Chapter 13. FPU's and multimedia

The computer is constantly performing *calculations*, which can be divided into two groups.

- Whole numbers
- Floating point numbers

The whole number calculations are probably the most important, certainly for normal PC use – using office programs and the like. But operations involving floating point numbers have taken on greater significance in recent years, as 3D games and sound, image and video editing have become more and more a part of everyday computing. Let's have a brief look at this subject.

Floating point numbers

The CPU has to perform a lot of calculations on decimal (or *real*) numbers when the PC runs 3D games and other multimedia programs.

These decimal numbers are processed in the CPU by a special unit called an FPU (*Floating Point Unit*). In case you are wondering (as I did) about the name, *floating point* – here is an explanation: Real numbers (like the number π, pi) can have an infinite number of decimal places. In order to work with these, often very large, numbers, they are converted into a special format.

First the required level of *precision* is set. Since the numbers can have an infinite number of decimals, they have to be rounded. For example, one might choose to have *five* significant digits, as I have done in the examples below (inside the PC, one would probably choose to have many more significant digits).

Once the precision has been set, the numbers are converted as shown below (the decimal point *floats*):

- The number 1,257.45 is written as 0.12575×10^{4}.
- The number 0.00696784 is written as 0.69678×10^{-2}.

Now the FPU can manage the numbers and process them using the arithmetic operators.

Normal form	Rewritten	In the FPU
1,257.45	0.12575×10^{4}	12575 +4
0.00696784	0.69678×10^{-2}	69678 -2

Fig. 92. Re-writing numbers in floating point format.

FPU – the number cruncher

Chapter 13. FPU's and multimedia

Floating point numbers are excessively difficult for the CPU's standard processing unit (the ALU) to process. A huge number of bits are required in order to perform a precise calculation. Calculations involving whole numbers (integers) are much simpler and the result is correct, every time.

That is why an FPU is used – a special calculating unit which operates with floating point numbers of various bit lengths, depending on how much precision is needed. FP numbers can be up to 80 bits long, whereas normal whole numbers can "only" be up to 32 bits (permitting 4,294 billion different numbers). So the FPU is a number cruncher, which relieves the load on the ALU's. You can experiment with large numbers yourself, for example, in a spreadsheet.

In Excel 2000, 2^{1023} (the number 2, multiplied by itself 1023 times), is the biggest calculation I can perform. The result is slightly less than 9 followed by 307 zeroes.

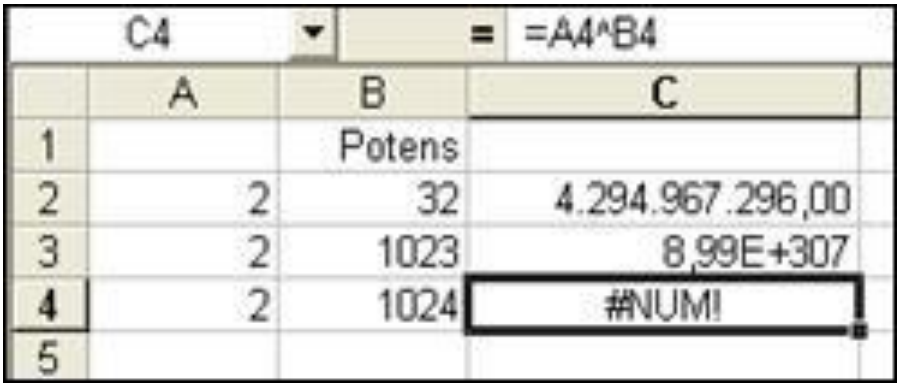

Fig. 93. Experiments with big numbers in Excel.

Modern CPU's have a built-in FPU (*Floating Point Unit*) which serves as a number cruncher. But it hasn't always been like this.

For example, Intel's 80386 processor didn't have a built-in FPU calculating unit. All calculations were done using the processor's ALU. But you could buy a separate FPU (an *80387*), which was a chip which you mounted in a socket on the motherboard, beside the CPU. However, in the 80486 processor, the FPU was built-in, and it has been that way ever since.

Fig. 94. A "separate" FPU, Intel's 80387 from 1986.

3D graphics

Chapter 13. FPU's and multimedia

Much of the development in CPU's has been driven by 3D games. These formidable games (like *Quake* and others) place incredible demands on CPU's in terms of computing power. When these programs draw people and landscapes which can change in 3-dimensional space, the shapes are constructed from tiny *polygons* (normally triangles or rectangles).

Fig. 95. The images in popular games like Sims are constructed from hundreds of polygons.

A character in a PC game might be built using 1500 such polygons. Each time the picture changes, these polygons have to be drawn again in a new position. That means that every corner (*vortex*) of every polygon has to be re-calculated.

In order to calculate the positions of the polygons, floating point numbers have to be used (integer calculations are not nearly accurate enough). These numbers are called *single-precision floating points* and are 32 bits long. There are also 64-bit numbers, called *double-precision floating points*, which can be used for even more demanding calculations.

When the shapes in a 3D landscape move, a so-called matrix multiplication has to be done to calculate the new vortexes. For just one shape, made up of, say, 1000 polygons, up to 84,000 multiplications have to be performed on pairs of 32-bit floating point numbers. And this has to happen for each new position the shape has to occupy. There might be 75 new positions per second. This is quite heavy computation, which the traditional PC is not very good at. The national treasury's biggest spreadsheet is child's play compared to a game like Quake, in terms of the computing power required.

The CPU can be left gasping for breath when it has to work with 3D movements across the screen. What can we do to help it? There are several options:

- Generally faster CPUs. The higher the clock frequency, the faster the traditional FPU performance will become.

Chapter 13. FPU's and multimedia

- Improvements to the CPU's FPU, using more pipelines and other forms of acceleration. We see this in each new generation of CPU's.
- New *instructions* for more efficient 3D calculations.

We have seen that clock frequencies are constantly increasing in the new generations of CPU. But the FPU's themselves have also been greatly enhanced in the latest generations of CPU's. The Athlon, especially, is far more powerfully equipped in this area compared to its predecessors.

The last method has also been shown to be very effective. CPU's have simply been given new *registers* and new *instructions* which programmers can use.

MMX instructions

The first initiative was called MMX (*multimedia extension*), and came out with the Pentium MMX processor in 1997. The processor had built-in "MMX instructions" and "MMX registers".

The previous editions of the Pentium (like the other 32 bit processors) had two types of register: One for 32-bit integers, and one for 80-bit decimal numbers. With MMX we saw the introduction of a special 64-bit integer register which works in league with the new MMX instructions. The idea was (and is) that multimedia programs should exploit the MMX instructions. Programs have to be "written for" MMX, in order to utilise the new system.

MMX is an *extension* to the existing instruction set (*IA32*). There are 57 new instructions which MMX compatible processors understand, and which require new programs in order to be exploited.

Many programs were rewritten to work both with and without MMX (see Fig 96). Thus these programs could continue to run on older processors, without MMX, where they just ran slower.

MMX was a limited success. There is a weakness in the design in that programs either work with MMX, or with the FPU, and not both at the same time – as the two instruction sets share the same registers. But MMX laid the foundation for other multimedia extensions which have been much more effective.

Chapter 13. FPU's and multimedia

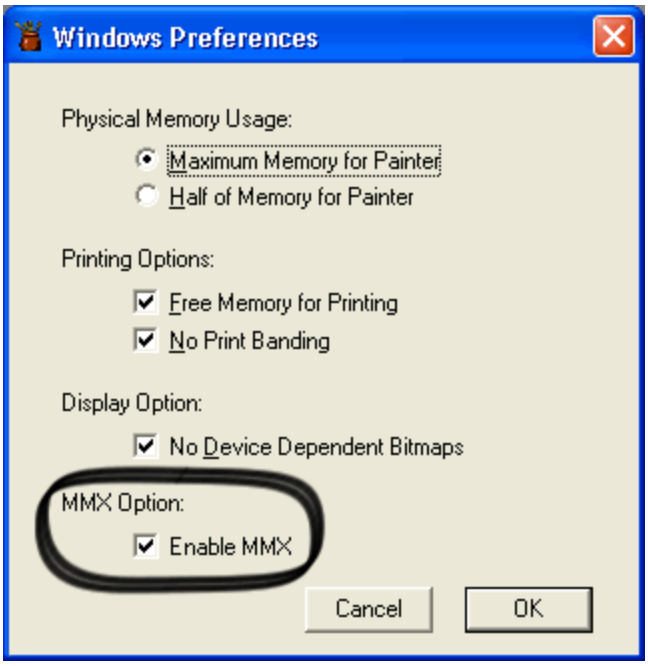

Fig 96. This drawing program (Painter) supports MMX, as do all modern programs.

3DNow!

In the summer of 1998, AMD introduced a collection of CPU instructions which improved 3D processing. These were 21 new SIMD (*Single Instruction Multiple Data*) instructions. The new instructions could process several chunks of data with one instruction. The new instructions were marketed under the name, *3DNow!*. They particularly improved the processing of the 32-bit floating point numbers used so extensively in 3D games.

Fig. 97. 3DNow! became the successor to MMX.

3DNow! was a big success. The instructions were quickly integrated into Windows, into various games (and other programs) and into hardware manufacturers' driver programs.

SSE

After AMD's success with 3DNow!, Intel had to come back with something else. Their answer, in January 1999, was SSE (Streaming SIMD Extensions), which are another way to improve 3D performance. SSE was introduced with the Pentium III.

In principle, SSE is significantly more powerful than 3DNow! The following changes were made in the CPU:

- 8 new 128-bit registers, which can contain four 32-bit numbers at a time.

Chapter 13. FPU's and multimedia

- 50 new SIMD instructions which make it possible to do advanced calculations on several floating point numbers with just one instruction.
- 12 *New Media Instructions,* designed, for example, for the encoding and decoding of MPEG-2 video streams (in DVD).
- 8 new *Streaming Memory* instructions to improve the interaction between L2 cache and RAM.

SSE also quickly became a success. Programs like Photoshop were released in new SSE optimised versions, and the results were convincing. Very processor-intensive programs involving sound, images and video, and in the whole area of multimedia, run much more smoothly when using SSE.

Since SSE was such a clear success, AMD took on board the technology. A large part of SSE was built into the AthlonXP and Duron processors. This was very good for software developers (and hence for us users), since all software can continue to be developed for one instruction set, common to both AMD and Intel.

SSE2 and SSE3

With the Pentium 4, SSE was extended to use even more powerful techniques. SSE2 contains 144 new instructions, including 128-bit SIMD integer operations and 128-bit SIMD *double-precision floating-point operations.*

SSE2 can reduce the number of instructions which have to be executed by the CPU in order to perform a certain task, and can thus increase the efficiency of the processor. Intel mentions *video, speech recognition, image/photo processing, encryption* and *financial/scientific programs* as the areas which will benefit greatly from SSE2. But as with MMX, 3DNow! and SSE, the programs have to be rewritten before the new instructions can be exploited.

SSE2 adopted by the competition, AMD, in the Athlon 64-processors. Here AMD even doubled up the number of SSE2 registers compared to the Pentium 4. Latest Intel has introduced 13 new instructions in SSE3, which Intel uses in the Prescott-version of Pentium 4.

We are now going to leave the discussion of instructions. I hope this examination has given you some insight into the CPU's work of executing programs.