

PARIS Explained: A Method of Parallel Pattern Fault Simulation for Synchronous Sequential Circuits

Jonathan Harris
ELEC 7250 - VLSI Testing
Dept. of Electrical and Computer Engineering
Auburn University
Auburn, AL 36849-5201
harri34@auburn.edu

ABSTRACT: Described is a method of parallel pattern fault simulation for synchronous sequential circuits that is based on parallel pattern single fault propagation for combinational circuits as described in [1]. This method of fault simulation was proposed in two independent works published in 1991 [2] and 1992 [3] that were a mere three months apart. In this paper this method shall be referred to as PARIS (PARAllel Iterative Simulator) as it is known in [2]. The method exploits the inherent parallelism present in modern microprocessors to speed up fault simulation times using clever techniques and utilizing methods to *look-ahead* during fault simulation to further decrease total simulation time.

1. INTRODUCTION

Due to increased complexity and cost in test development for VLSI (Very Large Scale Integration) chips, it is desirable to find a low cost and fast method of generating tests and simulating these circuits for fault detection purposes. Since fault simulation has been shown to be of quadratic time complexity [4], it is very much important to reduce the amount of time involved in the process of fault simulation. The objective of the PARIS is to reduce the fault simulation time by exploiting the inherent parallelism present in most modern microprocessors. This is accomplished by simulating 32 patterns in parallel. Results obtained by the simulator are compared to that obtained with the PROOFS algorithm [5]. The PROOFS algorithm simultaneously processes 32 faults, however, it is shown that the PARIS simulator provides a significant speed-up under most conditions. It is important to note some basic assumptions that are associated with PARIS. The algorithm is intended for use with faults modeled as single stuck-at faults with circuits containing D-type flip-flops and basic logic gates [2]. It is assumed that the flip-flops have no set or reset inputs and have only one output [2]. In addition, faults on the clock signal are omitted [2].

In this paper, the PARIS algorithm for parallel pattern fault simulation is presented. A discussion is given on the algorithm as it applies to logic simulation

in order to illustrate the basic concept. Following the logic simulation illustration, the algorithm is discussed in its application to fault simulation and how the results are obtained. The results obtained on selected benchmark circuits with the PARIS simulator are then compared to results obtained with PROOFS [5], FAUST [6], and ROOFS [7]. The paper is then summarized and concluded.

2. LOGIC SIMULATION ILLUSTRATION

In order to understand the PARIS algorithm as it applies to fault simulation, it is first important to understand the underlying concept in PARIS as it applies to basic logic simulation. For illustration, the status words, as they are referred to in [2], are only four bits instead of 32 in order to reduce complexity as shown in Figures 1 - 3. The status words are undefined for all but the primary inputs at the beginning of each simulation step [2]. It is important to note that this simulation algorithm uses three state logic (0, 1, X) for the purpose of its simulations [2]. The flip-flop outputs are assigned undefined values (X) at the onset of each step as well. Due to inherent feedback loops present in many sequential circuits, several iterations may be required in a simulation step in order to reach a stable point. During the first iteration levelized-event driven simulation begins at the primary inputs and pseudo inputs and stops at the pseudo outputs. These pseudo inputs and outputs are the inputs and outputs of the flip-flops, respectively, present in the circuit. In the succeeding iterations events are evaluated at the pseudo inputs where an event at a flip-flop causes a change of the status word.

Figure 1 shows the first step in the simulation of a simple circuit for illustration purposes. The bit marked with "C" represents a carry bit, which is the LSB (Least Significant Bit) of the pseudo input in step 2 (in Figures 2 and 3). This carry bit is saved from the previous step in simulation, where it was the MSB (Most Significant Bit) of the status word; the assumption is that the LSB is the right-most bit in a status word. In this example, three iterations are required to reach a stable point in the circuit as is demonstrated in the three iterations shown in Figure 1.

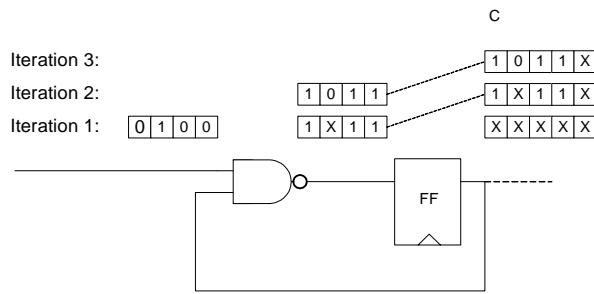


Figure 1. First Simulation Step [2]

Upon completing the first step in the simulation, and thus the first input vector, a second step is begun with a new input vector as is shown in Figure 2. As in the first step, three iterations are required to reach a stable point in the circuit.

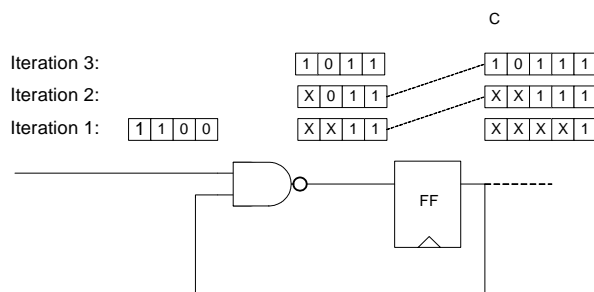


Figure 2. Second Simulation Step Without Look-ahead [2]

The object of the PARIS fault simulator is to reduce fault simulation time thus, three iterations or more per every input vector would require a lengthy test time. A novel approach was formulated in [2] and [3] to look ahead in the simulation and make "educated" guesses about the pseudo inputs and pseudo outputs during simulation in order to reduce the total simulation time. This idea is presented in both [2] and [3], however, in general, the approach is presented much more clearly and made much easier to comprehend in [2] than in [3]. The results are presented with many theorems and postulates in [3] whereas, in [2] a more visually simpler presentation of the method is presented, as is done in this paper.

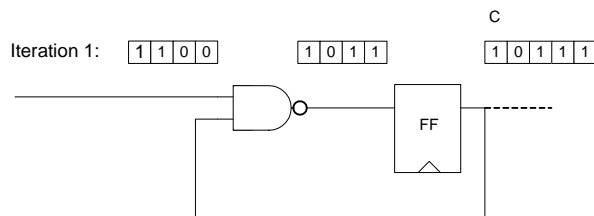


Figure 3. Second Simulation with Look-Ahead [2]

The look ahead portion of the algorithm, referred to as *heuristic look-ahead of signal values* in [2] is demonstrated in Figure 3. The basic idea of the look-ahead strategy is to use "educated" guesses for the values of status words based on the status words from the previous steps in the simulation (except for the first step). One of the assumptions in the algorithm is that the status words of all signals start each simulation step in an undefined state, except for primary inputs and the LSBs at the pseudo inputs. The novel idea in this approach is that "undefined" does not require unknown values, X, be used for status words at the beginning of simulation. The event-driven simulation mechanism is exploited to take advantage of its transformation of changes of a consistent initial state to a consistent final state, which is guaranteed using this method. Therefore, unknown values are not used at the beginning of the simulation step, but the status word is left unchanged from the previous simulation step.

The ingenuity in this approach allows simulation time to be greatly reduced since the method creates two large advantages over previous sequential fault simulation algorithms. The time required to reinitialize all status words at the beginning of each simulation step is eliminated, and, in addition, the number of iterations is frequently reduced for most circuits. Since status words from the previous simulation step are used for the status words in the current simulation step, there is no need for initialization and the previous status word is typically a much better guess at the resulting status word than the unknown values. As is demonstrated in Figure 3, the number of iterations required to reach a stable point in the circuit is reduced from three to one.

Another point of inadequacy in many fault simulators is the unnecessary multiple evaluations of circuit components during simulation. The PARIS simulator partitions the circuit into three parts, A, B, and C [2]:

- All primary inputs belong to part A.*
- A components with all predecessors in A is also part A.*

- The definition of C is similar:*
- All primary outputs belong to part C, except those in part A.*
- A component with all successors in C is also in part C.*

- Finally, part B is as follows:*
- All components not included either in part A or part C belong to part B.*

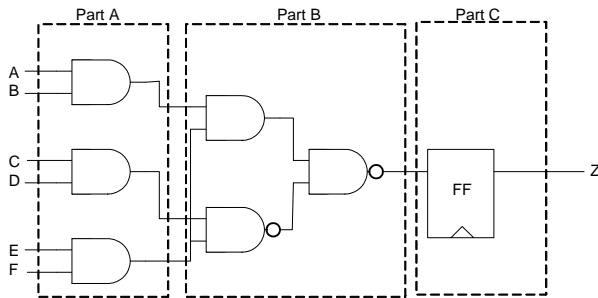


Figure 4. Circuit Partitioning

Figure 4 demonstrates the partitioning a simple circuit using the method described in [2] for the PARIS algorithm. As it is shown, it is clear that parts A, B, and C do not overlap and are independent of the other parts. In addition, it is clear that there are no feedback loops in parts A and C. PARIS takes advantage of this factor to evaluate parts A and C only once during simulation since these parts are effected by an event only once during a simulation step [2]. Part B of the circuit is handled slightly different than parts A and C. In part B of the circuit, only gates are leveled whereas in parts A and C, the flip-flops and gates are leveled [2]. In addition, events are stopped at pseudo outputs in part B until the next iteration is begun [2]. Therefore a simulation step in the PARIS simulator consists of three operations [2]:

1. Simulate part A.
2. Iterate simulation of part B until no more changes occur.
3. Simulate part C.

From evaluation of the simulation steps it is evident that only part B of the circuit requires iterations during simulation to reach a stable point and, therefore, is the only part of the circuit that requires the look-ahead algorithm described previously. Because of this fact, the PARIS simulator does not perform its heuristic look-ahead on parts of A and C [2].

3. FAULT SIMULATION

Fault simulation performed using PARIS basically follows the same procedure as in the logic simulation. Once the logic simulation is performed on the circuit, fault simulation is begun. Initial events are generated for a targeted fault site in the circuit and for the pseudo outputs (flip-flop outputs) where the MSB (Most Significant Bit, left most bit) of the input word was affected by the fault effect at the end of the previous step [2]. A further method of choosing the status words was also derived in [2]. The status word is chosen based on an assigned number. Based on this

number, either the current good status word is chosen or the faulty output status word from the last step is taken [2]. This is decided upon based on the number of differences between the good and faulty output word at the end of the previous step [2]. If the differences between the good and faulty words number smaller than 5, PARIS chooses the current good status word from the previous step, otherwise the current faulty status word is chosen [2]. This is done in order to aid in heuristically looking ahead to find the next correct status word thereby reducing the fault simulation time in a manner similar to that used in the logic simulation. The disadvantage of this approach is in the case where the wrongly guessed faulty output status word causes a resimulation of the current step [2]. The resimulation is necessary to correct the wrongly guessed status word and causes longer simulation time. According to [2] however, this case is extremely rare and the assumption is that the resimulation time is negligible due to the rarity of the occurrence.

4. COMPARISON OF RESULTS

To illustrate the effectiveness of the PARIS simulator, the algorithm was compared to other current algorithms of the day, ROOFS, PROOFS, and FAUST. Table 1 shows the results using deterministic vectors for selected ISCAS '85 and ISCAS '89 benchmarks [2]. The simulations were performed on a Sun 3/60 platform and the simulation times recorded relative to the CPU-seconds of the microprocessor. As can be seen in the table, PARIS provides a significant speed-up in fault simulation time for most of the benchmark circuits tested. With the exception of benchmark circuits s298, s349, s713, s832, and s35932, PARIS outperformed ROOFS, PROOFS, and FAUST by at least 3 times and as much as 18 times [2]. From evaluation of the results, it is evident that PARIS does not work well for smaller sequential circuits or for very large sequential circuits. The analysis would be that for average to medium size circuits, PARIS is the simulator of choice.

5. CONCLUSION

In this paper, a method of parallel pattern fault simulation (PARIS) has been presented. The method as described in [2] is a fairly simple yet novel approach to performing fault simulations on sequential circuits. The method has been described in its application to logic simulation for comprehension of its concept and then described as it applies to fault simulation. The method uses an interesting look-ahead approach to make very good "educated" guesses of future values in the circuits being simulated such that simulation times

can be drastically reduced. In comparison to other algorithms for fault simulation at the time of the invention of this concept, the PARIS simulator outperforms the other methods in the majority of circuits (as presented in Table 1). Suggestions for improvement would be to improve the algorithm for smaller circuits and for very large circuits where, by observing Table 1, it is shown to perform much slower than comparable algorithms.

Circuit	Test Vectors	Fault Cov.	Fault Simulation Time			
			ROOFS	PROOFS	FAUST	PARIS
c432	75	99.24	7.13	5.25	1.33	0.62
c499	71	98.94	5.82	3.65	1.33	1.32
c880	106	100.00	11.07	11.07	2.28	1.38
c1355	145	99.49	23.63	18.03	4.43	4.38
c1908	186	99.52	61.4	35.1	13.47	5.3
c2670	214	95.74	67.95	50.6	12.2	4.13
c3540	199	96.00	99.15	63.67	16.4	8.7
c5315	308	98.90	189.12	126.8	34.32	10.33
c6288	25	99.24	116.74	42.88	27.55	61.67
c7552	398	98.25	282.02	182.17	58.12	20.87
s298	162	85.71	25.77	11.45	6.68	8.03
s349	91	95.71	15.98	5.67	4.45	6.17
s526	754	75.32	456.12	166.25	120.72	94.35
s713	107	80.90	16.22	9.68	3.93	4.62
s832	377	81.38	96.72	78.28	22.25	23.8
s828	137	29.64	108.82	48.73	27.65	7.42
s1238	349	94.69	60.95	49.87	10.73	3.05
s1494	469	91.10	283.4	168.43	76.18	62.32
s5378	408	74.02	619.07	421.52	172.7	56.55
s35932	86	87.99	1999.2	1356.55	520.82	7597.76

Table 1. PARIS Simulation Time Comparisons [2]

REFERENCES

- [1] J. A. Waicukauski, E.B. Eichelberger, D.O. Forlenza, E. Lindbloom, and T. McCarthy, "Fault Simulation for Structured VLSI," *VLSI Systems Design*, pp. 20 - 32, Dec. 1985.
- [2] N. Gouders and R. Kaibel, "PARIS: A Parallel Pattern Fault Simulator for Synchronous Sequential Circuits," *ICCAD 91 Digest of Technical Papers*, pp. 542 - 545, Nov. 1991.
- [3] C.-P. Kung and C. -S. Lin, "Parallel Sequence Fault Simulation for Synchronous Sequential Circuits," *Proc. of European Conference on Design Automation*, pp. 434 - 438, March 1992.

- [4] P. Goel, "Test Generation Costs Analysis and Projections," *Proc. IEEE Design Automation Conference*, pp. 77 - 84, 1980.
- [5] W. Cheng, J.H. Patel, "PROOFS: A Super Fast Fault Simulator for Sequential Circuits," *Proc. European Conference on Design Automation*, pp. 475 - 479, 1990.
- [6] R. Kaibel, "Automatische Testgenerating fur kombinatorische Schaltungen," Ph.D. Thesis, University of Duisburg, 1991.

PEER REVIEWERS

- [1] Srinivas Garimella
- [2] John Sunwoo