PrimEx Note 27

Setup and Operation of Trigger Electronics for PrimEx

(Created by Kazem Edmond 8/01/01)
(Revised and added programs by Pjerin Luli 8/20/04)

Introduction

The trigger electronics in PrimEx will provide fast and accurate evaluation of neutral pion Primakoff events. The trigger electronics consists of:

Counting Room Hall-B:
A Windows XP Machine

Hall-B:
A National Instruments Ethernet-to-GPIB Controller
A CAMAC Crate
A Kinetic Systems 3988 GPIB Crate Controller
3 Units of CAEN Model C 542 Dual Programmable Logic Units
A Bi Ra Systems 8 channel 12-Bit Digital to Analog Converter (DAC), Model 5408

Extra devices added for other purposes are:
2 units of Phillips Scientific 16-channel Discriminator Latch, Model 7106
2 units of LeCroy Research Systems 16-channel Discriminator, Model 2313
2 units Lecroy Research Systems 16-channel Delay Unit, Model 4518

Installation and run of NI GPIB-Enet/100 Ethernet-to-GPIB Controller

The device of the setup is the National Instruments GPIB-Enet/100 Ethernet to GPIB controller, which comes with a manual and a NI-488.2 installation CD.

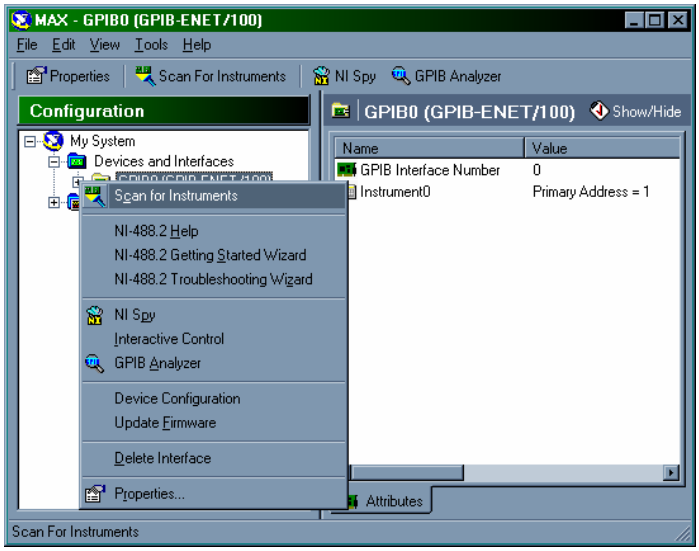
The NI GPIB-ENET/100 and NI-488.2 take advantage of network connectivity in instrument control applications. Using the NI GPIB-ENET/100, networked computers can communicate with and control IEEE 488 devices from anywhere on an Ethernet-based TCP/IP network. You can use the GPIB-ENET/100 to share a single GPIB system among many networked users or to control several test systems from a single networked host computer *.

Here are some simple steps to install the GPIB-Enet/100.

1. Plug the GPIB-Enet/100 to power socket, connect it into the Ethernet Network, but do not power it at this point.
2. Insert NI-488.2 CD into the Windows XP machine and chose to install NI-488.2 Software.
3. A GPIB Hardware Wizard will show up. GPIB-Enet/100 should be selected from the list of supported devices and the machine should be rebooted at this moment.

4. You need to record the specifications of the device. In order to acquire an IP address on the network for the GPIB-Enet/100, the network administrator might need these specifications such as model, part #, serial #, Ethernet address which can be found at the bottom of the device. The hostname at Hall-B for this device is primexenet2.
5. In order to manually assign the IP address to the GPIB-Enet/100 device, its needed to put it into configuration mode by pressing the inset pinhole button that is located on the back of the device and written beneath is "CFG Reset". The manual at chapter 6 gives some simple steps of this procedure.
6. Chapter 3 and 4 in the manual shows what needs to be done next in order to get the device running.

The NI-488.2 has installed a program called MAX (Measurement and Automation Explorer), which has been proved to be a useful program that can display the status of the controller at all times. A successful way to track the GPIB-Enet/100 through MAX is to launch the program and as shown in the image on your left, open the directory "Devices and Interfaces" that is located in the menu on the left-hand side of the screen. Right click on the subdirectory "GPIB0 (GPIB-Enet/100)", scroll down and click "Device Configuration".



If the configuration at step 6 was successful, you should see the device appear with its IP address. Now you are done with the setup procedure. Next time when the device is powered, it will take about 30 seconds to run the self-tests automatically and it will be ready to go.

Keep Max icon handy in order to checkup on the status everything that is enrolled in the network communication. A useful tool is spyware that records every step on the network and alarms on any error.

Note: If you select "Devices and Interfaces", click on "GPIB-Enet/100", scroll down and select "Properties". At "Network Settings" there is a "Search for GPIB-Enet/100". If you click that it will say it could not find a device on the subnet. Counting room and hall-B are in different subnets. You should disregard that and write by hand the name primexenet2.jlab.org under IP address/Hostname. When you selected "Device Configuration" above, it says "Discovering Ethernet Devices on the subnet" but instead the program searched beyond its own subnet.

A potential problem might be that Max cannot find the GPIB-enet/100 in the network. Make sure that the IP address you got from the network administrator has no possible constraints to certain devices. Also make sure that your computer and the GPIB-

Enet/100 have IP addresses manually set so there will be no discrepancies in the communication between them. If the problem persists, follow carefully the instructions at the GPIB-Enet/100 manual to reconfigure the device and try to communicate with it again.

Installation and run of KS3988 GPIB Crate Controller

The device is a Kinetics Systems 3988 GPIB Crate controller that is used to control the CAMAC crate, and its manual. Be sure everything is powered down beforehand. Slide the KS3988 controller into the CAMAC crate at slots 24-25. The KS3988 controller should have a jumper cable connecting the "Request" output to the "Grant In" input located on the device's front panel. Use of this jumper is noted in the controller's manual.

Connect tightly the KS3988 controller with the GPIB-Enet/100 together using a GPIB cable, by using the cable's screws. Power up the GPIB-Enet/100, CAMAC crate, and wait about 30 seconds for the GPIB-Enet/100 to run the self-tests in order to declare its place on the network.

Run Max program again, and open the directory "Devices and Interfaces". Right click on the subdirectory labeled "GPIB0(GPIB-Enet/100)" and select "Scan for instruments" option. "Instrument0" which is our KS3988 controller should appear on the right-hand of the window, which also shows the GPIB interface number and Instrument0's primary address. The controller's address which is10 must be specified in any application or VI that interfaces with it. KS3988 manual shows how to change the GPIB address.

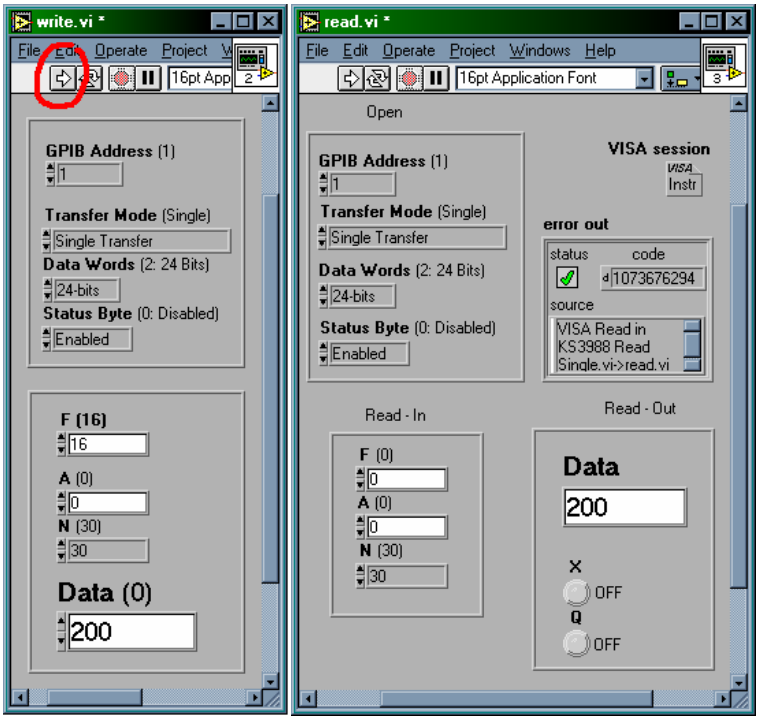
Note: NI Getting started wizard may tell you to communicate with the controller, Instrument0, using MAX. Communication using MAX will not work since the controller is using an older revision of GPIB than MAX is designed for. Originally GPIB used the IEEE 488 standard. Since its inception in 1978 there have been two revisions, putting GPIB at IEEE 488.2. All NI instrumentation and software including MAX 2.0 are consequently compatible with IEEE 488.2 protocol, so some commands, primarily ASCII commands, will not work correctly with our controller which follows the IEEE 488 standard.

Using LabView with the KS3988 GPIB Crate Controller.

Note: Every time you close a LabView VI, the program will ask if you want to save the changes. Say "no" unless you are sure about what you are doing. The Labview Programs under the name "Primex Trigger Electronics " same as MAX are located as shortcuts on the Desktop.

In order to see if you have established the communication with the KS3988 controller, open Read.vi and Write.vi Labview programs, which are two basic VIs that use a so called Visa protocol. These two programs will test the communication with the KS3988 controller and the Visa protocol that is installed in the Windows machine.

Here are two pictures of the VIs. Remember that our GPIB address is 10. To run the program you should single click on the white arrow in the upper left-hand corner of the Labview window (circled in red).



The Write VI sends a function call to slot 30 which is the CAMAC internal N value and writes any unsigned 16-bit number to the controller's Transfer Count Register (TCR). You can specify the 16-bit decimal in the "Data(0)" field. The Read VI sends a function call to the controller requesting the contents of the TCR and displays those contents as a 16-bit decimal in its "Data" field.

Note: When KS3988 is turned on, you will see some lights on including inhibit. The write and Read programs de-inhibit the device when they want to talk/listen with it.

Installation and run of the Digital to Analog Converter (DAC)

The device is a Bi Ra Systems Model 5408 8 channel 12-bit DAC and its manual. Install DAC right after you have installed the GPIB crate controller because it is simple and easy to use so it is a good way to make sure that the rest of our equipment is working. DAC is a device that converts digital information into a corresponding analog voltage or current. In our case we will be using it to set voltage threshold.

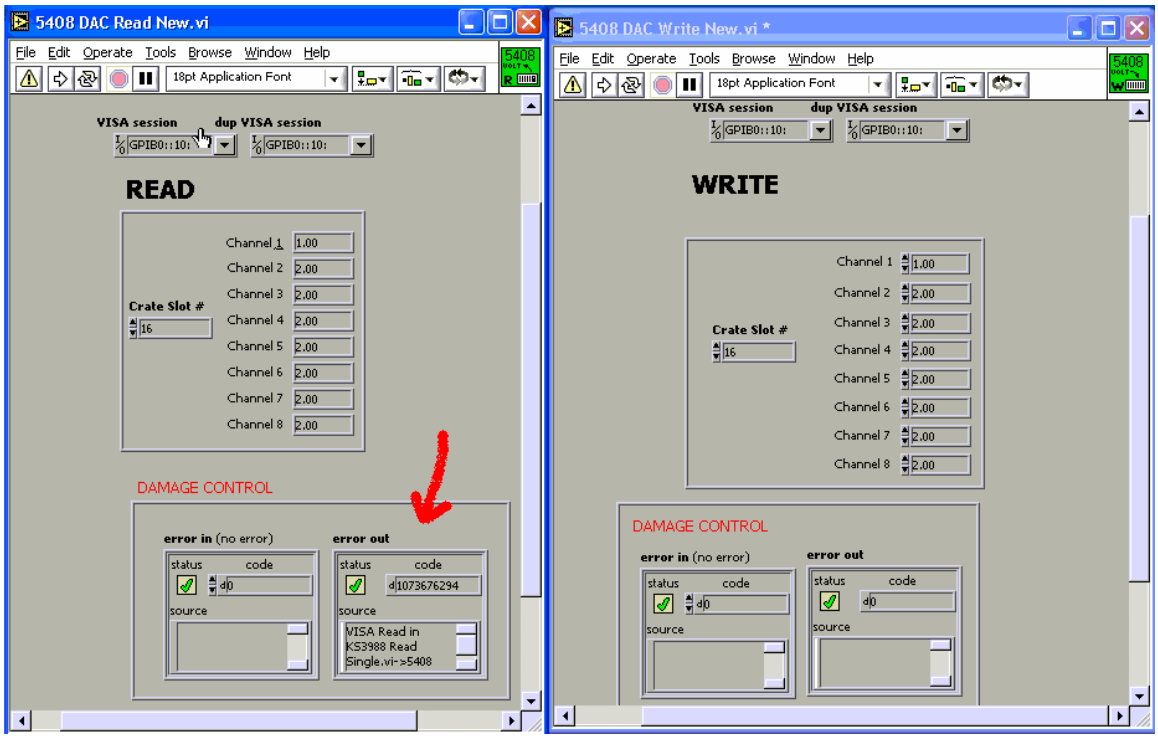
DAC has 8 NIM level output ports on the front panel. Each channel outputs the contents of a 12-bit register that can be written or read by using CAMAC functions that can be found in the manual. The default output range of the DAC's voltage is +/- 10V but it can be set manually to other values too, although our needs are for a range of 0 to 10 V. The voltage offsets can be adjusted, using OFFSET pots, quite easily. The pots are tiny flat-head screws located within the topside of the DAC.

The way DAC works is that it stores 12-bit binary values in each of its registers. A binary string of 000,000,000,000 will output the minimum voltage. A value of 2^{12}-1, or 4095, will output the maximum voltage. Likewise, 2048 will output the middle voltage. In the current configuration, 2048 outputs 0V, 4095 gives us 9.999V, and 0 gives us –10V. The DAC Read and Write VIs have the proper Read and Write function codes written in the manual in order to enter the voltage values and convert them to12-bit binary values using the appropriate transform equations.

The actual maximum value is 9.99V. If you enter 10V, or 4096, then the register will wrap around and store 0 instead.

Using Labview to set the DAC voltage.

The DAC Read and Write VIs are simple to use. First be sure to enter the correct slot number and GPIB address into their respective fields. The picture below shows the DAC Labview VIs. Use **5408 DAC Write New.Vi** to write the values to the DAC's 12-bit memory for all 8 channels simultaneously. Use **5408 DAC Read New.Vi** to read the voltage for all 8 channels.



Note: Notice that beneath the read arrow there is a combination of a green check mark and an error code. The combination of two means that Labview has issued a warning, but the warning did not fatally impact the program's operation.

We noticed that DAC was not working at a certain slot, and the reason was that the slot was bad. Take that into consideration, also the fact that CAMAC is a relatively old equipment.

*A note to developers: The **KS3988 Read Single.Vi** Problem.*

All of the VIs that operate with this setup use sub-VIs that were specifically written for the Kinetic Systems 3988 CAMAC Crate controller that we use. These sub-VIs were downloaded from
ftp://ftp.ni.com/support/instr_drivers/labview/windows/current/library/gpib/ks3988.zip
The sub-VIs are relatively new and use a protocol called VISA that serves as an abstraction layer for the programming of VIs. In essence it is supposed to make the job

of programmers easier. The problem with VISA is that programs that use VISA tend to be too general, often to the point that they do not work, which appears to be the case with one of the downloaded sub-VIs. One of these sub-VIs, **Read Single.vi**, is responsible for Single Transfers to and from the controller. At the core of this VI is a sub-VI called **VISA Read.vi**. One of its inputs is a byte count that tells it how many bytes it should expect to read. As installed VISA Read is expecting a string of data that is 10 bytes in length. Its impossible that a Single Transfer from the KS3988 Controller will ever amount to 10 bytes. Even with the longest word length, 24-bits which is 3 bytes, and a status byte, 8-bits which is 1 byte, you are still working with 4 bytes, far from the expected 10 bytes. Seeing this some logic was added to **Read Single.vi** that sends the "correct" byte count to **VISA Read.vi**, but since you will never send 10 bytes in a single transfer you will always receive the "byte count" warning.

A better solution is to use a VI library that does not use VISA at all. The library is titled **ks3988_old.vi** and has been included with this manual. For the same reason, **KS3988 Read Block VI** has not been used. Instead, all block transfers are done using the older VIs. The older VIs were discovered late into the development cycle so it was too late to rewrite everything. Although using the older VIs is like flying by wire, you can be sure that they will work more often than not.

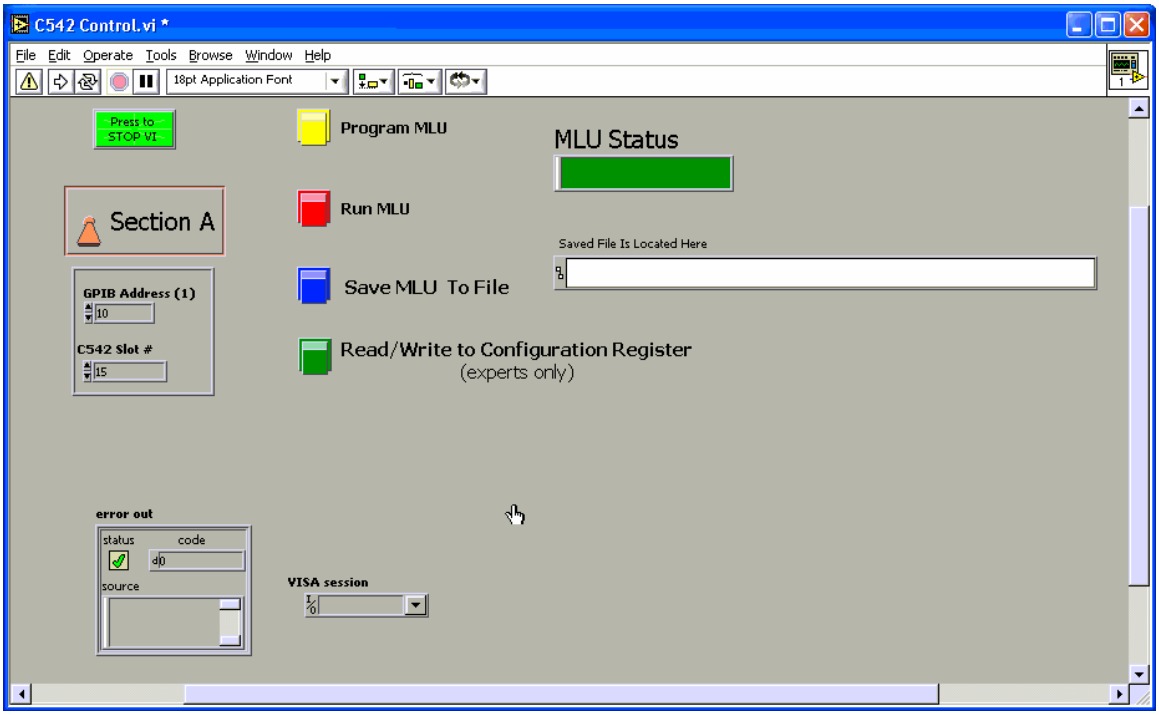
A potential factor that VISA does not agree with our controller is that our controller is IEEE 488 compliant as mentioned at the note above (the original GPIB standard) but is not compliant with the revised IEEE 488.2 standard, which is what VISA was designed for.

Installation of C.A.E.N C542 Dual programmable Units (MLU)

The CAEN Model C542 is a 1-unit wide CAMAC module that houses 2 independent sections with 8 Input and 8 Output channels each. We have three of these modules with the manual included. On the front panel of each section is a three-position output mode selector, a strobe input, a sync input, and an adjustable output width setting. Also, the unit has a clock input if an external clock device needs to be used. Each section contains an 8-bit 256 address memory cell and 3 data registers: an output register, a configuration register, and an address register.

By writing into the Configuration register we can select three different working modes for either section: lookup table mode, pattern generator mode, and the 8-bit input/output Register. We are most interested in the lookup table mode of this unit. Each lookup table contains 256 memory addresses within an 8-bit RAM unit that can be read and written to via CAMAC or via the input and output connectors.

The three-position switch that is located in each section is used to select the output mode and can be enabled or disabled using the configuration register. The switch's output modes are normal mode, strobe mode, and shape mode. You can ignore the switch, for the most part, since all the default configuration strings used in the VIs disable the switch (Normal Mode). The unit's functionality is described in greater detail in the C542 User's Manual.



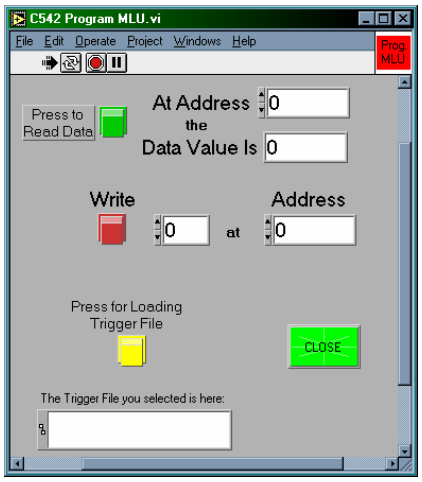
After installing the three units into the crate open **C542 Control VI** but before you start running the VI follow these steps:

1. Make sure rectangular button in the upper left-hand corner is **green**. Once the VI is running you must press it once more to make the VI cease operation. Upon pressing this button you will notice that it turns red indicating that the VI is not running and is not ready to run.
2. Make sure you have GPIB Address 10 written. Select the C542 slot number you want to use, and select the GPIB 10 address for Visa Session.
3. Notice the orange toggle switch on the left-hand side of the screen. Prior to any operation be sure that the section that you desire to read or write to is selected. MLU's two sections are completely independent from one another.

Once the VI is running we can begin to perform some operations and use the MLU. Also, notice the green field labeled "MLU Status". Each time you press a button, or perform any kind of operation, the Status field will tell you if you are in Run Mode or Program Mode.

Pressing the yellow button labeled "Program MLU" will launch a pop-up window that will give you access to read and write functions. The pop-up window **C542 Program MLU VI** is shown below. One way to use this window is to enter an address in the "At Address" field and then press the square green button. You will immediately see, in the "Data Value" field, the 8-bit decimal that is being stored at that address. Next, to

change the value stored at an address, enter the 8-bit value you would like to write in the field next to the red "Write" button, then type in the 8-bit address where you would like to store that value, then press the red button. If you would like, you could perform a read at that address by pressing the green button to make sure that the write function worked.



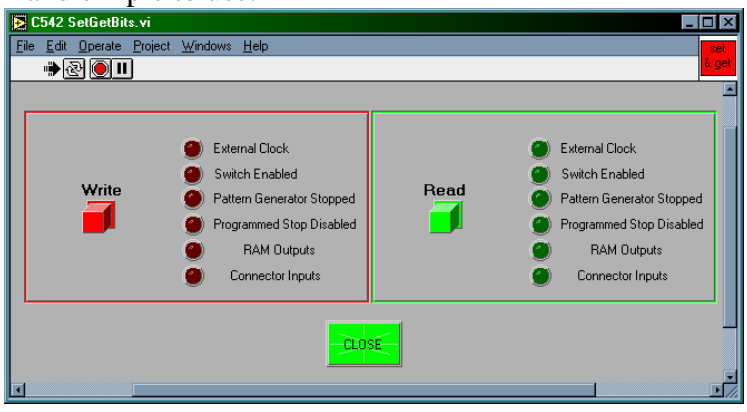
We are interested to put the MLU into Lookup Table Mode. In general after the window of **C542 Program MLU VI** shows, press the yellow button that says "Press for Loading the Trigger File". You should be presented with a dialog requesting a file to input into the MLU. The file that you select should be an ASCII file in column format, meaning that there is a single 8-bit decimal per line and there should be 256 lines. If you chose to input a bigger line ASCII file than 256, the VI will read only the first 256 lines. The file path to the file that you selected is displayed in the field located at the bottom of the window. Press the green button "Close" to close this window and return to C542 Cotrol panel.

If you want to run in Lookup Table mode but would like to enter the values by hand, as opposed to a file, then press "Close". In the C542 control panel (C542 Control.Vi) you will see a red button that is labeled "Run MLU". Pressing this button will run the MLU in Lookup Table mode using whatever values that have been entered into it.

The next button in this screen is a blue one labeled "Save MLU to File", and pressing it will do just that. You will be presented with a window requesting the name and location of where to save the MLU's contents. The MLU's contents, as described above, will be saved in an ASCII text file that contains 256 lines of 8-bit decimal values, each value being a separate entry located at a different memory address.

Note: The save function at "Save MLU to File" will either **replace** or **create** a new file. It will not append data to an existing file, so be careful not to replace any existing trigger files that may be important. The trigger files can be read using almost any word processing program. The "Notepad" program, which is included with every Windows machine, has proven to be particularly useful for viewing the contents of the trigger data files since it is quick and simple to use.

Now, if you wish to gain more control over the MLU's configuration, press the green button located at the bottom of the screen of **C542 Control VI**. You will be presented with a window that you can see right here **C542 SetGetBits VI**. Each section of the MLU has a configuration register. The



register stores a 6-bit binary number where each bit controls a different function of that section of the MLU. So in the pop-up window we can configure each of these bits graphically, without entering ones and zeroes. The text next to each bit-control describes the function currently selected. If you look at the figure, you can see a green and red box. The red box, on the left, is where you can enter a bit to be written to the register. After selecting your bits, press the red button labeled "Write".

You can select which section that you would like to write to by toggling the orange switch. You can read the current configuration of either MLU section by toggling the orange switch located in the green box and then pressing the square green button. When you are finished configuring the MLU by hand, press the large green rectangular button labeled "CLOSE" located at the bottom of the window. Doing so will return you to the **C542 Control VI**.

Note: In order to load the file into the MLU, the device needs to be set at certain bits, but you do not need to worry about that because the Program MLU does that for you. This is what happens step by step: when you click on Program MLU, the bits are set automatically which makes possible to load the file into the MLU device, when you close the window, the bits turn off and the file stays inside the device.

If for some reason the Program MLU is partially damaged and cannot set the right bits to the device, than you will have to set the bits manually at **C542 SetGetBits VI** by clicking on "Ram Outputs" and "Connector Inputs" which will respectively read after "Output Registers Outputs" and "Address Registers Inputs". At this point is crucial to check the Program MLU to see if you still can go through the loading process of the file.

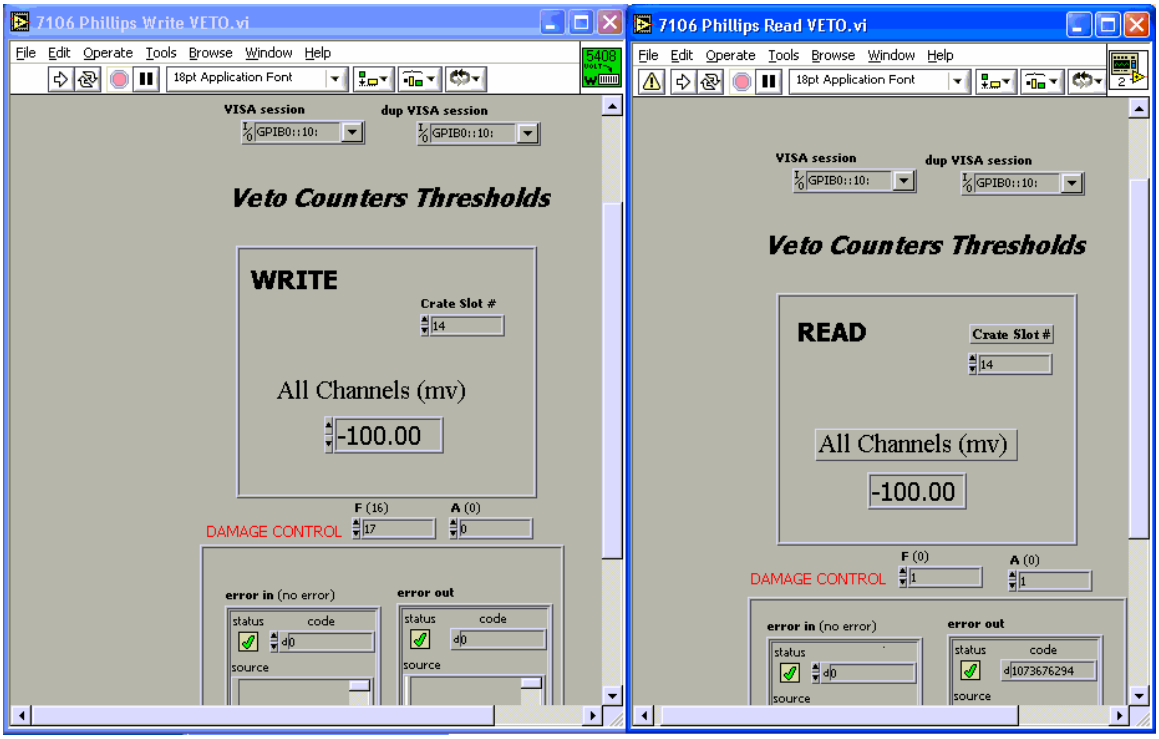
Always follow carefully the instructions/steps shown above to make MLU work properly. If you miss a step, chances are you have failed loading the files properly to the MLU. If so, stop, and start from the beginning.

Installation and Run of PS 16-Channel Discriminators

The device is a Phillips Scientific 16 Channel Discriminator Latch Model 7106. We have two of these devices and their manual.

The model 7106 is a high-performance, 16-channel leading edge discriminator packaged in a single width CAMAC module. The most commonly asked for features have been designed into this discriminator, making it extremely versatile instrument. The 7106 threshold is common for all 16 channels and is variable from –10mV to –1.033 V ±2% by either a front panel control when in local mode, or a 10-bit DAC via CAMAC when using in Remote mode.

Using Labview to set the 7106 thresholds.



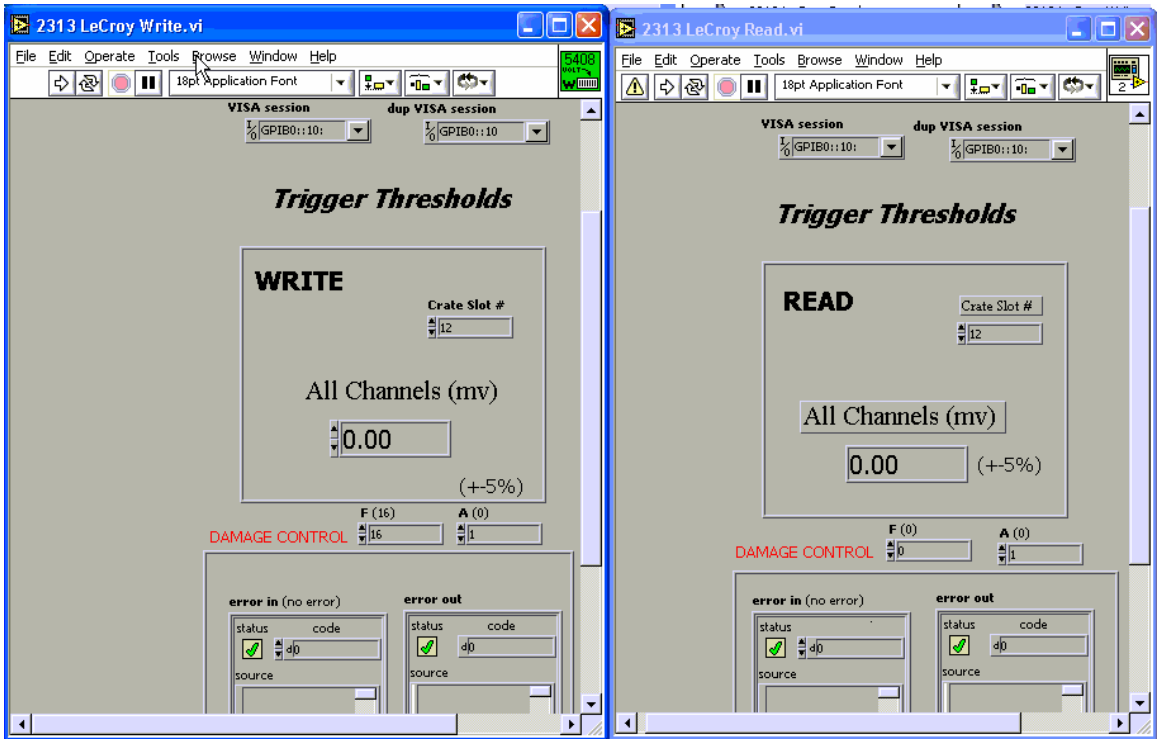
7106 Phillips Write VI uses the function codes and the translation bit equation mentioned in the manual and in an extra page document to set the device in Remote Mode and write the same threshold value to all 16 channels simultaneously through a the 10-bit DAC. **7106 Phillips Read VI** reads out the actual threshold level for all 16 channels from an internal 10-bit ADC. This is in the lowest 10 bits of the word.

Installation and Run of LeCroy 16 Channel Discriminators for Veto Counters

The device is a LeCroy Research Systems 16 Channel Discriminator Model 2313. We have two of these devices including their manual. Lecroy will be used to set the thresholds to the Veto Counter.

This device same as 7106 PS Discriminator sets a common threshold to all 16 channels. The voltage range for Lecroy 2313 is 0 to –1.023 V ±5%, and is adjusted either in local mode or via CAMAC through 12-bit DAC.

Using Labview to set Lecroy Thresholds.



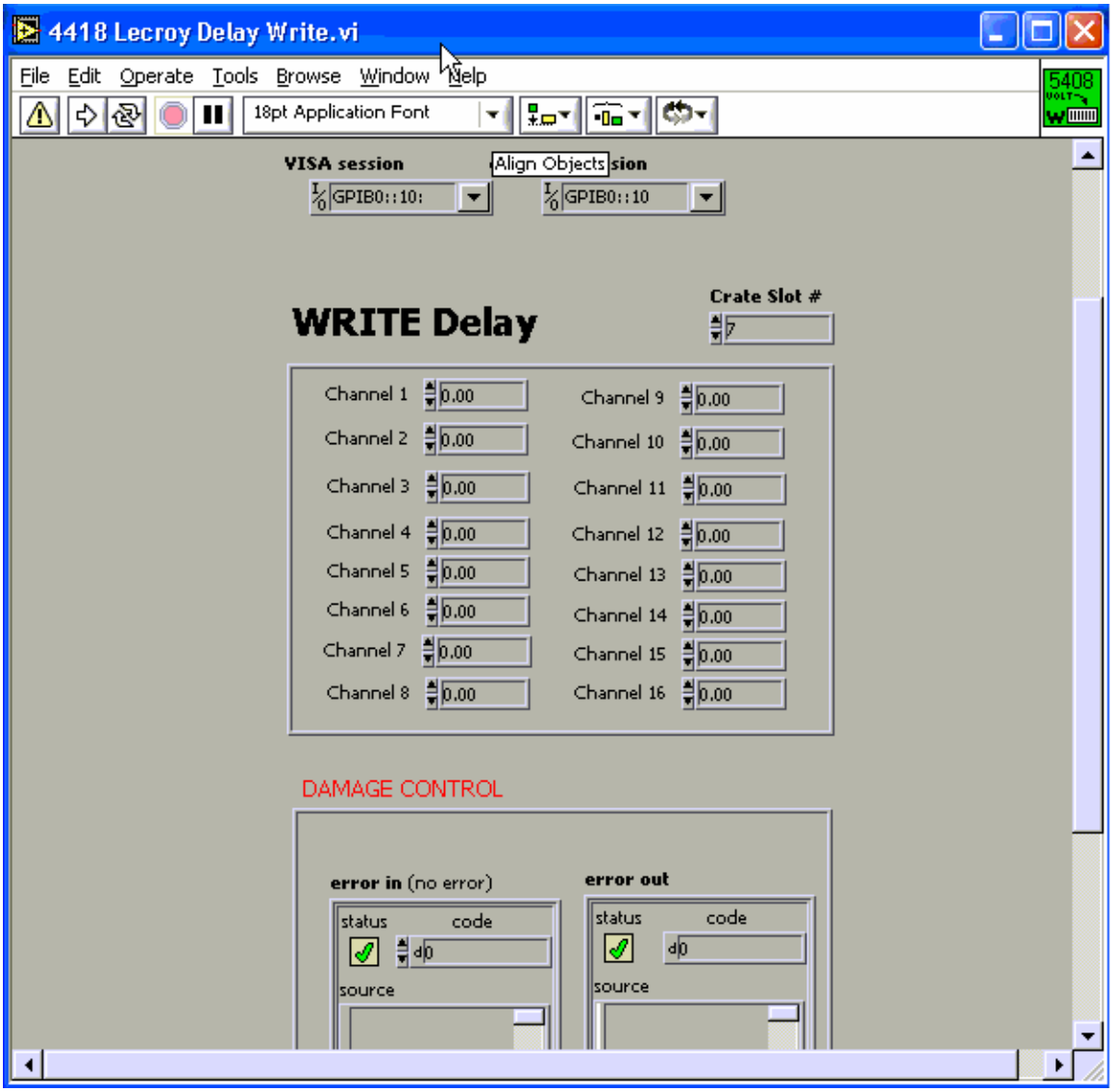
Veto 2313 Lecroy Write VI uses the function code and the translation bit equation to set the device into local mode and writes the threshold voltage to all 16 channels into the 12-bit DAC. **Veto 2313 Lecroy Read VI** reads back the threshold level.

Installation and run of LeCroy Delay Unit Model 4518

The device is a LeCroy Research Systems Delay Unit. We have two of these devices, one is spare, and the manual is included.

The 16 independent delay lines, each with a fan-out of three, are individually programmable via CAMAC in steps of two nanoseconds at a range 2-30 nanoseconds. Power supply backup ensures that delay settings are permanently retained even when CAMAC crate power goes down. Number of inputs is 16 all identical, and complementary outputs are three per channel.

Using Labview to set delay times.



4418 Lecroy Delay Write Vi writes the delay time to each channel of the 16-channel unit. Unfortunately this device does not offer a read back command.

Note: Refer often to the Labview Help/Explain error and device manuals to get a better understanding of how everything works and for troubleshooting.
The folder of Primex Trigger Control has all the software needed to run CAMAC including zip archives of particular programs:

1. **C542.zip**: A zip archive of LabView VIs for the control and operation of the CAEN Model C542 MLU.
2. **DAC 5408**: A zip archive of LabView VIs for the control and operation of the Bi Ra Model 5408 DAC.

3. **Ks3988.zip**: A zip archive containing a modified version of the sub-VIs that are used to interface with the Kinetic Systems 3988 CAMAC Crate Controller. All of the aforementioned VIs require this specific library to run. The KS3988 library is very similar to the one found on the LabView CD and the NI web-site, however some of the sub-VIs were changed so that they can work. The KS3988 sub-VIs use a protocol called VISA in order to communicate with GPIB devices.
4. **KS3988_old.zip**: An older version that performs identically to the KS3988 VI. The main difference is that these older VIs do NOT use VISA, meaning that they are guaranteed to work a majority of the time. They also work a good deal faster and use less computer resources since they do not require VISA.
5. **Read And Write VIs.zip**: A pair of extremely simple VIs that let you "manually" interface with the VIs. These VIs let you specify the CAMAC function codes (F() A() and N()) by hand and other VISA session settings. These VIs are extremely useful for seeing if you have successfully setup your GPIB network. If you cannot get these to work, forget about getting anything else to work since these two VIs are the soul of all other VIs.
6. **7106 Phillips Read and Write.Vi**: A pair of VIs to set the threshold for the Veto Counters.
7. **2313 Lecroy Read and Write.Vi:** A pair of other VIs to set the threshold for other uses.
8. **4418 Lecroy Delay Write.Vi:** Used to set time delays.