ILL

TN #: 238702

ILL #: 226416476

CUY

System ID: OCLC

Shipping Option: Article Exchange

Patron:

EFTS: No

CUY
UC Berkeley Libraries
University of California - Berkeley (CUY)

133 Doe Library - ILL Svcs. - Borrowing Unit
Berkeley, CA 94720-6000

Ariel:
Odyssey:
e-mail: cuy@library.berkeley.edu
E-MAIL: CUY@LIBRARY.BERKELEY.EDU
Fax: 510-643-8476
Phone: (510) 643-6152
Maxcost: 36.00IFM

Comments:
Notes: Caiasoft Error: Item is already present on the External Retrieval Queue
Need by: 09/16/2024

Journal Article

Journal Title: Proc. IEEE International Conference on Computer Design

Volume: 1 **Issue:**
Month/Year: 1983 **Pages:** 584-587

Article Author: BURGGRAFF T BURGGRAFF T

Article Title: The IBM los gatos logic simulation machine hardware

Imprint:
Item #:
Call #: 39072028215774
Location: Libraries Annex: Non-Circulating

Copyright Charges:

Date Received: 7/22/2024 9:32:35 AM

Status:________________

Date Cancelled:________________
Reason Cancelled:________________

Date Sent:________________
Number of Pages:________________

University at Buffalo Libraries, UC Berkeley Libraries
BUF NYUSBU
716-645-1334 libsharing@buffalo.edu

Lending - Lending

The IBM Los Gatos Logic Simulation Machine Hardware

Ted Burggraff, Al Love, Richard Malm, Ann Rudy

IBM Los Gatos Laboratory, Los Gatos, California

ABSTRACT

The IBM Logic Simulation Machine, also known as the LSM, is a high speed hardware Logic Simulator that simulates 64,512 logic expressions at a rate of 640 X 10^6 expressions per second. The LSM is a highly parallel system, composed of 64 processors operating simultaneously, each simulating part of a logic design. This paper presents the LSM architecture and discusses the decisions and changes made during development of the LSM.

INTRODUCTION AND OVERVIEW

The LSM is a complete computer architecture designed to perform logic simulation at high speed. It is not, however an exceedingly complex machine. It gains its speed through a highly parallel and modular architecture. The LSM's capacity can be increased incrementally with no speed degradation, and in most cases the addition of capacity can actually reduce simulation time by providing more processors among which a design can be spread.

Use of the LSM is made easy by an extensive software support package. The software package and other simulation facility considerations are discussed in three accompanying papers[1 2 3]. This paper describes only the LSM hardware architecture.

The LSM architecture provides specifically for:

- Simulation of Random Logic, PLAs, Memory.
- Variable delay simulation. Each gate can have a unique rise and fall delay time, from 1 to 256 times the basic unit of delay.
- Dotted Logic Gates with outputs tied together (dotted) can be simulated.
- Net activity trace The hardware can determine which gates in a design have changed value during a simulation. This may help determine the coverage of test patterns being used to stimulate the model.
- Conditional processing based on "events." When specified values for specified nets occur, simulation can be stopped.

Simulation in the LSM takes place in terms of a simulation time step which is one basic cycle through the LSM. During each simulation time step, all logic gates in the design are simulated, thus it takes many simulation time steps to simulate one real machine cycle.

As currently configured, the LSM can simulate 64,512 logic gates at a rate of 640 X 10^6 gates per second. This configuration consists of 64 processors, an inter-processor switch, and a host and an interface computer. The processors are distributed in the following manner:

1 required Control processor
x Logic Processors
y Array Processors
$x + y \leq 63$

The LSM is modular and flexible, allowing for low entry costs and incremental expansion of the system. The system can be configured to meet the needs of the simulation at hand, alteration of the configuration being simply a matter of plugging in or disconnecting boards.

The LSM can be extended indefinitely by adding more processors and a larger switch, and thus, can grow to accommodate future larger designs. There is no theoretical limit to the capacity it can obtain.

Figure 1 shows an overview of the LSM system. Each of the components is fully described below.

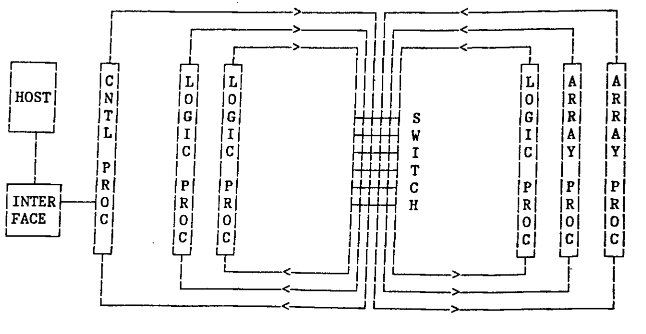

Figure 1. LSM Architecture, 6 Processor Configuration

Physically, the 64 processor LSM fits with its power supplies on six 19 inch square by 5 feet high racks. Each rack draws 190 amps at 5 volts. The total system contains approximately 42,000 TTL modules, 136,000 nets, and 612,000 wire points.

LOGIC PROCESSOR

The Logic Processor is the fundamental unit of the LSM and performs the actual simulation of a design. Sixty-three identical logic processors are incorporated into the LSM configuration. They operate in parallel, each stepping through its memories and simultaneously simulating a part of the total design.

The logic processor contains an instruction memory, a data memory, a logic unit, and a delay value memory. Figure 2 shows the basic structure of a logic processor.

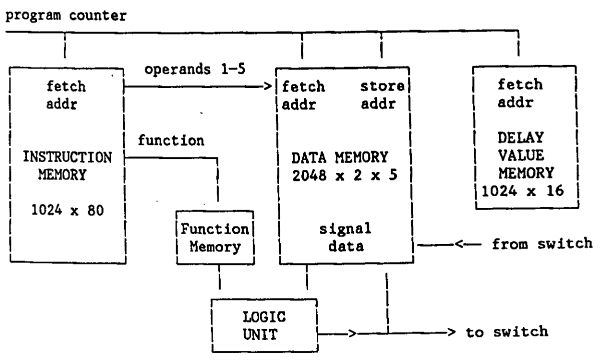

Figure 2. Overview of a Logic Processor: showing one of the two alternating data memories

The **instruction memory** contains 1024 80 bit instruction words. Each instruction word contains a function opcode, five input operand addresses, and other control values. The LSM executes only five input functions; any gate in a logic design with more than five inputs is decomposed into a sequence of five input functions.

Two **data memories** contain 2048 2 bit words each. The input operands addressed by the instruction word are fetched from the input data memory and results of the function evaluation are written to the output data memory. The two memories alternate as input and output memories. There are actually five copies each of these memories, each copy containing identical data, to facilitate the operand fetch. Addresses 0 through 1023 of the data memory contain signal data values calculated in the processor in which the data memory resides. Addresses 1024 through 2047 contain output from other processors.

The **function memory** contains 1024 64 bit words. Each bit in a word is an entry in a 64 entry truth table describing the output of a six input function. (One extra input is allowed for internal chaining of five input functions.) This means that the LSM can simulate up to 1024 native logic functions in each logic processor. The truth tables are fully programmable.

Simulation occurs as follows. The model and the test patterns containing model stimuli have already been loaded into the LSM processors. The model is distributed among the processors as determined by the LSM software and the number of instructions per logic processor (up to 1024) is known. Let this number be n.

An instruction is fetched from the instruction memory. The operands are fetched from the input data memory as addressed by the instruction. The function code and the operands are sent to the **logic unit** where the function is evaluated. The result (output) is sent to the inter-processor switch and results from other processors are received from the switch. The data are latched into the output data memory.

One fetch, evaluate, store cycle is called an instruction step. Instruction steps are executed in a seven stage pipeline, one instruction following another in the pipeline and one completing execution every 100 nanoseconds.

The required number (up to 1024) of instruction steps is executed, in this case n of them, and this makes up the simulation time step. Thus, the length of a simulation time step is not fixed for all simulations but depends on the design being simulated, the number of processors in the LSM configuration, and the distribution of the design in those processors.

The **delay value memory** contains rise and fall delay times associated with each instruction in the instruction memory. Thus, it contains 1024 words, each of which is 16 bits long. Eight bits each are used to represent the rise and fall delay time which range from 1 to 256 units of delay. A simulation time step is the basic unit of delay.

The minimum delay in the LSM is one unit, or one simulation time step, since results of the instructions in one time step are not available until the following time step. This is equivalent to a unit-delay simulator.

When more than one delay unit is specified for an instruction in the instruction memory, the output of the logic unit is not written to the output memory but is retained until the specified number of time steps have elapsed. The count for delay and the value being delayed are held in intermediate memories.

The instruction, output data, and delay memories are all accessed sequentially by the same program counter. No loops or branches occur. The input data and function memory are addressed by fields in the instruction word.

The input data and output data memories alternate in function each simulation time step, so that the memory that was written with new output values during the previous simulation time step is read during the current one. Thus arises the inherent unit delay propagation of output values in the LSM.

Also, because output values are always available for input the next simulation time step, no special ordering of instructions in the instruction memory is required to provide correct time sequencing.

The logic processor is implemented in low-power Schottky TTL on a 16 inch by 20 inch Multiwire board. Eleven boards are mounted vertically on one 19 inch square rack.

ARRAY PROCESSOR

The Array Processor is a special-purpose processor for the simulation of memory arrays such as RAM. These arrays can be simulated with the basic logic processor, but would soon exhaust the LSM's capacity. For example, to simulate a RAM, one LSM instruction would be required for each bit in the memory. This is clearly an inefficient use of the LSM.

The array processor was developed specifically to meet this need and greatly extends the abilities of the LSM. Up to 45 different memory arrays can be simulated on one array processor. The maximum size for any one array is 64K by 36 bits. Memories larger than 64K long or 36 bits wide must be broken into the basic 64K, 36 bit size. The total memory available on a single array processor is 512K 36 bit words.

The array processor was designed to look like a logic processor to the rest of the LSM system. Each array processor occupies one logic processor slot and communicates with the inter-processor switch in the same manner as a logic processor. Only its internal functions are different. The array processor may be substituted for a logic processor anywhere in the LSM configuration.

Arrays are defined by the user on the host computer using the LSM software. All read and write ports and address, data and control lines are defined. Each read and write port must be defined separately though they may share common inputs and controls. Then the array model is loaded into the array processor.

Once the array is defined and the model is loaded, address, data and control values are routed to the array processor and it performs the necessary reads and writes to simulate the modelled array.

CONTROL PROCESSOR

The control processor controls operation of the LSM. It starts, stops, and interrupts simulation; it reads and writes logic processor memories for simulation and diagnostics; and it communicates with the host computer through the interface computer, passing the results of simulation and loading new data into the LSM.

The control processor, like the array processor, is designed to look like another logic processor to the rest of the LSM system, so it passes data to and reads data from the logic and array processors through the same switching mechanism. The control processor occupies position 0 in the LSM rack configuration.

The control processor contains an **event mask memory** that provides for conditional simulation based on the results of instruction evaluation in other processors. The event mask memory contains 1024 4 bit words. Each bit corresponds to one of the four possible signal values and is turned on or off to indicate whether the occurrence of that value is to signal an event.

For example, if a certain instruction should never have an undefined output, an event can be triggered and simulation stopped if the output is undefined.

INTER-PROCESSOR SWITCH

The inter-processor switch is a 64 by 64 crosspoint switch that allows any processor (logic, array, control) to communicate with any other in the LSM. During each instruction step, all processors send simulation data to the switch and access the switch to obtain data needed from other processors. Any number of processors may access the output from any one processor simultaneously.

The switch is controlled by the **switch select memories**. There are 64 switch select memories in the interprocessor switch—one associated with each processor in the LSM. Each memory is a 1024 word 6 bit memory, and is stepped through sequentially by the same program counter used to access the logic processor memories. Each location in the memory contains a 6 bit address indicating the processor from which data is needed. During an instruction step, the address contained in the switch select memory selects a position in the interprocessor switch and routes the data to the memory's associated processor.

The switch select memories are loaded when the design model is loaded into the processor memories. The LSM software has determined where in the LSM the different pieces of a design will reside and loads the switch select memories so that proper communication takes place.

HOST AND INTERFACE COMPUTERS

External to the LSM system are two general purpose computers that handle user related functions.

The **host computer** provides the general purpose processing power necessary to take full advantage of the LSM. On the host computer, the user describes logic designs, creates test patterns, loads the LSM, starts simulation, and then views the simulation results. Diagnostic simulations and programs are also run from the host computer.

The **interface computer** links the LSM to the host computer, providing the necessary communication between the two and aiding data transfers. Simulations can also be loaded and run directly from the interface computer and many of the more detailed diagnostics are performed in this way.

At Los Gatos, the host computer is an IBM System 370 Model 3081; the interface computer is an IBM Series 1.

THE LSM AND THE YSE

A second generation LSM, called the Yorktown Simulation Engine, is being developed at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York. The YSE has been described fully in three papers[4,5,6] presented at the 19th Design Automation Conference.

The LSM and the YSE are based on the same architectural concept, and many of the enhancements made to the YSE are a result of experience with the LSM. The most significant difference between the two simulators is the increased speed and capacity of the YSE. The YSE provides simulation of up to 1 million gates at a rate of 3 billion gates per second, compared to 64,000 gates at 640 million per second with the LSM. To date, the LSM has proven sufficient for most logic simulations, but for large VLSI simulation the increased speed and capacity provided by the YSE will be needed.

The YSE provides the larger capacity in two ways. First, each logic processor can simulate 4096 gates, as compared to 1024 on an LSM logic processor. Second, while the LSM now supports a maximum of 64 processors, the YSE allows up to 256 processors, all operating in parallel. This greatly increased parallelism provides the higher rate of gate evaluation in the YSE.

It should be noted that evaluating 4K gates on each processor rather than 1K does cause a longer simulation time step since the length of the time step is a function of the number of instructions evaluated.

This time increase is thought to be made up by the inclusion of rank order simulation in the YSE. Rank order simulation means, in effect, that a circuit can be simulated with zero delay. Instructions can use the output of other instructions within the same simulation time step. Note that rank order simulation forces chronological ordering of instructions in the instruction memories. This ordering must be handled by the software so that the memories are loaded correctly. The LSM hardware supports rank order simulation, but this feature has not yet been supported by the software.

The YSE does not directly provide the same variable delay capabilities of the LSM. To add delay to a design in the YSE, instructions must be used that simply pass the output along, or counters must be simulated that create delay for each net. The LSM allows the user to define unique rise and fall delay times for each gate output.

The YSE and the LSM handle function evaluation differently. LSM equations have 5 two bit inputs; YSE equations have 4 two bit inputs. The LSM provides for 1024 different functions; the YSE provides 256, but also provides 16 generalized De Morgan functions on each input, giving it the power to implement the equivalent of the LSM's 1024 functions. For example, the YSE does not need to have both AND and NOR functions as primitives since it can invert the inputs to the primitive AND to get a NOR.

The YSE offers more flexible use of the four signal states output from the function unit. One of the four states may, for example, be assigned to represent high impedance. The LSM's use of the four states is restricted in that the uninitialized state is determined by the function unit and its use is not programmable.

The YSE does not support net activity trace.

ACKNOWLEDGEMENTS

The LSM was conceived in 1977 by John Cocke of IBM's Thomas J. Watson Research Center in Yorktown Heights, New York, by Richard L. Malm of Los Gatos Laboratory, and by John Schedletsky, also of Yorktown.

Richard Malm supervised development of the LSM at Los Gatos Laboratory. Warren Begas, Ted Burggraff, Jack Kohn, Al Love, Gary McGill, Greg Pfister, Jack Powers, Ann Rudy, and John Shedletsky all contributed to the hardware design. Technical support was provided by Mai Nguyen, Bobby Sloan, Don Whiteman, and Ron Williams. The LSM was developed at Los Gatos Laboratory under the management of John K. Howard and Larry Warren.

REFERENCES

[1] John K. Howard, Richard L. Malm, and Larry M. Warren, "Introduction to the IBM Los Gatos Logic Simulation Machine" Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers , Rye, New York, October 31 - November 3, 1983.

[2] Jack Kohn, Richard Malm, Chuck Meiley, and Frank Nemec, "The IBM Los Gatos Logic Simulation Machine Software," Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers , Rye, New York, October 31 - November 3, 1983.

[3] John K. Howard, Jack Kohn, Richard Malm, and Ann Rudy, "Using the IBM Los Gatos Logic Simulation Machine," Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers , Rye, New York, October 31 - November 3, 1983.

[4] Gregory F. Pfister, "The Yorktown Simulation Engine: Introduction," ACM IEEE 19th Design Automation Conference , June 1982, Paper 7.1, pp. 51-54.

[5] Monty M. Denneau, "The Yorktown Simulation Engine," ACM IEEE 19th Design Automation Conference, June 1982, Paper 7.2, pp. 55-59.

[6] E. Kronstadt and G. Pfister, "Software Support for the Yorktown Simulation Engine," ACM IEEE 19th Design Automation Conference , June 1982, Paper 7.3, pp. 60-64.