Synth engines
We will implement two synth engines, one for subtractive synthesis and one for granular synthesis.

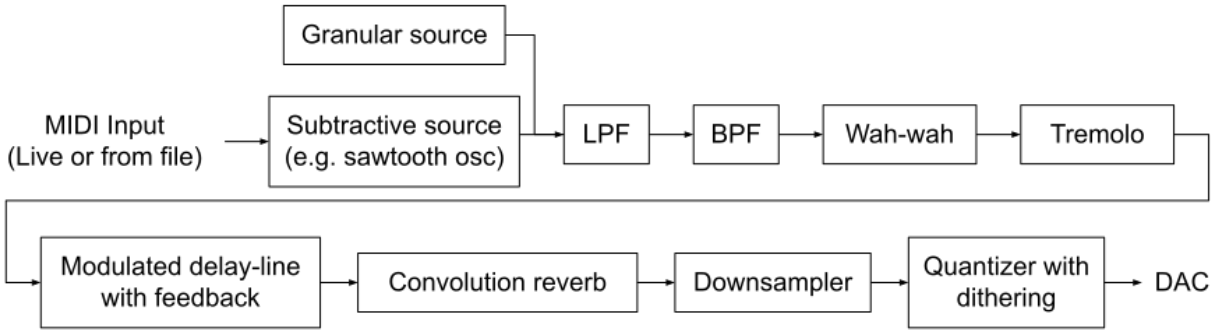
Effects
We will implement a reverb effect using convolution. We will choose the particular impulse response at implementation time (and, if time allows, we may make this user-configurable).

For our modulated effects, we will implement a wah-wah effect, tremolo, and modulated delay-line with feedback (suitable for vibrato, flanging, etc. depending on parameters).

Filters
We plan to implement the State-Variable Filter. This will give us access to several types of filters in one implementation, and we will include at least two of these (lowpass and bandpass) in the signal chain and allow the user to configure them.

Input/Output
We will take musical data as MIDI; the user will be able to supply a MIDI file or a connected MIDI device as the source of input. We will stream audio output in real time (and if time allows, we may also allow the user to record the output to a file).

Signal Chain



The order of the modules in the chain is subject to change during implementation.

Interface
We will provide a command-line interface where the user can get and set the values of all (user-exposed) parameters. The user will be able to modify parameters while the system is running.

Libraries
For MIDI input (from files and devices), we will use Mido. For streaming audio output, we will use sounddevice. If time allows for the other extensions described in extra credit, we may also use libraries such as pyOSC (for OSC) and PySide (for a GUI).

Extra Credit
As noted above, we intend to allow live musical input with live streaming output. (We may allow for latency-performance tradeoffs by letting the user select the blocksize.)

If time allows, we would like to enable control from other applications (such as Max or Pure Data) by having the system listen for OSC messages, and we would like to allow the user to change the signal chain (order and topology) at runtime, which may require a GUI to be usable.