto pedal to try sostenuto effect. You can instruct FluidSynth MIDI router to transform a MIDI sustain event to a MIDI Sostenuto event.
Using fluidsynth application, you need to enter the following commands in the shell to instruct the router.

 # Remove current rules (to remove cc sustain events):
router_clear
 # Set the rule to transform CC sustain (64d) to CC Sostenuto (66d)
***router_begin* cc**
router_par1 64 64 0 66
router_end
 # Set the rules to pass through other messages types (note, prog, pbend, cpress, kpress)
router_begin note
router_end
router_begin prog
router_end
router_begin pbend
router_end
router_begin cpress
router_end
router_begin kpress

router_end

To verify sostenuto behavior, please follow specifications in 1.2.

1.6. fix in fluid_synth_release_voice_on_same_note_LOCAL()

The purpose of **fluid_synth_release_voice_on_same_note_LOCAL()** is to force the previous note to enter the release stage.
This is done by calling fluid_voice_noteoff(). However in fluid_voice_noteoff(), behavior is pedals (sustain,sostenuto) dependent. For example, while sutain pedal is depressed if the same note is depressed several times, previous note remains sustained rather to be released.

So as we need to keep **fuid_synth_release_voice_on_same_note_LOCAL()**, pedals independent, the solution is to call only the part of code to force release. See chapter 1.6.1 for details.

Note: The same correction applies to other functions like:
- fluid_synth_damp_voices_LOCAL()
- fluid_synth_stop_LOCAL()

1.6.1. Steps to fix fluid_synth_release_voice_on_same_note_LOCAL()

The following table gives details.
- the code in **bold** need to be added.

to do	comments
done	In *fluid_voice.h* add : **/* *** fluid_voice_release** *** Force the voice into release stage. Usefuf anywhere the voice** *** needs to be damped even if pedals (sustain sostenuto) are pressed.** *** See fluid_synth_damp_voices_LOCAL(), fluid_voice_noteoff(),** ***/** **void fluid_voice_release(fluid_voice_t* voice);**

done	In *fluid_voice.c* add the function :
	**/*
	*** fluid_voice_release**
	*** Force the voice into release stage. Useful anywhere the voice**
	*** need to be damped even if pedals (sustain sostenuto) are pressed.**
	*** See fluid_synth_damp_voices_LOCAL(), fluid_voice_noteoff(),**
	***/**
	inline void fluid_voice_release(fluid_voice_t* voice)
	{
	unsigned int at_tick = fluid_channel_get_min_note_length_ticks (voice->channel);
	UPDATE_RVOICE_I1(fluid_rvoice_noteoff, at_tick);
	voice->has_noteoff = 1; // voice is marked as noteoff occured
	}

done done done	In fluid_synt.c, replace the call to *fluid_voice_noteoff () by a* call *to fluid_voice_release()* in the following functions: • fluid_synth_release_voice_on_same_note_LOCAL(). • fluid_synth_damp_voices_LOCAL(). • fluid_synth_stop_LOCAL().

1.7. Steps to add sostenuto code

The following table gives details.
- Changes or adding are in **bold**.

To do	comments

done

In fluid_voice.h
In enum fluid_voice_status: add **FLUID_VOICE_HELD_BY_SOSTENUTO**

Add this macro
#define _HELD_BY_SOSTENUTO(voice) ((voice)->status == FLUID_VOICE_HELD_BY_SOSTENUTO)

done

Change this macro

```
#define _PLAYING(voice)  (((voice)->status == FLUID_VOICE_ON) || \
                                  _SUSTAINED(voice)  || _HELD_BY_SOSTENUTO(voice) )
```

done

In fluid_synth.c , fluid_synth_release_voice_on_same_note_LOCAL()

```
unsigned int storeid = -1;  /* Id of previous note sustained by sostenuto */
for (i = 0; i < synth->polyphony; i++) {
{
    if (_PLAYING(voice) &&.....)
          && (voice->chan == chan)
          && (voice->key == key)
          && (fluid_voice_get_id(voice) != synth->noteid) ) {
       /* Id of voices that was sustained by sostenuto */
       if( _HELD_BY_SOSTENUTO(voice)) storeid = voice->id;
        /* Force the voice into release stage (pedaling is ignored) */
       fluid_voice_release(voice); /* Release voice */
    }
}
/* Set normal new voice id */
if (storeid == -1)   synth->storeid = synth->noteid++;
/* New voices id will be same as that of voices that was sustained by sostenuto */
else synth->storeid = storeid;
```

done

In fluid_synth_noteon_LOCAL(fluid_synth_t synth, int chan, int key, int vel) remove:*

done

~~synth->storeid = synth->noteid++;~~

In fluid_voice_get_overflow_prio(), change:

```
…..
} else if (_SUSTAINED(voice) || _HELD_BY_SOSTENUTO(voice)){
…..
```

done

In fluid_voice.c (fluid_voice_noteoff ()) add:

```
/* Sustain a note under Sostenuto pedal */
  if (fluid_channel_sostenuto(voice->channel) &&
              chan->SostenutoOrderId > voice->Id) { /* Sostenuto pressed after note */
    voice->status = FLUID_VOICE_HELD_BY_SOSTENUTO;
  }
  else if( fluid_channel_sustained(voice->channel)) {
    voice->status = FLUID_VOICE_SUSTAINED;
  }
else {  ....}
....
```

done

In *fluid_chan.h (*struct fluidchannel_t) add :

```
/*Sostenuto order id gives the order of SostenutoOn event .
This value is useful to known when the sostenuto pedal is depressed
(before or after a key note). We need to compare SostenutoOrderId  with voice id.
*/
unsigned int  SostenutoOrderId;
```

done

In fluid_chan.h add this macro:

```
#define fluid_channel_sostenuto(_c)          ((_c)->cc[SOSTENUTO_SWITCH] >= 64)
```

done

In fluid_chan.c (fluid_channel_init() add:

```
chan->SostenutoOrderId = 0;   /*  Reset Sostenuto order id */
```

done

In fluid_synth.c , add function

```
/* Damp all voices sustained by sostenuto on a channel (turn notes off) */
static int
fluid_synth_damp_voices_ by_sostenuto_LOCAL(fluid_synth_t* synth, int chan)
```

done

```
{
  fluid_voice_t* voice;
  int i;

  for (i = 0; i < synth->polyphony; i++) {
    voice = synth->voice[i];

    if ((voice->chan == chan) && __HELD_BY_SOSTENUTO(voice))
      fluid_voice_release(voice);
  }
  return FLUID_OK;
}
```

In fluid_synth.c (fluid_synth_cc_LOCAL())

```
case SOSTENUTO_SWITCH:
   /* Release voices if Sostetuno  switch is released */
    if (value < 64) /* Sostenuto is released */
    {
          fluid_synth_damp_voices_by_sostenuto_LOCAL (synth, channum);
    }
    else /* Sostenuto is depressed */
          // Update sostenuto order id when pedaling on Sostenuto
        chan->SostenutoOrderId = synth->noteid; /* future voice id value */
    break;

case SUSTAIN_SWITCH:
   /* Release voices if Sustain switch is released */
   if (value < 64 ) /* Sustain is released */
      fluid_synth_damp_voices_LOCAL (synth, channum);
   break;
```

Et voilà.