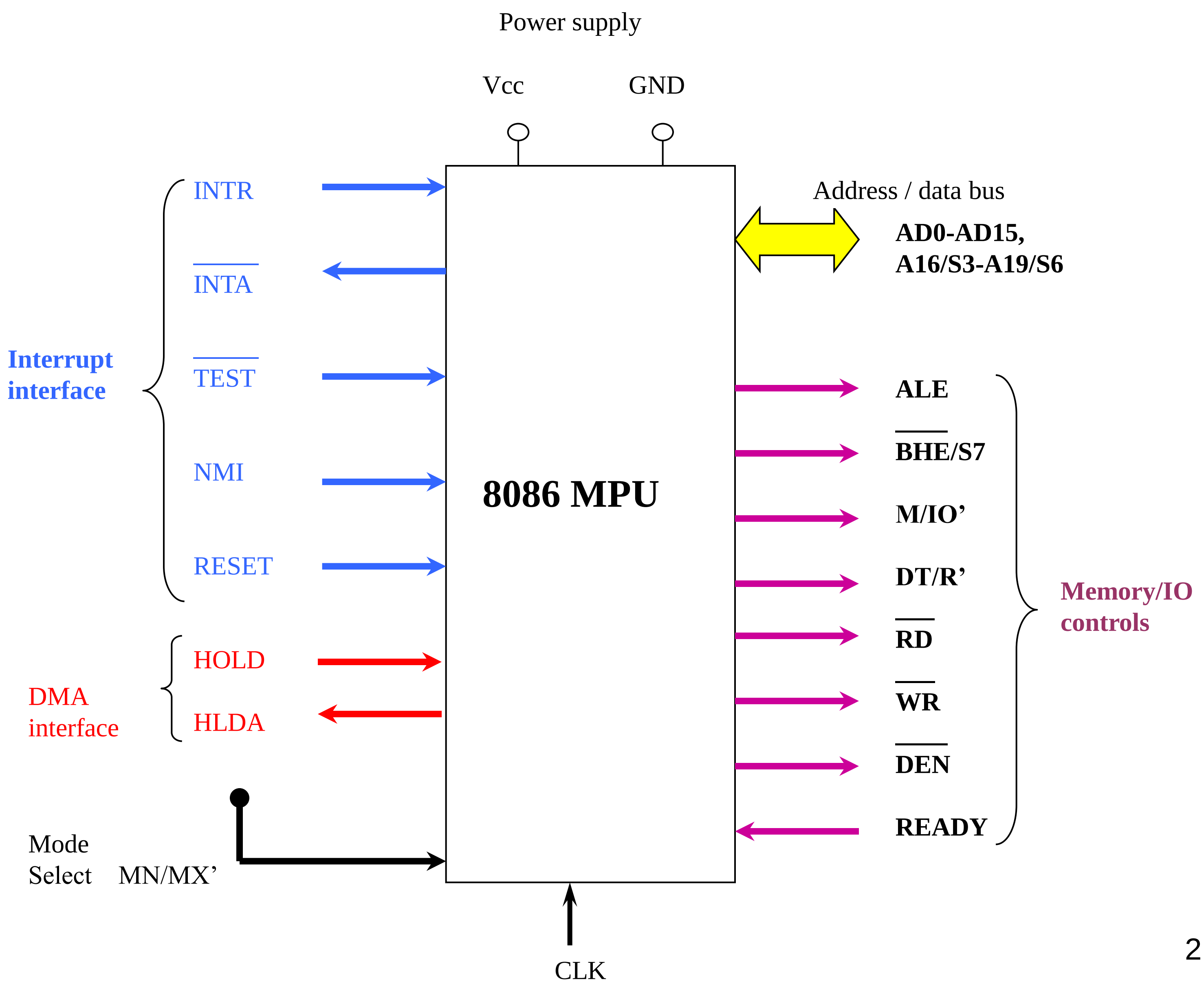
8086 Interrupt & DMA Interface

8086 Minimum-Mode Signals



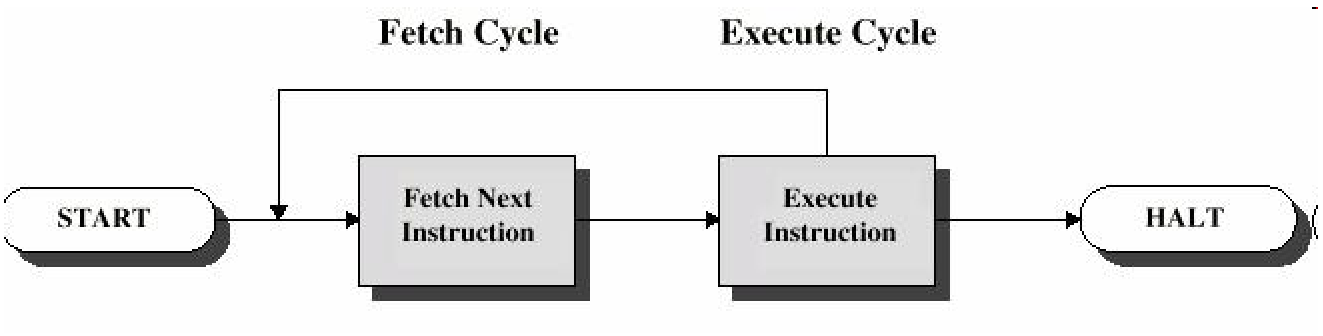
Interrupt

- An **interrupt** is a signal to the processor emitted by hardware or software indicating an event that needs immediate attention.

- 8088/8086 Interrupt Interface signals

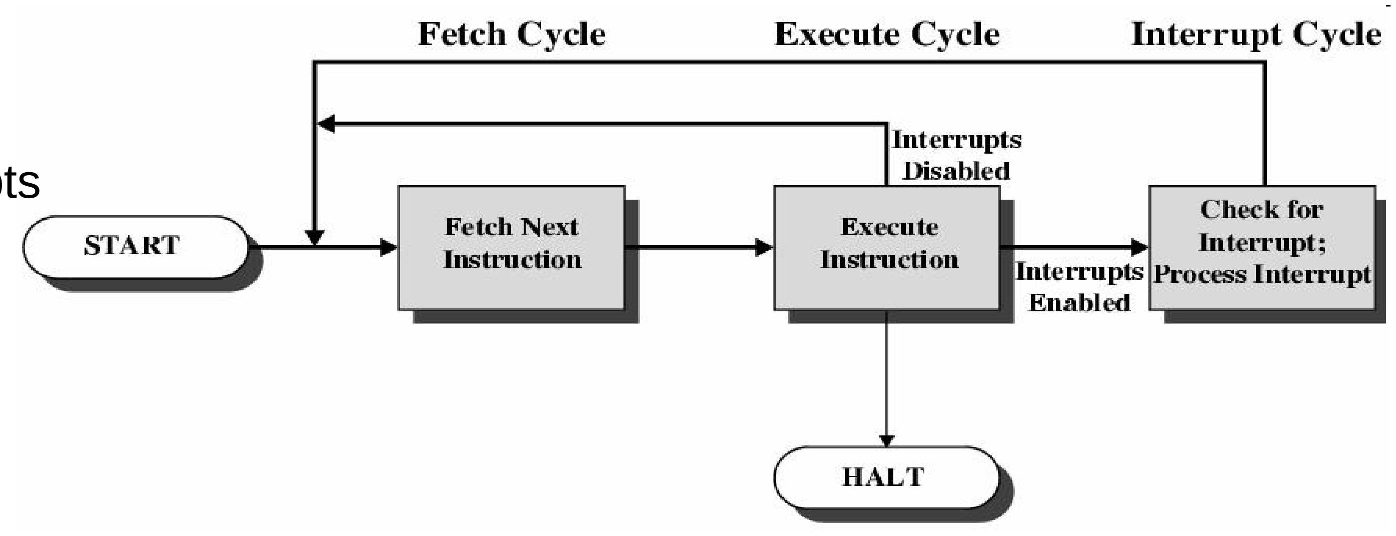
  - INTR – Interrupt Request
  - $\overline{\text{INTA}}$ – Interrupt Acknowledge
  - $\overline{\text{TEST}}$ – Test (can be use to synchronize MPU)
  - NMI – Nonmaskable Interrupt
  - RESET – Reset (hardware reset of the MPU)

Instruction Cycle

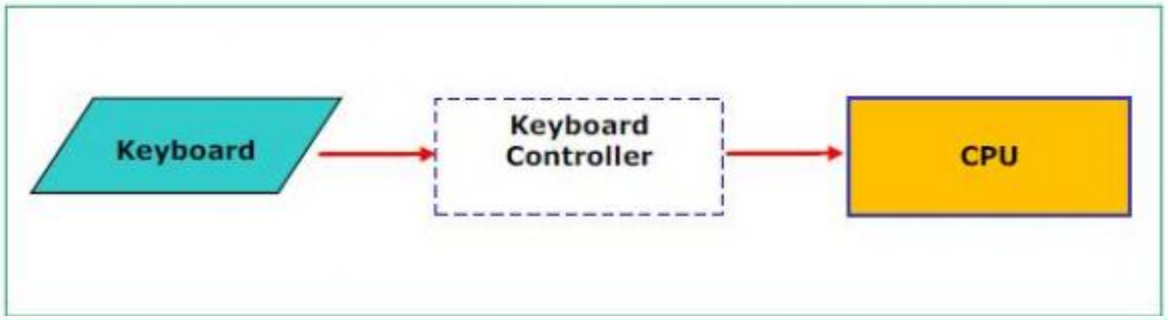


Without Interrupts

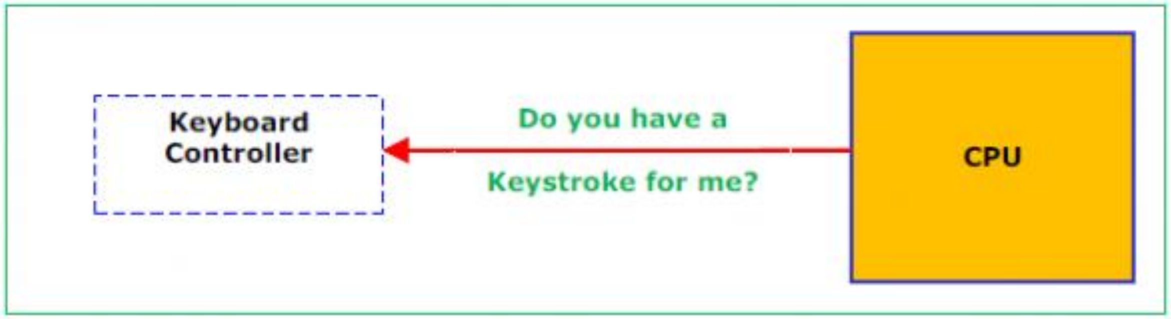
With Interrupts

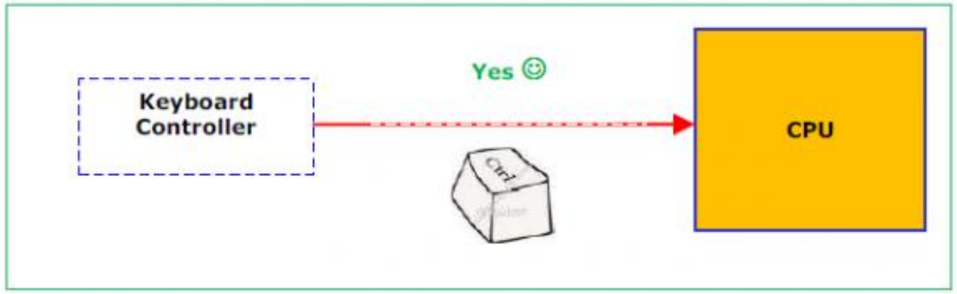


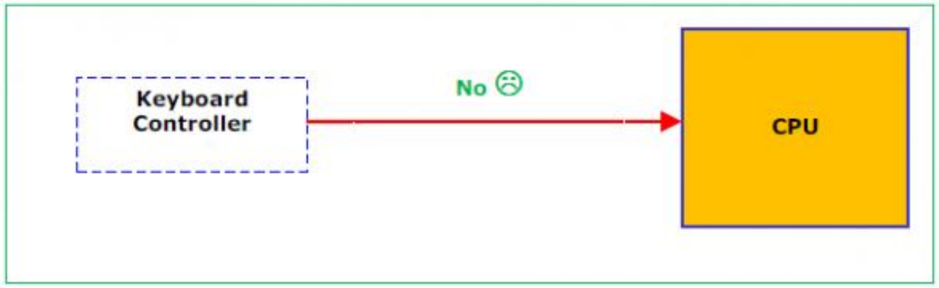
Need for Interrupts
Polling vs Interrupt



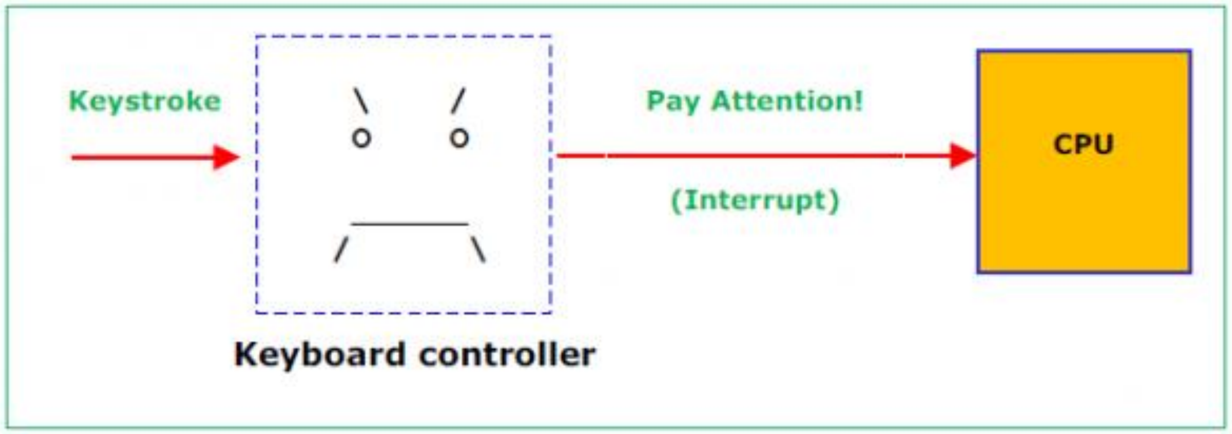
Polling





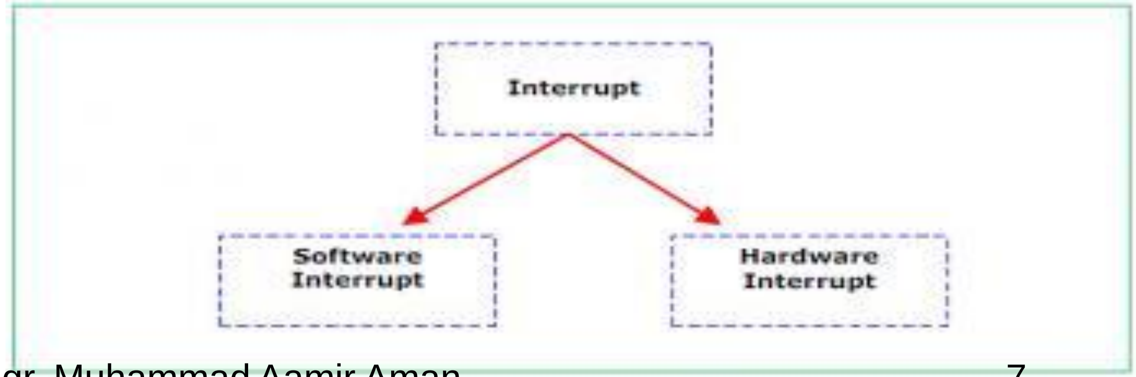


Interrupt saves processor time



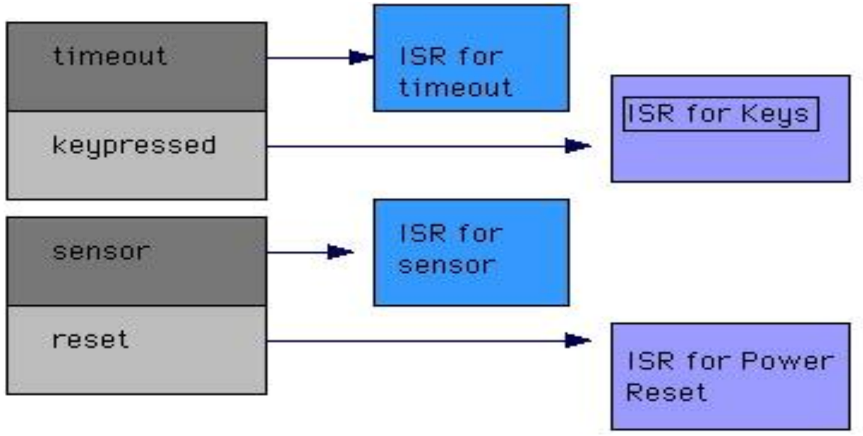
Types of interrupts

- **Maskable interrupt** (INTR): a hardware interrupt that may be ignored by setting Interrupt flag.
- **Non maskable interrupt:** (NMI): a hardware interrupt that can never be ignored. NMIs are used for the highest priority tasks such as timers, & memory hardware errors.
- **Software interrupt**: an interrupt generated within a processor by executing an instruction.



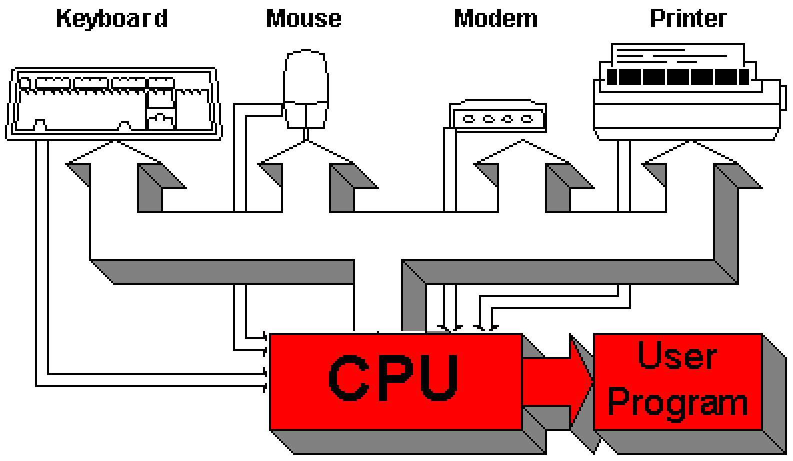
Interrupt Handler (IH) or Interrupt Service Routine (ISR)

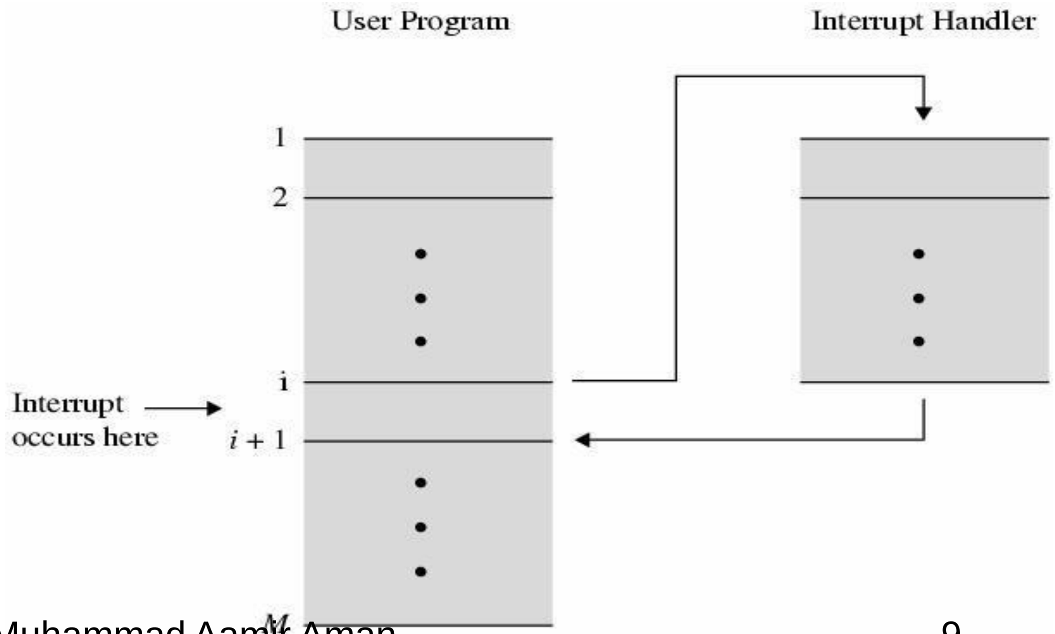
- It is a small program used to service the interrupting device (hardware or software).
- Each interrupting device has different service needs, so there are many service routines present in CS of memory.



Program flow control after interrupt

- When an interrupt service request reaches the CPU, it causes the CPU to interrupt the task that it is doing, and run interrupt service routine (ISR).
- When the ISR is completed, the CPU returns to the program it was executing before interrupt and resumes its previous task. So the stack is again used by CPU to store its program counter & registers temporarily.

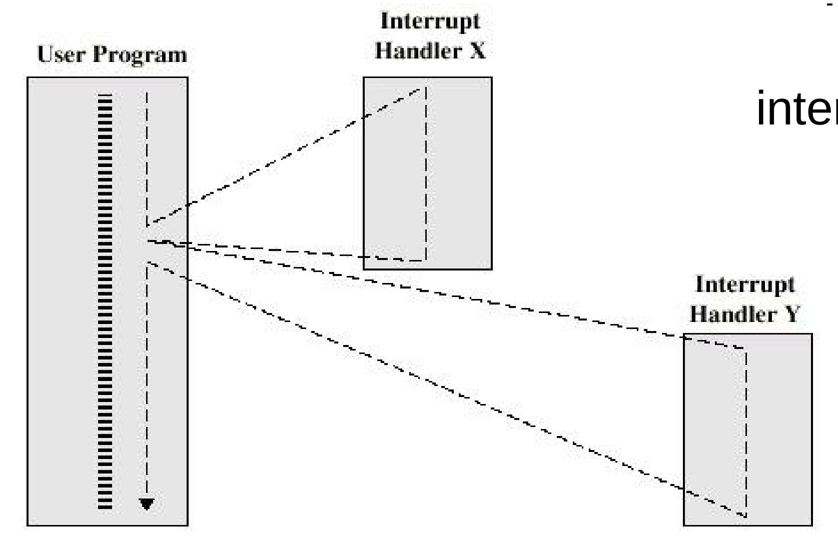




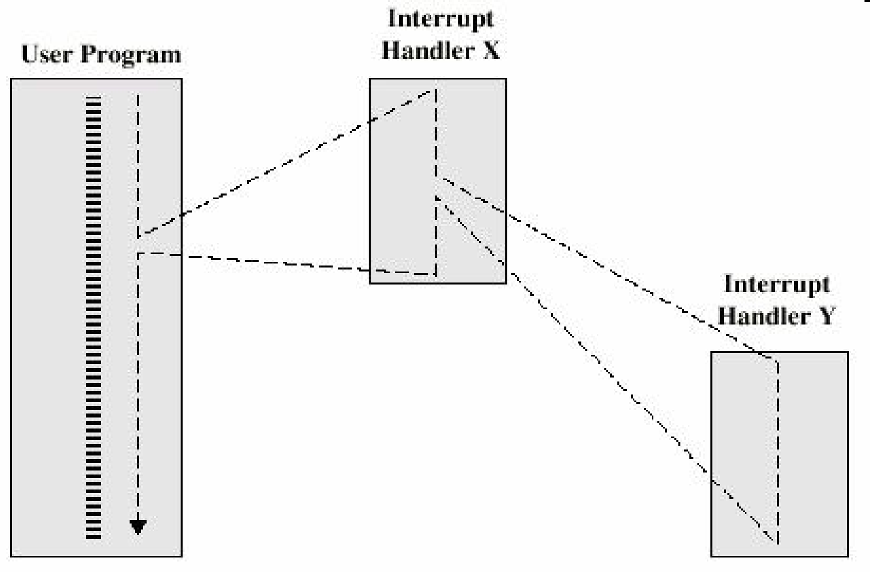
Multiple Interrupts

- A system can have multiple sources of interrupts. The PIC (Programmable Interrupt Controller) chip is used to handle multiple interrupts on the basis of their priority.
  - Once an IRQ is accepted by the CPU, it must identify the source of the request and locate the appropriate ISR in memory.
  - The PIC identifies the IRQ responsible for the interrupt and returns an 8-bit number identity of device called interrupt type code.
  - The CPU uses this type code to locate an entry in the Interrupt Vector Table (IVT). This table holds the starting address for all possible ISRs in main memory.

Sequential & Priority handling of interrupts

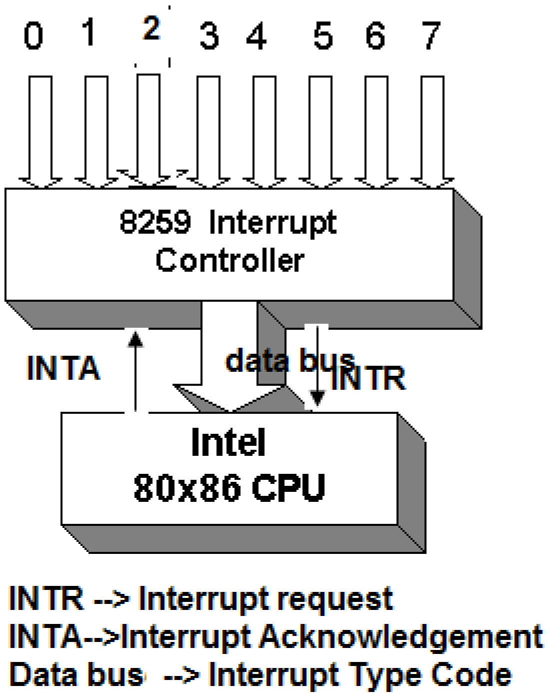


interrupt Y comes after X → Sequential service



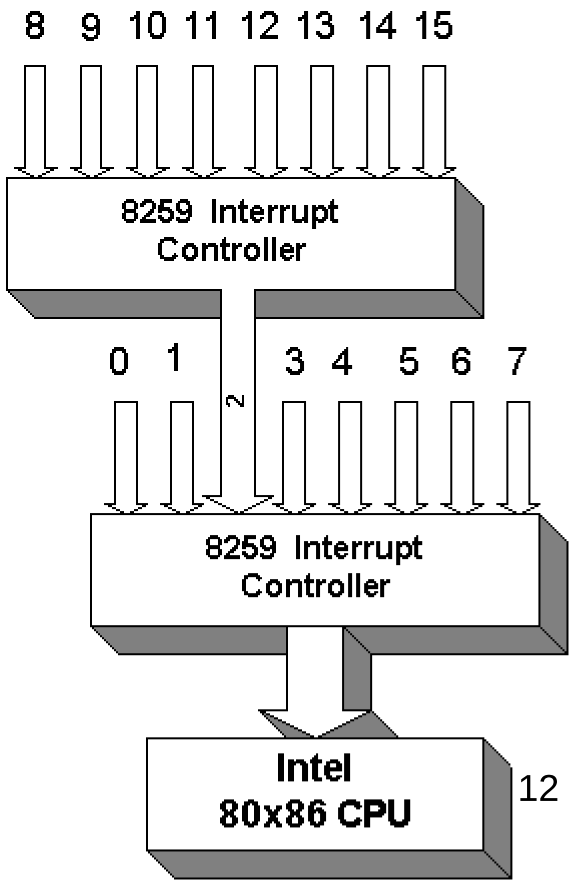
Interrupt Y priority is greater than X, Y comes after X → Priority service

Programmable Interrupt Controller 8259A for multiple interrupting devices



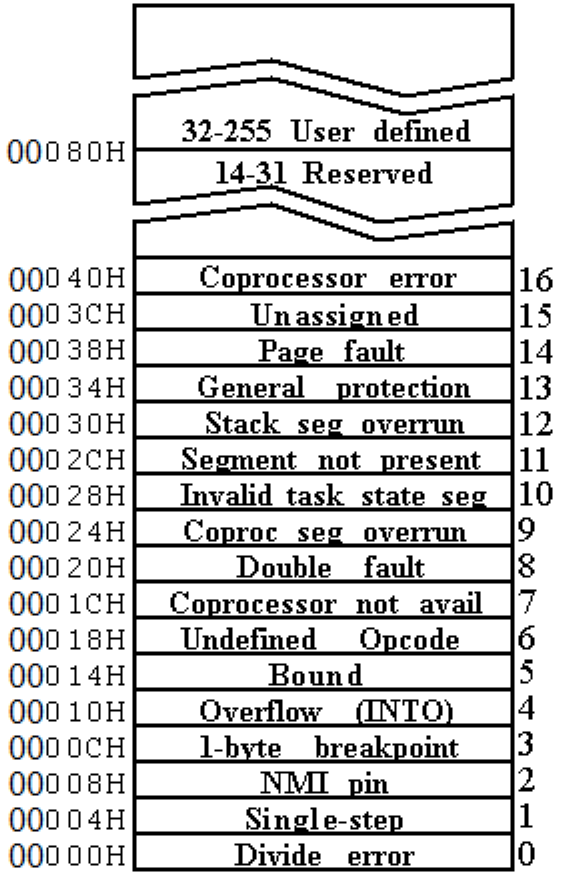
INTR --> Interrupt request
INTA-->Interrupt Acknowledgement
Data bus --> Interrupt Type Code

Cascaded PICs for connecting more than 8 interrupting devices



8086/8088 supports 256 interrupting devices
because interrupt type code is only 8 bits & **2^8** is 256

Interrupt Vector/ pointer Table

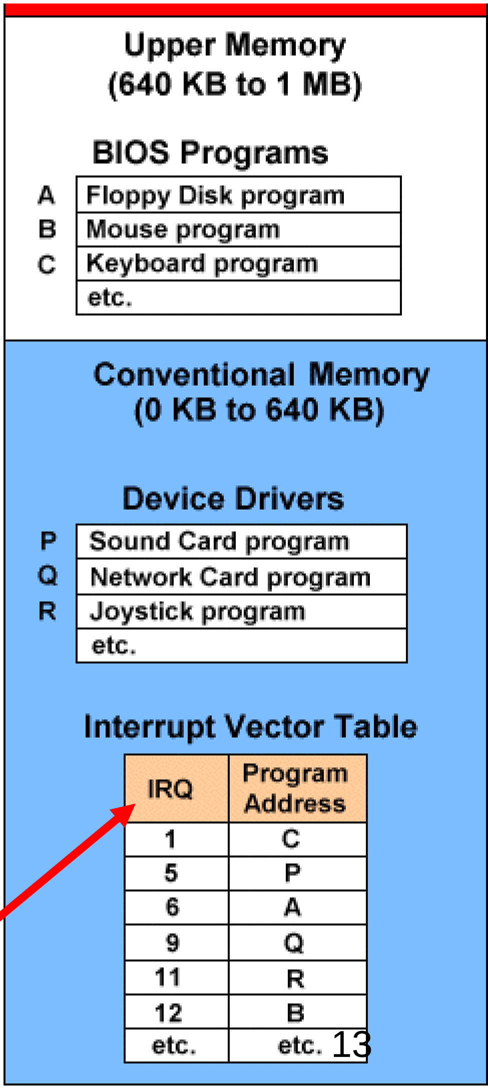


00080H	32-255 User defined	
	14-31 Reserved	
00040H	Coprocessor error	16
0003CH	Unassigned	15
00038H	Page fault	14
00034H	General protection	13
00030H	Stack seg overrun	12
0002CH	Segment not present	11
00028H	Invalid task state seg	10
00024H	Coproc seg overrun	9
00020H	Double fault	8
0001CH	Coprocessor not avail	7
00018H	Undefined Opcode	6
00014H	Bound	5
00010H	Overflow (INTO)	4
0000CH	1-byte breakpoint	3
00008H	NMI pin	2
00004H	Single-step	1
00000H	Divide error	0

The interrupt vector table is located in the first 1024 bytes of memory at addresses 00000H through 003FFH.

There are 256 4-byte entries (segment and offset in real mode).

Seg high	Seg low	Offset high	Offset low
Byte 3	Byte 2	Byte 1	Byte 0



Upper Memory
(640 KB to 1 MB)

BIOS Programs

A	Floppy Disk program
B	Mouse program
C	Keyboard program
	etc.

Conventional Memory
(0 KB to 640 KB)

Device Drivers

P	Sound Card program
Q	Network Card program
R	Joystick program
	etc.

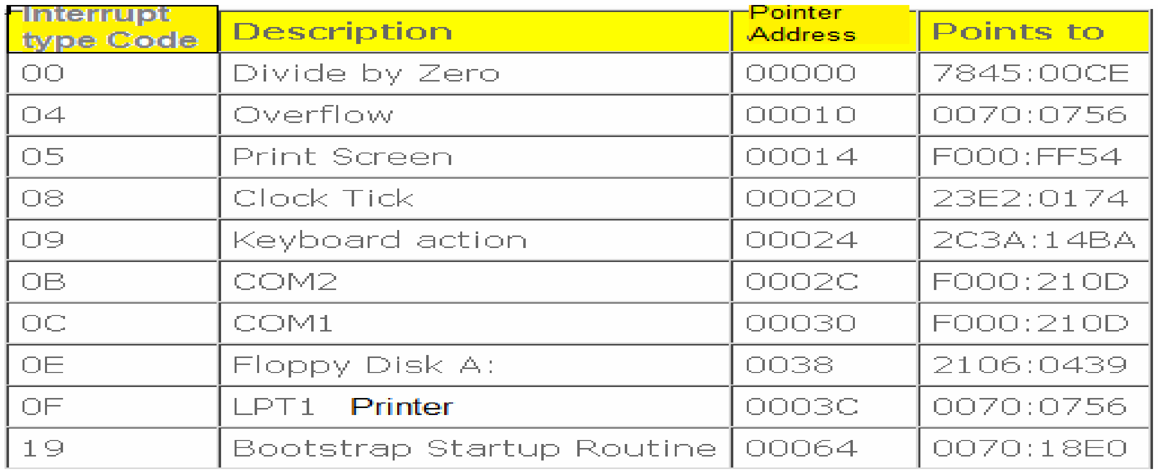
Interrupt Vector Table

IRQ	Program Address
1	C
5	P
6	A
9	Q
11	R
12	B
etc.	etc.

Interrupt Type Codes

Interrupt vector table (IVT)

- It is a table in the lowest 1 KB of memory which contain pointers to ISRs.
- The Type code received from PIC on data bus is multiplied by 4, to get the physical address from where it search for the pointer of ISR in this table for the device which had sent the interrupt request to CPU.



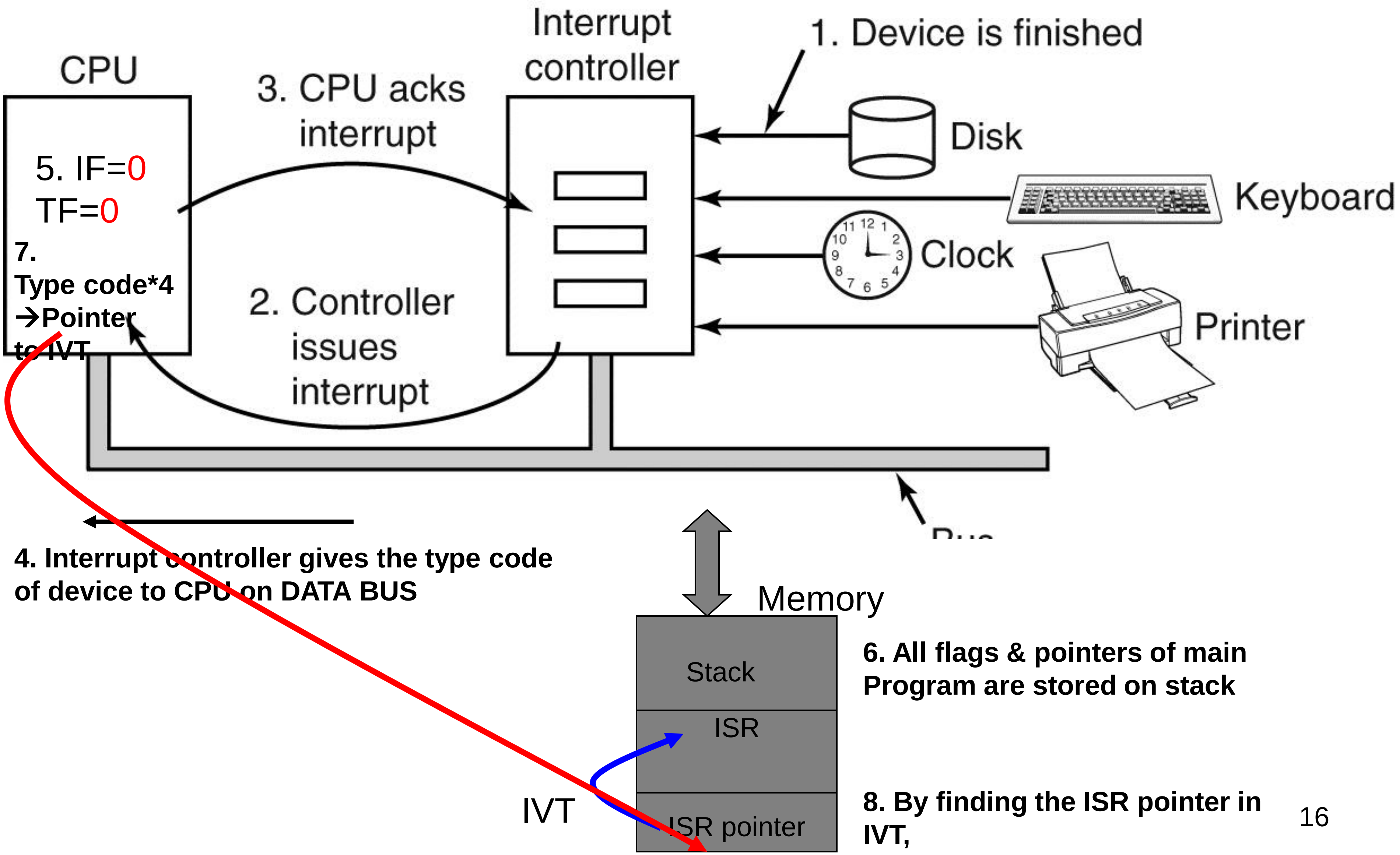
Interrupt type Code	Description	Pointer Address	Points to
00	Divide by Zero	00000	7845:00CE
04	Overflow	00010	0070:0756
05	Print Screen	00014	F000:FF54
08	Clock Tick	00020	23E2:0174
09	Keyboard action	00024	2C3A:14BA
0B	COM2	0002C	F000:210D
0C	COM1	00030	F000:210D
0E	Floppy Disk A:	0038	2106:0439
0F	LPT1 Printer	0003C	0070:0756
19	Bootstrap Startup Routine	00064	0070:18E0

Interrupts process in 8088/86

The Operation of an Interrupt sequence on the 8086 Microprocessor:

1. PIC sends an interrupt signal, to the Interrupt Request (INTR) pin.
2. The CPU finishes the present instruction and sends Interrupt Acknowledge (INTA) to PIC.
3. The interrupt type N is sent to the µProcessor Unit via the Data bus from the PIC.
4. 5. Both the interrupt (IF) and (TF) flags are cleared. This disables the INTR pin and the trap or single-step feature.
6. The contents of the flags, code segment register (CS) & instruction pointer (IP), related to currently running program are pushed onto the Stack.
7. The interrupt service routine address is fetched by using the interrupt type N. IP part is fetched, from (4 x N) and then placed into the IP and CS part is fetched from (4 x N +2) and then placed into the CS so that the next instruction executes from the interrupt service routine.
8. While returning from the interrupt-service routine by the Interrupt Return (IRET) instruction, the IP, CS and Flag registers are popped from the Stack and return to their state prior to the interrupt.

Steps of Interrupt



4. Interrupt controller gives the type code of device to CPU on DATA BUS

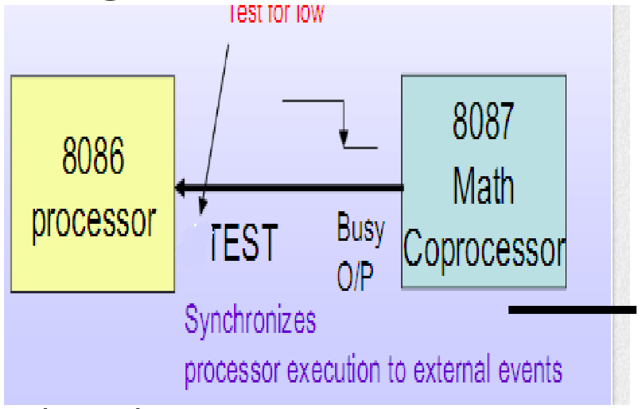
6. All flags & pointers of main Program are stored on stack

8. By finding the ISR pointer in IVT,
the program control shifts to ISR

Interrupt Interface signals Test signal

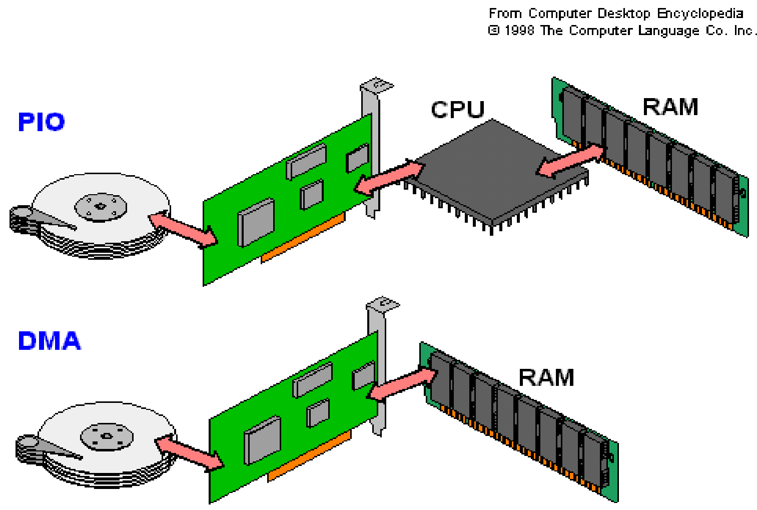
- Test input: Example

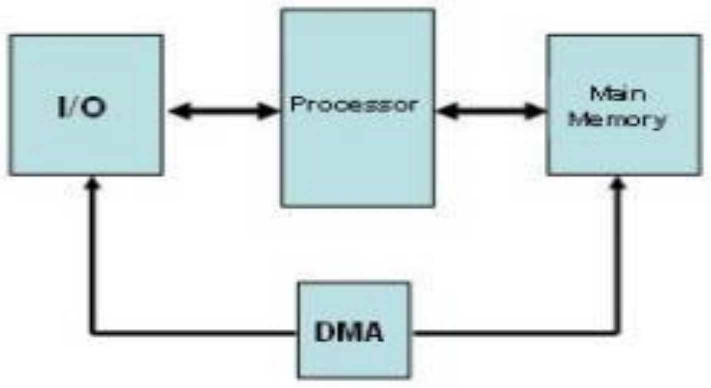
The 8086 cannot carry out floating point arithmetic. So before handling this task to 8087(Math Co-processor), the programmer has to include WAIT instruction in the program. The 8086 then keeps on testing the TEST input. As soon as it becomes low, it indicates that FP math processor has finished the Floating Point arithmetic.



Direct Memory Access DMA

- For large quantities of data or higher speed transfer, the direct memory access approach is preferred. Direct memory access (DMA) frees the CPU from managing the transfer of data between an IO device and main memory.


From Computer Desktop Encyclopedia
© 1998 The Computer Language Co. Inc.
PIO
CPU
RAM
DMA
RAM



DMA interface signals

1. HOLD:
   – When the DMA controller (DMAC) wants to take control of the system buses, it signals this fact to µP by switching HOLD to 1 logic level.
2. HOLA (Hold Acknowledge):
   – The µP releases the control of the buses & signals this fact to DMAC by switching its HLDA pin to 1 logic level.

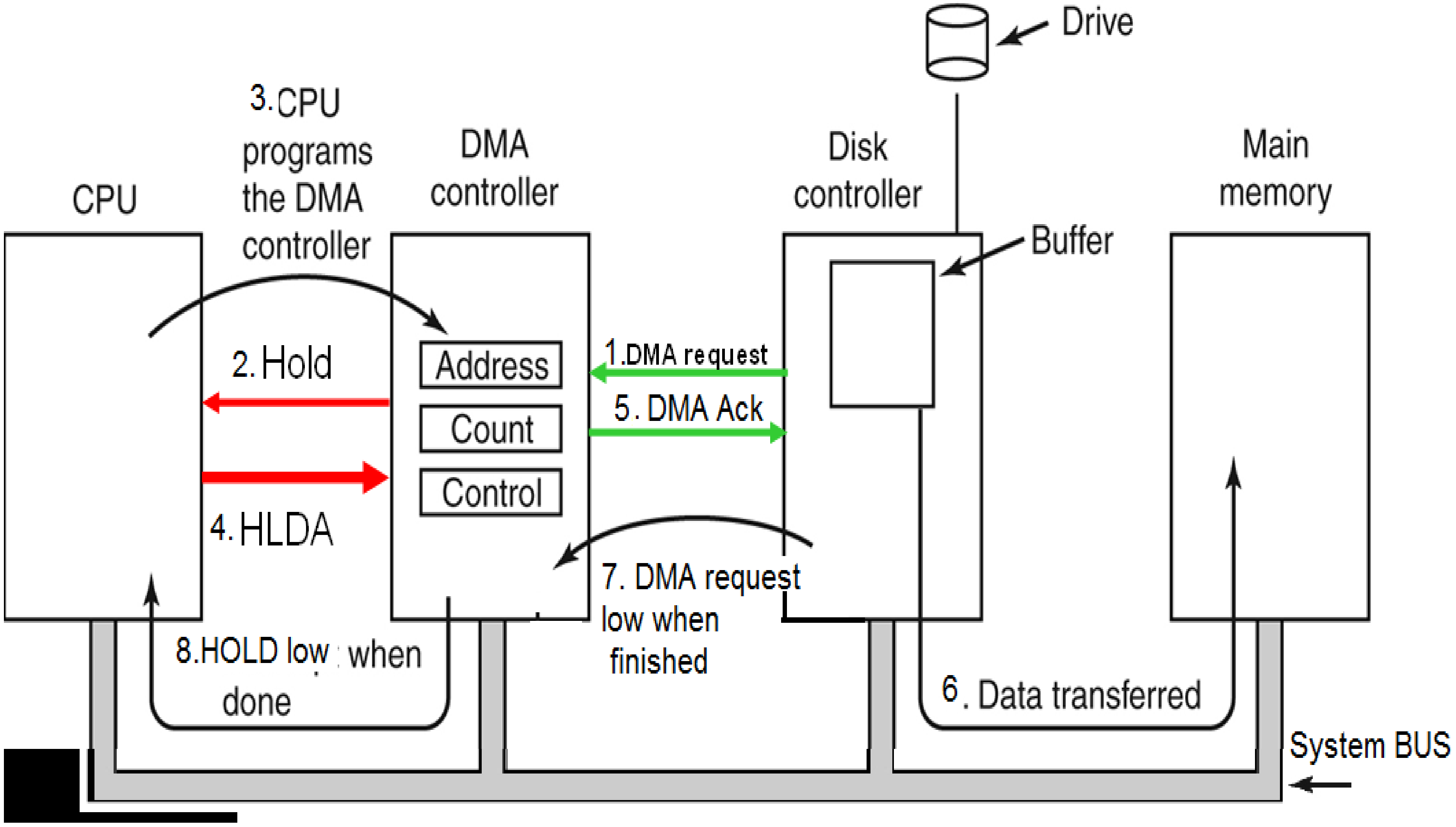
DMA Process

1. The Device sends DMAC request for data transfer.
2. The DMAC sends high on HOLD input of CPU.
3. The CPU programs the DMA controller

   CPU tells DMA controller:-

   – I/O Device address
   – Starting address of main memory block for data
   – Amount of data to be transferred
4. CPU sends HLDA to DMAC & release the control of System bus.
5. DMA controller sends ACK to the device which sends it request initially.
6. DMA deals with transfer.
7. When done, the device sends low on DREQ pin to DMAC.
8. The DMAC sends low on HOLD pin , telling CPU to acquire the control of System bus

DMA Process


Drive
3.CPU programs the DMA controller
DMA controller
Disk controller
Main memory
CPU
Buffer
2.Hold
Address
Count
Control
1.DMA request
5. DMA Ack
4. HLDA
7. DMA request low when finished
8.HOLD low when done
6. Data transferred
System BUS

Quiz # 01

- What do you know about Status Signals?

Quiz # 02

- How a word is transferred by 8088 microprocessor? Explain your answer with proper example.