Final project proposal

COSR 111501502 劉俊廷

Using haps to realize “FPGA Design and Implementation Flow”, “FPGA Verification Flow” , “FPGA Prototyping Flow” and “improve Debug Capability”. Also use Synopsys HAPS Platform, Synopsys Protocompiler and Xilinx Vitis Unified Software Platform to implement.

We have come up with a project involving the implementation of HDL Verilog, and we plan to use the HAPS-100 prototyping system, which is known to be the most scalable and high-performance system in the industry. It offers superior prototyping capabilities, including high-speed performance, efficient debugging, and enterprise-grade scalability, which can accelerate software development, system verification, and validation.

The HAPS-100 prototyping system, which is part of the Synopsys Verification Continuum Platform, enables designers, verification engineers, and software developers to collaborate seamlessly from any location using the HAPS Gateway. It facilitates multi-design, multi-user deployment, which increases productivity and reduces costs.

We also write a testbench in Verilog language to compare with HAPS-100. Since traditional testbenches are run through software simulation, while HAPS prototyping systems are hardware prototypes based on FPGAs, which can simulate real hardware behavior more quickly and provide better accuracy and reliability. Compared to traditional testbenches, HAPS systems have the following advantages:

1. Faster verification speed: The HAPS prototyping system can run hardware prototypes in real-time, making it possible to verify the correctness of designs more quickly and detect errors earlier.
2. Higher accuracy: Since the HAPS prototyping system is based on an FPGA hardware prototype, it can simulate real hardware behavior more precisely.
3. Higher reusability: Since the HAPS prototyping system is programmable, it can be reused in different design projects, saving time and cost.

Overall, the HAPS prototyping system provides a more reliable verification method that can help design teams identify and solve problems more quickly, thereby improving the quality and efficiency of the design.

Our project specifications are as follows:

1. Derive the logic circuit of a 16-bit adder. This involves designing a circuit that can accurately add two 16-bit binary numbers together.
2. Try to add Built-in Self-Test circuit into the 16 bit-adder, by using a 7-stage Linear Feedback Shift-Register as the automatic pattern generator of 127 pseudo random patterns, and a Multi-Input Shift Register as the signature analyzer.
3. Create a testbench to prove it correct by running through software simulation.
4. Use the HAPS-100 to prove it correct by running through hardware simulation. We can also use the HAPS-100 to verify the entire circuit, including the adder and any additional components. This will ensure that the circuit is reliable and meets all design specifications.

Block diagram: