CH - 6

By-

Kalyani N Pampattiwar

Asst. Prof., CE dept.

SIES GST,Nerul

Data Transfer Techniques

1. **Programmed I/O**

   - Processor issues a command to I/O module and must wait until the I/O operation is complete

2. **Interrupt driven I/O**

   - Processor issues a command to I/O module, continues to execute other instructions and is interrupted by I/O module when later has completed its work

3. **Direct Memory Access (DMA)**

   - I/O module and main memory exchange data directly without processor involvement

1. Programmed I/O

- Processor encounters an instruction relating to I/O
- It executes that instruction
- With the Programmed I/O, the I/O module will perform the requested action and set appropriate bits in the I/O status register
- It doesn't interrupt the processor
- Processor periodically check the status of the I/O module until it finds the operation is complete

I/O commands

- **Control:** used to activate the peripherals and tell it what to do

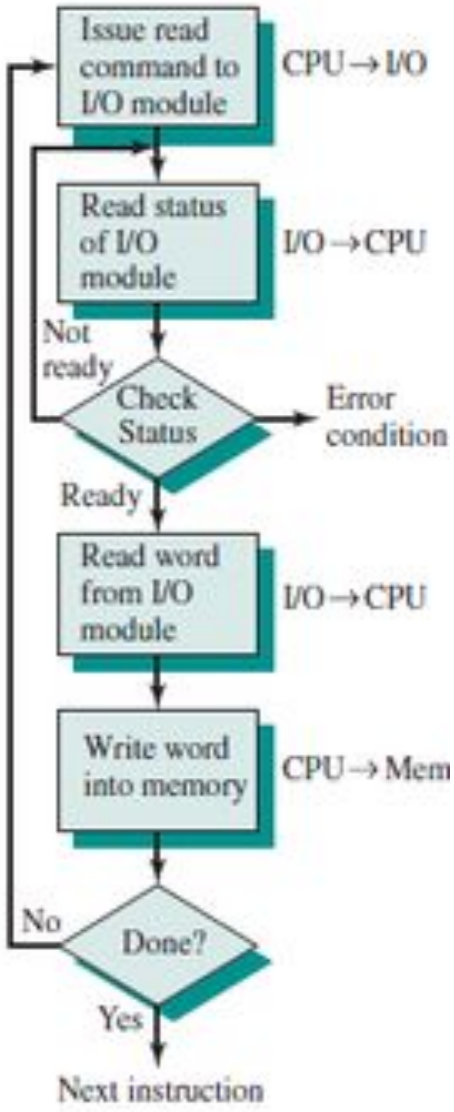
e.g., Magnetic tape

- **Test:** used to test various status conditions associated with an I/O module and its peripheral

- **Read:** Causes the I/O module to obtain an item of data from the peripheral and place it in a internal buffer

- **Write:** Causes the I/O module to take an item of data from the data bus and transmit that data item to peripheral

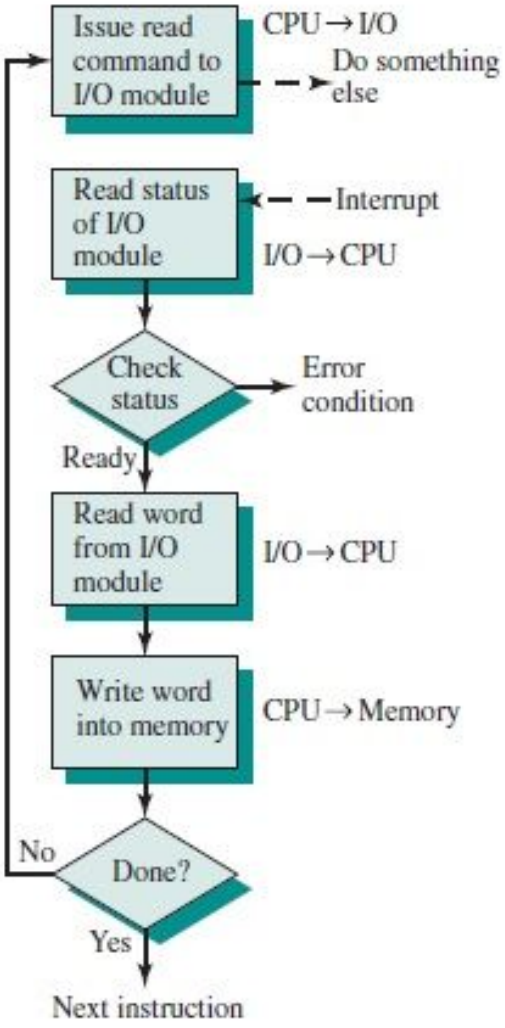
Programmed I/O…Cont..



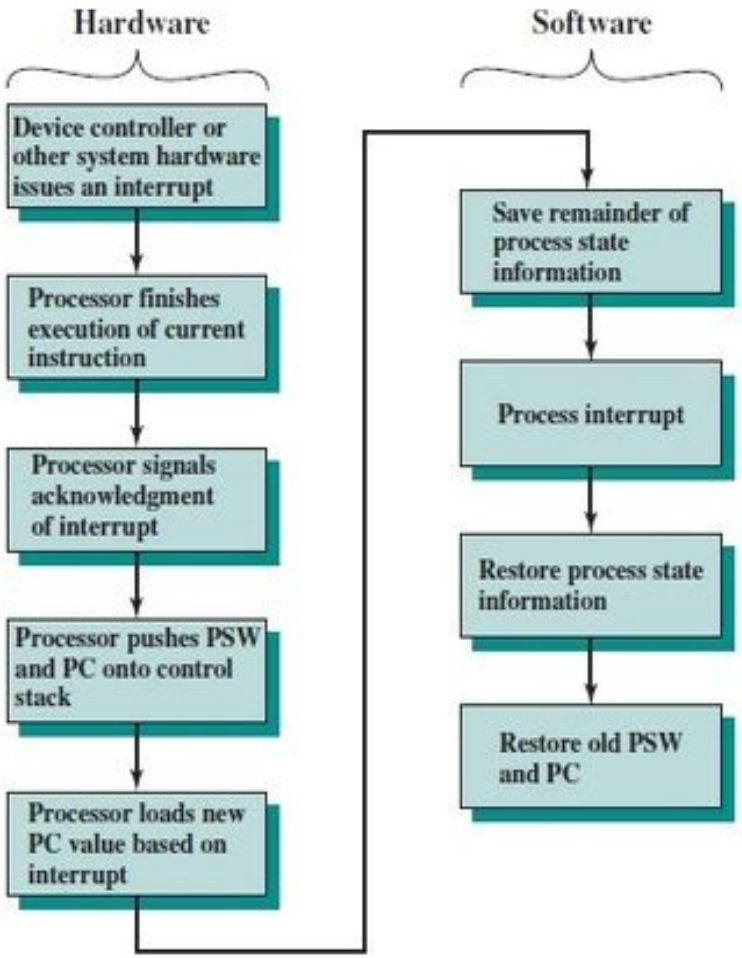
2. Interrupt Driven I/O

- Drawback of programmed I/O is processor has to wait a long time for I/O module to be ready for either reception or transmission of data
- System performance is degraded because of repeatedly interrogating status of I/O module
- Alternative is interrupt driven I/O
- After each instruction cycle processor checks the interrupt

2. Interrupt Driven I/O…Cont..



2. Interrupt Driven I/O…Cont..



2. Interrupt Driven I/O…Cont..

- **Advantage:**

More efficient as compare to programmed I/O because it eliminates needless waiting

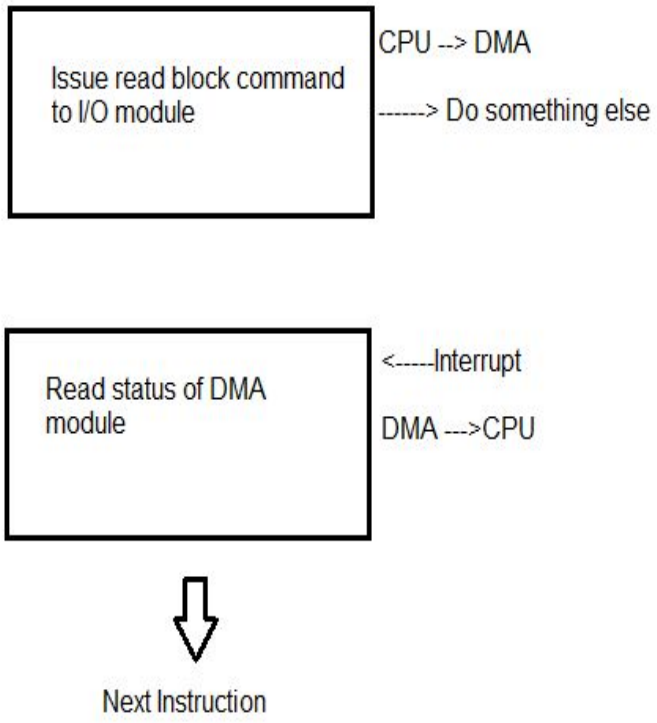
- **Disadvantage:**

Consumes lot of processor time

3. Direct Memory Access (DMA)

- Drawback of previous approaches are intervention of the processor to transfer data between memory and I/O module
- DMA is the solution to problem
- DMA is additional Module (hardware) on bus
- Capable of mimicking processor and taking control of the system from the processor

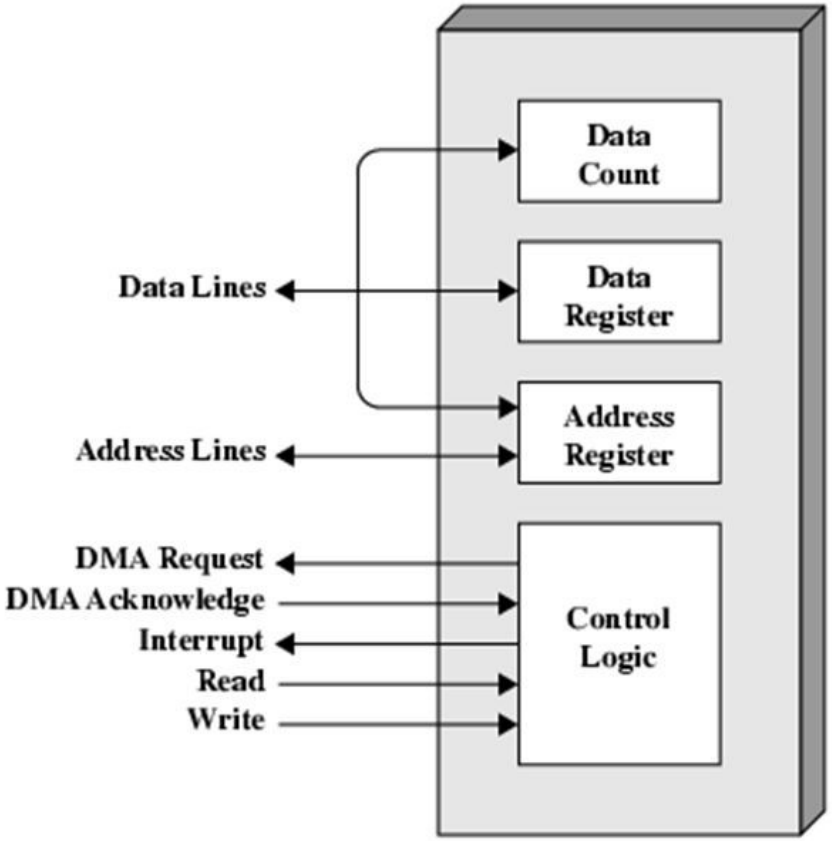
DMA…Cont..



- DMA takes control of system from processor
- DMA must use the bus only when processor does not need it
- Technique called as **cycle stealing** (DMA module steals bus cycle)

DMA Module

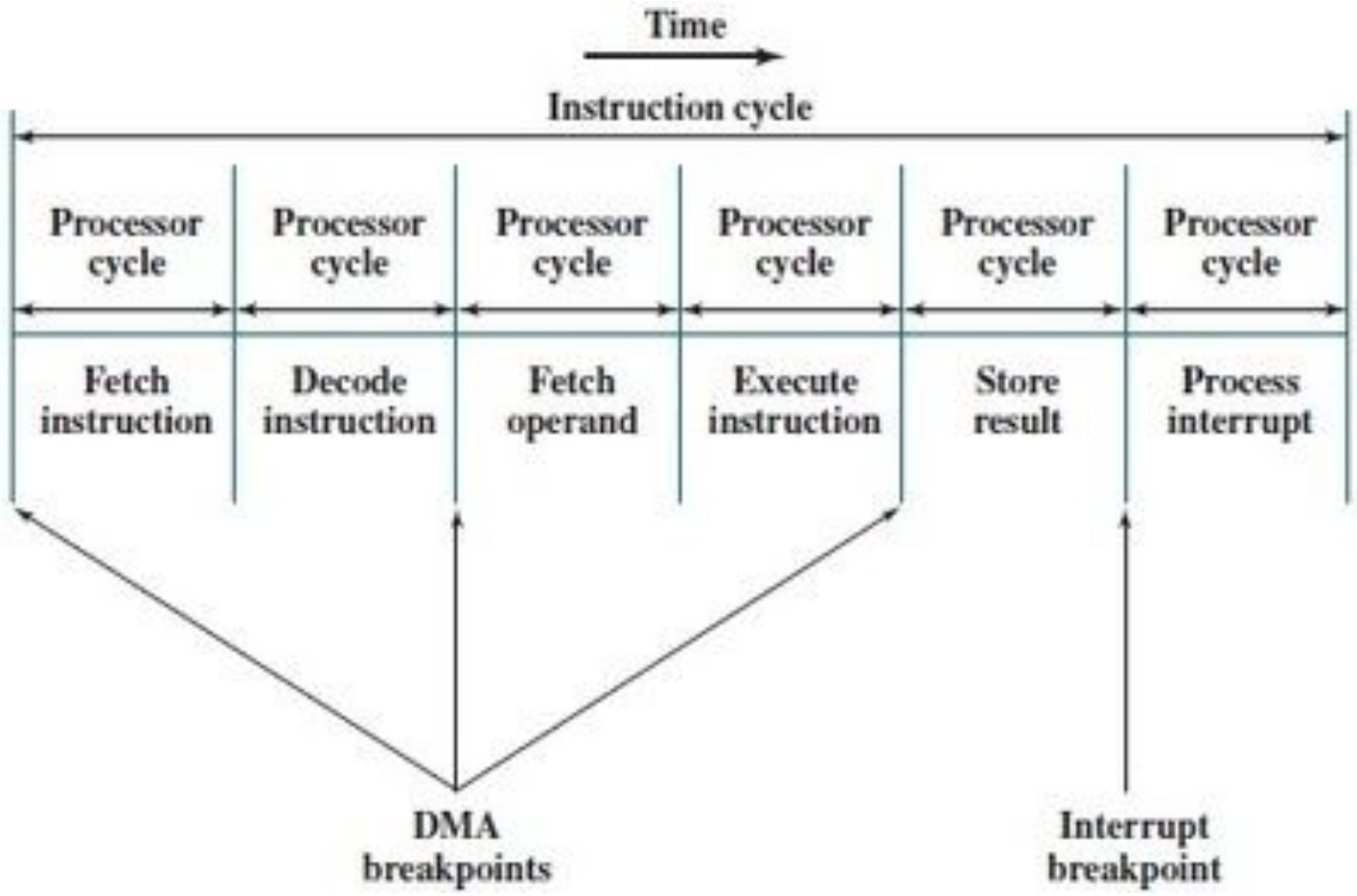
Typical DMA Module Diagram



DMA Operation

- CPU tells DMA controller:-
  - Read/Write
  - Device address
  - Starting address of memory block for data
  - Amount of data to be transferred
- CPU carries on with other work
- DMA controller deals with transfer
- DMA controller sends interrupt when finished

DMA & interrupt breakpoints during instruction cycle



DMA Configurations (1)

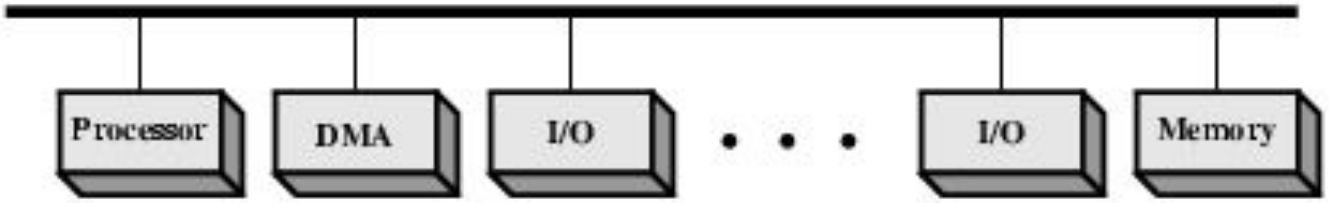


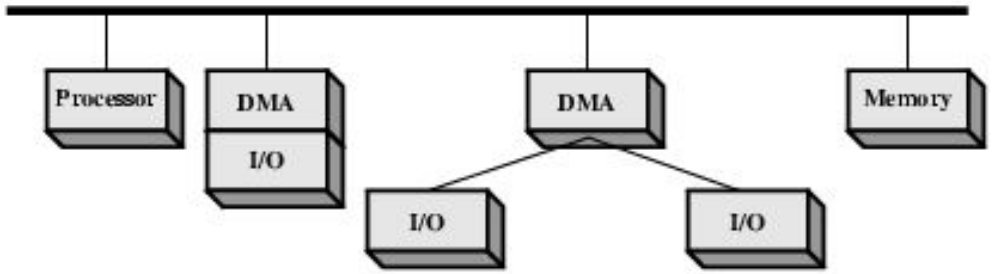
Fig a) Single bus, detached DMA

- Each transfer uses bus twice

  - I/O to DMA then DMA to memory

- CPU is suspended twice

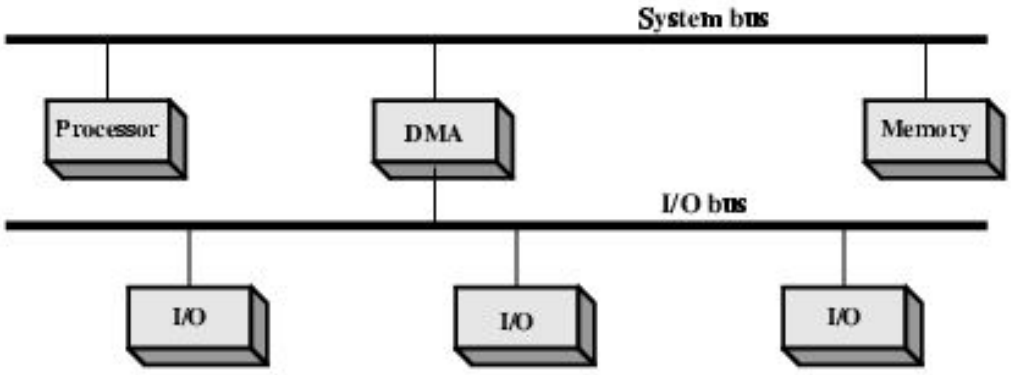
DMA Configurations (2)



(b) Single-bus, Integrated DMA-I/O

- Single Bus, Integrated DMA controller
- Controller may support >1 device
- Each transfer uses bus once
  - DMA to memory
- CPU is suspended once

DMA Configurations (3)



(c) I/O bus

- Separate I/O Bus
- Bus supports all DMA enabled devices
- Each transfer uses bus once
  -DMA to memory
- CPU is suspended once

INTERRUPT HANDLING

- Interrupt is signal or event inside a computer system
- Because of Interrupt CPU temporarily suspends the current program execution
- Interrupt handling is a function performed jointly by hardware and software

Interrupt Types

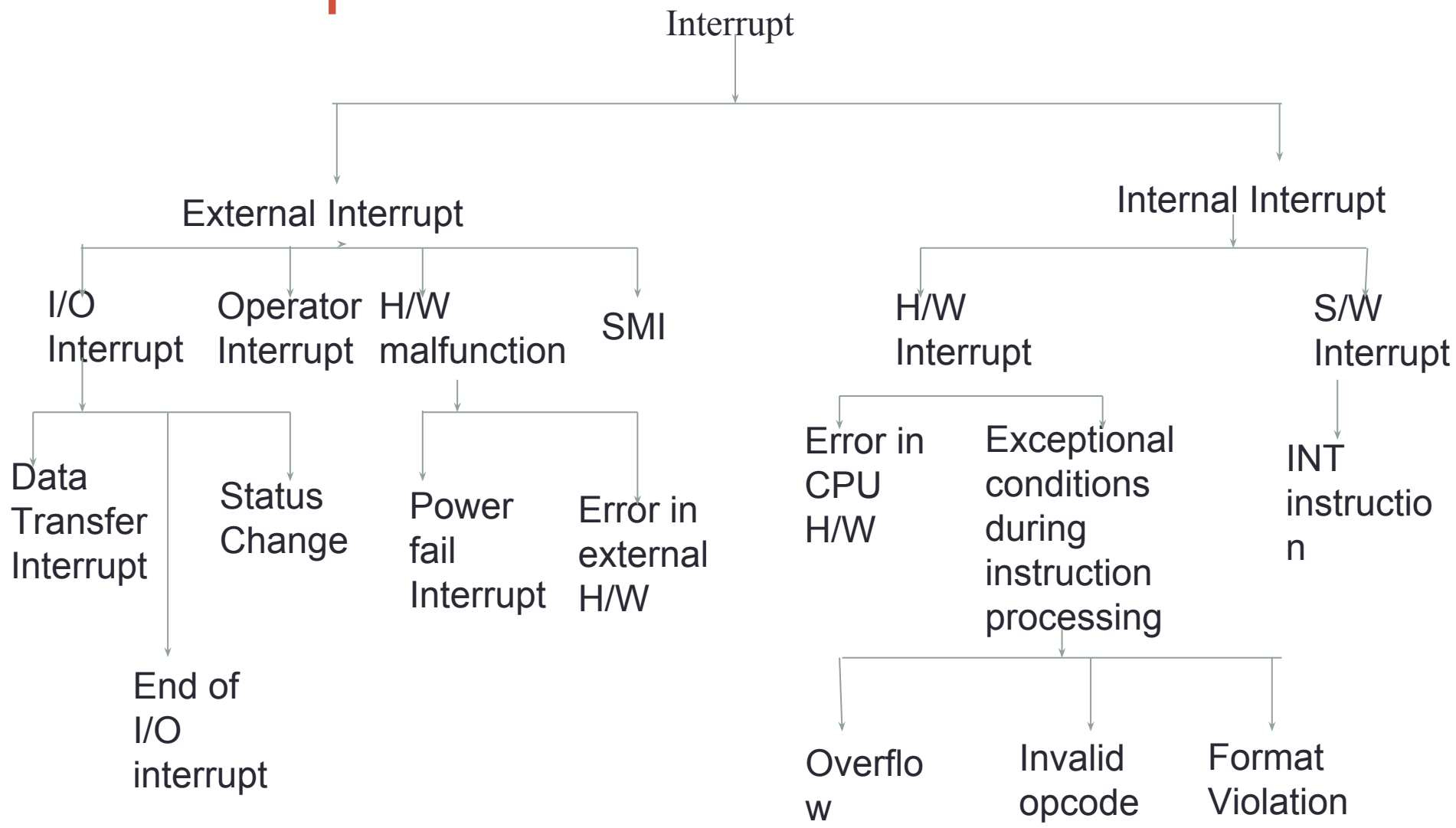
External Interrupt

- These are generated by hardware circuits outside the CPU
- These are hardware interrupts
- H/W interrupts are generated by the circuits

Internal Interrupt

- These are the interrupts generated within CPU
- These may be H/W or S/W interrupts
- S/W interrupts are generated by an instruction in the program

Interrupt


Interrupt
External Interrupt
Internal Interrupt
I/O Interrupt
Operator Interrupt
H/W malfunction
SMI
H/W Interrupt
S/W Interrupt
Data Transfer Interrupt
Status Change
End of I/O interrupt
Power fail Interrupt
Error in external H/W
Error in CPU H/W
Exceptional conditions during instruction processing
INT instruction
Overflow
Invalid opcode
Format Violation

Interrupt Types

- **Data transfer interrupt** is raised by an I/O controller to indicate to the CPU that it is ready for data transfer

- The **end of I/O interrupt** is raised by an I/O controller on completing a command to indicate to the CPU that the I/O operation is complete

- **Status change interrupt** is used to inform the software when there is a change of status of an I/O device

  (e.g., Changing device sate from READY to NON-READY)

- **Operator interrupt** is a special interrupt to the operating system to perform a specific action

- **Power fail interrupt** is raised by a special hardware which keeps monitoring the fluctuations in the AC power supply

  This interrupt is an advance information that power is going to fail in a short while

Interrupt Types

- **External H/W malfunction interrupt** occurs when there is a hardware malfunction (e.g., memory failure , bus error)
- **CPU Internal H/W error interrupt** indicates detection of some hardware malfunction within the CPU
- **Program exception interrupts** caused due to abnormal conditions encountered by the CPU during the execution of the program(e.g., illegal opcode, instruction format violation, operand format violation , overflow)
- **Software Interrupt** is created by the program so that the CPU can temporarily branch from current program to another program. INT(interrupt) instruction is used
- **System management interrupt(SMI)** used for minimizing power consumption by powering off the peripheral subsystems which are not accessed for a long time. They will be powered on whenever accesses are made.

Interrupt Types

- **Maskable Interrupt:** An Interrupt that can be disabled or ignored by the instructions of CPU are called as Maskable Interrupt.
  Eg: RST6.5,RST7.5,RST5.5 Of 8085 are maskable Interrupts.

Non-Maskable Interrupt: An interrupt that cannot be disabled or ignored by the instructions of CPU are called as Non-Maskable Interrupt.
Eg: Trap of 8085

1. Hardware Interrupts

- If the signal for the processor is from external device or hardware is called hardware interrupts

- Example: from keyboard we will press the key to do some action this pressing of key in keyboard will generate a signal which is given to the processor to do action, such interrupts are called hardware interrupts.

Hardware Interrupts…Cont…

- **Maskable Interrupt:** The hardware interrupts which can be delayed when a much highest priority interrupt has occurred to the processor.

- **Non Maskable Interrupt:** The hardware which cannot be delayed and should process by the processor immediately.

2. Software Interrupts

- **Normal Interrupts:**
  The interrupts which are caused by the software instructions are called software instructions.

- **Exception:**
  Unplanned interrupts while executing a program is called Exception. For example: while executing a program if we got a value which should be divided by zero is called a exception.

Interrupt handling

- Instruction cycle consists of fetch, decode, execute and read/write functions

- After every instruction cycle the processor will check for interrupts to be processed if there is no interrupt is present in the system it will go for the next instruction cycle which is given by the instruction

- If there is an interrupt present then it will trigger the interrupt handler, the handler will stop the present instruction which is processing and save its configuration in a register and load the program counter of the interrupt from a location which is given by the interrupt vector table

Cont..

- After processing the interrupt by the processor interrupt handler will load the instruction and its configuration from the saved register, process will start its processing where it's left.

- This saving the old instruction processing configuration and loading the new interrupt configuration is also called as context switching.

- The interrupt handler is also called as Interrupt service routine (ISR).

Features of the ISR:

- Interrupts can occur at any time they are asynchronous. ISR's can call for asynchronous interrupts.
- Interrupt service mechanism can call the ISR's from multiple sources.
- ISR's can handle both maskable and non maskable interrupts. An instruction in a program can disable or enable an interrupt handler call.
- ISR on beginning of execution it will disable other devices interrupt services. After completion of the ISR execution it will re initialize the interrupt services.
- The nested interrupts are allowed in ISR for diversion to other ISR.

Bus Arbitration

- Comes into picture whenever there are more than one processor requesting the services of bus
- Needed to reduce the problem of bus congestion
- The process of selecting a processor among requesting processors is known as **arbitration**
- Arbitration process can be centralized or distributed

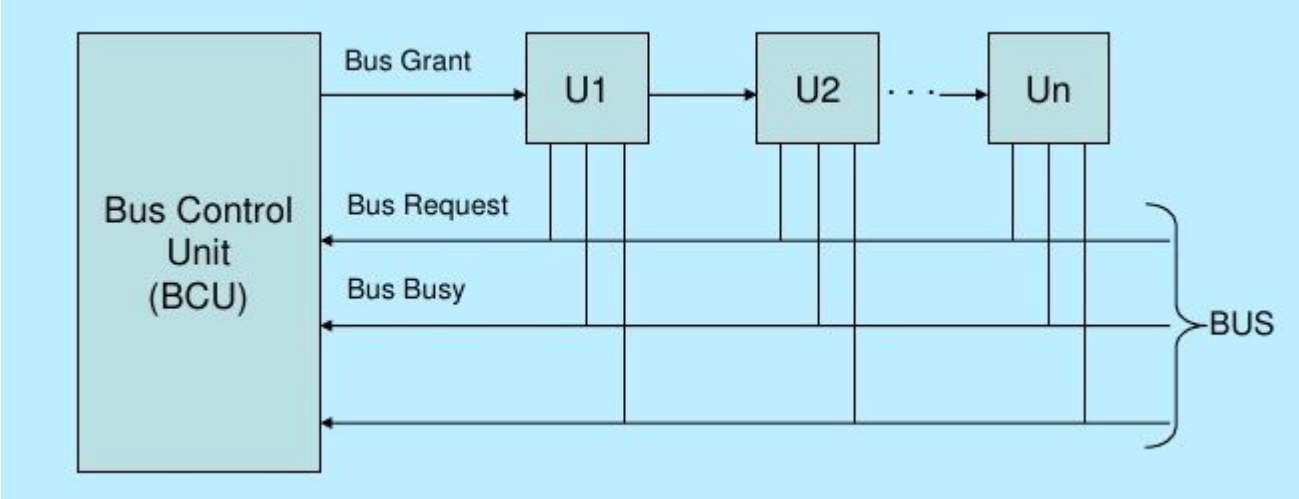
Arbitration schemes

1. Daisy chaining

2. Polling

3. Independent requesting

1. Daisy chaining



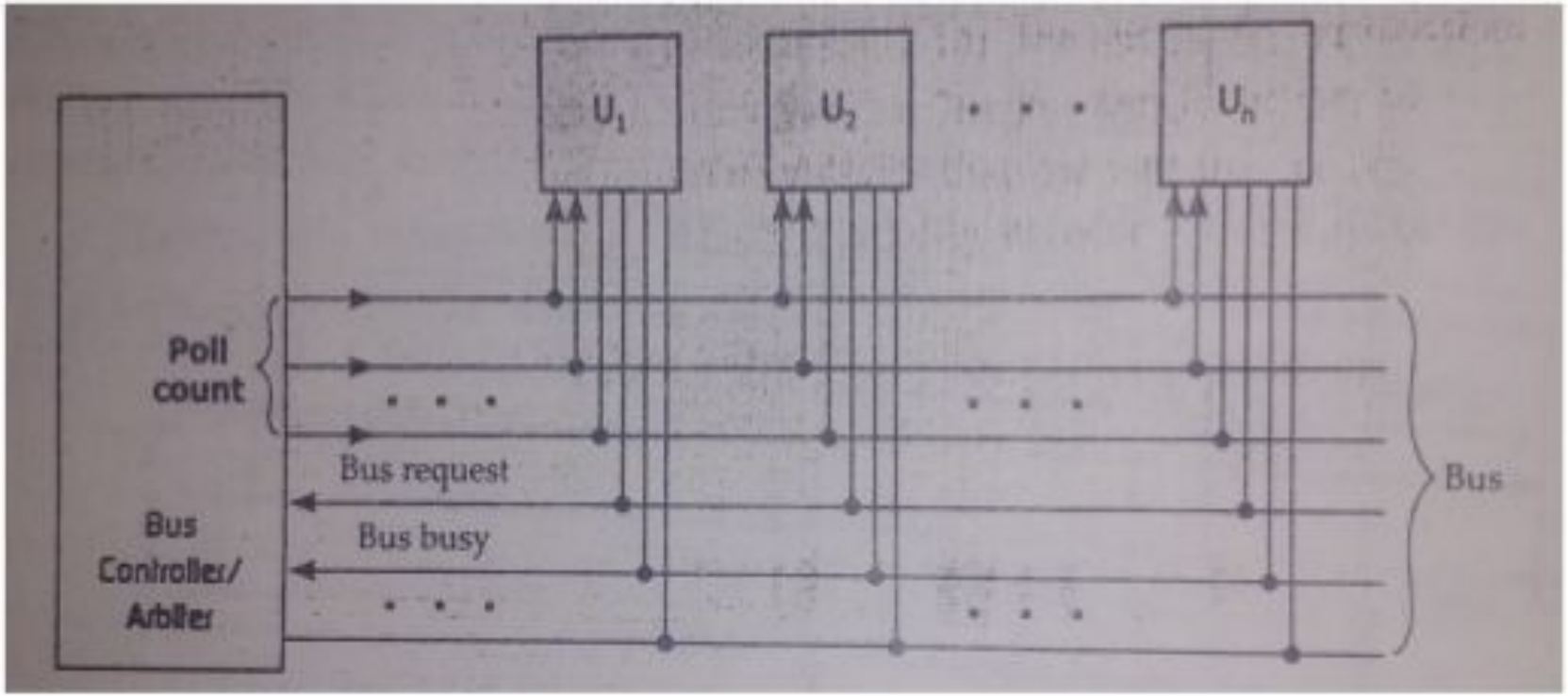
Cont..

- Characterized by Bus Grant signal connected serially from master to master
- 3 control signals are involved
  - Bus Busy
  - Bus request
  - Bus Grant
- Bus controller can respond to bus request signal only if bus busy signal is inactive
- Bus busy is active during the period it is being used by any of the processor
- Processors are connected to Bus request line
- Processor willing to use the bus will place request on Bus request line

Cont..

- Bus controller respond to the Bus request signal by placing Bus grant signal on the Bus grant line
- When the first processor willing to use the bus receives Bus grant signal it activates Bus busy signal
- When non requesting processor receives Bus grant signal it sends it to next processor
- If two processors simultaneously request bus , the processor closer to bus controller gains access to the bus
- **Advantage:** Simple, few control lines, device can be added easily
- **Disadvantage:** If one of the processor fails then all processors ahead will never get bus grant line

2. Polling


U_1
U_2
U_n
Poll count
Bus request
Bus busy
Bus Controller/ Arbiter
Bus

Cont..

- In this each master is called by its address turn by turn
- Address of master is generated on poll count lines
- Poll count lines are connected to each device
- Bus request which can be made on bus request line will not be responded to till Bus busy line is active
- Bus controller responds to the signal on Bus request line by generating addresses in sequence on poll count lines
- When poll count matches the address of a particular master that is requesting for the bus , the master activates the Bus busy signal and start using the bus

Cont..

- **Advantages:**
- Priority can be changed by changing the sequence of the generation of addresses on the poll count lines

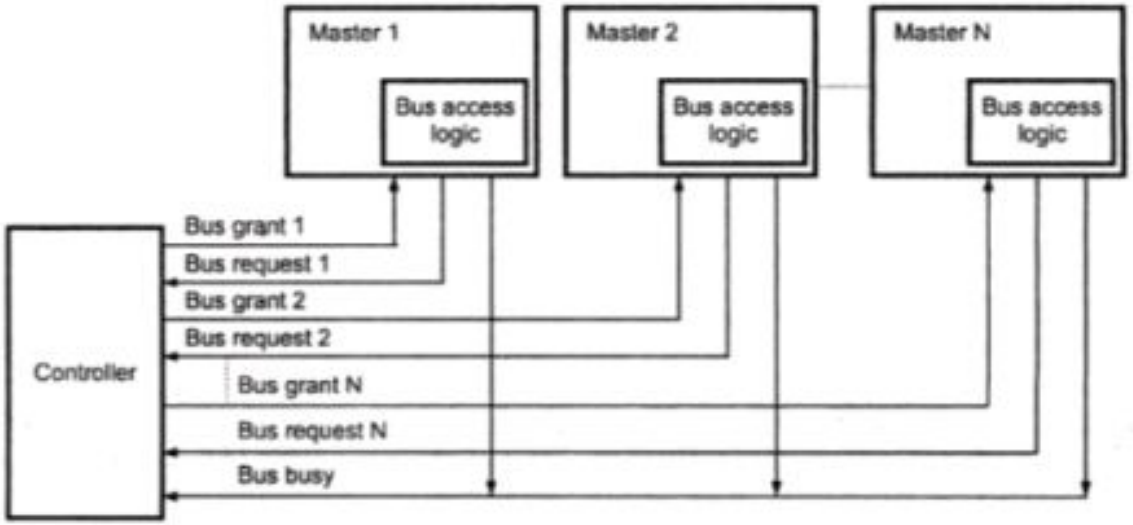
- Failure of one master will not affect any other master

- **Disadvantages:**
- Requires more control lines

- No. of poll count lines restrict number of masters connected to bus

(e.g., with n poll count lines only 2n masters can be connected)

3. Independent Requesting



Cont..

- In this scheme each master has a separate pair of bus request and bus grant lines and each pair has a priority assigned to it
- The built in priority decoder within the controller selects the highest priority request and asserts the corresponding bus grant signal
- Arbitration among the masters is carried out by central arbiter


Thank
you