

IBM's Cell Processor :

The next generation of computing?

By: David K. Every

February 11, 2005

Price: **\$1.95**

Published by Shareware Press

© David K. Every, 2005

Edited by: Roger Born & Tim Robertson

This document is a shareware report, normally these reports would sell for significantly more than I'm asking, but I'd rather make it widely available. If you read it and find it useful, valuable or entertaining, tell your friends, forward them copies, and please send \$1.95 (or more) to me via my paypal account (mevery@neo.rr.com). This modest fee helps pay for the effort and resources that went into producing this report, and will guarantee that I can afford to spend time on similar reports in the future.

About the Author

David Every is Director of Web Development for a midwestern media conglomerate and a long time computer industry observer, writer and analyst, with education and experience in hardware, software and business. He has a Masters Degree in Business Administration from Pace University in New York. He can be reached by email at: dke@mac.com or you can read his writings at: <http://www.mymac.com> or <http://www.igeek.com>



Legalese

This electronic report (eReport) can be sent in it's entirety but not excerpted or altered without the written permission of David K. Every.



TABLE OF CONTENTS

Introduction	2
RISCy Business	4
Parallel or Serial Growth	6
Hardware	7
Asymmetry : All things are not equal	8
Basics of Vector / SIMD	11
Is Asymmetry inelegant?	12
Taking a cell apart	13
Isochronous Cells?	15
Input/Output : Feed me Seymour!	15
Power Management	17
Putting it all together	18
Software	19
What's old is new again	19
Core Image & Core Audio	21
XGrid and Cellular Degeneration	22
Impact on the market	24
Mainstream or Niche?	24
What will Apple do?	25
Now or Later?	26
What does this mean to the PC?	27
Bibliography	29

Introduction

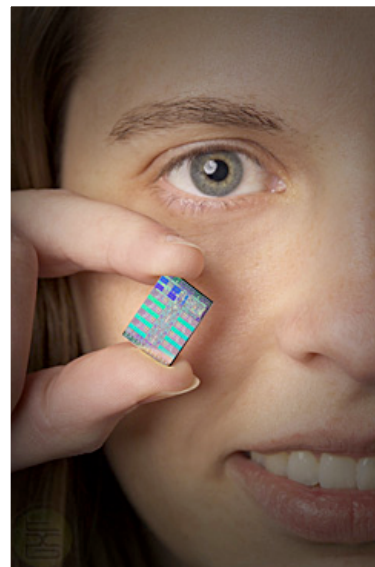
IBM got together with Toshiba and Sony in 2001 to create the STI (Sony Toshiba IBM) initiative, and created a specialized PowerPC alliance -- not unlike the AIM (Apple, IBM and Motorola) alliance of 1991. It does appear that Apple is being involved with STI or at least keeping their eye on developments, but it is not public knowledge in what capacity or whether they are intending on adopting it in the first iteration.

The idea behind the original alliance, AIM, was that if Motorola and IBM made and used the same chips, they could share development costs and create a new hardware and software standard to break open the PC market. Apple was the glue and motivator, as well as contributing the client side experience to the mix, and was a catalyst for the alliance itself.

We can debate whether AIM was a success or failure, but they did produce quite a few designs and worked together for a while, before Motorola and IBM went their separate ways. Now Motorola spun-off their Freescale division (processors/chips), and IBM started partnering with other companies. AIM certainly never achieved their goal of becoming the market leader in mainstream desktop processors, but they were briefly the performance leader and created many elegant and innovative designs that had a degree of success, and the PowerPC is one of the last RISC chips left standing while most others died out.

The idea of AIM was that RISC was the next big leap in computer technology, and the AIM partners as the pioneers could get ahead of the market, and once ahead could keep the lead. The first part was true, the latter was not.

Mainly, people didn't anticipate that Intel (and later AMD) could borrow the architectural improvements from RISC, and shoehorn them into the old x86 CISC design; but that's exactly what they did. While there's still an architectural superiority to the PowerPC over the Pentium and Athlon's older x86 design, and a far better ISA (Instruction Set Architecture) most of that is hidden from users. The advantages left are in areas like performance per watt, or transistor count, or heat, or things that end users don't care about as much in the mainstream market. Their attitude is "so it is



25% hotter, it runs my games pretty well” or, “so it costs a little more to manufacture, they just run thinner margins and higher volumes and it comes out in the wash”. The areas that do matter, like cost to manufacture, or cost to design are diminished by the huge volumes that Intel or AMD do, that helps minimize those benefits. So while the PowerPC dominance in things like embedded controllers, specialized designs, game consoles, and many other “niche” markets, they didn’t win the mainstream of desktops.

Enter STI. It’s strange that we’re talking about a 4 year old initiative like it is brand new; but they’re just getting the chip to market later this year, and announcing and showing the designs now. Realistically, it will be next year before the volumes start really ramping up with the second generation. (First generations tend to be as much proof of concepts, or lower volume working development and demonstration chips). But that 5 year head-start to market is going to translate to a serious competitive advantage.



STI has similar but wider goals than AIM. They have a new hardware design that will once again leap ahead of Intel. However, their design will give them a much larger leap over Intel and x86 designers. Last time, AIM tried to use Windows NT, IBM’s AIX and Apple’s MacOS to crack the Desktop PC market. But just the transition for Apple from 68000 to PowerPC cost them a few years. Microsoft sabotaged the Windows side, and IBM used the chips successfully in their machines, but they were more special purpose than general purpose. Many things are different this time.

The Cell Processor uses a PowerPC as the core processor, making it compatible with many of the tools and users that are familiar with that chip. RISC gave the PowerPC a 50% performance advantage that decayed over 5 years. With the cell we’re talking about 1,000% performance advantage, for certain things, that should scale up over time. STI still has a window of about 3-5 years before Intel or AMD can rip-off the idea, implement it, and catch up. But there was far less pre-announcements and giving away the details before hand, we’re learning the details this time months before the chip makes it to market, instead of a few years.

Last time, Microsoft was releasing Windows95 at the exact same time as Apple was releasing PowerPC support. That undercut the Mac and PowerPC’s advantages with coincidental perfect timing, and Apple couldn’t leverage the PowerPC’s advantage because they had a lot of legacy 68000 code. This time the software market is already being broken open. Apple is starting to make gains again with OS X (UNIX), IBM uses UNIX (Linux), Sony and Toshiba are using variants of UNIX, as are many

others. Software is now far more file compatible than it was in 1995, and Applications are far more portable from platform to platform. So whether Microsoft comes along with the STI initiative or not, this could be the start of a new race for platform dominance, with it being Intel (and possibly Microsoft) against lots of other companies at once. Whether the Cell starts this race in earnest, or just the market timing, it looks like Apple and others are already gaining ground from the back of the pack. OpenSource is making inroads, and there's nothing Microsoft can do to stop it, the Cell Processor could be the thing that gets people to jump in droves.

So that's what the Cell Processor is about. It is not just a cool embedded processor, or multimedia processor, or graphics processor, or I/O processor or signal processor -- though it will tear through those markets like a hot knife through butter. The Cell processor has a chance at ripping open the soft underbelly of the Wintel duopoly, and opening the market for many more competitors working on variants of a new standard. This could really change what the industry looks like in 10 years.

So let's dive in, and go over the different areas of the new architecture, and what it will mean to the PC industry.

RISCY BUSINESS

There are two design philosophies in any industry; old school and new school. New school is the idea that you can gain a competitive advantage by being the first in a new discipline/technique and become the market leader. Old school is the idea that things have gone too far, and the future is in the past, and by returning to your roots and the basics, you can increase efficiency and gain competitive advantages from that.



In the late 70's and early 80's computers instructions were getting more and more complex; which was taking more space and offering diminishing returns and costing more money in design and time to market (opportunity costs), and becoming harder to scale with each subsequent generation. Moore's law (really an observation by Intel's Gordon Moore) infers that you're going to double chip capacity every 18-24 months. With development windows that short, you need to plan for the future and the rapid scaling that brings. Some designers felt we needed to go "back to the basics" with RISC.

CISC or Complex Instruction Set Computing was the trend of the time. The idea was that people were adding more and more instructions and complexity to processors, while compilers and programmers were only using a small fraction of those instructions effectively. The instructions that were used were complex and tried to do many things at once (load and process and store). This took up logic (space), time to design, had bugs, and held the rest of the processor back because simple instructions had to wait for complex instructions to complete first. This made it far harder to scale, and so on.

RISC (Reduced Instruction Set Computing) was the idea that most instructions could be made out of a few smaller and simpler instructions; like they had been in the past. The reduced instructions were simpler thus it was easier to reduce bottlenecks; both in time to design and time to execute. Simplifying the steps of instructions (only allowing a load or store, but not both) meant that the computer could be clocked higher / run faster. Reduced meant complexity mostly, but they also reduced instruction count, both gave them space to do other things, which they spent on things that were simpler to design but offered better returns; like having more registers or cache (fast pool of memory) because memory accesses were a huge bottleneck. They could even reorder some instructions to keep things moving when some areas of the processor were traffic jammed and make better use of the internal resources, or just make the processor use less power or cost less to make (smaller is cheaper). It was a design philosophy of where you were going to spend your energy and transistors (space).

It worked; RISC chips ran faster, were easier to design, were more power efficient, cost less, and so on. The only downside was they tended to have simpler instructions and were a little less memory efficient; the simplification process also included making more things had to be the same size, instead of allowing them to be as small as possible. But this basic rule of applying business logic to computer logic, and weighing where you were going to spend your budgets; either transistors or design people was the real success.

Almost all processors since RISC design was popularized have either adopted the design philosophy or at least been influenced by it. Even CISC computers like Intel's x86 adopted the ideas; they just put a RISC engine behind an interpreter/emulator that let them use the old instructions (and breaks them down into simpler RISC instructions) creating a hybrid design. VLIW, EPIC and almost all new designs may try to extend RISC a little (grouping instructions, pushing branch hinting on the compiler, and other subtle changes), but RISC as the foundation philosophy won out.

Parallel or Serial Growth

Just because RISC, the design philosophy, won in the past does not mean the battle between old school and new school is over. There's still been entropy. RISC designs have gotten more complex over time; each subsequent generation of RISC chips has resulted in more transistors (space) being devoted to allowing the processor to look ahead more, reorder more, rename more, have artificial register sets (rename registers), avoid branches or take both at once, and so on. They increase this "instruction window" (big complex buffering) to try to increase the speed/efficiency at which each processor can run. They have tried many techniques to cram more parallelism (more instructions and data at a time), into a single serial stream. But it is a game of diminishing returns. Each generation doubles or more the complexity, for marginally better sequential/serial performance or efficiency, or just reducing the loss. (Processors are getting faster at a quicker rate than memory is, so processors have to do more just to minimize that bottleneck and not lose ground). Some designers want to go back to the basics and ask, "is it worth it".

The results are that we're back to the beginning again. Some people believe the future is in cleaning out the cruft; paring down the processor to simpler designs, again, and instead of having one large core (processor) in a chip, trying to break a large stream (set of instructions) into many smaller ones thus making the machine faster, we should simplify the processor and use the saved space to put multiple streams/cores on the same chip. The simplification implies that each core might run a little slower individually, but additively they are far faster, and far easier to scale.



The Cell is just an impure (hybrid) implementation of that "old school" philosophy. Instead of throwing transistors at making a computer serially faster, they want to simplify it and let a bunch of these smaller computers/cores all work at the same time, each on their own task, getting more work done. Would you rather have one giant or half a dozen elves working on a problem? In a pure design, you'd start over and make all the cores real simple; the Cell processor is more pragmatic and says you can do both, have a main processor that is fairly complex (and backwards compatibility), but have lots of little new helpers as well. A balance between the giant and the army of elves.

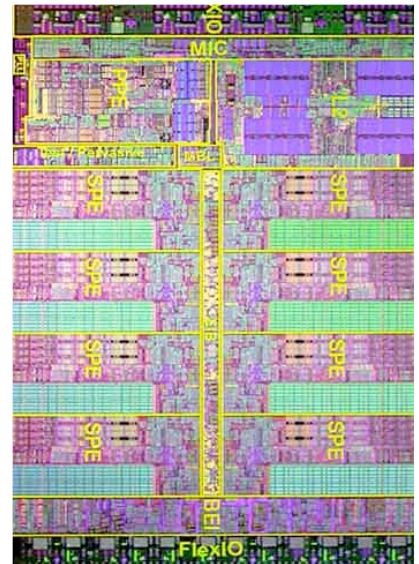
The “New School” people aren’t ready to throw in the towel either; Moore’s law is working for them too. They can’t keep making a single processor better at the rate they need to, but they can keep adding more complex cores to their designs every few years. So they’ll just add fewer big cores, instead of more little ones. But there’s no doubt that we are maxing out what we can do on a single processor and multi-streamed parallelism (more cores) is going to start taking over. Which leaves the true question about the success of the Cell; will the added efficiency gained by using simple specialized processors outweigh having fewer but more powerful processors?

Hardware

While STI has been keeping the wraps on the details, there is a lot of information available in the form of multiple articles, patent filings, with liberal amounts of educated speculation.

The technical specs include:

- 221 mm², making it a medium sized chip
- 234 million transistors
- fabricated with 90 nanometer SOI technology (Low K, 8 layers, Copper interconnect)
- runs at 4.6 GHz at 1.3v (50-80W estimates)
- has an 85° Celsius operating temp w/heat sink
- 6.4 Gigabit communication channel(s) to the outside world.
- 4 x 128 bit internal bus (ring), 96 Bytes/cycle
- 9 Cores / 10 Execution threads



All this is enough to get hardware designers attention. To give you an idea of what the specs mean (some relative measure), this is roughly twice the number of transistors as the newest Pentium4 and four times the G5. Normally, that is a yawner, Moore’s law infers that processors double in transistor capacity every 18-24 months, so transistor count by itself isn’t a huge deal. The number that gets heads to turn is “9 cores”, or the ability for it to work on 10 separate things at the same time. (The main core is dual threaded, so can do two things at once). Intel’s best processors can barely manage dual cores. With the cell, each of those cores requires much less

space, and seems to be much more efficient (based on projections) than what Intel can manage, or will be able to manage without a major change.

The operating temperature (heat) is an eye-opener too, as is the current physical size. They didn't imply whether that was a fanned heat sink, so the heat might not be that bad. Power consumption is projected to be in the 50-80 watt range; not exactly a chip. IBM is working on their "shrunk" 65 nanometer process, and you can bet this chip will be moved to that as soon as possible. When they shrink the process, that should help all the specs; meaning that the chip will decrease in physical size (cost), increase in performance, and decrease in power consumption and hopefully heat.

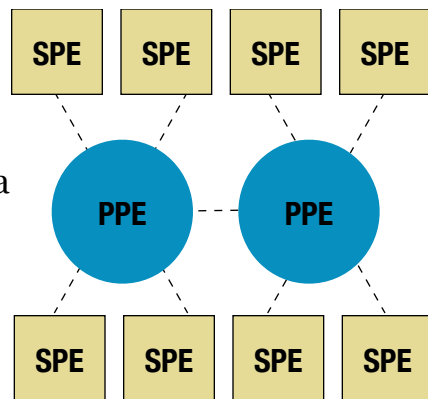
As it is, it is running at over 4.6 GHz, and they've maxed some out at over 5.6GHz, meaning this chip is fast. Technically, more GHz doesn't mean faster. GHz is like RPM's in your car -- more isn't necessarily better, it is a balancing act of running the processor fast, have a short and efficient pipe, higher IPC (Instructions per cycle), better instruction efficiency and keeping the processor fed. Theoretical numbers thrown out are 256 GFlops (peak) with single or double precision math. That's per chip! Current PowerPC and Pentiums are in the 5-10 GFlops (peak) range (single precision). So that's an improvement of 25 times, or more. The issue is keeping the monsters fed.

How does it manage to be faster in GHz, better in IPC, lower power per instruction, and use fewer gates per operation? The answer is in specialization and asymmetry.

Asymmetry: All things are not equal

The Cell is known as an asymmetric design; it doesn't have 10 PowerPC CPU's in it, it has two types of sub-processors in it -- a PPE (Primary Processing Entity - with dual cores) and eight helper units called SPE's (Synergistic Processing Elements). Leave it to IBM and a couple Japanese companies to create acronyms that roll off the tongue so easily.

Primarily the chip is a PowerPC, this PPE keeps the processor compatible with lots of older Applications, to the point that without much work Apple can run OS X on one, and so can most Applications written for the PowerPC. The design borrows some elements from newer PowerPC's like the G5 (64 bit), but has been

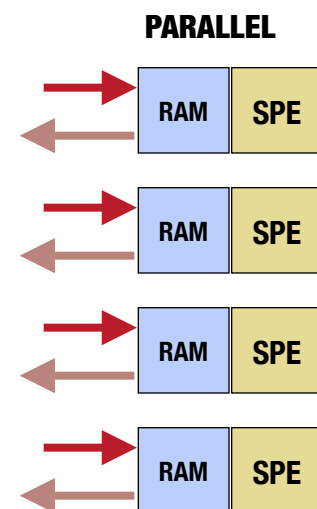


Note: the figure is meant to show the concept, not the actual topology or way that the processors are connected -- that is more dynamic or independent.

pared down with a smaller pipeline more like the G3. Its history is from an earlier research project (the GigaProcessor) with revisions over time. It is a two issue processor with SMT (Simultaneous MultiThreading) which acts like dual cores but shares some resources (like the current generation of high end Pentiums). This isn't likely to beat the newest PowerPC cores in instructions per cycle, but it is clocked fast and well fed to compensate. It retains its VMX or "Velocity Engine" vector processor (as Apple calls it), in order to offer full software compatibility.

While the PPE is an elegant microprocessor on it's own, it is not the exotic part of the Cell – it is just another PowerPC, with some dependency on its very high memory thru-put, high clock rate and low latency short pipelines to make up for its simpler design.

The exotic part is the rest of the chip; where they add up to 8 SPE's. These 8 little helper cores or SPE's, are what they call "Cells". Each cell has some memory and a 4 x 128 bit ALU's (Arithmetic Logical Unit which does the math in a processor), and it has 128 of the 128 bit registers. Think of these as vector processors both simplified (not as many modes/instructions), and on steroids (more registers, faster, more math units and their own flow control). Or in other words, instead of the main processor having to control the work being done by the AltiVec unit, there are stand alone APU-LETS (little applications for these Alternate Processing Units) that can run their own tasks without bothering the main processor as much. And there's 8 of them.

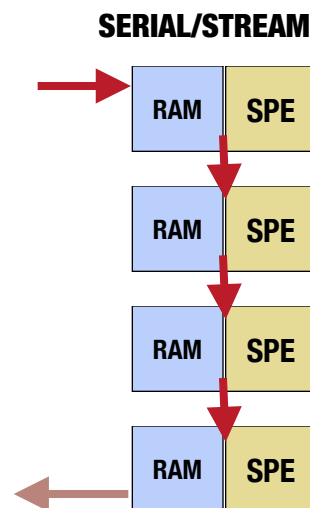


Each SPE can do simplistic things, like video or audio filters, moving things around, and stuff like that -- but they do their processing very fast. Before you discount this, there are many thousands of tasks like this that a computer wants to do each second -- so offloading them is a huge win. Basically, a super computer is doing lots of these simple filters being done very quickly. That's why STI is marketing this as "a Super-Computer on a Chip"; while a slight overstatement since it would take many of these to equal the performance of today's super computers, there's a strong element of truth because the basic design is that of a network clustered Super-computer. And before you scoff at the performance, remember one of these can keep up with a true Super-Computer from a decade or so

ago. So it is a Super-Computer on a chip, just not equal in performance to the latest ones.

Having all these units gives another level of flexibility, basically chaining these processors together (called streaming). The work on one SPE can be fed to another, which continues the work - making a Super-SPE or Stream. This can do more work on a single stream of data than a single SPE -- or what would have taken 8 passes, could take only one. Of course it is one pass with 8 stages, but there are many cases where this is faster or easier.

With 8 SPE's, it gives you a lot of variations in chaining, and of course the way the cells are streamed will be constantly changing over time, with the needs of the software. You just need an innovative software design to manage the cells (tasks).



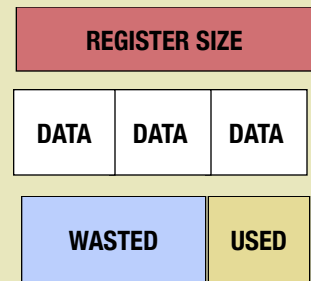
What was done is combine the “short vectors” of SIMD; being able to execute a few simple instructions in a pass, with “deep/long vectors” of the Cray Super-Computer, and being able to execute many more simple instructions in a single pass. And they did it in a dynamic way. You could sick 8 x 128 bit SPE's (doing 4 x 32 bit work) on the same task, making it behave like a massive 32 x 32bit Vector unit. Or you could make a stream processor that is doing dozens of instructions with very low latency and very high thru-put. Or have all the SPE's working on different things. Or any combination of the set. Simple, yet massively powerful and versatile.

The whole design seems to be a “what if” dream machine research project that got out of hand and became real. It was IBM saying, “what if we put 8 independent VMX / SIMD engines on a chip”. Or just as easily Sony saying “what if we made the next generation of our Emotion Engine, put a better main CPU on it, and instead of 2 SIMD engines we added 8 more powerful ones”. From the looks of it, it was a convergence of the two overlapping ideas.

If you want to know more on how SIMD works, just read the next section. If you're already know, you can just skip it.

BASICS OF VECTOR / SIMD

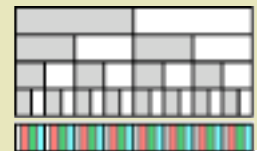
Normally a single instruction works on a single chunk of data (we can call this SISD; Single Instruction, Single Data). If the chunk of data is the same size as the register, everything is fine. Most of the time it is not. So you need to mask out the other information, then work with just the piece you want before you work with it. This results in a lot of wasted space, or sometimes instructions because things just don't line up right.



If that doesn't sound efficient, it is because it isn't. In most cases when you're dealing with large sets of data, the data does not naturally align with the register size. You either spend a lot of time packing and unpacking the data, or you artificially make it fit with lots of wasted space.

SIMD (Single Instruction, Multiple Data) means that a single instruction works with many chunks of data at once, even when the data doesn't match your register size. This tends to work better if all the data you're working with is the same size, as this is commonly the case, it works quite well. If you wanted to do a filter on a bunch of elements in a picture, instead of doing it one pixel or element at a time, you do it to as many as will fit in a 128 bit register (or two) at a time. One really cool instruction is called permute, which allows programmers to scramble, reorder, or pack/unpack data.

VMX is really versatile with data size; it can handle 64 bits x 2, 32 bits x 4, 16 bits x 8, 8 bits x 16, a bizarre 1/5/5/5 mode x 8 (that makes sense for some color functions), and even some 4 and 1 bit instructions.



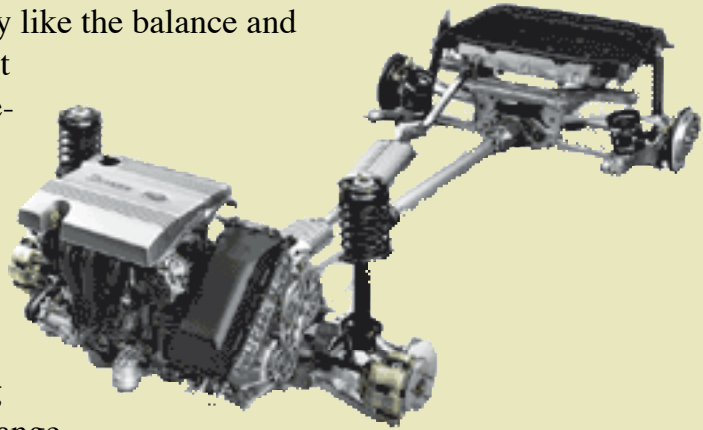
SIMD works extremely well if you have a stream of data, and in many applications there are lots of streams of data. SIMD is a really quick way to crunch, filter or manipulate it. For this type of work, this can be 10 or 20 times faster than the normal SISD part of the processor.

Since MMX, VMX and other vector processor designs have been released, they have been very successful and the fastest evolving area of processor design. The next evolutions are likely to be as dedicated coprocessors, with more registers and their own flow control with better hinting so that it requires less overhead. Heck, why not put many of them with a traditional processor; hey wait, that's what the Cell Processor is.

IS ASYMMETRY INELEGANT?

Asymmetry is foreign to some; they like the balance and harmony of sameness. But that isn't always the best thing in efficient design; more important is, "does the tool fit the problem".

In the auto world, the latest advances are in asymmetrical gasoline-electric hybrid motors. Two separate engines each catering to their strengths; electrics in low range stop and go situations, and gasoline in long range higher horsepower situations.

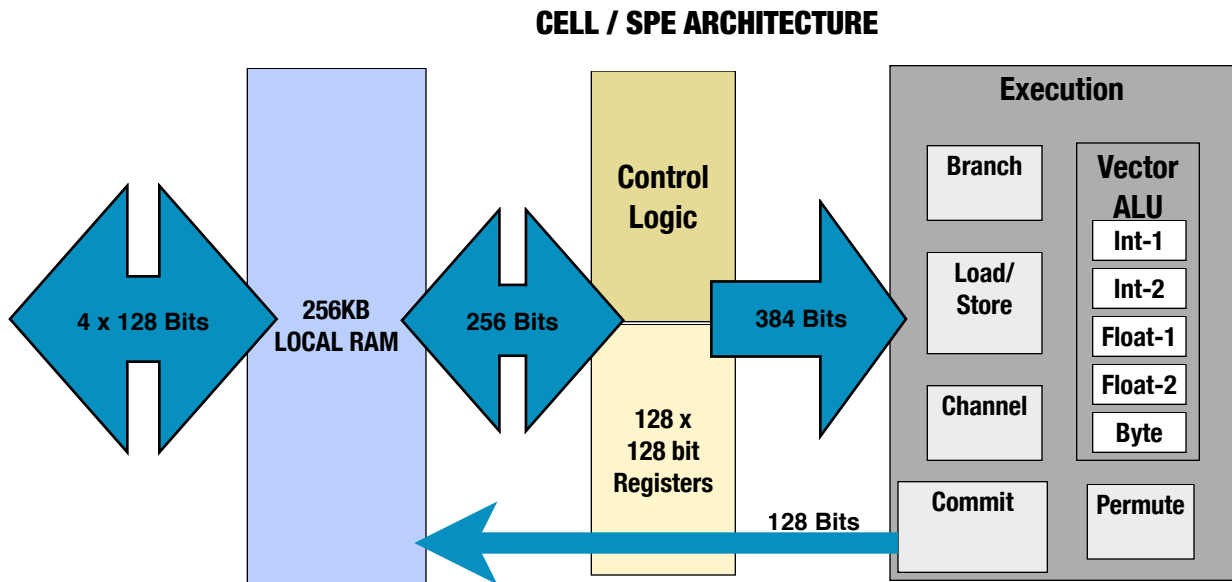


In the 1980's, Apple released the MacIIx, which had basically two AppleII's on board (as specialized DMA chip), that did nothing but manage serial input and I/O so the rest of the computer would be free to do it's work. The same with most GPU's (Graphics Processor Units / Graphics Chips), sound chips, network controllers and so on. Offloading specialized work to specialized processors can give you huge gains.

Even look into a modern processor like the PowerPC or Pentium; they have integer units, floating point units, and vector (SIMD) units. In the case of Pentiums there are differences between the integer (MMX) and floating point (SSE) SIMD units. Asymmetry is already in processors. The Cell Processor just decided that since you use the SISD and SIMD sides differently, they could break SIMD units into dedicated coprocessors.

Those that think "multiple same cores are better" are thinking about the versatility. You can use each core for anything, and they don't want to artificially limit themselves. But they aren't thinking of efficiency. If 90% of what you need to do, could be handled by a simpler processor, and you could get three of those processors into the space of one of the versatile ones, then why waste the resources? Why refuse the gains offered by asymmetry? **What matters most is how well the tool fits the problem.** I suspect there is a ton of analysis being done by the geniuses at IBM, Sony, Toshiba, that show that this tool fits the problem. If it doesn't, then people won't use it. But they designed it this way because past experiments like VMX itself have shown that asymmetry often fits the problems better. Really, Cells are just a way of responding to that realization and scaling it up.

Taking a cell apart



Each Cell/SPE is a Vector Processing computer in its own right with 21 million transistors: 14 million SRAM and 7 million logic, and capable of 32 GOps/GFlops each. Not a large core by today's standards, but respectable and significant. There are both single precision and double precision modes, so it can crunch some numbers. Face it, that's still a few times faster than today's PowerPC's and Pentium's, but it is a more limited/focused processor.

Each SPE has 256K of local memory, and which gets filled/cleared from the outside world through one of the 4 x 128 bit rings. (A ring is just a type of bus or network). You can think of this as a network computer, on four very high speed internal networks; with each network being able to transfer 16 bytes per clock (plus control tags). Resulting in about 73.6 GB/s. Did I mention a **REALLY** fast network?

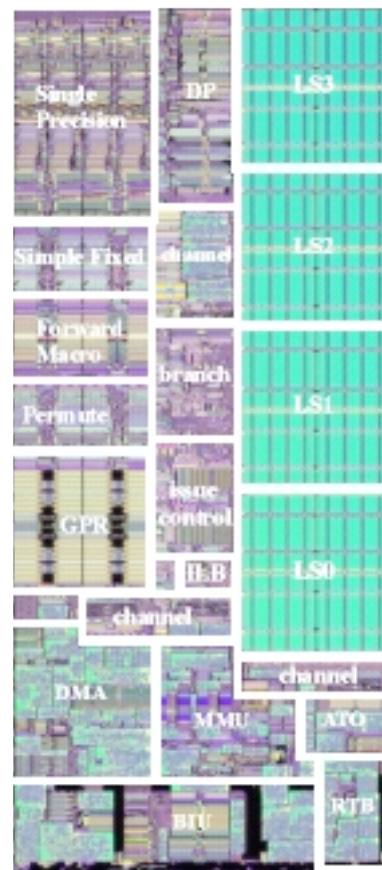
While each cell does not have it's own "cache", that is functionally what each one's local memory is working as. Functionally a cache is a local pool of something (that is quicker to access then a far away resource), which is what this locally mapped memory does. Technically a cache in a processor is more complex than normal memory, being dynamically mapped and tagged, with lots of associated logic -- so purists will

deny this is cache, while pragmatists will see what its purpose is more than how it is doing it. There isn't a lot of logic on it to handle collisions with other processors,

Furthermore, while each cell does not have its own dedicated cache, there is a large L2 cache (512K) that is used by all the processors; PPE and SPE alike. So there is a secondary cache that is just a shared pool for everyone.

The SPE is a RISC-like Load/Store architecture, with 32 bit instruction, with 3+1 operands (2 or 3 sources, and one target register). It has a DMA engine (Direct Memory Management) that can fetch or flush groups of data with a single instruction, and a simplistic RISC design (dual entry) in that it isn't as deep nor will it allow Out-of-order execution (beyond an instruction pair). This means it looks an awful lot like VMX (AltiVec/Velocity Engine), but both simpler and more powerful, without achieving compatibility.

An SPE (also called APU) has more units than VMX, but they are simpler (not as many data sizes or modes). It sticks with 32 bit single precision mostly, with some bytes and integer capabilities (conflicting results on double precision, with no being most likely). Since these are the more popular uses for VMX, many will not be bothered by the limitations. The branching and some more control flow should help more than compensate, not to mention having four times the number of registers (128 of them!). It has a simple commit, which makes sense as this processor not meant to do as much heavy/complex code flow and instead a simpler stream processor. So for a DSP or Vector Processor it is actually quite advanced/sophisticated, and is a "back to basics" RISC design, with a few VLIW concepts thrown in.



NOTE: The patent filing(s) showed each cell as having twice as many ALU's, and twice the bandwidth/rings. Either simulations showed this was the sweet spot, or scaling up in the future is on their mind.

ISOCRONOUS CELLS?

One subject that is in the patents that has not been discussed a lot in other resources is the isochronous nature of the cells. Isochronous is pronounced eye-sock-ra-nuss, and it refers to processes where data must be delivered within certain time constraints (guaranteed delivery). USB, FireWire, ATM and networks often do this to guarantee that a user can get the bandwidth that they need; and a video or audio stream isn't interrupted because something else is hogging all the shared resources.



While real-time Operating Systems have some isochronous behavior, meaning they will interrupt one thing to guarantee another gets its time; it is done in software and far from perfect. Modern operating systems try to mimic this behavior even looser; they will divide time among many tasks, but it is not guaranteed, as every user that has seen a video skip, or heard the audio breakdown knows.

The patent discusses each cell's ability to guarantee time to certain tasks, thus behaving isochronously. What this means is the cells are a superior media processor, network processor, or stream machine; because you can lock into the chip itself that it will give you time in this cell to do that task at this rate; then the rest of the tasks get to use the cell in the "left over" time. The patent does not discuss whether this is a hardware or software function (with some implications of a combination). Obviously the designers are thinking of how to make cells self-sustaining with as little overhead, and as much return as possible. Applying higher level thinking to lower level elements. There are many ways this could benefit the users of a computer that does this automatically, or at least better than current designs.

Input/Output : Feed me Seymour!

One of the primary issues of many new processors is the I/O bottlenecks. Each new processor is getting faster and faster, and at a rate that memory and memory controllers have not kept up with. This cell processor is a leap ahead in processing capability, but with 9 separate cores, it needs a huge leap in memory and cache designs to

keep these monsters fed? It has a voracious data appetite, and something has to keep shoveling.

It has a 32KB L1 cache and 512KB L2 cache, which is about the size of the PowerPC G5's; but this thing is running a lot faster, and has many more processors on it. The good news is that each of the 8 SPE's has it's own 256KB of SRAM (local pool of memory), bringing the total cache up to over 4 times more than the G5. But that's still not much based on the high clock rate and massive computing power; so the rest of the design is meant to help.

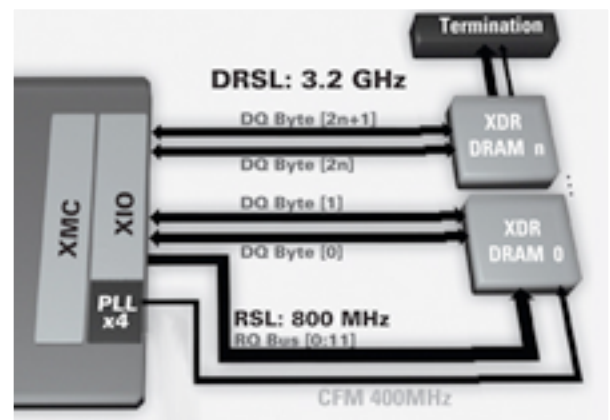
A lot of the data movement is intended to be internal. If you're streaming or chaining data from one SPE to another, you're not saturating your external bus or memory -- this means that you're getting a lot more computing power for a lot less memory access. This is why they spent more resources on four powerful ring networks inside the CPU itself.

They dealt with RAMBUS on their pipes both to memory and as a peripheral bus. RAMBUS is boasting that 90% of the pins for external access were designed by them.

The RAM is accessed through the new dual XDR interface. Each XDR is running at 3.2 GHz transferring 2 bytes at once (6.4 GB/s), by two banks of ram (12.8 GB/s) -- and this is a dual controller, so there are two of these. Resulting in a peak performance of 25.6 GBytes per second.

Compared to today's memory busses this is at least four times faster and closer to 8 times faster. They really want to keep this baby fed well. **But there's a HUGE caveat: the XDR is designed as an embedded subsystem and the current memory limitations are 256MB.** Either they are going to have to radically increase memory size, or change the architecture before it is viable in a mainstream application.

For a primary bus they are using RAMBUS's FlexIO, which can run from 400MHz to 8GHz, and is backwards compatible with HyperTransport, RapidIO, SPI-4, and so on. (Meaning it is easier to adapt to existing designs). Running at 8 GHz this asymmetric bus running has 12 byte lanes each transmitting at 6.4GB/s. With 7 outbound

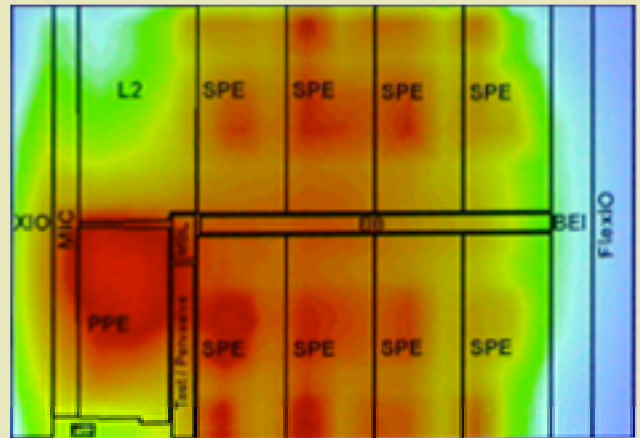


lanes it has 44.8 GB/s, and 5 inbound lanes gives it 32 GB/s, for a total of 76.8 GB/s total. Which is another world class bus.

The total I/O between these two is over 100 gigabytes-per-second (they are totally independent of one another), again, a new high water mark for a mainstream processor. The objective is that since these cores are all big eaters, they need to be able to access memory fast, and avoid memory and talk to each other very quickly.

POWER MANAGEMENT

The Cell Processor has dynamic digital power management. In fact, it has 10 heat sensors (presumably on each of the processors and cache), 5 power management states and a linear sensor. Assumably the chip can slow down individual units to reduce heat if spot heating, or start shifting load to other Cells if one side is getting too hot, and so on.

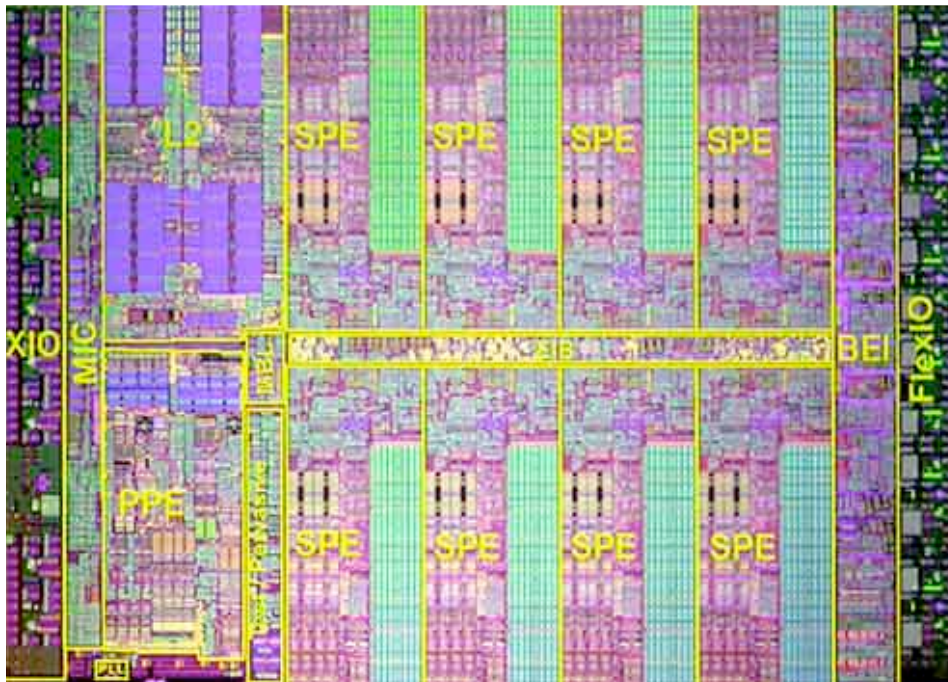
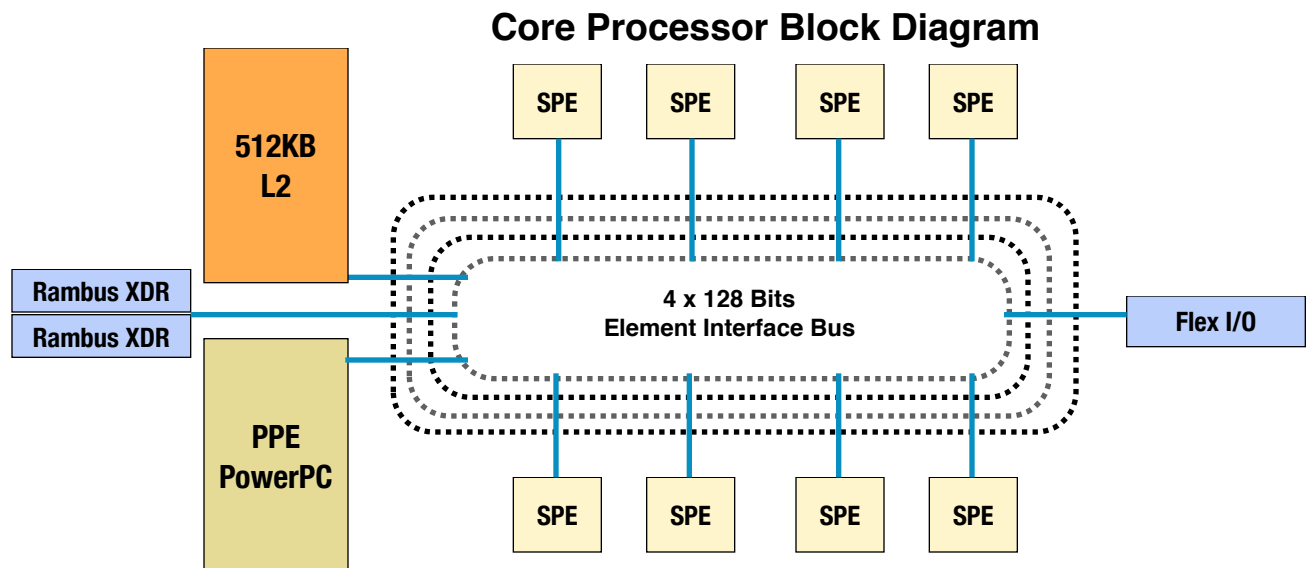


When most people hear of Power Management they think of portables. While you certainly could make a portable version, and power consumption is not excessive, it would more likely be a future version designed for that. Over time, they've stated the goal of using this architecture from handhelds on up to super-clusters.

For now, one of the primary concerns about having power management is that they need the added reliability that this brings. Also in super-computer clusters, performance per watt is very important. If you're magnifying your heat and power problems across a few hundred or few thousand processors, it can quickly add up. Early super computers required exotic power and cooling solutions; so it does not hurt to be safe and plan ahead.

Putting it all together

Here's the block diagram of the processors units and the die:



Software

The issue of cool hardware is always dampened when it collides with the realities of software. How long does it take to take advantage of these cool new ideas?

When Intel decided to increase parallelism and finally fix the architecture of their 30 year old x86 (IA32), they created their Itanium (IA64). Basically they decided to create a new RISC chip, that grouped multiple instructions into sets of 3 (with control) to make a very long instruction word (VLIW). They added some hinting, some more registers, and a few other ideas used in RISC or VLIW and gave it their own name; EPIC (Explicit Parallel Instruction Computing). It was new, it was cool, and it was much better than tired old x86. And it flopped like a pancake. It's still around, but adoption has been a fraction of what they thought, and market-share is rapidly approaching zero.

Itanium (IA64) didn't fail because they were being acronym happy, or that it wasn't a reasonable design. It was fair. For many more transistors they were getting a little better performance. It didn't realize the compiler optimizations or ease of scalability they had been promising, but it did work and was a big architectural leap over the x86 (IA32). The real gotcha was that it couldn't run any software (well) unless it was recompiled and tuned for EPIC. This is a huge expense. On top of that there were delays after delays, so by the time the IA64's made it to market, their performance was at best, modestly better. So the whole industry yawned, looked the other way, and the processor family was stillborn. If it had been more usable with older software, it might have been more viable. AMD released their 64 bit implementation to some success, because it still worked with old code. In other words, "it's the Software".

What's old is new again

Well the good news is that while the Cell Processor is a whole new architecture in regards to the SPE's, the PPE is a good ol' PowerPC with few changes. This means you can drop it into new designs, and it will still run old software with little modification. There's little performance penalty (if any) for adopting the Cell Processor. That removes a major barrier to entry for the processor. But to be a success, it needs to also add something of value; some upside return. (Realize the benefits of all that extra hardware and processing power it has).

I'm going to use Apple's software efforts as examples, because I know them the best and they are the biggest player in the market and most likely to mainstream and promote the Cell Processor. All these generalities should apply to Sony, Microsoft, CISCO, Ford, IBM, or any of the other PowerPC customers that might use the cell processor. Now before you comment, I realize, IBM is bigger than Apple, and that Sony's 100M Playstation3 projections are nothing to scoff at; but how are they likely to use this chip? They are likely to make specialized controllers, low end servers, or do a specialized super cluster with it. Sony is going to use it in their PlayStation3, CISCO, Ford and others are likely to stick these in other things like network routers or cars. All these are high volume applications, but Apple is the one that would get the most attention by using them in mainstream computers, and they have the most legacy (ISV's: Independent Software Vendors that this would impact) holding them back. So Apple is the one that challenges the PC status-quo the most.

When Apple coordinated SIMD (AltiVec) being added to the PowerPC, it was easy for them to also add SIMD support to the Operating System. Apple cherry picked a few of the most often used areas of the OS, and those that would gain the most from having SIMD support, and added it. Even without changing a line of code in any 3rd party Applications, the system ran faster. Over time, Apple added more support, and things just kept getting better for years.

Application Providers did pretty much the same thing. A few Apps, like 3D renderers, Games, Photoshop, Audio Processing programs and the like, all added SIMD support to a few key areas, and got a pretty massive improvement. This is known as the 80/20 rule; 20% of workers on any project do 80% of the work, or in computer programs the high-load/often-executed areas of code give high returns for low optimization investments.

Apple achieved success, by focusing on the low hanging fruit, and getting the large returns in small areas. Why wouldn't adapting cell processor work the same? In fact everything about the new adoption is easier:

1. Apple has already isolated parts of the Operating System that get the biggest bang for their buck. They just need to retune them for the more powerful Cells.
2. When they did this the first time, they created libraries that Application Programmers could use to make their programs faster. Well, it isn't like starting over; this is more like "VMX: the next generation". Take what you have and improve it. Evolution is easier than revolution (or pioneering).

3. Cells are more powerful than VMX; they are self-feeding, streamable, and have more flow control. There are more units (8 instead of 1) and they are faster, thus you get a better returns with lower CPU load.

So the returns on the cell processor are far greater, with less work. Apple has the infrastructure in place with their Core Image routines, Core Audio routines, their VMX libraries, many developers are already convinced of the value of such things (the human factor), and lastly but not least is XGrid. This is without getting into what STI is going to offer to help (compilers, libraries, and so on). So I'm pretty sure Apple has interest in taking their VMX capabilities to the next level.

CORE IMAGE & CORE AUDIO

Now that I mentioned Core Image and Core Audio, what are they?

Cells behave a lot like GPU's (Graphics Processors or Video Chips that are currently in computers) and vise-versa. So what if Apple wrote a set of routines that would use the GPU to apply filters or help the processor do things? Since they would be graphics based, they could call this "Core Image".

Now GPU's are very specialized. They work well for graphics, but not very well for anything else. Each cell will be more versatile, but a little less specialized. There are 8 of them, so they can do a lot of work; but then a lot of GPU's are adding multiple processors as well. In the first few generations of Cell, I wouldn't expect the cells to replace the GPU, but it certainly is a possibility; especially later as the Cell starts to scale and have 32 or 64 cells.

As long as Apple was using libraries to help optimize the multimedia behaviors of the OS, why not audio and network streaming? Those are simple tasks that can be easily optimized to use SIMD/VMX -- and that's what Apple did, and gave it the catchy name Core Audio.

XGrid and Cellular Degeneration



A cluster computer is just a set of computers, sitting on a network, waiting for someone to send them something to do. When given a small element of work, they crunch on it, and send back the results. For many types of work, where the data is huge, but the problem repetitive, this parallelism and scalability is really powerful. You need more power, you add more computers to the cluster. You can model nuclear explosions, analyze the human genome, test drugs in virtual space, do weather analysis, or just process radio, audio, video, 3D, speech or other things.

Well what is the Cell Processor? If you think of the Cell Processor as having Cells, and each cell is basically a little computer on a fast network, then basically all you have is a cluster computer. So we get that this is a microscopic cluster, but let's look macroscopically.

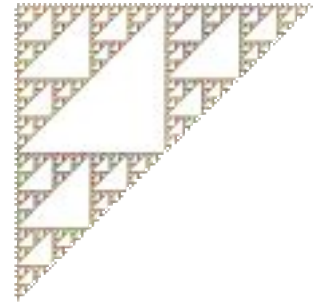
What if you packaged up the same work granule that you were going to send to a node on your network cluster, and sent that to something in the cell processor, or vice versa? What if lots of the work being done on your local computer could be sent to your neighbors computer when he wasn't using it, or split among your entire office, state, nation, or the entire world-wide-web?

On a simpler (higher level) scale that's what Apple's XGrid is all about. Basically you write a program once, and it spreads over the network you define and let's you utilize otherwise wasted cycles and power to do something useful. This idea is not new, I was doing similar experiments a decade or two ago, and I was not the pioneer in this area. In fact, one of the neatest ways to use the Cell Processor is to put a bunch of them in a cluster, and you cut the costs of making a super computer by an order of magnitude.

Sony, IBM and Toshiba are not daft, they're thinking big as well as small. The patent application that relates to the Cell Processor has a lot of references to this kind of distributed network of Cells. There are capabilities in each cell processor to handle Digital Rights Management (DRM) and security, which can be used for privacy management either individually (each cell) or in larger clusters. IBM and others certainly have it on their mind that you can do the same thing Apple is doing with XGrid on a larger scale. Now how well wrapped up this is as a package, how it discovers other machines, how secure it is, and so on are all open questions for now; but they're thinking about it, and they have some serious brainpower working on the problem.

Tipping the scales

Scalability is important in processors. You can make the next greatest design, but Moore's law says that in 18-24 months, you're going to have twice as many transistors to use. If your design is incredibly complex, it may work good for now, but be unmanageable in future generations, or in other words, too hard to scale up. What you want is most of the work being done in the simplest part of the processor, because those are the easiest to scale.



The Cell processor helps scalability a lot. A ton of the work will be done in all the SPE's (helper processors). You can keep dropping those in, and scaling up. Or plucking them out, and scaling down. The same with your memory channels, or the main processing unit, and so on. This first Cell Processor has half as many units in each SPE, and half as many rings as were referenced in the patent application. They're obviously thinking ahead. So this is a pretty modular design, in fact, the most modular and scalable design I've ever seen.

The scalability, for once, doesn't even stop at the box, as they're thinking about standard payloads between boxes. So you have a cluster of clusters. This versatility gives the processor far more legs (longevity) in design than other new processor designs have had, and add to that a few major industry players adopting it, and it is over the barrier to acceptance with a single leap.

What does this mean in 5 years? Well it depends on the acceptance rate and applications. For creative environments with high-latency demands, we will probably have the equivalent of SETI@home being applied to many more applications. (Distributed networks). In lower-latency specialized Super-Computing applications, I can see this processors design rocking their world and being littered all over the place; this brings super computer clusters to much smaller institutions and companies.

In the home, I doubt the ideas will catch on as much; there's administration complexities, setup time, and workloads that probably don't demand it. So unless they can radically simplify setup, and give some returns on enabling those capabilities, I see it more specialized for the next 5-10 years, or more, while society learns to adapt to the capability (the human factor), or wait until performance demands increase to where there's some return (the software factor), or if someone gets a micro-payment system working where you can get returns for the efforts (the greed factor).

Impact on the market

So what will the impact on the market be? I'm not a prognosticator, and no one knows for sure. But this is a specialized design of a generalized processor. The PowerPC is a success, and there's a low barrier to entry to try to use the Cell, and very high reward if it comes close to its hype.

The Cell will be highly successful in its niche, and change the way perceive processors. The real questions then become: Will it go mainstream? What will Apple do? How, and in what time frame? What will that do to the market and Intel, and how quick can Intel (or others) respond.



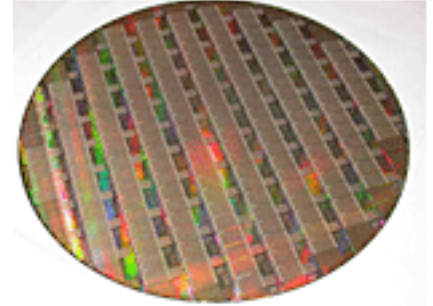
Mainstream or Niche?

So is it going mainstream? I think quite likely. There are a few that are bad-mouthing or at least discounting the design, they imply that this is basically a bunch of RISC chips (or designs) that are 4 to 8 years old, put together with a glue gun. Yeah, so what's your point? In the last 4 to 8 years we've had diminishing returns on highly expensive endeavors that have not as dramatically increased performance as they increased cost. Throwing transistors at single-stream parallelism is expensive and not efficient. Trimming back to some sweet spot just makes sense.

I think some nay-sayers are not doing the math or are just in-love with complexity and the Intel way of doing things. As much as 80% of a processors time is spent waiting for main memory. Focusing on improving the processor is focusing improving 20% of your performance. Focusing on improving memory performance is focusing on the other 80%. Focusing on getting more work done through SPE's passing data to each other and avoiding main memory is avoiding the 80% bottleneck. Small returns in this 80% can easily outweigh large returns in the other areas.

Take Apple and using the G5 processors over their G4's; there's no doubt that the better PowerPC design helped their number crunching abilities (thus speed). But I bet if you looked closely, even more of the gains were not from the improvements in the CPU, but that the G5 had a better memory system. So if you take PowerPC and simplify it, and cut performance with simpler reordering, shorter pipes and so on, you don't take much of a hit and make some gains like decrease the latency or reduce

branch/stall penalties as well. If you can keep the processor fed with high speed memory and clock it higher, you can easily come out way ahead. That's what the PPE/Main processor does, and that's before you even start throwing in the SPE's massive computing power. That doesn't mean I think a 4.6 GHz Cell Processors PPE (PowerPC) is going to outperform a 3 GHz G6, but if it is close, and the SPE's work nearly as well as expected, and they don't saturate your memory bus (slowing down the main processor), then all these hypothetical dream numbers become very real. STI wouldn't be pursuing this if the simulations and tests weren't panning out on some levels. But of course, Apple's interest or disinterest will be based more on analysis versus their needs.



The point is there are cross over points where bigger isn't better, and where going backwards can allow huge leaps forward. Is the Cell Processor this leap? Honestly, I don't know. The numbers released by IBM and Sony certainly imply that it is, and more. IBM has a history of understating its performance, so I have every reason to believe that the optimists' are going to beat the pessimists on this one. So it will probably be king of the cluster computers; I'm sure the lab work is being done to analyze if that translates to being Apple's future.

If it performs like predicted, then there's still issues; superiority doesn't mean adoption. But IBM has reduced the barriers to entry for the processor by offering backwards compatibility, with a very aggressive forward looking design. For Apple, it is about dollars, performance, and marketing opportunity.

What will Apple do?

I have no doubt that Apple is considering the processor, but the question is for what? They're probably working it over in labs, and going to decide based on how it performs, and based on what STI's goals are. If it passes those tests, then we get to decision phase:

- Apple could make a specialized cluster computer for it; something dedicated to its XGrid clientele.
- They could consider using it as an auxiliary processor. Back when NeXT came out with their machine they had a DSP that was included just to help them processor sound, video and other signals to make their computer more useful. There's no doubt this would make a cool support processor; however, the na-

ture of its current design is such that it would probably be starving for data unless you took the entire design as is; which means using it as a main processor not an auxiliary one, or using it as a network computer/cluster.

🔊 Lastly, they could be looking to replace all or some of their machines with this chip.

The things we want to add to computers today and tomorrow are features like Voice Recognition (signal processing), more graphics and 3D (signal processing), facial recognition (signal processing), networking, sound, video, and just moving stuff around are all things that the Cell Processor rules at. So why wouldn't Apple be considering it as a whole new machine architecture? I certainly would be. The memory issue on the current implementation is a red-flag issue, but these things can be fixed or designed around.

The cost benefit analysis will be based on project margins based on processor volumes and transistor counts; with Sony's volumes, I suspect the Cell can kill the G5 or G6 in the long term. So if performance numbers are realized, this isn't like it will be close.

Lastly is the marketing analysis. The Cell is a real hype generating machine; lots of ammunition for your reality distortion field. Intel for years has thrown out specs that few can come close to realizing in the real world, but people buy based on the hype. Well look at the specs; 10 threads of execution at once, 10 times faster than a current machine, 4.6 GHz, 3.2 GHz memory and 8 GHz main bus, 2.5 Megabytes of cache, 234 million transistors; this baby was meant to sell.

So everything points to this being the future of Mac processors. The question left is when?

Now or Later?

It would take Apple a year or three to prepare for using this type of processor change. But who's to say Apple hasn't been doing that for the last year and half, or more. A lot of Tiger's design (Apple next Operating System) seems to have elements that would work well with Cell. Apple could be ready to go ahead this fall, and use this as the G6 instead of the more expensive Power5 based chip, if they've already been secretly working on it. But I suspect that Apple has no reason to be that aggressive, especially right before the Christmas selling season.

Apple is probably waiting and working on things, and seeing if tests are panning out. I would expect a specialized machine from Apple (most likely after the die shrink in early 2006). There's a significant issue in that the XDR memory controller on the current version is designed for 256MB with current chips. Doubling or quadrupling that amount still doesn't make it a viable mainstream processor for Apple. Either they will be able to use the FlexIO to go to some memory bridge controller, or Apple is going to have to wait for a different variant with a different memory control design. I believe either path is viable, so this is not a huge hurdle, but it may indicate that the implementation phase for Apple would be further out.

Lastly, Apple might not want to bet their entire product line on such a radically new design and a single point of failure; so I suspect they'll bring in a new family or class of machine with this chip, and see how it works and see if IBM can meet the volumes and ramp up. If it ramps up and process shrinks down, by fall of 2006, I'd expect to see it spreading over the product line; making 2006 the year that Apple started the momentum to destroy the PC in performance and new classes of functionality in 2007. But I'm being conservative, if it is working in labs well, and Apple has IBM, Sony and Toshiba all committing to this chip; it is not unlike Steve Jobs to take the big risks and be ready to flood the market with these babies and ride the hype that goes with them.

What does this mean to the PC?

The traditional PC is already under attack. IBM and others are making specialized machines on the high end, servers, and niches. Linux is making the Wintel platform far less important that it once was. And the same can be said of OS X and Apple. Most people that try Linux or for servers or OS X for the desktop or server, prefer them to Windows. So there's far less holding people back on Wintel, less barriers to entry, and more reward. So the market is ripe for another phase of competition. Now mix in a radically new processor, that could empower new classes of Applications, and this is a major threat to Wintel that could break the market wide open again.



The PC has peaked, the question is how long can they hold on before they go down, and how far down will they come. But don't get ahead of yourself; I'm not talking Apple, IBM and other taking over the entire market in a year. The PC market is huge, and there's a lot of infrastructure and money built into the status quo. I could see Apple doubling to 5% this year with just the Mac Mini. Throw in the Cell processor

and improvements in Tiger, and we're talking about another 5% of the market in 2006, with momentum gaining for 2007. Throw in IBM, Sony and others picking at the niches (large niches), and we could see twice those numbers erode. 20% of your market in a couple years, and a huge momentum shift is a big deal. If that happened, I have full faith that Microsoft would have Windows (Longhorn) running on the cell, and that would completely open up the playing field with the best technology winning; something that would put the fear of God in Intel.

How can Intel respond?

With the Cell Processor, everything is in place. This processor can scale much easier and cheaper than x86 processors. Face it, you're getting 10 threads of execution in the place of two for the next version of the Pentium and 8 of those processors are 128 bit (instead of a 32/64 bit hybrid). Each processor is a much smaller core. This has a ton more potential if the software can take advantage of it. History has shown that Apple and IBM and Sony in their respective areas can all take advantage of it. So if they aren't sweating at Intel, they should be.

However, never under-estimate the huge amount of resources that a company like Intel or AMD can throw at a problem. If they have to copy the design, they will have the advantage of hindsight, and borrowing STI's successes and not the failures. They can do exactly what IBM did; rip their x86 core down to its basics, add in specialized DSP/RISC cores, or even license the cell processor cores from IBM or Sony and make the x86 "Cell compatible", or AMD could. It will still take time; say 3 years. And that's assuming they fully get the magnitude of the problem, and react to it. But they'll just have to stall, cut the price of their chips, make more multiprocessor based designs to buy time. But they have the money to buy time (or at least lease it).

Intel has had a few stellar flops of late, like the Itanium, and they are either eager to react by immediately starting their own versions, or are completely gun-shy. The politics of denial and the status quo has brought down more than a few companies the size of Intel. But I think Intel has enough healthy paranoia to respond in time, and enough experience at marketing inferior technology and stalling that even if Cell is everything it promises to be, Intel isn't going anywhere. Yet, I also believe that the market in 5 years will be more open than it is today. And that's a good thing.

Time will tell.

Bibliography

Cell Patents: 6,809,734, 6,526,491: [patent application](#), [patent application \(update\)](#)

Pending Patents:

- 20030229765, "Memory Protection System and Method for Computer Architecture for Broadband Networks," December 11, 2003
- 20020156993, "Processing Modules for Computer Architecture for Broadband Networks," October 24, 2002
- 20020138707, "System and Method for Data Synchronization for a Computer Architecture for Broadband Networks," September 26, 2002
- 20020138637, "Computer Architecture and Software Cells for Broadband Networks," September 26, 2002.

IBM: [IBM](#), [STI cell processor](#), [IBM, Sony and Toshiba unveil Cell processor](#), [Cell processor-based workstation prototype](#)

Sony: [Sony USA](#)

Toshiba: [TOSHIBA](#), [TOSHIBA Semiconductor Company](#)

RAMBUS: [XDR Memory Interface](#), [FlexIO Processor Bus](#)

Apple: [Core Audio](#), [Core Image](#), [Xgrid](#)

Microprocessor Report: [In-Stat - Processor Watch](#)

Microprocessor Report: [New Patent Reveals Cell Secrets](#) (\$50)

Cell Processor News: [Cell Processor: News](#)

The Register: [Multi-OS Cell CPU tops 4GHz](#), [The Cell chip - what it is, and why you should care](#)

The Register: [The Cell Chip - how will MS and Intel face the music?](#)

The Register: [Sony, IBM, Toshiba team on broadband supercomputing CPU](#) | The Register

EET: [Details trickle out on Cell processor, 'Cell' processor at 90-nm, : Interfaces one key to Cell success, claims Rambus](#)

Electronics Weekly: [IBM, Sony, Toshiba present Cell](#), [Cell processor uses Rambus I/O](#)

Blanchford Info: [Cell Architecture Explained](#)

The Inquirer: [Playstation3 architecture revealed](#), [Sony seeks out Rambus for Playstation3 technology](#)

Hardware Analysis: [Hardware Analysis - Sony, Toshiba License Rambus' Interface Technologies](#)

National University of Singapore: [SVU: A Beginner's Guide to Vector Processing](#)

Team XBox : [Cell Processor Unveiled - Xbox](#)

Ars Technica: Introducing the Cell Processor: [Part I: the SIMD processing units](#), [Part II: The Cell Architecture](#), [DRM notes](#)

New Scientist: ['Supercomputer-on-a-chip' microprocessor revealed](#)

Real World Technology: [ISSCC 2005: The CELL Microprocessor](#) - **[transistor details]**

PC Watch: [ISSCC 2005 Presentation 2 Slides](#) (Japanese site, english slides)