

W8. Hardware Testing Methods

Built-In Self-Test Architectures

1. Laboratory Objective

Configuring and implementing a Built-In Self-Test Architecture

2. Introduction

Errors are defined in relation to the services provided by a system [ALRH04]. The services of a system represent a succession of external states and an error occurs when at least one of the external states suffers a deviation from the correct behavior [ALRH04].

A **fault** is the hypothetical cause of an error and fault tolerance provides the means to achieve reliability and security in computing systems by avoiding system failures in the presence of faults [ALRH04].

3. Theoretical Background

3.1 Built-In Self Test Architectures

The Built-In Self-Test (BIST) error detection method transforms a design into a self-testing architecture, capable of detecting the presence of errors autonomously.

A typical BIST architecture can be viewed on the next page (see Fig. 1). After a brief analysis of the architecture we will be able to identify the following elements:

- Test Pattern Generator (**TPG**) – generates test vectors, which will be connected to the inputs of the Circuit Under Test unit.
- Output Response Analyzer (**ORA**) – analyzes the results of the CUT for error detection.
- Circuit Under Test (**CUT**) – in the case of a combinational CUT, for each test vector applied, a response vector at the CUT outputs is obtained.

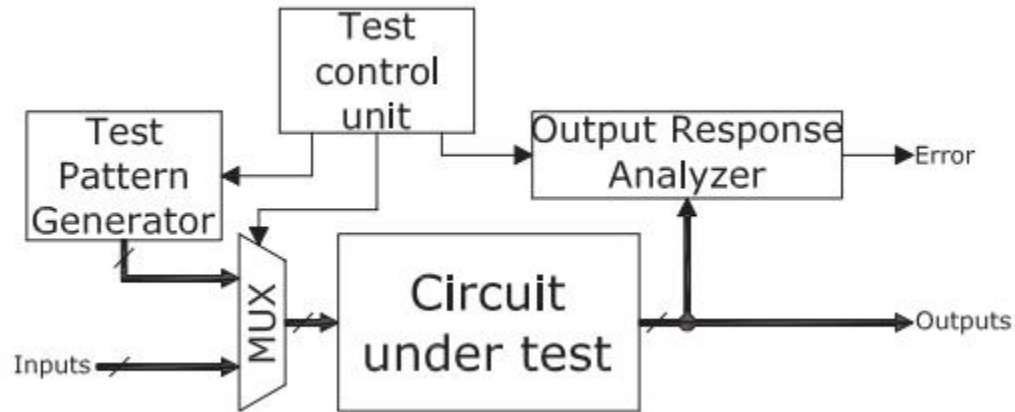


Fig.1 – Built-In-Self-Test Architecture

3.2 TPG Unit

The TPG unit can be built using:

- binary counters, or
- Linear Feedback Shift Registers (LFSRs)

Binary counters generate all the input configurations of the CUT, exhaustively.

LFSR is the conventional mechanism for generating test vectors in BIST structures. They are built as shift registers with a feedback connection, processed by EX-OR gates.

The figure below illustrates a 4-bit LFSR structure:

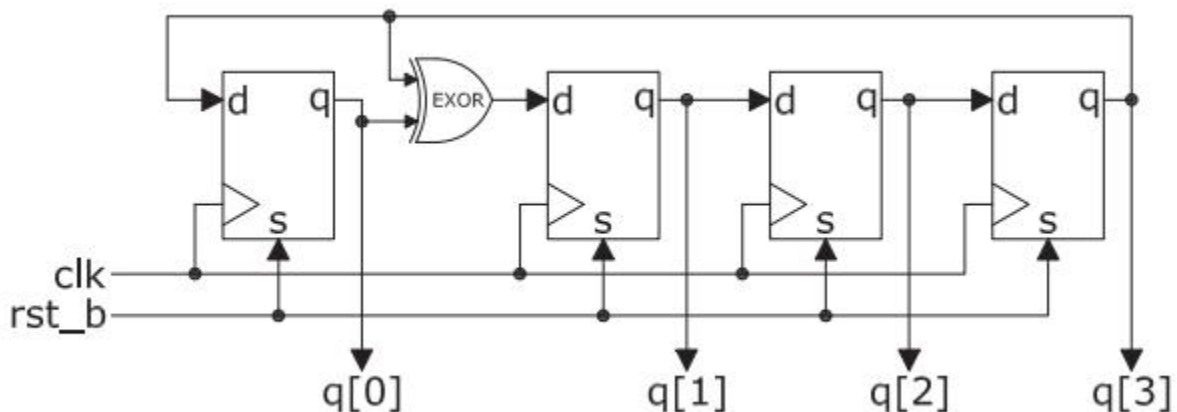

















Fig.2 – LFSR with EXOR feedback path

When initialized with a nonzero vector, an LFSR generates a pseudo-random, repetitive sequence at the output.

For the architecture of Fig. 2, the output sequence, composed of 4-bit vectors, is repeated with a periodicity of 15 (all non-zero 4-bit vectors are generated).

The 15 vectors, generated at the output of the given LFSR, are:

rst_b	clk	q[3]	q[2]	q[1]	q[0]
0	<i>d</i>	1	1	1	1
1		1	1	0	1
1		1	0	0	1
1		0	0	0	1
1		0	0	1	0
1		1	0	0	0
1		0	0	1	1
1		0	1	1	0
1		1	1	0	0
1		1	0	1	1
1		0	1	0	1
1		1	0	1	0
1		0	1	1	1
1		1	1	1	0
1		1	1	1	1
1		1	1	0	1

At the beginning, an LFSR is defined as a vector initialized to 1 at each position, for the value rst_b = 0 and at any clock signal (symbolized d or *). At the rising edge of the clock signal, when rst_b = 1 the first change in the bit vector occurs, q[3] takes over the old value of q[2],

q[2] takes the value of q[1]. At this point $q[1] = q[0] \oplus q[3]$ (see Fig.2), i.e. q[1] = 0. Finally q[0] takes the old value of q[3].

For the next iteration, we will repeat the above procedure, thus generating a number of 15 vectors. The periodicity of the output sequence is noticeable at the 15th iteration mentioned in the table.

3.3 ORA Unit

ORA performs the data compaction (with loss of information) processing all the results of the CUT when it is practiced with the test vectors generated by TPG. At the end of the compaction, ORA provides a signature. The signature is a narrow, fixed-length vector that characterizes the entire set of results.

The signature of a CUT is associated with the TPG unit that generates the inputs for the CUT. The gold signature refers to the signature obtained for a CUT unaffected by defects. It is usually obtained by simulation.

The presence of errors in a CAT is detected by comparing the signature obtained with the gold signature.

The ORA unit can be implemented using:

- counting techniques, or
- signature registers

3.4 Counting techniques

Counting techniques can count either the number of occurrences of a logical value at an output or the number of transitions of an output line. Binary counters are used to count a logical value (0 or 1) at an output.

A transition counter is shown below:

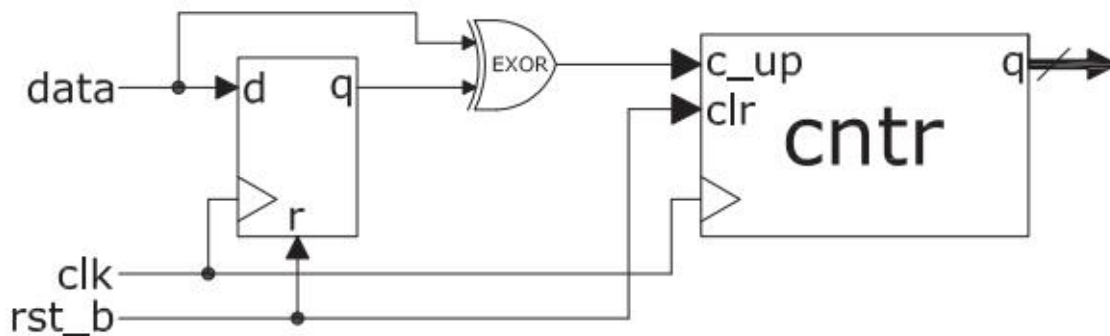


Fig.3 – Transition Counter connected to a Register

A counting unit will connect to each line of the CUT output. Consequently, a 4-bit output requires 4 counting instances. For a single line, the final signature is the contents of the counter after receiving all the bits of that line.

3.5 Signature Registers

A **Single Input Signature Register (SISR)** is built around an LFSR with an additional data entry. The SISR modeled from the LFSR architecture presented above is illustrated below:

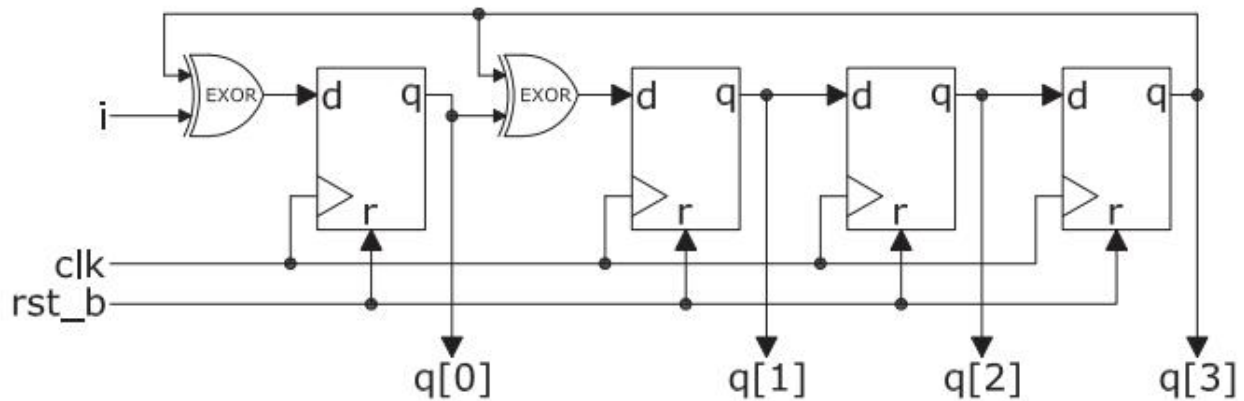




Fig.4 – Single Input Signature Register Architecture

The SISR, compared to the LFSR, starts from the initialization of the bit vector with the value 0 on all 4 positions (the SISR being of rank 4).

A SISR unit will be connected to each output line of a CUT and the final signature represents the contents of the SISR after processing all received bits.

4. Proposed Exercises

Exercise 1. Considering the completed table on page 3 for LFSR, 15 bit vectors will be generated for the SISR, starting from the table header below:

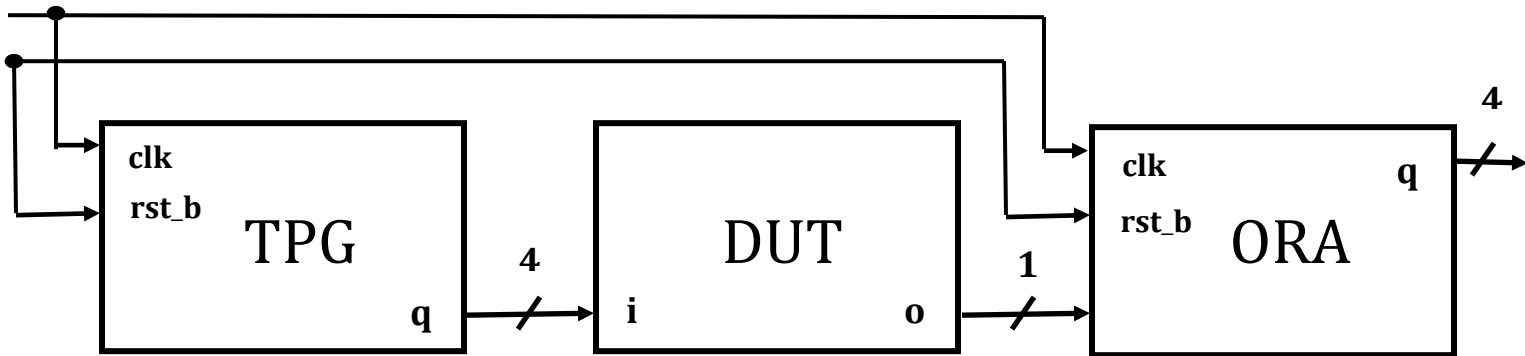
rst_b	i	clk	q[3]	q[2]	q[1]	q[0]
0	d	d	0	0	0	0
1	1	
1	1	

The input vector *i* will receive the following bit sequence [1 1 0 1 1 0 1 0 0 1 1 1 1 0 0]. The bits will be passed in the table on its column *i* from left to right the first bit 1 to the last bit 0.

Task: Complete the table constructed to the last iteration and determine the SISR signature vector.

Exercise 2. Implement, using Verilog language, the SISR architecture presented in Fig. 4.

Exercise 3. Construct, using Verilog language, the following BIST (Built In Self-Test) architecture:



The following elements are known for the given architecture:

- Test Pattern Generator (TPG) – is a 4-bit binary counter.
- Device Under Test (DUT) – is a purely combinational circuit in which the output of the block will be 1 if the input `i` is a multiple of 3 and 0 otherwise.
- Output Response Analyzer (ORA) – the block can be replaced by either a SISR or a transition counter.

Determine the golden signature of the output by simulating the implemented architecture.

5. References

[ALRH04] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, “Basic Concepts and Taxonomy of Dependable and Secure Computing,” *IEEE Trans. Dependable Secur. Comput.*, vol. 1, no. 1, pp. 11–33, Jan. 2004.