Pre-Alpha Build

External Interface

For the external interface, we have five main parts. The RCA input, the RCA output, the HDMI output, the USB output, and the user interface.

The RCA input will be conveyed through three signal processors, one for each cable, which will output raw binary data to be fed directly into the internal systems.

However, the RCA output will completely skip internal processing, and will be bridged directly from the RCA input.

The HDMI and USB outputs will be the result of outputs from the internal systems processing the digitized RCA input. Both of these will be simple connectors on the board for the user to connect to.

Finally, the user interface, consisting of a button, switch, and LED, will be simple binary inputs and outputs connected to the top level of the VHDL code to control the basic operation of the device, and allow for the user to monitor the device's state (on or off).

Persistent State

Our device won't have a persistent state in between power cycles, however, it might have a persistent state in the form of a buffer to hold data transmitted from the video decoder to the FPGA within a power cycle.

Internal Systems

The internal systems are the main focus of this project. The device will have two major processing aspects. The first will convert the RCA signal into an HDMI output, and the second will convert the RCA signal to a USB output. Each of the USB and HDMI components will be run by separate finite state machines in order to properly utilize and transfer the data according to each respective protocol.

For the USB output, we will need to format the data into USB protocol packets in order to easily transmit them. For video, this will mean cycling over the input data to output them as "VideoParticle" packets to render them onto the screen. These packets can take YCbCr data as a datatype, so no conversion will be necessary for the USB device to output the proper packets.

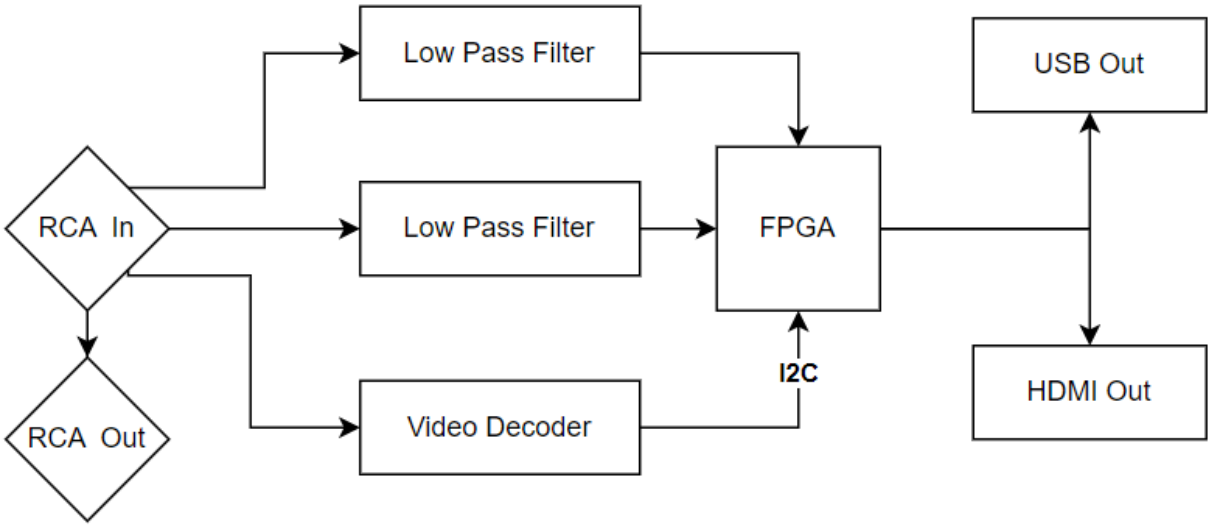
For the USB audio output, we will sample the input from the cable and send AudioSample packets over USB. A sufficient number of these sent together at the specified time should recreate the correct audio signal on the other end, with a high enough sample rate and output word size. However, we may need to compromise one for another.

For the HDMI output, the internal system will take the digitized RCA signal and encode it by reducing the number of transitions between on and off and DC balancing the signal. The algorithm used to encode this signal is called Transition Minimized Differential Signaling, or TMDS for short. This process also includes inverting the signal to be sent along the HDMI output port, which is required for the receiving device to decode the HDMI signal, and minimizes data loss over the transmitting cable.

Communication

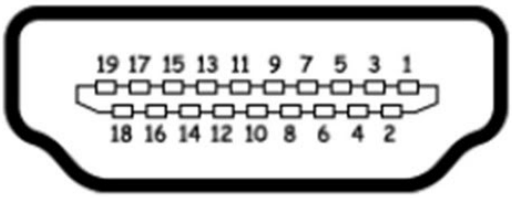
For communication between internal parts, we will simply use internal signals to hold the values of the YCbCr value, as well as the audio packets. For communication between the signal processors and the main board, we will likely use I2C based on our current prospective choices, though SPI is an option if we need a higher speed. I2C should be fast enough to reach the Nyquist frequency of the analog frequencies, but in the case that it doesn't, SPI will definitely reach the rate, though it may need more specialized parts and cause the PC to be more expensive to construct.

As for the button, switch, and LED, they will all be single-bit inputs and outputs to control the device. They should all only communicate with the main board, and won't connect directly to any other external parts.



Integrity & Resilience

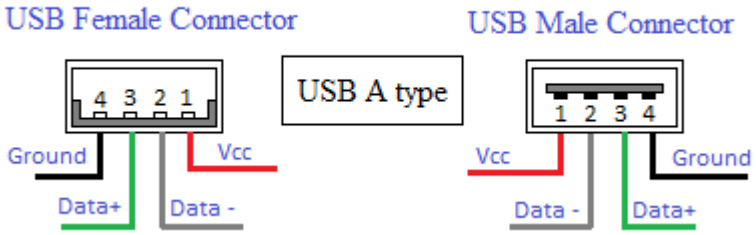
Most of the information we produce is sent out over HDMI or USB. Both HDMI and USB have protections against interference, as most of the data transmitted are sent in differential signal pairs which are negatives of each other. When merged back together, these pairs account for and eliminate benign interference that occurred along the way, as they are referenced against each other, rather than ground. The signals also travel in twisted pairs, which prevents both of the signals from being affected by interference, ensuring that one of the wires can eliminate interference from the other.



Pin#	Signal	Pin#	Signal
1	TMDS data 2+	11	TMDS clock shield
2	TMDS data 2 shield	12	TMDS clock-
3	TMDS data 2-	13	CEC
4	TMDS data 1+	14	No connected
5	TMDS data 1 shield	15	DDC clock
6	TMDS data 1-	16	DDC data
7	TMDS data 0+	17	Ground
8	TMDS data 0 shield	18	+5V power
9	TMDS data 0-	19	Hot plug detect
10	TMDS clock+		

HDMI also protects against faulty information conveyance by minimizing the transitions between high and low, making degradation of the signals less likely. The signals are also DC-balanced, which makes the signal resist switching bits.

As our device does not connect to any network, malicious intrusion should not be an issue.



USB-A 2.0 plug

Above is a pinout of the USB 2.0 plug that we intend to include as an output. The most notable signals are the Data+ and Data- pins, which represent the differential signals that packets will be sent with.