Dec Hex Bin
23 17 00010111

ORG ; TWENTY-THREE

PENTIUM AND RISC PROCESSORS

The x86 PC

assembly language, design, and interfacing

fifth edition

MUHAMMAD ALI MAZIDI
JANICE GILLISPIE MAZIDI
DANNY CAUSEY

The x86 PC
assembly language, design, and interfacing
fifth edition

Prentice Hall

OBJECTIVES

this chapter enables the student to:

- List the design enhancements of the x86 microprocessors from 80486 to Pentium® 4.
- Discuss the advantages of the 5-stage pipeline over the 2-stage pipeline.
- Explain how the burst cycle is used to increase memory cycle times for read/write operations.
- Compare the cache sizes of x86 processors from 486 to Pentium® 4.
- List three ways that designers can increase the processing power of a CPU.

OBJECTIVES
this chapter enables the student to: *(cont)*

- List design enhancements of the Pentium® over previous-generation x86 microprocessors.
- Describe the impact on performance of the 64-bit data bus of the Pentium®.
- Describe superscalar architecture and Harvard architecture and their use in the Pentium®.
- List the unique features of RISC architecture compared to CISC and describe the impact on processing speed and program development.

OBJECTIVES (*cont*)

this chapter enables the student to:

- Describe the main features introduced or enhanced in the Pentium® Pro.
- Give an overview of how MMX (Multimedia extension) technology is used in some Intel processors.
- Describe which aspects of DSP (digital signal processing) were incorporated into MMX technology.
- Code Assembly language instructions to identify the CPU.

23.1: THE 80486 MICROPROCESSOR

- The 80486 is the first 1-million-transistor (1.2 million) processor packaged in 168-pin PGA packaging.
  - Compatible with all previous Intel x86 microprocessors.
- The 32-bit core of the 386 is preserved in the 486.
  - External data bus size was increased from 16 bits to 32.
    - Address bus size became 32 bits instead of 24.
  - The 486 data bus is D0–D31.
    - Address bus is A2–A31, in addition to BE0–BE3.

23.1: THE 80486 MICROPROCESSOR
enhancement 1 – pipelining

- Heavily pipelined instruction fetching & execution, means 486 executes many instructions in ***one*** clock cycle instead of in *three* clocks as in 386.
  - Large numbers of transistors split fetching/execution of instructions into many stages, all working in parallel.
    - Allowing processing of up to five instructions to be overlapped.

23.1: THE 80486 MICROPROCESSOR
enhancement 2 - cache

- Putting 8K cache with the CPU core, all on a single chip, 486 eliminates external cache interchip delay.
  - 486 has 8K of on-chip cache to store both code and data.
    - Off-chip cache (level two) is commonly called secondary cache.
- The 486 8K, on-chip cache, has 2-way set associative organization & is used for storing both data and code.
  - It uses the write-through policy for updating main memory.

23.1: THE 80486 MICROPROCESSOR
enhancement 3 - math coprocessor

- In 80486, the CPU and math coprocessor are part of a single IC chip
  - Reduced the interchip delay associated with multichip systems such as the 386 and 387.
  - Increased cost since 80486 is two chips in one.
    - The main CPU and math coprocessor.
- For many people who did not need a math coprocessor, Intel introduced the 80486SX.
  - The main CPU, and a separate math coprocessor named 80487SX.

23.1: THE 80486 MICROPROCESSOR
enhancement 4 – data parity

- 80486 uses 4 pins for data parity (DP), allowing implementation of parity error checking on the system board.
  - The four pins DP0, DP1, DP2, and DP3 are bidirectional.
    - Each is used for 1 byte of the D31–D0 data bus.

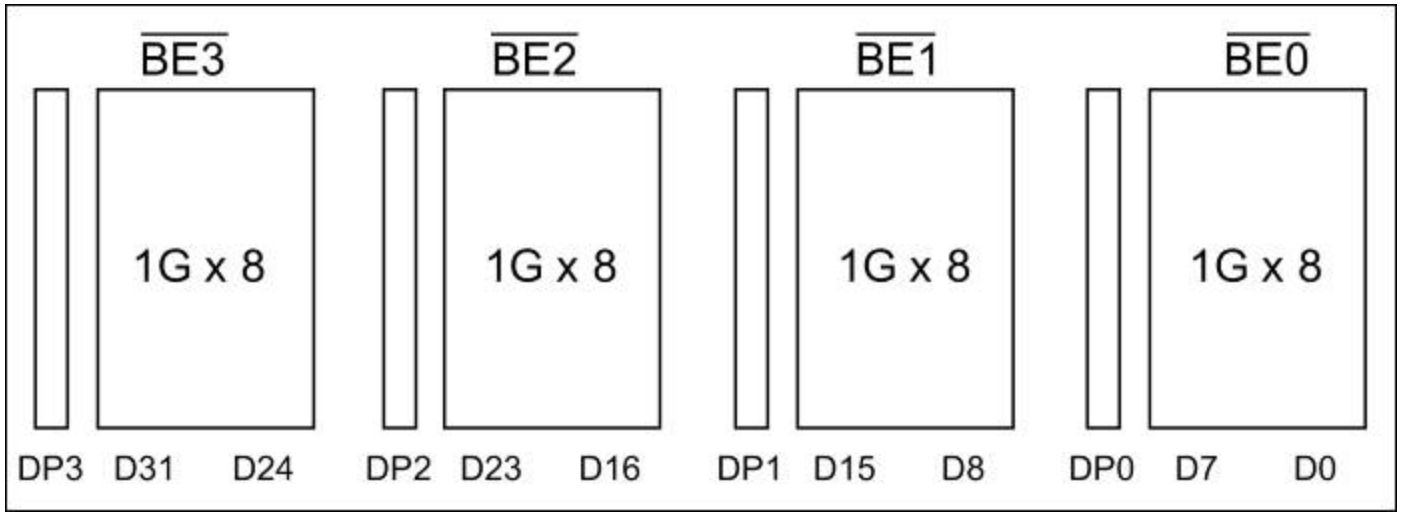


Fig. 23-1 Memory Organization with DP0-DP3

23.1: THE 80486 MICROPROCESSOR
enhancement 5 - burst mode

- The memory cycle time of the 486 with the normal zero wait states is 2 clocks.
  - To increase bus performance, Intel provides an additional option of implementing what is called a *burst cycle*.
    - 486 has two types of memory cycles, *nonburst* & *burst mode*.
  - In burst cycle, 486 performs 4 memory cycles in 5 clocks.

23.1: THE 80486 MICROPROCESSOR
enhancement 5 - burst mode

Two pins, **BRDY** (burst ready) and **BLAST** (burst last), specifically implemer the burst cycle.

BRDY is an *input* into the 486.

BLAST is an *output* from the 486.

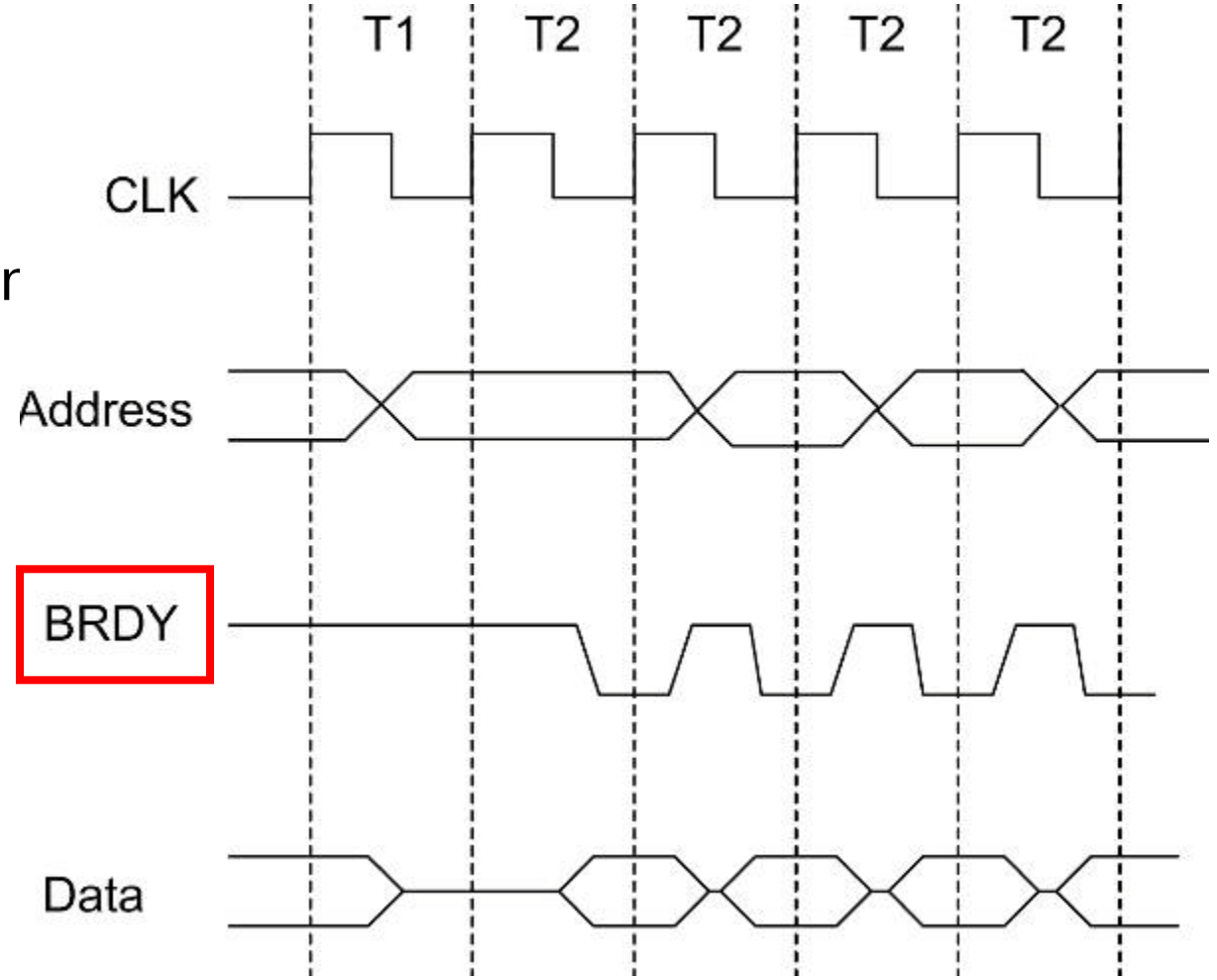


Fig. 23-2
Burst Cycle Read in the 486.

23.1: THE 80486 MICROPROCESSOR
enhancement 5 - burst mode

- Burst cycle can fetch a maximum of 16 bytes of information into the CPU in only 5 clocks.
  - Provided they are aligned on doubleword boundaries.

Example 23-1

Calculate and compare the bus bandwidth of the following systems. Assume that both are working with 33 MHz and that the 386 is 0 WS. Also assume that the data is aligned and is in 4 consecutive doubleword memory locations.
(a) 386 (b) burst mode of the 486

Solution:

(a) In the 386, since each memory cycle time takes 2 clocks we have memory cycle time = 2 (1/33 MHz) = 2 × 30 ns = 60 ns bus bandwidth = (1/60 ns) × 4 bytes = 66 megabytes/second
(b) In burst mode, the 486 performs 4 memory cycles in only 5 clocks; therefore, the average memory cycle time in burst mode is 1.25 (5/4 = 1.25) clocks for each 32-bit (doubleword) of data fetched as long as they are aligned and located in consecutive memory locations. This results in bus bandwidth = [1/(1.25 × 30 ns)] 4 bytes = 106.66 megabytes/second

23.1: THE 80486 MICROPROCESSOR
enhancement 6 – new instructions & CLK

- 486 supports all 386 instructions, with six new ones.
  - **INVD**, **INVLPG**, and **WBINVD** are added specifically for dealing with the on-chip cache and the TLB entries.
  - **XADD** first loads the destination operand into the source, then loads the sum of both destination & original source into the destination.
  - **CMPXCHG** compares the accumulator, AL, AX, or EAX, with the destination operand, either a register or memory.
  - **BSWAP** instruction converts the contents of a 32-bit register from the *little endian* to *big endian*, or vice versa.
    - See Example 23-2.
- In 80486, **CLK** is the *same* as the system frequency.

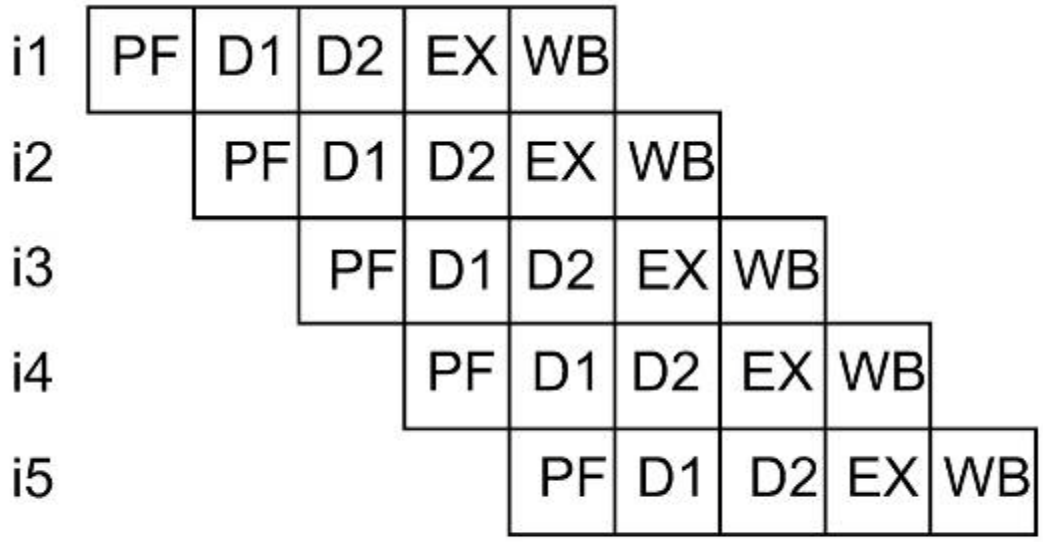
23.1: THE 80486 MICROPROCESSOR
more about pipelining

- In 80486, the pipeline stage is broken to 5 stages:
  - 1. Fetch. (prefetch)
  - 2. Decode 1.
  - 3. Decode 2.
  - 4. Execute.
  - 5. Register write-back.

A two-stage decoder is used for calculation & protection check of operand addresses.

Register write-back is the stage where the operand is finally delivered to the register.

23.1: THE 80486 MICROPROCESSOR
more about pipelining



PF = prefetch
D1 = decode 1
D2 = decode 2
EX = execute
WB = write back

Each stage takes 1 clock, but when the pipeline is full each instruction will execute in a single clock.

Fig. 23-3 486 Pipeline Stages

23.2: INTEL'S Pentium®

- Intel put 3.1 million transistors on a single piece of silicon using a 273-pin PGA package to design the next generation of x86.
  - It is called Pentium® instead of 80586 because it is hard to copyright a number such as 80586.

23.2: INTEL'S Pentium®

- There are three ways available to microprocessor designers to increase processing power of the CPU.
  - 1. Increase the clock frequency of the chip.
    - Higher frequency means more power dissipation, and more difficult/expensive design of microprocessor & motherboard.
  - 2. Increase the number of data buses to bring more information (code& data) into the CPU to be processed.
    - In today's PGA packaging, this is not a problem.
  - 3. Change the internal architecture of the CPU to overlap the execution of more instructions.
    - This requires a lot of transistors.

23.2: INTEL'S Pentium®

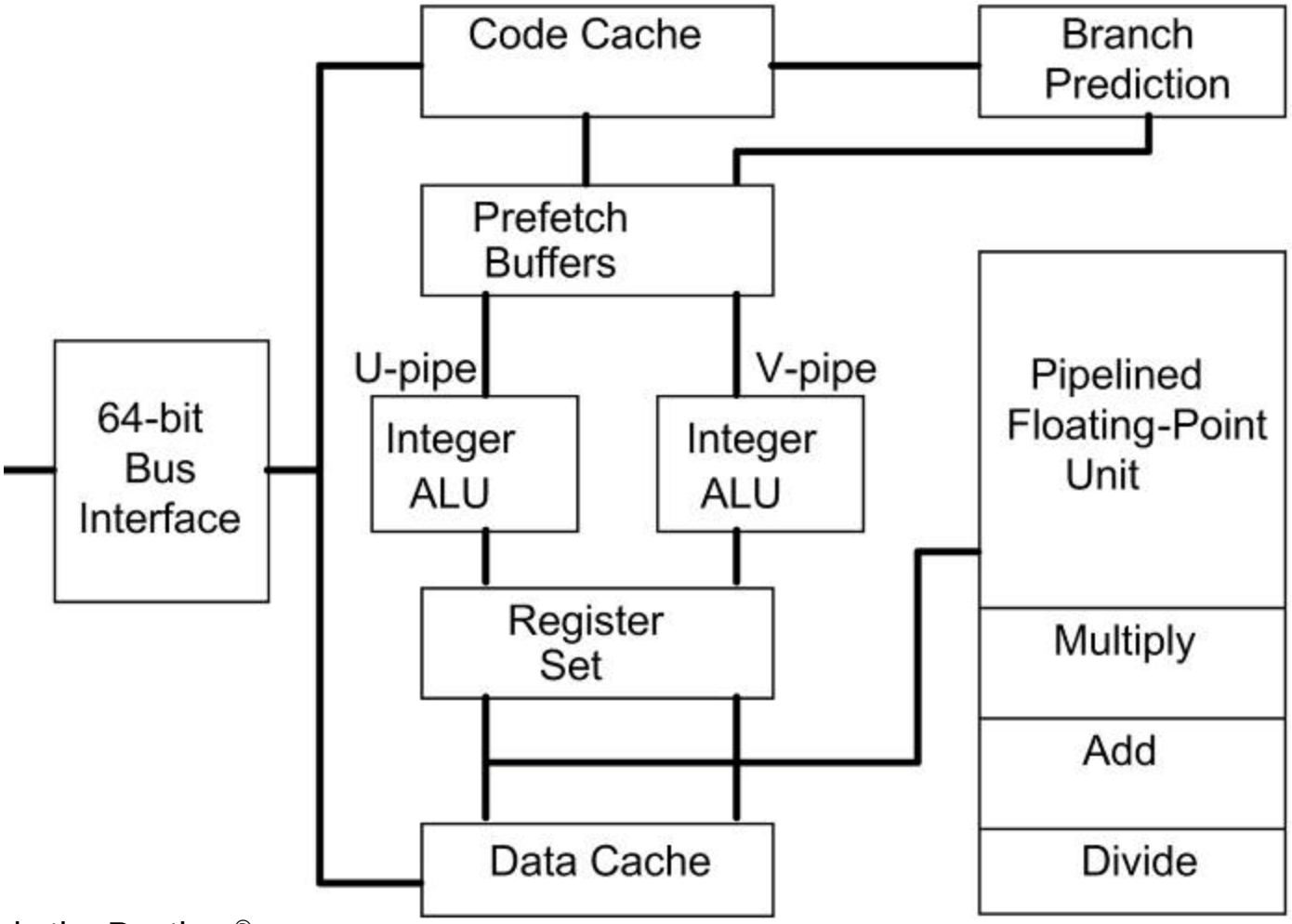


Fig. 23-4 Inside the Pentium®

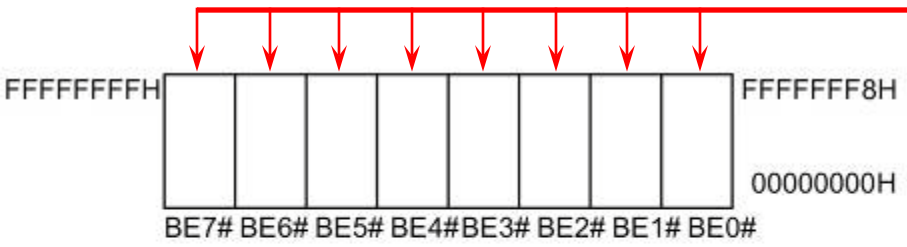
23.2: INTEL'S Pentium®

- There are two trends for changing internal architecture:
  - In **superpipelining**. fetching & executing instructions is split into many small steps, all done in parallel.
    - The execution of many instructions is overlapped.
    - Speed is limited to the the slowest stage of the pipeline.
  - In **superscaling**, the entire execution unit has been doubled and each unit has 5 pipeline stages.
    - Some superscalar processors have two execution units each with 4 pipeline stages.

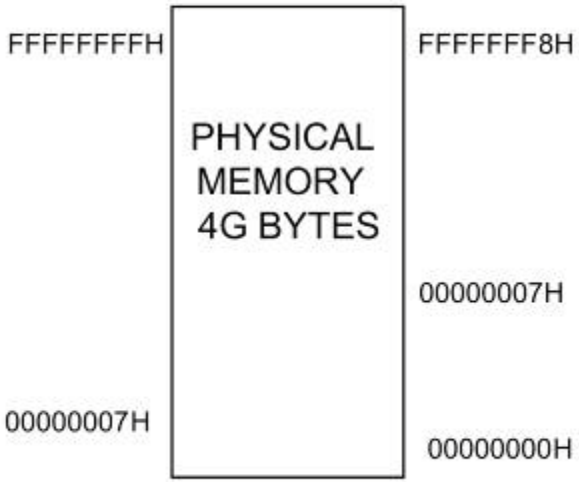
23.2: INTEL'S Pentium® feature 1 – 64-bits

- Pentium® external data buses are 64-bit, which will bring twice as much code/data into the CPU as 486.
  - Pentium® registers are 32-bit.
- Pentium® uses 64 pins, D0–D63, to access external memory banks 64 bits wide.
  - D0–D7 is the least significant byte.
  - D56–D63 is the most significant byte.
- Accessing 8 bytes of external data bus requires 8 **BE** (byte enable) pins, BE0–BE7.
  - Where **BE0** is for **D0–D7**, **BE1** for **D8–D15**, etc.

23.2: INTEL'S Pentium® feature 1 – 64-bits



64-BIT-WIDE MEMORY ORGANIZATION



PHYSICAL MEMORY SPACE

Fig. 23-5 Pentium® Memory Organization

Pentium® has 8 DP pins to handle the 8 bytes of data pins D0–D63.

Table 23-2: Pentium Byte Enable Signals

Byte Enable Signal	Associated Data Bus Signals
BE0#	D0–D7 (byte 0, the least significant)
BE1#	D8–D15 (byte 1)
BE2#	D16–D23 (byte 2)
BE3#	D24–D31 (byte 3)
BE4#	D32–D39 (byte 4)
BE5#	D40–D47 (byte 5)
BE6#	D48–D55 (byte 6)
BE7#	D56–D63 (byte 7, the most significant)

Pentium® has A31 to A3 for the address buses.

23.2: INTEL'S Pentium® feature 1 – 64-bits

- Like the 486, the Pentium® also has the A20M (A20 Mask) input pin for the implementation of HMA (high memory area).

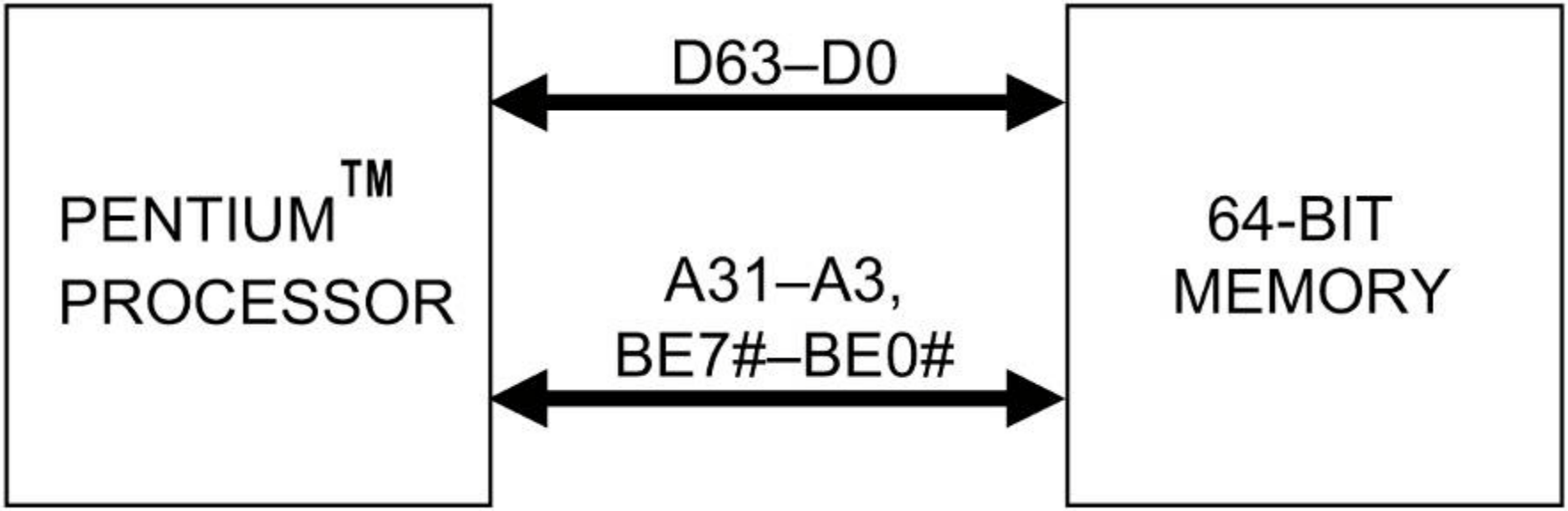


Fig. 23-5 Pentium® Address Buses

23.2: INTEL'S Pentium®
feature 2 – cache

- The Pentium® has a total of 16K bytes of on-chip cache—8K for code and 8K for data.
  - Data cache can be conFig.d write-back or write-through.
  - To prevent any accidental writing into code cache, code cache is write protected.
- When there is a cache miss for code cache, the CPU brings code from external memory & stores (writes) it in the cache code.
  - No instruction executing in the CPU can write anything into the code cache.
- Replacement policy for both caches is LRU.

23.2: INTEL'S Pentium®
feature 2 – cache

- Both on-chip data and code caches are accessed internally by the CPU core simultaneously.
  - Since there is only one set of address buses, external cache containing both data & code must be accessed one at a time, not simultaneously.
  - Some CPUs, notably RISC processors, use separate address & data pins (buses) for the data, and another set for the code section of the program.
    - Called Harvard architecture.
- Pentium® accesses the on-chip code & data caches simultaneously using Harvard architecture.
  - But not the secondary (external) off-chip cache and data.

23.2: INTEL'S Pentium®
feature 3 – coprocessor

- The on-chip math coprocessor of the Pentium® is many times faster than the one on the 486.
  - It has been redesigned to perform many of the instructions, such as add and multiply, ten times faster than the 486 math coprocessor.

23.2: INTEL'S Pentium® feature 3 – coprocessor

- On-chip coprocessors are commonly referred to as floating point units. (FPU)
  - Pentium® FPU section uses an 8-stage pipeline to process instructions.
- The section responsible for execution of integer-type data is called the integer unit (IU).

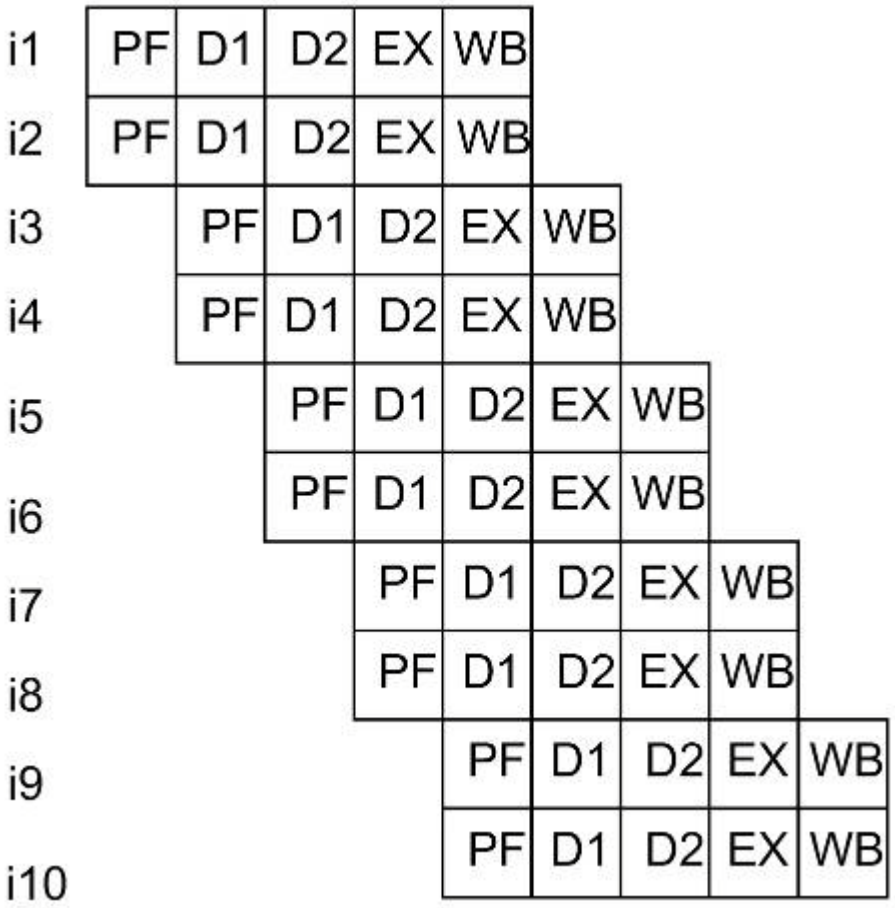


Fig. 23-7 Pentium Pipeline

23.2: INTEL'S Pentium®
feature 4 – superscalar architecture

- A unique feature of the Pentium® is superscalar architecture, in which alarge number of transistors were used to put two execution units in the CPU.
  - As instructions are fetched, they are issued to these two execution units.
- Issuing two instructions at the same time to different execution units can work *only* if the execution of one does not depend on the other one.
  - If there is no data dependency.

23.2: INTEL'S Pentium®
feature 4 – superscalar architecture

```
ADD    EAX,EBX        ;add EBX to EAX
NOT    EAX            ;take 1's complement EAX
INC    DI             ;increment the pointer
MOV    [DI],EBX       ;move out EBX
```

- ADD & NOT *cannot* be issued to two execution units, since EAX, the destination of the first instruction, is used immediately by the second.
  - This is called *read-after-write dependency*.
- The NOT instruction wants to read EAX, but must wait until after ADD is finished writing it into EAX.
  - ADD will not write into EAX until the last pipeline stage.
    - By then it is too late for the pipeline of the NOT instruction.

23.2: INTEL'S Pentium®
feature 4 – superscalar architecture

```
ADD    EAX,EBX        ;add EBX to EAX
NOT    EAX            ;take 1's complement EAX
INC    DI             ;increment the pointer
MOV    [DI],EBX       ;move out EBX
```

- This prevents the NOT instruction from advancing in the pipeline, causing the pipeline to be stalled until the ADD finishes writing.
- The NOT instruction can then advance.
  - This kind of register dependency raises the clock count from 1 to 2 for the NOT instruction.

23.2: INTEL'S Pentium®
feature 4 – superscalar architecture

```
ADD    EAX,EBX        ;add EBX to EAX
INC    DI             ;increment the pointer
NOT    EAX            ;take 1's complement of EAX
MOV    [DI],EBX       ;move out EBX
```

- *What if the instructions are rescheduled?*
  - Each can be issued to separate execution units, allowing parallel execution of both instructions by two different units of the CPU.
- Since the clock count for each instruction is one,, having two execution units leads to executing two instructions by pairing them together.
  - Using only one clock count for two instructions.

23.2: INTEL'S Pentium®
feature 4 – superscalar architecture

- The process of issuing two instructions to the two execution units is referred to as *instruction pairing*.
- The two integer execution units of the Pentium® are called "**U**" & "**V**" pipes—each has 5 pipeline stages.
  - The **U** pipe can execute any x86 instruction.
  - The **V** pipe executes only simple instructions:
    - INC, DEC, ADD, SUB, MUL, DIV.
    - NOT, AND, OR, EXOR, NEG.
- Simple instructions are executed in one clock as long as operands are "REG,REG" or "REG,IMM".
  - And have no register dependency.

23.2: INTEL'S Pentium®
feature 5 – branch prediction

- The penalty for jumping is very high for a high-performance pipelined processor like Pentium®.
  - Consider JNZ: if it jumps, the pipeline must be flushed and refilled with instructions from the target location.
    - This takes time.
  - In contrast, the instruction *immediately below* the JNZ is already in the pipeline and is advancing without delay.
- Pentium® has the capability to predict & prefetch code from *both* possible locations and have them advanced through the pipeline without waiting (stalling) for the outcome of the zero flag.

23.2: INTEL'S Pentium®
feature 6 – virtual memory

- Pentium® provides the option of 4K or 4M for the virtual memory page size.
  - The 4K page option makes it 386 and 486 compatible.
  - The 4M page size option allows mapping of a large program without any fragmentation.
    - The 4M page size reduces the frequency of a page miss in virtual memory.

23.2: INTEL'S Pentium®
feature 7 – translation lookaside buffer

- 386 & 486 have 32 entries for the TLB (translation lookaside buffer), which means the CPU has instant knowledge of the of only 128K of code and data.
  - If desired code or data is not referenced in the TLB, the CPU convert the linear address to a physical address.
- The Pentium® has two sets of TLB.
  - For data, the TLB has 64 entries for 4K pages.
  - TLB for code is 32 entries of 4K page size.
- Combining TLBs for code & data, Pentium® has quick access to 384K of code and data before it resorts to updating the TLB for the page miss.

23.2: INTEL'S Pentium®
feature 8 – burst mode

- Pentium® has burst read *and* burst write cycles.
  - Pentium® has features that lend themselves to implementation of multiple microprocessors (multiprocessors) working together.
  - It also has features called error detection and functional redundancy to preserve and ensure data/code integrity.

23.2: INTEL'S Pentium®
Intel's Overdrive™ technology

- To increase both CPU internal & external clock frequency requires faster DRAM, high-speed motherboard design, high-speed peripherals, and efficient power management due to a high level of power dissipation.
  - To resolve this, Intel developed the overdrive technology, also referred to as clock doubler & tripler.
    - To increase internal CPU frequency while the external frequency remains the same.
- The CPU processes code & data internally faster while the motherboard costs remain the same.

23.3: RISC ARCHITECTURE

- Since the 1960s, mainframe & minicomputers put as many instructions as they could into the CPU.
  - Some of these instructions performed complex tasks.
- These processors, using many instructions & performing highly complex activities, became known as CISC. (complex instruction set computer).
- The cost of implementing the many instructions, added to the 60% of transistors on the chip used by the instruction decoder, led to RISC systems.
  - RISC stands for (reduced instruction set computer).

23.3: RISC ARCHITECTURE
feature 1 – fixed instruction size

- RISC processors have a fixed instruction size.
  - The size of all instructions is fixed at 4 bytes (32 bits).
- In cases where instructions do not require all 32 bits, they are filled with zeros.
  - The CPU can decode the instructions quickly.

23.3: RISC ARCHITECTURE
feature 2 – load/store

- In RISC, instructions can only load from memory into registers or store registers into memory locations.
- There is no direct way of doing arithmetic & logic instructions between registers and contents of memory locations.
  - All these instructions must be performed by first bringing both operands into the registers inside the CPU, then performing the arithmetic or logic operation, and then sending the result back to memory.

23.3: RISC ARCHITECTURE
feature 3 – registers

- A major characteristic of RISC architecture is a large number of registers.
  - All RISC processors have 32 registers, r0–r31.
    - Each 32 bits wide.

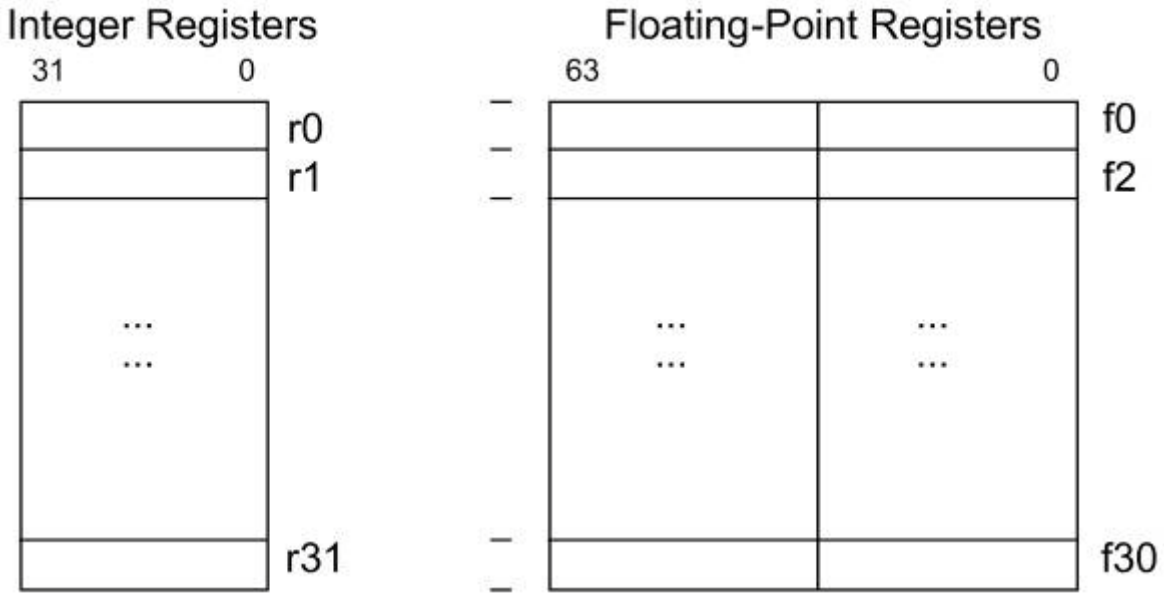


Fig. 23-8 RISC Integer and Floating Point Processors

23.3: RISC ARCHITECTURE
feature 3 – registers

- An advantage of a large number of registers is that it avoids the use of the stack to store parameters.
  - Although a stack can be implemented on a RISC processor, it is not as essential as in CISC since there are so many registers available.
- RISC processors, in addition to 32 general-purpose registers, also have another 32 registers for floating-point operations.
  - The floating-point register can be conFig.d as 64-bit in order to handle double-precision operands.

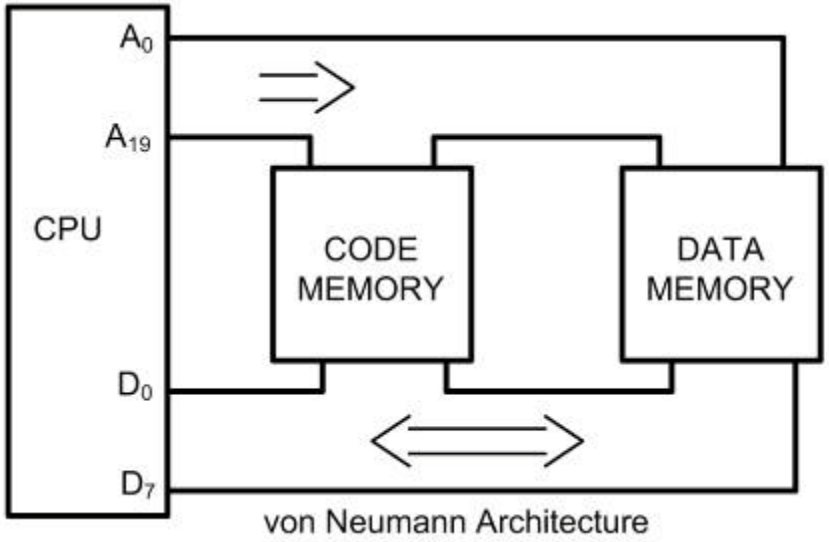
23.3: RISC ARCHITECTURE
feature 5 – gain

- *With all the difficulties associated with RISC programming, what is the gain?*
  - 99% of instructions are executed with only 1 clock.
    - The 1% of the RISC instructions executed with 2 clocks can be executed with *one* clock cycle by juggling instructions (code scheduling).

23.3: RISC ARCHITECTURE
feature 6 – hardwired instructions

- CISC has a large number of instructions, each with many different addressing modes, *microinstructions* (microcode) are used to implement them.
  - Implementation of microinstructions in the CPU takes more than 60% of transistors in many CISC processors.
- In RISC, due to the small instruction set, they are implemented using the hardwire method.
  - Hardwiring of RISC instructions takes about 10% of the transistors.

23.3: RISC ARCHITECTURE
Harvard and von Neumann architectures



von Neumann Architecture

Early computers used the same bus for accessing both the code and data.

Commonly referred to as *von Neumann* (Princeton) architecture.

To speed execution, some CPUs *Harvard architecture*.

Separate buses for the code and data memory.

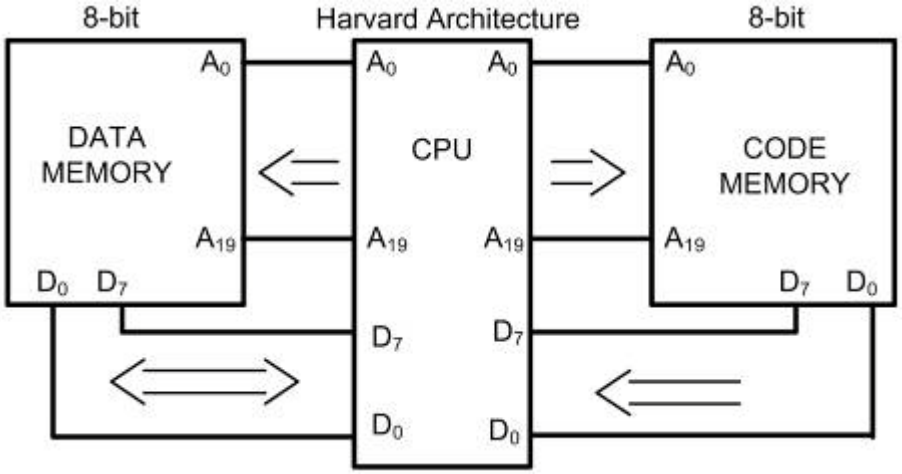


Fig. 23-9 von Neumann vs. Harvard Architecture

23.3: RISC ARCHITECTURE
IBM/Freescale RISC

- IBM and Freescale (formerly Motorola) together have a RISC processor called the Power PC 60x.
  - 2.8 million transistors, with a power consumption of 8.5 watts, vs. 16 watts in the Pentium.
    - Used for the last generations of Macintosh computers.
- No PC maker has used the RISC processors, since they must *emulate* to run Windows based software.
  - Rather than running native.
- Other notable RISC processors vying for a share of the embedded market:
  - ARM and PIC32.

23.4: Pentium® PRO PROCESSOR

- Pentium® Pro is the sixth generation of the x86 family, also referred to as P6.
  - Intel officially called this 1995, 5.5 million transistor, 150 Mhz chip, the Pentium® Pro.
- Intel also attached *level 2* (**L2**) cache to Pentium® Pro, all on a single package, with two separate dies.
  - Called dual cavity packaging by Intel, it reduces interchip delay between the L2 cache and the CPU.

23.4: Pentium® PRO PROCESSOR

Table 23-4: Comparison of Pentium and Pentium Pro

Feature	Pentium	Pentium Pro
Year introduced	1993	1995
Number of transistors	3.3 million	5.5 million
Number of pins	273	387
External data bus	64 bits	64 bits
Address bus	32 bits	36 bits
Physical memory (maximum)	4 GB	64 GB
Virtual memory	64 TB	64 TB
Data types (register sizes)	8, 16, 32 bits	8, 16, 32 bits
Cache (L1)	16K bytes (data 8K, code 8K)	16K bytes (data 8K, code 8K)
Cache (L2)	External	256KB/512KB
Superscalar	2-Way	3-Way
Number of execution units	3	5
Branch prediction	yes	yes
Out-of-order execution	no	yes

23.4: Pentium® PRO PROCESSOR internal architecture

- In Pentium® Pro, all x86 instructions are brought into the CPU, and broken down into one or more small and easy-to-execute instructions.
  - Called micro-operations (**µops**) by Intel.
    - Intel also had to maintain compatibility with all previous x86 processors, back to 8086.
- Converting x86 instructions into *micro-ops* internally uses what is called *triadic instruction format*.
  - In triadic instruction format, there are two source registers and one destination register.
    - A triadic instruction set means a large number of registers inside Pentium® Pro are not accessible/visible to the programmer.

23.4: Pentium® PRO PROCESSOR internal architecture

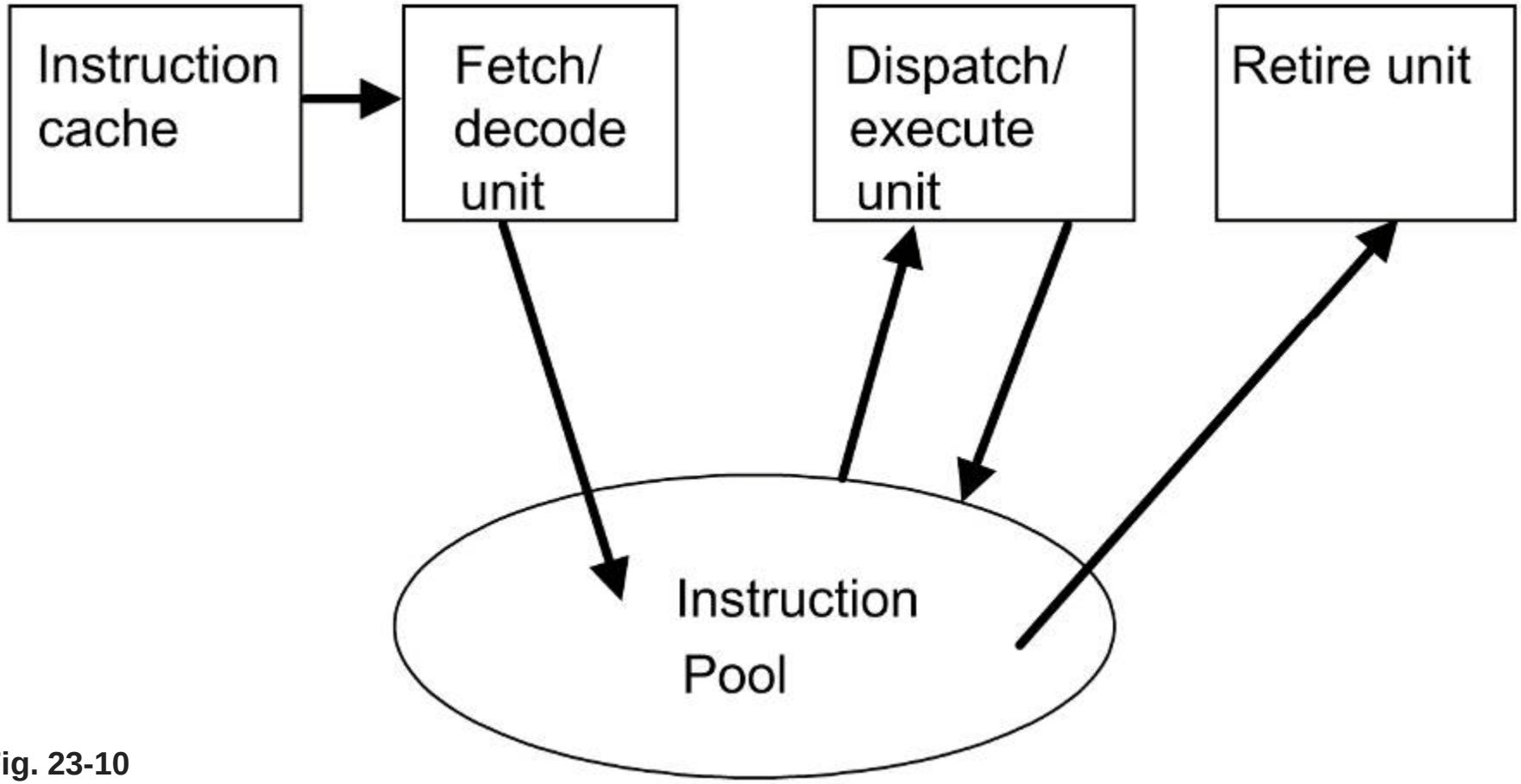


Fig. 23-10
Pentium Pro Instruction Execution

23.4: Pentium® PRO PROCESSOR both superpipelined and superscalar

- Pentium® Pro, with its 12-stage pipeline is referred to as s*uperpipelined*.
  - Since it also has multiple execution units capable of working in parallel, it is also *superscalar*.

23.4: Pentium® PRO PROCESSOR
what is out-of-order execution?

- In the Pentium®, when one pipeline stage is stalled, prior stages of fetch & decode are also stalled.
  - The fetch stage stops fetching instructions if the execution stage is stalled.
  - This dependency of fetch and execution has to be resolved in order to increase CPU performance.
    - Intel has done this with the Pentium® Pro, called *decoupling* the instructions fetch & execution phases.
- In Pentium® Pro, as instructions are fetched from memory, they are decoded (converted) to a series of micro-ops, or RISC-type instructions.
  - And placed into a pool called the *instruction pool*.

23.4: Pentium® PRO PROCESSOR
what is out-of-order execution?

- Fetch/decode of instructions is done in the same order as the program was coded.
  - When micro-ops are placed in the instruction pool, they can be executed in any order, as long as the data needed is available.
    - If there is no *dependency*, instructions are executed out of order, not the same order as the programmer coded them.
- Instructions are fetched in the same order they were coded, but are executed out of order if there is no dependency.
    - Ultimately retired in the same order as they were coded.
- Out-of-order execution *can* boost performance.

23.4: Pentium® PRO PROCESSOR branch prediction

- When Pentium® Pro encounters branch instructions (such as JNZ), it creates a list, in what is called the branch target buffer (BTB).
  - BTB predicts the target of the branch & starts executing from there.
    - When the branch *is* executed, the result is compared with what the prediction section of the CPU said it would do.

23.4: Pentium® PRO PROCESSOR bus frequency vs. internal frequency

- Frequently you may see an advertisement for a 1-GHz or 2-GHz Pentium® PC.
  - It is important to note that stated frequency is the internal frequency of the CPU—not the bus frequency.
- Such a design requires a very fast logic family and memory in addition to a massive simulation to avoid crosstalk and signal radiation.
  - Bus frequency for such systems is less than 1 GHz.

23.5: MMX TECHNOLOGY
DSP and multimedia

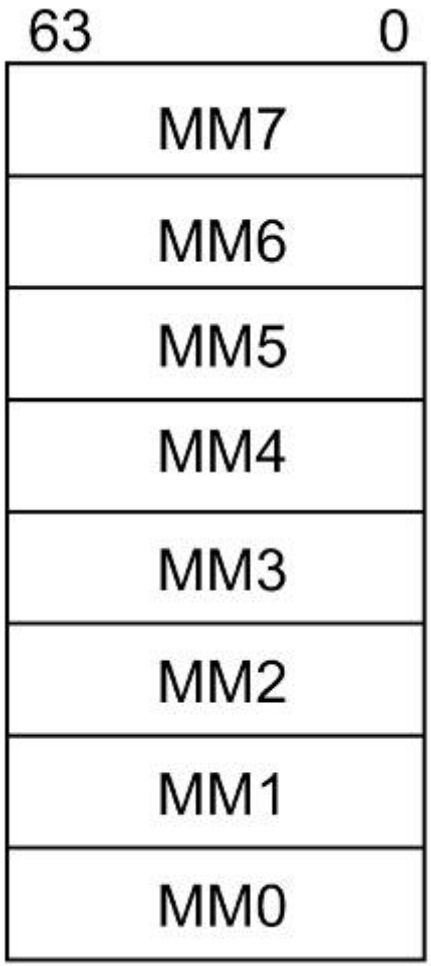
- High-quality multimedia applications with sound and graphics are normally performed by a highly specialized chip called DSP (digital signal processing).
  - DSP chips are the main engines performing tasks such as 2D and 3D graphics, video and audio compression, fax/modem, PC-based telephoning with live pictures, and image processing.

23.5: MMX TECHNOLOGY
DSP and multimedia

- There are three approaches to DSP capability:
  - 1. A full-fledged DSP chip on the board with the CPU.
    - The best and ideal approach since there are some very powerful DSP chips out there.
  - 2. Use the x86 & x87 FP (floating-point) instructions to emulate the function of DSP.
    - Slow and the performance is unacceptable.
  - 3. Incorporate some DSP functions into the x86.
    - In early 1997 Intel introduced a series of chips with somewhat limited DSP capability called MMX technology.

23.5: MMX TECHNOLOGY
register aliasing by MMX

- A main goal of MMX technology was to maintain compatibility with x86 processors with no MMX capability.
  - Intel uses the FP (floating-point) register set of the x87 coprocessor as the working register for MMX instructions instead of a whole new set of registers.
    - Called register aliasing.



23.5: MMX TECHNOLOGY
register aliasing by MMX

- Major implications of register aliasing by MMX:
  - 1. Registers must not be used to store MMX data & FP (floating point) data at the same time since they are the same physical registers.
  - 2. MMX instructions & FP instructions must not mix.
    - Slows down the application since it takes many clock cycles to switch between MMX and x87 instructions.
  - 3. On leaving an MMX program module, all MMX registers must be cleared before issuing any x87 instructions.
    - All FP registers must be popped to leave them empty.
  - 4. When FP registers are accessed by MMX instructions, each is accessed directly by its name, MM0–MM7.

23.5: MMX TECHNOLOGY
data types in MMX

- MMX uses 64 bits of the 80-bit wide FP registers.
  - The largest MMX data size is 64-bit, and can be used for four different data types:
    1. Quadword (one 64-bit)
    2. Packed doubleword (two 32-bit)
    3. Packed word (four 16-bit)
    4. Packed byte (eight 8-bit)

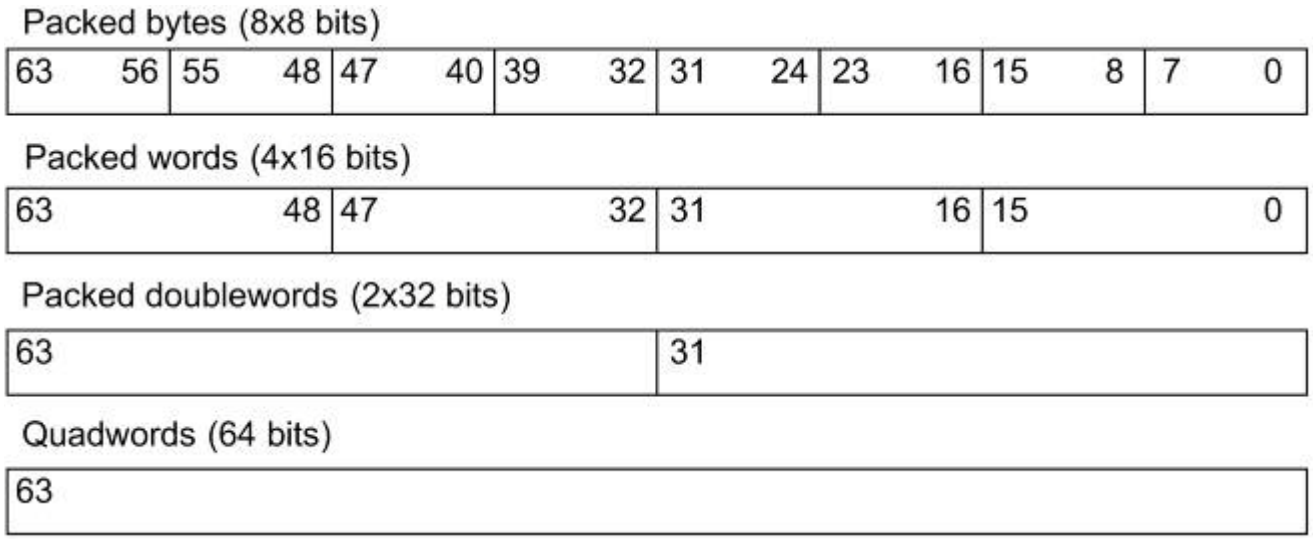


Fig. 23-12 MMX Data Types

23.5: MMX TECHNOLOGY
CPU identification for x86

- CPU identification is so important to new operating systems and software that starting with the Pentium®, Intel has introduced a new instruction.
  - The problem is identifying processors prior to Pentium®.
    - According to Intel, one must examine bits of the flag register.

Table 23-5: Flag Bits for CPU Identification

CPU	Flag Bits
8088/86	Bits 12 through 15 are always 1.
80286	Bits 12 through 15 are always 0 (in real mode).
80386	Bit 18 is always 0 (in real and protected mode).
80846	Bit 21 cannot be changed , therefore it is 486; if bit 21 can be changed to 1 and 0, then it must be a Pentium .
Pentium and Pentium Pro	Starting with the Pentium , one can use a new instruction , CPUID, to get information such as family and model of the processor . However, it is the ability to set or reset bit 21 of the flag that indicates whether the CPUID instruction is supported or not . The CPUID instruction can be executed any time in protected mode or real mode . For Intel's Pentium and higher microprocessors , prior to execution of the CPUID instruction we must set EAX = 1. After the execution of CPUID , bits D8– D11 of EAX have the family number . The family number is 5 for the Pentium and is 6 for Pentium Pro.

23.5: MMX TECHNOLOGY
CPUID instruction and MMX technology

- Not all Pentium® and Pentium® Pro microprocessors come with MMX technology.
  - To determine if a microprocessor is equipped with MMX technology, use Pentium® instruction CPUID.
    - According to Intel, on return from instruction CPUID, if D23 of EDX is high, the CPU has MMX technology.
  - MMX identification is performed in Program 23-1, on Page 617

Dec Hex Bin
23 17 00010111

ENDS ; TWENTY-THREE

The x86 PC

assembly language, design, and interfacing

fifth edition

MUHAMMAD ALI MAZIDI
JANICE GILLISPIE MAZIDI
DANNY CAUSEY

The x86 PC
assembly language, design, and interfacing
fifth edition

Prentice Hall