EC-252: COMPUTER ARCHITECTURE AND MICROPROCESSORS

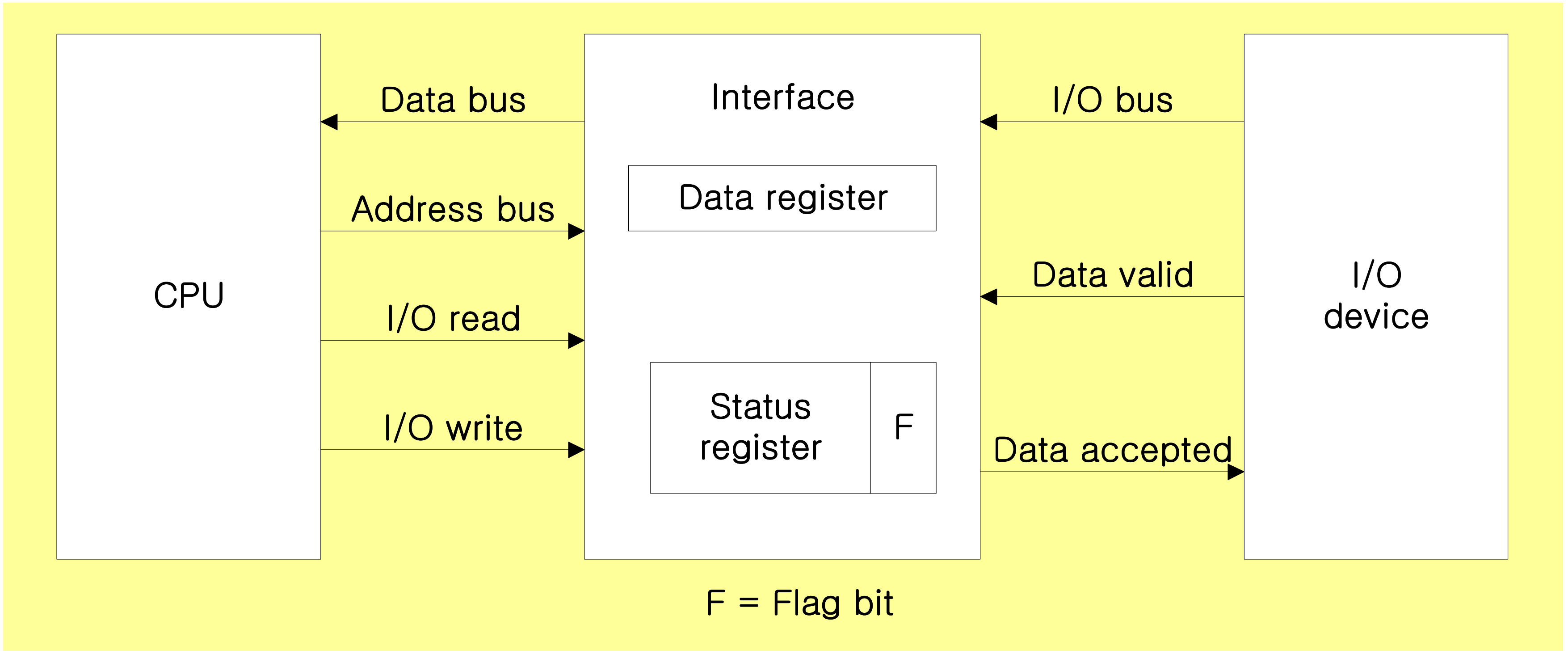
Vaskar Raychoudhury

Indian Institute of Technology Roorkee

Modes of Transfer

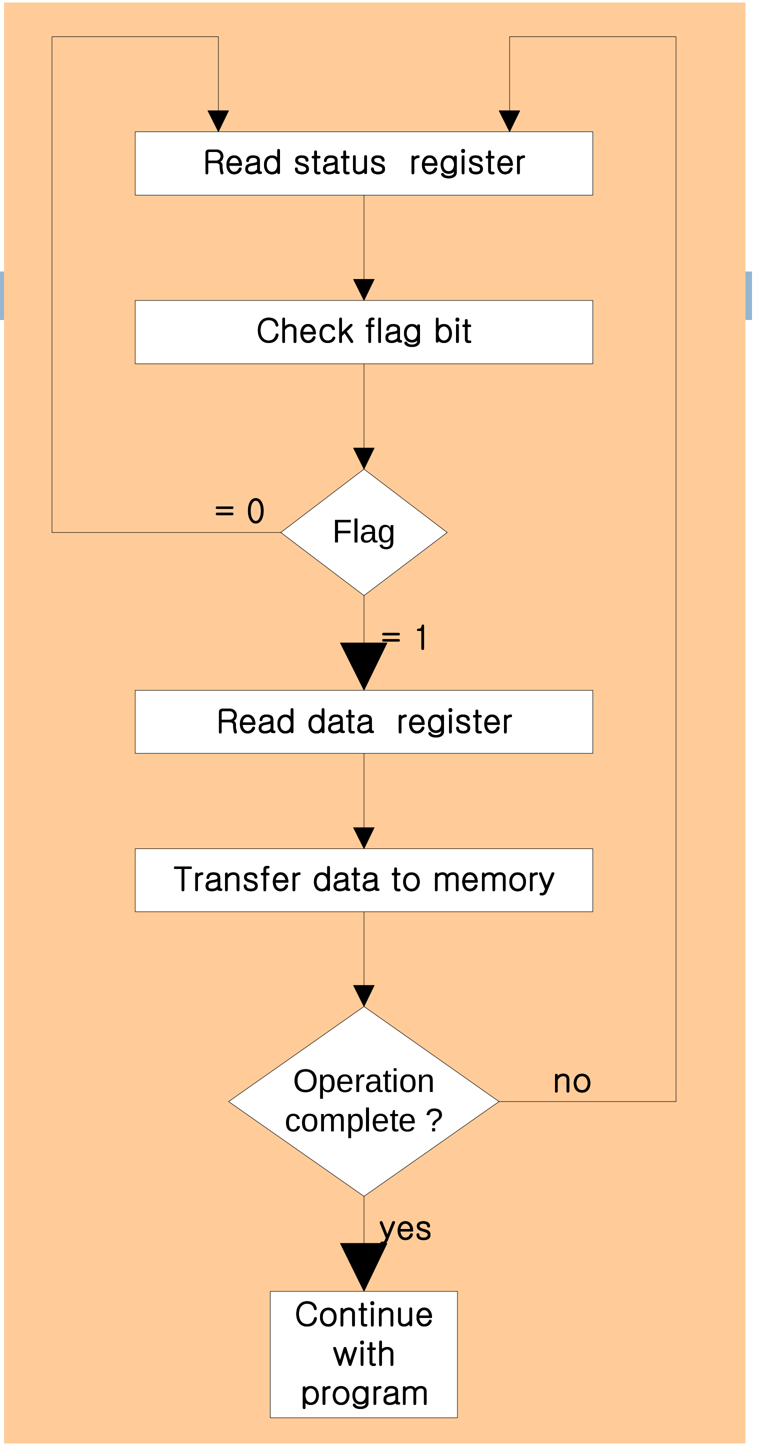
- Data transfer to and from peripherals
  - 1) Programmed I/O
  - 2) Interrupt-initiated I/O
  - 3) Direct Memory Access (**DMA**)

Example of Programmed I/O



CPU Program to Input Data

- Step1: Read Status register
- Step 2: Check status of Flag bit
  - Branch to step 1 if not set, or
  - To step 3 if set
- Step 3: Read the data register



Interrupt-initiated I/O

- Interrupt-initiated I/O
  - Non-vectored : fixed branch address
  - Vectored : interrupt source supplies the branch address (**interrupt vector**)
  - Interrupt vector may contain
    - the first address of the I/O service routine, or
    - A location in memory which contains the first address of the I/O service routine

Software Considerations

- I/O routines
  - software routines for controlling peripherals and for transfer of data between the CPU & peripherals
- I/O routines for standard peripherals are provided by the manufacturer (**Device driver, OS** or **BIOS**)
- I/O routines are usually included within the operating system
- I/O routines are usually available as operating system procedures (**OS** or **BIOS function call**)

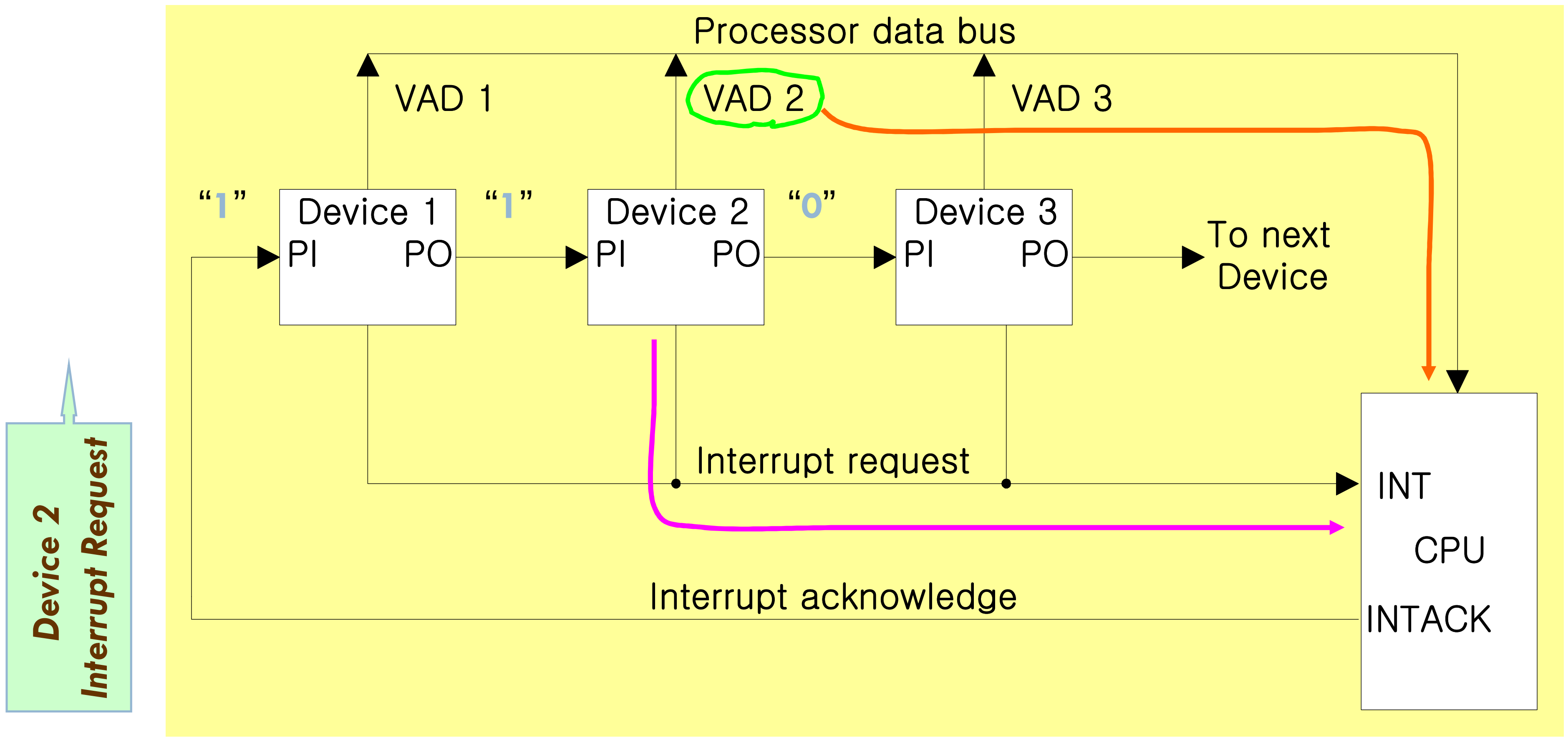
Priority Interrupt

- Priority Interrupt
  - 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
  - Methods to establish priority
    - 1) Software : **Polling**
    - 2) Hardware : **Daisy chain**, **Parallel priority**

Polling

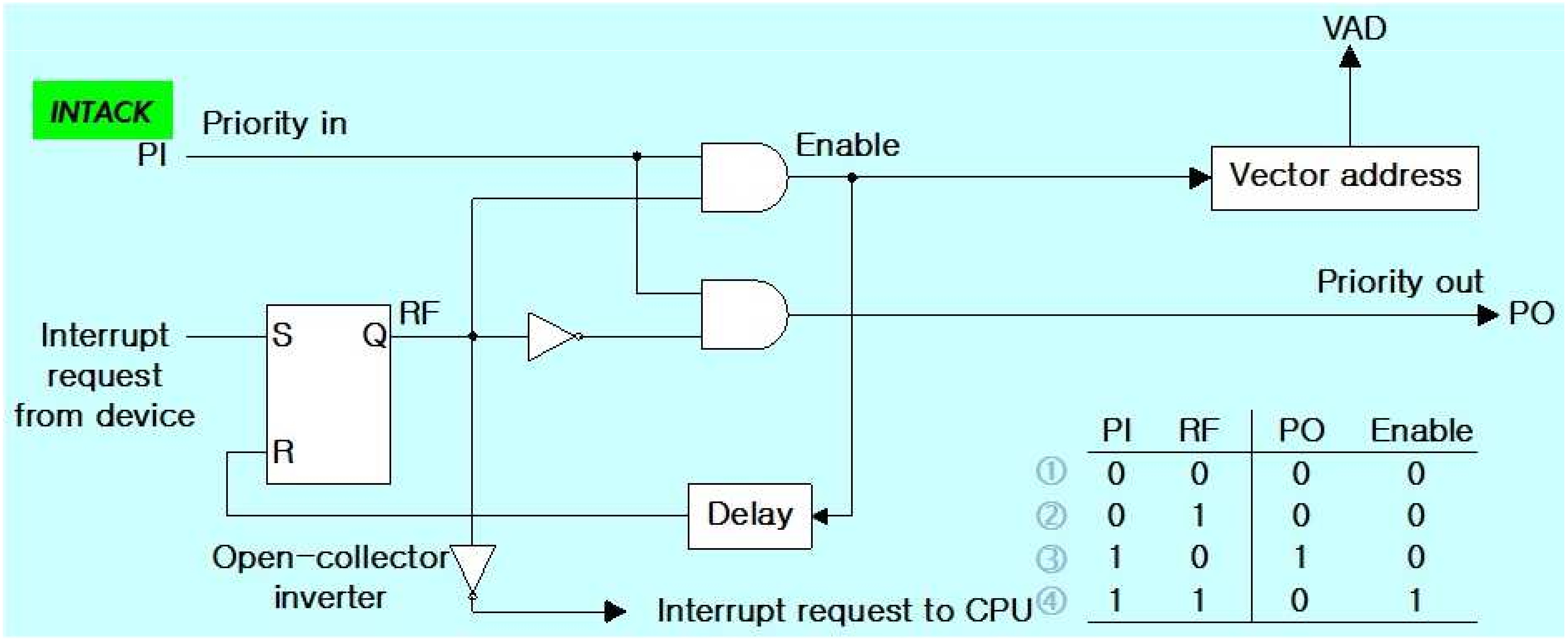
- Identify the highest-priority source by software means
  - One common branch address is used for all interrupts
  - Program polls the interrupt sources in sequence
  - The highest-priority source is tested first
- Polling priority interrupt
  - If there are many interrupt sources, the time required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

Daisy-Chaining



Daisy-Chaining (2)

- One stage of the daisy-chain priority arrangement
  - ① No interrupt request ② Invalid : interrupt request, but no acknowledge ③ No interrupt request : Pass to other device (*other device requested interrupt*) ④ Interrupt request

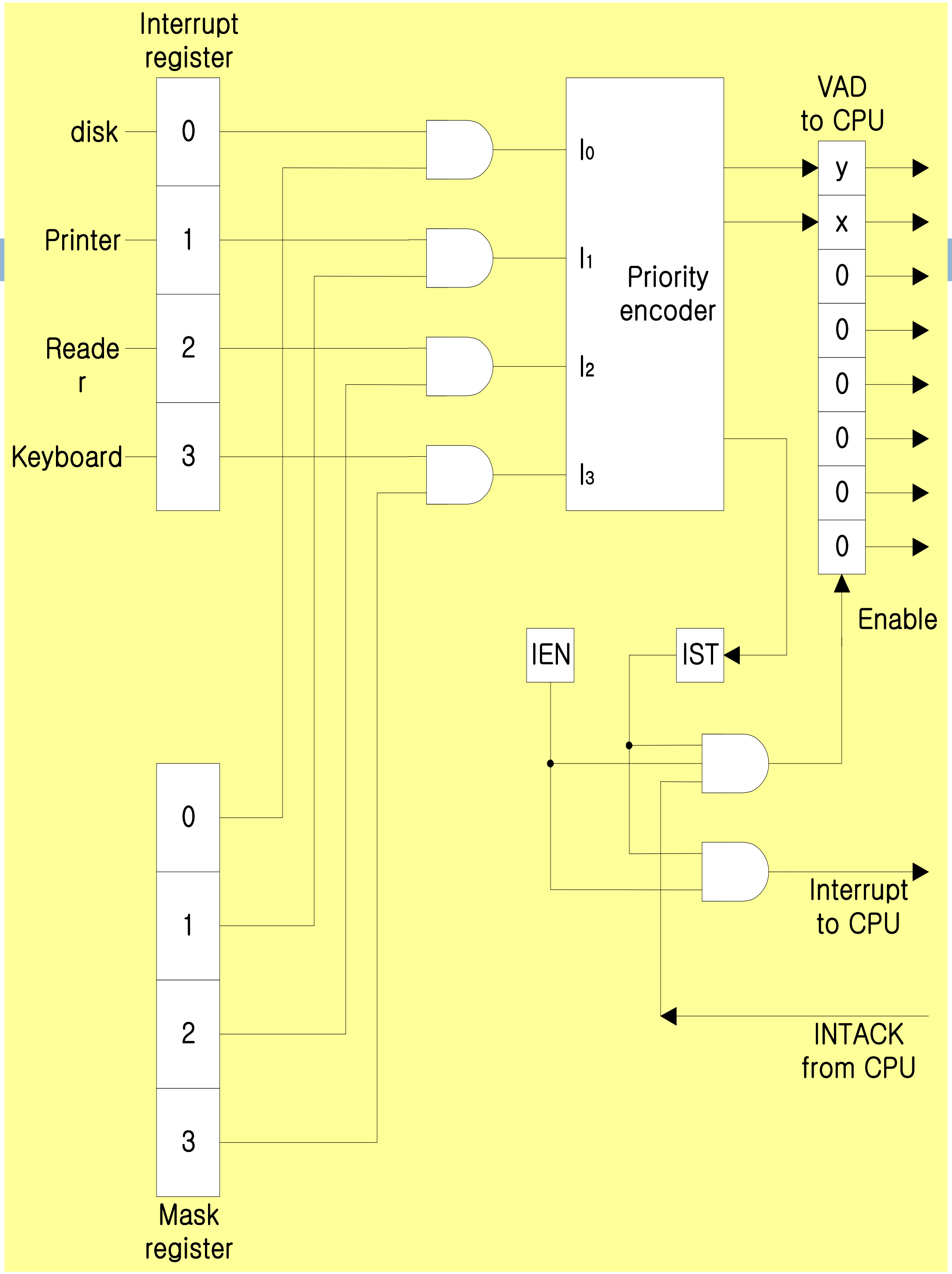


	PI	RF	PO	Enable
①	0	0	0	0
②	0	1	0	0
③	1	0	1	0
④	1	1	0	1

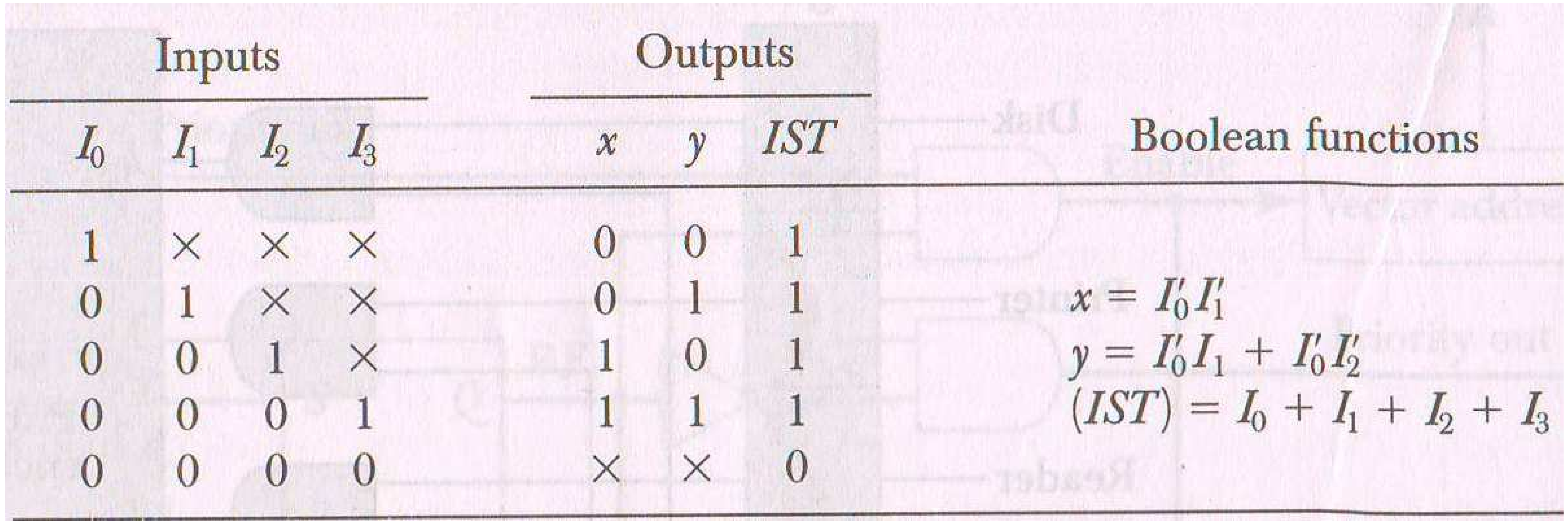
Parallel Priority

- Priority Encoder for 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 Interrupt Hardware



Priority Encoder Truth Table



Inputs				Outputs			Boolean functions
I_0	I_1	I_2	I_3	x	y	IST	
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	

Interrupt Cycle

- Interrupt Cycle
  - At the end of each instruction cycle, CPU checks IEN and IST
  - if both IEN and IST equal to "1"
  - CPU goes to an Instruction Cycle
    - Sequence of micro-operation during Instruction Cycle (next slide)

Interrupt Cycle

$SP \leftarrow SP - 1$

$M[SP] \leftarrow PC$

$INTACK \leftarrow 1$

Branch to ISR

$PC \leftarrow VAD$

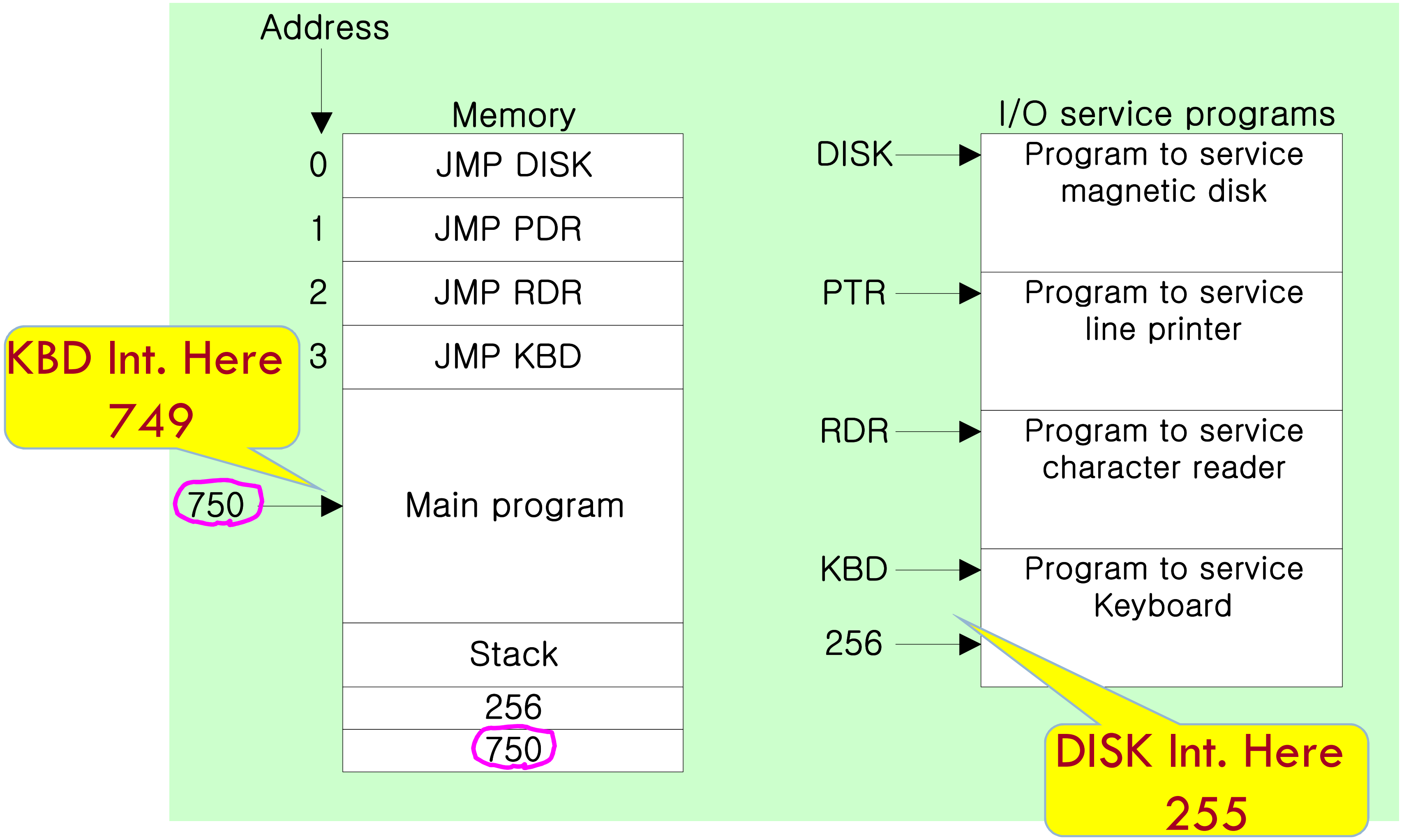
$IEN \leftarrow 0$

Go to Fetch next instruction

Software Routines

- Software Routines
  - CPU executing main program at address 749 when KBD interrupt arrives
  - KBD service program is at address 255 when DISK interrupt arrives

Software Routines



Initial Operations of ISR

- Initial Operation of ISR
  - 1) Clear lower-level mask register bit
  - 2) Clear interrupt status bit IST
  - 3) Save contents of processor registers
  - 4) Set interrupt enable bit IEN
  - 5) Proceed with service routine

Final Operations of ISR

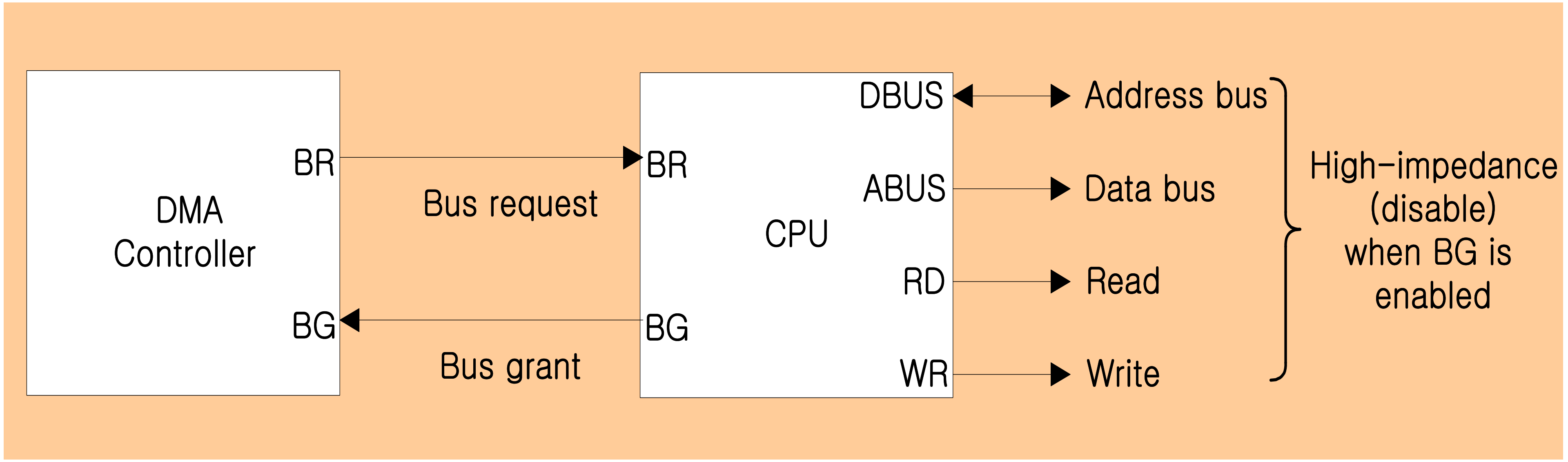
- Final Operation of ISR
  - 1) Clear interrupt enable bit IEN
  - 2) Restore contents of processor registers
  - 3) Clear the bit in the **interrupt register** belonging to the source that has been serviced
  - 4) Set lower-level priority bits in the mask register
  - 5) Restore return address into PC and set IEN

- The next part on DMA will not be included for the MTE 2 (no questions from that part will come)
- There will be NO questions from the MTE 1 part as well

Direct Memory Access (DMA)

- Removes CPU from the picture
- Peripheral devices directly manage memory buses
  - Improves data transfer speed
- Data is transferred from the I/O device directly to the memory
- DMA Controller takes over the buses from CPU

Direct Memory Access (DMA)



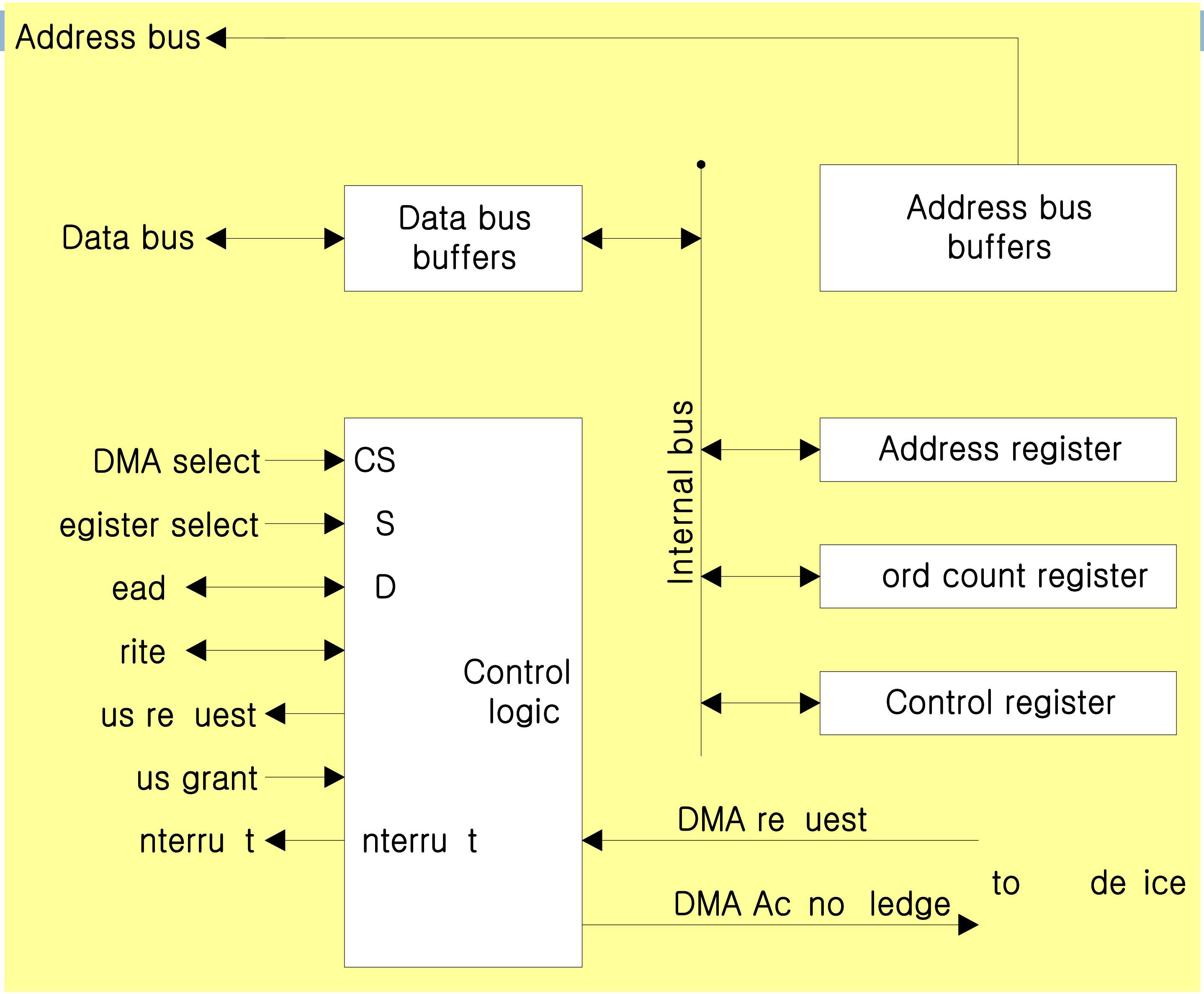
DMA Controller

- Transfer Modes
  - 1) Burst transfer : Blocks of data are sent
  - 2) Cycle stealing transfer : 1 byte sent in a memory cycle
- DMA Controller
  - DMA Initialization Process
    - 1) Set Address register :
      - memory address for read/write
    - 2) Set Word count register :
      - the number of words to transfer

DMA Controller

- 3) Set transfer mode :
  - read/write,
  - burst/cycle stealing,
  - etc
- 4) DMA transfer start
- 5) EOT (End of Transfer) :
  - Interrupt

DMA Controller



Initializing DMA

- CPU initializes DMA by sending the following:
  - The starting address of the memory block where data are available (for read) or where data are to be stored (for write)
  - The word count => no. of words in the memory block
  - Control to specify the mode of transfer
  - A control to start the DMA transfer

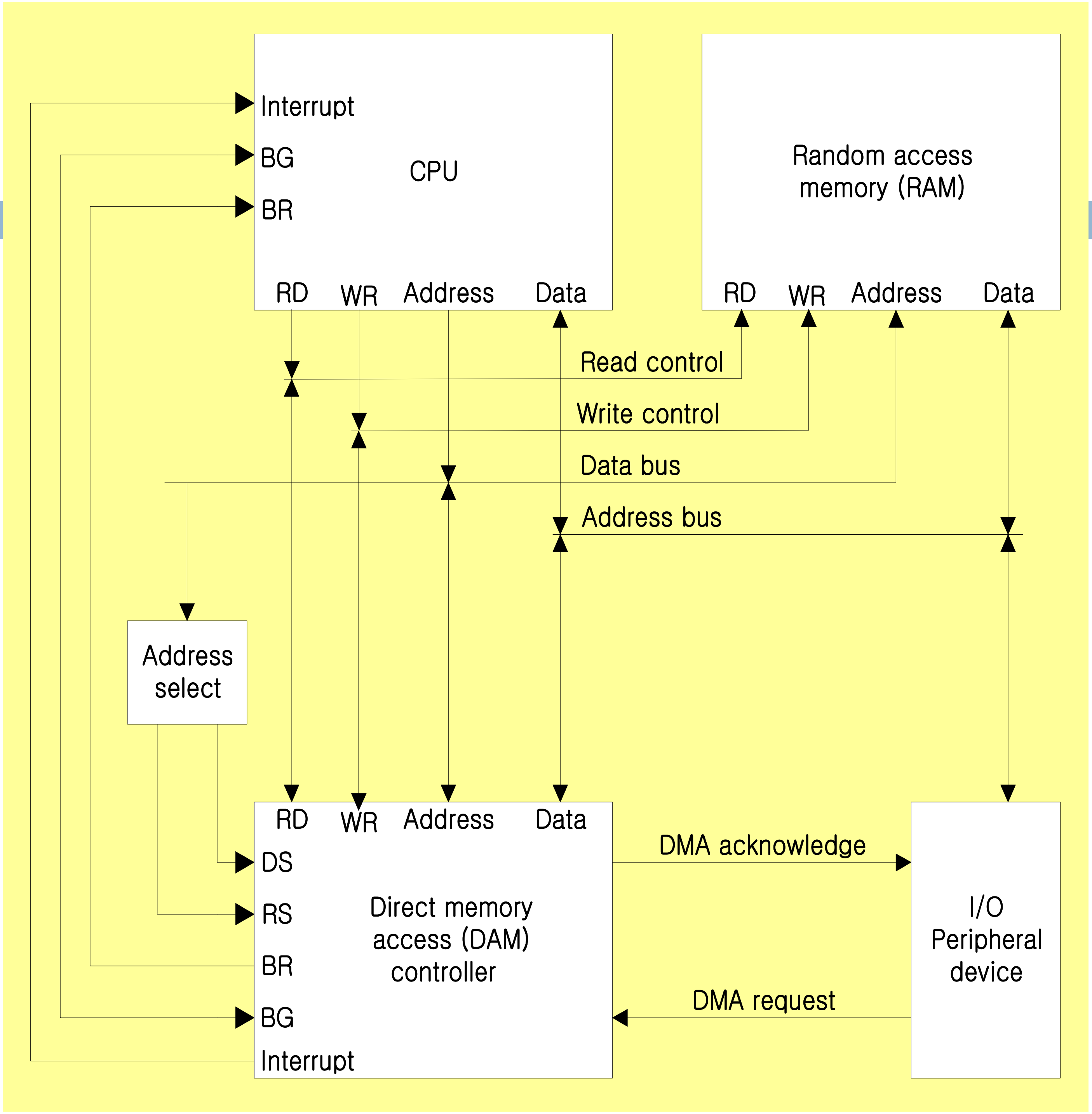
DMA Transfer

- 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 puts the current value of the address register into the address bus and initiates the RD or WR signal
  - 5) DMA acknowledge to the I/O device

DMA Transfer

- 5) I/O device puts a word in the data bus (*for memory write*)
- 6) Memory write takes place at the address specified by **Address register**
- 7) DMAC decrements **Word count register**
- 8) **Word count register** = 0 => **EOT** interrupt
- 9) **Word count register** ≠ 0

    DMAC checks the DMA request from I/O device


Interrupt
BG
BR
CPU
RD
WR
Address
Data
Random access memory (RAM)
RD
WR
Address
Data
Read control
Write control
Data bus
Address bus
Address select
RD
WR
Address
Data
DS
RS
BR
BG
Interrupt
Direct memory access (DAM) controller
DMA acknowledge
DMA request
I/O Peripheral device