

Design and Verification ARM 7 for FPGA prototype

Wannarat Suntiamorntut

Department of Computer Engineering

Prince of Songkla University, Kohong, Hatyai, Songkla 90112 Thailand

swannara@ratree.psu.ac.th

Abstract

This paper presents a methodology for microprocessor design and verification that significantly reduces a time-to-market and increases user's confidences. The research include the design of a 32-bit microprocessor ARM 7 only in user mode using VHSIC Hardware Description Language (VHDL), functional verification methodology in high-level design and build the FPGA prototype board. Following our method, we can get the synthesizable ARM 7 microprocessor core that has been error-free before download onto FPGA. Cause of this methodology, we used a short time in the board level testing. Almost of errors about 87.5% were a the result of the electrical problems on PCB board. The step in design could be accomplished through five approaches : top-level design, detail design, implementation, component verification and integrated system verification. Functional verification techniques were merged in each design process. The bugs were discovered, continuously decreased and no error in design finally. The verification techniques in this research are simulation-based and abstract data path graph(ADPG). Simulation-based was used in two paths of the microprocessor structure : component and pipeline verification.

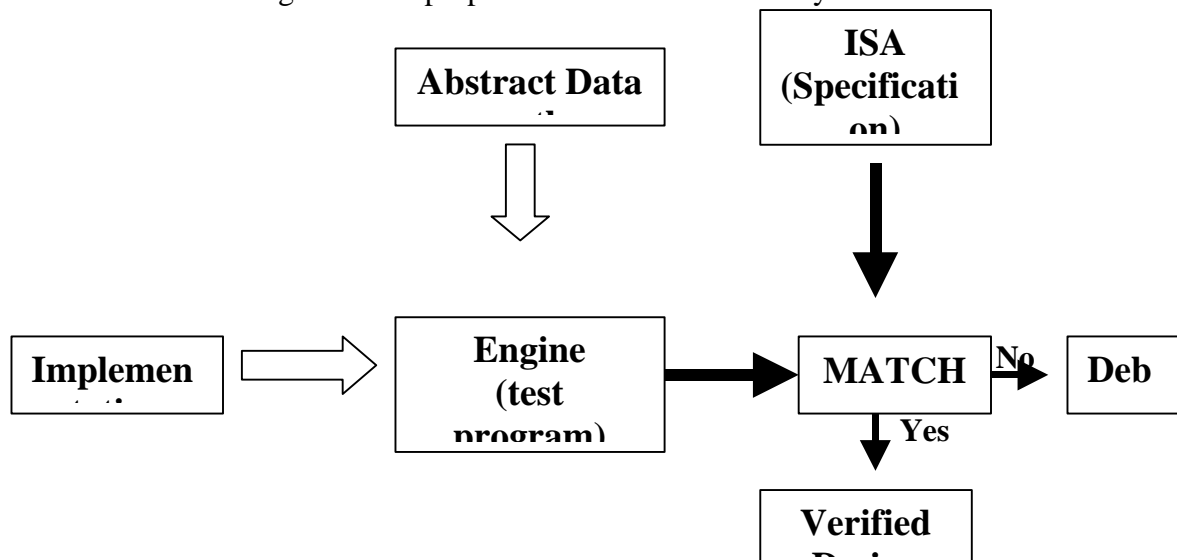
Top-level Design : The data path was the result of this step. Instruction Set Architecture (ISA) was explored and represented with data path. Also this step included the cycle time of each instructions. Survey and comprehension were called in top-level design step.

Detail Design : The data path of the microprocessor was partitioned into small component which we can represent with VHDL. The partition must be added in good design process especially in the complexity design such as microprocessor. .

Implementation : Each component was explained by VHDL and verified via simulation-based in Register Transfer Level (RTL) using hand-written test vectors.

Component Verification : When the error could not found in register transfer level, the VHDL codes were written back from synthesizer. New VHDL codes which were combined by gate delay of the FPGA structure were simulated with the same test vectors. The completed components are the result of this step.

Figure 1 The proposed ADPG verification system.



Integrated System Verification : Abstract data path graph(ADPG) is the methodology to verify a high-level design based on straightforward data path analysis as shown in figure 1. Otherwise, the pipeline could be verified by simulation with test vectors. The instructions which caused pipeline hazard were forced into implementation model for simulation.

As a microprocessor increases in complexity, It becomes harder to verify a design. In order to detect the design errors before building the FPGA prototype, the simulation-based method and ADPG were selected . The errors were detected in board level testing system as shown in table 1.

Table 1 The errors were found on FPGA board test.

Error Analysis	Amount
From function of FPGA	12
External Memory	9
Bugs in design	3

All these increased our confidence in microprocessor ARM 7 to be error-free core designs and easy for debugging FPGA board . With added communications capabilities the processor forms a basis for an ongoing work on a third generation cellular mobile terminal development.

ARM 7 instruction set architecture

The implemented ARM instruction set contains five types of instructions: (1) Data processing that perform a logical operation or an arithmetic operation on two operands. The first operand is always a register. The second operand can either be an immediate value, or a register. In either case, the second operand can be subject to a shift before being used. Compare instructions which do not store a result in register file but always affect the status flag. (2) Single data transfer instruction copy data between memory and register file. (3) Multiple data transfer instruction copy data between an arbitrary subset of register and memory. (4) Branch instruction allow the lines flow of the program execution to be interrupted. (5) Multiply instruction that multiply 32-bit operand to be 32-bit result and the remainder are thrown away.