ARAFE Build and Test Manual

Brian Clark and Patrick Allison

April 5, 2017

Abstract

In this document we will detail the procuedure for testing and debugging the ARA Advanced Front End (ARAFE). This includes the Power and Control (PC) boards and their associated DC-to-DC converters, as well as the Radio Frequency (RF) boards. We will describe how to test them as they are built to ensure they are functioning properly, give example oscilloscope and network analyzer traces of boards which are working correctly, as well as offer practical debugging advice as the boards are built.

1 General Comments

1.1 Introduction

A completed ARAFE module, or ARAFE quad, consists of one power and control (PC) board and one radio frequency (RF) board. Each RF board contains four channels of signal conditioning, with tunable attenuation for signal and trigger paths. As such, there are four quads per station for all sixteen antennas. The four quads are controlled by one ARAFE master board. Therefore, a complete ARAFE system consists of four PC boards, four RF boards, and 1 master board.

1.2 Powering and Communicating with the Tester Boards

Testing of the PC and RF board should be done with the ARAFE slave tester, which is the custom red "MSP430 Comms Test" board in figure 1. The communications tester provides power to the quad, and also provides communications-over-power, as detailed here and originally here. So, the comms-tester powers and talks to the ARAFE quads. Communication to the comms-tester is provided by a custom built USB-to-UART converter, the blue board in figure 2. The comms-tester and USB-to-UART bords can be plugged in to one another as in figure 3.

Power is delivered to the comms tester by providing a 15V rail and GND to J4. Power is provided from the comms-tester to the quads by the SMA output, or J1. Communication from laptop to the USB-to-UART bridge is provided by the mini-USB port, or J1. All of this can be seen in the functional diagram in the top of figure 3.

1.3 Powering and Communicating with the PC Boards

The PC boards are powered by +15V and GND, delivered to J1. The two left pins are power, and the two right pins are GND. A photo of a correct powered board can be viewed in figure 4. The board can be powered either by the comms-tester described in section 1.2, or can be powered directly by a DC power source.

Since the PC board contains a microcontroller (more on that later), it is equipped with two powering modes, controlled by the jumper J2. The orientation of the jumper J2 controls whether board is in "program" mode, for programming the microcontroller via TI Code-Composer studio/ MSP430 FET Pro LITE, or in "external power" mode, for testing with external power. When the left and middle pins of J2 are connected, the board is configured for external power. When the right and middle pins of J2 are connected, the board is configured for programming. The general procedure for powering the PC boards externally is the following:

1. Jumper together the leftmost pins of J2 on the PC board (the jumper setting for "powered externally").

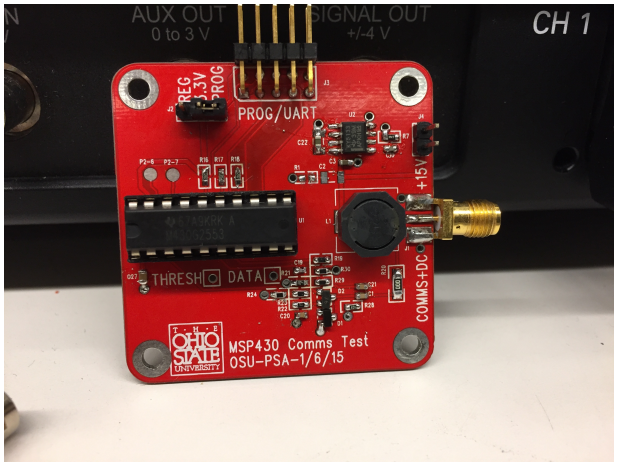

Figure 1: A photo of the MSP 430 Comms Tester. This board serves as a master to the PC slaves.

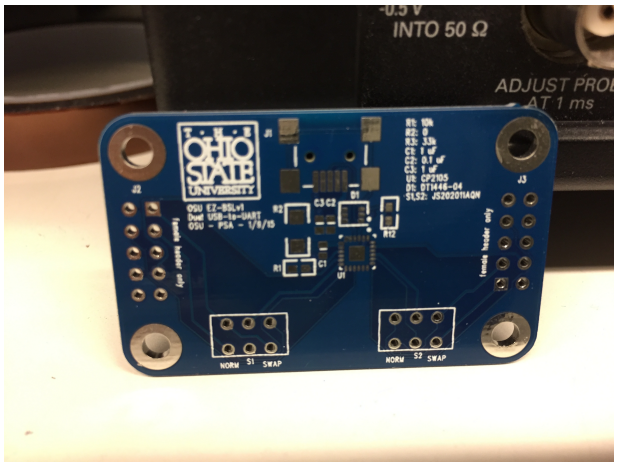

Figure 2: A photo of the USB-to-UART converter. This board serves as a USB interface between a computer (like a laptop) and the MSP430 Comms Tester.

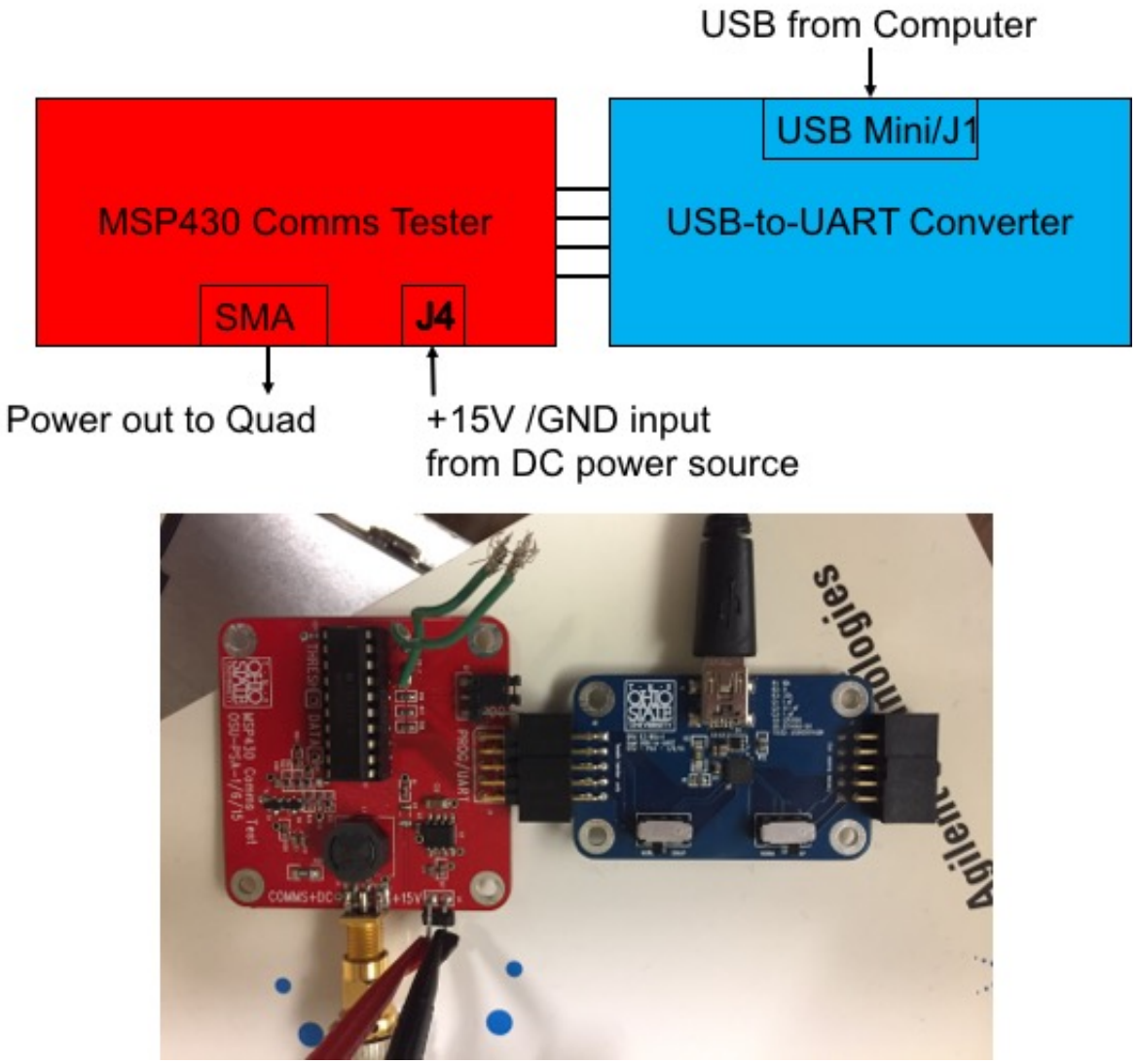

Figure 3: A photo of how to connect the MSP430 Comms Tester to the USB-to-UART converter, along with a functional diagram.

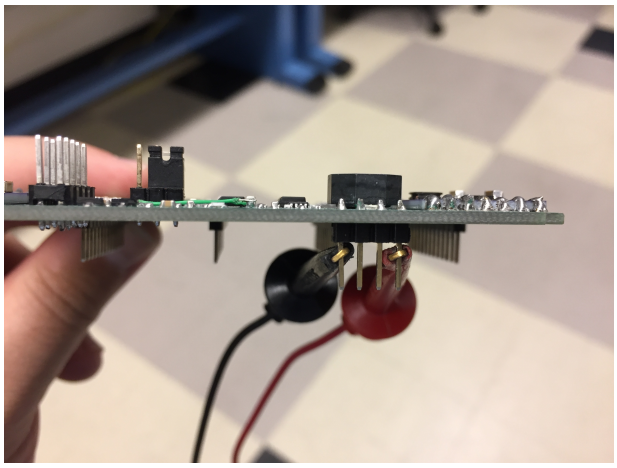

Figure 4: A photo of a correctly powered PC board. Red is +15V and black is GND. In the upper left, you can see the left-most pins of J2 have been jumpered together for "external power" mode. Note that because we are looking at the board edge on from the top, "left" and "right" are mirrored relative to the text.

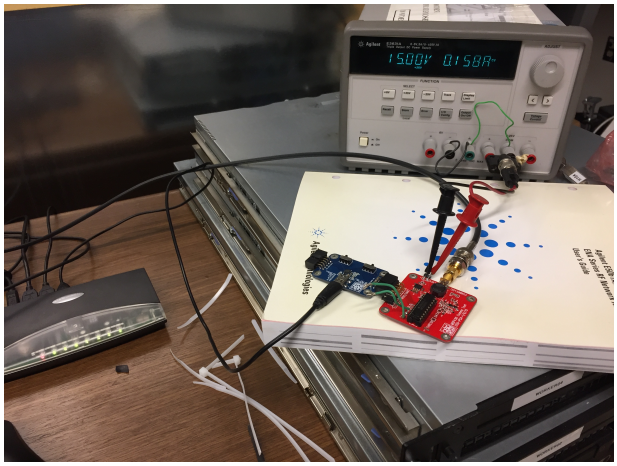

Figure 5: A photo of the complete power supply, USB-to-UART, and MSP430 comms-tester ready for testing.

2. Plug the USB-to-UART converter into the comms-tester. Wire the 15V intput and GND from a DC power source to the comms-tester, but do not turn the supply on yet. A photo of all devices plugged in and ready for use is given in figure 5.

3. From comms tester SMA output, deliver +15V to one of the two left pins of J1 on the PC board, and GND to one of the two right pins of J1 on the PC board. This is most easily done by splitting the SMA output to two micrograbbers.

4. Turn the power supply on. With only the PC board as a load (including DC-DC convereters) it should draw about 20 mA. (The photo in figure 5 shows larger current draw because it was powering an alternative set-up at the time.)

The PC board is reverse bias protected, so if the board does not power up, the polarity of the input is the first thing to check. A photo of the board correctly connected is in figure 4.

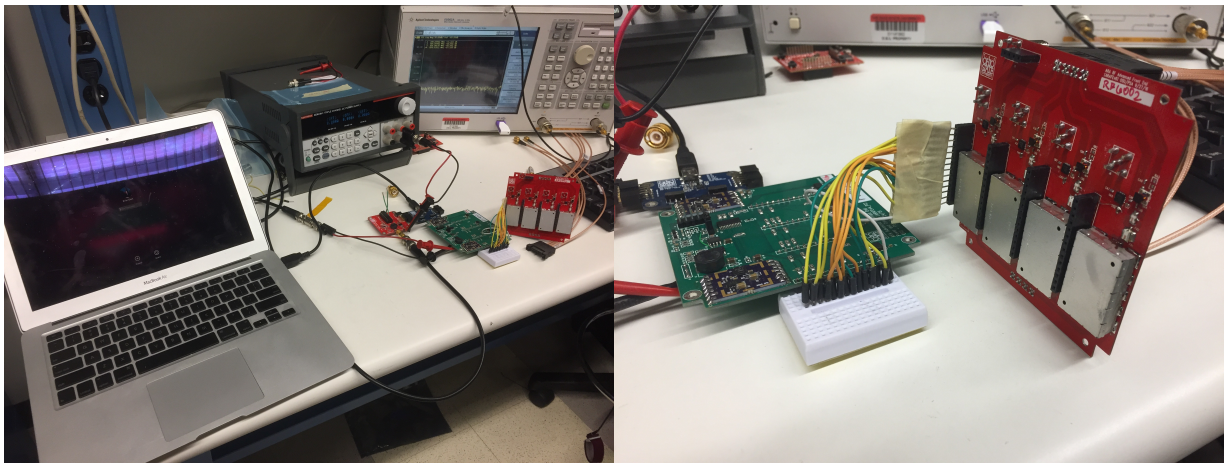

Figure 6: Figure of ARAFE RF testing setup at OSU.

1.4 Powering and Communicating with the RF Boards

Powering the RF board must be done carefully. Preferably, you should power the RF board only with the PC board, which will protect the RF board from any under/over-votlage issues, as well as reverse bias voltage. If you need to directly power an RF channel, be sure to get the polarity right, and to not apply more than +5V. Further, because the RF board is 31 mil FR4 (to keep traces thin), and because the header pins engage strongly between the PC and RF board, you should avoid plugging the PC board directly into the RF board during testing. Plying the two apart is difficult, and results in excessive flexing of the RF board. A better solution is to get a breadboard, plug in the male PC board, and wire "arduino type" jumper wires (eg: https://www.adafruit.com/product/758) to the female RF board. This narrower wires do not engage the female headers fully, and are much easier to remove, and even more so, can be removed without flexing the RF board. A photo of Brian Clark's test set-up in the Ohio State lab is visible in figure 6.

2 ARAFE Power and Control (PC) Board

2.1 Overview

The Power and Control (PC) board provides power to the RF board as well as controls power distrubtion downhole. The board is equipped with under-over voltage protection, reverse bias protection, and its central brain is a MSP430G2153 microcontroller. The board contains two custom, high efficiency DCDC converters; one converter steps down the 15V from the ASPS DAQ board to 5V, and the other to 12V. The 5V converter powers the RFSA variable attenuators and amplifiers for the RF board, while the 12V converter provides downhole power to the optical zonus via bias-tee. To vet a PC board, the board should be powered, and it's 5V convereter, 12V converter, and micronctroller should be tested.

2.2 Programming the PC Board

Programming the PC board is done via Code Composer Studio or CCS (http://www.ti.com/tool/ccstudio). The code repository for the PC board is called "arafe_slave_2" and is located in the ARA DAQ Github. I can be loaded into CCS using CCS's Git capability: File → Import → Git → Projects from Git.

To compile and upload the code to the microcontroller, you will use the 10-pin header on the board. If you have the TI USB FET Programmer you can use that directly. Plug the FET programmer into your computer, and hit Run → Debug, which should upload the code to the microcontroller. If you do not have the programmer, you can use almost any of TI's development boards as programmers also. That is explained here and here. You will jumper between the development board TEST, RESET, GND, and 3.3V to the corresponding pins on the PC board. Photos of how to do this are visible in figure 7 and figure 8.

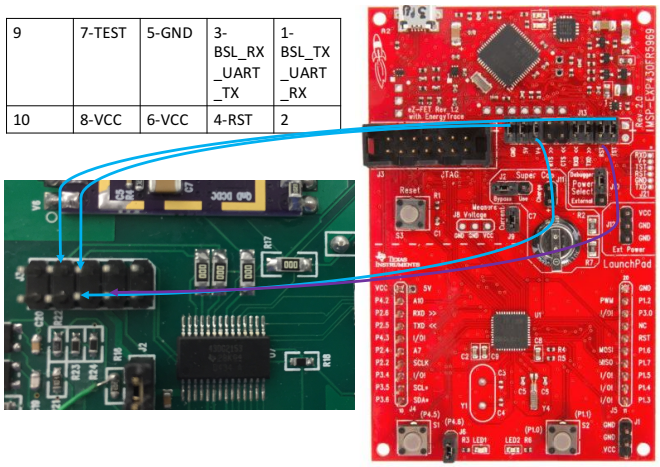

Figure 7: A diagram of the wire jumpering required to use a TI development board as a programmer.

Because the PC board/slave firmware uses the microcontroller information memory, CCS's default programming routine has to be altered. The memory management has to be changed from "Erase Main Memory Only" to "Erase Main and Information Memory." This can be done by doing Run → Debug Configurations → Target → MSP430x → Erase Options and select "Erase main and information memory." If when you try and program the board you get the following error "MSP430: File Loader: Verification failed: Values at address 0x1000 do not match Please verify target memory and memory map." this is the thing to correct.

2.3 5V DCDC Converter Testing

- Test the output of the 5V DCDC converter with a multimeter. It should register +5V. The output of the DCDC converter is labeled "Vout" in figure 11.
- Using a multimeter, probe pin 3, 8, and 14, on the jumpers, ensuring they are all +5V to begin.
- Issue the command "`5v 0`" to the slave tester command line. This will turn off the 5V converter; check with the multimeter that this is true. The current drawn on the power supply should decrease. Issue the comamnd "`5v 1`" to re-enable the 5V converter, and ensure the multimeter registers +5V, and check that the current draw on the power supply increases again.

2.4 12V DCDC Converter Testing

- Test the output of the 12V DCDC converter with a multimeter. It should register +12V.
- Issue the command "`12v 0`" to the slave tester command line. This will turn off the 12V converter; check with the multimeter that this is true. The current drawn on the power supply should decrease. Issue the comamnd "`12v 1`" to re-enable the 12V converter, and ensure the multimeter registers +12V, and check that the current draw on the power supply increases again.
- Using a multimeter, probe pin 1 of J4-J7, ensuring they are all grounded to begin.
- Issue the command "`control 0 [0-3]`" to the slave tester via command line. This will enable the individual 12V lines. Probe each line (0-3) in turn with a multimeter to ensure the voltage rises to +12V. There should also be an increase in current drawn from the power supply.

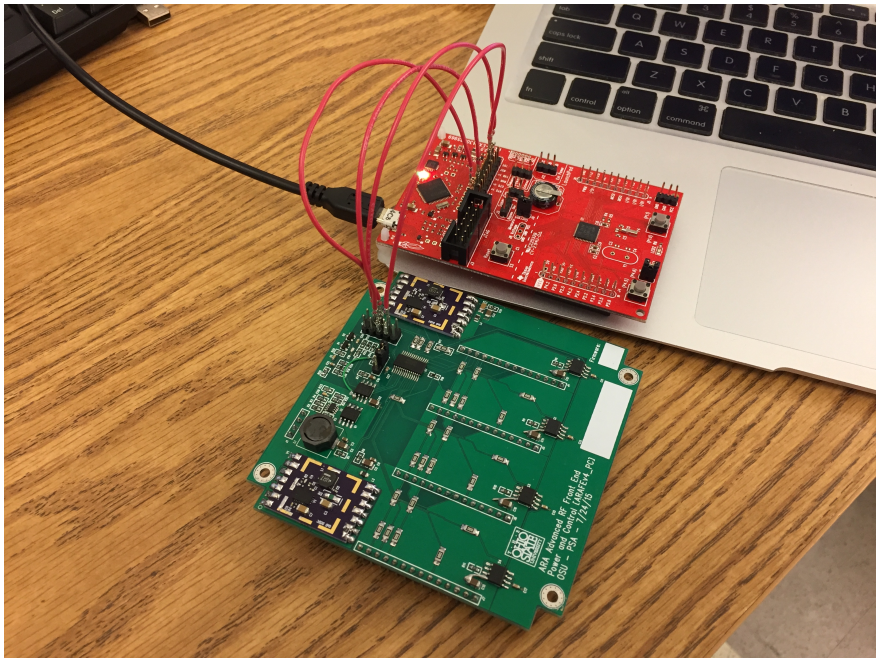

Figure 8: A picture of programming the PC board with a TI development board at OSU.

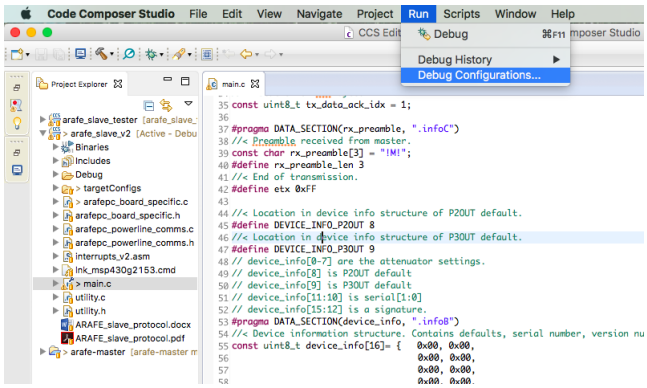

Figure 9: How to navigate to the debug configuration in CCS.

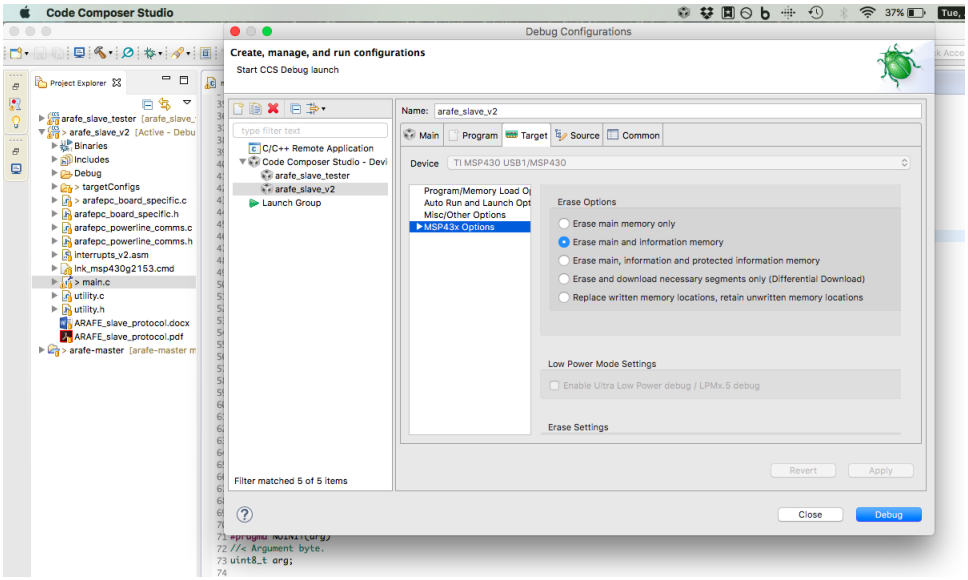

Figure 10: How to navigate to change the memory management in CCS.

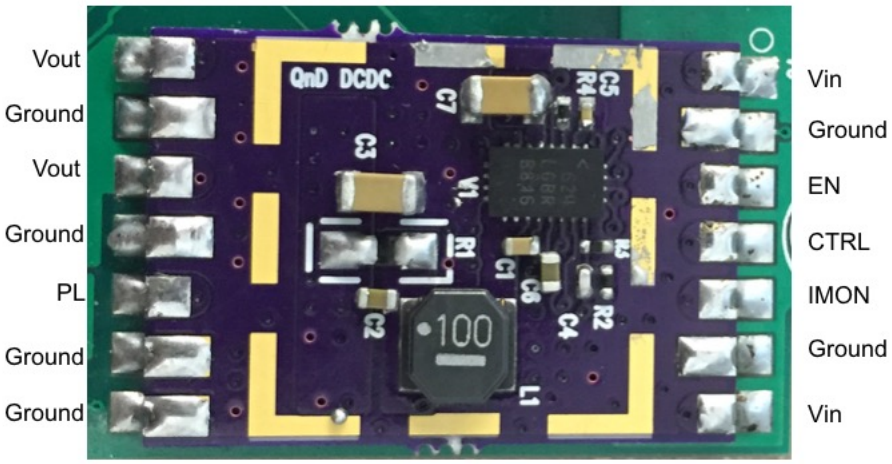

Figure 11: Photo of the DCDC converter with labeled inputs and outputs.

Atten Pin	Atten Function	Jumper Pin	uC Pin	uC Function
Chan 0				
12	Chan 0, Sig LE	JX-6	19	Chan 0, Signal LE
12	Chan 0, Trig LE	JX-12	18	Chan 0, Trigger LE
13	Chan 0, Sig and Trig Clock	JX-7 (SIG) & JX-13 (TRIG)	27	Global SPI Clock Pulse
14	Chan 0, Sig and Trig Data	JX-5 (SIG) & JX-11 (TRIG)	26	Gobal SPI Data Pulse
Chan 1				
12	Chan 1, Sig LE	JX-6	16	Chan 1, Signal LE
12	Chan 1, Trig LE	JX-12	17	Chan 1, Trigger LE
13	Chan 1, Sig and Trig Clock	JX-7 (SIG) & JX-13 (TRIG)	27	Global SPI Clock Pulse
14	Chan 1, Sig and Trig Data	JX-5 (SIG) & JX-11 (TRIG)	26	Gobal SPI Data Pulse
Chan 2				
12	Chan 2, Sig LE	JX-6	13	Chan 2, Signal LE
12	Chan 2, Trig LE	JX-12	12	Chan 2, Trigger LE
13	Chan2, Sig and Trig Clock	JX-7 (SIG) & JX-13 (TRIG)	27	Global SPI Clock Pulse
14	Chan2, Sig and Trig Data	JX-5 (SIG) & JX-11 (TRIG)	26	Gobal SPI Data Pulse
Chan 3				
12	Chan 3, Sig LE	JX-6	10	Chan 3, Signal LE
12	Chan 3, Trig LE	JX-12	9	Chan 3, Trigger LE
13	Chan 3, Sig and Trig Clock	JX-7 (SIG) & JX-13 (TRIG)	27	Global SPI Clock Pulse
14	Chan 3, Sig and Trig Data	JX-5 (SIG) & JX-11 (TRIG)	26	Gobal SPI Data Pulse

Table 1: Details of the connectivity between the RFSA pins, the jumpers on the PC/RF board, and the uC.

2.5 Microcontroller Testing

The microcontroller job is to program the RFSA3713 variable attenuators by toggling three analog lines: CLK (clock), DATA/SI (data), and LE (latch enable). The uC also allows for some current and voltage monitoring. CLK and DATA are mutual to all attenuators on all channels. Which attenuator will "listen" is controlled by setting the LE pin. We want to test if the uC is issuing necessary commands, to the right attenautor, by probing each of these three pins with an oscillosope, and looking for the analog bits in the oscilloscope traces. The pin mapping between the various inputs of the attenuators, various on board jumpers, and the pin output of the uC is detailed in table 1.

- Prepare an oscilloscope with two probes.
- Issue a command to control either the "signal" attenuator or the "trigger" attenuator. The command for the signal attenuator is "`sig [channel] [setting] = sig [0-3] [0-127]`" while the command for the trigger attenuator is "`trig [channel] [setting] = trig [0-3] [0-127]`".
- Check the CLK pins (J7 and J13) for every channel. You should see 16 clock pulses on every pin, no matter the command or channel to which the command was issued. We advise setting the trigger to this pin.
- Probe the two data pins J5 and J11 for every channel. The height of the transmitted pulse should be roughy half that of the +5V line.
  - Should see something like the SI line on the RFSA timing diagram 12. A picture of the oscilloscope trace is in figure 14.
  - There should be a high bit at the end of the bit train (A7) regardless of the command issued.

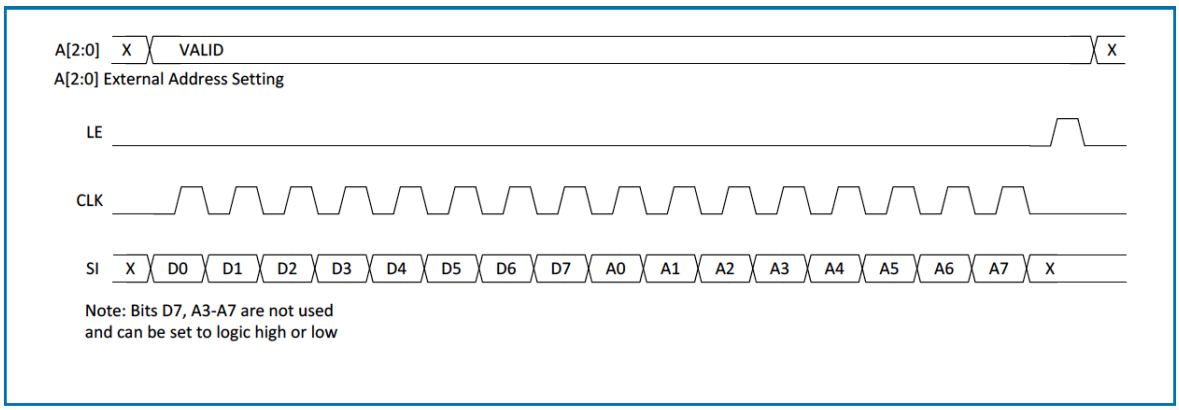

Figure 12: The bit timing diagram for programming RFSA3713 variable attenuator. From [1].

  - Bits D7-A6 should all be low, regardless of the command issued.
  - The first bits D0-D6 should be a combination of low and high, corresponding to the command issued. Note that the seven bits are exactly the number required to control the range of the RFSA; all 0's for setting zero, or minimal attenuation, and all 1's for setting 127, or maximal attenuation.
  - One should see the same command on every data pin, no matter the command or channel to which the command was issued.

- Check the LE pins (J6 for signal and J12 for trigger) for every channel, both signal and trigger. This is where the differences between the "sig" and "trig" commands, as well as the differences between channels, is important.
  - You should see one bit after the last clock bit (see timing diagram 12). A picture of the oscilloscope trace is in figure 13.
  - This bit should only be present in the channel to which the command was issued. That is, if you issue "sig 1" you should only see a bit on the signal attenuator pin of channel 1 (header pin JX-6 of channel 1), and no other. The firmware has already been debugged, so the only way this should not work is if a pin is shorted incorrectly.

2.6 Thermal Cycling and Programming the Serial Number in Flash Memory

Each board should be thermal cycled, and the microcontroller test in section 2.5 should be repeated for all channels to check for broken soldering connections. Once it has passed it's thermal cyclce, the uC needs to be assigned the board's serial number. The uC for each PC board has onboard flash memory to store a permanent serial number. This means that the serial number can be recovered after deployment if need be. At the end of testing (post thermal cycle), a serial number should be written on the board, and flashed into memory. In revision four of the boards, for example, a PC board might have the serial number PC4002, where the "PC" identifies it as a power and control board, the "4000" identifies it as board revision 4, and the 2 is the board specific identifier. In memory, device info 10 holds the most significant byte (MSB) of the serial number, and device info 11 holds the least significant byte (LSB) of the serial number. We want the numbering to start at 4000, which is 0x0FA0 in binary. In the following example, we will program the serial number for board 4002, or 0x0FA2. So:

- We set the MSB to 0x0F, which in decimal is 15. So we will issue the command "`write 10 15`". Now, we have the total value as 0x0F00, which is 3840 in decimal.

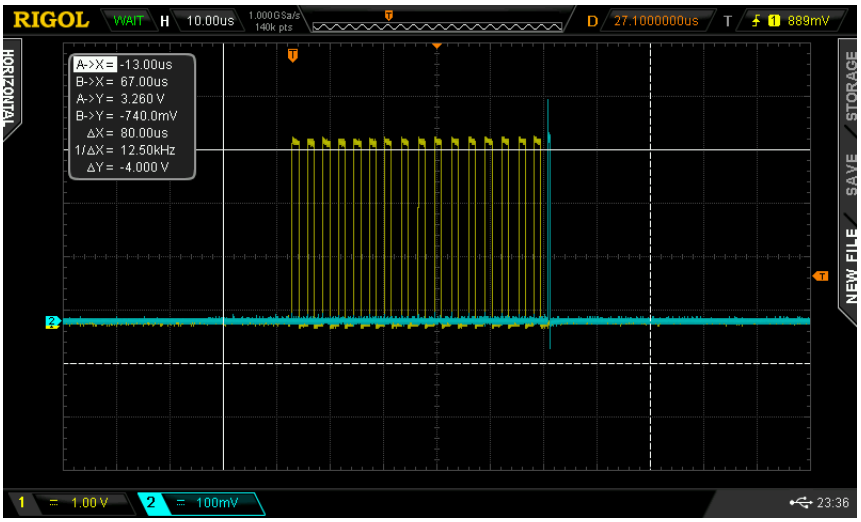

Figure 13: A photo of the uC clock (yellow) and latch enable (blue) pulses on an oscilloscope.

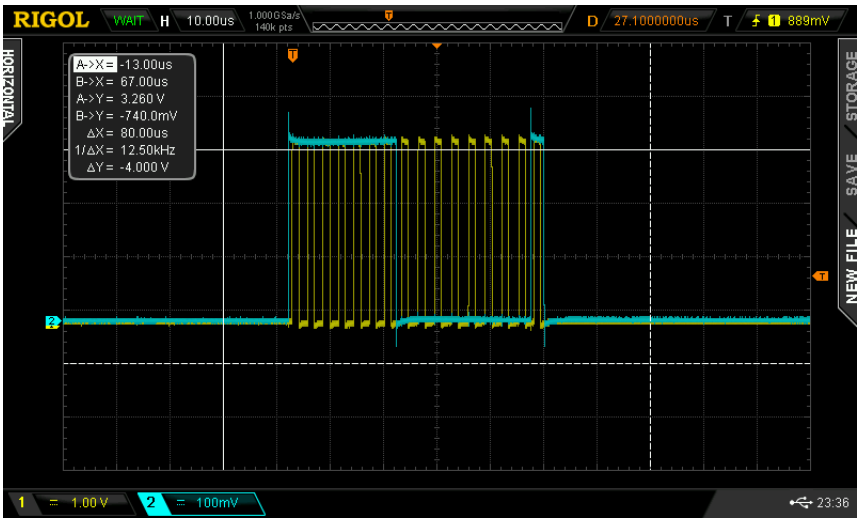

Figure 14: A photo of the uC clock (yellow) and data (blue) pulses on an oscilloscope.

- Now, we set the LSB to 0xA2, which is decimal 162. So we will issue the command "`write 11 162`". To summarize, what we have done is program the serial number to be 0x0FA2, which is 0xF000 (3840 in decimal) + 0x00A0 (160 in decmial) + 0x0002 (2 in decimal), or 4002 total.
- Issue the command "`flash`" which will store the serial number in the flash memory of the uC.
- Issue the dump command "`dump`" which will write out the content of the flash information memory to the terminal. Check that the third row, right two columns reflect the serial number you just programmed.

3 ARAFE Radio Frequency (RF) Board

3.1 Construction and Initial Testing

Testing of the RF board should be done channel by channel. After attaching surface mount components, testing of the RF channel requires installation of

- an iso-rate adapter
- a header through pin
- an SMA.

Here is some advice on building the boards

- Test channel 3 first, and channel 0 last. This allows you to power up the channels, while still having full access to the surface mount components.
- Even without power, the S11, S22 trigger path, S22 signal path, and S21 signal path all have distinct gain patters. They can be observed in figure 15,16,17, and18. Therefore, you can actually check many of the necessary connections on the board without powering it on.

Once powered:

- The S21 signal gain should be smooth across the band up to ±2dB as in figure 19. All network analyzer traces should be from channel 2 of RF4003 (this is here for OSU's documentation sake).
- The S21 trigger path will only be open once powered (this is because in most couplers the couple path is DC shorted to ground). The S21 gain should be as in figure 20. Note:
  - The trigger path is attenuated an additional 10 dB, consistent with the choice of coupler.
  - There are "ripples" in the response that are likely due to reflections off the coupler, and potentially the SMA. Given that the couple path is routed to the trigger, and the time integrator functions over ~5 ns, this reflection should not affect the triggering effectiveness.

3.2 Attenuator Testing

Now that we know the channels are working, we need to verify we can set the variable attenuators. The attenuators have a setting between 0 and 127, with each digit for a 0.25 dB increment of the 31 dB total attenuation possible with the RFSA 3713. The procedure is the following:

- Power a channel
- Set the signal attenuation with the command: "`sig [channel] [setting] = sig [0-3] [0-127]`" Play around with various value of attenuation, and you should see the curve move up and down, as in figure 21.

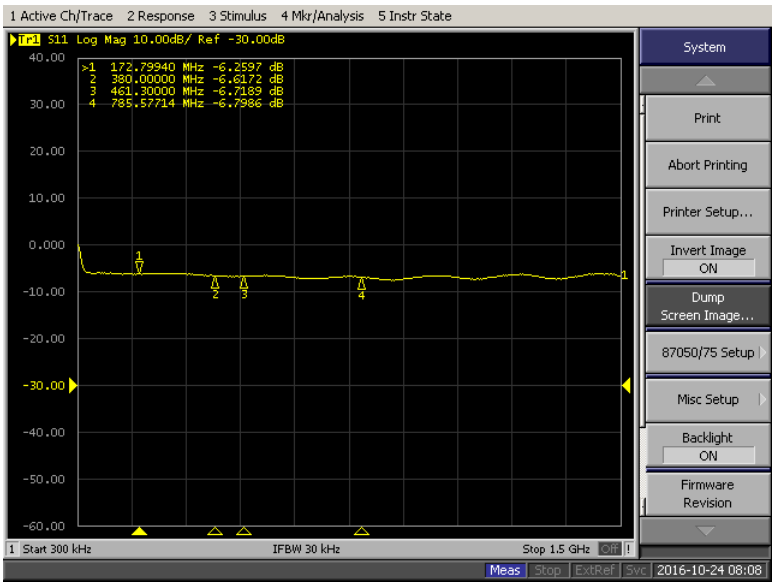

Figure 15: Ideal S11.

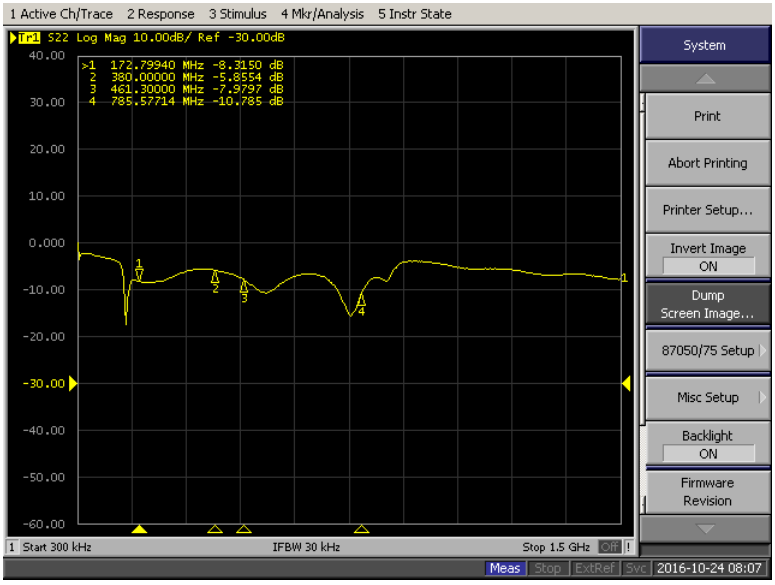

Figure 16: Ideal Signal Path S22.

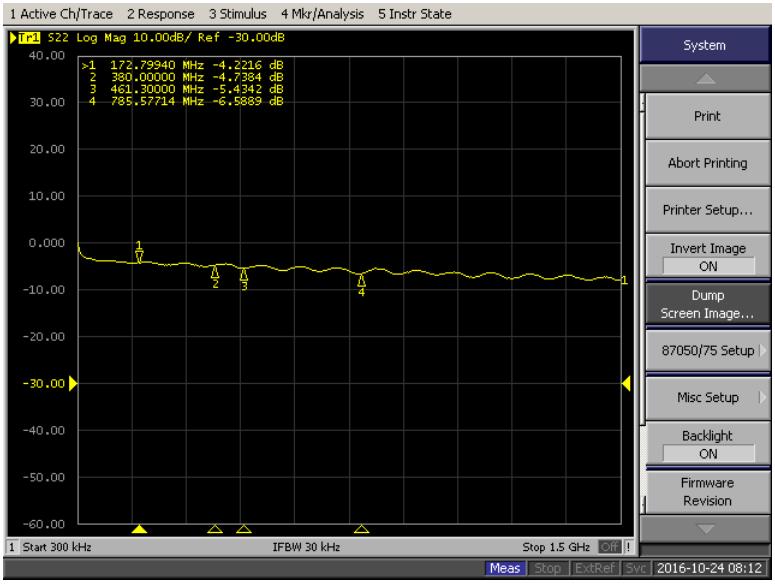

Figure 17: Ideal Trigger Path S22.

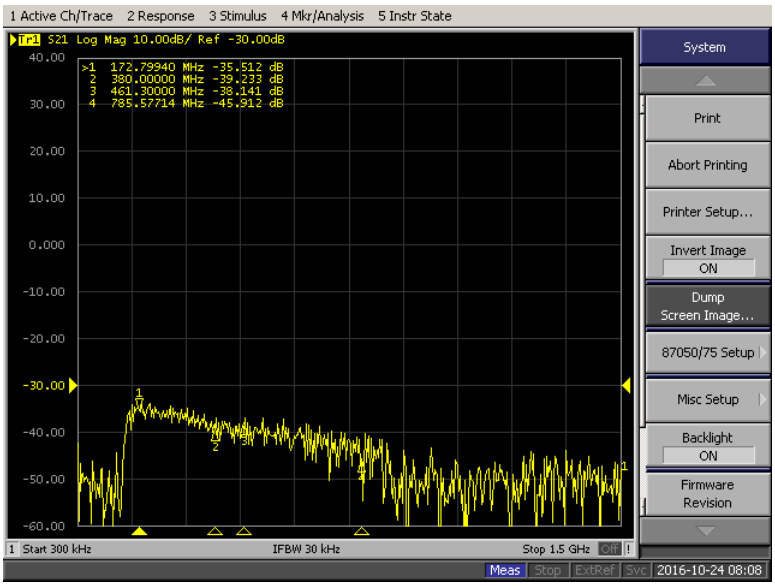

Figure 18: Ideal non-powered signal S21. Note that even with the power off, the system response can be observed in the gain.

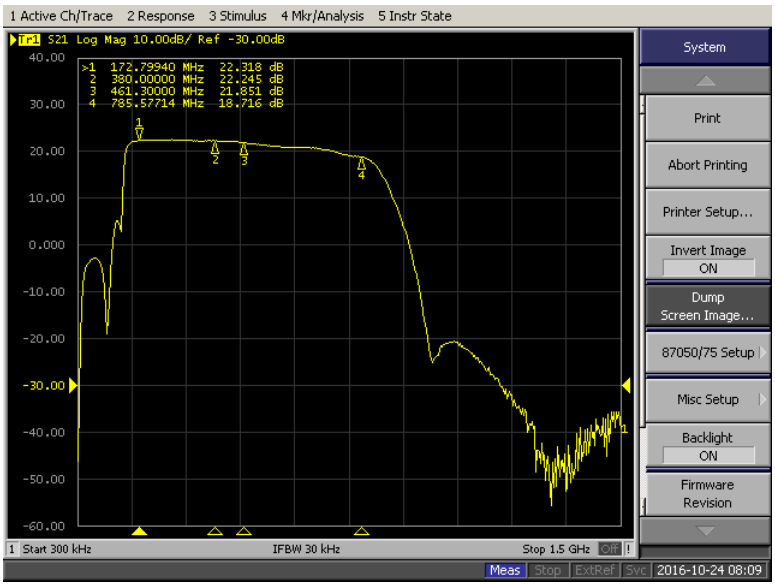

Figure 19: Ideal powered signal S21.

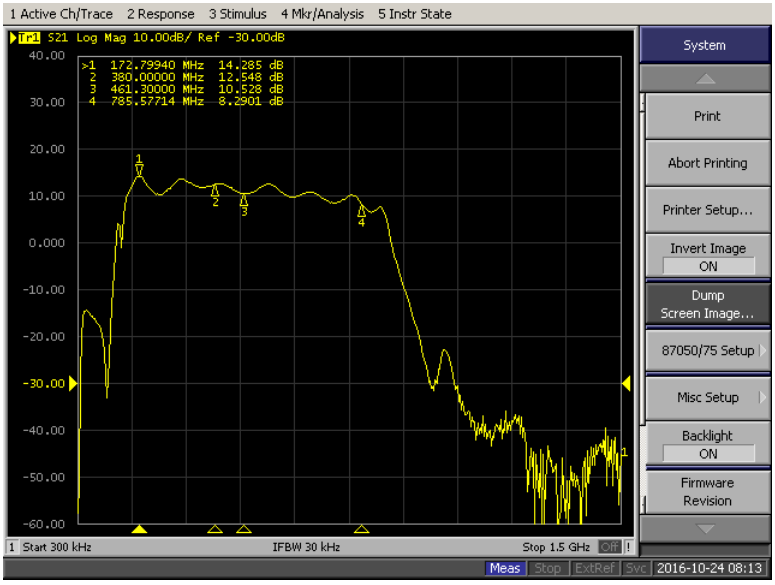

Figure 20: Ideal powered trigger S21.

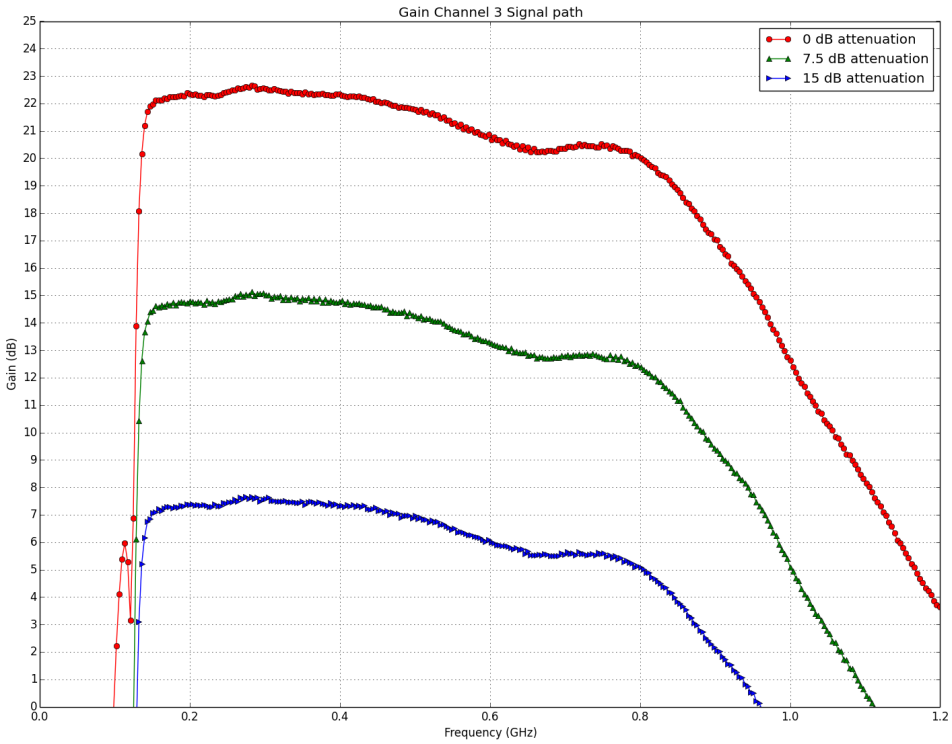

Figure 21: Example of a signal path gain pattern in an ARAFE at several different attenuation values. Note that this plot was made with a LFCN-800+ low pass filter. This was later replaced witha LFCN-630+ filter that has a sharper cutoff above 850 MHz. So the curve you observe should fall away more than 20 dB above ~850 MHz, as in figure 22. A sign that you have chosen the wrong filter is if the gain does not fall off fast enough.

- Set the trigger attenuation with the command: "`trig [channel] [setting] = trig [0-3] [0-127]`". Play around with various value of attenuation, and you should see the curve move up and down, as in figure 21.
- In both cases, make sure that setting the attenuator to "0" results in minimal attenuation, and that setting the attenuator to "127" results in a reduction in the gain by 30 to 31 dB.

Plotted system responses for channel 3 signal and trigger paths. This was done for RF4001 (this is here for OSU's documentation sake).

3.3 Thermal Cycle and RF Caging

Once all RF channels are working, you should thermal cycle the boards, and repeat section 3.2 to ensure no solder joints have broken in thermal test. If joints break, repair them. Once the boards have passed their thermal test, you should attach their RF cages, and give them a serial number on their silk screen. Attaching cages is easiest if you solder the corners first from the outside, and solder the inner pads near the headers last by poking the iron through the open cage. In all cases, you can use solder paste instead of a wire of solder if you like.

References

[1] RFSA3713 Variable Attenuator Datasheet. http://www.rfmd.com/store/downloads/dl/file/id/30109/rfsa3713_data_shee

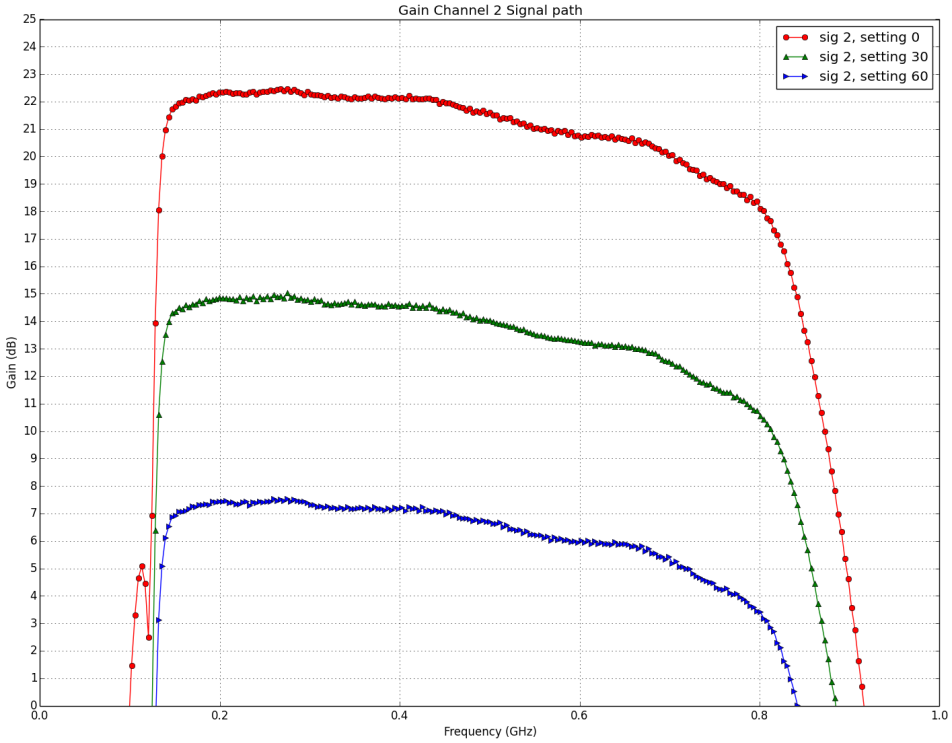

Figure 22: This is an example of the signal path gain with a correctly chosen filter, the LFCN-630+.

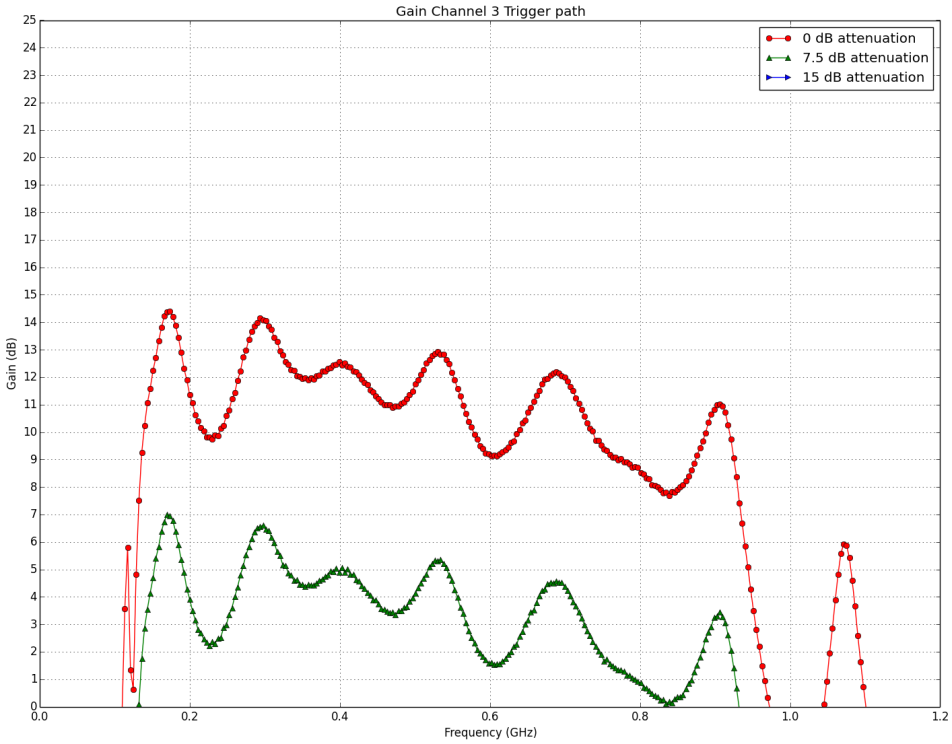

Figure 23: Example of a trigger path gain pattern in an ARAFE at several different attenuation values.