Replace {sum} with the summation sign, which is the Greek uppercase letter "sigma".

Digital's High-performance CMOS ASIC

by Jean H. Basmaji, Kay R. Fisher, Frank W. Gatulis, Herbert R. Kolk, and James F. Rosencrans

ABSTRACT

A high-performance ASIC has been developed to serve as the interface for the 10-ns bus in the new AlphaServer 8000 series server systems from Digital. The CMOS standard-cell alternative (CSALT) technology provides a timing-driven layout methodology together with a correct-by-construction approach for managing the complex device physics issues associated with state-of-the-art CMOS processes. The timing-driven layout is coupled with an automated standard-cell design approach to bring the complete design process directly to the logic designer.

INTRODUCTION

Today, high-performance microprocessors designed with complementary metal-oxide semiconductor (CMOS) processes are much more demanding on the support logic used to interface them to the rest of the system. Microprocessors, like Digital Semiconductor's Alpha 21164 chip, are extending the external logic cycle times to the point where custom-integrated circuits are necessary to realize the full performance potential. The CMOS standard-cell alternative (CSALT) technology developed at Digital satisfies these high-performance needs without resorting to a complex, custom design process.

CSALT technology provides a timing-driven layout methodology together with a correct-by-construction approach for managing the complex device physics issues associated with state-of-the-art CMOS processes. The timing-driven layout is coupled with an automated standard-cell design approach to bring the complete design process directly to the logic designer. Using CSALT, logic designers can take their application-specific integrated circuit (ASIC) designs from a concept on their desktops to a completed layout that is ready for fabrication.

Other design approaches address portions of the process, but the CSALT tool suite is complete and automated. Many ASIC vendors transfer the logic designs to a different set of engineers, using different tools and skills, to complete the physical implementation before post-layout timing analysis can take place. Any problems encountered after the layout tend to result in the design being returned to the logic designers. The artificial boundary erected between logic designers and layout implementers can result in delays. In complex designs, multiple iterations may

be necessary before the design converges into an acceptable solution. This convergence process becomes more complicated with the introduction of synthesized logic, because the process is extended to include the synthesis tools.

CSALT's timing-driven methodology eliminates the need for the many chip layout specialists and ASIC vendor experts who normally complete a multichip project. In addition, the timing-driven methodology eliminates the need for the traditional chip floorplanning step in which the designer maps the logical design onto the physical chip architecture. The floorplanning step often becomes a critical and time-consuming effort when the design is being optimized for performance.

The automated and batch-driven CSALT methodology can turn a logically complete design into a working, timing-correct chip layout within three compute-intensive days. Previous platform development projects used industry-standard ASICs, manual layouts, and hundreds of manual cell placements to meet the tight design timing requirements within their high-performance ASICs. These methods typically added months to the layout phase of these projects. CSALT's timing-driven layout was specifically developed to address these high-performance requirements and to make the complete design process available to logic designers.

This paper discusses some implementation pieces of CSALT technology and emphasizes the unique timing-driven approach and results. It explains the goals that were established for CSALT development as well as several features of the physical technology. The paper concludes with a discussion of the layout process operations and the process controller.

THE NEED FOR CSALT

During the technology evaluation phase of the AlphaServer 8000 series platform, various ASIC technology vendors were evaluated and compared against the aggressive performance needs demanded by the platform's designs and the customization that was necessary within these technologies to meet system bus timing. Based on the experience of developing designs for the previous platform generation and due to the anticipated months of iterative and interactive manual place and route necessary to meet timing, it became clear that technology was a high-risk item to the program. Requirements for the AlphaServer 8000 series systems exceeded the performance capabilities of existing ASIC technologies and the available CAD tools. In addition, access to the internal silicon structure of the ASICs was required to customize bus interface drivers. The risk and cost of developing these capabilities through working with ASIC vendors would have added months of valuable schedule time to the program.

As a result, the decision was made to focus the effort on CSALT technology and to move it from its advanced development stage to

a production-quality one. Given the selection criteria that were emphasized, a set of goals was established for the CSALT development:

- Incorporate an integrated timing-constrained driven placement.
- Implement technology in a 3.3-volt (V) stable CMOS process.
- Eliminate chip floorplanning and let timing constraints drive the placement.
- Eliminate manual interaction in the tools to reduce design time and defects.
- Develop very conservative layout rules to eliminate the need for cross talk and electromigration analysis.
- Automate the development and characterization of cell elements including thorough checking.
- Deliver more robust and accurate prediction of chip performance through integrated SPICE simulation and expanded cell library performance tables.[1]
- Use proven algorithms and software whenever possible.

OVERVIEW AND DESCRIPTION

The front-end logic design and verification process is based on the ASIC standard tools for gate array design that include schematics capture, timing and logic verification, pre-layout delay estimation, and post-layout delay feedback and analysis. The performance data for the library elements is housed in lookup tables that have multiple slope/intercept data entries based on output drive loading as well as input edge rate delay correction factors. Unique delays are calculated for each cell instance. CSALT supports a low-skew balanced clock distribution net.

The back-end layout tools for CSALT include several internally generated tools as well as research tools from academia. The heart of the place-and-route process is TimberWolf from the University of Washington.[2] One of the important features of the TimberWolf tools is their ability to be constraint-driven. These constraints are automatically generated from the timing verification step and then passed to the TimberWolf tools. TimberWolf prioritizes these critical path nets during the placement process in an attempt to meet the timing requirements. Constraints can also be manually generated through a separate user-generated file that feeds into the process. Once parameter files and constraint files are established, the place-and-route process proceeds in a completely automated and batch-driven

mechanism all the way to a completely verified design layout file (DLF). The speed of the process execution is limited only by the batch queues available and the performance of the underlying processor type.

The silicon fabrication process relies on Digital Semiconductor's CMOS line. All the physical design and process fabrication rules are built into the layout tools and driven through the parameter files specified at start-up. CSALT has built-in correct-by-construction custom design rules that guarantee all aspects of the automated layout to be free from any design rule violation. The tools account for all aspects of the physical design, such as electromigration rules, coupling capacitance effects on timing, as well as analysis of any electrical hot spots resulting from excessive logic switching in a dense localized area.

PHYSICAL TECHNOLOGY

The ASIC designs targeted for this technology needed to meet the physical, electrical, and thermal requirements of the AlphaServer 8000 series platform. The system functions that the ASIC designs satisfy belong to three classes:

- Class 1 -- Interface between the system bus and the CPU
- Class 2 -- Interface between the CPU and the local I/O
- Class 3 -- Interface between the local I/O and the Peripheral Component Interconnect (PCI)

An enhanced ASIC design style was used to reduce the time to market and to minimize design and verification resources. The enhancements to the conventional ASIC design (such as timing-driven layout and automated incorporation of SPICE delays) significantly improved ease of design for high-performance, 100-megahertz (MHz) very large-scale integration (VLSI) chips.[1]

Several features of the CSALT physical technology and their advantages are discussed in the following sections.

Low-skew Clock Distribution

There is one low-skew, single-phase clock net distribution available to the user. This is implemented through three stages. First, the buffered input clock receiver drives two high-power cells located on opposite sides of the chip. In the second stage, the high-power cells drive a central trunk that bisects the die and delivers the clock signals to each half row. In the third stage, separate local clock buffers in each half row are connected to the central trunk and deliver the clock signals to all logic elements in that particular half row.

Skew in this distributed net is controlled through automatic load balancing on the local clock buffers along each row. Cell capacitive loads are calculated for each row, and appropriate balance cells are added to bring the capacitive loads to a predefined value. This method equalized delays across the chip with less than 100 picoseconds (ps) of skew.

Other clock distributions, however, are available to the user. These clock nets are distributed through a single high-power cell driving a metal trunk along the chip. Skew within these clocks can be on the order of 300 ps, although this skew is more dependent on loading and cell distribution for each particular design.

5.0-V Compatible I/O Cells

CSALT arrays developed in Digital's fourth-generation CMOS process are powered by a 3.3-V supply for both I/O and internal core. CSALT ASICs can receive but not send 5.0-V I/O. The input receivers for both the bidirectional and the input-only cells have transistor-transistor logic (TTL) input levels and can be used in either a 3.3-V or a 5.0-V signaling environment. The CSALT PCI interface cell meets the PCI 5.0-V specification, without requiring the external module termination recommended by most ASIC vendors.

Performance-tuned Library Elements

The performance targets for the cell elements in CSALT were determined from a number of sources. First, previous ASIC designs, library performance, and heuristics were used to establish a baseline. The heuristics of the number of cell logic levels between two state elements in the DEC 7000 platform designs were analyzed. Second, the fourth-generation CMOS silicon process, electrical interconnect data, and transistor properties were used to arrive at new scaled estimates based on unit load, cell timing, and interconnect delay. Third, cycle times and system skews of the target platform were used to determine a new estimate of the levels of logic that can be placed between two state elements. The analyses resulted in the generation of baseline performance targets that were used in the design of an ASIC library tuned to cycle 100K gates at 100 MHz.

DELAY CALCULATION

CSALT post-layout timing analysis and net delay generation are based on conservative approximations and consist of six uncorrelated, additive components:

1. Intrinsic gate delay (also referred to as intercept)

2. Effect of lumped total net capacitance on delay

3. Effect of input edge rate on delay

4. Setup/hold time

5. Effect of input edge rate on setup time

6. Wire transit delay

The first five components are derived for each standard-cell type from lookup tables created using SPICE simulation.[1] The sixth component, wire transit delay, is calculated during layout for each net in each CSALT design using a specific methodology for bounding the solution.[3]

Both worst-case and best-case analyses are performed and are guaranteed to be more conservative than SPICE, because components 1, 2, and 6 of delay are measured in a conservative fashion. Paths that fail this timing analysis are then simulated with SPICE. These paths are automatically extracted from the timing analysis result files and submitted for SPICE simulation. The results of SPICE simulation are then back-annotated into timing analyses, and the design is reanalyzed using SPICE accuracy for delays on critical path nets that had failed previously. This strategy allows us to time designs quickly with the accuracy of SPICE where needed.[1]

SPICE Library Characterization

The entire cell timing data set and cell performance tables are generated automatically through a suite of automated tools called SPICE Library Characterization (SLiC). SLiC's automated procedure will create SPICE input files to fully characterize a library of CSALT cells, execute SPICE on these files, and post-process the results into a format readable by the timing tools.[1]

For cell delay slopes and intercepts, the SLiC process produces delay tables for each input-to-output path combination through each library macrocell. This is done by simulating in SPICE with 11 discrete output capacitance values attached to the cell output. The total range of loads is broken into four windows, and a best-fit line through each window is determined. Each line is then translated so that all discrete points within the window fall on or below this line (for worst-case parameters) or above this line (for best-case parameters). This translation is one mechanism for ensuring timing conservatism. Figure 1 shows the CSALT library performance approximation.

[Figure 1 (CSALT Library Performance Approximation) is not available in ASCII format.]

For edge-rate effect on delay, SLiC measures output edge rates for each of the 11 capacitance values attached to each output cell described above and stores them. In addition, SLiC creates ten simulations for each input-to-output path through each library macrocell to model the range of input edge rates that the macrocell is expected to see. These two sets of data are used to create (1) a table of delay additives to gate propagation delay as a function of input edge rate and (2) a table of output edge rates as a function of gate propagation delay. These tables are then used during the timing analysis step.

The last component of delay, wire transit delay, is the only one not determined by SLiC. During layout, the bounds on the transit delay through every net are calculated. These bounds are generated very quickly and are quite accurate for short and lightly loaded nets. For longer, more heavily loaded nets, SLiC calculates more conservative bounds.[3] This conservatism contributes to inaccuracy in path timing and is the primary reason why another methodology was developed for determining more accurate delays with SPICE.[1]

This alternative methodology for calculating delay has been verified through comparisons of thousands of path delays with SPICE. In all cases, the timing was found to be conservative. Sixty-five percent of all calculated delays are within 10 percent of SPICE prediction, and virtually all delays are within 20 percent. This methodology complements the power of the fast turnaround time of static timing analysis tools by modeling the delays more accurately and closely to SPICE prediction. Large chips can be analyzed in less than one hour and be fully timed in a few hours if any SPICE simulation of large nets becomes necessary.[1]

CONSTRAINT GENERATION OVERVIEW

After each timing verification run, a report is generated listing all paths that fail and detailing all nets and primitives within each of these failing paths. This information is then iteratively processed through an algorithm to shorten each net in the path proportionately to its original length in the path, such that it satisfies the allowable time requirement. First, the allowable total wire delay in a path is calculated in picoseconds:

[Production: In the following equation, replace {sum} with the summation sign and replace -- with the minus sign.]

W = MaxTimeLimit -- {sum} CellPrimitiveDelays -- SetupTime

where W is the total allowable wire delay for an individual failing path, MaxTimeLimit is the cycle time that the failing path needs to meet, CellPrimitiveDelays is the intrinsic delay through all the primitives that exist in the failing path, and SetupTime is the setup time required by the state element that

ends the failing path.

Then every net in the path is apportioned according to its contribution in the current (failing) total wire delay:

$$\text{NetNewDelay} = W \left(\frac{\text{NetFailingDelay}}{\text{ActualPathWire}} \right)$$

where NetNewDelay is the allowable delay on a particular net in a failing path, NetFailingDelay is the actual delay on a net within a failing path, and ActualPathWire is the total accumulated wire delay of all nets in the failing path.

Since wires can be shared by more than one failing path, a change in the length of a wire in one path will cause other paths that have the same wire as an element to be scheduled for recalculation. A wire length may change several times before it is stable. During recalculation, the smaller wire produced is the one that will be used. This iteration algorithm continues until no nets are scheduled for reevaluation, and convergence is achieved. The number of iterations can be limited if convergence is not achieved in a timely manner.

At completion, NetNewDelays are then converted into wire lengths:

$$\text{NetLength} = \frac{\text{NetNewDelay} - (\text{SlopeOfDriver} * \{\text{sum}\}\ \text{GateLoad})}{\text{SlopeOfDriver} * \text{CapUL}}$$

where NetLength is the calculated net constraint in unit length, SlopeOfDriver is the slope of the cell driving the failing net in unit time per capacitance, GateLoad is the sum capacitance of all cells tied to that net, and CapUL is the capacitance per unit length for interconnect metal.

NetLength is then compared to a quench value, and the larger of the two is used as the new net constraint feeding back to a new layout. Quench values define the minimum wire length that a net can have, based on the number of pins (fan-out) in that net.

PHYSICAL DIE ARCHITECTURE

The CSALT die architecture, as shown in Figure 2, consists of the following sections:

[Figure 2 (CSALT Die Architecture) is not available in ASCII format.]

- I/O cells -- The outermost region where the I/O cells are located is also called the pad ring. Bonding pads are built into the I/O cells.

- High-power and decouple cells -- This region of the array, also called the high-power ring, is filled predominantly with decouple cells. This region also allows for placement of a limited number of high-power driver cells designed to drive heavily loaded nets such as clock lines and reset lines.

- Core -- This region holds the majority of the logic in the array implemented as standard cells. All these cells are the same height but vary in width according to functional complexity. Core cells are arranged in rows numbered from the bottom of the array. The number of rows in the core is a design-dependent variable. The space between the rows varies from row to row and is used for routing channels.

  Generally, power to the core is distributed by cell abutment on metal 3 over the cell rows. Horizontal signal routing in the core channels takes place on metal 2. Metal 1 is used for vertical core routing. To route in the vertical direction, the rows contain feeds. By design, many standard cells have vertical feeds to provide pass through. In addition, a standard feed cell can be automatically inserted by the layout tools when the demand for feeds is high. I/O bristles for each of the core cells are made available on the top and bottom of the cells to enhance routability.

- Trunk -- The region splitting the core into left and right halves is referred to as the trunk. The trunk is a routing region used primarily to route clocks and power signals down the center of the core. These signals are then distributed to the left and right sides of the core on a row-by-row basis.

- Ring -- Although not indicated in Figure 2, the term ring refers to the I/O ring, the mini-moat, and the high-power ring regions as a group. Even though the physical size of the ring is fixed, the total dimensions are determined by the package size of the array. The size of the ring establishes the available area remaining in the center of the array for the moat and the core.

- Mini-moat -- The mini-moat is the region separating the I/O ring from the high-power ring. The layout process uses this region to route a small number of high fan-out nets that drive cells in the I/O ring. Layout parameters control the assignment of nets to the mini-moat.

- Moat -- The moat is a routing region used by the layout tools for attaching the ring to the core. The size of the moat is determined by the amount of space that is left over when the core and trunk are placed and routed. Small

arrays (low gate counts) result in small cores and large moat areas, and large arrays (high gate counts) result in large cores and small moat areas. During the layout of large chips, it is possible for the core to become so large that not enough moat space remains to make all the necessary routing connections.

Figure 3 is a photomicrograph of a CSALT die for one of the CPU gate arrays used in the AlphaServer 8000 series server systems.

[Figure 3 (Photomicrograph of a CSALT Die) is not available in ASCII format.]

PLACEMENT AND ROUTING

The function of the layout tool suite is to provide a fully placed and routed array that meets or exceeds all the design timing criteria and that satisfies all electrical and physical layout rules required to release the array for mask generation. Several place-and-route features are discussed in the following sections.

Constraint-driven Layout System

When an array is submitted to layout, it is accompanied by a set of timing constraints. Timing constraints can be thought of as estimated restrictions, on a per-net basis, for the amount of metal lengths allowed to interconnect the net in the layout. These constraints drive the TimberWolf placement tool and are ultimately responsible for the placement of core cells in the final layout. Because a working design may not be achieved on the first layout iteration, the overall CSALT methodology provides mechanisms for analyzing post-layout timing delays and for generating a refined set of constraints that can be fed back into the layout for another pass. The layout process is iterated in this manner until it converges on a layout solution that meets the timing constraints.

Routability

To ensure 100 percent routing, the routing process had to be kept simple, which required substantial planning during development of the chip architecture described above. As a result, the following elements of the architecture were defined: (1) Pins are available on both the top and the bottom of the cells; (2) Power and clock connections are defined by cell abutment; and (3) Total routing of the chip is divided into four areas (the core, the moat, the ring, and the megacell interface). This plan kept the routing problems similar from chip to chip, which allowed the routing tools to focus on particular solutions.

Quick Turnaround Time

One significant feature of the CSALT layout process is that it can complete a layout without manual intervention, saving time over manual processes. CSALT consistently demonstrated that the CAD suite can provide completed layouts in three to ten days from the time the wirelist enters the layout process. An array that has been in layout for ten days is likely to be one that is difficult to time and that has required four to six layout iterations to converge.

Cross-talk Effect Inclusion

In recent generations of ASIC technologies, interconnect metal widths and pitches have been shrinking while the clock frequencies have been on the rise. This raised some concerns about on-chip cross-talk effect due to the ability of signals traveling on one wire to affect the speed of signals traveling on adjacent or victim wires. In extreme cases, this cross talk can cause signals to spike on the victim wires. CSALT methodology compensates for such effects on wire delay calculation, and the compensation is integrated into the layout process.

The integration is implemented by factoring in a coupling capacitance extracted from layout and by using a worst-case signal-switching scenario. Conservative factors were chosen after analysis of cross talk on a representative cross section of CSALT layouts, using different routing pitches on signal interconnect metal 2. The goal was to find the right balance between metal pitch, area used, and chip timing. The study resulted in an optimum pitch definition of 3.75 micrometers um for metal 2, and a coupling capacitance multiplier of 2. The core area increase, from that of using the minimum pitch for metal 2 at 2.625 um, was less than 10 percent for the largest design. However, overall die area increase was negligible due to the designs being I/O-ring-limited in nature.

Free of Electromigration

When the current density in the aluminum interconnect used in today's high-density CMOS processes is too high, a detrimental physical phenomenon occurs. This phenomenon causes metal reliability problems in which metal molecules slowly migrate, resulting in open/short circuits inside ASICs. To eliminate the need for a long and manually tedious process of looking for these problems after layout, the CSALT strategy is to avoid electromigration problems during layout through analysis and implementation of built-in conservative layout disciplines.

Analysis of several CSALT arrays resulted in an increase in the contact capacity and the definition of maximum output load limits

for each macrocell. The limit set was a maximum of 130 unit loads (7.8 picofarads [pF]) switching at maximum frequency (100 MHz). In addition to that limit being available to designers during the design phase for proper fan-out implementation, the tools automatically flag all nets that exceed the limit.

A number of other features are designed into the CSALT process to guarantee that layouts are free from electromigration problems:

- Library data tables are used to dynamically assign metal widths and corresponding contact sizes according to driver strengths and loads. This eliminates electromigration problems for dynamically sized metal routes such as clock nets and other high fan-out nets.

- The bulk of the power distribution is achieved by cell abutment. Cell power rails are conservatively designed to handle the largest row's current demand.

- As a final check on correctness, one of the layout process steps incorporates a hot row tool. This tool flags any rows in the core whose cells collectively exceed a predetermined current threshold defined by the handling capability of the power cells in the rows. This information is used to flag a potential electromigration situation in the contact structure, distributing power from the trunk to the row. When the row is flagged, the user manually reviews the result files and analyzes the row. Out of 15 separate designs completed, not one had to be changed due to flagged hot rows. This is due to the extremely conservative assumption used by the tool -- it assumes all logic is switching at maximum frequency.

Correct-by-construction Concept

As it applies to all the critical device issues (for example, electromigration, cross talk, hot carrier injection, and latchup), acceptance of the concept of a layout being correct by construction has dramatically reduced turnaround time in the layout process by eliminating the need to perform these analysis operations on each array. Why does it work for CSALT? It works because the CSALT layout process is very deterministic, and correctness has been verified on a cross section of arrays. In the final analysis, all arrays use the same cells from a well-defined and characterized library. The architecture of the die is the same in all arrays. As a result, variation is likely to enter the system only during the routing process. This process incorporates conservative layout rules and checks to avoid and detect potential failure mechanisms.

CSALT LAYOUT PROCESS

As shown in Figure 4, the layout process encompasses five basic assembly and check operations: full wirelist preparation, pad-ring assembly, core assembly, chip assembly, and verification.

[Figure 4 (CSALT CAD Layout Overview) is not available in ASCII format.]

1. During wirelist preparation, the input wirelist is analyzed, names are manipulated to conform to layout naming conventions, and the design is partitioned into pad-ring and core components.

2. During pad-ring assembly, I/O and high-power cell/slot assignments are made according to bonding requirements. The ring is then globally routed.

3. During core assembly, floorplanning for the trunk and any random-access memory (RAM) devices takes place; timing constraints from several sources (pre-layout, user defined, current layout, and previous layouts) are merged into a worst-case set of composite constraints that are used by the TimberWolf tool to place and globally route the core. Also during this step, the balanced clock system and scan chain are synthesized and globally routed. The SCAR channel router is then used to route the standard-cell portion of the core. If the design contains RAMs, they are then placed in their floorplanned locations, globally routed, and finally attached to the core using the area router, Chameleon.

4. During chip assembly, the interfaces between the core, moat, and pad ring are refined. Chameleon is used to perform final routing of the ring and the moat. Thus far the chip has been completely placed and routed using cell outlines containing only dimension information, relative coordinates of bristles, and bristle names. The ring and core cell outlines are now replaced with the actual cell layout information from the library, and a complete design layout file for the chip is produced. As the file is generated, alignment block and substrate ring data is added to make the physical representation file ready for mask set production.

5. The following procedures occur during the verification phase of the layout:

    - Physical design rules are checked.

    - A wirelist comparison is performed to ensure that the layout file is an exact match to the logical representation of the design.

    - Capacitance information from the layout file is

extracted, and the process proceeds to calculate a conservative metal delay (including compensation for cross talk) for each net in the layout. These post-layout metal delays are fed back into the timing verification process. In addition, SPICE is run on clock nets and other critical nets predefined by the user, and the delays are made available for timing analyses.[1]

The entire process runs automatically and, when possible, steps are run concurrently. Manual intervention is unnecessary and is discouraged; it is used only during tool debugging or special customization.

PROCESS CONTROLLER

CSALT's fully automated process controller (PC) ensures optimum use of system resources, orchestrates the entire layout process, provides all necessary data management functions, and provides the user with a very simple set of commands for operating an otherwise complex process.

The power of the PC is in its continuous dynamic decomposition of every layout into parallel batch streams. The PC runs the entire layout process in batch mode, taking full advantage of opportunities to use multiple processors and run independent parts of the layout process in parallel streams. Because it hides all the CAD and process complexity from a user, no previous CAD or layout skills are required to iterate layouts once initial layout parameters have been established for a given array.

Figure 5 illustrates the flow for the CSALT layout process. (A and B indicate connectivity points.) Each name in the process flow represents the name of a single process step. Execution of the layout process is controlled by a single command procedure called CSALT place and route (CPR). Although other CPR command line options exist, CPR is most often used in its simplest and most powerful form, CPR PC. This command causes CPR to invoke the dynamic PC that will analyze the current relationships of a layout database and begin automatic execution of the layout process from the next eligible process step.

[Figure 5 (CSALT CAD Process Flow Diagram) is not available in ASCII format.]

RESULTS AND CONCLUSION

CSALT gate array technology was used extensively during the development of the AlphaServer 8000 server systems. This design methodology removed the product's critical dependency on the place-and-route portion of the design process. As a result, timing-correct ASIC layouts were produced in fewer than 72 hours.

In addition, the CSALT ASIC logic designers had access to the proven 3.3-V silicon structures developed by Digital Semiconductor. CSALT's timing-driven layout approach for designing and implementing a high-performance ASIC made the AlphaServer 8000 server systems' aggressive 10-nanosecond bus speed a reality, with minimum risk.[4]

Although CSALT's implementation pieces may not be unique, the approach that was taken to link the set of front-end design tools with the back-end layout has proven to be unique, with unmatched results. No ASIC vendor today (January 1995) can provide logic designers with the ability to do their own automated, timing-correct layouts from their desktops.

In less aggressive designs, a large number of working layout solutions exist. The number of these solutions starts to shrink when the technology is more aggressively used. Iterative timing-driven layout efficiently searches through the matrix of possible solutions to find a working layout. Coupling timing-driven layout with logic synthesis can bring us very close to achieving the "silicon compiler" goals of automatically producing working designs from high-level logic descriptions.

ACKNOWLEDGMENTS

The authors would like to acknowledge the efforts of the following people, without whom the project would not have been successful: Meaghan Engdahl, Dave Caffo, and Paul Janson. We would also like to acknowledge other members of the CSALT team: Linda Greska, Kevin Gamache, Dennis Litwinetz, Bill Gist, Dave Vanderbeek, John Drasher, John Kennedy, and certainly Dick Davis and Darrel Donaldson for their CSALT vision. We would also like to acknowledge Professor Carl Sechen and Bill Swartz for their continued TimberWolf support.

REFERENCES AND NOTES

1. SPICE is a general-purpose circuit simulator program developed by Lawrence Nagel and Ellis Cohen of the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley.

2. TimberWolf Mixed Macro/Standard Cell Floorplanning and Routing Package is a public domain automatic layout package. Developed and directed by Professor Carl Sechen from July 1986 through June 1992 at Yale University and later at the University of Washington, it is currently maintained and supported by TimberWolf Systems, Inc., Dallas, Texas.

3. J. Rubenstein, P. Penfield, and M. Horowitz, "Signal Delay in RC Tree Networks," IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems, CAD-2 (3) (July

1983): 202-211.

4. D. Fenwick, D. Foley, W. Gist, S. VanDoren, and D. Wissell, "The AlphaServer 8000 Series: High-end Server Platform Development," Digital Technical Journal, vol. 7, no. 1 (1995, this issue): 43-65.

BIOGRAPHIES

Jean H. Basmaji

Jean Basmaji is a hardware consulting engineer in the Server Platform Development Group. As the CSALT development project leader and ASIC technology manager, he was responsible for the transition of the CSALT technology project from advanced development to production. Jean has been the technical director of computer-aided engineering and design verification testing for the Server Platform Development Group since the introduction of Digital's first multiprocessing computer, the VAX 6200 system. Jean joined Digital in 1978 after receiving a B.S.E.E. from Lowell Technological Institute in 1977.

Kay R. Fisher

Kay Fisher joined Digital in 1973 as a hardware instructor and course developer. He has more than 20 years of software engineering experience in the areas of memory testing, fault-tolerant multicomputer systems, and VAX systems. Currently a principal software engineer in the Alpha and VAX Servers Group, Kay is responsible for the process control software for Digital's CAD system. His code runs programs in parallel and synthesizes clock drivers, networks, and scan chains. Kay received a B.S. in computer science (magna cum laude, 1978) from Boston University-Metropolitan College.

Frank W. Gatulis

As supervisor of the CAD segment of the CSALT technology team, consultant engineer Frank Gatulis is responsible for defining and implementing automated layout tools and processes. A graduate of Don Bosco Technical Institute, he also attended the B.S.E.E. program at Northeastern University. Frank has also supervised the I/O diagnostic team for DECsystem 10 and DECsystem 20 products as well as the development of the console-based fault detection system and automated isolation tool suite used on the VAX 8600 and VAX 8650. His architecture segments strategies were used to fabricate and test multichip units in VAX 9000 systems. Before joining Digital in 1973, Frank worked at EG&G where he developed real-time picture acquisition, processing, and compression software.

Herbert R. Kolk

A consulting software engineer who specializes in chip routing,

Herb Kolk currently supports and enhances the CSALT layout process. Prior to that, he was the architect of the Chameleon router, which was used for routing gate arrays, multichip modules, and CPU boards for the VAX 9000. This router is still being used in the CSALT process. Before joining Digital in 1983, Herb designed software systems for the Communications Systems Division of GTE. Herb received a B.S.E.E. (with honors) from the Rochester Institute of Technology in 1973.

James F. Rosencrans

Jim Rosencrans is a principal hardware engineer in the AlphaServer Engineering Group. As the ASIC technologist for server system development, Jim supports the AlphaServer 8000 series design teams and acts as liaison for both the teams and test engineering. In addition to leading the CSALT micropackaging definition and design, he contributed to the development of CSALT technology. His previous work includes contributions to ASIC and ASIC/silicon technology selection, definition, and design support. Before joining Digital in 1988, Jim was with NCR Microelectronics, where he worked on custom and CMOS ASIC design and silicon process development. A member of IEEE, Jim received a B.S.E.E. from the University of Wisconsin in 1980.

TRADEMARKS

==========

==========