Development of the firmware logic validation system using the FPGA accelerator

Ryugo Mizuhiki, Junpei Maeda, Seiya Marumoto (Kobe University)

Abstract

Validation of recent FPGA firmware logic used in particle physics is challenging, as the firmware logic becomes larger and more complex with the increasing FPGA resources. To validate the firmware logic efficiently, we developed a validation system using FPGA accelerators. We established a system supporting communication between the CPU in the host PC and the firmware we want to validate. Thanks to the fast and versatile data input/output, flexible validation with large amounts of data is possible. The details and implementation of the developed validation system are presented.

FPGA accelerator

FPGA accelerators are expansion cards with a built-in FPGA.

These cards connect via PCI-Express. They are gaining attention as an option for heterogeneous computing, which combines different types of processors to improve performance and efficiency.

Typically, these accelerators are used to exploit the flexibility and parallel computing capabilities of FPGAs, achieving faster results than using the CPU alone.

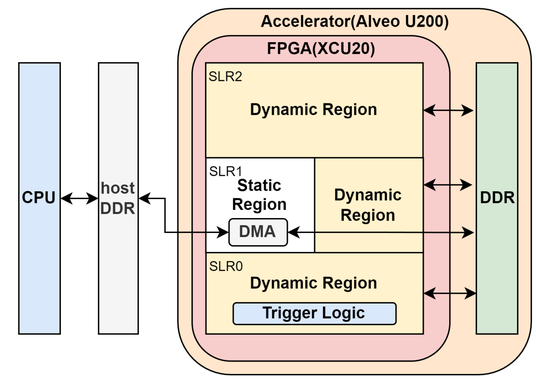
Alveo U200 ©AMD

	Alveo U200 [1]	Virtex Ultrascale [2]
FPGA	XCU200	XCVU9P
BRAM	77.8 Mb	75.9 Mb
URAM	276 Mb	270 Mb
LUT	1,182k	1,182k
FF	2,364k	2,364k
value	**$6,585**	**$64,542**

The Alveo U200 card has almost the same specifications as the Virtex Ultrascale+ XCVU9P, but it is approximately 10 times cheaper.

Data communication with FPGA accelerator.

- Communication between the host PC and the FPGA accelerator is controlled by DMA in the static region on one of Super Logic Region (SLR1). [3]
- We can implement logic to be validated in the Dynamic Region.
- A trigger logic communicates with the DDR using AXI interface, a handshake-based communication interface.
- The calculation results can be transferred to and read by the CPU.

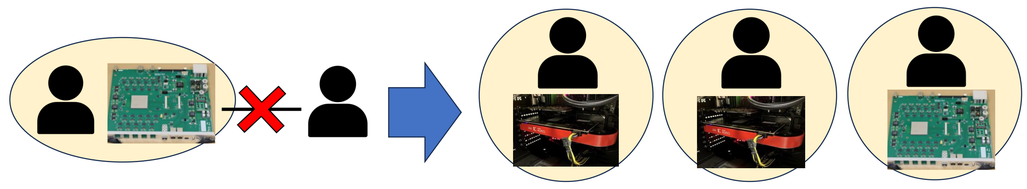


Motivation

Trigger logic verification using the current evaluation board has several issues, so an FPGA accelerator was used to solve the problem.

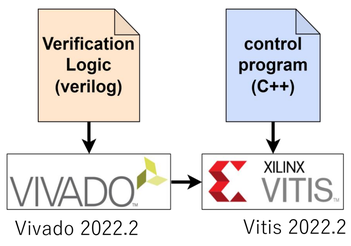
Only a few prototypes are available so multiple developers cannot work simultaneously.
→ **FPGA accelerators allow data to be read and written directly from a PC.**

Complex input/output settings are required.
→ **FPGA accelerators are inexpensive and can be easily introduced in several research institutes.**



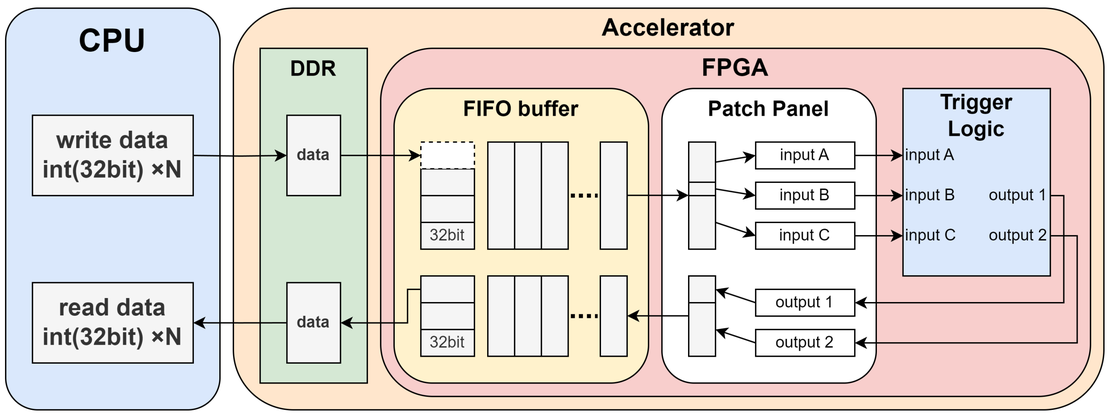
Development Procedure

1. The firmware logic is compiled into object files on Vivado.
2. Develop the control program for the FPGA in C++.
3. Use Vitis to combine the object files generated by Vivado with the C++ code to create a single application. [4]



Overview of the developed validation system

① Generate data in the CPU and transfer it to the accelerator's DDR.
② Write data to the FIFO buffer in the FPGA using the AXI interface.
③ Provide appropriate input to the Trigger Logic using the Patch Panel logic.
④ Perform calculations, output the results, and transfer the data back to the CPU.



- "FIFO buffer" logic combines the input data, divided into 32-bit chunks by the AXI interface, back into a single data stream. On output, the FIFO buffer splits a large data set into 32-bit chunks to match the AXI interface format.
- "Patch Panel" logic redistributes a large data stream into several input signals. At the output, multiple signals are bundled into one.

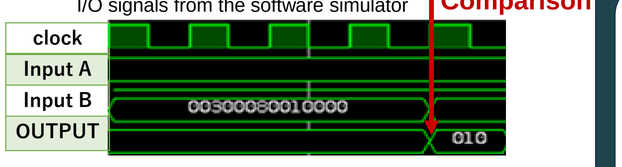
The latency of the signal output can be measured by sequentially repeating the data input and output.

Example of I/O signals from the FPGA accelerator

	1	2	3	4	5	6	7	8	9	10
clock	1	0	1	0	1	0	1	0	1	0
Input A										
Input B										
OUTPUT										

The results of **the software simulator and the accelerator can be compared** to validate that the triggering logic is implemented correctly.

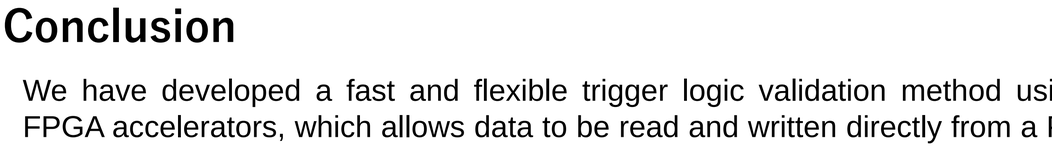
I/O signals from the software simulator



The developed system can validate that the trigger logic has been implemented correctly.

Example of trigger logic validation

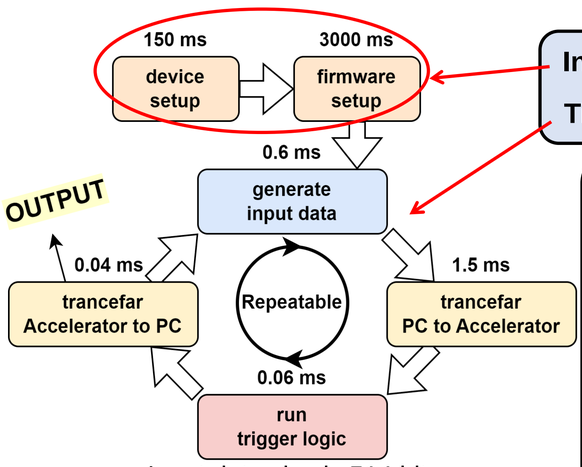
We have run the example logic used for the trigger.



Logic that calculates momentum using a LUT based on the detector's η difference.

Performance

The outputs and latencies of the trigger logic were obtained from the FPGA accelerator. The validation time was also measured.

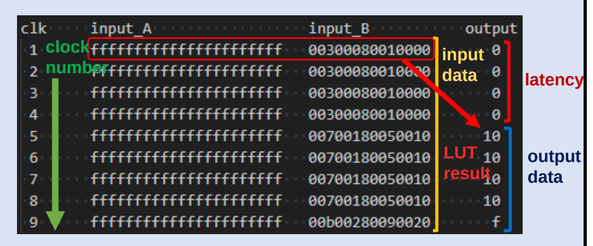


Input data size is 514 bits, output data size is 72 bits.

Initialisation (**3s** for the first time only).

Time per validation cycle. (2.2ms)

We could see the output results in the text format.



Conclusion

We have developed a fast and flexible trigger logic validation method using FPGA accelerators, which allows data to be read and written directly from a PC.

This method streamlines the validation of trigger logics that are time-consuming with conventional methods, such as large-scale LUTs, thereby accelerating trigger logic development.

Reference

[1] Digikey (25th Sept. 2024)
https://www.digikey.jp/en/products/detail/amd/A-U200-P64G-PQ-G/9645681
[2] Digikey (25th Sept. 2024)
https://www.digikey.jp/en/products/detail/amd/XCVU9P-2FLGB2104I/7604576
[3] Alveo Data Center Accelerator Card Platforms User Guide (UG1120)
https://docs.amd.com/r/en-US/ug1120-alveo-platforms/Overview
[4] AMD XILINX Vitis Tutorial
https://Xilinx.github.io/Vitis-Tutorials/2021-2/build/html/index.html

Topical Workshop on Electronics for Particle Physics, TWEPP-24
30 September – 4 October, 2024, Glasgow (United Kingdom)