MICROPROCESSOR REPORT

THE INSIDERS' GUIDE TO MICROPROCESSOR HARDWARE

VOLUME 8 NUMBER 14 OCTOBER 24, 1994

# AMD's K5 Designed to Outrun Pentium

## Four-Issue Out-of-Order Processor Is First Member of K86 Family

**by Michael Slater**

At the recent Microprocessor Forum, AMD unveiled its challenger to Pentium, setting the stage for AMD—along with Cyrix, IBM, and NexGen—to challenge Intel's domination of the high-end x86 microprocessor market.

The chip is the first in a new line from AMD that is based on an entirely AMD-developed core. By developing its own microarchitecture, AMD hopes to eliminate legal squabbles and gain a competitive edge. The first chip in the K86 family carries the project name of K5; the formal product name has not been released.

At the heart of the chip is an advanced four-issue superscalar core that supports speculative, out-of-order execution and register renaming *(see* ***081102.PDF****)*. The design goes beyond Intel's Pentium and Cyrix's M1, using fully decoupled instruction dispatch and execution in an effort to deliver more effective superscalar operation.

AMD was due to tape out the design as we go to press, so first silicon won't be seen until November. The chip is a static, 3.3-V design that is implemented in AMD's 0.5-micron, three-layer-metal CMOS process (the same as used for AMD's DX4-100). Die size for the chip, which uses 4.3 million transistors as compared with Pentium's 3.3 million, has not been disclosed.

AMD plans to deliver samples to customers by the end of the year, with production by the middle of 1995. Initial chips will come from AMD's Submicron Development Center, which has been equipped as a production facility, with volume production from the company's new Fab 25, now nearing completion in Austin. AMD plans to shift the K5 to a 0.35-micron process in 1996.

### 30% Faster at Same Clock Rate

AMD's simulations show that, at the same clock rate, the K5 should be at least 30% faster than Pentium (on integer code) and 2.5 times as fast as a 486. AMD has put less emphasis on floating-point performance but still expects the K5 to be roughly comparable to Pentium.

The K5 is a much more flexible, more aggressive microarchitecture than Pentium, so it is not surprising that it would achieve higher performance at the same clock rate. AMD expects to match Intel's current top rate of 100 MHz, but Intel probably will have higher-clock-rate Pentiums by the time the K5 is in volume production. It remains to be seen whether AMD will actually ship higher-performance processors than Intel at any point in time.

At the conference, Mike Johnson, AMD's director of microprocessor architecture, displayed pipeline simulations of Pentium and the K5 running actual traces from Microsoft Word, which showed the K5 to be more than 30% faster than Pentium on a per-clock-cycle basis. On selected code sections, the K5 is as much as three times as fast as Pentium.

### Tackling the x86 Bottleneck

Decoding multiple x86 instructions in parallel is challenging. RISC instructions have a fixed length, making it easy to decode as many of them in parallel as desired. For x86 instructions, on the other hand, the variable length means that the next instruction can't be decoded until the length of the previous instruction is known. Pentium proved that this challenge could be overcome, but it decodes only two instructions at a time.

AMD's architects avoided this problem by predecoding x86 instructions as they are fetched from memory and fed to the instruction cache, as Figure 1 shows. Since most instructions fetched come from the cache, the predecode information frees superscalar instruction dispatch logic from having to deal with the variable-length aspects of the x86 instruction set.

Since an average x86 instruction is about three bytes long, it takes the K5 an average of nearly three clock cycles to predecode the eight instruction bytes fetched in a single bus transaction. If the bus clock were the same as the processor clock, the predecoder wouldn't be able to keep up, but the fact that the CPU core runs at a multiple of the bus clock gives the decoder extra time. In a worst-case situation (short instructions, a CPU core running at only 1.5 times the bus clock, and a single-cycle burst rate from memory), the serial instruction decoder could limit performance during a cache miss, but this would have little impact, since the vast majority of instruction fetches are cache hits.

When instructions are written into the instruction cache, the predecoder adds five bits to each byte. These bits indicate whether the byte is the start or end of an x86 instruction; the number of microinstructions required to implement the x86 instruction; and the location of opcodes and prefixes.

The predecode bits increase the size of the instruction cache array by about 50% (the cache data array is increased by 5/8 but tags and prediction bits aren't increased). In return, they dramatically reduce the complexity of the parallel dispatch logic following the cache.

## Dispatching Four Instructions Per Cycle

After instructions are fetched from the cache, the K5 converts each instruction to one or more microinstructions, which AMD calls RISC operations or ROPs (pronounced "ar-ops" by Johnson). The x86 instructions are pulled from the instruction cache 16 bytes (plus predecode bits) at a time and converted to ROPs. Up to four ROPs can be issued per cycle.

The ROPs aren't quite like conventional RISC instructions, but they share two important characteristics: fixed length and simple, consistent encodings. The ROPs are essentially the same as microcode, except that the majority of them are generated directly by hardware decoders rather than fetched from ROM. Pentium also decodes simple x86 instructions directly into microinstruction sequences, but the K5's ROPs have more of a life of their own; they are not necessarily executed right away or even in order.

Unlike more limited superscalar machines, such as Pentium, there are no instruction grouping requirements for multiple issue. Even x86 instruction boundaries do not limit ROP dispatch. The K5 thereby avoids the need for specific compiler optimizations; 486-optimized code will run well, and Pentium-optimized code will run better (though many Pentium optimizations are unnecessary for the K5).

Figure 2 shows the instruction translation process in more detail. Instructions from the cache are fed into a 16-byte queue. The fetch logic tries to keep this queue filled, speculatively following branches as needed. As instructions are consumed from this queue, new instructions are added from the cache.

Up to four instructions can be pulled from the byte queue during each clock cycle. Because these instructions are already tagged with predecode bits indicating where instructions start and end and how many ROPs

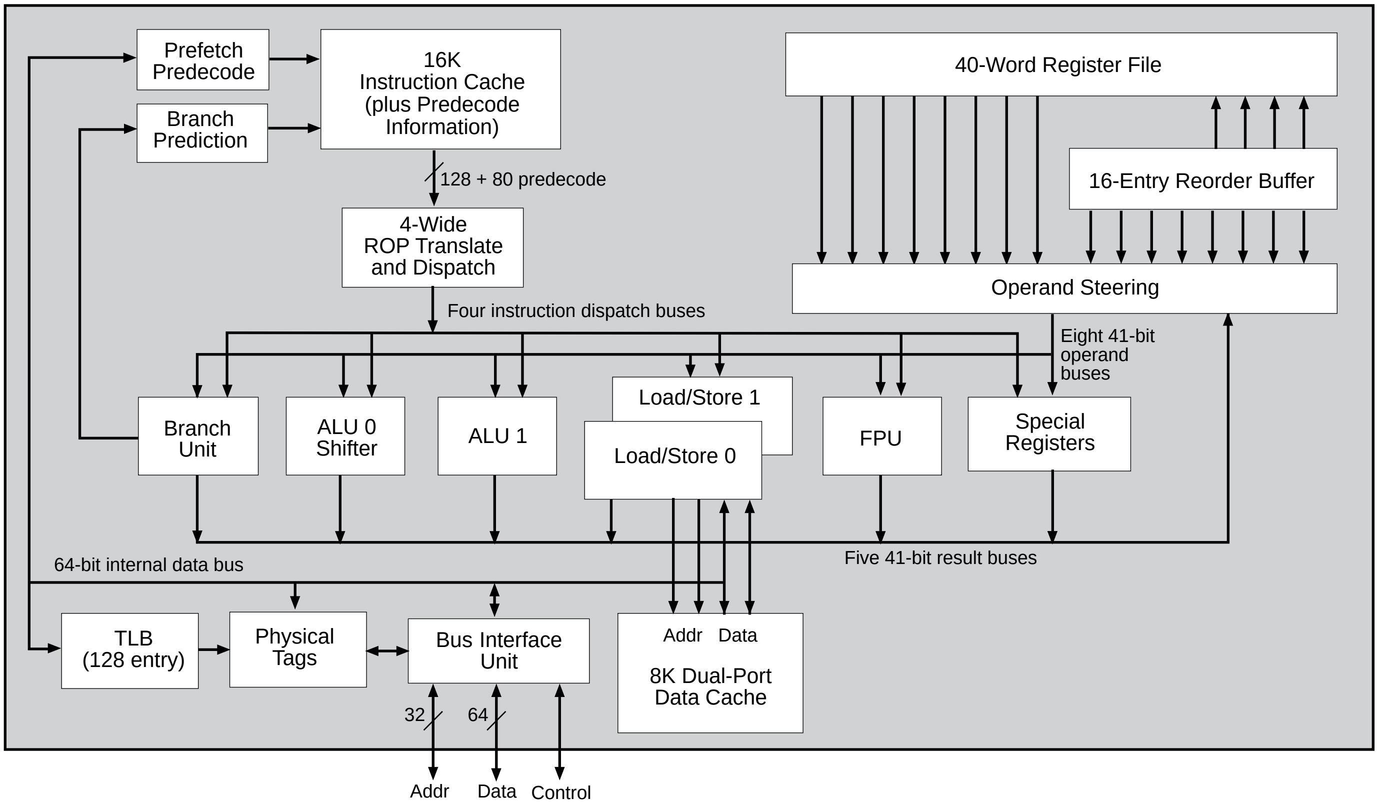

Figure 1. Block diagram of AMD's K5, which adds predecode bits to x86 instructions before caching them.

each needs, it is a relatively simple task for the byte queue to locate instruction boundaries and find four ROPs worth of instructions.

All four ROP converters are identical and can handle any instruction. The ROP converters translate most x86 instructions directly into ROPs, breaking the complex instructions into multiple ROPs and rearranging instruction fields for consistency.

In this example, the parse/duplicate logic sends the first instruction, which requires two ROPs, to the first two ROP converters, along with an indication of where in the ROP sequence each converter lies. The two ROPs produced by the add memory-to-register instruction are:

- Load temporary register from memory
- Add temporary register to target register

The second instruction, compare register to immediate value, translates into a single ROP. The final instruction, a register push, translates into two ROPs: one to decrement the stack pointer and a second to perform the store to memory.

Any instruction that can be executed with one, two, or three ROPs is handled entirely in hardware. For complex instructions, such as string move, that require four or more ROPs, ROP sequences (which are essentially the same as microcode) from the MROM are used. The MROM can dispatch four ROPs per cycle.

Once past the ROP converters, the K5 core is RISC-like in that it does not have to deal with variable-length instructions and memory-based operands. It does, however, have numerous special features to support the vagaries of the x86 instruction set—most notably, dual load/store units with full support for complex x86 addressing modes.

The limit of dispatching four instructions per cycle is based on ROPs, not x86 instructions. If each x86 instruction in the group requires only one ROP, then four x86 instructions can be dispatched at once. In the example shown in Figure 2, the first instruction requires two ROPs, the second requires one, and the third requires two. The processor will dispatch, in a single cycle, the first four of these ROPs—that is, all the ROPs for the first two instructions and the first of two ROPs for the third instruction.

On average, 16-bit x86 code produces 1.9 ROPs per instruction. Because 32-bit x86 code tends toward a simpler subset of the instruction set, it produces only 1.3 ROPs per instruction. Thus, in terms of x86 instructions dispatched per cycle, the peak rate for an average instruction mix is about two instructions for 16-bit code and three instructions for 32-bit code.

## Execution Resources Include Dual ALUs

Instructions are dispatched from the byte queue in order, without regard to the availability of the operands and execution resources required for the instructions. The ROPs are dispatched to the execution units, where they wait in reservation stations for the execution unit and the needed operands to be available. Each unit except the FPU has two reservation stations; the FPU has only one. The dispatch process stalls as soon as any ROP is blocked from being dispatched because no reservation station is available.

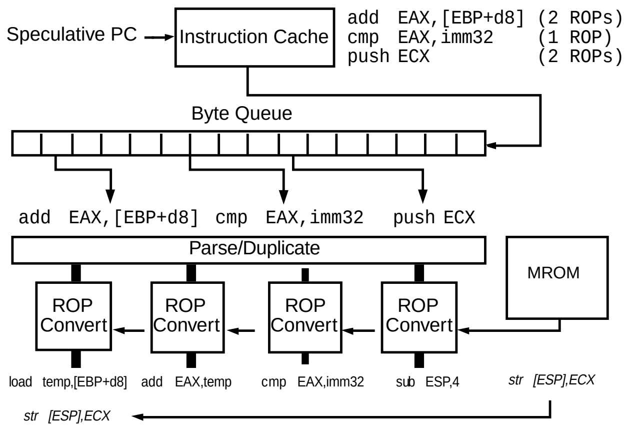

Figure 2. Instructions are issued from a queue (filled from the cache) into four parallel ROP instruction decoders.

The K5's six execution units are two ALUs, one FPU, two load/store units, and a branch unit. Only one ALU has a shifter, and only the other has a divider; otherwise they are identical. The floating-point unit does not include a hardware register stack, as in traditional x86 designs; the stack is emulated in the general register file with special register-renaming logic.

One area where the K5 has saved a little silicon and is therefore slower than Pentium is in floating-point multiplication, which has a latency of seven cycles (worst case, with a four-cycle issue rate) versus three for Pentium. Like Pentium, the K5 supports the parallel execution of the floating-point exchange (FXCH) instruction along with another floating-point operation—an important optimization aimed at mitigating the performance handicap of the stack architecture.

Eight operand buses feed the execution units, allowing four units to be fed two operands each on every cycle, thereby supporting the peak issue rate. There are five result buses. Each bus is 41 bits wide to support transfers of floating-point data; two buses are used in parallel to provide the 82-bit width needed for x86-compatible floating-point operations.

The register file holds 40 words, much bigger than the basic x86 register set. This is because the register file must also store the floating-point stack and temporary registers used for passing data between ROPs.

A 16-entry reorder buffer (ROB) stores results from instructions that have been speculatively executed (see sidebar below). All results are first written to the ROB;

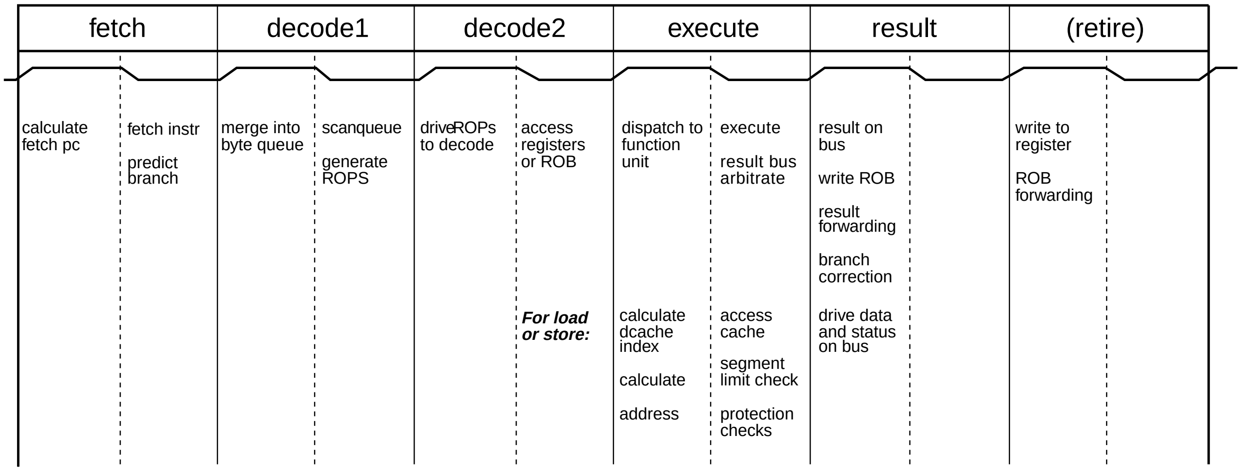

Figure 3. The K5's pipeline has six stages, but only five that affect instruction timing. It uses an extra decode stage for the byte queue but packs address generation into the execute stage.

writes to the register file are performed by the ROB when the result is known to no longer be speculative. Operands may be fetched from either the reorder buffer or the register file; each has eight ports, enough to service four two-operand instructions in every cycle.

## Pipeline Has Few Penalties

Figure 3 shows the pipeline timing. There are six pipeline stages, but it is effectively a five-stage pipeline because the final stage does not affect performance. The final stage, due to the reorder buffer, adds a cycle of latency before results are written to the register file, but it does not affect processor performance, since results are forwarded to any waiting execution units as soon as they are available from the ROB or a result bus.

Compared with a traditional x86 implementation, the K5 requires an extra decode stage for the x86-to-ROP translation. The K5 compensates for the added latency by combining address generation into the same stage as cache access; all other pipelined x86s use a separate stage for address generation.

The K5's load/store units are especially complex because they perform full x86 address calculations in addition to the data cache access, all in one clock cycle. In the first phase of the execute stage, the cache index (the least-significant 11 bits of the linear address) is calculated. In the second phase, the cache is accessed. The full 32-bit linear address calculation isn't complete until late in the second phase, just in time to compare it to the tag returned from the cache access. All the segment and protection checking is also done during the second phase, and if all goes well, the data is available at the start of the next clock cycle. As a result, there is no load-use penalty; data can be loaded by one instruction and used by the next without stalling.

Branches are predicted using a simple single-bit algorithm that is cache-line based: the branch history bit reflects the previous direction taken by the branch, except that a backwards branch that was predicted to be taken but is later not taken doesn't change the prediction bit. The K5 has one prediction entry per cache line, implemented as part of the cache array rather than a separate branch prediction cache.

Including the prediction bits in the cache array allows the chip to store predictions for 1,024 different cache lines—four times the size of Pentium's branch prediction cache. AMD believes that this approach is just as effective as the two-bit algorithm used in Pentium and recent superscalar RISCs; it has lower prediction accuracy per branch but stores prediction information for many more branches.

The prediction entry for each cache line includes a pointer to the target instruction, with its cache index and byte offset. This pointer enables the processor to follow a taken branch without a pipeline bubble. The penalty for a mispredicted branch is a minimum of three cycles, or 12 potential ROP instruction slots.

## Caches and Memory Management

The dual load/store units allow two accesses to the 8K data cache to be performed in a single clock cycle, provided that no two accesses are to the same bank. Dual load/store units are included because of the high incidence of loads and stores in the ROP stream, thanks to the paucity of registers in the x86 architecture.

The cache is divided into four banks. There are two access ports, one for each load/store unit, and both accesses proceed in parallel as long as they are to different banks. (Pentium uses a similar dual-access scheme with eight banks; Johnson says that AMD's simulations showed little benefit for eight banks instead of four.) Two accesses to the same cache bank are allowed in the same

## K86 Roots and Relatives

Although active work on the K86 processors at AMD did not begin until nearly three years ago, their roots go back to at least 1989, when Mike Johnson published his Ph.D. dissertation, *Super-Scalar Processor Design* (Stanford University). This work grew into the first book on the subject, published in 1991 by Prentice-Hall as *Superscalar Microprocessor Design.*

Johnson's book even includes an appendix titled "A Superscalar 386," which includes comments on why it would be "extremely painful to implement a superscalar version" of the architecture. Many seeds of the K5 design, including the use of a reorder buffer and the possibility of using predecode bits in the instruction cache, can be seen in this appendix.

Johnson joined AMD not to work on a superscalar x86 but to create the 29000 family. In parallel with the K5 effort, he also led the design of the first superscalar 29000 implementation *(see* ***081404.PDF****)*. While the two chips are very different, because of the differing demands of their instruction sets, some logic blocks—such as most of the reorder buffer and the ALUs—were shared between the two designs. The K5's FPU was borrowed from the 29050 and extended to 80 bits.

cycle if both are to the same cache line.

The instruction cache has a 16-byte line size, half the 32-byte line size used by Pentium. Since the Pentium bus, with which the K5 is compatible, performs cache fills in 32-byte bursts, the K5 includes a 16-byte buffer to hold the second cache line. Only if this line is subsequently requested (before another cache fill occurs) is it loaded into the cache. The smaller line size is better for the K5's branch prediction, which is performed on a cache-line basis, and it also yields a higher hit rate.

The caches are virtually addressed and virtually tagged to avoid the need to translate addresses before a cache access. In addition, a single set of physical tags is shared by both the instruction and data caches. When any changes are made to the virtual-to-physical mapping, the virtual cache tags are invalidated. To avoid the performance degradation usually associated with virtual caches, the physical tags continue to be checked on subsequent accesses, and if a match is found, the cache line is revalidated without reloading it from memory.

The physical tags are used for bus snooping, eliminating any conflicts with the CPU for cache access. They also serve to ensure consistency between the instruction and data caches. If a write occurs to a location that is in the instruction cache, that cache line is invalidated.

The bus interface is Pentium-compatible, using the P54C pinout but without the APIC functions (i.e., it uses the P5 signal set with the P54C pin arrangement).

## Reorder Buffer Operation

The reorder buffer used in AMD's K5 and its superscalar 29000 implements register renaming, facilitates branch prediction and precise exceptions, and serves as a central clearinghouse for the register values used and produced by instructions.

When ROPs are dispatched to an execution unit, an entry at the top of the reorder buffer is allocated for each instruction. Up to four entries are allocated simultaneously. Each entry keeps track of the program counter associated with its instruction and has a place to hold the result of the instruction (if it produces one). The reorder buffer acts as a true FIFO: as entries are deallocated from the bottom of the buffer, all other entries slide down, making room for the next group of instructions to be issued.

Up to four entries are deallocated from the bottom of the buffer each cycle, provided that their results are available; the branch prediction that led to their execution is validated as correct; and no exceptions were signalled along with the execution of the instructions (e.g., a page fault with a load or store). If the instruction associated with such an entry produced a result, the result is written to the register file.

The ROB helps branch prediction because results are not written to the register file unless they are guaranteed correct. If a branch is mispredicted, the results of instructions along the mispredicted path can simply be invalidated in the buffer.

The ROB naturally implements register renaming with hardware to implement associative lookup. Since each instruction issued has an entry with a place to hold a result, a unique storage location is available for every result. To implement register renaming, though, requires that the buffer function as an associative memory when presented with a register number. Given the number for a source register of an instruction that is being issued, the reorder buffer must find the entry that has the value associated with that register.

Further, it is possible that the ROB holds the results of two or more instructions that name the same result register; thus, when a new instruction is issued, the buffer must find the entry with the most up-to-date value for the instruction's source registers. If a register value is not yet available, the buffer must allocate a unique tag instead of delivering the value. When the required value is later available, tag-comparison logic forwards the value directly to the waiting execution unit.

The associative lookup requires a set of comparators and a priority encoder for each read port in the buffer. Since read ports, comparators, and priority encoders all require lots of metal routing, the die area consumed by the reorder buffer can quickly get out of hand. This is one reason that reorder buffers are kept as small as possible. The K5's ROB is more complex than the superscalar 29K's because it must support 8- and 16-bit writes to fields in registers.

—BC

## Design Technology

AMD is no stranger to the x86 microprocessor business, having produced (by its own accounting) 28 million of them in the past three years. All of them, however, are virtual duplicates of Intel's logic designs, so this record does not speak to AMD's ability to independently engineer a compatible processor. To keep the K5 design pure, AMD says that no one associated with the Intel-derived designs was involved. AMD borrowed heavily from the microarchitecture work that was done for the superscalar 29K, which was designed earlier, and much of the design team had 29K experience. Two things that were used from AMD's x86 experience are the compatibility test suites and validation methods that had been developed to test the clean-room 486 microcode.

To validate the design running real software, a Quickturn-based hardware emulator was used. Running on the hardware emulator, the K5 booted DOS in July and Windows in August. AMD believes that its investment in hardware emulation, which turned up about two dozen subtle bugs, will pay off in chips that have few problems in the initial silicon. Indeed, AMD's aggressive schedule requires sampling from first silicon.

AMD still could encounter stumbling blocks in making its design fully x86-compatible. Cyrix's and NexGen's examples, however, provide some assurance that compatibility is achievable. Any gaps in the chip's compatibility could be very damaging to AMD if they are not found early and corrected quickly.

CLARENCE TOWERS

**"The complexity of the x86 is not an impassable barrier. The x86 really isn't all that complex—it just doesn't make a lot of sense.... The biggest weakness in the x86 instruction set is the lack of registers coupled with an extremely painful addressing scheme."**

Mike Johnson, AMD

## Comparing Superscalar x86s

Although Intel likes to characterize all x86-compatible processors as imitators, the K5 is far from an imitator of Pentium. It executes the same instruction set—essentially that defined by the 386—and conforms to the same pinout and bus interfaces, but inside it is a radically different machine. Although nothing can guarantee that Intel won't deem its intellectual property to be infringed and file suit, there are no apparent grounds for doing so; AMD's independent design should avoid any copyright issues, and Intel does not dispute AMD's patent license.

The K5 goes further than any previously described design in combining x86 compatibility with a RISC-like core, achieving the best of both: a large software base and high performance. The overall style of the microarchitecture is most similar to that used by NexGen's Nx586, in which x86 instructions are decoded only one at a time but are translated into RISC-like operations that are executed in parallel *(see **080403.PDF**)*. Unlike the K5, however, the Nx586 does not cache predecode information, so it is limited to dispatching the ROPs (RISC86 instructions, in NexGen's lingo) for one x86 instruction at a time. Whereas AMD expects to exceed Pentium performance at the same clock rate by about 30%, NexGen's advantage is less than 10%.

The other Pentium competitor expected to debut soon is Cyrix's M1, which will be second-sourced by IBM Microelectronics. Although the design was described at last year's Microprocessor Forum *(see **071401.PDF**)* and shipments were promised for 1994, the chip has not formally debuted. Some press reports have put the M1 as much as six months ahead of AMD's K5, but indications are that both designs will see first silicon next month. Cyrix's delivery schedule appears to be based on a more aggressive plan for moving the chip quickly into production. Only time will tell who will get to market first with significant quantities of debugged chips.

Like AMD's K5, Cyrix's M1 is more advanced than Pentium in its use of register renaming and out-of-order execution. The Cyrix design appears more like Intel's in its use of dual pipelines rather than AMD's decoupled pool of execution resources; it is therefore more limited in its ability to execute instructions out of order. Cyrix uses deep, seven-stage pipelines. Until actual benchmark results are available, it will be impossible to judge the effectiveness of the strategies AMD and Cyrix have chosen, but AMD's design appears to be more capable.

Intel's P6 processor is likely to have a great deal in common with the K5. Because Pentium was developed years earlier, it had to be buildable in 0.8-micron technology, so Intel was more limited in what it could do. The P6 design, in addition to using techniques such as out-of-order execution and register renaming, is expected to boost performance by use of a proprietary second-level cache chip, connected to the CPU chip within a single IC package. Having a higher-bandwidth external cache should give the P6 a performance edge but will also make it more expensive to build.

## Competing with Pentium

AMD has created, on paper at least, a formidable challenger to Pentium. Taking advantage of coming to market two years later, AMD aimed for a more aggressive design point, enabling its chip to deliver higher performance at the same clock rate. If AMD is able to deliver on its promises in a timely manner, it should be a significant force in the Pentium-class CPU market by the end of 1995. Although the K5's die size is surely larger than Pentium's, the high margins in this market make modest cost differences relatively insignificant.

In the four years since the company began shipping 386 microprocessors, AMD has established business relationships with dozens of PC vendors, becoming the leading alternative to Intel. The credibility the company has established in this process, culminating in its partnership with Compaq, has created a fertile environment for the K5. If Compaq uses the K5, PC consumers and other PC makers will see it as a stamp of approval—making it much less intimidating for them to follow in Compaq's footsteps and leave the Intel fold.

With the 486, AMD's market share—which the company estimates at 13%—has been limited by its fab capacity. Only one small fab (the Submicron Development Center) is currently running AMD's most advanced processes, needed for 486 and K5 chips. By the middle of 1995, however, AMD expects to be in full production at its new Fab 25. Intel has several comparable fabs, giving it much greater capacity, but AMD's goal is not to overtake Intel as the market leader—just to be the leading alternative, with 25–30% of the market. A single large fab, plus the existing SDC and foundries such as Digital, should enable AMD to achieve this market share if it has the right products.

Marketing the K5 (or M1) against Pentium is an entirely different challenge than creating such a device. Ideally, AMD would like to be able to sell its x86 processors at similar prices to Intel's for comparable performance levels. Being the underdog, however, has pushed AMD to offer higher performance for the same price. When the 386 still had some life in it, AMD took over the market with 40-MHz chips at the price of Intel's top-of-the-line 33-MHz devices. With the 486, AMD didn't have a clock speed advantage for nearly two years. Now AMD is shipping 80-MHz 486 chips and sampling 100-MHz parts, catching up with Intel's DX4.

With the K5, AMD expects to match today's top Pentium clock rate of 100 MHz. The company has not announced any pricing or clock rates for the K5. Its natural pricing strategy, however, will be to price the K5 identically to Pentium at the same clock rate, offering higher performance as the incentive for buying from AMD.

Just how fast the chip will run has yet to be proved, however; the achilles' heel of an ambitious superscalar design is that, if not superbly executed, the increased efficiency in instructions per clock cycle can be negated by a reduced clock rate. AMD is confident in its simulations, but other vendors with complex superscalar designs have sometimes been unpleasantly surprised.

Intel won't stand still, either. It is unclear just when Intel will increase Pentium's clock rate, but a 120-MHz speed grade is likely early next year, and a 150-MHz version (P55C), using a new 0.4-micron process, has been promised for late 1995. AMD could market a 100-MHz K5 against a 120-MHz Pentium, saying that the K5 offers higher performance despite a lower clock rate, but this may be a hard sell with consumers. Competitive situations such as this one highlight the need for a standard benchmark for x86 processors, replacing clock rate as a measure of speed *(see **0814ED.PDF**)*.

How AMD fares against Cyrix's M1, with IBM behind it, remains to be seen. If both parts meet their expectations and are competitively priced, they should do well. Whether one or the other becomes dominant depends on the quality of each vendor's design and the strength of its relationships—factors that won't be tested until next year.

AMD has many challenges ahead. But if the company is able to produce K5 chips in the middle of next year, if the chips are significantly faster than 100-MHz Pentiums, and if there are no compatibility problems, then AMD will be well-positioned to keep increasing its x86 market share. ♦

## Price & Availability

AMD has not formally announced any chips in the K86 family, so no pricing information is available. AMD is promising samples by the end of this year and production in mid-1995.

Call AMD at 800.222.9323; fax 512.602.7639.