# Block Requirements and Verifications

## Software Interface

| Requirement | Verification | Points |
|---|---|---|
| Display menus, interfaces and user input fields appear on the screen, without error and can be updated appropriately. | 1. Input values into each user input field.<br>2. Set generators using these input values<br>3. Print out the generator values to ensure the values are being properly parsed by the code and stored in the designated buffer. | 5 |
| All menus must be togglable and feature each option allocated by the design.<br><br>*(cont'd)*<br>All menus must be togglable and feature each option allocated by the design. | 1. Click on each option in the drop-down menus on the interface.<br>2. Ensure all features are listed within the drop down function.<br>3. For drop-down functions that alter the screen, select these functions and ensure that the screen makes the necessary alterations.<br>4. For each drop down function, set the generator. Include random sample values for other parameters.<br>5. Print contents of the updated generator to ensure each feature selection is linked to the proper variable in the code that gets stored. | 5 |

## Microcontroller

| Requirement | Verification | Points |
|---|---|---|
| Produces no aliasing for any oscillator waveform and all notes up to C8 - highest note on a grand piano - with frequency 4186.01 Hz | 1. Display waveform with frequency >= 4186.01 on oscilloscope.<br>2. Rotate the horizontal scale knob, or otherwise change the horizontal scaling of the oscilloscope. If the waveform changes drastically, aliasing is present | 6 |

Wavetable Synthesis

| Requirement | Verification | Points |
|---|---|---|
| The wavetable synthesis program must not alter the single-pass waveform stored in the buffer. While not altering what's stored in the buffer, it also must use the buffer and properly process the waveform at the desired frequency to produce the correct audio signal. | 1. Store signal of single amplitude in the wave_form buffer.<br>2. Pass through the wavetable synthesis algorithm at a single frequency.<br>3. After passing the wave_form buffer through the algorithm 1000 times, check the buffer has not been altered<br>4. Ensure that the output signal is a single-tone audio output of a single-level. If fluctuations in pitch or frequency are detected, there is an error. | 10 |

Digital-to-Analog Interface

| Requirement | Verification | Points |
|---|---|---|
| 3.3Vp-p (+/- 0.3V error) digital waveforms at the input to the DAC during operation. | 1. Using an oscilloscope, ensure a digital version of the intended waveform is present at the input pin and conforms to the desired standard | 4 |
| 1.4Vp-p output (+/- 5% error) continuous Analog Output signal at DAC output pins | 1. Set to ready/inactive (through the GUI or underlying program if the GUI is not yet debugged) a simple output waveform to the GPIO connected to DAC1<br>2. Connect the I2C_SCL (PB10) and I2C_SDA (PB11) outputs of the pyBoard to the corresponding I2C inputs on the LMP92001 DAC<br>3. Ensure, through LMP92001 address bits 31:32, and by setting the I2C configurations in the underlying program to mirror the address of 31:32, that you are communicating with the proper DAC<br>4. Measure output at the corresponding output pin using a digital oscilloscope and ensure it conforms to the given requirement<br>5. Repeat for all DACs | 4 |
| Must properly convert any input digital signals to audio signals. | 1. Pass a single-tone audio signal into the DAC.<br>2. Test the audio it produces digitally<br>3. Connect the output of the DAC to an analog speaker.<br>4. The audio signal produced by the analog speaker must match that digitally produced during the aforementioned test.<br>5. Repeat with varied audio signals. | 6 |

Routing Matrix

| Requirement | Verification | Points |
|---|---|---|
| Verify that ADG2128 Chips Pass Signal Through All Connection Points | Test chips individually.<br>1. Manually verify that test DC voltages of +/-5V are passed through each of the 96 ports when operated individually on a +/-12V power supply.<br>2. Manually verify that a +/- 5V sine wave passes through all ports when operated on a +/- 12V power supply. | 2 |
| Verify that signals on separate ADG2128 chips carrying the same buffered output to different buffered inputs have no interaction with one another, or that signal interaction occurs at a low enough level as to be negligible for the listener/user (difference in peak-to-peak voltage of less than 2% from unconnected waveform) | 1. View signals on an oscilloscope individually, without the other connected. Measure peak-to-peak voltages.<br>2. View signals on the oscilloscope while both signals are connected. Ensure minimal or no change | 2 |
| Verify that user input into the routing matrix via Max/MSP produces the correct output hex and the correct number of output hexes | 1. Randomize switches that user sets in Max/MSP<br>2. Send serial message containing number of switches to be set and the matching I/O ports into the python code to be parsed<br>3. Print out the message to be passed over the SDA line by $I^2C$ for each switch to be set.<br>4. Verify that signals match the corresponding values in user datasheet<br>5. Run test minimum 5 times with varied number of switches, ensure 100% accuracy | 6 |