FPGA Classic Arcade Tester - Copyright 2024, Fred Cawthorne

Initial Draft 7/19/20 1.0 BETA software
Updated to support new version hardware 10/11/22
Added comments about Z80 2/12/23
Updated for software v3.0 & added Atari DVG Guide 3/26/24

This hardware and software is provided "as-is". Although it has been tested with several arcade boards, use of this hardware and software may cause damage to the device under test and the user assumes all risk in using the hardware and software. This project is currently in the beta phase, so please expect some issues and bugs.

High-frequency digital signals are present on the arcade board under test and on the interconnections between the tester and the arcade board. It is likely that RF interference will be produced during the testing process and it is the user's responsibility to comply with all applicable regulations.

Permission is granted to use the software in association with the accompanying FPGA tester hardware. The software may also be used to generate and view configuration files and other associated testing information as long as any testing functions are conducted through the provided FPGA tester hardware. You may install the software on multiple computers but you may not redistribute the software without the permission of the author.

DISCLAIMER OF WARRANTY -- This software and its associated hardware are provided 'AS IS'. There are no expressed or implied warranties, including, but not limited to, implied warranties of merchantability, of satisfactory quality, non-infringement of third party rights, and fitness for a particular purpose or use. The Licensor makes no representation that the provided hardware or software does not or will not infringe any patent, copyright, trade secret or other proprietary right. The entire risk as to the use, quality, and performance of any software and hardware shall be with You and not the Licensor. This disclaimer of warranty is an essential part of this License and a condition for the grant of any rights granted under this License.

EXCLUSION AND LIMITATION OF LIABILITY -- The Licensor shall, to the maximum extent permitted by law, have no liability for direct, indirect, special, incidental, consequential, exemplary, punitive or other damages of any character including, without limitation, procurement of substitute goods or services, loss of use, data or profits, or business interruption, however caused and on any theory of contract, warranty, tort (including negligence), product liability or otherwise, arising in any way in relation to the licensed software and/or hardware, even if advised of the possibility of such damages, and You shall hold the Licensor(s) free and harmless from any liability, costs, damages, fees and expenses, including claims by third parties, in relation to such use.

Table of Contents

Resources

Google Drive containing up to date software and logic as well as the latest version of this manual:

FPGA-CAT Shared - Google Drive

Fred’s YouTube video playlist is here:

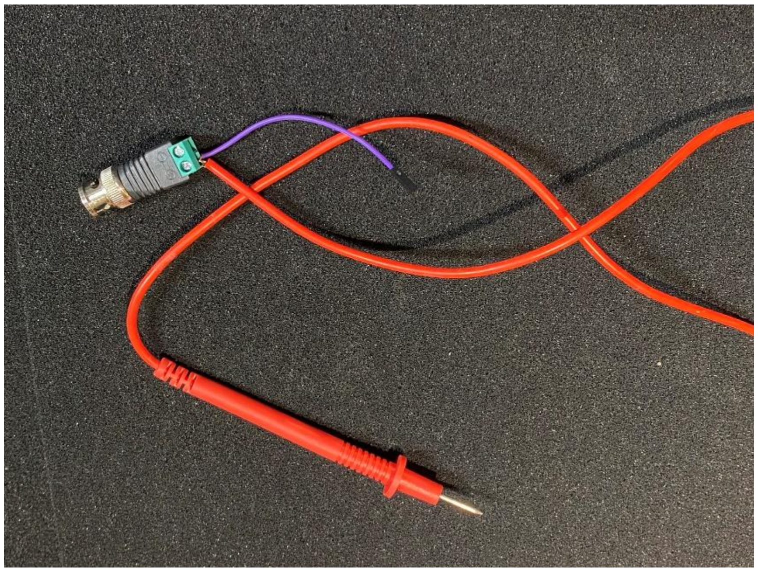
https://youtube.com/playlist?list=PLDe31EDCUSj19TSReRufoCrQVGdv7IZWS
The YouTube videos are "unlisted" since I'm trying to focus on groups like KLOV instead of having to answer a bunch of random questions from Facebook or YouTube etc. But if you have that link to the playlist you should be able to see all of my videos.

Douglasgb and Rotormoshun’s training videos playlist is here:

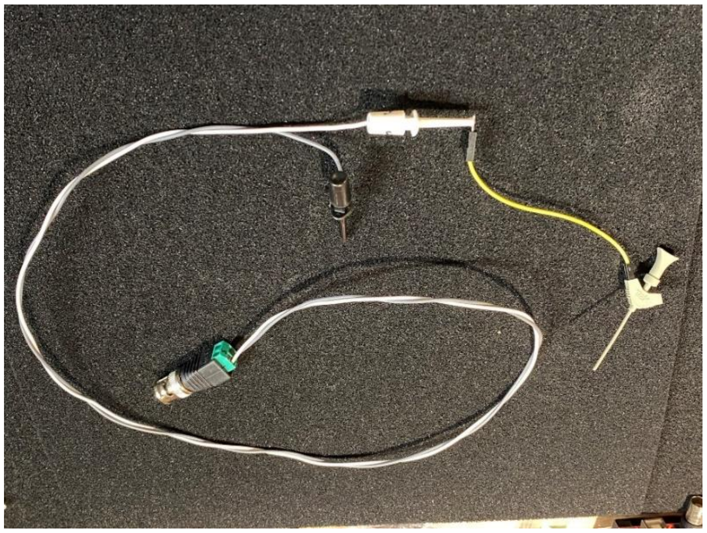
https://www.youtube.com/playlist?list=PLVD-zkkSvyEVWdcKeYNyCHcRgGxiabkLg

Signature Analyzer Connections (not included with the Tester)

We originally used low-cost oscilloscope probes for the SA connections and these are still useful but we have had a few situations like the Battlezone mathbox tests where the capacitance of the probe would cause an issue for the signals. The current approach is to make up some cables using “BNC Baluns” that go from BNC to screw terminals. They are not really “baluns” since they don’t have internal transformers but they can be purchased from amazon for less than 50 cents each: https://www.amazon.com/gp/product/B091BPZ7WZ/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1
For the data probe I’m using about 24” of an old test lead, plus a short pigtail with a “dupont” connector that I can use to plug in the pull-up resistor used in the Battlezone and Tempest mathbox tests. You can find “bare” test probes and other nice grabbers and hooks at e-z-hook.com



For the clock and start probes I’m using about 24” of 22 gauge wire that is twisted together as shown in the picture below. I soldered on mini grabbers to the end of the wires and I sometimes use a short Tektronix logic analyzer grabber (often available on eBay) for grabbing onto IC pins without shorting them. An alternative to the Tektronix grabber that seems to work well is the E-Z-Hook XKM series https://www.digikey.com/en/products/detail/e-z-hook/XKMGRY/528232
There is also a knockoff of the Tektronix grabbers that seems to work OK:
https://www.amazon.com/Goupchn-Grabber-Analyzer-Electronic-Testing/dp/B09TPBS7YF/ref=sr_1_18?keywords=SMD+grabber&qid=1676252550&sr=8-18



This is what I originally used:

I sometimes use two oscilloscope probes (one for CLK and one for DATA) and two BNC to mini-grabber cables. I did my testing with the very inexpensive ones below from Amazon shown in the links below. The oscilloscope probes are sold under different names but all seem to say "P6100" on them. I took out the little switch caps on mine to keep them in 1X mode because the covers can shift around and switch the probes into 10X. You can try your own oscilloscope probes if you want, but be aware that some that I tried (specifically some from Rigol) had a lot more capacitance than the P6100 ones so they loaded down the signals more. Because the TTL outputs have a low drive strength going high and are often fanned out on the arcade boards, driving the probes can cause a phase shift in the signal and cause the signature analyzer to not catch the right logic level when the clock transitions (or shift the clock itself if that's what you are connecting to the probe).

https://www.amazon.com/gp/product/B00XJH2M02/ref=ppx_yo_dt_b_asin_title_o04_s01?ie=UTF8&psc=1

https://www.amazon.com/gp/product/B077WTTMYT/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1

Here is a set that comes with two oscilloscope probes and one BNC to mini grabber cable in case you want an extra of those:
https://www.amazon.com/gp/product/B07Q88852L/ref=ppx_yo_dt_b_search_asin_title?ie=UTF8&psc=1

You will also likely need smaller grabber clips to grab on to IC pins – I like the ones that are used with Tektronix logic analyzers that are made for SMD testing (Search eBay for Tektronix 40V grabber). There are some SMD grabbers on Amazon that are OK but they don't grab as well as the Tektronix (the Tek ones hold on amazingly well). There are also some clones of the Tektronix ones on Amazon but I have not tried them. I have several short wires that plug in to an SMD grabber on one end and have a pin on the other end that you can grab with the big BNC grabbers.

Other accessories:

A ZIF socket plugged in to a dual-wipe socket (which destroys the dual-wipe socket for any other use) can then be plugged in to the arcade board CPU socket if you are going to be switching back and forth between CPU and tester. You can also make a test fixture with some protoboard that uses the smaller round headers like the adapters use to make a

ZIF socket with smaller pins on the bottom. Some users have also used 40 pin component carriers for this purpose. Mainly you want a way to plug a ZIF socket into the arcade board without damaging the socket on the board. I had to bend the arm of the ZIF socket to fit the 6502 adapter (I'll eventually redesign it to be narrower). The 6502 adapter doesn't have traces on the side of the lever so you can use some snips to make a notch where the ZIF arm goes if you want to do it that way.

Software Setup Instructions

The software is available here: https://drive.google.com/drive/folders/1oD4RSbiz7HLmX2tRbbnRT28lvkPk-RYp?usp=sharing

You need to install ni-visa runtime 18.5 (or higher) and the LabVIEW runtime engine 2018-SP1 32 bit. Links are provided below but they may change. Be sure you are downloading the "runtime" versions of both VISA and LabVIEW (not the full development versions).
https://www.ni.com/en-us/support/downloads/drivers/download/unpackaged.ni-visa.306119.html

https://www.ni.com/en-us/support/downloads/software-products/download/unpackaged.labview.309627.html

Extract the FPGACAT software to a folder of your choice. This zip file contains a folder with the latest config files up until the point when the software was released. It is recommended that you keep your ROMs folder and the configurations separate so that is easy to share your configurations without sending ROMs. The board configurations contain checksums but not the actual ROM files. Once you have tested the ROMs on the board against the checksums, you can use the tester to read the ROMs from a board and create your own ROM files as a backup or as a way to compare against another board. The most up to date tester configuration files are now available on GitHub. Software version 3.0+ can download these files automatically using the "Git/Paths" tab (see documentation below).

If you run fpgacat.exe, you should get a license window that you can read and accept, and then you will be at the "setup" tab in the software. In order to connect to the tester, you will need to specify the com port that is created when the tester is plugged in through USB. Since the tester USB port is isolated from the tester logic and powered by the computer, the tester does not need to be powered to show up as a USB com port on the computer. You can go to "device manager" on the computer to see which com port is assigned to the tester or you can select "refresh" in the pull down "COM port" menu and the largest number will probably be the tester unless you plugged in another USB serial device after plugging in the tester. The software will automatically save the com port you select so you should only need to do this once. In case you have any issues with the USB driver, the USB chip that is used is from FTDI and you can download drivers from https://www.ftdichip.com/FTDrivers.htm

Original version hardware: If you power up the tester using the included power supply and switch it on while it is plugged in to the USB port on the computer, you should see the "comm OK" LED lit up , and the "phase2 clock" LED will be blinking unless you have connected the tester to an arcade board. If you click "open" in the software, it should report the tester logic and hardware versions and the "port opened" light should illuminate in the software.

New version hardware: Once you click Open in the software, the tester will power on and boot up after a few seconds. You should then see the "comm OK" LED lit up , and the "phase2 clock" LED will be blinking unless you have connected the tester to an arcade board. No external power supply and switch are needed because the new version hardware is powered by an internal DC-DC converter which gets power from the USB port while maintaining isolation between the computer and the tester. Closing the connection will turn off the tester power.
Now that you have established communication to the tester, you can switch off the tester power and connect it to an arcade board to run tests.

General Guidelines:

1. Power: Original hardware: Please use the provided 5V power supply to power the tester. It can also be powered from your arcade board test setup if you rig something up but the 5V power input goes right to the 5V logic chips driving some of the bus so you can damage the board with over-voltage. It is also not protected very well against applying power backwards. There is a 5.6V Zener diode on the board power to prevent overvoltage from the input protection on the signature analyzer, so there is some level of protection but it is safest to use the provided power supply. In addition, the board only draws 100-120mA total and the included power supply is rated at around 500mA max which will prevent large currents in the event of a failure somewhere. (The new hardware is powered by the USB port so you don't need to worry about this section if you have the new version)

2. Grounding: Note that the tester is floating and is grounded through the arcade board to avoid ground loops especially with the computer's USB connection. The best practice is to connect one of the signature analyzer probe's ground (the BNC grabber black wire or the oscilloscope probe ground) to the arcade board ground before plugging in the tester. All of the "barrels" of the BNC connector are connected to the tester's ground plane. You can also just use an alligator clip jumper between a BNC outer shell and the arcade board. This additional grounding can also give a better ground connection when using the CPU adapter since the ground in that case is just passed through one or two pins on the CPU socket.

3. You should hit "close" on the setup tab and then power down the tester before plugging it into an arcade board or unplugging it. (On the newer testers closing the connection also powers off the board) Make sure that it is plugged in to the arcade board properly before powering it on. Once they are connected together, you should be able to power up the tester and arcade board in any order because the tester will drive all outputs to zero within a few microseconds if it does not detect a clock from the arcade board. If the board has a 6502 and you are using the edge connector, be sure to remove the CPU before connecting the tester to the edge connector. A 6809 can stay in the socket because it will be disabled by the tester edge connector.

4. The simplest way to use the tester is to just use the CPU adapter rather than the edge connector if you are testing Atari boards. The bottom line is that if you use the CPU adapter, you do not have to jumper phase0 and phase 2 or otherwise provide a clock as described in the Atari manuals. Details follow: The test procedures in the Atari manuals have some way of supplying a phase2 clock on the 6502-based boards. (this does not apply to the 6809E) Most of the time there are two test points that you can use to jumper phase2 and phase0. (but be careful – on boards like Battlezone the phase2 test point is an output and can't be connected to phase0!) I have found that in some cases the phase0 clock doesn't drive the clock network and tester ribbon cable very well so you could run in to problems doing it this way. To address this issue, the 6502 adapter board that comes with the tester has a buffer between phase0 and phase2 (just like the 6502 has internally). So **when you use the cpu socket adapter, do not jumper phase0 and phase2 on the arcade board**. If you use the edge connector, the preferred way to run the tester is using the adapter board in the CPU socket instead of jumpering the clocks: just leave the adapter's 50 pin connector open and plug in the edge connector. This has the advantage of not having to figure out what to jumper on the arcade board and you can just use the same method each time. Jumpering the clocks per the arcade manual works the vast majority of the time so you could try that and switch over to the adapter if you get any errors. Just beware of this issue because the error you are getting might be caused by the clock loading, especially if you are having trouble accessing the vector generator.
5. When doing signature analysis with a clock signal connected to phase2, it is best to just let the tester make the connection internally using the button in the software. This prevents the additional loading of the phase2 clock signal caused by the capacitance of the oscilloscope probe. I have seen a few examples of ringing on the clock

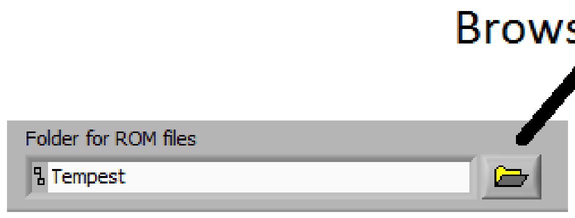
line when it was connected this way so just go the “easy way” and let the software and logic do it unless you suspect a problem with a signature (every sig. I have checked has been OK with the internal phase2 connection, but it does shift the timing a little). The same goes for “start” and “stop”. Most setups have start and stop connected together, so the logic and software lets you just use “start” and make the connection internally. This prevents the loading on the signals by two cables since you can just use one instead. Finally you can connect start and stop to A15 internally which is used for many address decoder tests and is sometimes hard to get to on the arcade board. This means that for some tests, you just need the data probe and you can leave the other three probes disconnected. You should still connect the ground of one or more other BNC connectors to the arcade board especially if you are using the data probe ungrounded. I typically leave the data probe ungrounded so I don’t have to constantly plug in the ground clip when moving it around – then if I get an incorrect or unstable signature I might try to ground it to see if that was causing an issue.

6. At the time of this writing, the configuration files are not complete with all ROM versions, etc. I would appreciate any efforts to clean things up and suggestions for file naming, etc. I’m sure that there are other bugs and issues so I appreciate your willingness to help out in this early beta phase of the project!
7. The Z80 adapter requires power, clock, and a WAIT signal (on the Z80 CPU socket) that is not stuck active (it is not stuck low) in order for the tester to communicate with the adapter and get the version when opening the port. If you get an error when opening the connection to the Z80, be sure to check that you have clock and wait is not stuck low. In the case when there is no clock or WAIT is stuck low, the clock LED on the tester will flash and clock detected will not light up when opening the connection.

Notes about interacting with the software:

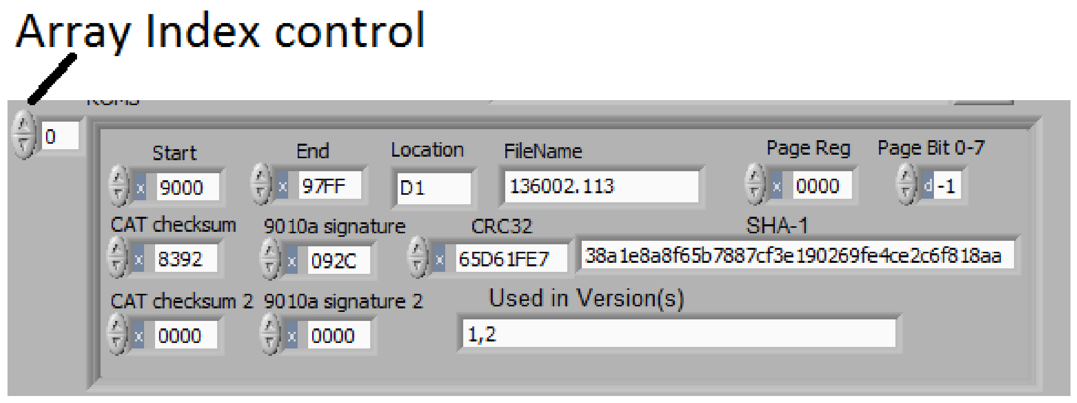
Most of the controls and indicators on the LabVIEW panel are fairly self-explanatory but there are a few things that would probably be a good idea to mention if you have not run a LabVIEW program before:

- On the file controls, if you click on the "browse button" which is the folder icon to the right of the control's text box, you will be able to browse to a location or file using a pop-up window.



Folder for ROM files: Tempest

- When there is an array control in LabView, you will normally have an index control in the upper left. This is the element of the array that you are currently looking at. You can use the arrows to scroll through the array or type in the index you want. Once you get to the end of the array, the elements will be grayed out. Some of the array controls and indicators in the software will display more than one element at a time, and in that case the index control shows you the index of the element at the top. Some arrays may also have a scrollbar to the right, so be sure that you scroll all the way through the list if, for example, the ROM test says that they are not all OK but you don't see one that has failed in the list that is showing.

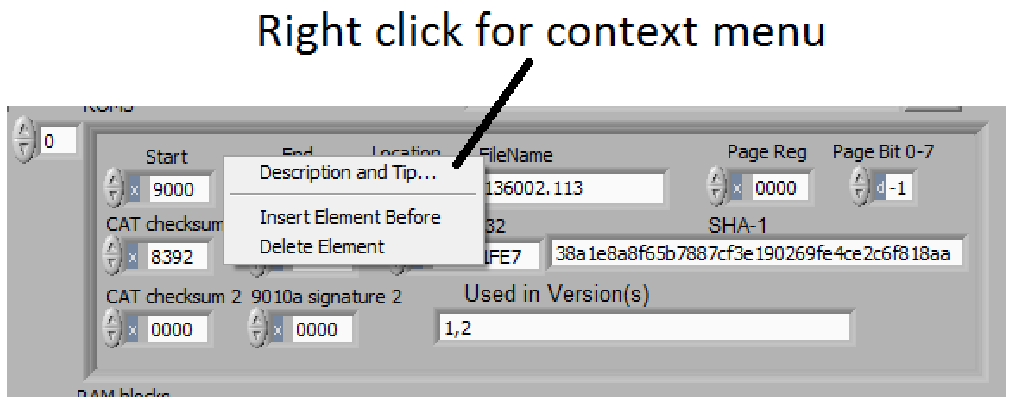


Start	End	Location	FileName	Page Reg	Page Bit 0-7
9000	97FF	D1	136002.113	0000	-1

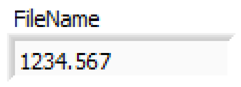
CAT checksum	9010a signature	CRC32	SHA-1
8392	092C	65D61FE7	38a1e8a8f65b7887cf3e190269fe4ce2c6f818aa

CAT checksum 2	9010a signature 2	Used in Version(s)
0000	0000	1,2

- For most controls, you can right click to pop up a context menu. The main thing you will need this for is deleting and inserting elements into the arrays.

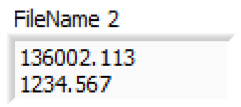


- **Watch out for hitting “enter” in a text box!**



FileName

1234.567



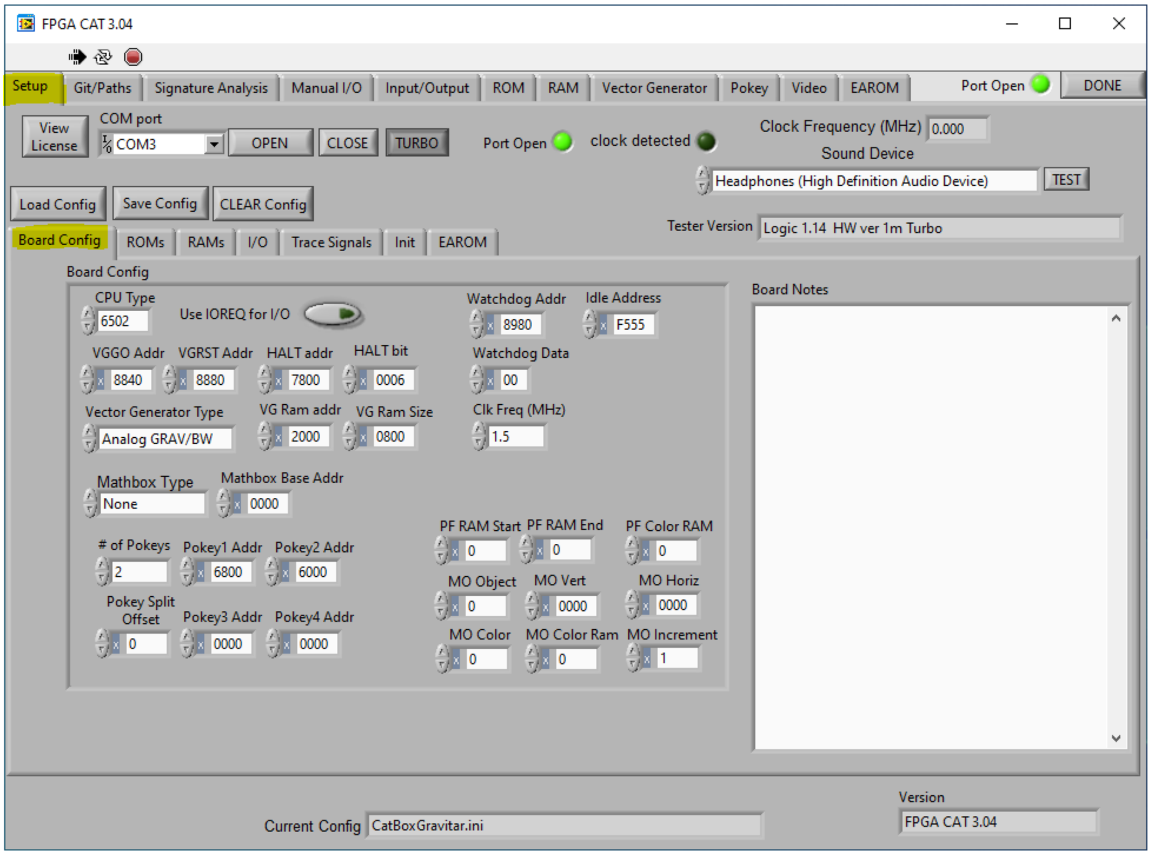
FileName 2

136002.113
1234.567

It looks like the filename is 1234.567, right? Well, I hit “enter” after the 136002.113 which created a new line, then I typed 1234.567. When I resize the text box I can see everything but otherwise I see just the last line. So the real string in the control is “136002.113<CR><LF>1234.567”. So make sure you hit “backspace” to delete what is in the text control if you are going to replace its content. If you just hit ”enter” with the cursor at the end of the string, it will look like it is now blank but there is really the original string still “above” where you can see!

Setup tab

Setup-Board Config tab


FPGA CAT 3.04

Setup | Git/Paths | Signature Analysis | Manual I/O | Input/Output | ROM | RAM | Vector Generator | Pokey | Video | EAROM | Port Open | DONE

View License | COM port: COM3 | OPEN | CLOSE | TURBO | Port Open | clock detected | Clock Frequency (MHz): 0.000

Sound Device: Headphones (High Definition Audio Device) | TEST

Load Config | Save Config | CLEAR Config

Tester Version: Logic 1.14 HW ver 1m Turbo

Board Config | ROMs | RAMs | I/O | Trace Signals | Init | EAROM

Board Config

CPU Type: 6502
Use IOREQ for I/O
Watchdog Addr: 8980
Idle Address: F555
VGGO Addr: 8840
VGRST Addr: 8880
HALT addr: 7800
HALT bit: 0006
Watchdog Data: 00
Vector Generator Type: Analog GRAV/BW
VG Ram addr: 2000
VG Ram Size: 0800
Clk Freq (MHz): 1.5
Mathbox Type: None
Mathbox Base Addr: 0000
PF RAM Start: 0
PF RAM End: 0
PF Color RAM: 0
# of Pokeys: 2
Pokey1 Addr: 6800
Pokey2 Addr: 6000
MO Object: 0
MO Vert: 0000
MO Horiz: 0000
Pokey Split Offset: 0
Pokey3 Addr: 0000
Pokey4 Addr: 0000
MO Color: 0
MO Color Ram: 0
MO Increment: 1

Board Notes

Current Config: CatBoxGravitar.ini

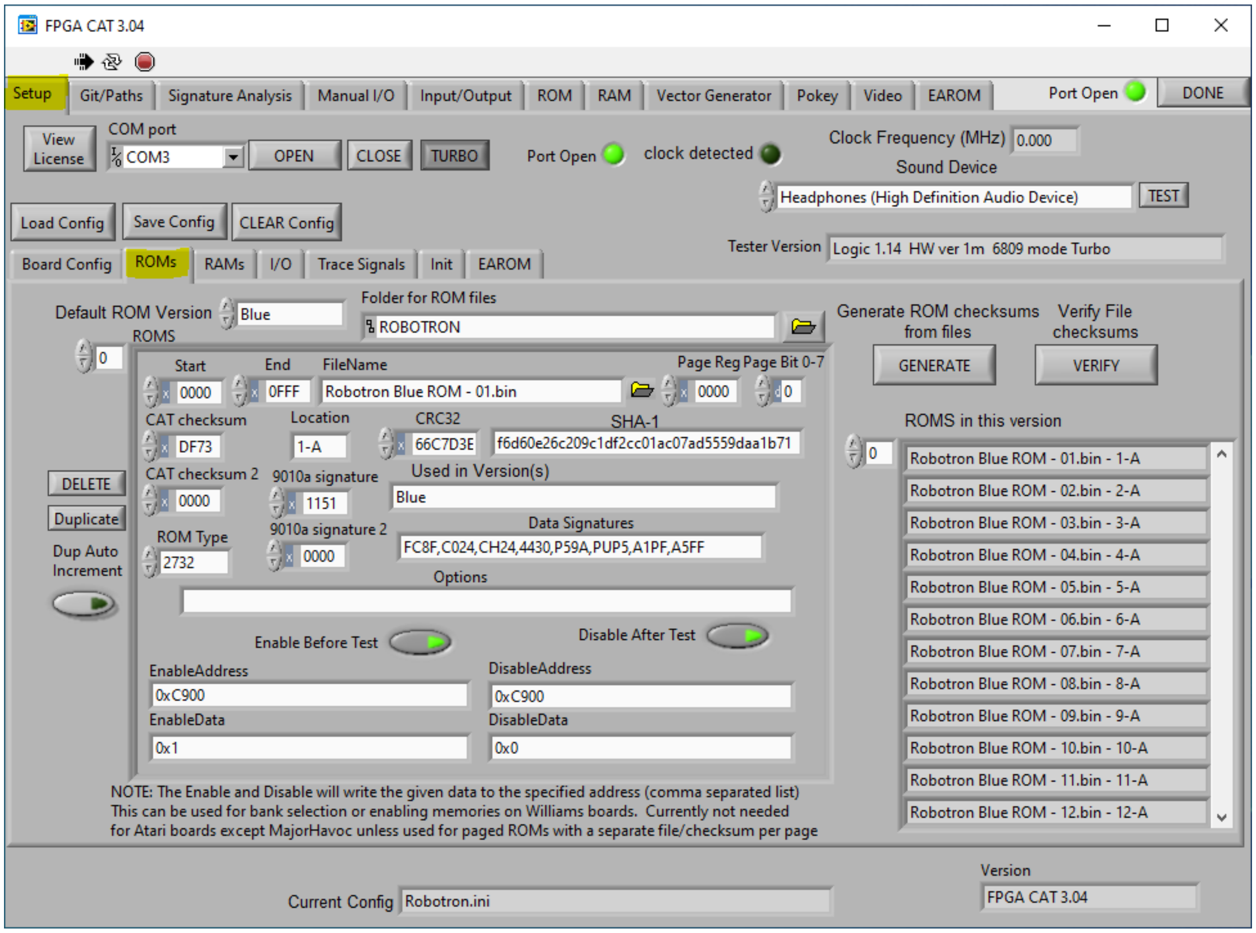
Version: FPGA CAT 3.04

This is where you set up, load, and save the board configuration. It is also where you open the connection to the tester board. The Board Config section lets you specify the memory addresses of several things on the board like the pokey chips and the vector generator flags. You can also specify the type of vector generator and the type of mathbox.   The entries at the bottom are for raster games like centipede and specify addresses for the motion object control and playfield. You can leave the non-applicable stuff at zero (like the motion object entries for a vector game).

The "Use IOREQ for I/O" option is required for testing of inputs and outputs on Sega G80 boards (Space Fury, Tac/Scan, Eliminator, Zektor, Battle Star, Star Trek, Astro Blaster, Space Odyssey, 005, Monster Bash) that use the Z80's /IORQ for I/O control.

The additional tabs let you set up various parts of the board like the ROM and RAM locations and address ranges that are discussed in the following sections.

Setup-ROMs tab


FPGA CAT 3.04

Setup | Git/Paths | Signature Analysis | Manual I/O | Input/Output | ROM | RAM | Vector Generator | Pokey | Video | EAROM | Port Open | DONE

View License

COM port: COM3 | OPEN | CLOSE | TURBO

Port Open

clock detected

Clock Frequency (MHz): 0.000

Sound Device: Headphones (High Definition Audio Device) | TEST

Load Config | Save Config | CLEAR Config

Tester Version: Logic 1.14 HW ver 1m 6809 mode Turbo

Board Config | ROMs | RAMs | I/O | Trace Signals | Init | EAROM

Default ROM Version: Blue

Folder for ROM files: ROBOTRON

Generate ROM checksums from files: GENERATE

Verify File checksums: VERIFY

ROMS: 0

Start: 0000
End: 0FFF
FileName: Robotron Blue ROM - 01.bin
Page Reg: 0000
Page Bit 0-7: 0

CAT checksum: DF73
Location: 1-A
CRC32: 66C7D3E
SHA-1: f6d60e26c209c1df2cc01ac07ad5559daa1b71

CAT checksum 2: 0000
9010a signature: 1151
Used in Version(s): Blue

ROM Type: 2732
9010a signature 2: 0000
Data Signatures: FC8F,C024,CH24,4430,P59A,PUP5,A1PF,A5FF

Options:

DELETE

Duplicate

Dup Auto Increment

Enable Before Test

Disable After Test

EnableAddress: 0xC900
DisableAddress: 0xC900
EnableData: 0x1
DisableData: 0x0

NOTE: The Enable and Disable will write the given data to the specified address (comma separated list)
This can be used for bank selection or enabling memories on Williams boards. Currently not needed
for Atari boards except MajorHavoc unless used for paged ROMs with a separate file/checksum per page

ROMS in this version: 0

- Robotron Blue ROM - 01.bin - 1-A
- Robotron Blue ROM - 02.bin - 2-A
- Robotron Blue ROM - 03.bin - 3-A
- Robotron Blue ROM - 04.bin - 4-A
- Robotron Blue ROM - 05.bin - 5-A
- Robotron Blue ROM - 06.bin - 6-A
- Robotron Blue ROM - 07.bin - 7-A
- Robotron Blue ROM - 08.bin - 8-A
- Robotron Blue ROM - 09.bin - 9-A
- Robotron Blue ROM - 10.bin - 10-A
- Robotron Blue ROM - 11.bin - 11-A
- Robotron Blue ROM - 12.bin - 12-A

Current Config: Robotron.ini

Version: FPGA CAT 3.04

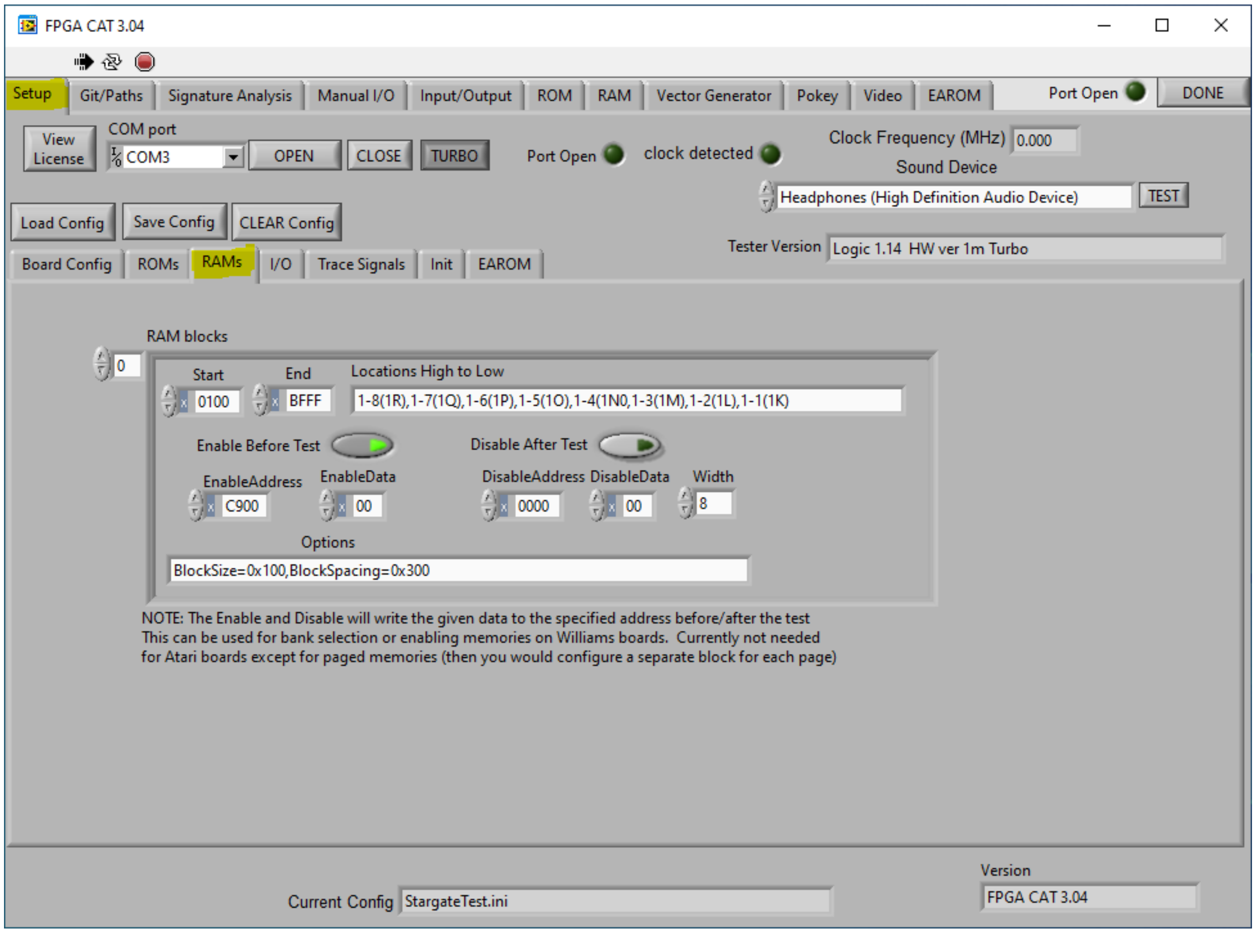
In this section, you have all of the ROMS in the game with their starting address, ending address, location, etc. If you have the ROM files available, you can generate the checksums and signatures from them with the “generate” button instead of typing them in. The page register and page bits are for the few games that have paged ROMs (like Star Wars). The “Cat checksum 2” and “9010a signature 2” entries are for the second page and should be zero if the ROM is not paged. The CRC32 and SHA-1 are calculated on the whole ROM even if it is paged. The “used in version(s)” control is a comma separated list of versions where the particular ROM is used. So you could have multiple entries with the same start and stop address but for different versions. The software automatically generates a list of all of the unique entries in the comma-separated list and gives you a selection box to choose the ROM version you are currently testing. It will also show you a list of the files that are in the version you are testing. Since most of the filenames match the labels on the ROMs, you can verify that you are working with the correct set before testing the board or the ROM test screen can try to detect the version automatically. There is also a “verify checksums” button that will go through all of the files and verify them against the checksums in the table. This is useful if you read the ROMs (on the ROM test page) and want to check that they are OK against checksums you got somewhere else. The enable and disable address and data fields along with the enable before test and disable after test buttons will cause the tester to write data to a specific address before and/or after the particular ROM is tested. The Robotron config screen above shows these options being used to switch the Williams board to ROM mode.

Note that for legal reasons, ROMS are not included in the tester software distribution. Be sure you understand any legal ramifications of downloading ROMS and please do not distribute ROMS as part of your configuration files when sharing them with other tester users.

The “folder for ROM files” is a folder in the ROM folder specified on the Git/Paths tab (Path to all ROM folders). To browse to a location, click on the folder icon to the right of the control. The “path to all ROM folders” is saved in the tester software configuration not the game configuration files. At this point I recommend just having all the ROMs for a particular game (all versions) in a folder with the game’s name (i.e. Tempest). The “folder for ROM files” is saved in each game configuration, and if it is within the “path to all ROM folders”, it will be saved as a relative path to make sharing configurations easier. You can set it to an absolute path but then if you share your configuration everyone will have to go and change that entry to point to their ROMs for that game.

Other buttons:
“Delete” will delete the current ROM entry, moving subsequent ROMs up on the list.
“Duplicate” creates a copy of the current ROM, pushing subsequent ROMs down on the list.
“Dup Auto Increment” will increment the address range, file name and ROM location of duplicates. This can be a huge time saver (in conjunction with “Generate”) when creating Config files for boards with a multitude or ROMs.

Setup-RAMs tab



Port Open

COM port: COM3

Port Open ●   clock detected ●

Clock Frequency (MHz): 0.000

Sound Device: Headphones (High Definition Audio Device)

Tester Version: Logic 1.14 HW ver 1m Turbo

RAM blocks: 0

Start: 0100   End: BFFF   Locations High to Low: 1-8(1R),1-7(1Q),1-6(1P),1-5(1O),1-4(1N0,1-3(1M),1-2(1L),1-1(1K)

Enable Before Test   Disable After Test

EnableAddress: C900   EnableData: 00   DisableAddress: 0000   DisableData: 00   Width: 8

Options: BlockSize=0x100,BlockSpacing=0x300

NOTE: The Enable and Disable will write the given data to the specified address before/after the test
This can be used for bank selection or enabling memories on Williams boards. Currently not needed
for Atari boards except for paged memories (then you would configure a separate block for each page)

Current Config: StargateTest.ini

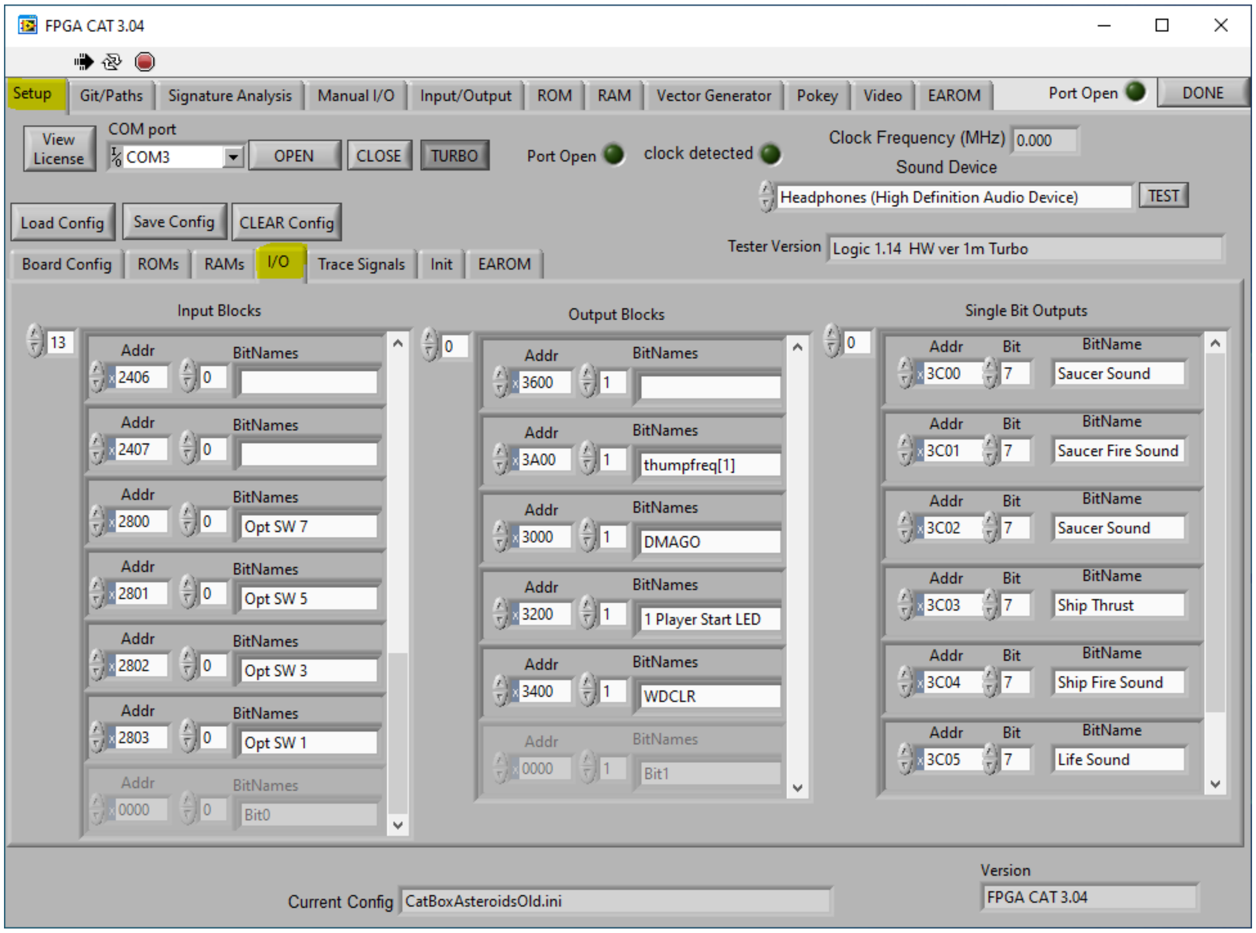
Version: FPGA CAT 3.04

The RAM blocks section is similar to the ROM blocks section except that it lets you set the location information for the bits of the memory since many of our games use 4-bit of 1-bit wide memory. In this case (Williams) there are 8 locations given in a comma-separated list. The first item in the list corresponds to bit 7, and the last bit 0. For a 4-bit memory, you would list the location of the high RAM first (bits 7-4) then the low RAM. For an 8 bit ram, just list one location. The options are as follows:

> **BlockSize=0x####** Set the block size in hexadecimal Above, the block size is 256 bytes (0x100 hex)
>
> **BlockSpacing=0x###** Set the spacing between the blocks.
>
> **NonBlock** If this option is specified, the entire ram will be written and read and the blocking will be sorted out later. This can speed up the ram test for boards like centipede which have lots of small blocks because the USB speed can be much faster when writing and reading fewer large blocks instead of lots of small ones.

In the example above (for Williams boards), the first block of this RAM is from 0x100-0x1FF, then the next block in this RAM would be 0x400-0x4FF, then 0x700-0x7FF etc.. In other words, the block spacing times the zero-based block number added to the start address is the start address of each block.

Setup-I/O tab



COM port: COM3
Port Open
clock detected
Clock Frequency (MHz): 0.000
Sound Device: Headphones (High Definition Audio Device)
Tester Version: Logic 1.14 HW ver 1m Turbo

Input Blocks: 13

Addr	Bit	BitNames
2406	0	
2407	0	
2800	0	Opt SW 7
2801	0	Opt SW 5
2802	0	Opt SW 3
2803	0	Opt SW 1
0000	0	Bit0

Output Blocks: 0

Addr	Bit	BitNames
3600	1	
3A00	1	thumpfreq[1]
3000	1	DMAGO
3200	1	1 Player Start LED
3400	1	WDCLR
0000	1	Bit1

Single Bit Outputs: 0

Addr	Bit	BitName
3C00	7	Saucer Sound
3C01	7	Saucer Fire Sound
3C02	7	Saucer Sound
3C03	7	Ship Thrust
3C04	7	Ship Fire Sound
3C05	7	Life Sound

Current Config: CatBoxAsteroidsOld.ini
Version: FPGA CAT 3.04

Input Blocks

Input blocks have a single address and let you specify an array of bits (bit 0-7) so you can test the inputs of the board. This is just a convenience so you don't have to "decode" the number that you read from that location into the bits you are testing.

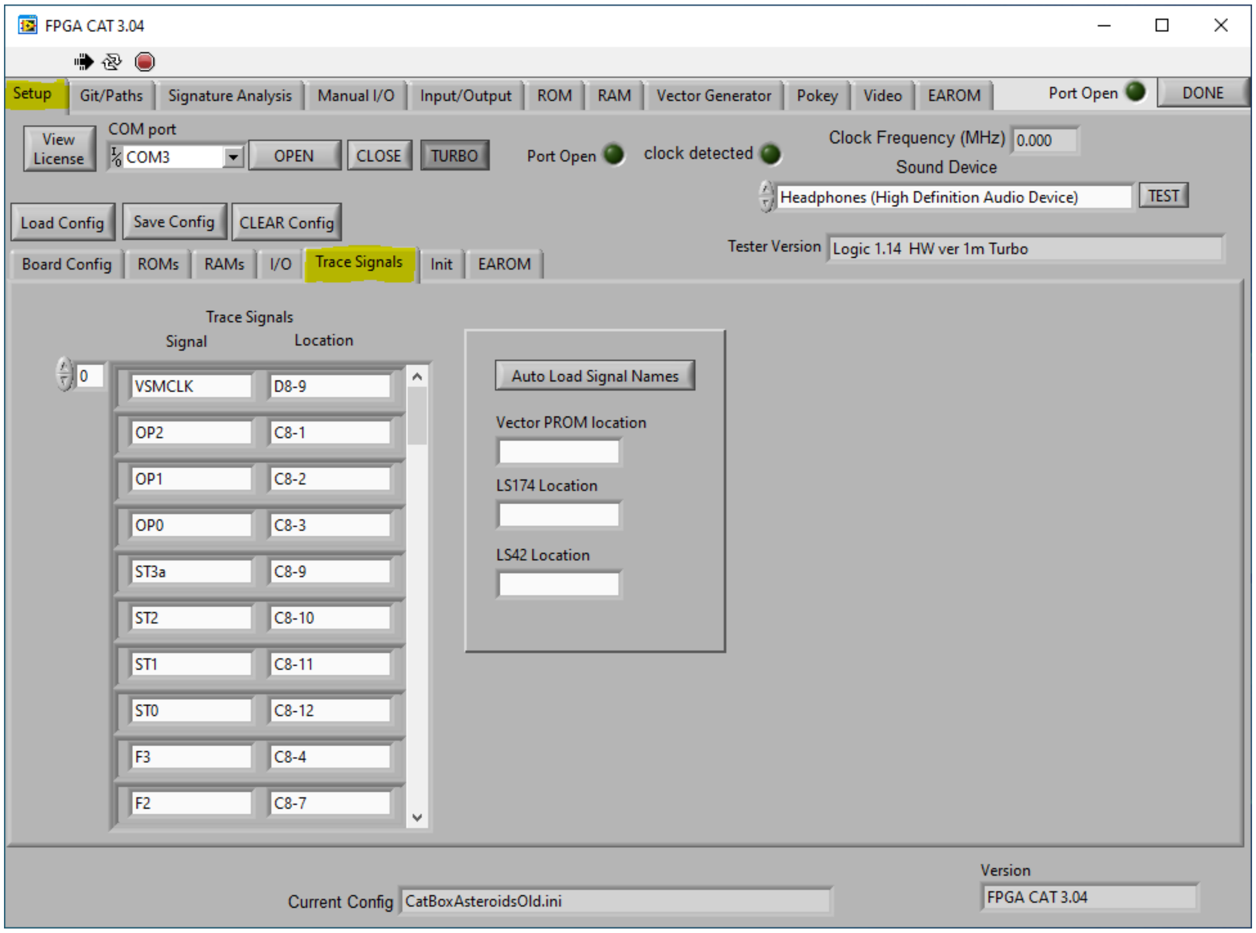
Output Blocks

Output blocks are for specifying the individual bits of a set of outputs at a single memory location. On the outputs tab, you can go in and toggle the bits individually to test them or do things like turn on the sound in Battlezone, etc.

Single bit outputs

These are for situations where there is just one bit (or no bits at all) that is needed for a specific output port address. Examples of these would be the vector generator "go" address, etc.

Setup-Trace Signals tab



COM port: COM3

Port Open

Clock Frequency (MHz): 0.000

Sound Device: Headphones (High Definition Audio Device)

Tester Version: Logic 1.14 HW ver 1m Turbo

Trace Signals

Signal	Location
VSMCLK	D8-9
OP2	C8-1
OP1	C8-2
OP0	C8-3
ST3a	C8-9
ST2	C8-10
ST1	C8-11
ST0	C8-12
F3	C8-4
F2	C8-7

Vector PROM location

LS174 Location

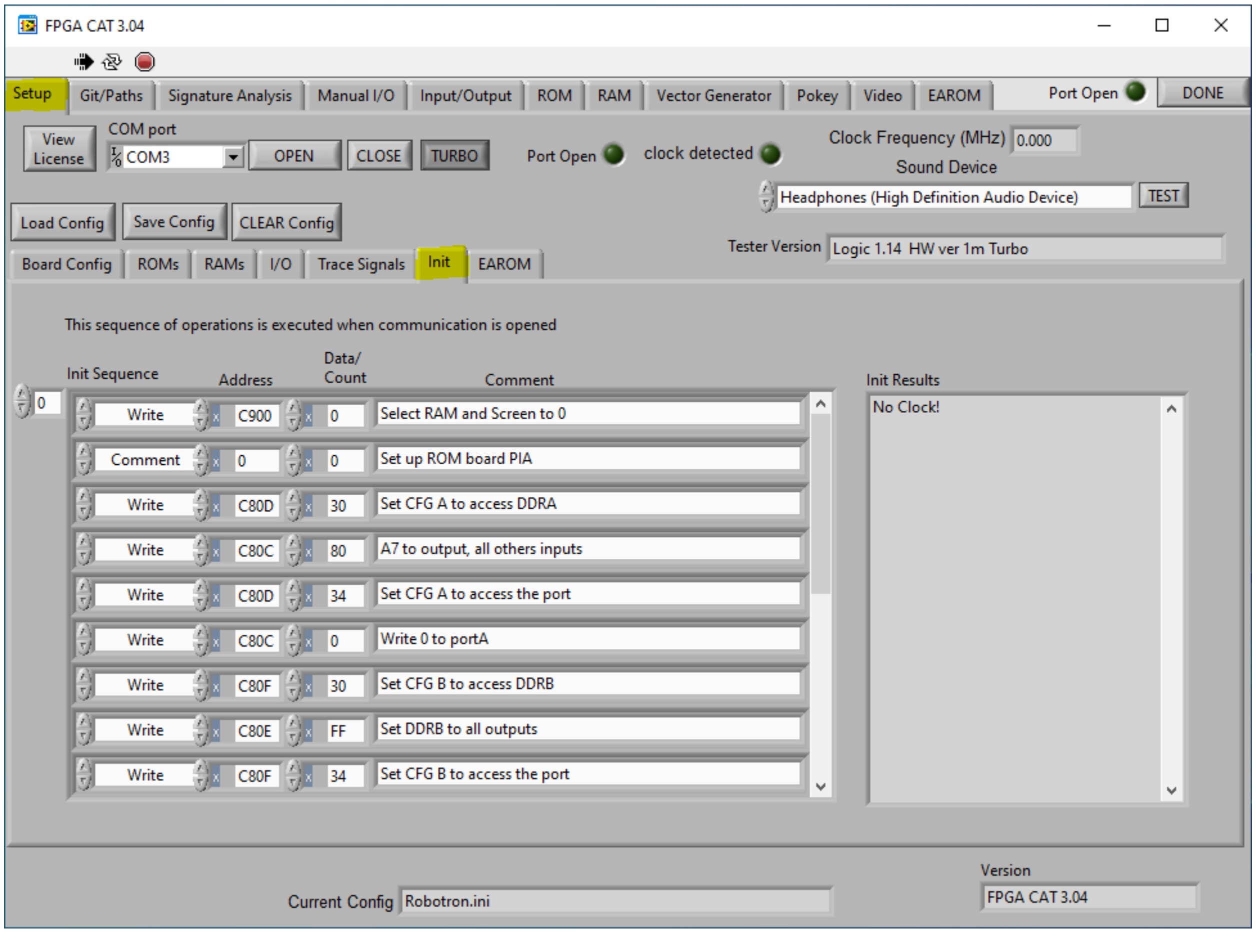
LS42 Location

Current Config: CatBoxAsteroidsOld.ini

Version: FPGA CAT 3.04

Trace signals are used by the Atari vector generator trace function. There is an option to auto-load most of the signal locations by giving the board location of the vector PROM LS174 and LS42 that are in the state machine section of the schematic.

Setup-Init tab


FPGA CAT 3.04

COM port: COM3

Port Open · clock detected

Clock Frequency (MHz) 0.000

Sound Device: Headphones (High Definition Audio Device)

Tester Version: Logic 1.14 HW ver 1m Turbo

This sequence of operations is executed when communication is opened

Init Sequence	Address	Data/ Count	Comment
Write	C900	0	Select RAM and Screen to 0
Comment	0	0	Set up ROM board PIA
Write	C80D	30	Set CFG A to access DDRA
Write	C80C	80	A7 to output, all others inputs
Write	C80D	34	Set CFG A to access the port
Write	C80C	0	Write 0 to portA
Write	C80F	30	Set CFG B to access DDRB
Write	C80E	FF	Set DDRB to all outputs
Write	C80F	34	Set CFG B to access the port

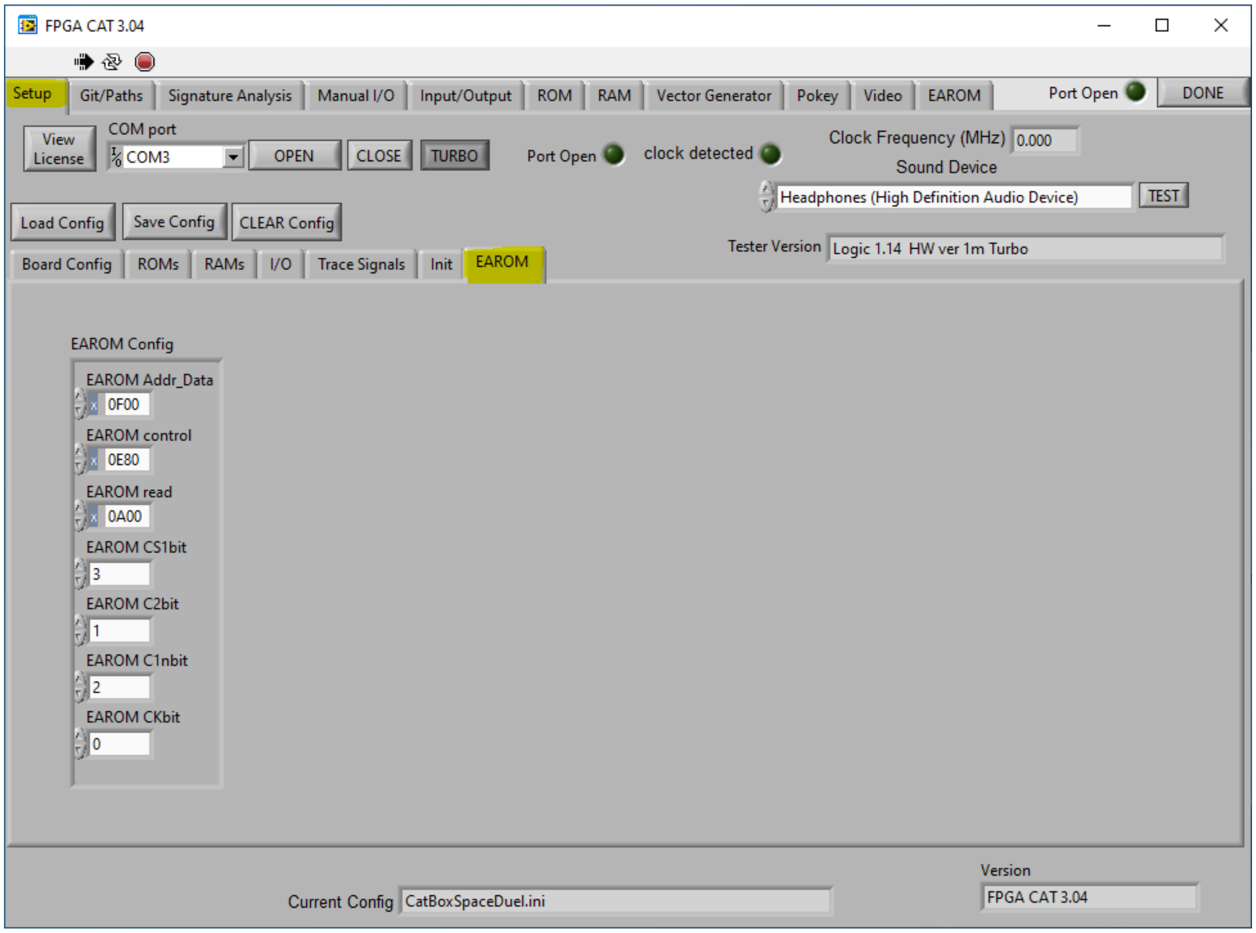
Init Results: No Clock!

Current Config: Robotron.ini

Version: FPGA CAT 3.04

The Init configuration will run a sequence of operations when the connection to the board is opened. If you have powered on the board after opening connection to the tester, you will need to re-open the connection for the Init sequence to run properly. The above screenshot shows the Init for a Williams board where the PIAs are setup to have the ports configured in the proper directions.

Setup-EAROM tab



COM port: COM3
Port Open
Clock Frequency (MHz): 0.000
Sound Device: Headphones (High Definition Audio Device)
Tester Version: Logic 1.14 HW ver 1m Turbo

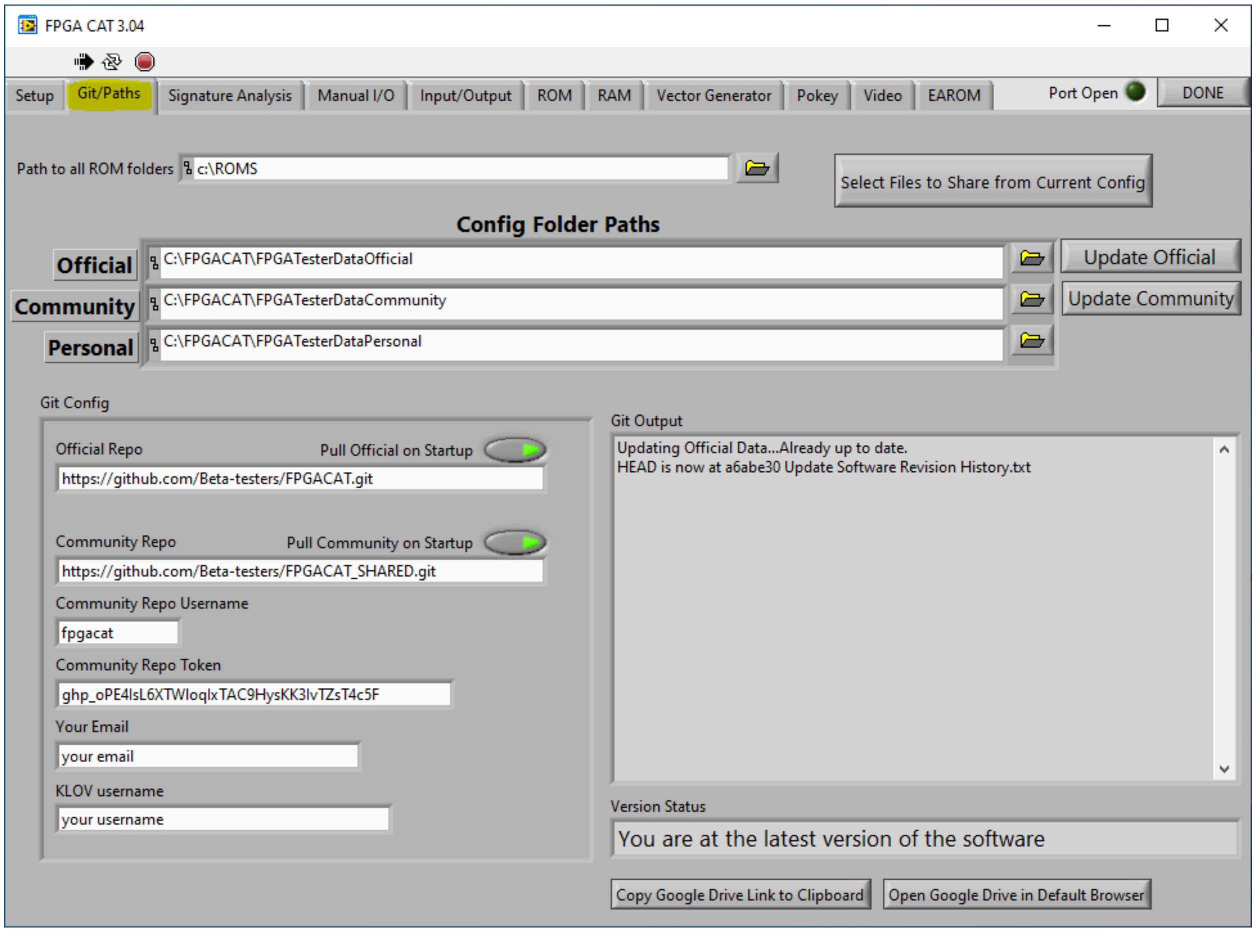
EAROM Config

EAROM Addr_Data	0F00
EAROM control	0E80
EAROM read	0A00
EAROM CS1bit	3
EAROM C2bit	1
EAROM C1nbit	2
EAROM CKbit	0

Current Config: CatBoxSpaceDuel.ini
Version: FPGA CAT 3.04

The EAROM tab allows you to set up the EAROM for Atari games that use the ER2055 EAROM chip to save high scores and settings. You will need to set up the bit locations since some Atari boards have different bits used for the EAROM control signals. The above example is for space duel, so you can look at that schematic to clarify the meaning of each signal.

Git/Paths tab



Path to all ROM folders: c:\ROMS

Config Folder Paths

Official: C:\FPGACAT\FPGATesterDataOfficial
Community: C:\FPGACAT\FPGATesterDataCommunity
Personal: C:\FPGACAT\FPGATesterDataPersonal

Git Config

Official Repo: https://github.com/Beta-testers/FPGACAT.git
Pull Official on Startup

Community Repo: https://github.com/Beta-testers/FPGACAT_SHARED.git
Pull Community on Startup

Community Repo Username: fpgacat

Community Repo Token: ghp_oPE4lsL6XTWloqlxTAC9HysKK3lvTZsT4c5F

Your Email: your email

KLOV username: your username

Git Output

```
Updating Official Data...Already up to date.
HEAD is now at a6abe30 Update Software Revision History.txt
```

Version Status: You are at the latest version of the software

This is a new feature introduced with software version 3.0. The tester can now use GitHub (an online software repository) to allow downloading and sharing of FPGACAT data files. Data storage is divided into three directory trees with common structure:

FPGATesterDataOfficial : these files have passed some level of "expert" review/approval and are stored on GitHub.

FPGATesterDataCommunity : these files have been submitted by members of the user community and are pending review/promotion to the Official folder. These files are also stored on GitHub, however please consider this the "wild west" of Configs.

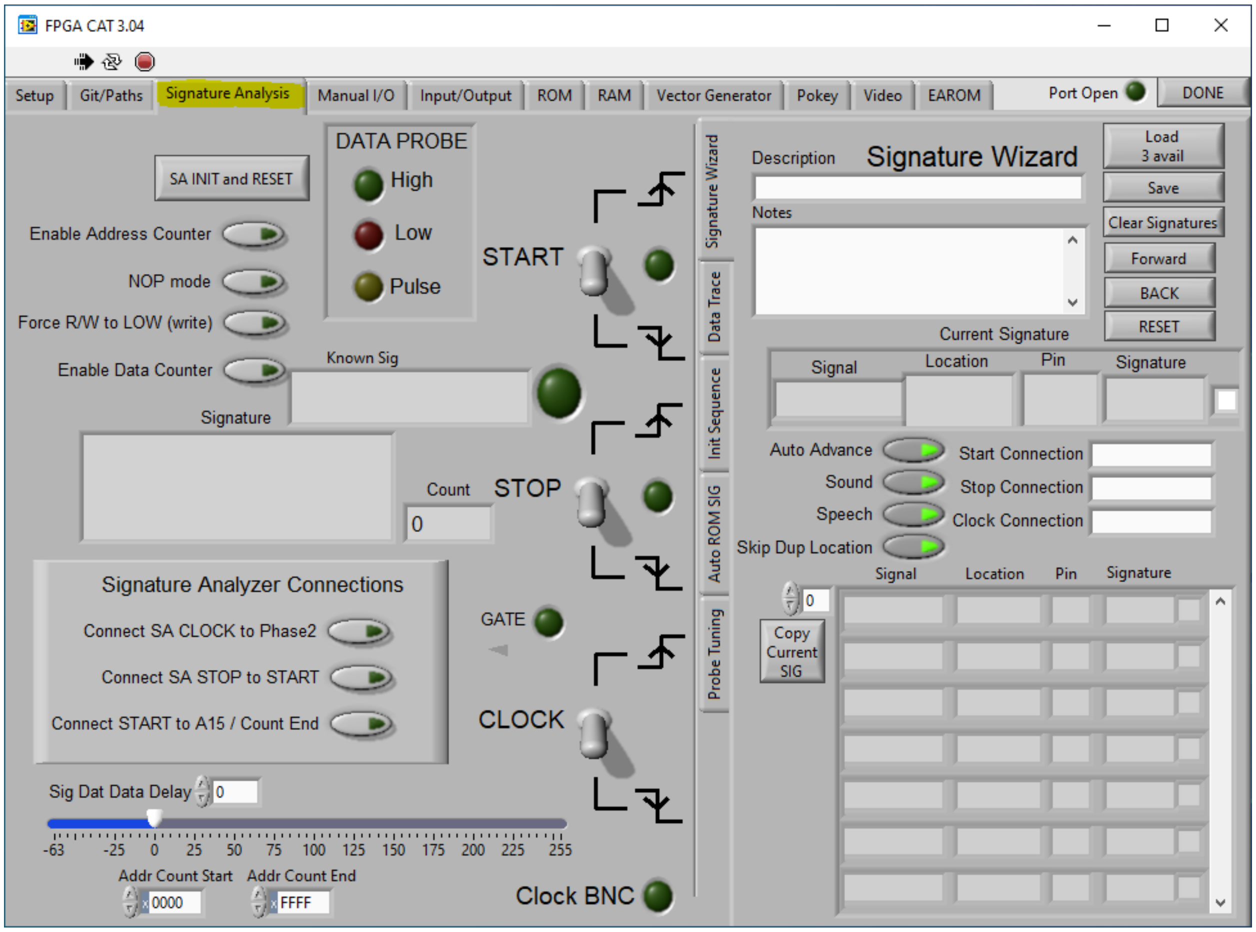
FPGATesterDataPersonal: these are your personal files which are only stored on your computer and not shared with other users.

The Official and Community folders on your computer can be updated manually (Update buttons to the right of the folder paths) or automatically using the "Pull on Startup" option. Users are asked to provide an email address and KLOV username to help avoid naming conflicts when sharing configuration files.

If you wish to contribute Personal Config files to the user community, the "Select Files to Share from Current Config" button gives you the ability to choose those specific files and upload them to the GitHub FPGATesterDataCommunity folder structure. Your KLOV name will be appended to the file name, which avoids overwriting others contributions.

The software also checks for newer versions on startup, which is then available on Google Drive as a zip file.

Signature Analysis Test tab



Here is where you do signature analysis. The “enable address counter” starts counting through all of the addresses in the memory sequentially to generate signatures for address lines, address decoding, ROMS, etc. You can select “NOP mode” to double-count each address to get signatures like using a NOP card instead of a CAT box. If you leave NOP mode off, you will get signatures that match the arcade manuals.

You can force R/W low with the switch above the signature display. Please use this instead of grounding R/W on the board like the troubleshooting guides tell you to do. I recommend that you remove any EAROM or other nonvolatile memory before using this option because it could cause lots of write cycles in the EAROM and mess up whatever data is on there since it will be doing a write to every address on the board.

The Sig Dat Data Delay control allows you to change the point where the data is captured relative to the clock. If you slide the slider to the right, the data will be delayed and so you will be capturing data slightly before the clock edge. If you slide it to the left of zero, the clock will be delayed so you will be capturing data slightly after the clock edge. If you get an incorrect signature, you can try to slide this to the right to introduce some delay in the signature analysis data signal. You can also see how much “margin” you have by playing with this control since you can sometimes see how close to the clock edge the data is changing. In many cases it won’t make a difference.

The clock, start, and stop switches match their corresponding functions in the other existing testers so should be set according to the test procedure you are following.

The connections section lets you connect signals internally within the logic instead of needing to wire them out to the arcade board. It is recommended that you use at least the phase2 clock option and not connect the clock probe to phase2 on the arcade board unless you suspect that you are getting an incorrect signature. This option uses the phase2 clock signal that already goes to the tester and avoids loading the signal down with the oscilloscope probe. You can also connect stop to start because most signature analysis has them tied to the same point, and it is also recommend you do this instead of double-loading that signal. Finally, you can connect start to A15 which is used for most address decoder testing. So, with all 3 buttons pressed it is equivalent to start and stop on A15 and CLK on phase2.

Signature Wizard Tab

This is a guided signature analysis tool where you can load a configuration file and then the tool will provide guidance to test each signature in the list. Most of these files are set up in a specific order so you normally want to stop once you get an incorrect signature and investigate before proceeding.

Data Trace

The data trace tab allows you to capture the signature count or address where the data transitions. If you have the address counter enabled, then the count in the trace will correspond to the address where the transition occurred for each transition of the data. This means that you can easily probe chip select lines to see what ranges of addresses they are decoding. Put the data probe on the signal then click “run trace” and then click “read trace” when the trace is complete. The trace system uses the same triggering and sampling as the signature analyzer - it just lets you look at the actual data rather than just the signature. Also there is a limit to the number of transitions that the data trace can capture (about 1024 transitions)

Init sequence

This allows you to run a sequence of operations to set up the conditions for signature analysis. It can be looped to generate signature signals with the sequence (see the Battlezone mathbox tests)

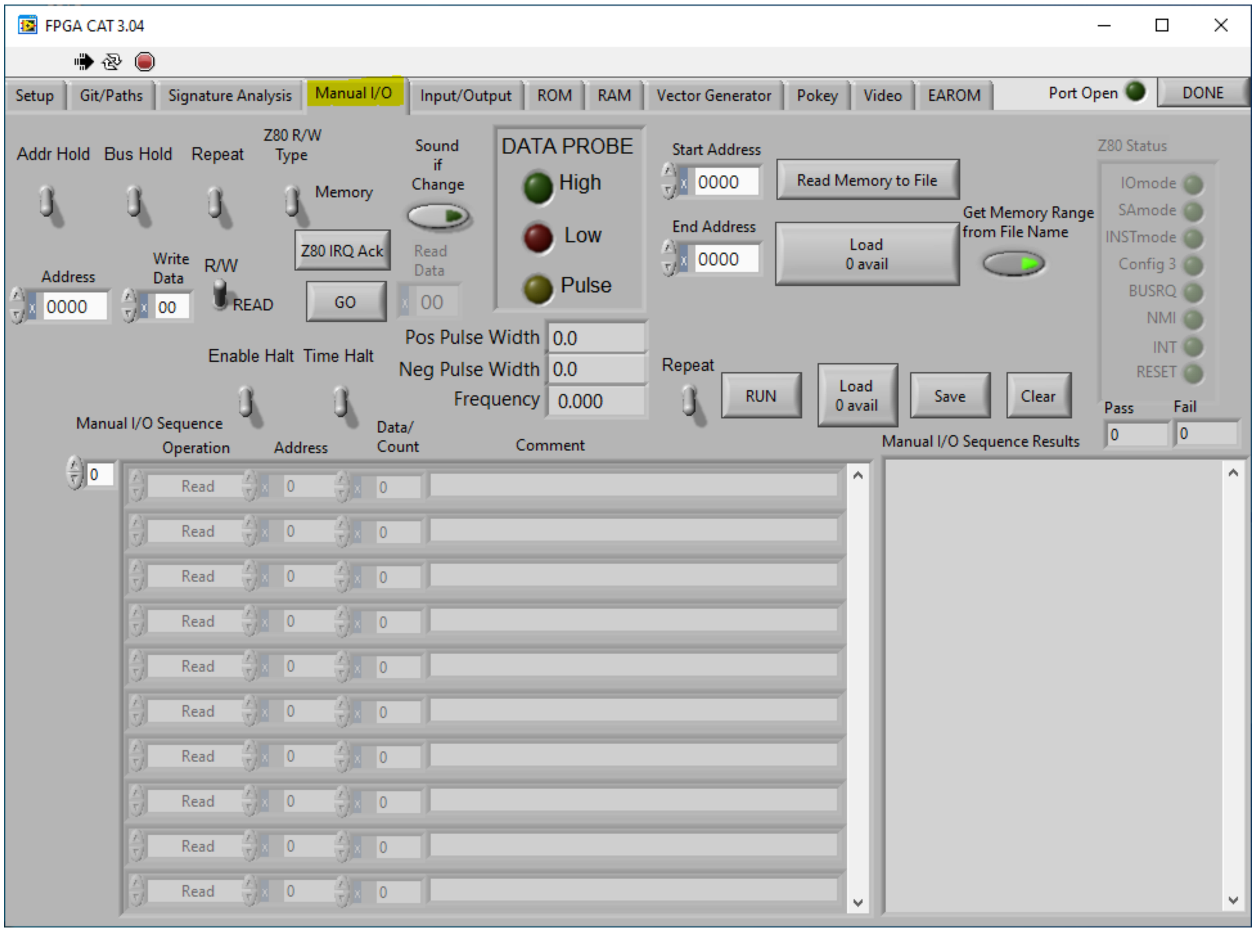
Auto ROM sig

If the ROM configuration is set up properly (the ROM type is correct and the signatures have been calculated from the ROM file), then you can automatically generate a signature wizard file for the ROMs on the board.

Probe Tuning

This is used to try to match the clock and data probes if you are using oscilloscope probes with compensation adjustments.

Manual I/O Test tab



On the manual I/O screen you can read and write individual memory addresses. The R/W switch controls if you are doing a read or a write. You would normally enter the address and click “go” to do a single read at that address. The read cycle will happen just once, then the board will go back to a “read” at the “idle address” that is set up in the board config (it is not possible for the 6502/6809 to do “nothing”, it has to be doing a read or write somewhere at all times). The same goes for write: you enter data to write to the address and click “go” and it will write once. If you click “repeat”, then the tester will repeat the operation over and over. Sound if Change will play a sound if the read value has changed – useful for testing inputs where you are paying attention to the board instead of the tester screen. There will be a delay (a few ms I think) between each read and write because of the interaction of the software with the tester.

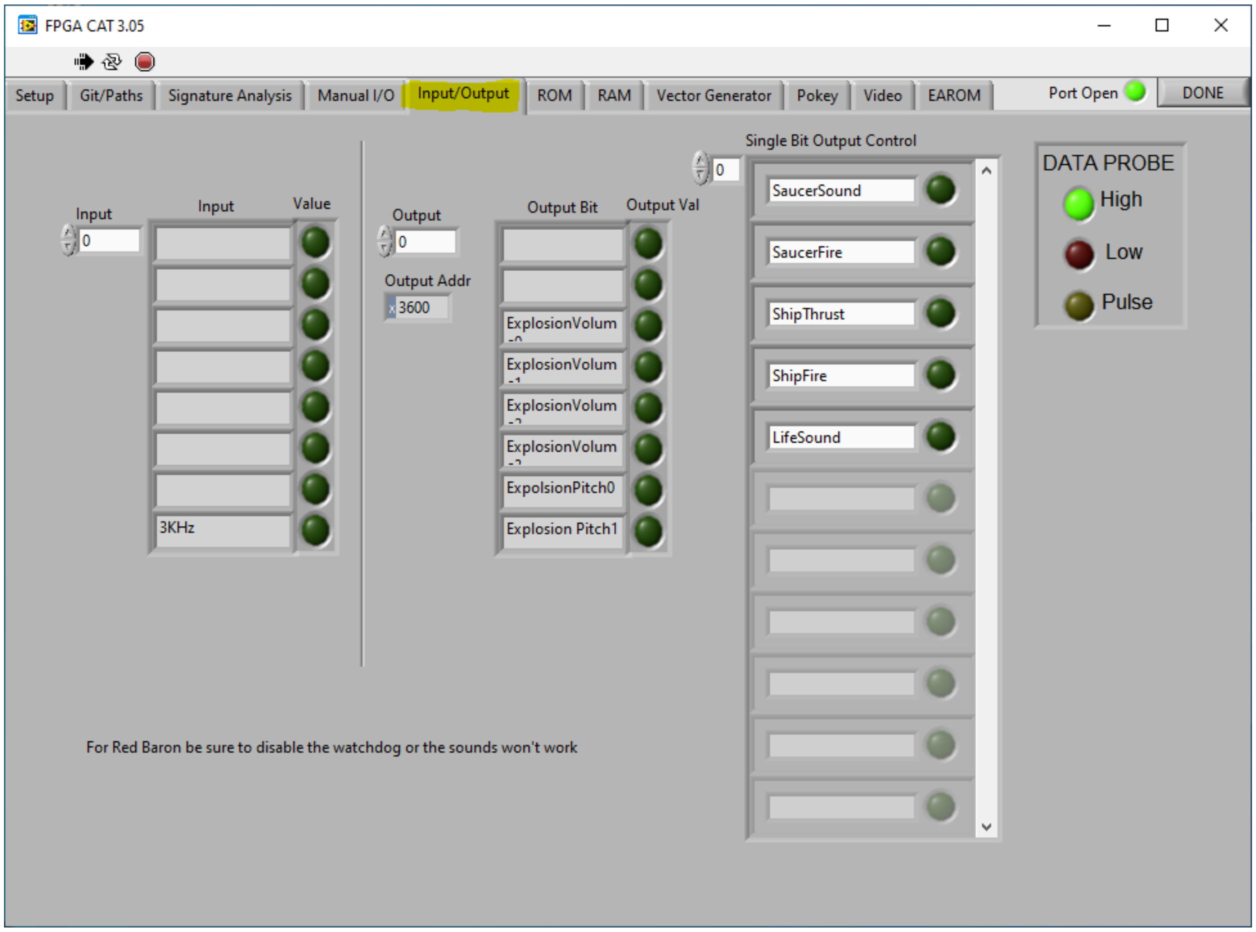
Clicking “address hold” will cause the address to stay on the address bus rather than going back to the idle address. This is useful for testing address bus buffers, etc. But if you do a write then it will still switch over to a read again and switch the data bus back to read mode. Bus hold will cause the data and R/W lines to “hold” after the read or write operation – this only really does anything for a “write” because it will continue to drive the data bus with data and continue to drive R/W low (write) continuously if bus hold is selected and you write something. This is useful for testing the data bus buffers in “write” mode if you want just static data on them. If you want to hold both address and data, do a write to some location with both bus hold and address hold switched on.

You can read and write memory to a file with the buttons on the top left. There is also an option to get the memory range from the file name. For example, the file name “DigDugScreenShot_8000-A007.bin” will put the contents of the file into the board’s memory from 0x8000-0xA007

You can also run an I/O sequence which uses a basic scripting environment to run a user-defined sequence of steps. This is a work in progress and needs more documentation but there are several examples especially for the Williams configs. (Play a tone out a sound board, cycle the digits on the 7 segment display of the ROM board, test the Blitter, etc).

The enable halt and time halt switches only apply to later versions of the tester which can grant the bus for 6809 Williams boards. You would want to switch both of these on if you are running the Blitter test.

Input/Output Test tab

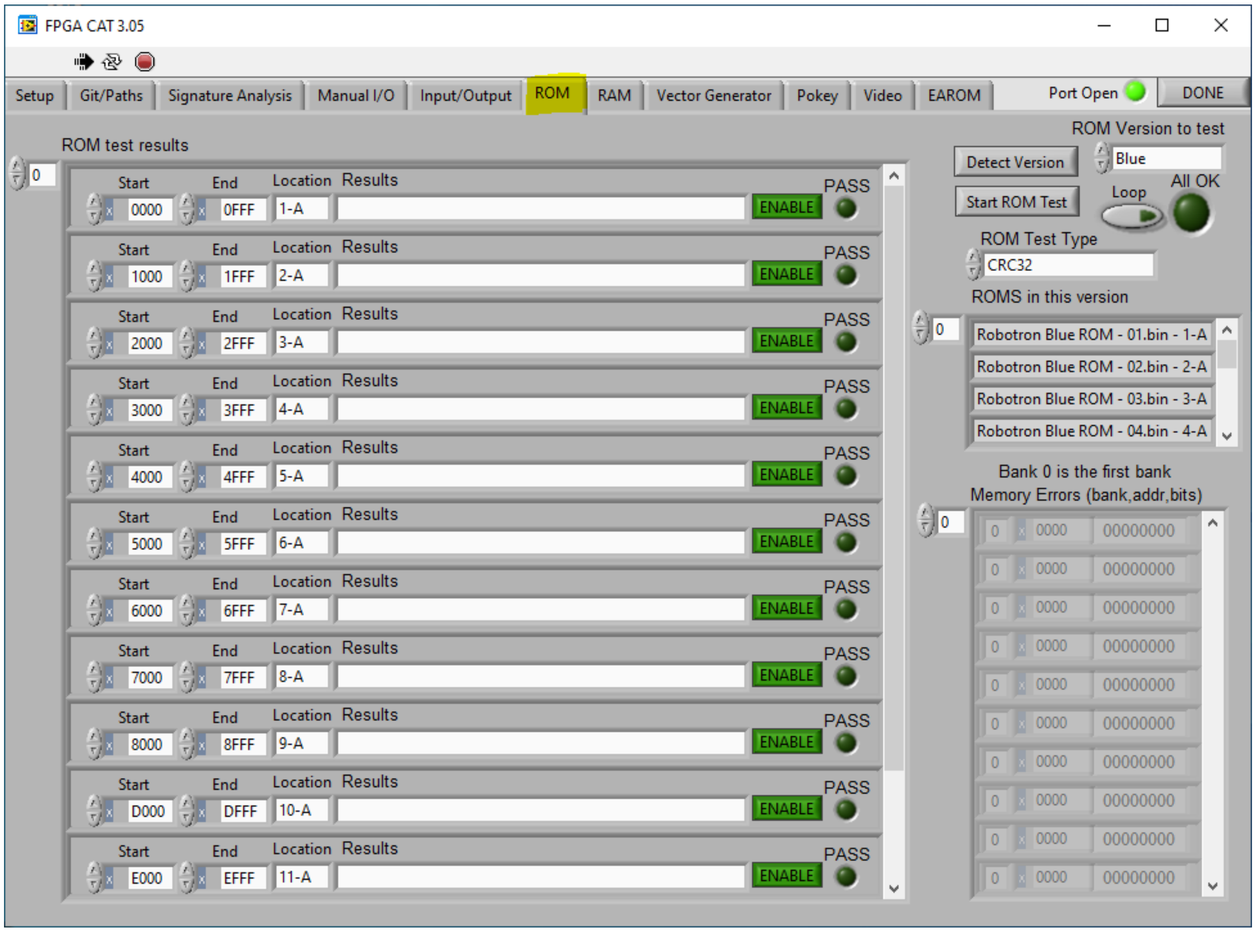


For Red Baron be sure to disable the watchdog or the sounds won't work

This screen will show the status of all of the inputs configured in the input blocks. I often look at the “3KHz” here as a first check to see that the board has at least some “life” in it.

For outputs, you can click the green LEDs next to the names of each of the bits to turn them on and off. If you are ever using this plugged in to an actual game cabinet then be careful not to leave the coin counters or other solenoids on for too long.

ROM Test tab



ROM Version to test: Blue

ROM Test Type: CRC32

Start	End	Location	Results		PASS
0000	0FFF	1-A		ENABLE	●
1000	1FFF	2-A		ENABLE	●
2000	2FFF	3-A		ENABLE	●
3000	3FFF	4-A		ENABLE	●
4000	4FFF	5-A		ENABLE	●
5000	5FFF	6-A		ENABLE	●
6000	6FFF	7-A		ENABLE	●
7000	7FFF	8-A		ENABLE	●
8000	8FFF	9-A		ENABLE	●
D000	DFFF	10-A		ENABLE	●
E000	EFFF	11-A		ENABLE	●

ROMS in this version

- Robotron Blue ROM - 01.bin - 1-A
- Robotron Blue ROM - 02.bin - 2-A
- Robotron Blue ROM - 03.bin - 3-A
- Robotron Blue ROM - 04.bin - 4-A

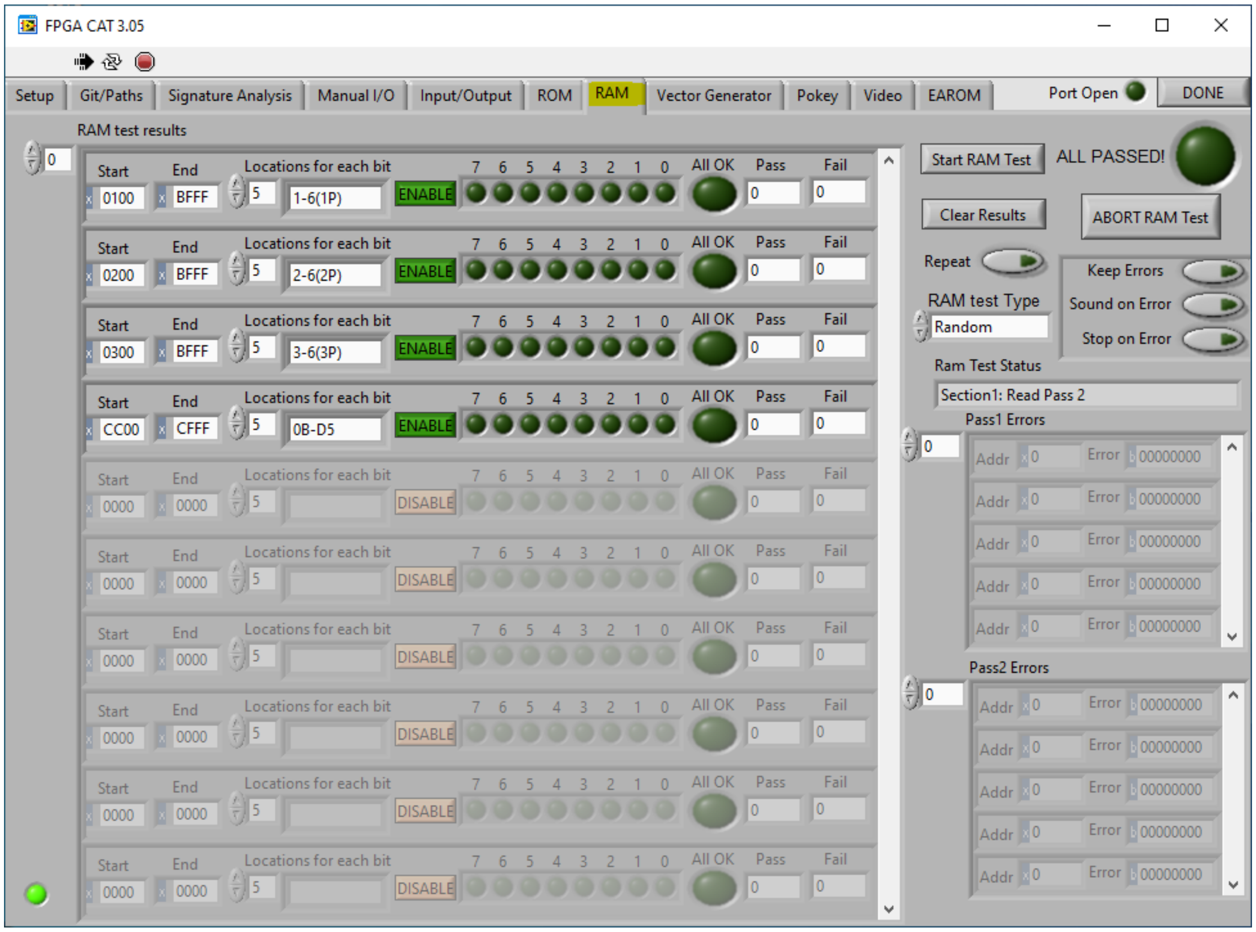
0	0000	00000000
0	0000	00000000
0	0000	00000000
0	0000	00000000
0	0000	00000000
0	0000	00000000
0	0000	00000000
0	0000	00000000
0	0000	00000000
0	0000	00000000

You should see all of the ROMs for the version you selected on the setup screen in the list, and then you can select the test type and hit "start rom test". You can compare to checksums/signatures, compare to full ROM files, and even read the ROMs from your board into files. When comparing to files, you will see any errors in the list to the right. A "1" in a certain bit position indicates an error in that bit. So if you have a failing data bus driver bit, you would see all the same bit wrong on the error list. Beware though - sometimes one ROM can mess up all of the others. I've even had a failed pokey that messed up the rom tests on one board.

You can disable some of the ROMS from the test process – This can be helpful when isolating one that is failing since you can just enable that one only and keep testing it. You can also loop the ROM test and the software will tell you how many times each has passed or failed.

One final note: Just remember that there is a scroll bar on the right – In the screenshot above you see that I can't see all the ROMs because there are 12 of them on Tempest. You can also have it scrolled down when there are fewer ROMs and not be looking at the top. That's why I put the big green "ALL OK" light on there.

RAM Test tab



FPGA CAT 3.05

Setup | Git/Paths | Signature Analysis | Manual I/O | Input/Output | ROM | RAM | Vector Generator | Pokey | Video | EAROM | Port Open | DONE

RAM test results

0

Start	End	Locations for each bit			7	6	5	4	3	2	1	0	All OK	Pass	Fail
x 0100	x BFFF	5	1-6(1P)	ENABLE	●	●	●	●	●	●	●	●	●	0	0
x 0200	x BFFF	5	2-6(2P)	ENABLE	●	●	●	●	●	●	●	●	●	0	0
x 0300	x BFFF	5	3-6(3P)	ENABLE	●	●	●	●	●	●	●	●	●	0	0
x CC00	x CFFF	5	0B-D5	ENABLE	●	●	●	●	●	●	●	●	●	0	0
x 0000	x 0000	5		DISABLE	●	●	●	●	●	●	●	●	●	0	0
x 0000	x 0000	5		DISABLE	●	●	●	●	●	●	●	●	●	0	0
x 0000	x 0000	5		DISABLE	●	●	●	●	●	●	●	●	●	0	0
x 0000	x 0000	5		DISABLE	●	●	●	●	●	●	●	●	●	0	0
x 0000	x 0000	5		DISABLE	●	●	●	●	●	●	●	●	●	0	0
x 0000	x 0000	5		DISABLE	●	●	●	●	●	●	●	●	●	0	0

Start RAM Test | ALL PASSED!

Clear Results | ABORT RAM Test

Repeat | Keep Errors | Sound on Error | Stop on Error

RAM test Type: Random

Ram Test Status: Section1: Read Pass 2

Pass1 Errors: 0

Addr	x 0	Error	b 00000000
Addr	x 0	Error	b 00000000
Addr	x 0	Error	b 00000000
Addr	x 0	Error	b 00000000
Addr	x 0	Error	b 00000000

Pass2 Errors: 0

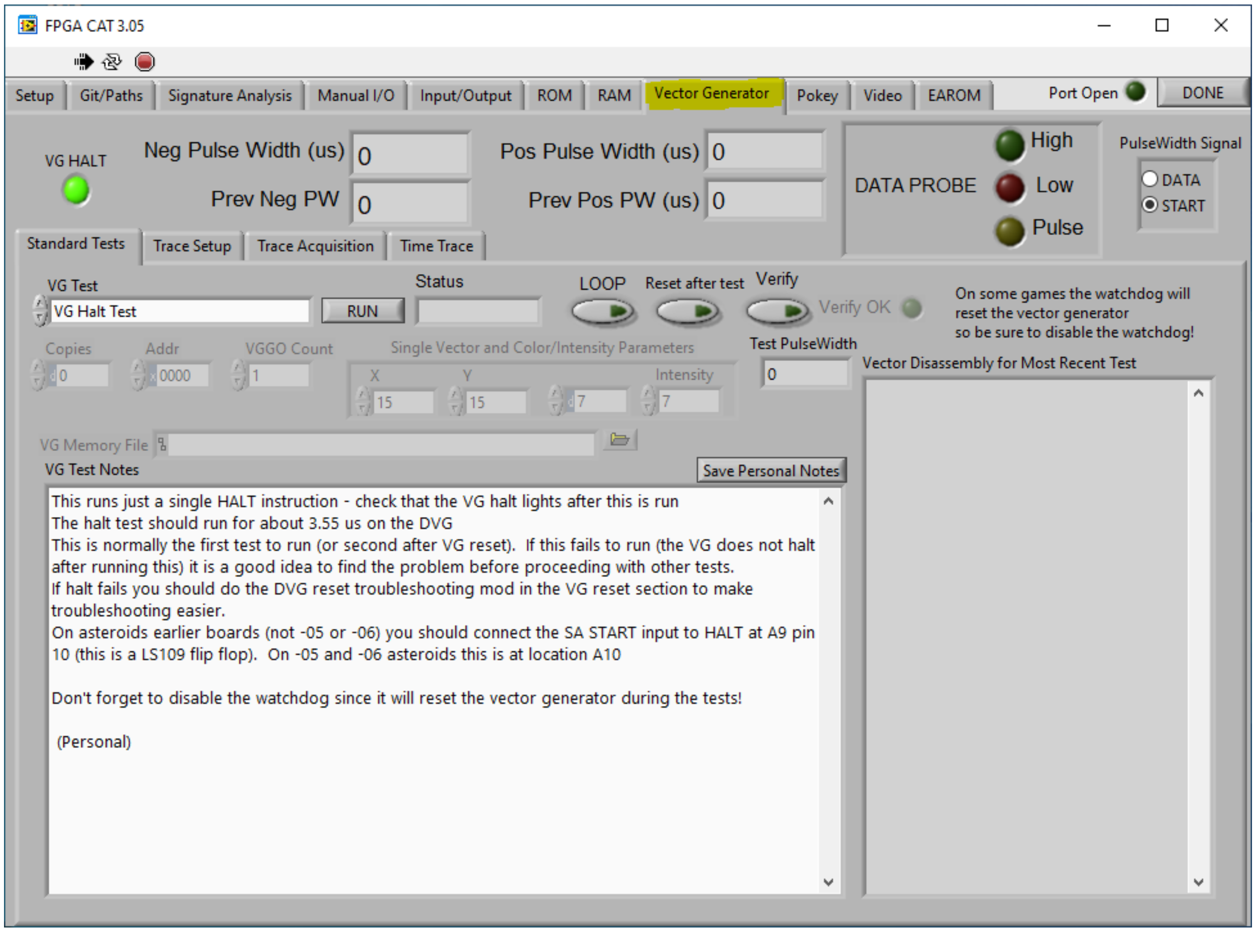
Addr	x 0	Error	b 00000000
Addr	x 0	Error	b 00000000
Addr	x 0	Error	b 00000000
Addr	x 0	Error	b 00000000
Addr	x 0	Error	b 00000000

The RAM test is similar to the ROM test. You can enable which RAMs you want to test, then it will test the low and high "nibbles" of each RAM. All tests write a pattern then read it, then write the inverse of the pattern and read it, and those are the two passes. Any differences show up in the errors list on the right. Like in the ROM test, a "1" in the binary Error list represents an error at that particular bit position. You can write random data, sequential data, FF then 00, or 00 then FF. The last two are interesting to try if you suspect an address bus problem since it will always write the same data everywhere in the memory so the test might pass even if there is a failing address line. Clicking "keep errors" will maintain errors between tests which might be useful for tracking down intermittent problems.

"Repeat" will continuously loop through the RAM test, tabulating pass and fail counts. "Sound on Error" plays an annoying sound for any RAM failure. "Stop on Error" will halt testing if a failure occurs.

The 8 green lights for each rom will turn green if that particular bit was 100% correct during the test. If you click on one of the green lights then the corresponding chip location will show up to the left of the lights.

Vector Generator Test tab



Neg Pulse Width (us) 0
Prev Neg PW 0
Pos Pulse Width (us) 0
Prev Pos PW (us) 0
PulseWidth Signal: DATA [✓ START]
VG Test: VG Halt Test
Status
On some games the watchdog will reset the vector generator so be sure to disable the watchdog!
Copies 0, Addr 0000, VGGO Count 1
Single Vector and Color/Intensity Parameters: X 15, Y 15, 7, Intensity 7
Test PulseWidth 0
Vector Disassembly for Most Recent Test
VG Memory File
VG Test Notes

This runs just a single HALT instruction - check that the VG halt lights after this is run
The halt test should run for about 3.55 us on the DVG
This is normally the first test to run (or second after VG reset). If this fails to run (the VG does not halt after running this) it is a good idea to find the problem before proceeding with other tests.
If halt fails you should do the DVG reset troubleshooting mod in the VG reset section to make troubleshooting easier.
On asteroids earlier boards (not -05 or -06) you should connect the SA START input to HALT at A9 pin 10 (this is a LS109 flip flop). On -05 and -06 asteroids this is at location A10

Don't forget to disable the watchdog since it will reset the vector generator during the tests!

(Personal)

The vector generator test section is used for testing Atari vector boards. This has been very useful to me when troubleshooting vector generator problems. This section of the manual needs to be updated but it should give a good starting point with focus on the Atari AVG games. A complementary section: “Vector Generator Testing - Atari DVG” can be found in Appendix A. The intent of which is to walk the user through vector generator testing of Atari DVG games (Lunar Lander, Asteroids and Asteroids Deluxe)

First, be sure that the vector RAM test passes before doing any of these tests. If it is not passing then you need to fix that before you can do much here.

It is recommended that you connect the signature analyzer start probe to the “halt” test point on the arcade board since you can get useful information even when you can’t yet get any vectors out. The “neg pulse width” indicator will show you how long the vector generator runs for each test. Be advised that it uses a 16 bit value clocked at 10MHz so you only get a maximum of 6553.5 microseconds.

Also it is strongly recommended that you use an X-Y oscilloscope and not an arcade monitor for these tests . The best way to see the vectors is using an older analog oscilloscope with X-Y mode but be careful to not crank up the intensity too high and leave the spot in one place for too long (like all day which I have done too many times on mine). If yours has a Z input you might be able to figure out how to blank the beam when the vector generator is halted. I am currently using a Keysight DSOX1102 digital scope which has the best X-Y mode of any sub-$1000 digital oscilloscope I have seen (you have to set it up in high resolution mode and play around with the settings a little and it isn’t as nice as analog but it doesn't burn in). The Rigol DS1054Z is a cheaper (and 4-channel) scope that can do X-Y enough to see an image for these tests. For some reason the Rigol does not let you use high resolution mode in X-Y, but you can improve the image by changing the memory depth and sweep rate. I have several scopes so I just leave one set up for the X-Y display and have another one I use for tracking down issues on the board.

Here is a brief overview of each test:

For all tests but reset : The "count" control lets you hit "VGGO" a certain number of times after writing the test to the vector memory – it helps to make the image more visible on the scope.

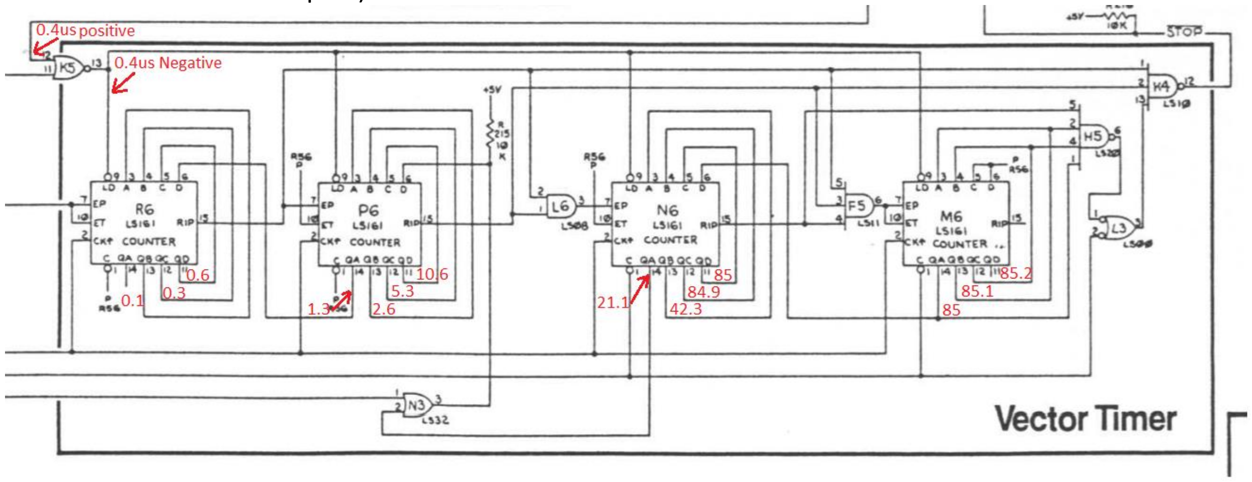
- VG Reset: This is the only test that doesn't write to "VGGO". It instead writes to the reset location to reset the vector generator. This is sometimes useful if the vector generator gets into a strange state during your testing and you want to stop it.
- VG Halt: This is always the first test I run on a vector generator. The vector program consists of just a single halt instruction which should take about 2 microseconds. I have seen situations where it will never halt, or where it will take a lot longer than 2 microseconds. If that happens, you should start troubleshooting the basics of the vector generator (address counter, halt flip flop, etc.) before moving further.
- VG Center: Now that you have the halt working, you can try to center the beam. On an analog vector generator (anything but Asteroids, Asteroids Deluxe, Lunar Lander) This should take about 89 microseconds or so (probably anywhere around this is fine). This uses some of the vector timers to zero the integrators. If this fails then you have a vector timer problem. I forget what I did on the digital vector generator for this test but I don't think it really applies (it will do something but not something very useful if I remember – maybe just go to the center?).
- VG Draw Single Short Vec from RAM: This draws a single short vector – the simplest thing you can do and actually get any output. It should take about 26 microseconds.
- VG DrawShort from RAM: Now we draw a capital "G" (might be different on the DVG) on the screen which is a little more complicated. All of the draws still use the short vector instruction which is the easiest thing for the vector generator to draw (since it only requires one 16-bit instruction)
- VG DrawShort from RAM BIG: What this does is allow you to put in lots of copies of the short draws from the previous test. You will notice that the "copies" control becomes active when you select this option. The image will look more "bold" on the screen the more copies you run. Be careful not to create too many copies or it will fill up the memory – but it is a good idea to "almost" fill up the memory because if that works then your address counters are good. I had a failing upper bit or bits in the address counter that this test caught once or twice. At the time of this writing, you can do 68 copies for the digital vector generator and it varies on the AVGs depending on the game (on the AVG each copy takes 14 bytes because it has 7 instructions). You can do 292 copies in Battlezone with its 4K vector memory. On the analog vector generator this uses just the first 2 vector timer counter chips so if this works but the regular vectors do not the vector timer is one possible place where there could be a problem.
- VG Draw from RAM: Now we are going to draw some "regular" vectors that are not short. This requires two 16 bit fetches from the vector memory plus involves all the vector timer counters so it is more complicated for the vector hardware to accomplish. Right now, on the DVG it will draw an asteroid (of course) and on the AVG it will draw the "gun sight" from Battlezone except on Space Duel which draws a "star thing". On the AVG it takes around 1173uS on Tempest, but the "star" in Space Duel takes about 454us.
- VG JMP Test RAM: I have actually gotten all the way to this point and still had issues a couple of times. The JMP test just creates a few of the short draws next to each other if it is working. Instead of them being all in a row in

the program, the program jumps around using the JMP instruction. If this fails you may have an issue with the address shift registers/latches. It should take about 360uS on the AVG.

- VG JMP only no draw. This creates a bunch of "JMP" instructions that just jump to the next address. Each one should take about 2uS. If it is taking longer than it should, then you may have a problem with the address shift registers/latches.
- VG JSR to address. This needs some work because right now it is the only way to do a JSR and you need to know a good address to jump to in the vector rom. I have had issues where everything works up to this point but the "register file" chips that hold the stack have failed and JSR does not return properly. In newer games the stack is in the AVG chip. I need to make another test that has some JSRs within RAM (maybe "copies" number of them nested to get to different levels of the stack) and I will try to remember to add that shortly.
- JSR test – on Battlezone and Tempest you should be able to get 4 levels of nested jumps. You can also place the location of the first jump at some offset in vector memory to test the full address space since the address is spread across multiple LS670 chips. On the AVG chip, you can only do 3 levels of nested jumps since it has just a 3 level stack (but it has one extra address bit compared to the "discrete" stack). The jump test is not yet implemented in the digital vector generator at the time of this writing.

Here are some vector generator tests that can be used with the data probe or an oscilloscope to narrow down problems in various parts of the circuit:

"Center" test - tests the vector timer and the normalization pulse circuit (Chip locations and pin numbers refer to space duel – see note below about Tempest)



We want to test the pulse width measured with the tester when running the "center" test. I connected the data probe ground to an adjacent ground test point since we will be probing K5, R6,P6,N6, and M6 (the 161 counters).

During the "center" operation, the "normalization flag" should pulse low for 400ns at K5 pin 12 If this is not happening then there's no need to go further since this is what shifts a bunch of "1"'s in to the vector timer to make it go shorter. There is not a lot of timing resolution but the FPGA cat box can be used to test this. Then we will test the counter outputs that should be counting so they will have longer and longer pulse width as we go "up" the counter until we get to the top where the 1's have already been shifted to make it stop at 89us.

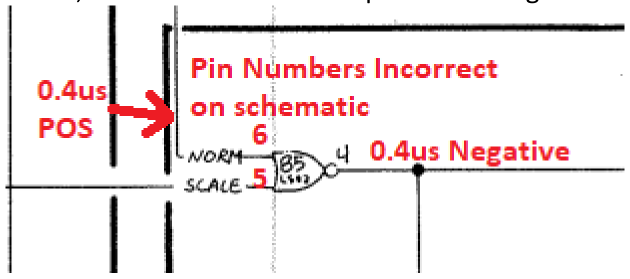
Sometimes I had to click "run" twice in the software to get these numbers - not exactly sure why it doesn't always get a good count the first time but maybe it is because those lower counters don't get cleared by anything.

Here are the pin numbers (where to put the data probe) and the timing you should read after running the VG center test on Space Duel. The first two are the "normalization" pulse that will shift some ones into the upper bits of the vector timer, making it expire much sooner.

K5-12	0.4us positive pulse width
R6-9	0.4us negative pulse width
R6-13	0.1us positive pulse width
R6-12	0.3us pos
R6-11	0.6us pos
P6-14	1.3us pos
P6-13	2.6us pos
P6-12	5.3us pos
P6-11	10.6us pos
N6-14	21.1us pos
N6-13	42.3us pos
N6-12	84.9us pos
N6-11	85us pos
M6-14	85us pos
M6-13	85.1us pos
M6-12	85.2us pos

I think this is a pretty good test of the vector timer counters that will probably work in all analog vector generators and I will probably put it into the manual.

NOTE: In the Tempest boards I've tested, there is an error in the pin numberings for the upper OR gate:



Scale test:

The scaling circuit is just used for vector drawing (not timing the "center" pulse). You can use the "AVG set scale" test in the software and set the bin scale from 0 to 7 and the 3 bit scale will show up on the output of the vector scaling latch (D7) (you can read it with a logic probe or even a voltmeter- it stays there after the test). Then if you do a "draw single short vector" you will see that the "load" pulse (pin 9 of the 161 counters) varies in length from no pulse (bin scale 0) to a 400ns long low pulse (bin scale 7). The other vector tests do a "center" at the beginning (I should remove this) so they generate a normalization "load" pulse along with the "scale" pulse so it is harder to measure them.

A note about the short vector test:

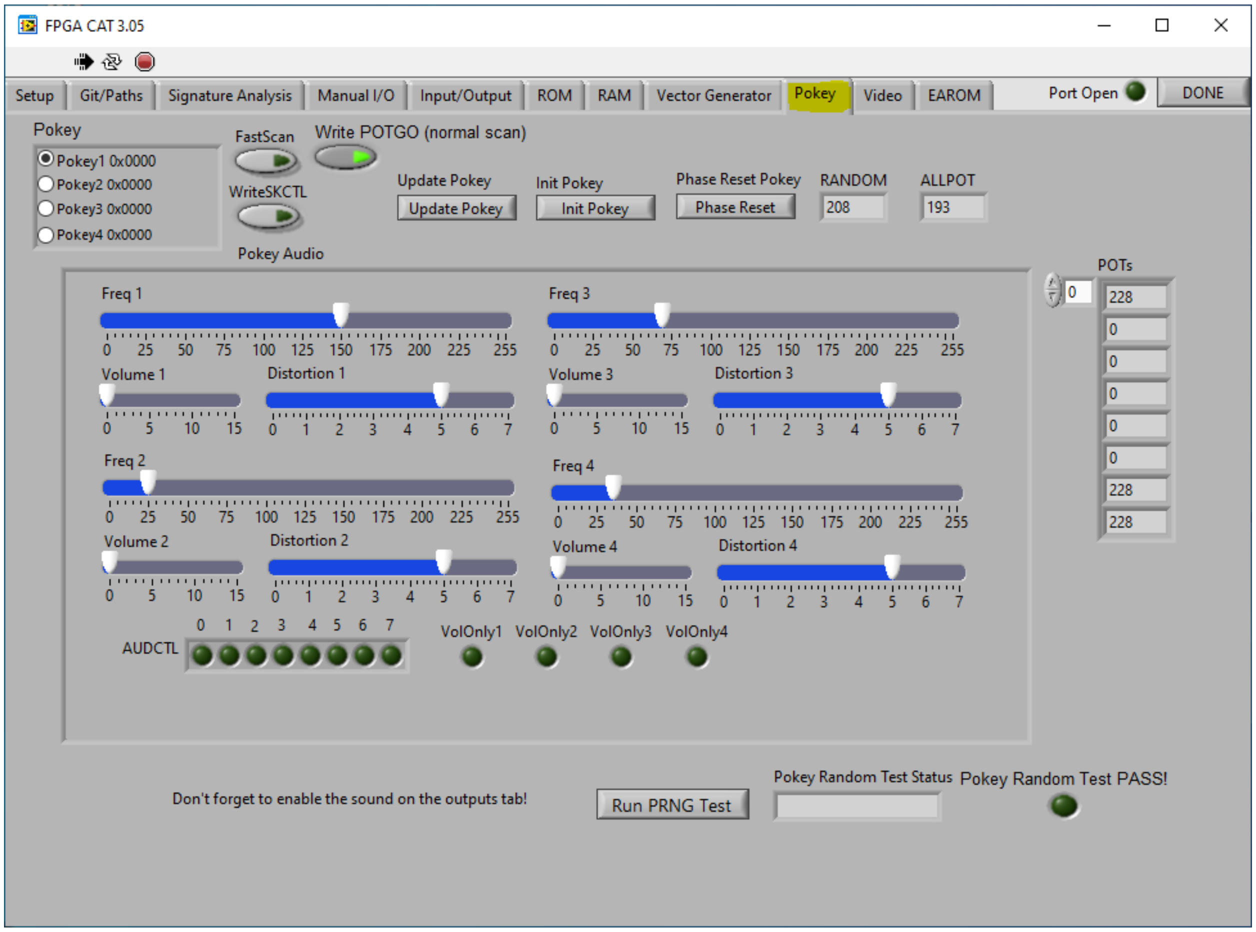
Short vectors only use the two lower-order vector timers and the upper two are ignored. If short vectors are working and long vectors are not, one or both of the upper two counters could be bad.

General notes about the vector timer on the AVG (DVG is different):

If you look at the vector timer, you will see that the upper-most load input is tied high (M6-pin5 on space duel). Also there is an "or" gate forcing the last load input of the second counter to "1" when in short vector mode (since just the first 2 counters are used for short vectors). Then each output bit of the timer is tied to the previous bit's load input. The LS161's have a synchronous load and are being clocked by a 12MHz clock, so this shifting action loads "1"s into the upper bits of the counter which get shifted down the longer the load bit is active. Since the vector timer "expires" when it is all 1's (or for short vectors when the lower 2 counters are all 1's) this has the effect of dividing the expiration time of the vector timer by powers of two. So if the load inputs (pin 9 on the LS161s) are active for one edge of the 12MHz clock, it will divide the vector time by 2, if they are active for 2 edges, it will divide by 4, etc. This variable-length load pulse is generated by the vector scaling counter (and LS191 at C7 on space duel). For double-size vectors (scaling of 0), there is no load pulse when vectors are drawn.

Because this is an analog vector generator, the outputs of the DACs are integrated and represent the "rate of change" of the actual X-Y vector output voltages. So driving the vector for half the time makes it half the size, etc.

Pokey Test tab



Pokey: (● Pokey1 0x0000) ○ Pokey2 0x0000 ○ Pokey3 0x0000 ○ Pokey4 0x0000

RANDOM 208

ALLPOT 193

POTs 0

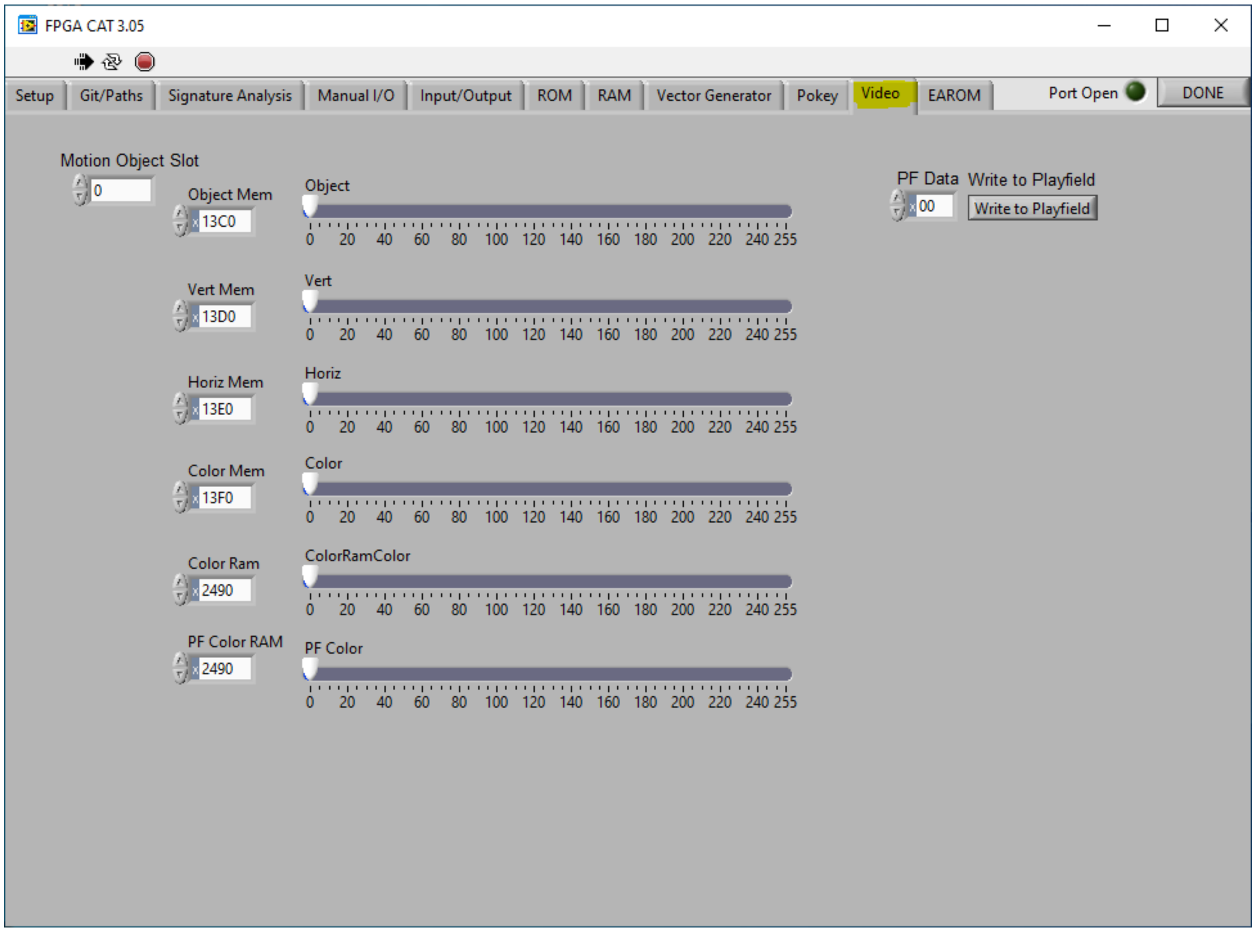
228
0
0
0
0
0
228
228

Don't forget to enable the sound on the outputs tab!

Pokey Random Test PASS!

The POKEY test lets you test just about everything arcade-related on the pokey. First you can select which pokey you are testing in the upper left corner. Then if you hit “Init Pokey” you should see the random numbers start changing. You can also hit the “Run PRNG test” button to test the pseudo-random number generator. The tester is able to reset the pokey and read back the random values one for each bus cycle (even the CPU can’t even do that since it has to fetch instructions). The results are compared against a known set of random numbers that should come out of the pokey. I have also seen pokeys with some channels “dead” and you can hook up some speakers (I use old computer speakers with a series capacitor) to the audio test points and turn up the volume on each channel and you should hear the different tones. By default it will play tones but you can set the distortion to play different types of noise, etc. You can also look at the “POTs” values which are the individual pot readings (usually connected to switches on the arcade boards) and ALLPOT should also read the “digital” value of the inputs at the POT pins of the pokey as long as you have “write POTGO” selected for a normal scan.

Video Test tab



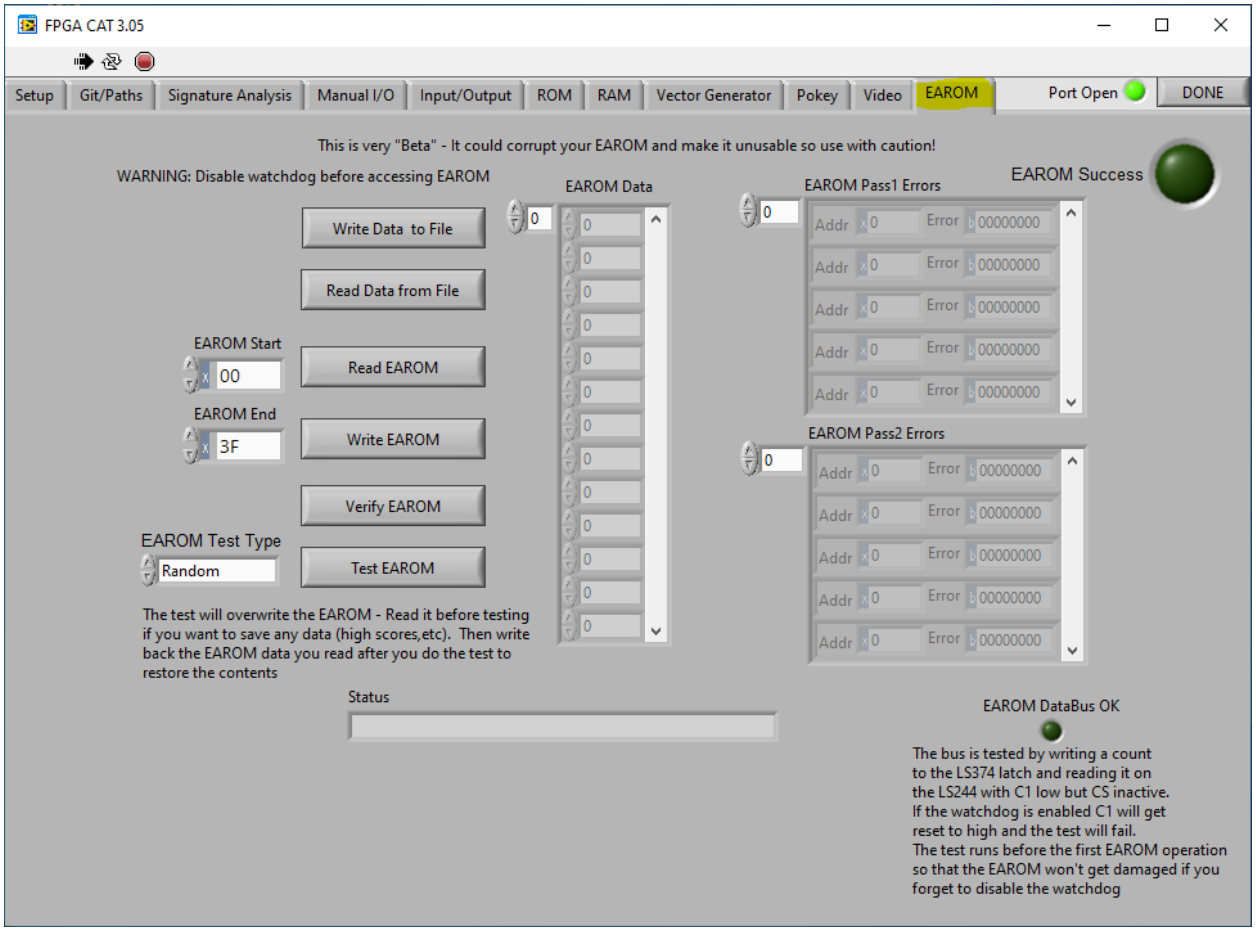
Motion Object Slot: 0

Field	Value	Slider
Object Mem	x13C0	Object
Vert Mem	x13D0	Vert
Horiz Mem	x13E0	Horiz
Color Mem	x13F0	Color
Color Ram	x2490	ColorRamColor
PF Color RAM	x2490	PF Color

PF Data: x00

The Video Test tab is a work in progress and is for raster games– I still need to figure out the details of the colors of the "Motion Objects" in Centipede and Millipede, but this lets you move the sprites (Motion Objects) around on the screen and pick which object is in each "slot", etc. I used this to fix some graphics problems in Millipede but I don't have it 100% figured out. (If someone has a good handle on programming the graphics on raster games please let me know). You can also fill the playfield with characters (or zero to blank it), etc.

EAROM Test tab



This is very "Beta" - It could corrupt your EAROM and make it unusable so use with caution!

WARNING: Disable watchdog before accessing EAROM

EAROM Start: 00

EAROM End: 3F

EAROM Test Type: Random

The test will overwrite the EAROM - Read it before testing if you want to save any data (high scores,etc). Then write back the EAROM data you read after you do the test to restore the contents

The bus is tested by writing a count to the LS374 latch and reading it on the LS244 with C1 low but CS inactive. If the watchdog is enabled C1 will get reset to high and the test will fail. The test runs before the first EAROM operation so that the EAROM won't get damaged if you forget to disable the watchdog

Atari used the ER2055 EAROM (512 Bit Electrically Alterable Read Only Memory) for saving high scores on a number of early 80's games (Asteroids Deluxe, Black Widow, Centipede, Dig Dug, Gravitar, Liberator, Millipede, Red Baron, Space Duel and Tempest). Here you can test, read, save and write the EAROM data. As noted you must disable the watchdog before accessing the EAROM.

Final Remarks:

If you are still reading this then thanks for reading through the manual such as it is! I hope to have more diagnostic information and new features added in response to everyone's feedback. Thanks for being a beta tester and for your interest in this project!

Appendix A: Vector Generator Testing – Atari DVG

Due to the circular nature of the Atari vector generator, traditional troubleshooting techniques are often ineffective. This document walks through Vector Generator testing for the Atari Digital Vector Generator (DVG). The VG Trace feature lets you assemble and run a vector program and trace the state machine in a "logic analyzer style", also allowing signal comparison with a known working board.

The DVG was used in early Atari vector games: Lunar Lander, Asteroids and Asteroids Deluxe. Later Atari vector games used the Analog Vector Generator (AVG) which has many similarities to the DVG.

Background/Reference material:

Fred's Introduction to Vector Generator testing with a Star Wars board (AVG):

Standard Tests:	https://youtube.com/watch/BcpguZCA9B0?start=19m17s
Trace Setup:	https://youtube.com/watch/BcpguZCA9B0?start=32m53s
Trace Acquisition:	https://youtube.com/watch/BcpguZCA9B0?start=38m15s
Time Trace:	https://youtube.com/watch/BcpguZCA9B0?start=40m20s

Fred's complete FPGACat video playlist… in addition to the above:

https://youtube.com/playlist?list=PLDe31EDCUSj19TSReRufoCrQVGdv7IZWS

Excellent training videos by Douglasgb and Rotormoshun using Trace with a Space Duel board (AVG):

Trace Setup:	https://youtube.com/watch/5OqD4fNd24M
Trace Acquisition:	https://youtube.com/watch/f9ZLQXmg93o
Trace Results:	https://youtube.com/watch/kyQKzRe4qC0

FPGACat complete training video playlist...in addition to the above:

https://www.youtube.com/playlist?list=PLVD-zkkSvyEVWdcKeYNyCHcRgGxiabkLg

DVG State Diagram (Figure 1), useful for determining where the Vector State Machine went awry:

https://drive.google.com/file/d/1mqwfwRpCVLqegHbSJx7lCfctvMnvues1/view?usp=sharing

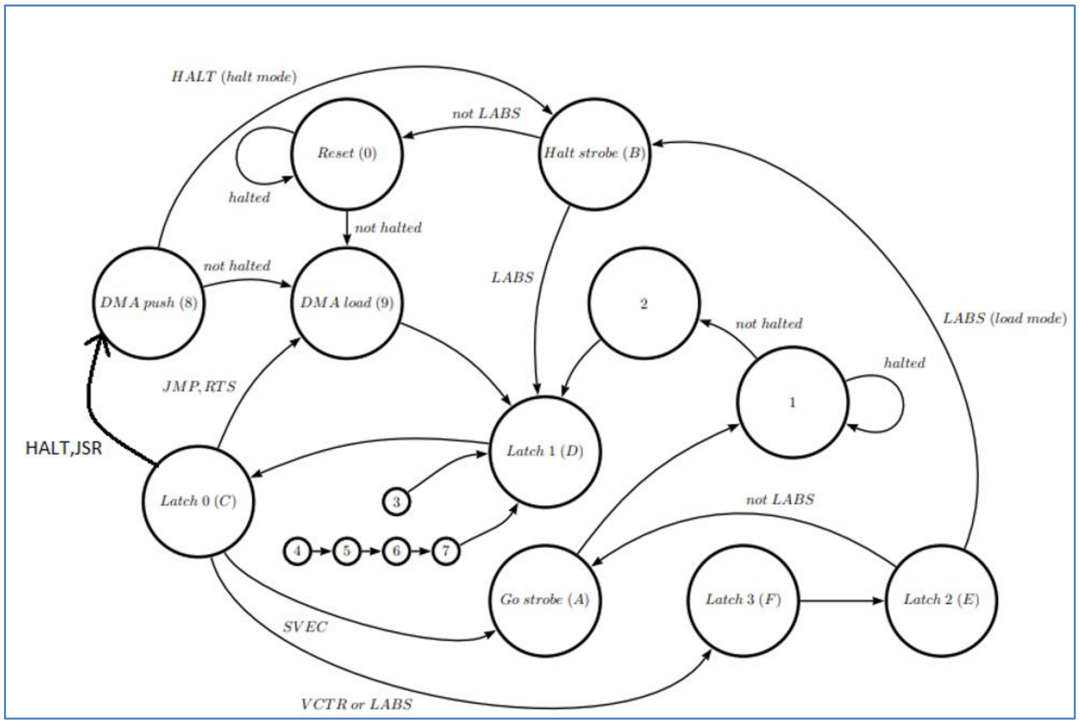


Figure 1 - DVG State Transition Diagram

Asteroids DVG State Machine schematic, annotated with signal names used in the configuration file (Figure 2)

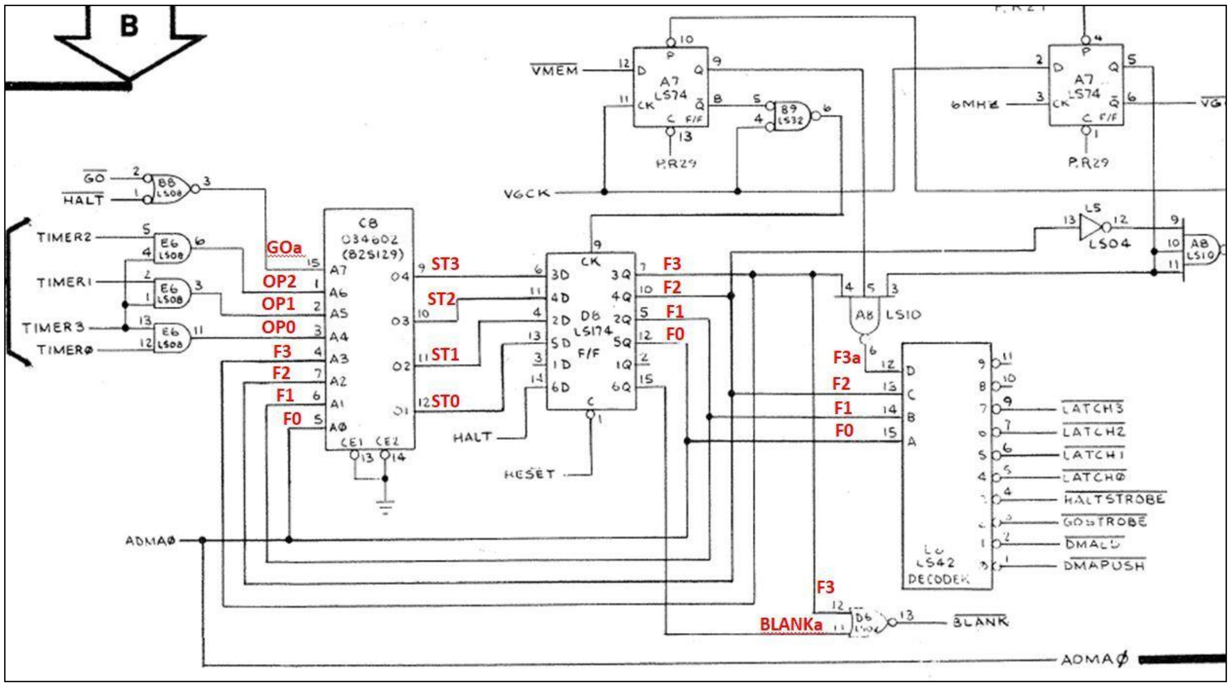


Figure 2 – Annotated State Machine Schematic

Asteroids & Lunar Lander board modification

For the Trace feature to work correctly the tester must be able to reset the DVG, otherwise the vector generator traces won't be starting from the same state each time. However there is no Vector Generator reset on either Asteroids or Lunar Lander, requiring the temporary board modification described below.

Lift the negative side of 1uF electrolytic capacitor C25 (C26 on Lunar Lander) located between A6 and then jumper 74LS42 L6 pin5 /DMACNT to the positive side of C25, this will allow the tester to reset the DVG (Figures 3A/B/C). Be sure to reconnect C25 when done testing.

Note: this modification is not required for Asteroids Deluxe or any AVG boards.

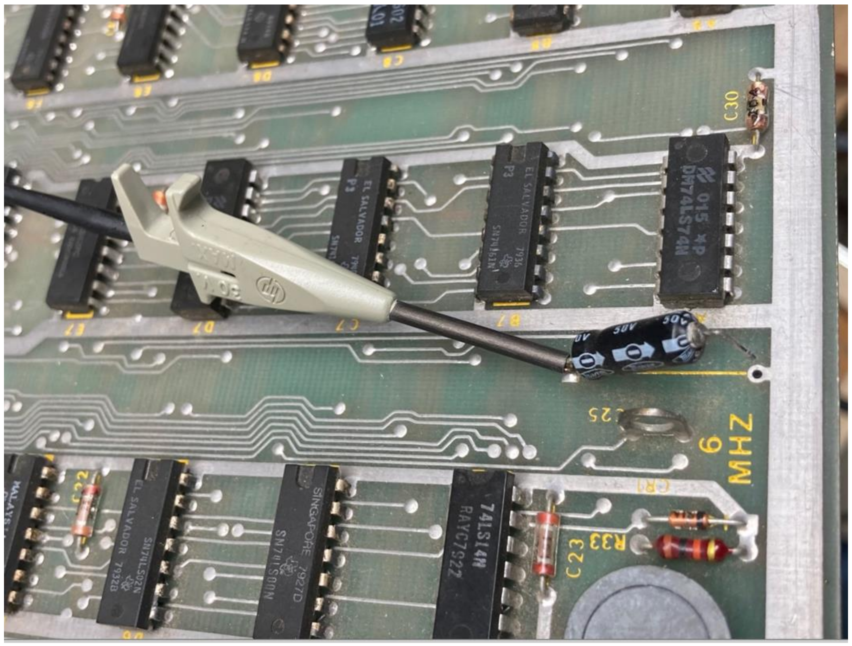
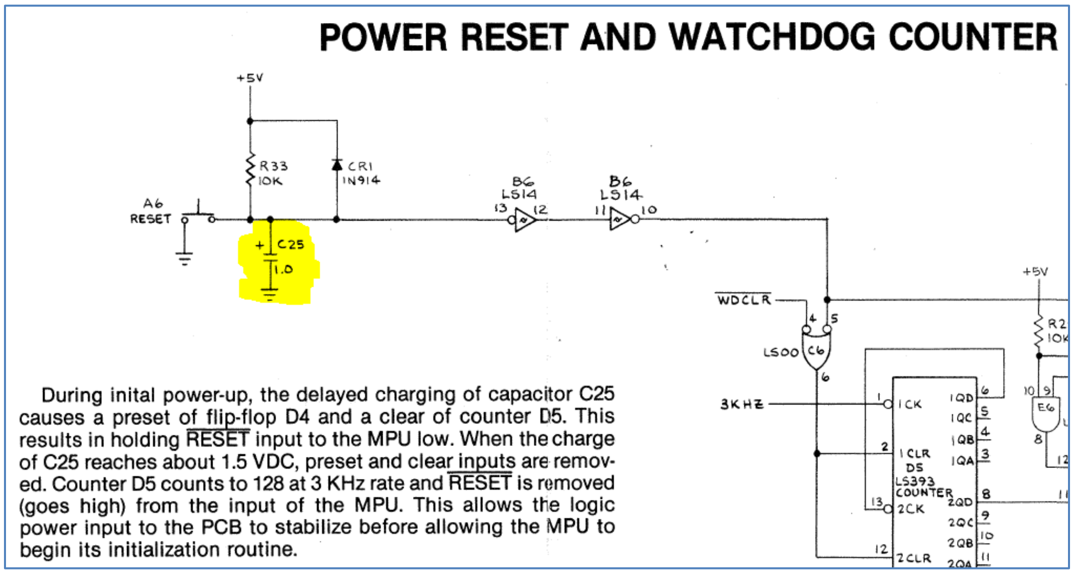


Figure 3B – C25 board modification



During inital power-up, the delayed charging of capacitor C25 causes a preset of flip-flop D4 and a clear of counter D5. This results in holding $\overline{\text{RESET}}$ input to the MPU low. When the charge of C25 reaches about 1.5 VDC, preset and clear inputs are removed. Counter D5 counts to 128 at 3 KHz rate and $\overline{\text{RESET}}$ is removed (goes high) from the input of the MPU. This allows the logic power input to the PCB to stabilize before allowing the MPU to begin its initialization routine.

Figure 3A – C25 Reset Circuit

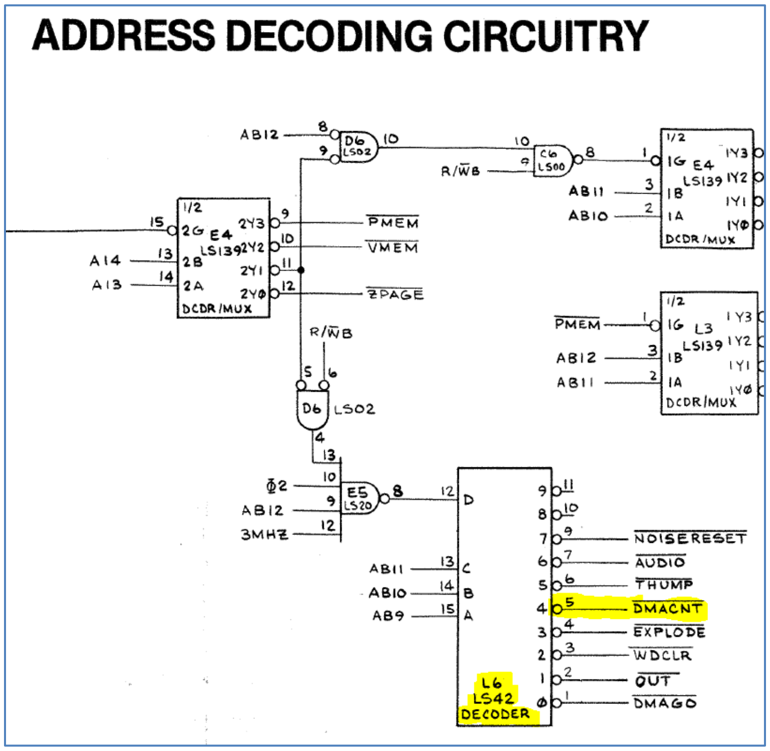
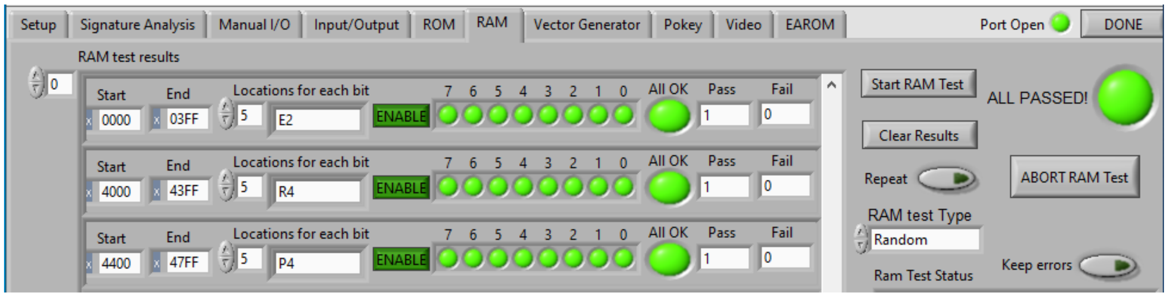


Figure 3C – Address Decoder /DMACNT

Vector Generator Testing Initial Setup

Step 1: Make sure the Vector RAM tests are all good, as you will obviously be writing to this memory.



RAM test results

0

Start	End	Locations for each bit			7 6 5 4 3 2 1 0	All OK	Pass	Fail
0000	03FF	5	E2	ENABLE	● ● ● ● ● ● ● ●	●	1	0
4000	43FF	5	R4	ENABLE	● ● ● ● ● ● ● ●	●	1	0
4400	47FF	5	P4	ENABLE	● ● ● ● ● ● ● ●	●	1	0

ALL PASSED! ●

Repeat

RAM test Type: Random

Ram Test Status

Keep errors

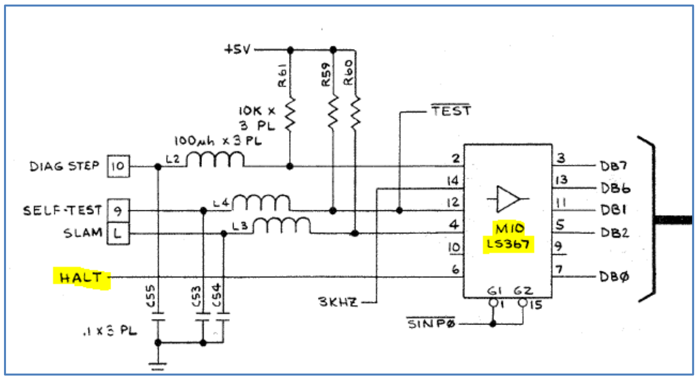
Step 2: Disable the Watchdog by grounding the WDOG DISABLE test point located at D5. If the Watchdog is not disabled it will keep the DVG from running intermittently, producing erratic results.

Step 3: Connect the tester's START probe to HALT (74LS109 A9 pin 10) and the CLOCK probe to the 6MHz Test Point (adjacent to C25) or use 74LS74 A7 pin3.

Note: For DVG testing it is necessary to use the 6Mhz clock instead of VSMCLK (used on the AVG) to allow detection of the 74LS42 Decoder outputs at E8 (i.e. LATCH0/1/2/3/etc.).

Step 4: Confirm the tester can both start and stop the vector generator. From the "Standard Tests" tab (covered in the next section) run the "VG Reset" test, the software VG HALT light should be on. Next run the "JMP 0" test, this creates a loop and should result in the VG HALT light turning off. Finally run "VG Reset" again, this should stop the running program and the VG HALT light should turn back on.

If you don't get these results, circle back and check your setup. It's also possible you have an issue with the Reset circuit, the A9 74LS109 circuit or the HALT signal to the CPU (Asteroids L10 74LS251 and Lunar Lander M10 74LS367 - shown in Figures 4 and 5).



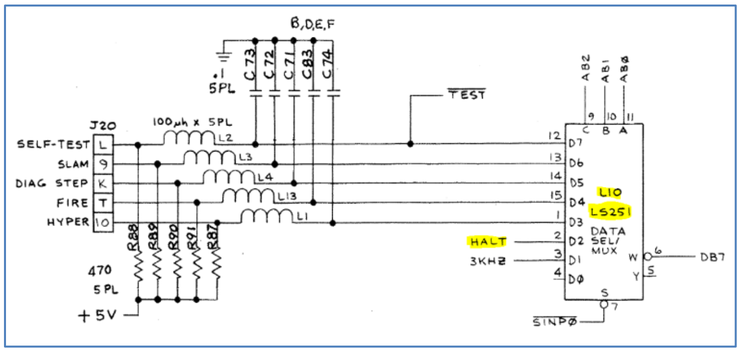


Figure 4 - Lunar Lander HALT to CPU

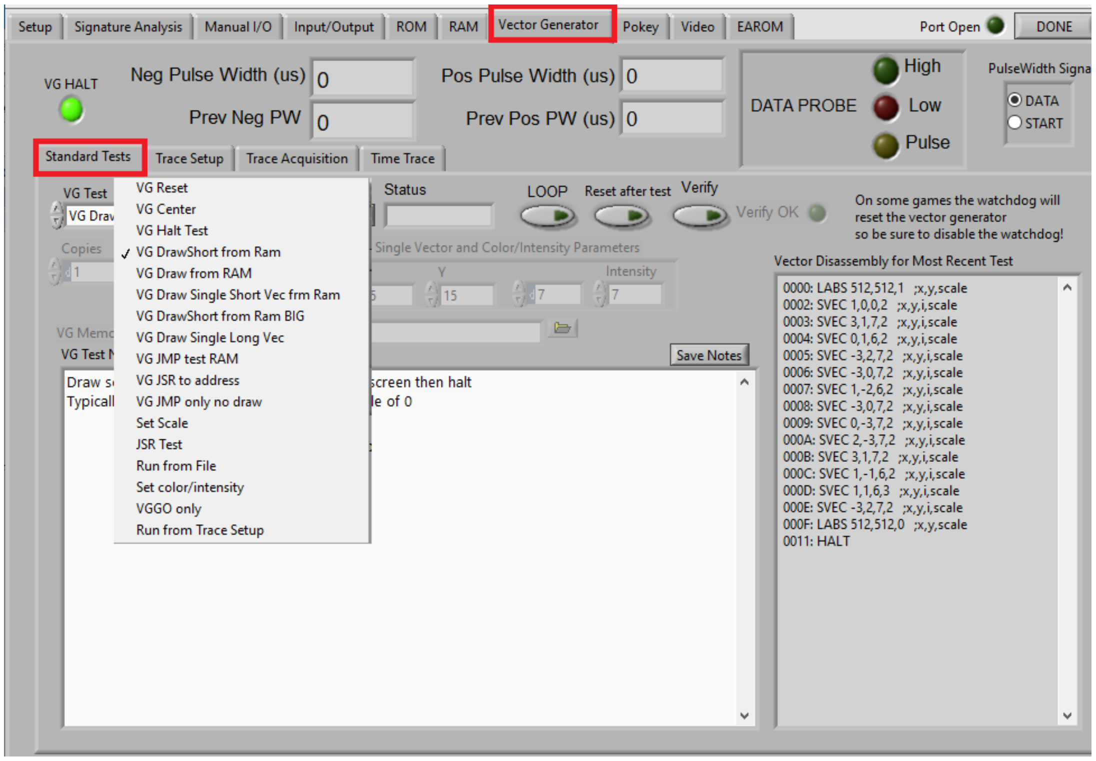
Figure 5- Asteroids HALT to CPU

(Note: inverted output on pin 6)

Vector Generator Standard Tests

https://youtube.com/watch/BcpguZCA9B0?start=19m17s

The “Standard Tests” tab is shown in Figure 6. The “VG Test” pull-down menu lets you choose a test of interest and execute it by hitting “RUN”. The “Neg Pulse Width” indicator will show you how long the vector generator runs for each test.



On some games the watchdog will reset the vector generator so be sure to disable the watchdog!

Vector Disassembly for Most Recent Test

```
0000: LABS 512,512,1   ;x,y,scale
0002: SVEC 1,0,0,2   ;x,y,i,scale
0003: SVEC 3,1,7,2   ;x,y,i,scale
0004: SVEC 0,1,6,2   ;x,y,i,scale
0005: SVEC -3,2,7,2   ;x,y,i,scale
0006: SVEC -3,0,7,2   ;x,y,i,scale
0007: SVEC 1,-2,6,2   ;x,y,i,scale
0008: SVEC -3,0,7,2   ;x,y,i,scale
0009: SVEC 0,-3,7,2   ;x,y,i,scale
000A: SVEC 2,-3,7,2   ;x,y,i,scale
000B: SVEC 3,1,7,2   ;x,y,i,scale
000C: SVEC 1,-1,6,2   ;x,y,i,scale
000D: SVEC 1,1,6,3   ;x,y,i,scale
000E: SVEC -3,2,7,2   ;x,y,i,scale
000F: LABS 512,512,0   ;x,y,scale
0011: HALT
```

Figure 6 – VG Standard Tests

- The “LOOP” button continually runs the program, which is useful for probing signals manually and maintaining a display on your XY scope (along with increasing the VGGO count - especially on a digital scope).
- “Reset after test” actually resets both before and after the test. This way the test always runs and shows Halt afterwards (to avoid getting stuck). It’s generally a good practice to use this option.
- “Verify” confirms the program was written to vector RAM correctly. This shouldn't really be needed if the RAM tests pass, but it provides an extra check.
- “Test Notes” describes the chosen test along with warnings, options and expected results
- “Vector Disassembly” shows the code that was run.

Best practice is to run the “VG Halt” test first, since if Halt fails any other test results are going to be irrelevant. Should the board fail one or more tests you’ll likely want to proceed to Trace testing, in the next sections.

DVG Standard Tests Results

Vector Generator Test	Time (usec)	Scope Display
VG Reset	0.05	no display
VG Halt	3.55	no display
VG Center	6.85	moves to center
VG Draw Short from Ram	637.5	Asteroid
VG Draw from Ram	9838	Test Grid and Intensity Bars
VG Draw Single Short Vec from Ram (global scale 0)	-----	-----
Scale=0	11.5	Short Vector per user input
Scale=1	16.8	same
Scale=2	27.4	same
Scale=3	48.6	same
VG Draw Short from Ram Big (time per copy)	637.5	Asteroid
VG Draw Single Long Vec (global scale 0)	-----	-----
Scale=0	8.85	Long Vector per user input
Scale=1	10.2	same
Scale=2	12.8	same
Scale=3	18.1	same
Scale=4	28.7	same
Scale=5	49.9	same
Scale=6	92.2	same
Scale=7	176.9	same
Scale=8	346.4	same
Scale=9	685.2	same
VG JMP Test RAM	1893	Asteroids
VG JSR to address (Asteroids v2: 0x862)	1450	Asteroids copyright notice
VG JMP only no draw (X=number of copies)	3.55 + 2.0X	no display
Set Scale	6.85	no display
JSR Test	----	----
Nest Level 1	575.1	0 1 0
Nest Level 2	944.8	0 1 2 1 0
Nest Level 3	1428.4	0 1 2 3 2 1 0
Nest Level 4	1866.9	0 1 2 3 4 3 2 1 0

Note: times are approximate – expect values within 1-2%

DVG Halt Troubleshooting

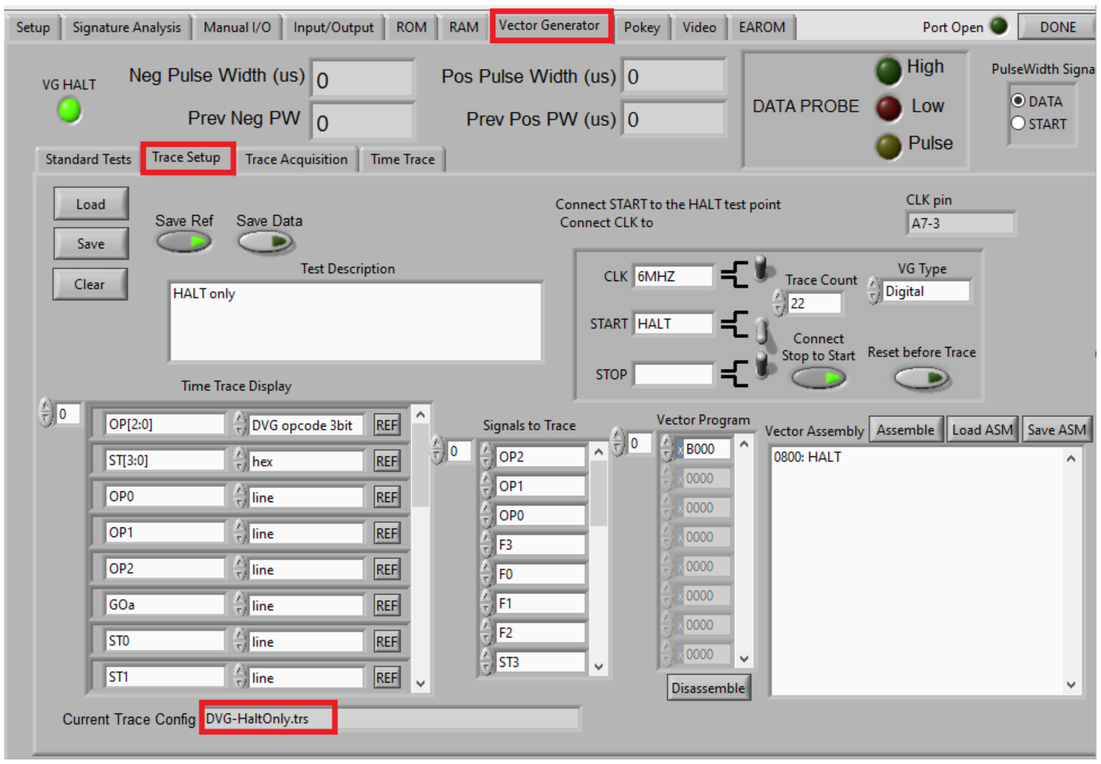
If your vector generator is messed up to the point it is not halting
Loop the halt test and check that the "ADMA" lines are either low or pulsing. If any are stuck high then the program counter is not being reset properly (it should start at 0) and the vector program will not see the "halt" instruction.

When looping the test (with reset) you should also see latch 0 and latch 1 pulsing as it latches in the 2 bytes of the "halt" instruction. You should also see DMALD pulse low as it loads the program counter when the state machine starts executing (the clear of the DVY latches and this DMALD pulse is what sets the program counter to 0). You should see the HALTSTROBE signal from the LS42 decoder in the state machine section, but you probably won't if it is not halting.

Vector Generator Trace Setup

https://youtube.com/watch/BcpguZCA9B0?start=32m53s

Moving to the “Trace Setup” tab and loading an existing Trace file (in this example: DVG-HaltOnly.trs) will produce the display shown in Figure 7.



Neg Pulse Width (us) 0
Prev Neg PW 0
Pos Pulse Width (us) 0
Prev Pos PW (us) 0
PulseWidth Signa: DATA [✓] START

Connect START to the HALT test point
Connect CLK to
CLK pin A7-3

Test Description: HALT only

CLK 6MHZ
START HALT
STOP
Trace Count 22
VG Type Digital

Time Trace Display:

OP[2:0]	DVG opcode 3bit	REF
ST[3:0]	hex	REF
OP0	line	REF
OP1	line	REF
OP2	line	REF
GOa	line	REF
ST0	line	REF
ST1	line	REF

Signals to Trace: OP2, OP1, OP0, F3, F0, F1, F2, ST3

Vector Program: B000, 0000, 0000, 0000, 0000, 0000, 0000, 0000

Vector Assembly: 0800: HALT

Current Trace Config DVG-HaltOnly.trs

Figure 7 – VG Trace Setup

- “Time Trace Display” shows the signals that will be plotted in “Time Trace”
- “Signals to Trace” is the data you will collect from the board under test in “Trace Acquisition”.
- The code to execute is displayed in the “Vector Assembly” and “Vector Program” windows.
- “Reset before Trace” actually resets both before and after the test, to avoid the getting stuck. It’s generally a good practice to use this option.

You can “Load” or manually enter your own program in the “Vector Assembly” window then hit “Assemble” to generate the vector program. You can even copy and paste a test of interest from the “Standard Tests” disassembly window for troubleshooting a bad test. https://youtube.com/watch/BcpguZCA9B0?start=32m53s

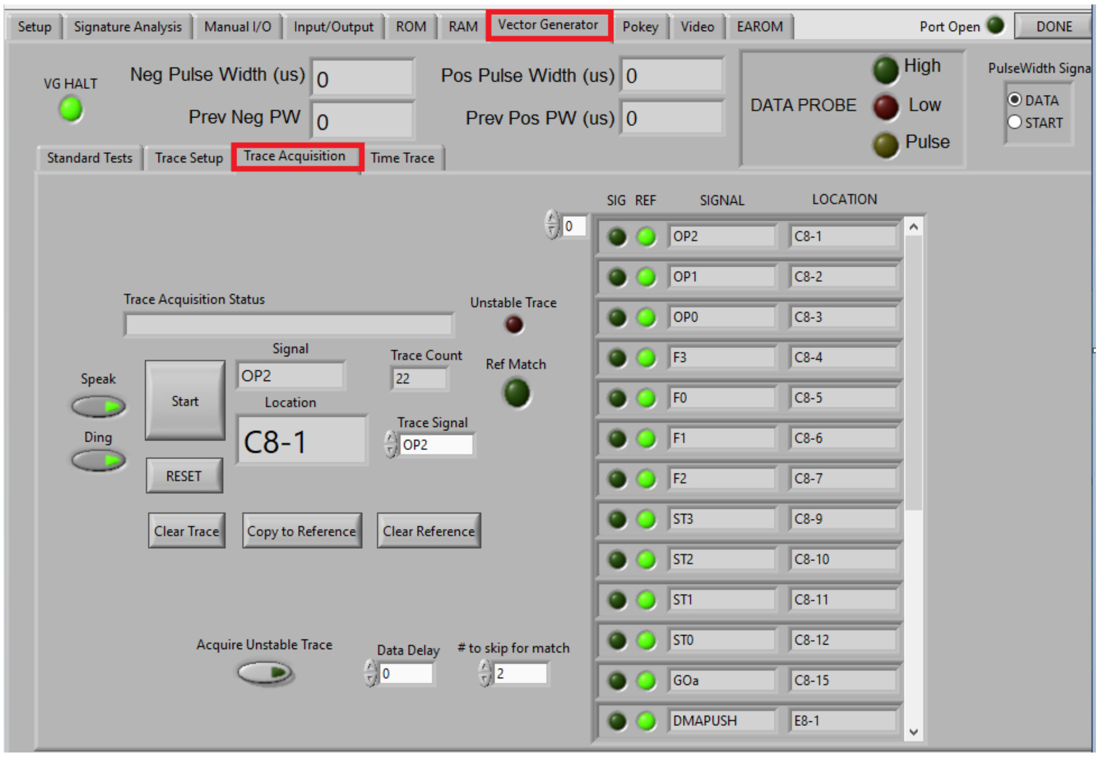
The “Save Data” and “Save Ref” buttons allow you to save the data you’ve collected as a new Trace file. This is useful for sharing the trace results from a non-working board or generating a new Trace test.

An existing Trace file (.trs) will often contain reference data from a known good board, which can then be used for Trace comparison.

Vector Generator Trace Acquisition

https://youtube.com/watch/BcpguZCA9B0?start=38m15s

Now it’s time to collect data from the board under test. Switch to the “Trace Acquisition” tab (Figure 8), then click “START” and begin collecting the specified signals. This process is nicely explained in the video link above.



Neg Pulse Width (us) 0
Prev Neg PW 0
Pos Pulse Width (us) 0
Prev Pos PW (us) 0
PulseWidth Signa [✓ DATA] START

Signal: OP2
Location: C8-1
Trace Count: 22
Trace Signal: OP2
Data Delay: 0
# to skip for match: 2

SIG	REF	SIGNAL	LOCATION
●	●	OP2	C8-1
●	●	OP1	C8-2
●	●	OP0	C8-3
●	●	F3	C8-4
●	●	F0	C8-5
●	●	F1	C8-6
●	●	F2	C8-7
●	●	ST3	C8-9
●	●	ST2	C8-10
●	●	ST1	C8-11
●	●	ST0	C8-12
●	●	GOa	C8-15
●	●	DMAPUSH	E8-1

Figure 8 – VG Trace Acquisition

- “Clear Trace” deletes any data you’ve collected from the board
- “Clear Reference” deletes the existing reference data
- “Copy to Reference” copies your board’s data to the reference location which is convenient for creating Trace files for a known good board. Use “Save” under “Trace Setup” to generate the Trace file.

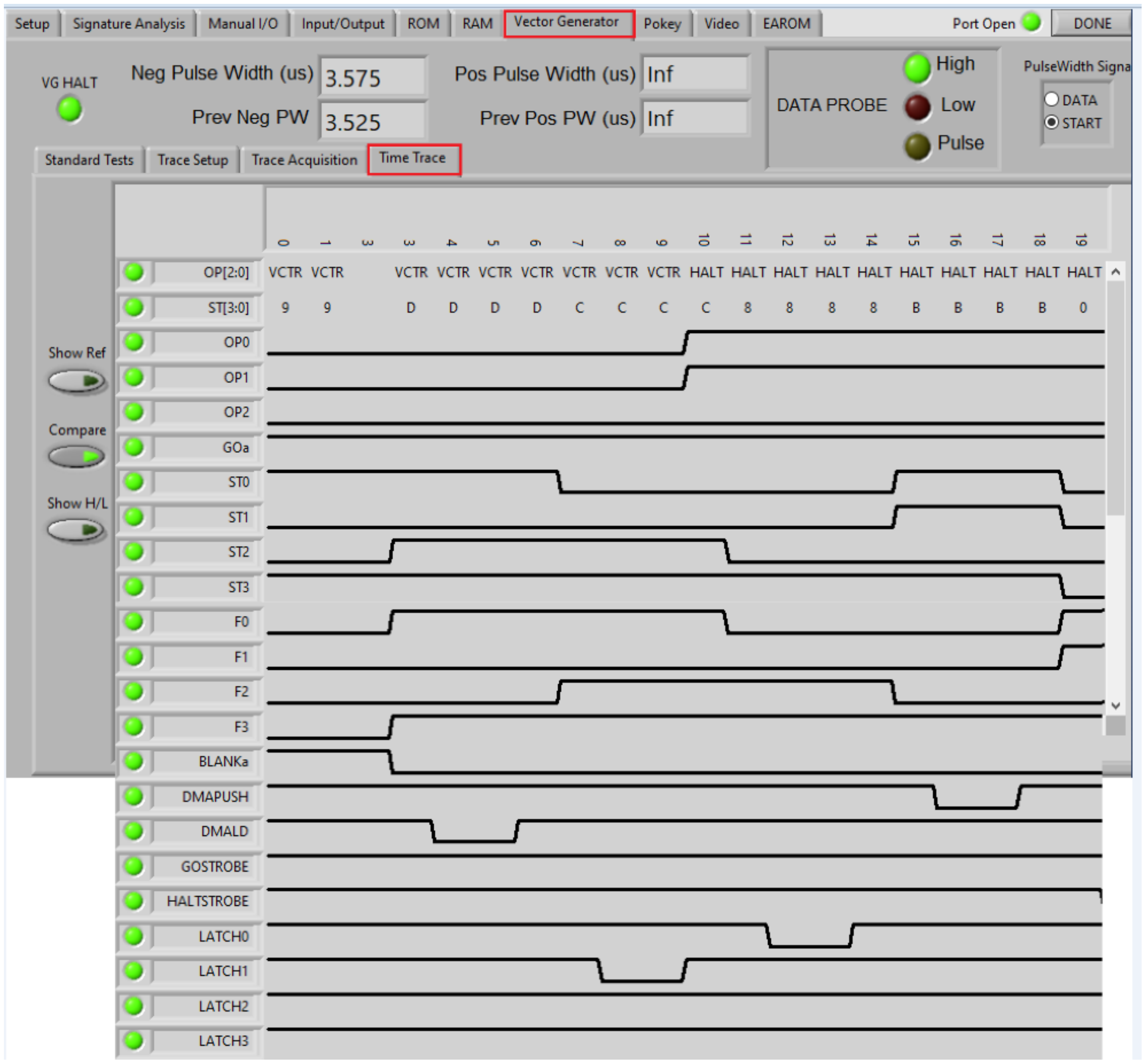
- A “Trace error – check connections” message in the status window means the trace was not ready after the vector generator program runs. This could either be the Clock or the Start/Stop signal since both are required for a trace to be acquired – this is the same logic that runs signature analysis.

- The “Unstable Trace” light indicates the current trace is not stable. The tester requires that two consecutive traces match before recording the data. If an unstable trace is detected you’ll need to advance manually using the “Trace Signal” pull down list (unless “Acquire Unstable Trace” is selected).
- The “Acquire Unstable Trace” option will attempt collect data a couple of times but acquire and move on even if the trace is unstable.

Vector Generator Time Trace

https://youtube.com/watch/BcpguZCA9B0?start=40m20s

Advancing to the “Time Trace” tab (Figure 9), you can now see the program execution in a logic analyzer format.



Neg Pulse Width (us) 3.575
Prev Neg PW 3.525
Pos Pulse Width (us) Inf
Prev Pos PW (us) Inf
PulseWidth Signa
DATA [✓ START]

	0	1	3	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
OP[2:0]	VCTR	VCTR		VCTR	VCTR	VCTR	VCTR	VCTR	VCTR	VCTR	HALT	HALT	HALT	HALT	HALT	HALT	HALT	HALT	HALT	HALT
ST[3:0]	9	9		D	D	D	D	C	C	C	C	8	8	8	8	B	B	B	B	0

Figure 9 – VG Time Trace

- “Show Ref” displays the reference program data if available.
- “Compare” then highlights any differences relative to the reference data.
- “Show H/L” simply annotates the signals for improved readability.

On the trace plot, Trace OP[2:0] is the 3 bit DVG OpCode and ST[3:0] is the current State.

For this example (running HALT), you can see the progression of signal traces and also follow on the State Diagram as highlighted in Figure 10.

State	**Signal**
0	(halted)
9	DMALD

D	LATCH1
C	LATCH0
8	DMAPUSH
B	HALTSTROBE
0	(halted)

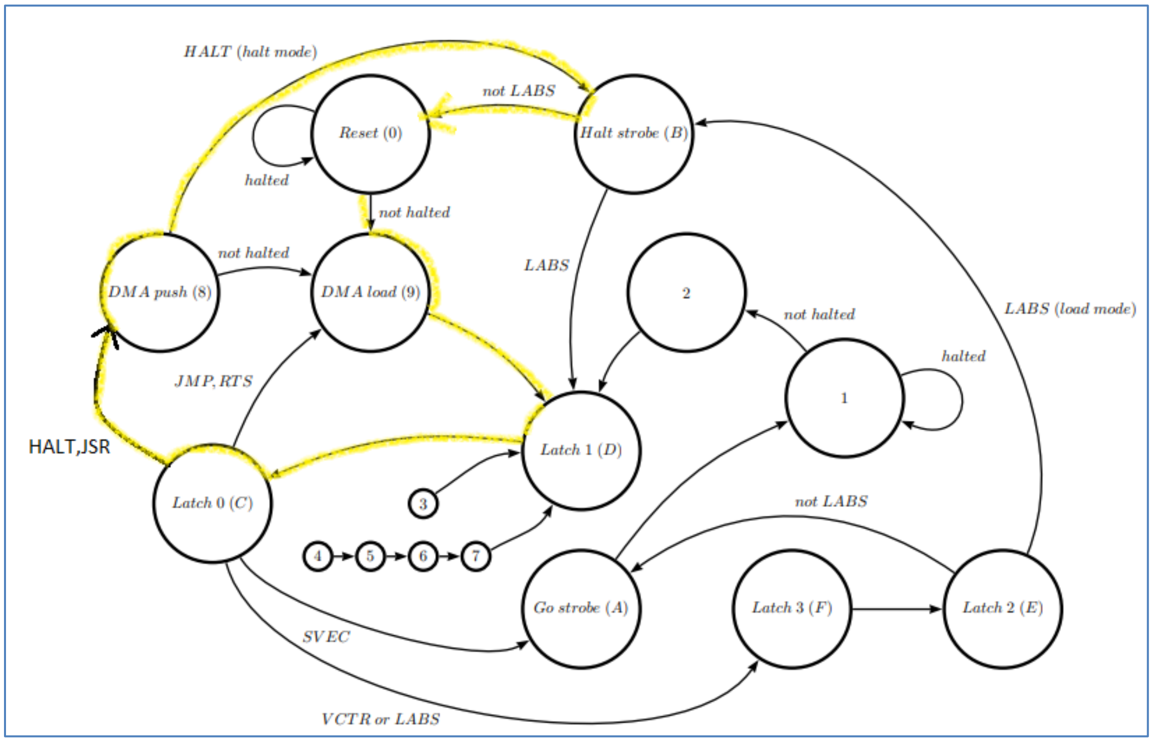


Figure 10 – HALT Path on State Diagram