Project Proposal: Loop Station

Evan Ziebart (erz2109)
Doga Ozesmi (do2330)
Lancelot Wathieu (lbw2148)
Varun Varahabhotla (vv2282)

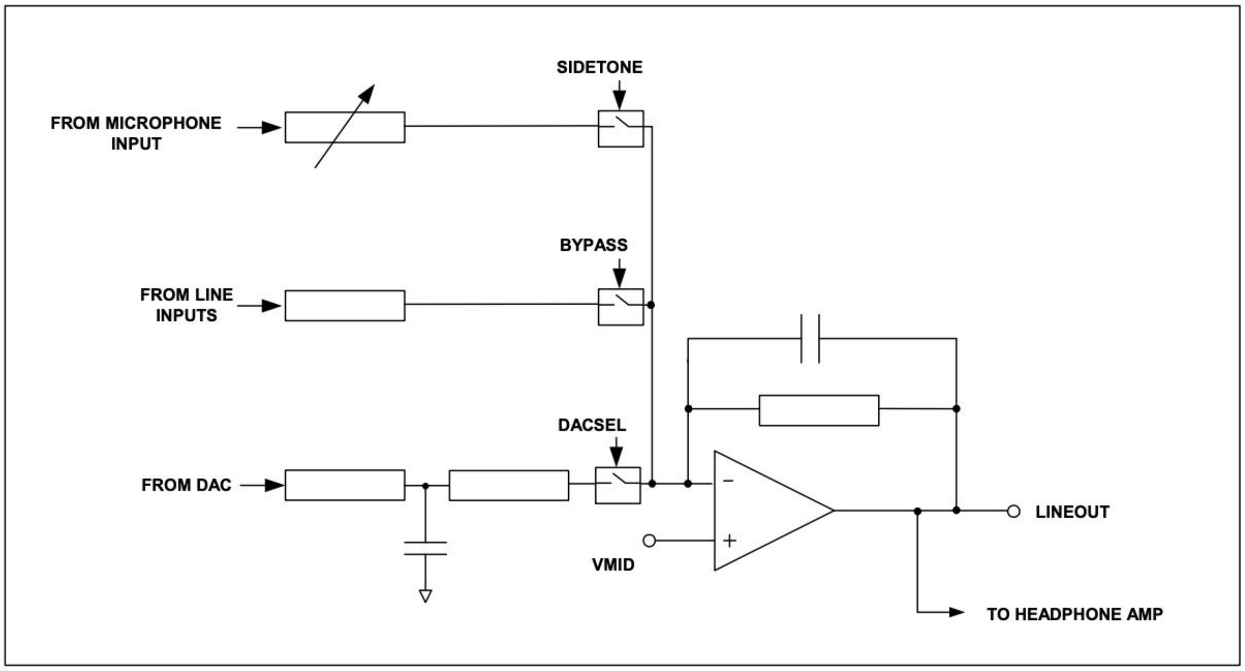
Overview: A loop station. Record audio from a microphone and electric keyboard and then play them back together to build a full soundtrack. Unlike in a typical loop station which dubs a loop onto a main track ("sound-on-sound") we will keep the tracks separate so they can be individually added or removed. Live playback may also be included so that you could make a backing track and then sing over it without recording.

I'm seeing if I can find any relevant images… Also, this should be formatted nicely and everything
References: recording and playing audio and karaoke machine examples of DE1 user manual (Section 5.2 and 5.3)

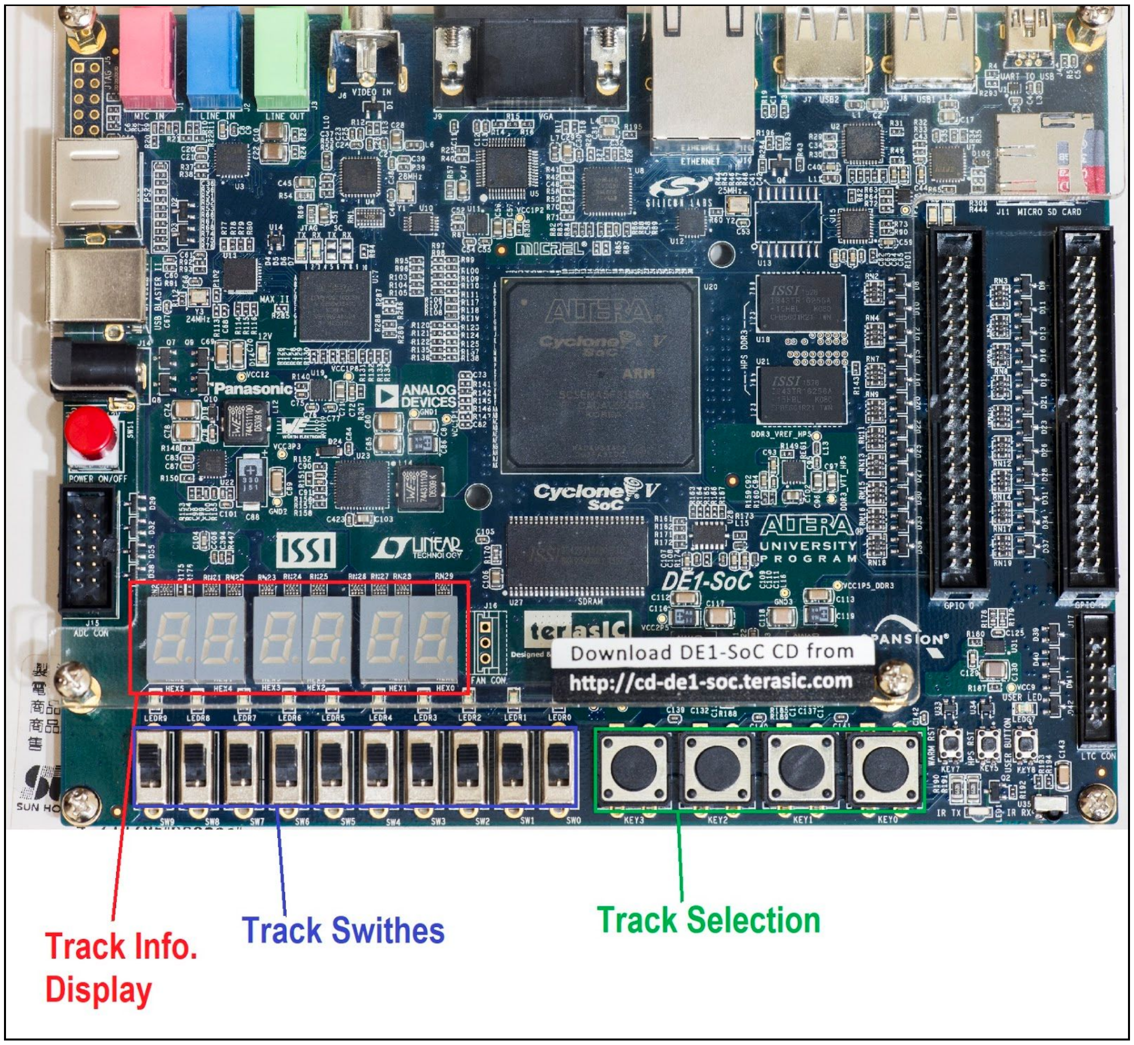
Controls:
Each slide switch will enable or disable a particular audio track
2 push button to record audio (1 for mic in, 1 for keyboard in) - press to record, press to stop recording. Press and hold for ~5 seconds to delete track
2 push buttons to turn volume up/down
Switches on the FPGA to enable the tracks we recorded
Current track number will be displayed on 7 segment display
LEDs will be utilized as needed for indicators, for example one will be on while recording.

Hardware Tools:
Audio Codec chip on the FPGA - WM8731
Mic connected to the MIC_IN line on Audio Codec
Optional mp3 device connected to LINE_IN on Audio Codec
Speakers connected to LINE_OUT on Audio Codec
MIDI Instrument can be connected via USB (eg: keyboard) (later translated to digital audio)
MIDI digital audio signal (translated to audio by software synthesizer) to DACDAT on Audio Codec

Memory to buffer audio during recording and playback.
Audio Controller to handle I/O with the Audio Codec.
SDRAM Memory controller to handle reading and writing audio in buffer



WM8731 Audio Codec Figure - Input from Mic, MIDI, and possibly mp3 device



Overview of hardware interface on device

Software Tools:

USB Driver for Yamaha YPG-635 keyboard

Synthesizer Software -- translate MIDI protocol coming from the keyboard to digital audio signal. Possibly TiMidity++ or Glitch by NaiveSound, both open-source linux-compatible MIDI synthesizers.

Linux device driver to get synthesizer to send bits to Audio Codec

Tasks:

1. Save and retrieve audio received from MIC_IN - makes use of the SDRAM
2. Send mic audio to speakers through the LINE_OUT port
3. Set up a series of audio tracks which can be switched on and off using the sliding switches. This will require a special device driver to communicate with the switching logic on the FPGA.

4. FPGA logic to switch tracks on and off: debouncing switches and buttons, tracking and displaying the currently selected track, and setting flags to indicate whether the software should play a certain track or not. Provide information about the current state on the 7-segment display
5. Mapped IO from MIDI instrument. In general, MIDI data should be sendable form any USB compatible instrument, which will include most electronic keyboards, provided the correct device driver is installed. For our purposes, we will download and utilize a device driver for a Yamaha YPG-635 keyboard.
6. Store MIDI data from USB instruments in temporary track storage alongside the mic tracks. These will be in MIDI format
7. Using a synthesizer program on Linux, convert MIDI data into audio which can be played over the speaker similar to the mic data.