NexGen Enters Market with 66-MHz Nx586

First Pentium Competitor Uses RISC-like Core and Optional FPU

by Linley Gwennap

Less than a year after Intel unveiled Pentium, the first Pentium-class competitor has emerged from an unlikely source: startup NexGen, which has labored for eight years to create its Nx586 processor. Although the chip is in no way a clone of Pentium, the company says that its product is fully compatible with the x86 instruction set and performs similarly to the Intel processor on a variety of benchmarks. NexGen will market 60- and 66-MHz versions of its new CPU, undercutting Intel's prices for 60- and 66-MHz Pentiums.

NexGen's CPU uses an innovative combination of a superscalar RISC-like core with an x86 instruction decoder. Each cycle, the decoder takes one x86 instruction and translates it into one or more "RISC86" instructions, which are then executed in parallel by multiple execution units. The RISC86 instructions can be executed speculatively and out of order to increase the potential for parallelism. The Nx586 uses register renaming to reduce the bottleneck of the small x86 register set.

NexGen originally planned to build high-performance multiprocessor systems but abandoned this effort a few years ago to focus on completing and marketing its processor chip set. Without shipping a product, the company has raised $90 million from a long list of backers including ASCII Corp., Compaq, Olivetti, and noted venture-capital firm Kleiner, Perkins, Caufield, and Byers.

The fabless company is currently delivering samples of its Nx586 built by IBM in its four-layer-metal 0.5-micron CMOS process, the same process used for the PowerPC 603. In preliminary press briefings, NexGen said that IBM had agreed to put the Nx586 into volume production, an arrangement that seemed to be a great coup for NexGen, given IBM's world-class production capability and its Intel patent license. But just two days before its public announcement, NexGen notified the press that the IBM deal was off; the company did not explain its previous misleading statements.

Negotiations between the two companies continue, and if an agreement is made soon, NexGen expects to put its chips into full production in the second quarter. If the deal falls apart, however, it would be a major blow to the company's credibility, legal strategy, and its ability to deliver its parts in volume.

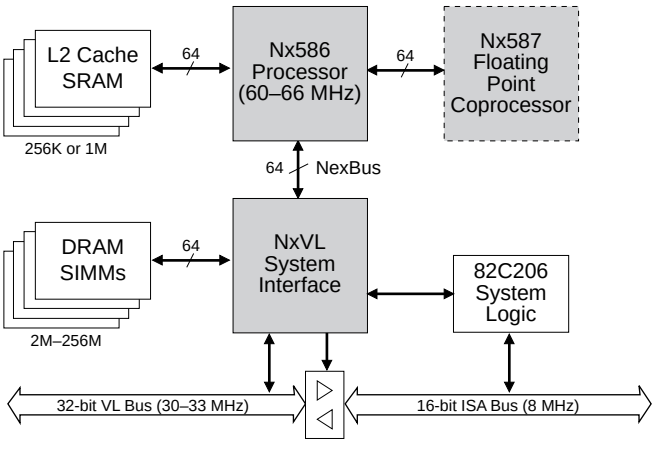

Figure 1. The NexGen 586 includes a cache interface but places the FPU on a separate chip. NexGen supplies a system-logic chip that connects to DRAM and to standard PC buses.

Unusual System Partitioning

NexGen has partitioned its chip set differently than a 486 or Pentium system. Like Intel, NexGen could not fit a complex x86 CPU, FPU, and 32K cache on a moderate-size die. Intel chose to include only 16K of cache on Pentium; NexGen abandoned the on-chip FPU instead.

Figure 1 shows that the Nx586 includes the CPU, while the optional floating-point unit (FPU) is relegated to a separate chip, much like the old 387. Instead of the FPU, NexGen has integrated a level-two (L2) cache controller on the CPU chip. The Nx586 includes separate 64-bit interfaces for the FPU and the external SRAMs, plus a third bus that connects to the rest of the system.

NexGen's partitioning offers several advantages over the traditional Intel arrangement of a combined CPU/FPU chip with a separate cache controller. The optional FPU lowers the cost of entry-level systems; since most programs make little (if any) use of floating-point math, these FPU-less systems will be adequate for most users. The dedicated cache bus eliminates bus conflicts with memory and I/O traffic and, in future versions, will allow the cache bus to run at the CPU frequency while the system bus stays at a more reasonable speed.

One disadvantage to this approach is that the three separate interfaces greatly increase the package pin count (and thus cost) compared with Pentium. NexGen's Nx586 uses a 463-pin PGA package that costs about twice as much as Pentium's 296-pin PGA.

Instead of a standard 486 or Pentium bus, NexGen uses its own NexBus to connect to the system, preventing system designers from taking advantage of standard system logic and existing motherboard designs. NexGen is the only vendor providing system logic for NexBus, and it currently offers a single product, which connects

to main memory, ISA, and VL-Bus, as Figure 1 shows. The company plans to deliver a second version, with a PCI interface instead of VL-Bus, later this year. Both chips require an external 82C206 for standard system logic such as interrupts and timers.

x86 Translated into RISC86

NexGen takes the unusual approach of translating x86 instructions into a proprietary instruction set called RISC86. The company would not release details of this instruction set. It said that RISC86 uses operations that are similar to standard RISC instructions but are modified to accommodate the idiosyncrasies of the x86 architecture. Although NexGen uses the "RISC" label, these instructions appear similar to microcode.

NexGen would not reveal the RISC86 instruction encodings or width. The company's original eight-chip implementation, which never shipped, used 104-bit instructions (see MPR 11/7/90, p. 6); NexGen says that its current design is similar to this original design but repartitioned into two chips. Unlike true RISC instructions, NexGen's internal instructions are mostly decoded to eliminate the need for a decoder in each function unit. Because these instructions are never stored in memory or transferred to other chips, their large size does not cause significant problems.

One of the bottlenecks of the x86 architecture is its limited register set: only eight general-purpose registers, compared with 32 for a typical RISC processor. The translation process provides an easy way to reduce this problem. The RISC86 instruction encodings support up to 32 registers; the Nx586 implements 22 physical registers. The x86 decoder maps the eight x86 registers onto these physical registers. This process, called register renaming, can eliminate many of the register conflicts common in x86 programs and is also used in Cyrix's M1 design *(see **071401.PDF**)*.

Many x86 instructions convert directly to a single RISC86 instruction. All register-to-register ALU operations, for example, have RISC86 counterparts. Because RISC86 uses a load-store model, however, many x86 instructions that access memory require two or three RISC86 instructions. For example, the x86 instruction:

```
ADD [mem], CX
```

translates into three RISC86 instructions:

```
LOAD R2, [R1]
ADD R2, R3
STORE [R1], R2
```

where the physical register numbers are assigned by the decoder to map the x86 registers appropriately.

Iterative x86 string instructions translate into an indefinitely long sequence of internal instructions. The decoder issues RISC86 instructions as fast as possible; when the core detects the termination condition, the remaining iterations are invalidated.

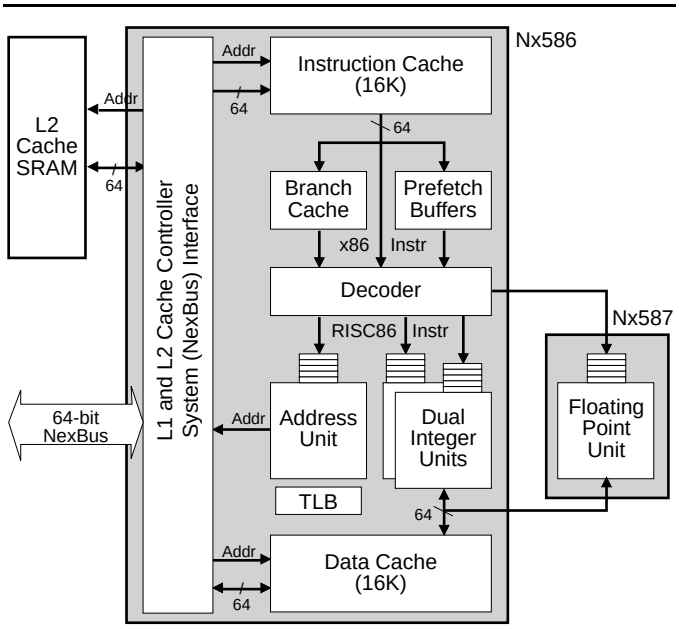

Figure 2. The decoder translates x86 instructions into NexGen's RISC86 instructions, which are then executed by the superscalar processor core and the off-chip floating-point unit.

The Nx586 uses an instruction prefetch buffer to solve the problems of variable length and alignment inherent in x86 code. Unlike Pentium's cache, which contains special logic to fetch up to 31 consecutive unaligned bytes, the Nx586 instruction cache delivers instructions in groups of eight aligned bytes. The prefetch buffer holds up to three groups of 24 bytes each, prefetching along the sequential path and two predicted paths. On each cycle, the Nx586 decoder can fetch up to eight unaligned bytes from the prefetch buffer. It can also fetch directly from the cache but is restricted to aligned accesses of eight bytes.

Superscalar Core Improves Efficiency

It is difficult to categorize the Nx586 due to its unusual design, shown in Figure 2. From the standpoint of x86 instructions, the Nx586 can be thought of as a scalar processor with a very deep pipeline that can execute most instructions in a single cycle. The CPU is not really superscalar because it fetches and decodes only one x86 instruction per cycle.

The Nx586 goes to great lengths to maintain this rate of one instruction per cycle, regardless of data dependencies, cache misses, branches, resource conflicts, and other events that cause glitches in most other processors. The key to this strategy is the superscalar processor core, which executes RISC86 instructions using four function units: two integer units, an address unit, and the optional floating-point unit.

Because one x86 instruction can turn into two or more RISC86 instructions, the decoder can issue multiple RISC86 instructions per cycle, one to each function

		NexGen Nx586	Intel Pentium	PowerPC 601
FP Add (DP)	latency	2	3	4
	throughput	2	1	1
FP Multiply (DP)	latency	2	3	4
	throughput	2	2	2
FP Divide (DP)	latency	40	39	31
	throughput	40	39	29
Integer Multiply	latency	8	10	9
Integer Divide	latency	39	41	36

Table 1. NexGen's chip set executes complex integer and double-precision floating-point arithmetic at about the same speed as Pentium or PowerPC. (Source: vendors)

unit. The function units are designed to work in parallel, executing multiple RISC86 instructions per cycle. NexGen calls the Nx586 superscalar, which it is from the RISC86 standpoint.

To further exploit the parallelism of the RISC86 instructions, a 14-entry queue precedes each of the function units, as Figure 2 shows. If instructions cannot immediately execute due to dependencies or resource conflicts, they simply wait in the queues; other function units can continue to execute. In this way, the superscalar core is similar to that of IBM's Power2 processor *(see **071301.PDF**)*.

The queues prevent the instruction decoder from stalling when a function unit is busy, as it can simply issue instructions into the queue. If the instruction at the front of a queue cannot be executed for any reason, however, that function unit stalls. This problem will completely tie up one unit while the others continue. The dual integer units provide some redundancy; if one stalls, the other can continue processing integer operations.

The Nx586 allows up to 14 RISC86 instructions to be pending at any one time; NexGen says that there are often eight or more instructions in process, and that it is not unusual to reach the limit of 14. One effect of the queues is that instructions can be executed out of order, though they are always issued in order. The CPU tags each instruction with a sequence number. The tags help ensure that instructions with dependencies are executed in the proper order. The use of register renaming reduces the number of dependencies, increasing parallelism in the instruction stream.

Proper handling of exceptions can be complex in an out-of-order machine. The Nx586 always retires instructions in order, even if their results were generated out of order. Exceptions are handled when the excepting instruction is retired; the results of all successive (unretired) instructions are nullified. Register renaming simplifies this process. Values in the physical registers are not overwritten until after the instruction that generated them is retired; intermediate values are kept in other physical registers. Nullifying instructions is simply a matter of updating the register mapping.

Although out-of-order execution does not require additional overhead for the general-purpose registers, the Nx586 must keep multiple copies of special registers, such as the flags and segment registers, to correctly handle exceptions. All writes are queued in an eight-entry write-reservation station and are not executed until the write instruction is retired, ensuring that the cache/memory system always sees in-order, nonspeculative writes. Reads can take data directly from the reservation station, bypassing the L2 cache.

Fast Floating-Point Unit

Like most microprocessors with two integer units, the Nx586 does not have symmetrical units. The primary unit can handle all RISC86 integer operations, including multiply and divide, while the second integer unit performs only simple (single-cycle) operations. The decoder has a load-balancing algorithm to allocate instructions that could be sent to either integer unit.

RISC86 load and store instructions are routed to the address unit, which calculates the target address and performs translation and validation according to the x86 standard. Since there is a single address unit, only one RISC86 load or store instruction can be executed on each cycle. The chip contains a 32-entry unified TLB for virtual address translation.

The optional 587 chip handles all floating-point operations. The FPU receives instructions from the decoder at the same time and in much the same way as the other three function units. The FPU can execute double-precision adds and multiplies in just two cycles, one fewer than Pentium; Table 1 shows the latencies for various math operations. NexGen did not implement Pentium's parallel FXCH feature and thus chose not to pipeline the FPU, since most code requires an FXCH between each math operation. Pentium's ability to issue an FXCH along with a math operation may balance out the performance advantage of the faster adds and multiplies.

Pipeline Depth Varies

It is difficult to pin down the amount of time that it takes to execute an instruction on the Nx586. The decoder can issue nearly all instructions in a single cycle, but execution may be delayed depending on interactions in the RISC86 core. Even basic pipeline concepts are difficult to apply to this design—but we'll try anyway.

The first few pipeline stages are the same for most instructions, as Figure 3 shows. During the first stage, an instruction is fetched from either the instruction prefetch buffer or the instruction cache. This stage will stall for two cycles if the prefetch buffer is empty and the requested instruction stretches across an eight-byte boundary, but this situation occurs infrequently.

Once the instruction is fetched, it takes two cycles (D1 and D2) to decode the x86 instruction and translate

it into RISC86 instructions. A third cycle (T) allows these instructions to transit to the function units and is mainly a vestige of NexGen's original eight-chip design (see MPR 4/89, p. 6), although the extra cycle permits instructions to reach the off-chip FPU at the same time as the on-chip function units.

At this point, things get complicated. The simplest case, shown in Figure 3(a), is a register-to-register integer calculation. Assuming that the queues are empty, it can be executed in a single cycle (EX) and retired in two cycles (WB and RET). The scoreboard and the register map are updated on the final cycle.

Figure 3(b) shows a memory-to-register calculation that is translated into two RISC86 instructions:

```
LOAD R2, [R3]
ADD R1, R2
```

The LOAD is sent to the address unit while the ADD goes to one of the integer units. Again assuming that the queues are empty, the LOAD begins processing immediately, but the ADD stalls until the LOAD completes. This stall ties up one integer unit, but the other integer unit (and the FPU) could process subsequent instructions during that period. The LOAD itself takes three cycles, two to generate, verify, and translate the address and one to access the data cache.

In practice, however, several RISC86 instructions usually are queued at any given time. In this situation, one or more delay cycles (Q) may be inserted into the execution of a particular RISC86 instruction, as Figure 3(c) shows. Because the core can execute multiple instructions per cycle, these delays usually are not reflected in the apparent execution of the x86 instruction stream.

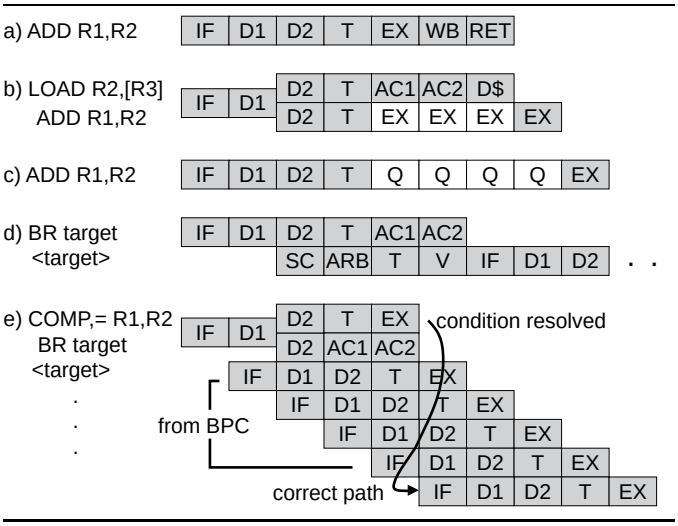

Figure 3. The Nx586 executes the simplest instructions in five cycles but can take many more cycles in other situations.

Branch Prediction Cache Avoids Penalties

When the Nx586 encounters a branch, it predicts the outcome and begins to execute subsequent instructions. This is called speculative execution, since these instructions may be incorrect if the branch condition is mispredicted. The Nx586 can speculatively execute beyond two predicted branches; in most cases, the first branch condition will be resolved by the time a third branch is encountered.

Figure 3(d) shows a branch predicted to be taken. By the end of the D1 phase, the target address has been calculated from the instruction. This address is then used to start an instruction fetch by assuming that the target is on the same virtual page as the previous address. The virtual target address is translated by the address unit in parallel with the fetch, and the fetch is restarted if the translation indicates that the target is on a different page.

In the meantime, it takes four cycles to transmit the address to the instruction cache and begin receiving data. This seems absurdly long for an on-chip access, but most of these cycles are a legacy from the old multichip design; an extra cycle is also required to update the scoreboard. In total, there are five cycles during which sequential x86 instructions could have been decoded and issued; these instructions must be invalidated.

To eliminate this five-cycle taken-branch penalty, NexGen implemented a 96-entry branch prediction cache (BPC). The company would not elaborate on the structure of the Nx586 BPC, but patent number 5,230,068, issued to NexGen, describes a BPC that contains the first 24 instruction bytes at each target address, along with the target address itself. This design is similar to the branch target cache in AMD's 29000 but different from Pentium's, which contains only target addresses.

The patented BPC is indexed by the address of the branch instruction, so it could be checked during the IF stage and immediately begin supplying instructions to the decoder. Figure 3(e) shows how the target instructions could flow from the BPC with no apparent delay. If the target address is read from the BPC and issued as in Figure 3(d), the 24 instruction bytes would be enough to bridge the gap until the instruction cache begins responding, even for most misses to the L2 cache.

Mispredictions Are Costly But Rare

Figure 3(e) represents a conditional branch broken into a compare instruction and a branch. As the branch itself is handled by the BPC as described above, the compare is dispatched to one of the integer units for evaluation. If the queues are empty, as in the figure, five cycles are lost if the result of the compare indicates that the branch was mispredicted. If the queues are not empty, 19 or more cycles can be lost before the misprediction is detected. These penalties give the Nx586 the appearance of a very deep pipeline.

To avoid these severe penalties, the Nx586 uses the now-standard two-bit Smith and Lee algorithm to pre-

Figure 4. The Nx586 die includes 3.5 million transistors and is built using 0.5-micron CMOS. In a fit of paranoia, NexGen would not reveal the die size or provide a die overlay.

dict branches *(see **070402.PDF**)*. According to the patent, each BPC entry contains two prediction bits. If a branch misses the BPC, an additional 2,048-entry, two-bit-wide branch history table is checked, increasing the predication accuracy compared with Pentium's 256-entry branch target buffer.

Although these two structures will correctly predict most conditional branches, they do not help RET instructions. These branches are hard to handle because the target address can change on each iteration. The Nx586 includes a return address stack, as in Digital's Alpha architecture, that handles up to eight subroutine calls. The combination of these three structures should push the prediction success rate above 90% on most code, compensating for the significant penalties that can occur when the Nx586 mispredicts a branch.

Cache Hierarchy Reduces Latency

By moving the FPU off-chip, the Nx586 has room for 16K each of instruction and data cache, twice the size of Pentium's caches. NexGen's caches are both four-way set-associative, further increasing the hit rate compared with Pentium's two-way caches. Both are physically indexed and tagged.

The speed of the 0.5-micron process allows accesses to occur in just one-half cycle at 66 MHz, or two accesses per cycle. Because the Nx586 has only one address unit, the second cache access is used for snooping or for moving data to and from the L2 cache or the system bus. Many processors block the cache when these events occur, stalling CPU accesses, but the Nx586 can handle them without slowing instruction execution.

The built-in cache controller connects to an external L2 cache constructed of standard asynchronous SRAMs. Only two configurations are supported: 256K or 1M, both using eight ×8 SRAMs. The L2 cache is unified (instructions and data) and, like the L1 caches, is four-way set-associative. The controller allows two cycles to access the external cache, requiring 15-ns SRAMs at 66 MHz.

The tags are stored in the same chips as the cache data, reducing the amount of memory available for data by 6%. A cache access requires two cycles to read the tags, then two cycles to read each quadword of data (4-2-2-2 access pattern). Reading the tags in series with the data simplifies the implementation of a set-associative cache, since the correct set is determined before the data is read; otherwise, the chip would have to support a 256-bit SRAM interface to read all four sets at once.

The Nx586 takes slightly longer to access its cache than a 66-MHz Pentium, which typically has a 3-2-2-2 access pattern using the same 15-ns SRAMs. NexGen's set-associative design will have a higher hit rate than a direct-mapped cache of the same size for a Pentium chip. Another advantage of the NexGen design is that it can maintain the same access pattern at higher frequencies because the cache bus can be clocked at a different speed than the system bus. A hypothetical 100-MHz Nx586 would require 10-ns SRAMs for a 4-2-2-2 access. A 100-MHz Pentium, on the other hand, has a 5-3-3-3 access because its system bus runs at 66 MHz, although it has the cost advantage of retaining 15-ns SRAMs.

The on-chip caches use a write-through protocol, taking advantage of the direct path to the L2 cache. The external cache uses a write-back design to reduce traffic on the system bus. Writes are sent to both the data and instruction caches to support self-modifying code.

All caches and the write buffer maintain coherency with other data in the system using a MESI protocol. This protocol allows other caches (typically other processors) to coexist in the system. The Nx586 snoops all transactions on NexBus; if a read snoop hits in any of its caches, it aborts the bus transaction and writes the dirty data back to main memory. Because of the double-speed L1 caches, most snoop transactions are transparent to the processor.

Nx586 Matches Pentium Performance

NexGen would not reveal the exact die size of the Nx586 but said that it is comparable to Intel's 163-mm^2 P54C. Even if the two chips have the same die area, the Nx586 will be more expensive to build because the 0.5-micron process costs more than Intel's 0.6-micron process. The Nx586 also uses a costlier package. Assuming a 163-mm^2 die, the MPR Cost Model *(see **071004.PDF**)* estimates that the Nx586 will cost around $220 to build,

50% more than the P54C. Adding the smaller Nx587 chip could bump the total to about $270.

System vendors, however, are most interested in system cost. At 60 MHz, NexGen's 586 is priced at $460, plus $86 for the NxVL system interface and a few dollars for an 82C206. A 60-MHz Pentium lists for $675; Intel's PCIset costs $84 with an ISA interface. Thus, NexGen's solution would save the system vendor more than $200.

The above comparison is somewhat unfair because it does not include floating point. An Nx587 coprocessor adds $128 to the cost of the NexGen system, reducing the difference to about $80. Many PC users do not need a floating-point coprocessor, however, and for these users the NexGen design offers a significant advantage.

This advantage depends on the Nx586's ability to keep pace with a Pentium of the same clock speed. On small benchmarks such as Landmark and PowerMeter, the performance of the Nx586 ranges from 14% worse than a Pentium to 28% better. NexGen has not run the SPEC benchmarks on its processor.

Assuming that the CPU cores are roughly equal, the NexGen processor should have better cache performance due to its larger on-chip caches and set-associative L2 cache, compensating for the extra penalty cycle on L2 accesses. NexBus adds overhead to memory accesses, however, so Pentium could have an edge on programs that frequently access memory. On FP-intensive code, the Nx586 can execute basic math operations 50% faster than Pentium but requires extra cycles for FXCH. Without any additional benchmark information, it appears that the two chips should offer similar performance on most applications.

Pentium, however, is able to operate at speeds up to 100 MHz, giving system designers a range of performance points from a single motherboard design. NexGen believes that, with additional characterization and modification of critical paths, it can get its Nx586 to 80 MHz and beyond without moving to a more expensive IC process. Still, the fact that the NexGen chip achieves a lower clock speed than Pentium while using a faster IC process seems to indicate an architectural deficiency.

NexGen hopes to remedy that deficiency with a 686 design aimed at 2–4×the performance of the Nx586. Although the company did not discuss details of this future product, the Nx586 architecture could be extended with a second decode unit and additional function units.

Little Immediate Market Impact

NexGen (and its investors) should be congratulated for persevering on the long road to shipment of its first product and for delivering that product at a competitive price/performance point. The company's goal, however, had been to deliver performance that Intel couldn't match, but Intel's new 100-MHz Pentium makes the Nx586 just another x86 contender.

Price and Availability

The Nx586 CPU, in a 463-pin PGA, lists for $460 in the 60-MHz speed grade and $506 at 66-MHz. The Nx587 FPU, in a 183-pin PGA, lists for $128 in either speed grade. The NxVL system-logic chip, in a 401-pin PGA, is priced at $86. All prices are in 1,000s.

The Nx586 and NxVL are currently sampling, and the company expects them to ship in 2Q94, with the Nx587 following by midyear. Contact NexGen (Milpitas, Calif.) at 408.435.0202; fax 408.435.0262.

Four small PC vendors—Adisys, Lucky, Compu-Tek, and Tangent—have announced systems based on the Nx586. Adisys (Santa Clara, Calif.) will offer a 60-MHz system with 8M of memory, 250M hard drive, and a 14" monitor for $1,795, about $800 less than a comparable Pentium system from Dell *(see **080401.PDF**)*. Nx586 motherboards will be supplied from seven Taiwanese vendors. To establish a significant market for these motherboards, NexGen must meet certain criteria.

Like all new x86 vendors, the company must first demonstrate uncompromising compatibility with x86 software. One advantage of the product's long development cycle is that the company has spent literally years testing its design with a wide variety of applications; it claims that the current version is fully compatible, but only time will tell. Cyrix and IBM already have proved that it is possible for new chips to be compatible with Intel's, priming the market for other competitors.

NexGen must also resolve its foundry situation and exhibit an ability to deliver an adequate supply of parts. So far, the company has had its best success in Taiwan, where Pentiums have been few and far between. But as Intel floods the market with millions of Pentium chips this year, system vendors' unit demands will increase, raising the bar for NexGen.

Finally, the startup must weather the inevitable legal challenges from Intel. If NexGen uses IBM as a fab, it can deploy the same patent cross-license defense that Cyrix has successfully used, although Intel is now challenging IBM's right to build 486 chips for Cyrix. If it is forced to an unlicensed foundry, NexGen claims that its independent design does not violate any Intel patents, although the company's reluctance to discuss address translation indicates a nervousness about the infamous '338 patent and other memory-management issues.

It will take most of this year to resolve issues of compatibility and the lack of a PCI interface. Even then, the company must maintain a price advantage over Intel. The apparently higher manufacturing cost of NexGen's chip will put it at a disadvantage if Intel continues to aggressively cut Pentium prices. If Intel falters, however, NexGen will be first in line to fill the gap. ♦

