Power Grid Non-destructive testing

Theoretical background

Power Grid stability and reliability are critical to the success of our mission. This is achieved through High Voltage Direct Current (HVDC) mixed grid connections between solar panels, small-module reactors, main base and the far-side outposts. Regular testing and maintenance are essential to this; however, the latest series of tests have revealed some instability in one of the receiving stations. It is not possible to simply replace it but rather the problem must be identified and then dealt with in place.

This task has been prepared by Hitachi Energy team. If you need support, feel free to reach out directly to Hitachi Energy experts!

Goal

Your task is to perform non-destructive testing on the affected station. A dedicated ultrasonic scanning tool has been provided to you which you need to utilize to get an analog response from the station. To achieve this, specialists will send you configuration data using one of the dedicated service communication channels. Be aware that those channels tend to be very unreliable, and the station will not respond to invalid configuration. Providing a valid one to the station will put it in test mode. During that time, you need to execute the scanning and return the received frequency response data to the specialists.

From the FPGA point of view, you will need to receive and check for valid packets, enable test mode, generate stimulus for the testing tool, analyze the analog output and return the frequency responses to the Judge.

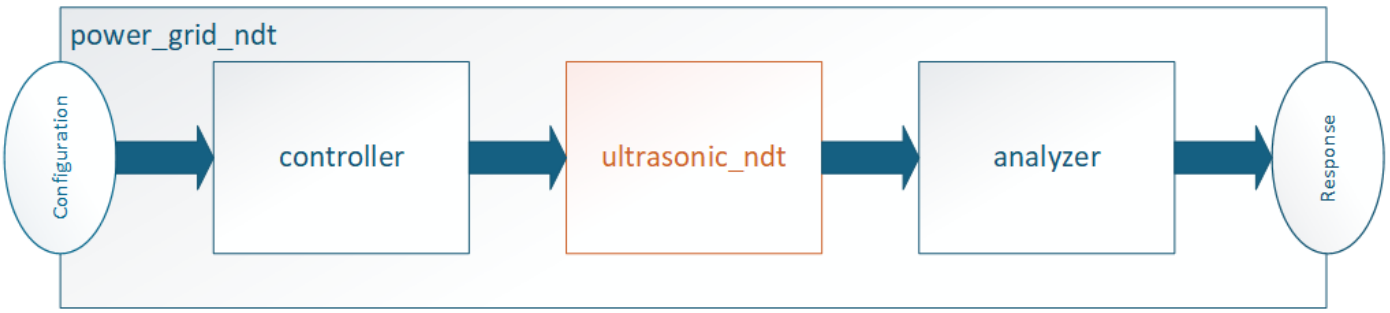

Figure 1 Conceptual diagram of non-destructive testing flow

Top module

You can only edit the **task_13.vhd** file. You may add other files for supportive submodules.

Group	Signal name	Signal direction	Bit width
Clock	i_clk	Input	1
Reset	i_rst	Input	1
Configuration	i_valid	Input	1
	i_first	Input	1
	i_last	Input	1
	i_data	Input	16
Response	o_valid	Output	1
	o_last	Output	1
	o_data	Output	16

Table 1 task_13 module interfaces

Input interface

In one input stream the task receives a sequence of configuration packets on the *i_data* port. The maximum amount of data is 1KB. The beginning of the data block is marked by the *i_first* signal and the end by the *i_last* signal. Throughout the data transfer, the *i_valid* signal is high. Backpressure is not supported.

A single sequence can start and end with any byte value, and it might contain multiple configuration packets as shown in Figure 2. A single valid configuration packet consists of: Start of Frame (0x1234), up to 32 data words (Little Endian), CRC (CRC-16-CCITT), End of Frame (0xFEDC). One valid packet is expected in the sequence.

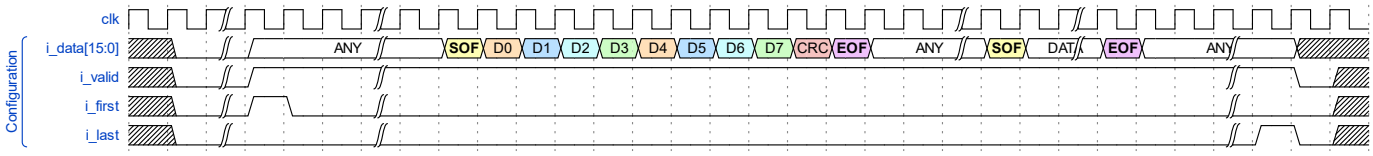

Figure 2 Input waveform

Output interface

The task should return a single sequence of data using the *o_data* port. Along with the data the *o_valid* signal should be set, and on the last cycle of data the *o_last* signal should go high (see Table1). A sequence of six data words is expected by the Judge.

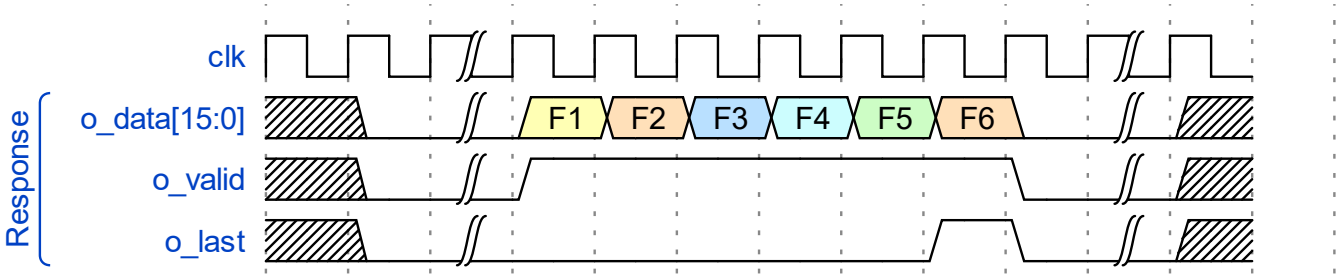

Figure 3 Output waveform

Non-destructive testing module

Main part of this task is realized through **ultrasonic_ndt** testing module provided as a blackbox IP located in the IP Catalog. Refer to the *Appendix* section at the end of this instruction for more details on correct instantiation of ultrasonic_ndt IP.

Group	Signal name	Signal direction	Bit width
Clock	i_clk	Input	1
Reset	i_rst	Input	1
Configuration	i_cfg_data	Input	16
	i_cfg_valid	Input	1
	o_enabled	Output	1
PWM	i_pwm	Input	1
Analog Response	o_valid	Output	1
	o_last	Output	1
	o_data	Output	16

Table 2 ultrasonic_ndt module interfaces

Configuration (Big Endian) needs to be passed to the device to enable test mode. If a valid one is provided *o_enabled* signal will go high within 16 clock cycles. Otherwise, a reset is required to retry the operation.

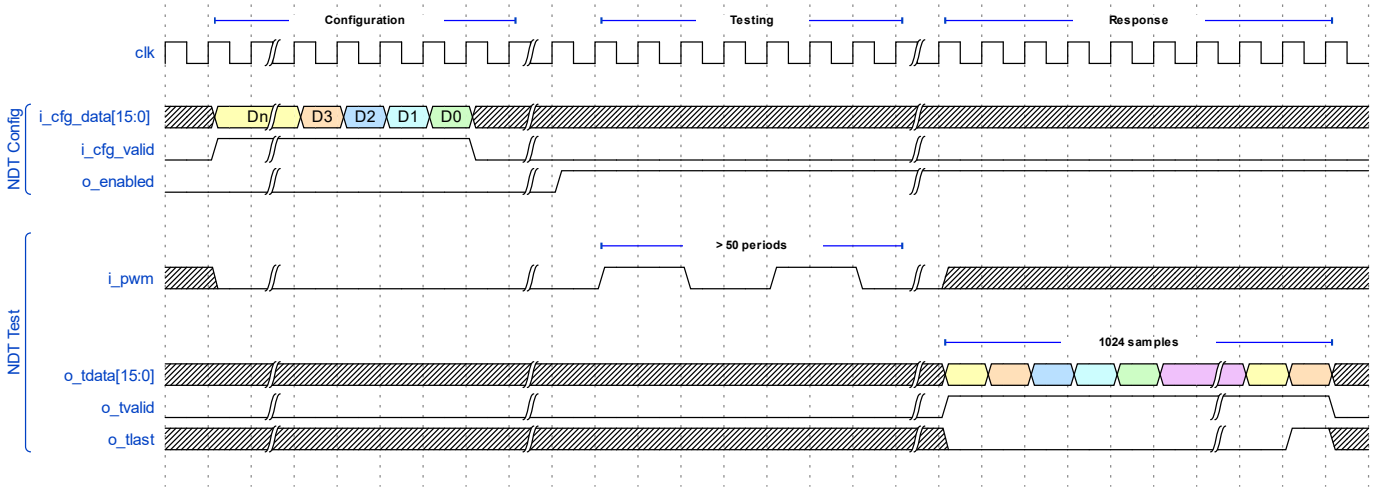

Figure 4 Testing device waveforms

In the test mode, ultrasonic device expects a PWM input in the range of 10 – 40 KHz with 1 KHz intervals and duty cycle of 25%, 50% or 75%. A very accurate signal is required for the device to function properly thus PWM calculations should utilize half up rounding.

When the PWM stimulus gets stable for over 50 PWM periods the device will return 1024 samples of analog response (262144 SPS) in the 16-bit signed fixed-point format with 15 decimal places.

Note: During simulation the PWM will only need to stay stable for around 10 to 16 periods. For reference, one of the expected PWM simulation frequencies is 20kHz.

Figure 5 Example of an analog output from the device

For two of the PWM inputs, the response will be a noisy signal with harmonic distortions or a noisy signal without harmonics if the wrong duty cycle is used. For all others the response will be constant. In each distorted signal three main frequencies can be detected. A total of six frequencies in descending order need to be returned to specialists in unsigned integer (16 bit) format.

Evaluating the task

The evaluation is done by comparing the sequence returned by the task with the expected sequence calculated by the reference model in the Judge. If the sequences are the same, the task is marked as correctly solved. Additional points will be given for the fastest solution.

In Development mode and in Evaluation mode, the task will be tested using a single set of data that is predefined, identical for every team and remains the same across all runs and judge's modes (for more

information, check chapter 2.3.1, “Testing modes”, in the FPGA_Hackathon_2025_Project_Guide document). You earn 10 points for correctly processing the test vector in both Development mode and Evaluation mode. Additional points are awarded if your latency score is satisfactory for such a challenging mission as ours!

Points: **10** for a correct solution + up to **18** for the solution with the smallest latency (see Table 3).

Latency	Scoring
Team with the smallest latency	18 points
Team with the 2nd smallest latency	16 points
Team with the 3rd smallest latency	14 points
Team with the 4th smallest latency	12 points
…	…
Team with the 8th smallest latency	4 points
Each team that achieves latency smaller than 2^{32} clock cycles	2 points
Each team that achieves latency equal or greater than 2^{32} clock cycles	0 points

Table 3 – Scoring table for task 13 (latency)

Appendix

In the task_13.vhd there is a commented-out section that contains instantiation of the *ultrasonic_ndt* ip. We recommend using this ready-to-use code section to have this instance dynamically linked.

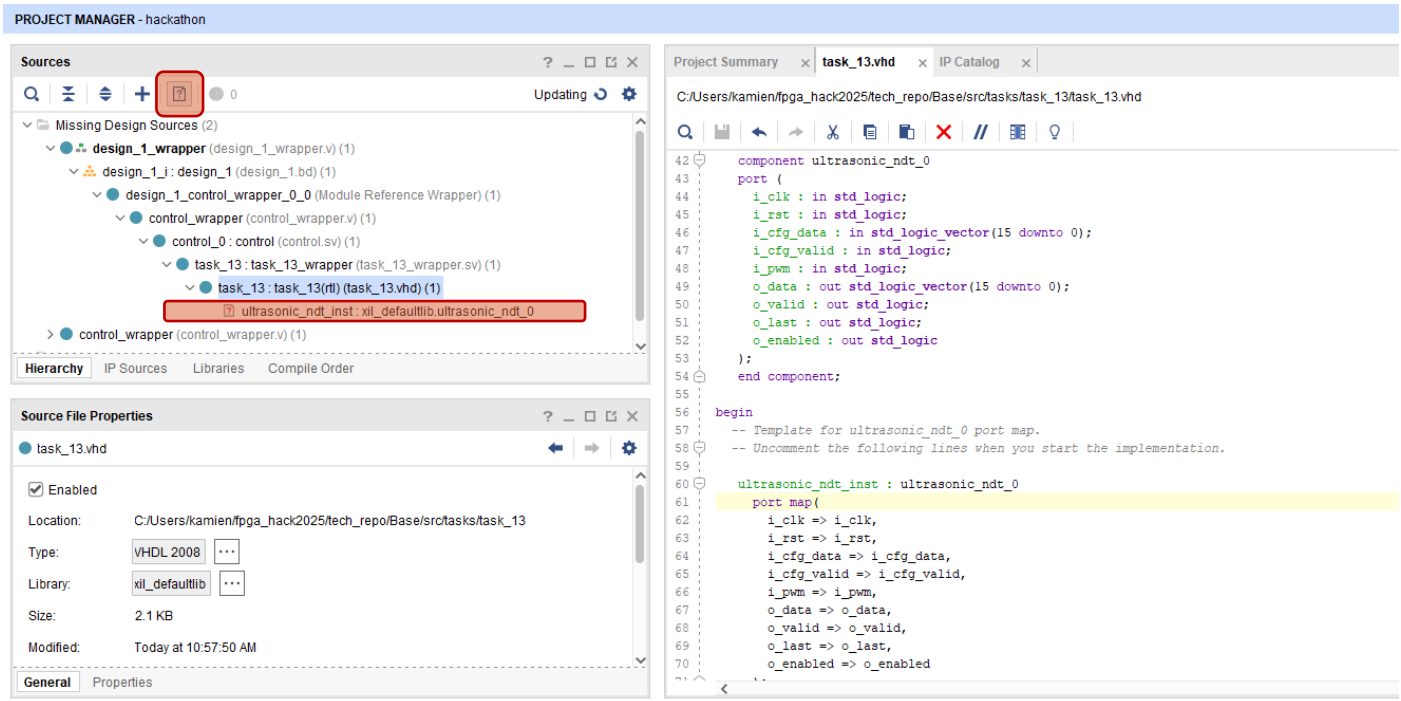

Figure 6 – Unrecognized module after uncommenting the ultrasonic_ndt instantiation

As you see in the Figure 6, Vivado does not recognize the ultrasonic_ndt type. To add this type, go to IP Catalog and double-click on the: User Repository -> Hitachy Energy -> ultrasonic_ndt_v1_2. Firstly, set the IP location to <ws_root>/Base/vivado/ip_repo **(obligatory!)**. If you leave the default path, the instance will saved in the project-related directory that is ignored when committing to the repository. Secondly, check if the "Component Name" is the same as the instance name in the task_13.vhd.

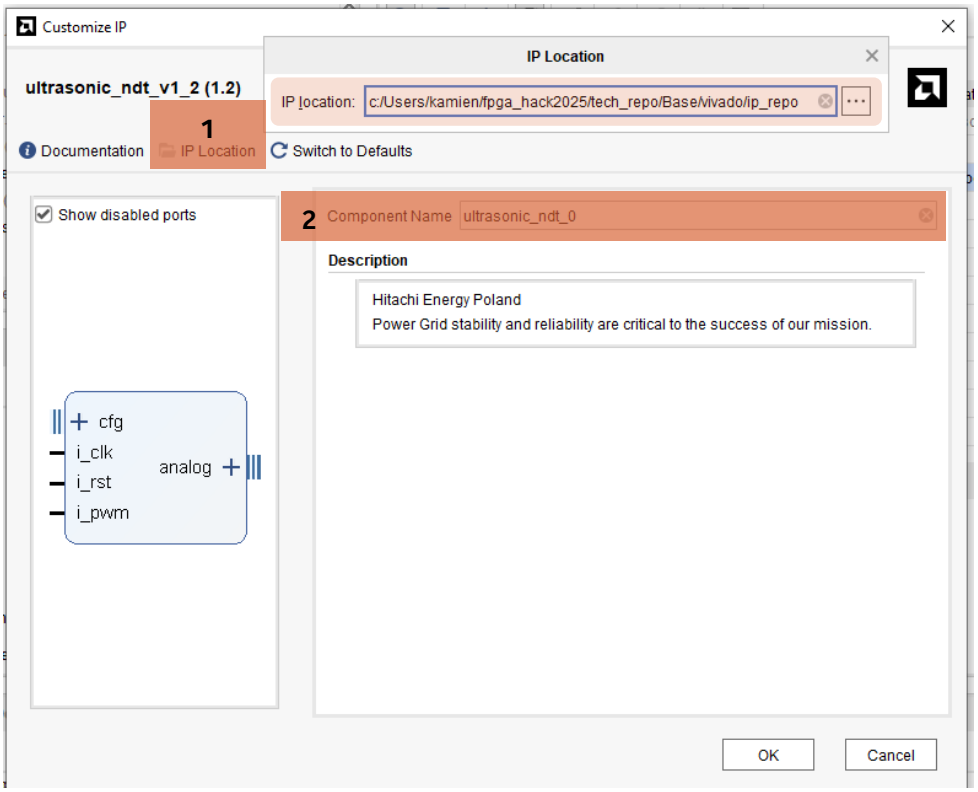

Figure 7 - Creating IP instance in the IP Catalog

Then agree to generate output products. When done, you should no longer see the question mark from Figure 6 and the *ultrasonic_ndt_0* instance should be linked to the correct type. If you need to repeat the process of IP's generation, keep in mind to appropriately update the instance's name.

As the last step, **do not forget to regenerate the project's tcl script**, otherwise the instance of the IP will not be recognized by Jenkins and the compilation will fail.