SRE3021

User Manual

Contents

1 Applicable Documents

[IDE-DS-SRE3021] SRE3021 Datasheet
[IDE-TB-M-BV0.26] IDEAS Testbench Software Reference Manual
[IDE-REP-Ref-V1.7] IDEAS Readout and Control Packet Protocol Reference

2 Revision History

Release Date	Revision	Author	Comment
2017-12-04	V0R0	Arne Fredriksen	Beta version of user manual created, based on beta firmware/software.
2018-02-27	V1R0	Tor M. Johansen	First release IDEAS Testbench under own heading Updated pixel map table Added configuration register table Added Gain Scan example
2019-04-09	V1R1	Tor M. Johansen	For version BETA V0.28 RC0 VAST

3 Introduction

The SRE3021, also referred to as VAST, is an OEM camera module for gamma and x-ray spectroscopy. It is designed to allow integration in end-user instruments and is delivered with a pixelated mono-crystal CZT sensor and readout electronics based on IDEAS' proprietary front-end IC technology.

The SRE3021 is designed to deliver information for reconstructing the 3-dimensional position of the events (photo-absorption, Compton scattering, etc.) occurring in the detector. The device is suited for users who intend to do Compton reconstruction of the detected events, as both the scatter and absorption events are detected by the same sensor. A block diagram is shown in Figure 1.

Multiple SRE3021 can be connected to build a synchronized system with a larger sensitive detector volume. Each camera module includes I/O interfaces for synchronization.

The event data is available over a standard Ethernet network interface. The detector system is delivered with the IDEAS Testbench testing software. APIs for C and LabVIEW can also be delivered together with the detector system – allowing the user to quickly make end-user applications.

This document describes how to install and operate the SRE3021 and is based on firmware version 1.2 and IDEAS Testbench Software BETA V0.28 RC0-VAST.

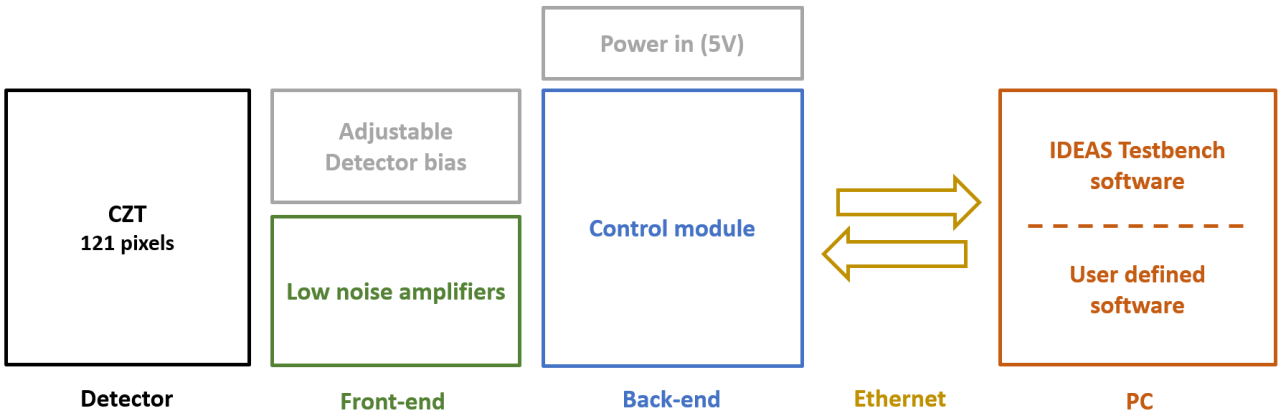


Figure 1: SRE3021 simplified block diagram.

4 Interfaces

4.1 Front-end Interface

The front-end interface is here defined as the detector interface. A photograph with of the SRE3021 with a superimposed pixel map is shown in Figure 2. The pixel map is here illustrated on top of the CZT crystal that is behind the detector cap. Please note the orientation of the X and Y axis, and that both axes are enumerated from 0 to 10. The 121 pixels are also referred to as the anode channels.

All channels that can be read out are listed in Table 1 and mapped toward the pixels described above. The X and Y coordinates are here in accordance with Figure 2, while the channel IDs correspond to the order of the channel readout, where channel 0 is read out first. The special channels are listed at the end of this table.

Figure 2: Pixel map orientation for SRE3021.

Table 1: Channel list with pixel map coordinates.

Channel	X	Y
0	2	3
1	2	0
2	1	2
3	0	0
4	0	3
5	3	0
6	1	4
7	0	1
8	2	4
9	3	1
10	0	5
11	1	0
12	0	2
13	4	1
14	1	1
15	3	2
16	2	2
17	4	0
18	1	3
19	5	0
20	0	4
21	5	1
22	3	3
23	4	2
24	2	1
25	6	0
26	7	1
27	7	0
28	10	0
29	6	1
30	9	0
31	8	0
32	8	1
33	10	1
34	9	1
35	8	2
36	7	2
37	9	2
38	6	2
39	10	2
40	5	2
41	8	3
42	7	3
43	9	3
44	4	3
45	10	3
46	6	3
47	8	4
48	5	3
49	9	4
50	7	4
51	10	4
52	5	4
53	7	5
54	6	4
55	8	5
56	4	4
57	10	5
58	6	5
59	9	5
60	5	5
61	7	6
62	6	6
63	10	6
64	4	5
65	9	6
66	5	6
67	8	6
68	4	6
69	9	7
70	6	7
71	10	7
72	4	7
73	10	8
74	5	7
75	8	7
76	3	7
77	7	7
78	6	8
79	9	8
80	4	8
81	9	9
82	5	8
83	10	9
84	3	8
85	10	10
86	6	9
87	8	9
88	8	10
89	7	8
90	7	10
91	7	9
92	9	10
93	8	8
94	6	10
95	5	10
96	4	10
97	4	9
98	1	7
99	5	9
100	3	6
101	1	5
102	3	9
103	3	4
104	3	10
105	1	6
106	0	9
107	2	8
108	2	9
109	0	7
110	2	10
111	0	10
112	1	10
113	2	5
114	0	8
115	0	6
116	1	8
117	3	5
118	1	9
119	2	6
120	2	7

Table 2: Pixel map with channel names.

y \ x	0	1	2	3	4	5	6	7	8	9	10
10	111	112	110	104	96	95	94	90	88	92	85
9	106	118	108	102	97	99	86	91	87	81	83
8	114	116	107	84	80	82	78	89	93	79	73
7	109	98	120	76	72	74	70	77	75	69	71
6	115	105	119	100	68	66	62	61	67	65	63
5	10	101	113	117	64	60	58	53	55	59	57
4	20	6	8	103	56	52	54	50	47	49	51
3	4	18	0	22	44	48	46	42	41	43	45
2	12	2	16	15	23	40	38	36	35	37	39
1	7	14	24	9	13	21	29	26	32	34	33
0	3	11	1	5	17	19	25	27	31	30	28

4.2 Back-end Interface

The back-end interface for the SRE3021 refers to all the connectors that are not associated with the detector. These connectors are pointed out in Figure 3, and described below.

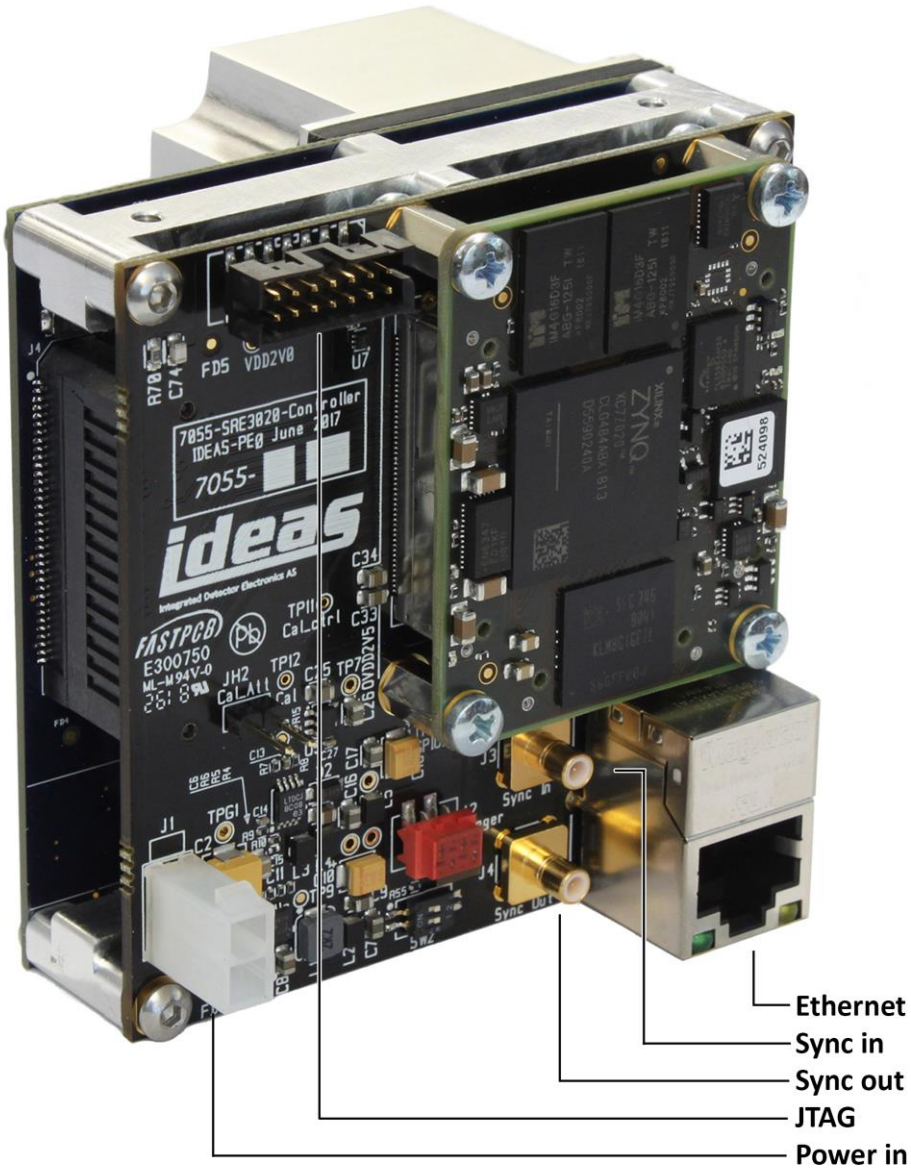


Figure 3: Back-end interface for SRE3021.

4.2.1 Power Supply

Power must be supplied to the module by a 2-pin MINI-FIT JR connector, with +5V and 0V like indicated in Figure 4. Note that the power supply quality affects the module performance. Use the switch mode power supply provided with the module for functional demonstration only.

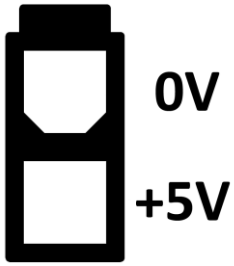


Figure 4: Power input connector.

4.2.2 Ethernet

The Ethernet is connected via a standard RJ45 Ethernet connector. Note that the Ethernet connection utilizes both the TCP and UDP protocol, and that the control computer must be set up appropriately. This is described further in Chapter 8.

4.2.3 JTAG

The JTAG interface is used for upgrading the module's firmware, which can be done with the provided JTAG programmer. The procedure for doing this is described in chapter 7 . Note that all modules are programmed with the latest firmware version available at the time of shipment.

4.2.4 Synchronization

There are two coaxial SMB connectors that can be used to synchronize the data acquisition with other SRE3021s or with external systems. The dip-switches are intended for setting the different modes for synchronization. This function requires a firmware upgrade.

4.2.5 Reset Button

The 'RESET' button on the module halts and then restarts the processor. Do not use this button while the High Voltage is turned on.

5 Functionality Overview

When the module is powered up, the Controller uses about 1.5 seconds to start up. The readout is event based. An event is either a trigger on an anode channel, or a pulse from the calibration generator. The control module will convert the amplitude of all anode signals and timing values and assemble an Ethernet packet.

5.1 Low Noise Amplifiers

The front-end is equipped with the IDEAS VASTAT4 ASIC. It is a 129-channel (5 special+123 normal+1 test) charge sensitive preamplifier-shaper for pulse amplitude-/time measurement with CZT Detectors. Dynamic range -80fC to +80fC. 300e ENC noise at 0.9us shaping time.
Serial data read-out and wire-or-ed trigger output.

5.2 Detector Bias

The SRE3021 can generate the cathode bias voltage for negative bias detectors. It is selectable via the system registers. The high voltage is limited to -2000V in software Detector Bias control. It is possible to set the bias higher (absolute value) by setting the control register in system configuration.

5.3 Calibration Pulse Generator

The SRE3021 has an onboard switch circuit with a selectable capacitor of 1pF, to allow testing with known charges.

5.4 Back-end Functionality

This section describes the functionality controlled by the back-end registers (system registers). These registers are divided into modules defined by the leftmost column in the system registers list. Below is a short explanation for how to operate these.

5.4.1 System Settings

These are registers for system information, such as serial numbers and firmware version.

5.4.2 Cal Pulse Generator Settings

Configures the properties of the calibration pulse generator. The polarity, number of pulses, length and interval can be set here. The pulse interval must be longer than the pulse length. If the number of pulses is set to 0, a continuous set of pulses is set, which can be cancelled by setting the number to another value, like for instance 1. Pulses with the configured settings are sent when a '1' is inputted to the 'Calibration execute' register.

5.4.3 DACs

Controls the voltage levels set by the DACs in the system. These are used to set biases for the VASTAT ASIC.

DAC	VTHR	VTHR0	VFP	VFP0	VRC	Cal	HV_ctrl	Gr_ctl
0	-1.84V	-1.84V	-1.84V	-1.84V	-1.84V	0V	0V	0V
2430	0V	0V	0V	0V	0V	1.48V	1.48V	1.48V
4095	1.27V	1.27V	1.27V	1.27V	1.27V	2.50V	2.50V	2.50V

5.4.4 VATA Readout

Configures the settings for the readout controller in the firmware. A trigger is received, and a readout of all channels is performed automatically. The output is transmitted via Ethernet.

6 IDEAS Testbench

This section gives a brief overview of the IDEAS Testbench, and how to operate it. A screenshot of the IDEAS Testbench is shown in Figure 6.

Detailed information about data formats and scripting functions is in IDEAS Testbench documentation. These documents are in the Testbench software folder:

- IDEAS-Readout_and_Control_Packet_Protocol-Reference.pdf
- IDEAS-Testbench-Software_Reference_Manual.pdf

6.1 Configuration Tab

In the configuration tab, both the back-end registers (also referred to as system registers) and the ASIC registers can be controlled and read. These are chosen by clicking on the text under the 'Units' area, like shown in Figure 5. The different register types are described below. Note that both register types can be written to by using Python scripting in the scripting window. The input for the register values are decimal numbers.

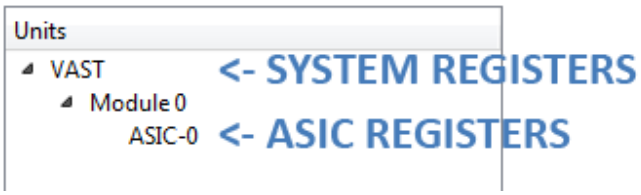


Figure 5: Choosing between viewing system/back-end registers and ASIC registers.

6.1.1 ASIC Registers

The ASIC Register settings are normally OK in the default state.

Name	Bit address	Description	Default Value
Channel 2 monitor enable	4	Selects the cathode channel to observe on the pre-amp monitor pin	0
Channel 0-4 monitor output enable	7	Turn on the pre-amp monitor	0
Current compensation enable	8	Reduces the effect of detector leakage current.	1
Test_on	9	Connects the calibration line to the preamp input.	0
Cathode Channel Disable	13	This disables the cathode channel from triggering. The amplifier is still on and is included in the read-out.	
Anode Chanel # Disable	16 to 138	This disables the channel from triggering. The amplifier is still on and is included in the read-out.	0

6.1.2 System Registers

Name	Description	Default Value
cfg_hold_dly	Programmable delay between the trigger signal and the Hold signal. The VASTAT ASIC holds the analog amplifier values as well as the analog timing values until they are read out.	142
calibration execute	starts the pulser	
calibration Num Pulses	If this value is '0' the calibration pulse will repeat until a different value is set in this register.	1
VTHR	Sets the threshold level for anode channels to generate trigger pulse. Applicable values 0-2430. 2400 represents ~150keV	2400
VTHR0	Sets the threshold level for cathode channel to generate trigger pulse. Applicable values 2430-4095 2450 represents ~200keV	2450
VFP	Preamp feedback resistance for anode channels.	1800
VFP0	Preamp feedback resistance for the cathode channel. Use this to stabilize the cathode signal when changing the detector bias.	1750
HV DAC	100 DAC step = -100V	2500
Grid DAC	T.B.D – Setting this value to max will set the detector anode guard pin to 0V	4095
Cal DAC	Sets the amplitude of the pulse used in the calibration circuit	1000
cmd_set_ch	enables the channel set in cfg_fixed_ch	
cfg_phystrig_en	enables/disables the VASTAT signals from starting a readout.	1
cfg_caltrig_en	When this bit is set, a readout will always start in coincidence with the calibration pulse.	0

6.2 Readout Data Tab

This tab shows the raw readout data coming from the module. You can view the data in different ways by using the sub-tabs for event data (latest event), histogram (accumulated) and trigger counts. It is possible to right-click the plot to adjust the axis or select different curve styles. A screenshot of the readout data tab is shown in Figure 6, where the sub-tab for histogram is selected.

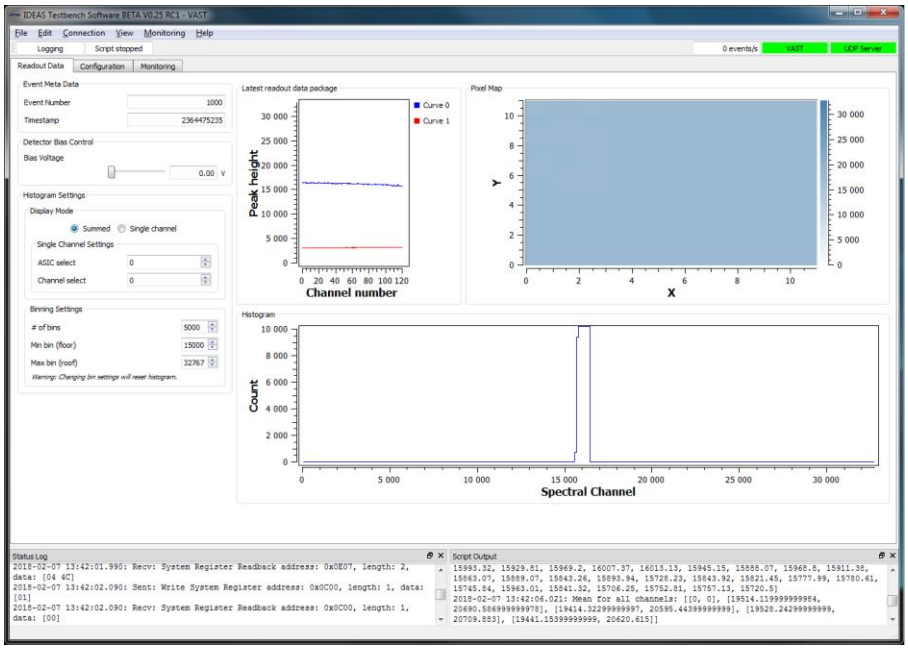


Figure 6: IDEAS Testbench readout data tab screenshot.

System registers (back-end registers)

These are the registers used to control firmware settings and board settings. They are used to perform tasks such as generating calibration pulses, setting DACs and controlling readout modules. You change the settings by writing in the "Current Value" column, and the registers are updated as soon as you press Enter or leave the currently selected cell.

ASIC registers (front-end registers)

ASIC registers are written to by entering values into the “New Value” column. Unlike the system registers, these are not loaded when Enter is pressed. To configure the ASIC, the “Send Config” button in the top right corner, shown in Figure 7, must be pressed. This programs the ASIC with the values listed in the “New Value” column, while the readback from the ASIC will appear in the “Current Value” column.

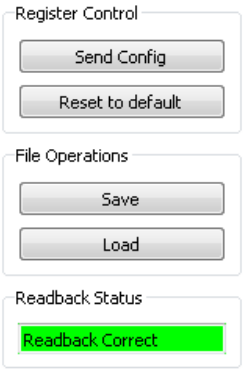


Figure 7: ASIC configuration control.

Note: The readback status will flash red if a new configuration is sent, because the new values are compared with the previously stored values in the ASIC. To verify that the new values have been

programmed correctly, the configurations should be sent twice, causing the readback status field to become green again with the text "Readback Correct".

6.3 Script Window

Scripting (View – Script Window) is a time-saver when it comes to setting system registers and ASIC registers. It can be used to control system registers, ASIC configuration, and getting readout data. IDEAS provides some sample scripts to perform simple functions. By studying these and the scripting reference document, the users have the tools to create elaborate scripts customized for their own needs. The scripting in IDEAS Testbench is done with the Python programming language. A screenshot of the script window can be seen in Figure 8.

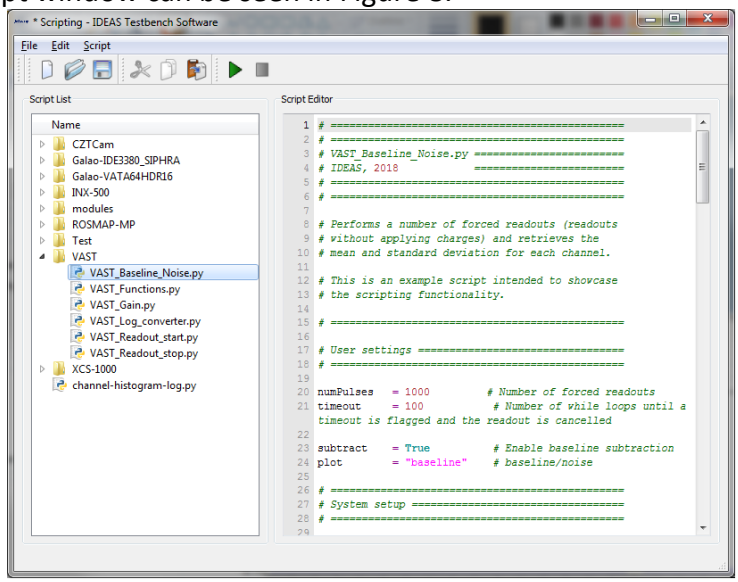


Figure 8: Script window.

The scripts must be located in the scripts/VAST folder within the IDEAS Testbench folder. They will then show up in the left area of the screen and can be loaded by double-clicking. One can edit the scripts in the built-in script editor, or in a separate editor of choice. Prints appear in the "Script Output" tab located at the bottom of the software window. Note that the script output can be resized and dragged outside the main window, or alternatively to the side of the "Status Log" window. This is done by pressing and dragging the gray bar with the tab name.

Note: Functions and variables are shared between the scripts, as long as the script holding the definitions are executed before the functions and variables are called upon. Hence, in the case of our sample scripts, the functions.py must be executed first in order to load the functions, since it contains functions used by other scripts.

6.4 ADC Value Format

The ADC values are 15 bit, including sign-bit. One unit at the ADC corresponds to approximately 1.09E-16C charge at the ASIC charge amplifier input.

7 Firmware Upgrade

To program the firmware, one must connect the provided JTAG programmer to the JTAG connector. After the programmer is connected, tone can proceed with the programming as described below. A software that can program the flash of a Zynq chip is needed – we recommend the free iMPACT software from Xilinx.

1. Download the .mcs file containing the desired firmware version to a computer. It should be saved locally on the computer, and not result in a long path-name, as this can cause iMPACT to fail when attempting to program.
2. Power up the system.
3. Start iMPACT.
4. Press ‘No’ when asked about automatically loading the last saved project.
5. Press ‘Yes’ when asked about automatically creating a project file.
6. Press ‘OK’ to configure devices using boundary-scan.
7. Press ‘No’ when asked about continuing and assigning configuration files.
8. Press ‘Cancel’ in the Device Programming Properties window.
9. Right click ‘SPI/BPI’ as seen in the picture below, and press ‘Add SPI/BPI Flash…’
10. Select the provided .mcs file for the target system.
11. Use QSPI in Single mode and press ‘OK’.
12. Right click the ‘FLASH’ box in the graphical view and press ‘Erase’. A blue box with ‘Erase Succeeded’ should show up.
13. Right click the ‘FLASH’ box in the graphical view and press ‘Program’. After progressing, ‘Program Succeeded’ should show up.
14. Power down the system.

After the programming is successful, the chosen firmware version is stored on the module, and will be stored until the module is programmed again.

8 Getting Started

This is a short guide giving an overview of how get started setting up and using the SRE3021. This section shows how to connect the module to a PC and presents details on how to take use of the various functionality in the module with the back-end configuration registers.
It is also possible to use the SRE3021 with custom readout software instead of the IDEAS Testbench. This requires the user to make use of the API provided by IDEAS, and it is not a procedure that is covered by this user manual.
In this guide there are several checkpoints that look like this:

CHECK

Text indicating a condition that should be checked before proceeding.

If such a check fails, something may have gone wrong in the setup. Should this happen, please contact IDEAS for support.

8.1 Requirements

For this system to function properly, the hardware and the PC software must be set up correctly. Please follow the steps below to set the system up.

8.1.1 Hardware

The only required hardware steps are:

1. Connect an Ethernet cable to the SRE3021 and to a PC.
2. Connect power to the SRE3021s power connector. Make sure that the polarity is correct.

CHECK

A LED on the back-end side of the SRE3021 starts flashing red, and then turns off after about 1 second.

8.1.2 Network Setup

The SRE3021's IP address is 10.10.0.50. To connect, the PC should be configured to have the IP 10.10.0.100, like shown in Figure 9.

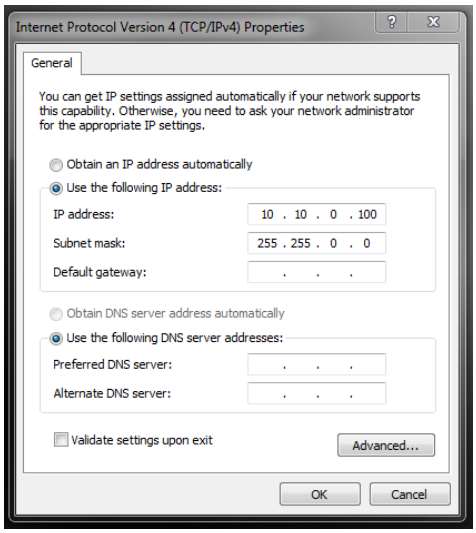
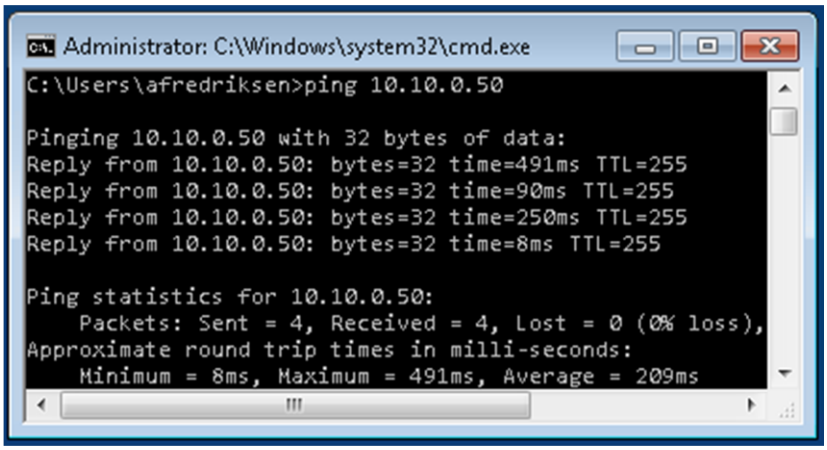


Figure 9: Network setup.

CHECK

Open Command Prompt (cmd.exe) and write: 'ping 10.10.0.50'. This shall result in an answer with an equal amount of sent and received packages, like shown in the screenshot below.



8.1.3 Python Install

IDEAS Testbench uses Python as a scripting language for running batch commands and creating custom control sequences. For this purpose, Python must be installed on the computer running IDEAS Testbench. The python support libraries used when building Testbench require compatibility with Python version 2.7.9 (32-bit). It is therefore strongly recommended that this version is used.

Python 2.7.9 32-bit can be found at the Python download page: https://www.python.org/downloads/

8.1.4 Firewall Configuration/Disable

Since IDEAS Testbench communicates over Ethernet, there may be issues if the firewall on the PC is activated. It is required that your firewall allows data on UDP port 50011, or alternatively that the firewall is disabled.

If you are using Windows, the “Windows Firewall” may block data on UDP port 50011 by default. Make sure that both any installed third-party firewall is deactivated and that the Windows Firewall is deactivated. Please note that Windows Firewall may be automatically activated when you deactivate your third-party firewall.

8.1.5 IDEAS Testbench Startup

The IDEAS Testbench folder can be added anywhere on your PC’s local storage, and it does not require an installation. It is started through batch files (.bat) in the root of the folder. The relevant batch file is **IDEASTestbench VAST.bat**.

8.2 Example: Baseline Readouts

To verify that all system components are up and running properly, and to give insight in how one may use the SRE3021, this section will walk you through some steps that demonstrate how to take use of some of the functionality of the SRE3021 and the IDEAS Testbench. It uses IDEAS' sample scripts to obtain forced readouts of the module, resulting in the baseline pedestal of each pixel. Insight into what is done to start the acquisition can be learned by studying the scripts used.

The scripts used here shows only one way of doing this, and are meant as examples only. They are not necessarily best practice on how to perform these tasks.

It is important that the steps in this small guide are carried out in the same sequence as they are written here.

1. Power up the module.
2. Open IDEAS Testbench by executing the VAST batch file.

CHECK
The fields 'VAST' and 'UDP Server' in the top right corner shall be green, indicating that the system is connected.

3. Go to the script window by pressing View – Script Window, open the VAST_Functions.py script and press execute.

CHECK
The script output tab in the bottom of the IDEAS Testbench main window shall display the text 'Functions loaded.'.

4. Open the VAST_Baseline_Noise.py script and press execute.

CHECK
The event rate in the top right corner of IDEAS Testbench shall go up to about 1000 events/s, and readout data shall appear in the Event Data window like shown in *Figure 10.*

When the baseline acquisition is done after 10 000 events are recorded, the means and standard deviations of every channel is recorded and listed in the Script Output window.

The data is stored in the same order as it is shown in the Event Data window during readout. The mapping towards the pixels is listed in Table 1. However, note that the events in Event Data are twice as many as the pixels. This is because the timing value for each channel is stored in the neighboring

channel. Hence, the data for channel 29 is in testbench shown in channel 29*2 = 58, while the timing data is shown in channel (29*2 + 1) = 59.

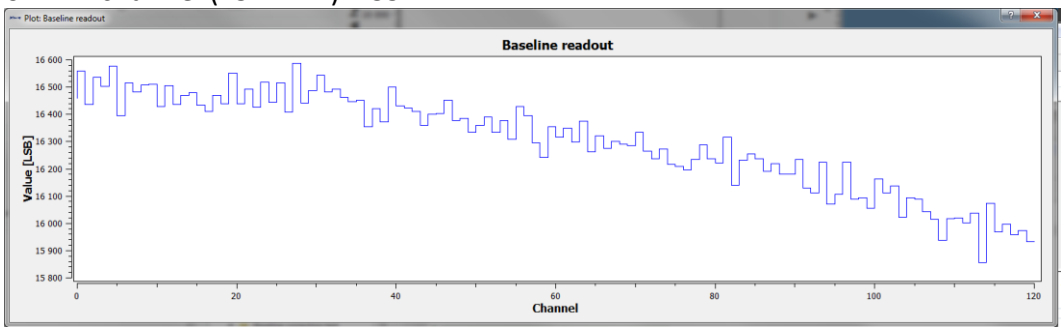


Figure 10: Baseline readout with beta software/firmware.

8.3 Example: Gain Scan

It is important that the steps in this small guide are carried out in the same sequence as they are written here.

1. Power up the module.
2. Open IDEAS Testbench by executing the VAST batch file.

CHECK
The fields 'VAST' and 'UDP Server' in the top right corner shall be green, indicating that the system is connected.

3. Go to the script window by pressing View – Script Window, open the VAST_Functions.py script and press execute.

CHECK
The script output tab in the bottom of the IDEAS Testbench main window shall display the text 'Functions loaded.'.

4. Open the VAST_Gain.py script and press execute.
5. After a few seconds the plot showing gain of all channels shall appear

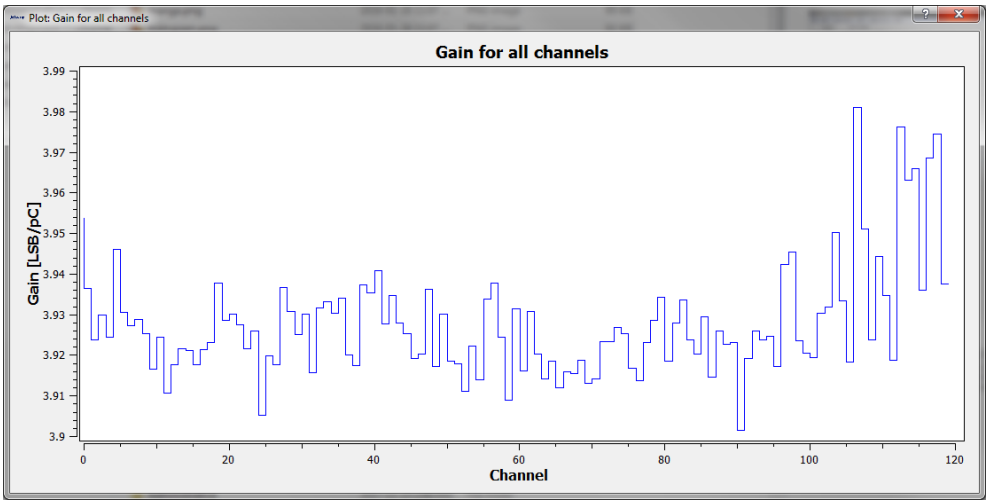


Figure 11: Gain Scan

8.4 Example: Readouts with a Radioactive Source

This small guide will show you one way of setting the module up for reading out a source, using one channel. It is done with only one channel because on-line baseline extraction is not yet implemented. It is possible to activate more channels, but this can lead to poor resolution because of the different channel baseline levels. Insight into what is done to start the acquisition can be learned by studying the scripts used.

The scripts used here shows only one way of doing this and are meant as examples only. They are not necessarily best practice on how to perform these tasks.

It is important that the steps in this small guide are carried out in the same sequence as they are written here.

6. Power up the module.
7. Open IDEAS Testbench by executing the VAST batch file.

CHECK
The fields 'VAST' and 'UDP Server' in the top right corner shall be green, indicating that the system is connected.

8. Go to the script window by pressing View – Script Window, open the VAST_Functions.py script and press execute.

CHECK
The script output tab in the bottom of the IDEAS Testbench main window shall display the text 'Functions loaded.'.

9. Open the VAST_StartUp.py script and press execute. The module is now ready to acquire data, and a radioactive test source can be placed in front of it.
10. Go to the Histogram tab in IDEAS Testbench.
11. Press the “Single channel histogram” button to view the histogram from one channel.
12. In the “Channel select” field, enter the value 58, since the only activated channel is channel 29 (58/2, as explained in the previous example).
13. Right click the histogram area and auto-scale the x-axis and the y-axis.

The accumulated histogram for *all channels* is shown in Figure 12.

Note that this is only a “raw” histogram taken without any correction efforts such as baseline extraction. Also note that it is possible to retrieve the histogram data by using the appropriate histogram functions in the scripts. This can be useful if one wishes to work further on the data after acquisition.

After using the module with high voltage enabled, we recommend disabling the high voltage with the script function “tb_product.set_HV(0)” before turning off the module. This function turns the high voltage of gradually. This can be done by simply entering this function into a script and pressing execute.

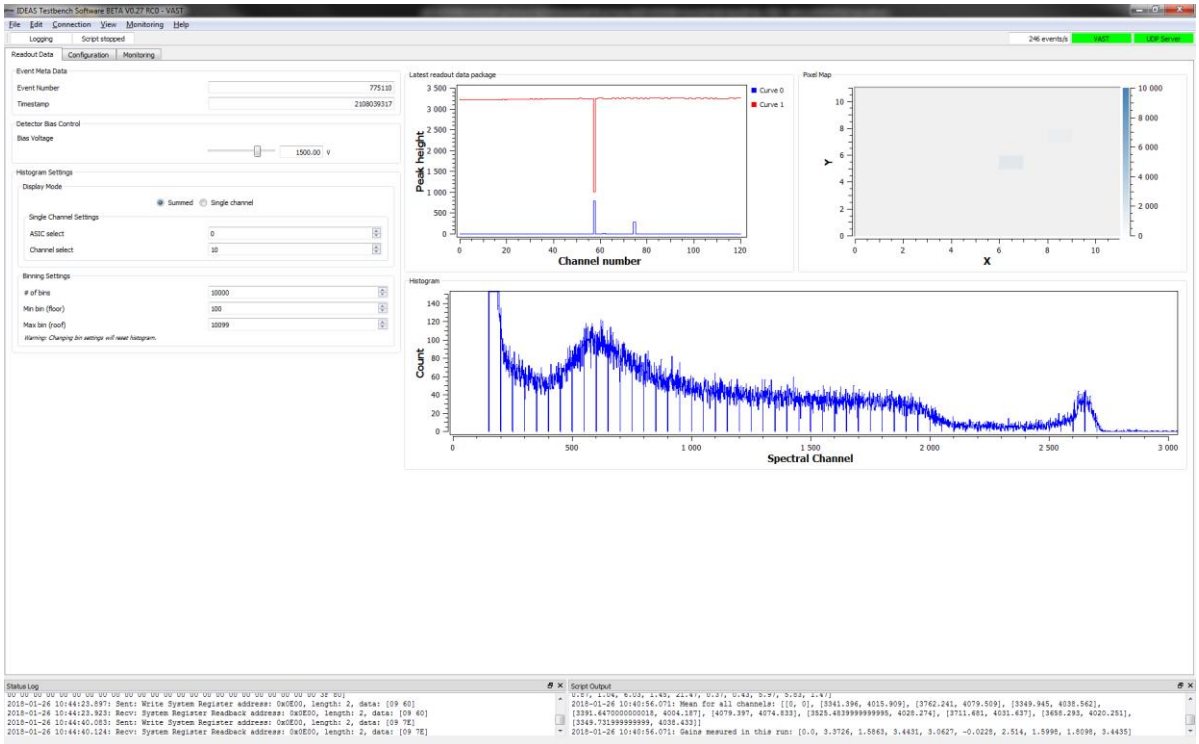


Figure 12: Readout in progress of Mn-54.

Appendix: Cooling Interface

One can only achieve the best performance by utilizing cooling, since the sensors operate optimally in the temperature range +20°C to +25°C.

We recommend using thermoelectric cooling by applying a Peltier device to this interface, attached with two screws. The dimensions of this interface are shown in Figure 13. Note that this component is electrically connected to 0V, the same potential as the chassis.

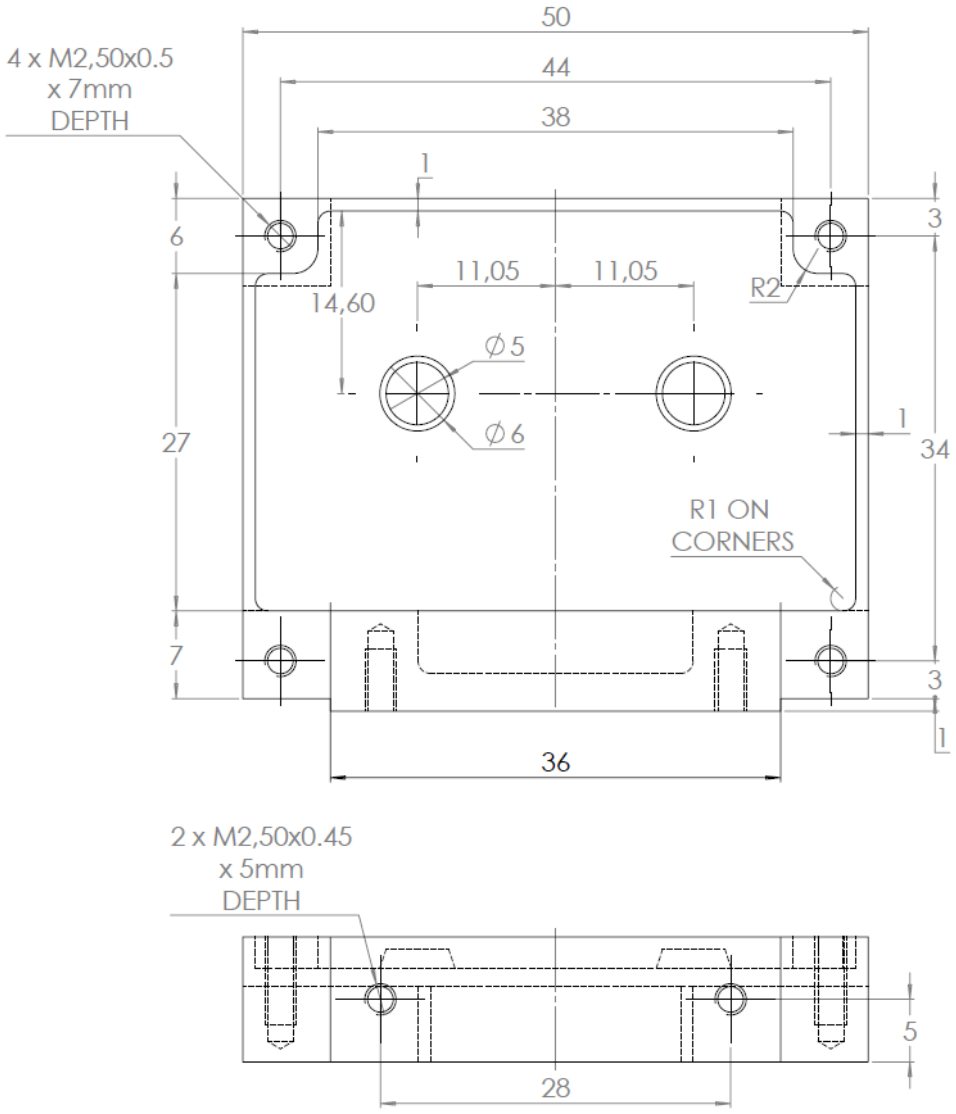


Figure 13: Cooling interface dimensions.