A PC-Based Algorithm for Testing a Configured PLD Using Boundary Scan Test

Bashar S. Mohamad-Ali and Ziyad K. Farej*

ABSTRACT

A comprehensive algorithm, and a data structure are proposed to test configured programmable logic devices (PLD) through JTAG port (IEEE 1149.1) using boundary scan testing (BST). An exhaustive test vectors set (all possible logic combinations of inputs) is used to test any configured combination logic circuit, so that all possible stuck-at faults are covered. The correct response is stored either from a good configured PLD or from a simulation file. Then the actual response of an PLD is compared to the correct response, which is stored previously. The only limitation to such algorithm is the test time, which is increased exponentially with the number of inputs. This algorithm can be implemented using either a low level language (Assembly language) or a high level language (Visual C, Visual BASIC) on any personal computer via its printer port.

Keywords: PLD Testing, Test Algorithms, Boundary Scan Test.

1. INTRODUCTION

As logic designers switched to use PLDs and field programmable gate array (FPGAs) in their designs, for their many advantages over fixed function ICs. The need for testing these PLDs and FPGAs becomes more apparent. Testing PLDs and FPGAs can be either manufacturing-oriented (carried out by producer), or application-oriented (carried out by user). In manufacturing-oriented test procedure, the PLD or FPGA device is partitioned into its resources (logic blocks, memory cells, interconnections, etc), and each type of resource is tested in certain test procedure. For examples the logic blocks are tested by (Abramovici et al., 2001), the interconnect network is tested by (Renovell et al., 1998), and the RAM memory modules are tested by (Renovell et al., 1998). While in application-oriented test procedure only used resources are tested (Renovell et al., 2000), (Rebaudengo et al., 2002). The application-oriented test procedure requires less test vectors for the same fault coverage, since a limited hardware is to be tested. (Renovell et al., 2000) presented the adoption of a classical test pattern generator considering a modified gate level description of the configured circuit in the FPGA. In that approach the model describing the possible faults is approximate, since it does not consider the faults affecting the values of the memory cells composing each Lock-Up Table (LUT). This drawback has been overcome by (Rebaudengo et al., 2002). They present a fault model to test a configured FPGA device which takes into account the stuck-at faults affecting the memory cells composing the LUT. In both test procedures, all the previously mentioned authors try to reduce the number of test vectors, and to obtain large fault coverage, using different test pattern generation, and redundancy methods. In fact they do not use the exhaustive test vectors due to the unfeasible test time.

In this work an exhaustive test vectors (all the possible logic combinations of the inputs), which detects all possible logic stuck-at faults, is used. However, the configured circuit which can be tested in this procedure should have a limited number of inputs. The number of inputs which can be tested in a feasible test time, is estimated.

2. IEEE 1149.1 JTAG BOUNDARY SCAN TEST

Boundary Scan is possibly the most resourceful test technique which is similar to the In-Circuit Test (ICT) but without physical contact. BST detects the failure location, tests thousands of test points and needs only four lines. After lengthy discussions about the principle of testing
through the stimulation and read-out of integrated thresholds in an Integrated Circuit (IC) in the 1980s, Boundary Scan (IEEE 1149.1) became a standard in 1991. Boundary Scan essentially means testing at the periphery (boundary) of an IC. Besides the core logic and the contact points some additional logic (scan cells) is implemented in a device (Fig. 1). These test points are integrated between the core logic and the physical pins. All Boundary Scan cells are combined in a shift register with parallel inputs and outputs and generate the serial scan path. In each Boundary Scan device a control logic (the Test Access Port (TAP) is integrated to stimulate and read-out the cells. The Boundary Scan device is controlled by four signals. Synchronous actions inside the TAP are executed at the edges of the Test Clock (TCK) signal; the TAP’s single states are fixed at each rising edge depending on Test Mode Select (TMS) signal. Test Data In (TDI) and Test Data Out (TDO) signals represent the input and output of the serial Boundary Scan shift register.

<table>
<thead>
<tr>
<th>Table 1. Total test time for different number of inputs, and different PCs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PC characteristics</strong></td>
</tr>
<tr>
<td>------------------------</td>
</tr>
<tr>
<td>Pentium III, 450 MHz</td>
</tr>
<tr>
<td>Pentium 4, 2.3 GHz</td>
</tr>
</tbody>
</table>

In order to have a clear picture about the Boundary Scan Test, a further discussion is needed for a particular device. Altera MAX 7000S series is chosen for this purpose. Each Scan Cell (Altera Corporation, 2005) consists of two sets of 3-bit registers (Fig. 2). The first set is called capture registers and the second is called update registers. The first capture register can store the signal (PIN_IN) at the pin connected to that cell. The second and the third capture registers can store signals from the internal logic (OEJ, OUTJ). The update registers can transfer the stored signals (PIN_IN, OEJ, and OUTJ) from the capture registers to the internal logic (INJ), to the control pin of the output buffer, and to the input of the output buffer (PIN_OUT) respectively. Three main phases of operations exist in BST; the capture, the shift, and the update phases. In the capture phase, the signals from the pin (PIN_IN), and from the core logic (OEJ and OUTJ) are stored in the capture registers. In the shift phase, new data can be shifted-in through TDI pin, while previous data at the capture registers can be shifted out of TDO pin. Finally in the update phase, the data at the capture register are applied to the inputs of core logic and to the pin. Each phase can be obtained sequentially by applying the appropriate CLK, and the TMS signals to the TAP (Altera Corporation, 2005). The TAP controller is a 16-state machine. At device power-up the TAP controller is at Test.Logic_State, which is the normal device operation. To switch into Boundary Test an appropriate TMS and CLK signal should be applied. An instruction code should be shifted into (Instruction Register) first, then test data can be shifted into (Boundary Scan Register) later. Many instructions are available for both boundary test and configuration; SAMPL/PRELOAD, EXTEST, BYPASS, IDCODE, USERCODE, ..etc. For our Boundary Scan Test the EXTEST instruction is required. Once the EXTEST instruction code is shifted and updated, the Boundary Scan register (cells) is selected. Hence test data can be captured, shifted, and updated.

### 3. THE PROPOSED TEST ALGORITHM

The proposed test algorithm consists of two stages. The first stage involves storing the correct response (Fig. 3). In this stage, an exhaustive test vectors are applied to the inputs of the configured circuit, and the response of the configured circuit is sampled and stored. The exhaustive test vector set is given by:

$$
\sum_{b=0}^{2^{i}-1} (b_{i-1}b_{i-2}b_{i-3}b_{0})
$$

(1)

Where \(b_{i}, b_{i-1}, b_{i-2}, b_{i-3}, b_{0}\) is the binary input test vector, \(i\) is the number of inputs of the configured circuit.
Fig. 1. Boundary Scan Device

Fig. 2. MAX 7000S Boundary Scan Cell (From Altera AN39)
The second stage involves verifying the response of the configured circuit (Fig. 4). In this stage, the same exhaustive test vector is applied to circuit inputs, and the response is sampled and compared to the correct response (previously stored). If there is any difference between the present response and the correct response, a faulty device is declared. Otherwise a functional device is declared.

### 3.1 APPLYING TEST VECTORS

In order to be able to apply test vectors, the boundary scan register should be selected. Selecting boundary scan register is accomplished by a number of steps. Each of these steps requires the appropriate TMS signal (or signals) synchronized with the TCK. For example to advance the TAP controller from RESET state to TEST mode (RUN_TEST_IDLE state), the TMS should be clocked with logic "0", and to advance to instruction mode (SELECT_IR_SCAN state) the pattern "11" should be clocked. Shifting in the instruction code of EXTEST, and updating the instruction, will select the boundary scan register.
scan register (boundary scan cells). Then for each test vector, the data is shifted in, and updated. Since an exhaustive test vector is chosen, the number of test vectors for combinational configured circuit is given by (Hawkins C.F., 1989).

\[ N_{tv} = 2^i \]  

Fig. 4. Verifying the response
Where \( i \) is the number of inputs of the configured circuit.

Each test vector should be shifted in the boundary scan cells together with stuff bits, so that each bit of the test vector is aligned with the input pins of the configured circuit. This process requires many parameters of the device under test, which are usually given in a Boundary Scan Description Language (BSDL) file. These parameters include device name, package type, number of pins, BST instructions codes, identity code, BS register attribute ...etc. Unfortunately, these parameters are different for different devices. Therefore a data structure (database) for each device should be implemented.

### 3.2 THE DATABASE

The database can be either interactively entered by the user, or previously stored within the implemented program. Figure (5) shows the Entity Relation Diagram of the proposed database. The database can be divided into two categories; device database \((D_{db})\), and configuration database \((C_{db})\).

The device database includes the following parameters:

\[
D_{db} = \{ \text{DN, DNP, SPIC, BSCM} \} \tag{3}
\]

Where:
- DN is the device number,
- DNP is the device number of pins,
- SPIC is the SAMPLE/ PRELOAD instruction code,
- BSCM is the boundary scan cells map.

The attributes of the above parameters can be obtained from Boundary Scan Descriptive Language (BSDL) file which is available from manufacturer.

The configuration database includes the following parameters:

\[
C_{db} = \{ \text{NIP, IP}_1, \text{IP}_2, ..., \text{IP}_i, \text{NOP, OP}_1, \text{OP}_2, ..., \text{OP}_o \} \tag{4}
\]

Where:
- NIP is the number of input pins of the configured circuit (positive number).
- \( \text{IP}_1, \text{IP}_2, ..., \text{IP}_i \) are the input pins of the configured circuit (positive numbers representing the used device pin numbers).
- NOP is the number of the output pins of the configured circuit (positive number).
- \( \text{OP}_1, \text{OP}_2, ..., \text{OP}_o \) are the output pins of the configured circuit (positive numbers representing the used device pin numbers).

The above parameters are to be entered by the configuration designer or user.

### 4. FEASIBILITY OF TEST ALGORITHM

In order to have an idea of how many inputs of the configured circuit are feasible in the suggested test algorithm, a test is carried out to determine the test time with different number of inputs, using the printer port of different personal computers. In this test, it is assumed that to shift 1 bit in the PLD through the JTAG pins requires two PC output operations, and one input operation to get 1 bit of the response. Notice that the number of bits which are required to be shifted into the PLD for one test vector does not depend on the number of test vector bits (the number of inputs), but it depends on the number of PLD pins. Hence the total estimated test time for the configured circuit is given by:

\[
T_{test} = T_{sv} \times 2^i \tag{5}
\]

\[
T_{sv} = N_{bsc} \times T_{sb} \tag{6}
\]

Where
- \( T_{test} \) is the total test time for the configured circuit,
- \( T_{sv} \) is the time for applying single test vector,
- \( N_{bsc} \) is the number of boundary scan cells in the PLD,
- \( T_{sb} \) is the time of shifting single bit into the PLD through JTAG pins.

Notice that the time required for initializing the boundary scan test is negligible compared with the time required for applying the test vectors. For the purpose of this test, a program is written in QBasic language to estimate the test time. The Altera PLD (EPM7128SL84) which has 287 boundary cells, is assumed, and the printer port is assumed to be used. The program involves a for-next repetition loop, which repeats (two output, and one input) instructions for a \((2^i \times 287)\) times. The test time is determined by measuring the duration of executing this program, and. The test results are shown in Table (1).
Fig. 5. The Entity Relation Diagram of the proposed database
5. CONCLUSIONS

A comprehensive test algorithm has been suggested to test a configured PLD. The suggested algorithm can be implemented by a suitable high level language. The connection between the JTAG device pins and the personal computer could be through the parallel printer port or a serial port. Feasibility test results showed that, for an FPGA of 84 pin and 287 boundary scan cells, a configured circuit of up to 13 inputs could be tested in one minute if Pentium 3 (450 MHz) PC is used, while 40 seconds if Pentium 4 (2.3 GHz )PC, and XP professional operating system are used.

REFERENCES


