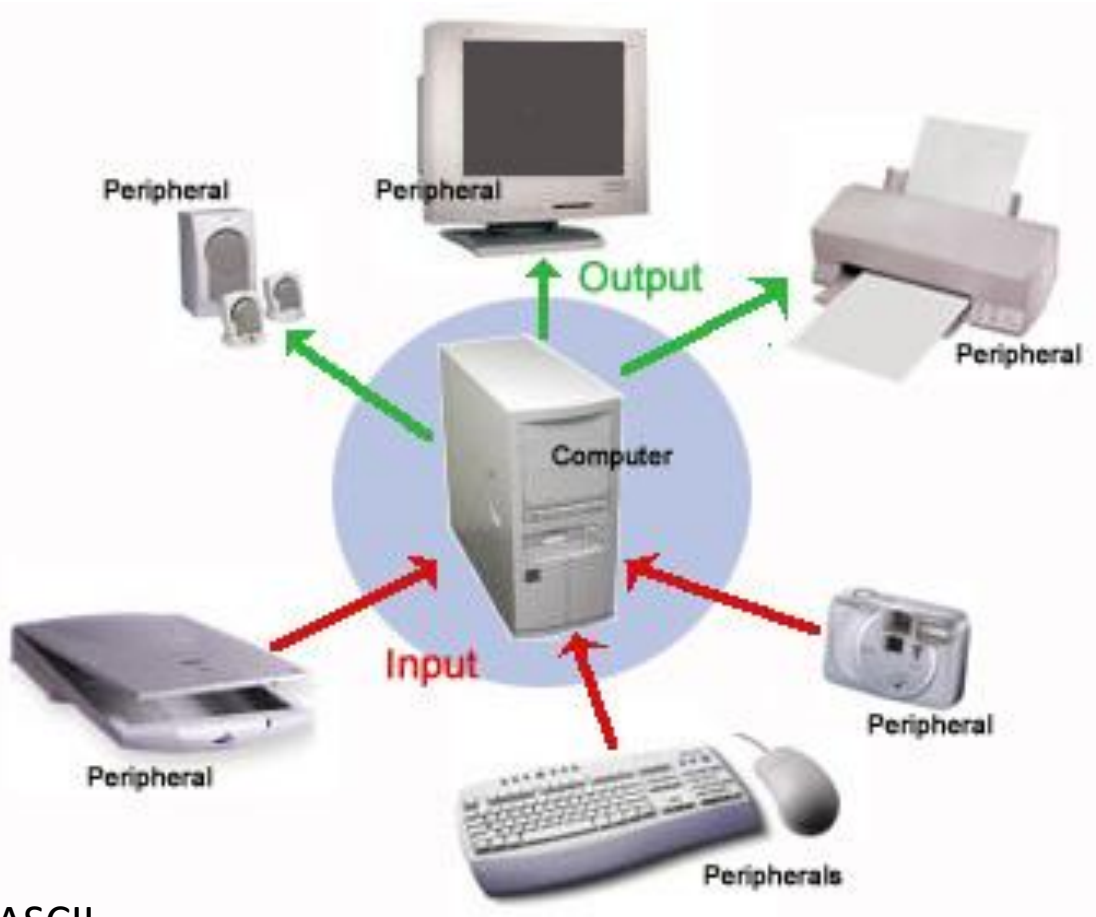
Unit – 2(a)
Input-Output Organization

Input - Output Organization

- Peripheral Devices
- Input - Output Interface
- Asynchronous Data Transfer
- Modes Of Transfer
- Priority Interrupt
- DMA

Peripheral Devices

- The input-output subsystem of a computer – I/O – Provides an efficient mode of communication between the central System and the outside environment
  - A computer serves no useful purpose without the ability to receive information from an outside source & to transmit results in a meaningful form
  - Input or output devices attached to the computer/CPU are called ***PERIPHERALS***



I/O Communication involves Transfer of alphanumeric Info

– ***The standard binary code*** - ASCII

Eg:

41 - A

61 - a

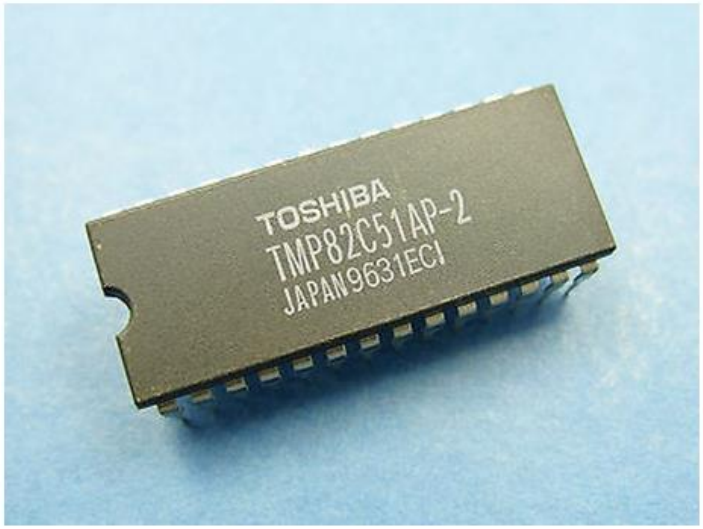
USASCII code chart

$b_7 b_6 b_5$ → / Bits	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4 b_3 b_2 b_1 ↓ / Column → / Row ↓	0	1	2	3	4	5	6	7
0 0 0 0	0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC4	$	4	D	T	d	t
0 1 0 1	5	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	6	ACK	SYN	&	6	F	V	f	v
0 1 1 1	7	BEL	ETB	'	7	G	W	g	w
1 0 0 0	8	BS	CAN	(	8	H	X	h	x
1 0 0 1	9	HT	EM	)	9	I	Y	i	y
1 0 1 0	10	LF	SUB	*	:	J	Z	j	z
1 0 1 1	11	VT	ESC	+	;	K	[	k	{
1 1 0 0	12	FF	FS	,	<	L	\	l	|
1 1 0 1	13	CR	GS	-	=	M	]	m	}
1 1 1 0	14	SO	RS	.	>	N	^	n	~
1 1 1 1	15	SI	US	/	?	O	_	o	DEL

Input - Output Interface

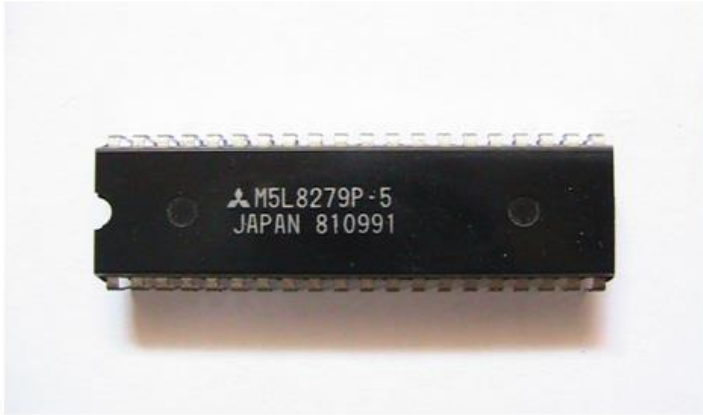
Proces

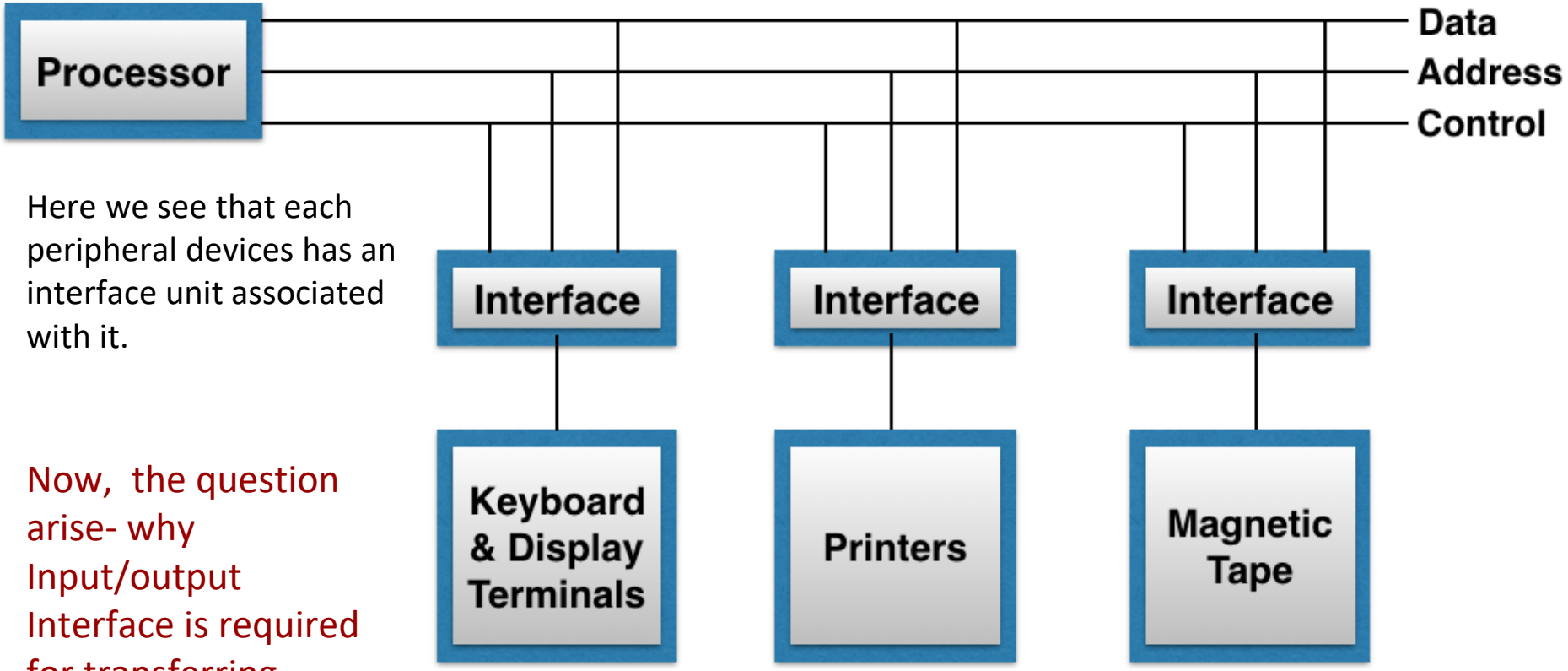
Here we s
peripheral
interface u
with it.



Data
Address
Control

Now, the
arise- wh
Input/ou
Interface
for transf
informati
processo
peripher





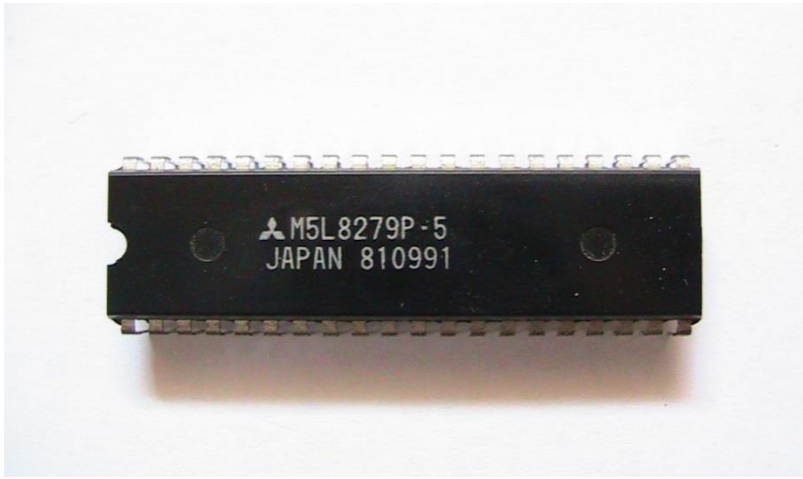
Here we see that each peripheral devices has an interface unit associated with it.

Now, the question arise- why Input/output Interface is required for transferring information between processor and peripherals?

Connection of I/O Bus to I/O Device


TOSHIBA
TMP82C51AP-2
JAPAN9631ECI


S/B8327
INS8255N
P8255


M5L8279P-5
JAPAN 810991


P8253-5
© INTEL 1976
0033
IC
PHIL

The answer is:

The Input/output Interface is required because there are exists many differences between the central computer and each peripheral while transferring information. Some major differences are:

- Peripherals are electromechanical and electromagnetic devices and their manner of operation is different from the operation of CPU and memory, which are electronic device. Therefore, a conversion of signal values may be required.

- The data transfer rate of peripherals is usually slower than the transfer rate of CPU, and consequently a synchronisation mechanism is needed.

- Data codes and formats in peripherals differ from the word format in the CPU and Memory.

- The operating modes of peripherals are differ from each other and each must be controlled so as not to disturb the operation of other peripherals connected to CPU.

These differences are resolved through input-output interface. As input-output interface(Interface Unit) contain various components, each of which performs one or more vital function for smooth transforming of information between CPU and Peripherals

IO Commands

Control

Status

Output Data

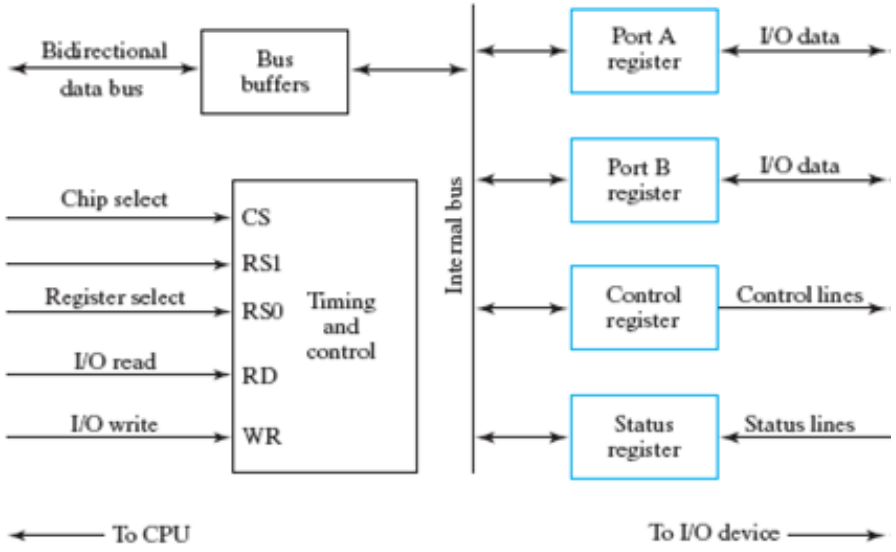
Input data

OUT 50,ALH

IN AL,51H

OUT 52,FFH

IN AL,52H



Example of a I/O interface unit

CS	RS1	RS0	Register selected
0	X	X	None: data bus in high-impedance state
1	0	0	Port A register
1	0	1	Port B register
1	1	0	Control register
1	1	1	Status register

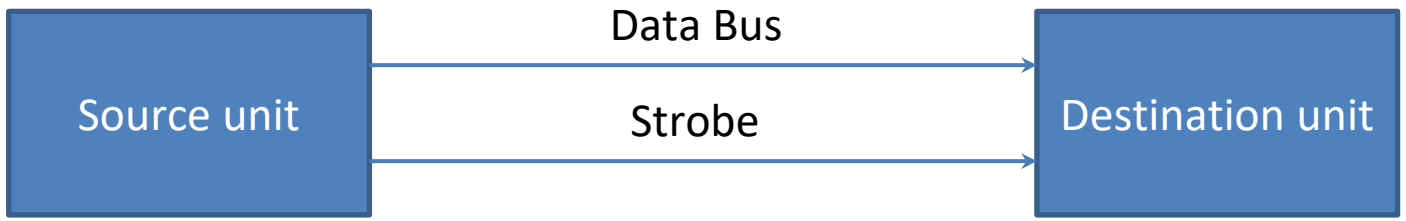
Asynchronous Data Transfer

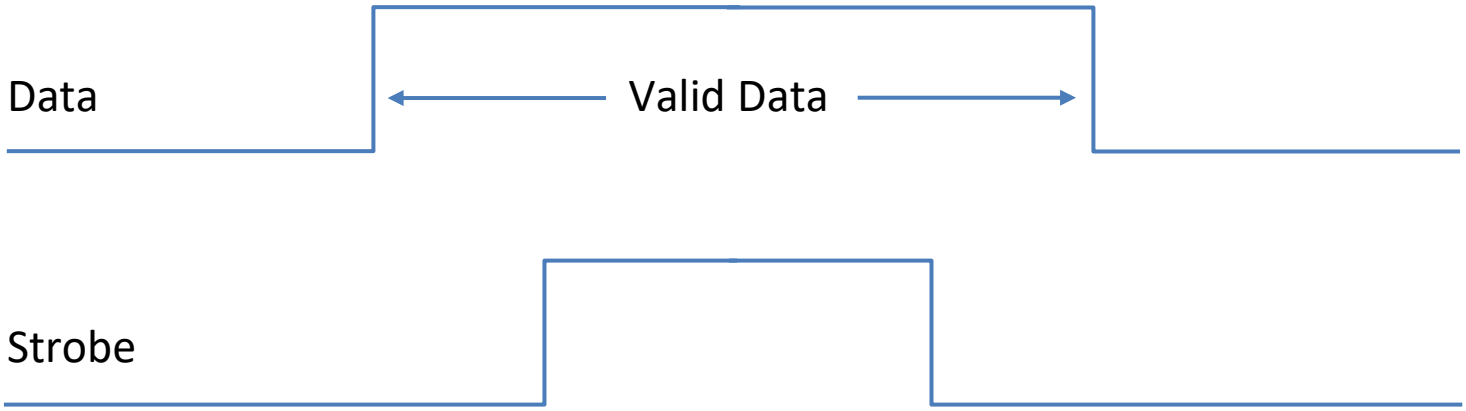
Asynchronous Data Transfer

- 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.
- Two ways of achieving
  1. Strobe
  2. Handshaking

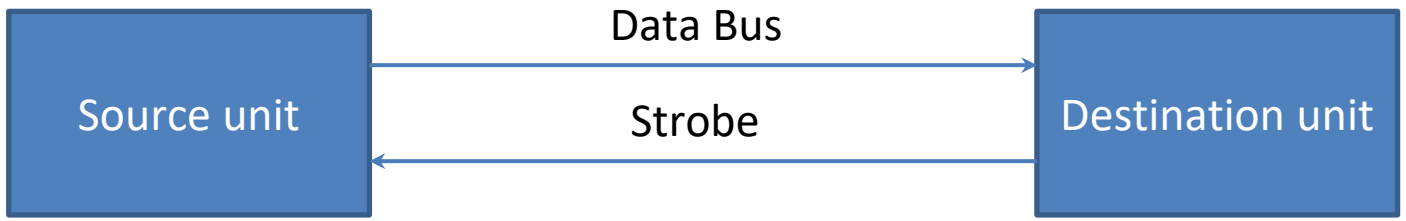
http://coagarage.blogspot.com/p/coa-asynchronous-data-transfer.html

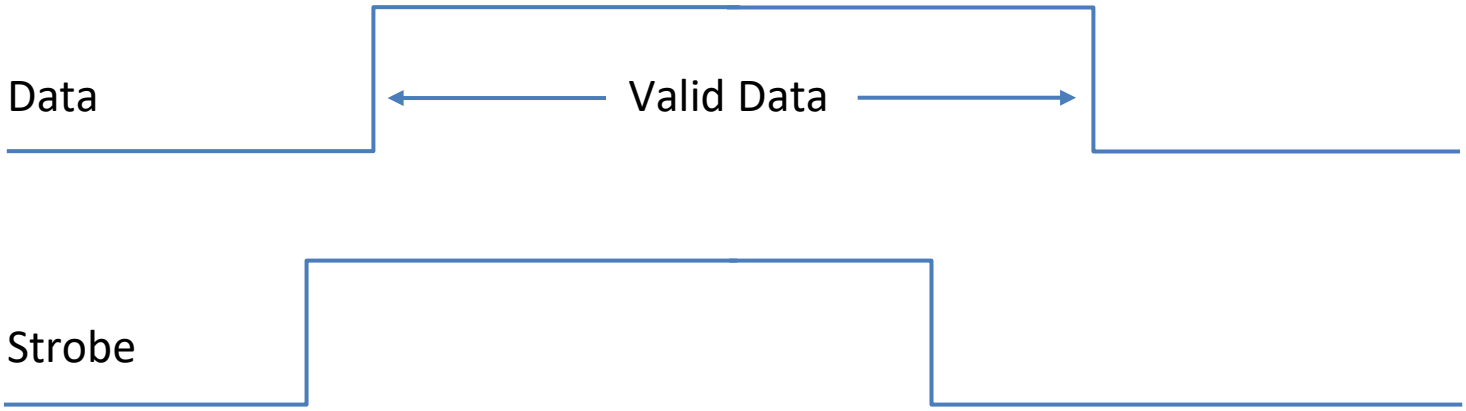
1.1 Source initiated Strobe



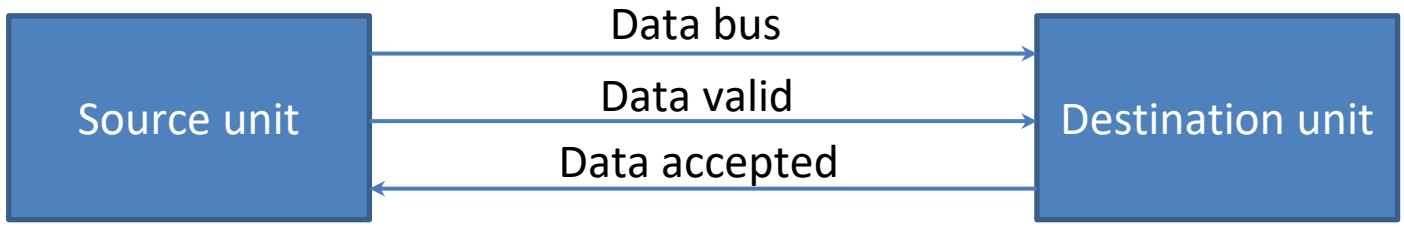


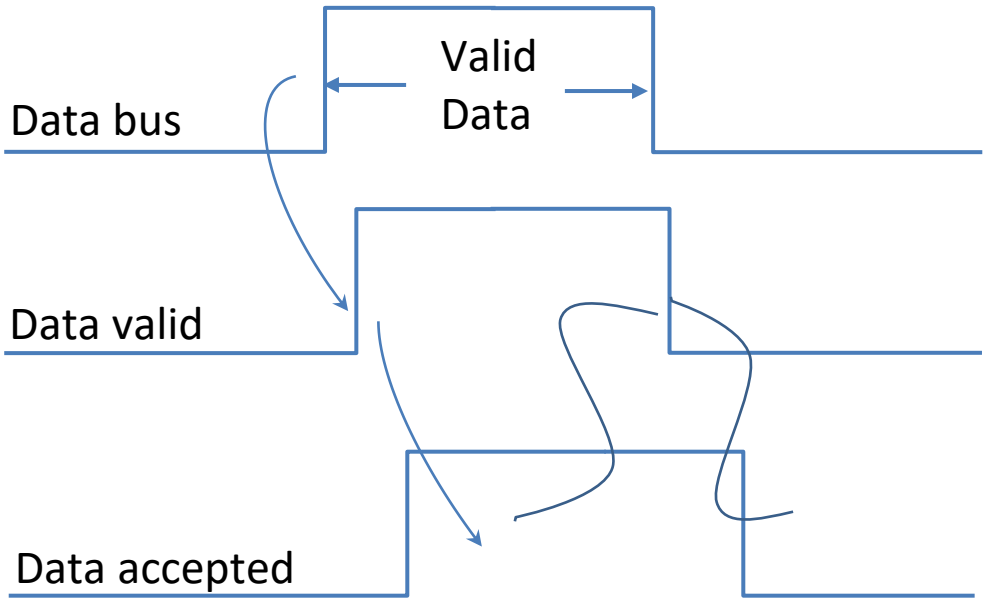
1.2 Destination initiated Strobe



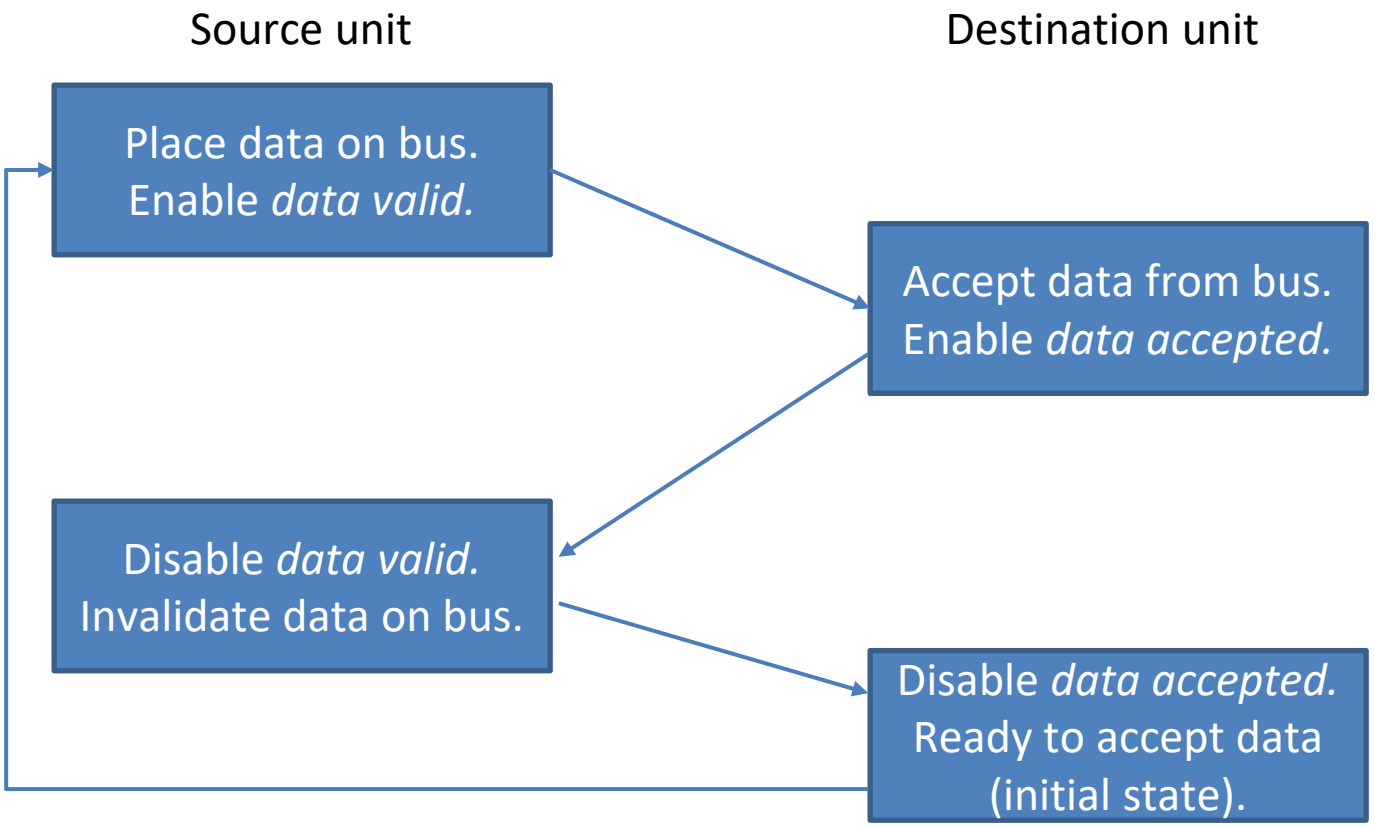


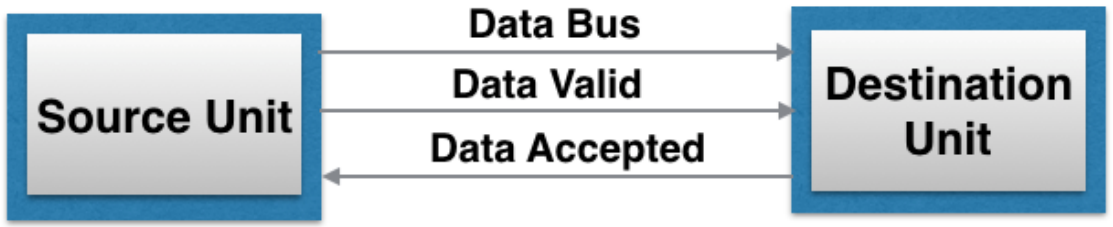
2.1 Source initiated Handshake



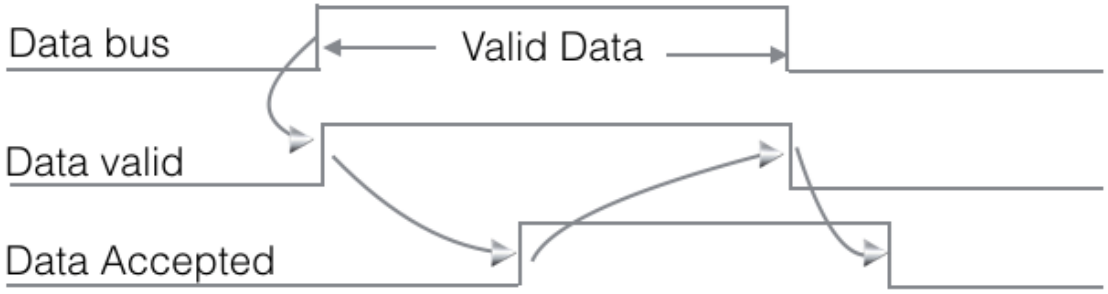


2.1 Source initiated Handshake

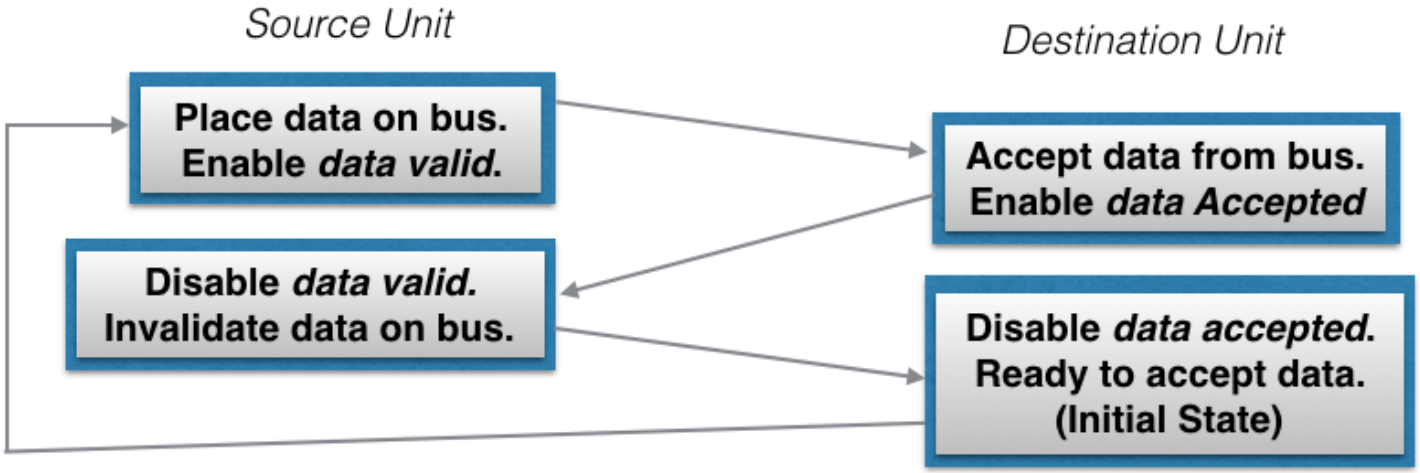




a) Block Diagram

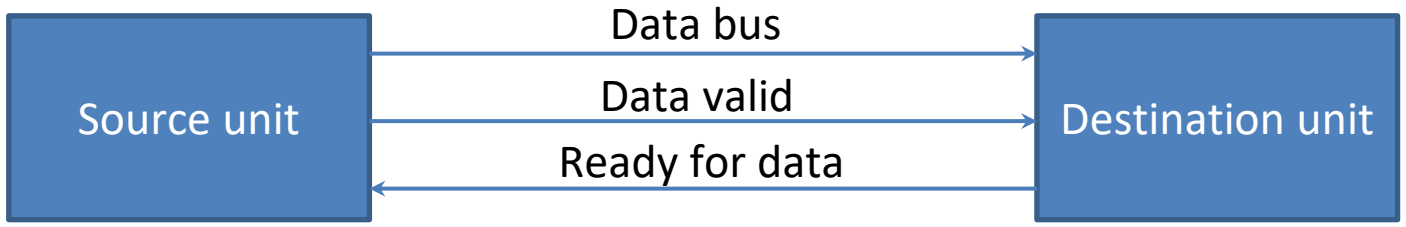


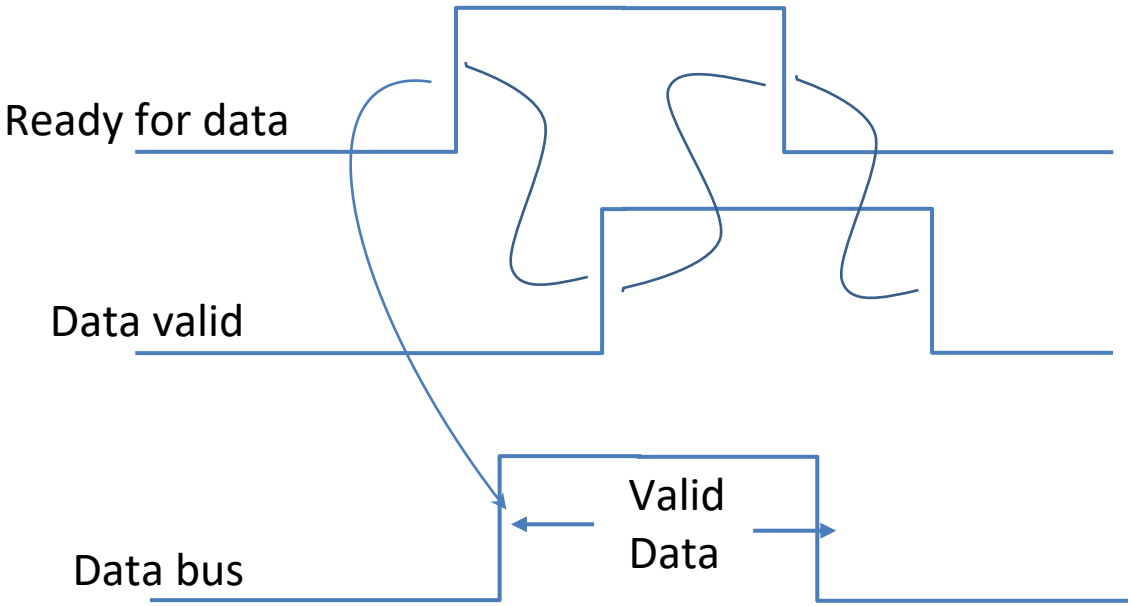
b) Timing Diagram



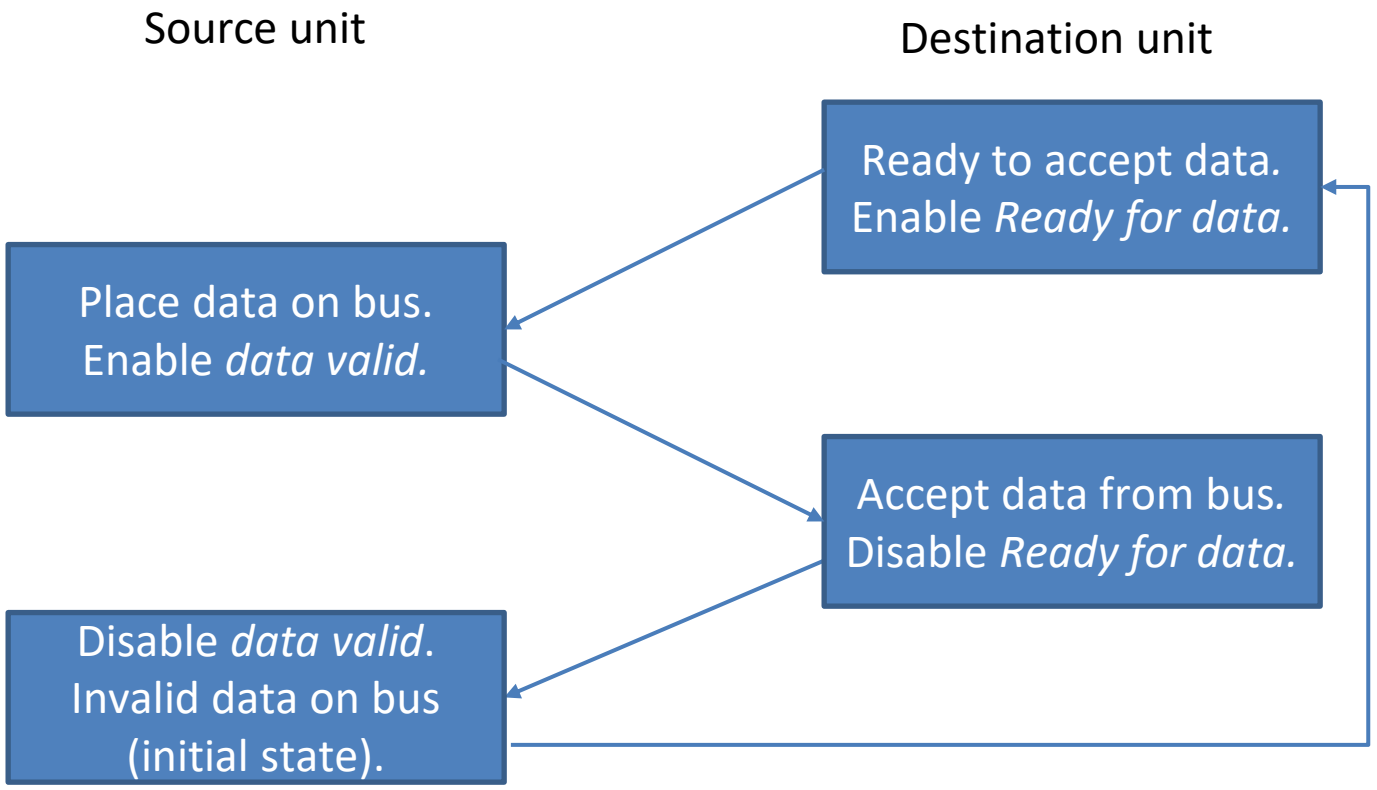
c) Sequence Diagram(Sequence of events)

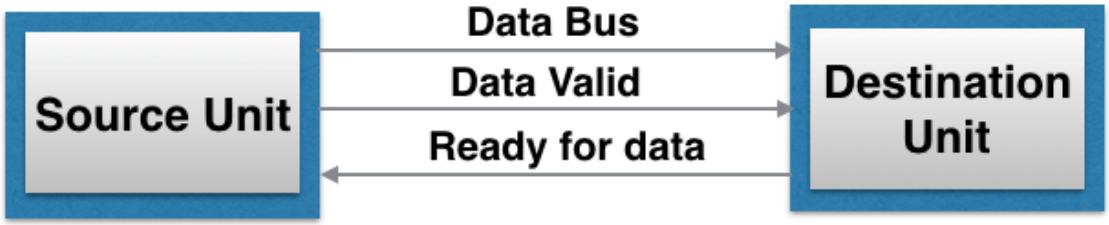
2.2 Destination initiated Handshake



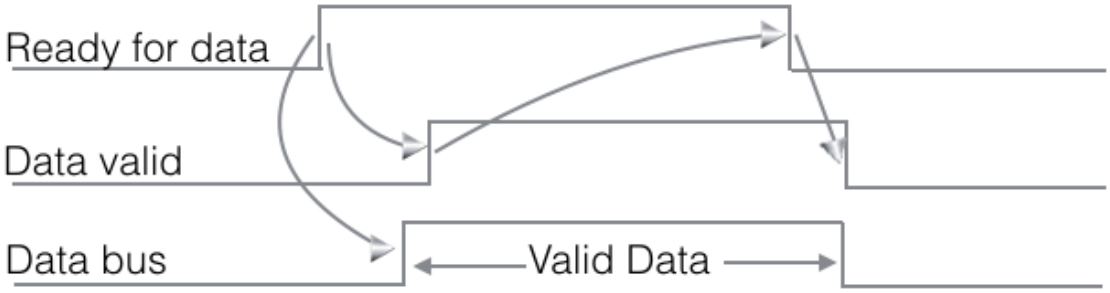


2.2 Destination initiated Handshake

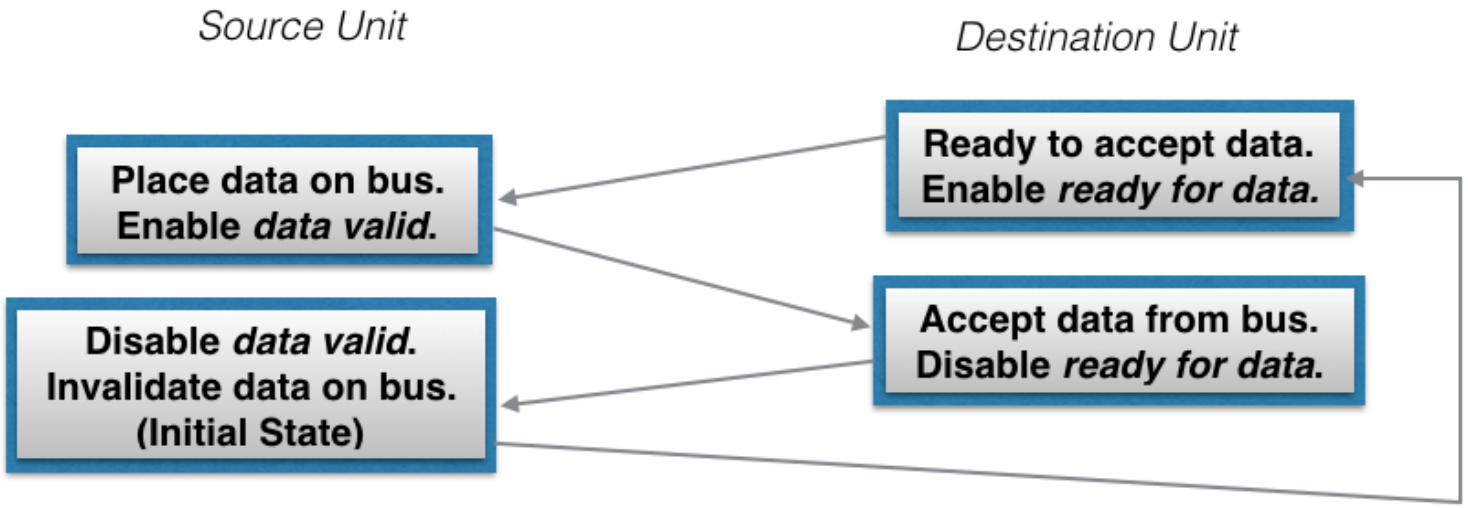




a) Block Diagram



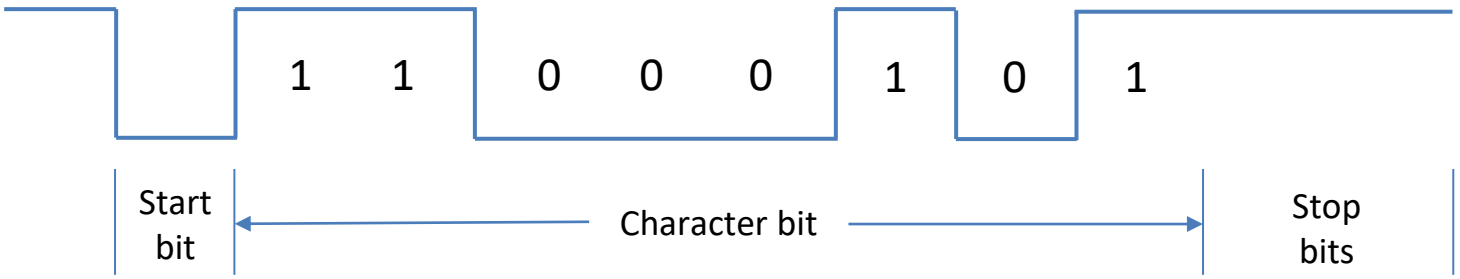
b) Timing Diagram



c) Sequence Diagram(sequence of events)

Asynchronous Serial Transfer

- Rules for transmission
  1. When a character is not being sent, the line is kept in the 1-state.
  2. The initiation of a character transmission is detected from the start bit, which is always 0.
  3. The character bits always follow the start bit.
  4. 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.

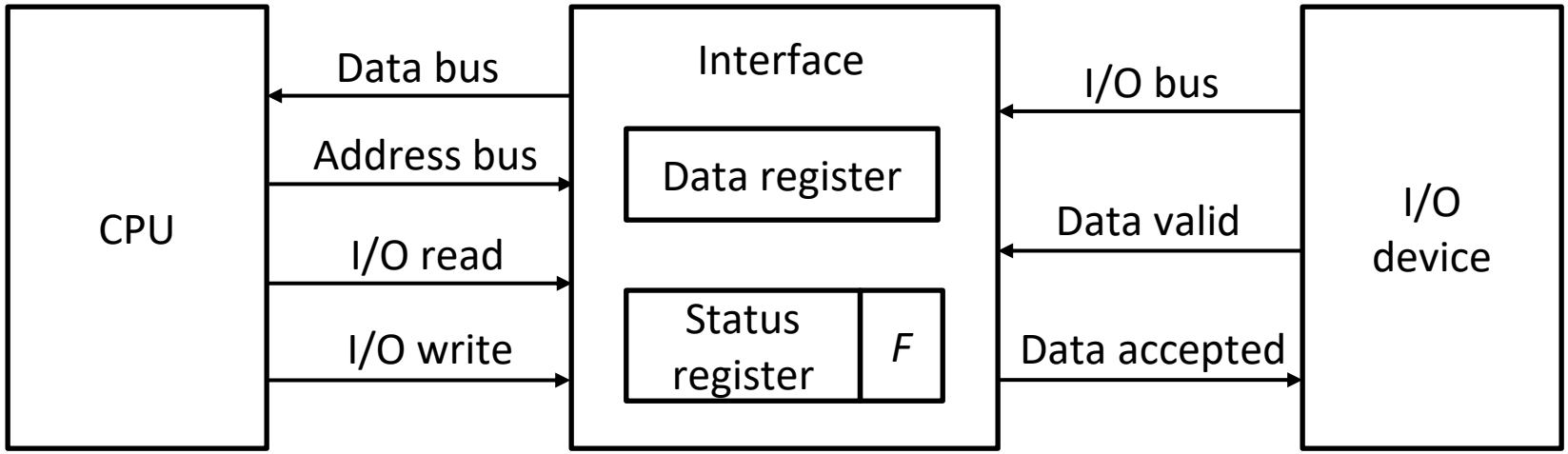


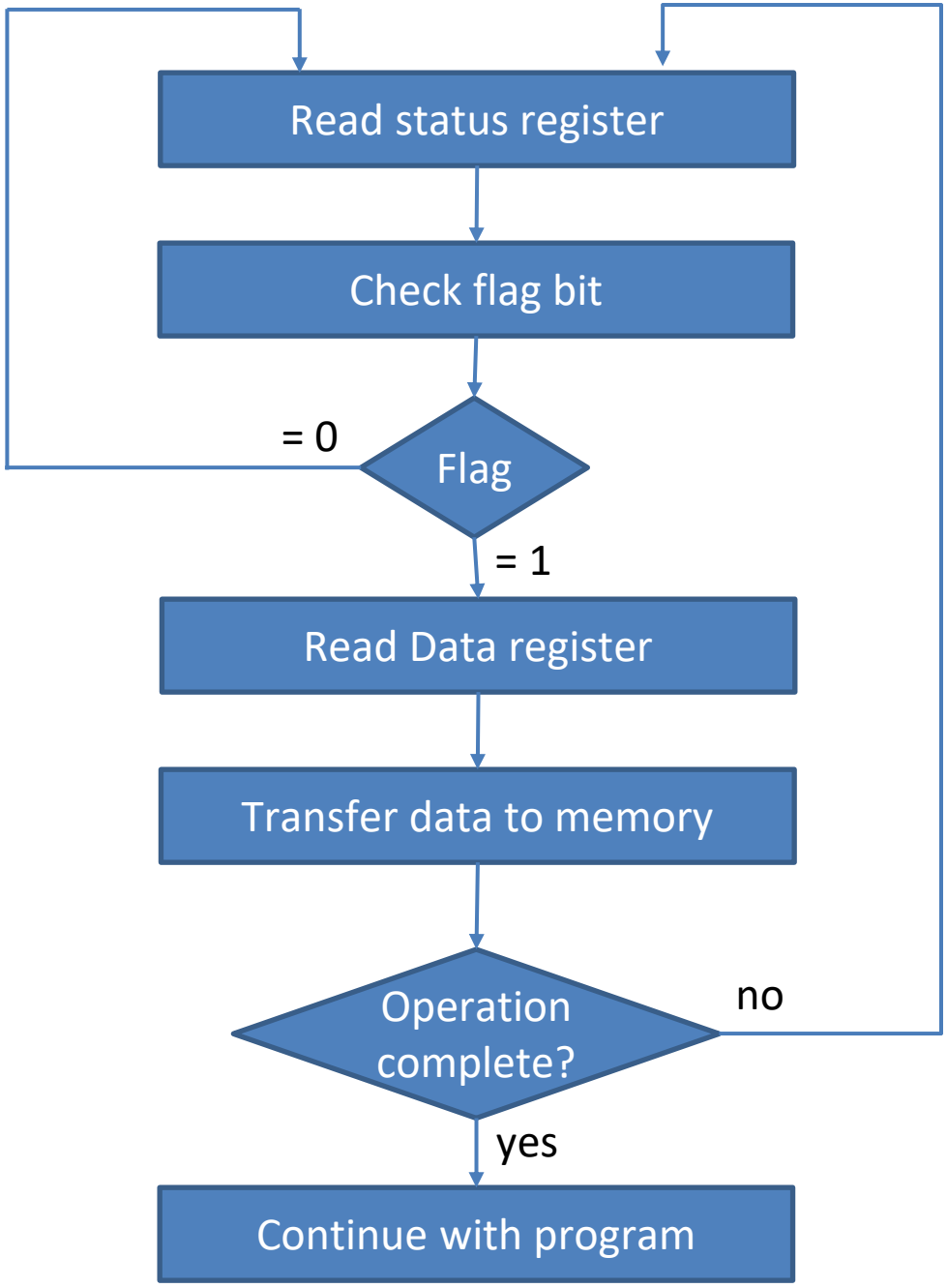
Modes of Transfer

Modes of Transfer

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

Programmed I/O




Read status register
Check flag bit
Flag
= 0
= 1
Read Data register
Transfer data to memory
Operation complete?
no
yes
Continue with program

Interrupt-initiated I/O

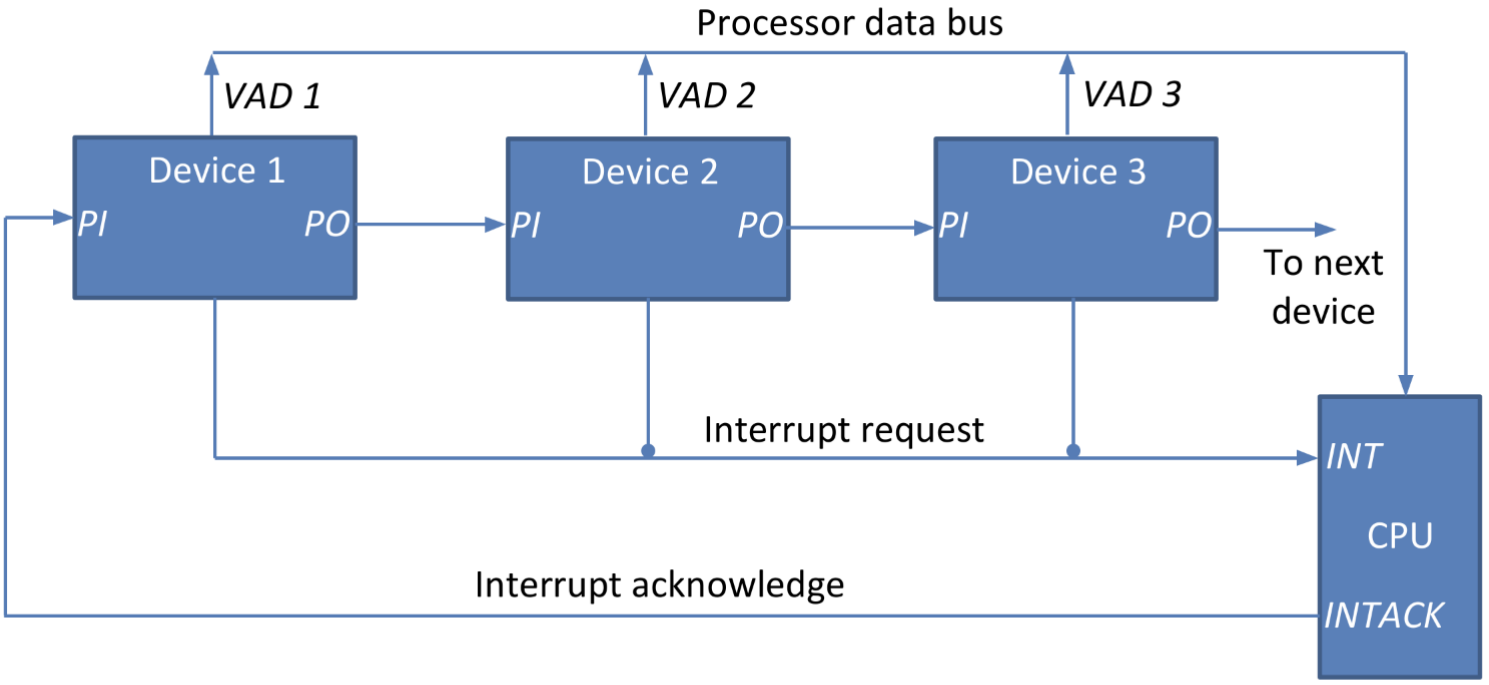
- An alternative to the CPU constantly monitoring the flag is to let the interface inform the computer when it is ready to transfer data.
- While the CPU is running a program, it does not check the flag.
- However, when the flag is set, the computer is momentarily interrupted from proceeding with current 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 transfer.
- After the transfer is completed, the computer returns to the previous program to continue what it was doing before the interrupt.

Priority Interrupt

Priority Interrupt (Daisy-Chaining Technique)

- Determines which interrupt is to be served first when two or more requests are made simultaneously
- Also determines which interrupts are permitted to interrupt the computer while another is being serviced.
- Higher priority interrupts can make requests while servicing a lower priority interrupt.

Daisy-Chaining Technique

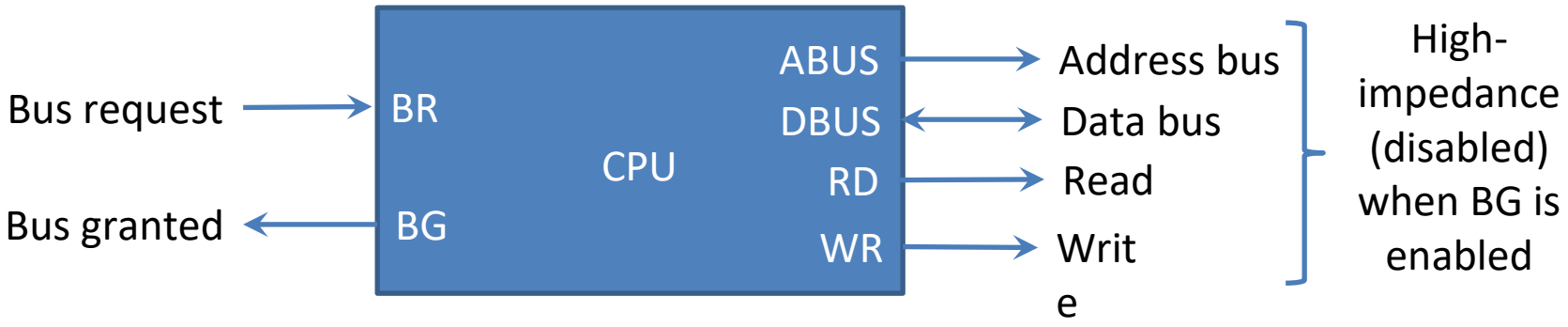


Direct Memory Access

DMA (Direct Memory Access)

- The transfer of data between a fast storage device such as magnetic disk and memory is often limited by the speed of the CPU.
- Removing the CPU from the path and letting the peripheral device manage the memory buses 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 between the I/O device and memory.

DMA (Direct Memory Access)

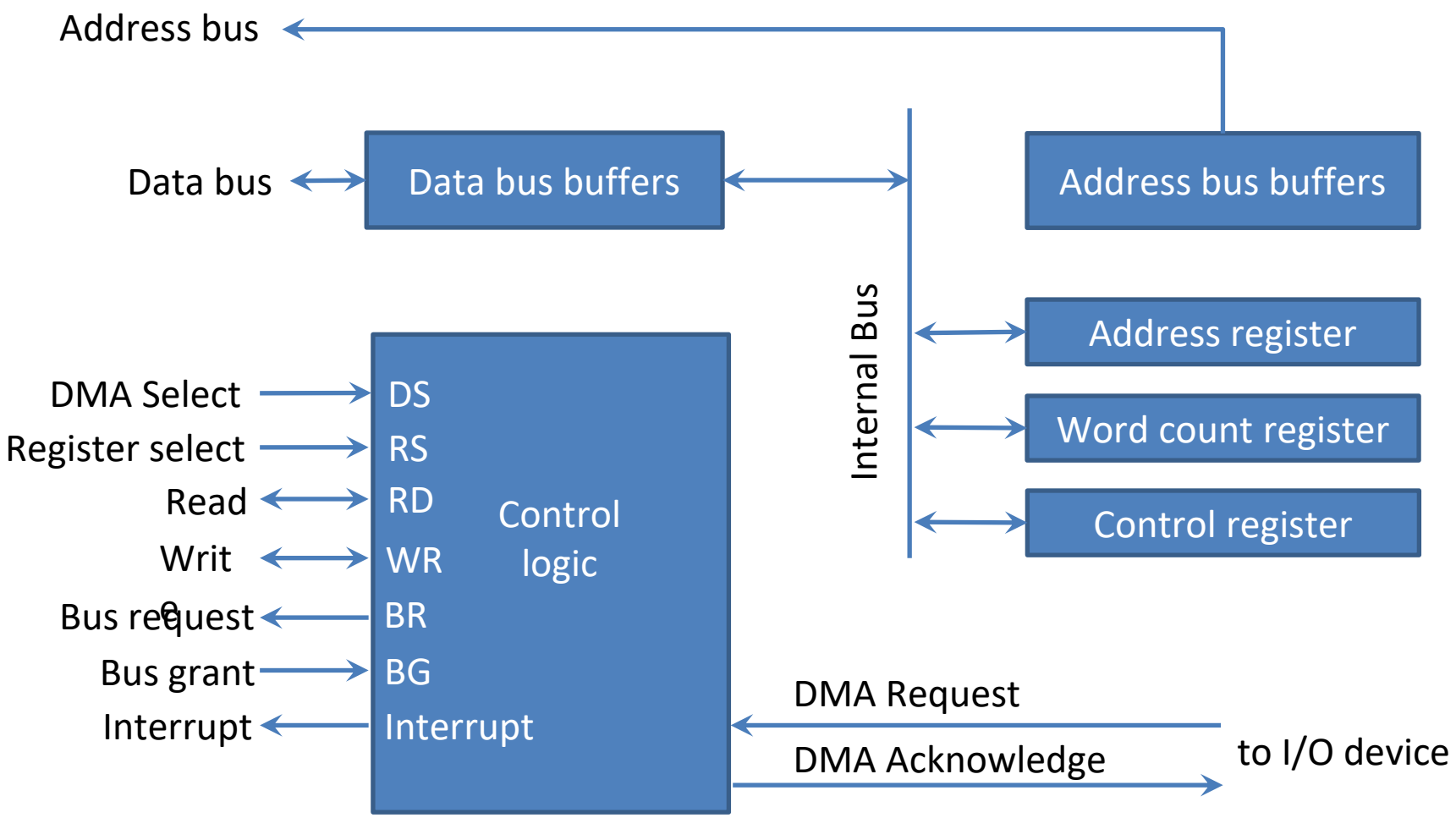


DMA (Direct Memory Access)

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).

DMA Controller



Questions asked in exam

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