Test Document
By Arlen Cox and Robert Hasegawa

Video System

Analog Output
The analog output will be tested in several parts. After construction, the output of each DAC channel will be tested against specifications by latching in low and high signals on each channel and taking a reading on a multimeter or oscilloscope. The values should range from 0 to 714mV on each channel.

If those signals are to specification, then the output of the composite signal will be checked by inputting a variety of different values into the RGB channels and then using a function generator to create the sync signal. The output will be compared with the NTSC specification, particularly for black and white inputs as these are simple to check on an oscilloscope.

Timer
The timer will be tested first by simulation in a logic simulator. The timer is a simple circuit, so a simple clock input should be all that it needs. It should create a pixel clock and sync signals. Once implemented, the circuit will be verified by testing the frequencies of the pixel clock and the sync signals using an oscilloscope.

Palette
The palette may be implemented in off chip RAM, and if that is the case, the RAM chip is the entire Palette system. The RAM chip will be tested by setting a clock signal and writing data to the RAM and then verifying that the data can again be retrieved. This may be best done using a microcontroller to send and receive the bits.

FrameBuffer Interface
The framebuffer interface will be tested first in logic simulation. It will be tested by providing a pixel clock and a reset value and verify that it correctly counts through the proper addresses. It will then be implemented and tested by inputting the proper clock signals and verifying the output on some sort of logic analyzer. It will be tested a low clock rates to ensure functionality and then tested at high clock rates in simulation to ensure that transient conditions don't cause problems.

Memory Interface
The memory interface will simply be tested to ensure that if the proper address lines are set, the correct data reads out. This will be similar to the Palette tests.

Video Processor
The Video Processor will be tested in several stages. First a software model of the video processor will be produced. This will be used to verify algorithms correct by rendering images on a PC. Then using that model, a test set of instructions will be generated and the corresponding control signals calculated for using in logic simulation. It will be functionally verified in a simulator. After functional verification, it will be implemented
and tested a slow clock rates for functionality using a logic analyzer. It will then be tested in simulation at high clock rates for transient behavior.

**Control**
Control should be tested simply by using the same process for other digital elements. It will be simulated for all potential test cases and verified that logic is correct. It will then be implemented and tested with a logic analyzer at low clock rates for functionality. It will then be tested for transient conditions using a simulator before final implementation.

**Framebuffer, Palette, Timer and Analog Output**
These will be tested together using a ROM input instead of the SRAM framebuffer to ensure that correct images can be displayed. A ROM will be programmed by a HEX file and will be placed where the RAM would be placed. The pixel clock will be slow enough for this to work with a fast ROM. The digital portion will be tested together using a simulator.

**Control, Memory Interface and Video Processor**
These will be tested together by running a sample set of instructions into the input. This set of instructions will generate data into a RAM which later can be polled to ensure correctness. This will also be tested all together using simulation.

**Audio System**

**Analog Circuit**
Test will be conducted to verify the operation of the analog circuit design. A digital function generator will be used apply a square wave into the DAC of the function generator. The output will be measured on an oscilloscope to check the accuracy of the DAC conversion circuit. The output square waveform should meet desired output characteristics. When the audio processor is finished, further testing will be done using the actual processor.

**Waveform Generator**

**Square Wave:**
Test will be made to verify the performance of the square function generation algorithm implemented within the audio processor. The algorithm will be implemented within the audio processor and set to output at a changing rate of frequency. When connected with the analog circuit, the output will be viewed and checked to meet desired output characteristics. A test can also be completed varying amplitude instead of frequency.

**Triangle Wave:**
Test to verify the performance of the triangle function generation algorithm. This test will be similar to the square wave test except with using the triangle wave generator instead.

**Sawtooth Wave:**
Test to verify the performance of the sawtooth function generation algorithm. This test
will be similar to the square wave test except with using the sawtooth wave generator instead.

Noise:
Test to verify the performance of the noise generation algorithm. This test will be similar to the square wave test except with using the noise generator instead.

Raw Audio:
Test will be made to verify the ability of the audio processor to pull a raw waveform from memory and use as output. A raw sound clip will be loaded into a memory module and the audio microcontroller programmed to use the raw audio algorithm to pull the sound clip from the memory and output it. The output can either be hooked up to a speaker or viewed on an oscilloscope to verify that the raw sound was output.

**Mixing Function**
Test will be made to check the performance of the digital mixing algorithm. Two or more wave functions will be programmed to be implemented within the audio microcontroller. Also programmed will be the algorithm to mix the various wave functions together, and then output to the analog circuitry. Either a speaker or oscilloscope will be used to verify that the functions are being mixed correctly. Various combinations of waveforms should be tested to verify the functionality of the mixer.

**Command Function**
Test will be made to check the performance of the command control functions of the audio processor. The full processor will be implemented and connected to a second processor programmed only to output a series of bits corresponding to the control function logic. The output of the processor will be fed into the analog circuit and the output monitored to verify the full functionality of the audio processor design.

**Game**

**Pong**
This can be tested by inputing many tests cases into the controllers. This can more or less be tested by random usage by many people pushing buttons at random intervals. If the game continues to operate under these conditions, it is working, otherwise, if all other systems are working, game code should be debugged.

**Central Processing Unit**

**Memory Interface**
Test to check the ability of the main microprocessor to interact with external memory. The microcontroller will be programmed with the ability to interface with a ROM and write the data on the ROM to attached RAM. The ROM will be loaded with different values and the test will be made to confirm that the RAM was written to with the correct values.
Input Interface
Test to check the ability of the main microprocessor to receive and processes external input. The microcontroller will be programmed with the ability to receive information from the gamepad controllers. It will also be programmed to output a high or low on a port pin depending on what it has detected for the button status on the controller. The pins can be wired to an LED circuit and button status seen visually.

Bootloader
Test to check the ability of the main microprocessor to initialize itself from startup to running of the program found on the ROM. The bootloader functionality will be implemented within the microcontroller and connected to a programmed ROM. The test will be to see if the bootloader function can take the data from ROM and write to its RAM and then begin processing the simple program found on the ROM.

Video Interface
Test to check the ability of the main microprocessor to interface with video hardware. The microcontroller will be programmed to output a series of data to the video hardware and checked to ensure that it outputs the data with proper form and clock rate.

Audio Interface
Test to check the ability of the main microprocessor to interface with the audio hardware. This test will be similar to the video interface test except that the microcontroller will be interfacing with the audio hardware instead.

Trig Functions
These can each be tested by comparing output to a known value from a calculator or a book. These can be debugged using a microcontroller debugging mechanism. The speed can be tested by testing the number of instructions/clock cycles needed to run them.

Other Math Functions
These are similar to the trig functions. They can all be compared to table look-up values to ensure their correctness. The speed of these functions can be measured by testing the number of cycles needed to run, or by using a random input and running many times through and dividing by the number of times through to get the speed.

Audio API
This can be tested by inputting all the different test cases to the function and ensure that the proper outputs are generated. This can algorithmically be verified by running in a simulator and can be physically verified with a logic analyzer.

Input API
The Input API can be tested by inputting a series of known events into the input system and verifying the results returned from this API. This is dependent on the input interface being implemented and functional.

Video API
This can be tested similarly to the Audio API. A series of test cases can be verified in
software and then physically tested using a logic analyzer. This is dependent on the video interface being tested and working. The error cases should be tested as well. These functions should never generate bad instructions.