PowerPC AS A10 64-bit RISC microprocessor

by J. W. Bishop
M. J. Campion
T. L. Jeremiah
S. J. Mercier
E. J. Mohring
K. P. Pfarr
B. G. Rudolph
G. S. Still
T. S. White

The PowerPC AS™ A10 64-bit RISC microprocessor is a 4.7-million-transistor integrated circuit design, using IBM CMOS 5L 0.5-μm, 3-V, four-level-metal ASIC technology. Support for the PowerPC AS architecture is implemented in a 213-mm^2 die using a semicustom design methodology. Chip density and speed are enhanced through the use of custom macros and multiport arrays. An on-chip phase-locked-loop circuit is used to reduce chip-to-chip clock skew. Full utilization of the four-level-metal interconnect technology was achieved through architectural floorplanning, performance clustering, and timing-driven placement and wiring, with a total wire length of over 102 meters placed on the 14.6 × 14.6-mm die. The microprocessor is a pipelined, superscalar design with five separate functional units, a 4KB instruction cache, and an 8KB data cache. The design includes parity, error-correction, and error-logging functions, as well as self-test for logic and arrays during power-on. The design is robust and implements a wide range of performance configurations at the system level, allowing direct attachment of DRAM to the processor, or high-performance L2 cache options using high-speed SRAM. An on-chip system I/O bus and bus controller are provided for attachment of peripherals.

Introduction

The IBM AS/400® mid-range computing system, a leader in the marketplace since its inception in 1988, is designed for ease of use and the preservation of the customers' investments. More than 300000 installed systems, with software provided by some 8000 software vendors, attest to the popularity of the system. One of the major achievements of the AS/400 is its ability to maintain software compatibility while providing growth in capacity, performance, and function.

To sustain a performance growth rate averaging 40% per year, the AS/400 switched from its original internal microprogrammed interface (IMPI) processor architecture to a reduced-instruction-set computing (RISC) platform. The PowerPC™ architecture was chosen as a base, with several instructions and features added to optimize performance and support unique AS/400 operating system requirements. The resulting architecture, PowerPC AS™, is implemented on two new RISC microprocessors, the PowerPC AS A30 and the PowerPC AS A10. The A30 is a high-performance one-way, two-way, and four-way multiprocessor for the high end of the AS/400 product line. The A10, described here, is a uniprocessor with a broad range of performance capabilities to support entry, low-end, and mid-range traditional systems and server systems.

When describing performance, AS/400 developers focus on overall system performance in the commercial environment. Typically, system performance is specified in relative performance ratio (RPR) units. The AS/400 9404

0018-8646/96/$5.00 © 1996 IBM

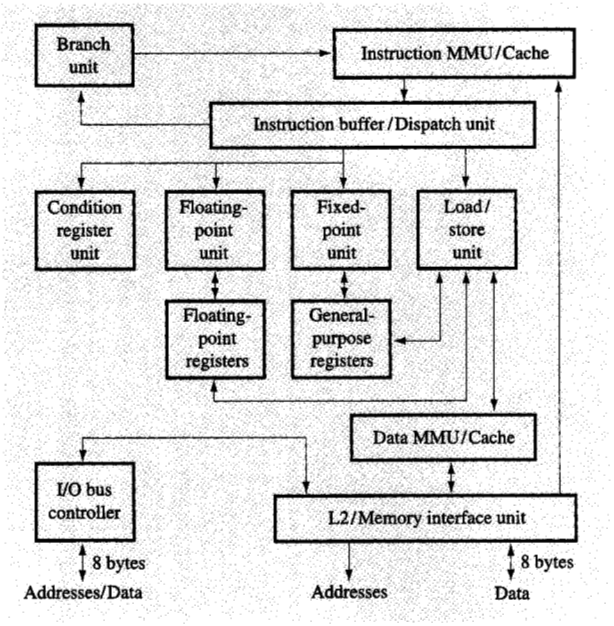

Figure 1

Architecture of A10 processor chip.

B10, with 16 MB of main storage and 945 MB of DASD, is the base unit of performance (*RPR* = 1.0), using the RAMP-C workload, with approximately 70% CPU utilization. The requirement for A10-based systems was to span the range from approximately 4 RPRs to 28 RPRs, while the A30-based systems would provide higher performance.

Not surprisingly, a number of conflicting objectives drove the decision-making process used to establish the design point and technology selection for the A10:

- As an entry processor, the A10 had to have absolute minimum cost in the processor and memory subsystem. As a result, direct attachment of DRAM single in-line memory modules (SIMMs) and a single-chip processor with minimum die size were of paramount concern. Up to a point, performance could be sacrificed to achieve low cost.
- Low-end systems could tolerate some additional cost, but required support for vastly more memory, implemented on cards instead of SIMMs.
- Mid-range systems performance requirements were in part driven by the capabilities of the more powerful A30. The A10 needed enough performance at its upper limit to prevent a major gap between it and the low end of the A30-based models. To achieve this level of performance in the commercial environment, a storage system with robust bandwidth was required.
- Available development resources and schedule, coupled with the fact that the A10 was a brand-new design, did not permit serious consideration of a full custom design, although such a design would likely have resulted in a somewhat smaller chip size and perhaps better performance.

To mitigate the effects of not being able to design a full custom chip, it was decided to use the best available CMOS chip technology, along with state-of-the-art ASIC design tools. Indeed, it turned out that the tools, technology, and processor design all underwent concurrent development. The A10 was the largest ASIC ever designed using the selected tools. As might be expected, the A10 design process explored the limitations of nearly every design tool used, and served as a vehicle to help develop technology rules.

On the basis of emerging trends, it was decided to survey the marketplace for a suitable 0.5-μm CMOS ASIC technology. Using a comparative study, the IBM 0.5-μm CMOS 5L ASIC product was chosen as the best match for performance, schedule, and cost objectives. One major advantage of CMOS 5L was its growable RAM (GRAM) and multiport growable register array (GRA) capability, which allows users to generate a wide variety of on-chip SRAMs and register arrays using a highly automated process, resulting in structures that are guaranteed correct by construction.

Logic generation is done primarily by synthesizing VHDL to the target technology, with a small number of custom logic macros such as adders, comparators, and memory ECC blocks used in critical areas. The bulk of the design is implemented in standard cell logic blocks, with gate array blocks used in support of I/O cells and metal-only engineering changes (ECs).

The remainder of this paper describes in further detail the process of designing and building what is possibly the largest and most complex ASIC to date, and the results of that effort.

A10 functions

The microprocessor chip, the architecture of which is shown in **Figure 1**, is a pipelined, superscalar design with separate fixed-point, floating-point, load/store, condition register, and branch processor pipelines. Up to three instructions can be dispatched in a single cycle. Instructions are dispatched in order. Instruction execution and completion are also performed in order, avoiding the need for complex register-renaming schemes and reorder buffers. On the basis of preliminary performance modeling, it was decided that measurable performance improvement could be achieved by employing branch prediction in conjunction with a simple superscalar design. It was further decided that out-of-order instruction

execution would introduce too much design complexity, and its effects on performance would be smaller in the commercial processing environment, which tends to have a large number of storage accesses compared to engineering/scientific workloads. The frequent storage accesses emphasize the characteristics of the storage system and somewhat obscure those of the execution pipelines. Also, the clock rate of the microprocessor should be balanced with the performance of the memory. The degree to which this strategy was successful has been quantified in an audited benchmark [1]. The report states that the AS/400 50S Model 2121 server based on the A10 microprocessor running at 77 MHz achieved a TPC-C rating of 914.05. In the same report, two other comparable systems with single-chip processors were benchmarked for comparison. The first system, running at 96 MHz, achieved a 728.73 TPC-C rating, while the second system, running at 100 MHz, performed at 735.27 TPC-C.

The pipelines are four stages deep: instruction fetch, decode, execute, and putaway. Most instructions require only a single execute cycle, except for a few fixed-point operations and most floating-point computational operations, which are relatively infrequent in current commercial workloads. To support an architected multiply–add instruction (MAD), a single four-pass add/shift data flow was implemented in the floating-point unit. The main floating-point dataflow consists of a carry-save adder (CSA) tree, a 160-bit B-operand shifter, and a 106-bit full adder with a 58-bit incrementer. The multiplier is relatively fast yet conservative in cell utilization, producing a multiply or multiply–add or multiply–subtract in five machine cycles. Results of all calculations are postnormalized and rounded to IEEE floating-point specifications.

A 4KB instruction cache and an 8KB data cache are provided on-chip, along with a DRAM controller which allows a wide variety of memory configurations supporting a broad performance spectrum. Entry systems use page-mode SIMMs, while low-end systems use page-mode memory cards. The higher-performance models support the use of toggle-mode DRAM cards. To extend the performance range even further, the processor chip is provided with an L2 cache interface which allows external SRAMs to be attached to its memory data port in lieu of DRAMs. The storage controller, a second ASIC chip, attaches to the same interface and takes over the task of interfacing with memory, along with providing the L2 cache directory arrays. The storage controller chips are used in pairs to double the size of the memory interface for increased bandwidth and performance.

A system I/O bus and bus controller are implemented on the chip, allowing attachment of AS/400 family I/O processors and devices.

Table 1 A10 chip CMOS 5L cell usage.

Function	*Cells used**	*Percentage of total used cells*	*X cell†* (μm)	*Y cell†* (μm)	*Cell area* (μm^2)
Logic	680,743	50	3.6	28.8	103.7
Custom macro	90,000	7	3.6	28.8	103.7
GRAM 2 port	272,912	20	8.1	13.3	107.7
GRA 1–2 port	69,625	5	54.0	21.6	1166.4
GRA 3–6 port	251,084	18	43.2	32.4	1399.7
Totals	1,364,364	100			

*Cells used are in logic cell equivalents.
†SRAM cell dimensions are for one bit of the respective memory element.

Substantial error checking is provided. Memory is covered by package-correcting ECC selected to handle the four-bit-wide memory modules used in all configurations. The ECC code can correct an entire module failure and can detect two module failures. Scrubbing is employed to clean memory of soft errors, while redundant bit steering provides additional capability to bypass hard failures. All GRAMs and GRAs on the chip are covered by ECC or parity, which allows most single-bit errors to be corrected and recovered. The floating-point unit is checked using a unique residue error checker. The majority of the dataflow also maintains parity, which adds to the error-detection capabilities.

The robust superscalar design, along with a highly configurable memory system and extensive error-handling capability, results in a substantial circuit count and a challenge for the design system.

Chip profile

The following is a brief summary of the salient features of the A10 processor, achieved as a result of the ASIC design process and CMOS 5L technology. Cell usage by circuit type is shown in **Table 1**.

Technology

- IBM Microelectronics Division (IMD) CMOS 5L 0.5-μm four-level metal [2].
- 32-mm module, 10-level enhanced ceramic ball grid array (CBGA) with 25 × 25 surface mount connection matrix.
- Custom macros used for critical functions, e.g., compare, add, error correction (ECC), multiport registers.
- Instruction and data caches built with two-port growable GRAMs.
- Extensive use of multiport grown register arrays (two to six ports).

Physical chip

- 14.6-mm-square die size.
- 517-module signal I/O.

- 1797408 cells available.
- Global connections = 399007.
- Number of signal nets = 155056.
- Total wire length on chip = 102 meters.

Circuit

- 1364364 cells used for functions—76% of available cells (Table 1).
- At 1.33 cells per circuit: 1025837 equivalent two-way AND-inverters [3].
- Number of transistors used = 4715495 out of 6000000 allocated in used cells.
- Average block power level 2.1 (min/max power levels = 1.0/4.0).
- 17.7 W worst-case (WC) chip power, 13.4 W nominal chip power at 77 MHz.

Performance

- Timing closure to 13.0-ns WC process and 60°C device junction temperature, at 3.4 V.
- Early- and late-mode statistical timing closure methodology (see the section on timing verification).

Design methodology

It was decided to make extensive use of timing-driven logic synthesis to facilitate design productivity. Delay rules for all array and logic components, including the custom macros, were generated and provided to the synthesis program.

The logic was initially partitioned into groups that were intended to have well-defined functional interfaces. The functional logic partitions were in some cases further subdivided to better distribute design workload or to obtain better logic synthesis timing results.

The chip design was specified using an IBM proprietary macro language as a preprocessing facility in generating VHDL 1076 concurrent style code. This resulted in a logical description of the chip in a register transfer language (RTL) format. The macro language offered advantages in coding style and quantity, as well as reducing the need for VHDL 1076 language expertise. The macro language also helped provide the required set of attributes for controlling logic synthesis. The IBM BooleDozer™ logic synthesis program [4] was used to transform the compiled VHDL, represented in a technology-independent (TI) logic structure, into technology-dependent (TD) logic. BooleDozer allowed the designer to help control the results of the synthesis, as explained below.

The first stage of BooleDozer synthesis was generating a TI logic model. This process would either transform the input logic model into NAND (AND-invert) logic blocks or retain higher-level logic specifications such as selector, AND–OR logic, and XOR (exclusive-OR) trees, under the control of the design style specified using the macro language. These transformation options were advantageous for control- and data-flow-type logic. BooleDozer then mapped the TI logic into TD logic.

An accurate timing analysis program and detailed timing specifications, which were coded in VHDL macro statements, were used to guide the generation of TD logic with appropriate consideration for path length. A key element of the timing-correction process was to move timing-critical signals to reduce the number of logic stages between the critical signal and an output of a cone (or tree) of logic blocks. Fan-out repowering, logic book power level selection, and noncritical path area optimizations were also performed. The designer was able to control the effort BooleDozer employed in this process by using options to work hardest on critical paths or to distribute the timing-correction effort across a wider portion of the logic. There were trade-offs involved in this process, and an iterative and somewhat experimental approach was often necessary to obtain desired timing results. A key feature in the whole design and synthesis process was the ability of the designer to code the equations precisely for timing-critical portions of logic and have BooleDozer generate the TD logic while preserving the precisely coded structure. This capability was achieved by features in the VHDL macro language which allowed the generation of appropriate directives to control BooleDozer.

Overall, the effectiveness of VHDL coding and the use of BooleDozer synthesis allowed for a fairly high-level coding style for noncritically timed logic, mixed with detailed logic coding, which was often necessary to achieve proper timing.

Table 2 shows the logic partition sizes used in BooleDozer. The number of TI logic blocks at the start of the logic synthesis and the final synthesized TD block partition sizes are shown. The rightmost column shows the normalized run time in seconds needed to synthesize each logic partition. Most synthesis was done on RISC System/6000® Model 550 servers, and the run-time statistics generated by the synthesis program are normalized to a value that would represent a machine running at 100 MHz, regardless of the actual processor speed. The table indicates that a partition size of 13085 logic blocks was synthesized successfully. The table also shows that synthesis CPU time is dependent on logic model size, but there were other factors (such as timing correction requirements) which tended to make synthesis CPU time vary considerably. The table shows actual partitioning, which was dependent on a number of factors, a significant one being the predilection of the designers working on a given functional entity.

Appendix A gives an example of results of BooleDozer logic synthesis on one of the logic partitions. Logic utilization summaries at the start of synthesis, after TI mapping, after TD mapping, and after TD timing optimization are shown.

A chip floorplan was developed and modified as synthesis output became available. As a complete chip design became available, it was stitched together, and full-chip timing runs could be performed. The final placement and timing runs were actually run only five weeks before the required chip release date. At this time, the clock tree was also incorporated into the design, timing problems were corrected, and final physical design performed, followed by test-pattern generation.

Custom logic macros

The need for several different custom logic macros was recognized early in the design phase. It has frequently been the case in past designs, as here, that critical paths in the logic often involve adders, comparators, and multiport data pipelining registers, so a 64-bit adder macro, several comparators, and several pipeline register macros were provided. Because of the complexity of the ECC package correction code, it became apparent that memory error correction could not be done in one machine cycle using synthesized logic. Consequently, it was decided to also provide an ECC generation macro and an ECC correction macro.

Processor logic designers provided a high-level design for the custom functions, along with timing objectives, to the circuit design group contracted to provide the circuits, layouts, and rules. The macros were developed by the PD group at the IBM Haifa Research Laboratory (HRL) during the initial design phase. Rules were provided in time to allow inclusion of the macros in the initial chip release. As timing rules became available, they were inserted into the chip timing closure process. The custom macros themselves have performed flawlessly from the first hardware built.

The 64-bit adder macro serves to illustrate the performance improvement possible through the use of careful application of customization in an ASIC environment. The adder receives two 64-bit inputs and a carry-in to produce a 64-bit sum and selected carry signals. To achieve maximum performance, a full carry-lookahead architecture was selected. The logic design of the carry-in stage and the most significant sum bit enabled the 64-bit sum to be computed in seven logic stages. At best, synthesis only produced a design with approximately 7-ns critical path delay. With custom circuit design and layout, tuning of device sizes, and careful distribution of the loads and fan-outs of each stage, a worst-case delay of less than 4 ns through the critical path was achieved for the adder macro.

Table 2 A10 chip 64-bit RISC microprocessor—logic partitions.

Logic partition	*TI blocks*	*TD blocks*	*Run time*
Bus Interface Unit	3293	3765	8738
Address Translation part 1	4351	5539	12202
Address Translation part 2	1867	2698	6579
Address Translation part 3	2743	4769	13412
Data Cache part 1	7324	7792	29662
Data Cache part 2	1561	1493	1484
Chip I/O	1505	1592	4535
Floating-Point part 1	4430	6676	5890
Floating-Point part 2	2848	2138	2680
Floating-Point part 3	9081	5965	11173
Floating-Point part 4	3269	2606	3207
Floating-Point part 5	4840	1666	2653
Floating-Point part 6	3925	2116	3491
Floating-Point part 7	1382	786	705
Floating-Point part 8	4377	1700	4311
Floating-Point part 9	1576	1781	2532
Fixed-Point Unit	12265	13085	34128
Display/Scan Control Unit	2044	2294	1061
Branch Predict Unit	3631	3196	8837
Instruction Unit part 1	7058	5021	14860
Instruction Unit part 2	14570	9690	27480
Instruction Cache part 1	1052	1148	1324
Instruction Cache part 2	44	62	114
Instruction Cache part 3	822	820	649
Load/Store Unit part 1	9632	9665	21779
Load/Store Unit part 2	2870	4332	29726
Memory Interface Unit part 1	10714	8560	26084
Memory Interface Unit part 2	4940	5630	15098
Storage Control Unit	6054	7105	22709

Clocking

The A10 chip is an LSSD (level-sensitive scan design) design. Each timing state element is an SRL (shift register latch) containing two discrete latches known as L1 and L2 latches. The use of separate clocks for the data capture (L1) and data launch (L2) latches requires distribution of clock pairs to the SRLs.

The clocking for the A10 chip consists of three primary domains and several secondary domains. The primary domains include the main processor domain, which comprises 75% of the SRLs clocked at 77 MHz, an I/O bus domain clocked at 26 MHz, and a separate DRAM refresh domain that is continuously clocked to support the continuously powered memory feature. The DRAM refresh domain clocks are cycled at 77 MHz in normal mode and at 10 MHz in the battery backup mode. The secondary domains include the clock control and JTAG (joint test action group) scan interface domains. In normal functional mode, the clocks for the primary domains and the clock control portion of the secondary domains are generated via an external analog clock module that resides on the processor card; they are distributed as low-voltage differential pairs. The clocks for the refresh domain in the battery backup mode are generated from an external

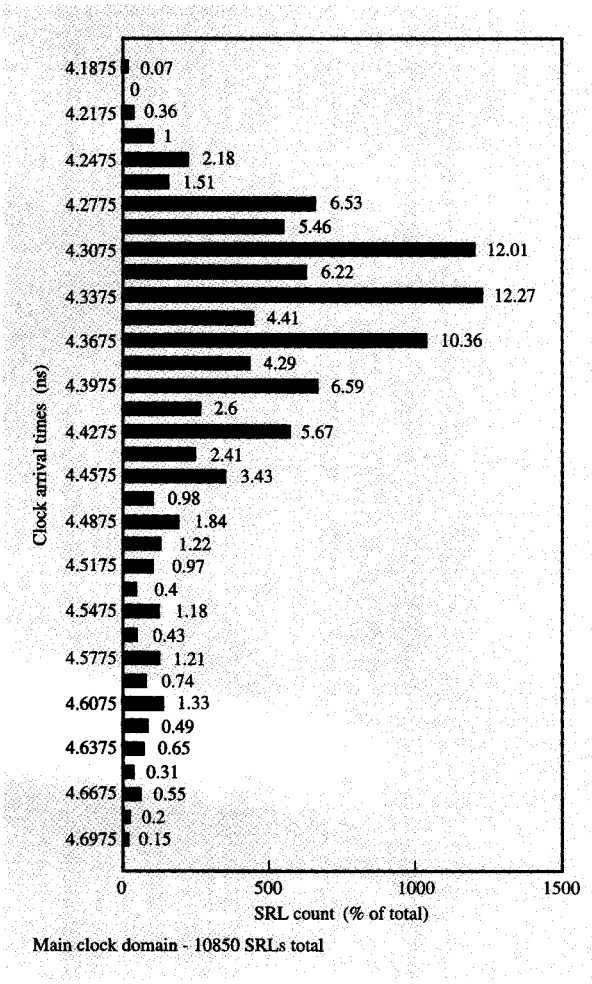

Figure 2

A10 chip clock arrival times vs. SRL count.

oscillator module, also located on the processor card. The clocks for the JTAG scan interface are distributed by a separate service processor.

An on-chip analog PLL (phase-locked-loop) circuit is utilized on the A10 chip and the synchronous storage controller chips to phase-align the clocks in the main processor domain. A 30% reduction in the chip-to-chip clock skew was achieved through the use of this technique. This 0.75-ns reduction in clock skew represented a 6% performance increase, which was worth the added design complexity and additional chip area required to accommodate the PLL [5].

The physical design tools and methodology utilized for optimizing the on-chip clock distribution [6] did not restrict the placement of the more than 15000 SRLs. The distribution was further complicated by the high percentage of relatively large and irregularly placed array macros. The first two levels of a four-level clock-repowering tree were preplaced to aid in the reduction of on-chip clock skew. The wire lengths and associated wire delays for the first two levels of the clock distribution were manually balanced to near zero skew. The first level of the clock distribution utilized wide wires to minimize the tracking component of skew. In addition, the wide and isolated wires helped to reduce the clock tree latency and the clock transition times feeding the second stage of the tree. The remaining two levels of the on-chip clock distribution and repowering tree were automatically optimized by the physical design tools. The latency of the on-chip clock tree was approximately 4.5 ns, with on-chip physical design skews in the 150–500-ps range. **Figure 2** shows a histogram of the data capture clock arrival times to the SRLs in the main processor domain of the A10 chip. The histogram demonstrates that approximately 75% of the clocks in the processor domain arrive at the SRLs within a 150-ps window, an additional 15% arrive within a 300-ps window, and the remaining 10% arrive within the 500-ps total window.

In order to help reduce the number of SRL-to-SRL short-path (hold time) timing problems on the chip (caused by clock overlap resulting from clock skew), an inverter circuit with a typical delay of about 150 ps was used to separate the launch clocks from the capture clocks. This helped to reduce the amount of chip area that was required to pad short logic paths between SRLs and resulted in only a slight impact on the chip performance.

Timing verification

Timing verification is the process used to ensure that the A10 chip achieved the manufacturing yields and product performance targets from the perspective of processor cycle time and off-chip timing requirements. A number of tests were employed which in turn were run at simulated environmental stress conditions to achieve testing at extremes of functionality.

The timing verification was divided into on-chip and off-chip categories. Both types of verification are concerned with the following types of timing analysis. Early-mode analysis is used for testing that signals do not arrive too soon (specifically, that a signal transition intended for a future clock pulse does not arrive during the current clock). In early-mode analysis, signals are propagated as fast as possible through logic, and the earliest arriving signal is the one considered to control the output of a logic block. Late-mode analysis is used for testing that signals propagate in time for the intended (typically, the current) clock event. In late-mode analysis, signals are propagated as slowly as possible through logic, and the latest-arriving signal is the one considered to control the output of a logic block.

In both timing analysis modes, the following tests are performed:

- Setup — Tests for data path arrival in time for an associated clock event, and is important in late-mode analysis.
- Hold — Tests for data stability during an associated clock event, and is important in early-mode analysis.
- Transition time — Tests for excessive signal rising or falling transition times with respect to limits imposed by the technology rules.
- Clock pulse width — Tests for clock pulse widths that are below acceptable technology limits.
- Clock separation and cycle time — Tests for other aspects of clocking defined by specific technology rules (e.g., static arrays).

In any test, the difference between data path timing and the appropriate test (clock or absolute time requirement) is termed the "slack" time for the test. A negative slack indicates that the timing requirements of a particular test were not satisfied. The objective of timing verification is to achieve zero or positive slack on all tests. An additional test step was to evaluate timing at extremes of voltage and temperature. This was achieved by use of voltage and temperature circuit delay modifiers in the CMOS 5L delay rules.

On-chip timing analysis was accomplished using ETE (Early Timing Estimator), which is a host-based IBM legacy tool. The timing tool was used in a statistical mode in both early-mode and late-mode analysis. In statistical mode, variations between clock and data path propagation delays based on worst- and best-case delay rules, and path-to-path tracking assumptions, were applied.

The A10 system worst-case environmental conditions were set at 3.47 V and 60°C chip junction temperature, but timing runs were made at 3.4 V and 85°C worst case, 3.8 V and 0°C best case, and 3.6 V and 25°C nominal. This provided a more pessimistic analysis to force work on the most negative slacks. In early-mode analysis, timing was closed to allow no negative slacks for both on-chip and off-chip timing. Late-mode on-chip timing tests at 13 ns verified that the chip met all timing specifications at worst-case environmental conditions.

Chip-to-chip late-mode critical nets were identified early in the design process. These nets were then modeled in an electrical simulator (ASX), and an optimum topology and resulting delay were obtained. The results were applied to classes of nets (e.g., SRAM address, cache control, DRAM data) and were used for generating a set of card net timing conditions. These conditions were then used in off-chip timing verification. Accurate A10 chip input times were used to verify that input signals met all of the test requirements specified by the on-chip timing analysis. A10 chip output signals were used with accurate net delay adders to ensure that other system-level timing model specifications, e.g., DRAM chip timing, were met. All chip-to-chip nets met the 13-ns system timing conditions. Chip-to-chip paths (driver plus card net delays) ranged from 3 ns to 8 ns. At the same time, the driver *DI/DT* was monitored to ensure that the design stayed within the simultaneous switch noise budget.

The hardware was tested and found to meet all performance criteria using parts that were specially fabricated to give a broad range of process variations (from slow to fast). A test matrix was specified which included these variations. In all systems configurations, the A10 microprocessor met performance specifications at worst-case environmental conditions.

Physical design

Thirty-four large macros (six 2-port-GRAM arrays, 23 multiport GRAs, three 64-bit adders, and two ECC macros) were preplaced on the basis of known critical paths and the chip dataflow. Primary chip inputs and outputs were also preassigned on the basis of the macro placement and card/module wiring constraints.

Placement of the remaining components was accomplished using MCPLACE, an internal IBM simulated annealing placement program. A two-pass placement approach was used. The first-pass placement ignored nets which are logically equivalent (i.e., clock nets and large logic net powering trees). An optimization step was run to reconnect these nets on the basis of the placement of the net sinks. The second-pass placement was made with these new net connections, ignoring placement information from the first pass. A cleanup optimization of logically equivalent nets was applied after the second-pass placement.

The placement steps included timing-driven capacitance targets which were derived from the prephysical design critical paths. Although floorplanning region constraints were not used, postplacement analysis (see **Figure 3**) showed that the major functional units remained within fairly tight groupings (especially those units that contained preplaced macros). After placement, the circuit power levels were optimized on the basis of timing runs with capacitance and *RC* (wire resistance times total capacitance delay adder) delay estimates derived from the placement.

At this point the design was converted to a Cadence CELL3 database. The CELL3 global router was used to produce more accurate capacitance and *RC* estimates. Additional critical path optimization was done at this point. Most of this path tuning was done on individual paths rather than applying a more comprehensive solution (e.g., power level optimization of all blocks). Also at this time, early-mode timing problems were solved using an

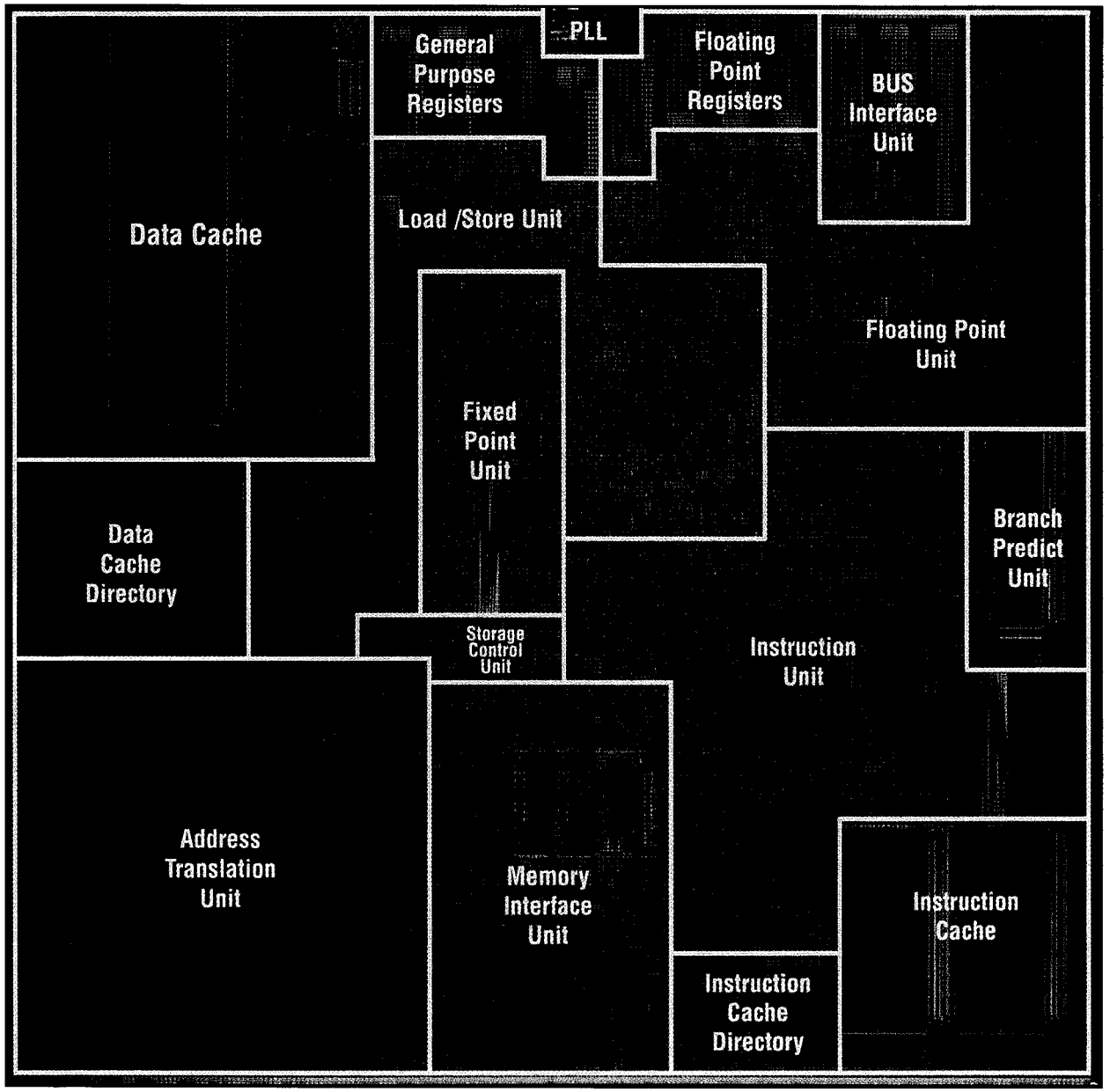

Figure 3

Microphotograph of A10 chip.

analysis tool which attempted to solve the early-mode problem without creating a new late-mode critical path.

The design was then final-routed using the Cadence CELL3 router. At this point, cycle-time-limiting paths were manually tuned and optimized using the engineering change order (ECO) capability of CELL3.

For the final pass of this part, a period of five weeks was spent in the physical design process. Measured run times (on a RISC System/6000 Model 590) for the CPU-intensive processes were approximately as follows:

- First placement job – 24 hours.
- Second placement job – 72 hours.
- CELL3 global route – 5 hours.
- CELL3 final route – 14 hours.

The remainder of the physical design process included working concurrently on

- Critical path optimization:
  - Net repowering.
  - Rerouting scenic wiring.
  - Critical path redesign/reimplementation.
- Early-mode timing closure.
- Meeting technology constraints with routed capacitance/*RC* values (e.g., net transition time violations).
- Incorporating functional changes for design problems found in simulation during the physical design process.
- Fixing routing problems (e.g., shorts, geometry violations).
- Running physical design checks and fixing errors.

The magnitude of the physical design process change activity is shown in **Table 3**.

Table 3 A10 chip physical design change summary.

Type of change	*Changes from PD entry to first B-release*	*Changes from first B-release to final B-release*
Block type changed	2592	116
Block power decreased	17451	0
Block power increased	10448	72
Block added	6548	2995
Block deleted	16	1
Net connection changed	63572	2289

The vehicle for specifying the chip design to manufacturing is known as Cadence LEF/DEF (Library Exchange Format/Design Exchange Format) release files. These release files can be thought of as two entities, an A-release, which specifies, among other things, the device layouts, and a B-release, which specifies the metallization and circuit interconnection information. When both the A- and B-release data are provided at the same time, it is known as an A/B release.

After the initial chip release (A/B), four additional B-releases were processed to incorporate functional logic fixes as well as additional timing cleanup. These B-releases were again processed using the ECO capability of CELL3 (i.e., all placement and wiring were maintained except for the logic affected by the changes). The changes were implemented by using spare gate array filler cells and by rewiring existing cells. Each B-release provided timing results equal to or better than those of the previous B-release. The magnitude of the B-release changes is also shown in Table 3.

Conclusions

Using the IBM CMOS 5L ASIC technology and associated design tools, a small design team was able to produce a 64-bit PowerPC AS microprocessor optimized for use across a wide range of AS/400 systems. A performance range of approximately 4 to 28 RPRs is supported, using various cache and memory configurations. At 4.7 million transistors, this chip represented the largest ASIC chip ever built within IBM.

The logic description language and synthesis tools allowed both high design productivity and detailed control in timing-critical areas. The ability to use logic macros allowed further optimization of timing-critical paths to achieve the A10 cycle time goals. The growable RAM and multiport growable register array capabilities of the technology also were very valuable in saving both design time and chip space.

Timing-driven placement and automated wiring tools were able to support aggressive performance requirements with a very large design. The ASIC design approach permitted a relatively short physical design period and fast turnaround of metal-only changes, reducing overall design schedule and resource investment.

Although treatment of it is beyond the scope of this paper, extensive simulation was done using random instruction patterns, as well as focused test cases in error-prone areas. The result was prototype parts that supported the OS/400® operating system development efforts. The A10 microprocessor achieved customer ship status in late 1995. Further, a robust logic organization, flexible design techniques, and the ASIC technology capabilities support design reuse and extendibility into future microprocessor offerings.

Acknowledgments

We would like to acknowledge our colleagues: In the IBM Microelectronics Division, Charles Ng, ASIC circuit development and applications, Endicott, NY, and John Chickanosky, CMOS 5L ASIC development, Burlington, VT, and their teams, for CMOS 5L technology applications support, and technology enhancements for the PowerPC AS A10 processor chip; Moshe Leibowitz, Ayal Bar-David, Alex Shnayder, Israel Wagner, Eyal Bin, and Rony Kay, of the circuit design (PD) group in the IBM Haifa Research Laboratory, Haifa, Israel, for the development of the A10 chip custom macros; John Doyle, formerly with the IBM AS/400 Division, Rochester, MN, for his outstanding contributions to the A10 chip placement, routing and checking, and physical design tool development; Jim Chambers and Mark Gildea in AS/400 Advanced Series ASIC CMOS processor development, for their review and helpful suggestions on the paper.

References and note

1. D. H. Brown Associates, Inc., "AS/400 $385/tpmC Challenges UNIX Servers," *Technology Trends*, pp. 1, 2 (November 21, 1995).
2. *CMOS 5L ASICS Logic Products Databook*, No. ADC05LDBU-01, IBM Microelectronics Division, Essex Junction, VT 05422, March 1995.
3. R. Gregor, C. Ng, J. Libous, E. Carter, R. Beaudoin, A. Chu, D. Grindel, J. Kinney, M. Lee, L. Mentes, J. Oppold, M. Russell, A. Secor, and G. Yenik, "One-Million-Circuit CMOS Basic Logic Family," *Proceedings of the Custom Integrated Circuits Conference*, 1993, pp. 23.1.1–23.1.4.
4. L. Stok, D. S. Kung, D. Brand, A. D. Drumm, A. J. Sullivan, L. N. Reddy, N. Hieter, D. J. Geiger, H. Chao, and P. J. Osler, "BooleDozer: Logic Synthesis for ASICs," *IBM J. Res. Develop.* **40**, No. 4, 407–430 (1996, this issue).
5. I. I. Novof, J. Austin, R. Kelkar, D. Strayer, and S. Wyatt, "Fully Integrated CMOS Phase-Locked Loop with 15 to 240 MHz Locking Range and Plus or Minus 50 ps Jitter," *IEEE J. Solid-State Circuits* **30**, No. 11, 1259–1266 (1995).
6. J. J. Doyle, J. A. Lawrence, and R. F. Lembach, "Method for Optimizing Clock Trees on VLSI Chips," *IBM Tech. Disclosure Bull.* **33**, No. 8, 288–291 (January 1991).

Received December 11, 1995; accepted for publication March 21, 1996

Appendix: BooleDozer synthesis output example

```
Instruction Unit Part 2 - BooleDozer Statistics
After initial mapping to TI logic:

This section shows the details of the initial logic model
that BooleDozer will transform.

The model (Instruction Unit Part 2) has:
 Primary Inputs       =                599
 Primary Outputs      =                503
 Average Fanout       =          2.335855
 Real signals         =              15420
 Real boxes           =              14570
Type Cnt   Boxname                    Function
 3659      AND                      >      AND
 1763      AO                       >       AO
  172      AOI                      >      AOI
    3      COMPARE                  >  COMPARE
    1      CONSTANT                 > CONSTANT
   44      DECODE                   >   DECODE
    6      CHTEB                    >   CUSTOM (COMPARE MACRO)
 4364      IDENT                    >    IDENT
 1102      IOPAD                    >    IOPAD
  862      NAND                     >     NAND
  158      NOR                      >      NOR
  213      NOT                      >      NOT
    1      OA                       >       OA
 1454      OR                       >       OR
 1286      REG                      >      REG
   48      SELECTOR                 > SELECTOR
   12      XNOR                     >     XNOR
  524      XOR                      >      XOR

After final mapping to TI logic:
This section shows the final TI model produced by BooleDozer.
Most of the logic has been transformed into NAND blocks. The
other logic types were preserved under the control of directives
passed to BooleDozer from source code VHDL macro directives.

The model (Instruction Unit Part 2) has:
 Primary Inputs       =                599
 Primary Outputs      =                503
 Average Fanout       =          2.533253
 Real signals         =              13781
 Real boxes           =              13181

Type Cnt   Boxname                    Function
   24      AND                      >      AND
 1761      AO                       >       AO
  172      AOI                      >      AOI
    1      CONSTANT                 > CONSTANT
    6      CHTEB                    >   CUSTOM (COMPARE MACRO)
   30      IDENT                    >    IDENT
 1102      IOPAD                    >    IOPAD
 9475      NAND                     >     NAND
   57      NOR                      >      NOR
   19      NOT                      >      NOT
    1      OA                       >       OA
   18      OR                       >       OR
 1286      REG                      >      REG
  331      XOR                      >      XOR
After mapping to TD logic:
This section shows that the logic model has been transformed
into CMOS5L technology low power level books.

The model (Instruction Unit Part 2) has:
 Primary Inputs       =                599
 Primary Outputs      =                503
 Average Fanout       =          3.082237
 Real signals         =              8550
 Real boxes           =              7949

Type Cnt   Boxname                    Function
  335      CHLEA, etc. (3 unique)   >      AND
 1206      CHE1A, etc. (15 unique)  >       AO
  335      CHM0A, etc. (14 unique)  >      AOI
   27      CHPMA                    >  COMPARE
   11      CHPUA, etc. (2 unique)   > CONSTANT
    6      CHTEB                    >   CUSTOM (COMPARE MACRO)
   30      CHQQA                    >    IDENT
 1102      IOPAD        IOPAD       >    OPAD
 1278      CHPGA, etc. (3 unique)   >     MUX
 1001      CHZBA, etc. (5 unique)   >    NAND
  175      CHLRA, etc. (4 unique)   >     NOR
 1164      CHZAA                    >     NOT
   26      CHN3A, etc. (6 unique)   >      OA
  435      CHOFA (8 unique)         >     OAI
   46      CHL4A, etc. (3 unique)   >      OR
 1286      CHRGA, etc. (4 unique)   >     REG
  458      CHMJA, etc. (2 unique)   >    XNOR
  130      CHQRA, etc. (6 unique)   >     XOR

After TD timing optimization:
This section shows the final logic synthesis result.
CMOS5L books have been added for timing correction and
optimization, specifically buffering and cloning of logic.
Block power level optimization has also been done.

The model (Instruction Unit Part 2) has:
 Primary Inputs       =                599
 Primary Outputs      =                503
 Average Fanout       =          2.803264
 Real signals         =              10291
 Real boxes           =               9690

Type Cnt   Boxname                    Function
  398      CHLEB, etc. (11 unique)  >      AND
 1191      CHE8A, etc. (34 unique)  >       AO
  323      CHM0B, etc. (22 unique)  >      AOI
   27      CHPMB, etc. (2 unique)   >  COMPARE
   11      CHPUA, etc. (2 unique)   > CONSTANT
    6      CHTEB                    >   CUSTOM (COMPARE MACRO)
 1050      CHQQB, etc. (9 unique)   >    IDENT
 1102      IOPAD                    >    IOPAD
 1217      CHPGA, etc. (6 unique)   >     MUX
 1488      CHK1B, etc. (17 unique)  >    NAND
  229      CHLRB, etc. (15 unique)  >     NOR
 1394      CHZAA, etc. (10 unique)  >     NOT
   26      CHN3A, etc. (11 unique)  >      OA
  397      CHOFB, etc. (14 unique)  >     OAI
   58      CHL4C, etc. (10 unique)  >      OR
 1286      CHRGC, etc. (12 unique)  >     REG
  450      CHMJA, etc. (7 unique)   >    XNOR
  139      CHQRB, etc. (12 unique)  >     XOR
```

James W. Bishop *IBM AS/400 Division, Endicott, New York 13760 (BISHOJ at ENDVM5; bishop@vnet.ibm.com).* Mr. Bishop is a staff engineer working in AS/400 Advanced Series ASIC CMOS chip processor development. He joined IBM in 1984 after receiving a B.S. degree in electrical engineering from the University of Cincinnati. He subsequently earned an M.S. degree in computer engineering from Syracuse University in 1993. While at IBM he has been a logic designer on processor and memory subsystems for the AS/400 and System/390 divisions. Mr. Bishop is the author of three technical disclosures and three patents.

Michael J. Campion *IBM AS/400 Division, Endicott, New York 13760 (CAMPION at ENDVM5; campion@vnet.ibm.com).* Mr. Campion is a senior engineer working in AS/400 Advanced Series ASIC CMOS chip processor development. He joined IBM in 1970 after receiving a bachelor's degree in electrical engineering from Manhattan College. While at IBM he has worked in design and verification of System/370™ and System/390® processors. Mr. Campion is the author of three technical disclosures.

Thomas L. Jeremiah *IBM AS/400 Division, Endicott, New York 13760 (tjeremiah@vnet.ibm.com).* Mr. Jeremiah is a senior engineer working in AS/400 Advanced Series ASIC CMOS chip processor development. He joined IBM in 1970 after receiving a B.S. degree in electrical engineering from Drexel University. While at IBM he has worked in System/370 and System/390 processor development on various systems. Mr. Jeremiah is the author of ten technical disclosures and six patents.

Stephen J. Mercier *IBM AS/400 Division, Endicott, New York 13760 (MERCIER at ENDVM5, mercier@vnet.ibm.com).* Mr. Mercier is a staff engineer working in AS/400 Advanced Series ASIC CMOS chip processor development. He joined IBM in 1985 after receiving a B.S. degree in electrical and computer engineering from Clarkson University. Mr. Mercier also holds a B.A. degree in computer science from Potsdam College. While at IBM he has been involved in processor design activities for both the System/390 and the AS/400 Divisions. Currently, he is responsible for floorplanning, place and route of PowerPC/AS processor chips for the AS/400 Division. Mr. Mercier is the author of two technical disclosures and one patent.

Edmond J. Mohring *IBM AS/400 Division, Endicott, New York 13760 (MOHRINGE at ENDVM5; mohringe@vnet.ibm.com).* Mr. Mohring is a senior engineer working in AS/400 CMOS processor development. He joined IBM in 1965 after receiving a B.S. degree in applied physics from Adelphi University. He subsequently earned an M.S. degree in electrical engineering from the University of Vermont. He is also a graduate of the IBM LSI Institute. While at IBM he has worked in high-performance electronic packaging development and semiconductor technology development for computer applications. Mr. Mohring is the author of one technical disclosure. He is a member of the IEEE and has chaired several technical conferences.

Kerry P. Pfarr *IBM AS/400 Division, Rochester, Minnesota 55901 (PFARR at RCHVMX, pfarr@vnet.ibm.com).* Mr. Pfarr is an advisory engineer working in the circuit technology area of the AS/400 system technology group. He is responsible for the application of CMOS VLSI at both chip and card levels. His past assignments include CMOS circuit and memory card design. He joined IBM in 1979 after receiving a B.S.E.E. degree from North Dakota State University. Mr. Pfarr is credited with one U.S. patent and five invention disclosures.

Bruce G. Rudolph *IBM AS/400 Division, Rochester, Minnesota 55901 (RUDOLPH at RCHVMX, rudolph@vnet.ibm.com).* Mr. Rudolph is an advisory engineer working in the circuit technology area of the AS/400 system technology group. He is responsible for system clock generation and distribution for AS/400 processors, memory controllers, and the processor I/O subsystem. Past assignments include bipolar and CMOS circuit design, chip logic design, technology application support for the AS/400 processor development area, and the design of clock trees and clocking methodologies for use on ASIC and semicustom chip designs. He joined IBM in 1983 after receiving a B.S.E.E. degree from the University of Wisconsin. Mr. Rudolph is credited with three U.S. patents and four invention disclosures.

Gregory S. Still *IBM AS/400 Division, Endicott, New York 13760 (STILLGS at ENDVM5; stillgs@vnet.ibm.com).* Mr. Still is a technical project manager working in AS/400 Advanced Series ASIC CMOS chip processor development. He joined IBM in 1983 after receiving a B.S. degree in electrical engineering from Ohio State University. While at IBM, he has worked in design and management of System/390 processors, management of System/390 I/O microcode subsystems, and management of low-end AS/400 microprocessor development. Mr. Still is the author of two technical disclosures and three patents.

Tennis S. White *IBM AS/400 Division, Endicott, New York 13760 (twhite at ENDVM5).* Dr. White is an advisory engineer working in AS/400 Advanced Series ASIC CMOS chip processor development. He joined IBM in 1991 after receiving a Ph.D. degree in electrical engineering from Virginia Polytechnic Institute and State University. While at IBM he has worked on availability, system design, and timing. Dr. White is the author of three technical papers. He is a member of the Sigma Xi Scientific Research Society.