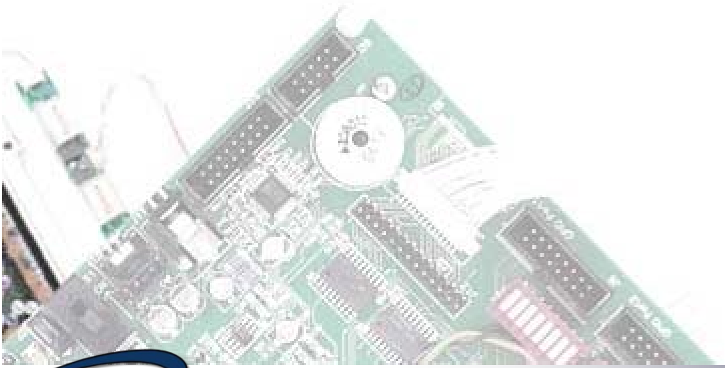
Basic Computing Concept and Instruction Level Parallelism (Part II)

2010.12.01

Nam Eyee Hyun



Contents

- Review
- CISC and RISC
- Pentium
- PowerPC
- Summary

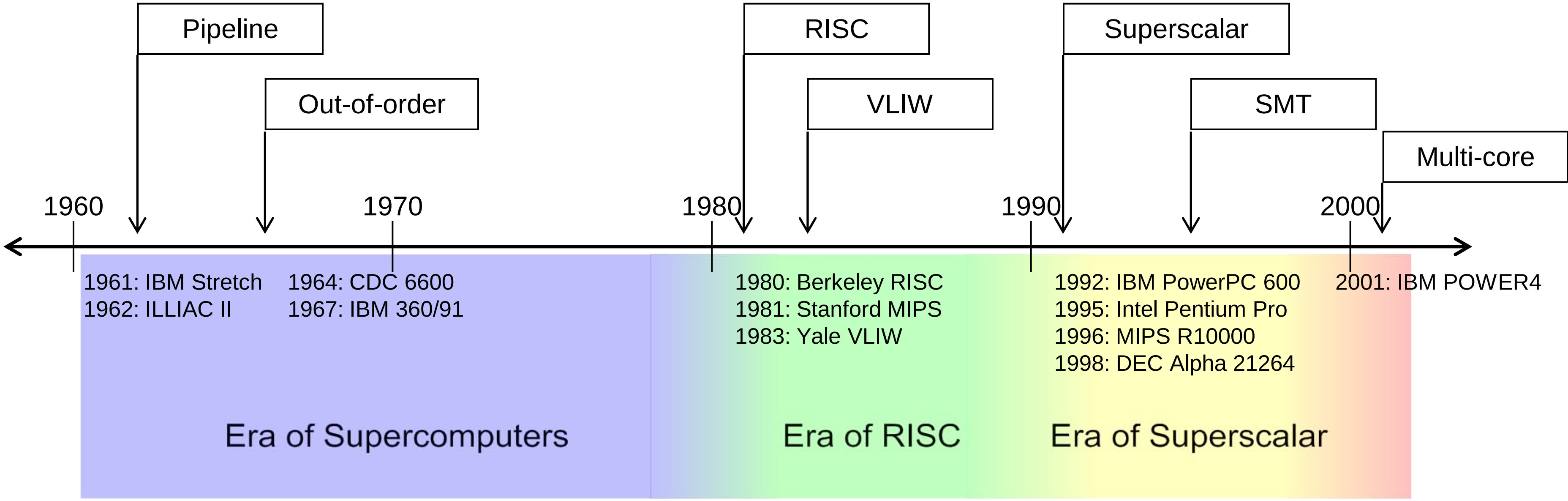
Review

- Basic computing concept
  - Introduced calculator model and file clerk model
  - Explained ISA and programming model
- Micro-architecture techniques used to exploit ILP

	Key concepts	Scheduling Mechanism	Main approach for Performance Enhancement	Relation between Frontend & Backend
Pipelined execution	In-order, partially overlapped execution	X	Increasing bandwidth	Lock step
Superscalar execution	In-order, parallel execution	Fixed, rule base scheduling	Increasing bandwidth	Lock step
Out of order execution	Out of order, parallel execution	Dynamic scheduling	Decreasing latency	Decoupled

- Hazards
  - Outlined the various issues on hazards

Historical prospective



Pipeline

Out-of-order

RISC

VLIW

Superscalar

SMT

Multi-core

1960 1970 1980 1990 2000

1961: IBM Stretch
1962: ILLIAC II

1964: CDC 6600
1967: IBM 360/91

1980: Berkeley RISC
1981: Stanford MIPS
1983: Yale VLIW

1992: IBM PowerPC 600
1995: Intel Pentium Pro
1996: MIPS R10000
1998: DEC Alpha 21264

2001: IBM POWER4

Era of Supercomputers

Era of RISC

Era of Superscalar

CISC vs. RISC

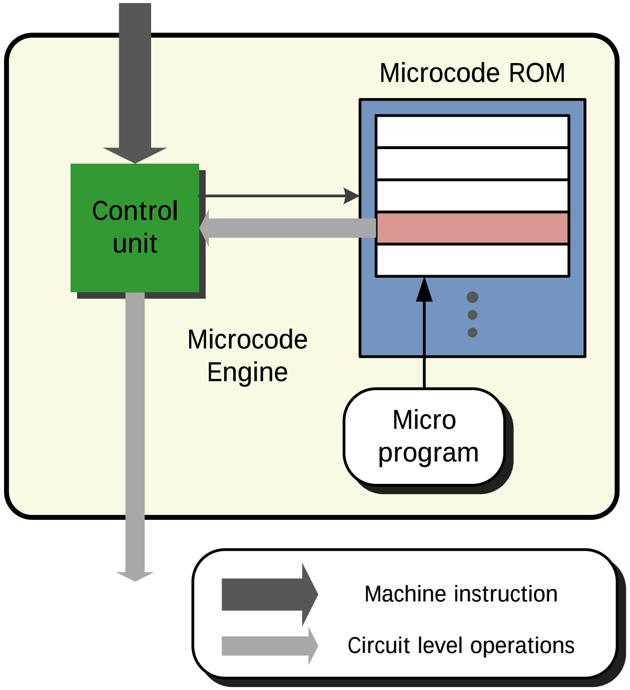
- Complex Instruction Set Computer (CISC) - In the 1960s and onward
  - Historical background
    - Programming was done in assembly language
    - Hardware design was more mature than compiler design
    - Memory was expensive, slow and small in capacity
    - Transistor budget was limited
  - Goal
    - To complete a task in as few lines of assembly as possible
      - Providing powerful and easy to use instructions
      - Supporting every possible addressing mode for every instruction
      - Encoding instructions more compactly
    - To provide an instruction set that closely supports the operation and data structures used by Higher-Level Languages
      - Simplifying compiler design

CISC vs. RISC

- CISC
  - Features
    - Complex instructions and operations
      - Need for microcode
      - Implicit load and store operation
    - Complex data types
    - Complex addressing modes
      - Register to memory , memory to memory operation
    - Small register set
    - Variable length and highly encoded instruction format
      - Small code size and fewer main memory access
  - Complexity management
    - Assembly language + microcode
  - Examples
    - X86, IBM system/360, PDP-11, VAX, 68000

Microcode

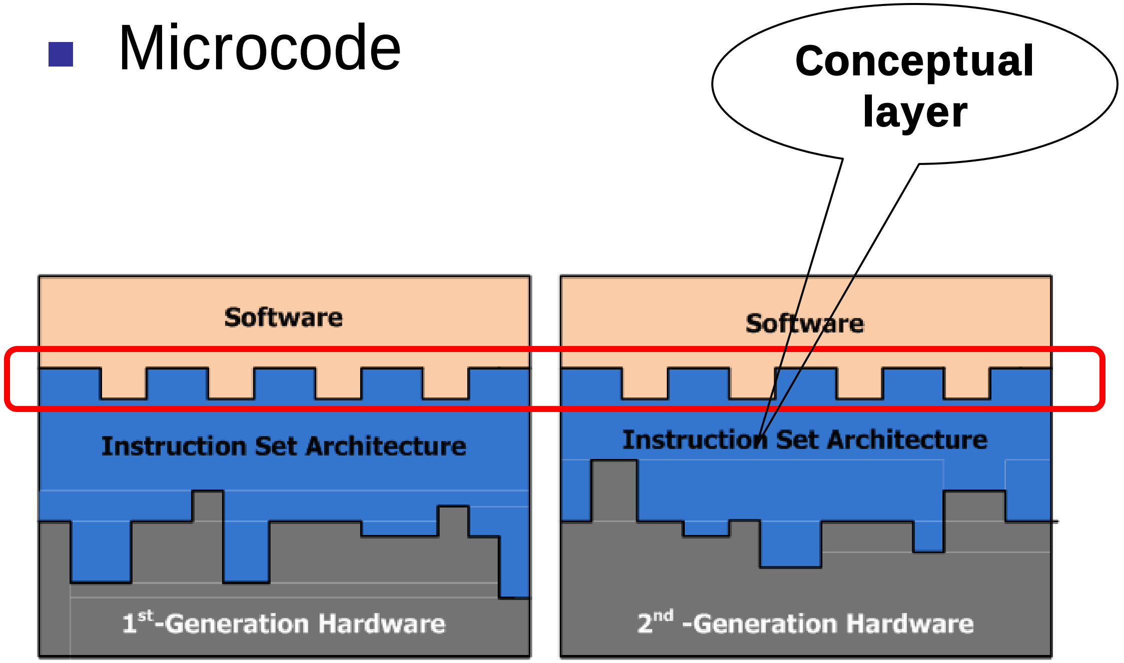
- Microcode
  - Is a layer of lowest-level instructions involved in the implementation of machine instruction in many CISC processors
  - Mainly used in a decode stage of CISC processor, translate machine instructions into a series of detailed circuit level operations

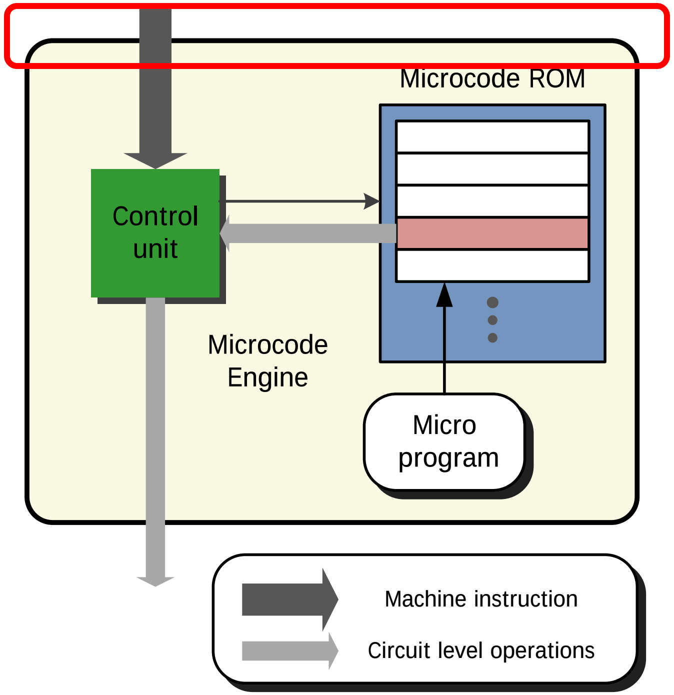


- Microcode engine
  - Sort of “CPU within a CPU”
    - Storage : Control store
    - Program : Microcode program
    - Execution : Instruction translation
  - Role
    - Translates a particular instructions into a series of commands that control the internal part of the chip

Microcode

- Microcode





Microcode

- Advantages
  - Separation of ISA design from micro-architectural implementation
    - Backward binary compatibility
    - Instruction can be designed and altered more freely
    - Ability to emulate other computer systems
  - Systematic & flexible control unit design
    - Implementing processor's behavior not by dedicated HW design but by programming methodology
    - Field modification without rewiring or changing hardware logic
    - Reconfigurable computing using a writable control store
  - Low marginal cost for additional function

- Disadvantage
  - Uneconomical for small systems
    - Due to relatively high fixed cost of microcode engine
  - Slower than direct decoding
    - Another level of interpretation with associated overhead
    - Instruction cycle time limited by control store access time

CISC vs. RISC

- Some observations on CISC
  - Complex instructions and data types
    - Were not used by most programs generated by compilers available at that time
    - Many real-world programs spend most of their time executing simple operations
  - Complex addressing mode
    - Caused variable instruction execution times, depending on the location of the operands
    - Led to variable length instruction and in turn leads to inefficient instruction decoding and scheduling
  - Complex operations
    - Tended to be slower than a sequence of simpler operations doing the same thing
  - Small register set
    - Procedure call/return activities generate a large number of memory references in a small register set environment

CISC vs. RISC

- Reduced Instruction Set Computer (RISC) - In the mid 1970s and onward
  - Historical background
    - High level language became generally used
    - Compiler design technology became mature
    - Advent of semiconductor memory reduced access speed gap between main memory and processor
  - Goal
    - To reduce the complexity of each instruction
    - To make each instruction execute very quickly

CISC vs. RISC

- RISC
  - Features
    - Simple instructions and operations
      - Each instruction can execute in one cycle
      - No need for microcode
    - Simple data types in hardware
    - Simple addressing mode
      - Explicit load and store operation
      - Register to register operation
    - Large general purpose register set
      - Provides ample opportunities for the compiler to optimize their usage
    - Fixed size and uniform instruction format
  - Complexity management
    - High level language + sophisticated compiler
  - Examples
    - PowerPC, ARM, MIPS..

CISC vs. RISC

- Post-RISC architectures (In the late 1980s and onward)
  - Historical background
    - Chip fabrication techniques have improved exponentially, according to Moore's law
    - Compilers have also become more sophisticated, and are better to exploit complex instruction on CISC architecture
    - Architectural improvements have been comparatively small
  - CISC and RISC convergence
    - What matters is performance and not principle
  - Post-RISC features
    - Additional registers
    - Increased on-die caches
    - Additional functional units for superscalar execution
    - Additional resources for out of order execution
    - Additional non-RISC but fast instructions
    - On-die floating point and vector processing units
    - Increased pipeline depth
    - SMT and SMP

CISC vs. RISC

- CISC
  - Chinese character
  - Ex) 孤掌難鳴
- RISC
  - English or Korean character
  - Ex) 손바닥으로는 소리를 내기가 어렵듯이, 혼자만의 힘으로는 일을 하기가 어렵다
  - Ex) One needs assistance to accomplish anything

Contents

- Review
- CISC and RISC
- Pentium
- PowerPC
- Summary

The chronicle of the Pentium and PowerPC

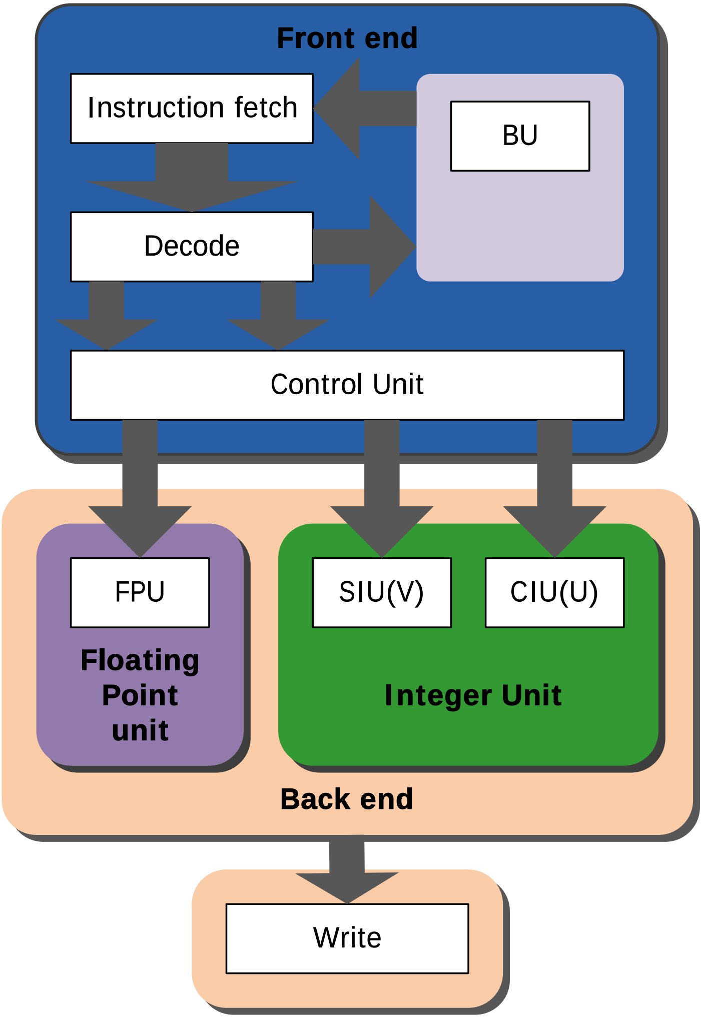
A(Apple)
I(IBM)
M(Motorola)

RISC		Intel	AIM
	1993	Pentium	
	1994		PowerPC 601
	1995	Pentium Pro	PowerPC 603 & 603e
Superscalar	1996		PowerPC 604 & 604e
	1997	Pentium II	PowerPC 750
	1999	Pentium III	PowerPC 7400
	2001	Pentium IV	PowerPC 7450 & POWER4
	2003	Pentium M & Intel Core	PopwerPC970
	2004		POWER5
Multi core & low power	2006	Intel Core 2	
			POWER6
	2008	Atom	

Historical Background in the early 1990s

- Background
  - Continuing Moore's law
  - Relatively small architectural innovations
- Paradigm shift
  - From "How do we squeeze all the hardware that we'd like to put on the chip into our transistor budget "
  - To "How on earth do we get this many transistor to do useful, performance-enhancing work?"
- Wide spread of PC environment
- Market situation
  - Intel was dominating PC market
  - RISC processors were dominating server and workstation market
  - AMD was rising as a serious competitor of Intel

P5 micro-architecture



	Pentium
Introduction date	March 22, 1993
Manufacturing Process	0.8um
Transistor Count	3.1million
Clock speed at introduction	60MHz and 66Mhz
L1 Cache sizes	8KB instruction, 8KB data

Features

- 5 stages basic pipeline
- Superscalar execution
- 2 x asymmetric IU, 1xFPU

Implementation

- Pentium

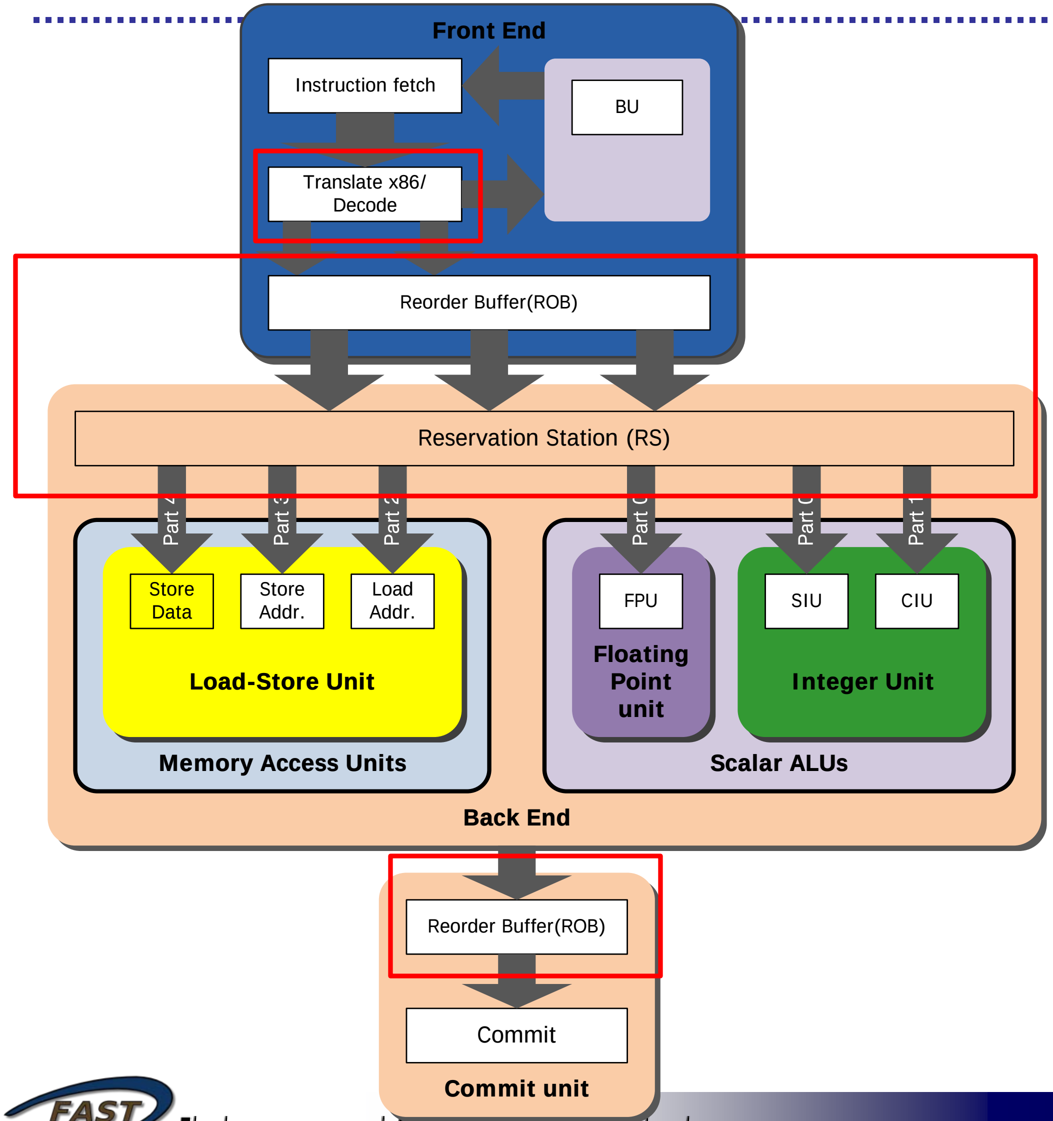
Competitor

- PowerPC 601

P5 micro-architecture

- Basic pipeline of P5 micro-architecture
  - Pre-fetch / Fetch
    - Variable instruction length from 1 to 17 bytes
    - Pre-fetch buffer
      - Instruction’s boundary is detected and marked
      - Then marked instructions are aligned
  - Decode 1
    - Decode machine instruction into the Pentium’s internal instruction format
    - Handle branch instruction
  - Decode 2
    - Complex instructions are decoded by means of a microcode ROM
    - Extra address calculation for multiple complex addressing mode
  - Execute
  - Write back

P6 micro-architecture



Common features

- 12 stages basic pipeline
- Instruction set translation
- Out of order execution
- Decoupled architecture
- 5 shared issue ports

Background

- Advent of AMD

Implementations

- Pentium Pro
- Pentium II
- Pentium III

P6 micro-architecture

	Pentium Pro	Pentium II	Pentium III(Katmai)
Introduction date	November 1, 1995	May 7, 1997	February 26, 1999
Manufacturing Process	0.6/0.35um	0.35um	0.25um
Transistor Count	5.5million	7.5 million	9.5million
Clock speed at introduction	150, 166, 180, and 200Mhz	233, 266, and 300MHz	450 and 500Mhz
L1 Cache size	8KB instruction, 8KB data	16KB instruction, 16KB data	16KB instruction, 16KB data
L2 Cache size	Off die 256KB or 512KB	Off die 512KB	Off die 512KB
X86 ISA extensions		MMX	SSE

Competitor

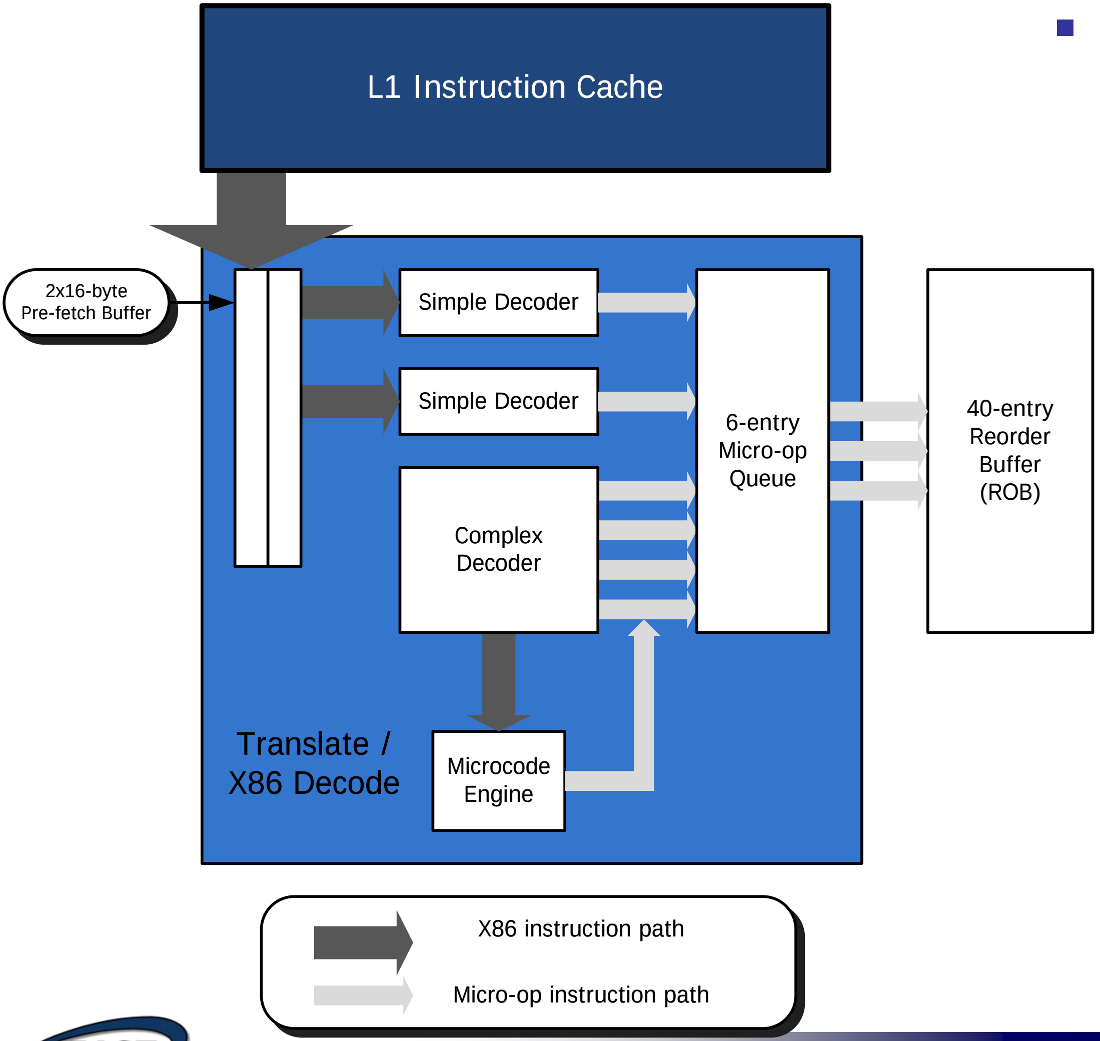
- Pentium Pro : PowerPC 604/604e
- Pentium II : PowerPC 750
- Pentium III : PowerPC 7400

P6 micro-architecture

- Basic pipeline of P6 micro-architecture

Name	# of pipeline stages	Description
Pre-fetch / fetch	3 and 1/2	Instruction fetching and branch processing
Decode	2 and 1/2	Translate x86 CISC instruction into the P6's internal RISC-like instruction format
Register rename	1	Register renaming and instruction logging in ROB
Dispatch from ROB to RS	1	3 instructions can dispatch into the RS buffer on a cycle
Issue from RS to EX unit	1 ~ multi	5 instructions can issue out of RS buffer on a cycle Instruction can sit in the RS for an unspecified # of cycles depending on the state of the processor and the resources currently available for execution
Execute	1 ~ multi	
Commit	2	Writing the result of the instruction execution back into the ROB and then committing the instructions by writing the results from the ROB into the architectural register file

P6 micro-architecture



- Instruction set translation
  - Speeding up x86-based architecture using the RISC-oriented dynamic scheduling techniques
    - Translating CISC operations into smaller, faster, more uniform RISC-like operations for use in the back end
  - All subsequent x86 processors have used instruction set translation

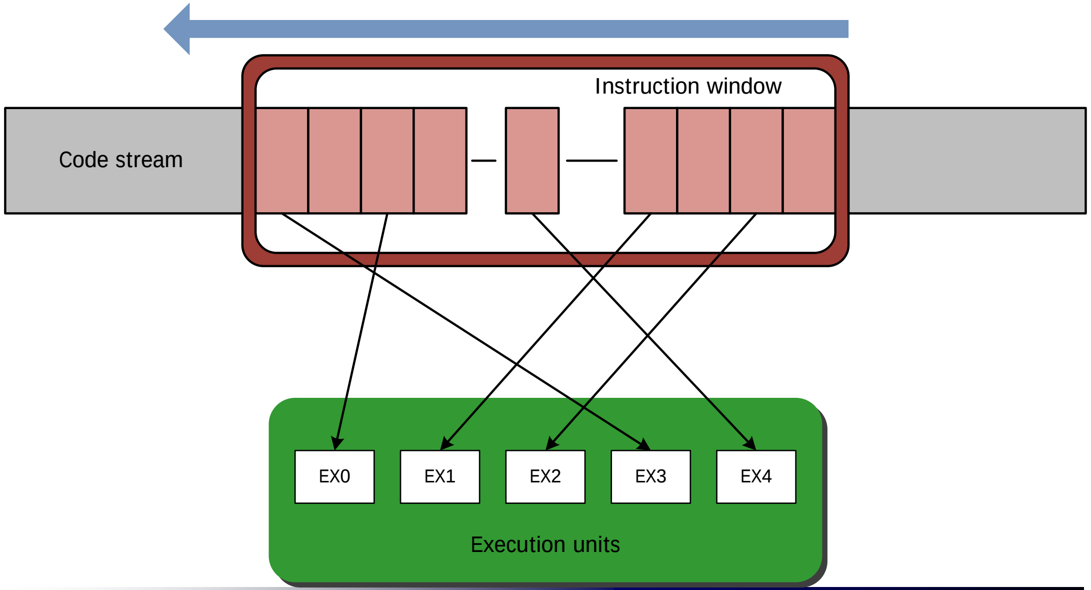
The P6 Micro-architecture’s decoding
  - Common and simple x86 instruction
    - Dedicates most of the decoding hardware
  - Less common and complex instruction
    - Complex and slow decoder works in conjunction with the microcode ROM
    - Translated sequences of micro-ops are read directly from the ROM

P6 micro-architecture

- Decoupled architecture of P6
  - The Reservation station
    - Buffer for shuffling and optimizing the code stream
    - Decoupling buffer
      - Is a buffer for performance improvements
      - Decouples the front end's fetch/decode bandwidth from the back end's execution bandwidth
  - The reorder buffer
    - Buffer for un-shuffling and reordering the code stream
    - Instruction tracker
      - Decoded instructions are given a tracking entry in the ROB
    - Register renaming
      - Decoded instructions have temporary rename register
    - Completion buffer
      - Executed instructions must wait in the ROB before they can commit

P6 micro-architecture

- Decoupled architecture of P6
  - Instruction window
    - Is a common metaphor for the combination of RS and ROB
    - P6 is looking through this window at visible segment of the code stream and thinking about how its execution units can optimally executes the instructions in the window



Evolution of P6 micro-architecture

- Pentium II
  - Features
    - 32KB split on die L1 cache
    - Multi-Media eXtensions (MMX)
      - Used for integer vector processing
      - 2 dedicated MMX units
  - Considerations
    - Furthered the trend of x86 commodity hardware's migration into the server and workstation realms
    - Still, couldn't stand up to RISC design built on the same process with similar transistor count

Evolution of P6 micro-architecture

Code name	Katmai	Coppermine
Introduction date	February 26, 1999	Early 2000
Manufacturing Process	0.25um	0.18um
Clock speed at introduction	450 and 500Mhz	Beyond 1GHz
L2 Cache size	Off die 512KB Using daughter card	On die 256KB

- Pentium III
  - Features
    - On die 256KB cache
    - Streaming SIMD Extensions(SSE)
      - Used for floating point vector processing
    - Processor serial number(PSN)
      - Intended for use in securing on-line commercial transactions
  - Considerations
    - Intel and AMD were locked in a “gigahertz race”, in 1999
      - Intel Pentium III vs. AMD athlon

X86 overhead on the P5 and P6

- Downside of P5 and P6 micro-architecture
  - Cause
    - Legacy x86 support for backward compatibility
      - 30% of the Pentium's transistor and 40% of Pentium II's transistor is dedicated solely to providing x86 legacy supports
        - E.g. Microcode ROM
    - X86 related bloat
      - Pre-fetch buffer has to consider the variable length instruction
      - Decode logic needs its own address calculation hardware
    - Limited transistor budget
  - Effect
    - Worse performance than its RISC contemporary competitors

X86 overhead on the P5 and P6

- Favorable trends to Intel
  - Relative cost of legacy x86 ISA was being reduced
    - More often ISA extensions such as MMX, SSE are required according to the increasing multimedia environment
    - The core legacy x86 ISA is fixed in size
    - Moore’s curve is extremely kind to the x86 ISA
      - E.g. Less than 10% of the Pentium IV’s transistor is used to providing x86 legacy supports

Summary of P5 and P6 micro-architecture

- Technical side
  - Deep pipeline depth
    - Super-pipelining
    - Need for more accurate branch prediction
  - Buffered decoupling
  - Less cache size compared with its contemporary competitor

- Comments
  - The advent of a serious competitor AMD stimulated the pace of progress in x86 architecture
  - Intel realized that “Clock speed sells”
  - Moore’s curve was extremely generous to the x86 ISA for several reasons

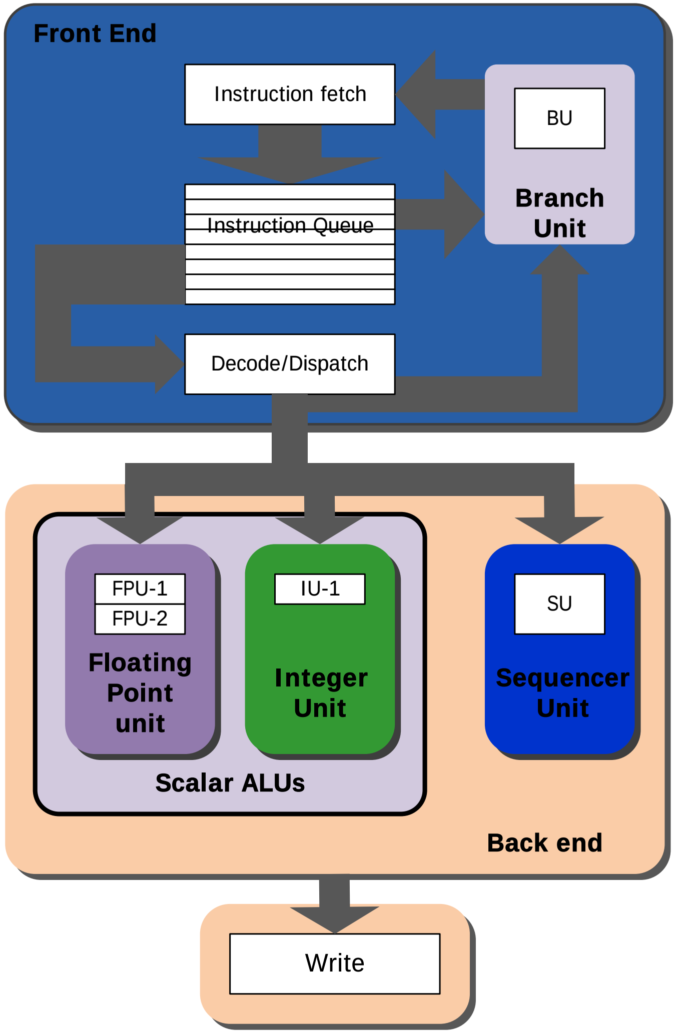
Contents

- Review
- CISC and RISC
- Pentium
- PowerPC
- Summary

Brief history of PowerPC

- Two roots of PowerPC architecture
  - POWER(Performance Optimization With Enhanced RISC)
    - IBM's RISC architecture developed for use in mainframe servers
  - 68000
    - Motorola's CISC architecture developed for Apple's desktop computing line
- AIM alliance
  - Apple needed a CPU for its personal computers that would be both cutting edge and backward compatible with the 68K
  - IBM needed a way to turn POWER into a wide range of computing products for use outside the server closet
  - Motorola needed a high-end RISC microprocessor in order to compete in the RISC workstation market

PowerPC 601



	PowerPC 601
Introduction date	March 14, 1994
Manufacturing Process	0.6um
Transistor Count	2.8million
Die Size	121mm2
Clock speed at introduction	60MHz - 80Mhz
L1 Cache sizes	32KB unified
First Appeared in	PowerMAC 6100/60

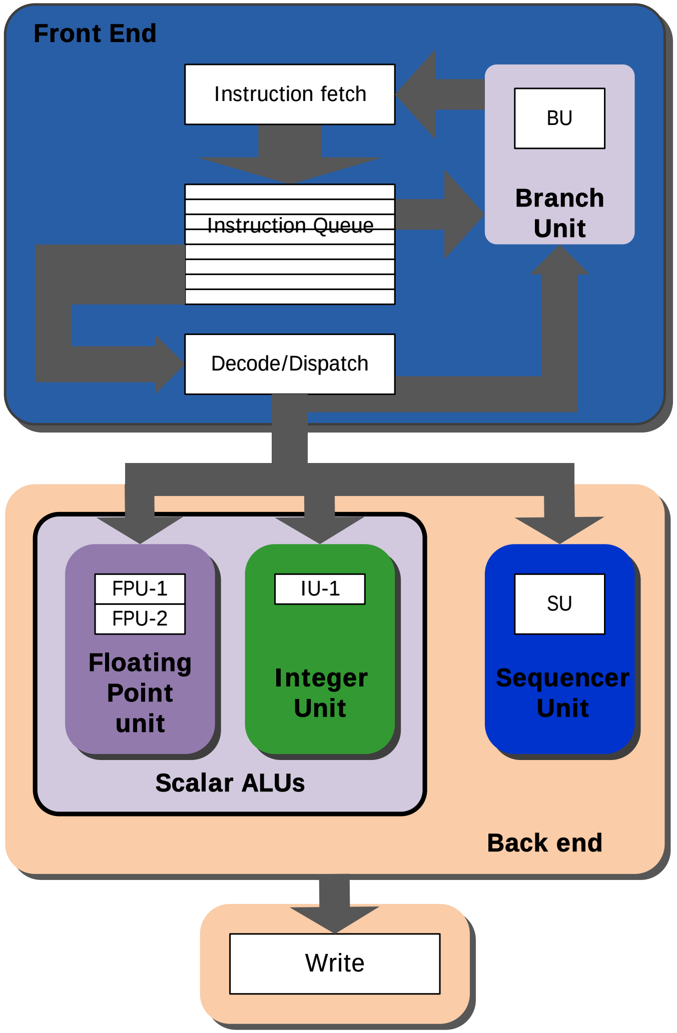
Competitor

- Pentium

Background

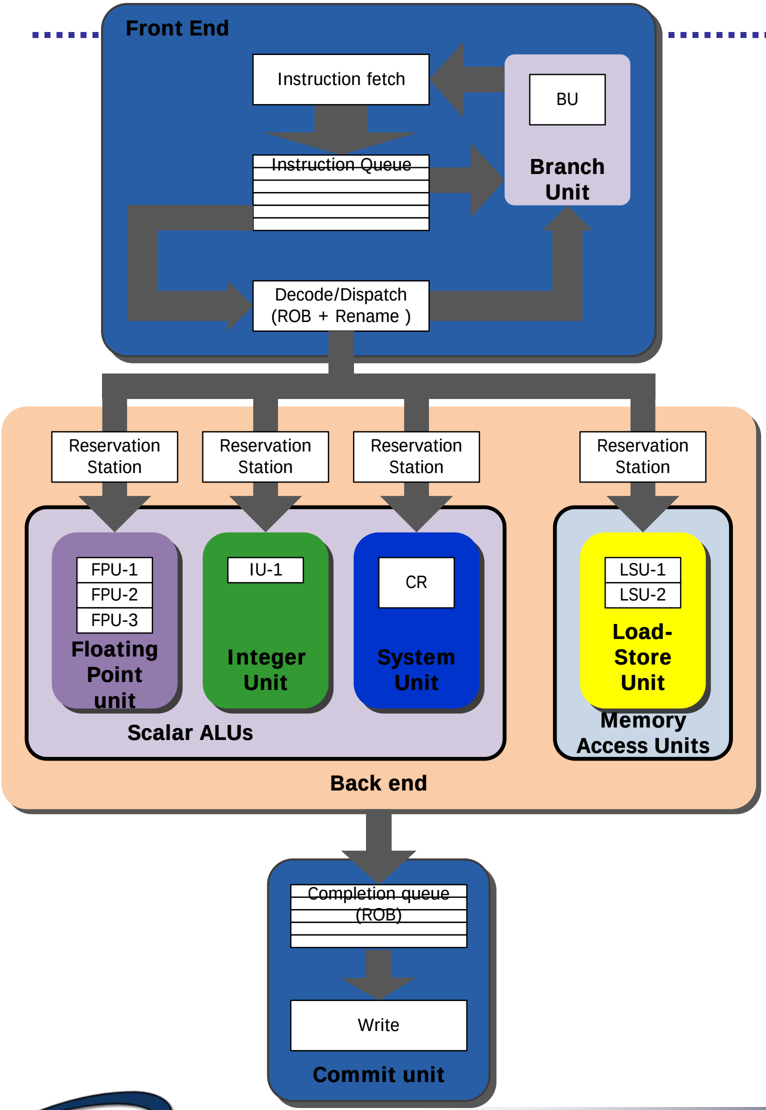
- 601 was based on IBM's older RSC (RISC Single Chip) processor
- 601 was designed as a bridge between POWER and PowerPC

PowerPC 601



- Features
  - 4 stages basic RISC pipeline
  - Frontend
    - Instruction Queue with 8 entries
    - Shared dispatch bus
      - Dispatch rate 3
  - Backend
    - 1xFPU, 1xIU, 1xBU
    - 1xSU(Sequencer Unit)
      - Small CISC-like sub processor with its own instruction set, 32 word RAM, microcode ROM, register file, and execution unit
  - Superscalar and rudimentary OOB execution
    - Out of order dispatch
    - No ROB and RS
    - Instruction tagging with program-order metadata

PowerPC 603/603e

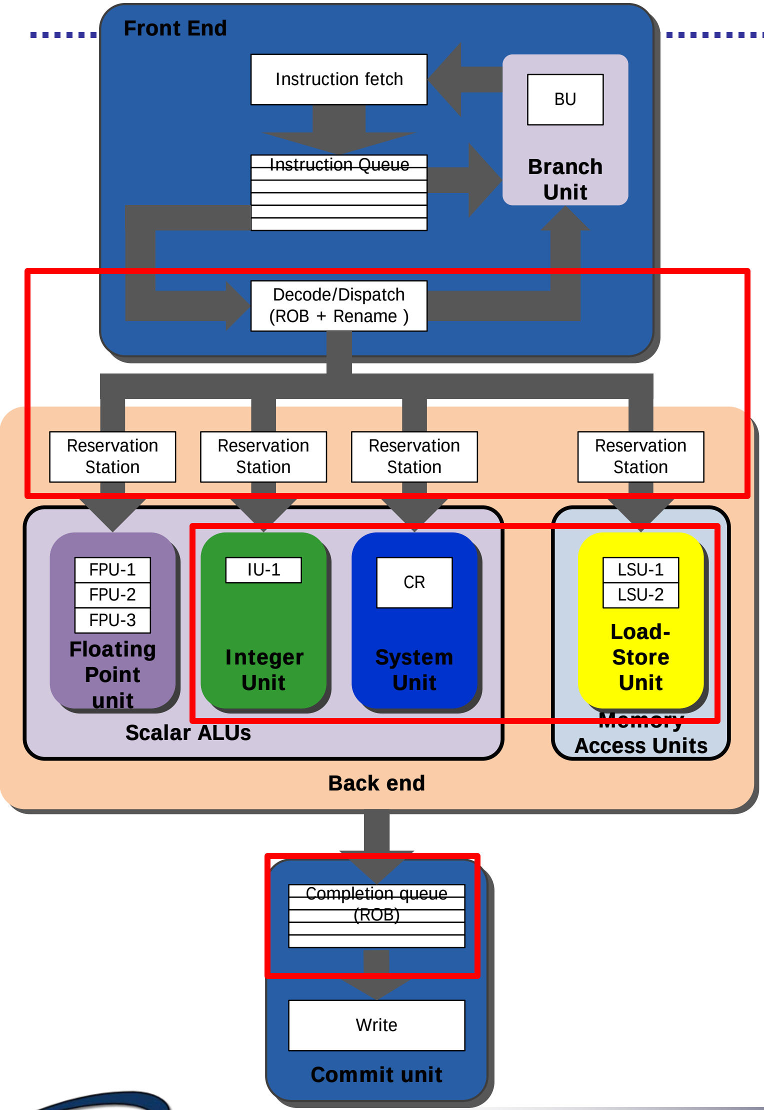


	PowerPC 603	PowerPC 603e
Introduction date	May 1, 1995	October 16, 1995
Manufacturing Process	0.50um	0.50um
Transistor Count	1.6million	2.6 million
Die size	81mm2	98mm2
Clock speed at introduction	75MHz	100MHz
L1 Cache size	8KB instruction, 8KB data	16KB instruction, 16KB data
First appeared in	Mac performa 5200CD	Mac Performa 6300CD

Background

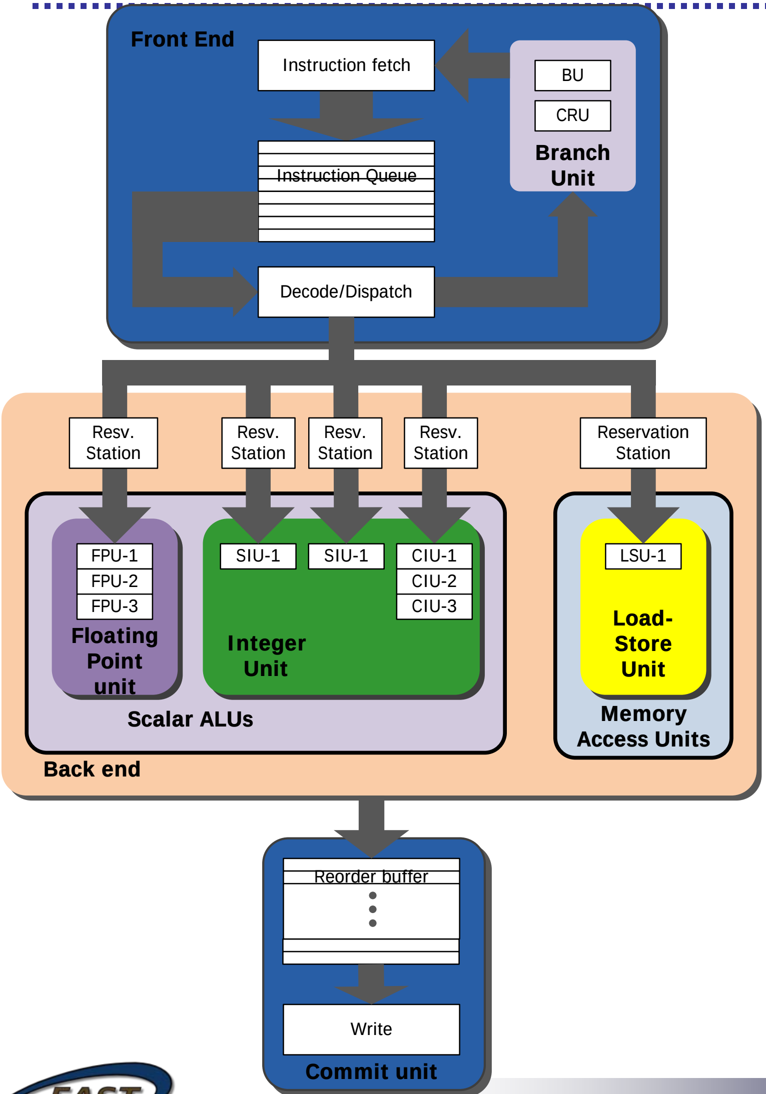
- 603 was designed for laptop computer or low to midrange desktop
- The first PowerPC processor to feature dynamic scheduling

PowerPC 603/603e



- Features
  - 4 stages basic RISC pipeline
  - Frontend
    - Instruction queue with 6 entries
    - Dedicated dispatch bus
      - Dispatch rate 2
  - Backend
    - 1xFPU, 1xIU, 1xBU
    - 1xLSU
      - Calculates load/store address
      - Handles load/store memory traffic
    - 1xSU(System Unit)
      - Used for update PowerPC condition register
  - Superscalar and OOB execution
    - Full-blown instruction window, complete with a ROB and RS
    - Very small instruction window
    - Relatively small rename register
    - 5 entries ROB
  - Good performance per watt ratio on native PowerPC code

PowerPC 604/604e



	PowerPC 604	PowerPC 604e
Introduction date	May 1, 1995	July 19, 1996
Manufacturing Process	0.50 um	0.35um – 0.25um
Transistor Count	3.6 million	5.1 million
Die size	197mm2	148mm2
Clock speed at introduction	120MHz	180MHz – 350MHz
L1 Cache size	16KB instruction, 16KB data	32KB instruction, 32KB data
First appeared in	PowerMAC 9500/120	PowerMAC 9500/180

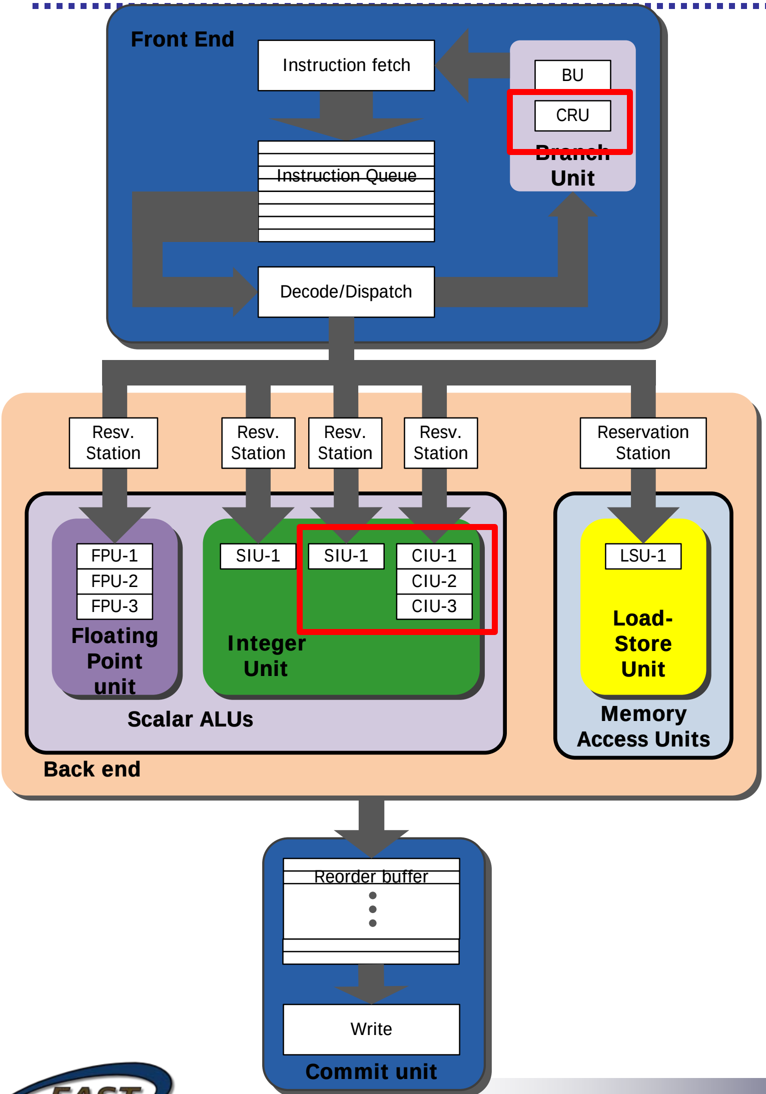
Competitor

- Pentium Pro

Background

- 604/604e was designed for high end desktop computer

PowerPC 604/604e



- Features
  - 6 stages basic RISC pipeline
    - Lengthened pipeline for higher clock speeds
  - Frontend
    - Instruction Queue with 8 entries
    - Dedicated dispatch bus
      - Dispatch rate 4
  - Backend
    - 1xFPU, 3xIU, 1xBU, 1xLSU
    - 1xCRU(Condition Register Unit)
      - The system unit was removed and CRU was contained in branch unit
      - CRU is used for update PowerPC condition register
  - Superscalar and OOB execution
    - 16 entries ROB

PowerPC 604/604e

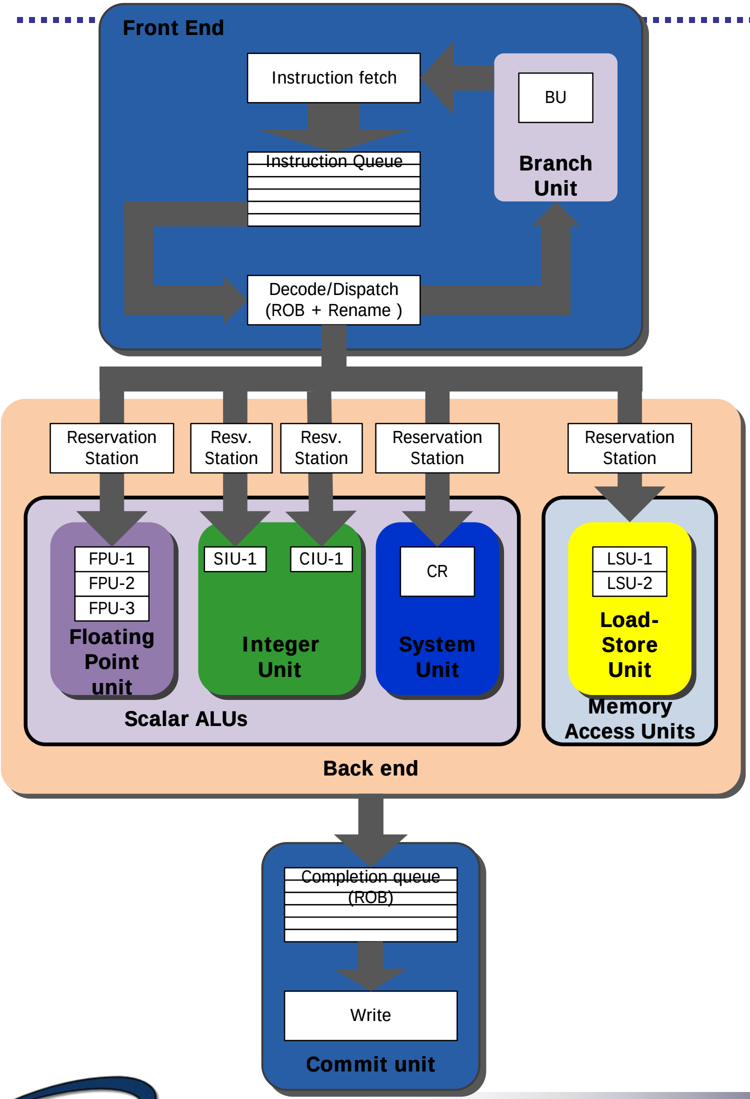
- Decoupled architecture of 604/604e
  - Characteristics of RS and ROB
    - Reservation station
      - Each execution unit has its own RS attached to it
        - *In order dispatch* and *in order issue* in the same reservation station thus *in order execution*
        - *Out of order issue* and *our of order execution* from the perspective of the overall program flow
      - Each reservation stations are relatively small and FIFO structure
        - Small size window is due to the relatively short pipeline compared with its P6 counterpart

    - Reorder buffer
      - Instructions that have finished executing wait in complete buffer to have its result written back to the register file
      - Reorder the code stream in completion buffer

PowerPC 604/604e

- Decoupled architecture of 604/604e
  - Mechanism
    - Dispatch phase (Dispatch + Issue)
      - Rename registers and a reorder buffer entry are assigned to each dispatching instruction
      - Instruction is sent either
        - Directly to an execution unit or
        - To an execution unit's reservation station
    - Complete phase
      - Instruction that have finished executing wait in ROB to have its result written back to the register file
      - Reorder the code stream in ROB
    - Write back phase (commit)
      - Write the instruction's result back to architecture register file and alter the programmer-visible machine state permanently

The PowerPC 750 (G3)



	PowerPC 750
Introduction date	September 1997
Manufacturing Process	0.25um
Transistor Count	6.35million
Die size	67mm2
Clock speed at introduction	200-300MHz
L1 Cache size	32KB instruction, 32KB data
L2 cache size	1MB
First appeared in	PowerMac G3

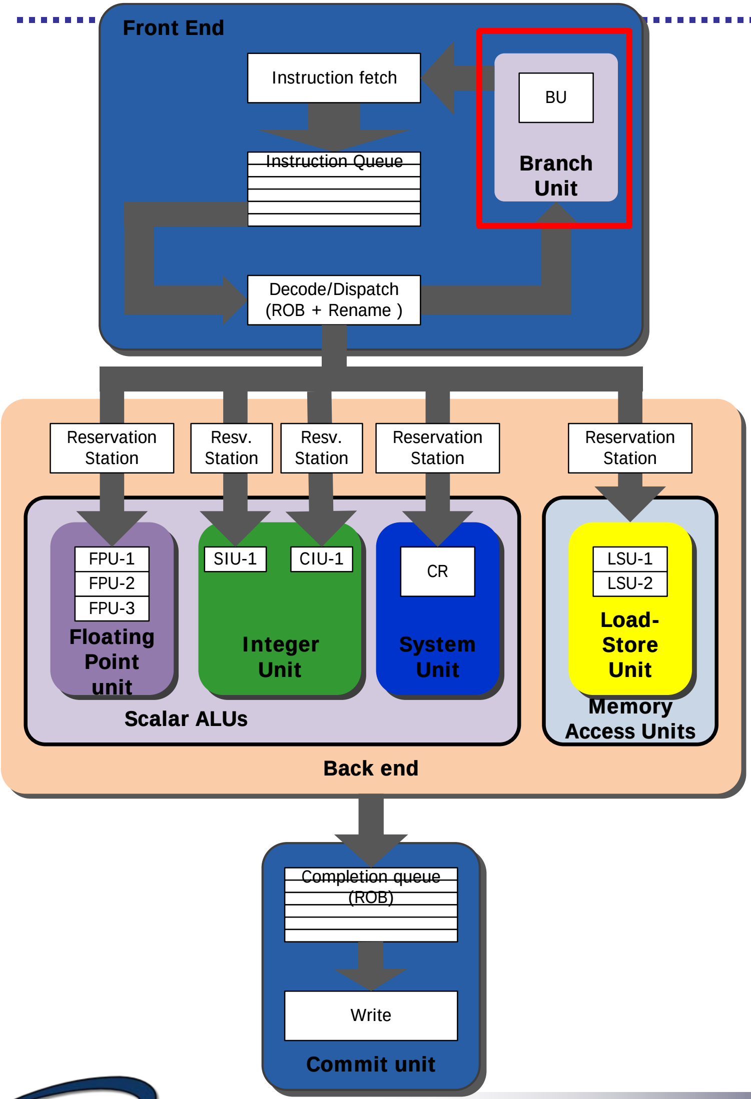
Competitor

- Pentium II

Background

- 750 is a design based heavily on the 603/603e

PowerPC 750



- Features
  - 4 stages basic RISC pipeline
  - Frontend
    - Instruction Queue with 6 entries
    - Dedicated dispatch bus
      - Dispatch rate 2
  - Backend
    - 1xFPU, 2xIU, 1xBU, 1xLSU
    - 1xSU(System Unit)
  - Superscalar and OOB execution
    - 6 entries ROB
  - More advanced dynamic branch prediction
    - Branch target instruction cache
  - Large L2 cache and dedicated L2 cache interface

PowerPC 7400 (G4)

	PowerPC 750
Introduction date	September 1999
Manufacturing Process	0.20um
Transistor Count	10.5million
Die size	83mm2
Clock speed at introduction	400-600MHz
L1 Cache size	32KB instruction, 32KB data
L2 cache size	2MB
First appeared in	PowerMac G4

Competitor

- Pentium III

Background

- 7400 is designed as a media processing powerhouse for desktops and portables

- Features
  - 4 stages basic RISC pipeline
  - Frontend
    - Instruction Queue with 8 entries
    - Dedicated dispatch bus
  - Backend
    - 1xFPU, 2xIU, 1xBU, 1xLSU
    - 1xSU(System Unit)
    - 1xVPU, 2xVIU, 1xVFU
  - Superscalar and OOB execution
  - More advanced dynamic branch prediction
    - Branch target instruction cache
  - Large L2 cache and dedicated L2 cache interface

Summary of PowerPC micro-architecture

- Technical side
  - Shallow-pipeline
  - Decoupling architecture
    - RS attached to execution unit
    - Shallow RS
  - Large cache size compared with its contemporary competitor
  - Advanced branch prediction
    - Branch target instruction cache

- Comments
  - It seems that IBM inconsistently modified PowerPC architecture
  - AIM didn't utilize the increased transistor budget constructively
    - Moore's curve is also kind to the PowerPC ISA but their effect was marginal to PowerPC
  - IBM stood outside the "gigahertz race"

Contents

- Review
- CISC and RISC
- Pentium
- PowerPC
- Summary

Summary - CISC vs. RISC

	CISC	RISC
Complexity	Hardware	Software
Performance	Decreased code size at the expense of a higher CPI	A lower CPI at the expense of increased code size
Instruction set	A large and varied instruction set includes complex, multi-cycle instructions	A small and uniform instruction set includes only simple, single-cycle instructions
HLL support	Support for HLLs is done in hardware	All HLL support is done in software
Addressing mode	Supports for memory-to-memory(register) addressing modes	Only support for register-to-register addressing mode
Control unit	microcode	Direct execution
# of Registers	Small	Large

- The RISC-CISC distinction has blurred significantly in practice

Summary - Pentium vs. PowerPC

- Design philosophy
  - Pentium
    - Narrow and deep approach
      - Less functional units
      - Deep pipeline
      - Large instruction window
  - PowerPC
    - Wide and shallow approach
      - More functional units
      - Shallow pipeline
      - Small instruction window