FPGA AUTO CONFIGURATION BASED ON AUTOMATED TEST EQUIPMENT

Qian Liu, Junwei Sun, Dan Wang

Wuhan Digital Engineering Institute, Wuhan, China, cmmt2008@gmail.com

Abstract: FPGAs (Field Programmable Gate Arrays) are highly integrated devices that can be programmed into variable functions. The application-level testing of FPGAs usually requires multiple reconfigurations and relevant functional tests respectively through ATEs (Automated Test Equipments). However, test engineers are facing a tough problem to re-configure FPGAs automatically by an ATE instead of interrupting the entire test process between every single functional test to change the configuration file manually. This paper takes example for XILINX Virtex-E series, presents two different methods for FPGA auto configuration based on an ATE and discusses both advantages and disadvantages of the two methods.

Key words: auto configuration, JTAG (Joint Test Action Group) interface, boundary scan

1. INTRODUCTION

FPGAs are large-scaled reprogrammable devices with high performance that are commonly used as key devices in electric systems.

With the rapid development of FPGAs, application-level tests for them are becoming more and more complicated. Usually for medium/small scale ICs (Integrated Circuits), test engineers write test patterns and programs according to their functional descriptions and AC/DC characteristics. However, FPGAs are highly integrated, very large scale, flexible devices. Traditional test methods are no longer suitable for them, and it's impossible for engineers to write test patterns manually. A complete test on a certain FPGA should basically cover multiple different functional tests which means the FPGA under test should be programmed several times to perform different functions[1]. There comes a problem that how to reconfigure the FPGA automatically during test process.

This paper takes XILINX Virtex-E series for example, presents two different methods for FPGA auto configuration based on V93000 test system using JTAG configuration interface and boundary-scan protocol. The entire auto configuration-test procedure can be accomplished by a single ATE.

2. CONFIGURATION METHODS ANALYSIS

2.1. Traditional Configuration Method

In most cases, XILINX FPGAs can be generally configured through Serial, SelectMAP and JTAG interfaces. Traditional configuration method is to program the specified bitstream file through specific tools and a XILINX download cable offered by the manufacturer. In another way, a platform flash PROM (Programmable Read-only Memory) can be used to store the configuration data as well. In this case, FPGAs automatically load the configuration data in bit-serial form from the PROM synchronized by a certain clock after power-up.

This method is very common in engineering because FPGAs usually operate as a fixed function in general applications. However in application-level test, FPGAs have to be re-configured several times for different functions. It is obviously very inconvenient to always interrupt the test, change a configuration file, download again and return to the test. It takes a lot of time especially in volume test.

This paper presents two solutions based on both advantages and disadvantages of the traditional method.

2.2. In-system Configuration Based on An ATE

In-system programming based on an ATE aims to implement the configuration process without specific tools and download cables. The idea is to convert each configuration datastream into a test pattern and implement the configuration process in the test program. An independent test item can be used for each different configuration. FPGAs can be configured directly by the ATE.

The entire configuration-test procedure can be accomplished by a single ATE using this method. It is also very efficient for volume automated test. Yet converting the pattern can still be very complicated. Each converted pattern may contain millions of vectors. It brings a huge challenge for vector compression and debugging.

2.3. Configuration through Parallel PROMs

In this method several PROMs are connected to a FPGA in parallel. Different configuration files are stored in different PROMs in advance. After finishing a certain function test, ATE controls the multiplexer to select the next PROM, disconnect FPGA and power-up again. Test program waits for the FPGA's reconfiguration and go to the next functional test automatically.

This work can also be done without specific tools and XILINX download cables during test process. It is much easier to carry out relative to in-system configuration method based on an ATE. However it requires pre-design of the test interface board.

3. IMPLEMENTATION OF IN-SYSTEM AUTOMATED CONFIGURATION

3.1. Boundary-scan Configuration Sequence

In-system automated configuration based on an ATE can be implemented according to boundary-scan timing. The boundary-scan logic is accessed through the Test Access Port (TAP). The TAP is comprised of four semi-dedicated pins: Test Mode Select (TMS), Test Clock (TCK), Test Data Input (TDI) and Test Data Output (TDO) [2].

TAP controller is a 16-state finite state machine as depicted in figure 1. The four TAP pins control how the data is scanned into the various registers. Virtex FPGAs can be configured through a set of 5-bit boundary-scan configure instructions in a certain sequence. All the instructions necessary can be found in the device datasheet.

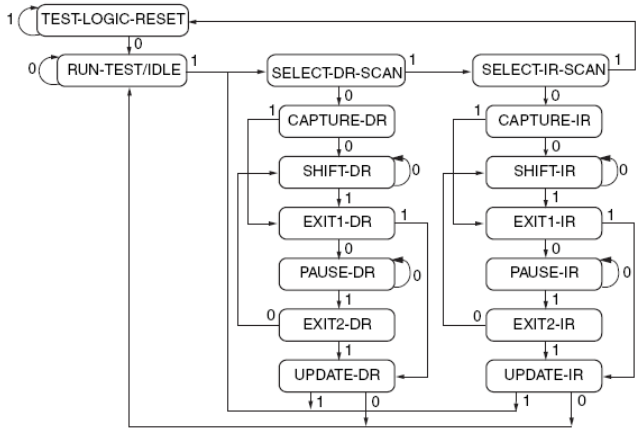

Fig. 1. State machine of TAP controller

Connect TDI, TDO, TCK and TMS pins of the FPGA to ATE's digital channels respectively. Set mode select pins M2, M1, M0 to boundary scan mode "101". Configure the FPGA as follows:

On power-up, place a "1" on the TMS and clock the TCK five times. Enter the SHIFT-IR state and start loading the "CFG_IN" instruction"00101" from LSB through TDI. Load the last bit of "CFG_IN" instruction while exiting SHIFT-IR, then enter the SHIFT-DR state and Shift in the configuration pattern. Enter the SHIFT-IR state and start loading the "JSTART" instruction which initializes the FPGA startup sequence. Finally move to the SHIFT-DR and clock the startup sequence by applying a minimum of 12 clock cycles to the TCK and return to the RTI state.

3.2. Auto Test Pattern Generation

The FPGA configuration file (bitstream file) is primarily compiled by XILINX ISE platform. Then a conversion tool must be developed to convert the bitstream file into the serial configuration pattern according to the above boundary-scan configuration sequence.

FPGAs can be configured directly by an independent test item of the test program, and then the corresponding functional test is performed next to it.

4. IMPLEMENTATION OF CONFIGURATION THROUGH PARALLEL PROMS

4.1. Test Interface Board Design

Take Virtex-E FPGA series and V93000 test system for example, assume the FPGA requires *n* times re-configurations. One can download different configuration files (bitstream files) into *n* PROMs in advance. Connections between parallel PROMs and the FPGA on the test interface board is shown figure 2.

"ADG709" can be used as the multiplexers that satisfy both the FPGA operating voltage and the supply voltage of V93000 Utility Lines. Utility lines are additional control lines of V93000 test system that may be used for a variety of purposes, such as controlling relays, solenoids, or indicators. Utility lines are not affected by standard test program or test flow, but are only controlled by dedicated test function commands. This allows the utility line to assume a steady state setup before the actual test flow takes control and to remain in this state.

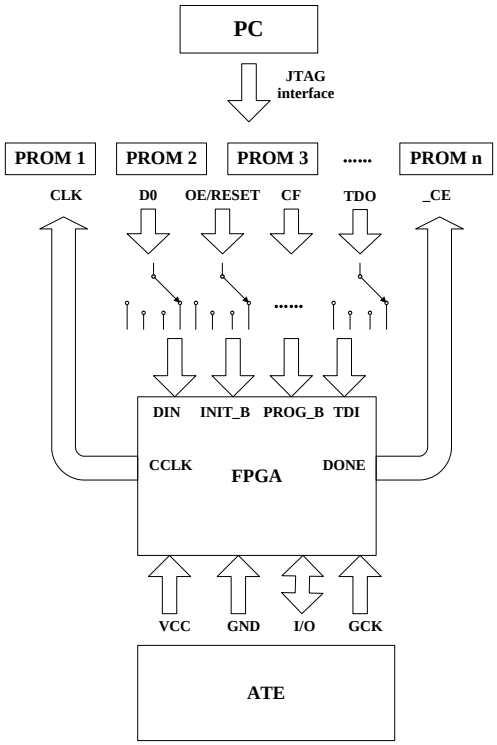

Fig. 2. Connections between parallel PROMs and the FPGA on the test interface board

Connect four output signals D0, OE/RESET, CF and TDO of each PROM to the input pins of ADG709 respectively. Connect the corresponding output pins of ADG709 to DIN, INIT_B, PROG_B and TDI of the FPGA at the same time. Connect the chip enable signals of ADG709 to V93000 Utility Lines.

Set the mode select pins M2, M1 M0 to master serial mode "000" and connect all the other I/O pins of FPGA to digital channels of V93000 test system.

The operating voltage of the FPGA and PROMs are supplied by DPS (Device Power Supply) of V93000 test system while the ADG709 multiplexers are supplied the Utility Lines.

2.5. Test Program Design

The flow chart of the test program is shown in figure 3. Perform continuity test, IDCODE verification, functional tests, DC parameters tests and AC parameters tests in turn. The functional tests include *n* times different re-configurations and *n* times corresponding functional tests.

Disconnect the FPGA power supply before every single re-configuration. The corresponding PROM is selected by the Utility Lines. The select sequence is determined by the functional test sequence.

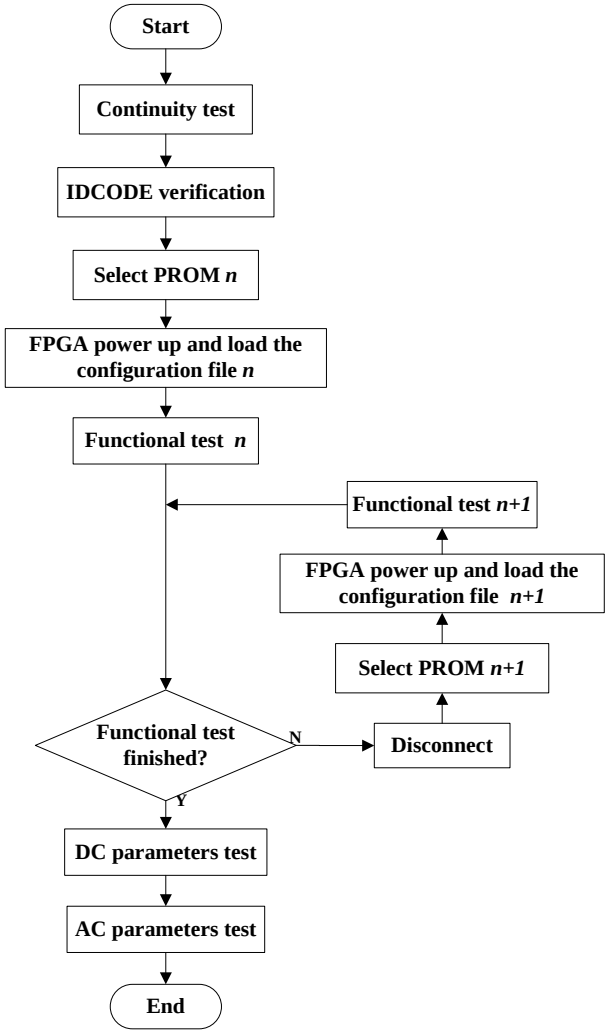

Fig. 3. Flow chart of the FPGA test program for configuration through parallel PROMs

5. CONCLUSIONS

This paper takes Virtex-E FPGA series and V93000 test system for example, presents two different solutions for FPGA auto configuration based on an ATE. The presented methods help implement completely auto reconfiguration without interrupting test procedure that benefit a lot especially in volume automated test.

In-system configuration based on an ATE is accomplished by boundary scan timing and TAP controller. Each different bitstream file for configuration is converted into a configuration pattern. The ATE is responsible for loading the configuration pattern directly into the FPGA. This method is much more efficient yet the conversion of configuration pattern is very complicated. It requires a thorough understanding of boundary-scan timing and the principle of in-system FPGAs programming.

Configuration through parallel PROMs is implemented by improving the hardware connection on the test interface board. PROMs with different configuration files are automatically selected by multiplexer and the Utility Lines of V93000 test system. This method is more flexible that one can update a configuration file any time by simply change a programmed PROM.

This paper aims to provide possible solutions for FPGAs auto re-configuration without specific tools and download cables. The entire configuration-test process can be accomplished by a single ATE without interrupting the test process between every functional test to change the configuration file manually. It provides huge advantage especially in volume automatic tests.

REFERENCES

[1] Huang W K, Meyer F J, Lombardi F. Array-based Testing of FPGAs: Architecture and Complexity. Proceedings IEEE Conference on Innovative Systems in Silicon, 1997. 249-258

[2] XILINX Inc. Virtex-E 1.8V Field Programmable Gate Arrays, 2002: Module 2, 1-8