# P6 Microcode Can Be Patched

## *Intel Discloses Details of Download Mechanism for Fixing CPU Bugs*

*by Linley Gwennap*

Taking an unusual approach to fixing bugs, Intel has implemented a microcode patch capability in its P6 processors, including Pentium Pro and Pentium II. This capability allows the microcode to be altered after the processor is fabricated, repairing bugs that are found after the processor is designed. Intel has already used this feature several times to correct minor bugs, and in the future, it may save the company from recalling CPUs if a major problem is discovered.

Although there have been rumors about downloadable microcode in previous Intel chips (see MPR 10/88, p. 6), the company says the P6 is the first of its chips to include this feature. The CPU maker originally intended the feature to be used only for debugging, but after dealing with the expense of the Pentium FDIV bug (see MPR 1/23/95, p. 4), Intel decided to make it usable in the field. The company plans to include this capability in all future processors.

High-end systems such as mainframes and some workstations often allow the system's boot code to be remotely patched, and some allow microcode updates as well. RISC-based systems, however, have no microcode, so this technique can't be used to fix bugs in the microprocessors themselves. CISC chips from AMD and Cyrix have microcode, but these vendors have not revealed any patch capabilities.

### Patches Stored in BIOS

Like other Intel processors, every P6 chip contains a complete set of microcode in an internal ROM. When it powers up, a P6 processor begins using this internal microcode. If a system does not have microcode patches installed, the processor will always use this internal microcode. So far, Intel is offering microcode patches only as part of BIOS (or other boot) code; the company is investigating whether the patches can be loaded by the OS, as a Windows DLL file, for example.

Intel has worked with all major BIOS vendors to get the patch feature into the BIOS of most P6-based PCs. In these systems, the BIOS writes a memory address into a special CPU register to trigger a download sequence. Upon receiving this command, the CPU downloads new microcode from the specified address.

P6 processors contain a small SRAM that holds up to 60 microinstructions. The patch code is downloaded into this SRAM. The processor also contains a set of "match" registers that cause a trap when a particular microcode address is encountered. (This is similar to the "instruction breakpoint" capability used to debug assembly code.) This trap, which takes a single cycle to process, vectors microcode execution into the patch RAM.

The downloaded microcode consists of two segments. The first is an initialization routine that is run immediately after the code is downloaded. This routine can be used to reconfigure certain aspects of the processor. This segment of microcode also initializes the match registers, if necessary.

The second segment contains one or more patches that remain in the patch RAM during normal operation and are accessed via a match-register trap. Since the original microcode is stored in ROM, it cannot be modified; the match registers allow the operation of the microcode to be changed. In this way, an x86 instruction that is operating incorrectly can be repaired, assuming it is implemented in microcode.

When an instruction needs to be repaired, a patch is created to replace a section of the original microcode, performing the correct operation and then jumping back to the original flow. The extra cycles spent jumping to and from the patch code will, of course, impact performance, but in some cases a bug can be corrected using this mechanism.

### Only Certain Bugs Can Be Fixed

Intel did not disclose the number of match registers, other than to say that there are more than one. The number of match registers and the size of the patch RAM limit the number of bugs that can be fixed for any particular version of the processor. A single bug, however, might require multiple patches, and some bugs are too complex to repair in this manner. Intel believes that the current mechanism could allow multiple bugs to be fixed, if necessary.

The P6 does not use microcode to implement most x86 instructions. These simpler instructions are the least likely to have bugs and are also the most heavily tested. In the past, postproduction bugs that affect specific instructions have been found in the more complicated microcoded instructions, such as FIST and FDIV. Intel admits that the Pentium FDIV bug could not have been fixed in this way if Pentium had patchable microcode, so it isn't certain whether the patch capability will save Intel from a major bug.

As another example, Intel encountered a problem with the P6's FIST instruction just after Pentium II was launched (see *www.MDRonline.com/P6/bug*). After much investigation, the company chose to offer a software workaround rather than a microcode patch. Intel would not say whether the FIST bug could have been fixed in microcode, but its actions suggest that a microcode patch would have been too large to fit, or at least too large to leave sufficient room for future use.

Another category of bugs is caused by the incorrect operation of certain features in the processor, such as instruction buffers, load and store buffers, and branch prediction. Many such features of the P6 processor can be disabled via a

special register. In the case of such a bug, the downloaded microcode can reconfigure the processor to turn off this feature. In these cases, the processor will then operate correctly, although often with some performance penalty.

A large majority of bugs, however, cannot be fixed by either type of microcode patch. If some part of the internal CPU or the bus interface is simply not functioning properly in certain circumstances, it is often impossible to correct via microcode. Intel says it has fixed a handful of P6 bugs using the patch feature, but this is a small percentage of the total number of reported errata. Thus, while the patch feature is useful, it is not a panacea for all processor problems.

### Security Features Block Tampering

The new mechanism provides the potential for devious hackers to reprogram other people's processors without their knowledge. To avoid this possibility, Intel has implemented several layers of security.

Patches are provided to BIOS vendors and system vendors as a 2,048-byte block of data. The block contains a 48-byte header—which includes a date code, the CPU ID (which includes the stepping level) of the target processor, and a checksum—and 2,000 bytes of data to be downloaded by the processor. The checksum allows the BIOS vendor to make sure the data block has not been corrupted during transmission, but it is not used by the CPU.

The 2,000 data bytes are encrypted in a way that Intel claims will be extremely difficult to break. The bytes are divided into blocks of varying lengths, each of which is encoded differently. Because the actual microcode patches are typically much smaller than 2,000 bytes, the remaining data is random noise intended to confuse anyone attempting to break the encryption.

Even if a hacker could successfully decipher the encryption, the next challenge would be constructing a legal patch. Intel has not published any information on the format of its microcode, which the company claims is deliberately designed to be difficult to understand. Only a small number of Intel employees know the P6 microcode formats. Thus, it would be nearly impossible to construct a microcode patch that would, for example, cause the processor to add when it was supposed to subtract.

If someone could crack the encryption algorithm, they could simply crash the CPU. If the downloaded byte stream successfully passes through the decryption process, the CPU does not verify that the downloaded microinstructions are valid; it simply executes them. Executing random microinstructions would most likely cause the processor to hang; if this code were written to flash BIOS, it would prevent the system from successfully booting. Of course, there are many far easier ways to crash a Windows PC.

### A New Way to Fix Bugs

The P6, like any microprocessor, has its share of problems. In the past, these bugs would be worked around in either hardware or software, or else simply tolerated by the end user, since most are incredibly obscure. In the case of the famous FDIV bug, however, Intel actually had to offer to replace millions of Pentium chips that had already been shipped to customers, ultimately costing Intel as much as $475 million.

The new capability in the P6 gives Intel another tool to correct such problems. The company has already used the method to fix bugs in the initial steppings of Pentium Pro and Pentium II. With each new stepping of the chips, the company incorporates these patches into the internal microcode, but as new bugs are found, new patches are needed.

One problem with this method is that newly discovered bugs may require adding patches to older versions of the processor. It may not be possible to bring the oldest steppings into full compliance, since the number of patches might exceed the resources available. As time goes by, fewer and fewer bugs are typically discovered, so hopefully this limit will not be reached.

BIOS vendors are most affected by the new patch capability. Intel periodically issues new patch codes to these vendors. Each processor stepping has its own 2K patch code; if a BIOS vendor wants to support all five existing steppings, it would need to set aside 10K of ROM. As the P6 continues to proliferate, this ROM space will balloon further. Instead, most BIOS vendors have decided to support only the two or three most recent steppings, assuming that the PC maker will not use an old CPU with a new BIOS ROM.

In multiprocessor systems, this simplification may be inadequate, forcing these systems to have a ROM large enough to accommodate several processor versions. In fact, many MP systems ship with empty processor slots that can later be filled. These systems should include a flash BIOS that can be updated with future patches. Upgrading a Pentium II PC with a new processor poses a similar problem; fortunately, most Pentium II PCs use flash BIOS.

### Upgrading Processors in the Field?

Intel has so far used the patch feature only to repair minor bugs in its processors. In theory, the feature could be used to upgrade processors in the field. For example, Intel could release a low-priced "crippled" P6 processor with many of the performance-enhancing features disabled via microcode. An end user could later purchase from Intel a software utility that would upgrade the processor's performance. This mechanism would require each CPU to have an individual encryption key, to prevent mass distribution of the upgrade utility, but such a protective method is surely possible.

Whether Intel will ever attempt such a radically new business model is unclear. The company will continue to use microcode patches to fix minor processor bugs. In the case of a major hardware problem, the ability to download a patch might save the company millions of dollars, improving customer satisfaction as well. Intel has found a creative way to protect itself against at least some of the inevitable hardware bugs—a way that its competitors have yet to emulate. M