Circuit Techniques in a 266-MHz MMX-Enabled Processor

Don Draper, *Member, IEEE*, Matt Crowley, *Member, IEEE,* John Holst*, Member, IEEE*, Greg Favor, Albrecht Schoy, Jeff Trull, Amos Ben-Meir, Rajesh Khanna, Dennis Wendell, *Member, IEEE*, Ravi Krishna, Joe Nolan, *Member, IEEE*, Dhiraj Mallick, Hamid Partovi, *Member, IEEE*, Mark Roberts, *Member, IEEE*, Mark Johnson, *Member, IEEE*, and Thomas Lee, *Member, IEEE*

***Abstract**: The AMD-K6 MMX-enabled processor is plug-compatible with the industry-standard Socket 7 and is binary compatible with the existing base of legacy X86 software. The microarchitecture is based on an out-of-order, superscalar execution engine using speculative execution. High performance and compact die size are achieved by using self-resetting, self-timed and pulsed-latch circuit design techniques in custom blocks and placed-and-routed blocks of standard cells. The 162 sq. mm die is fabricated on a 0.35-µm, five-layer metal process with local interconnect. It is assembled into a ceramic pin grid array (PGA) wing C4 flip-chip mounting. The processor functions at clock speeds up to 266 MHz.*

***Index Terms**—Cache memories, computer architecture, design automation software, flip-flops, integrated circuit design, logic design, phase-locked loops, programmable logic array, read only memories.*

I. Introduction

THE AMD-K6[1] MMX[2]-Enabled processor [1] achieves high performance while maintaining plug-compatibility with the hardware motherboard industry-standard Socket 7 and binary software-compatibility with the large, established base of legacy X86 applications. We will describe the circuit techniques used to achieve high performance in combination with innovative microarchitectural features. This processor carefully balanced the complexity of innovative circuit techniques with the regularity and modularity of a top-level standard-cell methodology. The circuit implementation utilized full-custom macroblocks designed with self-timed circuits and self-resetting techniques [2]-[4] for density and speed, while using a standard cell-based approach in the top-level modules to achieve the flexibility and fast implementation associated with place-and-route techniques.

Section II will describe the microarchitectural features showing how the processor achieves X86 compatibility with RISC performance. Emerging multimedia and communication software needs for increased integer data parallelism are addressed by the MMX unit, a new feature in this chip.

The circuit methods used for performance and testability in the caches are described in Section III along with the tags in Section IV. The design of the programmable logic array (PLA) in Section V and the Op-code ROM in Section VI is driven by coupling considerations. A fast flip-flop is achieved by the implementation of the edge-triggered latch described in Section VII. The clock distribution methodology described in Section VIII shows how low skew was achieved. To implement clock multiplication with low jitter, a high-performance phase-locked loop (PLL) as described in Section IX is needed. Section X describes the top level module implementation of standard-cell-based blocks. The five-layer metal, 0.35-µm CMOS process on which

Manuscript received April 28. 1997; revised August 7, 1997.

D. Draper, M. Crowley, l. Holst, G. Favor. A. Schoy,1. Trull, A. Ben-Meir. R. Khanna. D. Wendell, R. Krishna, J. Nolan, D. Mallick. H. Panovi. and M. Roberts are with Advanced Micro Devices. Inc.. Sunnyvale, CA 94088-3453 USA.

M. Johnson was with Advanced Micro Devices, Inc., Sunnyvale,CA 94088-3453 USA. He is currently with Transmeta Corp., Santa Clara. CA 95054 USA.

T. Lee was with Advanced Micro Devices. Inc., Sunnyvale, CA 94088-3453 USA. He is currently with Stanford University Center for Integrated Systems. Stanford. CA 94305 USA.

Publisher Item Identifier S 0018-9200(97)08030-X.

[1] AMD-K6 is a trademark of Advanced Micro Devices Inc.

[2] MMX is a trademark of Intel Corp.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

the processor is fabricated is described in Section XI, and the product performance is detailed in Section XII, followed by the conclusion in Section XIII.

II. Microarchitectural Overview

The AMD-K6 processor features a decode-decoupled superscalar microarchitecture designed to execute existing X86 instruction set architecture applications while achieving RISC performance [5]. The block diagram shown in Fig. 1 illustrates the 32-kbyte Level 1 instruction cache as well as the X86 decoder which issues RISC-like operations, called RISC86 Ops, to the seven parallel execution units. A new feature of the processor is the multimedia extension unit which uses new instructions to achieve higher performance parallel execution of integer arithmetic for multimedia functions.

Instruction predecoding to identify instruction boundaries begins during filling of the 32-kbyte two-way set-associative L1 instruction cache, after which the predecode bits are stored in the associated 20-kbyte predecode cache. Fetched X86 instructions are executed by being decoded on-the-fly by the instruction decoders into equivalent RISC86 Ops, which are then executed by what is essentially a RISC superscalar processing engine. During each cycle, up to 16 instruction bytes are fetched from the instruction cache to keep the instruction buffer full. From this buffer, up to two X86 instructions are decoded, any branches immediately recognized and predicted, and up to four RISC86 Ops generated.

The uncommon instructions are mapped into ROM-resident RISC86 sequences. The instruction scheduler buffers up to 24 RISC86 operations, using implicit register renaming onto a total of 48 physical registers. Each cycle, up to six RISC86 instructions are issued out-of-order to seven parallel execution units and speculatively executed (because they may not commit), depending on whether the branch prediction was successful or not. Up to four RISC86 Ops per cycle can be retired in order. The branch algorithm uses a two-level branch prediction algorithm utilizing an 8192-entry branch history table, a 16-entry branch target cache, and a 16-entry return address stack.

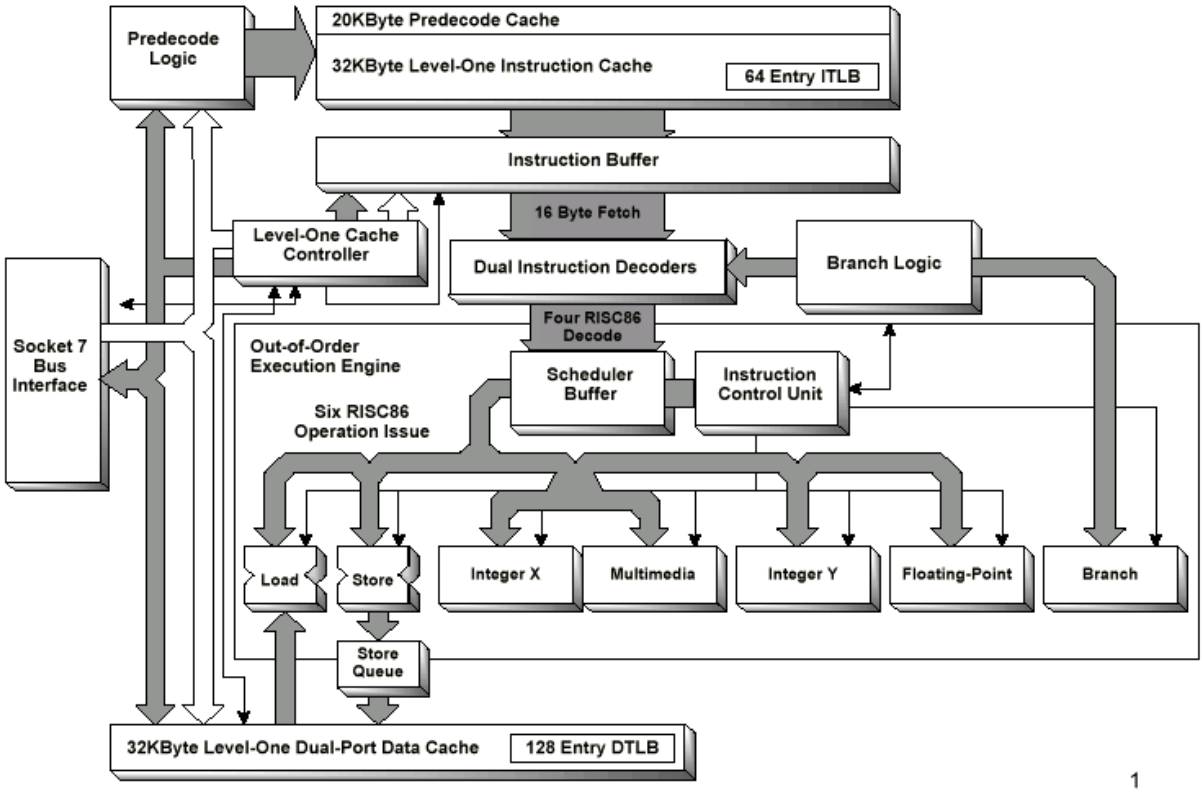

Fig. 1. Microarchitectural block diagram.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

Both instruction and data caches are 32 kbyte in size, two-way set-associative, and use a 64-byte line size with 32-byte subblocking. Cache fills are done on a subblock basis. Both caches are virtually indexed (bits 13 : 6, where bits 13 : 12 are translated address bits) and physically tagged. Synonyms and aliasing are handled in hardware. The I- and D-caches maintain mutual exclusion with respect to each other, which eases the handling of self-modifying code. Hit-under-miss (nonblocking) is supported in the l-cache, that is, a subsequent request can be serviced if it is a hit even if there is an older prior miss. There are 256 sets in each cache. Each set contains two ways (or lines) and each line contains two subblocks, for a total of $256 \times 2 \times 64$ bytes = 32 kbytes.

The instruction cache uses a most recently used (MRU) method to predict the way selection on cache accesses. A misprediction in the way selection causes a one-cycle penalty. The line replacement algorithm is least recently used (LRU). A simple prefetching algorithm is used: when a line miss (versus subblock miss) occurs, and the miss is on subblock 0 (bit [5] of address = 0) of the line, then both subblocks are fetched (as well as pipelined on the bus).

In contrast to the instruction cache which uses a prediction method for way selection, the data cache uses a way selection scheme based on an accurate tag comparison. The line replacement algorithm is least recently missed (LRM) or first-in, first-out (FIFO). In this approach, the line that came into the cache first is the next one to be replaced. Write allocation occurs assuming a memory access is cacheable. A number of cacheability control registers tell the processor whether the access is cacheable.

The data cache supports one read and one write operation per clock cycle. These operations can be to independent or dependent addresses. Stalls occur only on cache misses or on a data dependency that cannot be handled in hardware. (One notable case which stalls is a read with a superset dependency on an older write that has not yet made it into the data cache, where "superset dependency" means that the read is requesting more bytes than a write can provide.) Address dependencies are determined by address bits [9 : 0] and the number of bytes requested.

Because bits [13 : 12] of the index are virtual, each line can reside in any of a group of four sets where a group is defined by the combinations of bits [13 : 12]. The data and instruction tag RAM's are specially designed to support reading a group of four sets in one cycle. Effectively, the tag RAM's are indexed by bits [11 : 6] of the address and a total of eight tags are read out on every tag RAM access. All eight tags are compared against the corresponding physical address. This allows detection of virtual index aliasing during tag lookup. When an alias is detected, it is invalidated or written back if dirty. Then, the line or subblock is brought back into the cache and entered into the new set corresponding to the new linear index. In other words, the hardware disallows creation of aliases by making sure that a line can reside in only one of the four possible sets. In the instruction cache, aliases are handled during fills without any penalty. All that is required is invalidation of the appropriate line. In the data cache, aliases (for both reads and writes) are handled through a special state sequence that guarantees no disturbance from internal or external snoops while a line is being evicted (if necessary) into the write back buffer. Once the alias is eliminated, the line is fetched and deposited in its new set corresponding to the new linear index.

Each cycle, up to six Ops can be issued and executed—one memory read, one memory write, two integer or one multimedia register operation, one floating point operation, and one branch condition evaluation. These Ops are executed out of order, subject to dependencies and resource constraints. After completion, Ops are committed in order by the scheduler (which also functions as a reorder buffer), to ensure precise exceptions and full compatibility with the X86 architecture. Results from functional units are deposited in the scheduler reorder buffer in temporary registers. The process of finding the register containing the data requested (as an operand) is called implicit register renaming and is based on position within the queue rather than explicit tag assignments. In this why the scheduler module saves on renaming hardware. The FIFO physical structure and implicit register renaming of the scheduler module permit the use of fast position-based instruction issue and dependency tracking logic. The process of searching for the source instruction for a given operand is similar to propagating a carry through an adder and leads to hardware with characteristics similar to many fast adders.

The scheduler module contains logic to track RISC86 Ops throughout their history, determine dependencies, schedule execution, and commit architectural state. It is structured as a FIFO queue; Ops enter the top four at a time (matching the X86 instruction decoder bandwidth) and are retired up to four at a time from the bottom. Ops enter the scheduler after being decoded or fetched and remain there until the end of their life. The queue is general—any scheduler queue entry can be used for any category of Op, so there is no need for separate queues for Ops destined for different execution units. The determination of dependencies between Ops and the generation of operand forwarding controls takes advantage of the physical ordering of Ops within the scheduler (as well as the fact that all Ops reside in this common structure). An Op selected

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

for issue to an execution unit "generates" a "carry" into operand scan chains corresponding to each of its register operands. This carry propagates through progressively older queue entries until encountering an Op which "kills" the carry based on the fact that it modifies the desired register bytes. From the perspective of each queue entry, and for each operand scan chain, an Op is selected to supply a given operand if it receives a carry-in from the corresponding scan chain and its result register bytes matches the desired operand bytes. For maximum speed, a structure similar to the carry tree of a binary adder was used. Group generate Ggrp[7 : 0] and group propagate Pgrp[7 : 0] terms are created for 3-Op groups (convenient because there are 24 Ops in the scheduler). These eight group terms are combined in a three-level tree to create group carry in terms Cgrp[7 : 0] which are then used within each group to select the operand source Op.

In order to target the growing number of applications in the areas of communications and multimedia, the processor incorporates the Multi-Media Extensions (MMX) to the X86 instruction set. The MMX unit employs a single instruction, multiple data (SIMD) technique to process multiple operands of 8, 16, or 32 b in a 64-b data path to perform highly parallel and compute intensive algorithms involved in multimedia applications. The MMX unit supports 57 new instructions [6], [7] which allow additions, subtractions, multiplies, multiply-accumulates, shifts (logical or arithmetic), packing and unpacking functions, logical operations, moves, and several other operations, most of which can be executed on all three sizes of integers. Eight new 64-b registers are provided as part of the SIMD pipeline.

III. Cache Implementation

The high processor frequency requires large, on-chip caches to reduce the miss rate and supply the processor with instructions and data without stalling. To achieve single-cycle RISC operation at high clock rates, the cache design is optimized for high-speed access. Enhanced testability in addition to scan and built-in self test (BIST) was added to guarantee data retention.

The Level 1 caches consist of a 32-kbyte instruction cache, a 32-kbyte data cache, and a 20-kbyte predecode cache, all built out of 8-kbyte blocks (10-kbyte blocks in the case of the predecode cache), which can perform one read and one write each cycle. The predecode macro includes the first stage of the instruction decode logic. Bypass and storage buffers are provided for data, instruction, and predecode caches with 4×16, 4×16, and 4×20 bit sizes, respectively, which allow data-in to flow back to the cache outputs.

An 8-kbyte cache block, shown in Fig. 2, consists of two arrays of 256 rows by 128 columns. Several different features are implemented in the three basic cache designs. As required by the microarchitecture, the cache block supports a byte write capability. Row decode, column write decode, sense amplifier decode, and datapath circuits use dynamic logic held by weak keepers. The predecode expansion logic is implemented in dual rail dynamic nonclocked logic. Scan is implemented in the sense amps themselves to provide array bitmapping and datapath logic testability. Testability is further enhanced by separate array and wordline power supplies through which data retention is tested at wafer-sort to prove functionality of the p-channel pullups in the cells. In order to reduce power, dual wordlines per row are used with a one-of-two block select.

The array testability addresses the problem during test where data in a memory cell can be held dynamically and be passed by the tester even if a p-channel transistor is not functioning. In order to counteract this test problem, the following approach is used. The memory cell array *VDD*, called *VDDMX* (for matrix) and the word line supply, *VDDWL* are brought out to separate C4 bumps, as shown in Fig. 3. These are connected using separate power supplies at wafer sort, but are all tied to the core *VDD* power planes in the package at assembly. To illustrate this method, consider the word line supply to be set at l.5 V for example, when the array supply is set at a higher voltage, such as 2.9 V. A ONE is written to the true side and a ZERO to the complement side. If the p-channel transistor on the true side is not connected due to some process defect such as a missing contact, that ONE is a Vtn with body factor down from 1.5 V, at about 0.7 V and that ONE level will not be pulled up to 2.9 V. However, when the word line is turned off, that 0.7-V true side level causes the p-channel on the complement side to be turned on and pull the complement side up to 2.9 V, flipping the cell, which can then be easily detected as a fail.

A further enhancement to testability is the implementation of scan in the sense amps to allow array bitmapping and downstream logic test. Sense amp scan was chosen because extra scan flip-flops connected to the sense amps would have slowed them down. The operation proceeds as follows: pairs of sense amps can be considered as master-slave latches. As shown in Fig. 4, first the even columns are read when the sense amp mux (*samx*) goes low and *sceven* (for "scan clock even") goes high on the master sense amp. Then *sciodd* (for scan clock input for transfer devices) goes high, presenting data to the slave sense amp which then reads it when *scodd* goes high. When *scieven* goes high again, the data is propagated into the next sense amp, and so on down the chain. The operation is repeated to read the odd columns.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

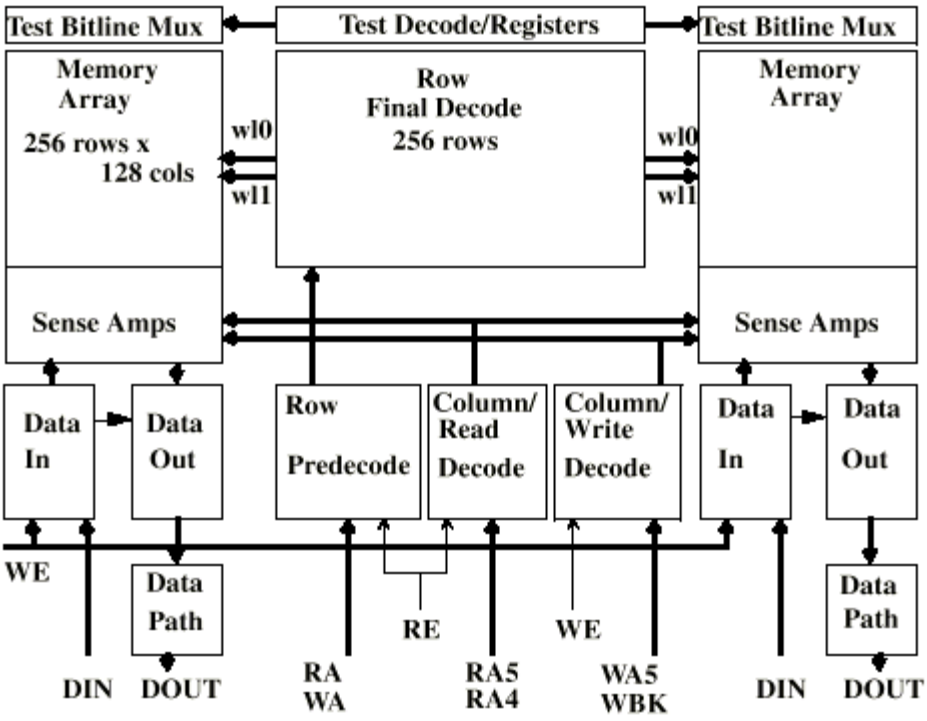

Fig. 2. An 8-kbyte cache block diagram.

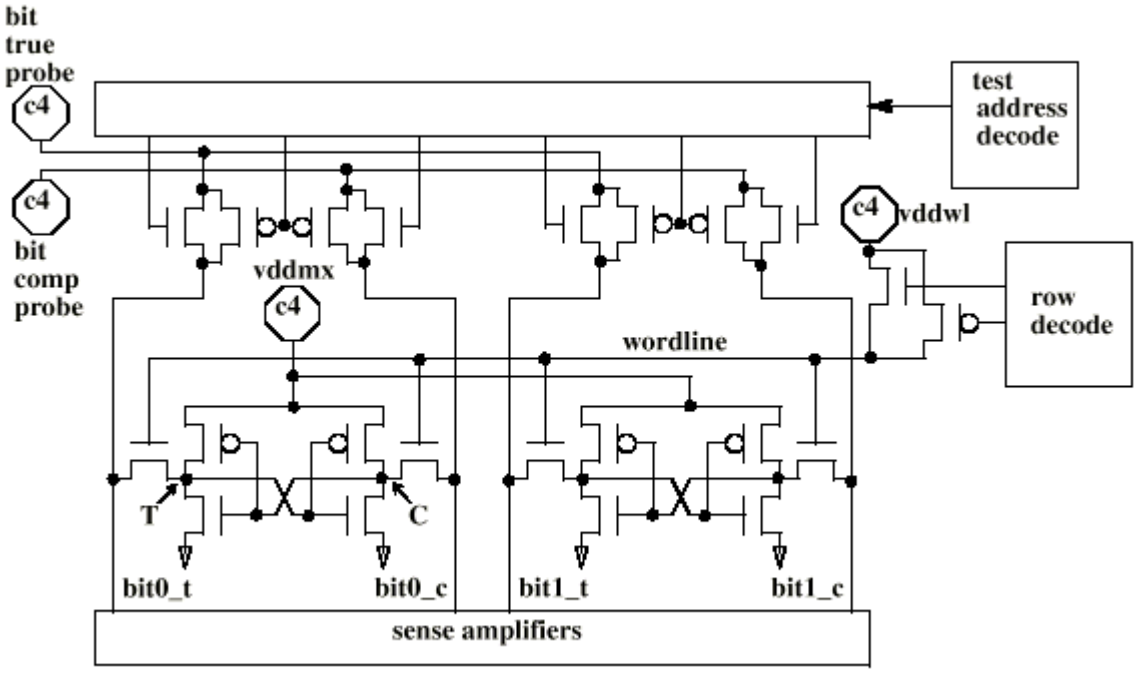

Fig. 3. Cache array test.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

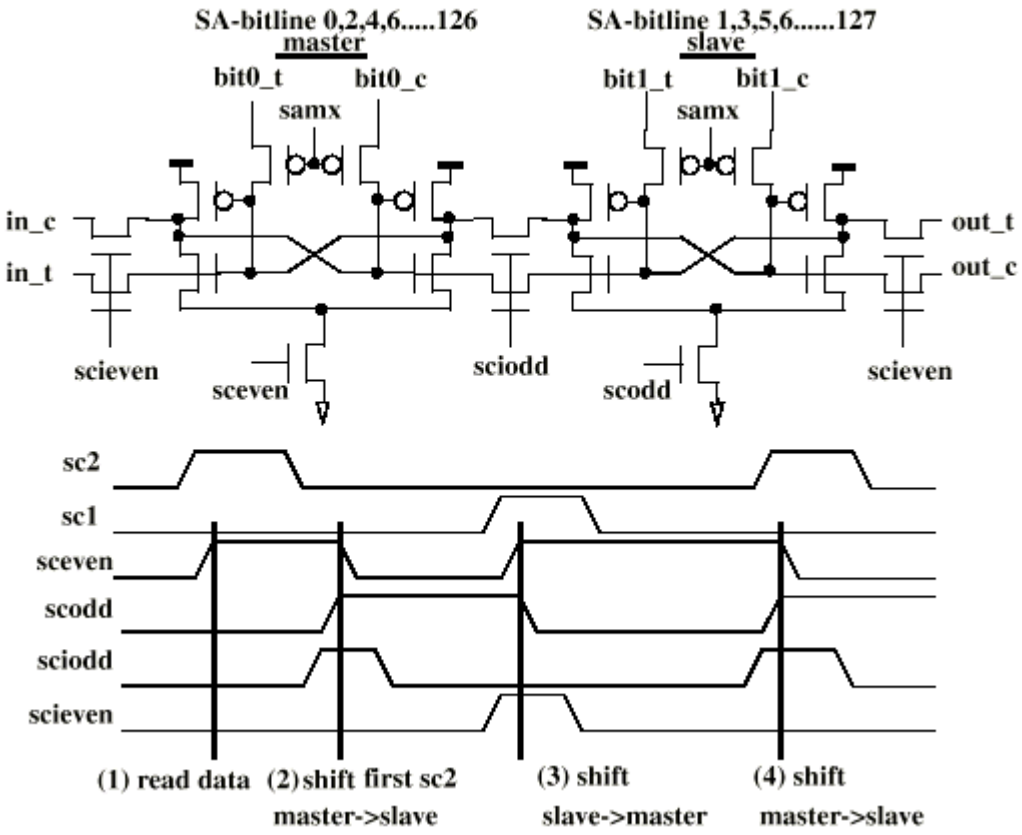

Fig. 4. Cache sense amp scan.

IV. Instruction Tag RAM

Critical to the single-cycle operation of the caches is a very fast tag access time and compare. The instruction tag RAM contains 512 20-b physical tags. Each tag has two additional bits for status. The tag RAM is logically two-way set associative, but is physically constructed with eight sets of tag-TLB comparators and eight sets of snoop comparators, with eight tags being read in each cycle. This allows all possible synonyms to be checked in a single cycle, at the expense of layout complexity and area. The tag RAM performs a read in the first half cycle and a write in the second half cycle. Write data is available at the beginning of the first half of the cycle and can be bypassed to the read outputs with no read access penalty. Fig. 5 shows how the bypass is implemented. The data to be bypassed is fed directly into the sense amp, which avoids the additional delay a follow-on max causes, and saves the area of the mux. The write data, present at the beginning of the cycle, is bypassed into data out, and then later written into the memory cell during the write phase.

V. PLA

The design requirements for the PLA, a control block used in the floating point unit (FPU), are speed and immunity to bit-line coupling, requiring specialized circuit solutions. The PLA shown in Fig. 6 contains 17 inputs and 104 outputs. It has the capacity for 800 minterms although only 760 minterms are used. The PLA uses twisted, fully differential bit lines in both the AND and OR arrays to improve speed and counteract bit line coupling. A single-ended bit line would have had severe coupling problems because, unlike a ROM, multiple pulldowns on a bitline are possible in a PLA which can cause a large voltage excursion on adjacent bit lines, hence large coupling on a victim bit line. Each row can be programmed to a particular bit line pair by connecting the drain of a transistor on the true bit line, and connecting a partial transistor, or “PLACAP,” on the complement bit line. The transistors and PLACAP’s are connected (or not) through the Contact 1 mask layer for late definition. A half-strength cell, driven by the read signal, pulls down each complement bit line which is then compared to its corresponding true bit line.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

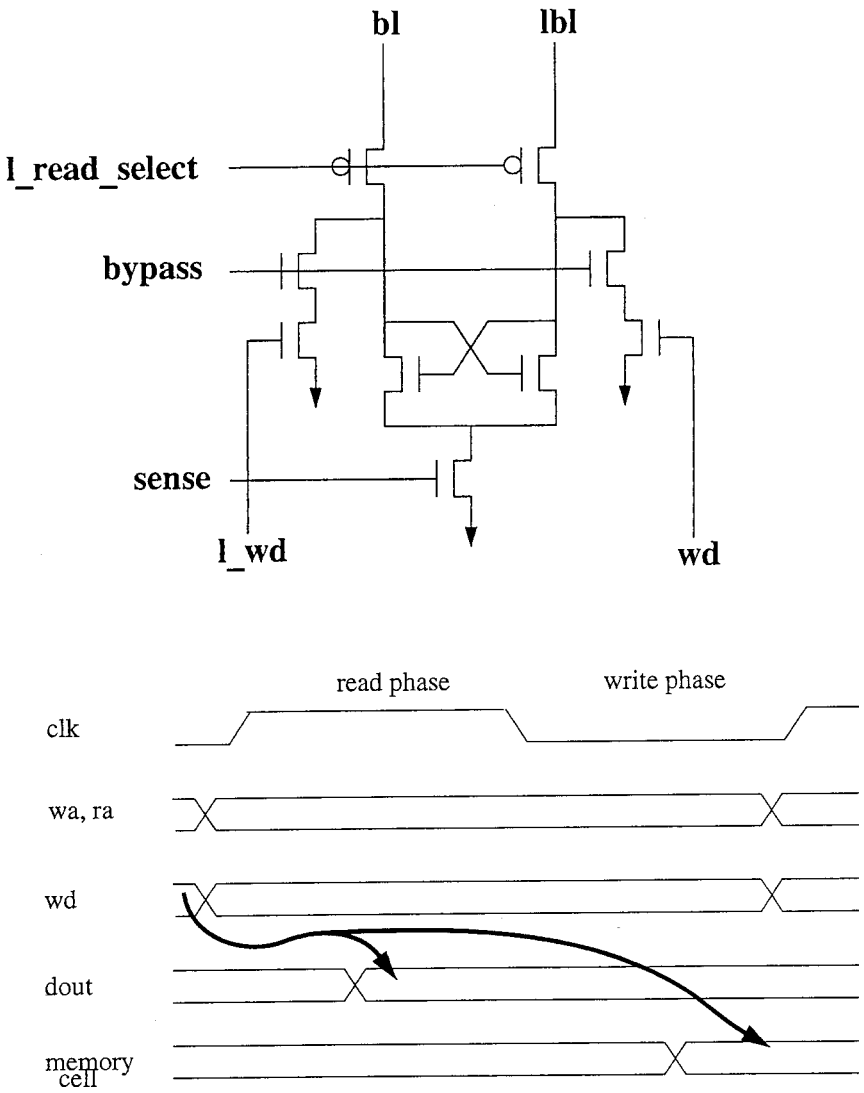

Fig. 5. Tag RAM bypass.

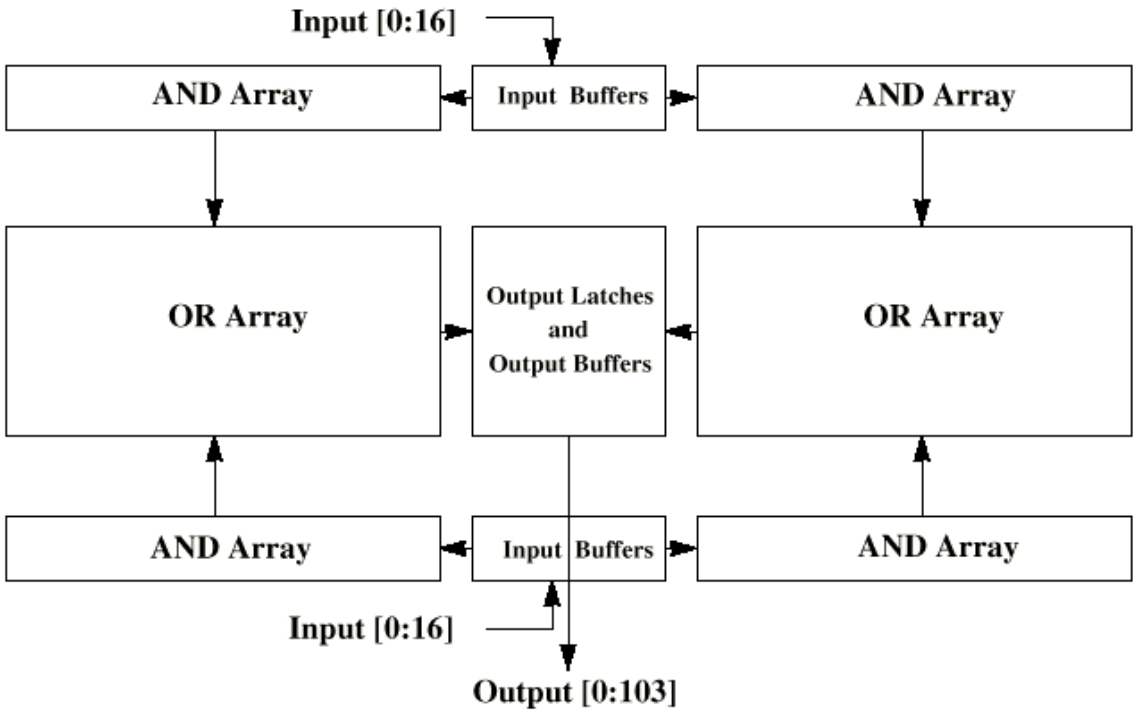

Fig. 6. PLA block diagram.

The PLACAP is used to provide capacitive matching between true and complement bit lines, as shown in Fig. 7. We chose this over a grounded gate transistor because the grounded gate transistor would have taken much more space as an array element. PLACAP was chosen over a diffusion capacitor because the diffusion would not have the matching drain-gate overlap Miller capacitance. The greater precision matching of PLACAP allows the building of at least two times larger arrays when designed to provide adequate margin over all simulation corners.

Both strobed and nonstrobed sense amps could be used for the AND array sense amp, as shown in Fig. 8. Although for the short AND plane bit lines, the nonstrobed sense amp would have been the simpler choice, for some process corners the true and complement bit lines fall very fast relative to the response time of the sense amp, producing a glitch on the output. To avoid this, we chose the strobed, self-timed sense amp, which also provided a faster access time for the PLA.

VI. OP Code ROM

In contrast to the PLA, which used fully differential bit lines, the op-code ROM, due to its large size of 4k words by 169 b, uses single-ended bit lines sensed differentially with respect to a reference line. To minimize area, the bit lines use minimum pitch metal 1 with no shielding lines between bit lines, where shielding is sometimes used to avoid coupling. This implementation uses a resistive pullup bit line load, realized by a grounded-gate p-channel, shown in Fig. 9.

The main difference in the method used to combat coupling in the ROM compared to the PLA is that in a ROM, each bitline has only one pulldown. Combined with the resistive pullup, the voltage excursion is minimized to about 0.7 V. This greatly reduces the amount of coupling into an unaccessed bit line. Fig. 9 shows an unaccessed bit line with two adjacent accessed lines (not shown), and the reference line. The bit line is first pulled down by the coupling, then pulled back up by the resistor, providing sensing margin when the sense amp fires. The small inflection on the bit line signal indicates when the sense amp fires. The two accessed lines now couple in as they are pulled up, coupling the unaccessed line higher. The resistive bit line load quickly brings the line back down to the proper equilibrium level before the next cycle.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

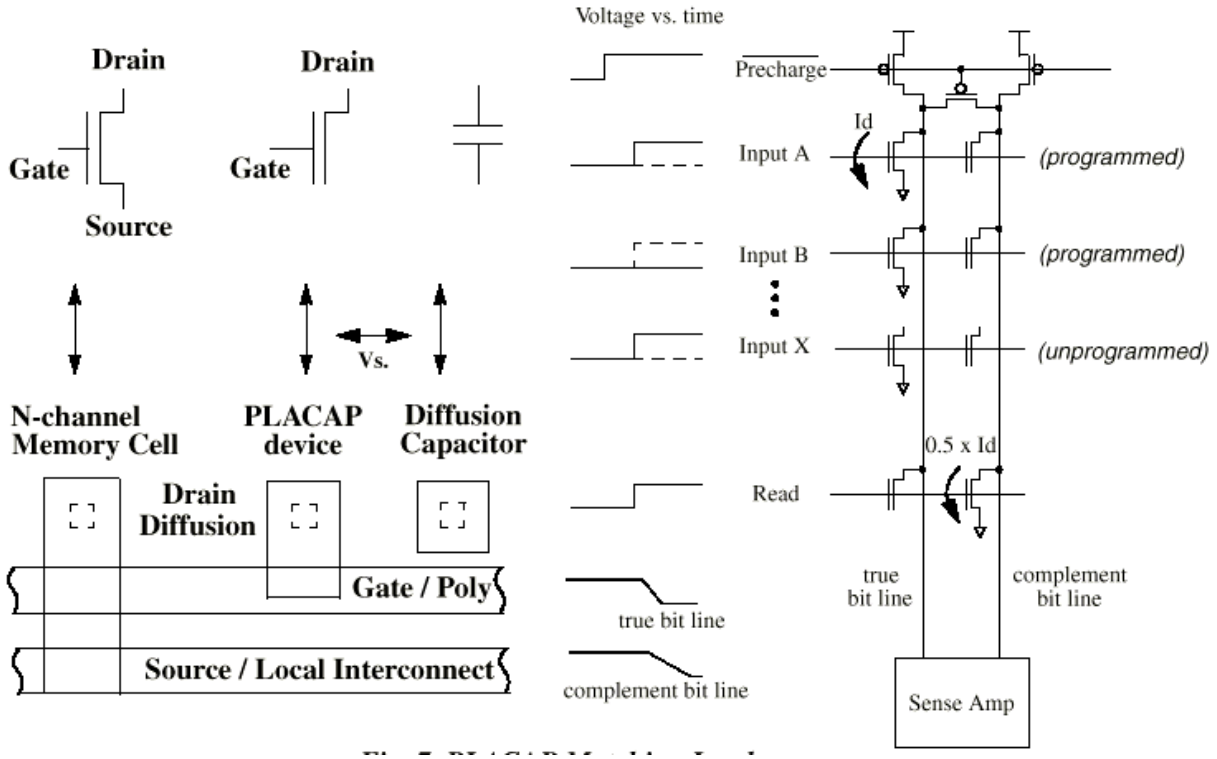

Fig. 7 PLACAP Matching load.

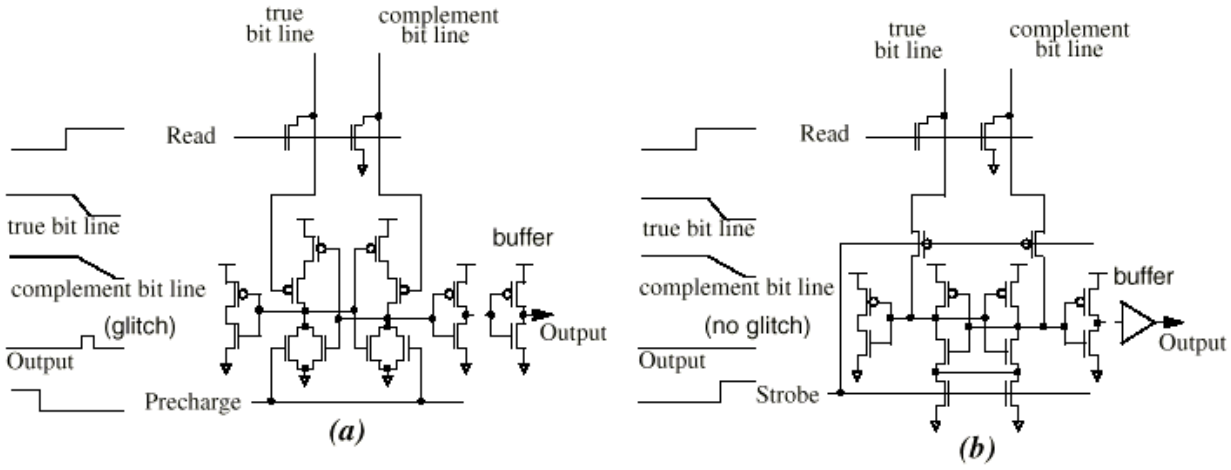

Fig. 8 (a) Nonstrobed and (b) strobbed sense amplifiers.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

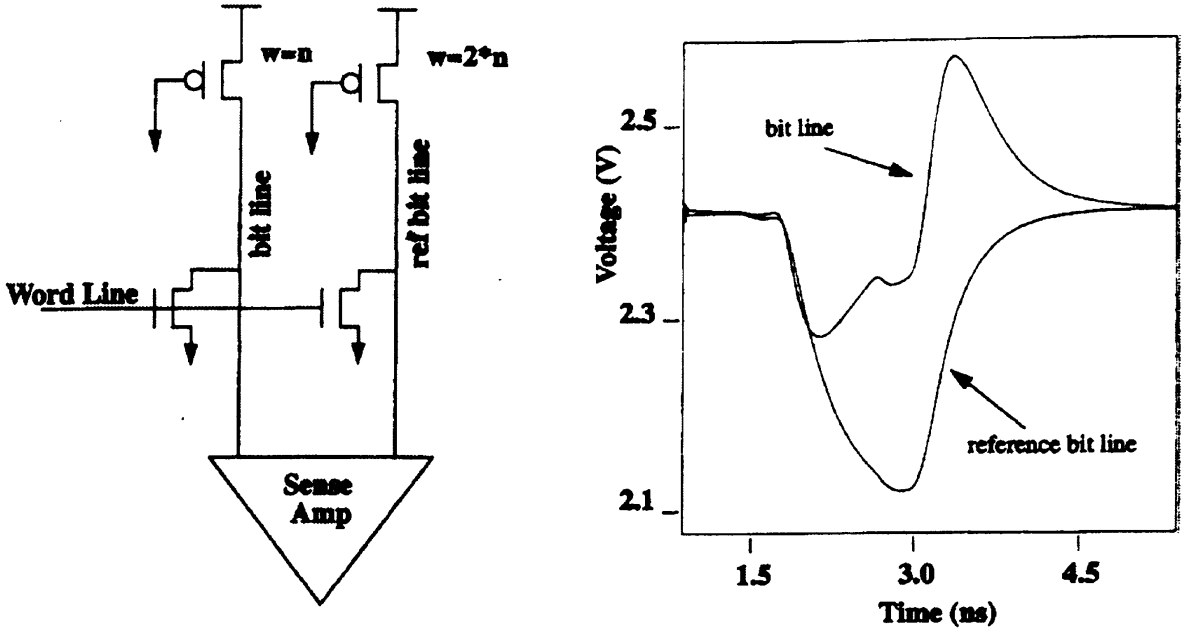

Fig. 9 Op-code ROM coupling analysis.

There are several advantages to resistive bitline loads in comparison to precharging and other methods. They do not require a half cell for the reference line and all bit lines can be minimum pitch. There is no requirement for 2× capacitances on the reference lines. Mismatches in bit line capacitance are tolerated. Bit line swing is clamped so that coupling "victims" recover. Resistive load bitlines with reference line have nearly the same density as cascade schemes, but without the traditional cascade drawbacks, such as sensitivity to coupling and supply noise. Cascade methods limit minimum VDD operation due to the Vtn drop. They often have static power dissipation and need tail currents to prevent bitlines from charging up during power-down modes. The robust operation and reduced sensitivity to coupling of resistive bit-line loads provide an extra measure of performance.

VII. Edge-Triggered Latch

The high processor clock frequency requires a flip-flop optimized for speed and reduced sensitivity to clock skew. The flip-flop topology is implemented by the edge triggered latch (ETL) [8]. As shown in Fig. 10, the ETL employs a precharged node, *X*, and a clock-derived, integrated one-shot which provides a transparency period. During this period, the data is sampled into the latch.

The transparency period, defined by inverter chain Il-I3 is typically about 250 ps in this design, and provides an attractive attribute of level-sensitive latches: data can arrive at the latch even after the assertion of the clock. This attribute allows for time-borrowing or slack passing, or alternatively, can be thought of as minimizing the effects of clock uncertainty on cycle time. The transparency period, though advantageous, causes an increase in the hold time, requiring a detailed hold timing analysis to avoid a timing failure.

In Table I, the performance of the ETL (tsu + tcq) is compared to two level-sensitive latch pairs [9], [10] and a sense-amplifier based flip-flop [11] (Fig. 11). As can be seen, the ETL outperforms the other elements. More precisely, with equal clock loads, the ETL can improve the cycle time centered at 250 MHz by 6.5% when compared to the latch pair type 2. Further, at l/2 the clock load, ETL still has a 5% advantage over the same latch pair.

Based on similar principles, the dual-rail pulsed ETL circuit was designed for use in dynamic self-resetting structures. As shown in Fig. 12, on the rising clock edge and depending on D, either of the predischarged nodes, *QP* or *QBP*, will be asserted. The assertion of one of the outputs activates a delay chain that finally resets the flop in the precharged, quiescent state.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

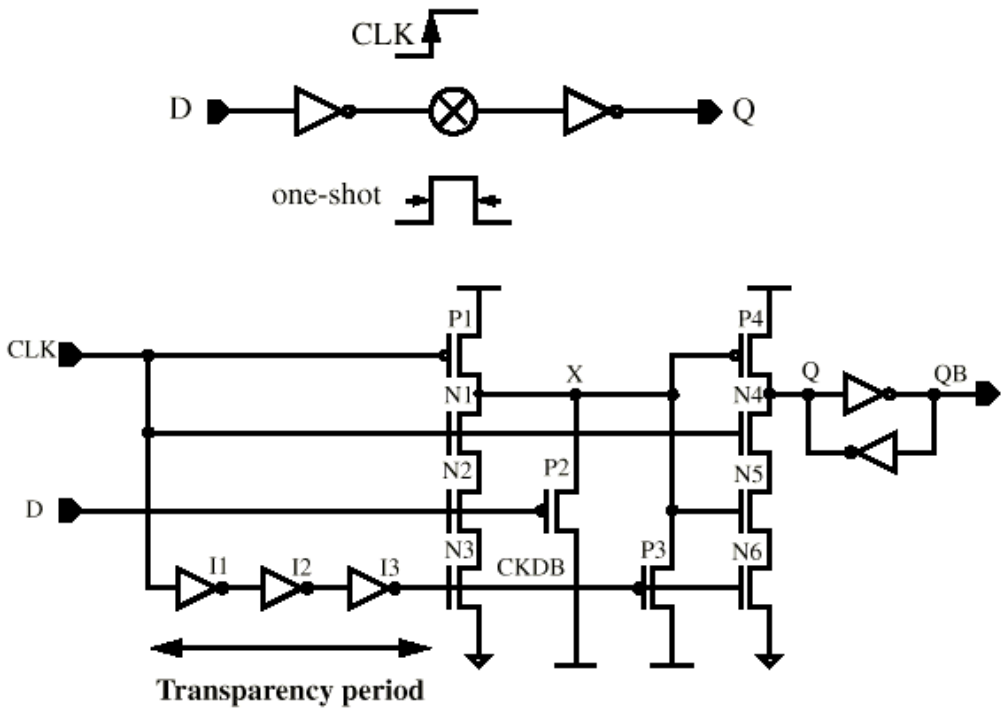

Fig. 10 Edge-triggered latch.

Table I ETL Speed Comparison with Other Latching Elements

	ETL	**ETL 1/2**	**SAFF 1/2**	**Latch Pair type 2**	**Latch Pair type 1**
$t_{su} + t_{cq}$ (ps)	270	340	440	530	750
Gain (%)	6.5	5.0	2.2	—	–4.5

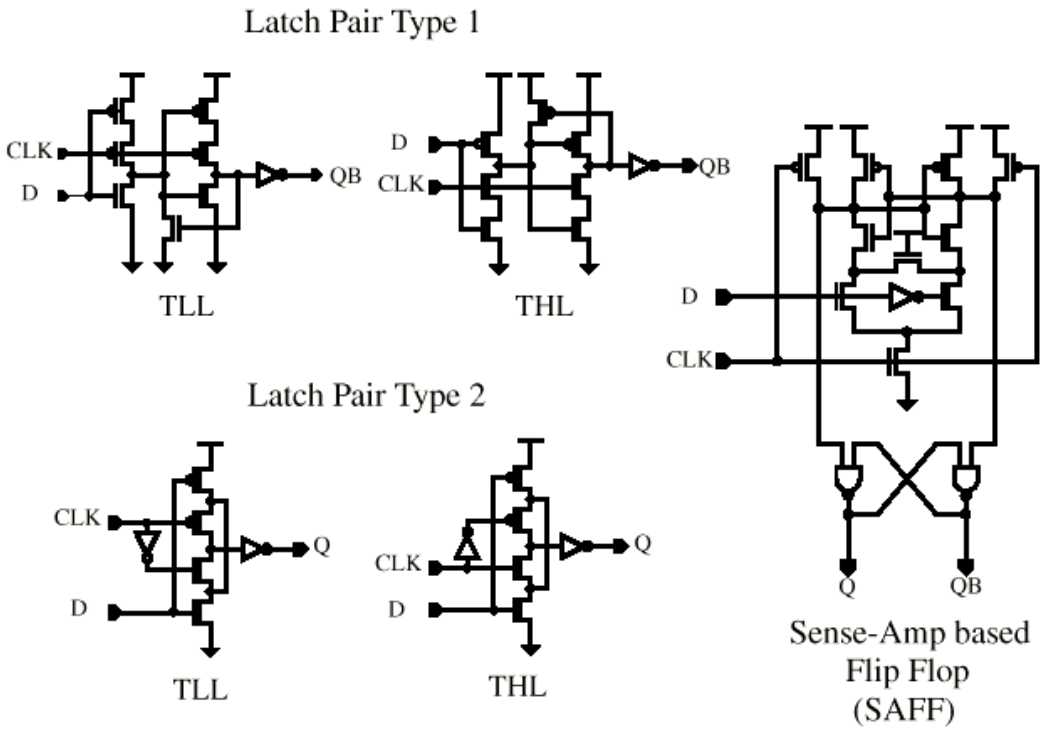

Fig. 11 Latch pairs and sense-amp-based flip-flops.

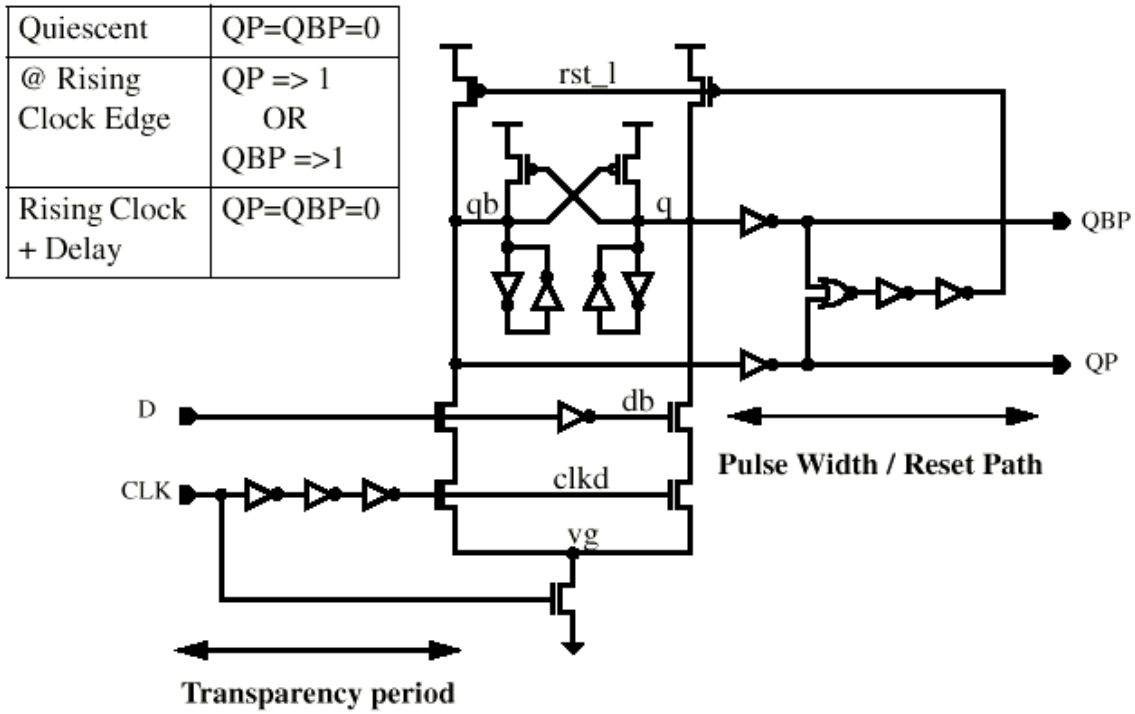

Fig. 12. Dual-rail pulsed ETL for self-resetting logic.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

The dual-rail pulsed ETL acts as a buffer stage between static and dynamic logic: it converts single-ended signals to dual-rail monotonic signals suitable for dynamic applications. Further, it acts as a flop for a brief period determined by the reset path. This path must be long enough such that the flop data can propagate to and be consumed by the succeeding stages of dynamic logic before it is reset.

VIII. Clock Distribution

Clock distribution needs to satisfy requirements of low-skew, variable clock loading, system management mode (SMM)-required power-down by clock gating, and synchronization with the PLL. The single-phase clock signal called *PCLK* is generated and synchronized to the bus clock (*BCLK*) by the PLL. The distribution through a tree of four levels of clock buffer, L0 through L3, as shown in Fig. 13, produces less than 150 ps of skew, as determined by simulation. For use in power-down mode, the Ll buffer can be gated off. To maintain synchronization, the clock signal is brought back to the PLL by a free-running clock replica path that delivers *PCLK2*, which is matched to *PCLK*. The final clock signal is distributed by a meshed grid of wires and is driven by two columns of L3 clock drivers. CAD tools were used to assess the regional clock load and to program the regional clock driver strength using the contact 2 mask layer, which minimized variation in clock rise time and clock skew.

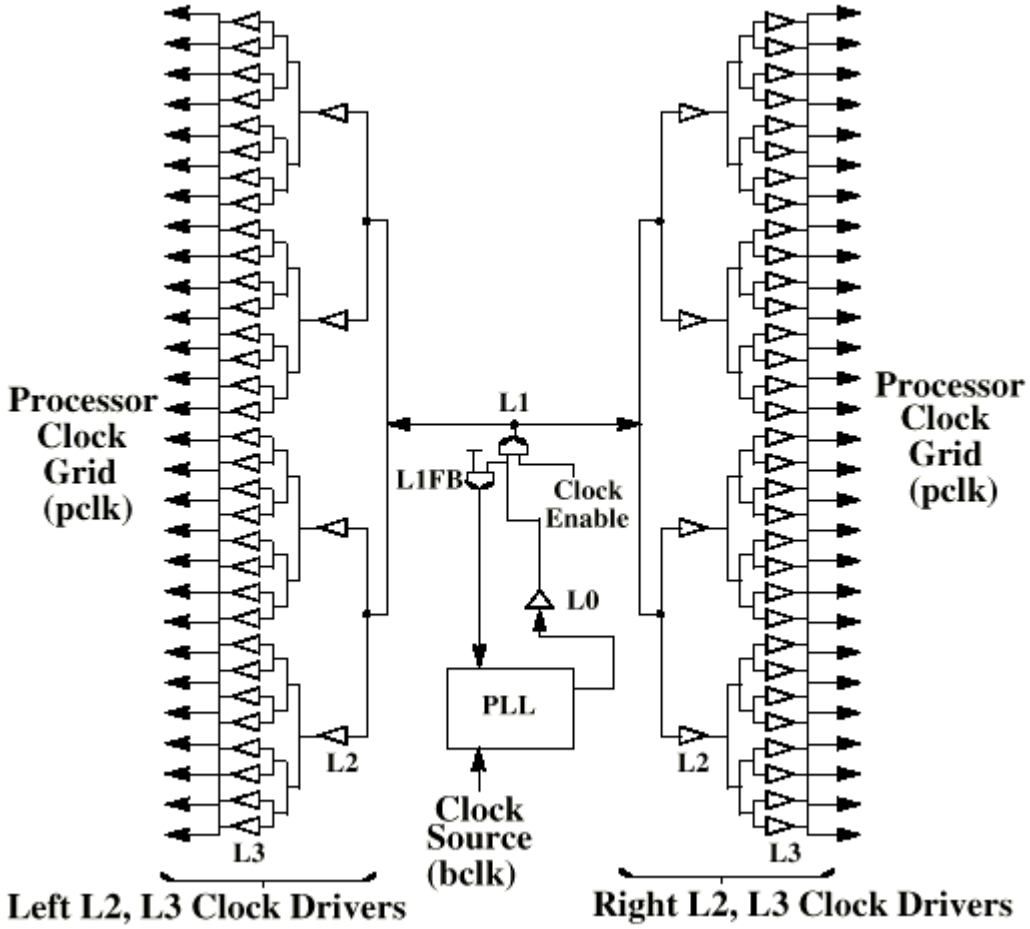

Fig. 13. Clock generation and distribution.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

IX. Phase-Locked Loop (PLL)

Low jitter, accurate synchronization, frequency multiplication, and high-frequency operation are required of the PLL to meet processor performance specifications. The PLL shown in Fig. 14 provides half-integer multiplication via the divide-by-two in the reference (*BCLK*) path. Power is provided by a dedicated pin called *VDDA* which is tied to *VDDIO* (3.3 V) on the motherboard. Using this copy (versus the chip level *VDDIO*) avoids power-supply switching noise generated by the I/O switching currents interacting with the package power and ground inductance. The VDDA supply can be set to VSS for test purposes, switching the MUX to allow the bypass clock to drive the clock distribution. The PLL itself is based on a fully differential voltage-controlled oscillator (VCO) design which aids common-mode noise rejection. The phase frequency detector (PFD) produces *UP* and *DN* pulses, each of which is translated to true and complement pulses that control the charge pump. These *UP* and *DN* pulses transmit phase and frequency information about *PCLK2*, and its alignment versus the reference clock *BCLK* to the charge pump; the charge pump controls the frequency of oscillation of the VCO via the loop filter voltage (*LFV*). In lock, *PCLK2* is phase aligned with *BCLK*, and the frequency of *PCLK2* will be M/2 times greater than the frequency of *BCLK*. The small swing amplifier takes the low voltage swing from the VCO and translates it to the core *VDD* voltage level. The feedback path is completed through an output divide-by-two (for 50% duty cycle), output mux, L0 and Ll global clock grid, and finally an internal replica L2 and L3 clock block.

The VCO frequency is controlled by the *LFV* and the VCO amplitude is controlled by *VCOA*, as shown in Fig. 15. CML-style triode loads [12] produce a fixed resistance (*R*) for a fixed *LFV*, and when used in conjunction with the input load capacitance of the next stage it is driving (*C*), a per-stage delay of *R* * *C* is produced, which is insensitive to supply noise-induced current fluctuations. Minor-loop feedback was employed to control *VCOA*, the end result being a constant amplitude of oscillation independent of the frequency of oscillation [13].

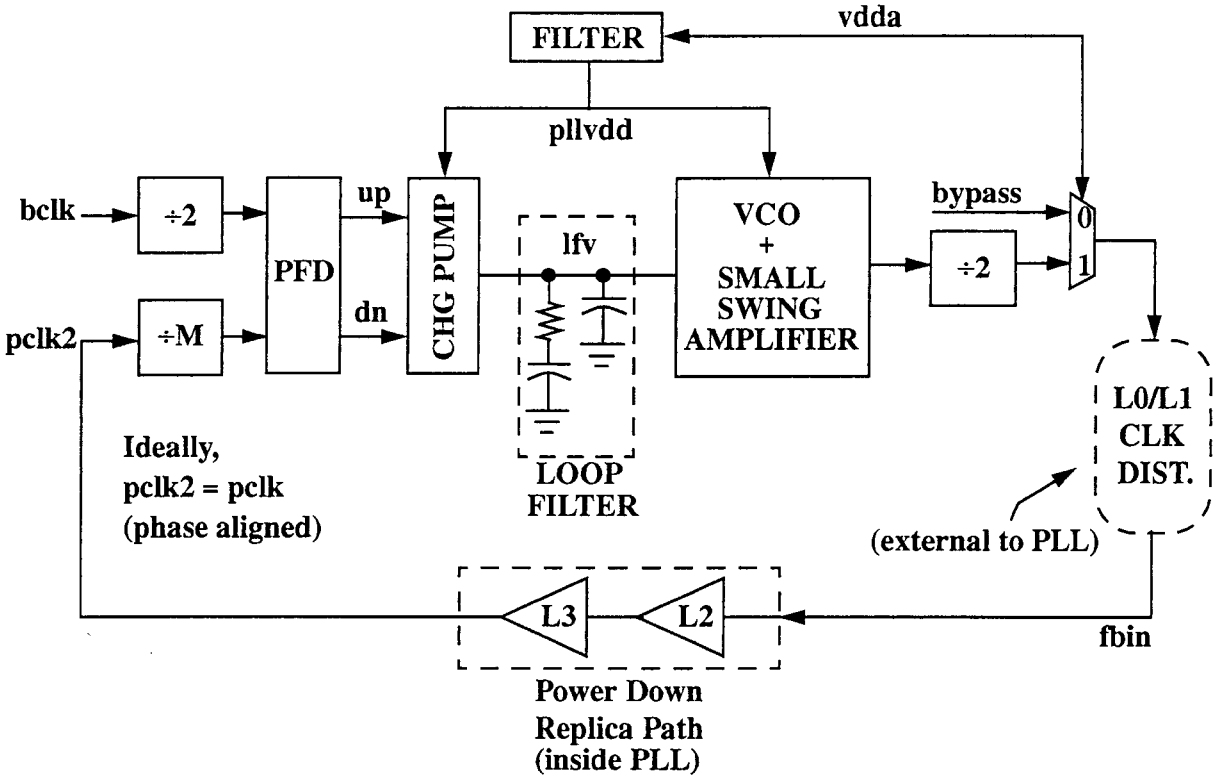

Fig. 14. Phase-locked loop block diagram.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

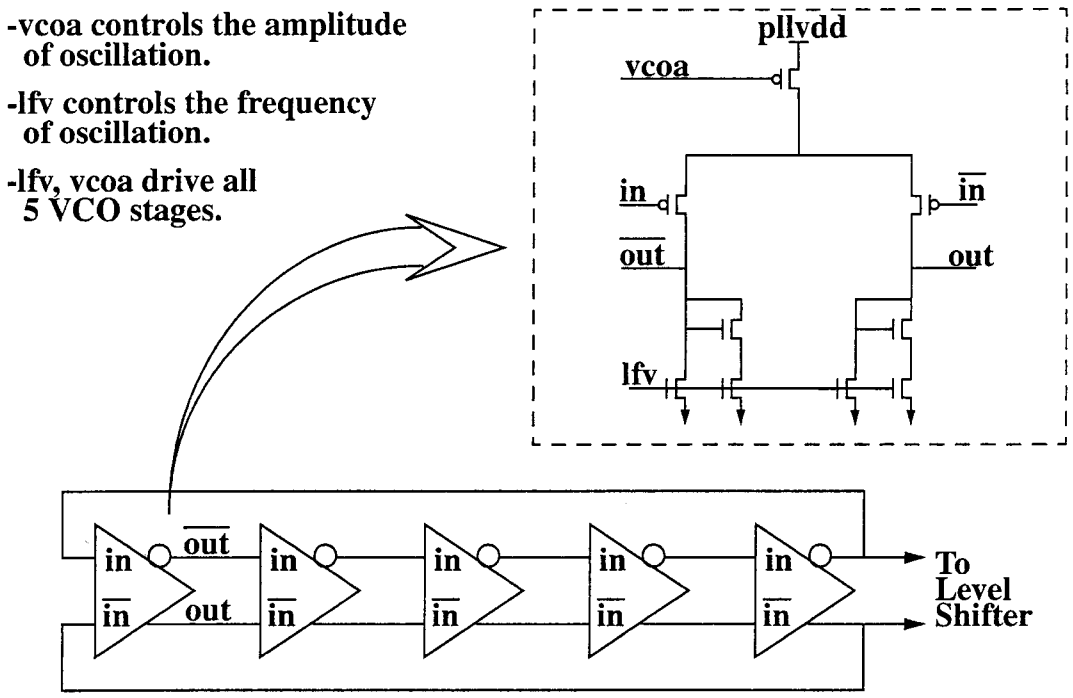

Fig. 15. Voltage-controlled oscillator using linearized loads.

The charge pump, shown in Fig. 16, uses a bandgap reference which is based on substrate PNP transistors to produce a process-, temperature-, and supply-independent pump current. The series switch topology, which employs a voltage follower buffer, was used to suppress charge sharing errors that can occur when the *UP* and *DN* pulses go from inactive (not controlling the *LFV* node) to active (controlling the *LFV* node) [14]. Both linearity and on-resistance of the switches was improved by operating them from the 3.3-V supply. A high-speed level translator (VDD-to-VDDA) and true and complement pulse generator, shown in Fig. 17, were developed to cleanly shift between voltage domains, as well as to generate tree and complement pulses that intersect at VDDA/2, which is the ideal crossing point for the charge pump switches. Crosscoupling PI and P2 ensures no crowbar current path from *VDDA* to ground. Unfortunately, this configuration also means nodes A and B cross at or near ground. The compound inverters of P3–N6 and P7–N10 compensate for this effect. When node A makes a high-to-low transition, transistor N6 is off. But when A transitions low-to-high, N6 is on. N6 modifies the effective beta ratio of the compound inverter circuit, giving a beta ratio (trip point) that is state dependent. Device W's of P3–N6 (as well as P7–N10) are selected so that *C* and *D* cross at VDDA/2 Output buffering inverters then drive the charge pump switches.

The power-supply filter shown in Fig. 18 was used to supply the VCO and charge pump to reduce jitter caused by supply noise. The filter is based on an n-channel pass transistor, MN0. The gate of MN0 is held at VDDA by the highly resistive MP0. To avoid noise coupling from *VDDA*, the well of MP0 is tied to the gate of MN0. This causes the p^+ to n-well diode to be forward biased during power-up, so a guard ring was needed around this device. Gate oxide capacitor MN1 and resistor MP0 provide a very stable MN0 gate voltage. The bleeder transistor biases MN0 in a high gm range, lowering the impedance on *PLLVDD*. Simulating a 100-ps risetime, 50-mV amplitude square wave on VDDA causes a ripple of only 22 mV on *PLLVDD*.

The jitter was measured by muxing out the *PCLK* signal through a standard output driver while the full clock system was running at 233 MHz. The measured cycle-to-cycle jitter is shown in Fig. 19 to have a peak-to-peak value of ±33.6 ps, for a total of 67.2 ps.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

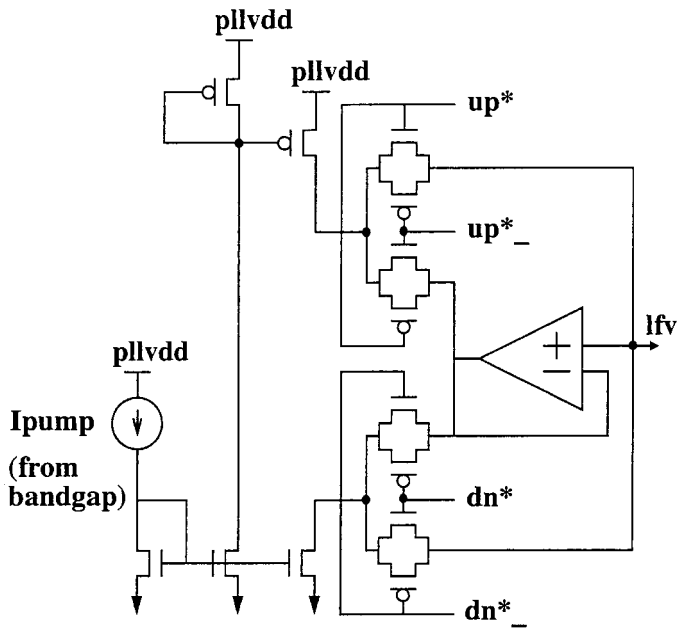

Fig. 16. PLL charge pump.

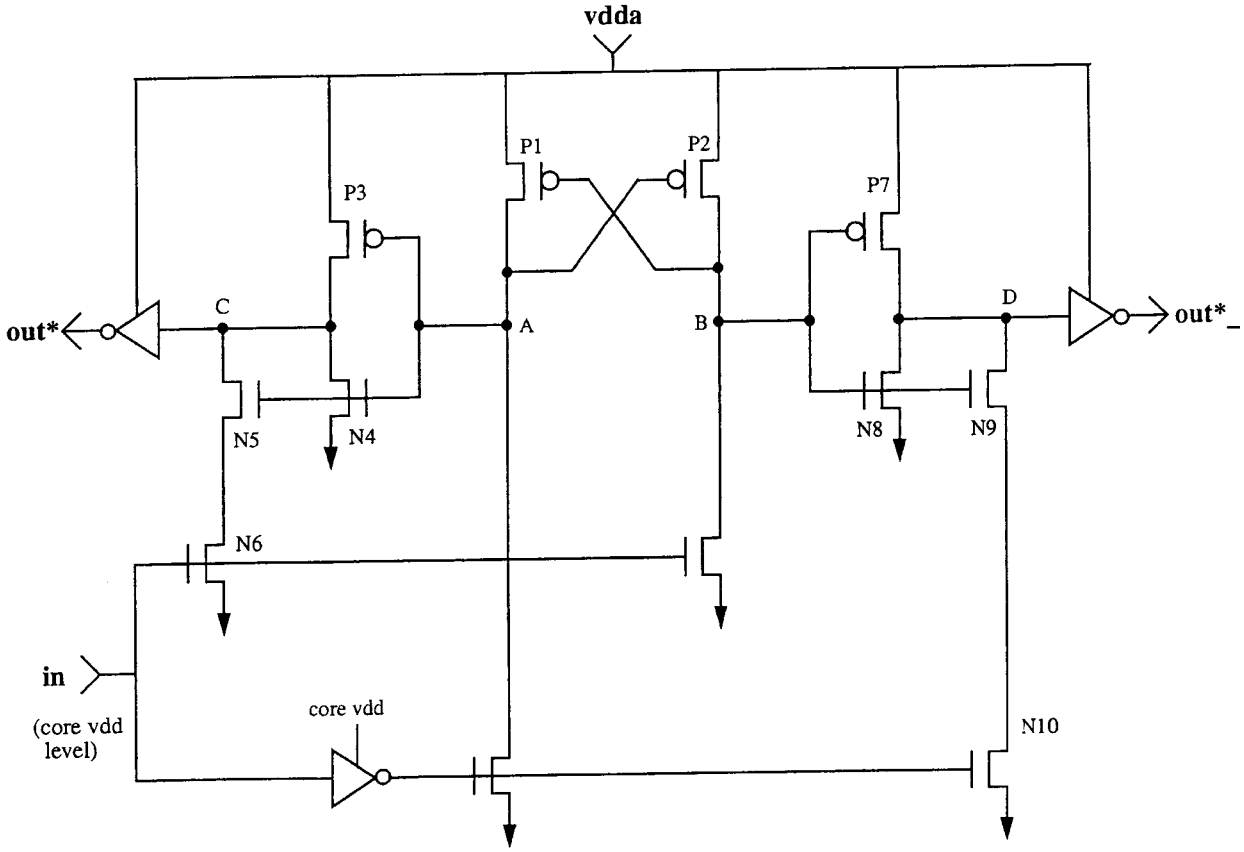

Fig. 17 PLL level translator and symmetric true/comp. pulse generator.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

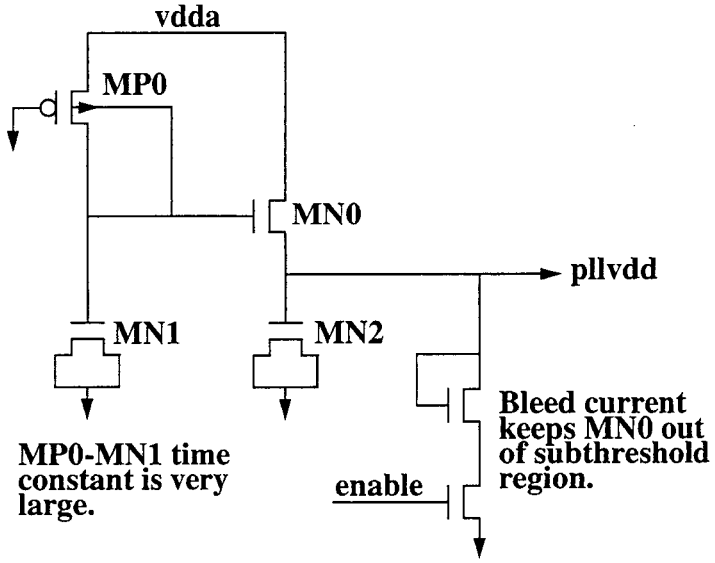

Fig. 18. PLL power-supply filter.

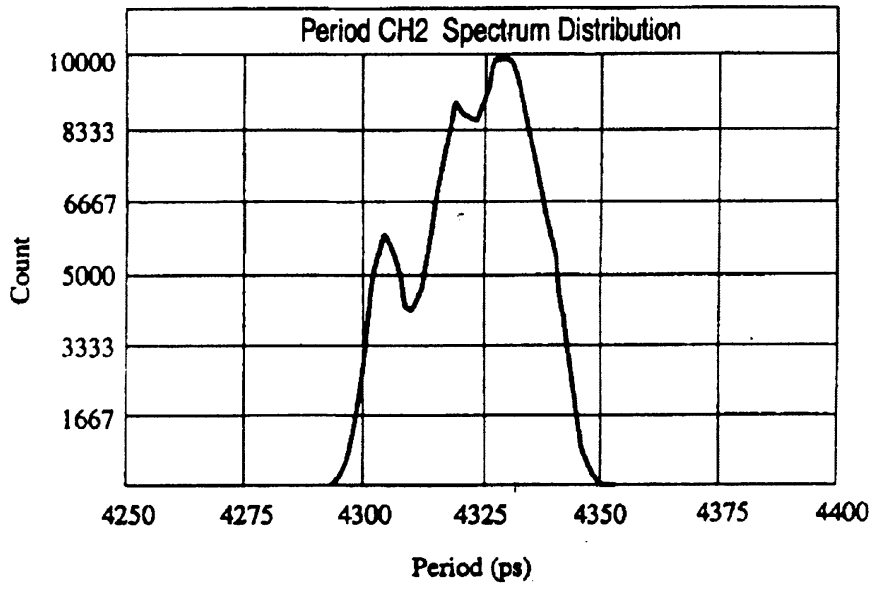

Fig. 19 PLL jitter measurement.

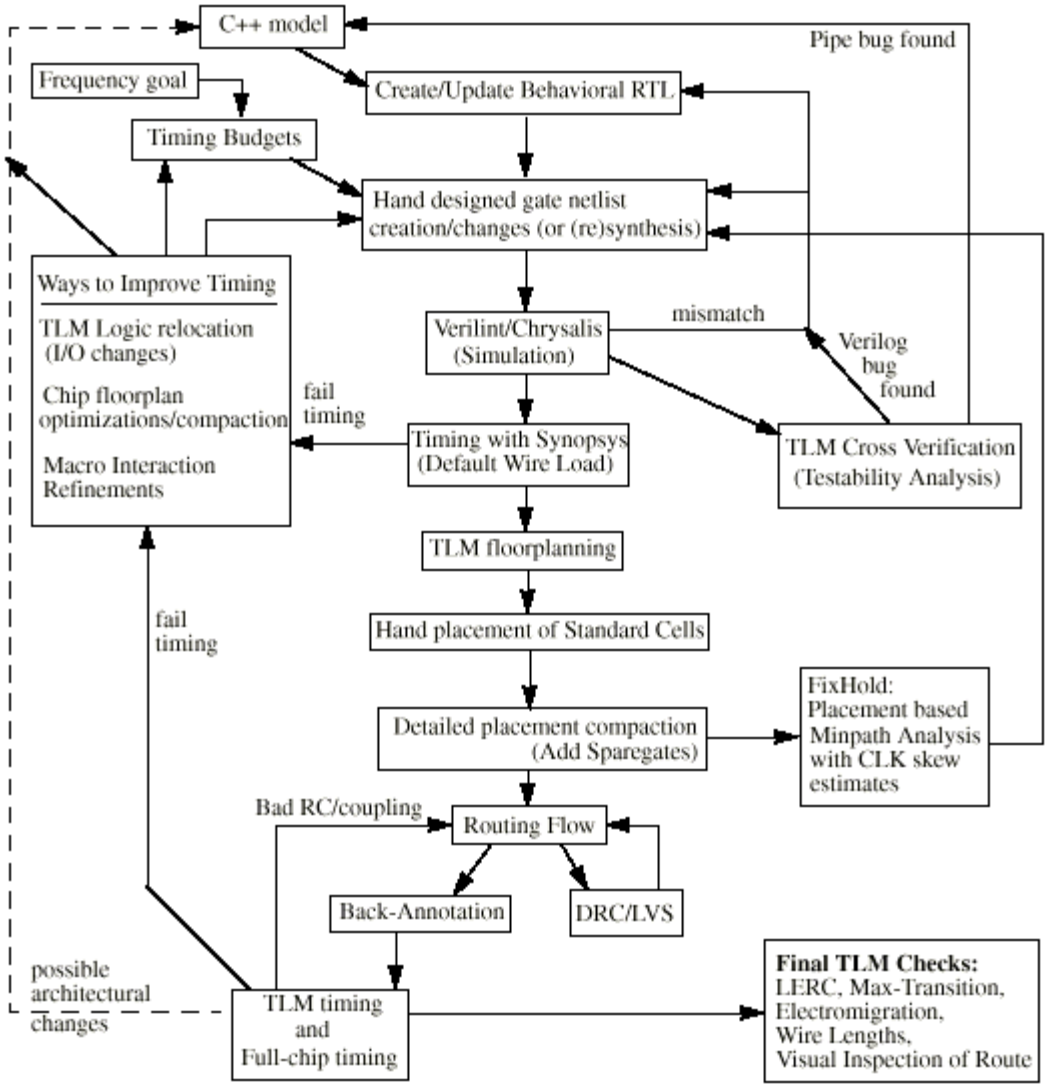

Fig. 20. Top level (TLM) design flow.

X. Timing Methodology and Top-Level Module (TLM) Design Flow

The datapaths not in the custom blocks and the control logic are implemented in standard cells and placed and routed as part of the top level module (TLM) design. The TLM design flow is shown in Fig. 20. The C++ pipeline model serves as the "golden" reference from which we create an intermediate behavioral RTL verilog file. A gate-level verilog file is then created by hand (synthesis is seldom used) and each of the three models are compared against each other. The gate and RTL models are compared using a formal verification tool and sometimes verilog simulations. Then the RTL and gate netlists are each separately compared against the C++ pipeline model using a cross-verification simulator (VCS or Speedsim).

The initial timing of the gate netlist is accomplished with a static timing analyzer using a default wire load model. Floor-planning and hand placement of the individual gates is then completed before giving the data to a routing tool (Cell3). Using the placement data, an internally developed minpath analysis tool (Fixhold) is run using *CLK* skew estimates to ensure holdtime constraints are met. The design is then routed and a parasitic-extraction tool is run on the routed database to extract distributed RC delay values. Timing analysis is then performed on each TLM and ultimately the entire chip. Vericheck design rule check (DRC) and layout

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

versus schematic (LVS) are also run on the completed route. Finally, the database is analyzed for electrical integrity purposes (wire lengths, electromigration, max-transition violations, electrical rules checking, etc.) Any type of undesired results along the design flow causes looping iterations to take place until the entire chip meets all of the design constraints and is ready for tapeout.

The timing analysis and allocation or budgeting methodology is based on a gate-level static timing analysis tool, an in-house budgeting tool, a more-accurate RC extraction tool, and a delay calculation tool. Various in-house programs are used to bind all of the above together. The timing analysis and budgeting methodologies are both designed to work together to provide consistency and accuracy throughout the evolution of the chip by making use of as much detailed design information as is available at any given time. As the design evolves, the timing methodology is required to support the following activities. In the early timing phase, time budgeting is done at the block and subblock level to derive and check the consistency of timing constraints for synthesis or manual design. In the interim timing phase, pre- and post-layout timing analysis of major design blocks is done in the context of the whole chip, before the entire chip is ready to be timed. In the late timing phase, post-layout RC extraction and timing analysis are done on the entire chip.

Most cells are placed by using text-based directives which are read by an in-house set of programs. A graphical display of the results aids in iteration. Most of the chip is placed using these scripts which greatly aid layout productivity and yield density and timing results similar to a full custom design.

The top level modules are constructed such that they include all logic and any wires passing through their "air-space." The inputs and outputs of each TLM are routed to a predefined I/O footprint which was derived by understanding the routing requirements of each top level net. This allows most of the chip construction to be done early during the construction of the TLM's. Construction of the chip then consists merely of placing the TLM's and stitching them together with very short final routes.

XI. CMOS Process Description

The AMD-K6 processor is fabricated on a 0.35-μm CMOS process with five layers of metal, shallow trench isolation (STI), tungsten local interconnect (LI), and C4 flip-chip die attach. A scanning electron microscope cross section of the process is shown in Fig. 21. The STI is required for tighter active area packing and smaller transistor width variation. Local interconnect is used to connect poly and diffusion without an intervening contact layer. It provides tighter layout of SRAM cells, standard cells, and custom macros. LI is realized by a Damascene tungsten process, which means the interconnect pattern is defined by trenches etched in the oxide, filled with tungsten, and later polished off. Chemical-mechanical polishing (CMP) is an integral part of the process, providing planarization, stacked vies, and metallization density unobtainable by any other means. C4 die attach allows much better power routing on the chip and low-inductance power supply connections to the package. The process is based on p-epi on a p+ substrate for less susceptibility to latchup. The process dimensions are listed in Table II.

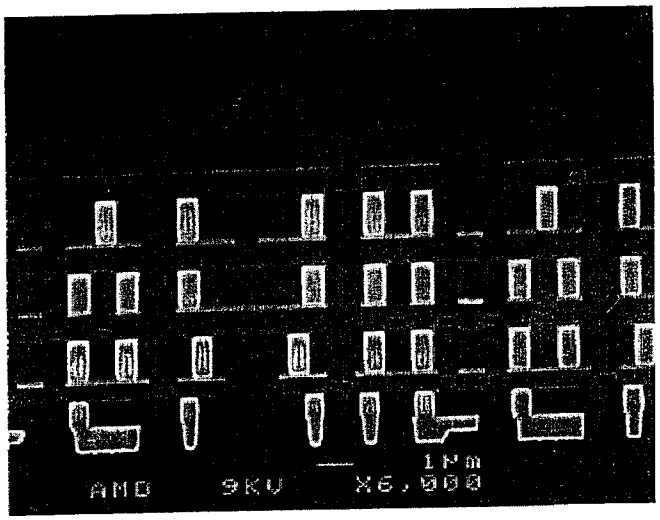

Fig 21. A 0.35-μm process cross section.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

Table II. 0.35-μm Process Dimensions

Gate Oxide Thickness	7.0 nm
Local Interconnect Pitch	1.0 um
Metal 1 Pitch	1.4 um
Metal 2 Pitch	1.4 um
Metal 3 Pitch	1.4 um
Metal 4 Pitch	1.8 um
Metal 5 Pitch	4.8 um

XII. Performance and Implementation

At a case temperature of 70°C and a supply voltage of 3.2 V, the processor operates at 266 MHz and dissipates 32.3 W. A room temperature shmoo plot is shown in Fig. 22. At a VDD voltage of 3.2 V, the external clock period of 12.3 ns translates to an internal frequency of 285 MHz using a PLL frequency multiplier of 3.5×. In the stop grant mode, the clock is shut down to most of the chip, but the PLL is kept running, allowing power to be reduced to 745 mW. When measured at 233 MHz, the processor performs at a rate of 73.5 Winstones 97 Business on Windows NT 4.0 and 53.87 Winstones on Windows 95.

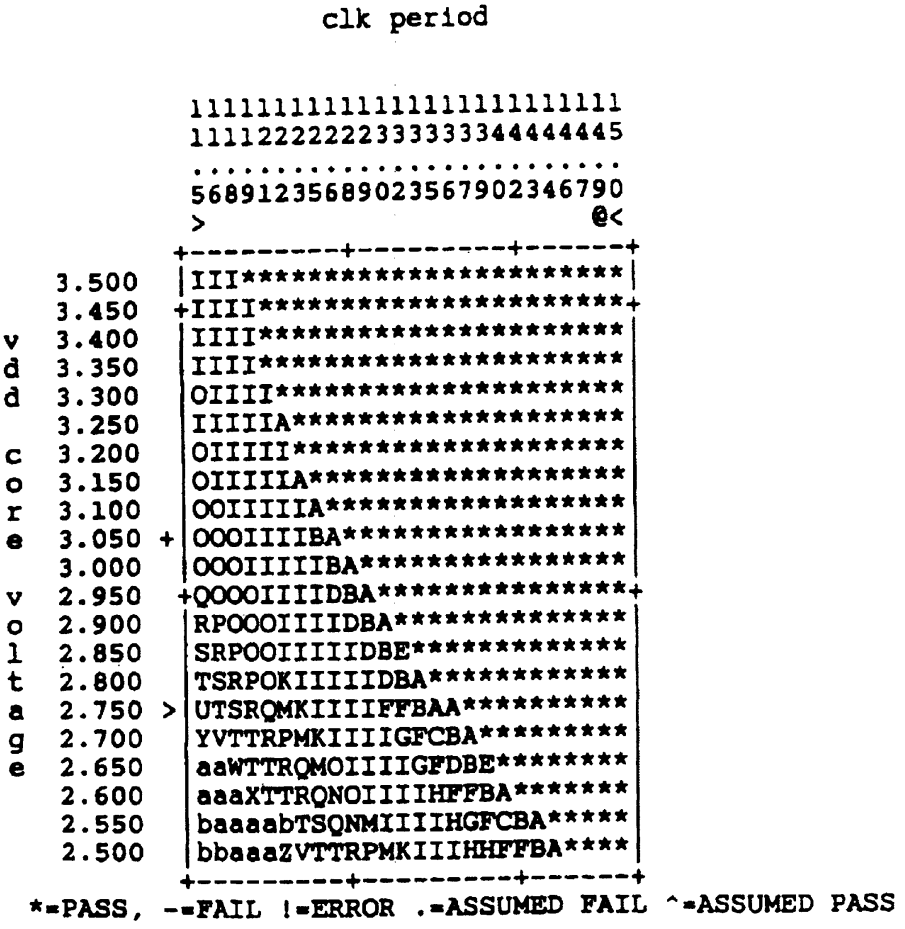

Fig. 22. Shmoo plot: Fmax versus power supply voltage at 3.5× clock multiplier.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

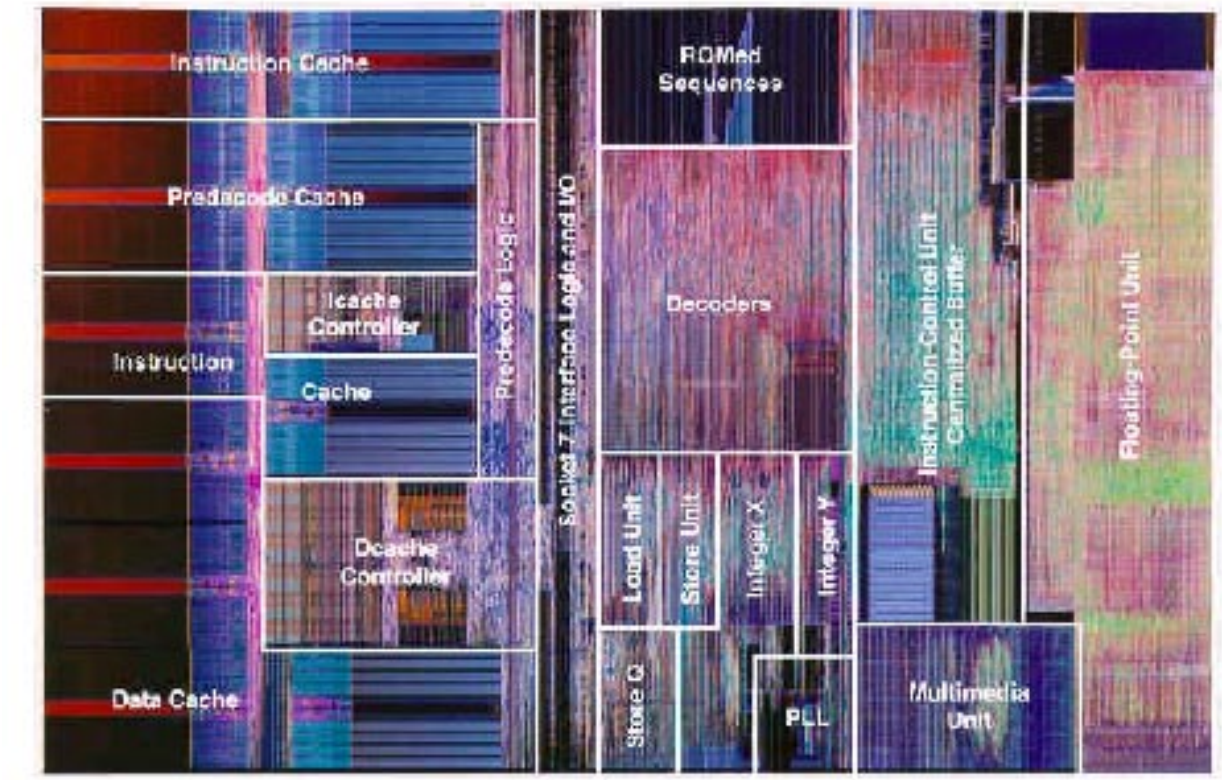

Fig. 23. Photomicrograph with functional overlay.

One of the key design concepts of the physical implementation was to delay into later mask layers some of the "programming" steps that had a reasonable probability of changing during design revisions. For example, the opcode ROM contents and the clock buffer strengths were both designed to be updated by only changing local interconnect and above, without requiring changes to the underlying base mask layers. Likewise, some options in the input and output cells were changeable in metal4 and above. These choices helped further the goal of a good balance between custom and ASIC approaches, and reduced time-to-market by allowing faster tapeout cycles.

By using the methods described in the TLM flow, a compact die size of 162 sq. mm was realized. A photomicrograph of the die, which contains 8.8 million transistors is shown in Fig. 23 with functional overlay describing the functions of the different die partitions. The processor achieves a density of 54 321 transistors/sq. mm compared to the current competitive, full-custom X86 processor [15] at 203 sq. mm and 7.5 million transistors for a density of 36946 transistors/sq. mm, where both chips are in 0.35-μm technologies.

XIII. Conclusion

High-speed circuit design techniques combined with an innovative microarchitecture and a high-speed process have enabled the AMD-K6 to be a very competitive, high performance processor which is plug-compatible with the existing socket 7 infrastructure and is binary compatible with the X86 legacy software base. The use of a standard cell methodology as well as a full-custom macro design approach yielded a compact, cost-effective die within the confines of a very tight schedule.

Acknowledgment

The authors are grateful for the contributions of U. Salim, B. Wong, J. Kumala, G. Frydel, A. Vuong, S. Yu, L. DiGregorio, E.-W. Tyan, Y.-R. Hwang, V. Nelson, L. Tsai, R. Domalpalli, R. Maley, B. Kauffmann, B. Burd, A. Scherer, F. Weber, M. Yamamura, M. Tamjidi, all the members of the K6 team, to J. Muleg for the Shmoo plot, and to T. Williams for his contributions to the manuscript.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

References

[1] D. Draper or al., "An X86 Microprocessor with Multi-Media Extensions," *SSCC Dig. Tech Papers*, 1997, pp. 172–173.

[2] T. Chappell et al., "A 2ns Cycle, 4 ns Access 512b CMOS ECL SEAMS," *ISSCC Dig. Tech. Papers*, 1991, pp. 50–51.

[3] R. Heald and J. Hoist, "6 ns Cycle 2S6kB Cache Memory and Memory Management Unit," *SSCC Dig. Tech Papers*, 1993, pp. 88–89.

[4] L. Lev et al, "A 64-bit Microprocessor with Multimedia Supports," *IEEE J. Solid-State Circuits*, Vol. 30, Nov. 1995, pp. 1227–1238.

[5] AMD-K6 MMX Processor Data Sheet, Mar. 1997.

[6] AMD-K6 MMX Processor Multimedia Extensions Manual, Mar. 1997.

[7] Intel Architecture MMXTM Technology Developer's Manual, Mar. 1996.

[8] H. Partovi at al., "Flow-Through Latch and Edge-Triggered Flip-Flop Hybrid Elements," *SSCC Dig. Tech. Papers*, 1996, pp. 138–139.

[9] D. Dobberpahl at al, "A 200MHz 64b Dual-Issue CMOS Microprocessor," *IEEE J. Solid-State Circuits*, Vol. 27, Nov. 1992, pp. 1555–1567.

[10] W. Bowhill at al., "A 300MHz 64b Quad-Issue CMOS Microprocessor," *ISSCC Dig. Tech. Papers*, 1995, pp. 182–183.

[11] J. Montanaro at al., "A 160MHz 32b 0.5W CMOS RISC Microprocessor," *ISSCC Dig. Tech. Papers*, 1996, pp. 214–215.

[12] B. Kim at al., "A 30-MHz Hybrid Analog/Digital Clock Recovery Circuit in 2-μm CMOS," *IEEE J. Solid-State Circuits*, Vol. 25, Dec. 1990, pp. 1385–1394.

[13] I. Young at al., "A PLL Clock Generator with 5 to 110 MHz of Lock Range for Microprocessors," *IEEE J. Solid-State Circuits*, Vol. 27, Nov. 1992, pp. 159–1607.

[14] M. Johnson at al., "A Variable Delay Line PLL for CPU-Coprocessor Synchronization," *IEEE J. Solid-State Circuits*, Vol. SC-23, Oct. 1988, pp. 1218–1223.

[15] M. Choudhury and J. Miller, "A 300 MHz CMOS Microprocessor with Multi-Media Technology," *ISSCC Dig. Tech. Papers*, 1997 pp. 170–171.

Don Draper (M'79) graduated from the University of British Columbia, Vancouver, with a Bachelors degree in engineering physics and from Carleton University, Ottawa, Canada with a Masters degree in electrical engineering.

He was employed by Bell Northem Research in Ottawa and American Microsystems in California before consulting to Siemens in Germany. Later he worked for Fairchild and Intergraph Advanced Processor Division on the Clipper Microprocessor. He joined NexGen in 1993 which was acquired by Advanced Micro Devices, Sunnyvale, CA, in 1996. He is K6 Circuit Design Manager and an AMD Fellow.

Matt Crowley (M'93) received the B.S. degree in electrical engineering from the University of Illinois, Urbana-Champaign. in 1990.

From 1990 to 1993 he worked as a Design Engineer at Amdahl Corporation, Fremont, CA, where he was responsible for board-level signal integrity as well as high performance package design for mainframe systems. In 1993 he joined NexGen/AMD, Milpitas, CA, as a Circuit Design Engineer. He is currently a Member of the Technical Staff with AMD, Sunnyvale, CA, where he is involved in both analog and digital CMOS design for high performance microprocessors.

John Holst (S'83–M'84) received the B.S. degree in electrical engineering from the University of Iowa, Iowa City, in 1984.

He was with the Memory Design Group at United Technologies MOSTEK, Carrollton, TX, as a Design Engineer of MOS memory circuits. In 1986, he joined UNISYS, Blue Bell, PA, where he designed CMOS processor circuits. He became a member of Intergraph Corporation's Advanced Processor Division in 1989, where he was part of the CLIPPER C400 processor development team. He is currently a Senior Member of the Technical Staff at AMD, Milpitas, CA, where he specializes in high-speed embedded memory design.

Greg Favor received the B.S. degree in computer science-electrical engineering from California Institute of Technology, Pasadena, in 1983.

He is a Senior Fellow at Advanced Micro Devices, Sunnyvale, CA, responsible for future X86 processor architecture development efforts. Prior to this, most recently, he was Chief Processor Architect and then Director of K6 Processor Development at NexGen Microsystems. Current technical interests include high performance microprocessor and system design.

Albrecht Schoy received the Diplom Ingenieur degree in 1976 from the Fachhochshule Furtwangen in Germany.

His current responsibilities at AMD, Sunnyvale, CA, include managing the design and validation on high-speed macros such as memories and I/O's for use on the K6 microprocessor. He started his career at the Robert Bosch GmbH, Stuttgart, Germany, designing microprocessors for automotive use in joint development with AMI in Santa Clara, CA. He subsequently worked in ASIC designs at Zymos and Dectroswiss in Switzerland where he was Design Manager specializing in mixed signal designs. Upon returning to the United States in 1989, he managed a number of teams in the field of ASIC and chipset design. Before joining AMD he was managing the memory compiler effort at VLSI.

Jeff Trull received the B.S. degree in electrical and computer engineering from Carnegie-Mellon University, Pittsburgh, in May 1989.

He worked on several generations of workstation processors at Hewlett-Packard in Ft. Collins, CO, in-

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

cluding the PA-8000. In October 1994 he joined NexGen (acquired by AMD), Milpitas, CA, where he was responsible for the implementation of the AMD-K6 Scheduler. His current interests include high-performance floating-point and software translation of program binaries between architectures.

Amos Ben-Meir received the B.S. degree from the Technion. Israel Institute of Technology, Haifa, Israel.

He worked on the NS32000 Microprocessor at National Semiconductor, Herriliya, Israel, and then on Sparc System Chipsets at Tera Microsystems, Santa Clara, CA. In 1992 he joined NexGen Inc. where he worked on a sixth-generation microprocessor which became the AMD-K6 after NexGen was acquired by AMD, Sunnyvale, CA, in 1996. His current interests include high-performance microprocessors, new circuit and logic design techniques, and process technology. He is now a K6 Logic Design Manager and an AMD Fellow.

Rajesh Khanna received the B.S.E.E. degree from the University of Cincinnati, OH, in 1991.

He then joined Digital Equipment Corp. where he worked on several commodity SRAM's and designed the Icache on the 21164 Alpha processor. In 1994 he joined Nexgen (now AMD), Milpitas, CA, and has been responsible for the implementation of the instruction decoder on the K6 microprocessor and has contributed extensively to the K6 design methodology. He enjoys challenging design work which requires a fusion of circuit design, logic design, and CAD.

Dennis Wendell (S'79-M'82) received the B.S.B.E. degree from the University of Nebraska, Lincoln, in 1979.

He joined Mostek Corporation in 1979 where he was involved with DRAM at the 64-kb and 256-kb technology generations. From 1984 to 1995, he worked on the design of CMOS and BiCMOS SRAM, post-charge, and delayed reset dynamic logic based on the SUN Ultrasparc microprocessor. Recently, he supervised the design and implementation of the I/D and predecode caches used for the AMD K6 686 microprocessor. He is a Senior Member of the Technical Staff at AMD, Milpitas, CA. He holds several patents.

Ravikrishna Cherukuri (Ravi Krishna) received the B.S.E.E. degree from the University of Roorkee, India, in 1986.

He designed multiprocessing systems for HCC-Hewlett Packard, India. He joined NexGen. Milpitas, CA, in 1990. His responsibilities include designing chipsets and multimedia enhancements in K6.

Joe Nolan (S'78–M'79)-received the B.S. degree in electrical engineering from the University of Florida, Gainesville, in 1978.

After graduation he designed bipolar and CMOS PROM's for Harris Semiconductor in Melbourne, FL. From 1983 until 1985 he was employed by Sprague Semiconductor, Willow Grove, PA, designing EPLD's. In 1985 he joined Sierra Semiconductor, San Jose, CA, where he designed EEPROM's, EEPLD's, microprocessors, analog-EEPROM-digital mixed signal ASIC's, and full-custom IC's. Later he became Sierra Semiconductor's Director of EEPROM designs. In 1989 he founded bASIC's Engineering, San Jose, CA, consulting in the design of analog integrated circuits, high-speed memories, EEPROM/FLASH-based PLD's and FPGA's, and microprocessors. Currently a consultant to Advanced Micro Devices, Milpitas, CA, he is involved in the design of high-performance microprocessors and EEPLD's. He is the author of several technical papers and patents.

Dhiraj Mallick received the B.S.E.E. degree, magna eum laude, from the University of Rochester, NY, in 1994.

He was with NexGen, Inc., Milpitas, CA, from 1994 to 1996, where he worked in the capacity of Circuit and Logic Design Engineer on fifth/sixth generation X86 microprocessors. Since January 1996, he has been with Advanced Micro Devices, Sunnyvale, CA, involved with the design of the AMD K6 microprocessor. Most notably he contributed to the aggressive design of the instruction decode and formulation of global tools and methodologies. He is currently a Module Lead on the K6 design team.

Hamid Partovi (M'89) received the B.S.EE. degree, magna cum laude, from the University of California, Berkeley, in 1981 and the M.S.E.E degree from the University of Michigan, Ann Arbor, in 1983.

He was with Advanced Micro Devices (AMD), Inc., Sunnyvale, CA, from 1983 to 1987 where be participated in the development of nonvolatile memories. He joined Digital Equipment Corporation, Hudson, MA, in 1987 where be was involved in the design of memories and microprocessors. Most notably, he designed the caches of a 100-MHz VAX processor and a 10-ns l-megabit commodity SRAM. After a year with Intergraph Corporation's Advanced Processor Division, Palo Alto, CA, he joined NexGen, Inc., Milpitas, CA (Later acquired by AMD) in 1994. Having participated in the design of NX586 and the K6 processors, he is currently the co-circuit design lead for the K7 microprocessor. He has authored or co-authored 11 technical papers; he holds 15 patents and has five pending.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.

Mark Roberts (M'93) received the B.S. and M.S. degrees in electrical engineering from the University of Michigan, Ann Arbor, in 1993 and 1994, respectively.

He is currently a Section Manager of microprocessor design at Advanced Micro Devices, Milpitas, CA. He led the integer execution unit design team on the K6 microprocessor and is currently the implementation leader of the K7 floating point execution unit.

Mr. Roberts is a National Science Foundation fellowship recipient. He is a member of Tau Beta Pi, Eta Kappa Nu, Phi Kappa Phi, and the ACM.

Mark Johnson (S'78-M'82) was born in Houston, TX, in 1957. He received the B.S.E.E. degree from Rice University, Houston, TX, in 1979 and the S. M. degree from the Massachusetts Institute of Technology, Cambridge, in 1982.

He joined Mostek Corporation in 1982, designing SRAM and DRAM products in both NMOS and CMOS technology. In 1986 he joined MIPS Computer Systems where he did circuit designs for floating point chips, phase-locked loops, and high-speed backplane buses. In 1991 he designed subnanosecond ECL FIFO circuits at MicroUnity before joining Rambus, Inc., in 1992. From 1992 to 1996 he designed high-speed mixed-signal circuits at Rambus, including reduced-swing I/O and PLL-DLL circuits in commodity DRAM technology. He worked as an independent consultant, designing PLL circuits and analog peripherals for microprocessors including the "K6" at AMD before joining Transmeta Corporation, Santa Clara, CA, in 1996. Currently he designs mixed-signal circuits, PLL's, and dynamic logic circuits for microprocessors.

Mr. Johnson is a member of Sigma Xi. Phi Beta Kappa, and Tau Beta Pi.

Thomas Lee (S'87-M'87) received the S.B., S.M., and Sc.D. degrees in electrical engineering, all from the Massachusetts Institute of Technology, Cambridge, in 1983, 1985, and 1990, respectively.

He joined Analog Devices in 1990 where he was primarily engaged in the design of high-speed clock recovery devices. In 1992 he joined Rambus Inc., Mountain View, CA, where he developed high-speed analog circuitry for 500 megabyte/s DRAM's. He has also contributed to the development of PLL's in the StrongARM, Alpha, and K6 microprocessors. Since 1994, he has been an Assistant Professor of Electrical Engineering at Stanford University, Stanford, CA, where his research interests are in low-power, high-speed analog circuits and systems, with a focus on gigahertz-speed wireless integrated circuits built in conventional silicon technologies, particularly CMOS.

Dr. Lee has twice received the "Best Paper" award at the International Solid-State Circuits Conference.

Originally published in *IEEE J. Solid-State Circuits*, Nov. 1997, pp. 1650–1664.