Computer Organization and Architecture

UNIT-5

Input-Output Organization: Peripheral Devices, Input-output Interface, Asynchronous Data Transfer, Priority Interrupt, Direct Memory Access (DMA), Input-Output Processor.

Input Output Organization

I/O Subsystem

- Provides an efficient mode of communication between the central system and the outside environment
- Programs and data must be entered into computer memory for processing and results obtained from computer must be recorded and displayed to user.
- When input transferred via slow keyboard processor will be idle most of the time waiting for information to arrive

- **I/O Devices (Peripherals)**
- **ASCII**

I/O(Peripheral) Devices

- **Devices that are under direct control of computer are said to be connected on-line.**
- **Input or output devices attached to the computer are also called peripherals.**
- **There are three types of peripherals :**
  - **Input peripherals**
  - **Output peripherals**
  - **Input-output peripherals**
- Examples-Peripheral (or I/O Device)
  - Monitor (*Visual Output Device*) : CRT, LCD, LED
  - Keyboard (*Input Device*) : light pen, mouse, touch screen, joy stick
  - Printer (Hard Copy Device) : **Daisy wheel**, **dot matrix** and **laser printer**
  - Storage Device : Magnetic tape, magnetic disk

Peripheral Devices

Input Devices

- Keyboard
- Optical input devices
  - - Card Reader
  - - Paper Tape Reader
  - - Bar code reader
  - - Digitizer
  - - Optical Mark Reader
- Magnetic Input Devices
  - - Magnetic Stripe Reader
- Screen Input Devices
  - - Touch Screen
  - - Light Pen
  - - Mouse
- Analog Input Devices

Output Devices

- Card Puncher, Paper Tape Puncher
- CRT
- Printer (Impact, Ink Jet, Laser, Dot Matrix)
- Plotter
- Analog
- Voice

ASCII (*American Standard Code for Information Interchange*)

- I/O communications usually involves transfer of alphanumeric information from the device and the computer.
- Standard binary code for alphanumeric character is **ASCII**
  - ASCII Code :
    - It uses 7 bits to code 128 characters (94 printable and 34 non printing)
    - 7 bit - 00 - 7F (*0 - 127*)
  - ASCII is 7 bits but most computers manipulate 8 bit quantity as a single unit called byte.
    - 80 - FF (128 - 255) : Greek, Italic type font
    - Three types of control characters:
      - Format effectors
      - Information separators
      - Communication control

ASCII Alphanumeric Characters

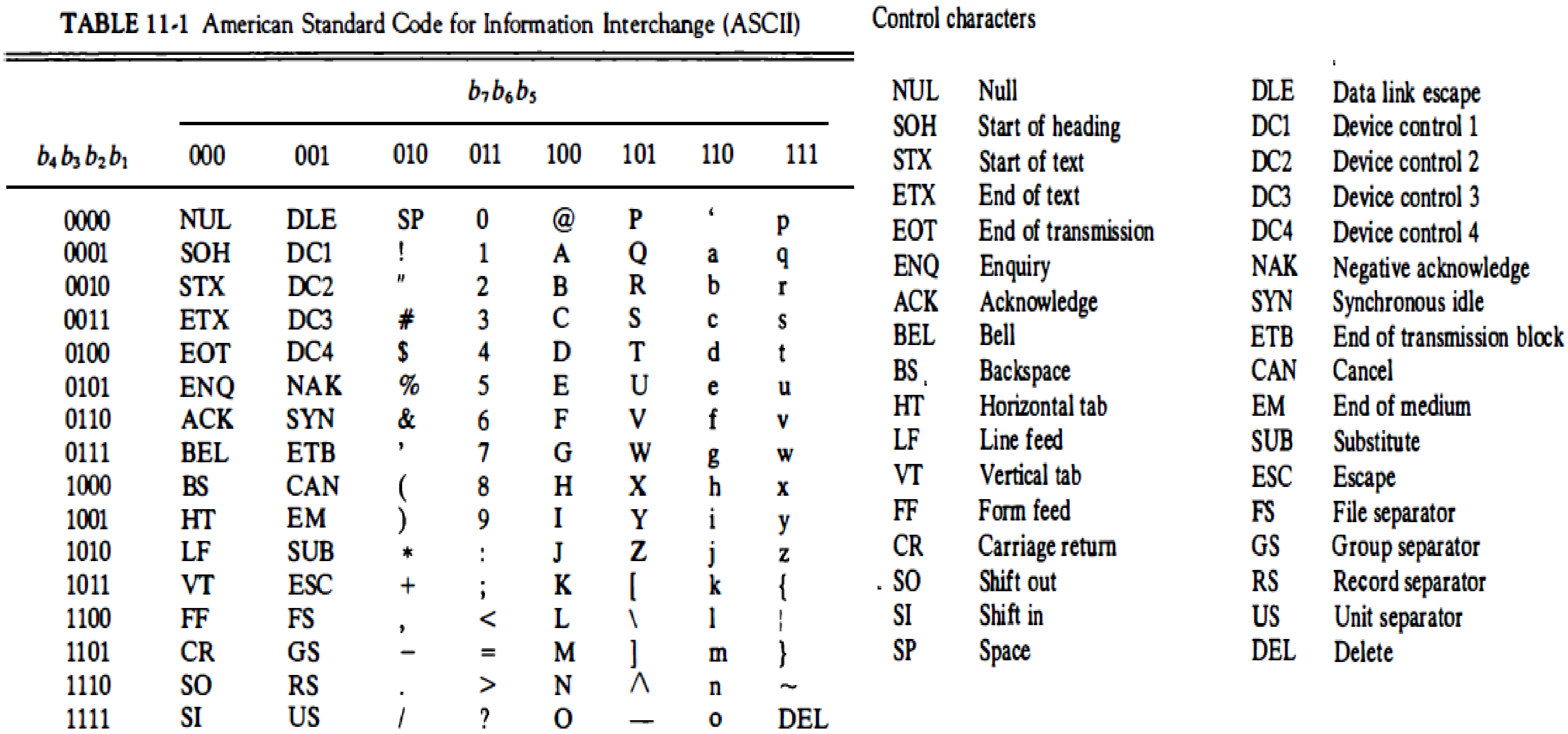


TABLE 11-1 American Standard Code for Information Interchange (ASCII)

	$b_7b_6b_5$							
$b_4b_3b_2b_1$	000	001	010	011	100	101	110	111
0000	NUL	DLE	SP	0	@	P	‘	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	"	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	’	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	¦
1101	CR	GS	−	=	M	]	m	}
1110	SO	RS	.	>	N	∧	n	~
1111	SI	US	/	?	O	—	o	DEL

Control characters

NUL	Null	DLE	Data link escape
SOH	Start of heading	DC1	Device control 1
STX	Start of text	DC2	Device control 2
ETX	End of text	DC3	Device control 3
EOT	End of transmission	DC4	Device control 4
ENQ	Enquiry	NAK	Negative acknowledge
ACK	Acknowledge	SYN	Synchronous idle
BEL	Bell	ETB	End of transmission block
BS	Backspace	CAN	Cancel
HT	Horizontal tab	EM	End of medium
LF	Line feed	SUB	Substitute
VT	Vertical tab	ESC	Escape
FF	Form feed	FS	File separator
CR	Carriage return	GS	Group separator
SO	Shift out	RS	Record separator
SI	Shift in	US	Unit separator
SP	Space	DEL	Delete

I/O Interface

- **Interface**
- **I/0 Bus and Interface Modules**
  - I/0 command
- I/O Bus versus Memory Bus
  - Isolated versus Memory-Mapped I/O

Interface

- **Interface: Provides a method for transferring information between internal storage (**such as memory and CPU registers**) and external I/O devices**
- **Resolves the *differences* between the computer and peripheral devices**

(1). Peripherals – Electromechanical or Electromagnetic Devices

CPU or Memory - Electronic Device

Conversion of signal values required

(2). Data Transfer Rate

Peripherals - Usually slower

CPU or Memory - Usually faster than peripherals

Some kinds of Synchronization mechanism may be needed

(3). Data formats or Unit of Information

Peripherals – Byte, Block, …

CPU or Memory – Word

(4). Operating modes of peripherals may differ

must be controlled so that not to disturbed other peripherals connected to CPU

Peripherals - Autonomous, Asynchronous

CPU or Memory - Synchronous

I/O BUS AND INTERFACE MODULES

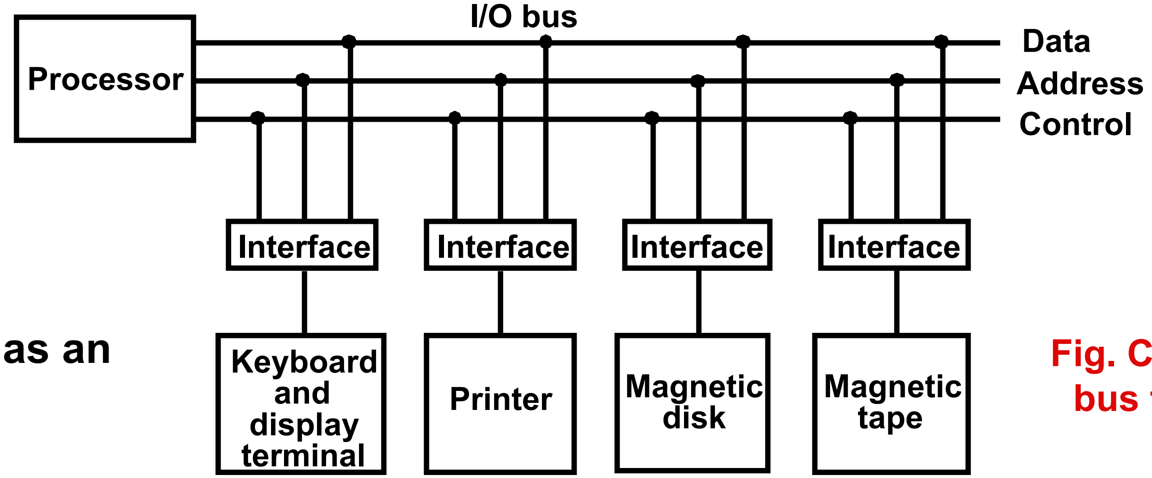


Fig. Connection of I/O bus to input-output devices

Each peripheral has an interface module associated with it

Interface

- **Decodes the device address (device code)**
- **Decodes the commands (operation)**
- **Provides signals for the peripheral controller**
- **Synchronizes the data flow and supervises the transfer rate between peripheral and CPU or Memory**

Typical I/O instruction

Op. code	Device address	Function code

(Command)

- The function code is referred to as an **I/O command** and is in essence an instruction that is executed in the interface and its attached peripheral unit

- There are **four types of commands** that an interface may receive.
  1. Control
  2. Status
  3. Data output
  4. Data input

I/O command

- Control command : is issued to activate peripheral and to inform what to do
- Status command : used to test various status condition in the interface and the peripherals
- Data output command : causes the interface to respond by transferring data from the bus into one of its registers
- Data input command : interface receives an item of data from the peripheral and places it in its buffer register.

I/O command

A control command is issued to activate the peripheral and to inform it what to do.
For example, a magnetic tape unit may be instructed to backspace the tape by one record, to rewind the tape, or to start the tape moving in the forward direction.
The particular control command issued depends on the peripheral, and each peripheral receives its own distinguished sequence of control commands, depending on its mode of

A status command is used to test various status conditions in the interface and the peripheral.
For example, the computer may wish to check the status of the peripheral before a transfer is initiated.
During the transfer, one or more errors may occur which are detected by the interface.
These errors are designated by setting bits in a status register that the processor can read at

A data output command causes the interface to respond by transferring data from the bus into one of its registers. Consider an example with a tape unit.
The computer starts the tape moving by issuing a control command.
The processor then monitors the status of the tape by means of a status command.
When the tape is in the correct position, the processor issues a data output command.
The interface responds to the address and command and transfers the information from the data lines in the bus to its buffer register.
The interface then communicates with the tape controller and sends the data to be stored on

The data input command is the opposite of the data output.
In this case the interface receives an item of data from the peripheral and places it in its buffer register.
The processor checks if data are available by means of a status command and then issues a data input command.
The interface places the data on the data lines, where they are accepted by the processor.

I/O Bus and Memory Bus

Functions of Buses

- ***MEMORY BUS*** **is for information transfers between CPU and the MM**
- ***I/O BUS*** **is for information transfers between CPU and I/O devices through their I/O interface**

Different ways of communications:

- **3 ways to bus can communicate with memory and I/O :**

(1). Use two separate buses, one to communicate with memory and the other with I/O interfaces

- Computer has independent set of data, address and control bus one for accessing memory and another I/O.
- Done in computers that have separate IOP other than CPU.

(2). Use one common bus for memory and I/O but separate control lines for each

(3). Use one common bus for memory and I/O with common control lines

Isolated vs. Memory Mapped I/O

Isolated I/O

- **Many computers use common bus to transfer information between memory or I/O.**
- **Separate I/O read/write control lines in addition to memory read/write control lines**
- **Separate (isolated) memory and I/O address spaces**
- **Separate input and output instructions**
  - Each associated with address of interface register

Memory-mapped I/O

- **A single set of read/write control lines (no distinction between memory and I/O transfer)**
- **Memory and I/O addresses share the common address space**
  - Reduces memory address range available
- **No specific input or output instruction**
  - The same memory reference instructions can be used for I/O transfers
- **Considerable flexibility in handling I/O operations**

I/O Interface: Asynchronous communication interface

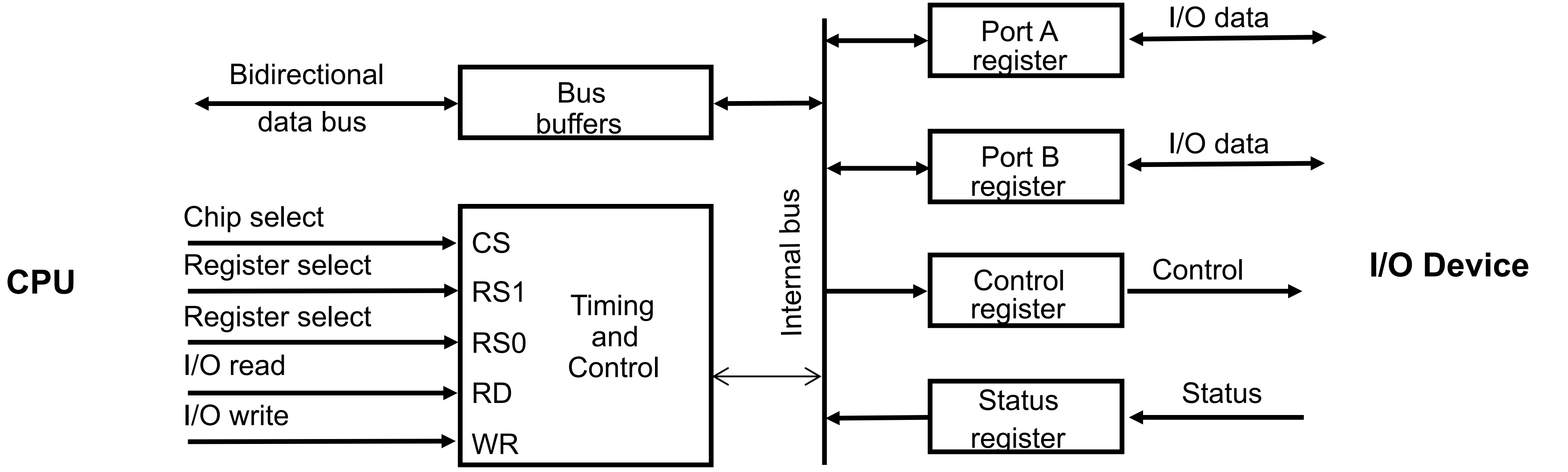


Fig. Example of I/O interface

CS	RS1	RS0	Register selected
0	x	x	None - data bus in high-impedence
1	0	0	Port A register
1	0	1	Port B register
1	1	0	Control register
1	1	1	Status register

Programmable Interface

- Information in each port can be assigned a meaning depending on the mode of operation of the I/O device
  - Port A = Data; Port B = Command; Port C = Status

- CPU initializes (loads) each port by transferring a byte to the Control Register
  - Allows CPU can define the mode of operation of each port
  - *Programmable Port*: By changing the bits in the control register, it is possible to change the interface characteristics

Asynchronous Data Transfer

- Synchronous Data Transfer
  - All data transfers occur simultaneously during the occurrence of a clock pulse
  - Two units such as CPU and I/O Interface are designed independently of each other. If the registers in the **interface** share a common clock with **CPU** registers. The transfer between the two are said to be synchronous.

- Asynchronous Data Transfer
  - Internal timing in each unit (*CPU and Interface*) is independent
  - Each unit uses its own private clock for internal registers

  - Asynchronous data transfer between two independent units requires that control signals be transmitted between the communicating units to indicate the time at which data is being transmitted.
    - **STROBE**
    - **HANDSHAKING**

⑤ Strobe : Control signal to indicate the time at which data is being transmitted

- ❶ 1) Source-initiated strobe : *Fig. 11-3*
- ❶ 2) Destination-initiated strobe : *Fig. 11-4*

* Employs a single control line to time each transfer
* The strobe may be activated by either the source or the destination unit

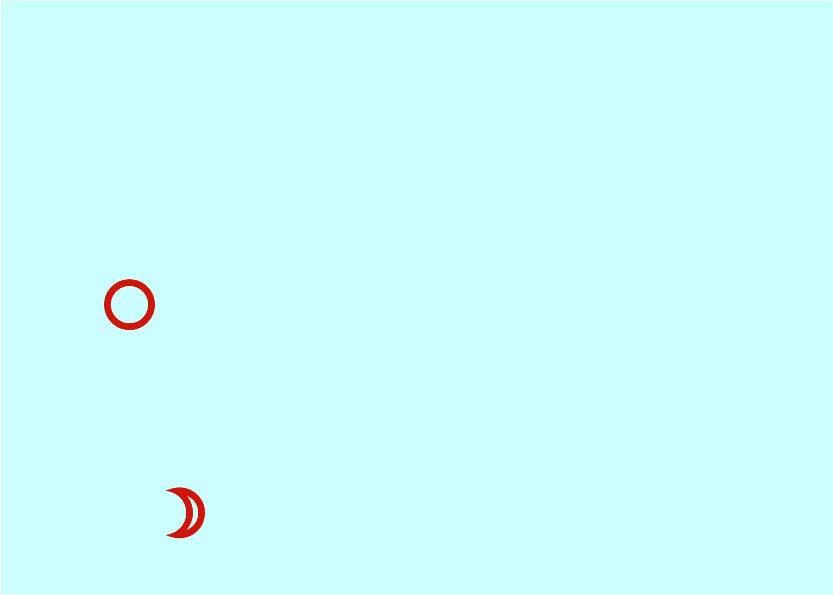


Fig. 11-3 Source-initiated strobe

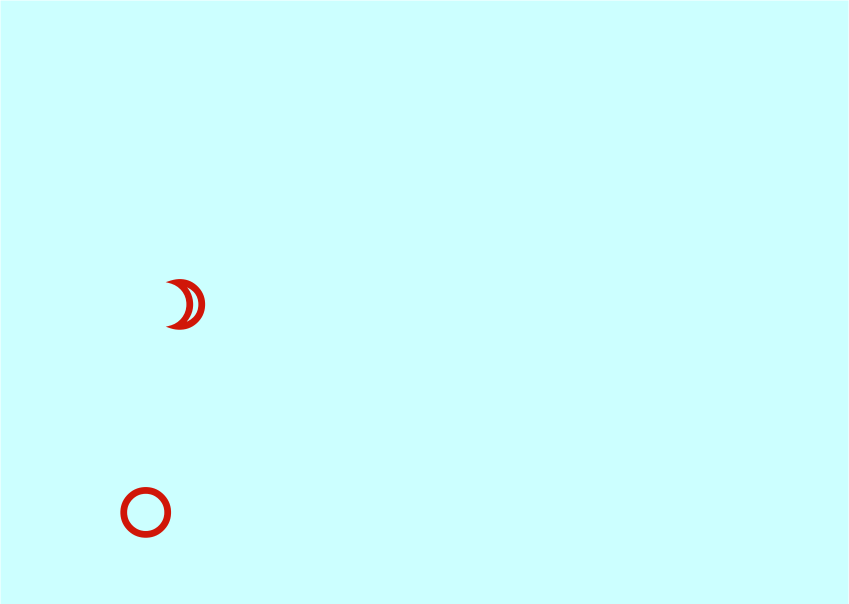


Fig. 11-4 Destination-initiated strobe

- ❶ Disadvantage of strobe method
  - ③ Destination Data
  - ③ Handshake method Data

HANDSHAKING

Strobe Methods

Source-Initiated

The source unit that initiates the transfer has no way of knowing whether the destination unit has actually received data

Destination-Initiated

The destination unit that initiates the transfer no way of knowing whether the source has actually placed the data on the bus

To solve this problem, the *HANDSHAKE* method introduces a second control signal to provide a *Reply* to the unit that initiates the transfer

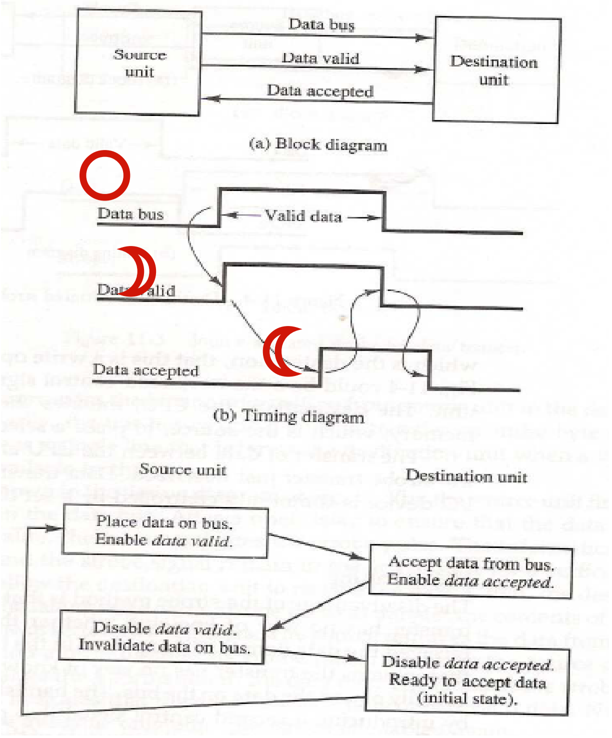
Handshake : Agreement between two independent units

1) Source-initiated handshake
2) Destination-initiated handshake

❶ Handshake : Agreement between two independent units

③ 1) Source-initiated handshake : *Fig. 11-5*

③ 2) Destination-initiated handshake : *Fig. 11-6*


Data bus
Source unit
Data valid
Destination unit
Data accepted
(a) Block diagram
Data bus
Valid data
Data valid
Data accepted
(b) Timing diagram
Source unit
Destination unit
Place data on bus. Enable *data valid*.
Accept data from bus. Enable *data accepted*.
Disable *data valid*. Invalidate data on bus.
Disable *data accepted*. Ready to accept data (initial state).

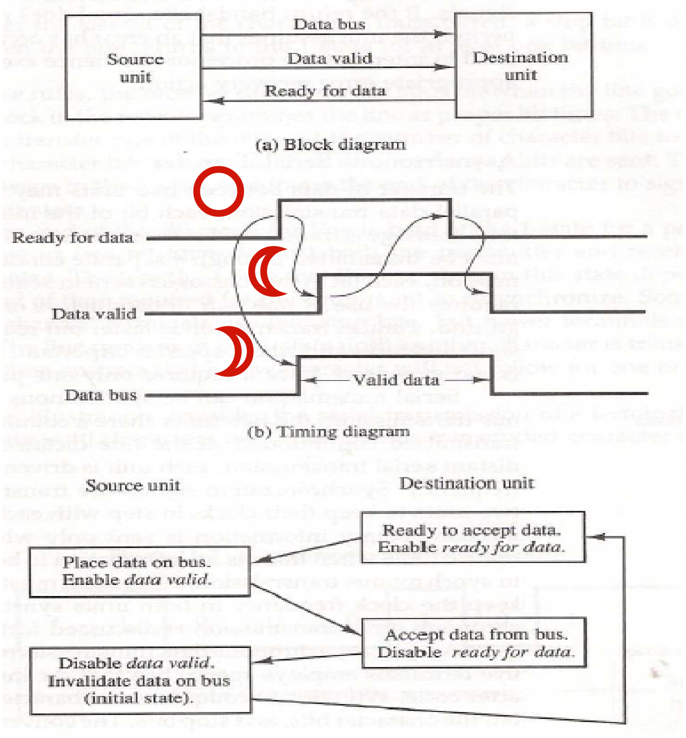
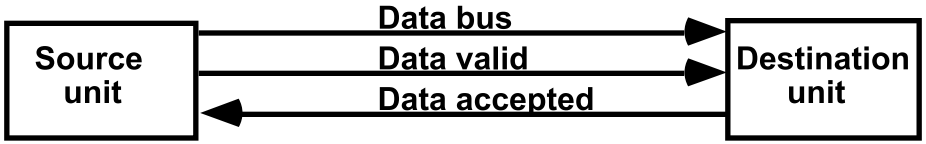

Data bus
Source unit
Data valid
Destination unit
Ready for data
(a) Block diagram
Ready for data
Data valid
Valid data
Data bus
(b) Timing diagram
Source unit
Destination unit
Ready to accept data. Enable *ready for data*.
Place data on bus. Enable *data valid*.
Accept data from bus. Disable *ready for data*.
Disable *data valid*. Invalidate data on bus (initial state).

Fig. 11-5 Source-initiated handshake *Fig. 11-6 Destination-initiated handshake*

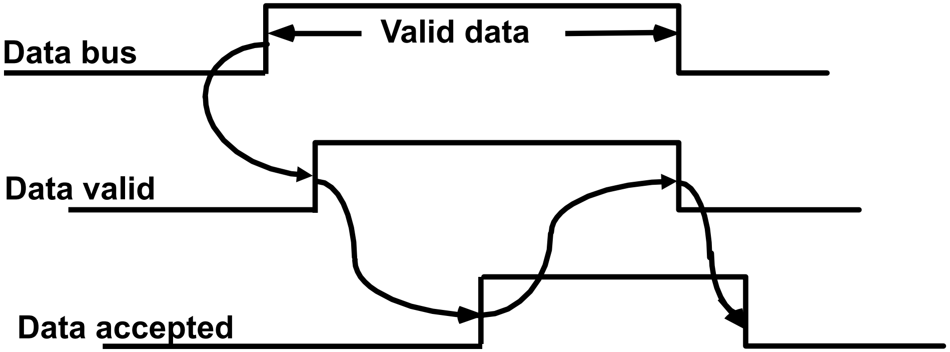
③ **Timeout** : If the return handshake signal does not respond within a given time period, the unit assumes that an error has occurred.

SOURCE-INITIATED TRANSFER USING HANDSHAKE

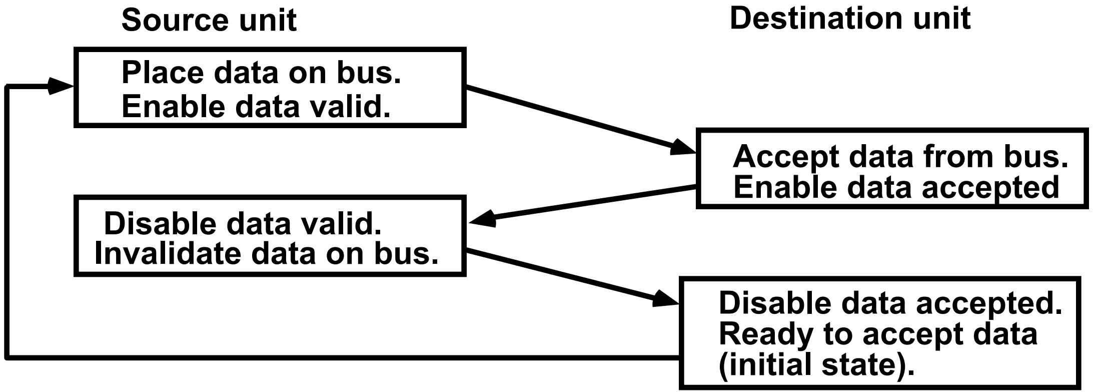
Block Diagram



Timing Diagram



Sequence of Events



Source unit

Destination unit

Place data on bus.
Enable data valid.

Accept data from bus.
Enable data accepted

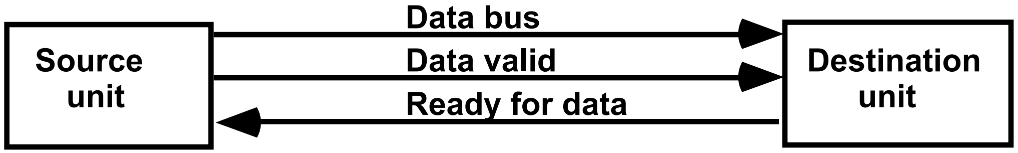
Disable data valid.
Invalidate data on bus.

Disable data accepted.
Ready to accept data
(initial state).

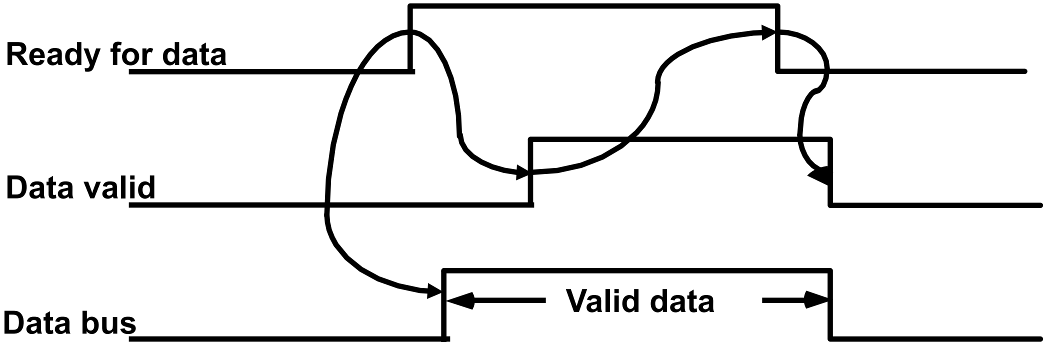
* Allows arbitrary delays from one state to the next
* Permits each unit to respond at its own data transfer rate
* The rate of transfer is determined by the slower unit

DESTINATION-INITIATED TRANSFER USING HANDSHAKE

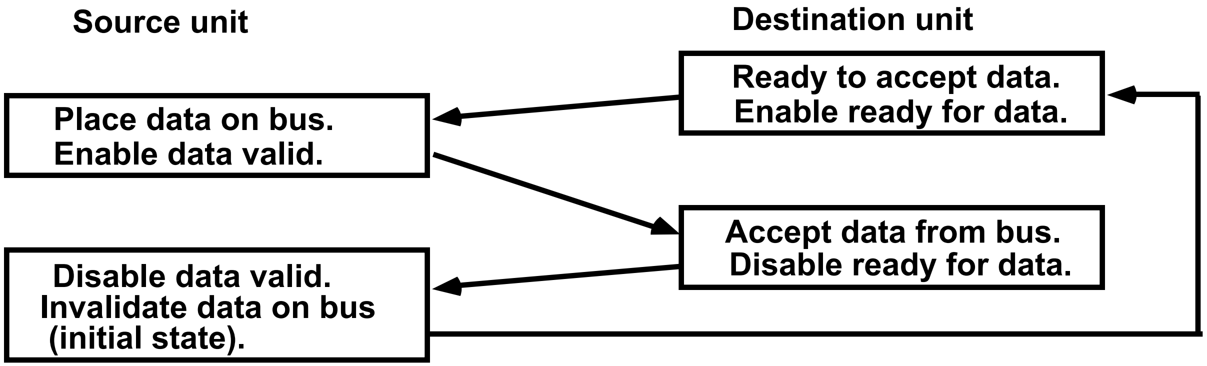
Block Diagram



Timing Diagram



Sequence of Events



Ready to accept data. Enable ready for data.

Place data on bus. Enable data valid.

Accept data from bus. Disable ready for data.

Disable data valid. Invalidate data on bus (initial state).

- * Handshaking provides a high degree of flexibility and reliability because the successful completion of a data transfer relies on active participation by both units
- * If one unit is faulty, data transfer will not be completed -> Can be detected by means of a *timeout* mechanism

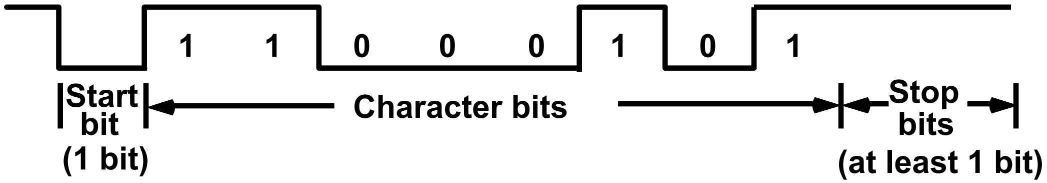
ASYNCHRONOUS SERIAL TRANSFER

Four Different Types of Transfer

Asynchronous serial transfer
Synchronous serial transfer
Asynchronous parallel transfer
Synchronous parallel transfer

Asynchronous Serial Transfer

- **Employs special bits which are inserted at both ends of the character code**
- **Each character consists of three parts; Start bit; Data bits; Stop bits.**



A character can be detected by the receiver from the knowledge of 4 rules;

- **When data are not being sent, the line is kept in the 1-state (idle state)**
- **The initiation of a character transmission is detected by a *Start Bit* , which is always a 0**
- **The character bits always follow the *Start Bit***
- **After the last character , a *Stop Bit* is detected when the line returns to the 1-state for at least 1 bit time**

The receiver knows in advance the transfer rate of the bits and the number of information bits to expect

❶ Asynchronous Serial Transfer

- ③ Synchronous transmission :
  - » The two unit share a common clock frequency
  - » Bits are transmitted continuously at the rate dictated by the clock pulses
- ③ Asynchronous transmission :
  - » Binary information sent only when it is available and line remain idle otherwise
  - » Special bits are inserted at both ends of the character code
  - » Each character consists of three parts :
    - ⑤ 1) start bit : always “0”, indicate the beginning of a character
    - ⑤ 2) character bits : data
    - ⑤ 3) stop bit : always “1”
- ③ Asynchronous transmission rules :
  - » ○ When a character is not being sent, the line is kept in the 1-state
  - » ☽ The initiation of a character transmission is detected from the start bit, which is always “0”
  - » ☾ The character bits always follow the start bit
  - » ¢ After the last bit of the character is transmitted, a stop bit is detected when the line returns to the 1-state for at least one bit time

- ③ Baud Rate : Data transfer rate in bits per second
  - » 10 character per second with 11 bit format = 110 bit per second
- ③ UART (Universal Asynchronous Receiver Transmitter) : 8250
- ③ UART (Universal Synchronous/Asynchronous Receiver Transmitter) : 8251

Modes of Transfer

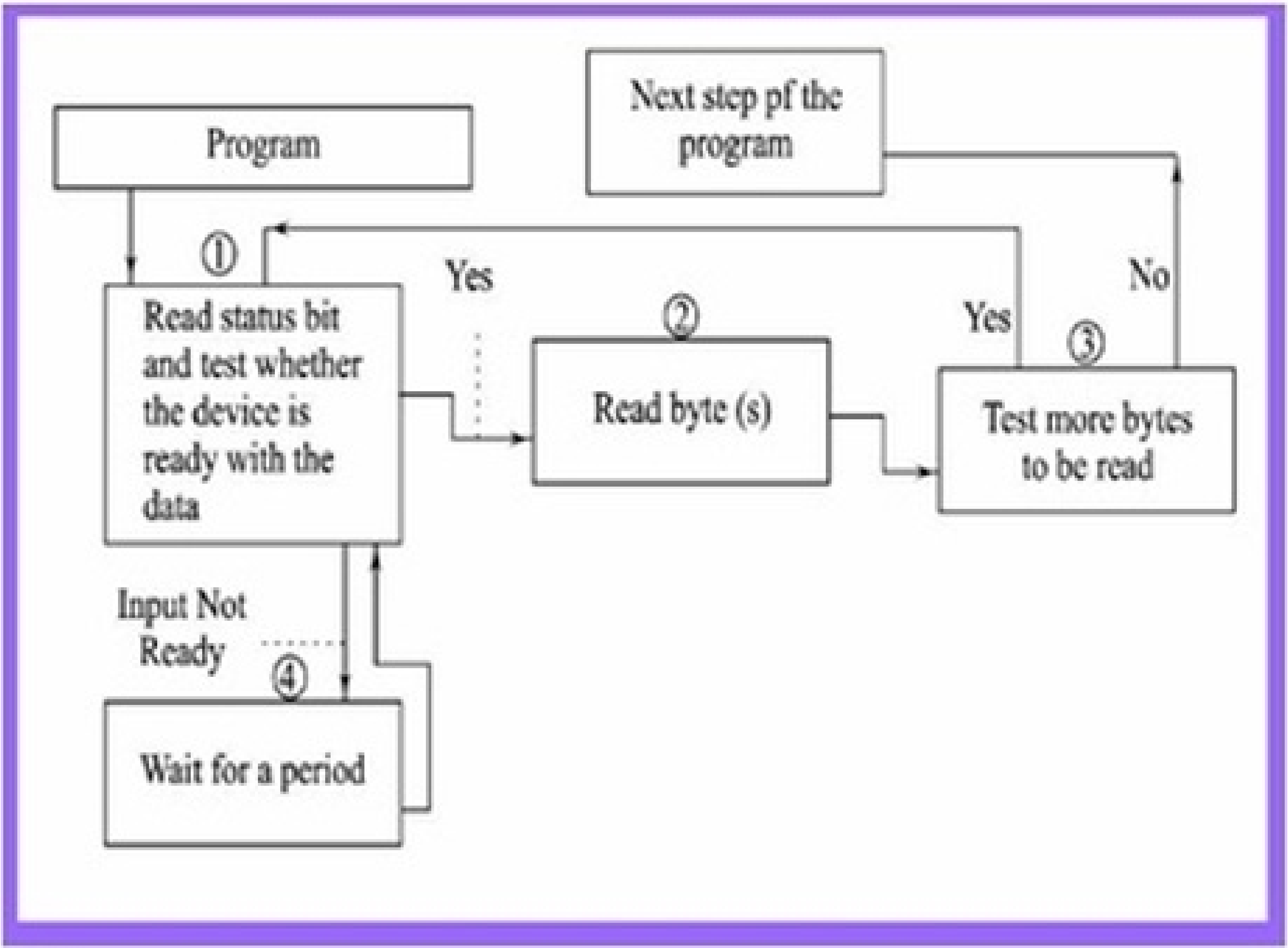
- Binary information received from an external device is usually stored in memory for later processing.
- Information transferred from the central computer into an external device originates in the memory unit.
- The CPU merely executes the i/o instructions and may accept the data temporarily, but the ultimate source or destination is the memory unit.
- Data transfer between the central computer and i/o devices may be handled in a variety of modes.
- Data transfer to and from peripherals
  1) Programmed I/O
  2) Interrupt-initiated I/O
  3) Direct Memory Access (**DMA**)
  4) I/O Processor (**IOP**)

Programmed I/O

- Programmed I/O operations are the result of I/O instructions written in the computer program.
- Each data item transfer is initiated by an instruction in the program.
- Usually, the transfer is to and from a CPU register and peripheral.
- Other instructions are needed to transfer the data to and from CPU and memory.
- Transferring data under program control requires constant monitoring of the peripheral by the CPU.
- Once a data transfer is initiated, the CPU is required to monitor the interface to see when a transfer can again be made.
- Example of Programmed I/O

Fig. Data transfer from I/O device to CPU

Programmed I/O


Program
Next step pf the program
①
Read status bit and test whether the device is ready with the data
Yes
②
Read byte (s)
Yes
No
③
Test more bytes to be read
Input Not Ready
④
Wait for a period

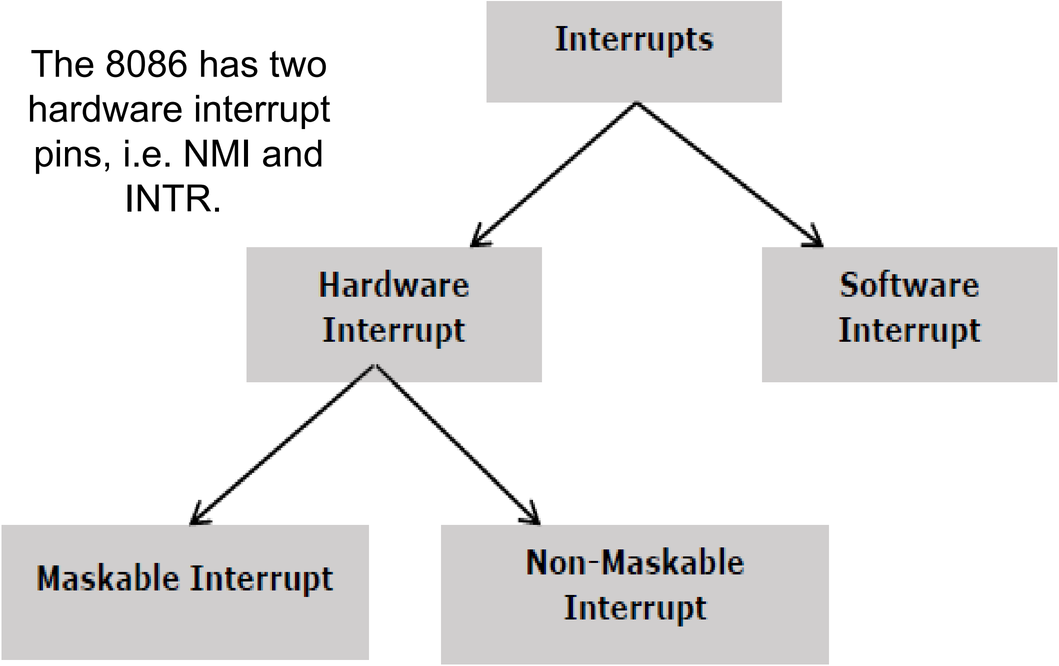
Programmed I/O Inefficient

- Consider a typical computer that can execute the two instructions that read the status register and check the flag in 1 μs.
- Assume that the input device transfers its data at an average rate of 100 bytes per second.
- This is equivalent to one byte every 10,000 μs (100,0000/100).
- This means that the CPU will check the flag 10,000 times between each transfer.
- The CPU is wasting time while checking the flag instead of doing some other useful processing task.

Interrupt-initiated I/O

- **Interrupt** is the method of creating a temporary halt during program execution and allows peripheral devices to access the CPU.
- The CPU responds to that interrupt with an **ISR** (Interrupt Service Routine), which is a short program to instruct the CPU on how to handle the interrupt.

The 8086 has two hardware interrupt pins, i.e. NMI and INTR.



Interrupts

Hardware Interrupt

Software Interrupt

Maskable Interrupt

Non-Maskable Interrupt

INT- Interrupt instruction with type number

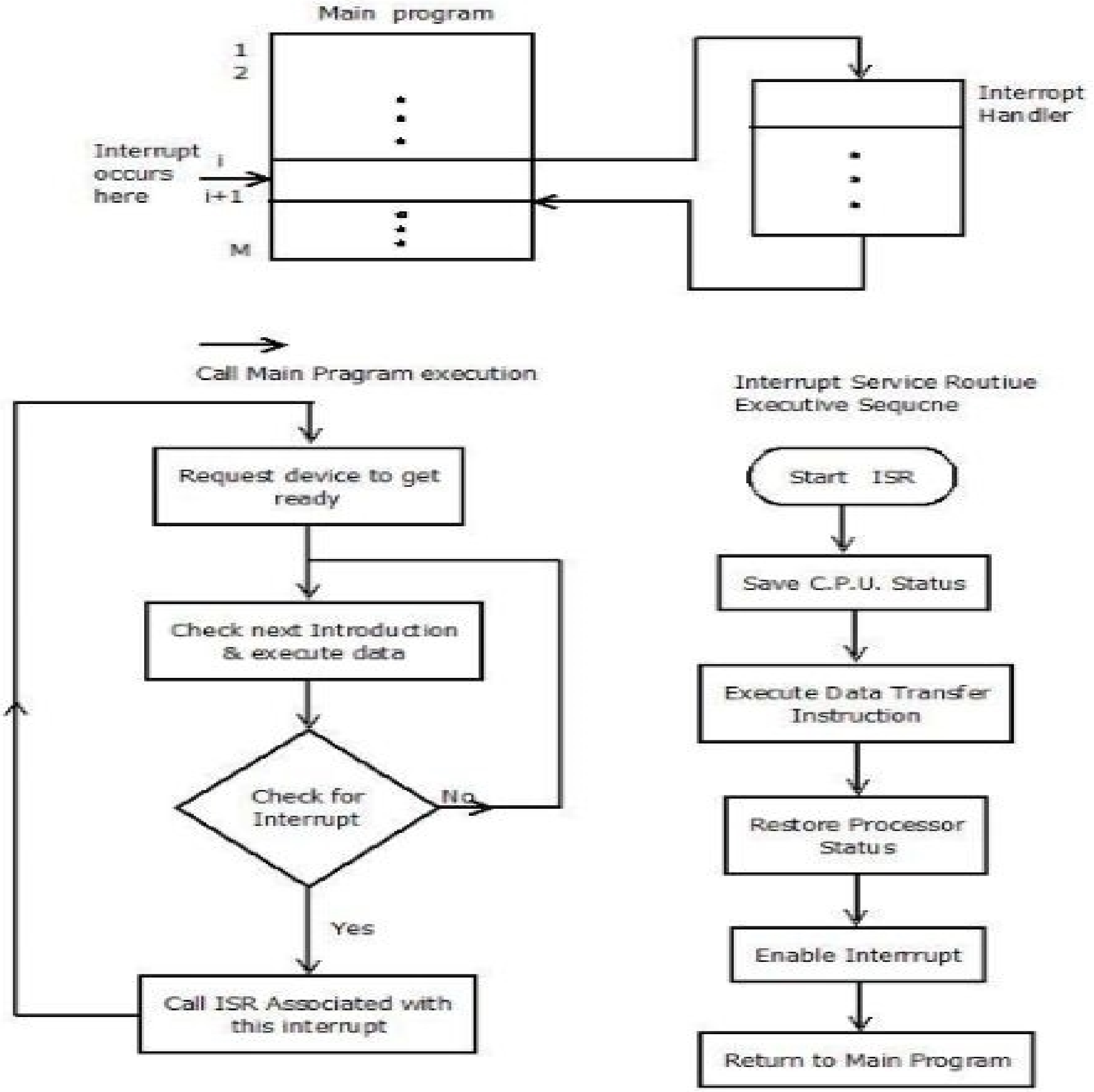
- **TYPE 0** interrupt represents division by zero situation.
- **TYPE 1** interrupt represents single-step execution during the debugging of a program.
- **TYPE 2** interrupt represents non-maskable NMI interrupt.
- **TYPE 3** interrupt represents break-point interrupt.
- **TYPE 4** interrupt represents overflow interrupt.

Interrupt-initiated I/O

- In the programmed i/o method, the CPU stays in a program loop until the I/O unit indicates that it is ready for data transfer.
- This is a time-consuming process since it keeps the processor busy needlessly.
- It can be avoided by using an interrupt facility and special commands to inform the interface to issue an interrupt request signal when the data are available from the device.
- In the meantime the CPU can proceed to execute another program.
- The interface meanwhile keeps monitoring the device.
- When the interface determines that the device is ready for data transfer, it generates an interrupt request to the computer.
- Upon detecting the external interrupt signal, the CPU momentarily stops the task it is processing, branches to a service program to process the I/O transfer, and then returns to the task it was originally performing.
- Interrupt-initiated I/O

  1) Non-vectored : fixed branch address

  2) Vectored : interrupt source supplies the branch address (**interrupt vector**)



Call Main Pragram execution

Request device to get ready

Check next Introduction & execute data

Check for Interrupt

No

Yes

Call ISR Associated with this interrupt

Interrupt Service Routiue Executive Sequcne

Start ISR

Save C.P.U. Status

Execute Data Transfer Instruction

Restore Processor Status

Enable Interrupt

Return to Main Program

Priority Interrupt

- Data transfer between the CPU and an I/O device is initiated by the CPU.
- However, the CPU cannot start the transfer unless the device is ready to communicate with the CPU.
- The readiness of the device can be determined from an interrupt signal.
- The CPU responds to the interrupt request by storing the return address from PC into a memory stack and then the program branches to a service routine that processes the required transfer.
- A priority interrupt is a system that establishes a priority over the various sources to determine which condition is to be serviced first when two or more requests arrive simultaneously.
- Identify the source of the interrupt when several sources will request service simultaneously
- Determine which condition is to be serviced first when two or more requests arrive simultaneously,
- Establishing the priority of simultaneous interrupts can be done by software or hardware.

  1) Software : Polling

  2) Hardware : Daisy chain, Parallel priority

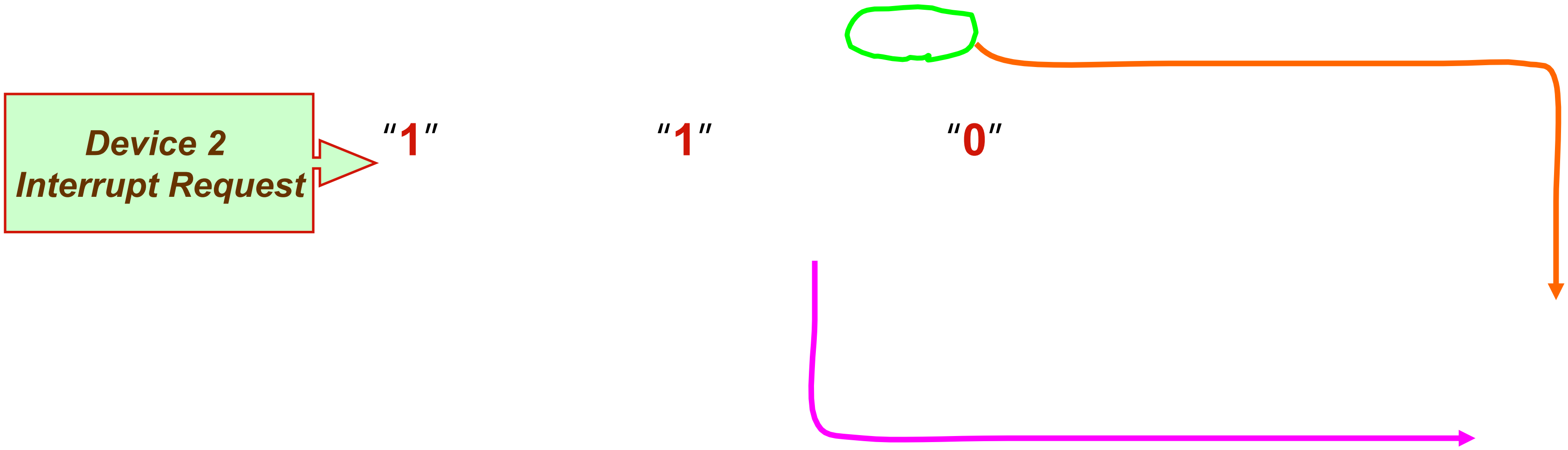
Polling

- Identify the highest-priority source by software means
  - One common branch address is used for all interrupts
  - Program polls the interrupt sources in sequence
  - The highest-priority source is tested first
- Polling priority interrupt If there are many interrupt sources, the time required to poll them can exceed the time available to service the I/O device.

Daisy-Chaining Priority

⑤ Hardware priority interrupt

- ❶ The hardware priority function can be established by either a serial or a parallel connection of interrupt lines called Daisy-Chaining



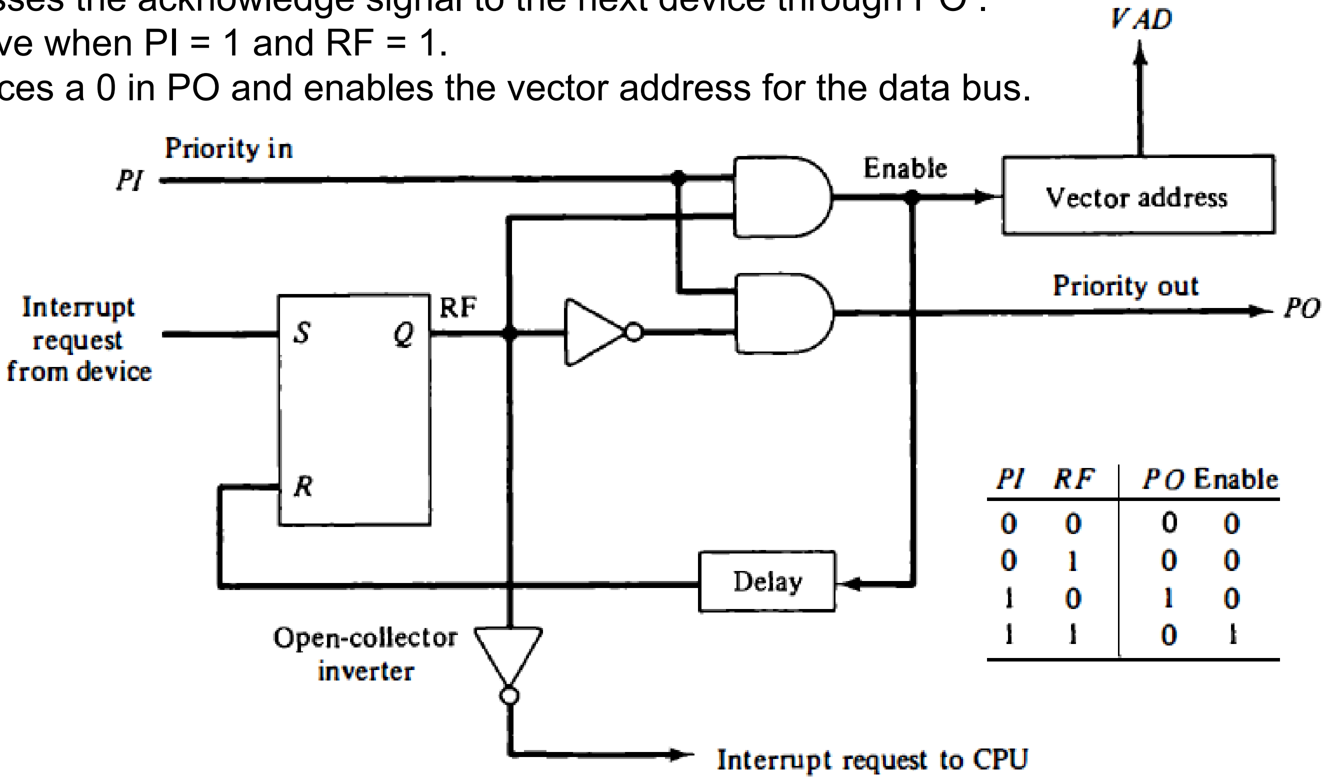
Device 2
Interrupt Request

"1"

"1"

"0"

- Figure shows the internal logic that must be included within each device when connected in the daisy-chaining scheme.
- The device sets its RF flip-flop when it wants to interrupt the CPU.
- The output of the RF flip-flop goes through an open-collector inverter, a circuit that provides the wired logic for the common interrupt line.
- If PI = 0, both PO and the enable line to VAD are equal to 0, irrespective of the value of RF.
- If PI = 1 and RF = 0, then PO = 1 and the vector address is disabled.
- This condition passes the acknowledge signal to the next device through PO .
- The device is active when PI = 1 and RF = 1.
- This condition places a 0 in PO and enables the vector address for the data bus.

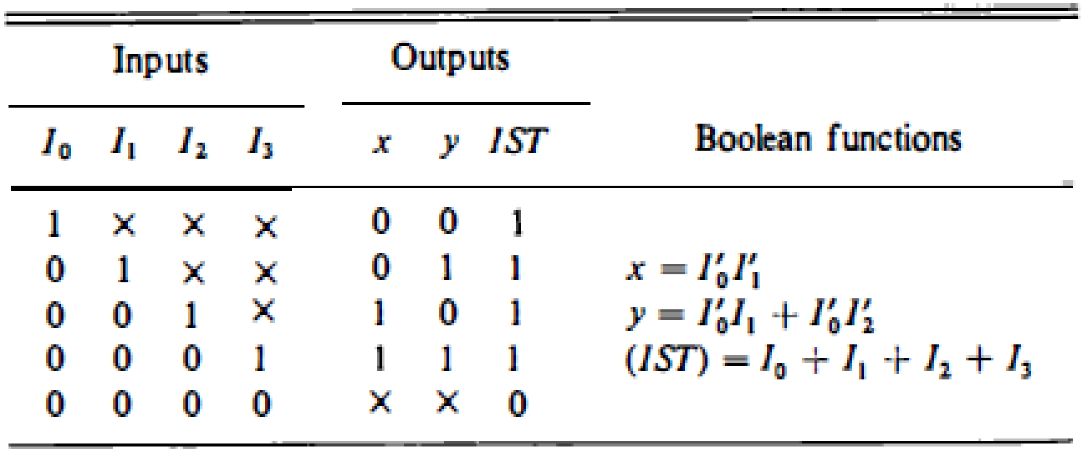


PI	*RF*	*PO*	Enable
0	0	0	0
0	1	0	0
1	0	1	0
1	1	0	1

Fig. One stage of the daisy chain priority arrangement

Parallel Priority

- Priority is established according to the position of the bits in the register.
- The mask register can be programmed to disable lower-priority interrupts while a higher-priority device is being serviced.
- Priority Encoder –The priority encoder is a circuit that implements the priority function.
- Parallel Priority :
  - Interrupt Enable F/F (**IEN**) : set or cleared by the program
  - Interrupt Status F/F (**IST**) : set or cleared by the encoder output
- Priority Encoder Truth Table :



Inputs				Outputs			
I_0	I_1	I_2	I_3	x	y	IST	Boolean functions
1	×	×	×	0	0	1	
0	1	×	×	0	1	1	$x = I_0' I_1'$
0	0	1	×	1	0	1	$y = I_0' I_1 + I_0' I_2'$
0	0	0	1	1	1	1	$(IST) = I_0 + I_1 + I_2 + I_3$
0	0	0	0	×	×	0	

Direct Memory Access (DMA)

- Transfer of data under programmed I/O is between CPU and peripheral.
- In direct memory access (DMA), the interface transfers data into and out of the memory unit through the memory bus.
- The CPU initiates the transfer by supplying the interface with the starting address and the number of words needed to be transferred and then proceeds to execute other tasks.

- DMA controller takes over the buses to manage the transfer directly between the **I/O device** and **memory** *(***Bus Request/Grant***)*
- When the transfer is made, the DMA requests memory cycles through the memory bus.
- When the request is granted by the memory controller, the DMA transfers the data directly into memory.

Direct Memory Access (DMA)

- Transfer Modes
  - 1) Burst transfer : Block
  - 2) Cycle stealing transfer : Byte
- DMA Controller (Intel 8237 DMAC) :
  - DMA Initialization Process
    - **1)** Set Address register :
      - » memory address for read/write
    - **2)** Set Word count register :
      - » the number of words to transfer
    - **3)** Set transfer mode :
      - » read/write,
      - » burst/cycle stealing,
      - » I/O to I/O,
      - » I/O to Memory,
      - » Memory to Memory
      - » Memory search
      - » I/O search
    - 4) DMA transfer start : *next section*
    - 5) EOT (End of Transfer) :
      - » Interrupt

Direct Memory Access (DMA)

⑤ DMA Transfer (I/O to Memory)

1) I/O Device sends a DMA request
2) DMAC activates the **BR** line
3) CPU responds with **BG** line
4) DMAC sends a DMA acknowledge
   to the I/O device
5) I/O device puts a word in the data
   bus (*for memory write*)
6) DMAC write a data to the address
   specified by **Address register**
7) Decrement **Word count register**
8) **Word count register**
   EOT interrupt CPU
9) **Word count register**
   DMAC checks the DMA request from
   I/O device

Input/Output Organisation

Outline

- Peripheral devices
- Input/Output Interface
- Asynchronous Data Transfer
- Modes Of Transfer
- Priority Interrupt
- DMA
- Input-Output Processor (IOP)
- Questions

Peripheral Devices

Peripheral Devices

- The input-output subsystem of a computer, referred to as I/O, provide communication between the central system and the outside environme must be entered into computer memory for processing and results obtai must be recorded or displayed for the user. A computer serves no use ability to receive information from an outside source and to translate form.

- The most familiar means of entering data into a computer is through a t that allows a person to enter alphanumeric information directly. Every ti terminal sends a binary coded character to the computer.

- The fastest possible speed for entering information depends on the perso other hand the CPU is extremely fast device capable of performing operat

Peripheral Devices

- Input or output devices that are connected to computer are called▪ **periph**
- For example:▪ *Keyboards*,▪ *display units*▪ and▪ *printers*▪ are common periphe

There are **three** types of peripherals:

- **Input peripherals▪** : Allows user input, from the outside world to th Keyboard, Mouse etc.
    - Keyboard, mouse, scanner, microphone, etc.,
  - **Keyboard:** A keyboard is an input device that allows users to enter text computer system.

Peripheral Devices

- **Mouse**: A mouse is an input device that allows users to control the curso

Scanner: A scanner is an input device that allows users to convert phy images into digital files.

Microphone: A microphone is an input device that allows users t

- **Output peripherals**: Allows information output, from the computer to the Printer, Monitor etc.,
  - Monitors, headphones, printers etc.

Monitor: A monitor is an output device that displays visual information fr

Peripheral Devices

- **Printer:** A printer is an output device that produces physical copies of c

Speaker: A speaker is an output device that produces a

- **Input-Output peripherals**: Allows both input(from outisde world to output(from computer to the outside world). Example: Touch screen etc.

Keyboard

- Input & Output devices that communicate with people and the computer the transfer of alphanumeric information to and from the device and the c
- The standard binary code for the Alphanumeric characters is ASCII…It characters.
- 94 printable and 34 non-printable characters
- The printing characters consists of 26 uppercase,26 lowercase,10 numeri printable characters such as %,*,$.
- 34 non printable characters are also called control characters
- 1) Format Selectors - are characters that controls the layou Backspace(BS),Horizontal Tabulation (HT),VT.
- 2) Communication Control Characters – Transmission of text betwe STX,ETX.
- 3) Information Seperators – used to separate data into paragraphs and pa

Input-Output Interface

Input-Output Interface

- Input-output interface provides a method for transferring information be and external I/O devices.
- Peripherals connected to a computer need special **communication lin** with the central processing unit.
- The purpose of the communication link is to resolve the differences central computer and each peripheral.
- The major differences are:
- 1. Peripherals are electromechanical and electromagnetic devices and th is different from the operation of the CPU and memory, which are electron conversion of signal values may be required.
- 2. The data transfer rate of peripherals is usually slower than the transf consequently, a synchronization mechanism may be needed.

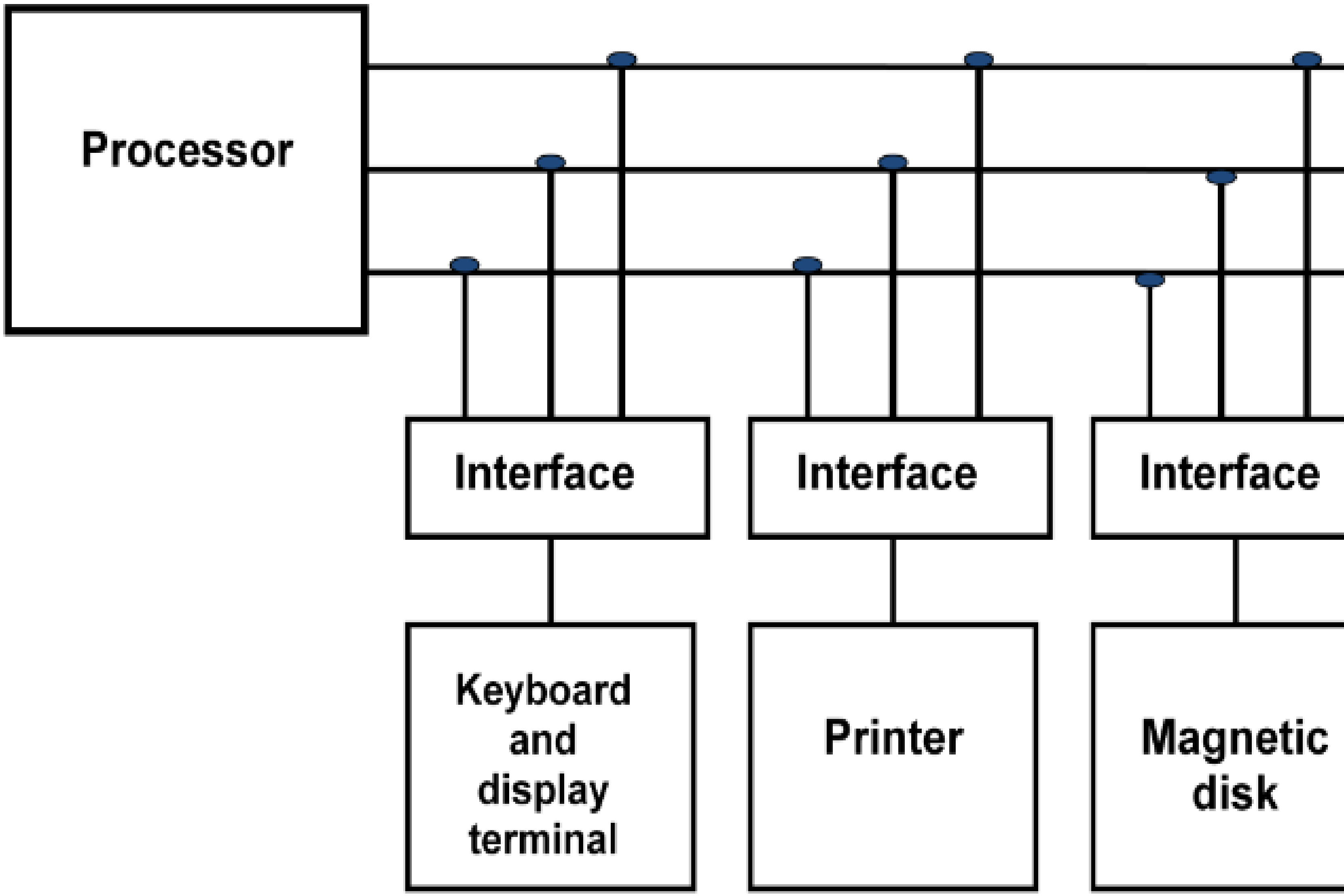
Input-Output Interface

- 3. Data codes and formats in peripherals differ from the word format in th
- 4. The operating modes of peripherals are different from each other and so as not to disturb the operation of other peripherals connected to the CP
- To resolve these differences, computer systems include special hardwar the CPU and peripherals to supervise and synchronize all input and output
- These components are called **interface** units because they interface bet and the peripheral device.
- In addition, each device may have its own controller that supervises particular mechanism in the peripheral.
- **I /O Bus and Interface Modules :** The I/O bus consists of data lines, ad lines. Each peripheral device has associated with it an interface unit. Eac address and control received from the I/O bus, interprets them for the signals for the peripheral controller.

Input-Output Interface

- It also synchronizes the data flow and supervises the transfer between peri
- Each peripheral has its own controller that operates the particular electro example, the printer controller controls the paper motion, the print timin printing characters.
- The I/O bus from the processor is attached to all peripheral interfaces. T particular device, the processor places a device address on the address lin
- Each interface attached to the I/O bus contains an address decoder that lines. When the interface detects its own address, it activates the path bet the device that it controls. All peripherals whose address does not corres the bus are disabled by their interface.

I/O Bus and Interface



Input-Output Interface

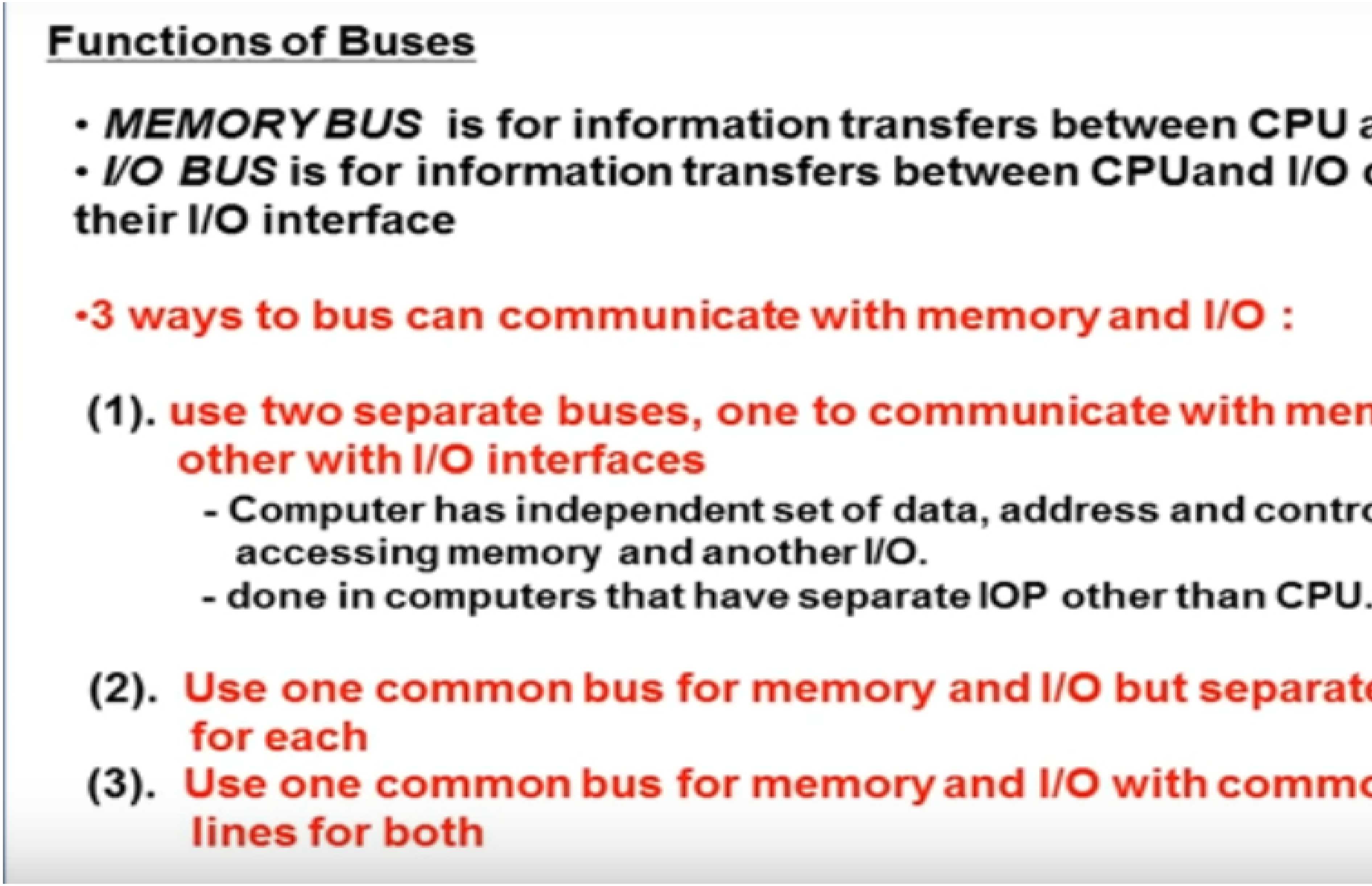
- At the same time that the address is made available In the address lines, a function code in the control lines. The interface selected responds to proceeds to execute it.
- The function code is referred to as **an I/O command.**
- The interpretation of the command depends on the peripheral that the pro
- There are **four types** of commands that an interface may receive. **They a status, data output, and data input.**
- **A control command :** is issued to activate the peripheral and to inform it w
- For example, a **magnetic tape** unit may be instructed to backspace the rewind the tape, or to start the tape moving in the forward direction. command issued depends on the peripheral, and each peripheral receive sequence of control commands, depending on its mode of operation.

Commands

- **A status command** is used to test various status conditions in the interfac
- For example, the computer may wish to check the status of the periph initiated.
- **A data output command :** causes the interface to respond by transferring one of its registers.
- Consider an example with a tape unit. The computer starts the tape mov command. The processor then monitors the status of the tape by means c
- When the tape is in the correct position, the processor issues a data interface responds to the address and command and transfers the info lines in the bus to its buffer register. The interface then communicates and sends the data to be stored on tape.

- **The data input command :** is the opposite of the data output.
- In this case the interface receives an item of data from the peripheral ar register.
- The processor checks if data are available by means of a status comman input command. The interface places the data on the data lines, where tl processor.

I/O BUS AND MEMORY BUS



Functions of Buses

- *MEMORY BUS* is for information transfers between CPU a
- *I/O BUS* is for information transfers between CPUand I/O c
their I/O interface

- 3 ways to bus can communicate with memory and I/O :

(1). use two separate buses, one to communicate with mem
other with I/O interfaces

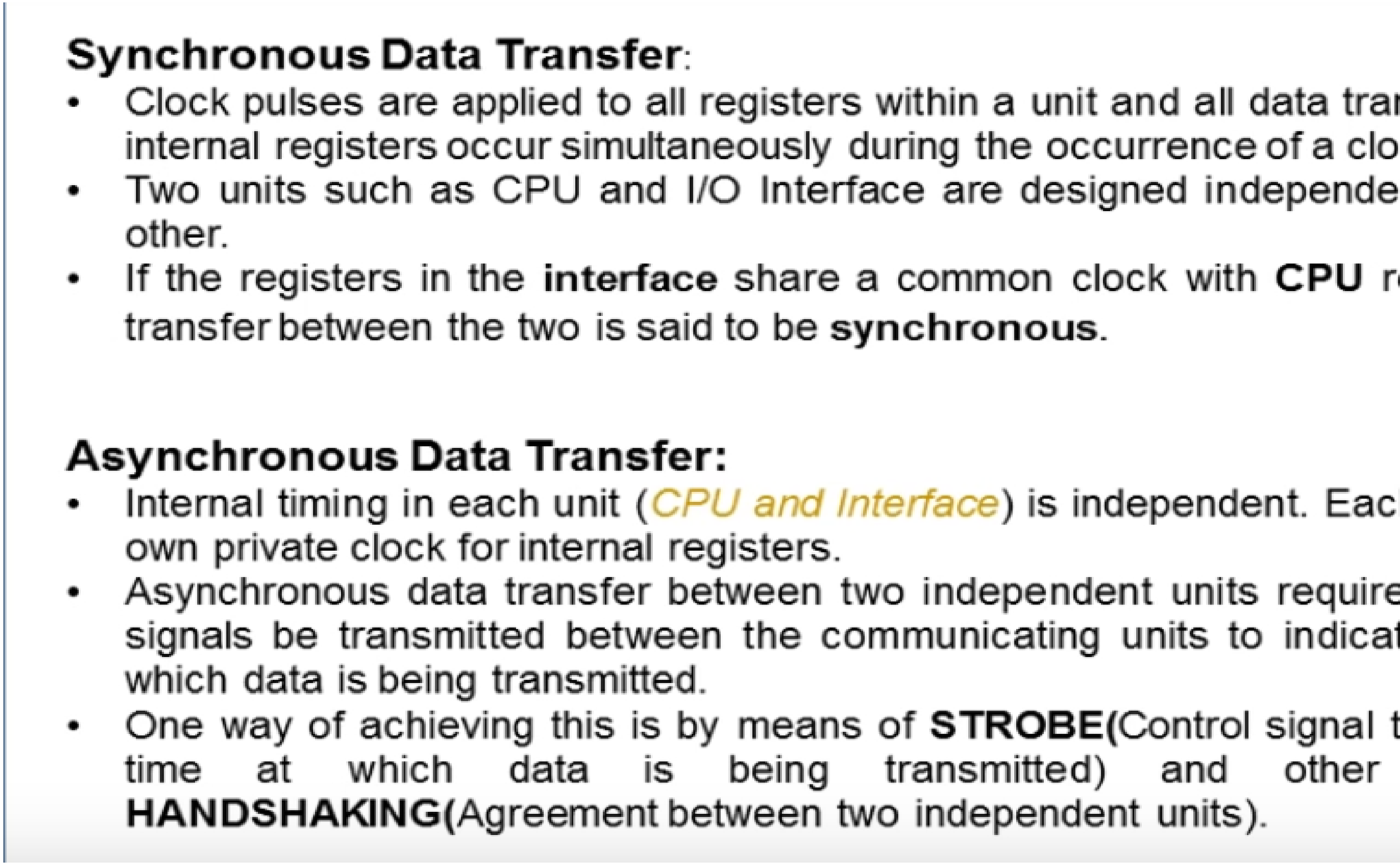
- Computer has independent set of data, address and contro
accessing memory and another I/O.
- done in computers that have separate IOP other than CPU.

(2). Use one common bus for memory and I/O but separate
for each

(3). Use one common bus for memory and I/O with comm
lines for both

Asynchronous Data Transfer

Asynchronous Data Transfer



Synchronous Data Transfer:

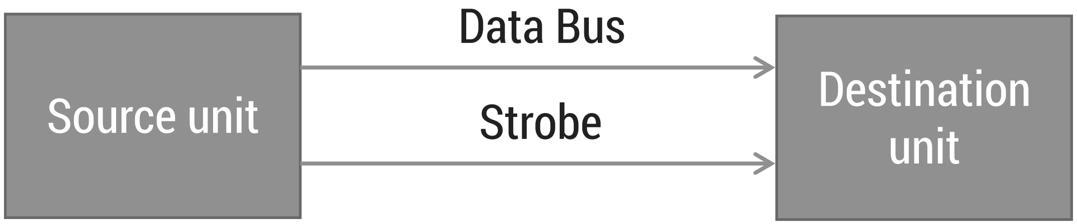
- Clock pulses are applied to all registers within a unit and all data tra internal registers occur simultaneously during the occurrence of a clo
- Two units such as CPU and I/O Interface are designed independe other.
- If the registers in the **interface** share a common clock with **CPU** r transfer between the two is said to be **synchronous**.

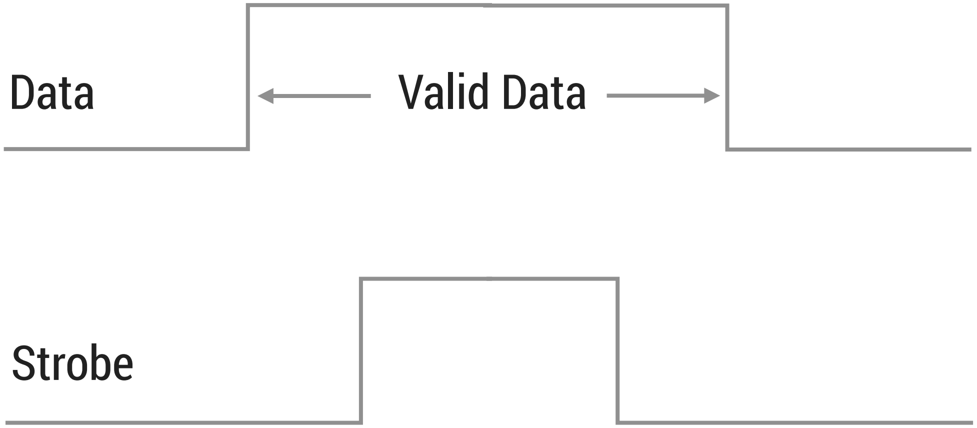
Asynchronous Data Transfer:

- Internal timing in each unit (*CPU and Interface*) is independent. Eac own private clock for internal registers.
- Asynchronous data transfer between two independent units require signals be transmitted between the communicating units to indica which data is being transmitted.
- One way of achieving this is by means of **STROBE(**Control signal t time at which data is being transmitted) and other **HANDSHAKING(**Agreement between two independent units).

Strobe Method

1.1 Source initiated Strobe

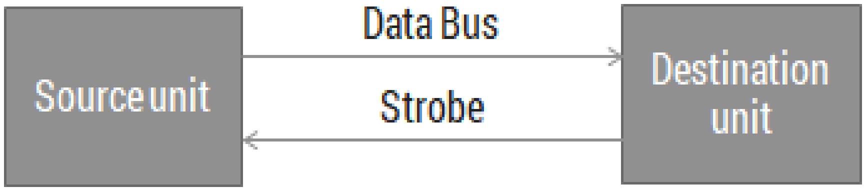


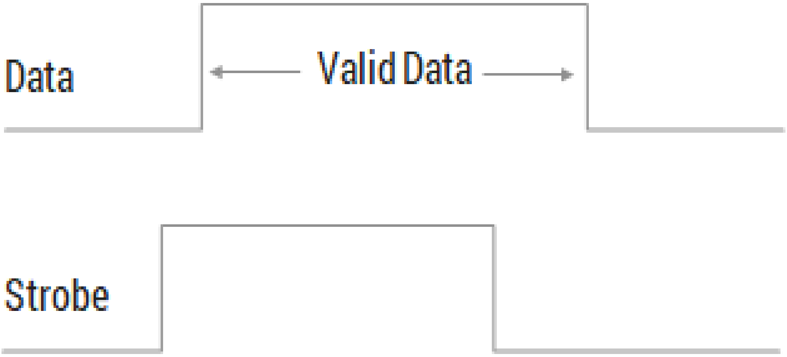


When source initiat
data transfer. Strob
(i) First, source put
bus and ON the stro
(ii) Destination on s
signal of strobe, rea
data bus.
(iii) After reading da
bus by destination,

It shows that first data is put on the data b
signal gets active.

1.2 Destination initiated Strobe





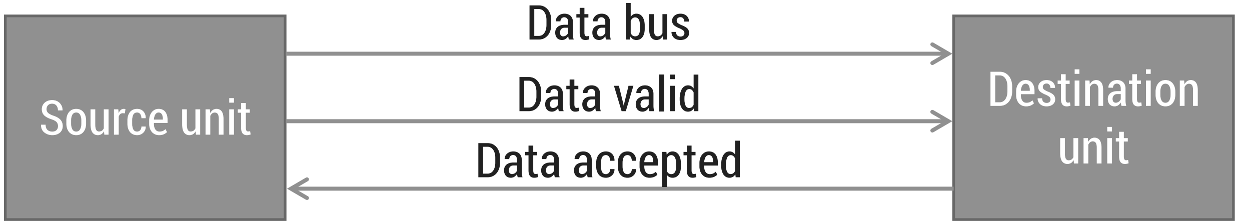
(▪ i)First, the destination ON the
ensure the source to put the fre
data bus.
(ii) Source on seeing the ON sig
on the data bus.
(iii) Destination reads the data
and strobe gets OFF signal.

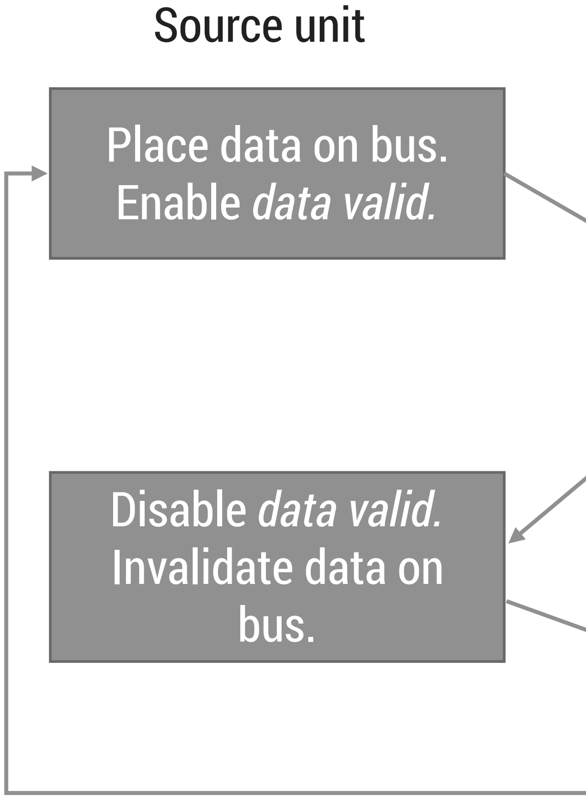
It shows that first strobe signal
data is put on the data bus.

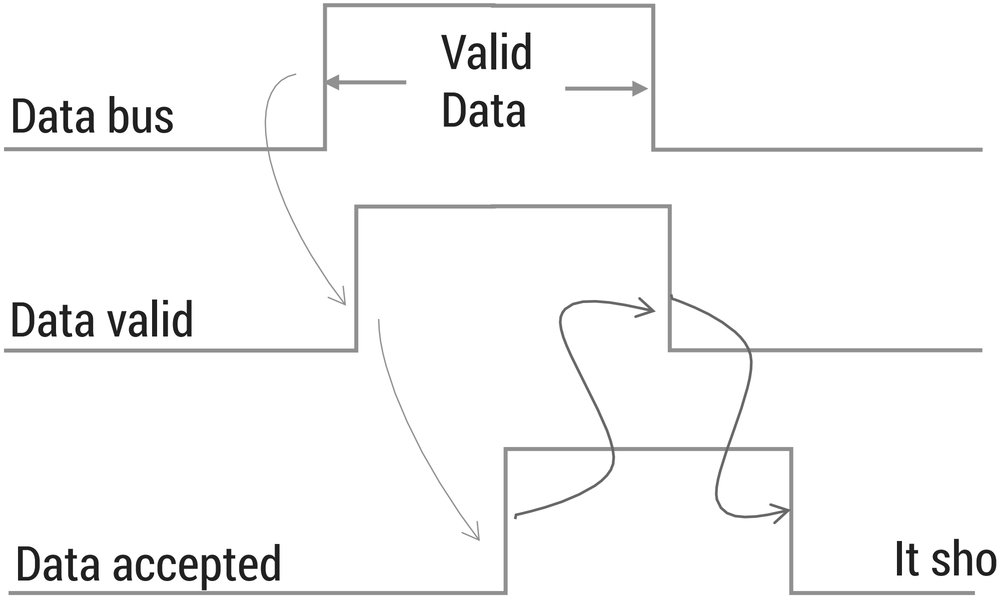
Disadvantage

- In Source initiated Strobe, it is assumed that destination has read the dat their is no surety.
- In Destination initiated Strobe, it is assumed that source has put the da their is no surety.

2.1 Source initiated Handshake

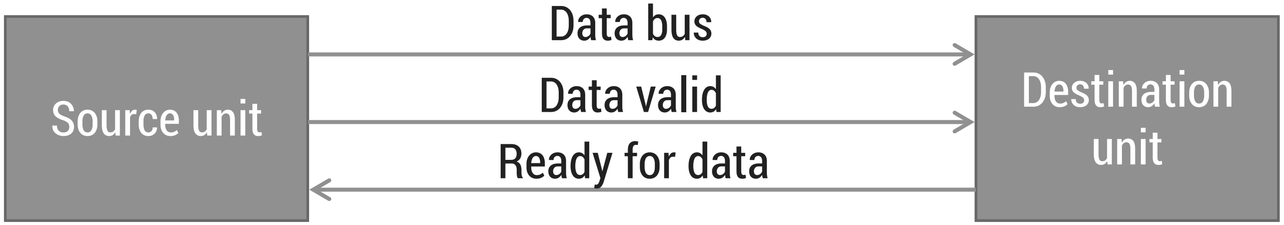


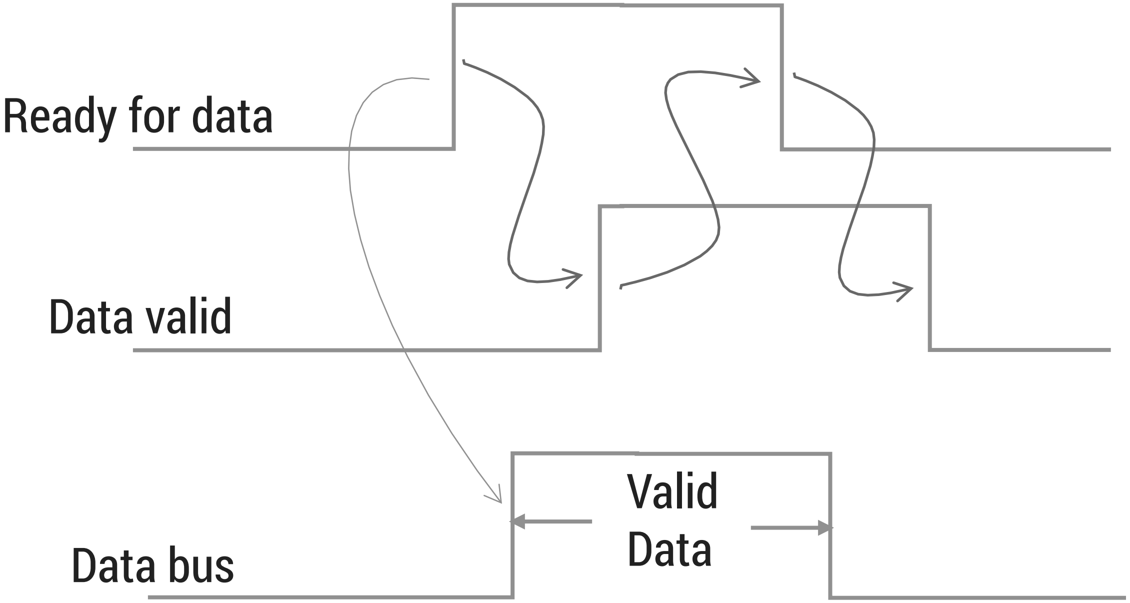

Source unit
Place data on bus. Enable *data valid.*
Disable *data valid.* Invalidate data on bus.

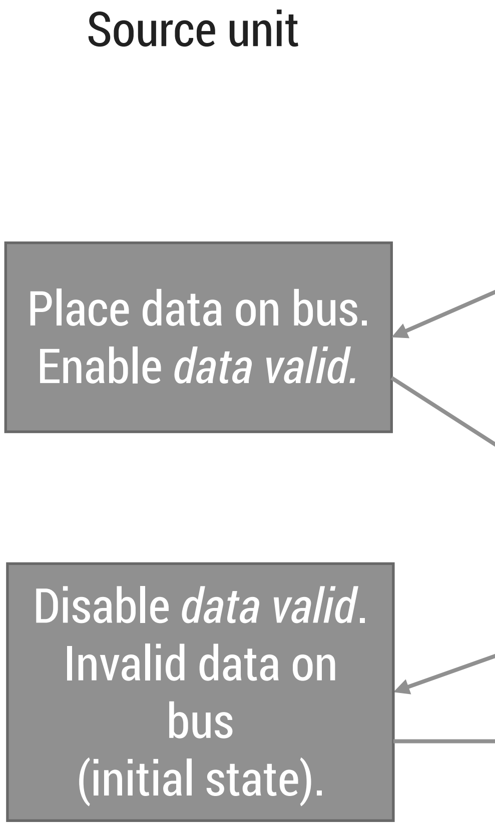


It shows that first data is put on the data bus th gets active and then data accepted signal gets accepting the data, first data valid signal gets c accepted signal gets off.

2.2 Destination initiated Handshake






Source unit
Place data on bus. Enable *data valid.*
Disable *data valid*. Invalid data on bus (initial state).

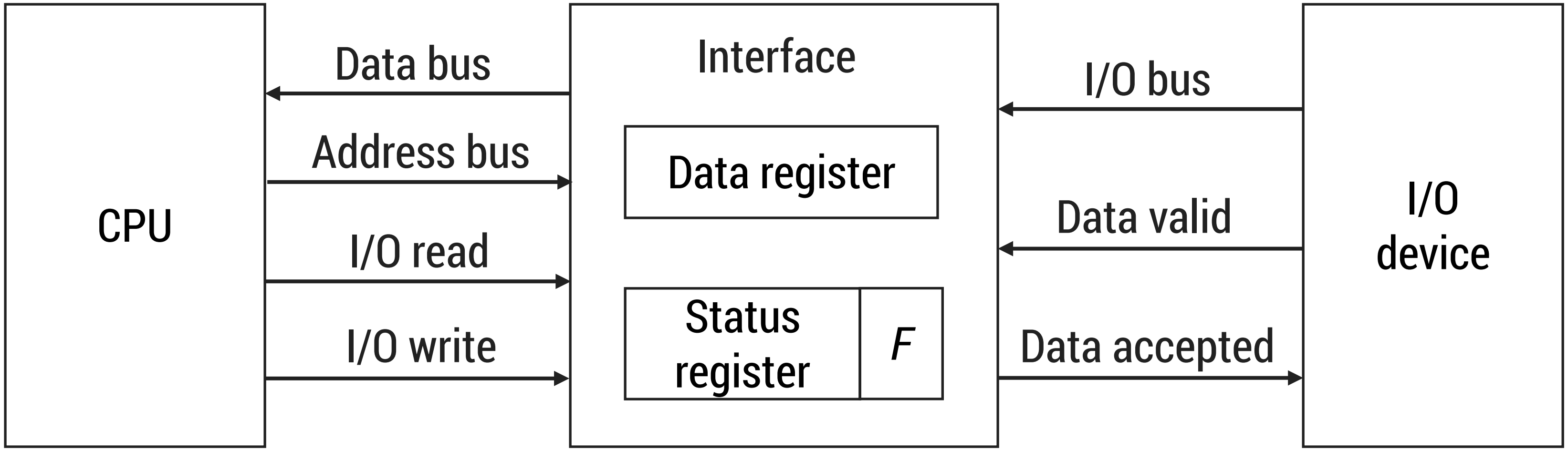
It shows that first Request for Data sig
put on data bus then Data valid signal
data, first Request for Data signal gets

Modes Of Transfer

Modes of Transfer

- Data transfer between the central computer and I/O devices may be modes.
- Some modes use the CPU as an intermediate path; others transfer the d the memory unit.
- Data transfer to and from peripherals may be handled in one of three poss
  1. Programmed I/O
  2. Interrupt-initiated I/O
  3. Direct memory access (DMA)

Programmed I/O



CPU

Data bus

Address bus

I/O read

I/O write

Interface

Data register

Status register

F

I/O bus

Data valid

Data accepted

I/O device

Programmed I/O

- It is due to the result of the I/O instructions that are written in the comput
- Each data item transfer is initiated by an instruction in the program.▪
- In this case it requires constant monitoring by the CPU of the peripheral d
- Example
- In this case, the I/O device does not have direct access to the memory uni
- A transfer from I/O device to memory requires the execution of several i including an input instruction to transfer the data from device to the CPU transfer the data from CPU to memory.
- In programmed I/O, the CPU stays in the program loop until the I/O unit i for data transfer.
- This is a time consuming process since it needlessly keeps the CPU bus avoided by using an interrupt facility.
-

Interrupt-initiated I/O

- An alternative to the CPU constantly monitoring the flag is to let th computer when it is ready to transfer data.
- While the CPU is running a program, it does not check the flag.
- However, when the flag is set, the computer is momentarily interrupte current program and is informed of the fact that the flag has been set.
- The CPU deviates from what it is doing to take care of the input or output
- After the transfer is completed, the computer returns to the previous prog was doing before the interrupt.

Priority Interrupt

polling

- what if multiple devices generate interrupts simultaneously. In that case, to decide which interrupt is to be serviced first. In other words, we have to the devices for systemic interrupt servicing.
- Polling is the software method of establishing priority of simultaneous int
- Establishes priority over the various sources to determine which conditio

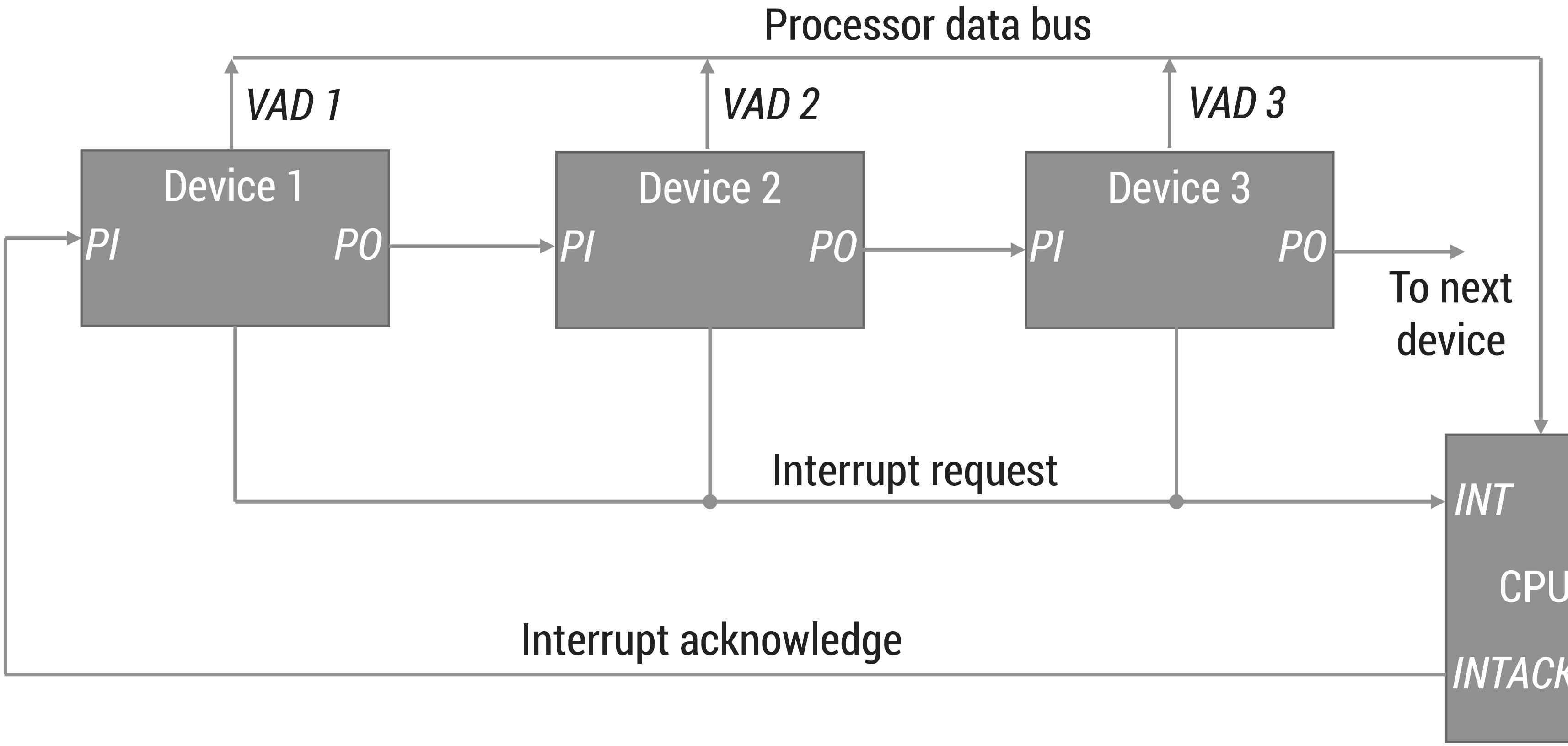
when two or more requests arrive simultaneously.

- •Highest priority source is tested first and if its interrupt signal is on, cont

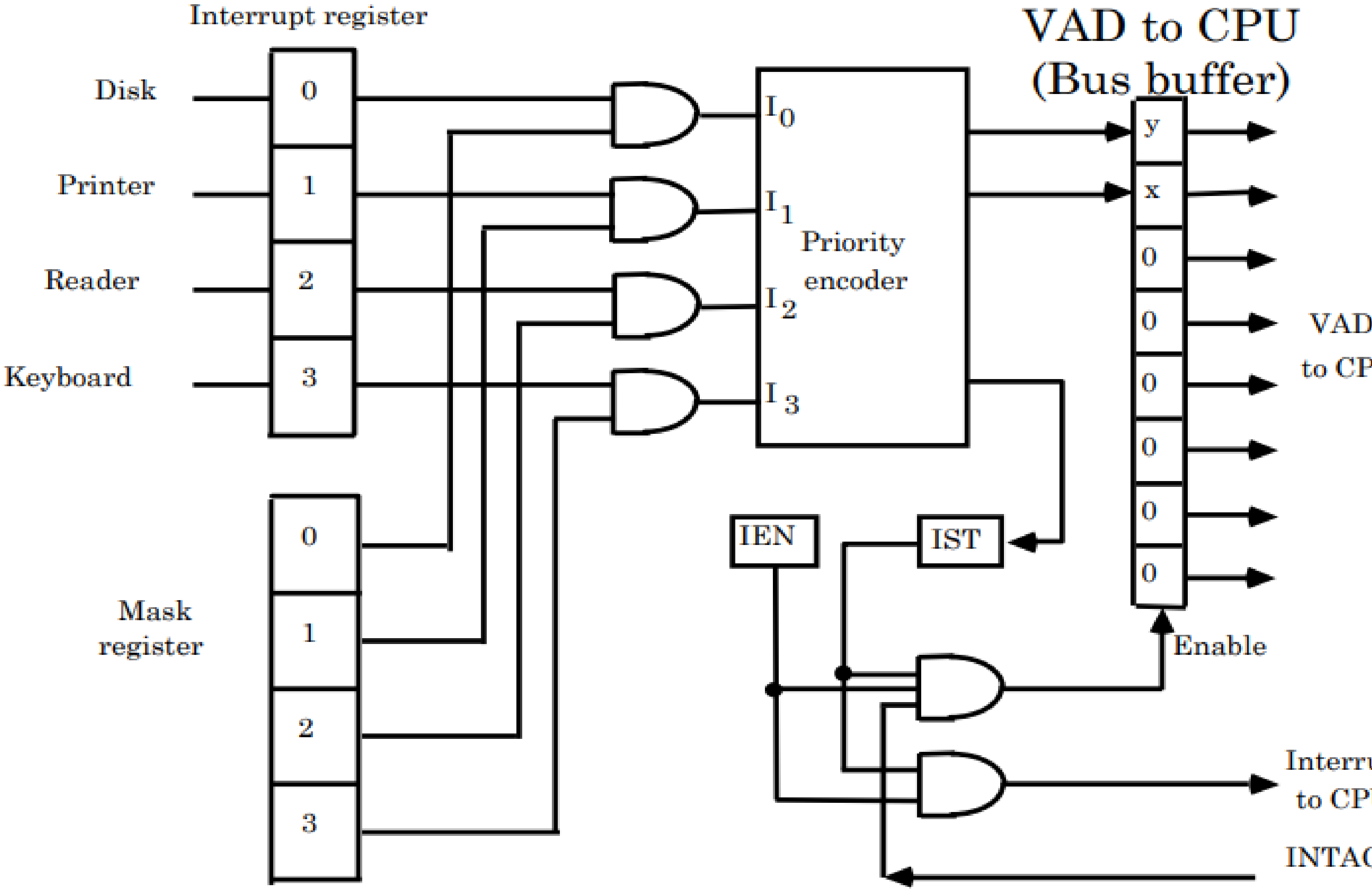
service routine for this source

Priority Interrupt (Daisy-Chaining Technique)

- Determines which interrupt is to be served first when two or more requests are m
- Also determines which interrupts are permitted to interrupt the computer while a
- Higher priority interrupts can make requests while servicing a lower priority inter



Parallel priority



Priority encoder

- Circuit that implements the priority function.
- Logic – if two or more inputs arrive at the same tin having the highest priority will take precedence.

Inputs			
I_0	I_1	I_2	I_3
1	d	d	d
0	1	d	d
0	0	1	d
0	0	0	1
0	0	0	0

Outpu	
d	Y
0	0
0	1
1	0
1	1
d	d

- The output of the priority encoder is used to form part of vector add source.

Interrupt cycle

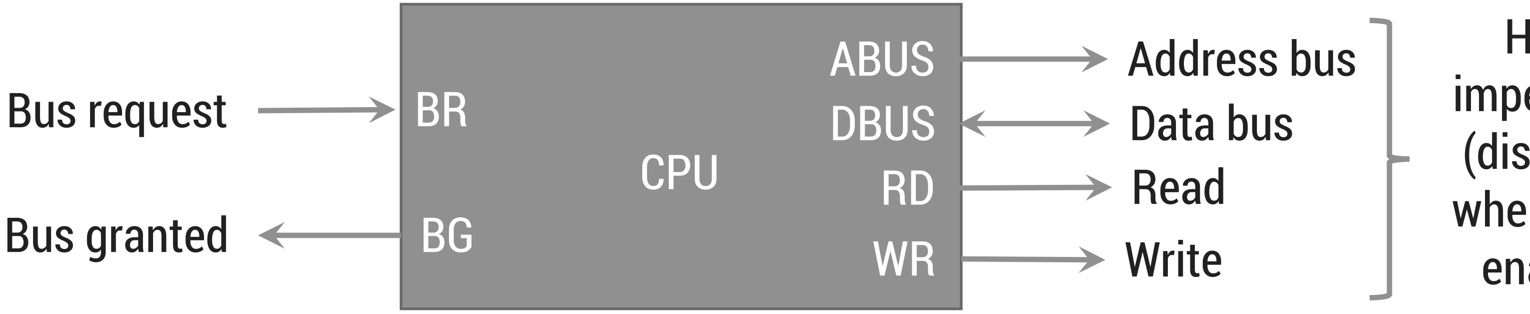
- The Interrupt enable flip-flop (IEN) can be set or cleared by program instr
- A programmer can therefore allow interrupts (clear IEN) or disallow interr
- At the end of each instruction cycle the CPU checks IEN and IST. If control continues with the next instruction. If both = 1, the interrupt is ha
- Interrupt micro-operations:
  - SP✱SP – 1 (Decrement stack pointer)
  - M[SP] ✱ PC Push PC onto stack
  - INTACK ✱ 1 Enable interrupt acknowledge
  - PC ✱VAD Transfer vector address to PC
  - IEN ✱0 Disable further interrupts
  - Go to fetch next instruction

DMA (Direct Memory Access

Section - 4

DMA (Direct Memory Access)

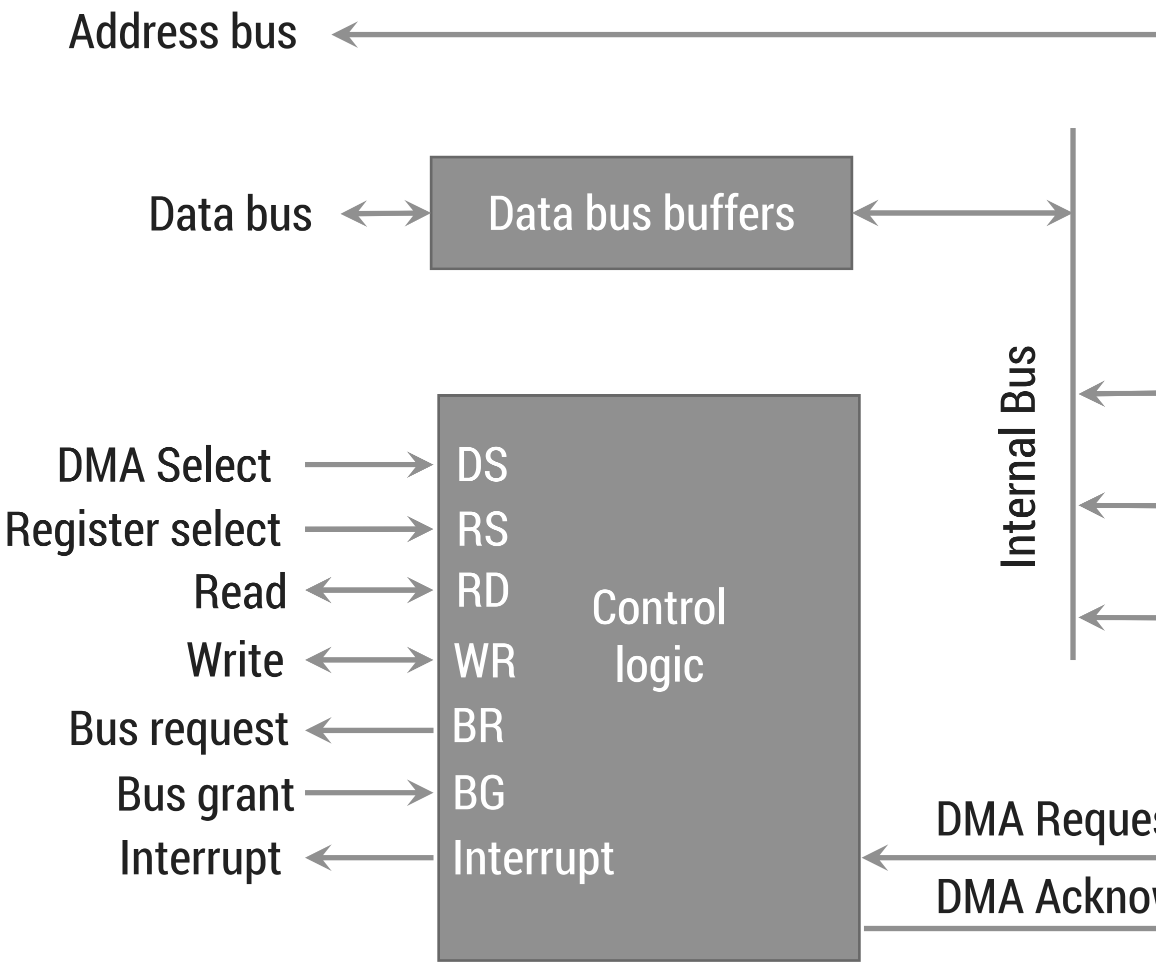
- The transfer of data between a fast storage device such as magnetic di limited by the speed of the CPU.
- Removing the CPU from the path and letting the peripheral device mar directly would improve the speed of transfer.
- This transfer technique is called direct memory access (DMA).
- During DMA, CPU is idle and has no control of the memory buses.
- A DMA controller takes over the buses to manage the transfer directly bet memory.

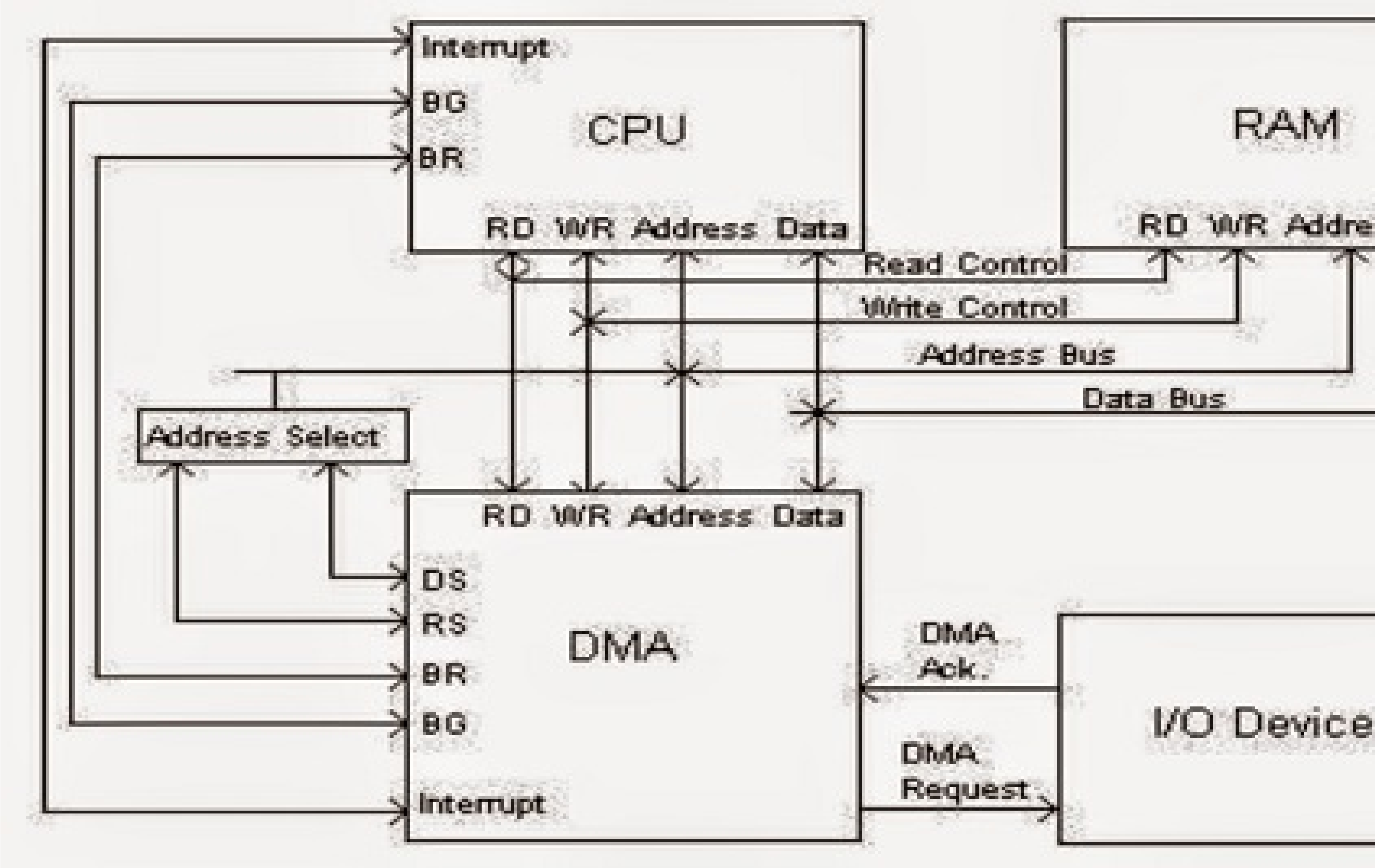


DMA Controller

DMA Controller

- DMA controller - Interface which allows I/O transfer directly between Memory and Device, freeing CPU for other tasks
- CPU initializes DMA Controller by sending memory address and the block size (number of words).




Interrupt
BG
BR
CPU
RD W/R Address Data
RAM
RD W/R Addre
Read Control
Write Control
Address Bus
Data Bus
Address Select
RD W/R Address Data
DS
RS
DMA
BR
BG
Interrupt
DMA Ack.
I/O Device
DMA Request

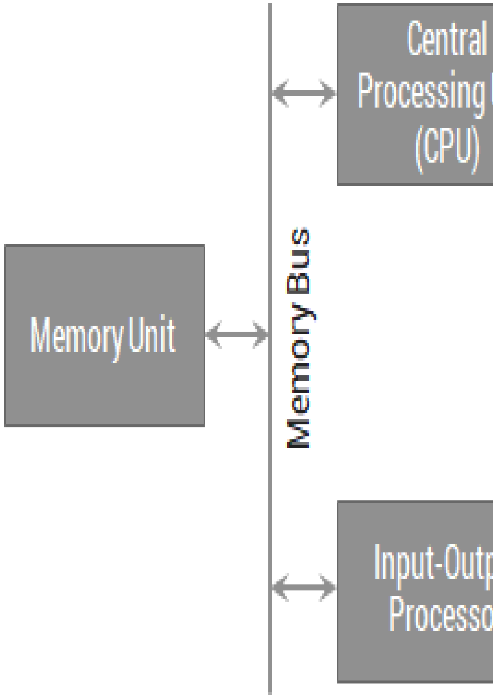
Input-Output Processor (IOP

Section - 5

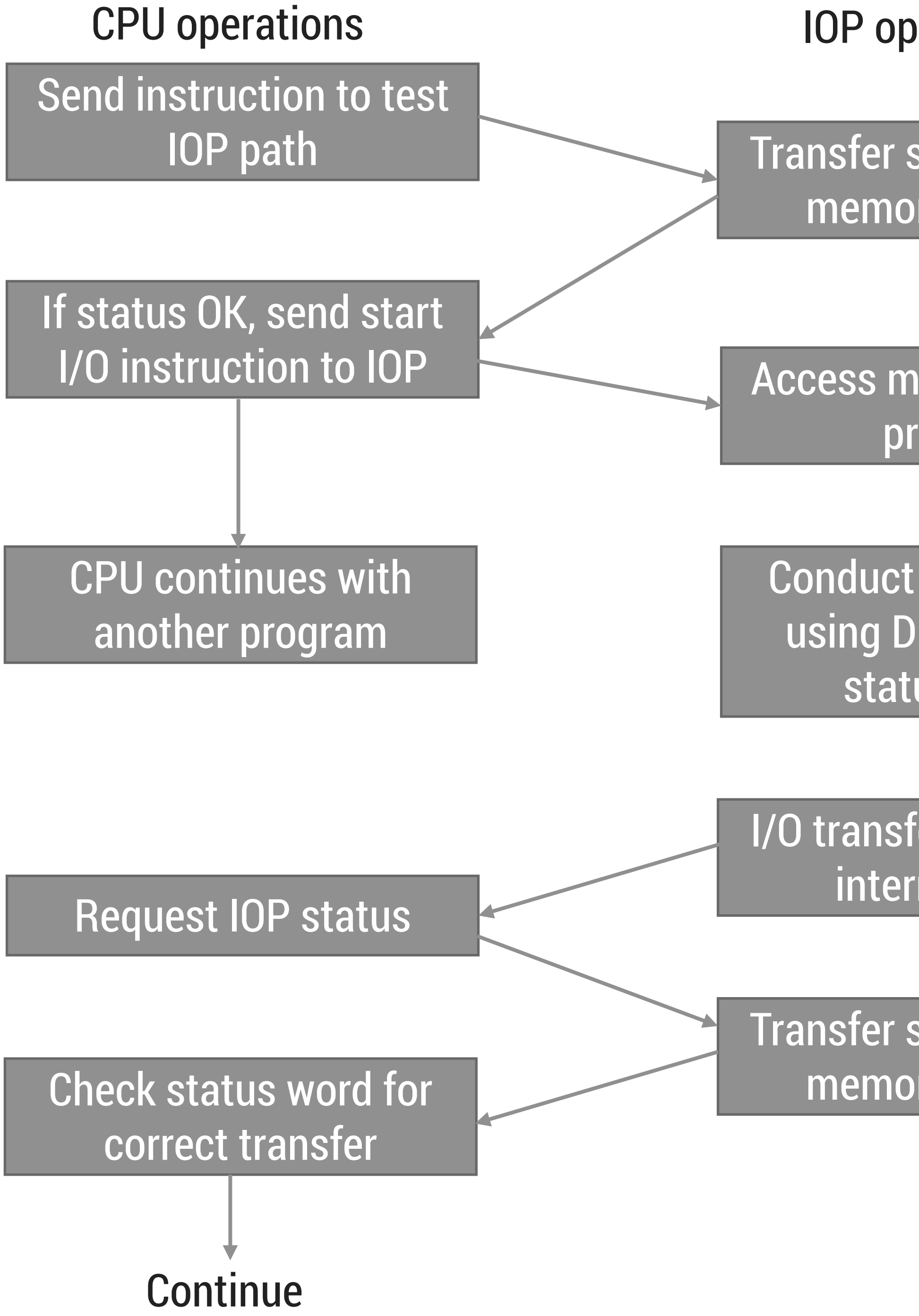
IOP

- An input-output processor (IOP) is a processor with direct memory access
- The IOP is similar to CPU except that it is designed to handle only I/O proc
- The IOP fetches and executes I/O instructions to facilitate I/O transfer
- Both CPU and IOP exists in the system however CPU is master,while IOP is
- The CPU only initiates the I/O program after that IOP operates independen
- The I/O processor is capable of performing actions without interruption CPU. The CPU only needs to initiate the I/O processor by telling it what a the necessary actions are performed, the I/O processor then provides the

- Memory occupies the central position and can communicate with each processor. •
- CPU is responsible for processing data. •
- IOP provides the path for transfer of data between various peripheral devices and memory. •
- Data formats of peripherals differ from CPU and memory. IOP maintain such problems. •
- Data are transfer from IOP to memory by stealing one memory cycle. •
- Instructions that are read from memory by IOP are called commands to distinguish them from instructions that are read by the CPU



CPU – IOP Communication



CPU operations

IOP op

Send instruction to test IOP path

Transfer s memo

If status OK, send start I/O instruction to IOP

Access m pr

CPU continues with another program

Conduct using D statu

I/O transf inter

Request IOP status

Transfer s memo

Check status word for correct transfer

Continue

Questions

Section - 6

Questions

1. Explain daisy chain priority interrupt.
2. Explain the DMA operation.
3. What is the use of IOP? Explain its communication with CPU.
4. Explain asynchronous data transfer using timing diagrams.
5. Differentiate isolated I/O and memory mapped I/O.
6. Differentiate Programmed I/O and Interrupt initiated I/O.
7. What are the advantages of Serial Data Transmission of data?
8. Briefly explain source initiated transfer using handshaking.
9. Enlist possible modes of data transfer to and from peripherals.

Input/Output Organisation

Outline

- Peripheral devices
- Input/Output Interface
- Asynchronous Data Transfer
- Modes Of Transfer
- Priority Interrupt
- DMA
- Input-Output Processor (IOP)
- Questions

Peripheral Devices

Peripheral Devices

- The input-output subsystem of a computer, referred to as I/O, provide communication between the central system and the outside environme must be entered into computer memory for processing and results obtai must be recorded or displayed for the user. A computer serves no use ability to receive information from an outside source and to translate form.

- The most familiar means of entering data into a computer is through a t that allows a person to enter alphanumeric information directly. Every ti terminal sends a binary coded character to the computer.

- The fastest possible speed for entering information depends on the perso other hand the CPU is extremely fast device capable of performing operat

Peripheral Devices

- Input or output devices that are connected to computer are called▪ **periph**
- For example:▪ *Keyboards*,▪ *display units*▪ and▪ *printers*▪ are common periphe

There are **three** types of peripherals:

- **Input peripherals▪** : Allows user input, from the outside world to th Keyboard, Mouse etc.
  - Keyboard, mouse, scanner, microphone, etc.,
  - **Keyboard:** A keyboard is an input device that allows users to enter text computer system.

Peripheral Devices

- **Mouse**: A mouse is an input device that allows users to control the cursor

Scanner: A scanner is an input device that allows users to convert phy
images into digital files.

Microphone: A microphone is an input device that allows users t

- **Output peripherals**: Allows information output, from the computer to the
Printer, Monitor etc.,
  - Monitors, headphones, printers etc.

Monitor: A monitor is an output device that displays visual information fr

Peripheral Devices

- **Printer:** A printer is an output device that produces physical copies of c

Speaker: A speaker is an output device that produces a

- **Input-Output peripherals**: Allows both input(from outisde world to output(from computer to the outside world). Example: Touch screen etc.

Keyboard

- Input & Output devices that communicate with people and the compute the transfer of alphanumeric information to and from the device and the c
- The standard binary code for the Alphanumeric characters is ASCII…It characters.
- 94 printable and 34 non-printable characters
- The printing characters consists of 26 uppercase,26 lowercase,10 numeri printable characters such as %,*,$.
- 34 non printable characters are also called control characters
- 1) Format Selectors - are characters that controls the layou Backspace(BS),Horizontal Tabulation (HT),VT.
- 2) Communication Control Characters – Transmission of text betwe STX,ETX.
- 3) Information Seperators – used to separate data into paragraphs and p

Input-Output Interface

Input-Output Interface

- Input-output interface provides a method for transferring information be and external I/0 devices.
- Peripherals connected to a computer need special **communication lin** with the central processing unit.
- The purpose of the communication link is to resolve the differences central computer and each peripheral.
- The major differences are:
- 1. Peripherals are electromechanical and electromagnetic devices and th is different from the operation of the CPU and memory, which are electro conversion of signal values may be required.
- 2. The data transfer rate of peripherals is usually slower than the transf consequently, a synchronization mechanism may be needed.

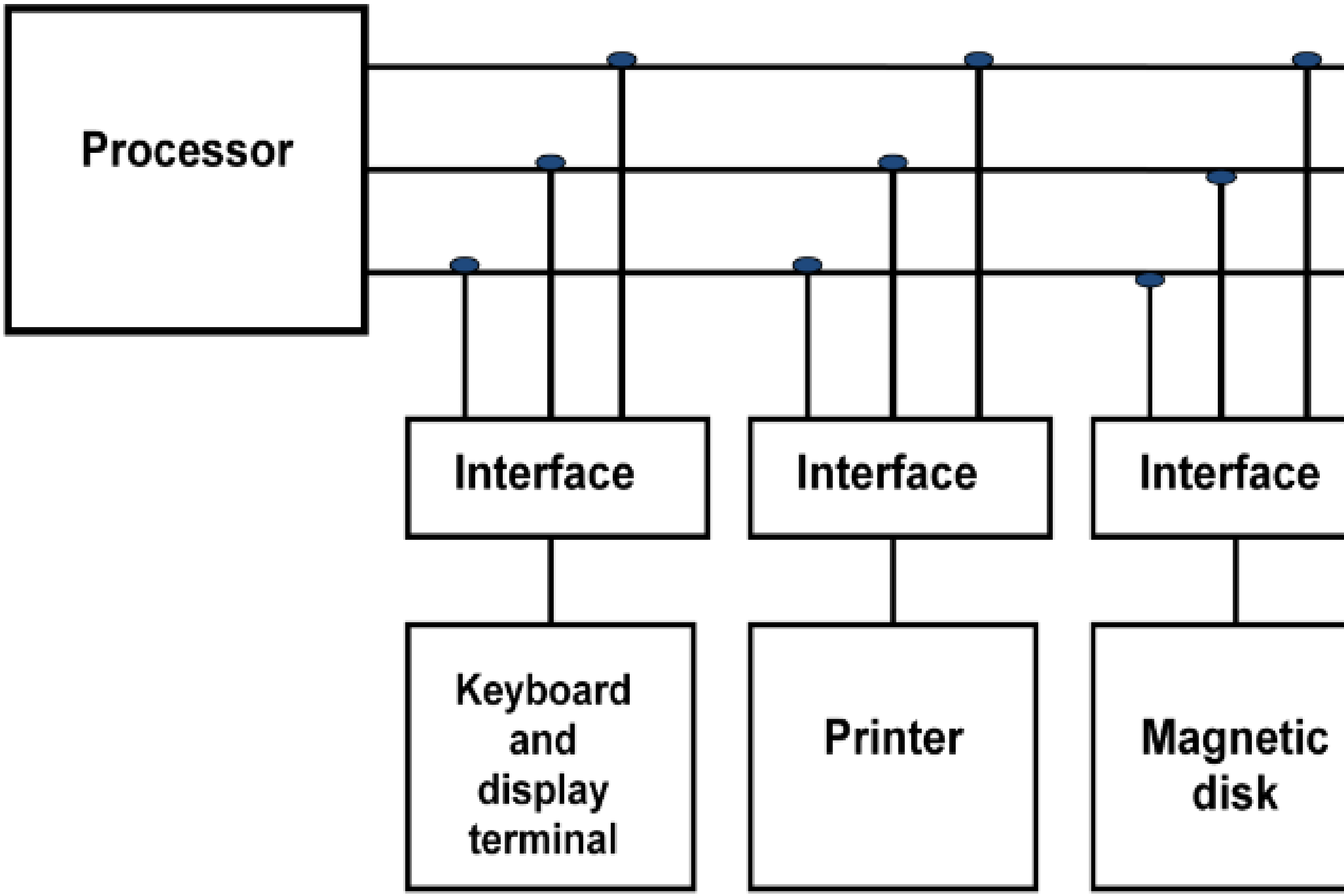
Input-Output Interface

- 3. Data codes and formats in peripherals differ from the word format in th
- 4. The operating modes of peripherals are different from each other and so as not to disturb the operation of other peripherals connected to the CF
- To resolve these differences, computer systems include special hardwar the CPU and peripherals to supervise and synchronize all input and output
- These components are called **interface** units because they interface bet and the peripheral device.
- In addition, each device may have its own controller that supervises particular mechanism in the peripheral.
- **I /O Bus and Interface Modules :** The I/O bus consists of data lines, ad lines. Each peripheral device has associated with it an interface unit. Eac address and control received from the I/O bus, interprets them for the signals for the peripheral controller.

Input-Output Interface

- It also synchronizes the data flow and supervises the transfer between peri
- Each peripheral has its own controller that operates the particular electro example, the printer controller controls the paper motion, the print timin printing characters.
- The I/O bus from the processor is attached to all peripheral interfaces. T particular device, the processor places a device address on the address lin
- Each interface attached to the I/O bus contains an address decoder that lines. When the interface detects its own address, it activates the path bet the device that it controls. All peripherals whose address does not corres the bus are disabled by their interface.

I/O Bus and Interface



Input-Output Interface

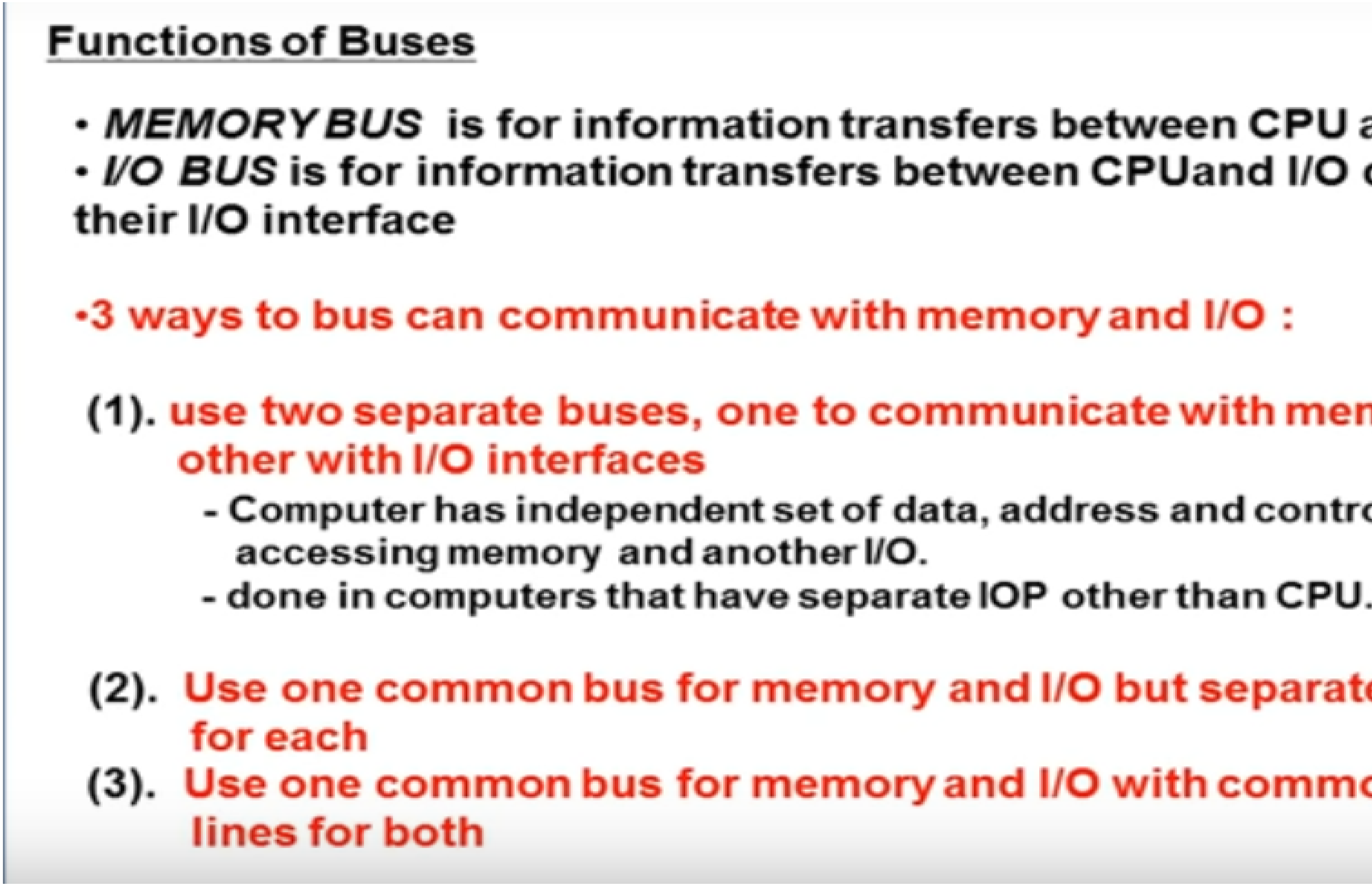
- At the same time that the address is made available In the address lines, a function code in the control lines. The interface selected responds to proceeds to execute it.
- The function code is referred to as **an I/O command.**
- The interpretation of the command depends on the peripheral that the pro
- There are **four types** of commands that an interface may receive. **They a status, data output, and data input.**
- **A control command :** is issued to activate the peripheral and to inform it w
- For example, a **magnetic tape** unit may be instructed to backspace the rewind the tape, or to start the tape moving in the forward direction. command issued depends on the peripheral, and each peripheral receive sequence of control commands, depending on its mode of operation.

Commands

- **A status command** is used to test various status conditions in the interfac
- For example, the computer may wish to check the status of the periph initiated.
- **A data output command :** causes the interface to respond by transferring one of its registers.
- Consider an example with a tape unit. The computer starts the tape mov command. The processor then monitors the status of the tape by means o
- When the tape is in the correct position, the processor issues a data interface responds to the address and command and transfers the inf lines in the bus to its buffer register. The interface then communicates and sends the data to be stored on tape.

- **The data input command :** is the opposite of the data output.
- In this case the interface receives an item of data from the peripheral ar register.
- The processor checks if data are available by means of a status comman input command. The interface places the data on the data lines, where t processor.

I/O BUS AND MEMORY BUS



Functions of Buses

- *MEMORY BUS* is for information transfers between CPU a
- *I/O BUS* is for information transfers between CPUand I/O d
their I/O interface

•3 ways to bus can communicate with memory and I/O :

(1). use two separate buses, one to communicate with mem
other with I/O interfaces

- Computer has independent set of data, address and contro
accessing memory and another I/O.
- done in computers that have separate IOP other than CPU.

(2). Use one common bus for memory and I/O but separate
for each

(3). Use one common bus for memory and I/O with comm
lines for both