MICROCOMPUTER INPUT/OUTPUT

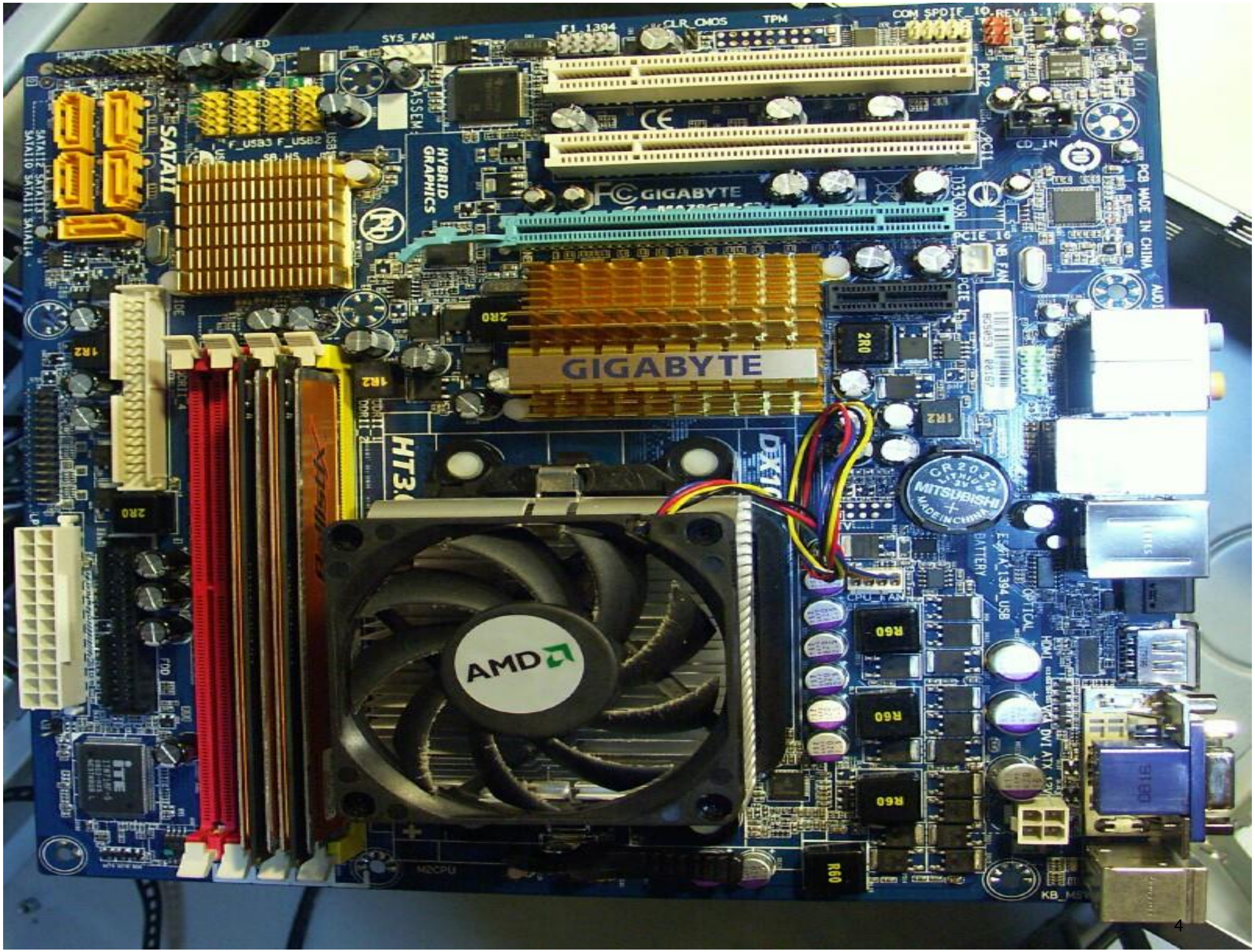
Microprocessors and microcomputer-based system design
Mohamad Rafiquzzaman
>Chapter 1 section 1.3.4

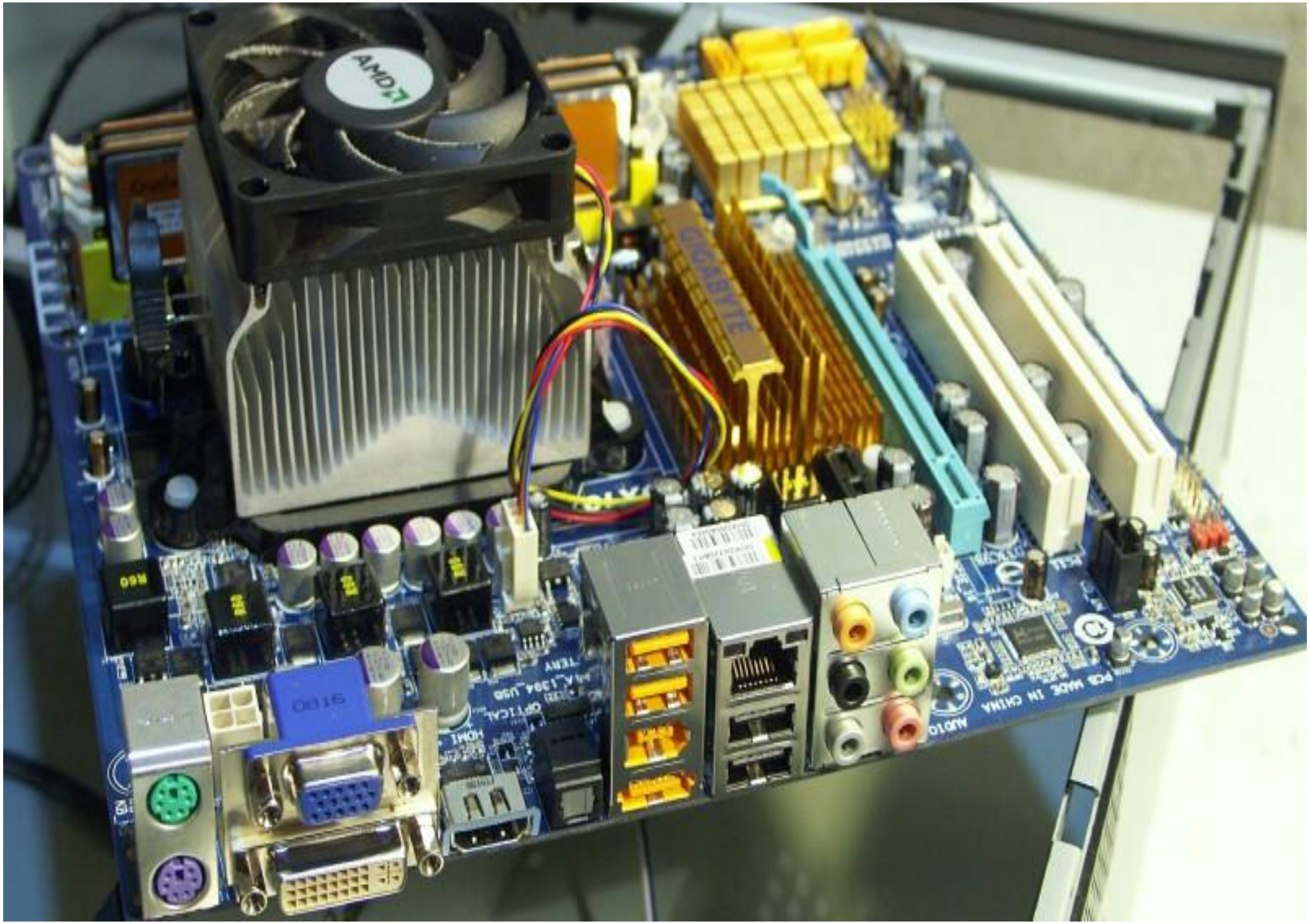
Outline

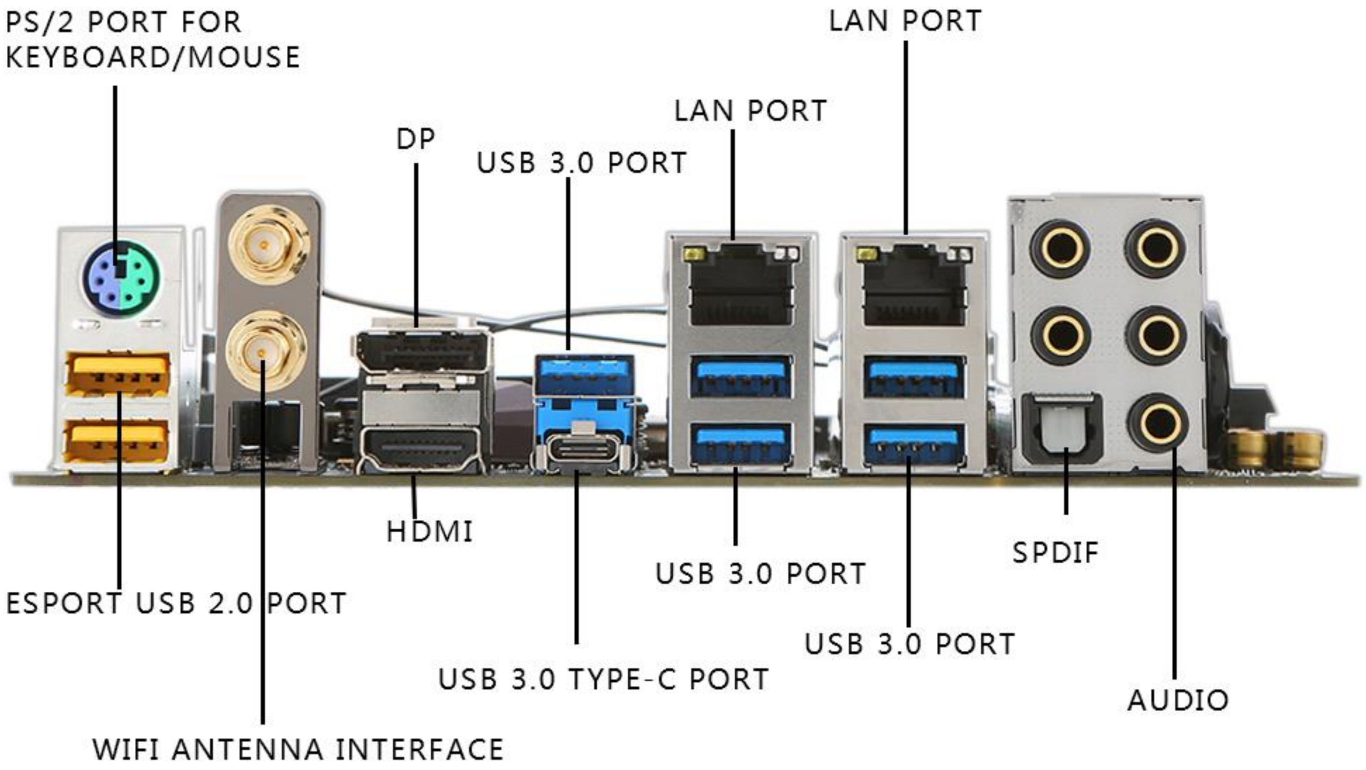
Introduction

- The technique of data transfer between a microcomputer and an external device is called *input/output (I/O).*

- ***Peripherals*** *are* the I/O devices that connected to a microcomputer and provide an efficient means of communication between the microcomputer and the outside world.


GIGABYTE
AMD


AMD
GIGABYTE
HDMI
OPTICAL
AUDIO
PCB MADE IN CHINA


PS/2 PORT FOR KEYBOARD/MOUSE
LAN PORT
LAN PORT
DP
USB 3.0 PORT
HDMI
USB 3.0 PORT
SPDIF
ESPORT USB 2.0 PORT
USB 3.0 PORT
USB 3.0 TYPE-C PORT
AUDIO
WIFI ANTENNA INTERFACE

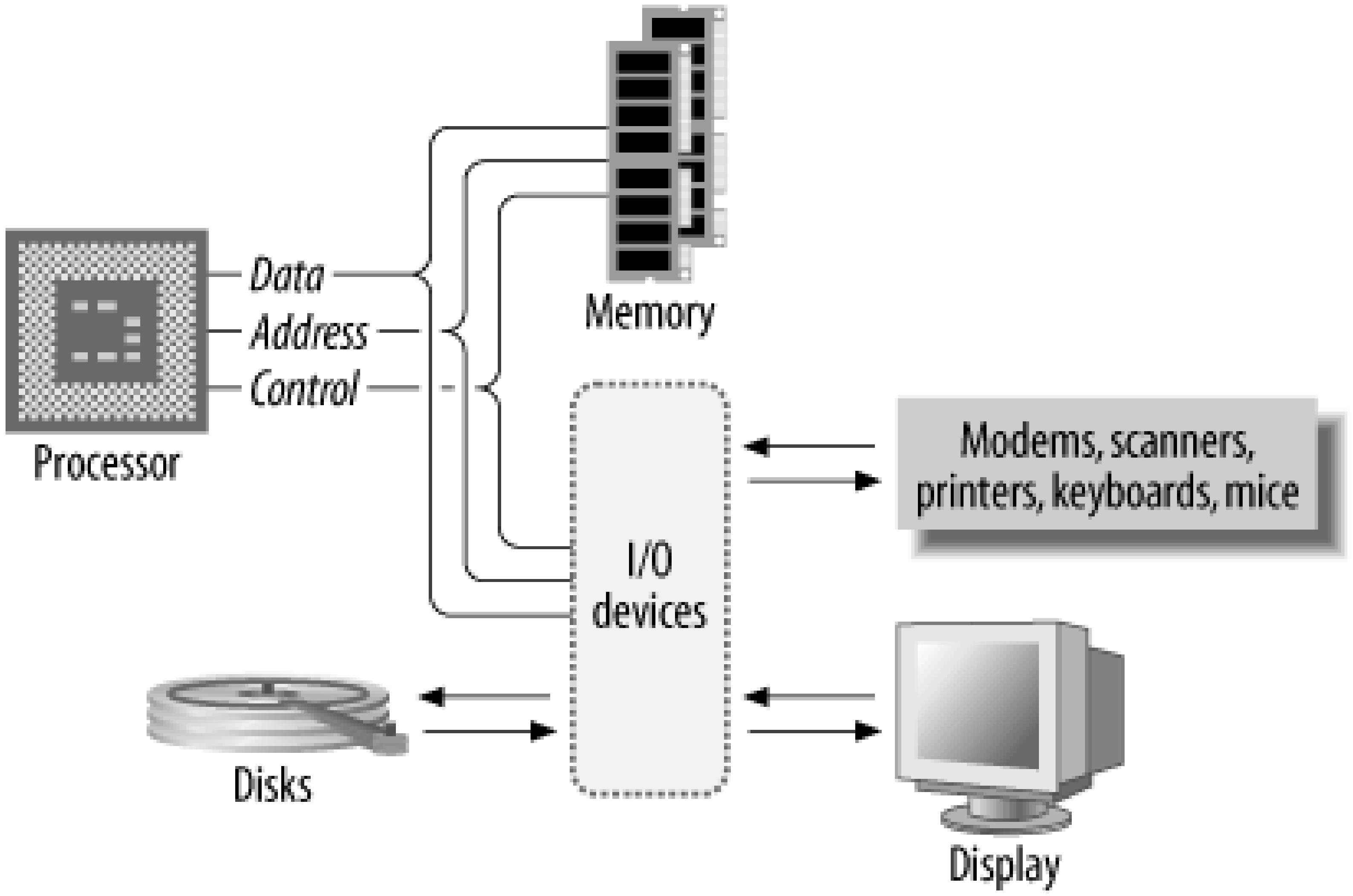
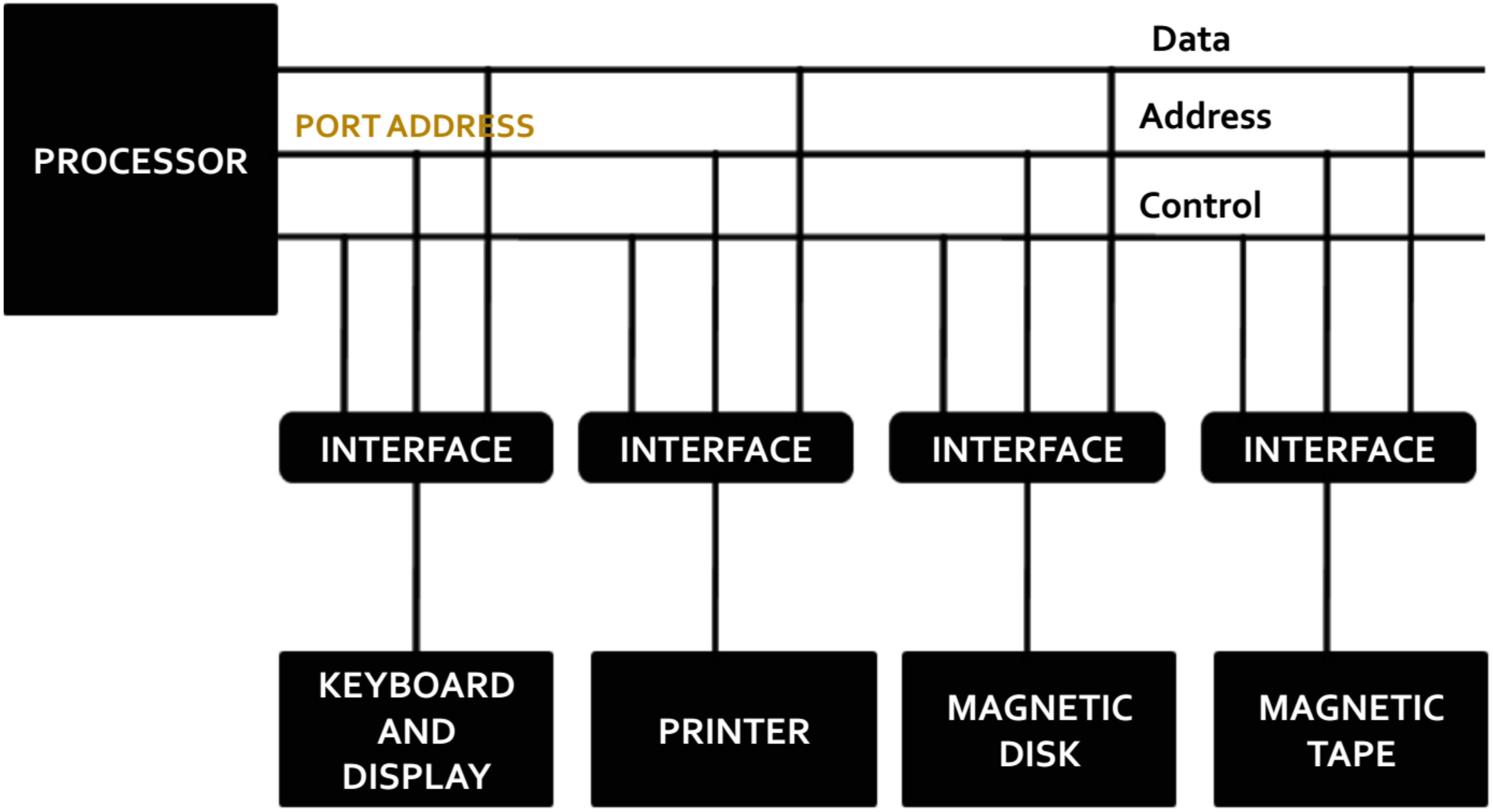


Figure. Basic computer system


PROCESSOR
PORT ADDRESS
Data
Address
Control
INTERFACE
INTERFACE
INTERFACE
INTERFACE
KEYBOARD AND DISPLAY
PRINTER
MAGNETIC DISK
MAGNETIC TAPE

Introduction

- The characteristics of I/O devices are normally different from those of a microcomputer like (speed and word length)

- we need interface hardware circuitry between the microcomputer and I/O devices.

- Interface hardware provide all input and output transfers between the microcomputer and peripherals.

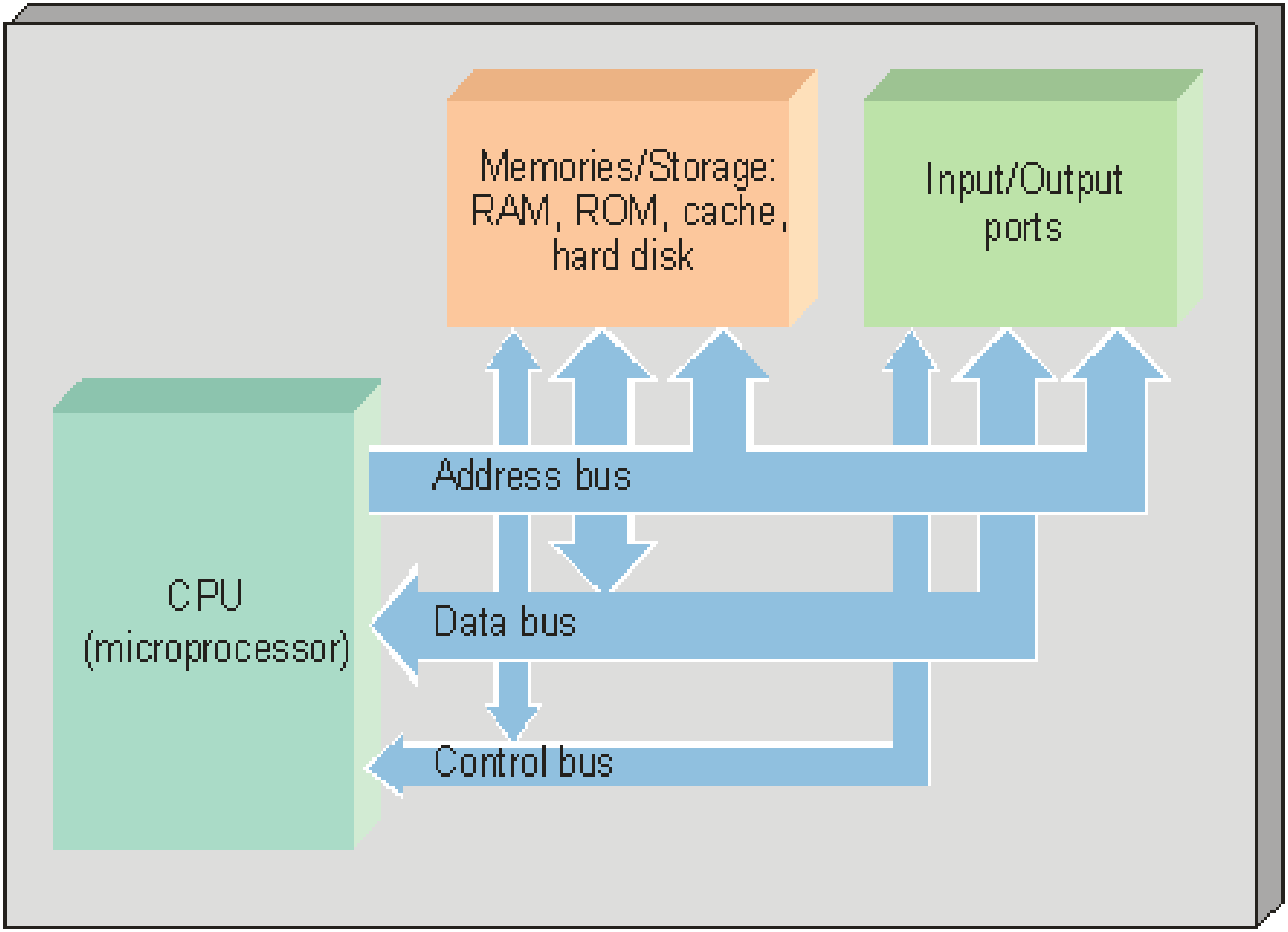
Introduction

There are three ways of transferring data between a microcomputer and physical I/O devices:

A. Programmed I/O,

B. Interrupt driven I/O and

C. Direct memory access.


Memories/Storage: RAM, ROM, cache, hard disk
Input/Output ports
Address bus
CPU (microprocessor)
Data bus
Control bus

Introduction

There are **three major types of data transfer** between the microprocessor and I/O device.

- Programmed I/O : In programmed I/O the **data transfer is accomplished through an I/O port** and controlled by software.
- Interrupt driven I/O : In interrupt driven I/O, the **I/O device will interrupt the processor,** and initiate data transfer.
- Direct memory access (DMA) : In DMA, the data transfer between memory and I/O can be **performed by bypassing the microprocessor.**

A. Programmed I/O

- *Programmed* I/O, the microprocessor executes a program to perform all data transfers between the microcomputer and the external device.

- The main characteristic of this type of I/O technique is that the external device carries out the functions dictated by the program inside the microcomputer memory.

I/O Port (1)

A microcomputer communicates with an external device via one or more registers called **I/O *ports*** *using* ***programmed I/O*****.** I/O ports are usually of two types.

i. For one type, each bit in the port can be configured individually as either input or output.

ii. For the other type, all bits in the port can be set up as all parallel input or parallel output bits.

I/O Port (2)

- I/O port is made up of group of 8 pins.
- Each pin can be configured as either input pin or output pin.
- If a pin is input pin, it accepts data from the device it is connected to.
- If a pin is output pin, it sends the data to the device it is connected to.

I/O Port (3)

Each I/O port has at least two special-function registers:

a. A control register called a data-direction register (such as DDRA or DDRB) that controls whether the port's pins are configured as inputs or as outputs.

b. A data register (such as PORTA or PORTB) that holds data traveling in or out through the port.

I/O Port (4)

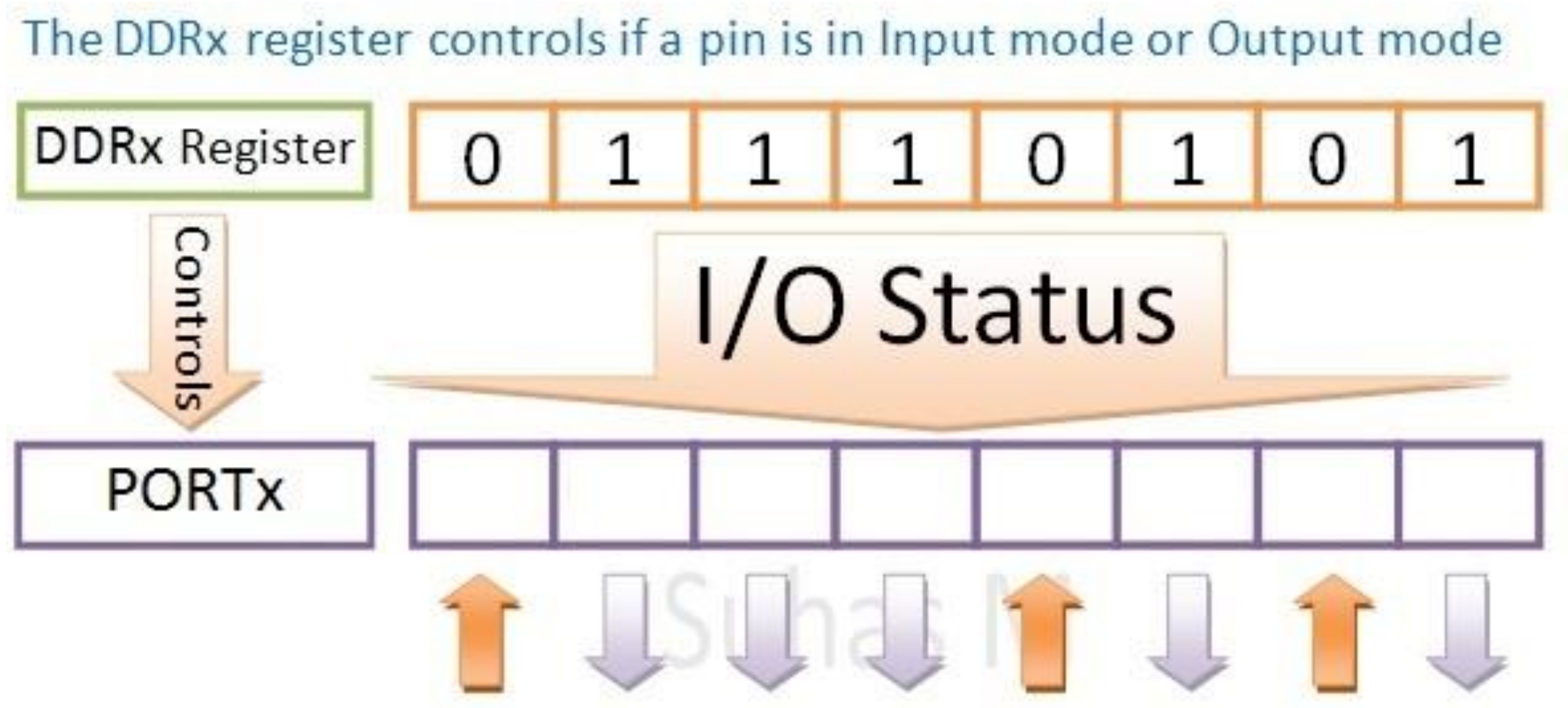
Command or data-direction register

- Determines the direction of individual pins of ports.

- If the bit of the DDR is **1** then the corresponding pin of the port is configured as **output pin**.

- If the bit of the DDR is **0** then the corresponding pin of the port is configured as **input pin**.

I/O Port (4)


The DDRx register controls if a pin is in Input mode or Output mode
DDRx Register
0
1
1
1
0
1
0
1
Controls
I/O Status
PORTx

A. Programmed I/O

As an example, if an 8-bit data-direction register contains 34H (34 Hex), then which pin's of the Port is set as Input and output.

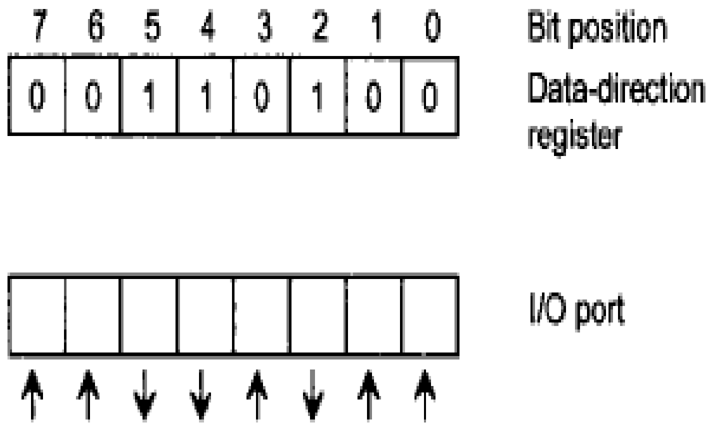


FIGURE 4.4 **Bit configurable I/O port along with a data-direction register.**

A. Programmed I/O

For parallel I/O, there is only one data directional register, known as the command register for all ports. A particular bit in the common register configures all bits in the port as either inputs or outputs. Programmable Peripheral Interface uses parallel I/O for configuration.

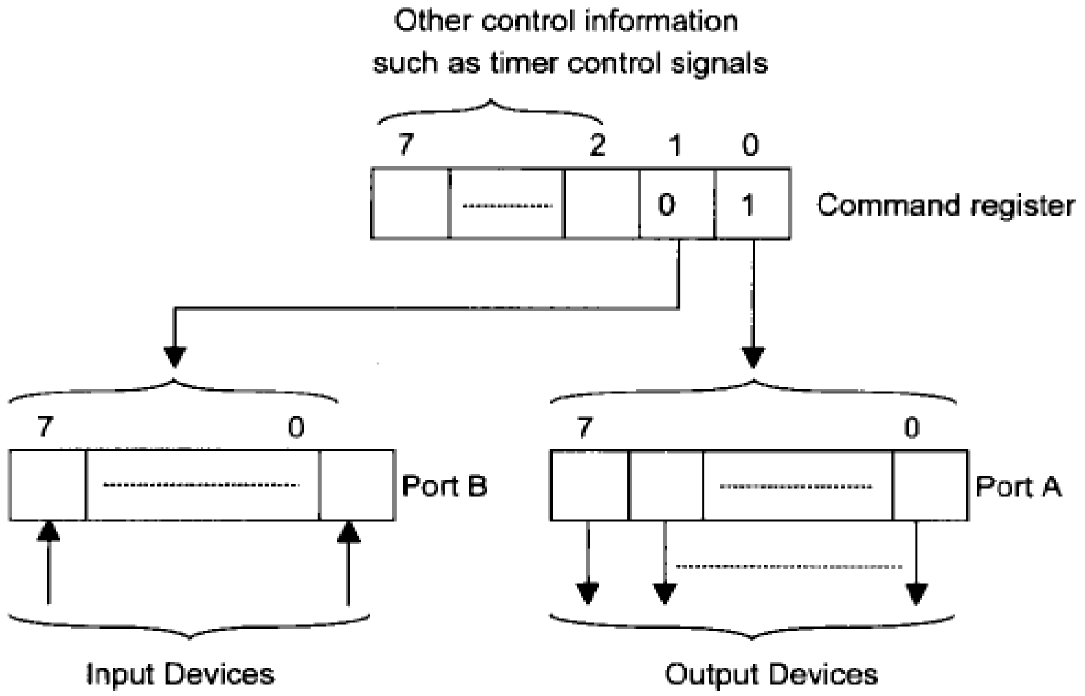


FIGURE 4.5 **Parallel I/O ports.**

A. Programmed I/O

I/O ports are addressed using two techniques.

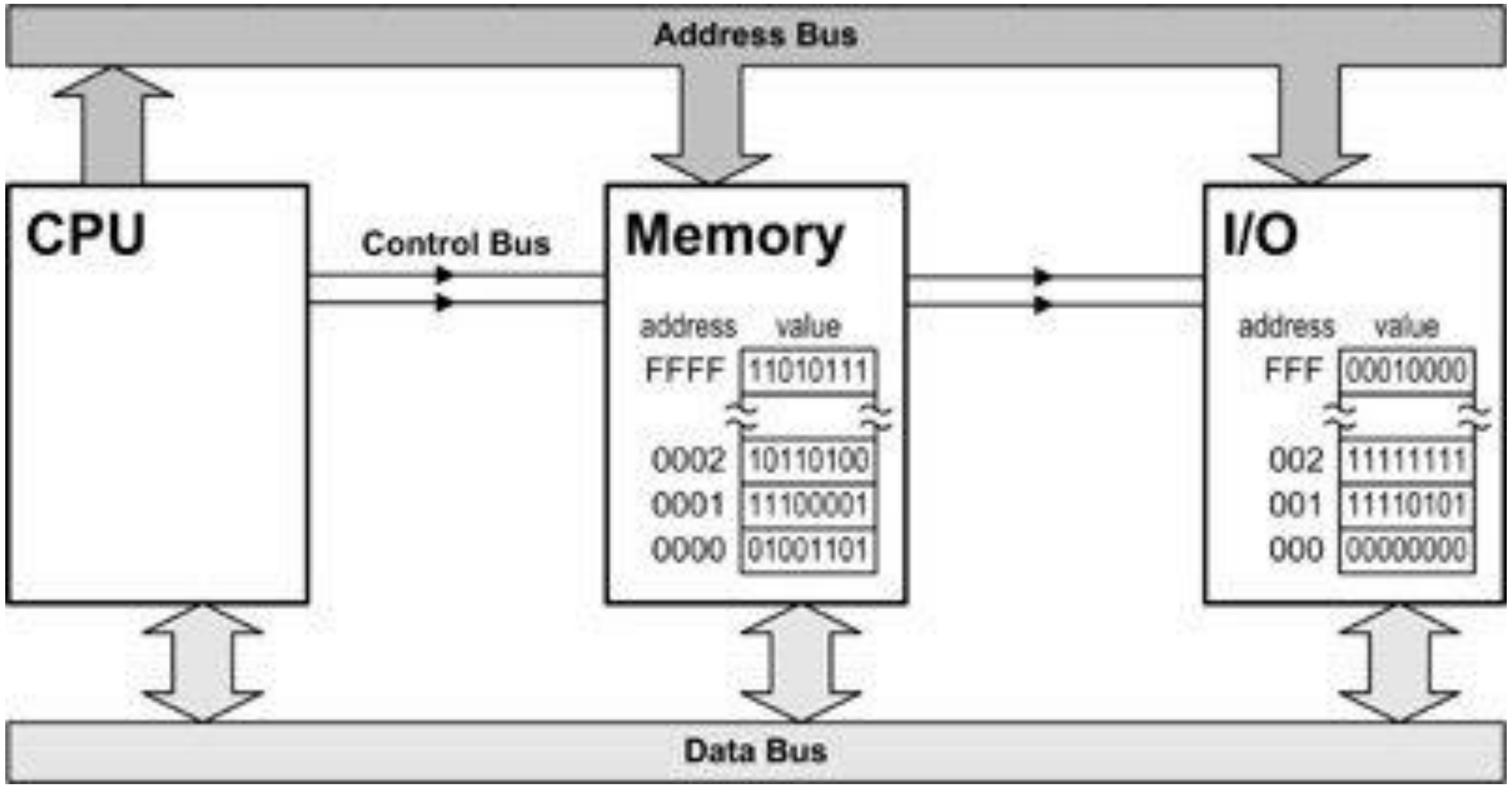
I. Standard I/O

II. Memory-mapped I/O

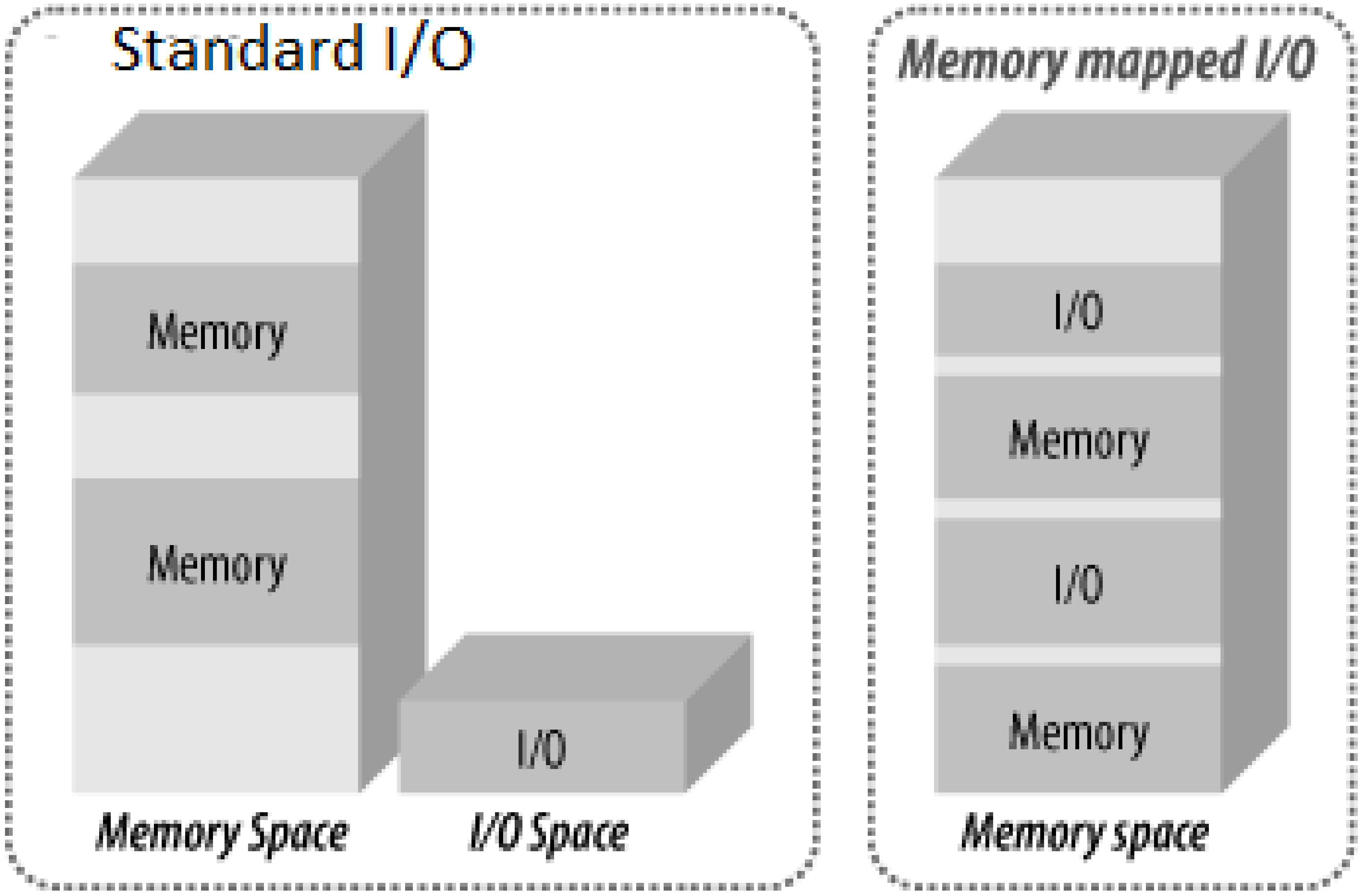
I. Standard I/O

Two separate address spaces are used – one for memory location and other for I/O devices.

- The I/O devices are provided dedicated address space.
- Hence, there are two separate control lines for memory and I/O transfer.
  a. I/O read and I/O write lines for I/O transfer
  b. Memory Write and Memory Read for memory transfer


Address Bus
CPU
Control Bus
Memory
address value
FFFF 11010111
0002 10110100
0001 11100001
0000 01001101
I/O
address value
FFF 00010000
002 11111111
001 11110101
000 00000000
Data Bus

Standard I/O


Standard I/O
Memory
Memory
I/O
Memory Space
I/O Space
Memory mapped I/O
I/O
Memory
I/O
Memory
Memory space

I. Standard I/O

Standard I/O or port I/O (called isolated I/O by Intel) uses an output control pin such as the M/ $\overline{IO}$ pin to distinguish between I/O and memory.

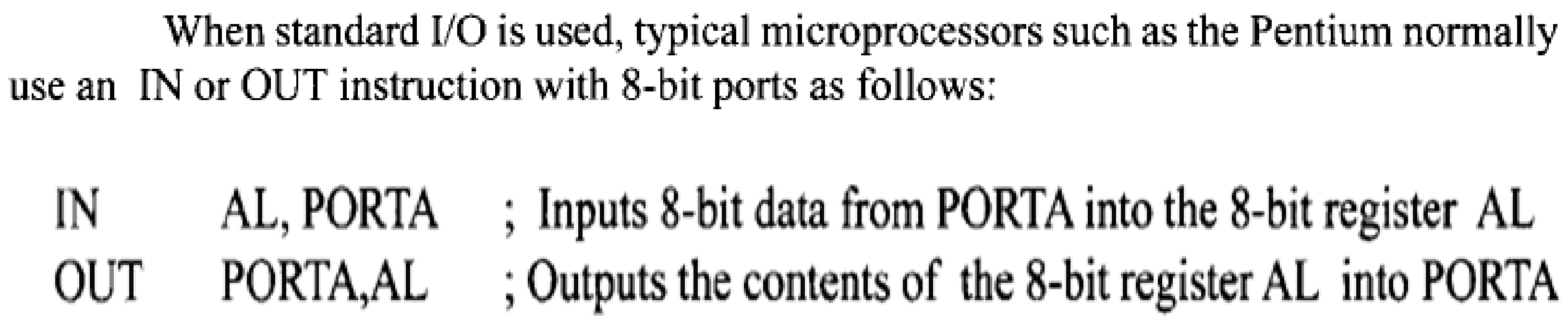
- M/ $\overline{IO}$ pin= High;

    The processor outputs a High to indicate that a memory operation is going on.

    - M/ $\overline{IO}$ pin = Low;

    The processor outputs a Low to indicate that a I/O operation is going on.

Standard I/O



When standard I/O is used, typical microprocessors such as the Pentium normally use an IN or OUT instruction with 8-bit ports as follows:

```
IN      AL, PORTA     ; Inputs 8-bit data from PORTA into the 8-bit register AL
OUT     PORTA,AL      ; Outputs the contents of the 8-bit register AL into PORTA
```

Memory oriented instruction

LDA 3000H (content of memory location 3000h is copied in accumulator)

STA 3000H (the content of accumulator is stored into the memory location 3000h)

Standard I/O

ADVANTAGES

- 1 MB memory address space is available for use with 8086 micro-pro. memory.
- Special Instructions for I/O operations maximize I/O performance.
- Used in system where complete memory capacity is required

DISADVANTAGES

- Data has to be transferred to the accumulator (any one of the internal register) to perform arithmetic and logic operation

II. Memory-mapped I/O

In memory mapped I/O scheme, CPU addresses an I/O device just like a memory location.

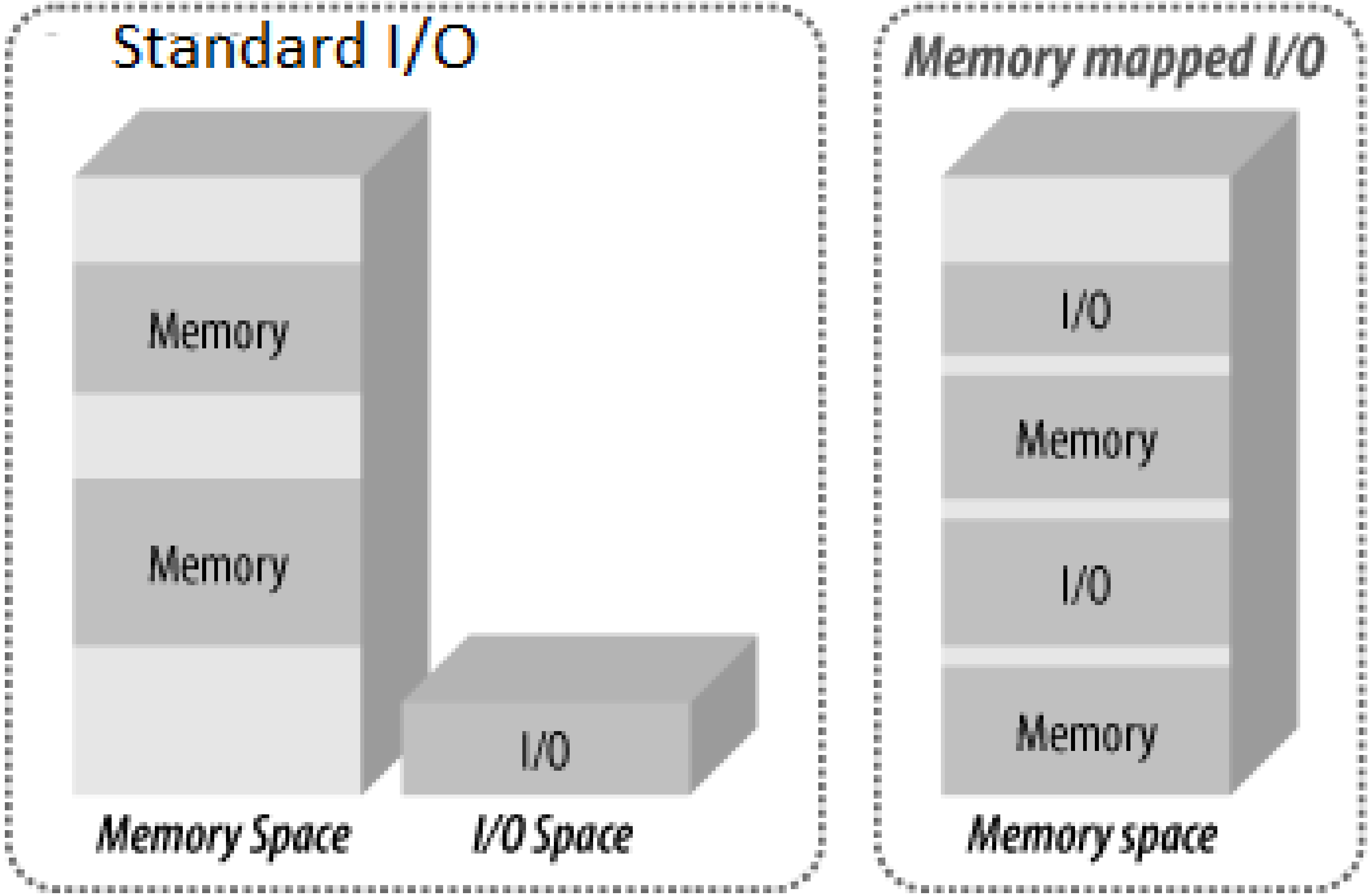
- In this scheme only single address space is used by CPU. Some addresses of the address space are assigned to memory location and other are assigned to I/O devices.
- The instructions used to manipulate the memory can be used for I/O devices.

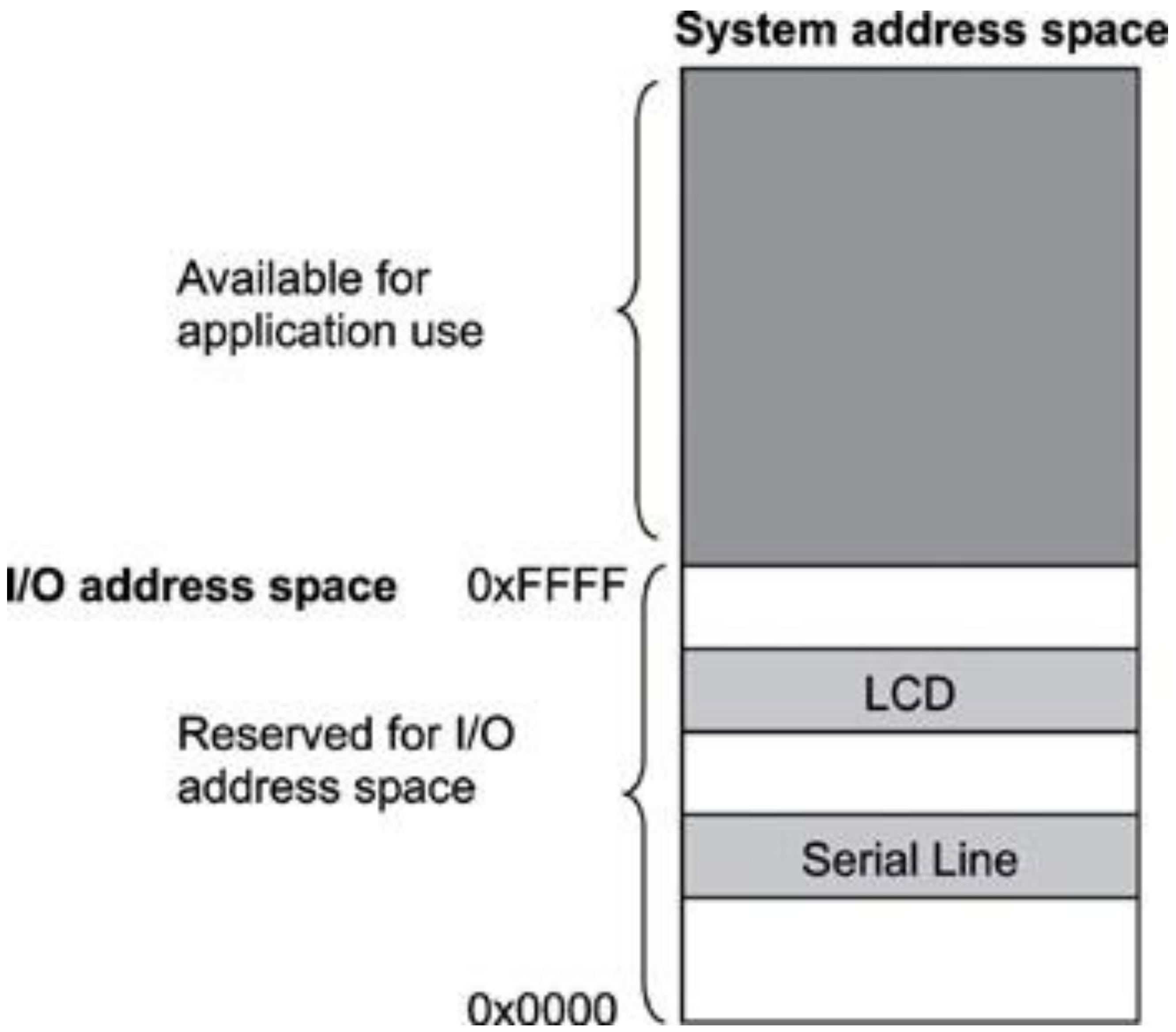
II. Memory-mapped I/O

- **In memory-mapped I/O**, the microprocessor does not use the M / $\overline{IO}$ control pin. Instead, the microprocessor uses an unused address pin to distinguish between memory and I/O.
- In memory mapped I/O, the most significant bit (msb) of the address can be used to distinguished between I/O and memory.

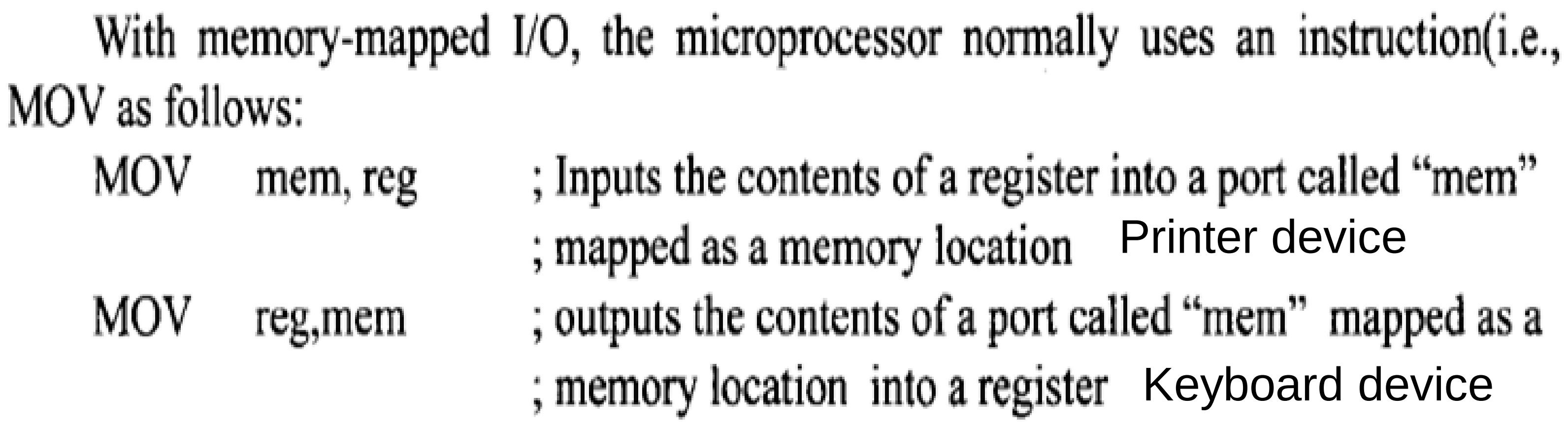
  Msb=1; I/O port selected

  Msb=0; memory location is selected
- This reduces the micro-processor memory by 50%.


Standard I/O
Memory
Memory
I/O
Memory Space
I/O Space
Memory mapped I/O
I/O
Memory
I/O
Memory
Memory space


System address space
Available for application use
I/O address space
0xFFFF
LCD
Reserved for I/O address space
Serial Line
0x0000

Memory mapped I/O

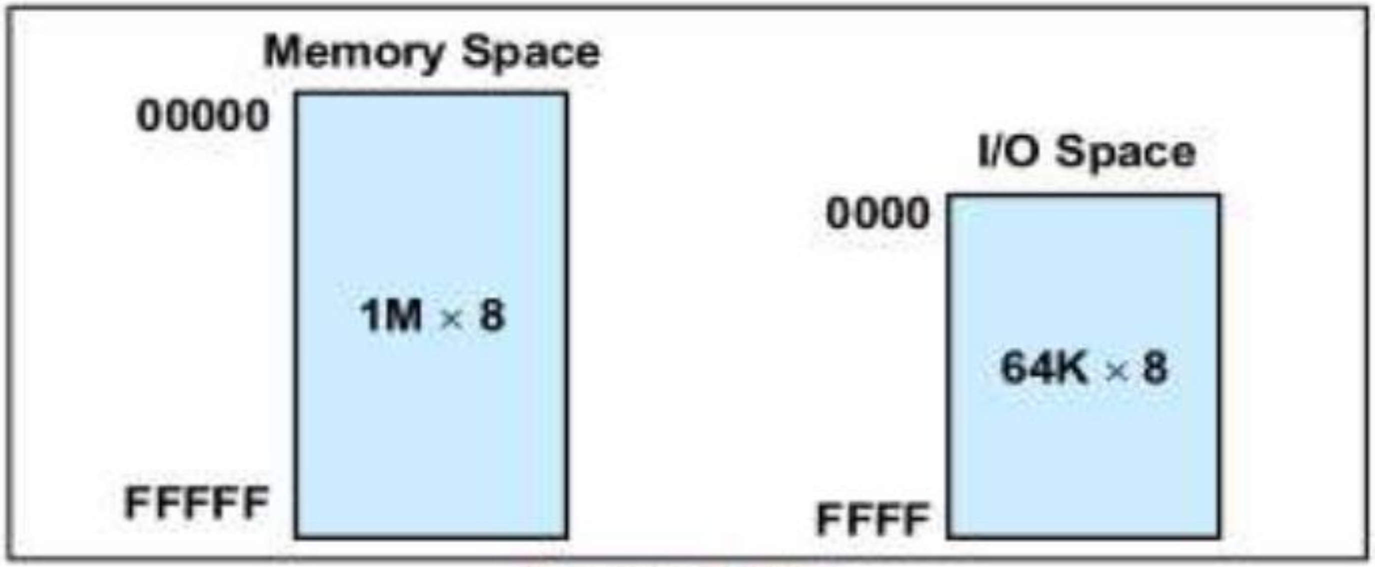


With memory-mapped I/O, the microprocessor normally uses an instruction(i.e., MOV as follows:

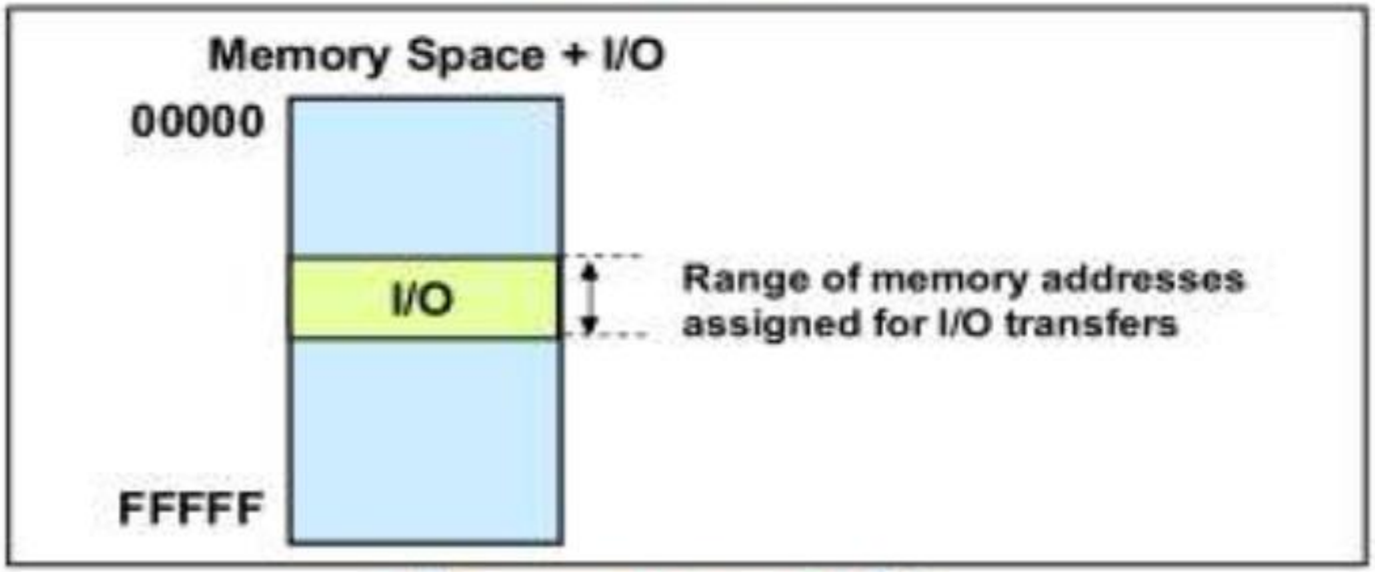
```
MOV    mem, reg     ; Inputs the contents of a register into a port called "mem"
                    ; mapped as a memory location
MOV    reg,mem      ; outputs the contents of a port called "mem"  mapped as a
                    ; memory location  into a register
```

Printer device

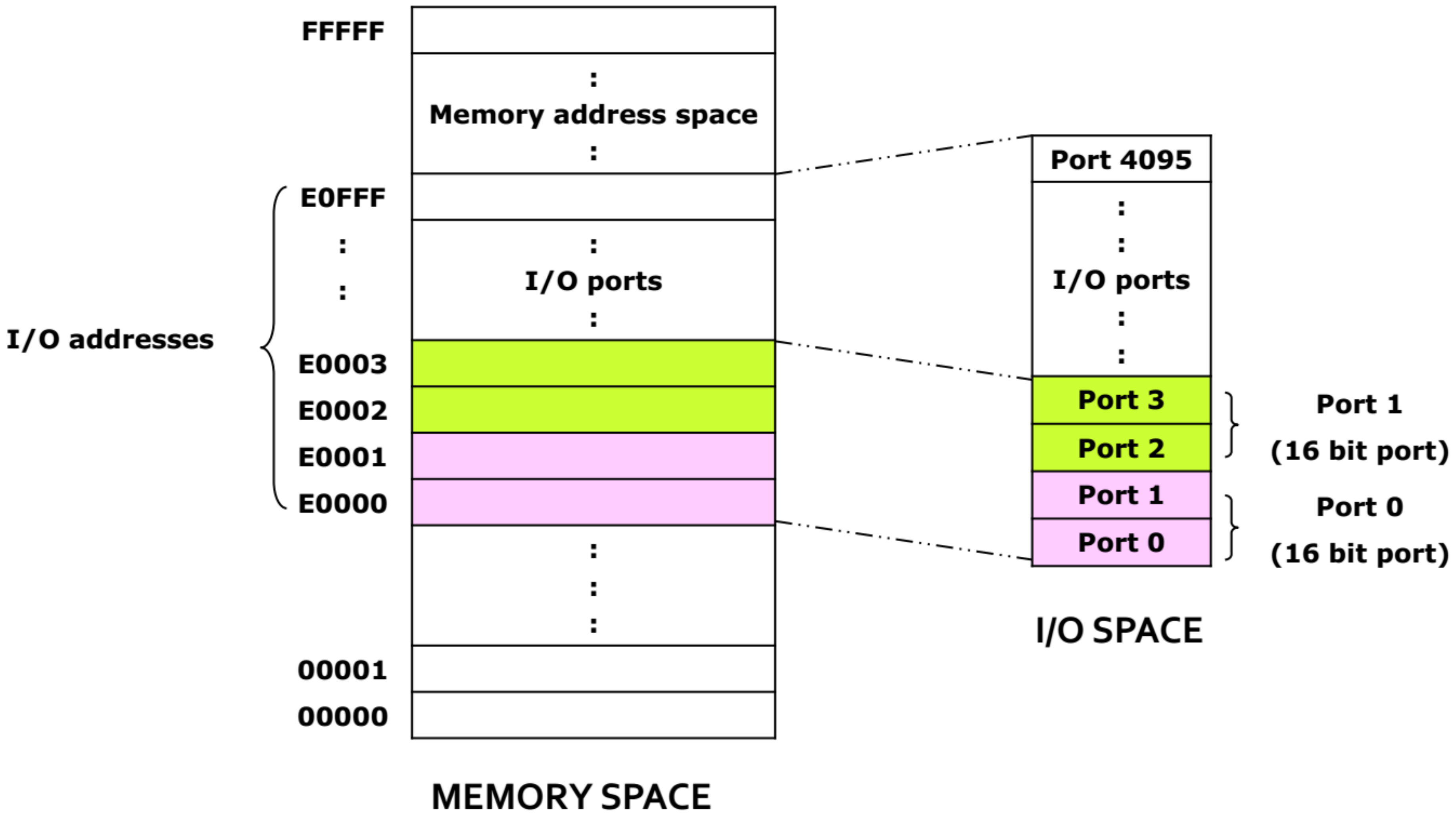
Keyboard device



Isolated I/O



Memory-Mapped I/O



Memory mapped I/O

Memory mapped I/O

ADVANTAGES

- All I/O locations are addressed in exactly the same manner as memory locations; no special I/O instructions is required .Thus the overall size of the instruction set is reduced.
- All arithmetic and logical operations can be performed on I/O data directly
- Used in system where memory requirement is small

DISADVANTAGES

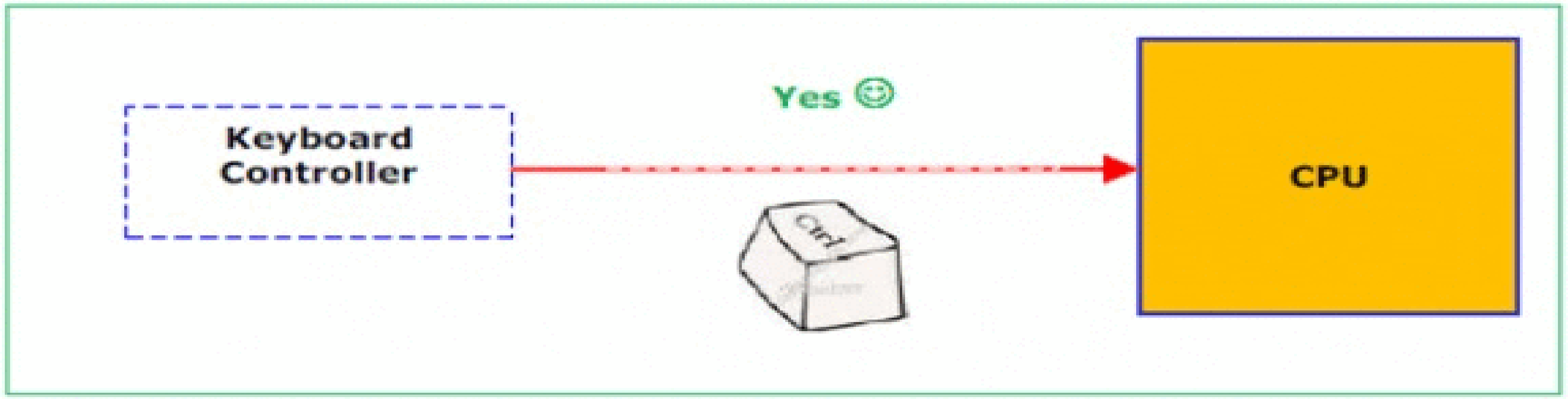
- Part of the memory address space is lost. (however, that with ported I/O systems, not all of the available I/O address space is always used.)

2. Interrupt driven I/O

- **Interrupt I/O**, *an external device can force the microprocessor to stop executing* the current program temporarily so that it can execute another program known as an **interrupt service routine**.

- *After* completing this program, a return from interrupt instruction can be executed at the end of the service routine to return control at the right place in the main program.

2. Interrupt driven I/O



- <u>The CPU executes other program</u>, as soon as a key is pressed, the Keyboard generates an interrupt. The CPU will response to the interrupt – read the data. After that returns to the original program.

2. Interrupt driven I/O

- Interrupts are useful when interfacing I/O devices at relatively low data transfer rates, such as keyboard inputs.

- Interrupt processing allows the processor to execute other software while the keyboard operator is thinking about what to type next.

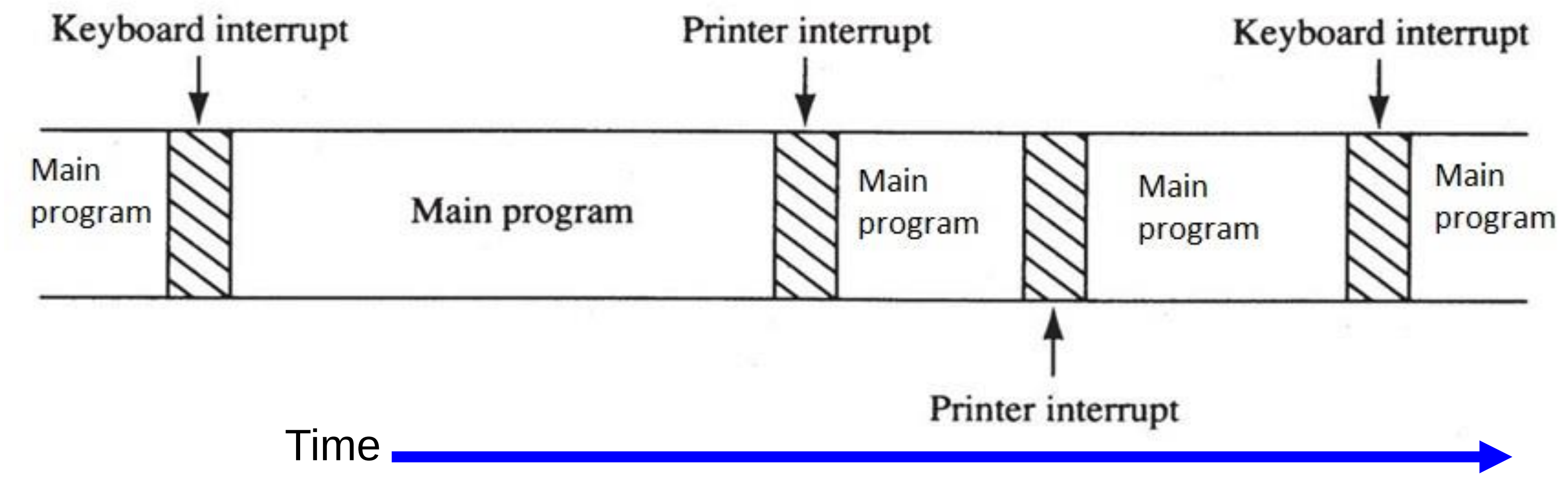


Figure: A time line that indicates interrupt usage in a typical system.

- A time line shows typing on a keyboard, a printer removing data from memory, and main program execution.
- The keyboard interrupt service procedure, called by the keyboard interrupt, and the printer interrupt service procedure called by the printer interrupt. Each interrupt take little time to execute.

2. Interrupt driven I/O

- ***Interrupt driven I/O*** is a device-initiated I/O transfer. The external device is connected to a pin called the interrupt (INT) pin on the microprocessor chip.

2. Interrupt I/O

There are typically two types of interrupts:

I. External interrupts,

II. Internal interrupts,

2. Interrupt I/O

External interrupts can be divided further into two types: maskable and non-maskable.

Maskable interrupt can be enabled or disabled by instructions.

The examples of maskable interrupt are: mouseclick, keystroke on keyboard etc.

2. Interrupt I/O

Non-Maskable interrupt cannot be enabled or disabled by instructions.

A non-maskable interrupt is typically used as a power failure interrupt.

Micro-processor normally use +5V which is transformed from 110V AC. If the voltage drops below 90V then +5V cannot be maintained. However, it will take a few milliseconds to drop power below 90V. In these few millisecond, the power failure-sensing circuit can interrupt the micro-processor. An interrupt service routine can be written to store critical data in non-volatile memory.

2. Interrupt I/O

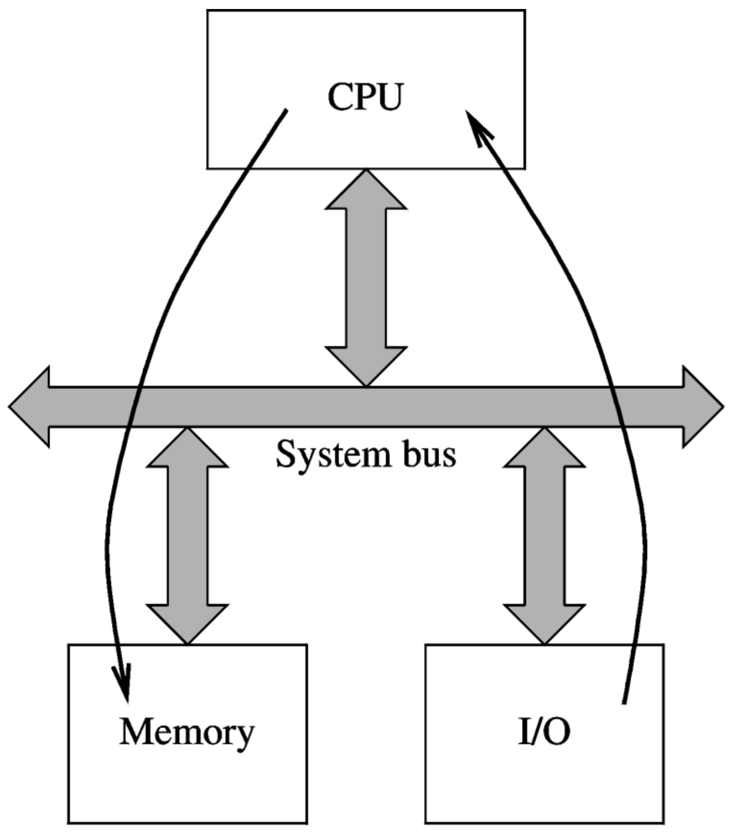
Internal interrupts, are activated internally by exceptional conditions such as overflow, division by zero, or execution of an illegal op-code.

C. Direct memory access (DMA)

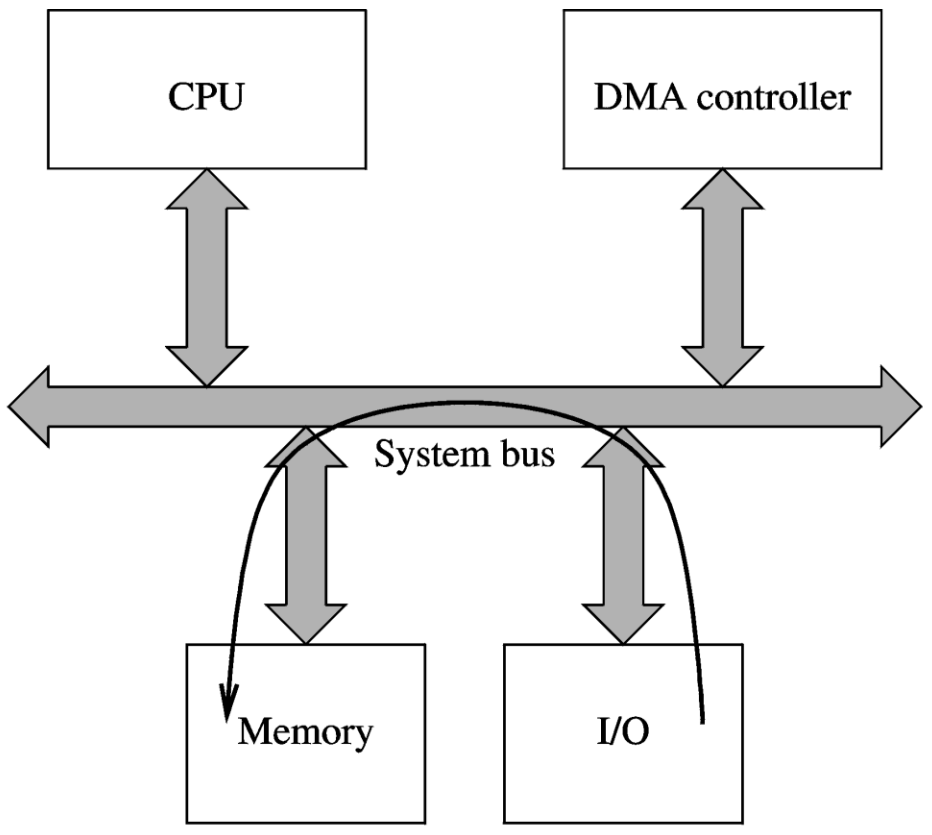
Direct memory access (DMA) is a type of I/O technique in which data can be transferred between microcomputer memory and an external device such as the hard disk, without microprocessor involvement.

A special chip called the DMA controller chip 8237 is typically used with the microprocessor for transferring data using DMA.

C. Direct Memory Access (DMA)



(a) Programmed I/O transfer



(b) DMA transfer

C. Direct Memory Access (DMA)

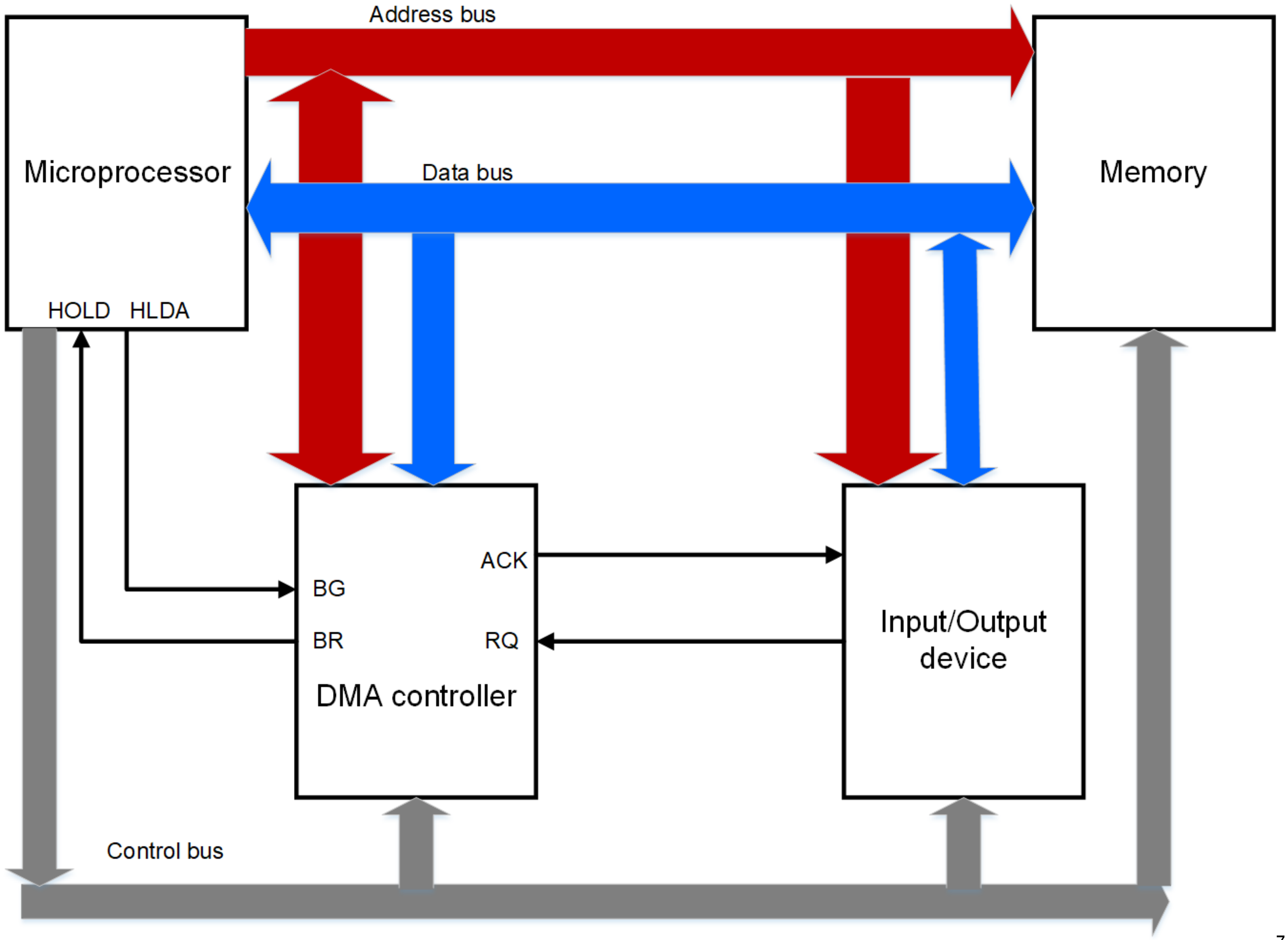


Figure: Micro-computer interfaced with DMA controller chip 8237

C. Direct Memory Access (DMA)

DMA operation

1. The I/O devices request DMA operation via the DMA request line of the controller chip.
2. The controller chip activates the microprocessor HOLD pin, requesting the microprocessor to release the bus.
3. The microprocessor sends HLDA (hold acknowledge) back to the DMA controller, indicating that the bus is disabled. <u>The DMA controller places the memory address on the address bus</u> and sends a DMA acknowledge to the peripheral device.
4. The DMA controller completes the DMA transfer and release the buses.

C. Direct Memory Access (DMA)

The DMA controller includes several registers :

- The DMA Address Register contains the memory address to be used in the data transfer.

- The DMA Count Register, also called Word Count Register, contains the no. of bytes of data to be transferred.

- The DMA Control Register accepts commands from the CPU.

C. Direct Memory Access (DMA)

Modes vary by how the DMA controller determines when to transfer data.

- BURST mode
- CYCLE STEALING Mode
- TRANSPARENT Mode

C. Direct Memory Access (DMA)

BURST mode

- DMA controller obtains access to the system buses using BR (Bus Request) & BG (Bus Grant) signals.
- An entire block of data is transferred in one contiguous sequence. Once the DMA controller is granted access to the system buses by the CPU, it transfers all bytes of data in the memory data block before releasing control of the system buses back to the CPU.
- This mode is useful for loading programs or data files into memory, but it make the micro-processor CPU inactive for relatively long periods of time.

C. Direct Memory Access (DMA)

CYCLE STEALING Mode

- Viable alternative for systems in which the micro-processor CPU should not be disabled for the length of time needed for Burst transfer modes.
- DMA controller obtains access to the system buses using BR (Bus Request) & BG (Bus Grant) signals.
- The DMA controller, after transferring one byte of data, releases control of the system buses by sending a bus grant signal (BG) through the control bus, and lets the CPU process an instruction and then requests access to the bus by sending the bus request signal through the control bus and then transfers another byte of data.

C. Direct Memory Access (DMA)

CYCLE STEALING Mode

- By continually obtaining and releasing control of the system buses, the DMA controller essentially interleaves (one-after-another) instruction & data transfers. The micro-processor CPU processes an instruction, then the DMA controller transfers a data value, and so on.

- The data block is not transferred as quickly as in burst mode, but the CPU is not idled for as long as in that mode.

C. Direct Memory Access (DMA)

TRANSPARENT Mode

- The DMA controller only transfers data when the micro-processor CPU is performing operations that do not use the system buses.

- This is the slowest mode to transfer a block of data, yet it is also the most efficient in terms of overall system performance, while the disadvantage is that the hardware needs to determine when the CPU is not using the system buses, which can be complex.

- Problem

Transfer of bus control in either direction, from processor to device or vice-versa, takes 250 ns. One of the IO device has data transfer rate of 75 KB/sec and employs DMA.

(a) Suppose we employ DMA in a burst mode. How long does it take to transfer a block of 256 bytes?

b) Data are transferred one byte at a time. Calculate the same for cycle stealing mode

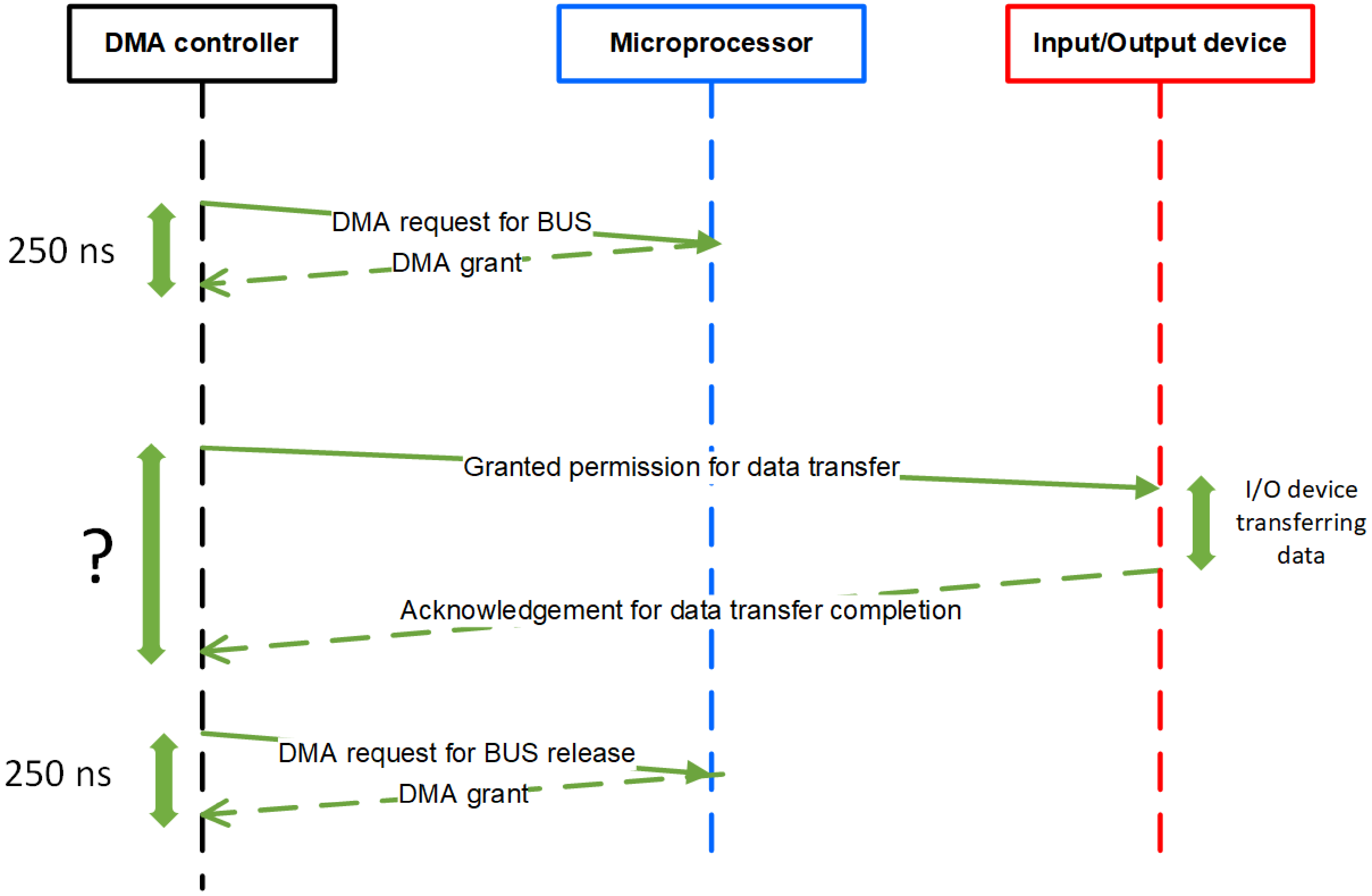

DMA controller
Microprocessor
Input/Output device
250 ns
DMA request for BUS
DMA grant
?
Granted permission for data transfer
I/O device transferring data
Acknowledgement for data transfer completion
250 ns
DMA request for BUS release
DMA grant

Figure : Sequence diagram showing DMA burst mode

1. Burst Mode (Block Mode)

- That is, the DMA interface gains bus mastership prior to the start of block transfer and maintains control of the bus until the whole block is transferred.
- Total time = (Transfer of bus control from Processor to DMA) + (Time taken to transfer 256 Bytes from DMA to mem) + (Finally transferring back bus control from DMA to Processor)
- Transfer of bus control from Processor to DMA = 250ns
- transferring back bus control from DMA to Processor= 250ns
- Time taken to transfer 256 Bytes from DMA to mem = 256/(75*1024) seconds = 3333333.33 ns
- Total = 250 + 3333333.33 +250

=3333833.33 ns

2.Cycle Stealing Mode -

- Let x=Time of Transfer of bus control from Processor to DMA
- Let y=Transfer time of 1 Byte(as Memory is Byte addressable) from DMA to Mem.
- Let z=Time of Transfer of bus control from DMA to Processor

- So the time sequence will be = xyz, xyz, xyz......so on until all 256 Bytes are transferred.
- 1 xyz transfers 1 Byte, total time for 1 xyz= 250ns + 13020.83ns (time to transfer 1 Byte) +250ns.

- 1 xyz takes a total of = 13520.83ns
- so 256 Bytes transfer will take = 13520.83ns * 256 = 3461333.33ns or 3461333330ps

Summary of I / O

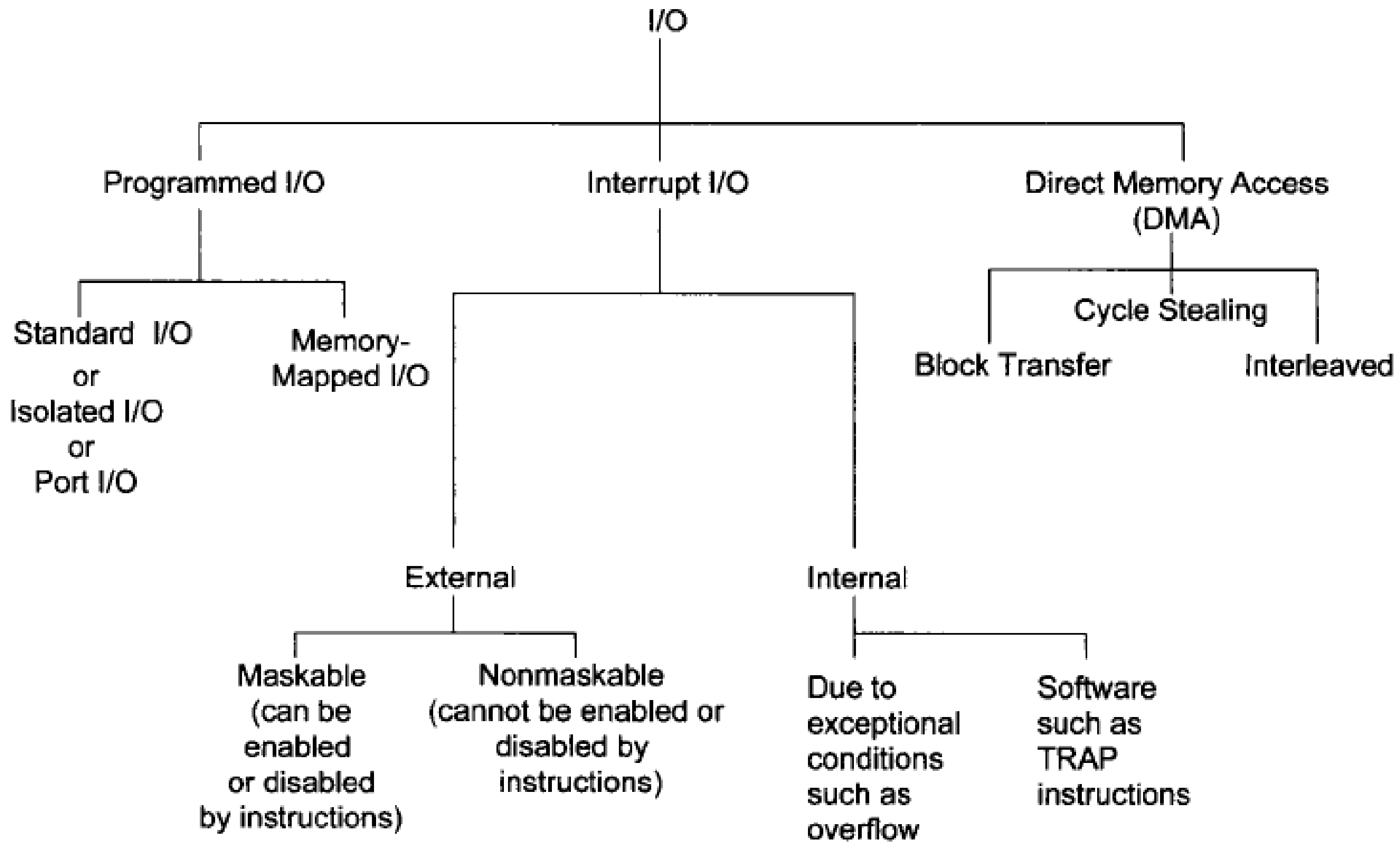
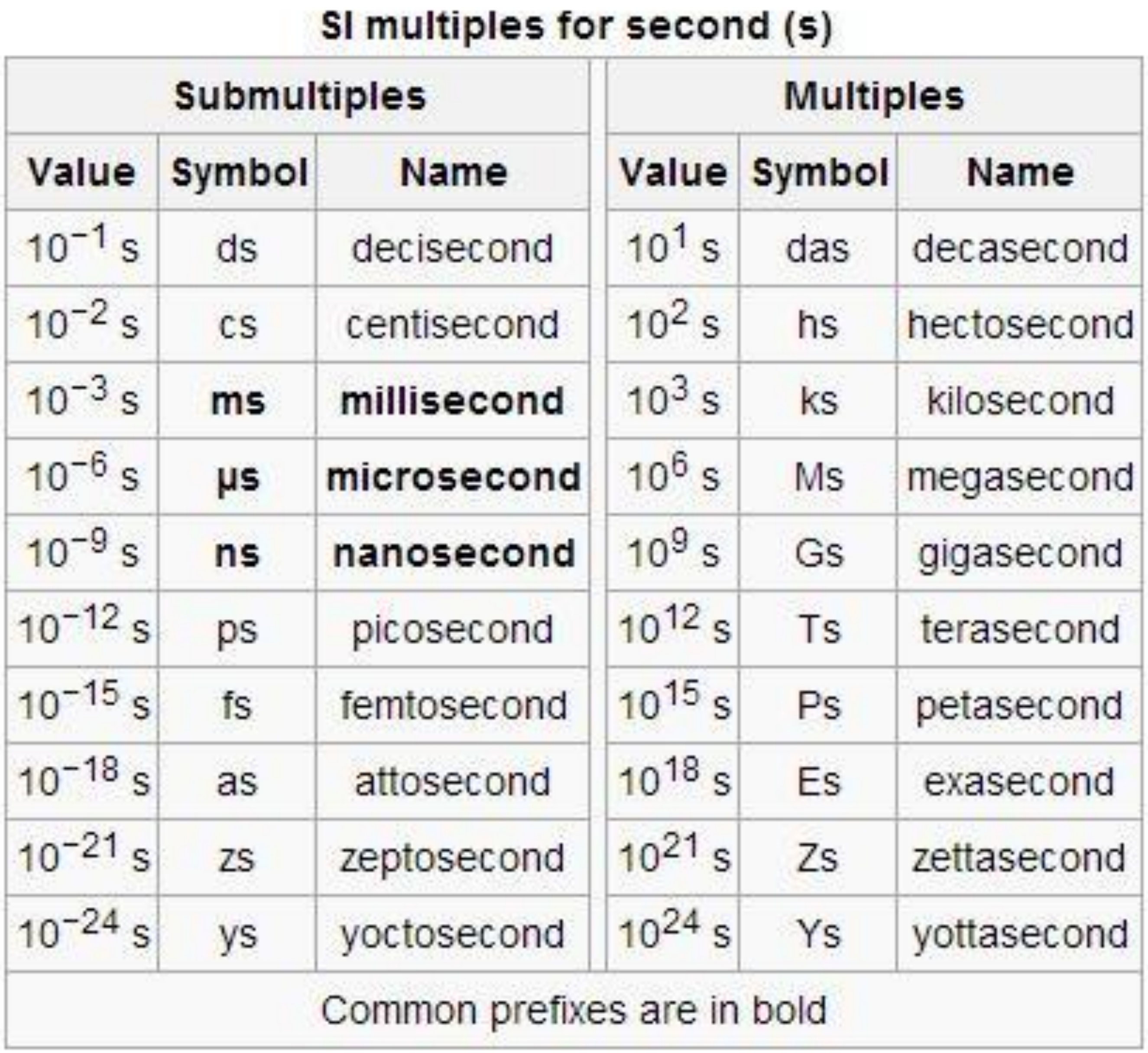

I/O
Programmed I/O
Interrupt I/O
Direct Memory Access (DMA)
Standard I/O or Isolated I/O or Port I/O
Memory-Mapped I/O
Cycle Stealing
Block Transfer
Interleaved
External
Internal
Maskable (can be enabled or disabled by instructions)
Nonmaskable (cannot be enabled or disabled by instructions)
Due to exceptional conditions such as overflow
Software such as TRAP instructions

FIGURE 4.15 I/O structure of a typical microcomputer.



SI multiples for second (s)

Submultiples			Multiples		
Value	**Symbol**	**Name**	**Value**	**Symbol**	**Name**
10^{-1} s	ds	decisecond	10^{1} s	das	decasecond
10^{-2} s	cs	centisecond	10^{2} s	hs	hectosecond
10^{-3} s	**ms**	**millisecond**	10^{3} s	ks	kilosecond
10^{-6} s	**µs**	**microsecond**	10^{6} s	Ms	megasecond
10^{-9} s	**ns**	**nanosecond**	10^{9} s	Gs	gigasecond
10^{-12} s	ps	picosecond	10^{12} s	Ts	terasecond
10^{-15} s	fs	femtosecond	10^{15} s	Ps	petasecond
10^{-18} s	as	attosecond	10^{18} s	Es	exasecond
10^{-21} s	zs	zeptosecond	10^{21} s	Zs	zettasecond
10^{-24} s	ys	yoctosecond	10^{24} s	Ys	yottasecond

Common prefixes are in bold