Chapter
2

Microprocessor Architecture

Contents

Chapter 2

Microprocessor Architecture

2-1. Introduction

The microprocessor architecture is the framework and the conceptual design of the microprocessor structure. There are two basic architectures of microprocessors, namely:

- The **Harvard architecture**: This architecture has two physically different memory storages for Data and Instruction (code). Although early computers used this architecture, it is still used in microcontrollers units (MCU), which are complete microprocessor systems on a chip.
- **Princeton** architecture (Von-Neumann architecture): This architecture has single memory storage for both data and instructions. .

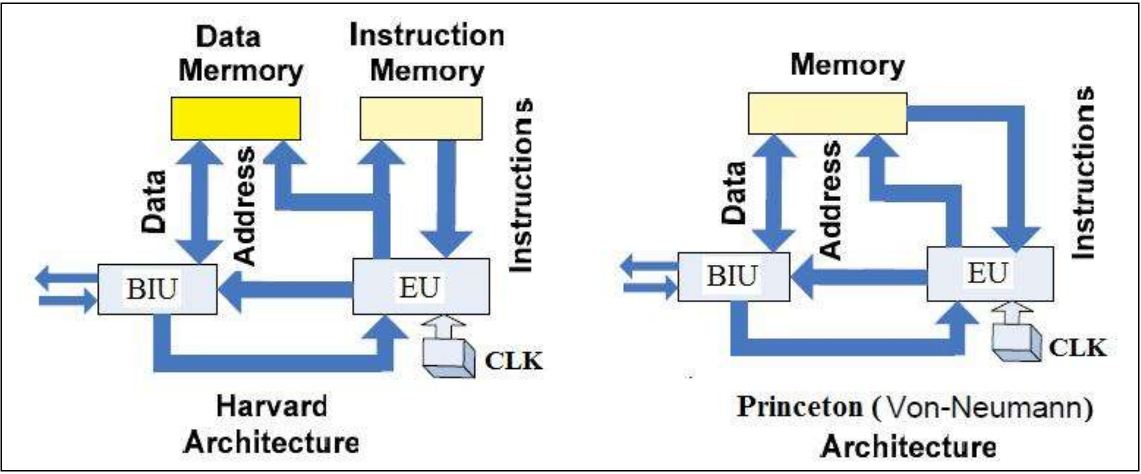


Fig. 2-1. Basic architecturess of microprocessors

Another way to divide computer architectures is into *Complex Instruction Set Computer* (CISC) and *Reduced Instruction Set Computer* (RISC). A CISC approach may only a single instruction taking values from memory, performing the addition internally and writing the result back. This means the instruction may take many cycles. A RISC approach is only performing operations on values in registers and explicitly loading and storing values to and from memory. All modern architectures would be considered RISC architectures.

2-2. Architecture of the 8086 / 8088 Microprocessors

The real PC history started with the introduction of Intel's 8086 microprocessor in 1978 and the 8088 in 1979. The 8086 is a 16-bit microprocessor, which means it processes 16-bit data (2 bytes), at a time. The 8086 was built in 6 micron NMOS technology and had 29,000 transistors on a single chip, with 40 pin package.

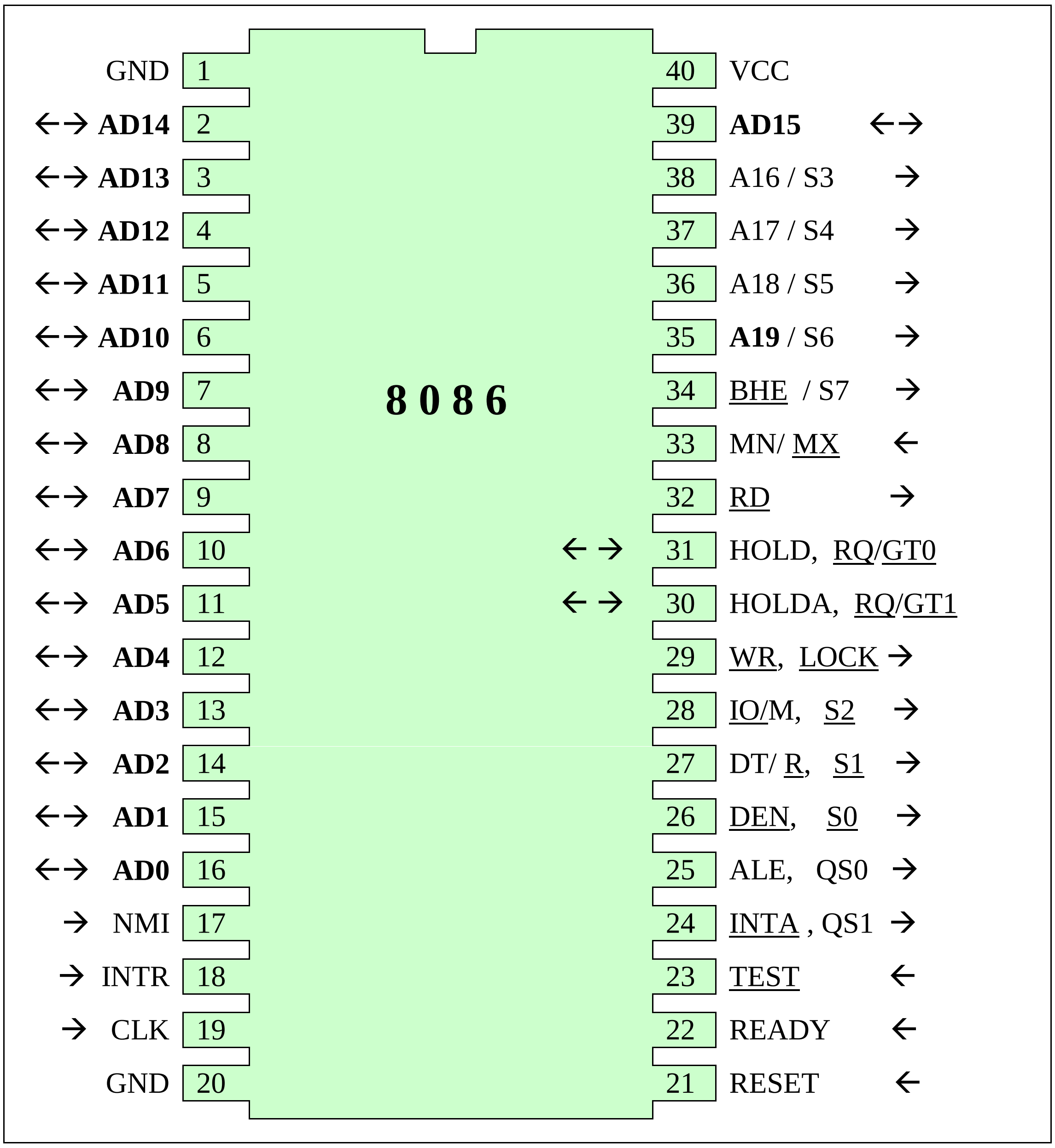


Fig. 2-2. Pin-out diagram of the Intel 8086 microprocessor.

Figure 2-2 shows the pin-out diagram of the 8086 microprocessor and table 2-1 depicts the functions of each pin.

As shown, the 8086 microprocessor, has a 16-bit data bus (D0 through D15), and a 20-bit address bus (A0 through A19), and hence can address up to 1 MB (2^{20} Bytes) of memory. The 16-bit data lines of the 8086 are multiplexed with the first 16 lines of the address bus (named AD0 through AD15). The clock frequency of the 8086 has been rated from 3 to 10 MHz.

Table 2-1. Pin assignment of 8086 microprocessor, in the so called minimum mode of operation. In this mode pin33 should be connected to ground.

Pin #	Pin Name	Function
1	GND	Ground
2	AD14	Address/Data bus
3	AD13	Address/Data bus
4	AD12	Address/Data bus
5	AD11	Address/Data bus
6	AD10	Address/Data bus
7	AD9	Address/Data bus
8	AD8	Address/Data bus
9	AD7	Address/Data bus
10	AD6	Address/Data bus
11	AD5	Address/Data bus
12	AD4	Address/Data bus
13	AD3	Address/Data bus
14	AD2	Address/Data bus
15	AD1	Address/Data bus
16	AD0	Address/Data bus
17	NMI	Non-maskable interrupt
18	INTR	Interrupt Request
19	CLK	Clock
20	GND	Ground
21	RESET	Reset signal
22	READY	Ready signal
23	$\overline{\text{TEST}}$	Test signal
24	$\overline{\text{INTA}}$	Interrupt Acknowledge
25	ALE	Address Latch Enable
26	$\overline{\text{DEN}}$	Data Enable
27	DT/ $\overline{\text{R}}$	Data Transmit/Receive
28	$\overline{\text{IO}}$/M	Input Output/ Memory
29	$\overline{\text{WR}}$	Write
30	HOLDA	Hold Acknowledge
31	HOLD	Hold
32	$\overline{\text{RD}}$	Read

Pin #	Pin Name	Function
33	MN/ MX	Min / Max mode
34	BHE / S7	Bus High Enable /Status
35	A19 / S6	Address bus / status
36	A18 / S5	Address bus / status
37	A17 / S4	Address bus / status
38	A16 / S3	Address bus / status
39	AD15	Address / Data bus
40	VCC	Vcc (+5V)

Figure 2-3 depicts the 8086 internal architecture. The Intel **8086** was based on the design of the Intel 8080 and Intel 8085 (it was source compatible with the 8080) with a similar register set, but was expanded to 16 bits.

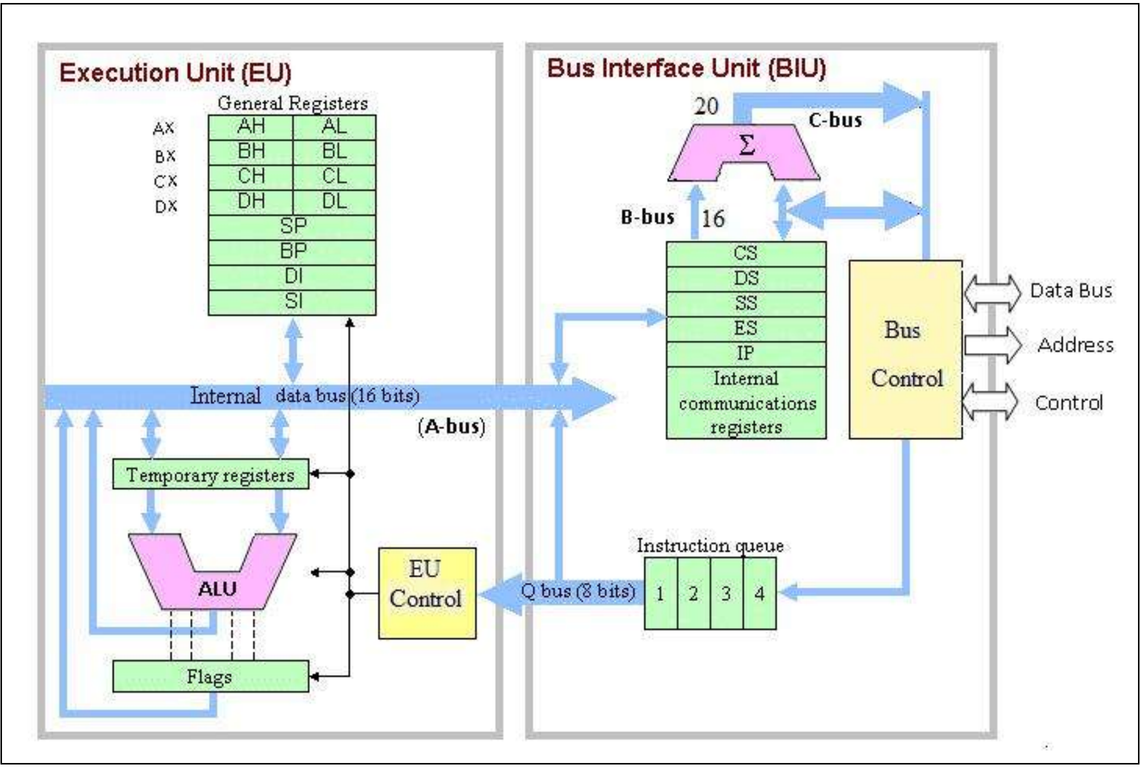


Fig. 2-3. Architecture of the Intel 8086/8088 microprocessors

The Bus Interface Unit (**BIU**) feeds the instruction stream to the Execution Unit (**EU**) through a 6-byte prefetch queue. So fetch and execution are concurrent (a primitive form of pipelining, which is a method of parallel processing). In fact the 8086 instructions vary from 1 to 4 bytes. It features 64k x 8-bit (or 32k x 16-bit) I/O ports and fixed vectored interrupts.

The Intel **8088** is a version of **8086**, with an 8-bit external data bus. Even though its internal architecture has also an internal 16-bit data bus, the 8088 is 30% slower than the 8086. The original IBM PC made use of the 8088 at a clock rate of 4.77 MHz. In 8088 microprocessors, the address and data busses are only multiplexed in the first 8 lines (AD0-AD7).

The **ALE** (address latch enable) signal, on pin 25, indicates whether the multiplexed address/data lines are carrying address or data. When the microprocessor sends an address, the ALE is set high. When the microprocessor sends or receives data, the ALE is set low. In order to de-multiplex the address signals (A0-A15) from (AD0-AD15), we use an octal **latch** (e.g., **74LS374**). The block diagram of 74LS374 is shown in figure (2-4). The 74LS373 octal latch is the same as 74LS374, except that the clock is replaced by a special enable line (control). The latch has to be enabled by the ALE signal as shown in figure 2-4.The bi-directional data lines (D0-D15) can be driven by the octal tri-state buffer **74LS245**, whose block diagram is shown in figure 2-5. Alternatively, the data lines (D0-D15) can be driven by the 8286 bus transceiver chip. In each case two chips are needed. The $\overline{\text{BHE}}$ pin (pin 34 in 8086) is used to indicate whether data appearing on AD8-AD15 should be latched or not. In fact, both BHE and A0 are used to indicate how data appear on data bus. The four possibilities are shown in the following table. Note that the $\overline{\text{BHE}}$ pin doesn't exist in 8088, which has only 8-bit external data bus.

Table 2-2. BHE pin signals.

Condition	**$\overline{\text{BHE}}$**	**A0**	Data on
R/W 16-bit at even address	0	0	AD0-AD15
R/W 16-bit at odd address	0	1	AD0-AD15
R/W 8-bit at even address	1	0	AD0-AD7
R/W 8-bit at odd address	1	1	AD0-AD7

The 8086/8088 microprocessors can operate in two basic modes:

- **Minimum mode**, is enabled by connecting pin33 (MN/$\overline{\text{MX}}$) to logic 1
- **Maximum modes**, is enabled by connecting pin 33 to logic 0.

Table 2-3 depicts the pin assignment of pins 24-31 in minimum and maximum modes. In minimum mode, the 8086/8088 processors work as stand-alone processors, that generate their own control signals. So, in minimum mode the control pins 24 to 31 are used as I/O control signals (INTA, ALE, DEN, DT/R, IO/M, WR, HOLD, HOLDA) which are generated from the 8086 (as 8085A control signals). Thus, in minimum mode, the 8085A peripheral support chips can be also used with the 8086/8088 microprocessors.

Table 2-3. Control signals of the 8086/8088, in minimum and maximum modes.

Pin	**Function**	
	Minimum Mode	**Maximum** Mode
31	HOLD (Hold for DMA request)	RQ/GT0 (Request/Grant)
30	HLDA (Hold Acknowledge)	RQ/GT1 (Request/Grant)
29	WR (Write control)	LOCK (Lock bus)
28	IO/M (I/O or Memory control , 8086) IO/M (I/O or Memory control , 8088)	S2 (Status signal)
27	DT/R (Data transmit/receive)	S1 (Status signal)
26	DEN (Data enable)	S0 (Status signal)
25	ALE (Address latch enable)	QS0 (Queue status)
24	INTA (Interrupt acknowledge)	QS1 (Queue status)

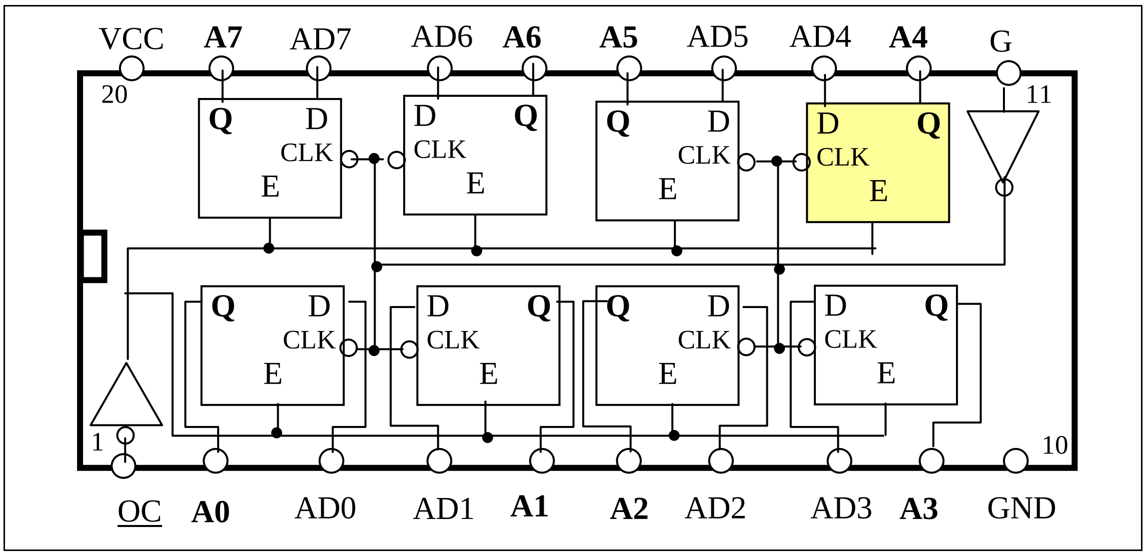


Fig. 2-4. Block diagram of the 74LS374 octal latch.

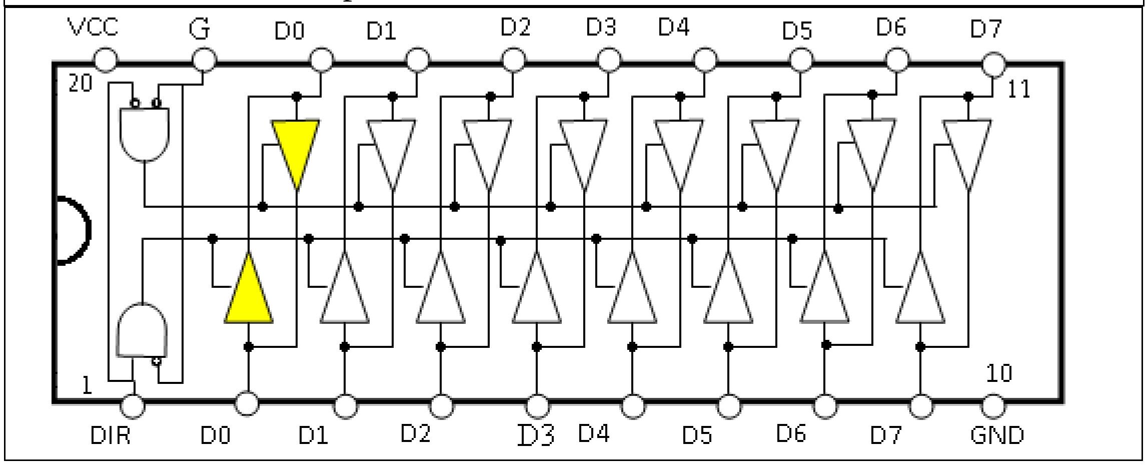


Fig. 2-5. Block Diagram of the 74LS245 octal tri-state buffer.

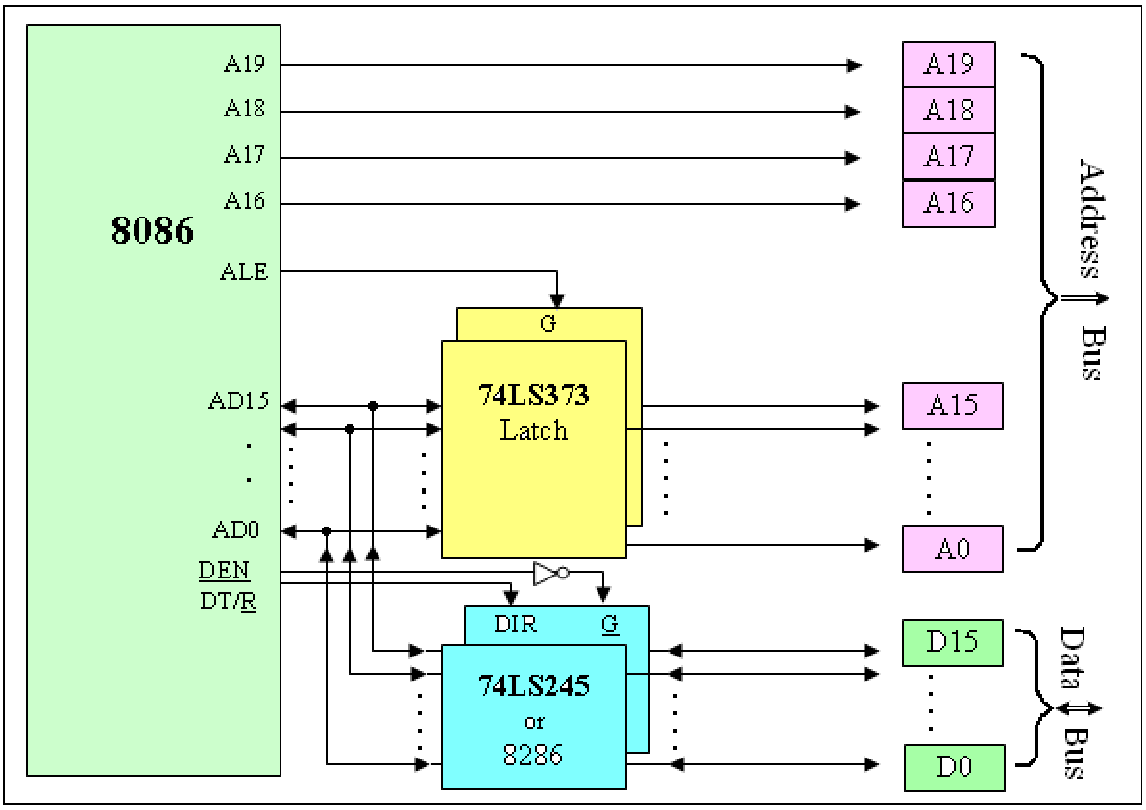


Fig. 2-6. Address de-multiplexing from address/data lines of 8086 processors.

Figure 2-7 depicts how the control signals MEMR, MEMW, IOR, and IOW can be generated from IO/M, WR and RD signals, for 8086/8088 microprocessors in minimum mode. Note that IO/M in 8086 is replaced with IO/M in 8088. The IBM PC/XT, made use of the 8088 in its maximum mode, while the IBM PCJR was running its 8088 in minimum mode.

In maximum mode, some I/O control signals have to be generated externally, using the **8288 bus controller** chip. In this case, the 8288 chip uses the S0, S1, S3 signals (pins 26-28) to generate I/O controls (INTA, IOR, IOW and other signals). Note that 8288 output signals MRD and MWT are equivalent to the usual control signals MEMR and MEMW, respectively. However, in this case, some new features are available, like the possibility of adding a coprocessor chip (8087) to the system.

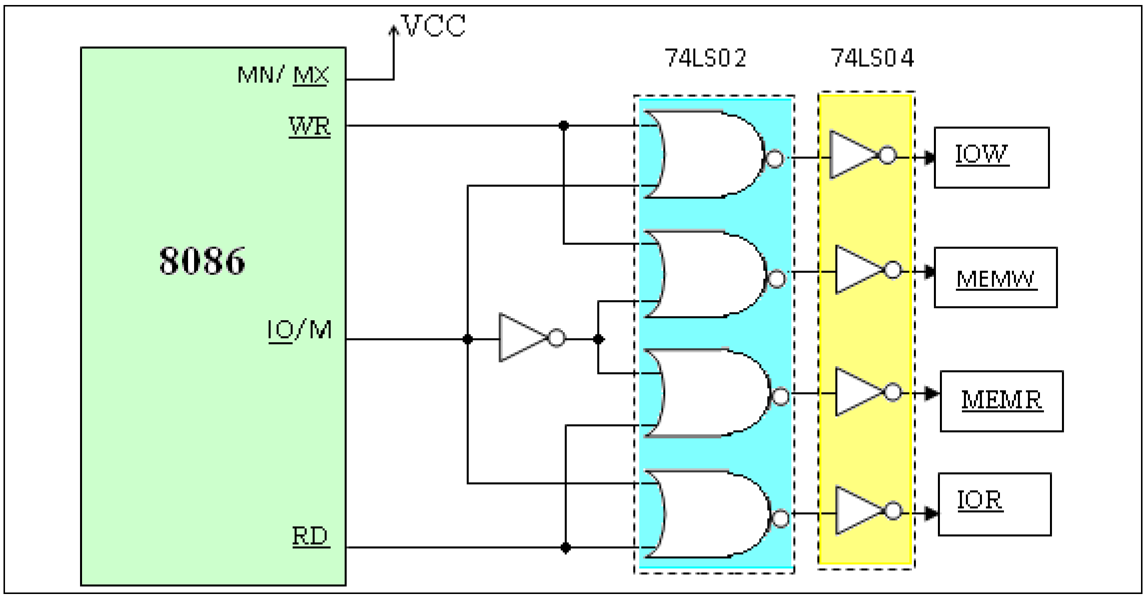


Fig. 2-7(a). Generating control bus signals in 8086 minimum mode.

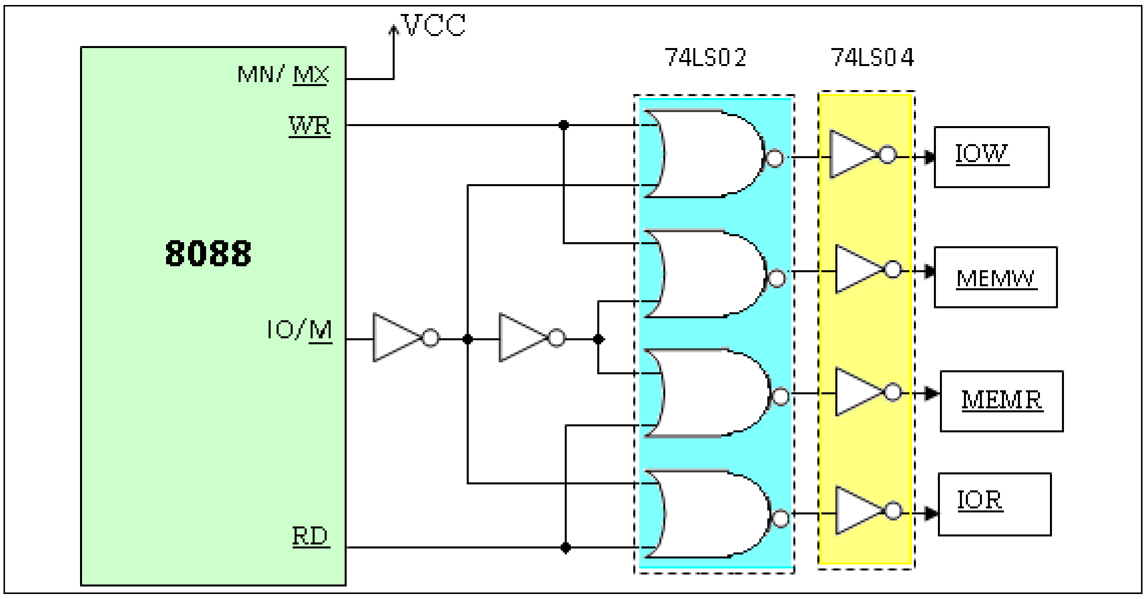


Fig. 2-7(.b). Generating control bus signals in 8088 minimum mode.

The **RQ/GT0**, and **RQ/GT1** pins coordinate the use of other processors (e.g., the math coprocessor 8087 or I/O processors 8089) on the same 8086 busses, in maximum mode. The request and grant RQ/GT0, RQ/GT1 lines provide some sort of handshaking between other processors and the 8086. The **LOCK** signal (pin 29) is used in the maximum mode to prevent other processors from gaining control on the system bus. In fact, some 8086 instructions need that no other processor can intervene or takes over the busses, while they are executed. Note that bus mastering in minimum mode is performed by **HOLD** and **HLDA** signals (instead of RQ/GT0, RQ/GT1).

The queue status lines **QS0**, and **QS1** tell whether the 8086 is taking its next byte from internal byte queue or from external memory. The **Ready** pin is usually pulled low when the 8086 is communicating with a slow memory or input/output devices. When the memory or input/output device finishes the reading or the writing operation it pulls ready high.

The 8086/8088 processors have four 16-bit general purpose registers, which could also be accessed as eight 8-bit registers, and four 16-bit index registers (including the stack pointer). There are also four segment registers. As shown in figure 2-3, the 8086/8088 registers are all 16-bits, as follows:

i) 4 General-purpose Registers : AX, BX, CX, DX
*AX (AL/AH): Accumulator, used in multiplication, division and I/O
*BX (BL/BH): Base register,
*CX (CL/CH): Count register, used as counter in loop operations
*DX (DL/DH): Data register, used in multiplication, division and I/O

ii) 5 Pointer / Index Registers:
SP: Stack Pointer, **BP**: Base Pointer, **IP**: Instruction Pointer
SI: Source Index, used as pointer in string operation
DI: Destination Index, used as pointer in string operation

iii) 4 Segment Registers : to access memory and I/O
CS: Holds Code segment address (the beginning of the code segment)
DS: Holds Data segment address (the beginning of the data segment)
SS: Holds Stack segment address (the beginning of the stack segment)
ES: Extra segment

iv) 1 Flag Register **FLAGS**: to monitor the result of ALU instructions.

The FLAGS register of the Intel 8086/8088 is described in figure 2-8. FLAGS is unlike other registers of the 8086. This register is simply a collection of one-bit values (flip flops), which help us to determine the current state of the processor. Although the FLAGS register is 16-bits wide, the 8086 processor uses only **nine** of those bits. Of these flags, four flags you use all the time: carry **C**, zero **Z**, sign **S**, and overflow **O**. For example, after an addition of two numbers, if the sum is larger than 8 bits, the Carry flag (CF) is set to one. When an arithmetic operation results in zero, the Zero flag (ZF) is set to one.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	**-**	**-**	**-**	**OF**	**DF**	**IF**	**TF**	**SF**	**ZF**	**-**	**AF**	**-**	**PF**	**-**	**CF**

- **CF**: Carry Flag (contains carry out of MSB of result)
- **PF**: Parity Flag (indicates if result has even parity)
- **AF**: Auxiliary carry Flag (contains carry out of bit 3 in AL)
- **ZF**: Zero Flag (indicates if result is zero)
- **SF**: Sign Flag (indicates if result is negative)
- **OF**: Overflow Flag (indicates if overflow occurred in result)
- **IF**: Interrupt Flag (indicates if interrupt is enabled or disabled)
- **DF**: Direction Flag (controls pointer updates during string operation)
- **TF**: Test Flag (provides single-step execution capability for debugging)

Fig. 2-8. Illustration of the FLAGS register in Intel 8086/8088 microprocessors.

The instruction pointer (**IP**) is a 16-bit register which contains the address of the current executing instruction. The microprocessor uses this register to sequence the execution of instructions. The **IP** and **FLAGS** are two special registers on the 8086 CPU. You do not access these registers the same way you access the other registers. Instead, it is the CPU which manipulates these registers directly.

The 8086/88 microprocessors can handle **hardware interrupts**, via **INTR** (interrupt request) and **NMI** (non-maskable interrupt) pins. Interrupts are external events (like mouse movement or memory failure) that need CPU attention. When NMI pin is edge triggered, the CPU will finish its current instruction and handle the interrupt. Similarly, when the INTR is activated, by an external device, the CPU will finish its current instruction and respond by interrupt acknowledge signal (**INTA**). The INTA signal is received by an **interrupt controller** (the **8259** chip), as shown in Fig. 2-8. This chip puts an interrupt vector byte on the data bus and the CPU uses it to determine the address of the appropriate interrupt service routine (**ISR**).

The interrupt signal INTR, can be masked (and hence ignored) if interrupt flag (**IF**) is cleared by CLI instruction. Figure 2-9 depicts how the microprocessor handles the hardware interrupt signals.

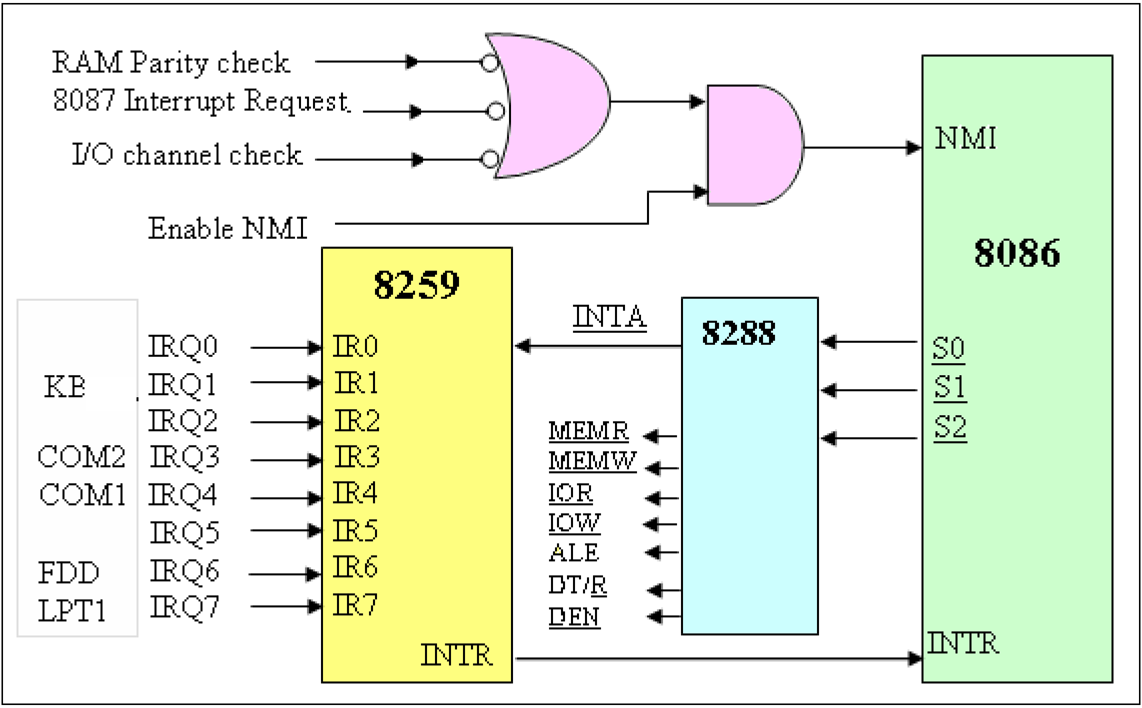


Fig. 2-9. Handling hardware interrupts in 8086/8088 systems, using the 8259 interrupt controller chip. The INTA signal and other control signals are generated in the maximum mode via the 8288 bus controller.

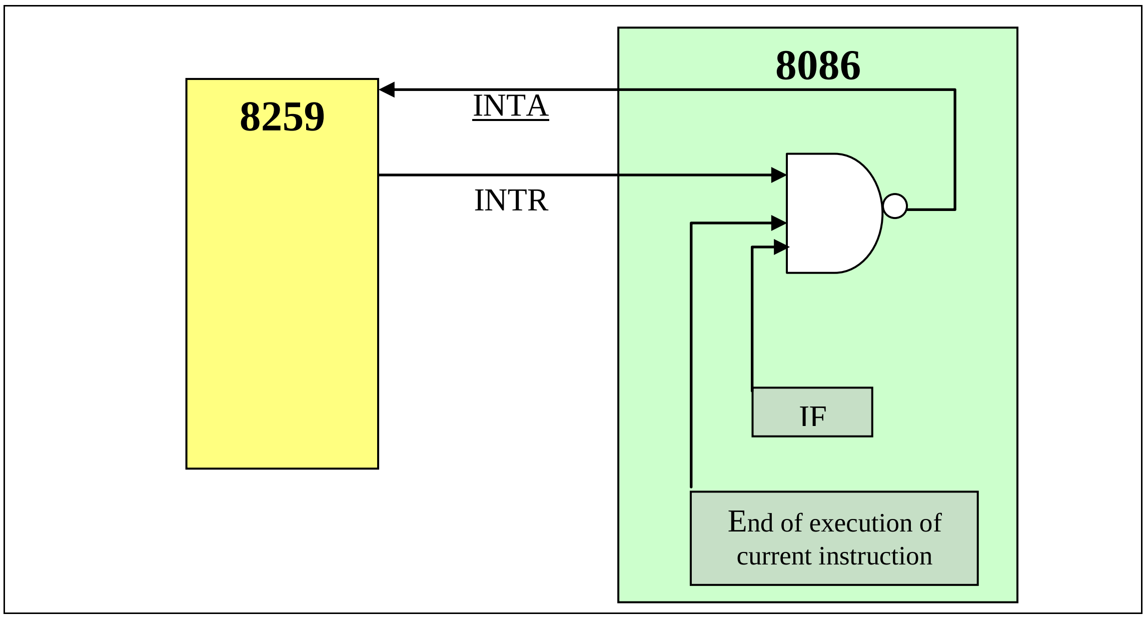


Fig. 2-10. Gating the hardware Interrupts inside the 80x86 systems.

The internal clock of the 8086/8088 is equal to one third of the input frequency signal at pin 19 (**CLK**). The CLK signal is usually supplied via the **82284** clock chip, as shown in figure 2-11. For instance, if the 8088 is to be operated at 5MHz, then the 8284 chip should generate 15MHz. The 8284 is also used to manage READY and RESET signals of the 8086/8088.

The **RESET** pin of the 8086/8088 is used to restart the microprocessor system when the system is crashed. When the RESET line goes high momentarily the 8086/8088 microprocessor turns all its IP, DS, ES and SS registers zeroes and makes the CS points to 0FFFFH. This address should contain the startup routine of the microprocessor system. In IBM PC and compatible clones, this is usually the beginning address of the BIOS ROM. The reset signal can be generated via a simple pushbutton or via the 8284 clock chip, as shown in figure 2-11.

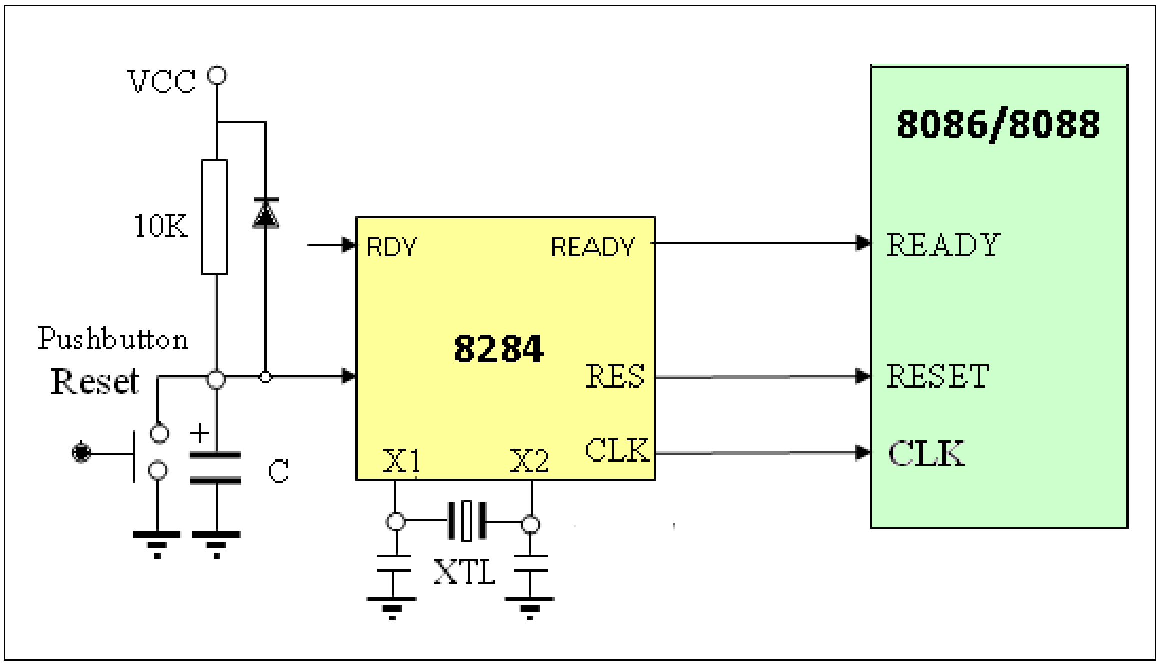


Fig. 2-11. RESET, CLK and READY pins in 8086/8088 microprocessors and their connection to the 8284 clock chip.

2-3. Architecture of the 8087 Math Coprocessor

The ALU of the 8086/8088 could only do integer math. For data that is floating point, the calculation was handled by writing programs to break down the math operation into a series of integer operations. As this was a slow calculation, special purpose arithmetic chips were developed. These circuits were developed specifically to do mathematic functions on floating point data. These chips were referred to as co-processors or floating point processors, **FPU**.

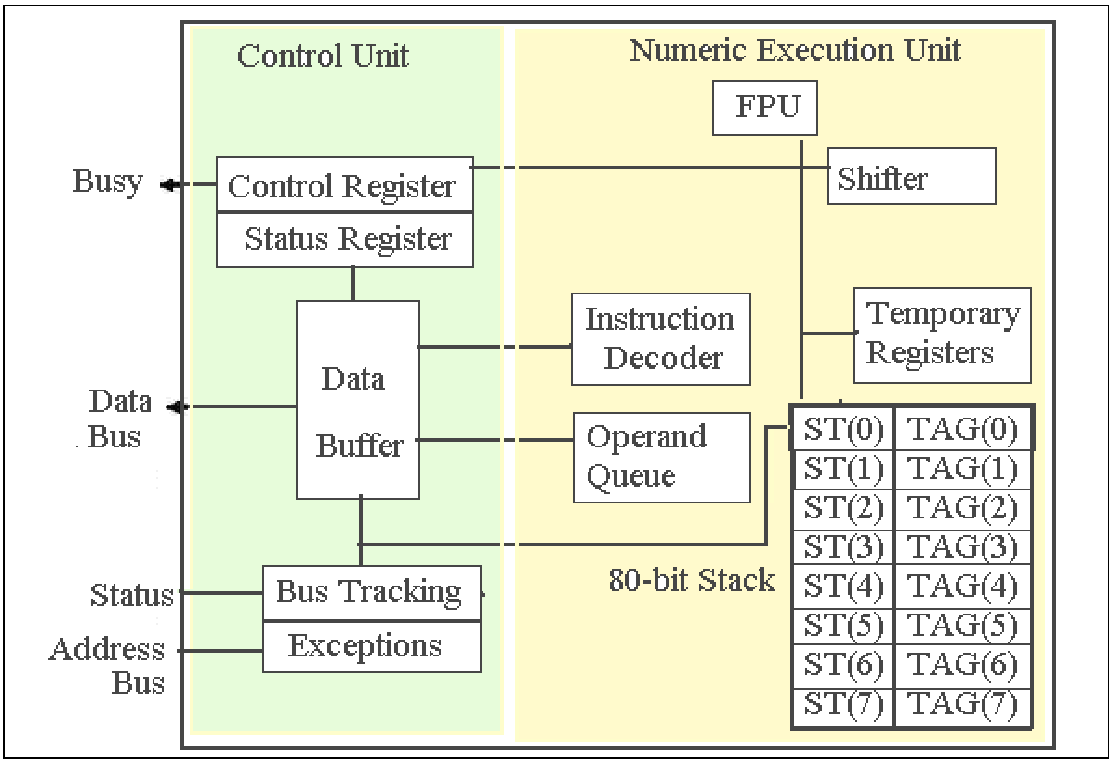


Fig. 2-12. Architecture of the 8087 math coprocessor.

The 8087 math coprocessor was one of the innovations of the x86 family of microprocessors. The 8087 is capable of performing mathematical floating point operations in 80-bit precision. It was designed to monitor the instruction stream and watch for *ESC* sequences (instructions which are proceeded by ESC prefix). Whenever an ESC sequence is detected, the coprocessor knows that the following instruction involves a floating point operation and can execute it more efficiently than the 8086/88. In the meantime, the 8086/88 has to wait the result of the 8087, by issuing a *Wait* instruction until the 8087 is done and its BUSY signal goes low. Other (non ESC) instructions are ignored by the 8087 and normally handled by the 8086/88.

As shown in figure 2-12, the 8087 coprocessor has eight 80-bit data registers, named **ST**(0) through **ST**(7), which can be accessed randomly or as a stack. The connection of 8087 with 8086/88 microprocessor in maximum mode is shown in the following figure.

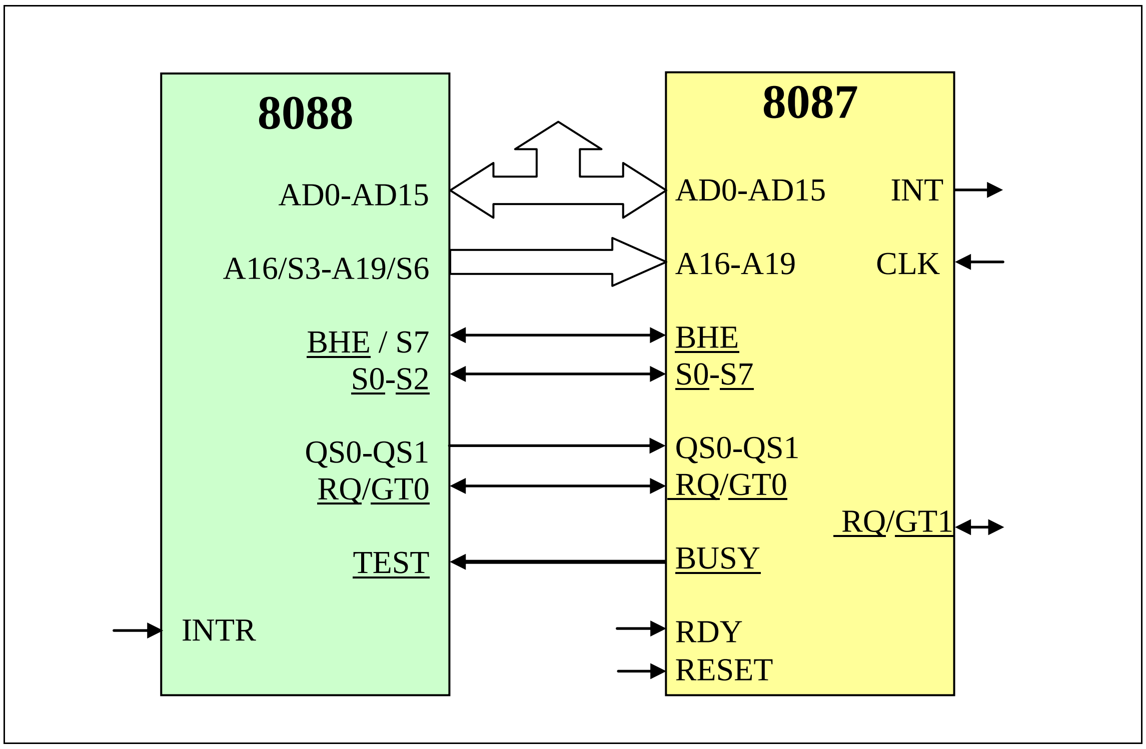


Fig. 2-13. Connection of the 8087 math coprocessor with 8088 microprocessor, in maximum mode.

2-4. Architecture of the 80286 Microprocessor

With the IBM PC great success story, Intel wanted to continue to be the preferred supplier of CPU chips for all the IBM PC models. In order to do that, Intel found it necessary to introduce subsequent improved versions of the 8086, such as the 80186 and 80286.

The 80286 (also known as **iPAX 86**) is a 16-bit microprocessor that was introduced in 1982. It was built in 1 micron NMOS technology with 134,000 transistors in a 68-pin quad-flat pack (**QFP**) package. The 80286 features a 16-bit data bus, and a 24-bit address bus and hence can address up to (2^{24}) 16 MB of memory space. The processor was used in IBM PC/AT, which has been introduced in 1984. The 80286 performance was more than twice that of its predecessors (8086 and 8088) per clock cycle. For instance, the model operated at 12 MHz had a benchmark of 2.66 MIPS (million instruction per second). Figure 2-14 depicts the architecture of the 80286 microprocessor. As shown, the 80286 has 5 additional control registers for segmented memory management and multiple processing:

- **GDTR**: Global Descriptor Table Register
- **LDTR**: Local Descriptor Table Register
- **IDTR**: Interrupt Descriptor Table Register
- **TR**: Task Register
- **MSW**: Machine Status Word

The additional control registers enabled 80268 to work in the protected virtual address mode (**PVAM**). The machine status word (**MSW**) is a register that contains information indicates whether the machine operates in real mode or in PVAM. When bit 0 of the MSW register (termed **PE**) is set to zero, the machine enters the PVAM. In this case and the 80286 can address up to 16 MB of memory. The **GDTR** and **LDTR** registers point to the general and local segment descriptor tables. **IDTR** register points to a table of entry points for interrupt. The **TR** register points to the information needed by the processor to define the current task.

Beside additional registers, there are three additional bits present in the 80286 flags register. The I/O Privilege Level is a two bit value (bits 12 and 13). It specifies one of four different privilege levels necessary to perform I/O operations. These two bits generally contain 00 when operating in real mode on the 80286.

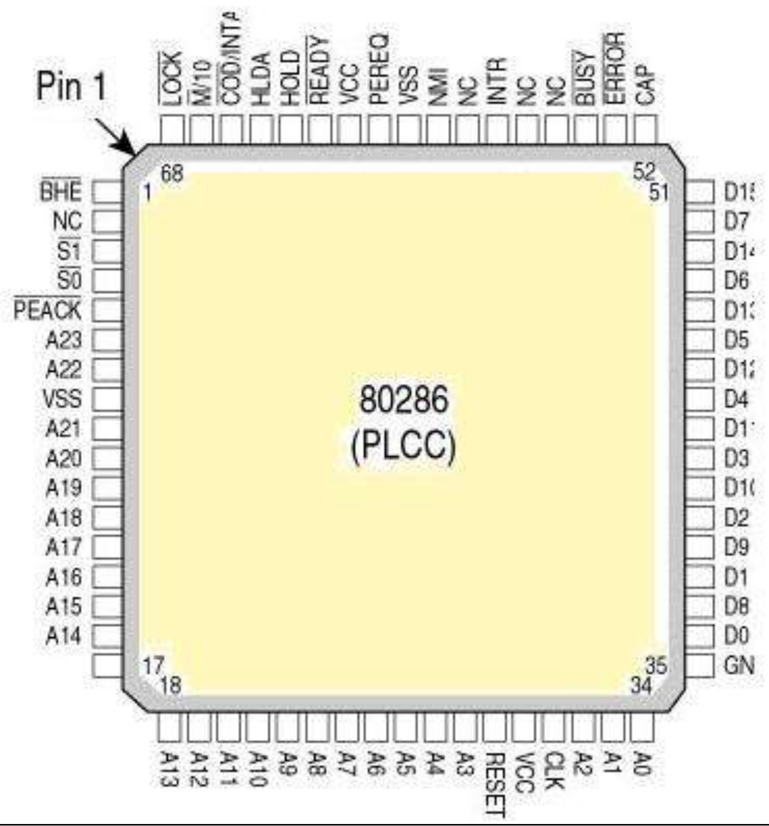


Fig. 2-14(a). Pinout of the 80286 microprocessor.

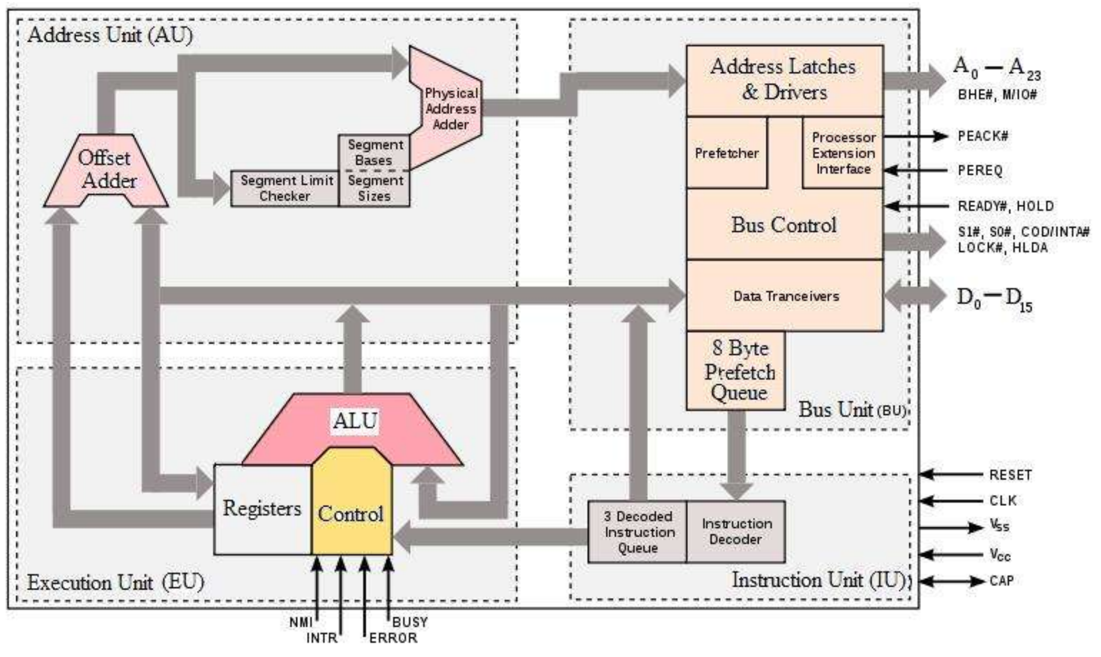


Fig. 2-14(b). Architecture of the 80286 microprocessor.

The **NT** (*nested task*) flag controls the operation of an interrupt return (**IRET**) instruction. The benefit of the protected mode on the 80286 is to access more than one megabyte of RAM. However, as the 80286 is now virtually obsolete, and there are better ways to access more memory on later processors, programmers rarely use the PVAM mode.

MSW (80286)															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-												PE

Fig. 2-14(c). Machine status word register (MSW) in the 80286 microprocessor.

2-5. Architecture of the 80386 Microprocessor

The 80386 processor was a significant step in the line of Intel's 80x86 processors. The 80386 is a 32-bit microprocessor that was introduced for the first time in 1985. It has 275,000 transistors, built in 1μm CMOS technology, in 132 pin package. Figure 2-15(a) and table 2-4 depict the pin assignment of the 80386 chip. The 80386 features a 32-bit data bus, a 32-bit address bus, and hence can address up to (2^{32}) 4 GB of memory. This 32-bit architecture added better memory management and multiprocessing. Before that time, personal computers based on Intel processors were almost exclusively driven by the DOS systems that could run only one program at a time. The 386 allows multiple application programs to run at the same time using the so called "protected mode". In this mode, each application has its own protected memory area. In addition, the 80386 microprocessors have a virtual memory mode, in which the microprocessor can address up to 64TB of external memory. In order to employ all these new facilities, Intel has equipped the 80386 microprocessors with new opcodes in a kludgy fashion similar to the Zilog Z80. In addition to all the registers on the 80286 (and therefore, the 8086), the 80386 added several new registers and extended the definition of the existing registers. There are different versions of the 80386 CPUs, among which one can cite:

- 80386DX - which works with 16-bit and 32-bit external buses.
- 80386SX – which is a low cost version of the 80386. It has 16 bit external data bus and 24-bit external address bus.

Figure 2-15(a) shows the pinout diagram of the 8386SX microprocessor and figure 2-15(b) depicts the internal architecture of the 8386 CPU.

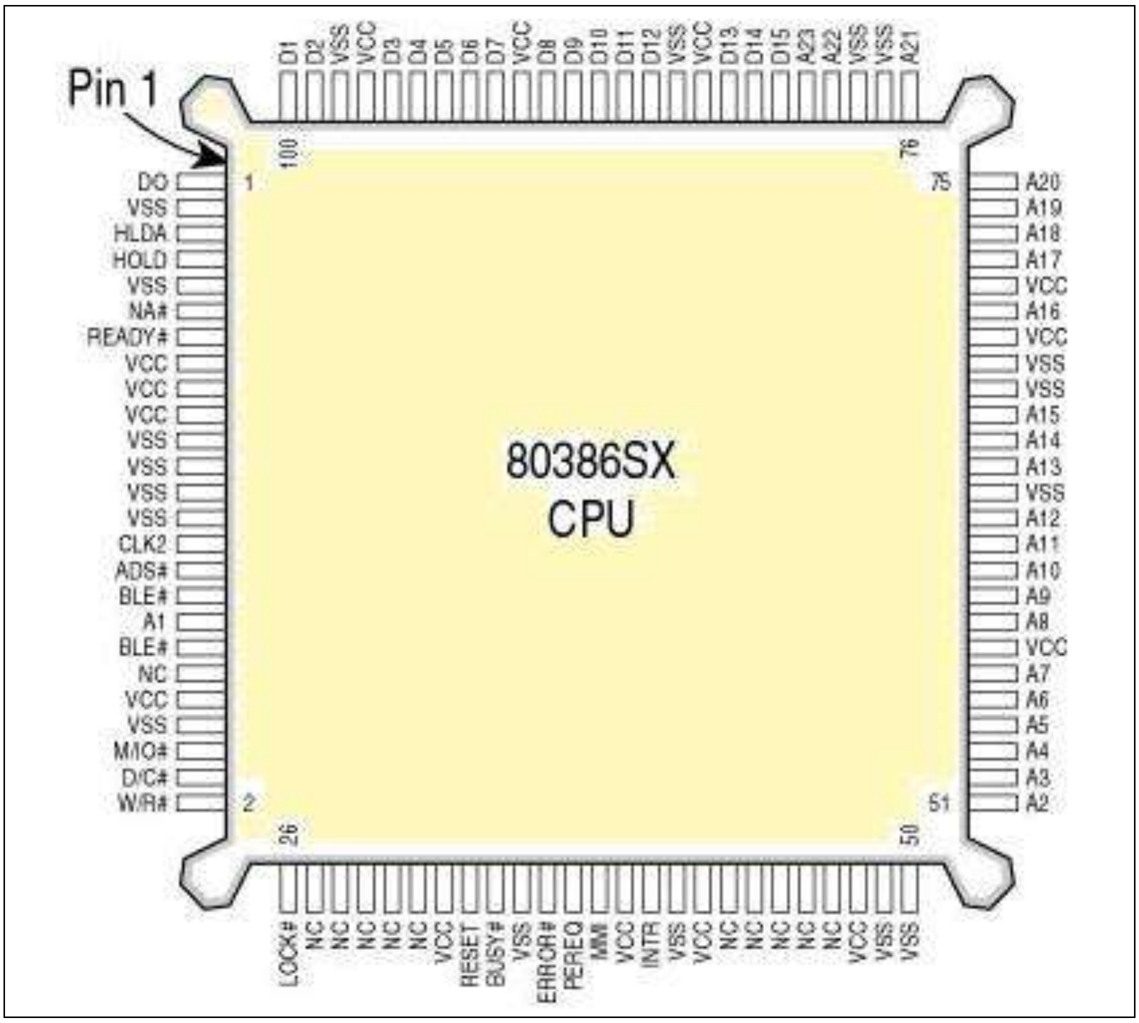


Fig.. 2-15(a). Pin-out diagram of the 80386SX microprocessor.

Table 2-4. Pin assignment of the 80386 microprocessor.

Pin	**Assignment**	**Pin**	**Assignment**
A2-A31	Address Bus	BE0 -	Bus Enable encoded from A0-A1 to
D0-D31	Data Bus	BE3	select 4 (8-bit) memory banks
W/R	Write/Read	ADS	Address Data Strobe. Combined with
M/IO	Memory/IO		W/R to generate MRD, MW, IOR, IOW
D/C	Data / Control	BS16	Bus Size 16. Select 16-bit or 32-bit data
NA	Next Address	BUSY	Input used by coprocessor to Wait
CLK2	2X clock	PEREQ	FPP Request to 80386
INTR	Interrupt Request	NMI	Non-maskable Interrupt
ERROR	Error signal from coprocessor.	READY	Ready signal. Controls the number of wait states to lengthen memory access.
HLDA	Hold Acknowledge	HOLD	Hold requests a DMA action
INTR	Interrupt Request	NMI	Non-maskable Interrupt

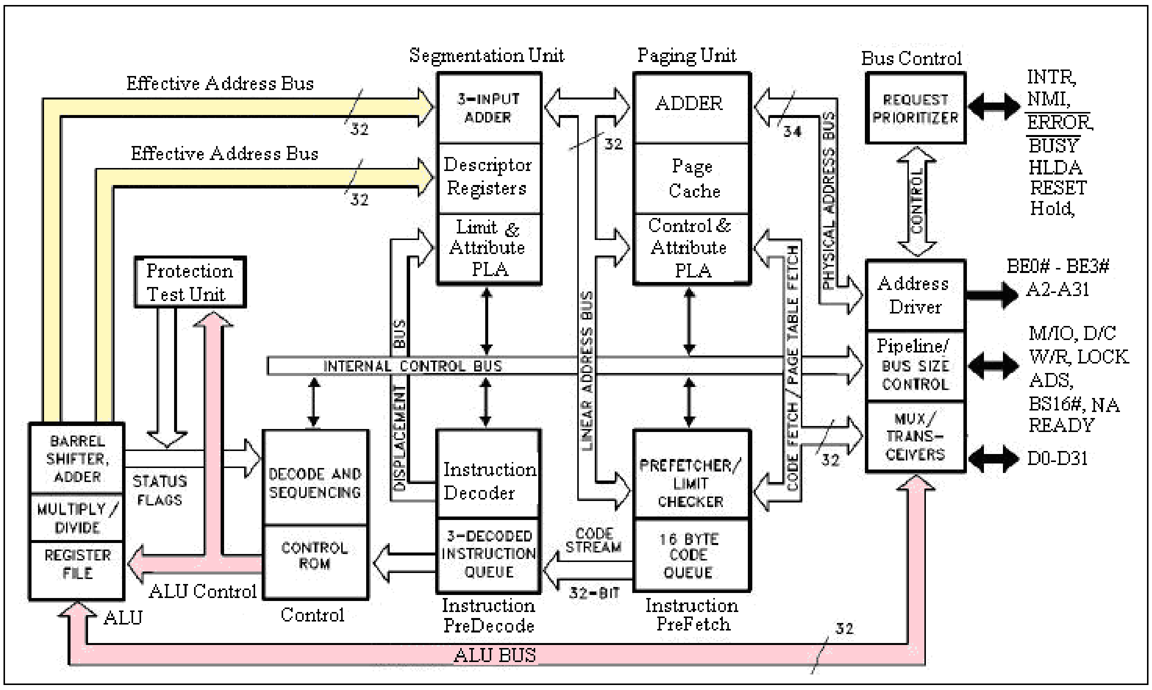


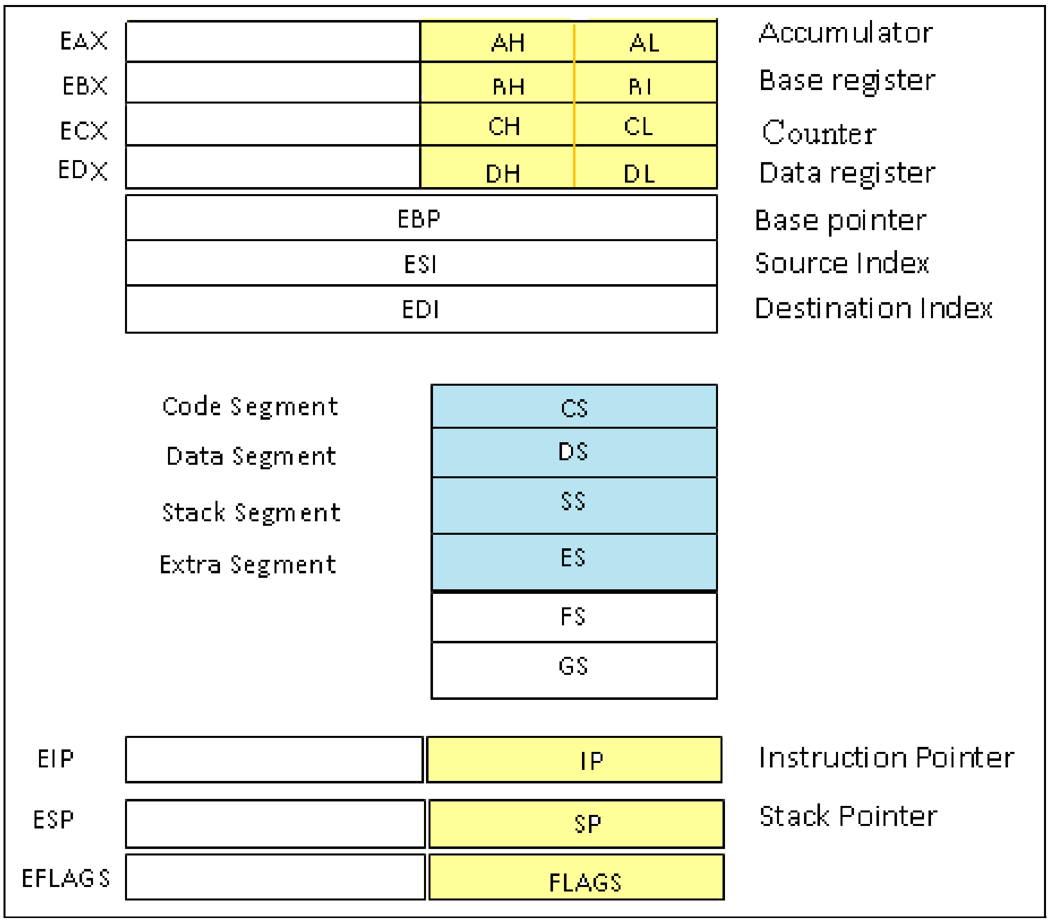
Fig. 2-15(b). Internal architecture of the 80386 microprocessor

As shown in figure 2-15(c), some of the internal registers of 80386 microprocessors are extended to 32 bit as follows:

i) 4 Data Registers (32 bit): **EAX, EBX, ECX, EDX**
ii) 6 Segment Registers (16 bit): **CS. DS, SS, ES, FS** and **GS**
iii) 5 Pointer/Index Registers (32 bit):
  ESP: Stack Pointer
  EBP: Base Pointer
  ESI: Source Index
  EDI: Destination Index
  EIP: Instruction Pointer
iv) 1 Flag register **EFLAGS** (32bit): to indicate result of ALU instructions (like addition, subtraction, comparison, etc.)

The most important change, from the programmer point of view, to the 80386 was the introduction of a 32-bit register set. The 16-bit AX, BX, CX, DX, SI, DI, BP, SP, FLAGS, and IP registers were all extended to 32 bits. The 80386 calls these new 32-bit versions EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP, EFLAGS, and EIP to differentiate them from their 16-bit versions (which are still available on the 80386). Besides the 32-bit registers, the 80386 also provides two new 16-bit segment registers, **FS** and **GS**, which allow the programmer to concurrently access six different segments in memory without reloading a segment register.

Note that all the segment registers on the 80386 are still 16 bits. The 80386 did not extend the segment registers to 32 bits. The 80386 microprocessor extended the flags register to 32 bits (renamed EFLAGS) and defined bits 16 and 17. Bit 16 of the EFLAGS register is the debug resume flag (**RF**) used with the set of 80386 debug registers. Bit 17 is the virtual 8086 mode flag (**VM**), which determines whether the processor is operating in virtual-86 mode (that simulates an 8086) or standard protected mode. Furthermore, the 80386 has additional special registers for control and memory management. So, in addition to the 5 special registers of the 80286 microprocessor: GDTR, LDTR, IDTR, TR, and MSW, the 80386 has added 16 registers, as shown in table 2-5. As shown, the 80386 added four control registers (CR0-CR3). These registers extend the MSW register of the 80286 (the 80386 emulates the 80286 MSW register for compatibility, but the information really appears in the **CR**x registers). These registers control functions such as paged memory management, and protected mode operation.



Fig, 2.15(b). Internal registers in 80386 microprocessors.

The 80386 added eight *debugging* registers (DR0-DR7). A debugging program like Microsoft's **CodeView** can use these registers to set breakpoints when you try to locate errors within a program. While you do not use these registers in an application program, you will notice that such debuggers reduce the time to eradicate bugs in programs. Also, the 386 processor has a set of test registers (TR6-TR7) to the system, which test the proper operation of the processor when the system powers up. Intel put these registers to allow chip testing after fabrication. However, system designers can use these registers to do power-on self test (POST).

EFLAGS (386+)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	NT	IOPL		OF	DF	IF	TF	SF	ZF	-	AF	-	PF		CF
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
-	-	-	-	-	-	-	-	-	-	ID	VIP	VIF	AC	VM	RF

- **CF**: Carry Flag (contains carry out of MSB of result)
- **PF**: Parity Flag (indicates if result has even parity)
- **AF**: Auxiliary carry Flag (contains carry out of bit 3 in AL)
- **ZF**: Zero Flag (indicates if result is zero)
- **SF**: Sign Flag (indicates if result is negative)
- **OF**: Overflow Flag (indicates if overflow occurred in result)
- **IF**: Interrupt Flag (indicates if interrupt is enabled or disabled)
- **DF**: String Direction Flag (controls pointer updates during string operation)
- **TF**: Test or Trap Flag (provides single-step execution capability for debugging)
- **IOPL**: I/O Privilege Level. Used in protected mode to select priority level of I/O devices. IOPL takes 2 bits (12, 13). So, there exist 4 privilege levels (0,1, 2, 3)
- **NT**: Nested Task flag. Indicates if current task is nested
- **VM**: Virtual Mode flag. Selects virtual 80386 mode operation in protected mode.
- **RF**: Resume Flag. Used with the set of 80386 debug registers.
- **AC**: Alignment Check flag (added in 80486)
- **VIF**: Virtual Interrupt Flag. Copy of IF bit in Pentium (80586) and later processors.
- **VIP**: Virtual Interrupt Pending. Used in multithreading (80586).
- **ID**: Identification flag. Indicates whether the microprocessor supports CPUID instruction, which provides information about the microprocessor (80586)

Fig, 2.16(a). Structure of the EFLAGS register in 80386 microprocessors.

Table 2-5. Special registers in 80386 and later microprocessors.

CR0 Control Register 0	**DR0** Debug Register 0
CR1 Control Register 1	**DR1** Debug Register 1
CR2 Control Register 2	**DR2** Debug Register 2
CR3 Control Register 3	**DR3** Debug Register 3
	DR4 Debug Register 4
	DR5 Debug Register 5
	DR6 Debug Register 6
TR6 Test Register 6	**DR7** Debug Register 7
TR7 Test Register 7	

CR0 (386+)																			
31	30	29	...	18	17	16	...	10	9	8	7	6	5	4	3	2	1	0
PG	CD	NW	…	AM	-	WP	…	-	-	-	-	-	NE	ET	TS	EM	MP	PE

- **EM**: Emulation Mode (of 80x87)
- **ET**: Extension Type (presence of 80387)
- **MP**: Math Processor
- PE: Protection Enable
- **PG**: Paging Gate (disabled when PG = 0)
- **TS**: Task switch
- **CD**: Cache Disable (486)
- **NW**: No-Write through (486)
- **AM**: Alignment Mask (486)
- **WP**: Write Protect (484)
- **NE**: Number Exception (486)

Fig, 2.16(b). Structure of the CR0 control register in 80386 microprocessors.

2-6. Architecture of the 80486 Microprocessor

The 80486 is also a 32-bit microprocessor that contains both the 80386 microprocessor and the 80387 coprocessor on a single chip. The first version of 80486 was introduced in 1989 using the 1-micron CMOS technology. It had 1.2 million transistors in a 168-pin package, called pin-grid array (**PGA**). The 80486 has a 32-bit data bus and a 32-bit address bus and hence can address up to 4 GB (2^{32} bytes) of physical memory space. Figure 2-17 depicts the architecture of the 80486 microprocessor system. The 80486 performance is 20 times faster (20X) than 8086/8087 on integer numbers and 40 times faster (40X) than 8086/8087 for floating numbers operations.

Exactly as 80386, the 80486 processor has eight 32-bit general-purpose registers. It is also equipped with a floating point unit (**FPU**) to perform mathematical operations with floating point numbers. The FPU has a **register file** of eight 80-bit floating-point registers. The 80486 processor is also equipped with internal 8kB of **cache memory** to accelerate the processor instructions handling. As we'll see in details in chapter 7, the cache memory is a sort of fast static RAM (SRAM) which is usually integrated on the microprocessor chip. This memory is sometimes called level-one cache (**L1-Cache**). The 80486 did not add any new registers to the 80386 basic register set, but it defined a few bits in some registers left undefined by the 80386. The 80486 adds a third bit to the EFLAGS register at position 18 - the alignment check flag (**AC**). Along with control register zero (CR0) on the 80486, this flag forces a trap (program abort) when the processor accesses non-aligned data (e.g., a word on an odd address). The 80486 extends the control functions of the control registers. In addition to their use for paged memory management, and protected mode operation, the 80486 uses control registers for cache enable/disable operation. It should be also noted that the 80486 microprocessor can execute two instructions per cycle. It also does pipelined floating-point and performs branch prediction as well as clock doubling (like the Zilog Z80).

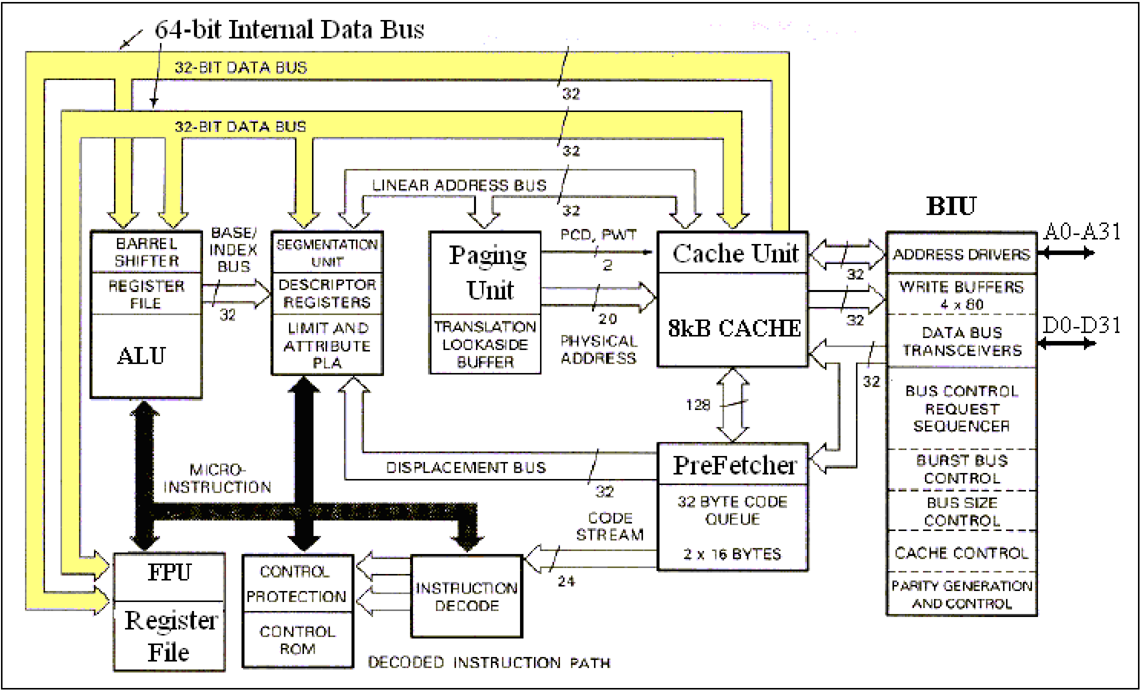


Fig. 2-17. Architecture of the Intel 80486 microprocessor.

2-7. Architecture of Intel's Pentium (80586) Microprocessor

The Intel Pentium (80586) is a 32bit microprocessor and was introduced for the first time in 1993. It is called "Pentium" because it is the fifth in the 80x86 lines. It would have been called the 80586 unless a US court had ruled that you can't trademark a number. The Pentium processor has internal 32bit data bus, a 32bit address bus and can address up to 4GB of memory. However, this processor has 64bit memory interface and an external data bus of 64bit. It has 3.1 million transistors, and built in 0.18μm BICMOS technology. The clock of this CPU was initially rated at 66MHz. Figure 2-18 shows the architecture of the Pentium processor.

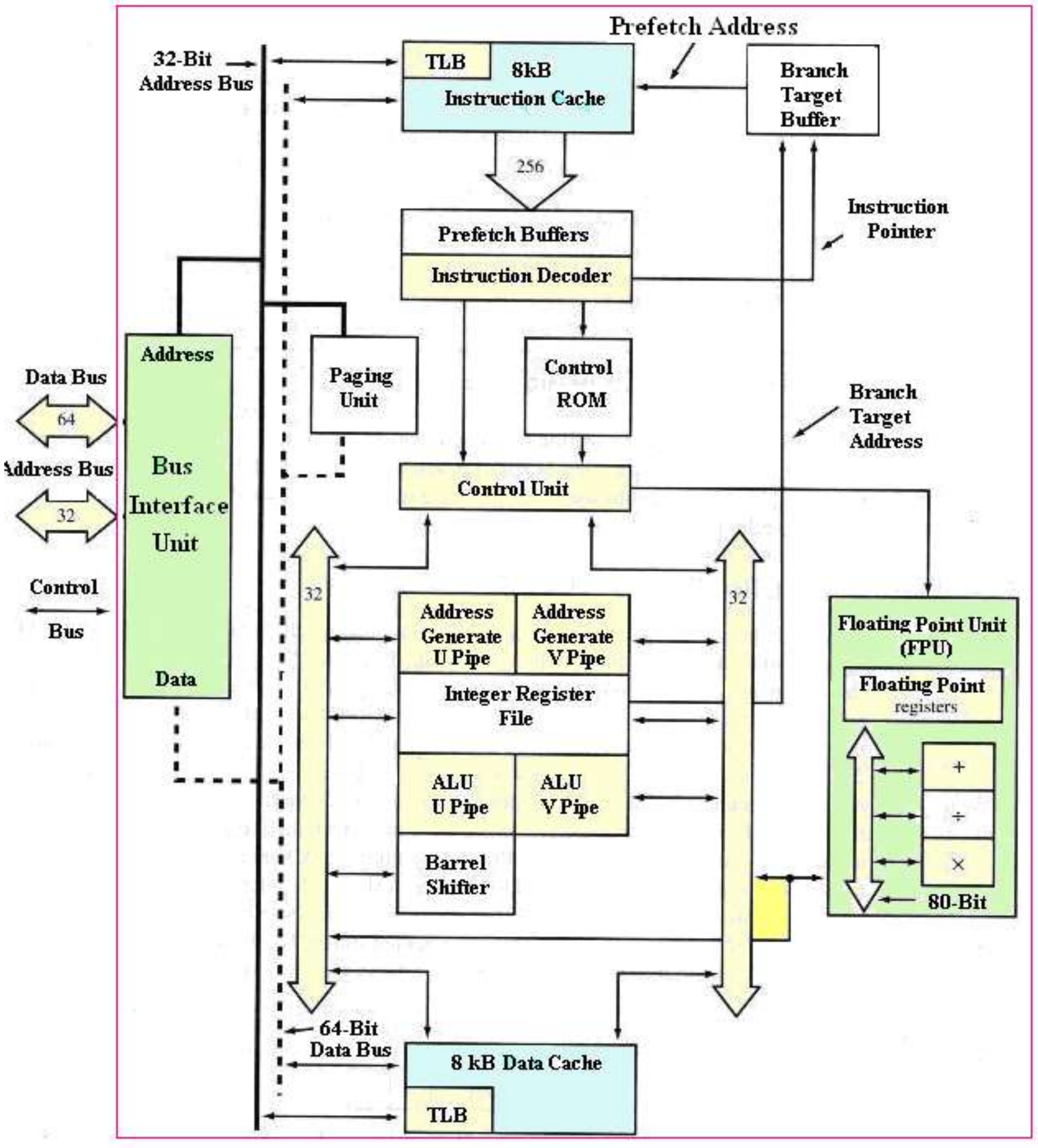


Fig. 2-18. Architecture of the Intel Pentium (80586) microprocessor.

As shown in figure, the Pentium processor is equipped with a floating-point unit (**FPU**) with an 80-bit bus. It has two 32-bit integer pipelines. The Pentium processor has also 16kB of cache memory; 8kB for data (**D-Cache**) and 8kB for instructions (**I-Cache**), to accelerate instructions handling and execution. The branch prediction Unit (**BPU**) predicts which instruction block will be executed after a conditional branch instruction. The instructions in that block are then fetched and their execution can begin even before the branch decision is taken. Whenever the prediction is correct, the instruction execution continues smoothly, saving so much time. Otherwise, the processing in the prediction path is abandoned,

The Pentium surpasses the 80486 processor in speed by using internal 256bit data paths and pipelined processing that lets all operations happen at once. In addition, instruction processing is split into dual arithmetic logic units. In pipelining, the instructions are broken into small tasks, as shown in figure 2-19, and their execution is shared between various stages as follows:

- The Bus Interface Unit (**BIU**) retrieves code and data from RAM.
- The **BIU** sends code along one 64-bit path to the 8kB I-Cache and sends data along another 64-bit path to the 8kB D-Cache. The two caches collect code and data until other components request them.
- The BPU inspects the code in the I-Cache to determine which of the two pipelines, or data paths, can most efficiently carry each instruction.
- The instruction **Pre-fetch Buffer** obtains new instructions in 256-bit bursts and the Instruction Decode Unit prepares the code for execution.
- The **FPU** calculates any non-integer math and puts the result in D-Cache.
- The two integer **ALU**'s simultaneously take two sequential instructions of up to 32 bits each from the Instruction Decode Unit.
- The instructions are **executed** using data placed in the Execution Unit's Registers from the D-Cache.
- The D-Cache receives the results of the calculation. The Cache sends the results to the BIU, which in turn **stores** the results to RAM.

The heat dissipation of first Pentium processors was about 16W and needed to special cooling fans

	Instruction 1					**Instruction 2**				
Task / Time Cycle	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Instruction **F**etch	IF1					IF2				
Instruction **D**ecode		ID1					ID2			
Operand **L**oad			OL1					OL2		
Instruction **E**xecute				IE1					IE2	
Operand **S**tore					OS1					OS2
(a) Non-pipelined processing										
Task / Time Cycle	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Instruction Fetch	IF1	IF2	IF3	IF4		IF5	IF6	IF7	IF8	
Instruction Decode		ID1	ID2	ID3	ID4		ID5	ID6	ID7	ID8
Operand Load			OL1	OL2	OL3	OL4		OL5	OL6	OL7
Instruction Execute				IE1	IE2	IE3	IE4		IE5	IE6
Operand Store					OS1	OS2	OS3	OS4		OS5
(b) Pipelined processing (with 4 pipes)										

Fig. 2-19. Pipelined and Non-pipelined instruction processing.

The performance (or **benchmark**) of a Pentium processor is about 100 million instructions per second (**MIPS**) when it is operated at 66 MHz.

Benchmarks are standard programs or set of programs which can be run on different computers to give a measure of their performance. The count of executed instructions per second, by millions (**MIPS**), was a measure of performance for the first microprocessor generations. However, as microprocessors have been widely produced by several manufacturers and their instruction sets are different in length, the MIPS has no longer been an acceptable benchmark in the microprocessor literature. Alternatively, the SPECint92 and SPECfp92 have been introduced as benchmarks for the evaluation of microprocessor performance. The **SPECint92** is a performance evaluation standard, whose result is derived from a set of integer benchmarks. So, the SPECint92 is a benchmark, which may be used to estimate a machine's single-tasking performance on integer operations. Similarly, the **SPECfp92** is a benchmark, which is used to estimate the microprocessor single-tasking performance on floating point numbers. The performance of the 80486 in terms of the above benchmarks is 64.5 SPECint92, and 56.9 SPECfp92 at 66 MHz.

2-8. Architecture of Intel's Pentium II Microprocessor

Pentium II is the successor of the Pentium (80586) and Pentium Pro (80686) microprocessors. The first produced Pentium II processors were code named **Klamath**. They were manufactured using a 0.35 micron CMOS process with about 7.5 million transistors on a single chip. The initial versions of Pentium II supported clock rates of 233 to 300 MHz at a bus speed of 66 MHz. The Pentium II second-generation (code-named **Deschutes**) were made with the 0.25 micron CMOS technology and supported clock rates of 350 to 800 MHz at a bus speed of 100 MHz. The Pentium II can execute all the instructions of all the earlier x86 processor family. There are four versions targeted at different user markets. The Celeron processor is the simplest and cheapest version. The standard Pentium II is aimed at mainstream home and business users. The Pentium II **Xeon** is intended for higher performance business servers. There is also a mobile version of the Pentium II for use in portable computers.

All versions of the Pentium II are packaged on a special **daughter-board** that plugs into a card-edge processor slot on the motherboard. The daughter-board is enclosed within a rectangular black box called a Single Edge Contact (**SEC**) cartridge.

The low cost Celeron may be sold as a card only without the box. Consumer line Pentium II requires a 242-pin slot called **Slot 1**. The Xeon processor uses a 330-pin slot called **Slot 2**. Intel refers to Slot1 and Slot2 as SEC-242 and SEC-330 in some of their technical documentation. The daughter-board has mounting points for the Pentium II CPU itself plus various support chips and cache memory chips. You can find a recapitulation of the features of all the above mentioned sockets and interfaces, as well as their photos, in Chapter 11 of this book.

Pentium II is a super-scalar CPU. A **superscalar** architecture is a uni-processor that can execute two or more scalar operations in parallel. Super-scalar architectures require multiple functional units, which may or may not be identical to each other. In some superscalar processors the order of instruction execution is determined statically (purely at compile-time), in others it is determined dynamically (partly at run time). All Pentium II processors have Multimedia Extensions (**MMX**) and integrated level-1 and level-2 cache controllers. Additional features include Dynamic Execution and Dual Independent Bus Architecture, with separate 64 bit system and cache busses.

MMX is a set of 57 extra instructions built into some versions of Intel's Pentium microprocessors for supporting SIMD (single instruction / multiple data) operations on multimedia and communications data types. MMX is not an acronym for "**MultiMedia eXtension**", according to Intel, but an Intel brand name. MMX-enhanced processors have been released in 1997. They are fully compatible with previous Intel processors and software but software only benefit if it is written to use the new MMX instructions. They can handle many common multimedia operations, such as digital signal processing, normally handled by a separate sound card or video card.

2-9. Architecture of Intel’s Pentium III Microprocessor

The Pentium III microprocessor was introduced in 1999 as Intel Corporation's successor to the Pentium II. The Pentium III is very similar to the Pentium II in architecture. It has a 500 MHz to 1 GHz clock rate and its external bus can be clocked at 100 or 133 MHz. It can have up to 512kB of secondary cache (**L2-Cache**) and it comes in various packages, including Flip-Chip Pin Grid Array (**FC-PGA**)

The Pentium III has Pentium Pro dynamic execution microarchitecture. **Dynamic Execution** consists in fetching and decoding several instructions and preparing them for eventual non-ordered execution. Dynamic execution is actually a combination of different techniques (e.g., multiple branch prediction and data flow analysis). Intel implemented Dynamic Execution, for the first time, in the Pentium Pro after analyzing the execution of billions of lines of code.

The Pentium III also features a multi-transaction system bus, and MMX, like the Pentium II. It adds 70 new instructions, Dual Independent Bus (**DIB**) architecture, the Intel processor serial number, internet streaming and Single Instruction/Multiple Data (**SIMD**) Extensions. Some Pentium III versions also include an Advanced Transfer Cache and Advanced System Buffering.

The **SIMD** is a technology used in parallel processors, where many processing elements (functional units) perform the same operations on different data. There is often a central controller which broadcasts the instruction stream to all the processing elements. When Intel released a 1.1 GHz version of the Pentium III processor using a 0.18 micron fabrication process on July, 2000, it was the world highest performance microprocessor for PCs.

2-10. Architecture of Intel's Pentium 4 Microprocessor

The Pentium IV microprocessor was introduced in 2001. It comes in various packages including 423-pin FC-PGA and has 1 to 3 GHz clock rate. The Pentium 4 processor features the **NetBurst** micro-architecture, which operates at higher clock speeds and delivers performance levels that are higher than previous generations of IA-32 processors. While based on the Intel NetBurst micro-architecture, the Pentium IV still maintains the tradition of compatibility with IA-32 software. The Intel NetBurst micro-architecture features include hyper-pipelined technology, a rapid execution engine, either 400/533/800/1066 MHz system bus, and an Execution Trace Cache (**ET-Cache**).

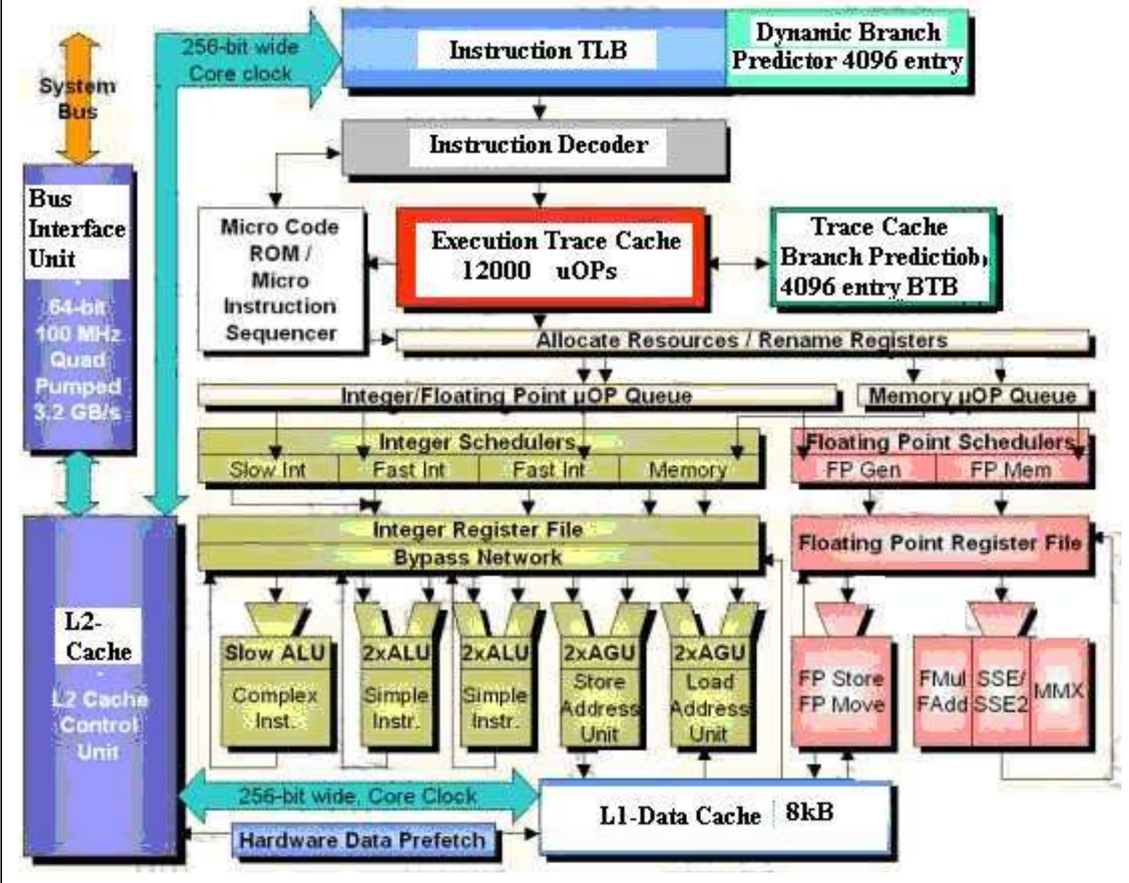


Fig. 2-20. Architecture of the Intel Pentium 4 microprocessor.

The **hyper-pipelined** technology doubles the pipeline depth in the Pentium 4 processor, allowing the processor to reach much higher core frequencies. The rapid execution engine allows the two integer ALU's in the processor to run at twice the core frequency, which allows many integer instructions to execute in 1/2 clock tick.

The 400 MHz system bus is a quad-pumped bus running off a 100 MHz system clock making 3.2 GB/sec data transfer rates possible.

The ET-Cache is a level-1 cache (**L1-Cash**) that stores approximately 12 kB decoded micro-operations (**µOP**s), which removes the decoder from the main execution path, thereby increasing performance.

Improved features within the Intel NetBurst micro-architecture include advanced dynamic execution, advanced transfer cache (**AT-Cache**), and enhanced floating point and multimedia unit, and Streaming SIMD Extensions (**SSE2**).

The advanced dynamic execution improves speculative execution and branch prediction internal to the processor. The AT-Cache is a 256 kB, on-die level-2 cache (**L2-Cash**) with increased bandwidth over previous micro-architectures. The floating point and multimedia units in Pentium 4 have been improved by making the registers 128 bits wide and adding a separate register for data movement.

The **SSE2** enables break-through levels of performance in multimedia applications including 3-D graphics, video decoding/encoding, and speech recognition. The new packed double-precision floating-point instructions enhance performance for applications that require precision, including scientific and engineering applications as well as advanced 3-D geometry techniques. Finally, the SSE2 adds 144 new instructions for double-precision floating point, SIMD integer, and memory management. Figure 2-20 depicts the architecture of the Pentium 4. The Branch Target Buffer (**BTB**) table contains all the addresses to where a branch will or could be made.

The Micro-Operations **(µOP**) is the name that Intel gives to its new instructions, which can be directly understood by the execution units of the microprocessor. These RISC-like instructions represent very simple instructions that can be quickly carried out by the processor. Unlike x86-instructions, those µOPs are of a defined size and can thus easily be fed into the execution pipeline. The decoder translates an x86-instruction into one or many more µOPs, unless the x86-instruction is so complex. In this case, the Micro Instruction Sequencer (**MIS**) has to produce a sometimes rather long sequence of µOPs, using the Micro Code ROM (**MCR**). In average, most x86-instructions get decoded to about two µOPs.

The Pentium 4 processor uses a scalable system bus protocol referred to as the “system bus”. The system bus uses Source-Synchronous Transfer (**SST**) of address and data to improve performance. Whereas the P6 processor family transfers data once per bus clock, the Pentium 4 processor transfers data 4- times per bus clock (4X data transfer rate).

Along with the **4X-data bus**, the address bus can deliver addresses two times per bus clock and is referred to as a ‘double-clocked’ or **2X-address bus**. Working together, the 4X-data bus and 2X-address bus provide a data bus bandwidth of up to 3.2 G Byte/sec (3200 MB/sec).

Most system bus signals of the Intel Pentium 4 processor in the 423-pin package system bus signals use the so-called Assisted Gunning Transceiver Logic (**AGTL**) signaling technology. This signaling technology provides improved noise margins and reduced ringing through low voltage swings and controlled edge rates.

Unlike the P6 processor family, the termination voltage level for the Pentium 4 processor AGTL signals is V_{CC}, of the processor core. The AGTL inputs require a reference voltage (called **GTLREF**), which is used by the receivers to determine if a signal is a logic 0 or a logic 1. GTLREF must be generated on the system board.

Unlike previous processors, the Pentium 4 uses a differential clocking implementation. The system bus clock (**BCLK**) directly controls the system bus interface speed as well as the core frequency of the processor. As in previous processors, the Pentium 4 processor core frequency is a multiple of the BCLK frequency. The Pentium 4 processor bus ratio multiplier is set at its default ratio at manufacturing. No jumpers or user intervention is necessary; the processor will automatically run at the speed indicated on the package.

In 2005, Intel has released an extreme edition of Pentium 4 microprocessor running at 3.73 GHz, which was likely the last of its kind before the Core design. As far as we know today, there is no further frequency speed jump on this family of chips. This processor was based on the so-called **hyper-threading** (**HT**) technology. The Pentium D processor belongs to the dual core Pentium microprocessors. It operates with 800MHz front bus clock and sits in the LGA 775 pin socket, with a thin heat-sink on it.

2-11. Intel's Core and Core2 Microprocessors

The Intel **Core** microarchitecture is a multi-core processor architecture issued by Intel in Jan 2006. It is may be considered as the latest iteration of the Intel 32-bit P6 architecture, which traces its history back to Pentium Pro. The extreme power consumption of NetBurst-based processors and the inability to increase clock speed was the primary reason, which made Intel abandon the NetBurst architecture.

The Intel **Core2** is the eighth-generation of Intel x86 microprocessors. The first wave of Core2 processors was released on July 27, 2006. The Intel **Core2** is a multi-core 64-bit microprocessor, which allows for true parallel computing capabilities on the desktop PC's. It has two processor cores in one package running at the same frequency. Each core has its own 2MB L2-Cache (total 4MB), as shown in Fig. 2-21(a).

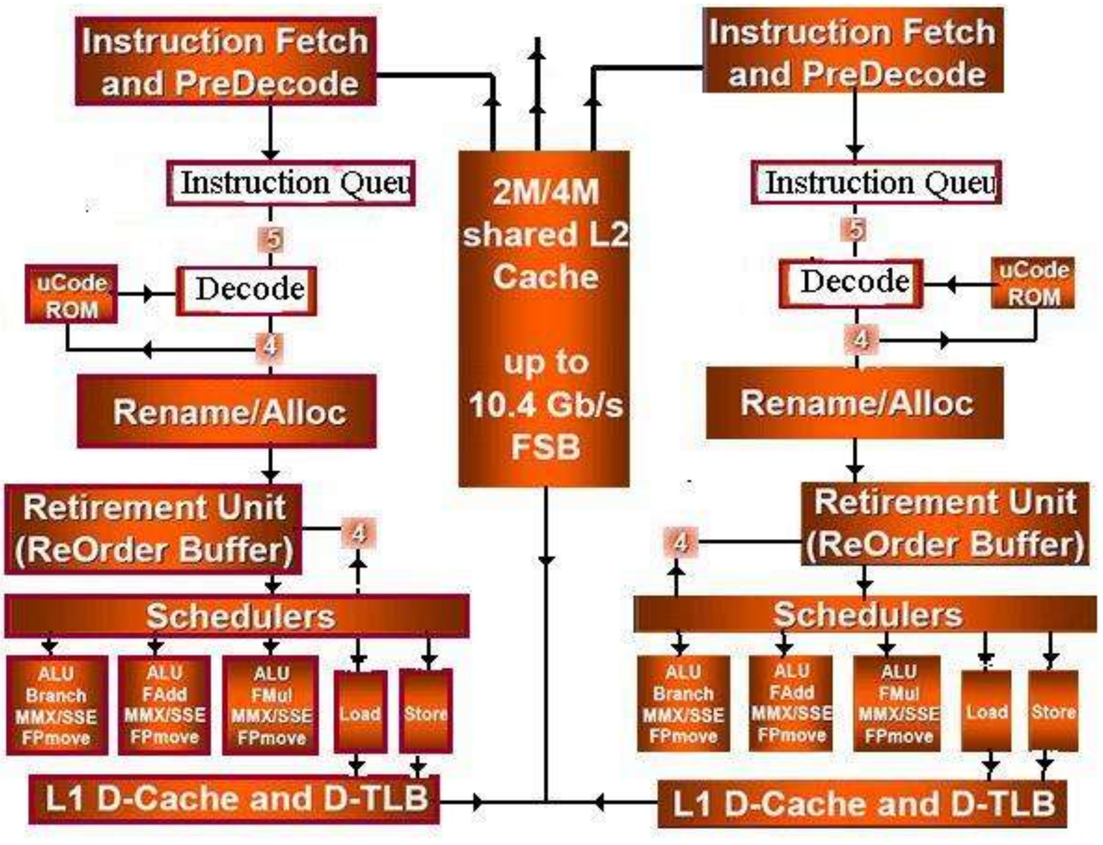


Fig. 2-21(a). Architecture of the Intel dual Core (Core2) microprocessor

The Core architecture has 14-stages pipeline, but it can decode, fetch and issue up to 4 instructions per clock cycle. Like recent PC processors, Core2 translates x86 instructions (using a μCode ROM) into RISC-like short instructions (μOps). One new technology included in the Core design is **Micro-Ops Fusion**, which combines two x86 instructions into a single micro-operation. For example, a common code sequence like a compare followed by a conditional jump would become a single μOp. Other new features include 1 cycle throughput of all 128-bit SSE instructions and a new power saving design.

The so–called Intel's Virtualization-Technology (**VT**) permits one hardware platform to function as multiple virtual platforms. For high-end CPU's, the front side bus (**FSB**) runs at 1333 MHz. However this is scaled down to 1066 MHz for lower end 1.60 and 1.86 GHz variants. The Core2 Duo has a 1066 MHz front bus clock and dissipates only 65W at frequencies up to 2.93GHz.

Fig. 2-21(b). Photograph of the Intel dual Core2 microprocessor

2-12. Core i7 Processors

The Core i7 processors are based on the Nehalem microarchitecture, which is successor to the Core microarchitecture. As shown in figure 2-22, this architecture offers 6 dispatch ports: one Load, two Store and three universal ports for 4 decoders, one for complex instructions (CD) and three for simple instructions (SC). The desktop Core i7 was released on 2009. Server and mobile Core i7 processors followed in 2010 and 2011. The initial Core i7 processors used the 45 nm technology and had 731 million transistors. The processor has 64 kB L1-Cache (32kB code, 32kB data), 256 kB L2-Cache per core and 2-3 MB L3-Cache per core, shared by all cores.

The Core i7 processors also have an integrated memory controller supporting DDR3 SDRAM and 3 memory channels as well as an integrated graphics processor (**IGP**) in the same CPU package. It has also a new point-to-point processor interconnect, the Intel **QuickPath** Interconnect (**QPI**), replacing the legacy front side bus.

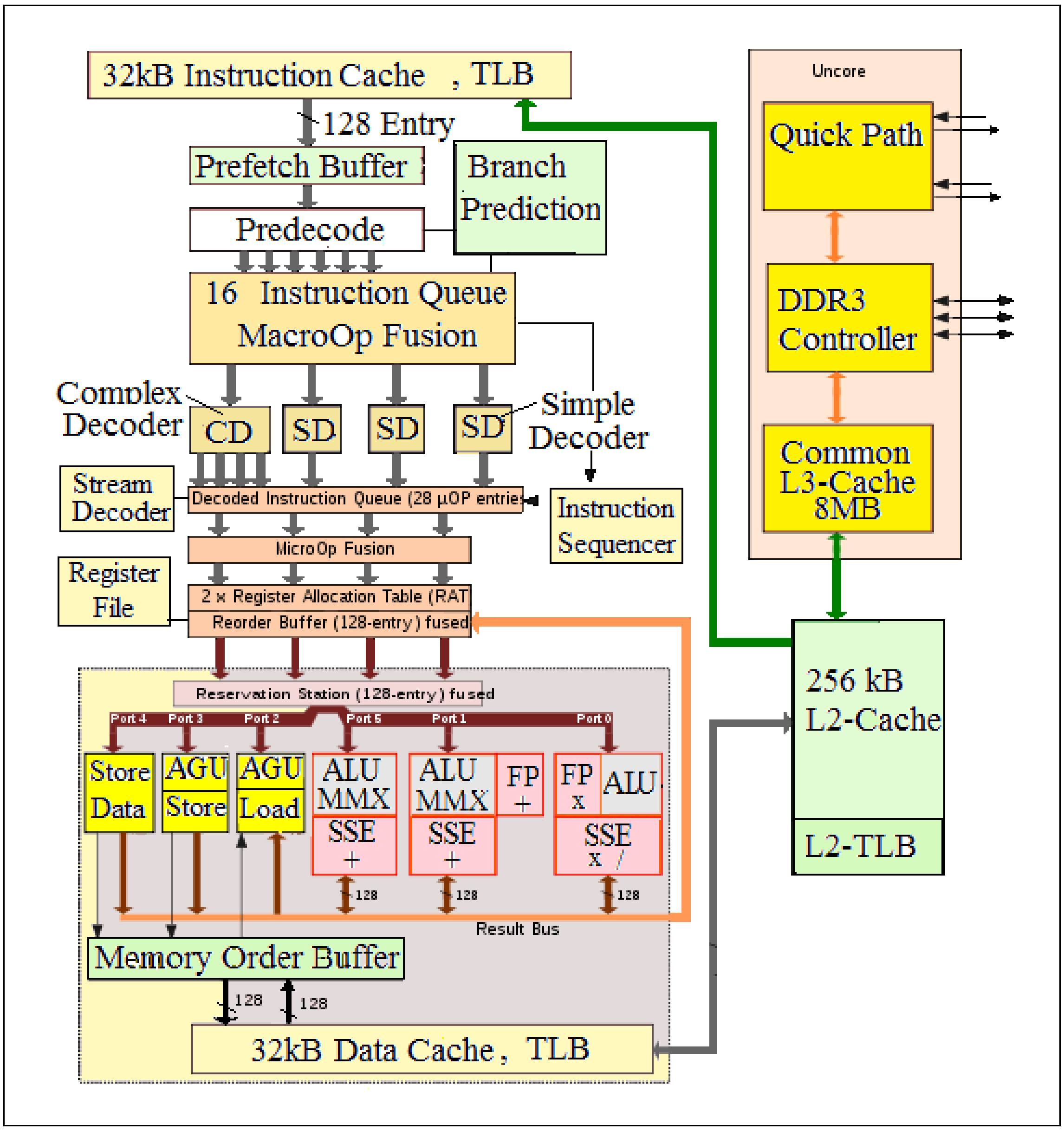


Fig. 2-22. Nehalem microarchitecture (Core i7)

2-13. Architecture of 64-bit Microprocessors

The architecture of Intel x86 series, up to Pentium 4, has been traditionally called **IA-32**. However, in 2003, Advanced Micro Devices (AMD) has introduced the so-called **AMD64**, which is a 64-bit superset of the x86 instruction set architecture. The AMD64 architecture is a simple yet powerful 64-bit, backward-compatible extension of the industry-standard (legacy) x86 architecture. The need for 64-bit architecture was driven by giant applications that address large amounts of memory, such as high-performance servers, database management systems, and CAD tools.

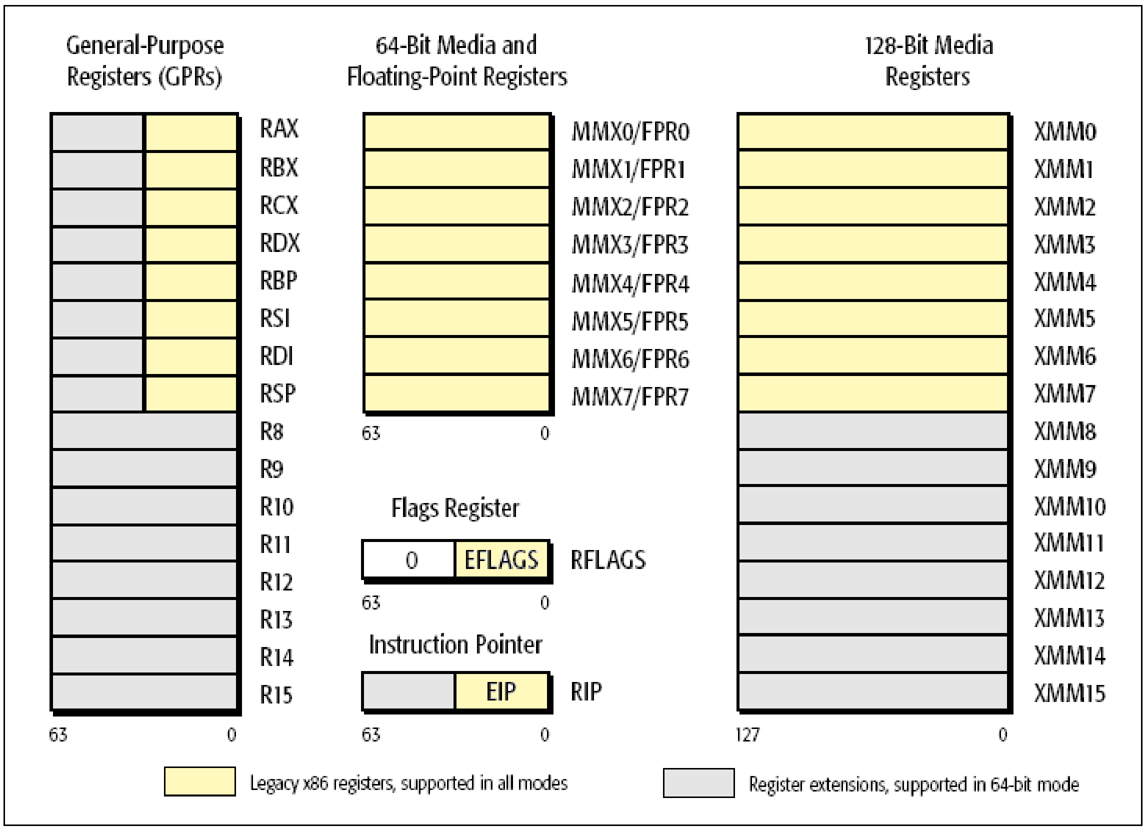


Fig. 2-23. Application registers in x86-64 (64 bit) microprocessors.

The AMD64 architecture has been cloned by Intel under the name **Intel 64**. This leads to the common use of the names **x86-64** to collectively refer to the two nearly identical implementations. Note that x86-64 is not the same as **IA-64**, which is the architecture of Intel's Itanium processors. Intel 64 is Intel's implementation of AMD64 (x86-64). It is used in newer versions of Pentium 4, Pentium D, Pentium Extreme Edition, Celeron D, Xeon and in all versions of the Core2 and later processors.

The x86-64 (x86 in **64-bit**) processors extended the 32-bit registers in a similar way that 32-bit protected mode did before it. The register names become: RAX, RBX, RCX, RDX, RSI, RDI, RBP, RSP, RFLAGS, RIP. However, x86-64 processors also added 8 additional 64-bit general registers (R8, R9, R15), as shown in figure 2-23.

2-14. AMD K10 Architecture

The **AMD K10** is AMD's latest microprocessor architecture. Actually, AMD has used the K-nomenclatures (which stand for Kryptonite) up to *K8* or Athlon 64 processor family. The third-generation **Opteron** was launched on September 2007 and the **Phenom** processors for desktop followed as the successors to the AMD K8 series (Athlon 64, Sempron 64). The following figure describes the AMD K10 architecture.

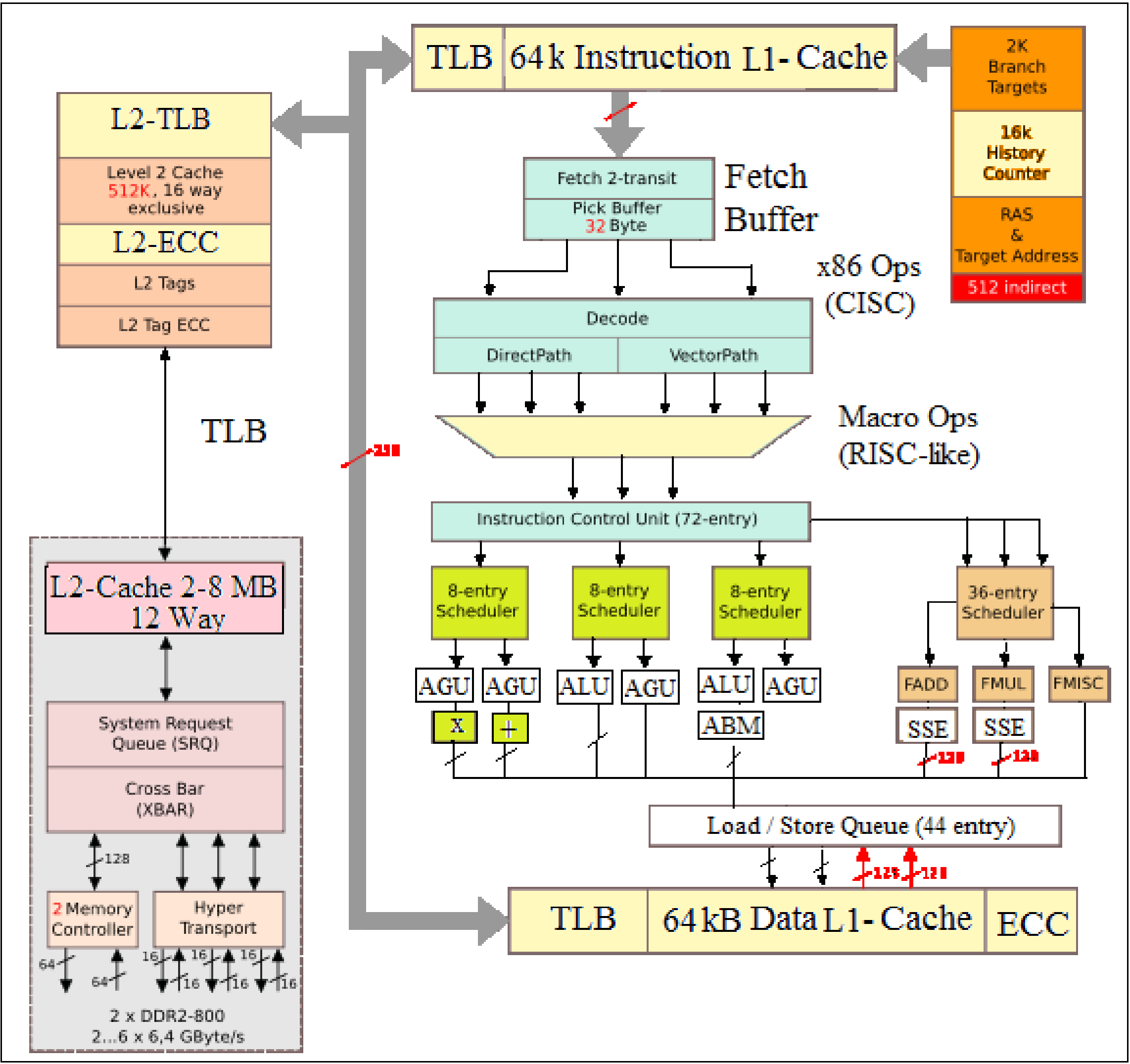


Fig. 2-24. Architecture of the AMD K10 microprocessors.

As shown in figure, the K10 architecture features 128-bit wide SSE units, Wider L1 data cache interface allowing for two 128-bit loads per cycle (as opposed to two 64-bit loads per cycle with K8), Lower integer divide latency, 512-entry indirect branch predictor and a larger return stack (size doubled from K8) and branch target buffer.

2-15. Summary of Intel & AMD Architectures

In this section, we summarize the historical developmental of Intel architectures starting from the Intel 8086 processor to the latest Intel Core processors. Note that the object code created for processors released as early as 1978 still executes on the latest x86-64 architecture.

2-15.1. Intel 16-bit Processors (1978)

The Intel 16-bit architecture (IA-16) family started with the 16-bit processors, the 8086 and 8088. The 8086 has 16-bit registers and a 16-bit external data bus, with 20-bit addressing giving a 1-MB address space. The 8088 is similar to the 8086 except it has an 8-bit external data bus. The 8086/8088 introduced segmentation to the Intel microprocessor architecture. With segmentation, a 16-bit segment register contains a pointer to a memory segment of up to 64 KB. The 20-bit addresses that can be formed using a segment register (CS or DS or SS or ES) and an additional 16-bit pointer provide a total address range of 1 MB.

2-15.2. Intel 286 Processor (1982)

The Intel 286 processor introduced protected mode operation into the Intel architecture. Protected mode uses the segment register content as selectors or pointers into descriptor tables. Descriptors provide 24-bit base addresses with a physical memory size of up to 16 MB, support for virtual memory management on a segment swapping basis, and a number of protection mechanisms. These mechanisms include:

- Segment limit checking
- Four privilege levels

2-15.3. Intel 80386 Processor (1985)

The Intel386 processor was the first 32-bit processor in the **IA-32** architecture family. It introduced 32-bit registers. The lower half of each 32-bit Intel 386 register retains the properties of the old 16-bit registers of earlier generations, permitting backward compatibility. The processor also provides a virtual-86 mode that allows executing programs created for 8086/88 processors. In addition, the 80386 processor has support for:

- A 32-bit address bus that supports up to 4-GB of physical memory
- A segmented-memory model and a flat memory model
- Paging, with a fixed 4-KB page size, providing a method for virtual memory management
- Support for parallel stages

2-15.4. Intel 80486 Processor (1989)

The Intel 80486 processor added more parallel execution capability by expanding the 80386 processor instruction-decode and execution units into five **pipelined** stages. Each stage operates in parallel on up to five instructions in all stages of execution. In addition, the processor added:

- 8 kB on-chip first-level cache (L1-Cache) that increased the ratio of instructions that could execute at the scalar rate of one per clock,
- Integrated 80x87 floating point unit (FPU),
- Power saving and system management capabilities

2-15.5. Intel Pentium Processor (1993)

The introduction of the Intel Pentium processor added a second execution pipeline to achieve superscalar performance (two pipelines, known as U and V, together can execute two instructions per clock). The on-chip L1-Cache has doubled, with 8 KB for code and another 8 KB for data. The data cache (**D-Cache**) uses the MESI protocol to support more efficient write-back cache in addition to the write through cache of 80486. Branch prediction with on-chip branch table was added to increase performance in looping constructs. In addition, the Pentium processor added the following features:

- Extensions to make the virtual-8086 mode more efficient and allow for 4-MB as well as 4-kB pages
- Internal data paths of 128 and 256 bits add speed to data transfers
- Burst external data bus was increased to 64 bits
- An APIC to support systems with multiple processors
- A dual processor mode to support glue-less two processor systems

A subsequent stepping of the Pentium family introduced Intel **MMX** technology. Intel MMX technology uses the single instruction, multiple-data (**SIMD**) execution model to perform parallel computations on packed integer data contained in 64-bit registers.

2-15.6. Intel P6 Family of Processors (1995-1999)

The P6 family of processors was based on a superscalar microarchitecture that set new performance standards. One of the goals in the design of the P6 family microarchitecture was to exceed the performance of the Pentium processor significantly while using the same 0.6 micron, 4-layer, metal BiCMOS process. Members of this family include the following processors:

- The Intel Pentium-Pro processor is 3-way superscalar. Using parallel processing, the processor is able to decode, dispatch, and complete execution of (retire) 3 instructions per clock cycle.
- Pentium Pro introduced the dynamic execution (micro-data flow analysis, out-of-order execution, superior branch prediction, and speculative execution) in a superscalar implementation. The processor was further enhanced by its caches. It has the same two on-chip 8-KB L1-Caches as the Pentium processor and an additional 256KB L2-Cache in the same package as the processor.
- Pentium II processor added the MMX technology to the Intel processors along with other enhancements. The processor core is packaged in the single edge contact cartridge (SECC). The L1- data and instruction caches were enlarged to 16kB, and L2-cache of 512KB and 1MB are supported. Low-power states such as Sleep, Deep-Sleep and Auto-HALT are supported to conserve power when PC is idle.
- Pentium II Xeon processor combined the characteristics of previous generations of Intel processors. This includes: scalability and a 2 MB L2-Cache running on a clock speed backside bus.
- The Intel Celeron processor family focused on the value PC market segment. It offers an integrated 128 KB of L2-Cache and a plastic pin grid array (**PPGA**) form factor to lower the PC system cost.
- The Intel Pentium III processor introduced the Streaming SIMD Extensions (**SSE**) to the IA-32 architecture. SSE extensions expand the SIMD model introduced with the Intel MMX technology by providing a new set of 128- bit registers and the ability to perform SIMD operations on packed single-precision floating-point values.
- The Pentium III Xeon processor extended the performance levels of the IA-32 processors with the enhancement of speed, and Advanced Transfer Cache (**ATC**).

2-15.7. Intel Pentium 4 Processor Family (2000-2006)

The Intel Pentium 4 processor family is based on Intel **NetBurst** micro-architecture. The Pentium 4 processor introduced Streaming SIMD Extensions 2 (**SSE2**), Extensions 3 (**SSE3**) and Hyper-Threading Technology (**HTT**). The Intel Virtualization Technology (**IVT**) was also introduced in the late Pentium 4 processors. Note that Pentium processors are based on the IA-32 (32-bit) architecture. However, some late versions of Pentium 4, like Pentium 4E and Pentium D support Intel 64bit technology (**EM64T**).

2-15.8. Intel Xeon Processor (2001- 2007)

Intel Xeon processors (except for dual-core and, Xeon 5100 series) are based on the Intel NetBurst microarchitecture, which is a variant of IA-32. The Xeon family is designed for multi-processor servers and workstations. The 64-bit Intel Xeon processor running at 3.6 GHz has introduced the **Intel 64 architecture** (IA-64). The Intel Xeon processor 70xx series include Intel Virtualization Technology. The Intel Xeon processor 5100 series introduced the **Intel Core microarchitecture**, which is based on Intel 64 architecture. This processor includes Intel Virtualization Technology and dual-core technology. The Intel Xeon processor 3000 series are also based on Intel Core microarchitecture. The Intel Xeon processor 5300 series introduces four processor cores in a single package. They are also based on Intel Core microarchitecture.

2-15.9. Intel Pentium M Processor (2003-now)

The Intel Pentium M processor family is a low power mobile processor family with **microarchitecture** enhancements over previous generations of IA-32 mobile processors. This family is designed for extending battery life and seamless integration with platform innovations that enable new usage models (such as integrated wireless networking). Its enhanced micro-architecture includes:

- Support for Intel Architecture with Dynamic Execution
- A high performance, low-power technology with copper interconnect
- On-die, 32-KB instruction cache and 32-KB write-back data cache
- On-die, L2-Cache (up to 2 MB) with Advanced Transfer Cache
- Advanced Branch Prediction and Data Prefetch Logic
- Support for MMX technology, SIMD and SSE2 instructions.
- A 400/533 MHz, Source-Synchronous Processor System Bus
- Advanced power management using Intel SpeedStep technology

2-15.10. Intel Pentium Processor Extreme Edition (2005-2007)

The Intel Pentium processor Extreme Edition introduced dual-core technology. This technology provides advanced hardware multi-threading support. The processor is based on Intel NetBurst microarchitecture and supports SSE, SSE2, SSE3, Hyper-Threading, and Intel 64 architecture.

2-15.11. Intel Core Processors (2006-2007)

The Intel Core processor offers power-efficient and multi-core performance with a low-power design that extends battery life. The Intel Core processors offer micro-architectural enhancements over Pentium M processors. Its enhanced microarchitecture includes:

- Intel Smart Cache which allows for efficient data sharing between two
- processor cores,
- Improved decoding and SIMD execution
- Intel Dynamic Power Coordination and Enhanced Intel Deeper Sleep to reduce power consumption
- Intel Advanced Thermal Manager which features digital thermal sensor interfaces
- Support for power-optimized 667 MHz bus

The dual-core Intel Xeon processor LV is based on the same micro-architecture as Intel Core Duo processor, and supports IA-32 architecture.

2-15.12. Intel Xeon 5x00 & Multi-Core Processors (2006-now)

The Intel Xeon processor 3000, 3200, 5100, 5300, and 7300 series, Pentium Dual-Core, Core Duo, Core Quad and Core Extreme processors support Intel 64 architecture; they are based on the Intel Core micro-architecture built on 65 nm process technology. The Intel Core micro-architecture includes the following innovative features:

- Intel Wide Dynamic Execution to increase performance
- Intel Intelligent Power Capability to reduce power consumption
- Intel Advanced Smart Cache which allows for efficient data sharing between two processor cores
- Intel Smart Memory Access to increase data bandwidth
- Intel Advanced Digital Media Boost which improves application performance using multiple generations of SSE technology.

The Intel Xeon processor 5300 series, Core2 Extreme processor QX6800 series, and Core2 Quad processors support Intel quad-core technology.

2-15.13. Intel Xeon 5200, 5400 and Core2 Processors (2007-now)

The Xeon processor 5200, and 5400 series, Core2 Duo processor E8000 series and Core2 Quad processor Q9000 Series, support Intel 64 architecture; they are based on the Enhanced Intel Core micro-architecture. The Enhanced Core microarchitecture provides the following features:

- A radix-16 divider, faster OS primitives further increases the performance of Intel Wide Dynamic Execution.
- Improves Intel Advanced Smart Cache with larger L2-Cache.
- A 128-bit shuffler engine improves the performance of Advanced Digital Media Boost and SSE4

2-15.14. Intel Atom Processor Family (2008 - Current)

The Intel Atom processors are based on a new microarchitecture, called **Intel Atom microarchitecture**, which is optimized for ultra low power devices. Intel Atom processors are built on 45-nm technology. For instance, the Atom Processor N270 (code named Mobile Diamond) was the first generation of low-power IA-32 microarchitecture designed for **Netbook** Platforms. The processor supports low power states at the thread level and the package level. Package low power states include Normal, Stop Grant, Stop Grant Snoop, Sleep and Deep Sleep.

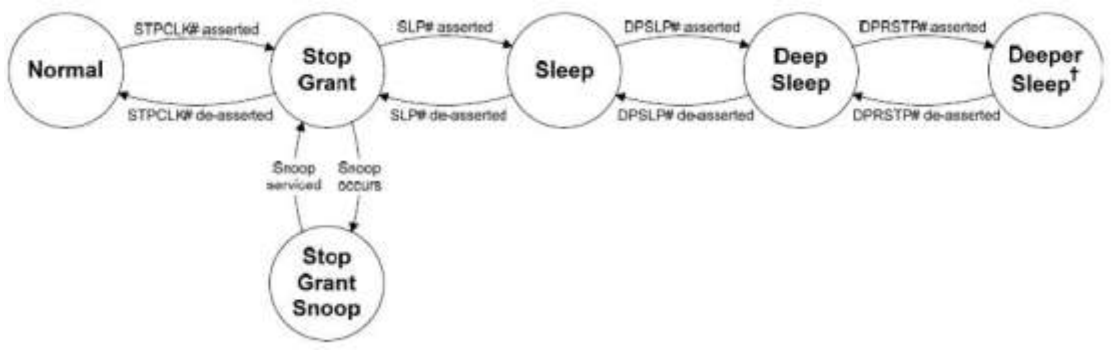


Fig. 2-25. Low power states of Atom processors

The Atom microarchitecture has also the following features:

- Deep Power Down Technology with Dynamic Cache Sizing
- Hyper-Threading and Enhanced SpeedStep Technology
- Support for new instructions including Supplemental Streaming SIMD Extensions 3 (SSSE3) and Intel Virtualization Technology.
- Support for Intel 64 Architecture

2-15.15. Intel Nehalem Microarchitecture (2009 - Current)

Nehalem is the codename for the Intel microarchitecture, which is successor to the Core microarchitecture. The architecture is named after the Nehalem River in Northwest Oregon, which is in turn named after the Nehalem Native American tribe in Oregon. As shown in figure 2-22, this architecture offers 6 dispatch ports: one Load, two Store and three universal ports for 4 decoders, one for complex instructions (CD) and three for simple instructions (SC). The first processor released with the Nehalem architecture is the desktop Core i7, which was released on 2009. Server and mobile Core i7 processors followed in 2010 and 2011.

2-15.16. Sandy Bridge & Ivy Bridge Microarchitectures (2011-now)

The Sandy Bridge microarchitecture was developed by Intel to replace the Nehalem microarchitecture. Intel demonstrated a Sandy Bridge processor in 2009, and released first products based on this architecture in 2011 under the Core brand. Originally, implementations were in 32nm technology process using double-gate MOSFET transistors. One of the main features of this microarchitecture is the integration of the memory controller, integrated graphics and processor into single die. This architecture permits up to 8 physical cores or 16 logical cores through Hyper-threading.

The **Sandy Bridge** architecture was followed by ***Ivy Bridge*** micro-architecture, which makes use of the 22nm technology. The Ivy Bridge is based on 3D tri-gate MOSFET transistors and was demonstrated by Intel in 2011. However, lately in 2013, Intel demonstrated the **Haswell** architecture, which is a successor to Sandy Bridge and Ivy Bridge. **Skylake** is the Intel 6th generation of Core microarchitecture which was launched in **2015**. Skylake uses the 14nm manufacturing technology.

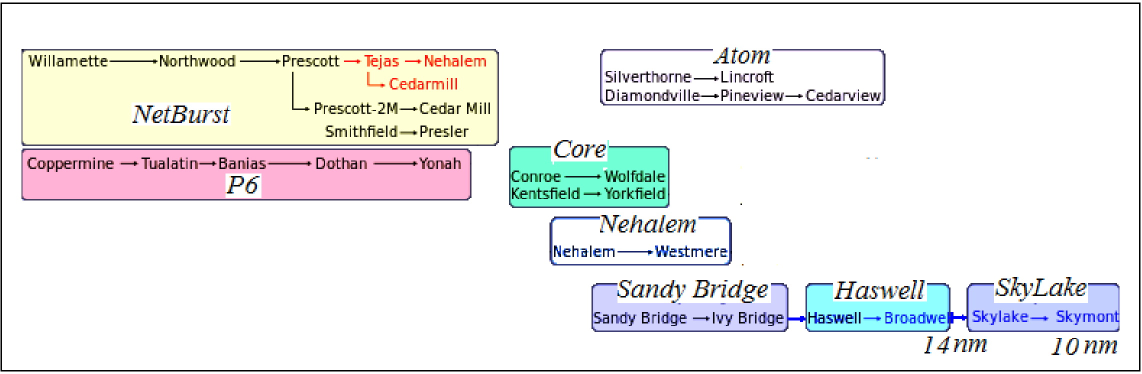


Fig. 2-26. Timeline of Intel architectures

2-16. Evolution of x86 Processors from CISC to RISC Architecture

The 80x86 family of microprocessors has been traditionally *CISC* (Complex Instruction Set Computer) devices. In such processors, each instruction needs multiple clock cycles to be read, decoded and executed. For instance, the *STA r* (store accumulator in register *r*) instruction takes 2 cycles to be read, 1 cycle to be decoded and 2 cycles to be executed. If the processor clock frequency is 100MHz, then the clock duration is 10ns and this instruction needs 50ns to be executed. Such processors have always a large number of opcodes (400 for 8086).

On the other hand, the so-called *RISC* (Reduced Instruction Set Computer) processors (like **ARM**, **SPARC**, **Alpha**, and **PowerPC**) have a few number of simple opcodes. RISC processors are faster than CISC processors because they use simpler fixed-length instructions and their architecture enables higher performance through the use of pipelining and superscalar execution. In order to increase the performance of the latest generations of processors, Intel and its competitors have borrowed from RISC technology.

The first hint of the incorporation of RISC technology into the x86 family came about in 1990 with the integration of a floating-point unit (**FPU**), and by incorporating more hard-wired instructions and pipelining. The FPU was Intel's response to the superior floating-point performance of RISC processors. Pipelining and reduced micro-code enabled 80486 processor to process many instructions at an effective rate. RISC processors achieve the same result by using simpler instructions that require fewer clock cycles.

Cyrix Company adapted these techniques to its improved 80386 chips and created hybrids like the 486SLC. Texas Instruments and IBM took the technology even further with IBM using a larger cache memory and introducing the clock doubling technology. Because the 80486 processor has only one pipeline its theoretical throughput limit is one instruction per clock cycle and so Intel provided the Pentium with two pipelines so it could handle two instructions simultaneously. This allows the Pentium to issue some instructions at a rate of greater than one per clock cycle. The nature of CISC instructions (like *compare* and *jump*) makes multiple pipelines difficult to implement.

RISC processors generally use fixed length instructions whereas CISC processors use variable length instructions ranging in length from 8 to 120 bits. This means a CISC processor must decode each instruction before it fetches the next one.

By overcoming these limitations of a CISC processor, the Pentium and later Core processors may be considered as hybrid CISC/RISC processors. Another limitation imposed by the CISC origin of the x86 family is the shortage of registers. This family of processors has only 8 **g**eneral **p**urpose **r**egisters (**GPR**) but the **Cyrix** M1 chip overcame this limit with 32 GPR's that are dynamically renamed, making it appear as there are only eight general purpose registers.

As the development of the **PowerPC** microprocessors advances and IBM and the rest of its consortium (IBM, HP, Motorola) espouse the virtues of RISC technology we started to see more and more RISC technology incorporated into Intel's microprocessor chips. However, for a given level of performance, such hybrid RISC/CISC processors have a higher count of transistors. This will generally translate to a larger chip die size, more power dissipation and more heating problems.

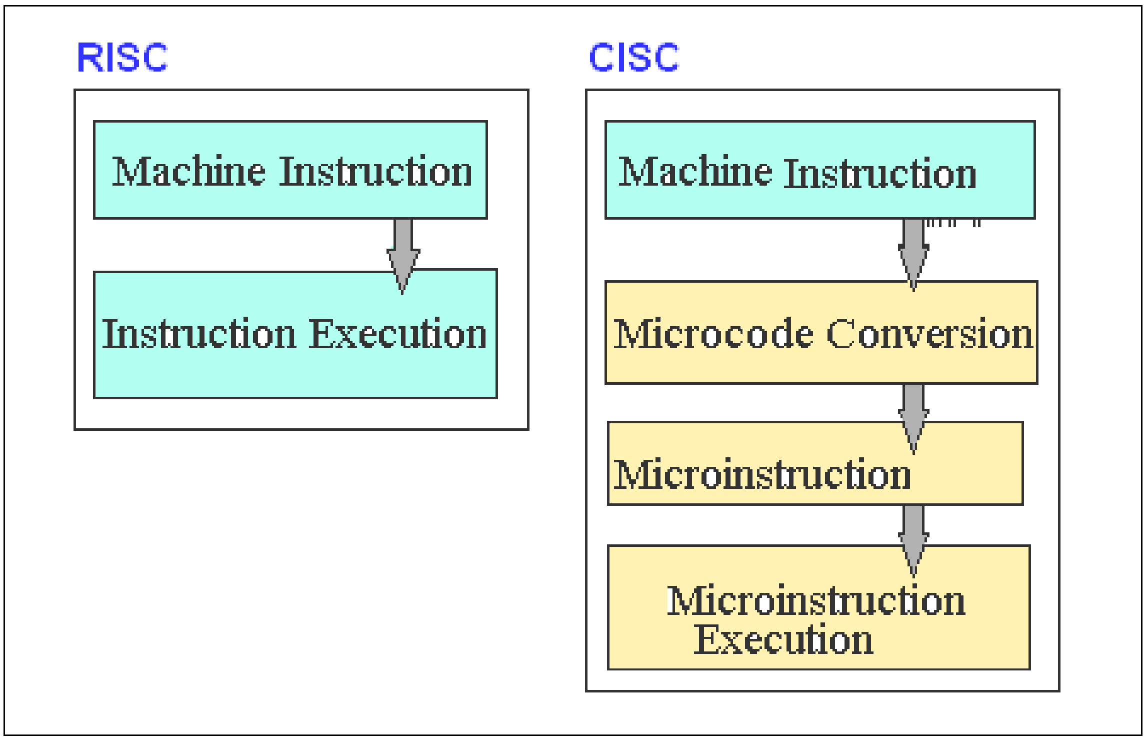


Fig. 2-27. Comparison between the execution cycles of RISC and CISC machines.

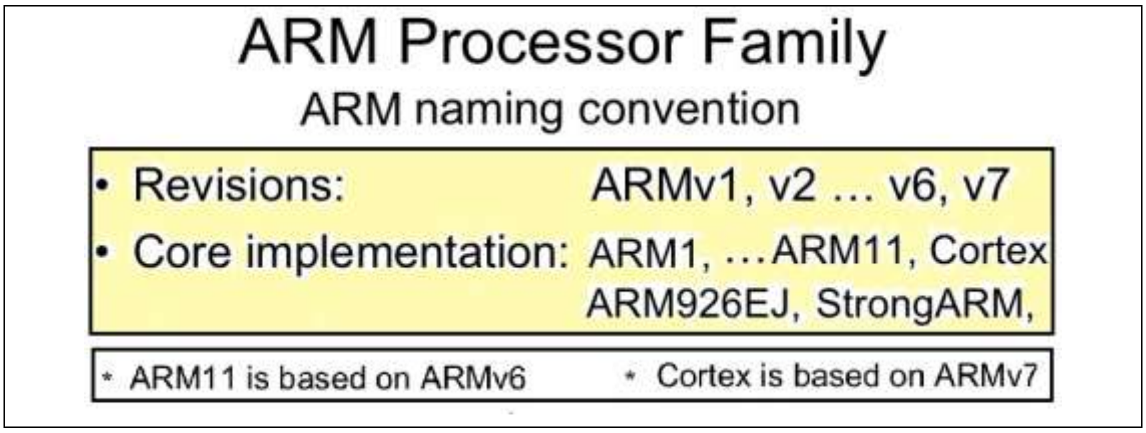
2-17. Architecture of RISC Processors

We have seen so far in Chapter 1 that the SPARC**,** PowerPC**,** MIPS, ARM and DEC Alpha have RISC architectures**.** We have also pointed out that RISC microprocessors are typically designed to have a relatively small number of instructions that can be decoded and run quickly. To enable the instructions to run quickly, the number of memory accesses is kept to the smallest amount possible. To help limit the number of memory accesses, RISC machines typically have a large number of registers. Both MIPS, ARM and SPARC machines have a relatively large number of registers for the programmer/compiler to use. In the MIPS and ARM machine, there are 32 registers that the program can use. In the SPARC machine there are 32 registers that the program can use at a time.

Generally speaking, RISC architecture offers power in even small sizes, and thus has become dominant for low-power 32-bit CPUs. However, other 64-bit RISC processors are also available in applications ranging from laptops to supercomputers. In this section we depict the architecture of some famous RISC processors, such as ARM and SPARC machines.

2-17.1. Architecture of ARM Processors

The ARM is a *Reduced Instruction Set Computer* (RISC), as it incorporates the typical RISC architecture features. The ARM architecture has large uniform register file and a small **orthogonal** instruction set, like most RISC processors. Orthogonal instructions set have the same format and all registers and addressing modes can be used interchangeably. Over time, the ARM architecture has evolved to include architectural features to meet the growing demand for new functionality and the needs of new and emerging markets (e.g., smart mobile phones).



ARM Processor Family

ARM naming convention

- Revisions: ARMv1, v2 … v6, v7
- Core implementation: ARM1, …ARM11, Cortex ARM926EJ, StrongARM,

* ARM11 is based on ARMv6 * Cortex is based on ARMv7

Fig. 2-28. ARM processor families

The first ARM processor, the **ARM1** was a prototype, which was never released. The ARM2 was originally called the Acorn RISC Machine. It was designed by Acorn Computers and used in the Archimedes. Their successor to the **BBC Micro** and **BBC Master** models were based on the 8-bit 6502 microprocessor. It was clocked at 8 MHz giving an average performance of 4.7 MIPS. Development of the ARM family was then continued by a new company called **Advanced RISC Machines** Ltd.

The **ARM3** added a fully-associative on-chip cache and some support for multiprocessing. This was followed by the ARM600 chip which was an ARM6 processor core with a 4kB 64-way set-associative cache, an MMU based on the MEMC2 chip, a write buffer and a coprocessor interface. The **ARM7** processor core uses half the power of the ARM6 and takes around half the die size. In 1994 VLSI Technology, Inc. released the ARM710 processor chip. The subsequent ARM11 micro-architecture represented a major step in embedded systems. By scaling both the clock frequency and the supply voltage, the developer can control power consumption and performance. First ARM11 processors are implemented in 0.13µm process technology and dissipate less than 0.4 mW/MHz when they are powered by 1.2V. Figure 2-28 depicts one of the ARM11 processors and a roadmap to over 1GHz. As shown, the ARM11 processor contains an AMBA interface, which improves memory bus performance and facilitates "right-first-time" development of embedded systems with multiple peripherals.

Instructions for ARM cores have 32-bits wide fixed-length instructions, but later versions of the architecture also support a variable-length instruction set that provides both 32 and 16 bits wide instructions for improved code density. Instructions are split into ***load*** and ***store*** which access memory and arithmetic/logic instructions which work on registers (two sources and one destination. The ARM has 27 registers of which 16 (R0-R15) are accessible in any particular processor mode. The ALU includes a barrel-shifter allowing, single-cycle shift and add.

The ARM processor has four processor modes:

1- **User** mode,
2- **Interrupt** mode (with a private copy of R13 and R14),
3- **Fast interrupt** mode (private copies of R8 to R14) and
4- **Supervisor** mode (private copies of R13 and R14).

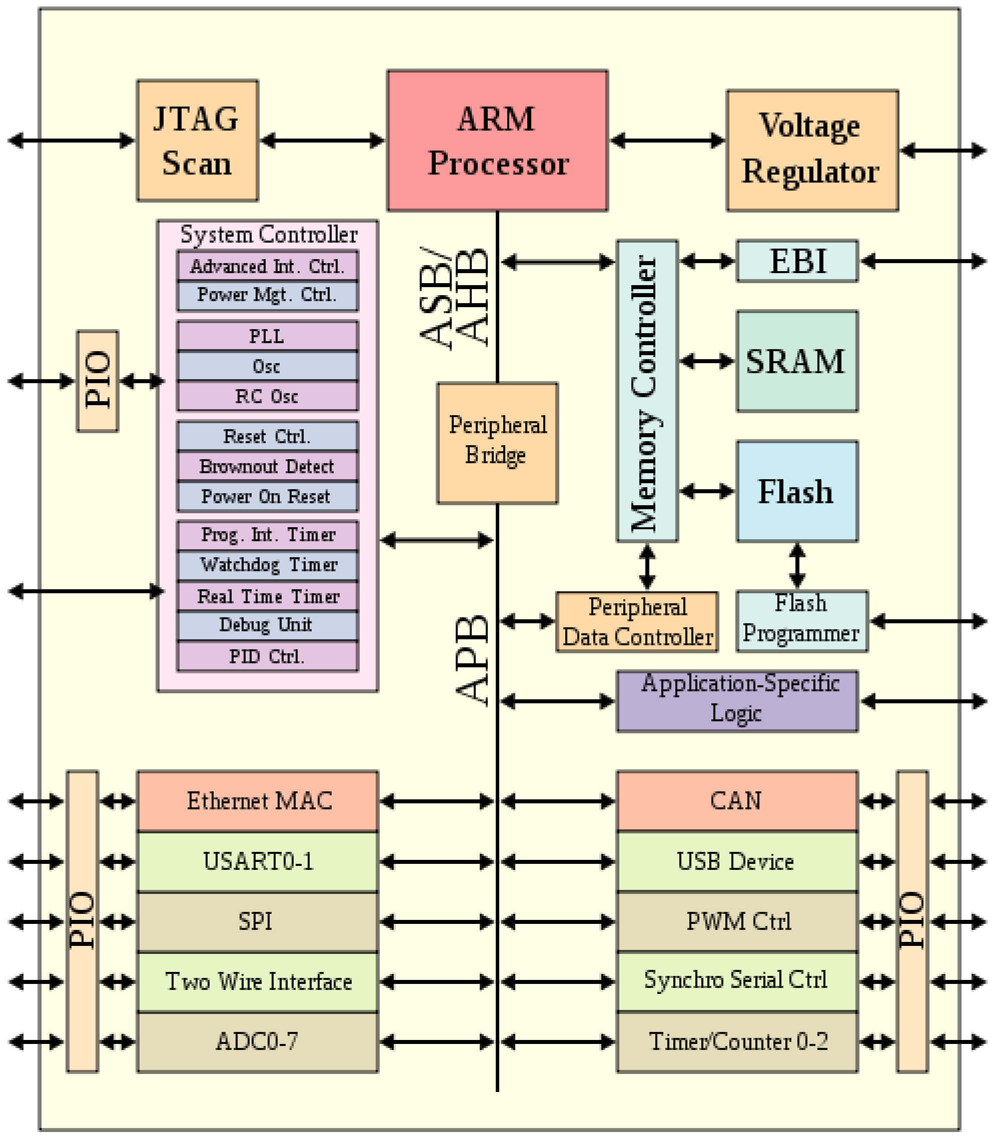


Fig. 2-29. Architecture of ARM1176JZ processor with ARM11 core.

All modern ARM processors include hardware debugging facilities, allowing software debugging operations such as halting, stepping, and break-points. These facilities are built using JTAG support.

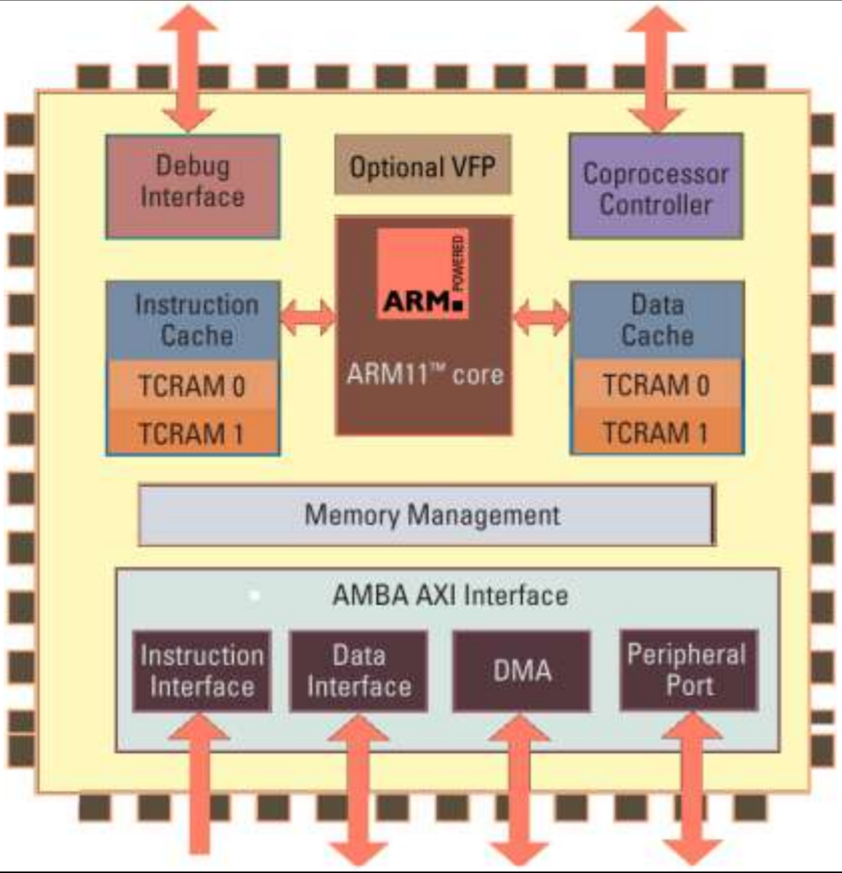


Fig. 2-30. Architecture of ARM1176JZ processor with ARM11 core.

The ARM architectures used in smartphones, PDAs and other mobile devices range from ARMv5, used in low-end devices, through ARMv6, to ARMv7 in current high-end devices. ARMv7 includes a hardware floating-point unit (FPU), with improved speed compared to software-based floating-point. In 2009, some manufacturers introduced netbooks based on ARM architecture CPUs, in direct competition with netbooks based on Intel Atom. ARM's CPU cores dominate the mobile space. This year (2015) the core of choice for high-end smartphones and tablets is ARM's Cortex A9 and late next year it'll be probably the Cortex A15

The first 32-bit ARM-based personal computer ran an interim operating system called Arthur. The 32-bit ARM architecture is supported by a large number of embedded and real-time operating systems, including Linux, Symbian, Debian, Windows CE, Windows Phone, Windows RT, Android, OS-9 and RISC OS

2-17.2. Architecture of SPARC Processors

The Scalable Processor ARChitecture (**SPARC**) is a CPU instruction set architecture (ISA), which is derived from the reduced instruction set computer (RISC) line of machines. Actually, the SPARC architecture was formulated at Sun Microsystems in 1984 through 1987. The SPARC architecture is based on the RISC I and II designs engineered at the University of California at Berkeley (**UCB**) from 1980 through 1982.

The SPARC architecture was first implemented by Sun in 1987, using version 7 architecture. Since then, SPARC architectures have gone through two major changes: SPARC V8 in 1990, and the latest implementation, SPARC V9, first published in 1994, which introduces 64-bit addressing support.

SPARC was licensed by SPARC International Inc., a consortium of computer makers who control the design. The scalable part of the name means that the design allows for forward compatibility of programs. Nowadays, SPARC processors are designed and implemented by Sun Microsystems, Texas Instruments, Toshiba, Fujitsu, Cypress, and Tatung, among others. These companies use the chip in applications ranging from laptops to supercomputers.

The SPARC processor can operate in either of two modes:

1- **User** mode or
2- **Supervisor** mode.

In supervisor mode, the processor can execute any instruction, including the privileged (supervisor-only) instructions. In user mode, an attempt to execute a privileged instruction will cause a trap to supervisor software. “User application” programs are programs that execute while the processor is in user mode.

A SPARC processor logically comprises an integer unit (**IU**), a floating-point unit (**FPU**), and an optional coprocessor (**CP**), each with its own registers. This organization allows for implementations with maximum concurrency between integer, floating-point, and coprocessor instruction execution. All of the registers, with the possible exception of the coprocessor’s, are 32 bits wide. Instruction operands are generally single registers, register pairs, or register quadruples.

In all SPARC machine there are 32 registers that the program can use at the same time (actually, 28 of the 32 are generally available). Since the SPARC has been optimized for subroutine calls, the actual number of registers in the microprocessor is much larger than 32 (often 124 registers exist), but only 32 registers are visible to the program at any given time (see the subroutine overview section for more information). Registers can be categorized by function since they are typically created for a given purpose. On RISC machines, if the register is not being used for the purpose it is designed for, it is available to use by any instruction. The most general registers are the "temporary" registers. These registers exist to store values loaded from memory or calculated by the ALU before being stored in memory. These registers are not preserved across subroutine calls. Basically, the program should use these as the "general use" registers for the program.

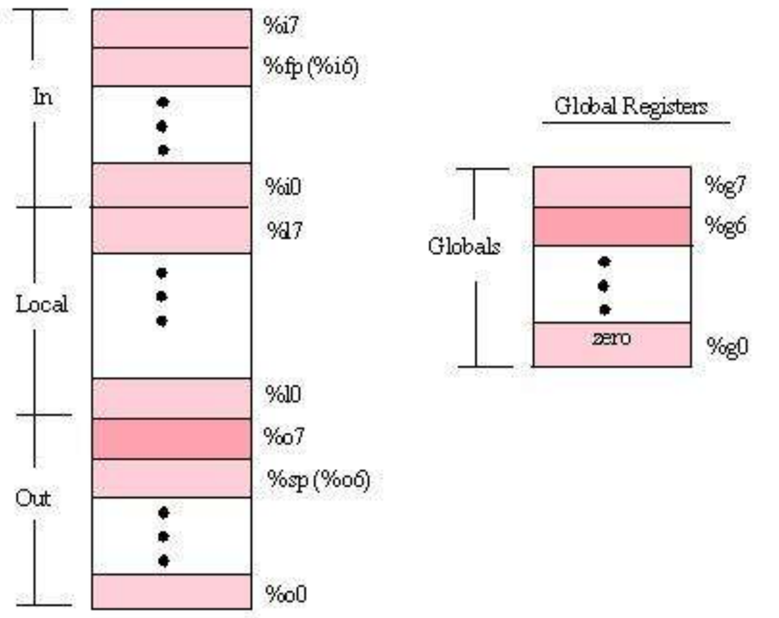


Fig. 2-31. Layout of SPARC processor registers

The SPARC machine has only eight temporary registers designated *%l0-%l7* (registers 16-23), but the SPARC often has other registers available that can also be used as temporaries if the program needs them. For memory accessing and for subroutine calls, SPARC provide a few specialized registers. Since they are keys to the operation of the machine, the program should use the registers as designed, unless they are known to not be needed. The SPARC provides a stack pointer register, *$sp* and *%sp*. It also both provides a frame pointer register, *$fp* and *%fp*. Like MIPS, SPARC reserve one register for the constant zero. This register, designated %g0 on SPARC, will always return zero when read from and will not change if written to. To provide for subroutines, there needs to be registers available for passing the arguments to the registers, returning values from the registers and returning from the subroutine. The SPARC machine has six registers for function arguments, designated %i0-%i5, six registers for function return values, designated %o0-%o5, and a function return address, designated %i7. When a subroutine call requires more arguments than the machine provides registers for, then the program must use the stack to save the additional information. The following figure depicts the layout of the SPARC registers.

i. Integer Unit (IU)

The IU contains the general-purpose registers and controls the overall operation of the SPARC processor. The IU executes the integer arithmetic instructions and computes memory addresses for loads and stores. It also maintains the program counters and controls instruction execution for the FPU and the CP. An implementation of the IU may contain from 40 to 520 general-purpose 32-bit registers. This corresponds to a grouping of the registers into 8 global registers, plus a circular stack of 2 to 32 sets of 16 registers each, known as **register windows**.

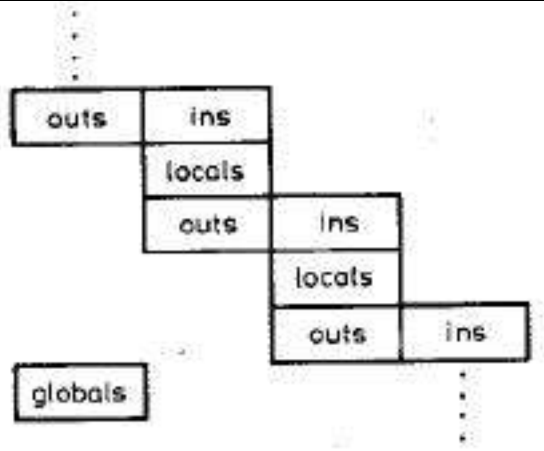


Fig. 2-32. Register organization of the Integer Unit (IU) of SPARC processors

At a given time, an instruction can access the 8 globals and a register window into these registers. A 24 register window comprises a 16-register set, divided into 8 IN registers (ins) and 8 local registers, together with the 8 IN of an adjacent register set, addressable from the current window as its OUT registers (outs).

The specified register organization is illustrated in Figure 2-28(d). The current window is specified by the current window pointer (**CWP**) field in the processor state register (**PSR**). Window overflow and underflow are detected via the window invalid mask (**WIM**) register, which is controlled by the supervisor software. The actual number of windows in a SPARC implementation is invisible to user programs. When the IU accesses an instruction from memory, it appends to the address an address space identifier (**ASI**), which encodes whether the processor is in supervisor or user mode, and whether the access is to instruction memory or data memory.

ii. Floating-point Unit (FPU)

The FPU has 32 32-bit floating-point registers. Double-precision values occupy an even-odd pair of registers, and quad-precision values occupy a quad-aligned group of 4 registers. Thus, the floating-point registers can hold a maximum of either 32 single-precision, 16 double-precision, or 8 quad-precision values. Floating-point load/store instructions are used to move data between the FPU and memory whereas Floating-Point operate (FPop) instructions are used to perform the actual floating-point arithmetic. The floating-point data formats and instruction set conform to the IEEE Standard for Binary Floating-point Arithmetic, ANSI/IEEE Standard 754-1985. An implementation can indicate that a floating-point instruction did not produce a correct ANSI/IEEE Standard 754-1985 result by generating a special floating-point unfinished or unimplemented exception. Software must emulate any functionality not present in the hardware. If an FPU is not present, or if the enable floating-point (**EF**) bit in the PSR is 0, an attempt to execute a floating-point instruction will generate an fp_disabled trap.

iii. Coprocessor (CP)

The instruction set includes support for a single, implementation-dependent coprocessor. The coprocessor has its own set of registers, the actual configuration of which is implementation-defined but is nominally some number of 32-bit registers. The Coprocessor load/store instructions are used to move data between the coprocessor registers and memory. For each floating-point load/store in the instruction set, there is an analogous coprocessor load/store instruction.

2-17.3. Architecture of SuperSPARC Processors

A SuperSPARC microprocessor is a superscalar SPARC microprocessor, which is compatible with the SPARC V8 (version 8) architecture. A block diagram of the device is shown in Figure 2-29. The processor contains an integer unit (IU), double precision floating point unit (FPU), fully consistent instruction and data caches, a SPARC reference Memory Management Unit (**MMU**), and a dual mode bus interface.

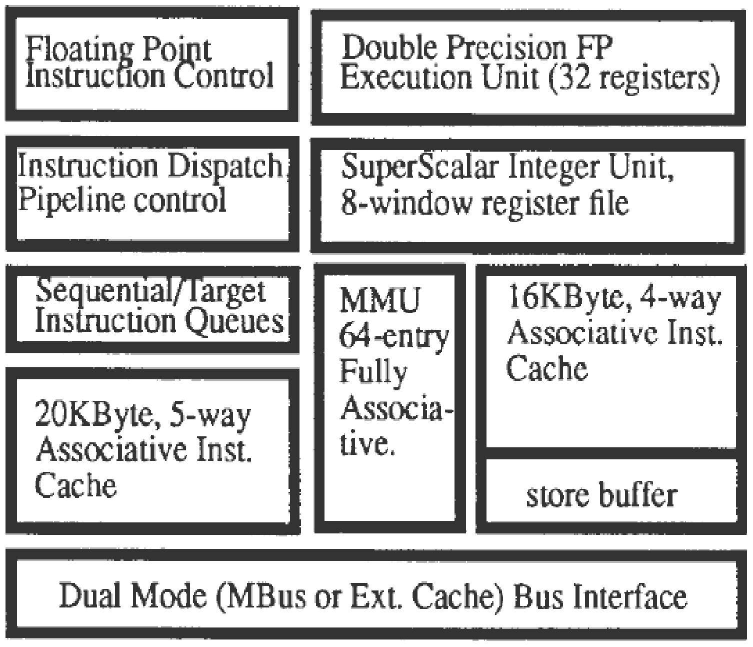

Floating Point Instruction Control
Double Precision FP Execution Unit (32 registers)
Instruction Dispatch Pipeline control
SuperScalar Integer Unit, 8-window register file
Sequential/Target Instruction Queues
MMU 64-entry Fully Associa-tive.
16KByte, 4-way Associative Inst. Cache
20KByte, 5-way Associative Inst. Cache
store buffer
Dual Mode (MBus or Ext. Cache) Bus Interface

Fig. 2-29(c). Block diagram of a SuperSPARC processor

2-17.4. Architecture of SPARC64 Processors

Since 1998 and for quite some time, Fujitsu has been making its own SPARC implementation, called SPARC64. Presently the SPARC64 is in its 6th generation (SPARC64 VI). The processor is able to execute the SPARC instruction set but the processor internal design is different from Sun's implementation. Since 2008, SPARC Enterprise mid-range and high end models incorporated Quad cores processors SPARC64 VII, together with SPARC64 VI. Figure 2-30 shows the block diagrams of the dual core SPARC64 VI and V2 SPARC64 architectures. Also, figure 2-31 depicts the details of the dual core SPARC64 VI architecture. Figure 2-32 illustrates the block diagram of the Fujitsu SPARC64 VI processor **chip**. Two cores share the L2 cache. What is not shown in the diagrams is that, like the IBM and Intel processors, the SPARC VI is dual-threaded per core.

The type of multithreading is similar to that found in the Intel processors and is called Vertical Multithreading (**VMT**). The multithreading technology is illustrated in section 2-16.7.

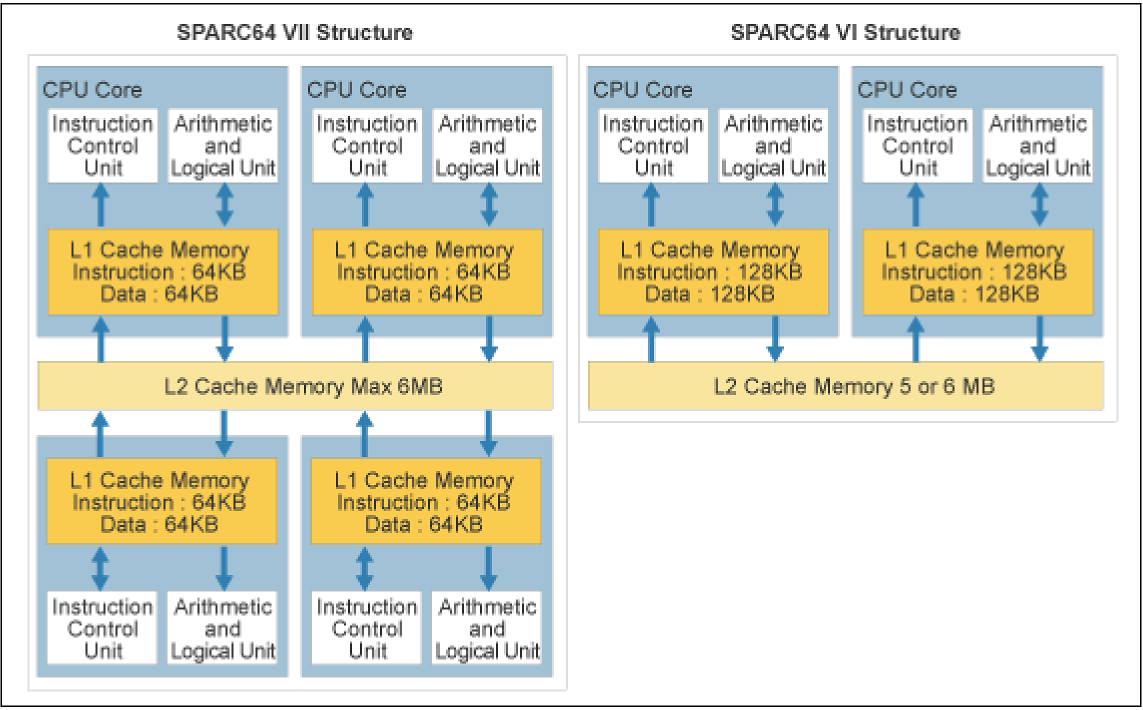

SPARC64 VII Structure
SPARC64 VI Structure
CPU Core
Instruction Control Unit
Arithmetic and Logical Unit
L1 Cache Memory Instruction : 64KB Data : 64KB
L2 Cache Memory Max 6MB
L1 Cache Memory Instruction : 128KB Data : 128KB
L2 Cache Memory 5 or 6 MB

Fig. 2-30. Block diagram of the SPARC64 architectures (VI and VII)

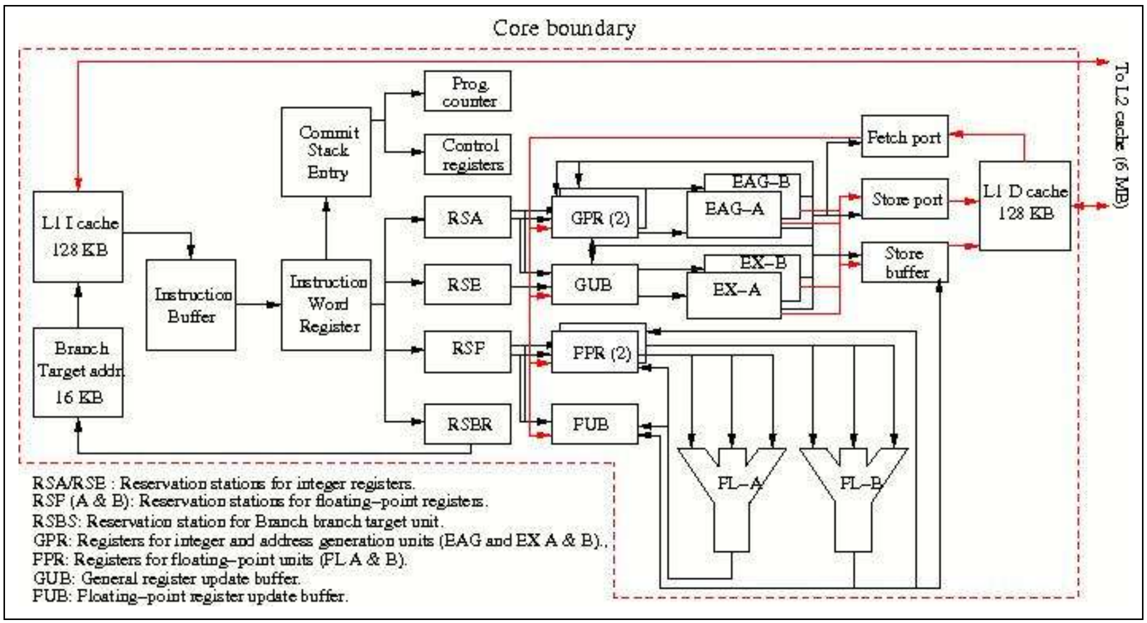

Core boundary
To L2 cache (6 MB)
Prog. counter
Control registers
Commit Stack Entry
L1 I cache 128 KB
Instruction Buffer
Instruction Word Register
Branch Target addr 16 KB
RSA
RSE
RSF
RSBR
GPR (2)
GUB
FPR (2)
FUB
EAG-B
EAG-A
EX-B
EX-A
Fetch port
Store port
Store buffer
L1 D cache 128 KB
FL-A
FL-B
RSA/RSE : Reservation stations for integer registers.
RSF (A & B): Reservation stations for floating-point registers.
RSBS: Reservation station for Branch branch target unit.
GPR: Registers for integer and address generation units (EAG and EX A & B).
FPR: Registers for floating-point units (FL A & B).
GUB: General register update buffer.
FUB: Floating-point register update buffer.

Fig. 2-31. Details of SPARC64 VI architectures

Currently, the highest clock frequency of SPARC64 is 2.4 GHz. The theoretical peak performance is presently 9.6 GFlops/core. In 2008 Fujitsu has brought out a dual core SPARC64 VI+ at 2.7GHz.

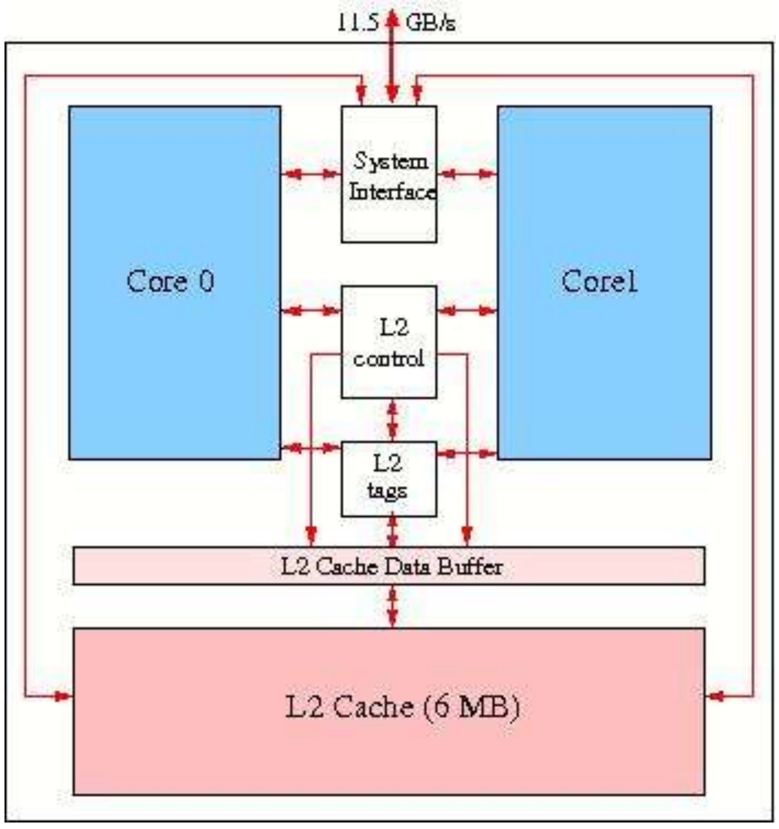


Fig. 2-32. Block diagram of the SPARC64 chip

2-17.5. Architecture of UltraSPARC Processors

The UltraSPARC Architecture 2007 architecture is derived from the SPARC architecture V9. Like its ancestors, UltraSPARC architecture consists of an integer unit (IU) and a floating-point unit (FPU), each with its own registers. This organization allows for implementations with concurrent integer and floating-point instruction execution. However, the new version of SPARC introduced 64-bit addressing, superscalar execution, instruction and data pre-fetching, and handling of nested traps.

Since the sizes of data sets and programs continue to increase, 64-bit addressing support became necessary, and SPARC took on this challenge. Integer registers have become 64 bits wide; floating-point registers are 32, 64, or 128 bits wide.

Instruction operands are single registers, register pairs, register quadruples, or immediate constants. In addition to pipelining, the new architecture calls for superscalar execution, in which more than one instruction may be fetched at a time. This causes far more serious hazards in branching than with the previous design, and version 9 solves this with branch prediction, a feature available in other processors such as MIPS during their 32-bit lives. Furthermore, instruction pre-fetching in UltraSPARC can be done before taking a known branch or call to another program. Without pre-fetching, the instructions at the branch or called location probably are not in the same area of memory as the calling routine. So, the instructions to be called are probably not in cache, either. Pre-fetching allows the programmer to speed up transition over a call by pre-fetching the instructions in the new routine to cache. .

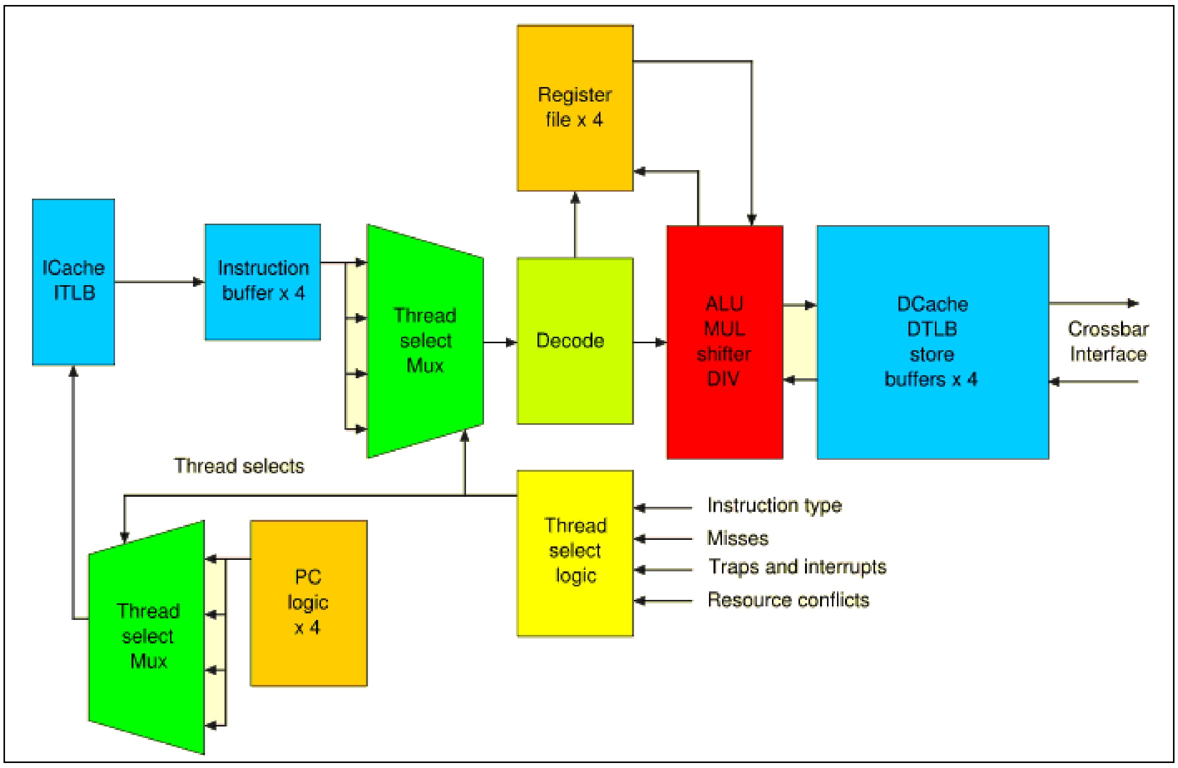


Fig. 2-34. Architecture of UltraSPARC.T1 processor and its package photograph

The UltraSPARC Architecture virtual processor can run in *nonprivileged* mode, *privileged* mode, or *hyperprivileged* mode. In hyperprivileged mode, the processor can execute any instruction, including privileged instructions. In privileged mode, the processor can execute nonprivileged and privileged instructions. In nonprivileged mode, the processor can only execute nonprivileged instructions.

In nonprivileged or privileged mode, an attempt to execute an instruction requiring greater privilege than the current mode causes a trap to hyperprivileged software.

The UltraSPARC T1 was the first multicore and multithreaded SPARC processor. The processor is available with 4, 6 or 8 CPU cores, each core able to handle 4 threads concurrently. Thus, the T1 can handle up to 32 simultaneous threads. This is called simultaneous multithread technology (**SMT**). The UltraSPARC T1 processor was designed to lower the energy consumption of Sun servers, the CPU typically uses 72 W of power at 1.4 GHz. Figure 2-34 shows the architecture of UltraSPARC T1 processor.

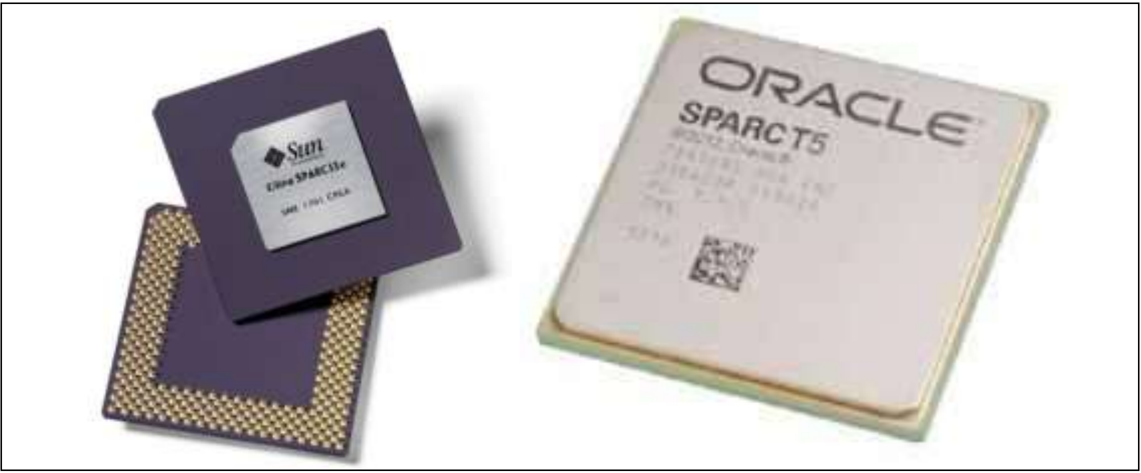


Fig. 2-35. Photographs of UltraSPARC.T1 and Oracle T5 processors

Actually, the T1 cores are less complex than those of high end processors in order to allow 8 cores to fit on the same die. The UltraSPARC T1 and T2 are designed for single CPU systems. Recent UltraSPARC processors such as **Rock** (2009) support multiple chip server architectures. The most recent commercial iterations of the SPARC processors is SPARC64 X *"Athena"* introduced in 2012, and the 16 core SPARC T5 introduced by Oracle Corporation in 2013, and running at 3.6 GHz.

2-17.6. Multithreading Technology

The UltraSPARC T1 processor is slow on single threaded work but shines on multi-threaded work. In fact, the studies which were carried out by Intel showed that even under full load, a typical x86 CPU is idle 50 to 60% of the time. This is due to cache misses whom all CPU architectures suffer from; they must wait for data to arrive from RAM..

However, CPUs belonging to the T1 family do not suffer from this problem. Instead, as soon a T1 thread stalls due to a cache miss, the T1 switches thread in 1 clock cycle and continues to do work while waiting for the data. Typically on a modern CPU, a thread switch takes a much longer time than 1 clock cycle. This is the reason a T1 can work 95% of the time and only waits for data 5% of the time. Compare this to an x86 CPU at 3 GHz. Because the x86 CPU can only work at half speed due to cache misses, it can be compared to a 1.5GHz CPU at full speed. The following figure depicts the effect of multithreading on the speed of CPU. Both virtual threading (**VMT**) and simultaneous multithreading (**SMT**) technologies are illustrated.

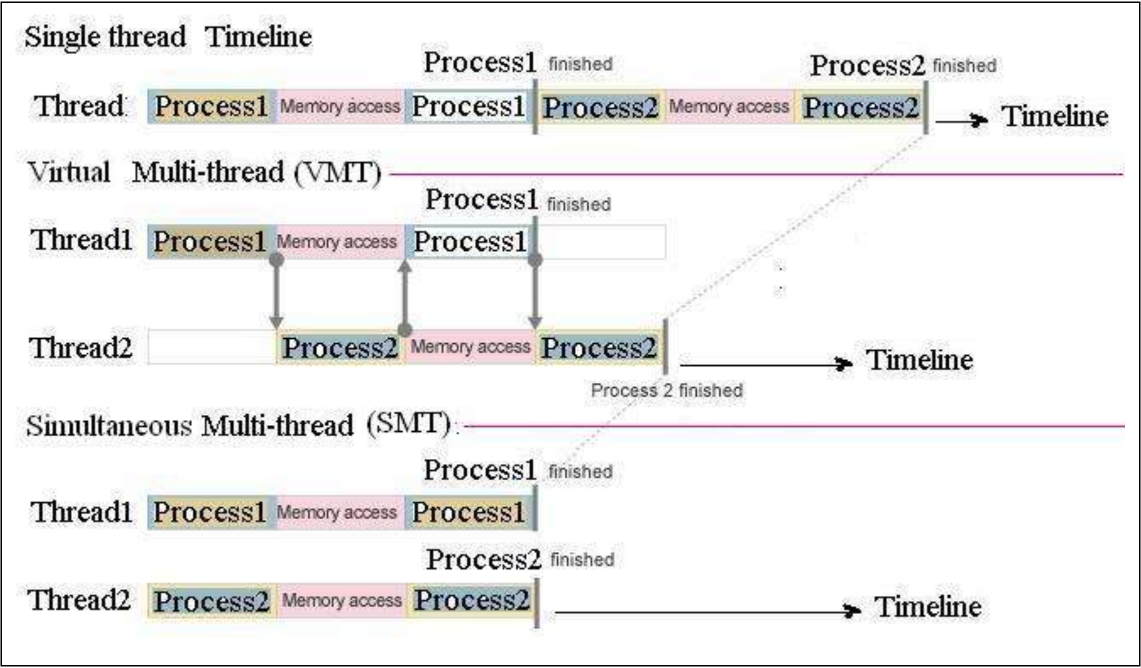


Fig. 2-36. Illustration of the virtual threading (VMT) and simultaneous threading (SMT) technologies.

2-18. CPU Market Share

Both Intel and its intimate rival AMD processors have been extensively used in the microcomputer and workstation market since relatively long time. The following figure depicts the Market shares of x86 microprocessors manufacturers, from Intel, AMD and other manufacturers, according to Mercury Research. Obviously, the CPUs based on Intel Core micro-architecture stimulated rapid changes in the computer market. As we have seen so far, these processors set new performance records for mainstream PC systems.

Thus, Intel has earned the title of the today's fastest x86 processors developer. In the meantime, the AMD processors are pushed back and started to be just a good solution for inexpensive systems. In order to retain the sales volume, AMD undertook an unprecedented reduction of the pricing on their solutions. For instance, the price of **AMD Phenom X4** 9600 (quad core with 512 kBx4 L2-Cache at 2.3 GHz), is $145.99, as declared on AMD official website, in 2009. The price of the corresponding Intel Core2 Quade, running at 2.4GHz, ranges from $184.99-$310, depending on the Cache size.

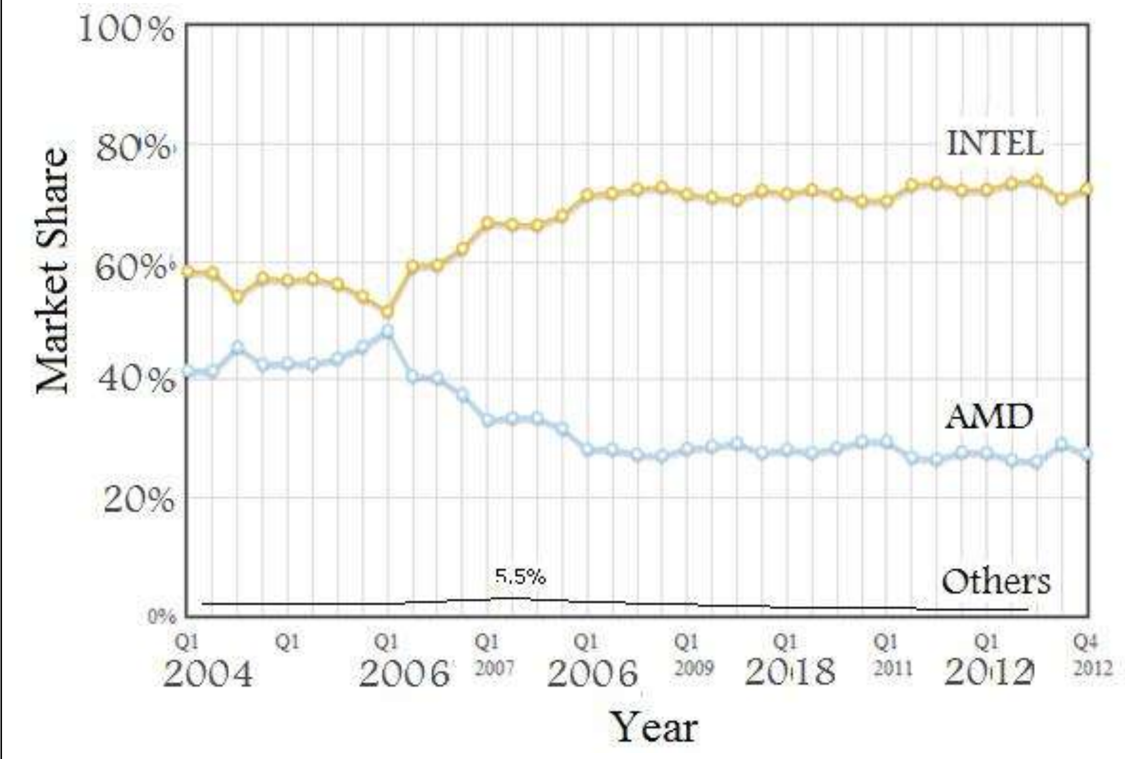


Fig. 2-37. Market shares of x86 microprocessors manufacturers, according to Mercury Research.(March 2013)

As for SPARC processors, they have failed long ago on the desktop and still being insignificant in the overall notebook market (despite the availability of technically impressive products), Therefore, unlike Intel and AMD architectures, SPARC is best viewed solely as a server processor architecture. The market prospects for **all** servers (not just SPARC) is driven by the following considerations.

- The Credit Crunch: for most enterprises uncertainty about their future survival and lack of funding - will mean that spending on new servers will be done only as a last resort when all other options are exhausted.
- Virtualization: time sharing applications capacity into a common pool of less servers is already a well established trend in the market. This will continue to a more rigorous degree.
- Fatter Multi-Core Processors: As the number of core heads into double digits it satisfies many customer needs by reducing the physical and energy footprint of server installations - as well as reducing cost.

2-19. Moore's Law

Along the above discussion, we have seen that each generation of microprocessor chips is more powerful than its predecessor. This is because the electronic industry is constantly increasing the complexity (the number of integrated transistors on the same chip) and reducing the feature size of the integrated circuit.

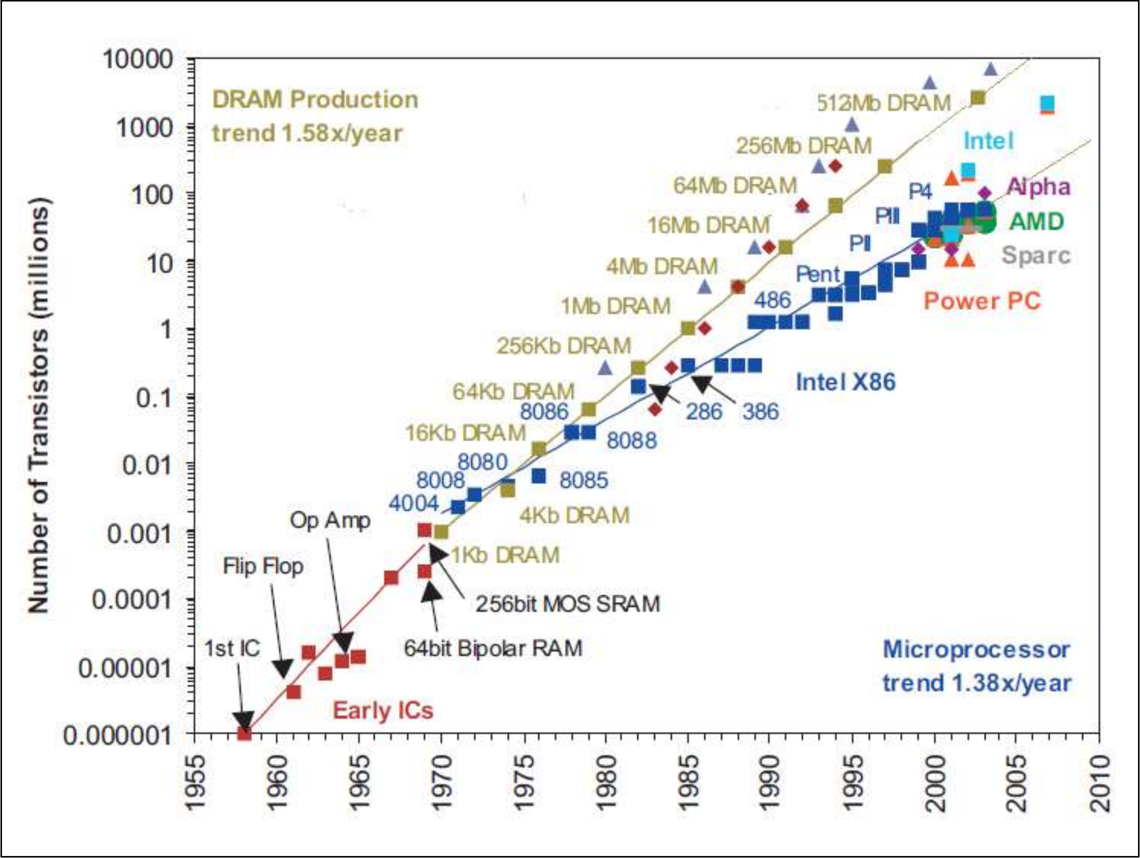


Fig. 2-38. Microprocessors and DRAM roadmap.

In the mid-1960s, the Intel chairman of the Board Gordon Moore deduced a principle or "law" which has continued to be true for over three decades: *the computing power and the complexity* (roughly, the number of transistors per chip) *of the silicon integrated circuit microprocessor doubles every one to two years, and the cost per CPU chip is cut in half.* This law is the main explanation for the computer revolution, in which the Intel Architectures (**IA**) play such a significant role.

2-20. Summary

In this chapter we described the architecture of some famous microprocessors, with emphasis on x86 microprocessors. The jargon computer terms, such as *pipelining*, *threading* and *virtualization*, which usually appear in microprocessor datasheets and the advertisements of CPU vendors, have been explained in a simple didactic manner.

The **80x86** is the generic name of Intel microprocessor architecture. The generic term **x86** refers to the instruction set of the most commercially successful CPU architecture in the history of personal computing. The Intel 8086 CPU was the first of the x86 architecture, which appeared in 1978. Three years later, the 8088 (an eight-bit data bus version of 8086), was chosen as the main CPU for the IBM PC.

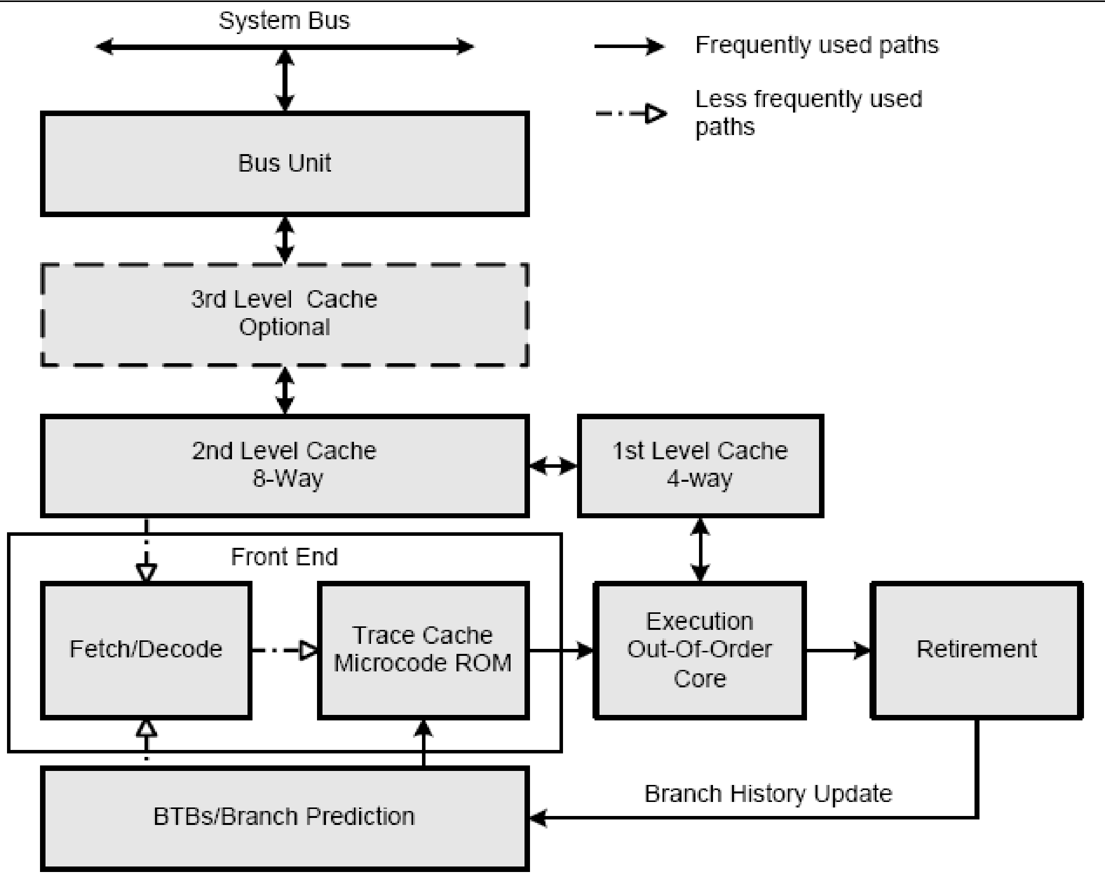
The architecture has twice been extended to larger data bus sizes. In 1985, Intel released the 32-bit 80386 to replace the 16-bit 80286. This extension to the x86 architecture is commonly called **IA-32** (**I**ntel **A**rchitecture, 32-bit). In 2003, AMD further extended the architecture to 64 bits, variously called **x86-64** or AMD64. Intel 64 should not be confused with the unrelated IA-64 architecture

The x86 architecture is a variable instruction length, CISC design with emphasis on backward compatibility. The instruction set is not typical CISC however, but basically an extended and orthogonalized version of the simple eight-bit 8085 architecture. Words are stored in little-endian order and 16-bit and 32-bit accesses are allowed to unaligned memory addresses. To conserve ***opcode*** space, most register-addresses are three bits, and at most one operand can be in memory (in contrast with some highly orthogonal CISC designs such as PDP-11 where both operands can be in memory), but this memory operand may also be the *destination*, while the other operand, the *source*, can be either *register* or *immediate*. This contributes, among other factors, to a code footprint that rivals 8-bit machines and enables efficient use of instruction cache memory. During execution, current x86 processors employ a few extra decoding steps to split most instructions into smaller pieces, micro-ops (μOps), which are readily executed by a micro-architecture that may be described as a RISC-machine without the usual load/store limitations. The small number of general registers (inherited from 8085) has made register-relative addressing (using small immediate offsets) an important method of accessing operands, especially on the stack.

Much work has therefore been invested in making such accesses as fast as register accesses, i.e. one cycle instruction throughput in most circumstances. The following table summarizes the common architecture steppings of the x86 processors:

Generation	**Date**	**CPU Brands**	**Linear / Physical Address space**	**Features**
1st (IA-16)	1978	Intel 8086, Intel 8088	**16-bit** / 20-bit (segmented)	first x86 microprocessors
2nd	1982	Intel 80186, 80188, NEC V20	as above	fast address, fast mul/div etc
2nd	1982	Intel 80286	**16-bit** (30-bit virtual) / 24-bit (segmented)	MMU, protected mode and larger address space
3rd (IA-32)	1985	Intel386, AMD Am386	**32-bit** (46-bit virtual) / 32-bit	32-bit instruction set, MMU, paging
4th	1989	Intel486	see above	RISC-like pipelining, FPU, on-chip cache
5th	1993	Pentium, Pentium MMX	see above	superscalar, 64-bit data bus, faster FPU, MMX
5/6th	1996	Cyrix 6x86, Cyrix MII	see above	register renaming, speculative execution
6th	1995	Pentium Pro, AMD K5	**36**-bit physical (PAE)	μ-op translation, integrated L2 cache
6th	1997	AMD K6, Pentium II/III	see above	L3-cache support, 3D Now, SSE
7th	1999	Athlon, Athlon XP	see above	superscalar FPU, wide design
7th	2000	Pentium 4	see above	deep pipelined, SSE2, hyper-thread
6/7th -M	2003	Pentium M	see above	optimized for low power
8th (x86-64)	2003	Athlon 64	**64-bit** / 40-bit physical.	x86-64 instruction set, on-die memory controller
8th	2004	Prescott	*see above*	very deeply pipelined, high frequency, SSE3
9th	2006	Intel Core, Intel Core 2	*some versions are 32-bit only*	low power, multi-core, lower clock frequency
10th	2007-2008	AMD Phenom, K10	*see above*	4-core, 128 bit FPUs, SSE4 , on-die L3 cache

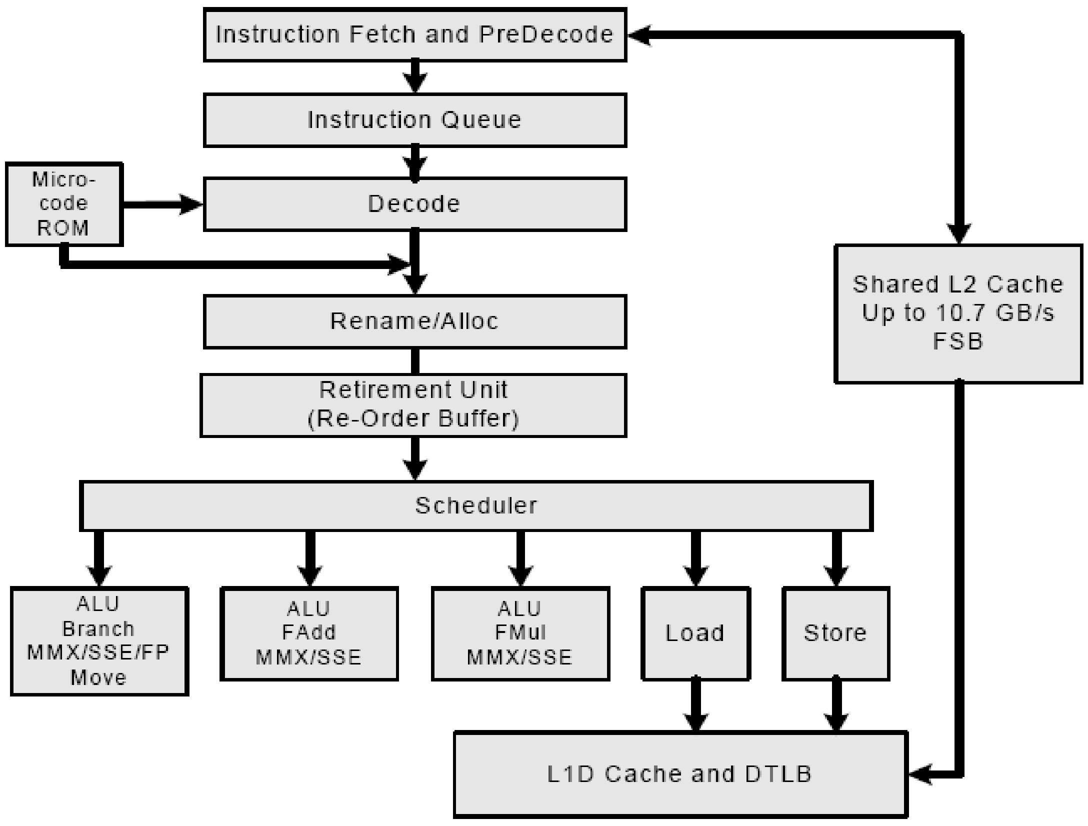
It should be noted that the P6 family processors are 32-bit Intel Architecture (**IA-32**) processors. This includes the Pentium Pro, Pentium II, Pentium III, and Pentium III Xeon processors. The Pentium 4, Pentium D, and Pentium processor Extreme Editions are based on the Intel **NetBurst** Microarchitecture. Most early Intel Xeon processors are also based on the Intel **NetBurst Microarchitecture**.



Intel Netburst MicroArchitecture

The Intel **Core** (Solo and Duo) and dual-core Intel Xeon processor are based on an improved Pentium **M processor architecture**.

The Intel Pentium dual-Core, Intel Core Duo, Core Quad and Core Extreme, Intel Xeon 3x00 and 7x00 series processors are all based on **Intel Core Microarchitecture**. The Intel Core2 Duo, Core2 Quad and Core2 Extreme, Xeon 5x00 series are based on **Enhanced Intel Core microarchitecture**.


Instruction Fetch and PreDecode
Instruction Queue
Micro-code ROM
Decode
Rename/Alloc
Retirement Unit (Re-Order Buffer)
Scheduler
Shared L2 Cache Up to 10.7 GB/s FSB
ALU Branch MMX/SSE/FP Move
ALU FAdd MMX/SSE
ALU FMul MMX/SSE
Load
Store
L1D Cache and DTLB

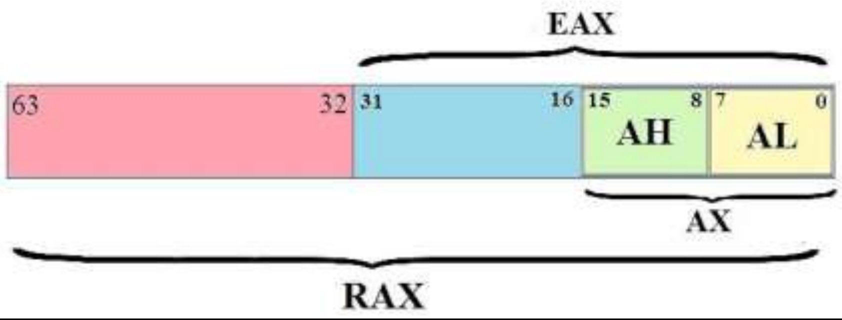
Intel Core MicroArchitecture

The Intel Penryn mircoarchitecture, which included the Core 2 family of processors, was the first mainstream Intel microarchitecture based on the 45nm fabrication process. This allowed Intel to create higher performance processors that consumed less power than previous processors. The Intel Nehalem microarchitecture encompasses the Core i7 class of processors and uses a 45nm fabrication process. The 2nd generation of Nehalem processors uses 32nm technology and was codenamed **Sandy Bridge**. The **Sandy Bridge** architecture was followed by ***Ivy Bridge*** micro-architecture, which makes use of the 22nm technology. The Ivy Bridge is based on 3D tri-gate MOSFET transistors and was demonstrated by Intel in 2011. However, in 2013, Intel demonstrated the **Haswell** architecture. **Skylake** is the Intel 6th generation of Core microarchitecture which was launched in **2015**. Skylake uses the 14nm manufacturing technology.

Through the continuous development of x86 architecture, the processor internal registers have been changed from 16-bit to 64-bit wide. Although the general-purpose registers in x86 processors can be used for anything, they were envisaged to be used for the following purposes:

- AX (16-bit) / EAX (32-bit) / RAX(64-bit): accumulator
- BX / EBX / RBX: base
- CX / ECX / RCX: counter
- DX / EDX / RDX: data/general
- SI / ESI / RSI: "source index" for string operations.
- DI / EDI / RDI: "destination index" for string operations.
- SP / ESP / RSP: Stack pointer for top address of the stack.
- BP / EBP/ RBP: stack base pointer to hold the address of stack frame.
- IP/EIP/RIP: Instruction pointer. Holds the current instruction address.

However, the segment registers (CS, DS, SS, ES) are still 16-bit wide for the matter of compatibility with previous processor generations.



The x86 processors also include various special/miscellaneous registers such as **control registers** (CR0 through CR4), **debug registers** (DR0 through 3, plus 6 and 7), **test registers** (TR4 through TR7), **descriptor registers** (GDTR, LDTR, IDTR), and a **task register** (TR). In addition to the increase of the size of internal registers, new technologies have been introduced, with the advent of new processor architectures.

Floating point unit (FPU): Initially, IA-32 included floating-point capabilities only on co-processors (8087, 80287 and 80387.) With the introduction of the 80486, the 8 floating point registers, known as ST(0) through ST(7) were built into the CPU. Each register is 80 bits wide and stores numbers in the double precision format of the IEEE floating-point standard. These registers are accessible like a LIFO stack. The register numbers are not fixed, but are relative to the top of the stack; ST(0) is the top of the stack, ST(1) is the next register below the top of the stack, etc. This means that data is pushed down from the top of the stack, and operations are always done against the top of stack. So you couldn't just access any register randomly, it has to be done in the stack order.

MMX is an SIMD instruction set designed by Intel, introduced in 1997 for Pentium microprocessors. It first appeared in the Pentium MMX. It is supported on most subsequent IA-32 processors by Intel and other vendors. MMX is typically used for video applications. **MMX** added 8 new registers to the architecture, known as MM0 through MM7 (referred to as **MMn**). In reality, these registers were just aliases for the existing x87 FPU stack registers. Hence, anything that is done to the floating point stack will also affect the MMX registers. Unlike the FP stack, these MMn registers were fixed, not relative. Each of the **MMn** registers are 64-bit integers. The upper 16-bits of the stack registers are unused in MMX, and these bits are set to ones, which makes it look like NaN's or infinities in the floating point view.

Single Instruction/Multiple Data (**SIMD**): SIMD is a technology used in parallel processors, where many processing elements perform the same operations on different data. There is often a central controller which broadcasts the instruction stream to all the processing elements. SIMD technology was introduced into x86 processors, starting from Pentium III.

Streaming SIMD Extensions (**SSE**): Streaming SIMD extensions instruction set. The **SSE** enables high levels of performance in multimedia applications like 3-D graphics, video decoding, and speech recognition In 1999, Intel introduced the SSE instruction set, following in 2000 with **SSE2**. The first addition made MMX almost obsolete and the second allowed the instructions to be realistically targeted by conventional compilers. Introduced in 2004 along with the Prescott revision of Pentium 4, **SSE3** added specific memory and thread-handling instructions to boost the performance of Intel's processors. **SSE** discarded all legacy connections to the FPU stack.. The designers created eight 128-bit registers, named XMM0 through XMM7. In AMD64, the number of SSE XMM registers has been increased from 8 to 16. However, the operating systems had to have an awareness of this new set of instructions to save their register states. Intel created a slightly modified version of Protected mode, called **Enhanced mode** which enables the usage of SSE instructions, whereas they stay disabled in regular Protected mode.

SSE2 is much more suitable for scientific calculations than previous technologies such as SSE1 or 3DNow! From AMD, which were limited to only single precision. **SSE4A** is a further extensions to the SSE instruction set.

Virtualization: x86 virtualization is difficult because the architecture did not meet the so-called "Popek and Goldberg requirements" until recently. Nevertheless, there are several commercial x86 virtualization products, such as VMware, Parallels and Microsoft Virtual PC.Intel and AMD have introduced x86 processors with hardware-based virtualization extensions that overcome the classical virtualization limitations of the x86 architecture. These extensions are known as Intel VT (**IVT**) and **AMD-V**. Although most modern x86 server-based and many modern x86 desktop-based processors include these extensions, the technology is generally considered immature at this point.

Multithreading: The studies showed that even under full load, a typical x86 server CPU is idle about 50% of the time. This is due to cache misses which all CPU architectures suffer from; they must wait for data to arrive from RAM. However, CPUs belonging to the SPARC T1 family do not suffer from this problem. Instead, as soon a T1 thread stalls due to a cache miss, the T1 switches thread in 1 clock cycle and continues to do work while waiting for the data. Typically on a modern CPU, a thread switch takes a much longer time than 1 clock cycle.

Quick Path Interconnect (QPI): In the development of Nehalem's microarchitecture, Intel had to remove some of the limitations of last generation's Core2 Duo and Quad processors. The first thing to go was the Front Side Bus, the old system interconnect that allowed the CPU, motherboard and memory to talk to one another. With processors getting faster and memory sizes getting larger, the FSB became increasingly choked up transferring data between the CPU and RAM. Intel's solution is a new point-to-point bi-directional bus called Quick Path Interconnect that transfers data directly between the CPU and the chipset. Every Core i7 processor is actually equipped with two QPI links, which could potentially allow for future multi-processor systems. It's exciting, but we're not quite there yet. For now though, QPI simply connects the CPU to the motherboard over a set of 20-bit wide connections that operate at either 4.8GHz (Core i7) or 6.4GHz (Core i7 Extreme Edition). Since these links are bi-directional and allow the CPU and the chipset to both send and receive information simultaneously, the end result is 19.2GB/s of bandwidth between the Intel Core i7 and the Intel chipset. The important thing to take away is that the Core i7 won't become bandwidth bottlenecked anytime, thanks to QPI.

2-21. PROBLEMS

2-1) Draw a general block diagram describing the internal architecture of the 8086 microprocessor and explain briefly each block.

2-2) Calculate the maximum address space of the 8086 microprocessor and the maximum number of segments that can be located in this space .

2-3) List and explain the significance and use of the general-purpose registers in the 8086 microprocessor.

2-4) Explain how the address/data lines can be de-multiplexed in 8086 and 8088 microprocessors using octal latches like the 8282 and bus transceivers 8286 chips.

2-5) In which microprocessor does the concept of pipeline was first introduced?
(a) 8086 (b) 80286 (c) 80386 (d) 80486

2-6) Explain how the bus controller chip 8288 can be used to obtain main control signals from 8086 microprocessors.

2-7) What's meant by a 32-bit microprocessor? What are the main features of 80386 and 80486/8088 microprocessors?

2-8) SIMD is:
(a) used in standard Pentiums but not in the MMX versions.
(b) A way of preventing wraparound.
(c) Single in-line multimedia data.
(d) Single instruction multiple data.
(e) All the above

2-9) Describe how INTR signals can be generated by the interrupt controllers and how can it interfaced to the 8086 microprocessor.

2-10) What do you know about Cash Memory and how does it operate in a microprocessor? What is meant by **L1-Cash** and **L2-Cash** memory?

2-11) Branch prediction logic**:**
(a) is another name for the prefetch register.
(b) is only used in MMX versions.
(d) attempts to guess the future steps to be taken by a program.
(e) All the above

2-12) Check the right phrases with (√) sign and false ones with (x)

[1]	The early Intel x86 processors were generally CISC processors because they made use of complex instructions sets	[ ]
[2]	The 8088 is a 8-bit microprocessor while 8086 is a 16-bit PU	[ ]
[3]	The Core 2 Duo is a dual core microprocessor, which belongs to Intel's x86 microprocessors	[ ]
[4]	RISC processors are faster than CISC processors because they use simpler fixed-length instructions and their architecture enables pipelining and superscalar execution	[ ]
[5]	In the flat memory mode, the whole memory of a 80386 micro-processor may be considered as one segment of 4GB	[ ]
[6]	The interrupt service routines are called by the CPU when IF=0	[ ]
[7]	The AF is raised (AF=1) when the addition of 8-bit numbers results in a carry	[ ]
[8]	The parity flag helps to correct memory errors, in x86 microprocessor system	[ ]
[9]	Core2 Duo is a dual core microprocessor with shared L2-Cache	[ ]
[10]	The 80486 has a built-in FPU	[ ]

2-13) Describe the main features of Pentium processors, with respect to their precursors. What's the difference between Pentium 4 and Itanium microprocessors?

2-14) Describe the meaning of the following terms:

- Pipelining,
- Super-scalar architecture,
- SEC, SIMD, MMX, SSEE, SSEE2

2-15) What are the main power saving modes, which are supported in Pentium microprocessor.

2-16) Describe the operation of the main **support chips** in the 8086/8088 – based microcomputer systems, and show how they're interfaced to the microcomputer system. **Hint**: The bus controller 8288, the programmable timer / counter PTC 8243/8244, the programmable interrupt controller PIC 8259, the programmable peripheral interface PPI 8255

2-17) Explain the difference between a directive, an operation, and an instruction. Give an example of each.

2-18) How are the integer registers named on the SPARC?

2-19) How many integer registers are there on the SPARC? For each of the integer registers that have special attributes, explain the special attributes.

2-22. Bibliography

[1] C. **MORGAN** and M. **WAITE**, 8086/8088 16-bit microprocessor primer, McGraw-Hill, **1986**.

[2] M. **Sergent**, IBM PC Inside Out, McGraw Hill, **1986**

[3] **Intel** Workshop, IAPX 286 Microprocessors, Intel corporation, **1986**.

[4] M. **THORNE**, Computer organization and assembly language programming for IBM PCs and Compatibles, Benjamin-Cummings, **1991**.

[5] J. E. **Uffenbeck**, Microprocessors: The 8080, 8085, and Z-80 Programming, Interfacing, and Troubleshooting (3rd Edition), **1988**.

[6] 1982K. **Hwang**, Advanced Computer architecture, McGraw Hill, **1993**

[7] K. **AYALA**, The 8086 Microprocessor: Programming and Interfacing the PC, CENGAGE, West Publishing, **1995**.

[8] E. **TRIEBEL**, 80386/80486 and Pentium Processors, Prentice-Hall, **1995**.

[9] Barry B. **Brey,** The Intel Microprocessors 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, and Pentium Pro Processor Architecture, Programming, and Interfacing, Book News, NY, **1999**.

[10] W. A. **Triebel**, 80386, 80486, and Pentium Microprocessor: The Hardware, Software, and Interfacing, **1999**.

[11] T. **Shanley**, Pentium Pro and Pentium II System Architecture (2nd Edition), Mind share Inc. **2000**.

[12] ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition, issue C.b, Section A2.10, 24 July **2012**

[12] http://www.intel.com
[13] http://www.x86-guide.com
[15] ARM milestones, ARM company website. Retrieved 8 April **2015**