Neuromorphic Auditory Sensor User Manual

The Neuromorphic Auditory Sensor (NAS) has been developed by Robotics and Technology of Computers (RTC) Lab. in the University of Seville, Seville (Spain).

This user manual has the aim to introduce the user about what NAS is, what hardware/software components/tools does it need and how to configure and use the entire system. Furthermore, it is also indicated the software which can be useful to analyze the neuromorphic sensor output events.

More information about NAS, AER protocol, related hardware and software development, and neuromorphic stuff research papers can be found in the References section.

NAS user manual, version 1.0. May 2018, by RTC group.

Contact:

Angel Jimenez-Fernandez, ajimenez@atc.us.es
Antonio Rios-Navarro, arios@atc.us.es
Juan Pedro Dominguez-Morales, jpdominguez@atc.us.es
Daniel Gutierrez-Galan, dgutierrez@atc.us.es
Alejandro Linares-Barranco, alinares@atc.us.es

robotics &
computer technology lab.

Index

Hardware setup

This section resume, in a quick view, the hardware components needed to use the Neuromorphic Auditory Sensor. First, the following list show the name of each part, and in the next paragraph, a brief description of each one.

Components

2x USB Type A/B cable*.

1x Ribbon cable 20 way*.

1x AERNode board.

1x OKAERtool expansion board for AERNode.

1x Opal Kelly board.

1x PDMicrophone/Line-in board.

1x Audio stereo cable*.

1x Xilinx JTAG.

1x Power Supply 5V 4A.

***Component not included**.

AERNode board

Communication platform for AER systems. Its two PAER ports allow to connect an IN and an OUT AER system. Different AER protocols are supported. PAER to Serial-AER through LVDS SATA with bidirectional ports. Powered with 5V. It has Flash memory for FPGA programming support. The FPGA chip mounted is a Xilinx Spartan-6.

In our case, it will be used to implement the Neuromorphic Auditory Sensor as neuromorphic sensor, making use of the AER OUT interface.

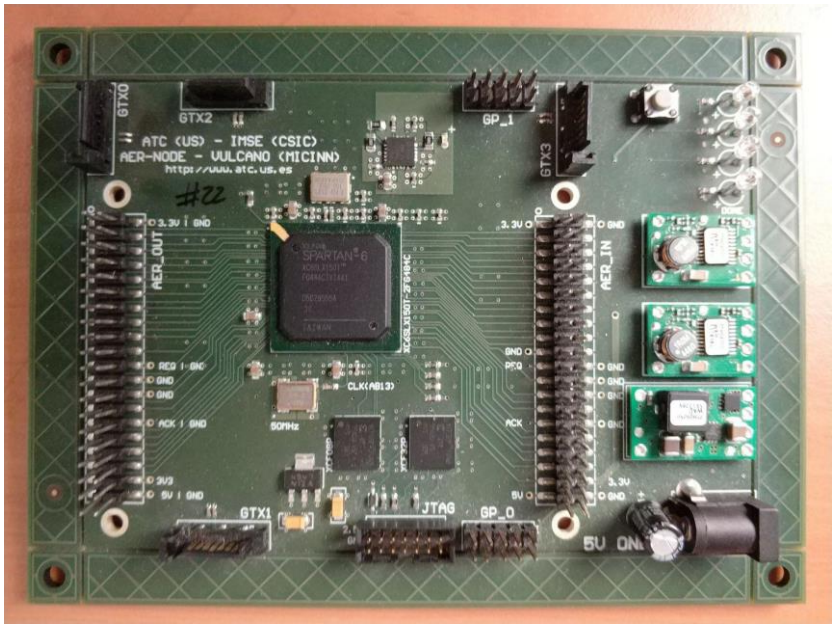


Figure 1: AERNode board.

Power Supply 5V 4A

To power up AERNode. The voltage input range should be 100-240 V, 50-60 Hz and 0.8 A, and the output must be 5V 4A. AERNode voltage regulator has an input range of XX V and XX A. **Be careful when a power supply will be selected. A bad selection could cause a malfunction or even damage the AERNode board.**

Figure 2: AERNode board power supply.

OKAERtool expansion board for AERNode

Daughter board of the AERNode for AER systems monitoring and debugging. It is able to merge, monitor, sequence, log and play AER events both to/from USB and to/from on board DDR2 memory. It is based on the OpalKelly XEM 6010.

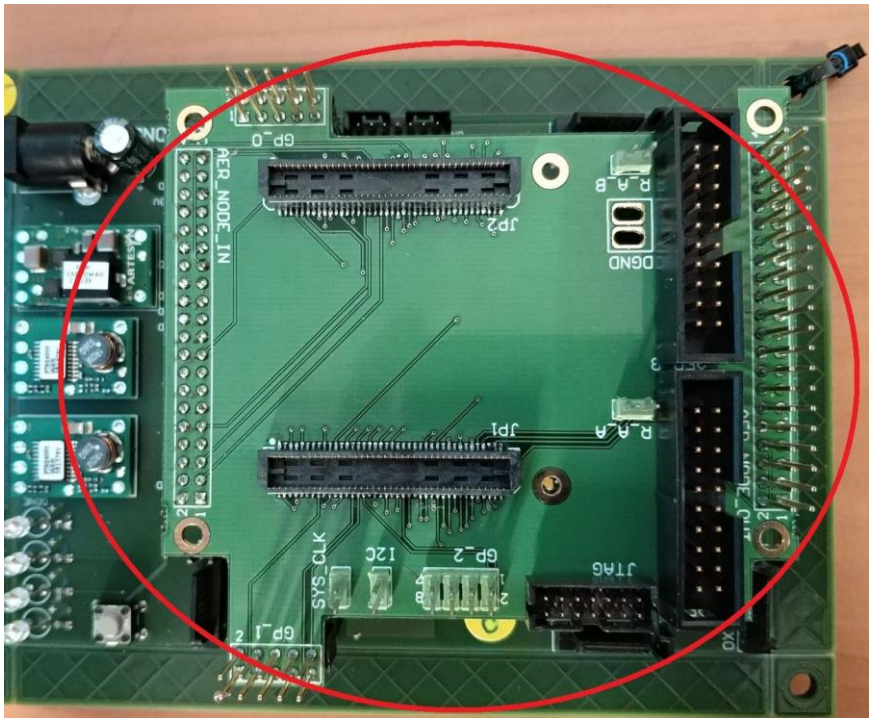


Figure 3: OKAERTool expansion board.

Opal Kelly board

Opal Kelly XEM 6010 (150LXT)board, which has a Spartan-6 FPGA chip. **It can be also powered up by the same power supply device selected for AERNode board**.

Figure 4: OpalKelly board.

Xilinx JTAG

A JTAG is a device which allows to program a FPGA. It also makes possible to debug a design when it is under development. This Xilinx JTAG uses the USB connection for both powering up and communicating the FPGA with the computer.

Figure 5: Xilinx JTAG device.

USB type A/B cable

This cable is used to:

-Communicate to the Opall Kelly board.

-Connect and power the JTAG.

Figure 6: USB type A/B cable.

PDMicrohpone/Line-In board

For mono/stereo audio application on the NAS sensor for the AERNode board, this expansion PCB is needed. It provides two pulse-density modulation (PDM) microphones and an audio codec (96KHz) for line-in audio input. User can select the input source by means of a jumper.

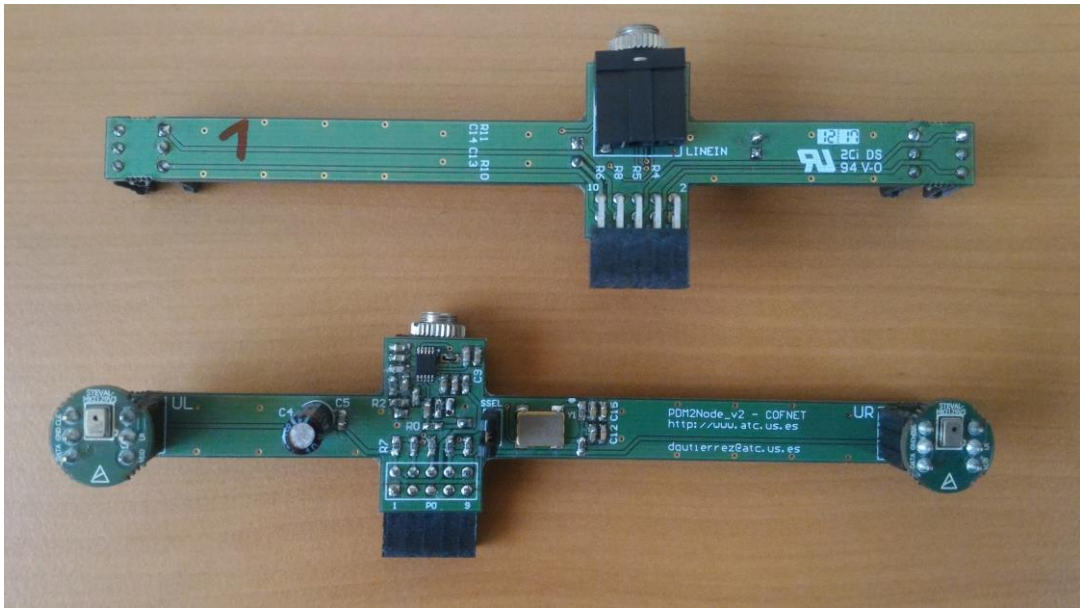


Figure 7: PDM microphones/Line-in audio input board.

Audio stereo cable.

The audio cable is used to connect the PDMicrophone/Line-in board to an audio source. It could be mono (for only one “ear” applications), but it is recommendable to use a stereo cable, even if the audio is mono.

Figure 8: stereo audio cable 3.5mm.

Ribbon cable 20 way

The OKAERTool has two 20-pin ports, which can be used to connect the AERNode board to other platforms that supports AER protocol that send/receive AER events. In that case, the ribbon cable should be used.

Figure 9: ribbon cable 20 way.

How to assemble

-Step 0:

Get the AERNode board. **Make sure that it is no powered up. If it is, please disconnect the power supply to avoid future problems.**

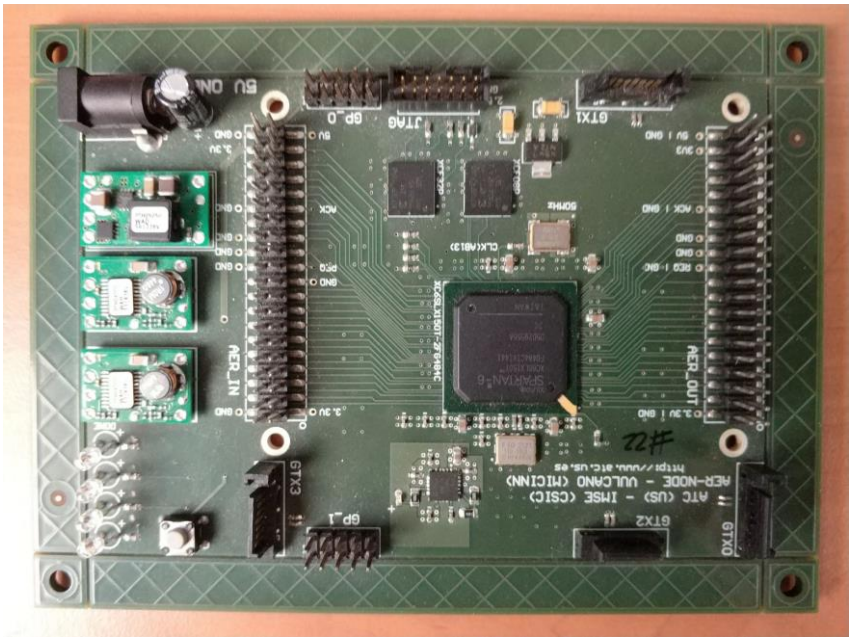


Figure 10: step 0, get AERNode board.

-Step 1:

Get the OKAERtool expansion board without the OpalKelly board, and connect it as it is shown in the following picture. The AER_NODE_IN, AER_NODE_OUT, GP_0 and GP_1 connectors of OKAERtool have to be connected properly with the AERNode board AER_IN, AER_OUT, GP_0 and GP_1 ports, respectively. Make sure that both boards are correctly assembled making the right pressure.

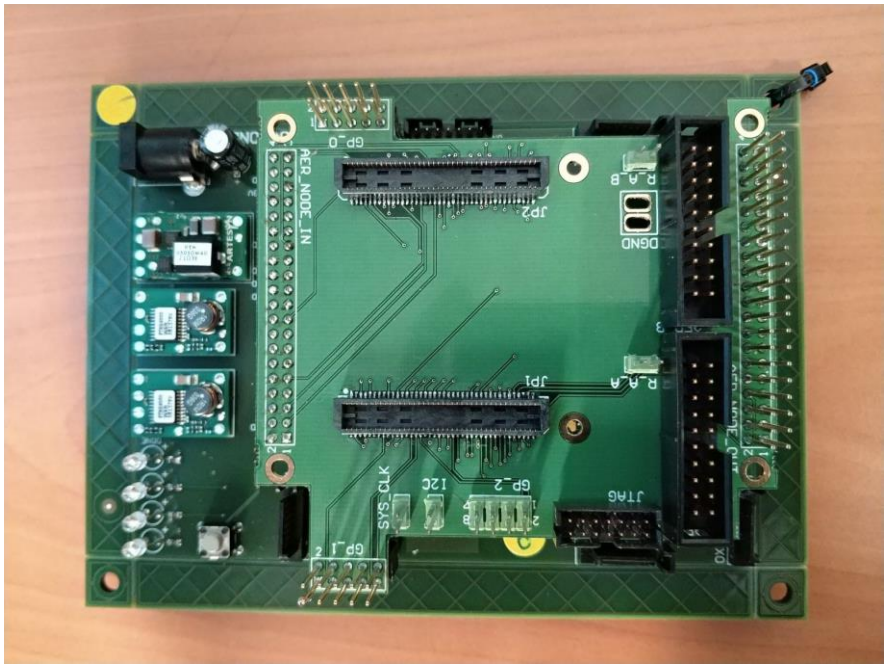


Figure 11: step 1, get OKAERTool board.

-Step 2:

Put OpalKelly board on OKAERtool central connectors, in the same way that you can see below. Note that OpalKelly's connectors (USB and power) position are in the same side that AERNode board (on the left side of the image).

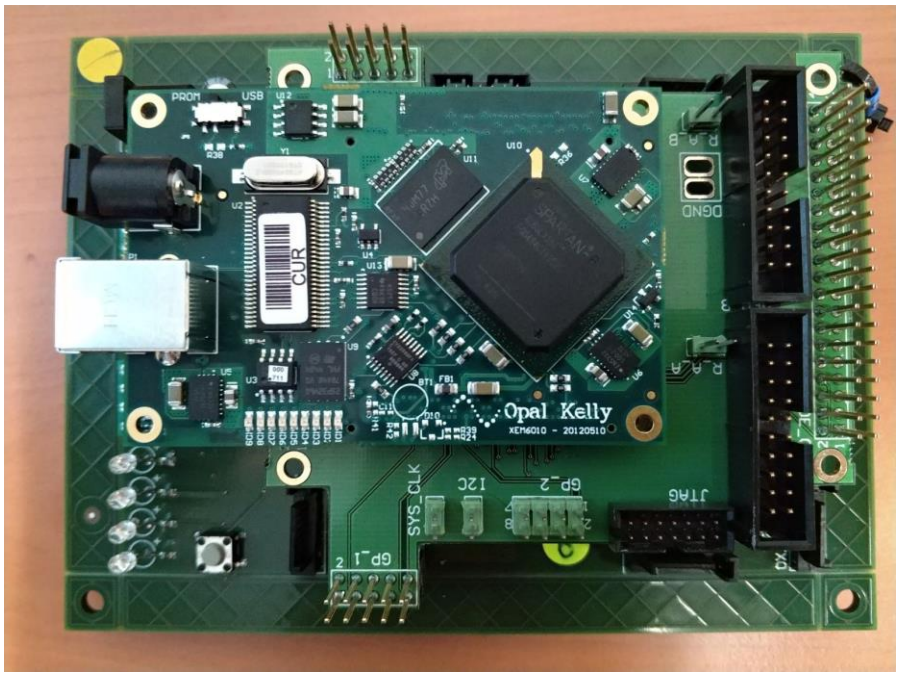


Figure 12: step 2, mount OpalKelly on OKAERTool board.

*COBER (http://t-cober.es/es/home.html) supplies this setup mounted.

-Step 3:

Assemble the PDMicrophones board on GP_1 port of OKAERtool board. The microphones should be facing outside de board.

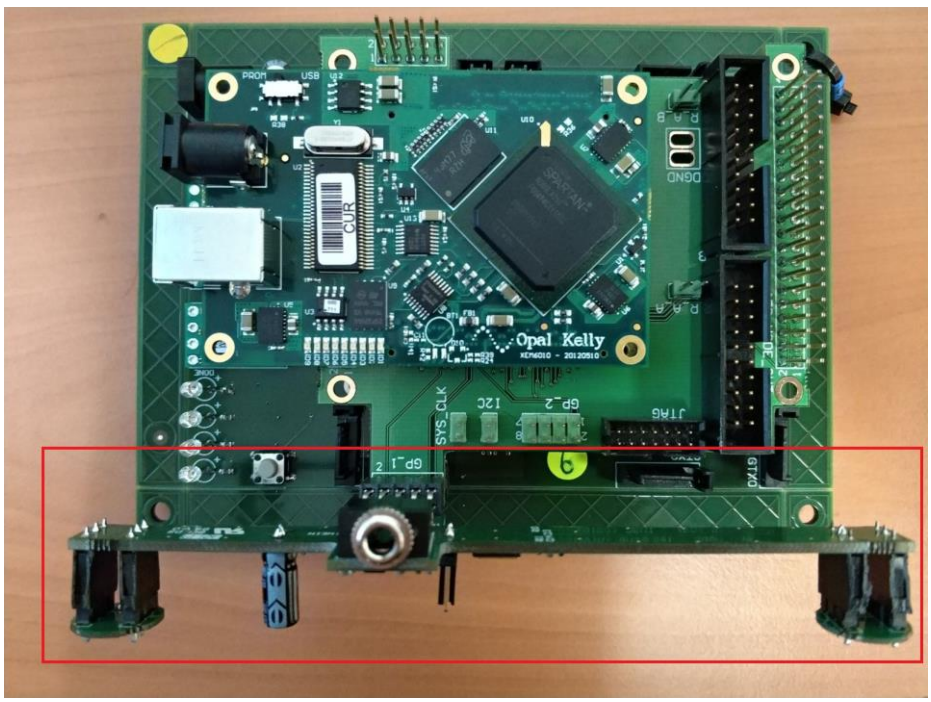


Figure 13: step 3, assemble PDM microphones/Line-in audio input board.

-Step 4:

Connect the audio cable into the jack 3.5mm connector. Important note: check the microphone position. A wrong position may damage the microphones: **STMicroelectronics logo should be at the bottom**.

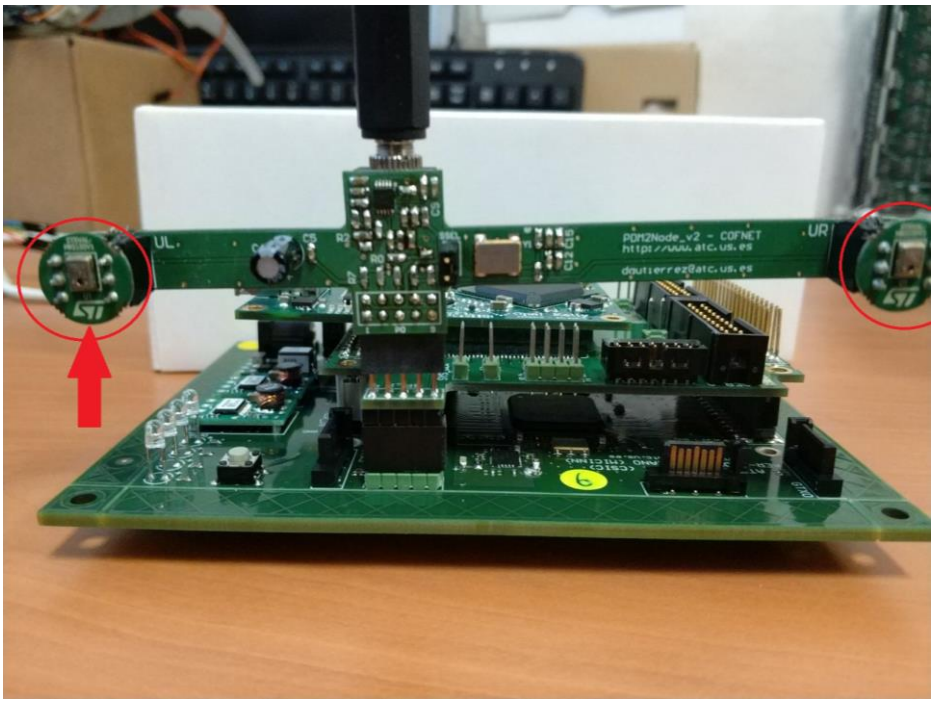


Figure 14: step 4, connect audio cable and check PDM microphones.

-Step 5:

Once all boards are assembled, it is time of connect first the USB from OpalKelly to the computer (to receive AER events and to be able to visualize them in JAER software) and finally power up the AERNode board. **It is possible to power either the AERNode board or the Opal Kelly, but power only one**.

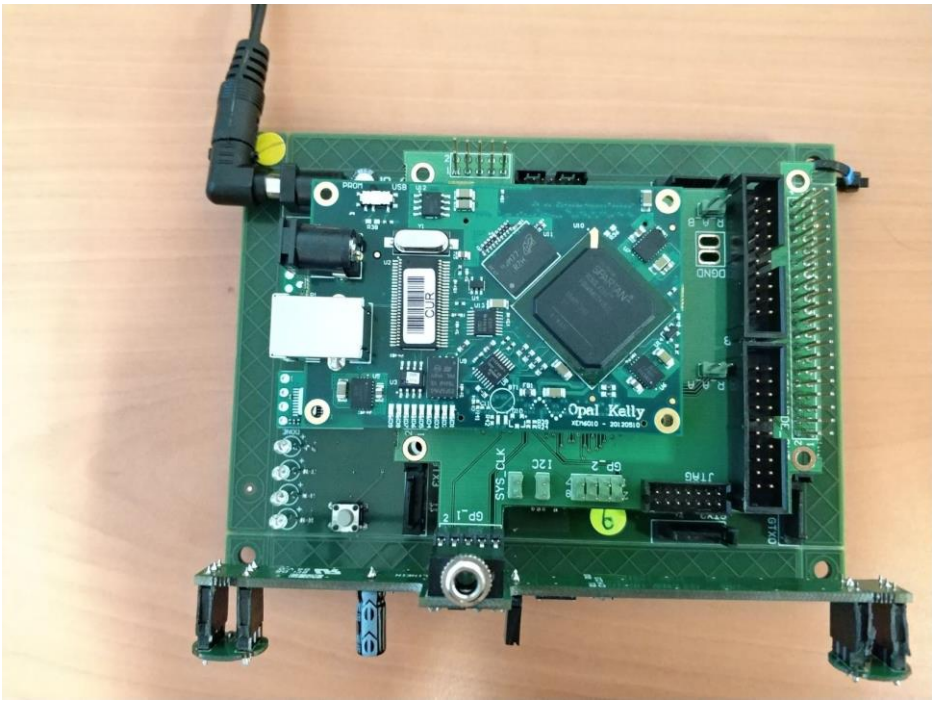


Figure 15: step 5, power up AERNode board.

-Step 6:

At this moment, assemble process is done! Output events could be visualized using JAER (it will be explained later). AERNode board is provided with a NAS design loaded on the flash memory*. However, if another design would like to be loaded, **the Xilinx JTAG (Downloading cable) must be connected to JTAG port of AERNode board**. OpalKelly board JTAG port is for programming itself, not for programming AERNode board. Xilinx USB downloading cable must be used. It has two connectivity options as shown below. If the OpalKelly plug-in board is mounted and connected you must use the 10-pin cable, plus the small PCB adapter to 14-pin AERNode PCB connector.

Figure 16: step 6, load a .bit file.

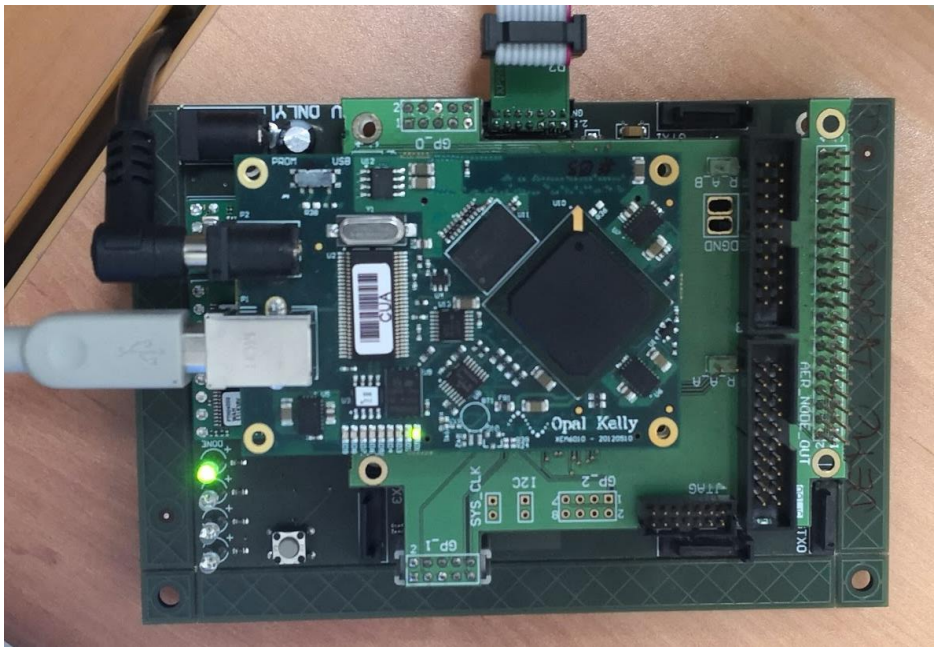


Figure 17: final hardware setup should be like this.

*By default, stereo NAS with 64 channels, input source PDM microphones or line-in (can be selected by the user with the jumper), cascade architecture, output AER events through USB (filter's output attenuation -15 dB)

Software tools

This section indicates what software do you need to install to work with the NAS: configuring FPGA, developing new designs and analyzing the output data.

Xilinx iMPACT

For FPGA programming. Download and install Xilinx ISE Design Suite from:

https://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/design-tools/14_7-windows.html

Once the installation process is done, you should have both ISE and iMPACT (among others). It allows to program the FPGA.

JAER

For AER output events visualization and recording through Opall Kelly (OKAERtool). Install JAER from:

https://github.com/RTC-research-group/jaer

Follow the user guide.

NAVIS

For AER output events analysis from JAER recordings. Download and run NAVIS tool from:

https://github.com/jpdominguez/NAVIS-Tool

Follow the user guide.

NAS .bit files

What is it?

This kind of file is the result of performing the complete design flow using FPGA. When the hardware engineer has the VHDL design finished, the synthesis, implement and generation steps have to be carried out. As a final result, the tool generates a bitstream file, which can be loaded on the FPGA.

This file contains the design described in the VHDL file, but translated to a logic circuit, mapped and placed into the logic blocks array, and finally, routed for communication between blocks. Therefore, for every change performed on VHDL design files, the whole steps must be carried out again.

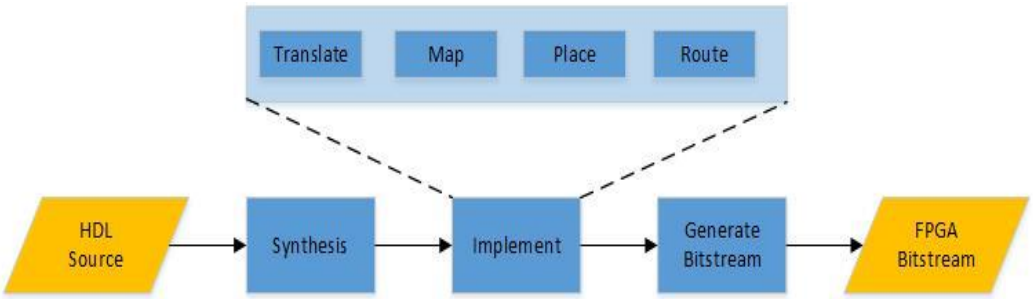


Figure 18: Xilinx design flow.

NAS implementations

For making easier to the final user the test/research process, many NAS .bit files have been generated using the most common NAS configurations, selecting between mono/stereo NAS, how many frequency channels the NAS has, band-pass filters cut frequency, output attenuation, etc, among others parameters. The following list show some of these files:

- NAS stereo, 32 channels, input source using PDM microphones/line-in.
- NAS mono, 64 channels, input source using PDM microphones/line-in.
- NAS stereo, 64 channels, input source using PDM microphones/line-in.
- NAS mono, 128 channels, input source using line-in.

You can find all these .bit files ready to be loaded on the NAS_implementations_files folder on our RTC repository:

https://github.com/RTC-research-group/AERtools

But, how is it possible to have many NAS configurations? Since our neuromorphic sensor is completely digita (although it is spike-based processing instead of digital processing), and FPGA-based development, all of configuration parameters can be defined by the user. Using a software tool named “OpenNAS”, which allows to researchers to configure NAS parameters and generate all VHDL files needed to start the Xilinx design flow for, finally, obtaining the .bit file to be loaded on the FPGA. This tool is still under development process, but it is functional.

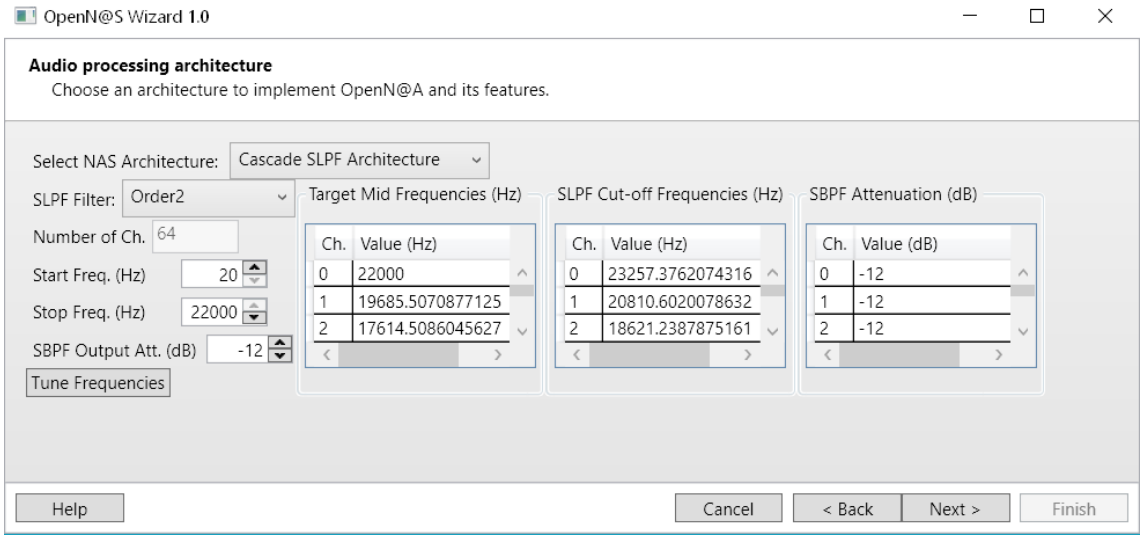


Figure 19: OpenNAS tool screenshot.

First steps

Power up FPGA

As it was aforementioned, assemble process must be performed without connecting the AERNode board to the power supply. Once that process is done, connect the power supply to either the AERNode board or Opal Kelly board. **If there is a file loaded on the flash memory**, green leds which are placed on AERNode board will begin to blink, and they will stop when the FPGA is configured and ready to use (**only "DONE" led will be ON**). If there is not a configuration file (.bit) loaded on flash memory, no leds should be keeped on.

If it is the first time using NAS, or you just want to show a demo, you can skip the next section "Configure FPGA" and go to "How to visualize NAS output". Remember AERNode board has loaded a NAS design by default. But if you want to try with another provided NAS configuration, or test your own NAS design, please continue with "Configure FPGA" steps.

Configure FPGA

Open Impact

When iMPACT is initialized, many pop-up windows could appear. We are going to select always "No" or "Cancel" option. So, the main view after launch iMPACT should be:

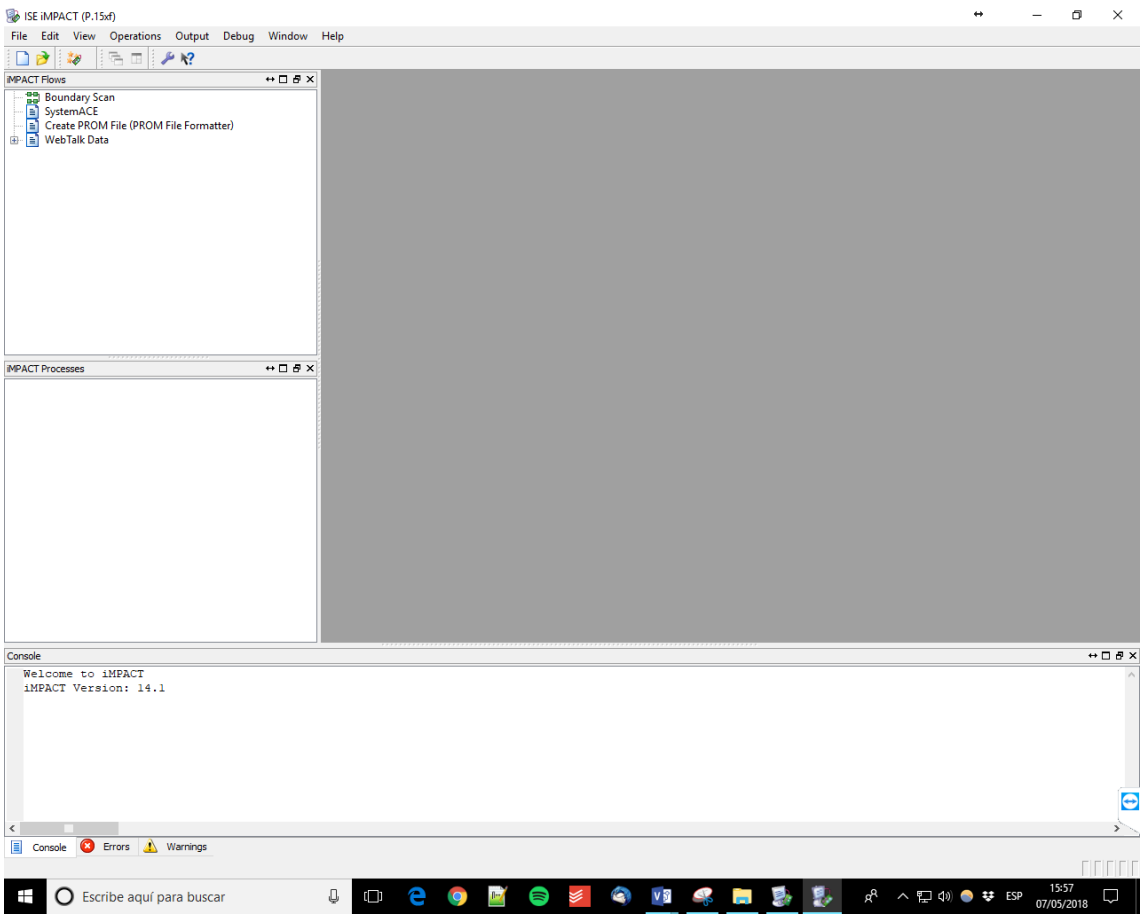


Figure 20: first view after opening iMPACT.

After that, double left click in “Boundary Scan”, at top left corner. Then, the grey background changes to white background:

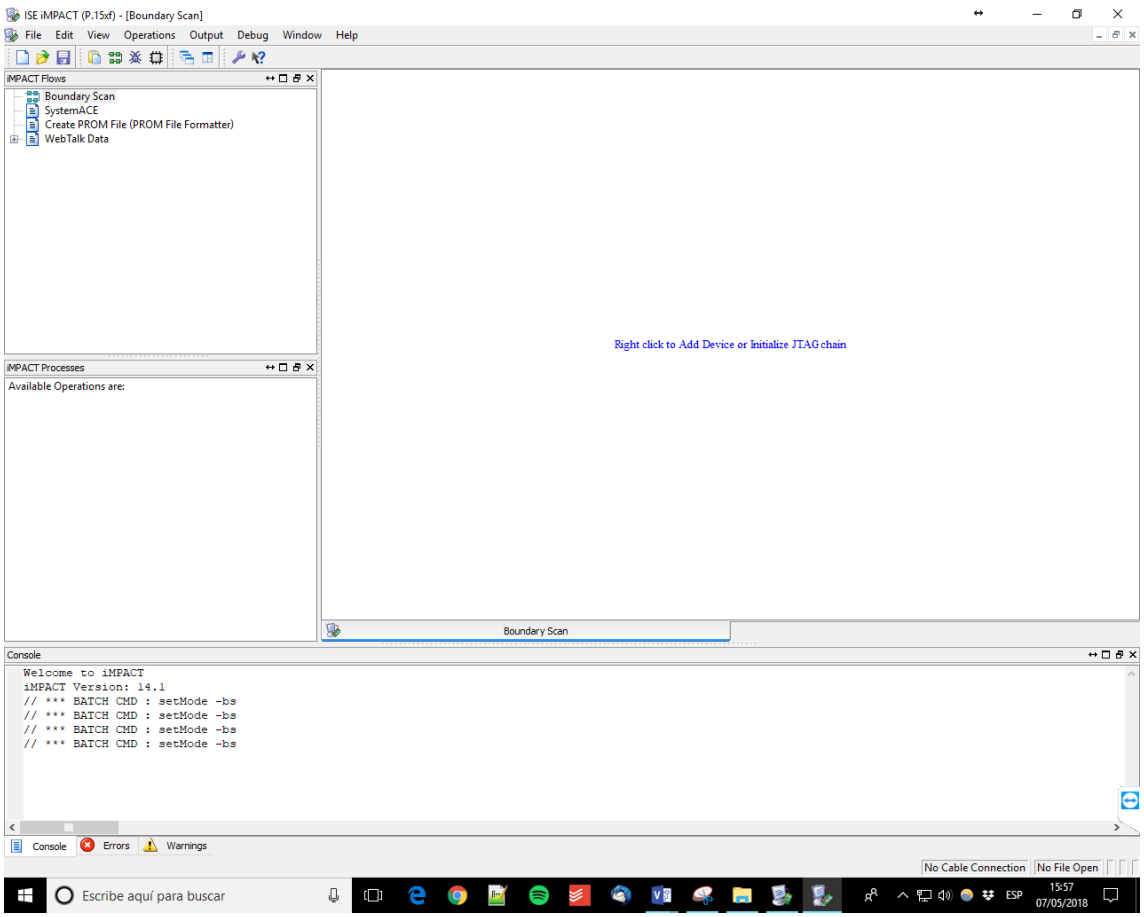


Figure 21: iMPACT after running Boundary Scan.

Initialize chain

Such as it is indicated, right click into main window to “Initialize JTAG chain”:

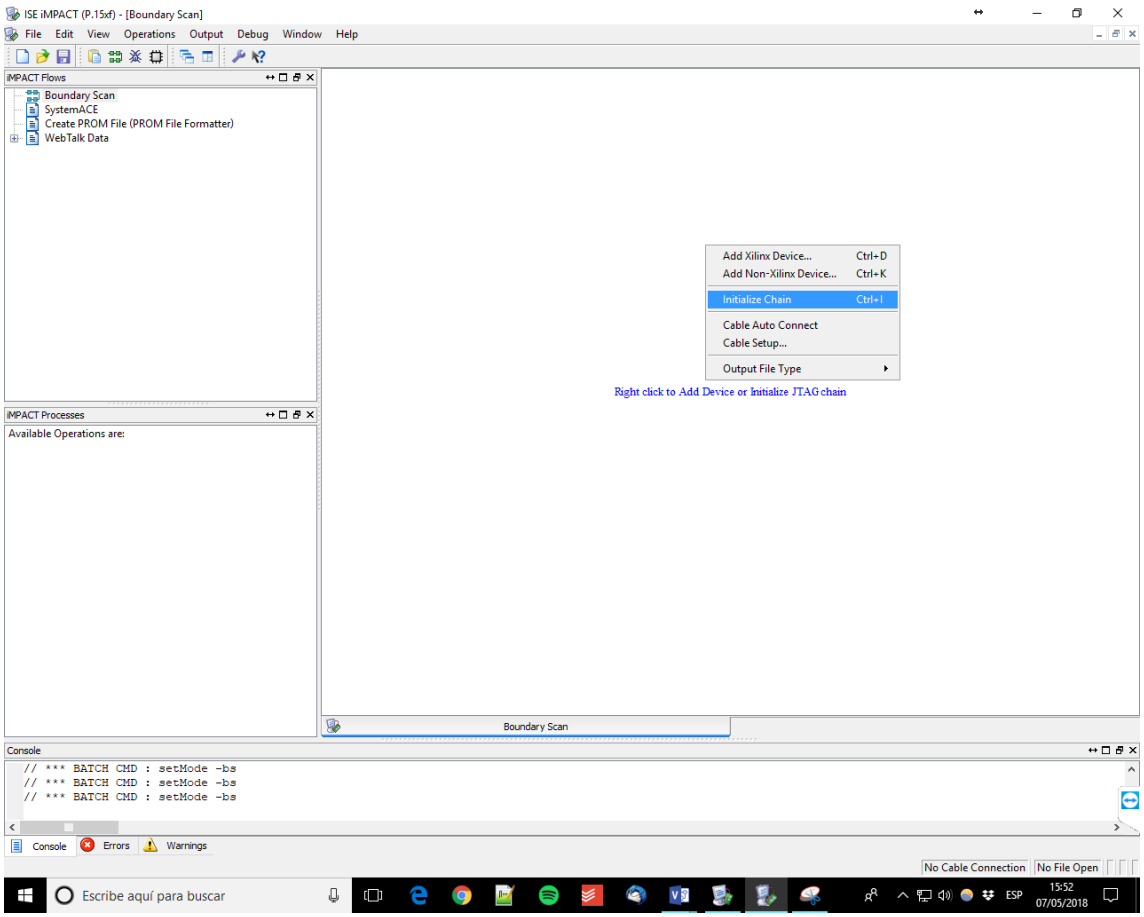


Figure 22: selecting Initialize Chain.

After select that option, iMPACT show us the result of the identification process with a message of “Identify Succeeded” if JTAG devices is correctly connected:

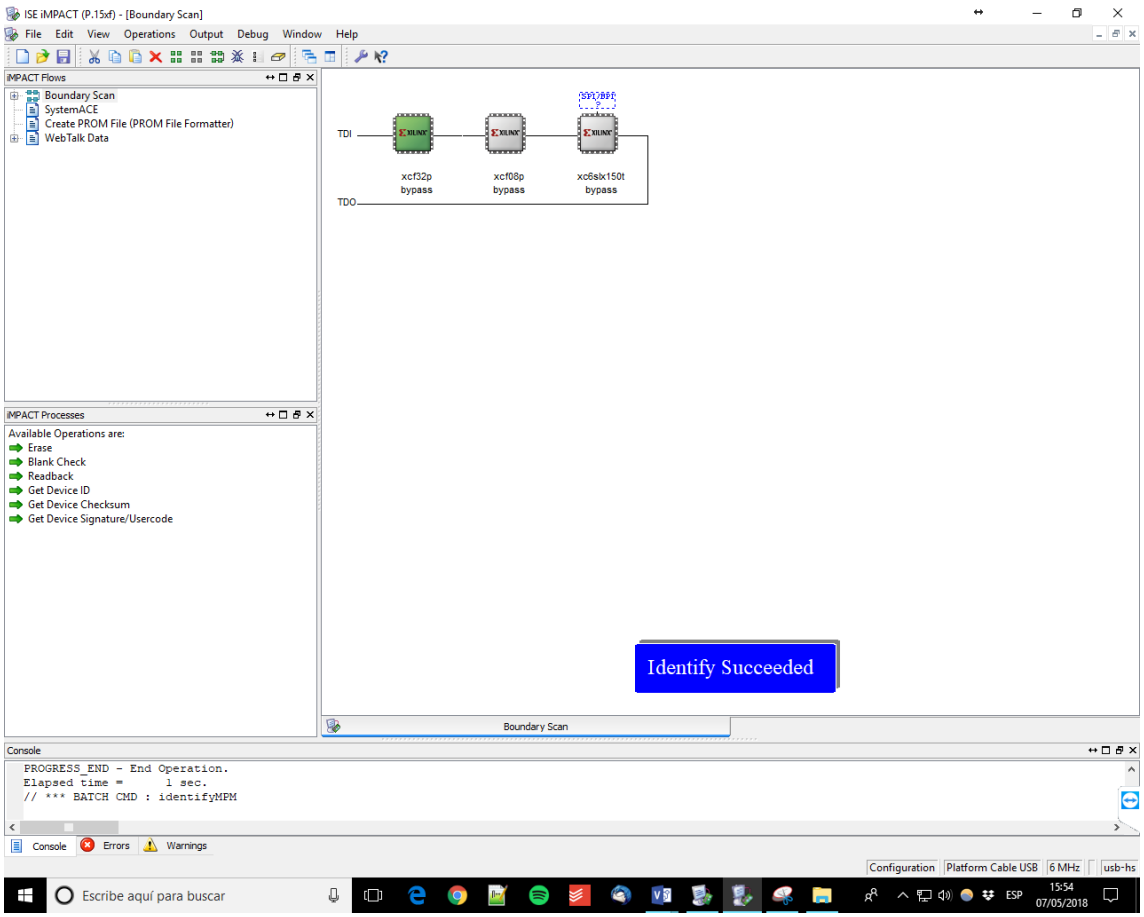


Figure 23: device identification succesfully done.

As we can see, three chips are listed. The first two chips are flash memories, for loading .bit files permanently, and the last one is ram memory (volatile memory). So, **note that when AERNode board is power off, and power on again, the user will have to perform these steps if a design was loaded on ram memory**.

A new pop-up window appears, asking “Do you want to continue and assign configuration file(s)?” User must select “No”. After that, another pop-up window appears; select “Cancel”.

Select and assign a .bit file

Since it is loaded a design by default on flash memory, we want to load a new design on ram memory. But first, we need to indicate the location of the .bit file. First, we select the third chip (ram memory):

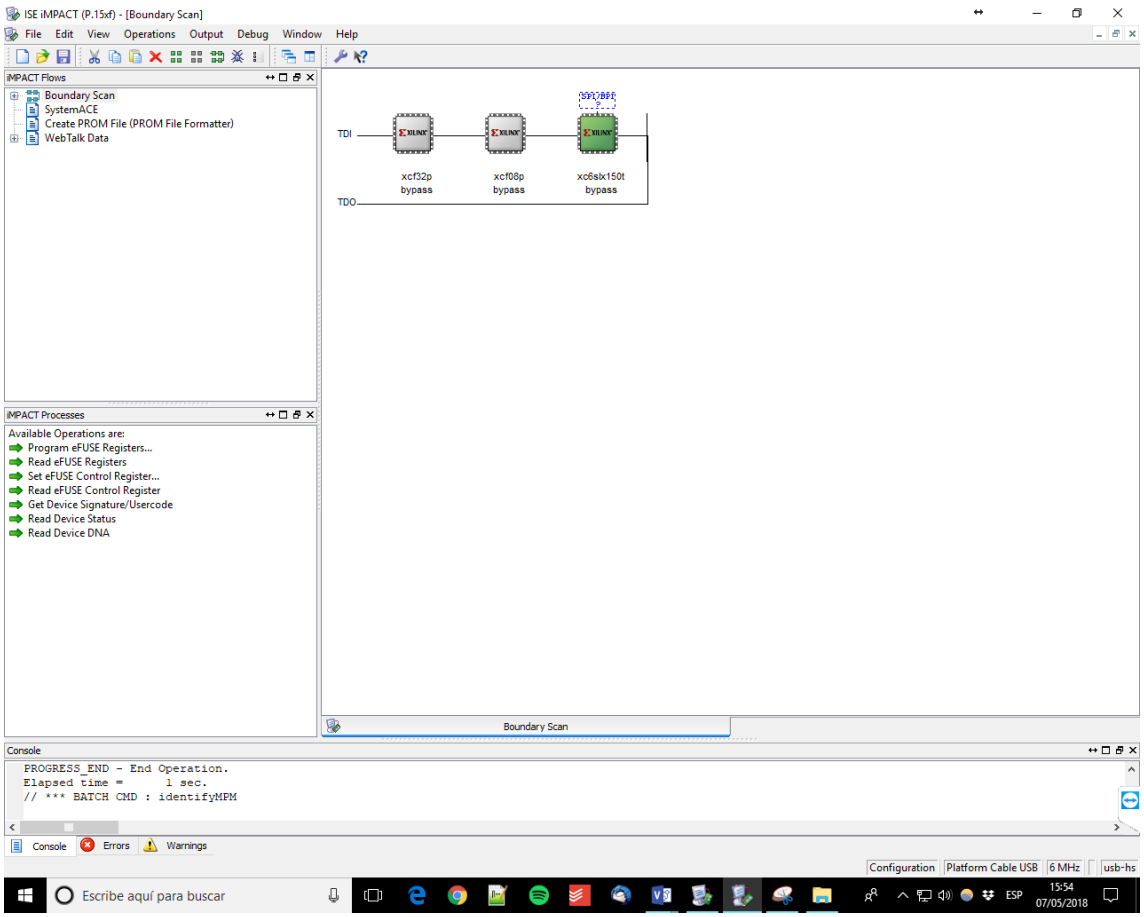


Figure 24: selecting the correct memory.

And now right click on it, and select “Assign new configuration file...”:

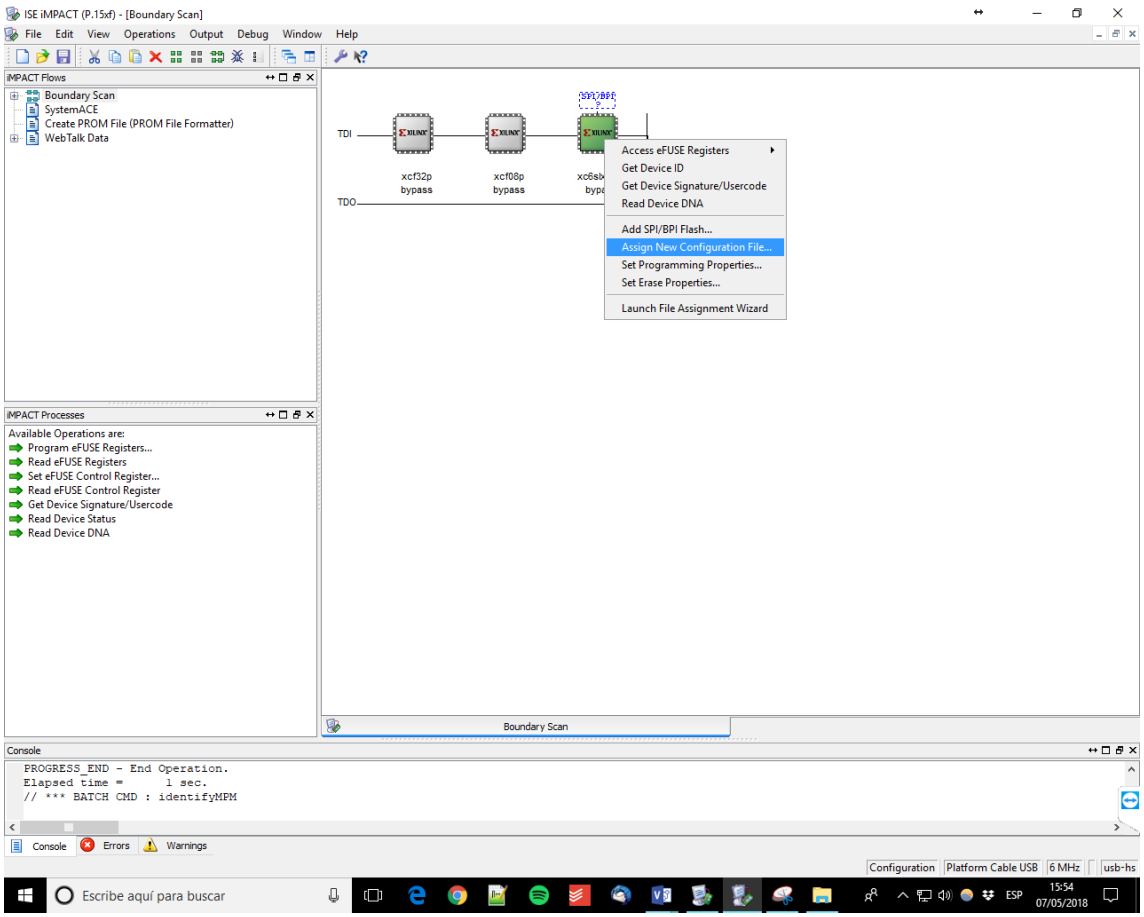


Figure 25: assigning New Configuration File.

A new file selector windows should appear, and the user have to navigate to the .bit file location folder, and select the correct .bit file:

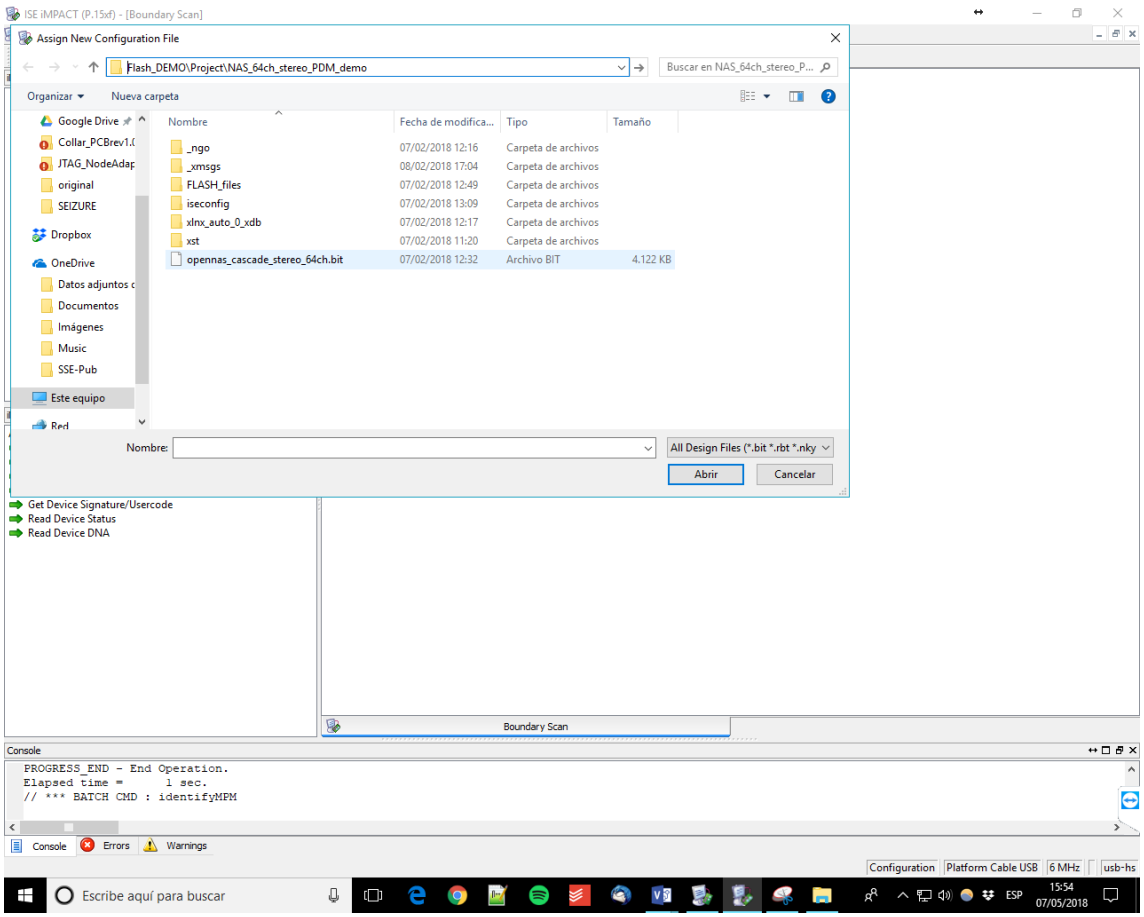


Figure 26: selecting .bit file.

This step should be done without errors, and we can check it in the console, where appears “Loading file ‘C:/Users/.../opennas_cascade_stereo_64ch.bit”... done. In this step, just after click on “Open”, a pop-up window asking “Do you want to attach an SPI or BPI PROM to this device?” would appear, so select “No”.

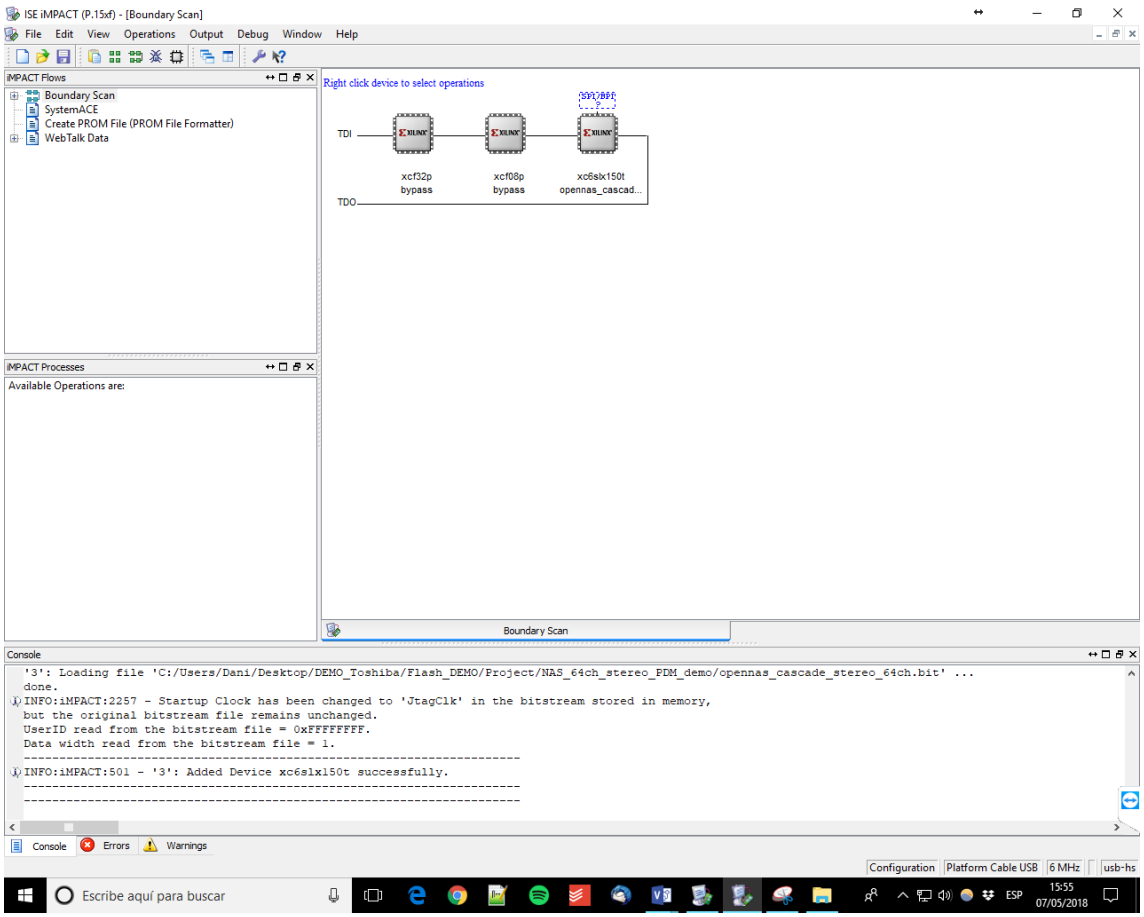


Figure 27: .bit assignation was done.

The .bit file name is very descriptive, because it indicates if the NAS has cascade or parallel architecture (bank filter architecture), mono or stereo, and the number of channels.

Program FPGA (RAM)

However, assign a new configuration file does not mean to program the FPGA. We need select that option by right clicking again on the third chip, and selecting “Program”:

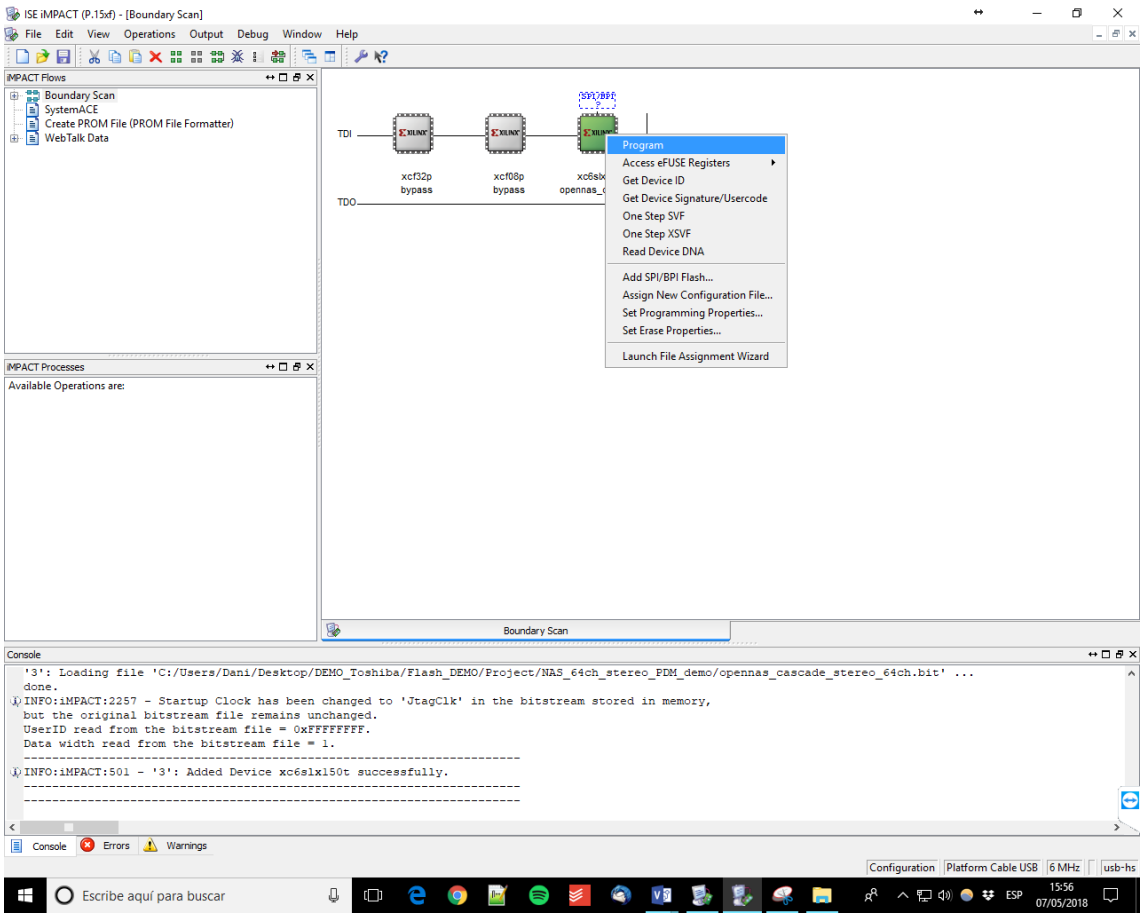


Figure 28: programming FPGA.

And in the “Device programming properties” window, click on “OK” button:

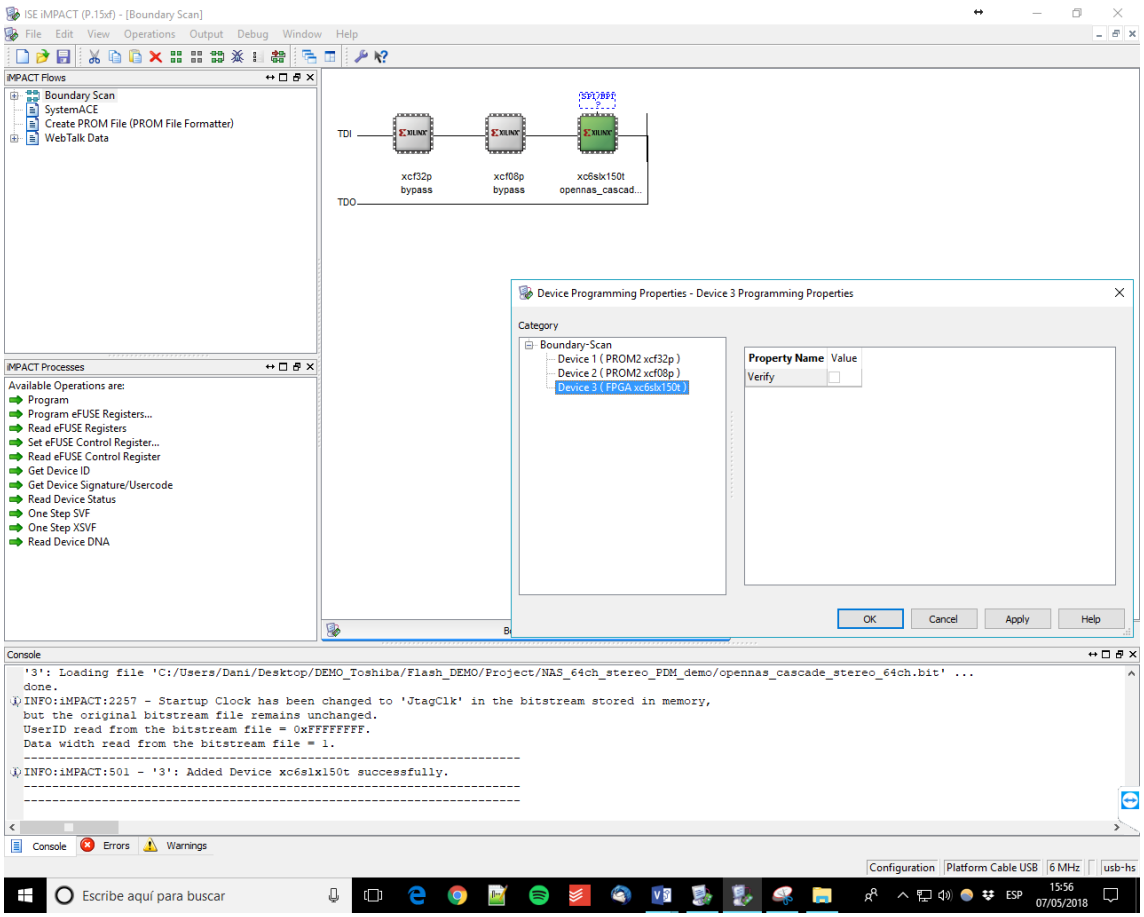


Figure 29: confirmation of FPGA programming.

Done!

If the program process is done successfully, we can see in the console the message “Programmed successfully”:

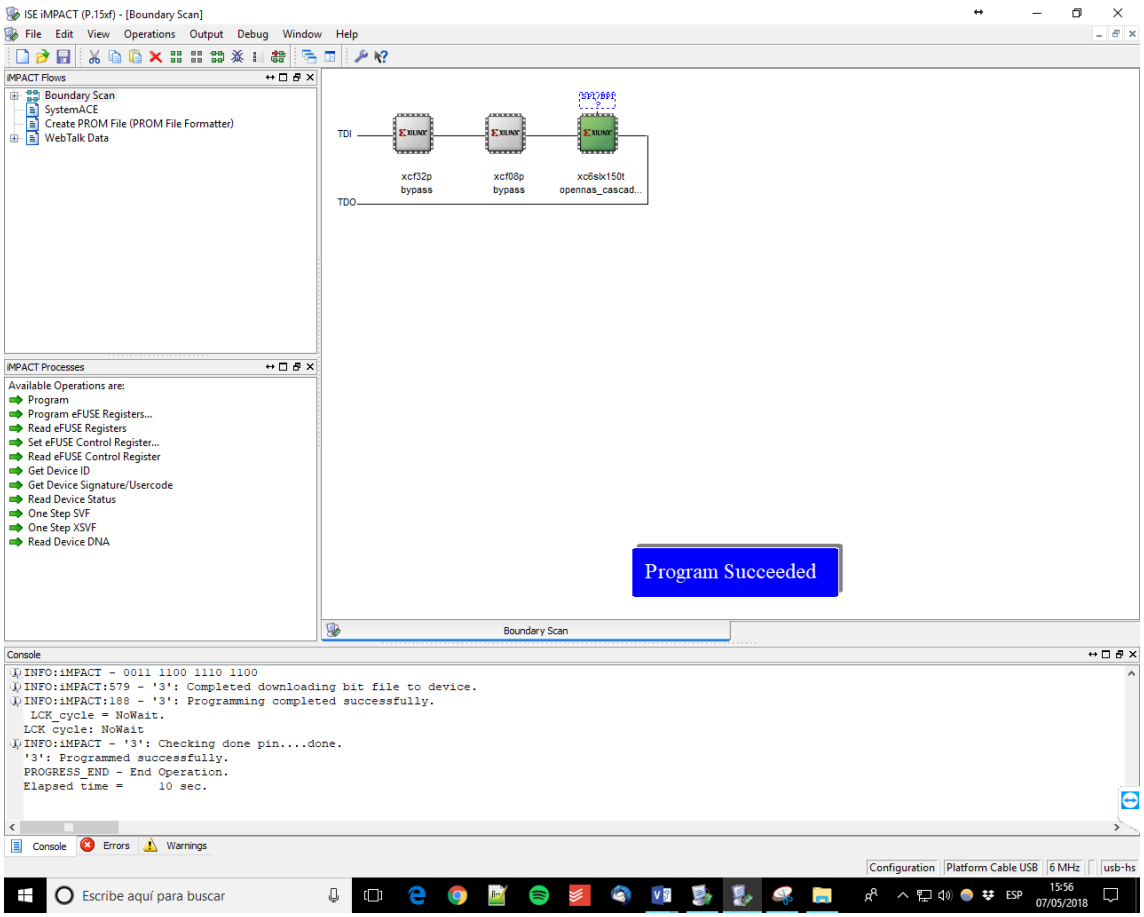


Figure 30: programming process was done without errors.

And now our neuromorphic stuff should be working! It is recommendable to do a global reset, to ensure the system starts correctly, although it is not obligatory.

If we want to close iMPACT, a new pop-up windows appears, and we select “No”.

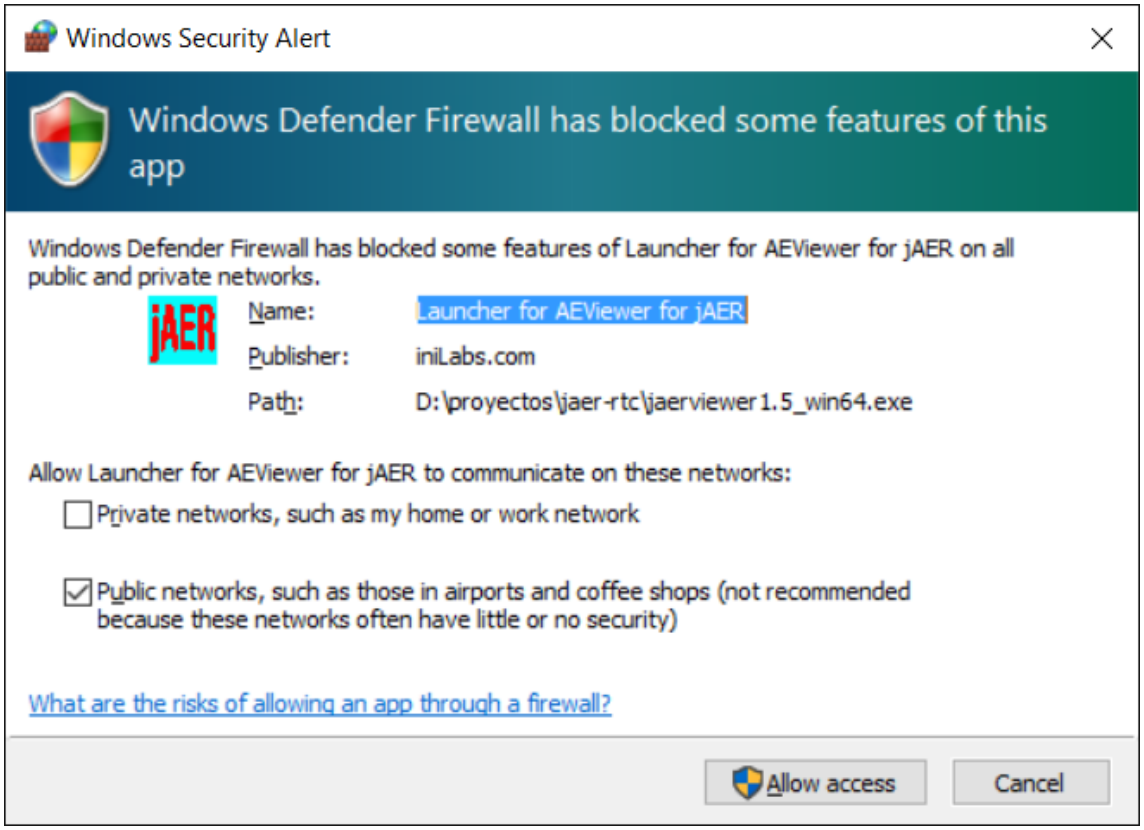
As it was aforementioned, AERNode board has four leds. Three of them are “users leds”: user can use those leds in a design. But those leds were used for indicating FPGA programming steps by default. **However, “DONE” led mustn’t be used by the user, because it is connected by hardware to specific FPGA pin to indicate that it is programmed**.

How to visualize NAS output

Open JAER

Before opening jAER application make sure that OpalKelly drivers have been installed in your system. For this task, download the correct drivers (x86 or 64) from OpalKelly support webpage and install them.

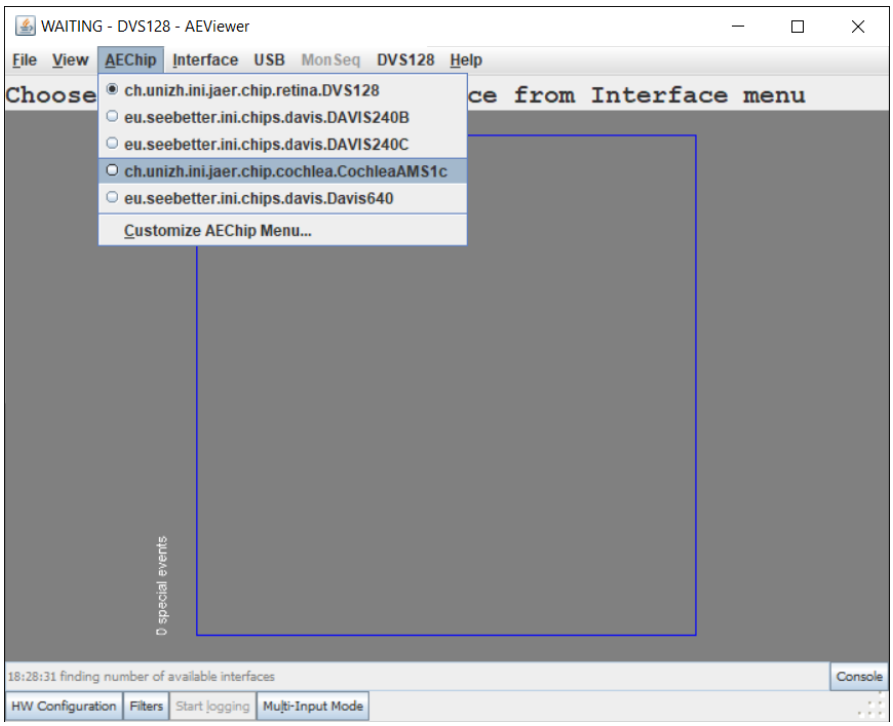
The jAER is an open source application for event based processing algorithm. To launch jAER, open de root folder that has been clone form the Github repository (https://github.com/RTC-research-group/jaer) and double click on *"jAERViewer1.5_winXX"*, where XX is 32 or 64 depend of the driver version that have been installed before. The first time that jAER is opened the following windows is shown. Just allow access of jAER through windows firewall.



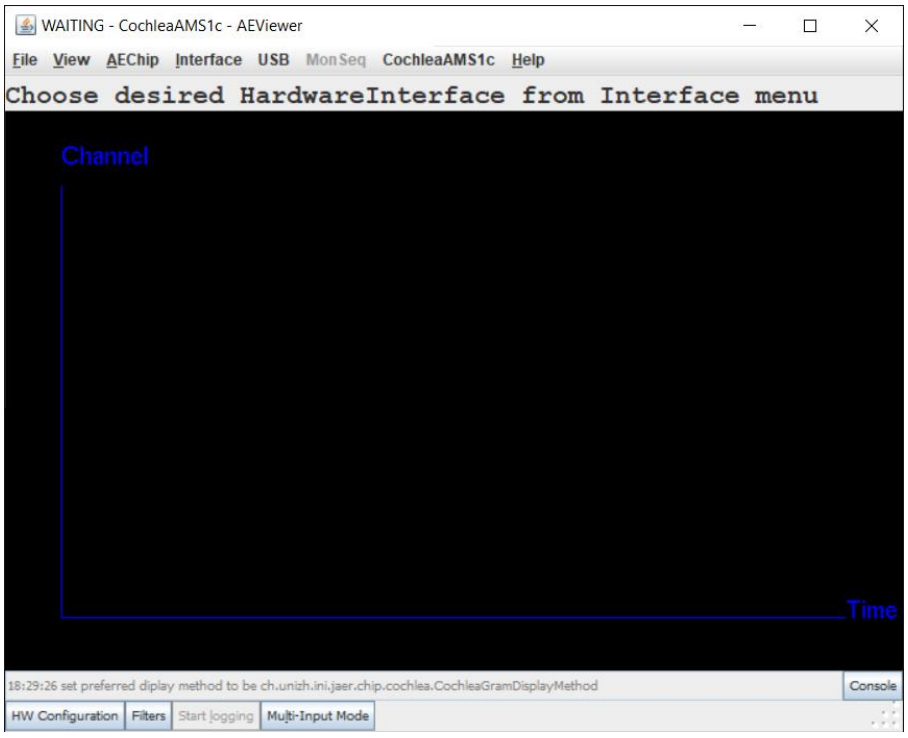
After allow the firewall access, the main jAER windows is shown.

Select Cochlea chip

To select the cochlea chip go to the menu AEChip →ch.unizh.ini-jaer.chip.cochlea.CochleaAMS1c



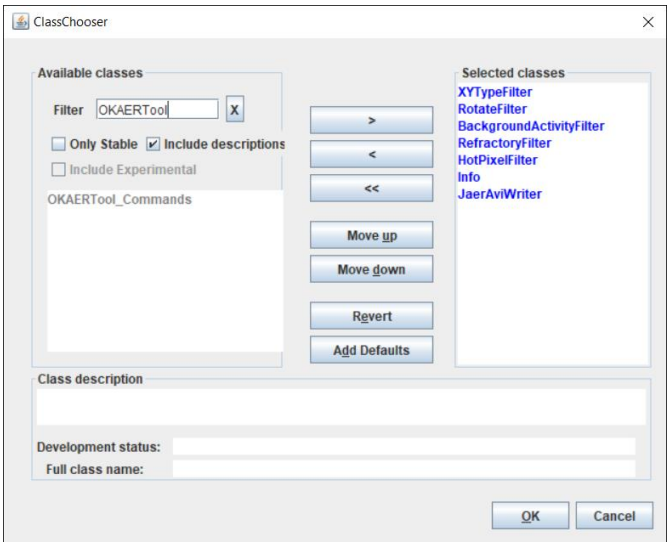
Then select the correct display method under the menu: View → Display Method → CochleaGramDisplayMethod. The main windows should looks as follow.



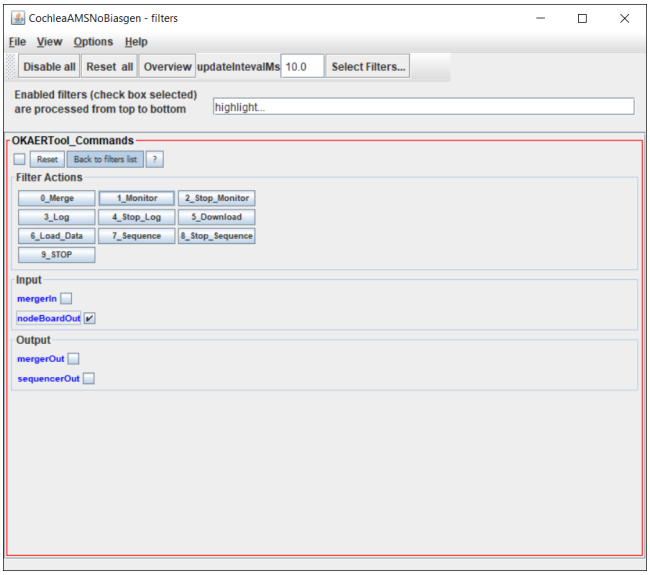
Select interface

After chip selection and the display method, the OKAERTool hardware interface have to be selected under the menu, “Interface”.

To manage all OKAERTool functionalities, the OKAERTool filter has to be added to Filters list. Go to Filters button in the bottom of the main windows and a new window will be opened. Click to “Select Filters” button placed on the top of this windows, and write the name of the filter that we want to add to the list. In our case we should write “OKAERTool” that next figure shows. Then click to the “>” button to add the filter to the filter list.



The main functionality of the OKAERTool is monitor the event coming from event based sensor to jAER application. In this case, the sensor is the cochlea that is running in the AER-NODE board. To monitor the events coming from the cochlea, the OKAERTool option have to be configured as follow, and click on the “monitor” button.



Once “1_Monitor” button is pressed using Input/Output configuration, we can what the event on the main windows.

If you need more information about the OKAERTool functionalities, please contact us.

How to analyze NAS output

After the whole system has been set up and configured, the NAS output can be recorded for further treatment and analysis. For that, two applications are used: jAER and NAVIS.

Record events using JAER

First, the output of the NAS has to be recorded in the computer. We use jAER for that purpose. Open jAER and if you have set up the sensor and the application correctly you will see a button in the bottom part of jAER with the text “Start logging”.

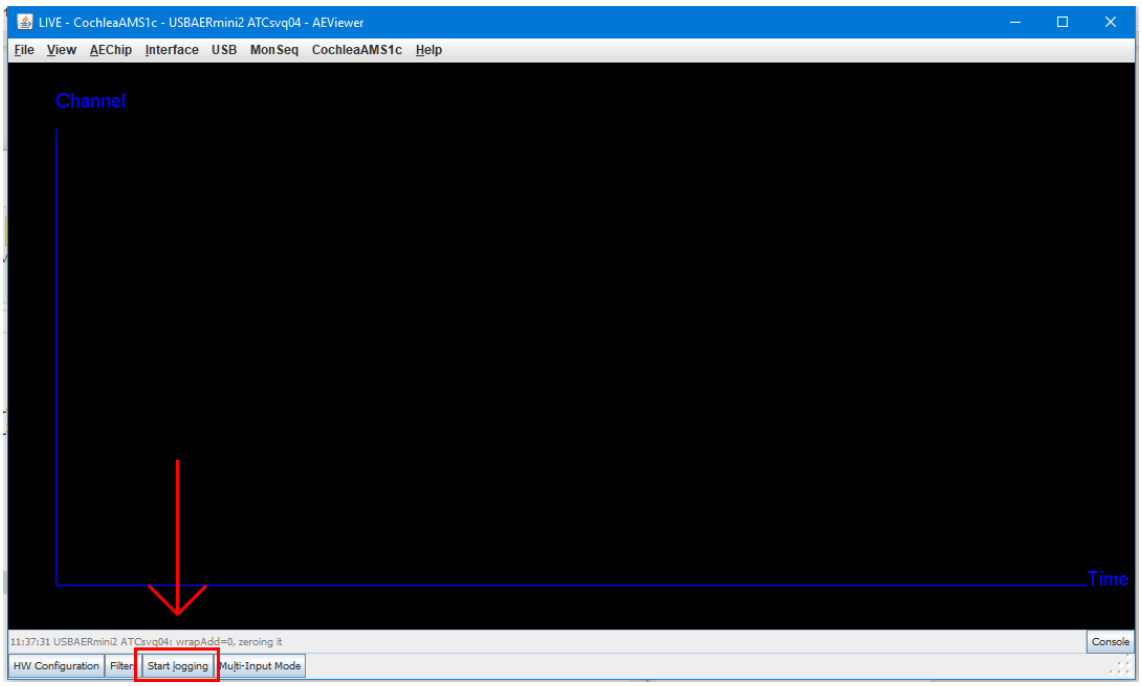


Figure 31: how to record events.

Click on that button to start the recording and, after the time that you want to record has passed, press again the same button to stop it and record the information in a .aedat file. These files can be opened using NAVIS.

Once jAER is configured correctly, you would see something like this:

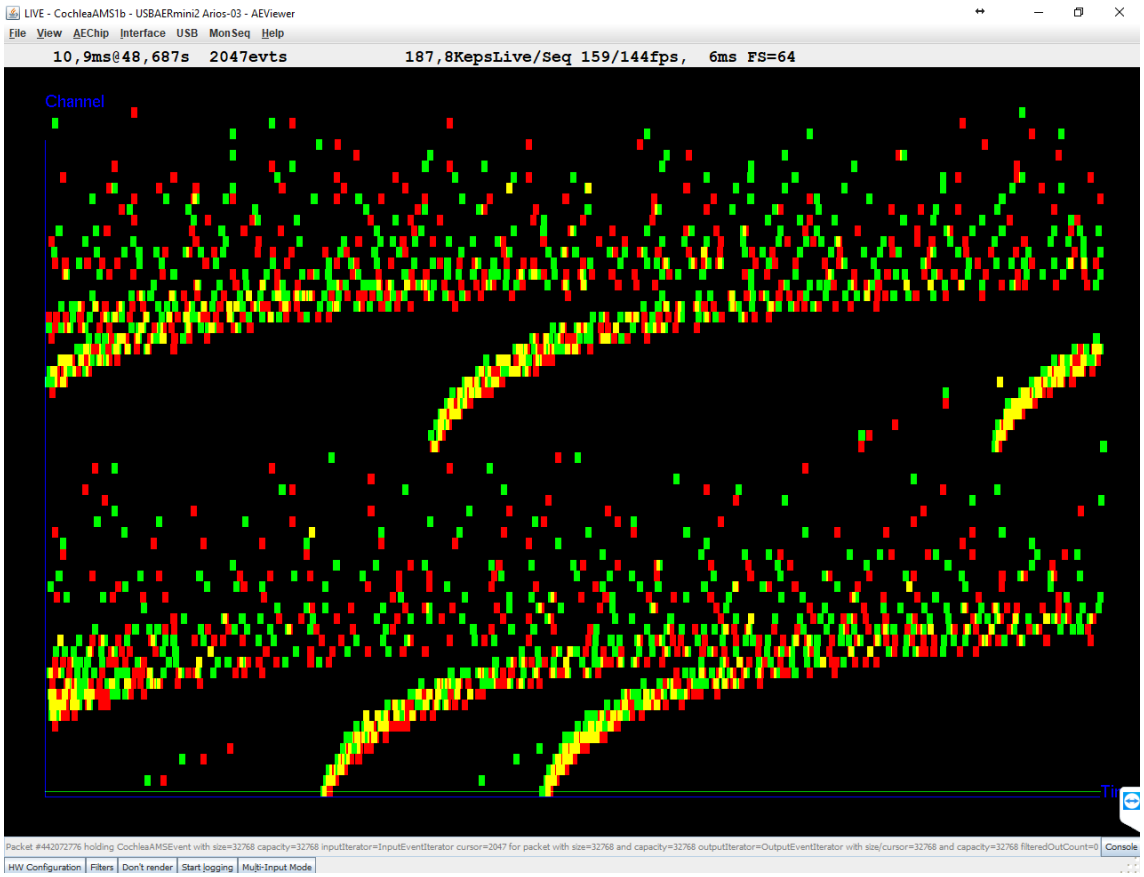


Figure 32: visualizing NAS real time output using jAER.

At the bottom of the screen it is showed the left "ear" NAS output (for both stereo and mono, due to when NAS is configured as "mono", left channel is selected by default). And at the top of the screen it is showed the right "ear".

Load file using NAVIS

NAVIS allows to post-process and analyze the information that is stored in an aedat file. In the GitHub page of the project (https://github.com/jpdominguez/NAVIS-Tool) there's an user manual an a "Getting started" in which it is explained how to load an aedat file using NAVIS and make the most out of its tools. There's also a link to the video tutorial (https://www.youtube.com/watch?v=xJ27ZqqyDRo) explaining this information.

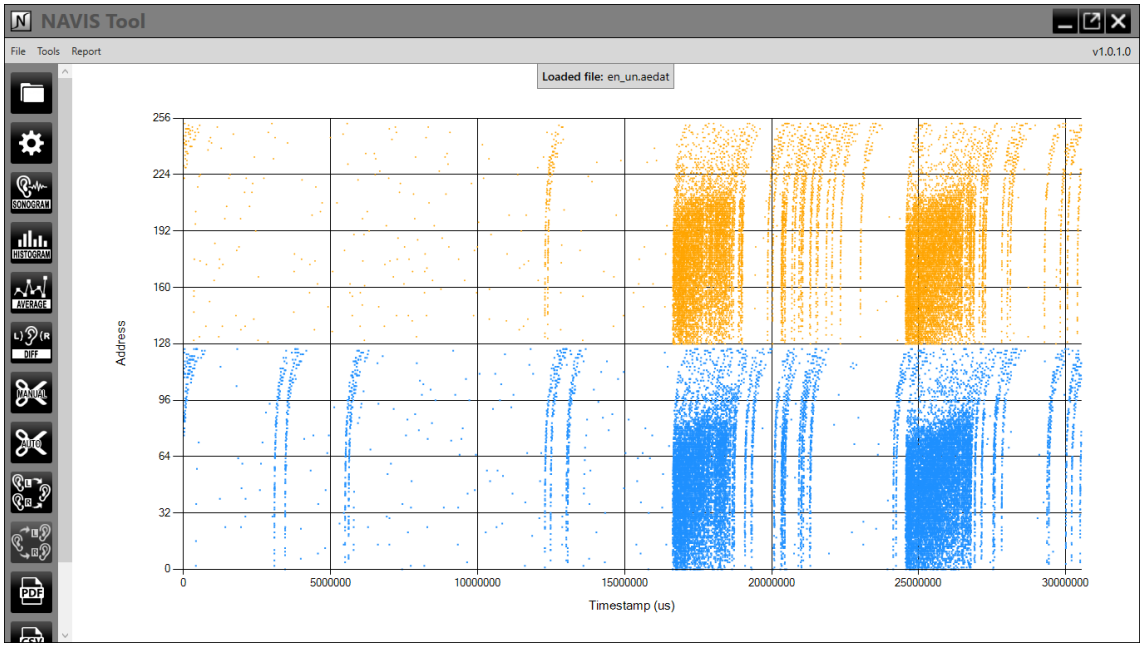


Figure 33: aedat files visualization using NAVIS tool.

References

[1] Fernández, Á. F. J. (2010). *Diseño y evaluación de sistemas de control y procesamiento de señales basados en modelos neuronales pulsantes* (Doctoral dissertation, Universidad de Sevilla).

[2] JIMÉNEZ-FERNÁNDEZ, Angel, et al. A binaural neuromorphic auditory sensor for FPGA: a spike signal processing approach. *IEEE transactions on neural networks and learning systems*, 2017, vol. 28, no 4, p. 804-818.

[3] DOMÍNGUEZ-MORALES, M., et al. On the designing of spikes band-pass filters for FPGA. En *International Conference on Artificial Neural Networks*. Springer, Berlin, Heidelberg, 2011. p. 389-396.

[4] CEREZUELA-ESCUDERO, Elena, et al. Musical notes classification with neuromorphic auditory system using FPGA and a convolutional spiking network. En *Neural Networks (IJCNN), 2015 International Joint Conference on*. IEEE, 2015. p. 1-7.

[5] Dominguez-Morales, J. P., Jimenez-Fernandez, A., Dominguez-Morales, M., & Jimenez-Moreno, G. (2017). NAVIS: Neuromorphic Auditory VISualizer Tool. *Neurocomputing*, *237*, 418-422.

[6] DOMINGUEZ-MORALES, Juan Pedro, et al. Multilayer spiking neural network for audio samples classification using SpiNNaker. En *International Conference on Artificial Neural Networks*. Springer, Cham, 2016. p. 45-53.

[7] CEREZUELA-ESCUDERO, Elena, et al. Sound Recognition System Using Spiking and MLP Neural Networks. En *International Conference on Artificial Neural Networks*. Springer, Cham, 2016. p. 363-371.

[8] MIRÓ-AMARANTE, Lourdes, et al. A spiking neural network for real-time Spanish vowel phonemes recognition. *Neurocomputing*, 2017, vol. 226, p. 249-261.

[9] ESCUDERO, Elena Cerezuela, et al. Real-time neuro-inspired sound source localization and tracking architecture applied to a robotic platform. *Neurocomputing*, 2017.

[10] DOMINGUEZ-MORALES, Juan P., et al. Deep Neural Networks for the Recognition and Classification of Heart Murmurs Using Neuromorphic Auditory Sensors. *IEEE transactions on biomedical circuits and systems*, 2018, vol. 12, no 1, p. 24-34.

[11] A. Yousefzadeh *et al.*, "On Multiple AER Handshaking Channels Over High-Speed Bit-Serial Bidirectional LVDS Links With Flow-Control and Clock-Correction on Commercial FPGAs for Scalable Neuromorphic Systems," in *IEEE Transactions on Biomedical Circuits and Systems*, vol. 11, no. 5, pp. 1133-1147, Oct. 2017. doi: 10.1109/TBCAS.2017.2717341

[12] T. Iakymchuk *et al.*, "An AER handshake-less modular infrastructure PCB with x8 2.5Gbps LVDS serial links," *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, Melbourne VIC, 2014, pp. 1556-1559. doi: 10.1109/ISCAS.2014.6865445

[13] A. Rios-Navarro, *et al.*, "A 20Mevps/32Mev event-based USB framework for neuromorphic systems debugging," *2016 Second International Conference on Event-based Control, Communication, and Signal Processing (EBCCSP)*, Krakow, 2016, pp. 1-6. doi: 10.1109/EBCCSP.2016.7605248