# Efficient Test Circuit to Qualify Logic Cells

R. P. Ribas, S. Bavaresco, M. Lubaszewski
PGMicro - PPGC / UFRGS
Av. Bento Gonçalves, 9500, Porto Alegre, 91501-970, Brazil
rpribas@inf.ufrgs.br, simoneb@inf.ufrgs.br, luba@ece.ufrgs.br

A. I. Reis
Nangate Inc.
155-A Moffett Park DriveSunnyvale, CA, 94089-1321, USA
are@nangate.com

*Abstract* - **This work proposes a simple, efficient and easy-to-use test circuit for evaluating and validating any set of logic gates in terms of functionality, performance, power consumption and impact in operation of sub-nanometer physical effects**.

## I. INTRODUCTION

The cell-based approach is definitely the most applied strategy in ASIC design nowadays. This IC design technique is based on the reuse of library cells to build more complex digital circuits. Logic cells generated by software (liquid cells) are also a very interesting and promising way to improve existing cell libraries. Liquid cells can provide efficient in-place optimization (IPO) by adding to the library specific cells which combine different transistor arrangements and sizing [1], and new cell drive strength options [2], for instance.

In both, fixed libraries and liquid cells, a complete electrical characterization of logic gates is required before using them in the IC design flow. The characterization shall be accurately performed by considering different input slopes, output loads and design corners (temperature, power supply and process parameters variations), for timing information (delay and output transition) and power consumption components (static and dynamic). These characteristics are essential for the circuit mapping task, the circuit static timing analysis (STA) and power consumption estimation. For this reason, on-silicon validation of logic cells in terms of functionality and performance is a crucial step before using them into consumer products [3-7].

Moreover, since the continuous process scaling down greatly impacts the circuit reliability, electromigration, gate oxide breakdown, NBTI and DFM rules must be carefully verified in terms of circuit operating degradation and fatal failures [8,9]. This is the reason why the process parameters variability represents today an important design issue and requires statistical static timing analysis (SSTA) to guarantee acceptable parametric yield.

In standard cell libraries, three groups of logic gates co-exist: (1) inverters/buffers; (2) combinational cells and (3) storage elements (latches and flip-flops). Mainly due to the large number of different Boolean functions and driving strength options needed for typical designs, the largest of the three aforementioned groups is naturally the one of logic gates, which is the focus of this work.

To evaluate the cell library quality and perform the electrical characterization, some guidelines have been pointed out in [3], where authors propose that open test structures and logic paths are directly connected to I/O pins for delay measuring. A similar approach based on open structures is proposed in [5]. The main drawback of these approaches is that an external test environment makes somewhat difficult to accurately extract time propagation signals. To overcome that problem the principle of monitoring pulse widths can be used [4]. Generating pulses for delay test is commonly based on the implementation of ring oscillators to reduce the impact of external parasitic capacitances [10,11]. Nevertheless, ring oscillators and single logic paths are not suitable for validating the functionality of combinational gates. Several works propose specific structures for that [3,5,6]. These approaches usually require extensive ATE test pattern generation, while delay and power consumption cannot be precisely obtained. The use of benchmark circuits (open-cores) could be considered to cover both the functionality and the electrical characterization. However, the use of the entire set of available gates (considering different drive strength variations) in the benchmark design is rarely attained, as well as the full logic stimuli of each instantiated cell cannot be guaranteed.

This work presents a new test circuit for complete functional and electrical characterization of a whole set of logic gates. It presents a straightforward operation and measurement procedure. Limited number of I/O pads is required. The use of ATPG tools is significantly reduced. The proposed circuit is useful to evaluate cell libraries in the design environment, to check the accuracy of electrical cell data and the correlation to STA and power prediction, as well as to test on-silicon for physical validation. Moreover, it can also play as a benchmark to compare different layout templates and transistor topologies for the same set of cells. Additionally, the proposed test circuit can serve as a degradation monitoring scheme to evaluate the impact in design of emerging sub-nanometer physical mechanisms, such as NBTI and oxide breakdown, for instance. Finally, a diagnosis operating mode, which allows the configuration of a wide range of different logic paths, turns to be a very useful vehicle for library cell yield learning [12].

The proposed test circuit is based on combinational blocks that reproduce the same input data at the output nodes. Each library cell is placed at least once in one of these blocks and is connected to its primary inputs, as described in Section II. This scheme makes it possible that all input stimuli combinations are applied to these cells, and that the blocks can be directly cascaded to test other cells. Section III presents the circuit architecture which is based on a ring configuration for the synchronous and asynchronous operating modes. Section IV presents the circuit validation, and conclusions are drawn in Section V.

## II. Combinational Blocks

The combinational blocks are built in such a way to guarantee the complete functional excitation of the entire set of gates under test. To achieve that, each block is composed of two stages, as illustrated in Fig. 1: a first stage (with single logic depth) where every instantiated cell 'Ci' is connected to (shared) primary block inputs; and a second stage where the signals 'Wi', provided by the cells in the first stage, recreate the same input logic information at the block outputs. Using such strategy, the full functional verification of the cells in the first stage is achieved by providing at the block inputs all possible input combinations. Furthermore, the same input information will be available at the block output, allowing for easy verification of the correct logic behavior. Additionally, once the output vector of any block provides all signal combinations, it can be applied to test the following cascaded block, as demonstrated in Section III.

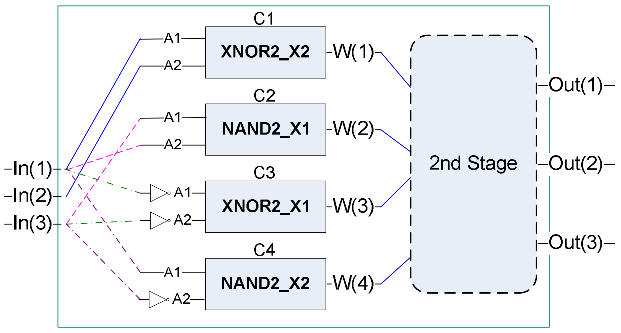

Figure 1. 3-input combinational block: an example.

Under these fundamental principles, the main constructive guidelines for the combinational blocks of the test circuit can be summarized as follows:

a) all combinational blocks present the same number of input and output nodes, which is necessarily equal or higher than the largest number of inputs in a single logic cell;

b) input and output vectors have equal steady state values (this characteristic is explored in the circuit architecture and operating modes, in Section III);

c) every cell has to be instantiated at least once in the first stage of a combinational block to guarantee it is fully stimulated during the circuit operation;

d) cells with the same logic behavior but different drive strengths are considered as distinct cells;

e) the total number of combinational blocks depends on the total amount of cells in the library under test, and also on the number of cells used to compose the first stage of each block;

f) the second stage is synthesized using the cells of the same library under test.

The critical task in generating the combinational blocks is the construction of the first stage. The cells instantiated at this part can be connected directly or inversely to the block primary inputs at any order, as depicted in Fig. 1. For a given number of 'n' block inputs and considering the application of $2^n$ input combinations, the goal is to use the minimum number of instantiated cells ('m') necessary to obtain $2^n$ different values at the W(m..1) internal nodes. Thus, the synthesis process of the second stage will be able to reproduce the block input data at its output nodes.

The procedure to generate the first stage of a combinational block is the following:

1) The cell list can be initially ordered according to different criteria: alphabetic order, number of cell inputs, amount (or rate) of 0s and 1s provided by the logic function, or simply a random order.

2) A first cell is taken from the list and instantiated in the first stage of the block.

3) A second cell is then picked and placed, and its input connectivity is exercised considering input permutation and negation. The goal is to obtain the largest number of different W(m..1) vectors by instantiating this cell. If no new distinct vector can be obtained, then this cell is let down and not considered in this block.

4) A new cell is then evaluated, repeating the step (3).

5) Step (4) is repeated until $2^n$ different vectors are obtained in the W(m..1) nodes, being 'n' the number of inputs/outputs in the blocks.

Once the first stage of the combinational block has been created, the same procedure is applied to the next block disregarding the cells already used to build the previous blocks. The generation of the last block is an exception. Eventually, the remaining unused cells are not enough to conclude the construction of its first stage, and thus already used cells can be applied again.

After the generation of the first stage of different combinational blocks, the information provided by the W(m..1) vectors, as illustrated in Table I, are then used to synthesize the multi-level second stage of every block. A standard mapping tool can be used for this step. The *don't care 'X'* values in Table I are chosen to cover all single stuck-at faults at the W(m..1) internal nodes, thus covering all stuck-at faults at the outputs of the cells under test.

Table I - Truth table used to synthesize the second stage of the combinational block illustrated in Fig. 1.

| In(3) | In(2) | In(1) | W(4) | W(3) | W(2) | W(1) | Out(3) | Out(2) | Out(1) |
|---|---|---|---|---|---|---|---|---|---|
| *no input vector* | | | 0 | 0 | 0 | 0 | *X* | *X* | *X* |
| *no input vector* | | | 0 | 0 | 0 | 1 | *X* | *X* | *X* |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 |
| *no input vector* | | | 0 | 1 | 0 | 0 | *X* | *X* | *X* |
| *no input vector* | | | 0 | 1 | 0 | 1 | *X* | *X* | *X* |
| *no input vector* | | | 0 | 1 | 1 | 0 | *X* | *X* | *X* |
| *no input vector* | | | 0 | 1 | 1 | 1 | *X** | *X* | *X* |
| *no input vector* | | | 1 | 0 | 0 | 0 | *X* | *X* | *X* |
| *no input vector* | | | 1 | 0 | 0 | 1 | *X* | *X* | *X* |
| 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

Obs.: *Don't cares 'X'* are set to '0' logic value, except the one indicated as *X**, which is set to '1'.

Minimizing the number of cells at the first stage, as previously mentioned, decreases the complexity of the second stage of the block. The size of the second stage is proportional to the number of bits of the intermediate code W(m..1). Moreover, the number of block inputs/outputs 'n' also directly impacts the complexity of the second stage.

## III. Circuit Architecture and Operation

The combinational blocks construction, as described in the previous section, guarantees the functional validation of the entire set of logic gates, if each cell has been instantiated at least once in the first stage of a block, and the $2^n$ input signal combinations are applied at the 'n' primary inputs of each block. Once the block primary outputs reproduce the same input values, they can then be arranged in different ways in the test circuit architecture. Long paths can be built, for instance, by cascading the combinational blocks. This way, the primary input values should be observed at the output of the chain in the case of correct functional behavior.

The circuit architecture is presented in Fig. 2. To provide a sequence of test vectors with minimum external intervention, the signals at the end of the chain are reconnected to the circuit primary inputs in a ring configuration. A register barrier (D-type flip-flops) is added into the feedback path to avoid racing. An adder is used to increment the binary vector, also making the circuit to modify the data in the ring and play as a counter. The adder is able to perform '+K' increments providing other than just 1-by-1 counting operation. This allows for different signal transitions that are important to check charging and discharging conditions at intra- and inter-cell nodes.

This basic architecture also includes a comparator and multiplexers to configure different operating (and thus data evaluation) modes. They are:

a) **Synchronous mode** - In this operating mode, the register barrier is controlled by an external clock signal ('Ext_CK'). The maximum circuit operating frequency indicates the worst case path delay, that might be different for different values of 'K'. This mode is useful to evaluate the dynamic power dissipation, at a given operating frequency, and the static consumption (leakage currents), for different steady state values measured at very low frequencies.

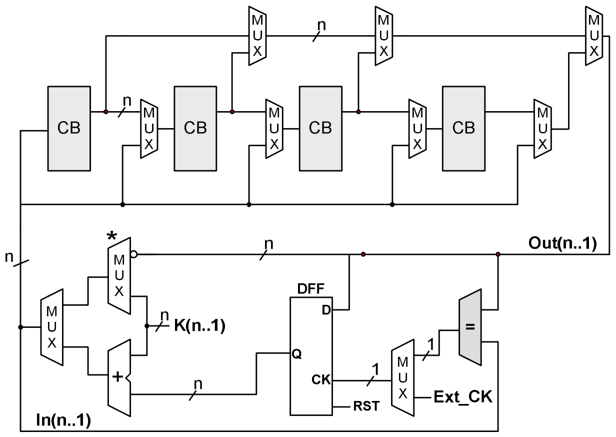

Figure 2. Block diagram of the test circuit.

b) **Asynchronous mode** - In the self-timed ring configuration, the clock signal of the flip-flops is provided by the comparator that checks whether the input data In(n..1) has propagated to the end of the chain Out(n..1). When the same vector applied to the circuit inputs gets to the end of the chain, the comparator switches from '0' to '1', clocking the register. The new data is stored in the register and passed to the adder. It increments the register output and applies the new vector to the chain. At this moment, since In(n..1) no longer equals Out(n..1), the comparator output is back to '0' and remains at this state until the new vector propagates through the whole chain of combinational blocks. In this mode, no external signal is necessary to keep the circuit running and, if a cell is defective, the data at the end of the chain will not equal the data at the circuit inputs. This way, the comparator will not switch to '1' and the self-timed execution will stop. Thus, the correct operation of the circuit, and consequently the cells, can be easily checked by just monitoring the internal clock signal or just one data bit in the ring. Moreover, the asynchronous mode can be run continuously without external intervention, and timing information can be obtained by observing complete '+K' counting cycles. This is useful to detect, for instance, circuit performance degradation along the lifetime cycle.

c) **Oscillation BIST mode** - Since the same binary value of the i-index input is expected to re-appear at the corresponding i-index output of the circuit, when the $i_{th}$-output is inverted before reconnecting to the $i_{th}$-input, a negative polarity logic feedback occurs and the $i_{th}$-path oscillates [11]. According to this principle, in oscillation mode the feedback loop is closed such that at least one of the primary inputs of the chain receives the negation of its previous value. Only one bit from the vector Out(n..1) is selected at a time to provide only one bit oscillation, while the other ones are fixed to the value provided at the input In(n..1). The oscillation condition depicted in Fig. 3 is ensured by the multiplexer marked with '*' in Fig. 2. In the oscillation BIST mode, a large number of different logic paths can be configured and tested for time delay propagation. This mode is useful for monitoring the circuit performance degradation due to aging.

d) **Diagnosis mode** - Multiplexers can be included in the circuit architecture to select part of the combinational chain. Multiplexers at the inputs of each combinational block can select the signal from the previous block or directly from the beginning of the chain, removing the influence of the previous blocks in the ring loop. Similarly, multiplexers at the output of the blocks can send the data from the middle of the chain directly to its end. This way, the chain can be easily reduced to a single block or even none, allowing in this case the verification of the counter and register barrier without the influence of the combinational chain. The possibility to reduce the number of combination blocks under test can be explored for yield learning of cell libraries [12].

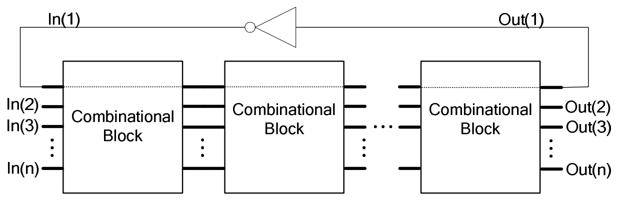

Figure 3. Oscillation BIST path representation.

## IV. Circuit Validation

The test circuit has been validated considering two different sets of logic gates: an open standard cells library containing 64 conventional cells, and another one corresponding to 208 distinct functions with up to 7-inputs extracted from the GenLib_44-6 SIS library. A Java program was developed to generate the first stage of all combinational blocks. The information resulting from the first stage construction was used to create the Boolean equations representing the second stage functionality, to be then synthesized using a commercial mapping tool. Both VHDL and electrical descriptions were verified. In terms of circuit complexity, for the small library (64 cells) a circuit with around 500 cell instances was obtained, while for the library with 208 cells ten combinational blocks were created, leading to a test circuit with approximately 8,000 cell instances. HSPICE simulations were carried out for the different circuit operating modes, and different input data (adder input and multiplexers control) in order to exercise different logic paths and signal transition sequences. To illustrate, Fig. 4 shows the circuit behavior in asynchronous mode when a fault occurrence is detected in a particular gate under test.

## V. Conclusions

An efficient test circuit has been proposed to qualify logic gates of library cells. It is somewhat simple to design and straightforward to operate and measure. Very few I/O pins are required to control the different operating modes and to extract the output data. Each operating mode provides facilities for a wide range of logic transition measurements in several configurations of logic paths. As a result, all cells available in the library have their functionality fully validated, as well as timing and power consumption data are easily obtained for different design corners. This circuit is also quite useful for evaluating the impact of physical effects in the circuit operation, and for yield learning of standard cell libraries.

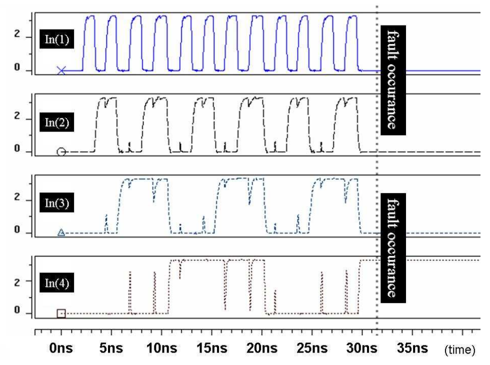

Figure 4. Electrical simulation of fault occurrence during the self-timed counting (asynchronous mode).

## References

[1] R. Roy, D. Bhattacharya and V. Boppana. "Transistor-level optimization of digital designs with flex cells". IEEE Computer, Feb. 2005, vol. 38, no. 2, pp. 53- 61.

[2] P. de Dood, B. Lee and D. Albers. "Optimization of circuit designs using a continuous spectrum of library cells". US Patent 7107551. 2003.

[3] R.-B. Lin, I. S.-H. Chou and C.-M. Tsai, "Benchmark circuits improve the quality of a standard cell library", Asia and South Pacific Design Automation Conference (ASP-DAC), 1999, pp.173.

[4] W. Agatstein, K. McFaul and P. Themins, "Validating an ASIC standard cell library", Third Annual IEEE ASIC Seminar and Exhibit, Sep. 1990.

[5] W. D. Heavlin, "System and method for designing, fabricating and testing multiple cell test structures to validate a cell library", US Patent 5724251, 1998.

[6] N. Salgunan, "Library test circuit and library test method", US Patent 0157141, 2007.

[7] K. S. Kim, S. Mitra and P. G. Ryan, "Delay defect characteristics and testing strategies", IEEE Design & Test of Computers, Sep.-Oct. 2003, pp.8-16.

[8] D. M. H. Walker, "The essential role of test in DFM", IEEE International Test Conference (ITC), 2004, panel 4.4.

[9] S. V. Kumar, C. H. Kim, S. S. Sapatnekar, "NBTI-aware synthesis of digital circuits". Design Automation Conf. (DAC), 2007. pp.370.

[10] S. Long, "Test structures for propagation delay measurements on high-speed integrated circuits", IEEE Trans. Electron Devices, vol. ED-31, no. 8, Aug. 1984.

[11] E. Arabi , H.n Ihs I, C. Dufaza and B. Kaminska, "Digital oscillation-test method for delay and stuck-at fault testing of digital circuits", International Test Conference (ITC), 1998, pp. 91-100.

[12] M. Keim *et al.*, "A rapid yield learning flow base don production integrated layout-aware diagnosis", IEEE International Test Conference (ITC), 2006, paper 7.1.