Brad Burgess, Nasr Ullah, Peter Van Overen, and Deene Ogden

The PowerPC 603 **Microprocessor**

In October 1993, Motorola and IBM unveiled the first low-power version of the PowerPC family—the PowerPC 603 microprocessor. Measuring a mere 85mm^2 (7.4 x 11.5 mm) in size, the 603 contains 1.6 million transistors and consumes less than 3 watts of power when operating at 80MHz. With estimated performance values of 75 SPECInt92 and 85 SPECfp92, the 603 is comparable in performance to present-day high-end personal computer and workstation processors. This member of the PowerPC family is designed to bring high-performance and low-power capabilities to the laptop and low-cost desktop computer markets.

Following closely on the heels of its predecessor, the PowerPC 601 microprocessor [1], the 603 microprocessor was developed at the joint Motorola/IBM/Apple Somerset Design Center in Austin, Texas. The 603 microarchitecture evolved from Apple, IBM, and Motorola's collective experience on several past designs. The similarity of the POWER and PowerPC architectures permitted the use of sample traces generated by RISC System/6000 machines for evaluation of design trade-offs. The compiler groups also provided their insight to ensure the traces from the past generation of processors and compilers, with their own specific peculiarities, did not misguide the 603's microarchitecture definition, and that trade-offs selected were appropriate for the next generation of compilers.

To accelerate the design and test process, engineers employed a formal VLSI design methodology derived from the best of both IBM and Motorola's CAD tools. These tools enable both the rapid design and dense packing capability necessary to produce very high-volume, high-yield microprocessors for the commercial market. The 603 design team employed a combination of custom circuitry (for arrays), library components (for data paths), and standard cell place and route (for random logic) to accomplish the 603 design.

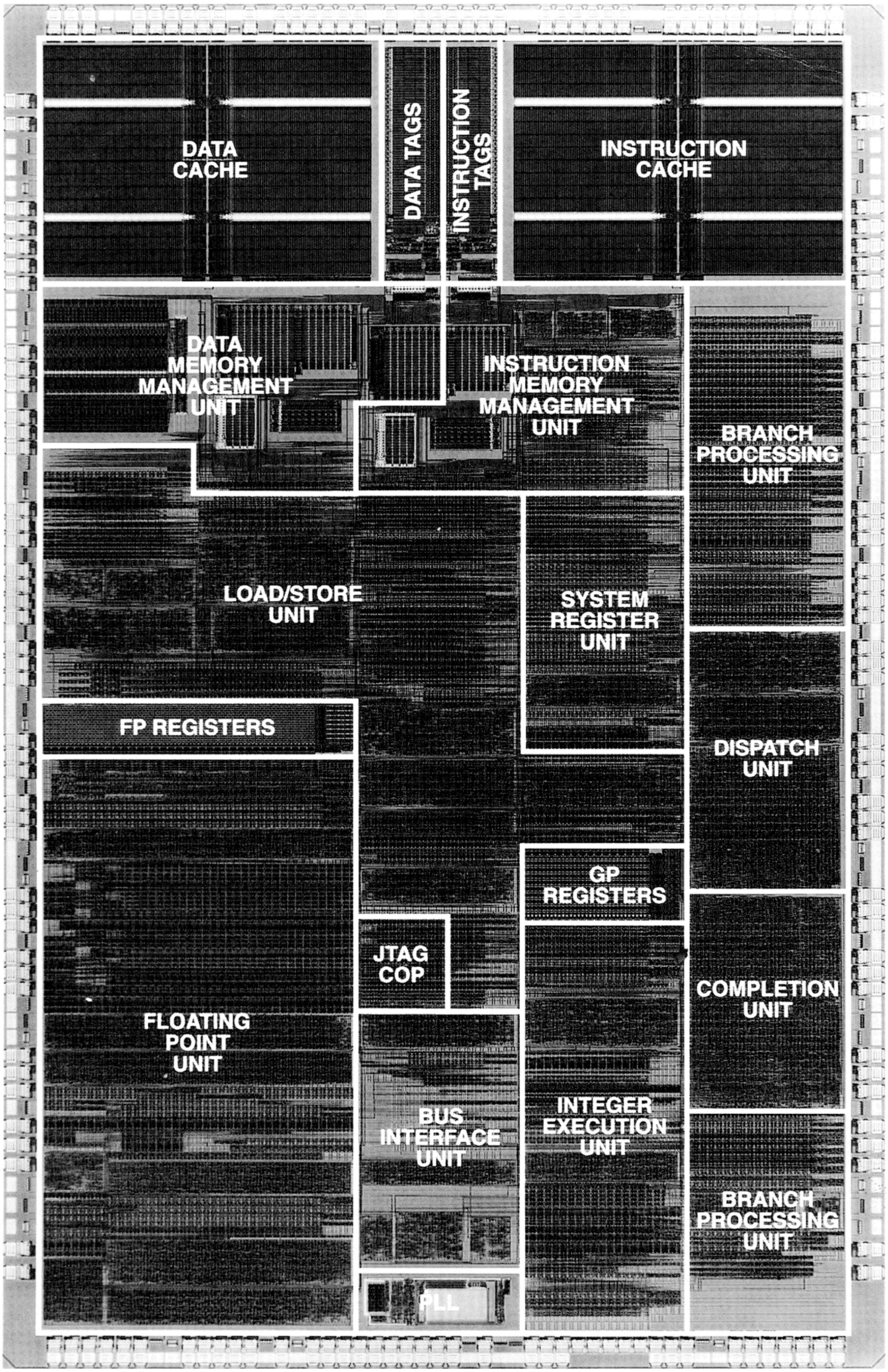
Using the best tools and methodology available, the design team took the 603 from concept to working silicon in 18 months. Ongoing design evaluation and debugging, including simulation of 28 billion processor cycles prior to tape-out, provided fully functional first-pass silicon that ran at the design target speed of 80MHz.

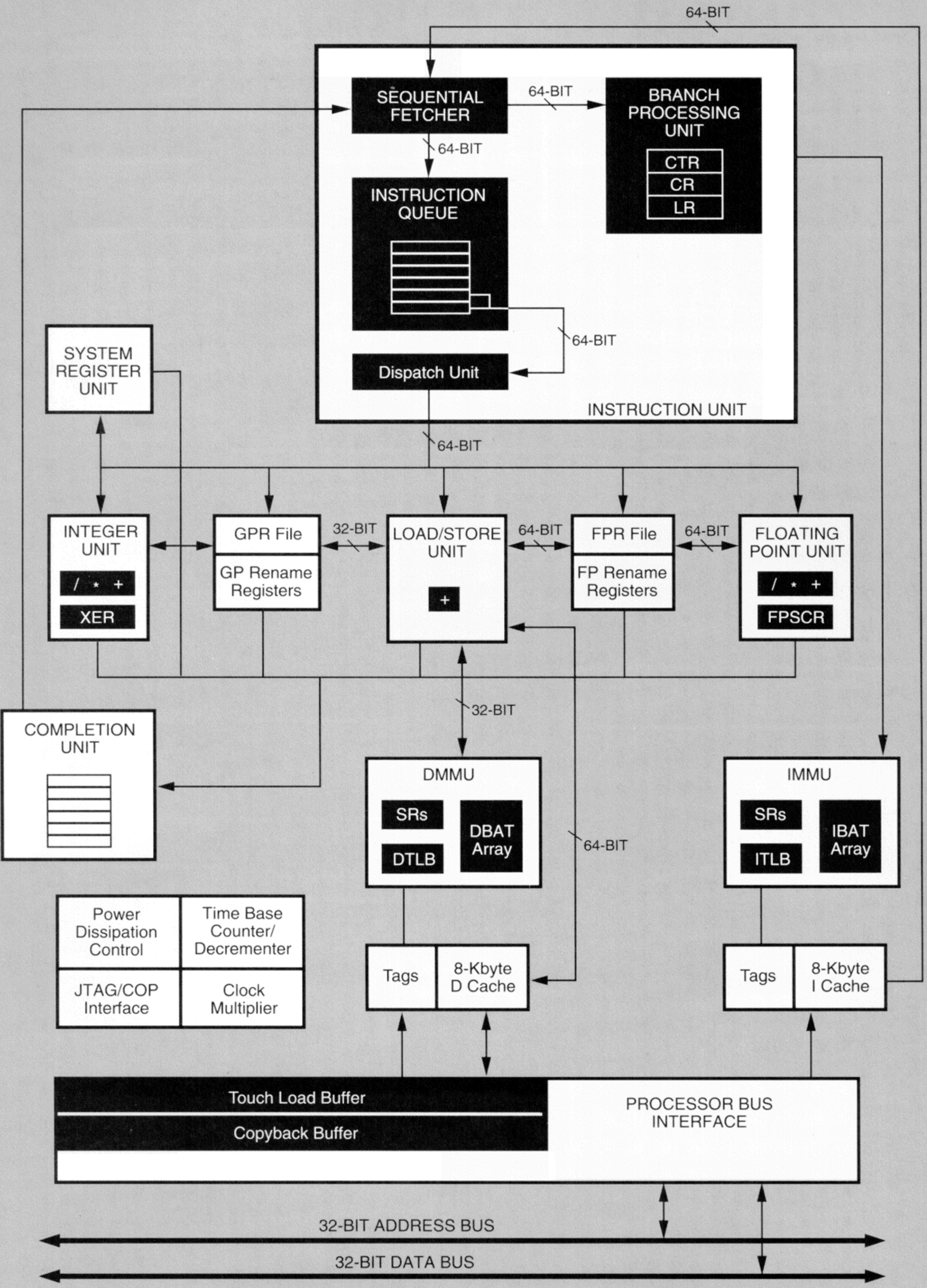
The PowerPC 603 microprocessor is manufactured by Motorola in Austin, Texas, and by IBM in Burlington, Vt. Motorola and IBM both fabricate the 603 using a 0.5μm, 4-level metal, 3.3VDC CMOS process with design rules compatible with both companies' semiconductor processes. The die is designed to be packaged in either a 240-pin ceramic quad flat pack or a ball-grid array package. Figure 1 is a photograph of the 603 die.

Functional Overview

The 603 is the first processor in the PowerPC family to fully support the PowerPC Architecture. It incorporates five execution units: branch, integer, floating-point, load/store, and system register; and a pair of on-chip 8KB instruction and data caches.

Figure 1. PowerPC 603 microprocessor die photograph


DATA CACHE
DATA TAGS
INSTRUCTION TAGS
INSTRUCTION CACHE
DATA MEMORY MANAGEMENT UNIT
INSTRUCTION MEMORY MANAGEMENT UNIT
BRANCH PROCESSING UNIT
LOAD/STORE UNIT
SYSTEM REGISTER UNIT
FP REGISTERS
DISPATCH UNIT
GP REGISTERS
JTAG COP
COMPLETION UNIT
FLOATING POINT UNIT
BUS INTERFACE UNIT
INTEGER EXECUTION UNIT
BRANCH PROCESSING UNIT
PLL


64-BIT
SEQUENTIAL FETCHER
64-BIT
BRANCH PROCESSING UNIT
CTR
CR
LR
64-BIT
INSTRUCTION QUEUE
64-BIT
Dispatch Unit
INSTRUCTION UNIT
SYSTEM REGISTER UNIT
64-BIT
INTEGER UNIT
XER
GPR File
GP Rename Registers
32-BIT
LOAD/STORE UNIT
64-BIT
FPR File
FP Rename Registers
64-BIT
FLOATING POINT UNIT
FPSCR
32-BIT
COMPLETION UNIT
DMMU
SRs
DTLB
DBAT Array
64-BIT
IMMU
SRs
ITLB
IBAT Array
Power Dissipation Control
Time Base Counter/ Decrementer
JTAG/COP Interface
Clock Multiplier
Tags
8-Kbyte D Cache
Tags
8-Kbyte I Cache
Touch Load Buffer
Copyback Buffer
PROCESSOR BUS INTERFACE
32-BIT ADDRESS BUS
32-BIT DATA BUS

Since the 603 is a super-scalar microprocessor, it is capable of issuing and retiring as many as three instructions per clock to these execution units. For increased performance, the 603 allows instructions to be executed out-of-order. Additionally, the 603 provides programmable power reduction modes that permit systems designers the flexibility of implementing a variety of power management techniques. A block diagram of the 603 is shown in Figure 2.

Instructions are dispatched in-order to one of the five execution units. If there are no operand dependencies, execution occurs immediately. The integer unit executes most instructions in one cycle. The floating-point unit is pipelined and executes both single and double precision floating-point operations. Branch resolution is handled by the branch unit. If the branch conditions are available, branches are immediately resolved; otherwise, instruction execution continues speculatively. Instructions that modify the processor control registers are executed by the system register unit. Finally, data movement between the data cache and the general-purpose and floating-point registers is handled by the load/store unit.

In case of cache misses, the caches access main memory through a 64-bit high-performance bus similar to that of the MC88110 [8]. To maximize throughput and thus increase overall performance, the cache communicates with memory mostly via burst operations that allow a cache line to be filled in one transaction.

After an instruction finishes execution in an execution unit, its results are forwarded to a completion buffer, and then subsequently written to the appropriate register file set when the instruction is retired from the completion buffer. To avoid register contention, the 603 provides separate 32-entry integer general-purpose registers (GPRs) and floating-point register (FPRs) sets for the storage of operands.

The following sections discuss in more detail the factors that contribute to the efficient flow of instructions and data through the 603.

Figure 2. Block diagram of the PowerPC 603 microprocessor

Instruction Pipeline

Figure 3 shows the 603's instruction pipelines for several types of instructions.

Fetch stage. During this stage, the instruction fetcher retrieves two instructions at a time from the instruction cache (regardless of alignment), unless the address points to the last word of a cache line, in which case only a single word is returned.

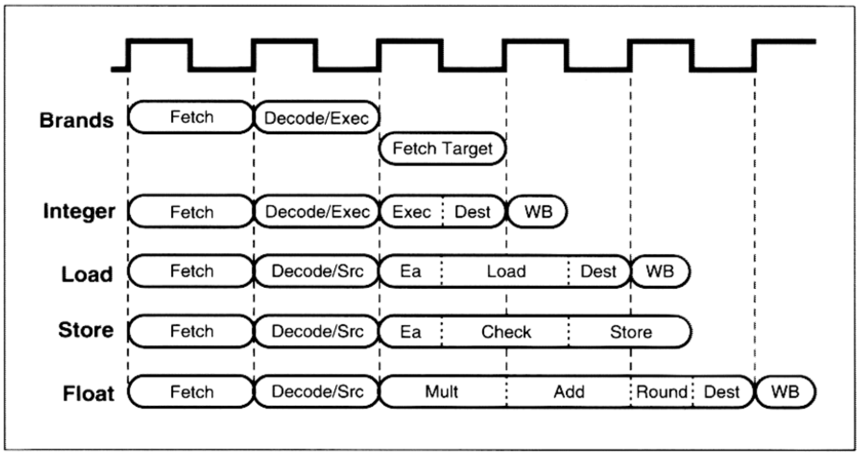


Figure 3. PowerPC 603 microprocessor master instruction pipeline

Decode/Source stage. During this stage, the dispatcher and branch unit decode instructions, allocate rename registers, read available source operands, and dispatch instructions to their respective execution units (or reservation stations).

Execute stage. During this stage, the execution units execute instructions and write results back to the destination rename registers. If the data is needed as a source operand for another instruction, the data is forwarded immediately to the requesting unit. When the execution unit finishes with an instruction it notifies the completion buffer that the instruction is finished and tags the instruction if any exceptions occurred.

Completion stage. During this final stage, the completion buffer logic writes the contents of any renamed registers into the architectural registers. It then deallocates rename registers and returns them to the pool for future use. If the completion logic detects that an instruction is tagged as having caused an exception, it flushes the pipeline and initiates exception processing. Otherwise, it retires completed instructions and removes them from the completion buffer. Because the completion logic retires all instructions in program order, all exceptions are fully precise.

Functional Units

Dispatch unit. The instruction flow diagram is shown in Figure 4. Instructions are fetched two per cycle from the instruction cache, and placed in a six-entry instruction queue. On arrival at the instruction queue, branch instructions are immediately forwarded to the branch processing unit for resolution. All other instructions are issued from the instruction queue at the rate of two per cycle if there is no resource contention (for example, a busy execution unit, or a full completion queue).

The instruction dispatcher decodes the bottom two entries of the queue and dispatches up to two instructions per cycle, in program order, to either the integer unit, floating-point unit, load/store unit, or system unit. If the instruction dispatcher finds an execution unit busy, it does not dispatch the instruction and stalls. There are several mechanisms in the 603 to avoid dispatch stalls.

To avoid dispatch stalls due to operand dependencies, the 603 has single-entry zero-latency reservation stations associated with each execution unit. The reservation station holds the instruction until all operands are available. This allows the dispatcher to dispatch subsequent instructions to other execution units without stalling the instruction queue.

In addition to dispatching instructions, the dispatcher allocates rename buffers and coordinates pipeline stalls. Rename buffers provide temporary storage for the results of an instruction's execution. Register renaming helps avoid stalls on register write-after-write and write-after-read hazards. In addition, the rename buffers simplify exception recovery by allowing the 603 to invalidate results of speculative instruction execution without affecting the contents of the general registers, floating-point registers, and processor control registers.

When the instruction queue is relatively full, the branch unit decodes instructions upstream from those being decoded by the dispatcher. For taken branches, the instruction queue usually contains enough instructions to keep the dispatcher busy until the new instruction stream is fetched. Thus, from the dispatcher's perspective this allows basic blocks to be connected without ever seeing the branch instruction or the fetch penalty. Once dispatched, the dispatcher transfers control of the instruction's execution to the execution unit, and control of the overall instruction stream to the completion buffer logic.

Execution units. The *branch unit* decodes the fetched instructions and executes most branch instructions in a single clock. It also retires and eliminates from the instruction queue (branch folding) branches that do not modify branch-related resources such as the Link or Count registers. The Link register contains the return address from a branch instruction. The Count register is a loop counter that is used by some branch instructions.

The branch unit has its own facilities for calculating the branch target address. Conditional branches in the PowerPC Architecture depend on a counter-register (CTR) value and/or any one of 32 condition register (CR) bits.

If the CTR value is unavailable, the branch unit and instruction fetching is stalled (not likely to occur very often). If a CR bit is unavailable, the branch unit predicts either the taken or fallthrough path based on bits in the branch opcode. These predicted instructions are tagged as speculative and proceed down the pipeline normally. The unavailable CR bit that initiated speculative execution is checked each cycle. When the CR bit becomes available and the branch resolves, the branch unit flushes all speculative tagged instructions from the pipeline if the branch was mispredicted, or simply clears all speculative tags if the branch was correctly predicted. The branch unit can only resolve a single CR bit at a time, thus it can only speculate down one conditional branch path at a time.

The *integer unit* processes integer arithmetic, logical, and bit-field instructions. All integer instructions are single cycle with the exception of multiply and divide, which requires 2 to 6 (data-dependent) and 37 processor clock cycles respectively.

The *load/store unit* handles load and store instructions to and from both the integer and floating-point registers. It contains a dedicated adder for the calculation of effective addresses, and the logic required for data alignment to and from the cache.

The load/store unit is fully pipelined so that loads can be dispatched at the rate of one per clock cycle. Loads have a two-clock-cycle latency, a half cycle to compute the effective address, one cycle to access the data cache and MMU, and another half cycle to write the result into the rename register.

Since the load/store unit cannot write to the cache until after checking for memory protection violations, it does not execute store instructions in a fully pipelined manner. During the execution stage, the load/store unit calculates the effective address and translates the address to check for memory protection violations. On the next clock cycle, the store advances to a holding buffer, where it waits for the completion logic to retire the instruction and enable writing data to the cache.

The 603 takes advantage of the PowerPC Architecture's weakly ordered memory model and allows load instructions to bypass pending stores in order to minimize stalls due to load data hazards. (Of course, loads which may potentially access the same location as the pending store are not allowed to bypass in order to preserve correct program operation.)

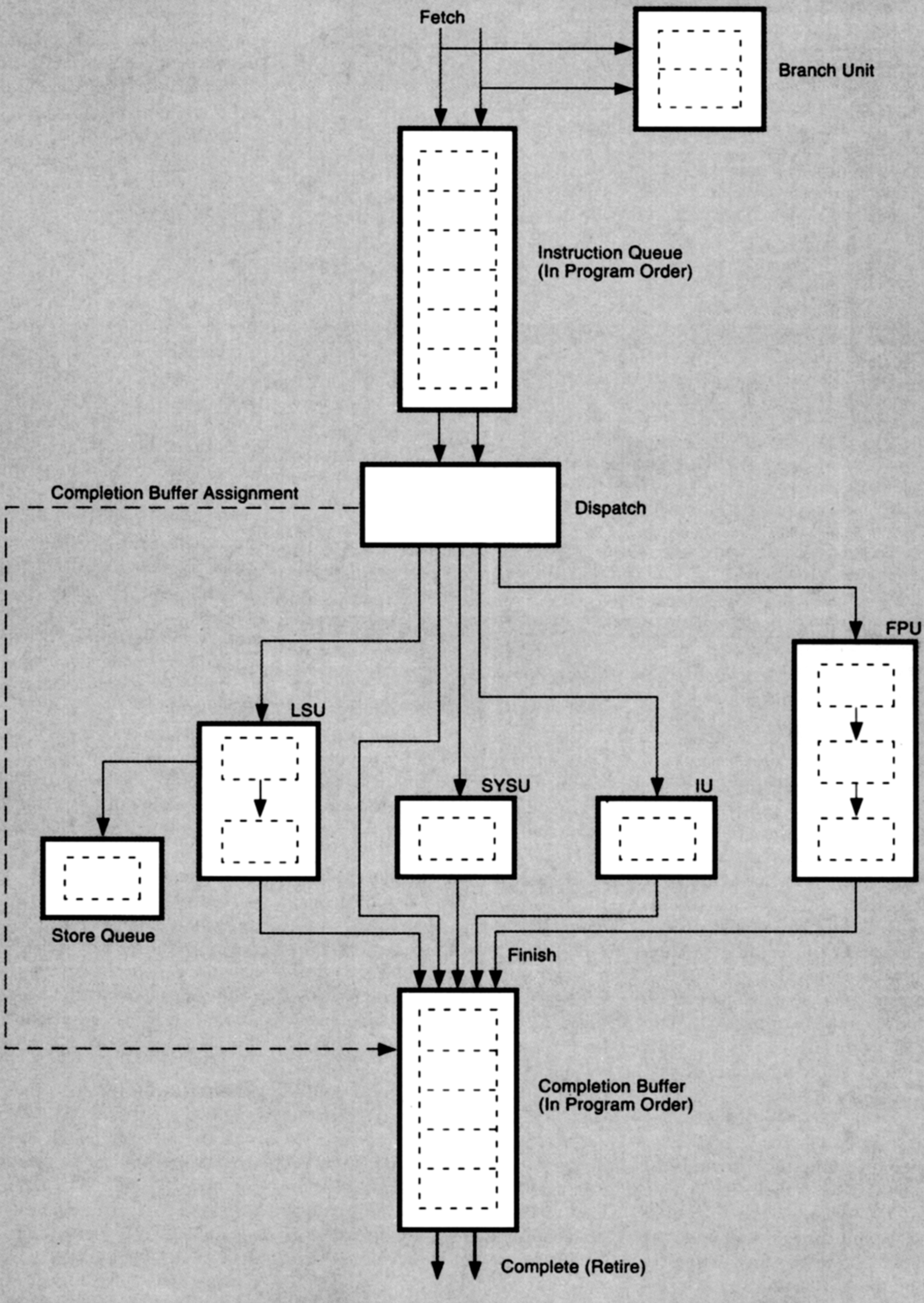
The *floating-point unit* is fully pipelined with single-cycle throughput and three-clock-cycle latency for all single-precision instructions (except divide) and double-precision adds, subtracts, and compares. The first stage of the pipeline performs operand alignment and multiplication, the second stage performs addition, and the third stage rounds and normalizes the result. For double-precision multiplies and multiply-adds, the first pipeline stage is iterated twice, providing two-clock-cycle throughput and four-clock-cycle latency. Divides are nonpipelined and stall the floating-point unit for 18 clock cycles for single-precision operations and 33 clock cycles for double-precision operations.

The floating-point unit supports denormalized numbers in hardware. Since denormalized results require more time to process, a special non-IEEE mode provides a means to coerce denormalized results to zero. This eliminates data-dependent instruction execution time that would disrupt real-time data processing.

The primary function of the *system register unit* is to process the condition-register-logical instructions and the move operations to or from the SPRs. These instructions execute in one to three cycles, and are serializing in nature (that is, all preceding instructions must have been completed and retired). As a result they have somewhat less performance than might otherwise be expected. However, given the relatively infrequent use of instructions executed by the system register unit, the trade-off was made in favor of reduced complexity and silicon costs.

Completion buffer. The completion

Figure 4. PowerPC 603 microprocessor instruction flow


Fetch
Branch Unit
Instruction Queue
(In Program Order)
Completion Buffer Assignment
Dispatch
FPU
LSU
SYSU
IU
Store Queue
Finish
Completion Buffer
(In Program Order)
Complete (Retire)

603 Features

Superscalar Instruction Processor

- 2 instructions fetched per clock
- 5 independent execution units
  —branch, integer, floating-point, load/store, and system
- 3 instructions issued per clock (2 dispatched plus 1 branch)
- 2 instructions retired per clock plus branch folding
- 32, 32-bit general registers and 32, 64-bit floating-point registers
- 3-stage pipelined floating-point unit
  —floating-point denorm support in hardware
- 2-stage pipelined load-store unit
  —loads bypass stores
  —single entry store buffer
- Static branch prediction and speculative execution

Memory System:

- 8K, 2-way set associative, 32-bytes/line, instruction cache
- 8K, 2-way set associative, 32-bytes/line, copyback data cache
- Support for big-endian and little-endian addressing
- 64-entry, 2-way set associative, instruction TLB
- 64-entry, 2-way set associative, data TLB
  —hardware-assisted software table walk for TLB reloads
- 4 instruction block address translation registers (IBAT)
- 4 data block address translation registers (DBAT)
- 16 segment registers
- 64-bit, pipelined, split transaction, burstable external bus with parity
- 32-bit data bus option
- Internal clock multiplier 1x, 2x, 3x, 4x

System Features:

- Power Down modes (DOZE, NAP, SLEEP)
- JTAG Interface
- On-chip hardware debug support

buffer tracks instruction execution, retires finished instructions, and controls writing of the contents of renamed registers to the architectural registers. The completion buffer tracks up to five instructions at a time and can retire up to two instructions each clock cycle. Because instructions may execute out of order, the completion buffer provides an ordering mechanism that makes instruction completion appear sequential, and provides a mechanism for precise exception handling for PowerPC 603 microprocessor systems.

Memory Subsystem

The memory subsystem provides the instructions for the instruction fetcher and data for the load/store unit. Efficient access between the caches and memory systems is provided by the MMU, and the external bus interface.

Caches and Memory Management

The 603 incorporates two 8KB, 2-way set associative, 32-byte per line on-chip caches, one for instructions and one for data. On a cache hit, the instruction cache can provide 2 instructions per cycle to the instruction queue, and the data cache can provide up to a double-word of data to the load/store unit per cycle.

The data cache supports copy-back or write-through policies. Both caches use a least recently used (LRU) replacement policy. On a cache miss, the 603's cache blocks are filled in four beats of 64 bits each. The burst fill is performed as a 'critical-double word-first' operation; the critical double word is simultaneously written to the cache and forwarded to the requesting unit, thus minimizing stalls due to cache fill latency. In the case of the data cache, the burst operation is also used to write-back a modified cache line to memory.

Since the 603 was not specifically targeted for multiprocessing applications, the 603 design restricted certain aspects of data cache coherency in order to save silicon area. Although the data cache only supports a three-state MEI (modified, exclusive, invalid) cache coherency protocol, it is compatible with full MESI (modified, exclusive, shared, invalid) caches on the same bus. Additionally, the data cache implements only a single set of cache tags which must be arbitrated for between snooping operations and load/store activity, with priority given to snoop operations.

The load/store unit provides the data transfer interface between the data cache and the GPRs and the FPRs. In addition the load/store unit provides all logic required to calculate effective addresses, handles data alignment to and from the data cache, and provides sequencing for load-and-store string and multiple operations. The caches provide a 64-bit interface to the instruction fetcher and load/store unit.

For faster translation of addresses, the 603 provides two four-entry, fully associative block address translation registers, and two 64-entry, 2-way set associative translation look-aside buffers (TLBs) for instructions and data. The 603 uses software to perform TLB replacements. A hash function is used for the replacement policy. When a TLB miss occurs, the processor takes a special exception to the software tablewalk handler. This handler walks through the page tables to locate and reload the necessary page table entry into the TLB. Additionally, the 603 has dedicated scratch pad registers that can be used to shadow general-purpose registers during software tablewalks. Through hardware assist, the entire tablewalk routine fits in two cache lines of memory and provides a low-cost, flexible, and fairly fast TLB reload capability.

External Bus Interface

The 603's external bus is compatible with the 601 [1], which was derived from Motorola's 88110 multiprocessor bus [8]. The 603's bus interface unit (BIU) receives requests for bus operations from the instruction and data caches, and executes the operations per the 603 bus protocol. Memory accesses can occur in single-beat (1 to 8 bytes) and four-beat burst (32 bytes) data transfers when the bus is configured as 64 bits, and in single-

With the flexibility built into the PowerPC Architecture, we expect to deliver a wide range of microprocessors all the way from low-cost embedded controllers through massively parallel supercomputers.

beat (1 to 4 bytes), two-beat (8 bytes), and eight-beat (32 bytes) data transfers when the bus is configured as 32 bits.

The BIU provides address queues, and prioritization and bus control logic. It consists of an independently arbitrated 32-bit address and 64-bit data buses (the data bus can optionally be configured as 32 bits). The 603's bus transaction consists of separate address and data tenures. This allows a variety of bus arbitration schemes to be supported. Specifically, the 603 supports address pipelining, where the address tenure of a new transaction is allowed to begin before the data tenure of the current transaction has completed. The 603's bus interface also supports split transactions, where the address and data tenures can be arbitrated for and controlled by different masters; and enveloped transactions, where the address and data tenures of a new transaction can occur after the address tenure of a previous transaction has ended, but before the data tenure for the previous transaction has begun. Enveloped transactions can be used for deadlock prevention in hierarchical bus environments, and to speed snoop push operations.

The 603 bus protocol also supports an address retry capability to support an efficient snooping protocol for memory coherency across the system. In a multiple processor system, the address retry can be used by a snooping master to interrupt another master's transaction on the bus. This becomes necessary when a bus master begins a transaction to access data that has been locally modified in the 603's cache. The 603 (referred to as the snooping 603) detects the access to this modified memory region, and uses the retry capability to force the first master to abort its transaction and retry it later. This enables the snooping 603 to write back the modified data to memory for use later by the bus master that has been retried. The 603 bus interface also contains a clock multiplier that enables the processor to run at twice, three times, or four times the external bus clock speed.

Debug Features

The 603 incorporates a JTAG/IEEE 1149.1 boundary scan interface to facilitate board-level testing. The 603 also incorporates a special interface (accessible through the JTAG port) that allows an external service processor to read or write memory or any of the 603's internal registers. A special mode allows pipeline status information to be displayed for tracking the instruction stream in real time. Additionally, a programmable instruction address breakpoint is provided to assist in software debugging.

Conclusion

The combined efforts of Apple, IBM, and Motorola have been focused on creating PowerPC, a new RISC architecture that will form the basis of a whole new generation of high-performance, low-cost computers. With the flexibility built into the PowerPC Architecture, we expect to deliver a wide range of microprocessors all the way from low-cost embedded controllers through massively parallel supercomputers. The 603 is the first in a series of PowerPC microprocessors targeted at high-volume, low-cost, portable and desktop personal computers. It provides performance heretofore available only in significantly higher-priced, higher-cost, and higher-powered processors. C

References

1. Allen, M. and Becker, M. Multiprocessing aspects of the PowerPC 601 Microprocessor. In *Proceedings of COMPCON* (Feb. 1993).
2. Becker, M.C., Allen, M.S., Moore, C.R., Muhich, J.S., and Tuttle, D.P., The PowerPC 601 Microprocessor. *IEEE Micro. 13*, 5 (Oct. 1993), 54–68.
3. Chang, A., Mergen, M.F., Rader, R.K., Roberts, J.A., and Porter, S.L. Evolution of storage facilities in AIX Version 3 for RISC System/6000 processors. *IBM J. Res. Develop. 34* (Jan. 1990).
4. Cohen, D. On holy wars and a plea for peace. *IEEE Comput., 14*, 10 (Oct. 1981).
5. Dubois, M.C. and Scheurich, Briggs, F. Synchronization, coherence, and event ordering in microprocessors. *IEEE Comput., 21*, 2 (Feb. 1988), 9–21.
6. ANSI/IEEE-1985 Standard for binary floating-point arithmetic, IEEE, Piscataway, N.J., 1985.
7. Groves, R.D. and Oehler, R.R. RISC System/6000 processor architecture. *Microprocessor & MicroSystems 14*, 6 (July/Aug. 1990).
8. Gullette, B. The Design of the 88110 Bus Interface. In *Proceedings of RISC '92* (Feb. 1992).
9. Lamport, L. How to make a multiprocessor computer that correctly executes multiprocess programs. *IEEE Trans. Comput., C-28*, 9 (Sept. 1979), 241–248.
10. Oehler, R.R., and Groves, R.D. IBM RISC System/6000 processor architecture. *IBM J. Res. Develop. 34*, 1 (Jan. 1990), 23–36.
11. Patterson, D.A., and Ditzel. The case for the reduced instruction set computer. *Computer Architecture News* (Oct. 15, 1980).
12. Patterson, D.A. and Sequin C.H. RISC-1: A Reduced Instruction Set VLSI Computer. In *Proceedings of the Eighth Annual International Symposium on Computer Architecture* (May 1981), pp. 443–457.
13. *The PowerPC Architecture*. Morgan-Kaufmann, San Mateo, Calif., Dec. 1993.
14. Radin, G. The 801 Minicomputer. In *Proceedings of the Symposium on Architectural Support for Programming Languages* (March 1982), pp. 39–47.
15. Simpson, R.O. and Hester, P.D. The IBM RT PC ROMP processor and

memory management architecture. *IBM Syst. J. 26* (1987), 346–360.

16. Stone, J.M. and Fitzgerald, R.P. An overview of storage in PowerPC. IBM Res. Rep. RC 19133, Aug. 1993.

About the Authors:
BRAD BURGESS is a member of the technical staff at Motorola involved in both instruction set architecture and microarchitecture implementation. Current research interests include microarchitecture implementation and optimization, performance analysis, and compiler optimization. email: bradb@ibmoto.com

NASR ULLAH is a RISC applications designer in Motorola's RISC microprocessor division who has been working on PowerPC microprocessor board designs. Current research interests include parallel processing, computer design, computer architecture, performance measurement, evaluation and prediction. email: nasr_ullah@riscgate.sps.mot.com

PETER VAN OVEREN is a RISC applications designer at Motorola who has been developing user manuals and associated documentation for the PowerPC microprocessor family. Current research interests include electronic information distribution, computer architecture, performance evaluation, and systems design. email: pvo@daffy.sps.mot.com

Authors' Present Address: Motorola, 6501 William Cannon Drive West, Austin, TX 78735-8598.

DEENE OGDEN is a senior engineer at IBM's Somerset Design Center who was the 603 control logic team leader. Current research interests include microprocessor and computer design. **Author's Present Address:** International Business Machines Corporation, 11400 Burnet Road, Austin, TX 78758; email: deene@austin.ibm.com