Comparative study of the Pentium and PowerPC family of micro-processors

Chan Kit Wai and Somasundaram Meiyappan

1. Introduction

We live in an era of computers. From the smallest embedded system to the complex servers that take care of the world economy, we need microprocessors to run them. As time passed by, applications needed more processing power and this lead to an explosive era of research on the architecture of microprocessors to make use of all that the latest semiconductor process technology has got to offer. Different micro-architectures, the core of the micro-processor, debuted over the years. In this report, we study and compare two processor families that have been at the core of the world's most popular desktops – the Intel's Pentium family and IBM/Motorola's PowerPC family.

2. Introduction into the processor families

2.1. Introduction to IA-32 microprocessor family

2.1.1. History of IA-32 family

The most popular microprocessors in the history of man-kind so far had come from Intel. It all really started with 8088 processor and then the 8086, 80186 and 80286processor. The now famous *x86* instruction set actually started evolving from the 8086 microprocessor. This family of x86 processors started off as 16-bit CISC machines. The first real 32-bit processor from Intel was the 80386 processor. Intel introduced 80486 with a 5-stage pipeline and then came the Pentium (or 80586) with a second integer pipeline. And this really set the stage for super-scalar IA-32 family. Pentium family was extended with some multimedia extensions (MMX) which is actually a set of new instructions to perform the same arithmetic operation on a set of packed integer values. Pentium Pro, Pentium II and Pentium III used a micro-architecture called the P6 micro-architecture and used a 10-stage pipeline with Pentium III adding the SSE extensions. This helped Intel to increase the frequency of the processors because the deeper the pipeline for an instruction, the lesser it takes (or the less time we have) to complete a stage in the pipeline since the work to do in a stage is now half as before. Of-course, the reduction in the minimum feature size of the transistors also helped.

The latest additions to this line-up of IA-32 desktop processors are the Pentium 4, based on the NetBurst micro-architecture and the Pentium-M, reportedly based on the P6 micro-architecture. The Pentium 4, which would be our main point of focus from the IA-32 family, was designed to offer the highest level of performance while the Pentium-M (part of the Centrino set) was designed to offer a good performing; but a lesser power-hungry processor for mobile computers like notebooks.

2.1.2. Instruction Set Architecture

The IA-32 family (as mentioned above) has traditionally been designed on the principles of CISC (Complex Instruction Set Computing). So, these processors differed from simplistic approach taken the RISC machines like MIPS and PowerPC. Some of the traditional characteristics of CISC machines are

- Small register file
- Complex instructions that perform more than one or two simple operations
- Some systems like the IA-32 family had variable length instruction encoding schemes.
- Complex addressing modes like memory-indirect addressing that might cause multiple memory access from a single instruction

The IA-32 family had all these characteristics. Something that some CISC machines, like 68K family from Motorola, had/have; but the IA-32 processors do not have is memory-indirect addressing, in which an instruction is capable of fetching data from an address in the memory that is computed based on another value stored in the another address in the memory. But IA-32 processors do have instructions to move a string, available at one memory location, to another memory location. The ISA also has instructions that could save a lot of work when programming the infrastructure for shared memory multi-processor systems made of Pentium 4 or Xeon.

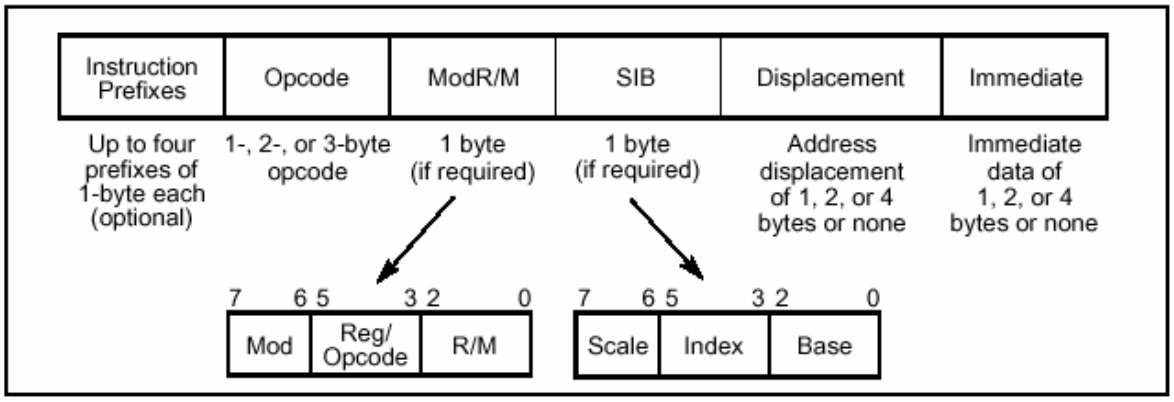

Figure 1: IA-32 general instruction format

The first biggest enhancement to the ISA came in the form of MMX (Multimedia extensions) that enabled IA-32 processors to perform SIMD (Single Instruction Multiple Data) integer operations. Though SIMD was introduced to only integer operations 64-it packed structures, enhancements like SSE (Streaming SIMD Extensions introduced by Pentium III) and SSE2, introduced by Pentium 4, added SIMD support for floating point operations and 128-bit registers to this ISA. Figure 1, above, shows the general instruction format of IA-32 processors. For more information, please refer to *IA-32 Architecture Software Developer's Manual.*

2.2. Introduction to the Power PC family

The PowerPC started off with a RISC architecture that defines all instructions to have a fixed instruction length and that all instructions are simple. It basically divided the instruction set into 3 sets, namely the user instruction set, virtual environment architecture and the operating environment architecture.

The PowerPC instruction set has several salient features compared to other RISC architectures. First, the instruction set was organized around the idea of superscalar instruction dispatch. Second, several compound instruction are added to reduce the instruction path length. The third key feature is the removal of the 'branch and execute' capability because the 'branch and execute' is found to be ineffective for advanced superscalar machines. The details are found in [Keith94].

Another interesting feature about the ISA is that, the complier provides some hints to the pipeline architecture about the branch instruction. In a conditional branch instruction, a special field "BO" in the instruction provides a hint about whether a conditional branch is to be taken. This form of compiler aided branch prediction mechanism which helps to save hardware resources and at the same time reduce the chances of branch penalty incurred in the pipeline.

It is also noted that the ISA provides synchronization instructions for the semaphore operations of the system. Two special instructions are provided to enforce memory synchronization that is often required in advanced operating system.

Taking the PowerPC 750 as a reference, the number of floating point instructions is found to be relativity large and comprehensive. Being a RISC originated processor; the number of instructions should be kept low. Instead, a complicated ISA is implemented in 750, making the job of the hardware architecture with increasing complexity. However, from another point of view, it simply the software development and reduces the program memory as a whole.

3. Details of the Micro-architectures

The term 'micro-architecture' is used to refer the internal design of a micro-processor. The performance of microprocessor for a given semiconductor process technology is determined by the micro-architecture of the processor. Modern architectures use the concept of pipelining to extract higher throughput and employ more execution units So that more operations can be done in parallel.

3.1. Micro-architecture: IA-32 processors

In the Intel processors, evolution of modern micro-architecture started with the introduction of the second integer pipeline in the Pentium processor. But the significant jumps in the micro-architecture of IA-32 processors came with the introduction of the P6 family and then the NetBurst micro-architecture. The following section will give the reader a brief understanding of the P6 micro-architecture before the paper describes the latest NetBurst micro-architecture.

3.1.1. Micro-architecture of the P6 family

The P6 micro-architecture introduced several new concepts to the IA-32 processor family. Some of the key novel features are

- 3-way super-scalar design with a speculative fetch unit that is controlled by a branch predictor
- 10-stage execution pipeline (12-stage pipeline including the retirement unit)

- Dividing some complex IA-32 instructions into some or several RISC style operations referred to as μ-ops (micro-opeartions).

Other features include

- Out-of-order execution
- Internal registers are coupled with re-order buffers
- Dedicated L1 Data cache and Instruction cache to store data and instructions from memory
- Two floating point units that are floating point/MMX/SSE/SSE2 capable

The figure below shows the 10-stage execution pipeline of the P6 micro-architecture.

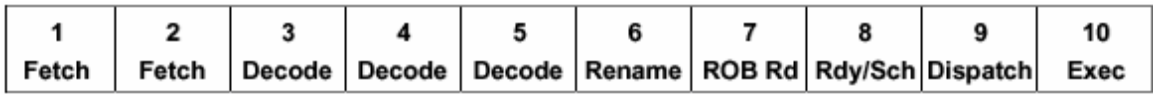

Figure 2: P6 "master" pipeline

One of the biggest improvements over the earlier architecture in Pentium is the use of μ-ops that eliminate the pipeline dedicated decoding stage as in Pentium. Since the μ-ops are RISC style operations, it now becomes more pipeline-friendly. It is interesting to note that a μ-op is estimated to be 111bit wide which is much larger than most of the instructions themselves. [6.13]

μ-ops stored in reservation stations are allow for out-of-order execution unit for effective schedule and dispatch the μ-ops for execution as execution units become free and as the instructions are deemed to be ready for execution. The use of reservation stations increases the probability that the out-of-order execution core would find independent μ-ops for dispatch and execution.

The use of μ-ops also introduces little complexity in the retirement, which is the act of committing the results of an instruction, phase. Since the out-of-order execution unit actually executes the μ-ops and since the program/software does not have the notion of μ-ops, the retirement unit should take care to retire the set of μ-ops that make an instruction together and allow interrupts/exceptions to take place only at instruction boundary.

3.1.2. NetBurst micro-architecture:

The NetBurst micro-architecture is the latest high-performance micro-architecture used in the Pentium4 and Xeon processors. The problem statement for design team seem to have been – *design a micro-architecture for high performance and capable of running at a very high clock.*

Given the process used to manufacture a chip, the definite way to use a higher frequency clock in the processor is to increase the depth of the pipeline. And that is the choice that the NetBurst micro-architecture team has made. It has a 20-stage hyper-pipelined pipeline. But such a deep pipeline has its pit-falls. This section would describe the pipeline itself and the methods used to realize the potential of the deep pipeline and out-of-order execution unit.

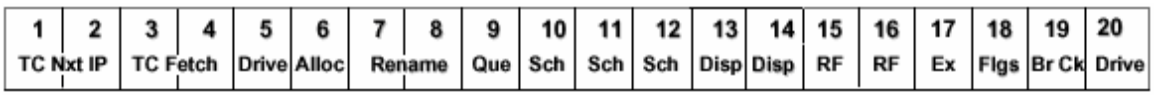

Figure 3: Pentium 4's pipeline

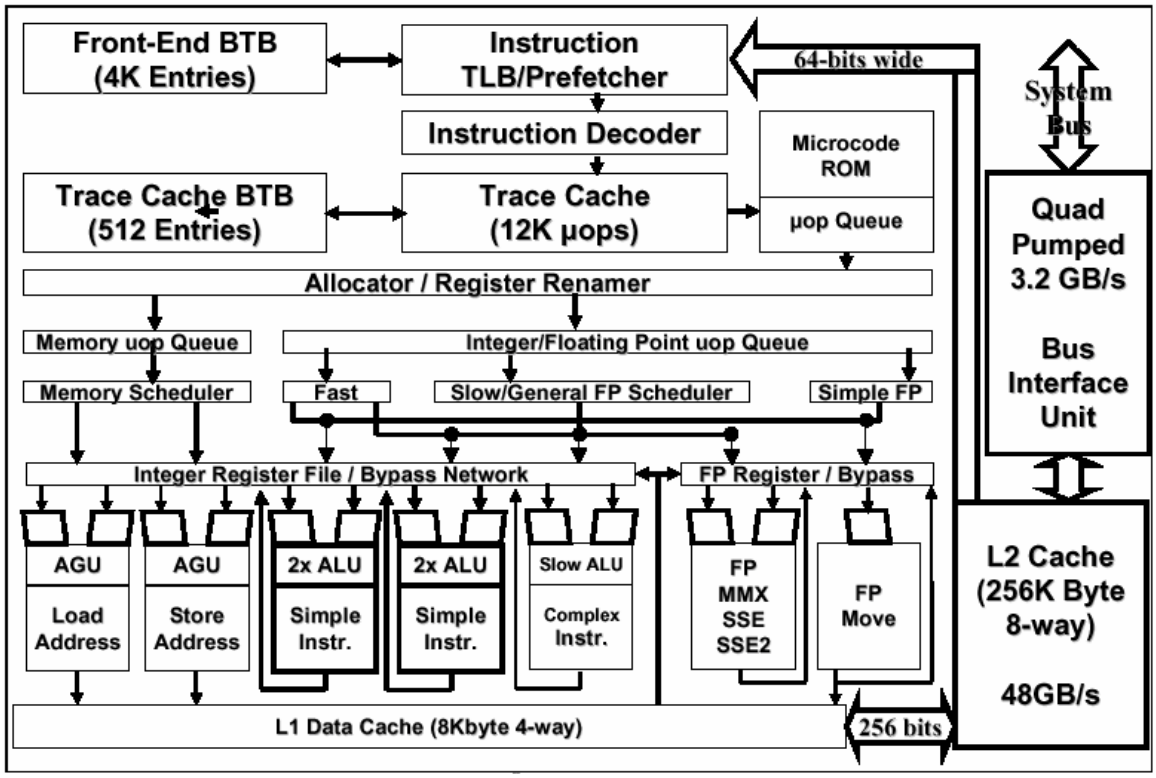

Figure 4: Conceptual view of execution in Pentium 4

Some of the novel and notable features introduced in the NetBurst micro-architecture are:

- Decoupling the instruction decoding stage from the pipeline
- Transforming the L1 instruction cache into a bigger; but very useful *Advanced Trace Cache*
- Advanced branch prediction using a large branch history buffer (BTB – branch target buffer)
- Internal register file is decoupled from the reorder buffer (ROB) entries
- Rapid execution which virtually executes some integer operations in ½ clock cycle
- Hyper-threading, that allows one execution core to act as two logical processors so that the available execution units are kept as busy as possible
- And a very deep pipeline

Hyper-Threading is the name Intel has given to Simultaneous Multi-Threading (SMT). And it is the first commercial instance of SMT. The motivation is that when executing a single thread of execution, the processor is not able to keep all the execution units occupied with work all the time. So, these free execution slots can be utilized by executing another thread in the hardware. This means that the second thread should have an active context (its own set of registers) available along with the context of the first thread. So, two parallel streams of instructions can be

executed at the same time whenever possible. Further discussion on hyper-threading is beyond the scope of this paper.

Just like the P6 micro-architecture, the NetBurst micro-architecture also has a front-end that tries to feed the out-of-order execution core with more instructions to find scope of parallelism. And just like the older architecture, *NetBurst* also speculatively fetches instructions for decoding and execution. But unlike the P6 micro-architecture, the NetBurst decodes the instruction into μ-ops and stores these μ-ops in a cache. Since the decoded instructions are cached and stored as a trace (the predicted sequence of execution which also involves branches), loops would benefit a lot as the instructions do not have to be decoded every time after a fetch from the cache like in P6. The use of the *Trace cache* turns the decoupled Penitum 4's pipeline into a pipeline that deals only with μ-ops and not the complex instructions of the x86 instruction set. From the size of the *Trace cache* in the die against the on-core L2 cache, it is estimated that the *Trace cache*, which can hold 12,000 μ-ops, is 96KB in size and a μ-op could be around 64-bit wide [6.13]. Intel does not publish any information on this though. But some complex instructions still are not completely decoded and placed in the trace cache. The microcode ROM takes care of furnishing the μ-ops for those instructions that are marked so in the trace cache. Please note that since the μ-ops in the trace cache are stored as a sequence in the program order, only instructions that would be executed will be brought inside the trace cache. Hence, all instructions brought inside the trace cache would execute unless there is a branch mis-prediction. This keeps the out-of-order execution core happy.

Please refer to the following table for information on what each stage of the pipeline does.

Stage	Name	Function
1-2	Trace Next Instruction Pointer	Computes the next pointer to fetch μ-ops from. The computation is based on the branch prediction and is governed by the Trace BTB (that is a subset of the larger front-end BTB)
3-4	Trace cache fetch	Fetches μ-ops from the address computed by the previous stage
5	Drive	A stage to pass the fetched μ-ops to the next stage. Please note at one cycle only 1 load, 1 store and 1 non-memory operation can be driven to the allocator (3 μ-ops).
6-8	Allocate and rename	Allocations are made in the various buffers like the μ-op queue, registers, etc… Pentium4 can have up to 126 instructions active (in-flight) at a time and have up to 48 loads and 24 stores allocated in the machine at a time.
9	Que	This stage places the μ-ops into the one of the two μ-op queues viz., a memory operation queue and a non-memory operation queue. *Dependence checking is done in the queue itself.*
10-12	Scheduler	Depending on the availability of the execution unit, μ-ops are selected from the queues. *Please note that the two queues are strict FIFO; but each queue is independent of each other thus enabling out-of-order execution.*
13-14	Dispatch	The selected μ-ops are dispatched are dispatched to the respective execution units. *Please note that there are two separate schedulers that work independently each with two dispatch ports. We can execute as much as 6 μ-ops every cycle with 2 μ-ops every cycle to each of the two ports in the ALU.*
15-16	Register File	These stages are used to either read data from the register file or read the operand from that was bypassed by the previous

		operation.
17	Execute	The actual execution takes place here.
18	Flags generate	Generates condition flags like Negative, Zero, Positive, etc…
19-20	Branch Check and Drive	This stage is used to compare and update the branch target buffer, which holds the history of the branches.

Since the IA-32 processors (unlike Itanium or PowerPC) have only 5 general purpose registers, the register pressure will be high. This would cause register write-backs that are to memory stores of values in registers and when that variable is has a *use* again in the program, there would be memory load. And as one can see above, with a possible 24 stores active at a time, there would have been lots of loads on the same addresses of the stores. But unfortunately, one of the main reasons why we have so many stores pending at a time is because they are either not yet retired or they are not committed to the memory yet. Since loads are high latency operations and also because of the fact that we are executing all instructions speculatively anyways, the values in the store buffers can be forwarded to the load instruction when it needs. This helps the execution to move forward as much as possible without having to wait for the store to be committed to the memory. This feature is called *store-to-load forwarding* and improves performance.

It can also be noted from figure 4 that we have only one SIMD capable unit instead of the two that was available in Pentium III. Another notable feature is the mention of '2xALU' in figure 4. Though there actually is only two integer execution units (apart from the slow integer unit), this shows that the ALU for simple instructions can virtually complete an operation in ½ clock cycle. Actually, the ALUs, say the adder, performs only 16-bit operations in ½ clock and takes three ½ clocks to complete the entire operation. So, we can look at this as a 3 stage pipe with each stage taking ½ clock cycle. Since these ALU units are fed by register files and the output also go to the register file, we would need a really fast register file and a good bypass/forwarding network to forward the result of an arithmetic operation to its dependent instruction even before the value is written into the register file. Such forwarding is needed because of the fast execution in the ALU.

As noted above, the interesting part is that the execution pipeline looks just like a deeply pipelined RISC machine. Though this deep pipeline can be its biggest killer of performance when branch happens, the supposedly good branch prediction scheme deployed in NetBurst architecture and the use of trace cache can help us to reduce the occurrence of the mis-predicted branch. The branch predictor, when we cannot dynamically predict the branch target, assumes that all forward branches are not taken (benefits the 'then' part of the 'if' condition) and also that the backward branches are taken (benefits the loops). But in this micro-architecture, a mis-prediction could still cost us dearly, as the fall-through instructions of a branch might not in the trace cache as the trace cache only has the predicted trace. So, the instruction fetch, at best, have to happen from the costlier L2 cache. But in the P6 micro-architecture, some part of the fall-through code could be in the faster L1 cache itself resulting in faster recovery. This seems to be the only disadvantage of using the trace cache over having a L1 instruction cache. Of course, another big source of performance impact on a branch mis-prediction is the deep pipeline itself which takes a longer time to hit the maximum throughput state. But with the branch prediction that is believed to be 3 times more accurate and with the trace cache having lots of other advantages, it is worth the risk.

3.2. Micro-architecture: PowerPC family

The first PowerPC is design based on the IBM's POWER (Performance optimization with enhanced RISC) architecture used in IBM's RISC system/6000. It inherits the characteristic of POWER architecture that incorporated fundamental RISC features of fixed width instruction set, register to register architecture, a few simple memory-addressing modes, and simple hardwired

instructions. It is designed with the aim to minimize the total time required to complete a task. The total time is the product of 3 components: path length, number of cycles needed to complete an instruction, and cycle time.

The PowerPC clearly distinguishes architectural features from implementation features. This classification is necessary to maintain a consistent basic architecture organization as well as the emphasis of different variants of implementation. The architecture is group into 3 parts or books and the fourth book includes all implementation features. The detailed specifications of the 4 books are discussed in [Keith].

The unique feature of the PowerPC is that, it defines a 64bit architecture that is a superset of the 32bit architecture, providing application binary compatibility for the 32 bit applications. This approach assures upward compatibility for all applications. The concept of the 64bit extension is described in [Keith].

The Motorola PowerPC 601 is the first implementation of the PowerPC family of reduced instruction set computer (RISC) microprocessors. The architecture aims to improve performance by exploiting instruction level parallelism by incorporating more resource in specific areas. It supports out of order execution with the multiple independent EUs. The architecture integrates three execution units to increase the level of parallelism. Each EUs having their own set of register resources to reduce the communication and synchronization required between the EUs. The details can be found in [MPC601].

The instruction unit contains an instruction queue of 8 instructions that is refilled in a single cycle from the instruction cache. The lower half of the instruction queue is analysis by the BPU for any potential branches.

The BPU performs condition register (CR) look-ahead operations on conditional branches. Branch folding is used for to eliminate a branch instruction and issue the next instruction. Branch instructions that are executed are decoded but not issued for any further processing. The BPU attempts to resolve branches early which help to achieve the effect of a zero-cycle branch in many cases. In the case of a mis-predicted branch, the instruction fetcher flushes all predicted path instructions that are issued and reissue instruction from the correct path.

The pipeline structure in PowerPC demonstrates a fully interlocked instruction pipeline, which permits proper operation of the pipeline without introducing no-ops instructions for unused pipeline delay. The design helps to isolate the software from the dependence on the underlying hardware structure from future compatibility issues. In additional, it reduces program length, instruction bandwidth and cache entries. Additional information about the POWER and PowerPC 601 architecture can be found in [IBMHist] [IBM601]. A comparison of the PowerPC 601 and the latest PowerPC, 970 is also evaluated in the following section.

The latest PowerPC, the 970 is based on POWER4 architecture. PowerPC 970 has significant increases the data, address bus and EUs to 64bit, making it a true 64bit machine. It is also the 1st PowerPCs with AltiVec extension to specifically handle multimedia applications.

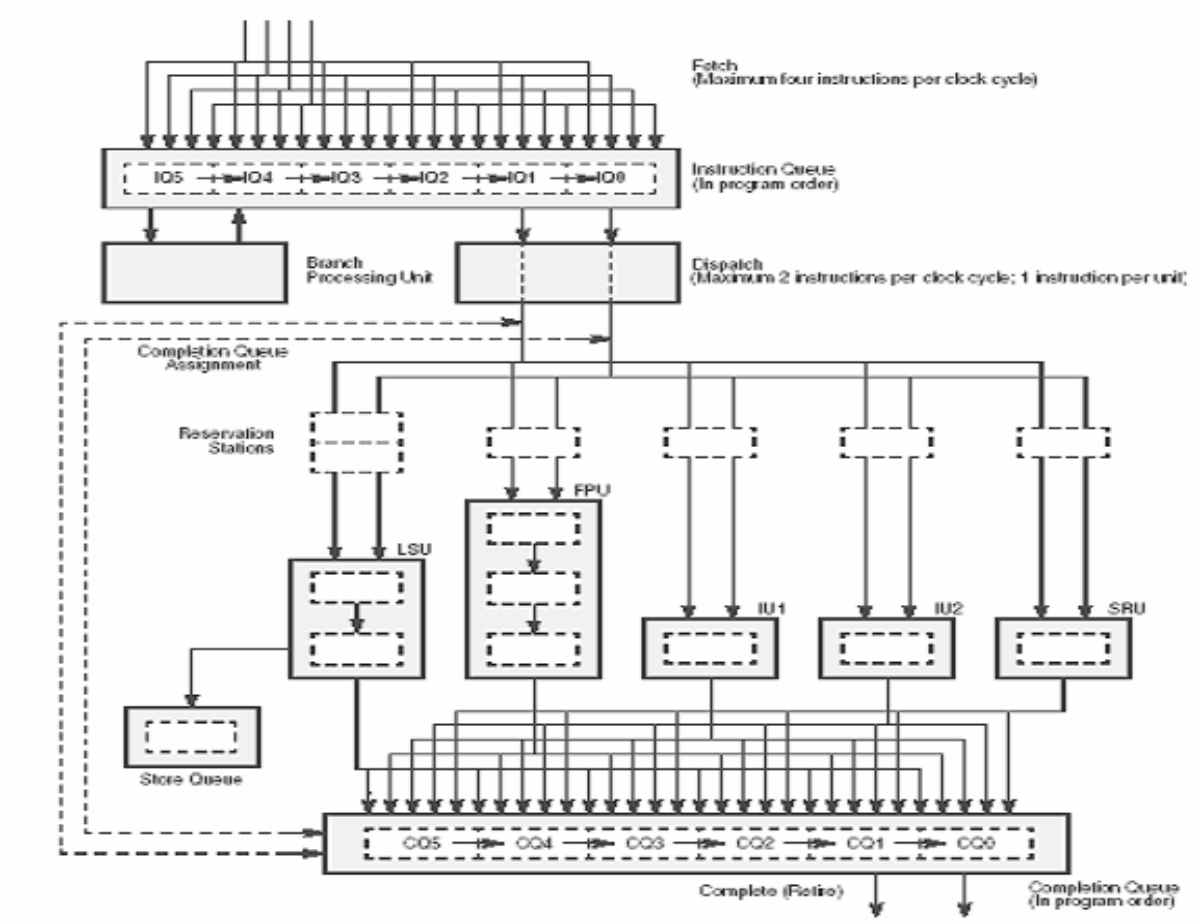

Figure 5: PowerPC 750 pipeline with buffers

The deep pipelined design potentially increases the parallelism as well as the perils of a mis-predicted branch. Hence, it is observed that a substantial branch prediction and speculative logic are implemented that is necessary to increase the accuracy of the predicted branches. The 3-component branch prediction logic details can be found in [AppleG5].

The 970 fetches 8 instructions from L1 cache which shows a double increase in the fetch rate compare with the 750. Further information of 970 can be found in [Tom02] [Pow4] [Jon03] [AppleG5].

From the survey conducted on the PowerPC series, it is can be seen that some of the architecture must "fall back" in technology to cope with the market demands. Introducing an architecture that does not match with other technologies (e.g compilers, OS and applications) results in a failure. For instance, the 64 bit 620 is introduced in 1991 never succeeded. Hence, designers are forced to "fall back" to 32bit architecture until the market is ready for 64bit architecture. The history and the range of PowerPC series can be found in [PowHis].

	MPC 601	IBM 750	IBM 970
Clock speed	50-135Mhz	300-500Mhz	1.4-2Ghz
Pipeline Stages	4	4	16(int), 21(float), 25(SIMD)

# of instr fetch	8 (a cache block)	4	8
# of instr dispatch	2+1 BR	2+1 BR	4+1 BR
# of EUs	3	6	8+4 velocity engine
Register files	GPRs, FPRs	GPRs, FPRs	64bit wide GPRs, FPRs, 128 bit VRFs
Reservation station (LSU)	N.A	2	not published
Reservation station (FPU)	1	1	not published
Reservation station (IU)	0	1	not published
# LSU stages	N.A	2	not published
# FPU stages	6	3	not published
# IU stages	5	2 IU, 1 stage	not published
# of retired instr per clock	3	2	5

Table 2: Comparison of the 601, 750 and 970 pipeline structure.

PowerPC 601

The PowerPC 601 pipeline architecture consists of 4 stages with 3 separate EUs. Each EUs have dedicated register files (GPRs and FPRS) to maintain operands and allowing the operations of both units to execute simultaneously without interference.

The FPU is pipelined so that most single-precision and many double-precision instructions can be issued back to back. It contains 2 additional, instruction queues allowing the instructions to be issued from the decode stage even if the FPU is busy. The introduction of such "buffer" helps to relieve the blocking of floating point execution and as well as allowing other non-floating point instruction to be issue to their respective units.

Interestingly, the Integer instructions are only dispatched from IQ0 whereas branch and floating point instructions can be dispatched from IQ1-IQ3. Integer instruction is only dispatched in IQ0 in order to maintain an ordered flow of instruction through the integer pipeline, at the same time ensuring the completion in order. In contrast, the branch and floating point instructions are executed out of order. These instructions are tagged to instruction in integer pipeline for tracking purpose. The completion unit checks the tag and reorders the instruction sequence to ensure a completion in order.

PowerPC 750

The aim of the PowerPC (G3) 750 is to improve floating point and data transfer performance. Although it remains as a 4 stages of pipeline, the EX stage increases the no. of EUs to 6. In additional, it is also noted that the floating point unit is further divided into 3 stages to implement a floating point pipeline within the FU. The 750 separates the load/store operation into an independent EU. The additional of resource are attempts to segregate tasks into concurrent process to ensure a smooth flow in the pipeline. The 750 retires instructions and dispatch instructions both at rate of two. This is especially true to maintain the retire rate equal or lower than the dispatch rate. Otherwise, the completion unit will be under utilized or idle most of the time.

An interesting difference between the 601 and the 750 is the reduction of instruction fetch rate. The 601 has an instruction queue of 8 and it fetches 8 instructions per cycle from the cache line. Surprisingly, the 750 only fetch 4 instructions! The reason can be explained as follows. The 601 can only fetch 8 instructions after the last instruction is dispatched, hence for every 8 instructions, an unknown delay is incur when those 8 instructions are being fetched. As a result, the performance of the pipeline will be affected. In the 750, a fetch rate of 4 instructions is chosen

instead to balance between the time required to fetch from cache and the instruction dispatch rate. While the 2 instructions at the bottom of the queue are being dispatched, new instructions are being fetched and place on top of the queue, hence creating a kind of pipelining in the queue.

It is also observed that several attempts are made to ensure that the pipeline maintains a high dispatch rate. Efforts to increase the no of EUs, introduction of pipeline within FPU and LSU and increasing the number of reservation stations are made. To reduce dispatch unit stalls due to instruction data dependencies, the 750 also provides a single entry reservation station for individual EUs. For additional information, pls refer to [IBM750].

PowerPC 970

The latest series of the PowerPC is introduced in 2002. The PowerPC 970 has significant deeper pipeline compare to the rest of its family. It is interesting to note that the architecture has a different number of pipelines depending on the type of instructions. There are 16 stages of Integer, 21 for floating point and 25 for SIMD operations. The variable pipeline stages objective is to achieve a consistent throughput for all the varying computational loads for different instructions.

The 970 inherits the POWER4 wide superscalar execution unit. It has 2 FPUs, 2 SIMDs, 2 LSUs, a Branch and a condition-register unit. Although it have 8 EUs, it is still technically regarded as 5 way superscalar, since the instruction retired rate is 5. The execution units are designed with additional resources (e.g 2 parallel ALU units) to ensure a smooth flow of instructions throughout the entire pipeline, reducing control hazards that lead to stalls. The number of reservation station and stages for each EUs are yet to be published. However one could estimate the number based on the POWER4 architecture.

We can see that architecture designers are exploiting the necessary hardware resource that is required by the application in the market. The AltiVec was first introduced in Motorola's G4 processor and IBM later put this in the PPC 970 processor to meet the increasing computation demands of multi-media applications. It is therefore emphasizes that additional of multi-media instruction set to be included into general purpose computers. The AltiVec vector-processing unit has a 128bit wide datapaths and 32 registers of the same width to support multi-media operations.

Another special feature about the 970, is the grouping of internal operations. The 970 breaks instructions into internal operations (IOPs) and these IOPs are packaged into a group of 5 before dispatching into the execution core. The IOPs are grouped together for tracking purpose for the completion of in-order write back process. Instead of tracking individual IOPs, the completion unit tracks the IOP grouping [Jon03], hence reducing the overhead associated with the tracking and reordering of instruction in the deep and wide pipeline.

4. Penitum4 Vs PowerPC 750/970

The IA-32 family and the Power/PowerPC family are two processor families that started off with two different design philosophies. The IA-32 processor's ISA was originally based on the philosophy of CISC (Complex Instruction Set Computing) and PowerPC was based on RISC (Reduced Instruction Set Computing). So, in the earlier 90s, the RISC based processors which were more suitable for pipelining and produced a better throughput because of the following reasons

- More registers that are visible to the program than in the CISC machines like IA-32. So, we have fewer memory access for data
- Relatively smaller instruction set (and a fixed length instruction encoding) which also leads to a lesser effort in decoding than the IA-32 family, instructions of which can be of variable length.
- With a fixed instruction length and with a careful designed instruction cache line, RISC machines also lead to a more predictable instruction cache performance than CISC processors

But there are other advantages that CISC processors had like the following.

- With complex instructions, a CISC machine might be able to do a set of simple instructions by execution only one instruction in its ISA; but traditional RISC machines would need a few instructions to do the same. This means that CISC processors would have better code density that their RISC counterparts.
- More addressing modes normally provided by the CISC processors mean that not only we have a more flexible way; but this further helps in reducing the number of instructions (atleast in the user program) needed to perform an operation. This is more important as most of the CISC machines like Intel's are accumulator based machines.

With improvement in semiconductor process technologies, more transistors could be packed into a die of the same size. This enabled the RISC machines to bring in the basic CISC idea of having instructions that perform more than on simple operation. And it also enabled traditional CISC processors include some of RISC characteristics. Some like the Pentium Pro and its successors have become a RISC machine at the core; but a CISC machine at the ISA level. Chronologically, the design of Pentium 4 is sandwiched between the designs of Power PC 750 (its closer relative with some enhancements is in Power Mac G4) and Power PC 970 (Power Mac G5). Now, let as compare the processors that represents the world's most popular desktops viz., x86 PC and Apple Power Mac.

As mentioned earlier, the micro-architecture of the Pentium4 processor looks the same as a RISC machine. But one of the most notable differences between the two processors in question is the depth of the pipeline. The Pentium4 pipeline is a 20-stage pipeline as compared to essentially a 4-stage pipeline in PowerPC750. It would be interesting to note that the latest PowerPC970 has at least 15 stages in its integer pipeline to 25 stages for its SIMD (AltiVec / Velocity Engine) pipeline. And one of the main reasons that PowerPC970 (which is a 64-bit processor) has been designed this way is apparently for reaching higher clock speeds and to close its gap with Pentium 4.

One of the significant advantages that the RISC processors had over the CISC processors is the ease of decoding instructions. This is now no longer valid with Pentium4 around and with Power PC adding some complex instructions to its ISA. As noted before with the NetBurst micro-architecture, with the instruction decoder decoupled from the main pipeline through the use the *trace cache*, that advantage that RISC enjoyed has been nullified. Interestingly, the Power PC 970 processor dedicates the first 9 stages of its pipeline for fetch and decode. But Pentium 4 uses only the first 4 Stages for its fetch and decode. This is made possible by the use of *trace cache* in Pentium 4. Also, the PPC970 processor breaks complex instructions in to instructions that can he executed in a single cycle. This would look just like Pentium Pro and its IA-32 successors dividing the instructions provided by their ISA into μ-ops for execution. But we feel that for very

high frequency designs such division of instructions, be it CISC or RISC architecture based processor, is going to inevitable in the future.

PowerPC750 also tries to access the operands when issuing the instructions or when in the reservation stations. But Pentium4 access the register file for operand fetch only after the dispatch. This increases the size of a line in the reservation station for PowerPC750. It will be interesting to see this part of the implementation in PowerPC 970 which can have as much as 215 instructions in-flight compared to the 126 in-flight instructions in Pentium 4.

PowerPC also introduced another interesting approach to branching. In PowerPC750, the branch prediction unit can be given some hints as a part of the branch instruction itself to specify the most likely option (is the branch taken or not?). It also has a special branch processing unit (BPU) unlike the Pentium4. That BPU is also capable of resolving branches very early and this helps us to recover the mis-prediction pipeline faster. PowerPC 970 extends this even further with a 3 separate 16K branch prediction buffer compared to the 4K branch prediction buffer in Pentium 4.

PowerPC had and still does use individual reservation stations for each EU. But Pentium 4 uses a separate queue for memory loads and stores and another queue for all other kinds of instructions. It should also be noted that Pentium 4 takes a longer time to schedule and dispatch (5 stages) to the EUs.

PowerPC still enjoys a particular advantage of Pentium 4. PPC has lots of user visible registers and this can reduce the memory access considerably as procedure or function local variables can mostly be kept in the registers itself. Of course, wider registers (64 bit) in PPC 970 means that structures of that size can now be stored in a single register. The introduction of *Hammer*, which has 64 bit extensions to x86, by AMD would mean that we would begin to have Intel like machines with 64 bit registers. But we feel that Intel is not likely to make a 64 bit version successor to Pentium 4; but rather promote an Itanium 2 derivative like *Deerfield*.

Pentium 4 has a strict FIFO order of dispatch from a queue. This means that if an instruction in the front of the queue cannot be dispatched, then the scheduler and dispatcher would not look further down in that queue. But because the loads and stores are implemented in the separate queue, the operations which could cause the most latency (memory loads) can execute faster. But PowerPC 750 on the other hand issue looking at only 2 entries in the queue and the reservation stations hold the entries for each EU. This means that when PowerPC really relies on executing the instructions in parallel, Pentium 4 relies on the deep pipeline behaviour within the execution units. And the rapid execution core in Pentium 4 helps this cause as well.

As we can see, PowerPC not only attacked the performance problem with a deep pipeline matching Pentium 4's; but also has more EUs than Pentium4. So, it packs more power in the hardware. Something that PowerPC could add in the future for making full use of the resources is on-chip multi threading (SMT). POWER5 processor is supposed to have that. So, we can hope the successor of PPC970 to have that.

5. Conclusion

The two micro-processor families described here really compete for the top spot in modern high performance desktops and workstations. The latest 64 bit PowerPC and the Pentium 4 seem to be very close in terms of performance. PowerPC 970 was not only designed to run at higher speeds;

but they are also designed for graphics and media applications by providing a power Altivec engine. But Intel aims to run all applications better with its hyper-threading technology and providing optimal amount of hardware resources. But, if Intel continues for long to be backward compatible to IA-32, future design's performance might be constrained by this need for backward compatibility. So, we might begin to see the PowerPC processors on Apple Power Macs outperforming Pentium 4 in the years to come.

6. References

[Keith] D.Keith, O.Rich and H.Ron, " Evolution of the PowerPC Architecture", IEEE Micro, Vol 14, no 2, April 1994, pp. 34-49

[IBM601] PowerPC 601 RISC Microprocessor Technical Summary.

[MPC601] MPC601 PowerPC 601 User Manual.

[IBM750] PowerPC 750 RISC Microprocessor Technical Summary.

[IBMhist] PowerPC Architecture, a high performance architecture with a history. http://www-1.ibm.com/servers/eserver/pseries/hardware/whitepapers/power/ppc_arch.html

[Keith94] D.Keith, "History of the PowerPC Architecture", Communications of the ACM, Vol 37, Issue6, June 1994, pp. 28-33.

[Tom02] R.Tom, "IBM Trims Power4, Adds AltiVec", http://www-.ibm.com/chips/techlib/techlib.nsf/products/PowerPC_970_Microprocessor

[AppleG5] PowerPC G5, "The World's First 64bit Desktop Processor", White paper. July 2003.

[Jon03] H.S Jon, "Inside the IBM PowerPC 970, http://arstechnica.com/cpu/02q2/ppc970/ppc970-4.html, 2003

[Pow4] M.T. Joel, D.Steve and F.Steve, "POWER 4 System Microarchitecture", Technical white paper, Oct 2001

[PowHis] Wikipedia, http://www.wikipedia.org/wiki/PowerPC

[6.12.] Intel technology Journal, 2001 – The micro-architecture of the Pentium 4 processor

[6.13.] Tom's hardware guide on Pentium4 processor - http://www6.tomshardware.com/cpu/20001120/index.html

[6.14.] Modern Processor design: Fundamentals of Superscalar processors by Shen and Lipasti

[Moz] F.Darin, HTTP/1.1 Pipelining FAQ, http://www.mozilla.org/projects/netlib/http/pipelining-faq.html