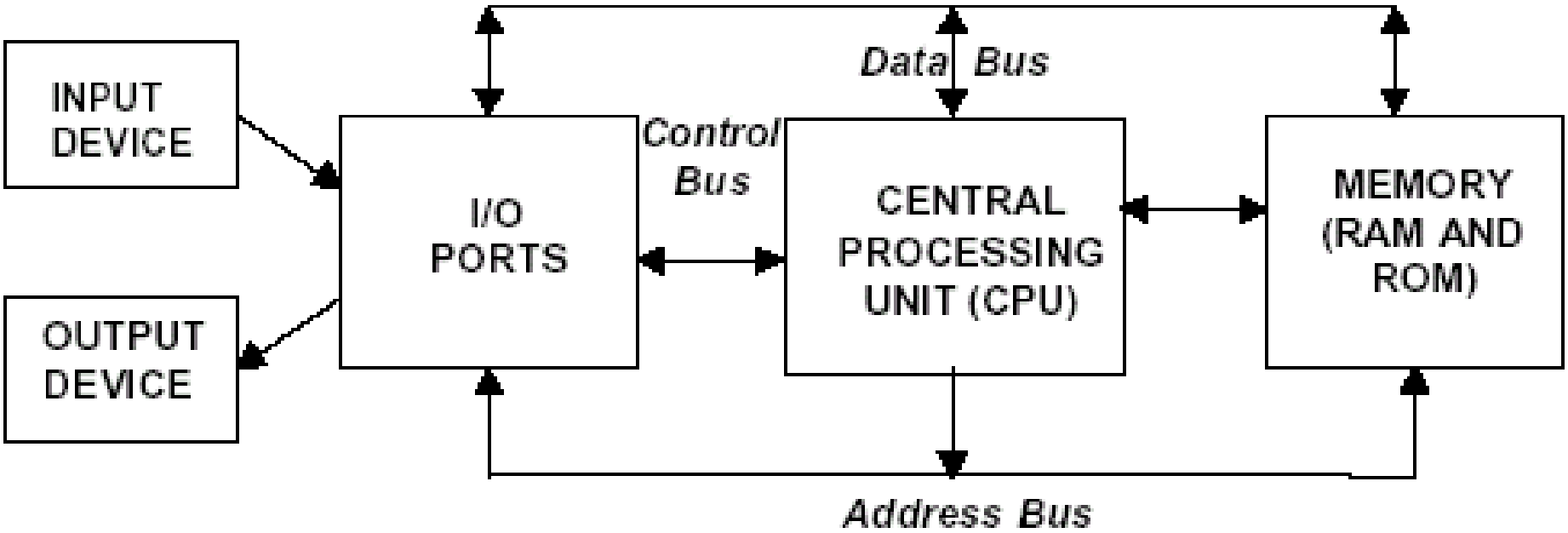
A Simple Microcomputer



Major parts: *CPU*, *Memory*, *I/O*

Buses: *address bus*, *data bus*, and *control bus*.

CPU

- The central processing unit (CPU) controls the operation of the computer.
- **Operation of the CPU**
  - fetches the instruction from memory
  - decodes the instruction into a series of simple actions (Performed in the Instruction decoder)
  - executes the instruction by carrying out these actions in a sequential manner (performed in the logic control unit)

CPU

- A typical CPU consists of
  - arithmetic-logic unit (ALU) which executes the instructions.
  - logic control unit which interprets and sequences instructions and generates the bus control signals.
  - special purpose registers such as *program counter* (PC) that holds the address of the next instruction or data item to be fetched from memory.
  - general purpose registers which are used for temporary storage of binary data- such as *accumulator A*, *status register*, etc.

Memory

- Two purposes :

  (1). to store the binary codes for the sequences of instructions that you want the computer to execute;

  (2). to store the binary-coded data with which the computer is going to work with.

- Types:

  Usually it consists of a mixture of RAM, ROM and sometimes bulk storage devices such as floppy disks, magnetic hard disks or optical disks.

Input/Output

- Allows the computer to take data from or to the outside world, through the interface. Peripherals such as keyboards, video display terminals, printers, and modems are connected to the I/O sections.
- The actual physical devices used to interface the computer buses to external systems are often called *ports*.
- The simplest type of I/O port is a set of D flip-flops.
  - If they are being used as an input port, the D inputs are connected to the external device, and the Q outputs are connected to the databus.
  - Data will then be transferred through the latches when they are enabled by a control signal from the CPU.
  - The same concept applies when using the D flip-flop for output.

Buses

- Address bus
  - Unidirectional parallel signal lines connecting CPU and the memory
  - Bus width; number of lines (wires), can be 16, 20, 24, 32, or 64 parallel signal lines.
  - On these lines the CPU sends out the address of the memory location that is to be written to or read from.
  - A CPU with *n* address lines can address 2^n memory locations. For example, a CPU with 16 address lines can address 2^{16} or 65,536 (64K) memory locations.

Buses

- Data bus
  - bus width: 4, 8, 16, 32 or 64 bits
  - *Bidirectional* parallel signal lines (wires)
  - many devices can be connected to the data bus; but only one at a time can have its output *enabled*.
  - Any device that is connected to the data bus must have *three-state* outputs, so it can be *disabled* if not being used.

Buses

- Control bus
  - consists of 4 to 10 parallel signal lines.
  - CPU sends out signal on the control bus to enable the outputs of addressed memory or port devices
  - typical control bus signals: *memory read*, *memory write*, *I/O read* and *I/O write*.

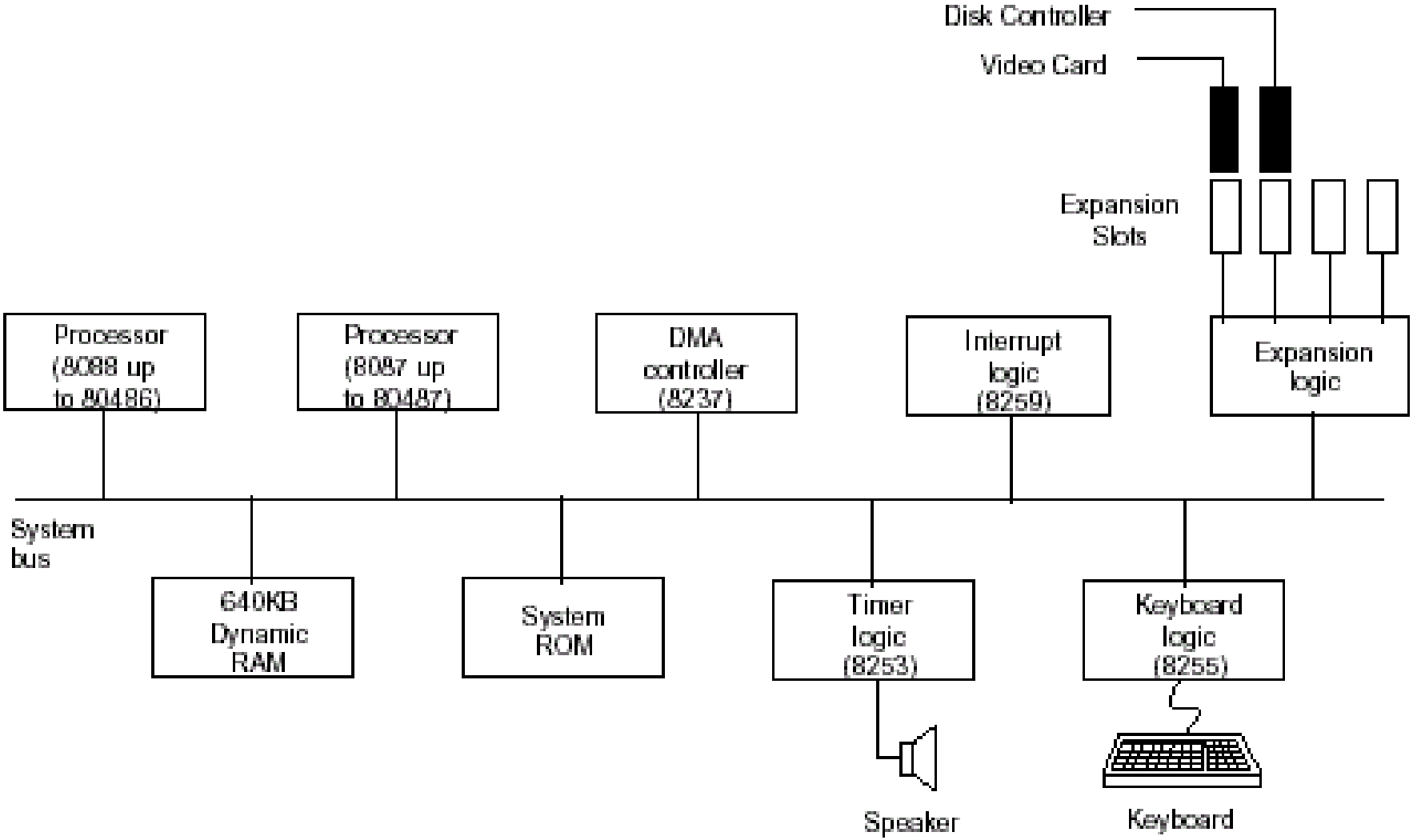
Buses

- To read a byte of data from memory location, for example, consists the following activities:
  - CPU sends out the memory address of the desired byte on the address bus.
  - CPU sends out a *Memory Read* signal on the control bus.
  - The *Memory Read* signal enables the addressed memory device to output a data word onto the data bus.
  - The data word from the memory travels along the data bus to the CPU.

Hardware, Software, and Firmware

- Hardware is the physical devices and circuitry of the computers.
- Software is referred to the program written for the computer.
- Firmware is the term given to programs stored in ROMs or in other devices which permanently keep their stored information.

A Typical PC Motherboard



Historical Background

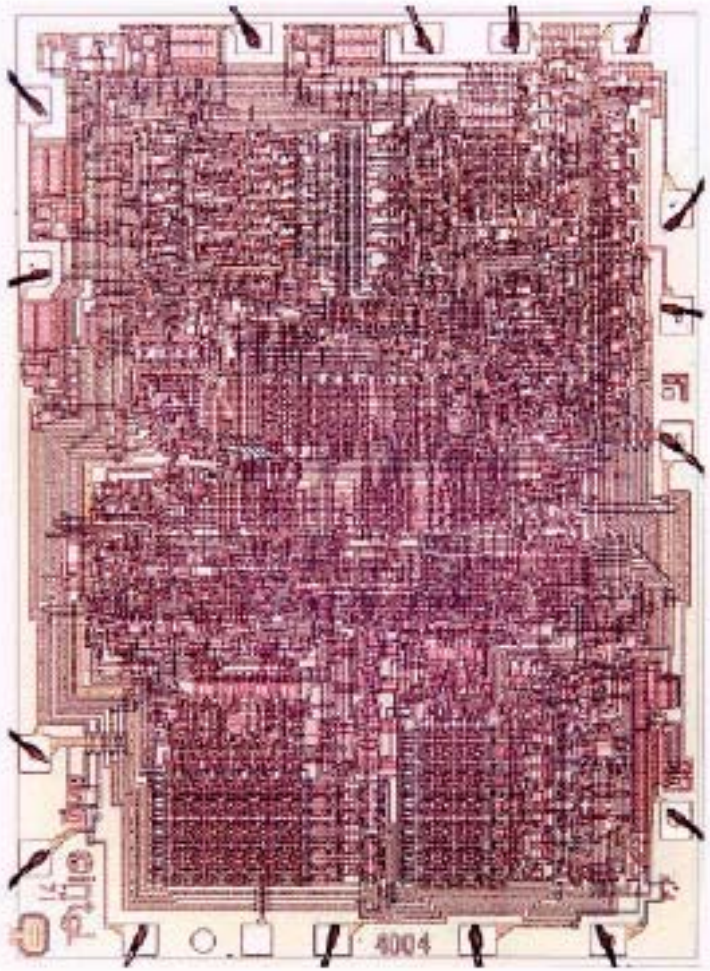
- **1969/70** Intel 4004, **first Microprocessor** (M.E.Hoff) 4 bit microprocessor, originally developed for Busicom, a small Japanese calculator company. Limited to 4096 memory location (of 4 bit data), 45 instructions; integrated 2300 PMOS transistors.
- **1971** Intel 8008, **first 8 bit microprocessor** (16K x 8bit)
- **1973** Intel 8080, 10 x faster than 8008, more memory (64k)
- **1974** Other 8 bit processors: Motorola 6800, Fairchild F8,
- **1975** Signetic 2650, MOS Technology 6502 (used in Apple II), Rockwell PPS-8
- **1976** National IMP-PACE, **first 16 bit microprocessor**, followed by Texas Instrument, TMS9900, **Zilog Z80** (8 bit) (used in Radio Shack TRS-80)
- **1977** Intel 8085 (8080 with built-in clock & system controller)
- **1978** Motorola 6809 (8 bit), **Intel 8086** (16 bit processor, 1M)
- **1978/79** **Intel 8088** - variant of 8086 with 8 external data pins
- **1981** IBM adopts Intel 8088 for IBM PC/XT

(see http://infopad.eecs.berkeley.edu/CIC/archive/cpu_history.html)

2 most popular microprocessor series:

- INTEL 8086, 80186, 80286, 80386, 80486, Pentium
- Motorola 68000, 68010, 68020, 68030, 68040

4004

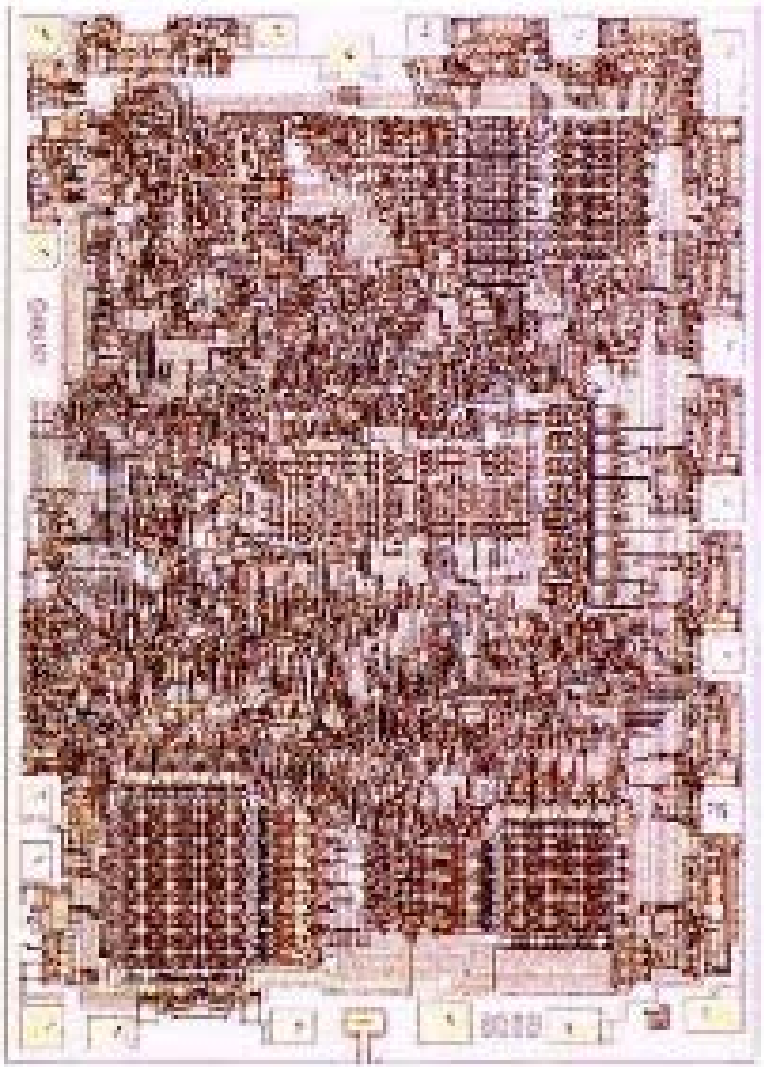


1971: 4004 Microprocessor

The 4004 was Intel's first microprocessor.

This breakthrough invention powered the Busicom calculator and paved the way for embedding intelligence in inanimate objects as well as the personal computer.

8008

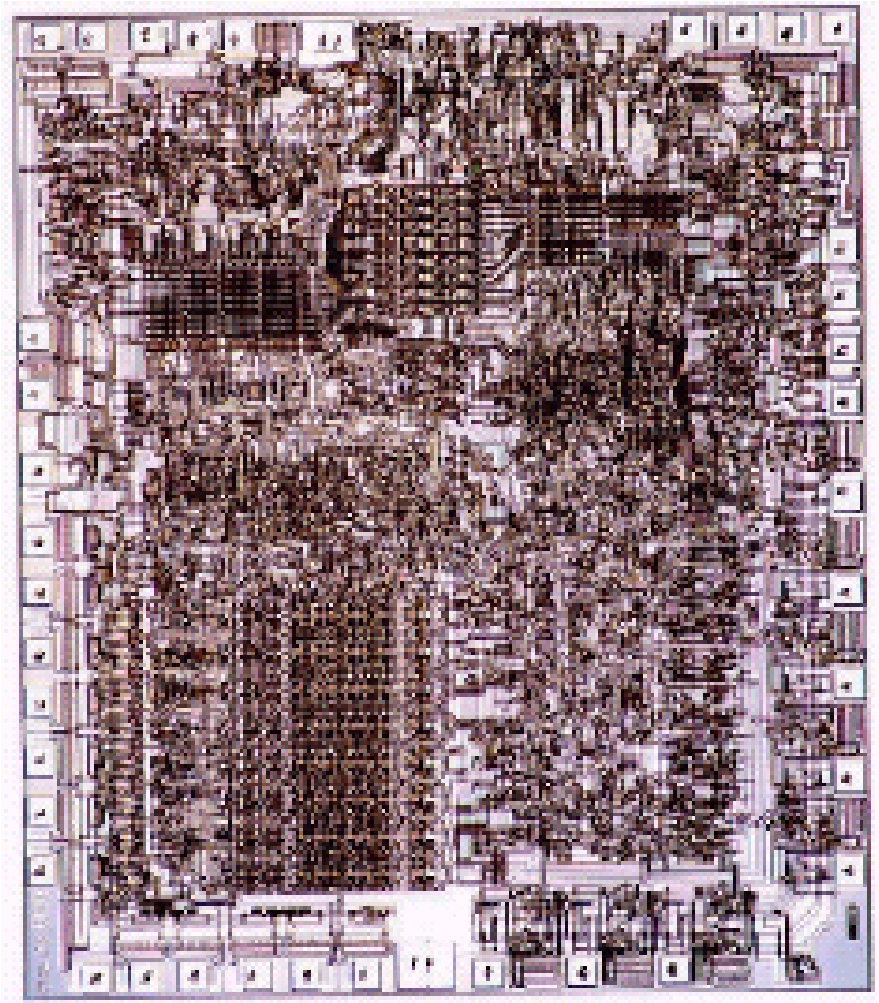


1972: 8008 Microprocessor

The 8008 was twice as powerful as the 4004.

According to the magazine *Radio Electronics*, Don Lancaster, a dedicated computer hobbyist, used the 8008 to create a predecessor to the first personal computer, a device *Radio Electronics* dubbed a "TV typewriter." It was used as a dumb terminal.

8080

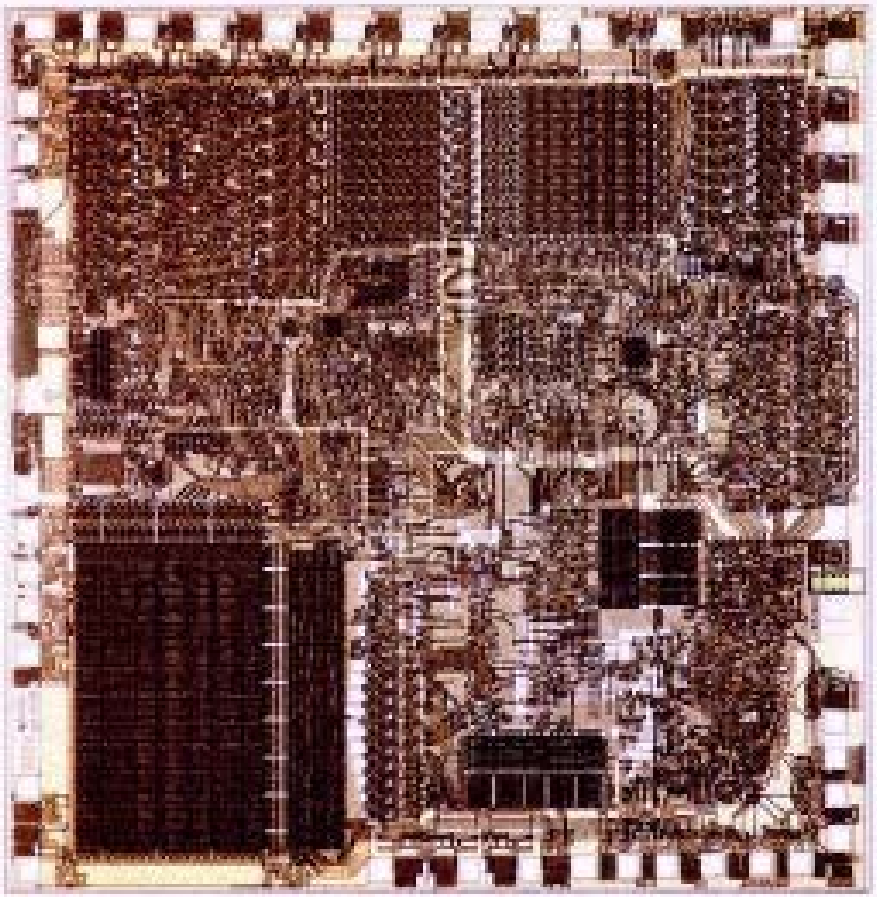


1974: 8080 Microprocessor

The 8080 became the brains of the first personal computer--the Altair, allegedly named for a destination of the Starship *Enterprise* from the *Star Trek* television show.

Computer hobbyists could purchase a kit for the Altair for $395. Within months, it sold tens of thousands, creating the first PC back orders in history.

8086-8088



1978: 8086-8088 Microprocessor

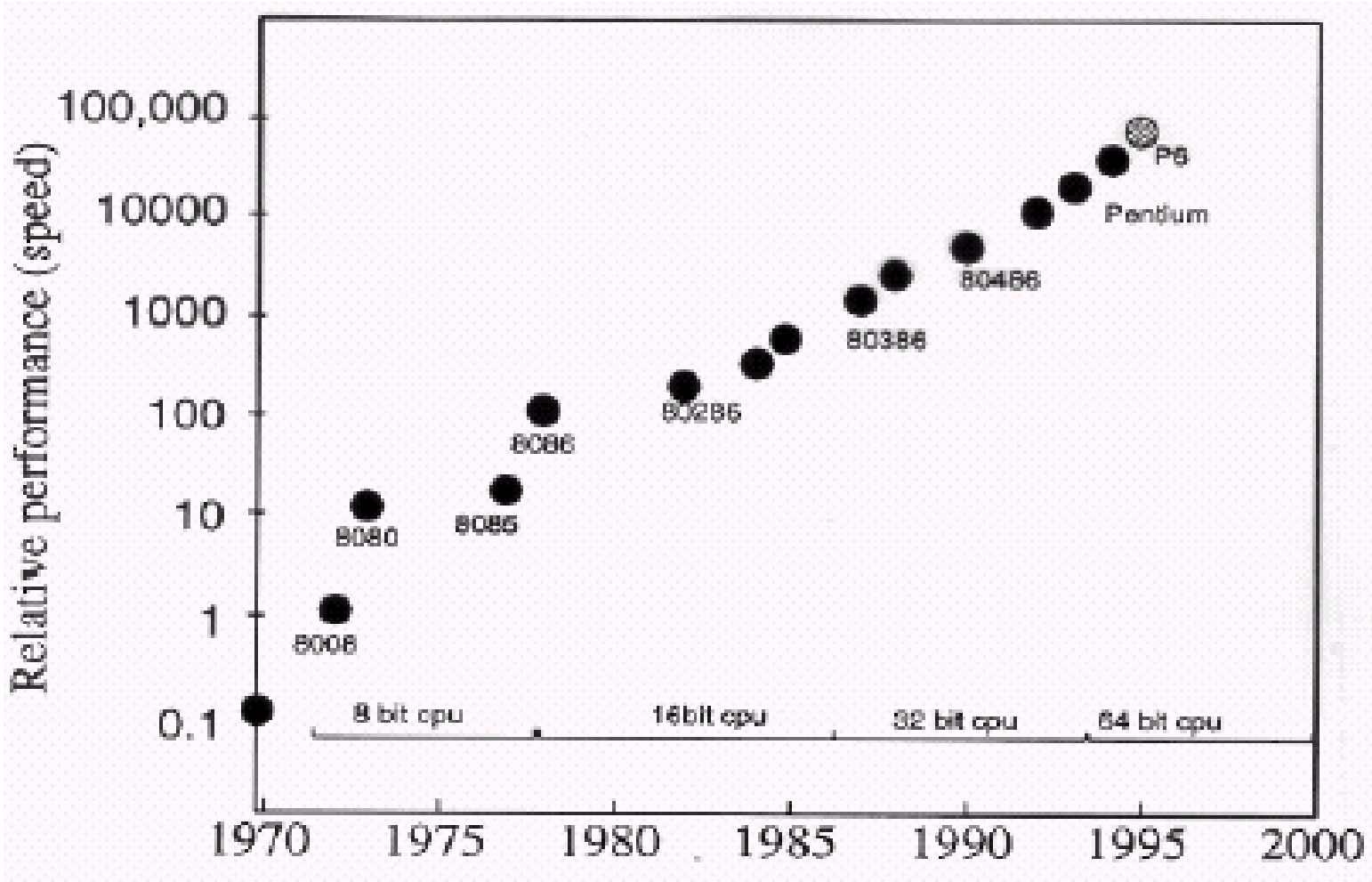
A pivotal sale to IBM's new personal computer division made the 8088 the brains of IBM's new hit product--the IBM PC.

The 8088's success propelled Intel into the ranks of the *Fortune 500*, and *Fortune* magazine named the company one of the "Business Triumphs of the Seventies."

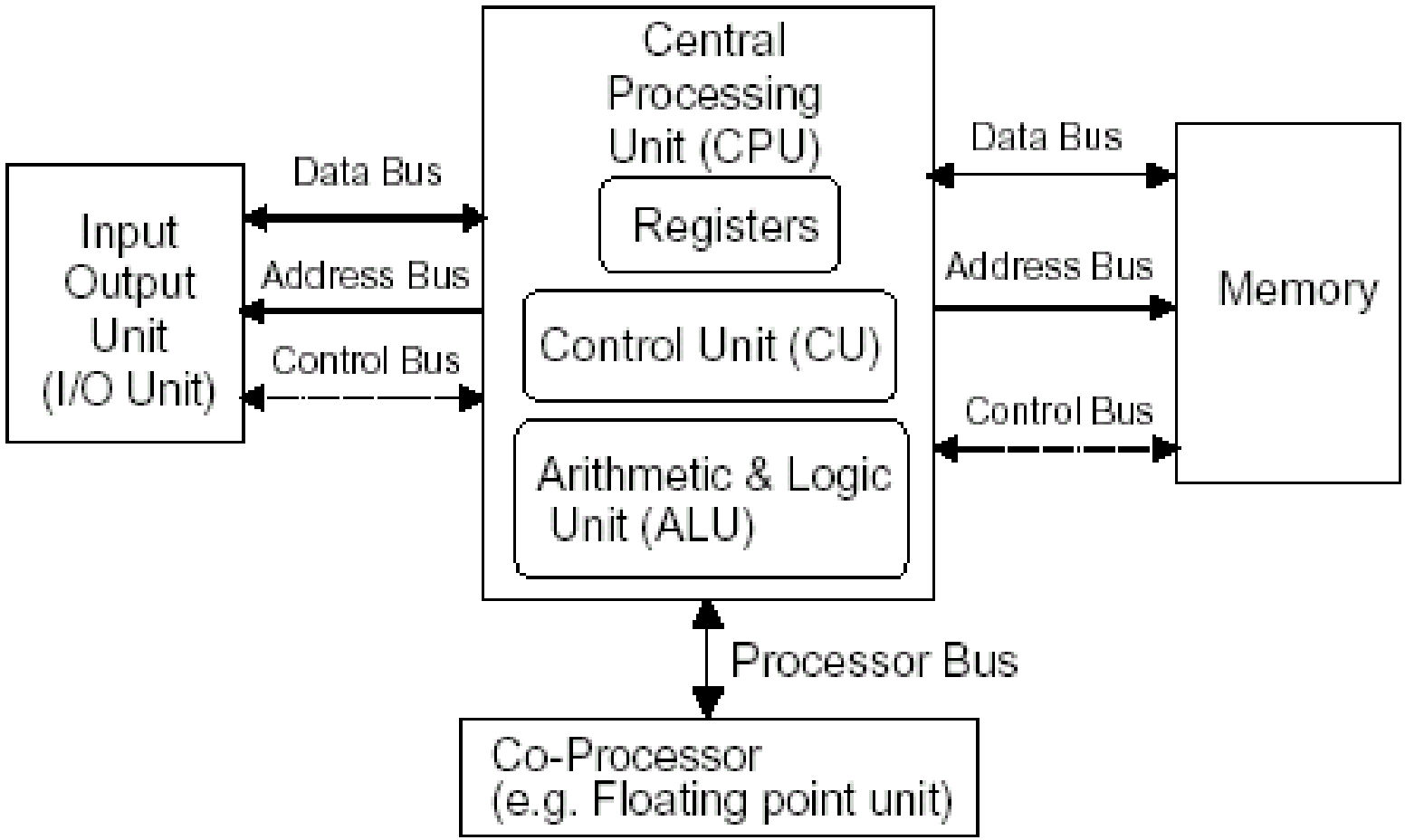
Evolution of Intel Microprocessors

Feature	8008	8080	8086	80396	80486	Pentium	P6	P-II	Celeron	PII Xeon
Year introduced	1972	1974	1978	1985	1989	1993	1995	1997	1998	1998
Number of instructions	66	111	133	154	235					
Address bus width	8	16	20	32	32	32	64	64	32	64
Data bus width	8	8	16	32	32	64	64	64	64	64
Number of flags	4	5	9	14	14					
Number of registers	8	8	16	8	8					
Memory addressability	16 KB	64 KB	1 MB	4 GB	4GB	4G	64G	64G	4G	64G
I/O ports	24	256	64 K	64 K	64K					
Bus bandwidth (MB/sec)	-	0.75	5	32	32					
Register-to-register add time (µsec)	-	1.3	0.3	0.125	0.06					

Intel Microprocessors Relative Speeds



Microprocessor Computer System



Microprocessor Computer System

- **Control Unit** (CU) generates all the control signals within the CPU. It initializes the registers on power-up, generates the signal to fetch instructions for the ALU.

  The Control unit may be implemented completely by hardware (hard-wired controller e.g. using a state counter and a Programmable Logic Array) or a mixture of software instructions (microcode stored in CPU) and hardware (microprogrammed control). Both the Intel 8086 family and Motorola 68000 family use microprogrammed controllers.

- **Registers** - small, fast memory which usually store data and addresses associated with the instruction being carried out.

- **ALU** performs arithmetic and logic operations

Instruction Execution Cycle

Two main steps in the cycle:

1. Fetch the next instruction from main memory
2. Decode and Execute the instruction

The Fetch cycle consists of

i) Use the instruction pointer (IP) to set the address bus with the address of the next instruction and increment the instruction pointer

ii) Wait (few hundred nanoseconds) for data to be transferred to the data bus from memory

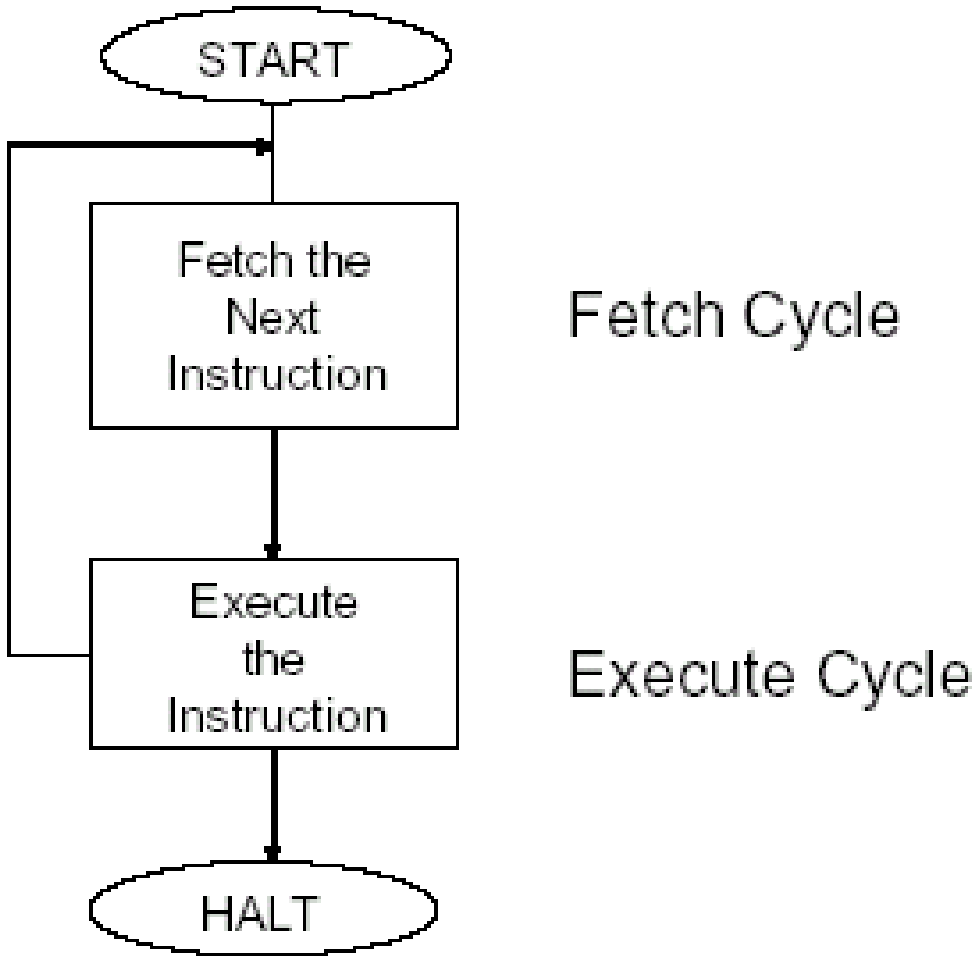
iii) Read the data from the data bus

The Execution Cycle consists of

i) Decode the instruction and generate the correct sequence of internal and external signals

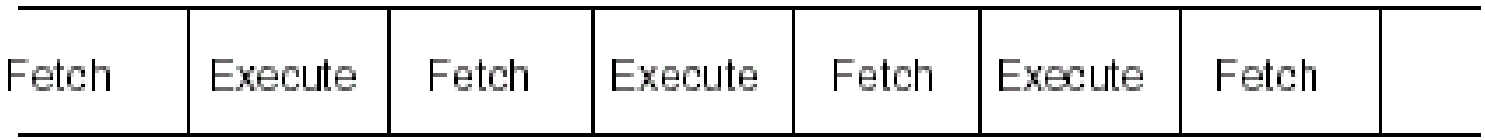
ii) Execute the instruction and restart the Fetch Cycle

Basic Instruction Cycle



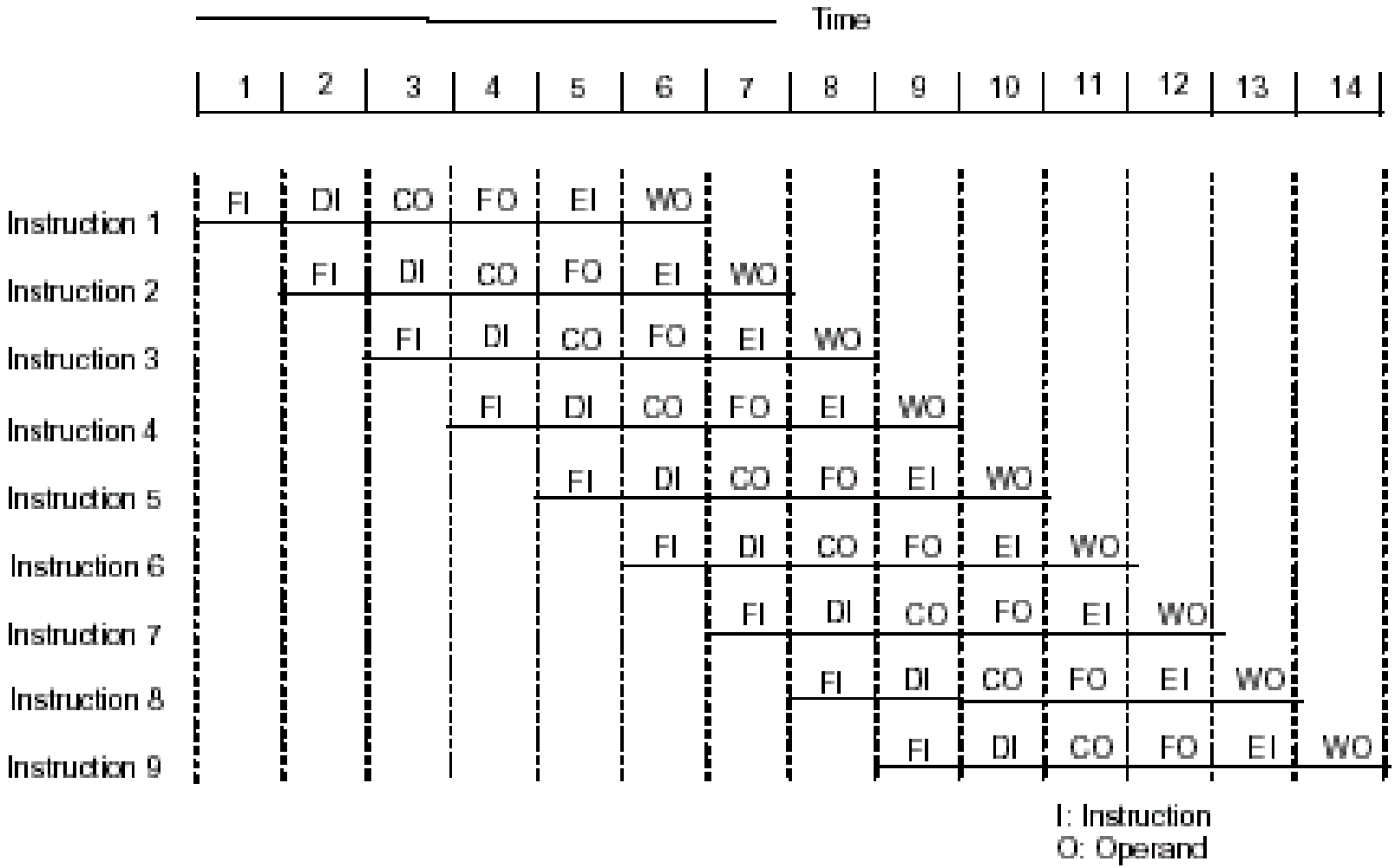
Pipelined Instruction Fetch and Execution Cycles

Instruction Fetch and Execution pipeline

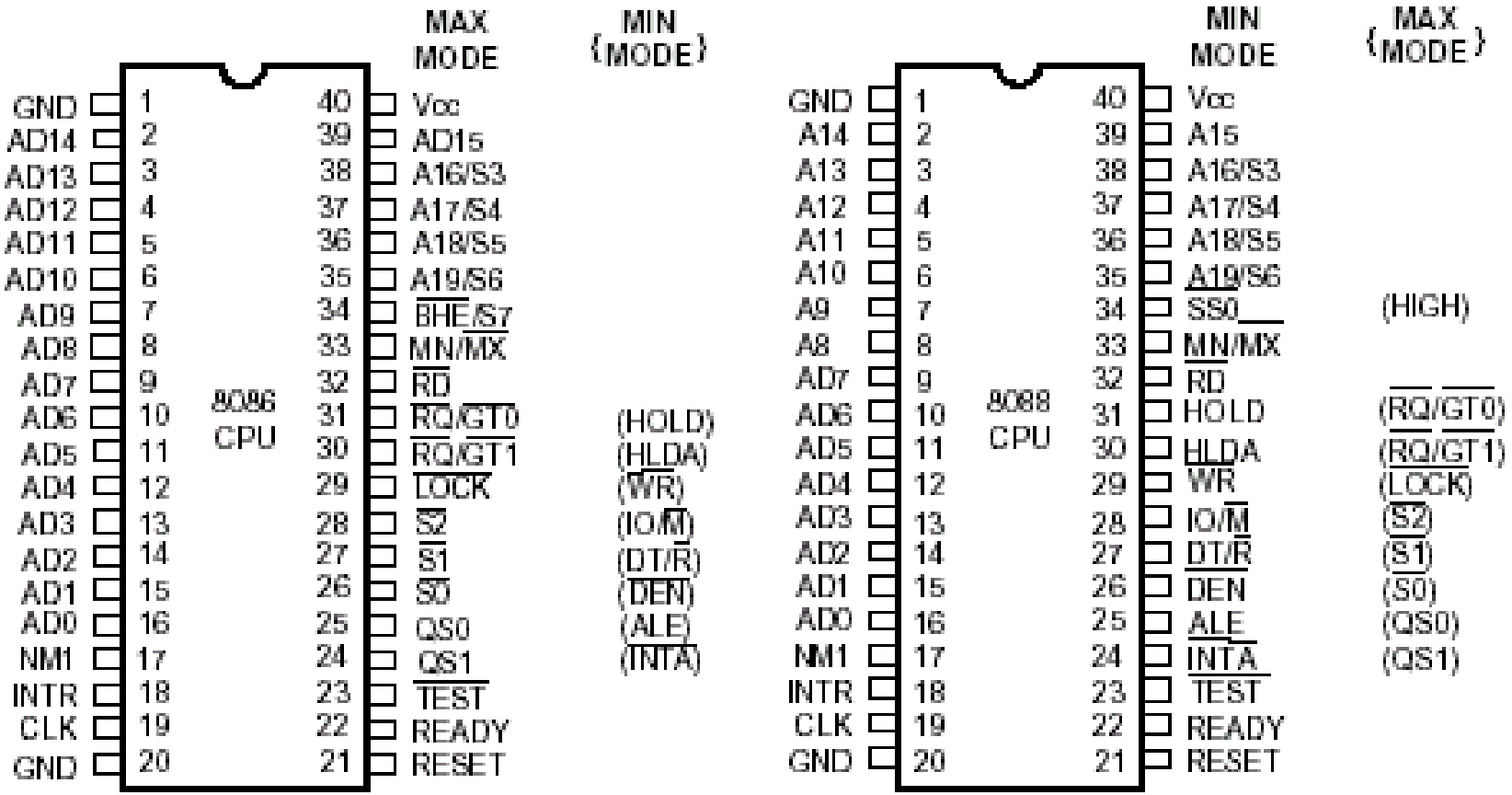


- **Bus Interface Unit (BIU)** fetches instructions from memory, passes the instruction to the instruction stream byte queue and starts to fetch the next instruction immediately
- **Execution Unit (EU)** removes instructions from the instruction queue
- Both BIU and EU may be working simultaneously in time (pipelined parallel processing)

Timing Diagram for Instruction Pipeline Operation



Introduction to Intel 8086/8088 Microprocessors



8086 pin diagram

8088 pin diagram

Introduction to Intel 8086/8088 Microprocessors

8088 and 8086 are almost identical except that 8088 has only 8 external data lines whereas the 8086 has 16 external data lines.

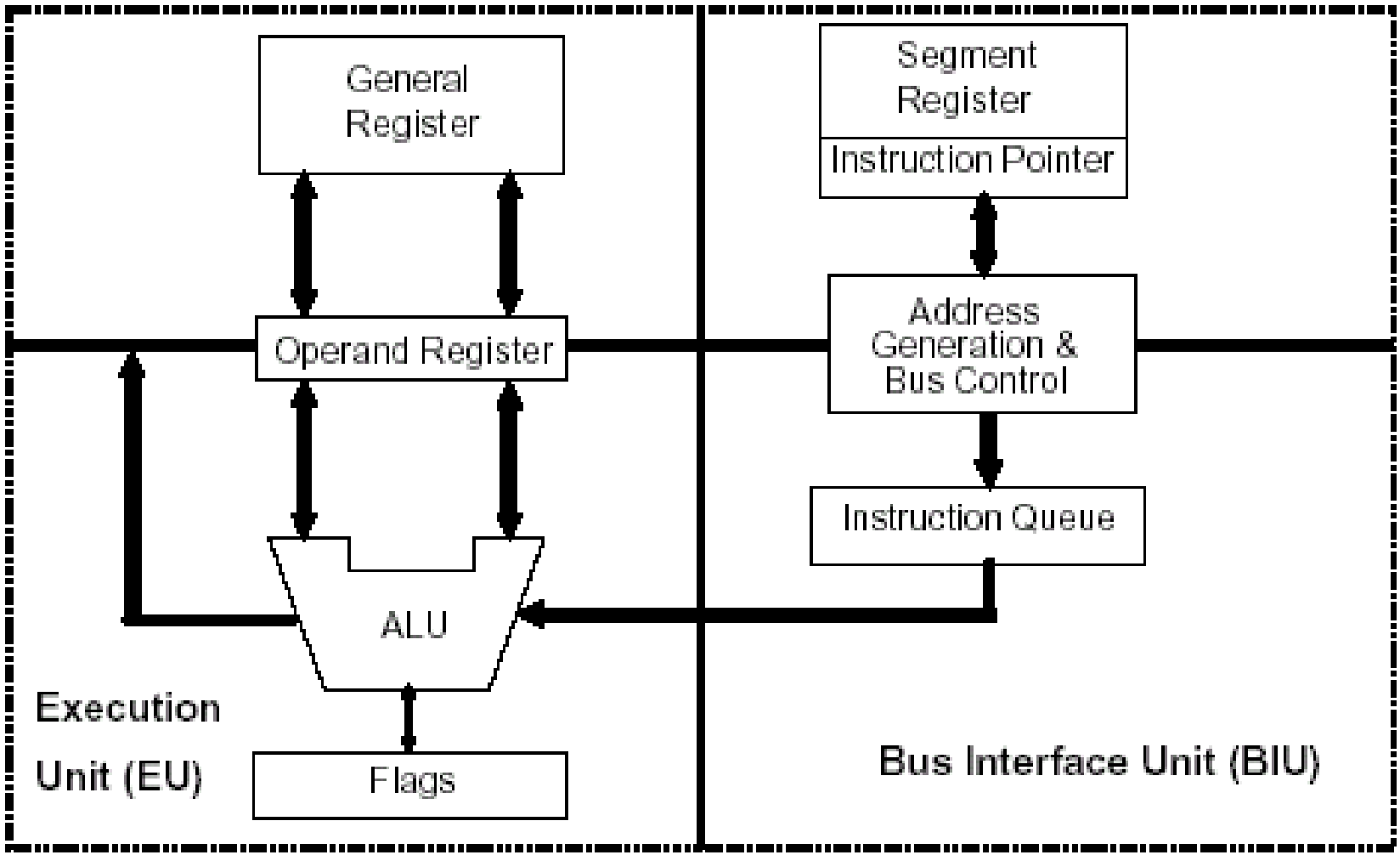
Both have

- 16 bit wide data bus internally
- 20 address pins (16 address/data + 4 address/status) allowing a maximum memory address range of 1MByte
- multiplexed address/data pins (8088 only multiplexes 8 pins)
- 2 modes of operation (maximum and minimum mode)
- Same instruction set

Internal Architecture of the 8088

- Both 8088/8086 employ *parallel processing.*
- Contains two processing units: *Execution unit* (EU) and *Bus interface unit* (BIU); operate at the same time.
- The BIU sends out addresses, fetches instructions from memory, reads data from ports and memory, and writes data to ports and memory, i.e. *the BIU handles all transfers of data and addresses on the buses for the execution unit*
- The EU tells BIU where to fetch instruction or data from, decodes instructions, and executes instructions.

Internal Architecture of the 8088



8086 Internal Block Diagram

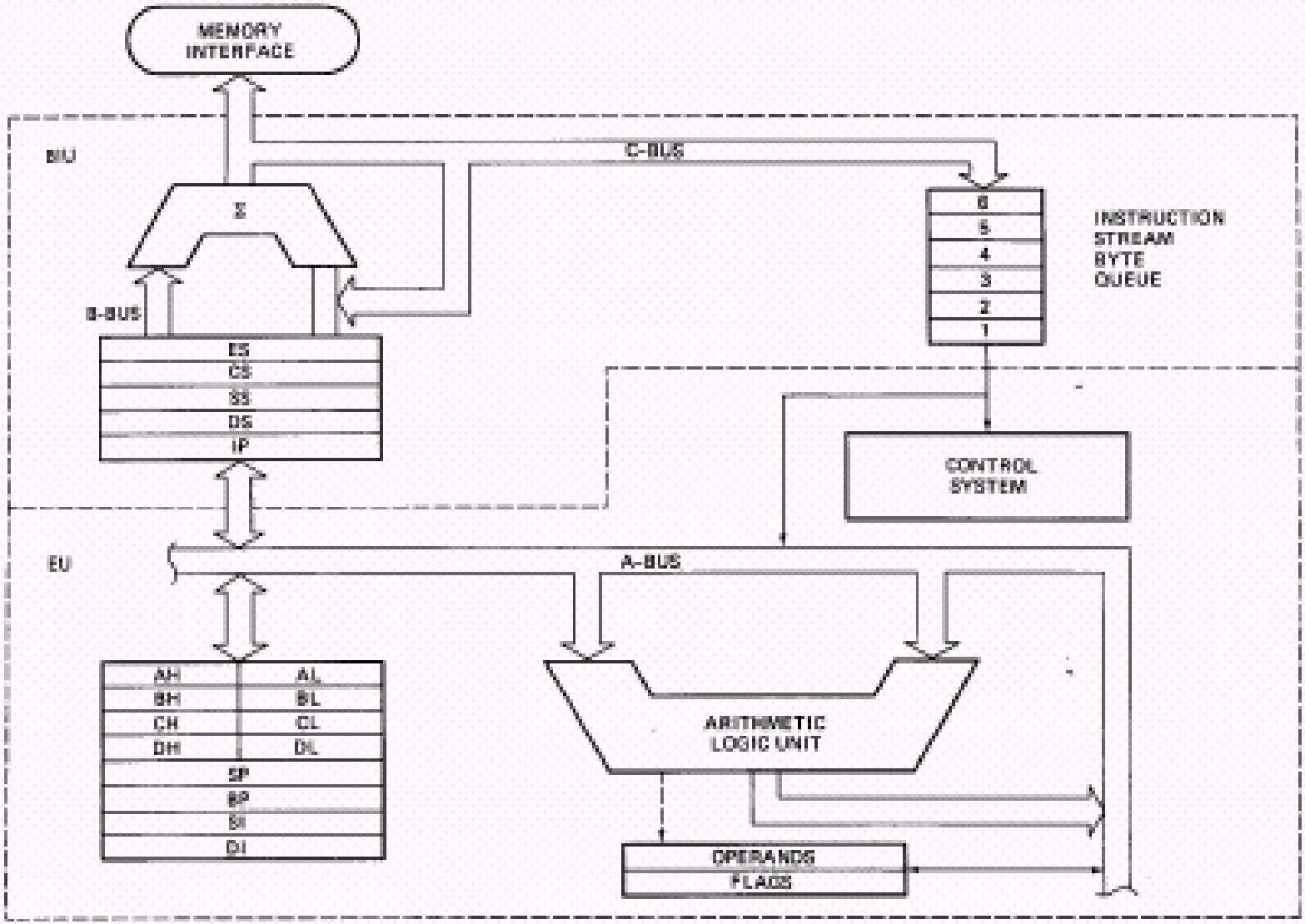


FIGURE 2-7 8086 internal block diagram. (*Intel Corp.*)

Bus Interface Unit (BIU)

- Perform bus operation such as instruction fetching, reading/writing of data operand for memory, inputting/outputting data for I/O peripherals.
- Perform other functions such as instruction queuing and data acquisitions.
- 8-bit (16-bit) bi-directional data bus for 8088 (8086).
- 20-bit address bus → can address any one of the 2^{20} (1,048,576) bytes.
- Contain *segment register*, *instruction pointer*, *address generation adder*, *bus control logic*, and an *instruction queue*.
- Use *instruction queue* to implement a *pipelined architecture* (prefetch up to 4 (6) bytes of instruction code for 8088 (8086) and then store and access the codes in FIFO order).

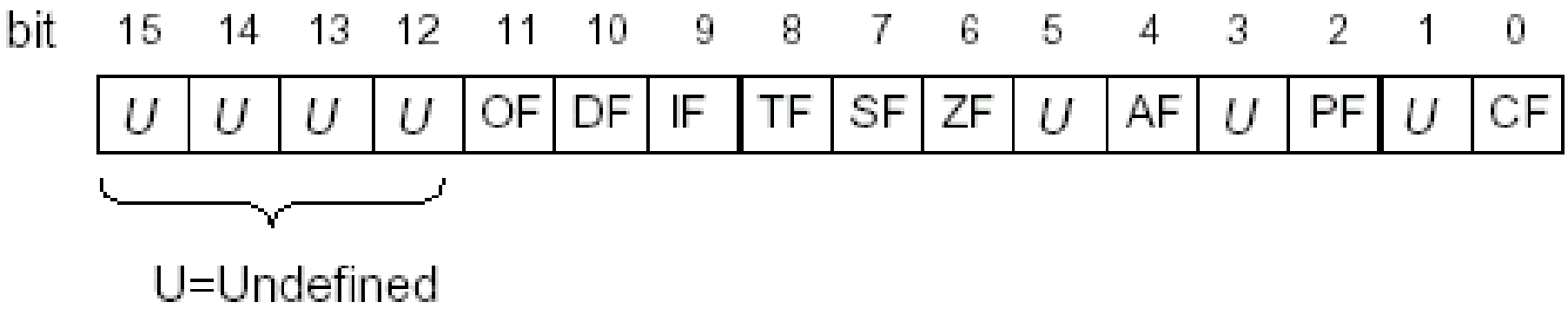
Execution Unit (EU)

- Responsible for decoding and executing instruction.
- Contains: *arithmetic logic* unit (ALU), status and control flags, general purpose registers, and temporary-operand register.
- EU access the instruction from output end of the instruction queue and data from general-purpose register.
- It reads one instruction at a time, decodes them, generates operand address if necessary, passes them to BIU and request to perform the read/write cycle to memory or I/O, and performs the operation specified by the instruction on operand.
- During execution, EU may test the status and control flags and update these flags based on the results of execution.

Flag Registers

- The flags indicate the condition of the microprocessor as well as controlling its operations.
- A flag register is a flip-flop which indicates some conditions produced by the execution of an instruction or controls certain operations of the EU. A 16-bit flag register in the EU contains nine active flags.
- *conditional flags*: Six flags are *conditional flags*. They are set or reset by the EU on the basis of the results of some arithmetic operation.
- *control flags* : The three remaining flags in the flags register are used to control certain operations of processor. They are called the *control flags*.

Flag Registers



Carry Flag (CF)- set by carry out of MSB.
Parity Flag (PF)- set if result has even parity.
Auxiliary carry Flag (AF)- for BCD
Zero Flag (ZF)- set if results = 0
Sign Flag (SF) = MSB of result
TF- single step trap flag
IF- interrupt enable flag
DF- string direction flag
Overflow Flag (OF)- overflow flag

Conditional Flags

- **carry flag (CF)-** indicates a carry after addition or a borrow after subtraction, also indicates error conditions.
- **parity flag (PF)-** is a logic "0" for odd parity and a logic "1" for even parity.
- **auxiliary carry flag (AF)-** important for BCD addition and subtraction; holds a carry (borrow) after addition (subtraction) between bits position 3 and 4. Only used for DAA and DAS instructions to adjust the value of AL after a BCD addition (subtraction).
- **zero flag (ZF)-** indicates that the result of an arithmetic or logic operation is zero.
- **sign flag (SF)-** indicates arithmetic sign of the result after an arithmetic operation.
- **overflow flag (OF)-** a condition that occurs when signed numbers are added or subtracted. An overflow indicates that the result has exceeded the capacity of the machine.

Control Flags

- The control flags are deliberately set or reset with specific instructions YOU put in your program. The three control flags are:
  - **trap flag (TF)** - used for single stepping through a program;
  - **interrupt flag (IF)** - used to allow or prohibit the interruption of a program;
  - **direction flag (DF)** - used with string instructions.

- No specific instruction to set TF. See example 11-1 in Brey's for more details.

General-Purpose Registers

- EU has eight 8-bit *general-purpose registers*, labeled AH, AL, BH, BL, CH, CL, DH, DL. These registers can be used individually for temporary storage of 8-bit data.
- Register pairs AH-AL, BH-BL, CH-CL, and DH-DL can be used together to form register AX, BX, CX, and DX and can be used to store 16-bit data words.
- The AL register is also called the *accumulator*. It has some features that the other general-purpose registers do not have.
- The advantage of using internal registers is that it can be accessed more quickly than from external memory. No memory reference or memory cycle is needed to get the data.

Block Diagram of a Simple 8086-based Microcomputer