CONF-801005--1

Delay Modeling in Logic Simulation

by

John M. Acken and Lawrence H. Goldstein

ABSTRACT

As digital integrated circuit size and complexity increases, the need for accurate and efficient computer simulation increases. Logic simulators such as SALOGS (SAndia LOGic Simulator) which utilize transition states in addition to the normal stable states, provide more accurate analysis than is possible with traditional logic simulators. Furthermore, the computational complexity of this analysis is far lower than that of circuit simulation such as SPICE. An eight-value logic simulation environment allows the use of accurate delay models which incorporate both element response and transition times. Thus, timing simulation with an accuracy approaching that of circuit simulation can be accomplished with an efficiency comparable to that of logic simulation.

DISCLAIMER

INTRODUCTION

As developing technology enables designers to produce larger, denser, and more complex integrated circuits, the importance of simulation as a tool in verifying these integrated circuit designs increases. However, the very size of these circuits makes simulation using such computer programs as SPICE infeasible. Logic simulators may be used to analyze much larger circuits than circuit simulators can handle, but the resulting reduction in accuracy can mask potential timing hazards. As an approach to the problem of generating sufficiently accurate yet computationally tractable timing simulations of complex LSI circuits, this paper will present a delay element model sophisticated enough to support reasonably accurate timing simulations in a logic simulation environment.

SIMULATION

Circuit simulation provides a detailed analysis of the voltages and currents within a circuit. This level of accuracy is required for a careful examination of critical components. Logic simulation merely presents logical activity, enabling a less accurate analysis of the behavior of an entire LSI circuit. Over a period of time, a wide range of accuracies have become available in logic simulators depending upon the number of logic states used.

Since the earliest logic simulators evaluated Boolean equations, they had two states (0 and 1).[1] An undefined state (*) was soon added to model the output of an uninitialized flip-flop, a node the logical value of which is unknown.[2] Simulation of high-impedance switches such as those used in bus-oriented structures, required the addition of an entirely new state--the high-impedance state.[3] The H state (high impedance) is also used in the modeling of an MOS pass transistor or transmission gate. The high-impedance state is the nondominant value when two outputs are tied together in a WIRED-OR or a WIRED-AND configuration. Thus the low-frequency logic operation of digital circuits could be completely analyzed using four stable logic states. As more accuracy was needed in the higher-frequency simulation of LSI circuits, additional states[4] were developed to model transient behavior. They include negative slope (D), transient undefined (X), transition to high impedance (A), and positive slope (U). This eight-value simulation in such computer programs as SALOGS (SAndia LOGic Simulator), with explicit representation of transitions, allows delay modeling required for accurate timing analysis of a complete integrated circuit.

MODELING DELAYS

Timing analysis of a circuit with a logic simulator requires the modeling of delays. Logic simulators based on the evaluation of Boolean equations need no sophisticated delay models.[1] A unit or one-time step delay was introduced[2] to avoid hazards and problems with feedback loops. To model time more accurately in later simulators, variable integer delays were introduced.[5] To model delay through different paths, a delay element with separate rise and fall time delays was introduced.[6] Another approach was to specify the delay and an ambiguity region, allowing best and worst case simulation simultaneously.[7] One author even used a statistical approach to model propagation delay with a mean and variance.[8] However, more accurate modeling of delays is needed for the timing analysis of very large-scale integrated circuits.

An accurate delay element model must be capable of describing the activity on the output node of a circuit as a function of the input to that circuit and time. For any stimulus on the input of an element, the output change can be divided into two components: a response time and a transition time. An eight-value logic simulation environment allows this division by the explicit representation of transition states. For example, the response time of an element to an upward transition on the input consists of the interval from the initial input rise (U-state) until the output begins to respond. The rise time is measured from the initial output rise (U-state) until the output is stable (1-state). Similarly, the response time to a downward transition on the input is the time from when the input begins to fall (D-state) until the output begins to change, and the fall time is the time from when the output begins down (D-state) until it is stable (0-state). These four times can be clearly related to MOS transistor operation. Response times represent the delay from an input voltage change to a significant change in device current. The rise or fall time begins when the device output voltage starts to change and lasts until the device is either completely on (output = 1 state) or completely off (output = 0 state), respectively.

As an example, consider the depletion mode n-channel transistor in Figure 1. The source is grounded, the gate is the input, and the drain is the output. Initially, the gate is at ground (0 state) and the output is floating (H state), Figure 1a. The entire area under the gate is depleted of n-type carriers. As the gate voltage is increased (U state), the area under the gate becomes intrinsic, but the output is still floating (H state), Figure 1b. When the gate voltage exceeds the threshold voltage

(1 state), the channel begins to form and some current is seen at the drain (D state), Figure 1c. Finally, with the gate voltage high (1 state), the channel is completely formed, the drain is connected to the source, and the output is grounded (0 state), Figure 1d. Using a response time/transition time delay of this type, timing simulation accuracy can be achieved for elements at various levels of complexity in a hierarchical, eight-value logic simulator.

EXAMPLE

As a further example, circuit simulation using SANCA (DEC 10 version of SPICE 2) and eight-value logic simulation using SALOGS will be compared. Figure 2 illustrates three representations of an inverter: the CMOS transistor circuit (Figure 2a), the SANCA model with parasitics (Figure 2b), and the logic diagram with accurate delay element (Figure 2c). Figure 3 presents the SANCA simulation for a CMOS metal gate inverter with VDD = 10V, temperature, 27°C, and a load capacitance of 10 pf,[9] where t_{RR} is the rising response time, t_F is the output fall time, t_{FR} is the falling response time, and t_R is the output rise time. Figure 4 contains the analogous eight-value logic simulation with 2 ns time steps. This example demonstrates that eight-value simulation with accurate response time/transition time delay modeling can produce accurate timing simulations with a computational complexity comparable to that of logic simulation.

CONCLUSIONS AND EXTENSIONS

Large scale integration has led to greater use of computer simulation by integrated circuit designers. The complexity of today's circuits has made circuit simulation of a complete chip infeasible. Although logic simulation is feasible, it requires some additions to support accurate timing analysis. Within an eight-value logic framework, modeling of the components of delay is straightforward and accurate. The Sandia Logic Simulator (SALOGS) is a working computer program which implements a response time/transition time delay model. Within the framework of a complete CAD system, a program could be developed to calculate chip delays from layout information and input these values to SALOGS. Furthermore, in a fault simulation environment, nonclassical delay faults could be analyzed with a great deal of accuracy and flexibility. Thus the use of a response time/transition time delay model in an eight-value logic simulation environment enables the logic designer to perform timing simulations of entire LSI circuits with a computation time comparable to that of less accurate logic simulations.

REFERENCES

1. Breuer, M. A., "Techniques for the Simulation of Computer Logic," Comm. of the ACM, Vol. 7, No. 7, pp 443-446 (July 1964).

2. Jephson, J. S., McQuarrie, R. P., and Voqelsberg, R. E., "A Three-Value Computer Design Verification System," IBM Systems, pp 178-188 (June 1969).

3. Crist, S. C., "A Tri-State Logic Family," Proceedings of the Eighth International Symposium on Multiple-Valued Logic (May 1978).

4. Acken, J. M., Stauffer, J. D., "Part 1: Logic Circuit Simulation," IEEE Circuits and Systems Magazine, Vol. No. 1, pp 9-13 (March 1979).

5. Ulrich, E. G., "Exclusive Simulation of Activity in Digital Networks," Comm. of the ACM, Vol. 12, No. 2, pp 102-110 (February 1969).

6. Moen, D. M., Simulation of Digital Logic Circuits, Masters Thesis, University of Arizona, 1976.

7. Szygenda, S. A., Rouse, D. M., Thompson, E. W., "A Model and Implementation of a Universal Time Delay Simulator for Large Digital Nets," Proceedings of the Spring Joint Computer Conference, pp 207-216 (1970).

8. Magnhagen, B., Flisberg, R., "A High-Performance Logic Simulator for Design Verification," Proceedings of the Summer Computer Simulation Conference, pp 724-726 (July 1976).

9. Brown G., et al., Sandia Laboratories Metal Gate CMOS Design Handbook, Internal Document, Sandia Laboratories, September 1979.

JMA:lm:4536A:04/30/80X

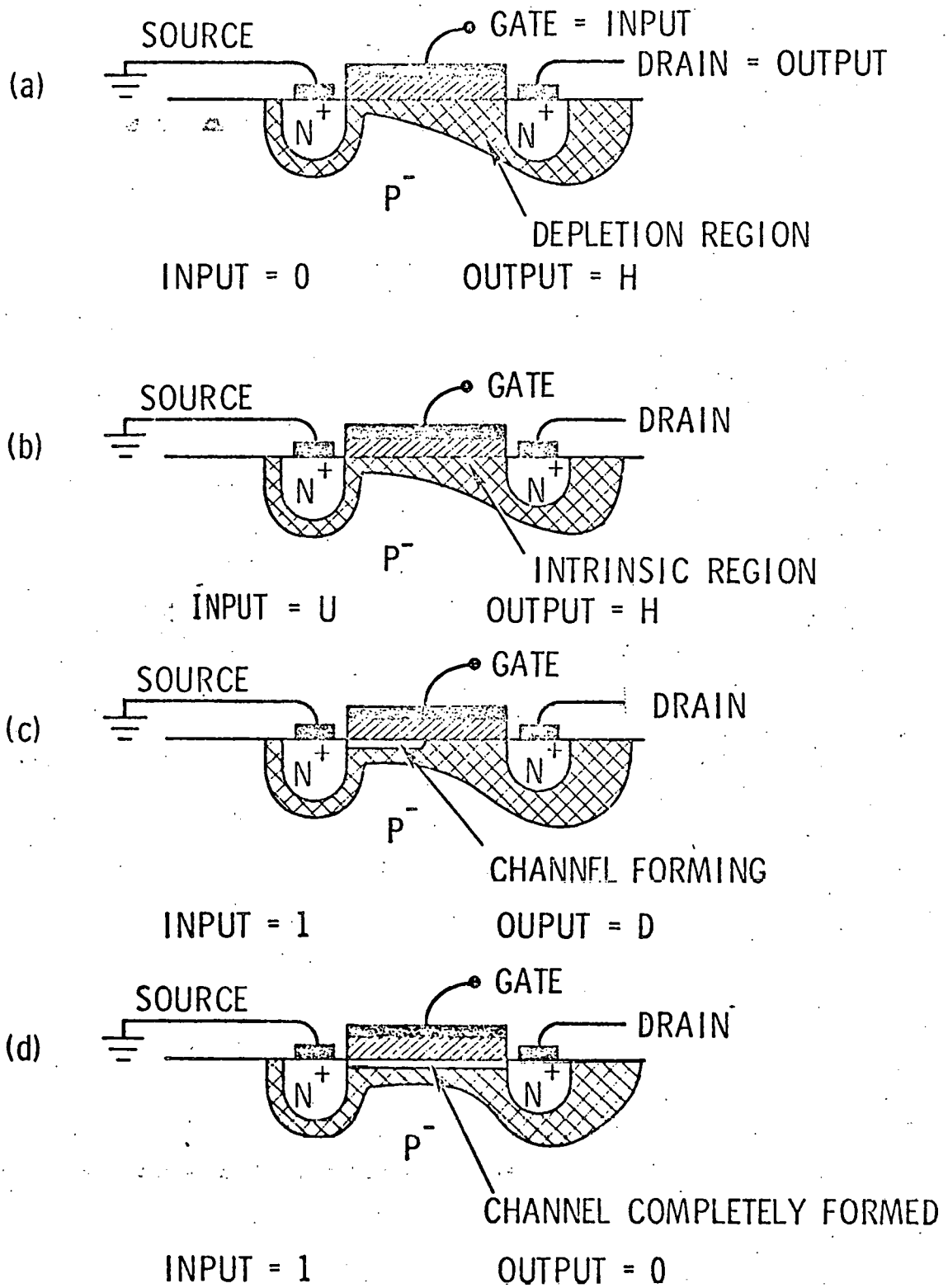
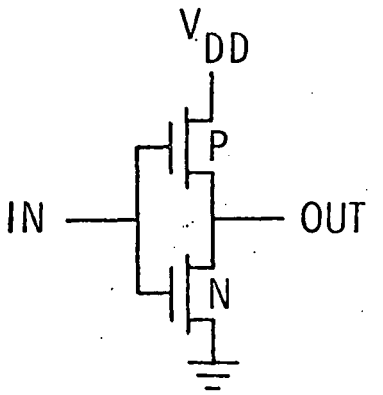
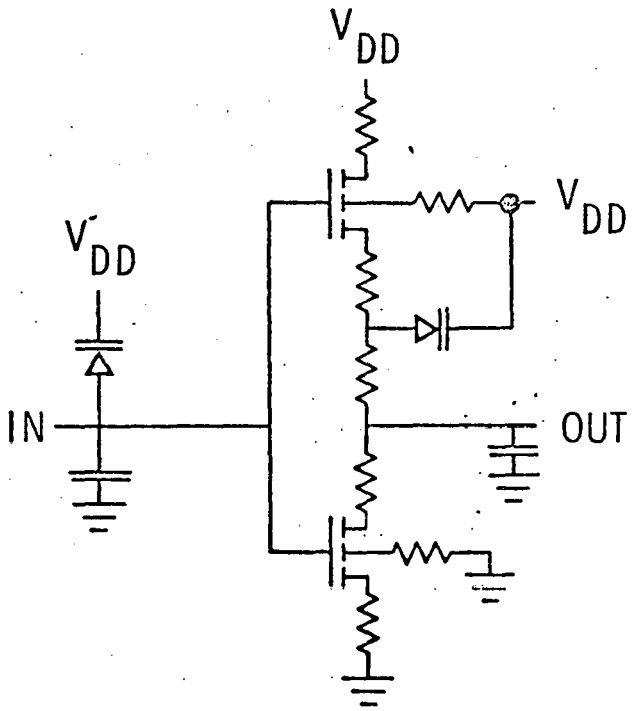


FIGURE 1. MOS TRANSISTOR



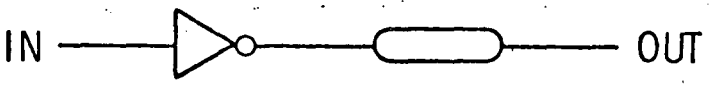
CMOS INVERTER

(a)



SANCA MODEL

(b)



SALOGS MODEL

(c)

FIGURE 2. INVERTER REPRESENTATIONS

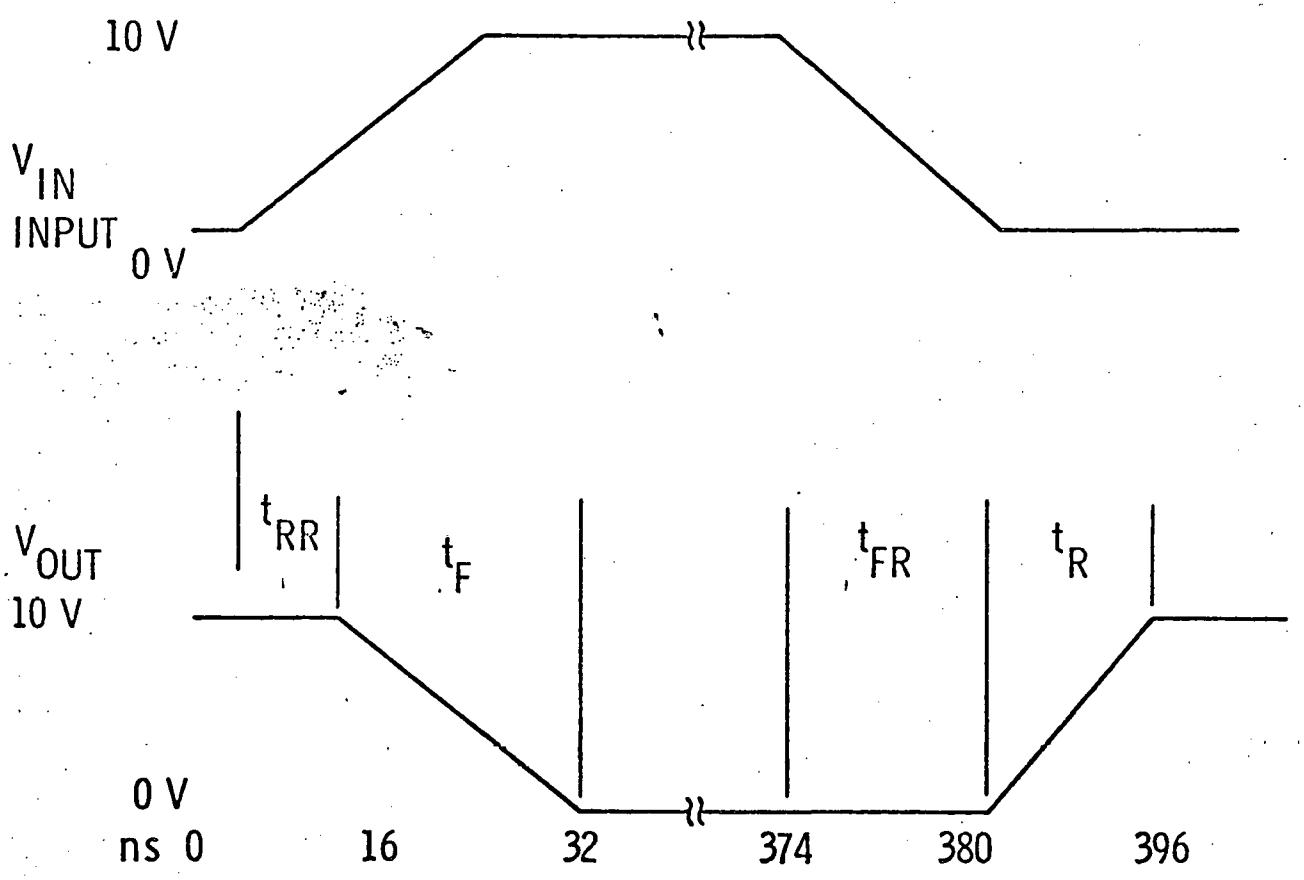


FIGURE 3. SANCA CIRCUIT SIMULATION

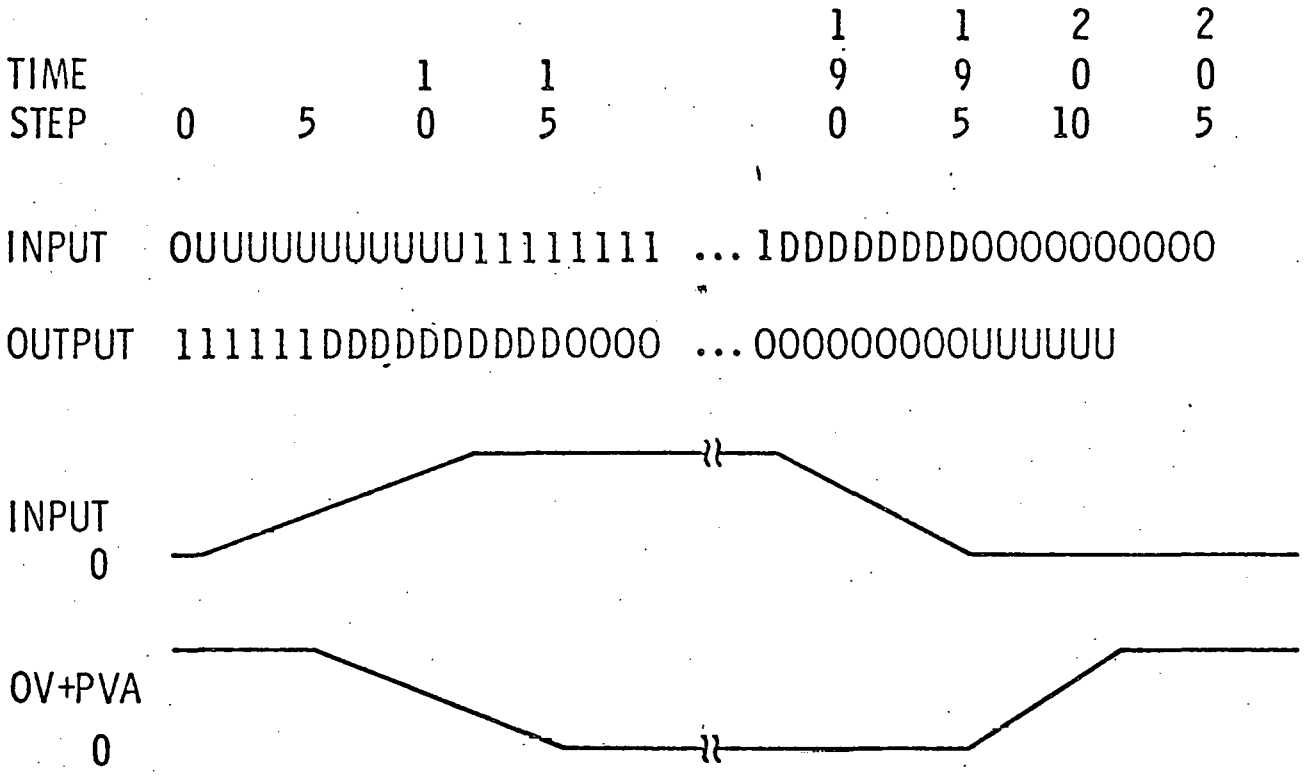


FIGURE 4. SALOGS LOGIC SIMULATION