A 200-MHz 64-bit Dual-issue CMOS Microprocessor

1 Abstract

A reduced instruction set computer (RISC)-style microprocessor has been designed and tested that operates up to 200 megahertz (MHz). The chip implements a new 64-bit architecture, designed to provide a huge linear address space and to be devoid of bottlenecks that would impede highly concurrent implementations. Fully pipelined and capable of issuing two instructions per clock cycle, this implementation can execute up to 400 million operations per second. The chip includes an 8-kilobyte (KB) I-cache, 8KB D-cache and two associated translation buffers, a four-entry, 32-byte-per-entry write buffer, a pipelined 64-bit integer execution unit with a 32-entry register file, and a pipelined floating-point unit (FPU) with an additional 32 registers. The pin interface includes integral support for an external secondary cache. The package is a 431-pin pin grid array (PGA) with 140 pins dedicated to V(DD)/V(SS) (power supply voltage/ground). The chip is fabricated in a 0.75-micrometer (m) n-well complementary metal-oxide semiconductor (CMOS) process with three layers of metalization. The die measures 16.8 millimeters (mm) x 13.9 mm and contains 1.68 million transistors. Power dissipation is 30 watts (W) from a 3.3-volt (V) supply at 200 MHz.

2 CMOS Process Technology

The chip is fabricated in a 0.75-m, 3.3-V, n-well CMOS process optimized for high-performance microprocessor design. Process characteristics are shown in Table 1. The thin gate oxide and short transistor lengths result in the fast transistors required to operate at 200 MHz. There are no explicit bipolar devices in the process as the incremental process complexity and cost were deemed too large in comparison to the benefits provided - principally more area-efficient large drivers such as clock and I/O.

The metal structure is designed to support the high operating frequency of the chip. Metal 3 is very thick and has a relatively large pitch. It is important at these speeds to have a low-resistance metal layer available for power and clock distribution. It is also used for a small set of special signal wires such as the data buses to the pins and the control wires for the two shifters. Metal 1 and metal 2 are maintained at close to their maximum thickness by planarization and by filling metal 1 and metal 2 contacts with tungsten plugs. This removes a potential weak spot in the electromigration characteristics of the process and allows more freedom in the design without compromising reliability.

3 Alpha AXP Architecture

The computer architecture implemented is a 64-bit load/store RISC architecture with 168 instructions, all 32 bits wide.[1] Supported data types include 8-, 16-, 32-, and 64-bit integers and both Digital and IEEE 32- and 64-bit floating-point formats. Each of the two register files, integer and floating point, contains 32 entries of 64 bits with one entry in each being a hardwired zero. The program counter and virtual address are 64 bits. Implementations can subset the virtual address size, but are required to check the full 64-bit address for sign extension. This ensures that when later implementations choose to support a larger virtual address, programs will still run and not find addresses that have dirty bits in the previously "unused" bits.

The architecture is designed to support high-speed multi-issue implementations. To this end the architecture does not include condition codes, instructions with fixed source or destination registers, or byte writes of any kind (byte operations are supported by extract and merge instructions within the CPU itself). Also there are no first-generation artifacts that are optimized around today's technology, which would represent a long-term liability to the architecture.

4 Chip Microarchitecture

The block diagram (Figure 1) shows the major functional blocks and their interconnecting buses, most of which are 64 bits wide. The chip implements four functional units: the integer unit (IRF plus E-box), the floating-point unit (FRF plus F-box), the load/store unit (A-box), and the branch unit (distributed). The bus interface unit (BIU), described in the next section, handles all communication between the chip and external components. The microphotograph (Figure 2) shows the boundaries of the major functional units. The dual-issue rules are a direct consequence of the register file ports, the functional units, and the I-cache interface. The integer register file (IRF) has two read ports and one write port dedicated to the integer unit, and two read and one write port shared between the branch unit and the load/store unit. The floating-point register file (FRF) has two read ports and one write port dedicated to the floating unit, and one read and one write port shared between the branch unit and the load/store unit. This leads to dual-issue rules that are quite general:

- Any load/store in parallel with any operate

- An integer operate in parallel with a floating operate

- A floating operate and a floating branch

- An integer operate and an integer branch
  except that integer store and floating operate and floating store and integer operate are disallowed as pairs.

NOTE

> Figure 2 (Microphotograph of Chip) is a photograph and is unavailable.

As shown in Figure 3a, the integer pipeline is 7 stages deep, where each stage is a 5-nanosecond (ns) clock cycle. The first four stages are associated with instruction fetching, decoding, and scoreboard checking of operands. Pipeline stages 0 through 3 can be stalled. Beyond 3, however, all pipeline stages advance every cycle. Most arithmetic and logic unit (ALU) operations complete in cycle 4, allowing single-cycle latency, with the shifter being the exception. Primary cache accesses complete in cycle 6, so cache latency is three cycles. The chip will do hits under misses to the primary D-cache.

The I-stream is based on autonomous prefetching in cycles 0 and 1 with the final resolution of I-cache hit not occurring until cycle 5. The prefetcher includes a branch history table and a subroutine return stack. The architecture provides a convention for compilers to predict branch decisions and destination addresses, including those for register indirect jumps. The penalty for branch mispredict is four cycles.

The floating-point unit is a fully pipelined 64-bit floating-point processor that supports both VAX standard and IEEE standard data types and rounding modes. It can generate a 64-bit result every cycle for all operations except divide. As shown in Figure 3b, the floating-point pipeline is identical and mostly shared with the integer pipeline in stages 0 through 3; however, the execution phase is three cycles longer. All operations, 32- and 64-bit (except divide) have the same timing. Divide is handled by a nonpipelined, single bit per cycle, dedicated divide unit.

In cycle 4, the register file data is formatted to fraction, exponent, and sign. In the first-stage adder, exponent difference is calculated and a 3 x multiplicand is generated for multiplies. In addition, a predictive leading 1 or 0 detector using the input operands is initiated for use in result normalization. In cycles 5 and 6, for add/subtract, alignment or normalization shift and sticky-bit calculation are performed. For both single- and double-precision multiplication, the multiply is done in a radix-8 pipelined array multiplier. In cycles 7 and 8, the final addition and rounding are performed in parallel and the final result is selected and driven back to the register file in cycle 9. With an allowed bypass of the register write data, floating-point latency is six cycles.

The CPU contains all the hardware necessary to support a demand paged virtual memory system. It includes two translation buffers to cache virtual-to-physical address translation. The instruction translation buffer contains 12 entries, 8 that map 8KB pages and 4 that map 4-megabyte (MB) pages. The data translation buffer contains 32 entries that can map 8KB, 64KB, 512KB, or 4MB pages.

The CPU supports performance measurement with two counters that accumulate system events on the chip such as dual-issue cycles and cache misses or external events through two dedicated pins that are sampled at the selected system clock speed.

5 External Interface

The external interface (Figure 4) is designed to directly support an off-chip backup cache that can range in size from 128KB to 8MB and can be constructed from ordinary SRAMs. For most operations, the CPU chip accesses the cache directly in a combinatorial loop by presenting an address and waiting N CPU cycles for control, tag, and data to appear, where N is a mode-programmable number between 3 and 16 set at power-up time. For writes, both the total number of cycles and the duration and position of the write signal are programmable in units of CPU cycles. This allows the module designer to select the size and access time of the SRAMs to match the desired price/performance point.

The interface is designed to allow all cache policy decisions to be controlled by logic external to the CPU chip. There are three control bits associated with each backup cache (B-cache) line: valid, shared, and dirty. The chip completes a B-cache read as long as valid is true. A write is processed by the CPU only if valid is true and shared is false. When a write is performed, the dirty bit is set to true. In all other cases, the chip defers to an external state machine to complete the transaction. This state machine operates synchronously with the SYS_CLK output of the chip, which is a mode-controlled submultiple of the CPU clock rate ranging from divide by 2 to divide by 8. It is also possible to operate without a backup cache.

As shown in the diagram, the external cache is connected between the CPU chip and the system memory interface. The combinatorial cache access begins with the desired address delivered on the adr_h lines and results in ctl, tag, data, and check bits appearing at the chip receivers within the prescribed access time. In 128-bit mode, B-cache accesses require two external data cycles to transfer the 32-byte cache line across the 16-byte pin bus. In 64-bit mode, it is four cycles. This yields a maximum backup cache read bandwidth of 1.2 gigabytes per second (GB/s) and a write bandwidth of 711MB/s. Internal cache lines can be invalidated at the rate of one line per cycle using the dedicated invalidate address pins, iAdr_h<12:5>.

In the event external intervention is required, a request code is presented by the CPU chip to the external state machine in the time domain of the SYS_CLK as described previously. Figure 5 shows the read miss timing where each cycle is a SYS_CLK cycle. The external transaction starts with the address, the quadword within block and instruction/data indication supplied on the cWMask_h pins, and READ_BLOCK function supplied on the cReq_h pins. The external logic returns the first 16 bytes of data on the data_h and error correcting code (ECC) or parity on the check_h pins. The CPU latches the data based on receiving acknowledgment on <keep(rdAck_H.) The diagram shows a stall cycle (cycle 4) between the request and the return data; this depends on the external logic and could range from zero to many cycles. The second 16 bytes of data are returned in the same way with rdAck_h signaling the return of the data and cAck_h signaling the completion of the transaction. cReq_h returns to idle and a new transaction can start at this time.

The chip implements a novel set of features supporting chip and module test. When the chip is reset, the first action is to read from a serial read-only memory (SROM) into the I-cache via a private three-wire port. The CPU is then enabled and the program counter (PC) is forced to 0. Thus with only three functional components (CPU chip, SROM, and clock input), a system is able to begin executing instructions. This initial set of instructions is used to write the bus control registers inside the CPU chip to set the cache timing and to test the chip and module from the CPU out. After the SROM loads the I-cache, the pins used for the SROM interface are enabled as serial in and out ports. These ports can be used to load more data or to return status of testing and setup.

6 Circuit Implementation

Many novel circuit structures and detailed analysis techniques were developed to support the clock rate in conjunction with the complexity demanded by the concurrence and wide data paths. The clocking method is single wire level sensitive. The bus interface unit operates from a buffered version of the main clock. Signals that cross this interface are deskewed to eliminate races. This clocking method eliminates dead time between phases and requires only a single clock signal to be routed throughout the chip.

One difficulty inherent in this clocking method is the substantial load on the clock node, 3.25 nanofarad (nF) in our design. This load and the requirement for a fast clock edge led us to take particular care with clock routing and to do extensive analysis on the resulting grid. Figure 6 shows the distribution of clock load among the major functional units. The clock drives into a grid of vertical metal 3 and horizontal metal 2. Most of the loading occurs in the integer and floating-point units that are fed from the more robust metal 3 lines. To ensure the integrity of the clock grid across the chip, the grid was extracted from the layout and the resulting network, which contained 630,000 RC elements, was simulated using a circuit simulation program based on the AWEsim simulator from Carnegie-Mellon University. Figure 7 shows a three-dimensional representation of the output of this simulation and shows the clock delay from the driver to each of the 63,000 transistor gates connected to the clock grid.

The 200-MHz clock signal is fed to the driver through a binary fanning tree with five levels of buffering. There is a horizontal shorting bar at the input to the clock driver to help smooth out possible asymmetry in the incoming wave front. The driver itself consists of 145 separate elements, each of which contains four levels of prescaling into a final output stage that drives the clock grid.

The clock driver and predriver represent about 40 percent of the total effective switching capacitance determined by power measurement to be 12.5 nF (worst case including output pins). To manage the problem of di/dt on the chip power pins, explicit decoupling capacitance is provided on-chip. This consists of thin oxide capacitance that is distributed around the chip, primarily under the data buses. In addition, there are horizontal metal 2 power and clock shorting straps adjacent to the clock generator, and the thin oxide decoupling cap under these lines supplies charge to the clock driver. di/dt for the driver alone is about 2 x 10[11] amperes per second. The total decoupling capacitance as extracted from the layout measures 128 nF. Thus the ratio of decoupling capacitance to switching cap is about 10:1. With this capacitance ratio, the decoupling cap could supply all the charge associated with a complete CPU cycle with only a 10 percent reduction in the on-chip supply voltage.

Latches

As previously described, the chip employs a single-phase approach, with nearly all latches in the core of the chip receiving the clock node, CLK, directly. A representative example is illustrated in Figure 8. Notice that L1 and L2 are transparent latches separated by random logic and are not simultaneously active; L1 is active when CLK is high and L2 is active when CLK is low. The minimum number of delays between latches is zero and the maximum number of delays is constrained only by the cycle time and the details of any relevant critical paths. The bus interface unit, many data-path structures, and some critical paths deviate from this approach and use buffered versions and/or conditionally buffered versions of CLK. The resulting clock skew is managed or eliminated with special latch structures.

The latches used in the chip can be classified into two categories: custom and standard. The custom latches were used to meet the unique needs of data-path structures and the special constraints of critical paths. The standard latches were used in the design of noncritical control and in some data-path applications. These latches were designed prior to the start of implementation and were included in the library of usable elements for logic synthesis. All synthesized logic used only this set of latches.

The standard latches are extensions of previously published work, and examples are shown in Figures 9 to 11.[2] To understand the operation of these latches, refer to Figure 9a. When CLK is high, P1, N1, and N3 function as an inverter complementing IN1 to produce X. P2, N2, and N4 function as a second inverter and complement X to produce OUT. Therefore, the structure passes IN1 to OUT. When CLK is low, N3 and N4 are cut off.

If IN1, X, and OUT are initially high, low, and high respectively, a transition of IN1 falling pulls X high through P1 causing P2 to cut off, which tristates OUT high. If IN1, X, and OUT are initially low, high, and low respectively, a transition of IN1 rising causes P1 to cut off, which tristates X high leaving out tristated low. In both cases, additional transitions of IN1 leave X tristated or driven high with OUT tristated to its initial value. Therefore, the structure implements a latch that is transparent when CLK is high and opaque when CLK is low. Figure 9c shows the dual circuit of the latch just discussed; this structure implements a latch that is transparent when CLK is low and opaque when CLK is high. Figures 9b and 9d depict latches with an output buffer used to protect the sometimes dynamic node OUT and to drive large loads.

The design of the standard latches stressed three primary goals: flexibility, immunity to noise, and immunity to race-through. To achieve the desired flexibility, a variety of latches like those in Figures 9 to 11 in a variety of sizes were characterized for the implementors. Thus the designer could select a latch with an optional output buffer and an embedded logic function that was sized appropriately to drive various loads. Furthermore, it was decided to allow zero delay between latches, completely freeing the designer from race-through considerations when designing static logic with these latches.

In the circuit methodology adopted for the implementation, only one node, X (Figure 9a), poses inordinate noise margin risk. As noted above, X may be tristated high with OUT tristated low when the latch is opaque. This maps into a dynamic node driving into a dynamic gate that is very sensitive to noise that reduces the voltage on X, causing leakage through P2, thereby destroying OUT. This problem was addressed by the addition of P5. This weak feedback device is sized to source enough current to counter reasonable noise and hold P2 in cutoff. N5 plays an analogous role in Figure 9c.

Race-through was the major functional concern with the latch design. It is aggravated by clock skew, the variety of available latches, and the zero delay goal between latches. The clock skew concern was actually the easiest to address. If data propagates in a direction that opposes the propagation of the clock wave front, clock skew is functionally harmless and tends only to reduce the effective cycle time locally. Minimizing this effect is of concern when designing the clock generator. If data propagates in a direction similar to the propagation of the clock wave front, clock skew is a functional concern. This was addressed by radially distributing the clock from the center of the chip. Since the clock wave front moves out radially from the clock driver toward the periphery of the die, it is not possible for the data to overtake the clock if the clock network is properly designed.

To verify the remaining race-through concerns, a mix-and-match approach was taken. All reasonable combinations of latches were cascaded together and simulated. The simulations were stressed by eliminating all interconnect and diffusion capacitance and by pushing each device into a corner of the process that emphasized race-through. Then many simulations with varying

CLK rise and fall times, temperatures, and power supply voltages were performed. The results showed no appreciable evidence of race-through for CLK rise and fall times at or below 0.8 ns. With 1.0-ns rise and fall times, the latches showed signs of failure. To guarantee functionality, CLK was specified and designed to have an edge rate of less than 0.5 ns. This was not a serious constraint since other circuits in the chip required similar edge rates of the clock.

A last design issue worth noting is the feedback devices, N5 and P5, in Figures 10c, 10d, 11a, and 11b. Notice that these devices have their gates tied to CLK instead of OUT like the other latches. This difference is required to account for an effect not present in the other latches. In these latches, a stack of devices is connected to node X without passing through the clocked transistors P3 or N3. Referring to Figure 11a, assume CLK is low, X is high, and OUT is low. If multiple random transitions are allowed by IN1 with IN2 high, then coupling through P1 can drive X down by more than a threshold even with weak feedback, thereby destroying OUT. To counter this phenomenon, P5 cannot be a weak feedback device and therefore cannot be tied to OUT if the latch is to function properly when CLK is high. Note that taller stacks aggravate this problem because the devices become larger and there are more devices to participate in coupling. For this reason, stacks in these latches were limited to three high. Also, note that clocking P5 introduces another race-through path since X will unconditionally go high with CLK falling, and OUT must be able to retain a stored ONE. So there is a two-sided constraint: P5 must be large enough to counter coupling and small enough not to cause race-through. These trade-offs were analyzed by simulation in a manner similar to the one outlined above.

64-bit Adder

A difficult circuit problem was the 64-bit adder portion of the integer and floating-point ALUs. Unlike a previous high-speed design, we set a goal to achieve single-cycle latency in this unit.[3] Figure 12 has an organizational diagram of its structure. Every path through the adder includes two latches, allowing fully pipelined operation. The result latches are shown explicitly in the diagram; however, the input latches are somewhat implicit, taking advantage of the predischarge characteristics of the carry chains. The complete adder is a combination of three methods for producing a binary add: a byte long carry chain, a longword (32-bit) carry select, and local logarithmic carry select.[4] The carry select is built as a set of n-channel metal-oxide semiconductor (NMOS) switches that direct the data from byte carry chains. The 32-bit longword lookahead is implemented as a distributed differential circuit controlling the final stage of the upper longword switches. The carry chains are organized in groups of eight bits.

Carry chain width was chosen to implement a byte compare function specified by the architecture. The carry chain implemented with NMOS transistors is shown in Figure 13a. Operation begins with the chain predischarged to V(SS), with the controlling signal an OR of CLK and the kill function. Evaluation begins along the chain length without the delay associated with the V(gs) - V(t) threshold found in a chain precharged to V(DD). An alternative to a predischarged state was to precharge to V(DD) - V(t), but the resulting low noise margins were deemed unacceptable. From the least significant bit to the most significant bit, the width of the NMOS gates for each carry chain stage is tapered down, reducing the loading presented by the remainder of the chain. The local carry nodes are received by ratioed inverters. Each set of propagate, kill, and generate signals controls two carry chains, one that assumes a carry in and one that assumes no carry in. The results feed the bit-wise data switches as well as the carry selects.

The longword carry select is built as a distributed cascode structure used to combine the byte generate, kill, and propagate signals across the lower 32-bit longword. It controls the final data selection into the upper longword output latch and is out of the critical path.

The NMOS byte carry select switches are controlled by a cascade of closest neighbor byte carry outs. Data in the most significant byte of the upper longword is switched first by the carry-out data of the next lower byte, byte 6, then by byte 5, and finally byte 4. The switches direct the sum data from either the carry-in channel or the no-carry channel (Figure 13b). Sign extension is accomplished by disabling the upper longword switch controls on longword operations and forcing the sign of the result into both data channels.

I/O Circuitry

To provide maximum flexibility in applications, the external interface allows for several different modes of operation all using common on-chip circuitry. This includes choice of logic family (CMOS/transistor-transistor logic [TTL] or emitter-coupled logic [ECL]) as well as bus width (64/128 bits), external cache size and access time, and BIU clock rate. These parameters are set into mode registers during chip power-up. The logic family choice provided an interesting circuit challenge. The input receivers are differential amplifiers that utilize an external reference level which is set to the switching midpoint of the external logic family. To maintain signal integrity of this reference voltage, it is resistively isolated and RC-filtered at each receiver.

The output driver presented a more difficult problem due to the 3.3-V V(DD) chip power supply. To provide a good interface to ECL, it is important that the output driver pull to the V(DD) rail (for ECL operation V(DD)

= 0 V, V(SS) = - 3.3 V). This precludes using NMOS pull-ups. P-channel metal-oxide semiconductor (PMOS) pull-ups have the problem of well-junction forward bias and PMOS turn-on when bidirectional outputs are connected to 5-V logic in CMOS/TTL mode. The solution, as shown in Figure 14, is a unique floating-well driver circuit that avoids the cost of series PMOS pull-ups in the final stage, while providing direct interface to 5-V CMOS /TTL as well as ECL.[5]

Transistors Q1, Q2, and Q6 are the actual output devices. Q1 and Q2 are NMOS devices arranged in cascode fashion to limit the voltages across a single transistor to no more than 4 V. Q6 is a PMOS pull-up device that shares a common n-well with Q7 through Q10, which have responsibility for supplying the well with a positive bias voltage of either V(DD) or the I/O pin potential, whichever is higher. Q3 through Q5 control the source of voltage for the gate of Q6-either the output of the inverter or the I/O pad if it moves above V(DD). R1 and R2 provide 50-ohm series termination in either operating mode.

Caches

The two internal caches are almost identical in construction. Each stores up to 8KB of data (D-cache) or instruction (I-cache) with a cache block size of 32 bytes. The caches are direct mapped to realize a single cycle access, and can be accessed using untranslated bits of the virtual address since the page size is also 8KB. For a read, the address stored in the tag and a 64-bit quadword of data are accessed from the caches and sent to either the memory management unit for the D-cache or the instruction unit for the I-cache. A write-through protocol is used for the D-cache.

The D-cache incorporates a pending fill latch that accumulates fill data for a cache block while the D-cache services other load/store requests. Once the pending fill latch is full, an entire cache block can be written into the cache on the next available cycle. The I-cache has a similar facility called the stream buffer. On an I-cache miss, the I-box fetches the required cache block from memory and loads it into the I-cache. In addition, the I-box will prefetch the next cache block and place it in the stream buffer. The data is held in the stream buffer and is written into the I-cache only if the data is requested by the I-box.

Each cache is organized into four banks to reduce power consumption and current transients during precharge. Each array is approximately 1,024 cells wide by 66 cells tall with the top two rows used as redundant elements. A six-transistor, 98-[m]^2 static RAM cell is used. The cell utilizes a local interconnect layer that connects between polysilicon and active area, resulting in a 20-percent reduction in cell area compared to a conventional six-transistor cell. A segmented word line is used to accommodate the banked design, with a global word line implemented in third-level metal and a local word line implemented in first-metal layer. The global word line feeds into local decoders that decode the lower two bits of the address to generate the local word lines. As shown in Figure 15, the word lines are enabled while the clock is high, and the sense amplifiers are fired on the falling edge of the clock.

7 Summary

A single chip microprocessor that implements a new 64-bit high-performance architecture has been described. By using a highly optimized design style in conjunction with a high-performance 0.75-m technology, operating speeds up to 200 MHz have been achieved.

The chip is superscalar degree 2 and has 7- and 10-stage pipelines for integer and floating-point instructions. The chip includes primary instruction and data caches, each 8KB in size. In each 5-ns cycle, the chip can issue two instructions to two of four units, yielding a peak execution rate of 400 mips and 200 MFLOPS.

The chip is designed with a flexible external interface providing integral support for a secondary cache constructed of ordinary SRAMs. The interface is fully compatible with virtually any multiprocessor write cache coherence scheme, and can accommodate a wide range of timing parameters. It can interface directly to standard TTL and CMOS as well as 100K ECL technology.

8 References

1. Alpha Architecture Handbook (Maynard: Digital Equipment Corporation, Order No. EC-H1689-10, 1992).

2. J. Yuan and C. Svensson, "High-Speed CMOS Circuit Techniques," IEEE Journal of Solid-State Circuits, vol. 24, no. 1 (February 1989).

3. R. Conrad et al., "A 50 MIPS (peak) 32/64b Microprocessor," ISSCC Digest of Technical Papers (February 1989): 76-77.

4. J. Sklansky, "Conditional-Sum Addition Logic," IRE Transactions on Electronic Computing, vol. EC-9 (1960): 226-231.

5. H. Lee et al., "An Experimental 1Mb CMOS SRAM with Configurable Organization and Operation," ISSCC Digest of Technical Papers (February 1988): 180-181.

9 Trademarks

Alpha AXP and VAX are trademarks of Digital Equipment Corporation.

10 Biographies

Daniel W. Dobberpuhl Dan Dobberpuhl received a B.S.E.E. degree from the University of Illinois in 1967. Subsequent to positions with the Department of Defense and General Electric Company, he joined Digital's Semiconductor Engineering Group in 1976. Since that time, he has been active in the design of four generations of microprocessors, including the first single-chip PDP-11 and the first single-chip VAX. Most recently, Dan was the project leader for the first VLSI implementation of Digital's new 64-bit Alpha AXP computing architecture. He is coauthor of the text, The Design and Analysis of VLSI Circuits.

Richard T. Witek Rich Witek joined Digital in 1977 to work on DECnet network architecture during Phase II. In 1982 he joined Digital's Semiconductor Engineering Group where he worked on CAD development, MicroVAX VLSI chips, and a variety of internal RISC projects. Rich was a codesigner of the Alpha AXP architecture and the principal microarchitect of the DECchip 21064 CPU chip. He received a B.A. degree in computer science from Aurora College. Rich is currently employed by Apple Computer, Inc.

Randy Allmon After receiving a B.S. degree in electrical engineering from the University of Cincinnati, Randy Allmon joined Digital in 1981. As a circuit designer in the Semiconductor Engineering Group, he has contributed to the development of numerous high-performance CMOS processors. Currently, Randy is responsible for the technical design and management of a next-generation processor based on the Alpha AXP architecture. He is the coauthor of four high-performance processor papers given at ISSCC and has one patent pending.

Robert Anglin Robert Anglin received S.B. and S.M. degrees in electrical engineering in 1989 from the Massachusetts Institute of Technology. In the same year, he joined Digital's Semiconductor Engineering Group, where he has worked on the design of high-performance microprocessors. Robert is a member of Sigma Xi. He is currently pursuing an M.B.A. degree at Harvard University.

David Bertucci David Bertucci received a B.S.E.E. degree in 1982 from Wayne State University and an M.S.E.E. degree in 1988 from Michigan State University. He joined Digital's Semiconductor Engineering Group in 1989 and worked on advanced CMOS microprocessor design. Currently, he is employed at Sun Microsystems, Inc.

Sharon Britton Sharon Britton received a B.S.E.E. degree from Boston University in 1983 and an M.S.E.E. degree from the Massachusetts Institute of Technology in 1990. She joined Digital in 1983 to work on the design and development of 80186-based controllers for read-only and write-once optical disk drives. Sharon's graduate research involved the development of an integrated content addressable memory system with error detection capability. Currently a member of the Semiconductor Engineering Group, she is involved in the design and implementation of high-performance CMOS microprocessors.

Linda Chao Linda Chao received a B.S.E.E. degree from the Massachusetts Institute of Technology in 1987. Since joining Digital in the Semiconductor Engineering Group/Advanced Development in 1987, Linda has been engaged in the design of microprocessors based on the VAX and Alpha AXP architectures. She is currently pursuing master's degrees in electrical engineering and management through the MIT Leaders for Manufacturing Program.

Robert A. Conrad Robert Conrad received a B.S. degree in electrical and computer engineering from the University of Cincinnati in 1984 and an M.S. degree in electrical and computer engineering from the University of Massachusetts in 1992. In 1981 he joined Digital's Semiconductor Engineering Group, where he worked as a co-op student in the Architecturally Focused Logic Group. Since 1984 Rob has been engaged in the research and development of VLSI microprocessors, including the MicroVAX CPU, a 50-MHz RISC CPU, and most recently the DECchip 21064 microprocessor.

Daniel E. Dever Dan Dever received a B.S.E.E. degree in 1988 from the University of Cincinnati. He joined Digital's Semiconductor Engineering Group in 1988, where he worked on the design and logic verification of CMOS VAX microprocessors. Since 1990 he has been involved in the design of RISC architecture microprocessors, including the floating-point unit of the DECchip 21064 microprocessor. Dan is currently involved in the design of integer arithmetic logic for the next-generation processor based on the Alpha AXP architecture.

Bruce Gieseke Bruce Gieseke received a B.S. degree in electrical engineering from the University of Cincinnati in 1984, and an M.S. degree in electrical engineering from North Carolina State University in 1985. In 1986 he joined Digital's Semiconductor Engineering Group, where he has been engaged in the implementation and circuit design of RISC microprocessors.

Soha M.N. Hassoun Soha Hassoun received a B.S.E.E. degree from South Dakota State University in 1986, and an S.M.E.E. degree from the Massachusetts Institute of Technology in 1988. From August 1988 to August 1991 she was employed at Digital as a custom design engineer in the Semiconductor Engineering Group. She contributed to the design of the floating-point unit of the DECchip 21064 processor. Soha was the recipient of a Digital Minority and Women's Scholarship in 1991 and is pursuing a Ph.D. degree at the University of Washington, Seattle, Computer Systems Engineering Department.

Gregory W. Hoeppner Gregory Hoeppner graduated with distinction from Purdue University in 1979. His research topic was ion-implanted optical waveguides. In 1980 he worked at General Telephone and Electronics Research Laboratory, where he performed basic properties research on GaAs for fabrication of submicrometer FETs. From 1981 to 1992 he held a number of positions at Digital Equipment Corporation's Hudson, MA site, including co-implementation leader of Digital's DECchip 21064.

Kathryn Kuchler Kathryn Kuchler received a B.S. degree in electrical engineering from Cornell University in 1990. Upon graduation, she joined Digital's Semiconductor Engineering Group, where she worked on the first implementation of a RISC microprocessor based on the Alpha AXP architecture.

Maureen Ladd Maureen Ladd received a B.S. degree in computer engineering from the University of Illinois in 1986. She then joined the Semiconductor Engineering Group within Digital and worked on a 32-bit RISC microprocessor. Maureen received an M.S.E. degree in electrical engineering from the University of Michigan in 1990 through Digital's Graduate Engineering Education Program. Upon her return to Digital, she worked on the implementation of the first microprocessor based on the Alpha AXP architecture.

Burton M. Leary Mike Leary is currently a consulting engineer in the Semiconductor Engineering Group/Advanced Development Memory Group. He designed the instruction and data caches for the DECchip 21064 CPU and is currently working on the design of advanced memory products. Mike joined Digital in 1980 after receiving a B.S.E.E. degree from the University of Massachusetts.

Liam Madden Liam Madden joined Digital in 1984 and has since designed both CISC and RISC microprocessors and contributed in the area of CMOS process development. He is currently a consultant engineer in Digital's CPU Advanced Development Group and his interests include circuit design and CMOS technology development. Prior to joining Digital, Liam designed industrial microcontrollers for Mahon and McPhillips, Ireland, and worked for Harris Semiconductor. He received a B.S. degree from University College Dublin in 1979 and an M.E. degree from Cornell University in 1990.

Edward J. McLellan Ed McLellan is a principal engineer in the Semiconductor Engineering Group. He has contributed to the design of several processor chips. Ed joined Digital in 1980 after receiving a B.S. degree in computer and systems engineering from Rensselaer Polytechnic Institute, where he was elected to Eta Kappa Nu. He holds three patents in computer design and has one application pending.

Derrick R. Meyer Dirk Meyer joined Digital's Semiconductor Engineering Group in 1986. He was initially involved in the design of the cache and memory systems for a chilled CMOS VAX processor. He has since been involved in the development of microprocessors based on the Alpha AXP architecture. Prior to joining Digital, he was employed at Intel Corporation, where he was involved in the design of various CMOS microcontrollers, including the 80C51 and 80C196. Dirk received a B.S. degree in computer engineering from the University of Illinois in 1983.

James Montanaro James Montanaro received B.S.E.E. and M.S.E.E. degrees from the Massachusetts Institute of Technology in 1980. He joined Digital Equipment Corporation in 1982. He was a circuit designer on the floating-point chip for the LSI 11/74 and a MicroVAX peripheral chip. He led the physical implementation of the uPRISM CPU, a 70-MHz prototype RISC CPU completed in 1988. James also led the physical implementation of the first CPU chip based on the Alpha AXP architecture and then contributed as a circuit designer for the DECchip 21064 CPU. He is currently with Apple Computer, Inc.

Donald A. Priore After receiving an S.M. degree in electrical engineering and computer science from the Massachusetts Institute of Technology, Donald Priore joined Digital in 1984. Initially, he worked on device characterization, yield enhancement, and yield modeling of NMOS and CMOS processes in manufacturing. Subsequently, he joined a CMOS design group, working first with low-temperature CMOS technology and later with conventional CMOS in high-performance microprocessor design. His interests include signal, clock, and power integrity in the on-chip environment.

Vidya Rajagopalan Vidya Rajagopalan received a B.E. degree in electronics engineering from Visvesvaraya Regional College of Engineering, Nagpur, India, in 1986, and an M.S. degree in electrical engineering from the University of Maryland in 1989. She was with Norsk Data India Ltd. from 1986 to 1987 as a systems design engineer. In 1989 she joined Digital's Semiconductor Engineering Group and was a member of the design team of the DECchip 21064 RISC microprocessor. Vidya is currently involved in the design of high-performance microprocessors.

Sridhar Samudrala Sridhar Samudrala is a consulting hardware engineer in the Semiconductor Engineering Group, where he is currently working on a new CPU chip. He joined Digital in 1977. Since that time, he has worked on the design and verification of PDP-11/23 chips, VAX 8200 microcode development, and on the architecture and design of floating-point chips. He holds two patents and has three patent applications pending, all on floating-point design. Sridhar received an M.Sc. (Tech) degree from Andhra University, India, and an M.S.E.E. degree from the University of Wisconsin.

Sribalan Santhanam Sri Santhanam received a B.E. degree in electrical engineering from Anna University, Madras, India, in 1987, and an M.S.E. degree in computer science and engineering from the University of Michigan in 1989. Upon graduation, he joined Digital as a design engineer for the Semiconductor Engineering Group, responsible for the full-custom design and development of high-performance CMOS VLSI processors. Sri worked on the design of the floating-point unit of the DECchip 21064 CPU. He is currently involved in the design of another high-performance microprocessor.