AWG Instrument Development

Sunday 12th April, 2015

Chapter 1

AWG Upgrades: Notes, Useful Ideas, and Implementation

1.1 AWG Detection Train

1.1.1 List of Necessary Improvements

1. ☐ Order DC - 1 GHz video amplifiers for use. Follow up with Ilia where these are actually from.
2. ☐ Measure the recovery time of the limiter.
3. ☐ Get a quote and order the limiter the Kurt Warnake uses in his spectrometer. Found that the site claims 10 - 20 ns recovery of the limiters this is good however they're only rated to 100 W pulse.
4. ☐ Order 2 gigatronics power supplies model number GPD 33032.
5. ✓ Buy the F914A switch, 10 ns rise and fall time, 80 dB isolation, 1.6 dB insertion loss. However with this switch we'll have to use a diode limiter to protect the protection switch.
6. ✓ Need to determine the tolerance for the switch that you actually need it might not be nearly as high as you think because much of the pulse power may just be reflected back to the pulse source due to the over coupled cavity. *14/07/23* You need at least 400 W tolerance in the switch.
7. ✓ Follow through with general microwave about the high power tolerance switch. *14/07/26* They don't have one that will suffice.
8. ✓ Measure insertion loss from passive components e.g. DC block, high pass filter etc. *14/07/26* This is just really not necessary.
9. ✓ Deal with DC offset for the on carrier signals. This is possibly due to a DC offset due to the IQ mixer, here just placing a DC block after the mixer or video amplifiers may deal with this. If this does not fix the issue moving a DC block around the detection train may fix this issue. Here you want to be on carrier but off EPR resonance condition this will show oscillations in signal but also the DC offset that's seen. *14/07/25* Actually it's the DC block that is causing the baseline issue, If we want to drop the base line we'd be best to modulate the LO input to the detection mixer by something like 200 MHz then use a RF high pass filter after the mixer to drop all of the DC components from the scope voltage.
10. ✓ Figure out what's up the weird base line after a given pulse. *07/24/14* This seems to be a problem with the video amplifier bandwidth causing these distortions. You still need to go forward and measure what this is like at high power 1 W and what the base line does with changing the pulse phase and AWG frequency relative to the carrier.
11. ☐ Talk to Tim about the sensitivity measurement that Thorsten was talking about, this would be a nice experiment to do to compare our detection train to what Bruker gets.

12. ☐ The phase roll and I Q imbalance of the detector is still an issue you can see it in your background traces of the ELDOR type experiment. This needs dealt with. I think if you did the calibration where you find the IQ imbalance as a function of phase and then just normalized the I and Q channels, this should at least give you a nice flat background.

1.1.2 Follow up on baseline issue 07/25/14

I check and see that with the video amplifiers I still see this weird baseline artifact. I think that it may either be due to the band width of the amplifier or possibly due to the fact that the amplifiers reflect anything that they do not amplify. Here verify that you see the same weirdness with the DC block, then insert an isolator between the DC block and the IQ mixer to see if that fixes the issue. *400pm* I do see very similar weirdness with the DC block inserted, now you should place an isolator between the DC blocks and the mixer. *442 pm* I place the isolators in line between the IQ mixer and the DC block and see that the baseline artifact remains. *450pm* I turn the DC block around and see the same issue. I really do think this is a band pass thing. Next take DC blocks out capture nice baseline, ft and drop anything below 50 kHz ift and see if you see a weird baseline. Interesting to note, the DC blocks do indeed drop the DC component of the spectrum. *500 pm* I try using data analysis to show that I'm running into a bandwidth thing but find that if I drop 100 kHz I don't see any change in the spectrum. This would suggest that what I'm seeing is not a bandwidth issue but something else.

Impedance matching? *520 pm* I take a 150 MHz bandpass filter and see that it removes the baseline artifact where as the DC block is actually the cause of the baseline artifact. Is it possible there is some weird impedance mismatch? I also try the 500 MHz bandpass and see that it completely removes any signal as well as the baseline artifact. This cannot be a bandpass thing.

540 pm I try the nicer video amplifiers and I get the same baseline artifact. What is this? I put a 150 MHz bandpass filter after the video amplifier and I see all of the baseline artifact go away. I think what you need is a highly flat amplification across the frequency range of interest. I also notice that attenuating before the video amplifier does nothing.

I really really think that this is a problem with the video amplifier flatness. What I mean is the flatness across all frequencies being amplified.

Explanation: The DC offset comes from the IQ mixer, this is why a DC block before the mixer at XBand does nothing. The baseline drift is because we're not attenuating enough of a bandwidth, this is why with the DC block you see funky baseline but with the 150 MHz HP filter you see absolutely flat baseline. With the video amplifier you are 0.05 - 500 MHz this is cause the baseline weirdness because you aren't amplifying all frequencies equally, here you could get a really nice flat amplifier or just use a high pass filter that bocks sufficient frequencies and just make sure your signal never is in that frequency range. Here you should buy different high pass filters and test them on the system from a look at minicircuits it looks like the lowest possible is 20 MHz which is ok but with this solution you should go with higher bandwidth video amplifiers.

1.1.3 Determine the necessary tolerance for the switch 07/23/14

Here I want to use the solid state amplifier and measure the power reflected from the resonator to the detector on the sampling scope. I should do this for various conditions of coupling and of resonance. This will tell me the relative amount of power that is reflected at the detector. I notice that the reflection is almost half of the voltage as the input pulse. You should measure this to get a verifiable number and do for multiple resonator resonance offsets.

1.1.4 Determine what component of the detection Train is causing baseline issue 07/23/14

The base line after a pulse of any power, even low is not! flat! it should be. Here go through component by component and check what is causing this.

I see that when I'm off resonance by 100 MHz I see that the baseline after the pulse has a slow oscillation to it.

First just figure this out by pulling out the elements that may be causing the problem.

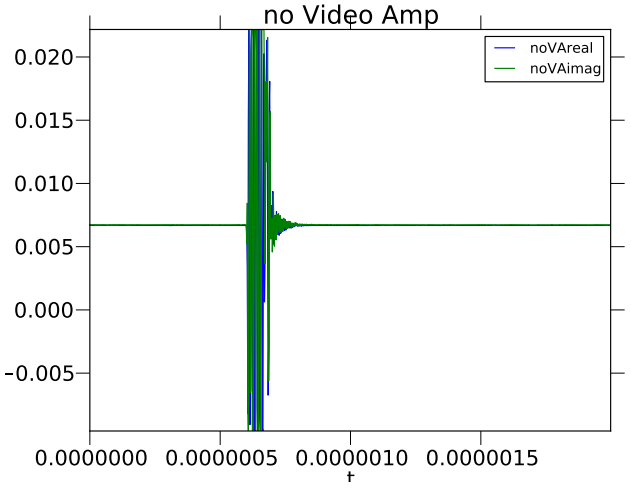
1. ✓ LNA *5pm* I pull this out and see that the baseline issue is still present only to a lesser extent, likely this is just amplifying the baseline problem.

2. ✓ limiter *540 pm* I plug directly into the sampling scope and see that there is microwave leakage, you need to be using the switch so you know what is exactly the problem. It is interesting to note that I see a sort of

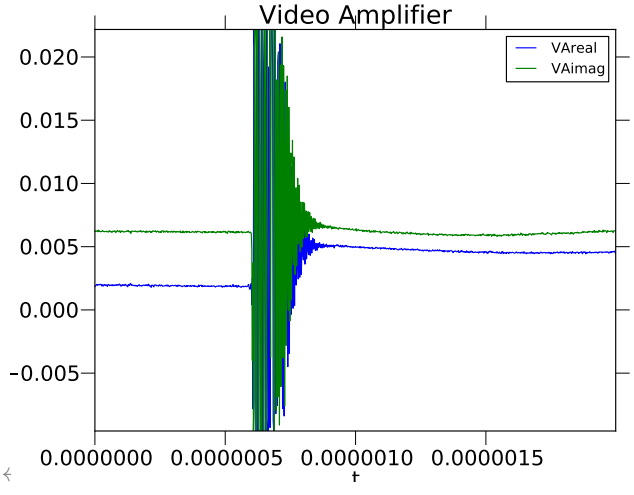
exponential decay on the agilent where as I see an initial ripple then a DC offset on the sampling scope. I may not see the DC offset issue on the agilent because of the video amplifiers. *550pm* With the switch I effectively kill any DC offset and on the sampling scope I really just see what looks like ring down. Plug back into the agilent and see if the baseline is still there. *600 pm* I still see weirdness with the baseline. Note that when there is no microwave power in the system there is a slight disturbance with the baseline at 2 mV. This could still be the issue.

3. ✓ High pass filter *520 pm* This does not cause the baseline issue.
4. ✓ Isolator *530 pm* I unplug everything and go cavity -¿ limiter -¿ mixer -¿ VA -¿ scope, I still see this base line weirdness here. It's possible that this is actually due to the limiter itself, run from the limiter to the sampling scope itself.
5. ☐ IQ mixer. Here just plug the end into the sampling scope
6. ✓ Video Amplifiers *610 pm* I go cavity -¿ limiter -¿ mixer -¿ scope and leave out the video amplifiers, the baseline issue crap is gone or seemingly at least. Put the LNA back in place to see if that gets more power to the scope. *620 pm* I put all of the detection stuff back in and see that all of the weird baseline stuff has gone away. I also see that the DC block we have in place does nothing to X-Band which makes sense. When I put the DC blocks in I see a weird Baseline…is it possible that I'm saturating the block?
7. ☐ Video Amplifiers with high power:
8. ☐ Without the video amplifiers in the detection train, change the pulse phase and also change the AWG frequency and see what this corrected baseline does with changing these parameters. Actually make a plot of the baseline DC against the AWG frequency.
9. ☐ Make a plot of the 50 ns pulse with the DC block in to show that this baseline issue looks very much like a poor bandpass filtering issue.

Video Amplifiers: I find that the video amps are causing some sort of baseline issue I see that with out the amplifiers I have a baseline that looks very flat as shown in 1.2.1 where as when I put the amps in place I see that the base line does some weird shit as shown in 1.1.4. You really need to verify that the issue is no longer present when using higher power at least 1 - 5 Watts.



file:140723noVA.pdf

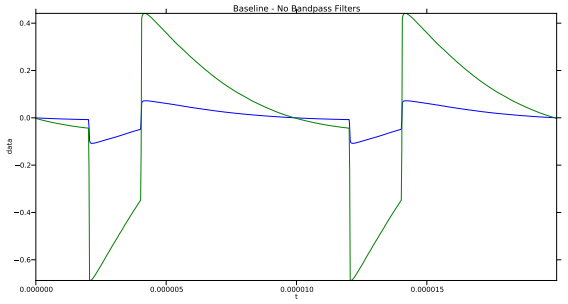


file:140723VA.pdf

1.1.5 Implementation of 1 GHz bandwidth video amplifiers and bandpass filters

Like the title says I implement the high bandwidth video amplifiers and the bandpass filters to deal with the baseline issue we're currently facing.

The figure show what we're currently dealing with. I put out a 2000 ns long pulse at low power approximately 1 mW and I see this issue.



{fig:nova}

file:150407BaselineNoBandpass.pdf

I note that the high pass filters that I ordered from minicircuits never came in...

1.1.6 Testing the reciever 15/04/09

I setup to test the reciever with a BDPA PS sample. I tune the cavity and the

We setup with the synthesizer switch and the new reciever configuration and take some signal meausrements.

I save a representative data set as '150410Experiments.h5/phaseCycleWithBandPassWithSynthSwitch'. *Here we have the synthesizer switch in place and we're using the TWT. Currently the reciever switch is sitting on the shelf and not being used. I'm also using the 200 MHz bandpass filters. I set the YIG 300 MHz off resonance and use the DAC board to modulate the frequency by 300 MHz. I see that this very much cleans up the base line to my signal as I take this signal with a 4 step phase cycle. The sample is solid BPDA and I'm lookin at an echo.*

In preparation, I pull the synthesizer switch so the waveforms have a very high offset and I see that I still record fine signal. I repeat the same measurement and save the signal as '150410Experiments.h5/phaseCycleWithBandPassWithOutSynthSw

Also note I have a script written up to do a quick phase cycle with the modulation of the carrier by the DAC board. The script name is *'bandpassSignalTest.py'.*

1.2 AWG Synthesizer

1. ☐ Place low power (roughly 10 - 100 mW) switch and isolator after the IQ mixer to kill any leakage.
2. ☐ Possibly use a phase shifter and variable attenuator

1.2.1 Testing the Synthesizer Switch 15/04/09

I place the switch from miteq in the synthesizer train and test to find the best configuration possible.

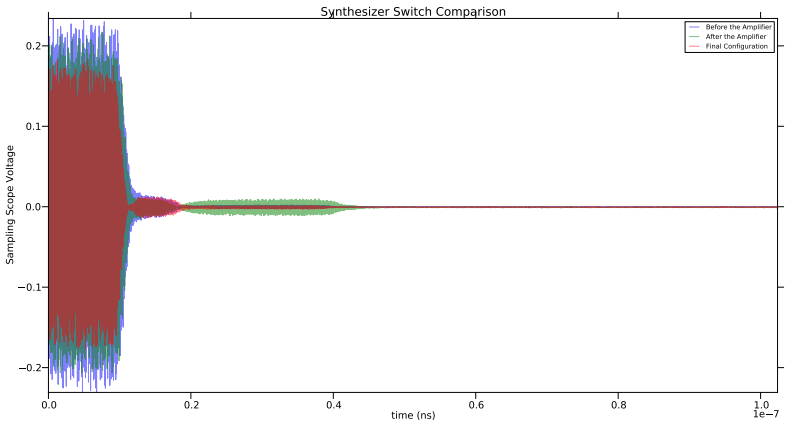
I find the following

1. The switch puts out a video transient at MHz frequencies when the switch turns both on and off. *This is why the high pass filter is in place and is bloody necessary!* USE IT
2. The performance of the switch is better before the low noise amplifier as evidenced by the graph below.
3. The input line to the switch is not impedence matched to 50 ohms. I placed a 50 ohm load to ground on the sma input and this cleans up the switch transient that is imprinted on the microwave waveform.

I think this issue is wraped up for the moment as we have very high isolation after the pulse.

Below shows the comparison of the synthesizer switch placed before and after the input amplifier. What I find is that I get better isolation if I place the switch before the amplifier, plot shown below. *The Data for the plot is saved in '150409Experiments.h5'*

Also note that this is taken without doing any zero calibration on the DAC board.



{fig:novA}

file:150409SynthesizerSwitchComparison.pdf

1.3 Bruker Bridge

Chapter 2

DAC Board

2.1 Implementation of Daniel's new FPGA code and server software

Daniel Sank and the Martinis group have developed some new FPGA firmware and server software to run longer pulse programs and also possibly run on board phase cycling.

2.2 Time Line of Jump Tables Implementation

I honestly think the best, most efficient, way to do this is in context of CPMG as opposed to writing nice software to handle general instances. Here I write code pragmatically to do my CPMG experiments and learn the in's and out's of using the jumptables firm ware. Once I have a handle on the firmware, I will then sit down to write code (pulsegen Class) to act as a more general implementation. At this point it would be good to get in contact with Bryan from Jayich's lab and share our code.

Monday (15/03/09):

1. ✓ Work with Daniel, get DAC and server program to work together. *Here ask Daniel the best way to do looping, imagine a CPMG sequence where you want to increment the interpulse delay, how would the jumptables vector look?* The DAC and the server talk now. The fix was ultimately in the registry dictionary, to do with the board's fifo buffer size... Also we've updated labrab and the direct ethernet program. The DAC still works with the current digitize class.

2. ☐ Demonstrate *jumptables.py* working appropriately. *Need to install the current version of firmwave. V.11 not V.8*

3. ☐ Save software example code with sufficient comments and notes to pick up project at a later date.

4. ☐ Nuke py26.

Here I've come across, in a bad way, how poorly the files are organized on this box. Every module that we use is brought into the current working directory, this shouldn't be. We should store the necessary modules pulsegen, matlablike, etc. in a directory that is on the PYTHONPATH environment variable. This directory should stay clean and script that's nicely wrapped should go here and that's it!! PERIOD!!

Also you need to go through and nuke py26 and anycalls that your win scripts make to py26!!

3 days: Organize or Nuke Current Version of Windows... Definitely NUKE IT!! I list thoughts below on how I think best to do this safely but this just needs done. The file organization is a mess and the computer runs like a sloth!

1. ☐ Safely back up the HD. Detailed in **This Fucking Windows Box**.

2. ☐ Reformat the harddrive and re-install windows 7.
3. ☐ Install all the necessary crap to make windows behave like a real development platform mingw, git, putty, python27, gcc etc.
4. ☐ Copy registry keys.
5. ☐ Clone pylabrad, twisted, and servers from Martinis git repo.
6. ☐ Setup appropriate software directory on PYTHONPATH.
7. ☐ Detail a file organization procedure so we don't end up with this mess again.
8. ☐ Also set up administrator privileges to install programs, I don't really know what else to do but there is so much crap on this box.

1-3 days: Code up and test device for CPMG experiment

1. ☐ Use jumptabels format to produce a CPMG sequence verify all delays on scope.
2. ☐ Code up a CPMG experiment where I loop through both phase of inversion pulses and the intepulse delays. *Do this by hand, don't attempt some fancy code structure as this will inevitably lead to a mess at this point.*

2-5 days: Run a CPMG experiment

1. ☐ Use 50% Glycerol 4-OHT sample. cool to 85 K
2. ☐ Optimize Hahn echo pulse sequence for a 200 ns interpulse spacing. Nutation
3. ☐ Quickly measure a Hahn echo decay, just for the note book.
4. ☐ Run a CPMG experiment with a 200 ns interpulse spacing. *Most likely you're going to have to do this the dumb way and use the scope to capture each echo individually, which kinda blows but it's what will be easiest. - I'm call this 'dumb-CPMG'*

Now, as proof of concept I should show the CPMG spectral filter concept. I do CPMG experiment varying the interpulse spacing and plot the decay rate $(1/T_2)$ as a function of pulse frequency. **1 week: Spectral Filter**

1. ☐ Measure CPMG signal as a function of pulse spacing for three different temperatures. Say 85 - 110 K, just so that I can see if what I calculate for the rotational correlation time makes sense.
2. ☐ Use the $(1/T_2(\nu_\pi))$ data and follow Suter's method of inverting the series of equations to get my spectral density.
3. ☐ Fit the spectral density to a simple lorentzian and get correlation time and plot as a function of temperature.
4. ☐ Fit the spectral density to the FFHS model

Here, I argue, is the right time to do instrument development (e.g. write generally implementable jump table code and configure detector to handle the CPMG experiment appropriately.) Now this is also the time to evaluate whether this is worth doing. If the Bruker setup is still orders of magnitude better in signal to noise and can also do the CPMG experiment, I argue this development is not worth doing.

If we decide it's worth doing. **2 weeks: Code up Agilent ADC board**

1. ☐ Demonstrate that you can capture long data packets, should be able to do $20\mu s$ at a time with on board averaging.
2. ☐ Wrap c functions to do this long data on board averaging in swig to call from python.

1 week: Wrap up jump tables into pulsegen

1. ☐ Write general class instance to handle parsing a given waveform into memory blocks and a list of calls for the jump tables firm ware. *Right now I think the easiest way is to add into the waveform class a parsing function that the user defines repeat sequences, then the class calls a function which determines from the waveform specefied and the repeat sequences the necessary list to hand jumptables so that the given waveform is reproduced.*

2.3 Jump Tables implementation

You need to pull the data from the git hub. Pull from branch jump_table_bench_test/GHzDACs

There is a readme in HanLabNewFPGA folder on desktop, it outlines what needs done.

I have the fpgaServer.py running. *Note that I had to comment out some code. Also on another note you might want to talk to daniel about how they organize their code.*

I import fpgaTest successfully and run fpgaTest.get_build_number(fpga,board) and cannot connect to the board because their are no fpgas present.

So far I've:

1. Installed the new firmware onto the FPGA. *firmware located in the C:*
   Users
   hanlab
   Desktop
   HanLabNewFPGAStuffs

2. Created fpgaServer from ghz_fpga_server.py, I had to comment out several lines that imported stuff to do with the ADC as well as imported stuff the my version of labrad does not have.

3. Commented out lines of the fpgaTest.py for the same reasons as above. I can now import fpgaTest.py.

4. Tried running fpgaTest.test_jump_table_idle(fpga,de) *Here I don't know what de is I handed it cxn but I'm not sure that's what was needed.* I also note that the mac address for the board is hard coded - I need to input the correct mac address for board.

5. Tried running fpga.select_device() as well as fpga.select_device('Han Lab DAC 1') which is the board name and I get an error telling me that no devices are present.

6. I note that our previous method of using the startup dac board script works to bringup the DAC with the new FPGA firmware. *Maybe I can use this to figure out how to configure the new FPGA server program.* I test and find the problem is isolated to the new FPGA server program. I test and use the old server program and see that I can run bringup, I close the old server and open the new one and cannot run bringup because it cannot find the DAC, I close new server open old server and run bringup successfully.

7. I test the dependent modules, dac.py and util.py and get the same error that bringup cannot find any board. I try moving the working server program such that it uses the new modules and see that it does not work, suggesting the new modules dac.py or util.py are to blame.

Things I need to do:

1. ☐ Find the correct mac address for the board and insert it, either hardcoded or as a global variable in the fpgaTest.py

2. ☐ Possibly install the new labrad server.

3. ☐ Talk to Daniel about file organization. There seems to be some structure you're missing that might make life easier in the long run.

2.4 15/02/18 - Get the board fucking working

For some reason I can no longer connect to the DAC board. I keep getting an error when I try to connect to the DAC board *FAILURE: The board was off when you started the GHz DAC server. Turn on (and) connect the DAC board, close and restart the DAC server and press any key...*

Also, eventhough this is running on the local host loop back address this DAC server throws the same error with or without the putty ssh tunnels up and running.

I check to see that I can connect to other instruments. *I can connect the agilent scope, which is also setup on the loopback address. However I cannot connect the sampling scope, or more formally the prologix gpib-eth controller (setup on loopback) which is a problem in and of itself, fuck.*

I find that the 10 MHz oscillator is unplugged, thus the DAC is not recieving a clock signal (most likely not able to communicate because of the lack of clock signal). I plug in the clock, let it warm up, and retry connecting. What do you know, I can connect to the DAC board!!

I now try to use the DAC board, and I can synthesize waveforms good news.

The ecl port is j24 you idiot!

2.5 We're in a rush to do things as always... Flash the old firmware and switch back to using the old server program.

I flash the old firmware back onto the board and setup with the old server program and things work.

I make sure to flash the .pof file because we need to be able to restart the board.

I flash the old firm ware and switch to what I think is an old version of labrad. The problem is the server program does not like when I do a long delay...

Specifically I cannot run functions '_sendDAC()' and 'fpga.run_sequence()' with the current build, server program, and labrad implemnetation.

I need to fix this as we need to be able to run sequences longer than $15\mu s$. The things I try are below.

1. ✓ Try updating to the new version of the labrad. *This didn't work for allowing the use of sending and playing memory* Balls
2. ✓ Try loading the old labrad in site˙packages. *Again this didn't work*
3. ✓ Try the old server progam and make sure it's using the old labrad version. *Yes, using old version of labrad and the old server. Still Didn't work*
4. ✓ Flash what you think to be the old build onto the board, I found *V5˙b8* on the Martinis website and tried that but I still get the 'Paging off: SRAM too long.' error. *I see the registry has definitions for the sram length that look right but I now check what length of sram I'm sending to the DAC.* This isn't a build problem...This is something to do with the server or the registry.
5. ✓ Check sram length, maybe this error is real and not something to do with the old server / build. *No it's the proper length.* FUCK!
6. ✓ Do build 14. The new server. The new labrad. Daniel got back to me about the issue being resolved...Maybe this will work now? *Make sure to test with the 'testBuild14.py', this code runs through a jump_table example that Daniel gave me and also plays an sram sequence. Also make sure you try all build numbers.* I got through and try all build numbers and cannot get the board to function when sending jumptable commands.

2.5.1 15/04/12

Conclusion to previous day of work. Jump tables will not work at this time due to continued development at the Martinis group...The problem is I need to loop over memory to do long delays which I currently cannot do.

Make a test script for Daniel that shows the sram and the memory I send to the DAC board. Do this running the new server and the newest version of pylabrad.

2.5. WE'RE IN A RUSH TO DO THINGS AS ALWAYS…FLASH THE OLD FIRMWARE AND SWITCH BACK TO USING THE OLD SERVER PROGRAM.

1. I'm now running the newest version of pylabrad on master branch and the newest version of servers on jump_table_test_bench branch.

Chapter 3

The Prologix Controller

This does deserve it's own chapter as you spent two days figuring this out... The power supply is truly important. You found that it is very important that it supply the right power, if it's not right the prologix controller will still light up and 'look' normal however it will not function properly as an tcp / ip socket.

I can now access the field controller and I setup the gpibtunnels in putty to forward or tunnel or whatever it's called through this new address.

I can now access the gpib controller via 'conn = g.gpib()' and the field controller through 'fc = g.field_controller()'. The new computer is still a problem but atleast this is fixed.

Chapter 4

This Fucking Windows Computer - Rebooting and Getting the DAC to work with New Comp and Firmware

Seriously, it needs reformatted. . . BADLY!

I am hesitant to do this bing bang boom as it will be a pain in the ass to get resetup and working nicely again. I list what I think would be a good plan of action for re-formating this piece of shit.

Didn't go this route.

1. ☐ Mirror the current hard drive on an external drive.
2. ☐ Verify that you can boot into this drive off of the windows box.
3. ☐ Make sure you have access to pylabrad and servers repositories from the Martinis group on the github.
4. ☐ back up the registry files so that you can reconnect to the board.
5. ☐ Make sure you can access quartus on the new computer, or get an access key.
6. ☐ Now reformat that shit box and install a fresh version of win7 free of any crapware.
7. ☐ Go through the installation of python and get setup.
8. ☐ Install the scripts from Martinis group and go.

Went this route instead I decided to use Anna's old computer, reformatted it and am now setting it up to run the AWG from. I do this and the new windows box is alive and up to date.

1. I install mingw. I install all of the developer tools along the way.
2. I install git and clone the repositories *servers* , *pyspecdata* , and *pylabrad* to c: /extraPythonLibs
3. I install pythonxy.
4. I install tex Live
5. I install twisted
6. I move necessary contents from old computer *dac_board* to new computer.
7. I run labrad and directEthernet. I find that when I start direct ethernet I do not see the typical 'Han Direct Ethernet' connection popup in the labrad instance.

8. I find this is because labrad is not listening on the correct port which is defined in *env_vars.reg* . I set the labrad port to 7682 and restart the labrad manager and the direct ethernet and see my 'Han Direct Ethernet' connection in labrad!
9. I notice that if I power cycle the DAC I revert back to build 8. This is because what I'm loading to the board is volatile memory.
10. I send the pof via active serial programming and this keeps the memory in a non volatile state.
11. I keep getting an timeout error on fpga.pll_init()
12. I try running other fpga firmware deals to see

Things left to do

1. ☐ Get DAC to work with new firmware. This might involve bugging Daniel more.
2. ☐ Setup freesshd to route ssh traffic.
3. ☐ Get putty setup to host the ssh tunnels.

4.0.2 FreeSSHD Putty and tunneling

I'm setting up the system to do tunneling. This, to me, means essentially ssh on windows. What this also allows is for an outside computer, say a linux box, to ssh to the host windows PC and operate the spectrometer (essentially the associated instruments with the spectrometer).

So what happens is the host windows computer 'brokenBandit' communicates with all of the instruments through the loop back port 127.0.0.1.

For this to work the winows computer needs to host an ssh server, this is provided with freeSSHd.

Chapter 5

Ralph and the Elexsys

Ralph, from Bruker, is coming to visit sometime in the future. . . Definitly, hopefully, before I graduate. However he knows lots of things EPR wise and even more so how-to-hack-the-Elexsys wise. I have some things that I need to do on the Elexsys that I cannot currently and I need to make sure I wring this knowledge out of Ralph when he is here.

Things to Get Working While Ralph is Here

1. ☐ Make xepr talk to python or at least make system calls.
2. ☐ Temperature control from xepr. Minimally, I need to set the temperature however I'd like to also be able to log the temperature. I can live without temperature logging as long as I can communicate between xepr and python.
3. ☐ Three dimension experiment (dim1=Temperautre dim2=RunNumber dim3=Time) for the T_2 experiment.
4. ☐ CPMG experiment.