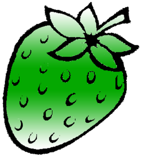
STRAWBERRY



/strawberrydevelopers

/strawberry_app

For more visit:

Strawberrydevelopers.weebly.com

INTERRUPTS

Richa Upadhyay Prabhu

NMIMS's MPSTME

richa.upadhyay@nmims.edu

February 2, 2016

Introduction

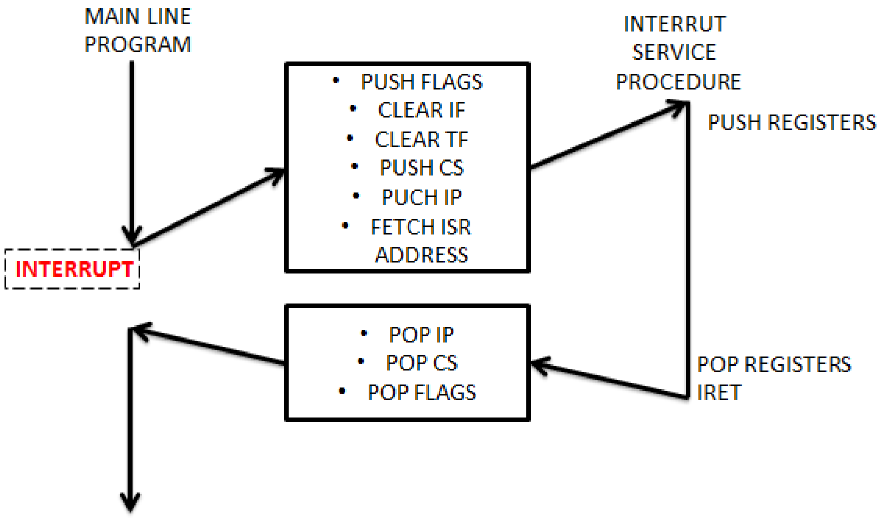
- Microprocessors allow normal program to be interrupted by some external signal or special functions
- As a response to interrupt, it stops executing current program
- Calls a procedure which **services** the interrupt
- These are called as **interrupt service procedures**(ISP)
- An IRET instruction at the end of ISP returns the execution to interrupted program

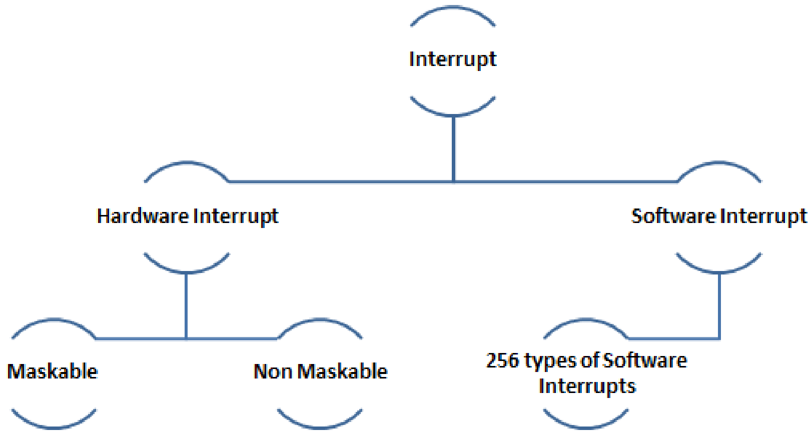
8086 Interrupts

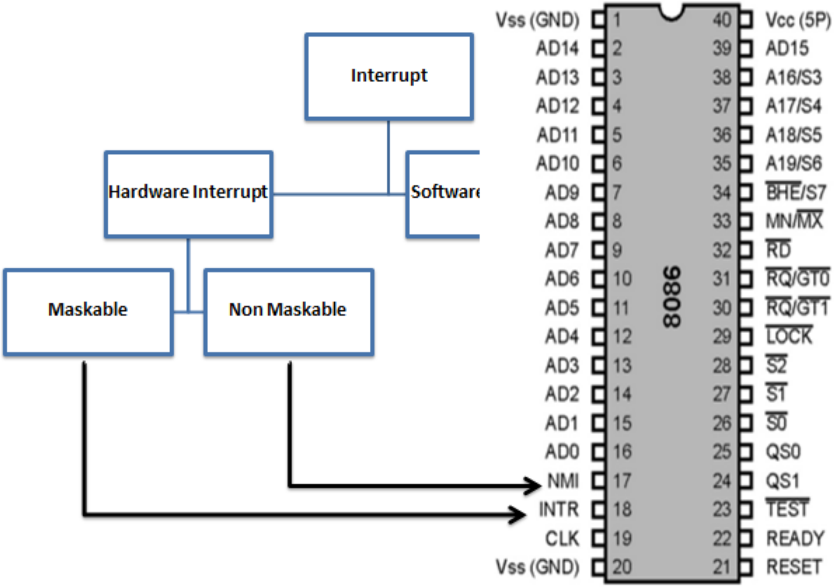
An interrupt can come from any of these three sources:

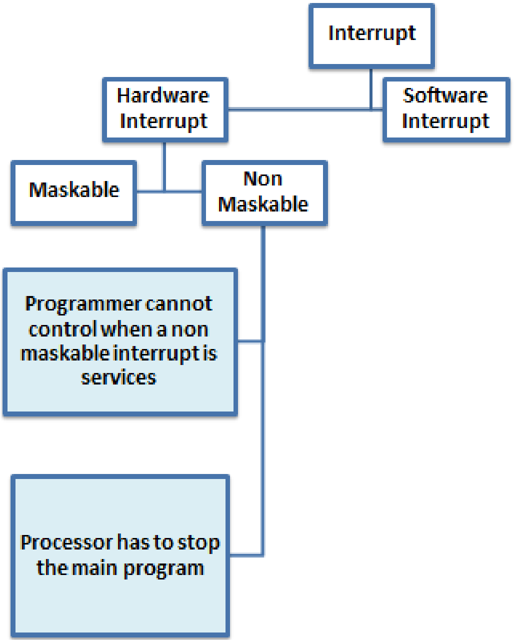
1. Hardware Interrupt
   - an external signal applied to *nonmaskable interrupt* **(NMI)** input pin or *Interrupt* **(INTR)** input pin
2. Software Interrupt
   - Execution of Interrupt instruction : INT
3. Error Condition produced during 8086 (e.g. divide by zero)

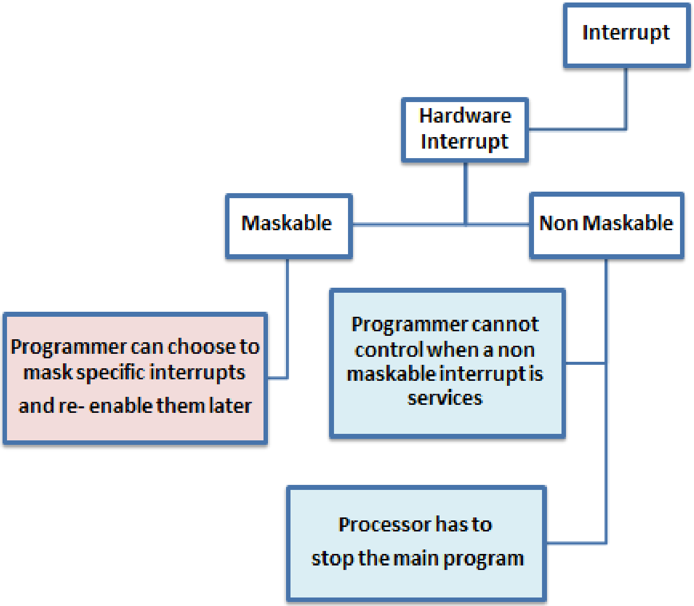
8086 interrupt response

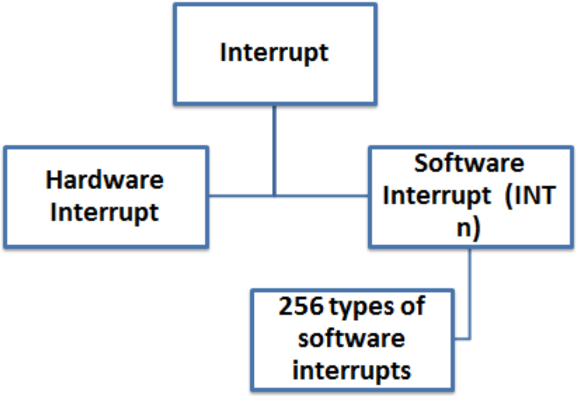



Interrupt
Hardware Interrupt
Software Interrupt
Maskable
Non Maskable
256 types of Software Interrupts


Interrupt
Hardware Interrupt
Software
Maskable
Non Maskable
Vss (GND) 1
AD14 2
AD13 3
AD12 4
AD11 5
AD10 6
AD9 7
AD8 8
AD7 9
AD6 10
AD5 11
AD4 12
AD3 13
AD2 14
AD1 15
AD0 16
NMI 17
INTR 18
CLK 19
Vss (GND) 20
8086
40 Vcc (5P)
39 AD15
38 A16/S3
37 A17/S4
36 A18/S5
35 A19/S6
34 BHE/S7
33 MN/MX
32 RD
31 RQ/GT0
30 RQ/GT1
29 LOCK
28 S2
27 S1
26 S0
25 QS0
24 QS1
23 TEST
22 READY
21 RESET


Interrupt
Hardware Interrupt
Software Interrupt
Maskable
Non Maskable
Programmer cannot control when a non maskable interrupt is services
Processor has to stop the main program

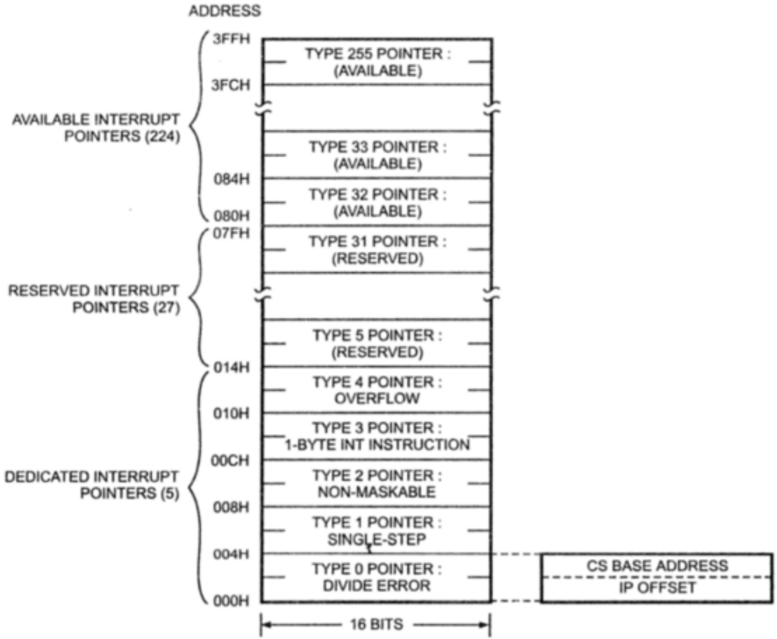

Interrupt
Hardware Interrupt
Maskable
Non Maskable
Programmer can choose to mask specific interrupts and re- enable them later
Programmer cannot control when a non maskable interrupt is services
Processor has to stop the main program



INT 00H to INT 0FFH comprises 256 kinds of Interrupts

Software Interrupts INT n

- In 8086, the first 1Kbyte of memory is reserved as a table for storing the starting addresses of interrupt service procedures
- 4 bytes are required to store CS:IP values of each interrupt service procedures
- So,1 Kbyte(0000H to 03FFH) can hold starting addresses for up to 256 interrupt procedures
- Starting address of ISP is often called **Interrupt Vector** or **Interrupt Pointer**
- so the table is referred as **Interrupt Vector Table**


ADDRESS
3FFH
3FCH
084H
080H
07FH
014H
010H
00CH
008H
004H
000H
AVAILABLE INTERRUPT POINTERS (224)
RESERVED INTERRUPT POINTERS (27)
DEDICATED INTERRUPT POINTERS (5)
TYPE 255 POINTER : (AVAILABLE)
TYPE 33 POINTER : (AVAILABLE)
TYPE 32 POINTER : (AVAILABLE)
TYPE 31 POINTER : (RESERVED)
TYPE 5 POINTER : (RESERVED)
TYPE 4 POINTER : OVERFLOW
TYPE 3 POINTER : 1-BYTE INT INSTRUCTION
TYPE 2 POINTER : NON-MASKABLE
TYPE 1 POINTER : SINGLE-STEP
TYPE 0 POINTER : DIVIDE ERROR
CS BASE ADDRESS
IP OFFSET
16 BITS

Interrupt vector table

The 256 interrupts are divided into three groups:

1. **Type 0 to Type 4 Interrupts**
   - These are used for fixed operations and hence are called as DEDICATED INTERRUPTS
2. **Type 5 to Type 31 Interrupts**
   - Are reserved by Intel for higher processors
3. **Type 32 to Type 255 Interrupts**
   - Available for user called as user defined interrupts. They can either hardware or software interrupts.

8086 Interrupt Types

TYPE '0', DIVIDE BY ZERO INTERRUPT

- 8086 will automatically do a Type 0 interrupt if result of division is too large to fit in destination register
- This interrupt is automatic and cannot be disabled

8086 Interrupt Types

TYPE '1', SINGLE STEP INTERRUPT

- If the trap flag is set, then 8086 automatically do a type 1 interrupt after each instruction executes
- Wherein the processor will execute only one instruction and stop, the contents of register and memory location can be monitored after each instruction
- so when in single step mode, system will stop after it executes each instruction and wait for further direction

8086 Interrupt Types

TYPE ’2’, NON MASKABLE INTERRUPT

- 8086 will automatically do a type 2 interrupt response when it receives a low to high transition on its NMI pin
- it cannot be disabled by any instruction

8086 Interrupt Types

TYPE ’3’, BREAKPOINT INTERRUPT

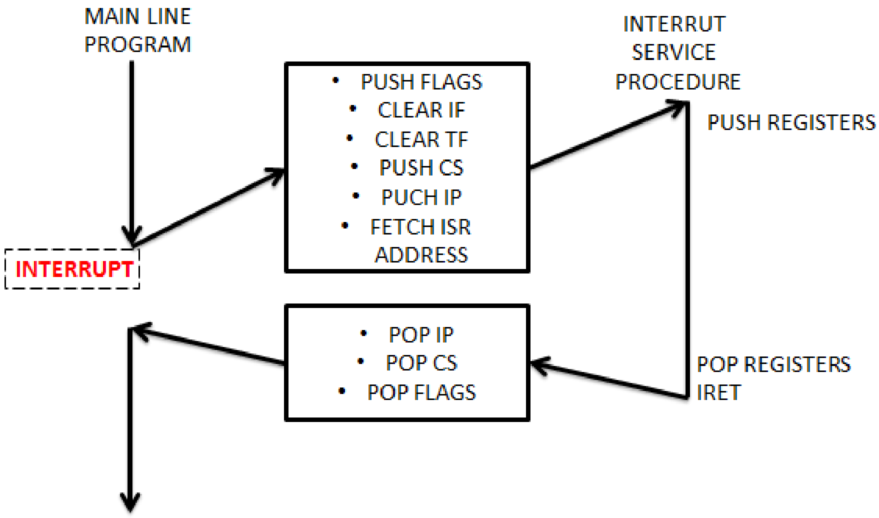
- Type 3 interrupt is produced by execution of a INT 3 instruction
- Implements a breakpoint function in a system
- Breakpoint feature executes all the instructions up to the inserted breakpoint and then stops execution

8086 Interrupt Types

TYPE '4', OVERFLOW INTERRUPT

- There are two ways to detect and respond to an overflow error:
  - Use Jump If Overflow (JO) instruction
  - Use *Interrupt on Overflow* (INTO) Instruction
- if overflow is set,8086 will do a type 4 interrupt after it executes an INTO instruction

8086 interrupt response



8086 Interrupt Response

Step 1:
External interface sends an interrupt signal, to the Interrupt Request (INTR) pin, or an internal interrupt occurs.

- Unlike NMI input, INTR can be disabled by setting IF $= 0$
- Instructions Clear Interrupt (CLI) and Set Interrupt(STI) can be used to reset and set the IF
- If 8086 is reset, IF is automatically cleared

8086 Interrupt Response

Step 2:
The CPU finishes the present instruction (for a hardware interrupt) and sends Interrupt Acknowledge (INTA) to hardware interface.

Step 3:
The interrupt type N is sent to the Central Processor Unit (CPU) via the Data bus from the hardware interface.

8086 Interrupt Response

Step 4:
The contents of the flag registers are pushed onto the stack. Both the interrupt (IF) and (TF) flags are cleared. This disables the INTR pin and the trap or single-step feature.

- IF is automatically cleared, to prevent another interrupt while a current higher priority ISP is in progress
- Also to avoid the 8086 to continuously interrupt itself (since INTR is activated by a high level)

8086 Interrupt Response

Step 5:
The contents of the code segment register (CS) and instruction pointer (IP)are pushed onto the Stack.

Step 6:
The interrupt vector contents are fetched, from (4 x N) and then placed into the IP and from (4 x N +2) into the CS so that the next instruction executes at the interrupt service procedure addressed by the interrupt vector.

8086 Interrupt Response

Step 7:
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.

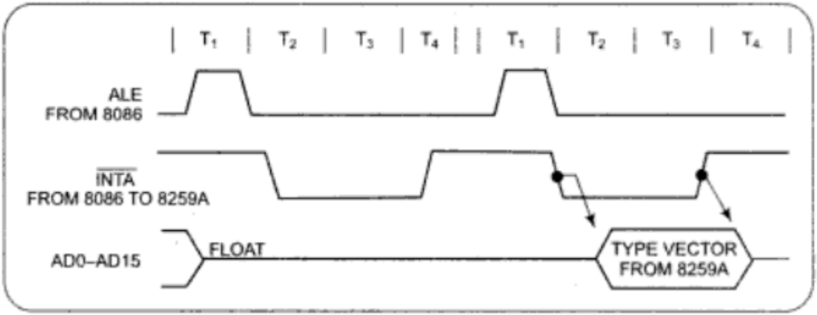
It also re-enables the INTR input (Set IF)

So, make sure that the signal causing interrupt is made low (else it would again cause an interrupt)

Priority Interrupt Controller

- In an INTR input, the input type is sent to 8086 from an external hardware device: **8259A** *Priority Interrupt Controller*
- we will be discussing it later

Interrupt Acknowledge machine cycles



Interrupt Acknowledge machine cycles

- 8086 responds to an interrupt with the above waveform
- It does TWO interrupt acknowledge machine cycles, purpose is to get interrupt type from the external device
- At start of first machine cycle:
    - 8086 floats the data bus
    - sends out an acknowledge pulse on $\overline{INTA}$ pin (This tells 8259 to get ready)
- During the second cycle
    - 8086 send another pulse on $\overline{INTA}$ pin
    - In response to it, 8259 puts the interrupt type on the lower eight lines of data bus, where it is read by 8086

Priority of Interrupts

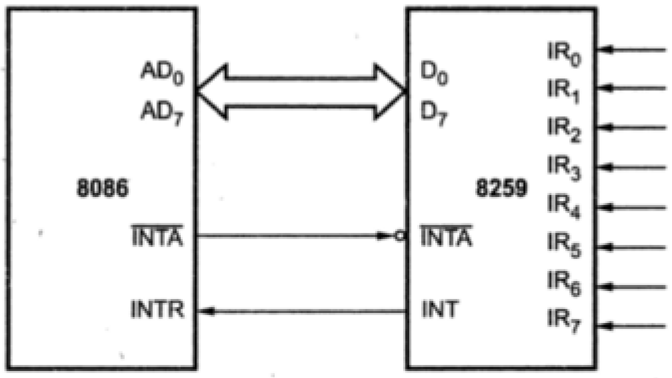
Interrupt	**Priority**
DIVIDE Error, INT n, INTO	Highest
NMI	
INTR	
SINGLE STEP	LOWEST

8259 Priority Interrupt Controller

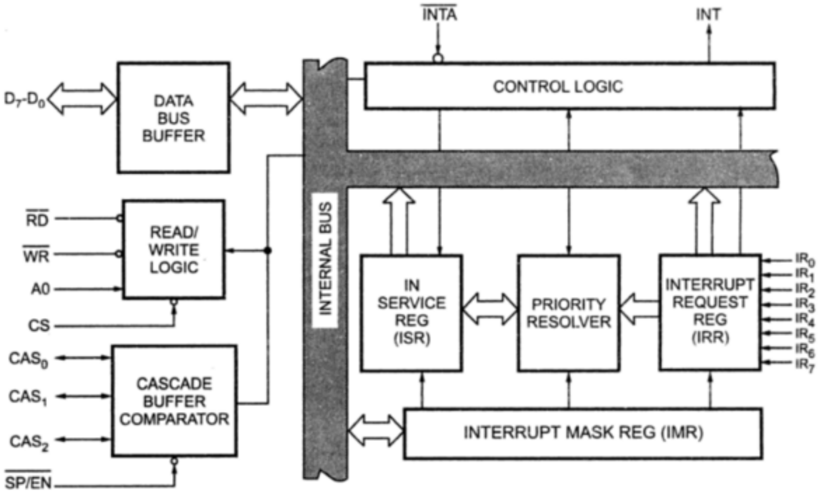
- Interrupts can be used for variety of applications
- 8086 has only two interrupt inputs (NMI and INTR)
- 8259 Priority Interrupt Controller is used to handle multiple interrupt sources
- 8259 "funnel" ~~the~~ the interrupt signals into a single interrupt input
- It can manage eight priority inputs
- It helps to get information regarding pending interrupts, in-service interrupts and masked interrupts
- It is designed to minimize software and real time overhead in handling multilevel priority interrupts

8259 Priority Interrupt Controller

Connection between 8086 and 8259



Block diagram 8259



Block diagram 8259

Internal block diagram of 8259A consists of 8 main blocks :

- data bus buffer
- read/write logic
- control logic
- Three registers (TRR, ISR, IMR)
- priority resolver
- cascade buffer

Block diagram 8259

DATA BUS BUFFER

- 8 bit data bus is used to send interrupt type to 8086
- Used by 8086 to send control word to 8259 and read status word from 8259

READ/WRITE LOGIC

The $\overline{RD}$ and $\overline{WR}$ input control data flow on data bus when the device is selected by asserting its chip select $\overline{CS}$ input low

Block diagram 8259

CONTROL LOGIC

- This block has both an input (INT) and output($\overline{INTA}$) line
- Interrupt request asserts a logic high on INT output pin
- If this is connected to INTR of 8086 and interrupt flag is set
- then this high signal will cause the 8086 to respond INTR

IR_0 **TO** IR_7

These are 8 interrupt input pins. If 8259 is enabled, an interrupt input on any of these pins will cause 8259 to assert INT pin high

Block diagram 8259

INTERRUPT REQUEST REGISTER(IRR)
It keeps track of which interrupt input is asking for service. If there is an interrupt of any of the pins then the corresponding bit goes high.

INTERRUPT SERVICE REGISTER (ISR)
It keeps a track of which interrupt inputs are currently being serviced, by setting the corresponding bit high in in-service register.

INTERRUPT MASK REGISTER (IMR)
It is used to disable or enable individual interrupt inputs. Any input can be masked by setting the corresponding bit high.

Block diagram 8259

PRIORITY RESOLVER
It acts as a judge, that determines if and when an interrupt request on one of the IF input gets serviced.

Remember IR_0 has the highest priority and IR_7 has the lowest.

8259 Interrupt Procedure I

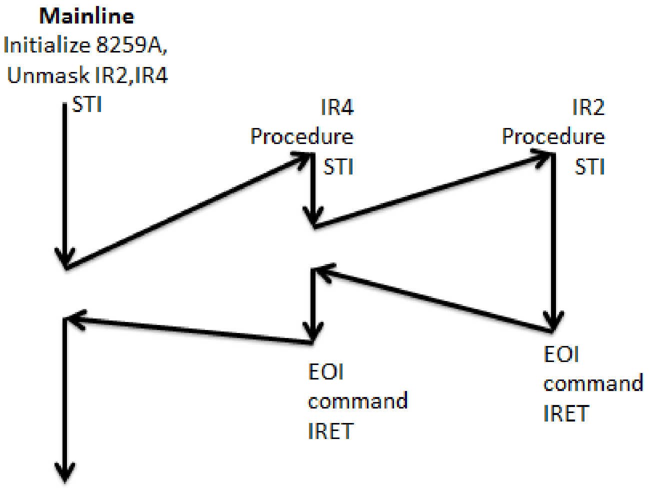
1. One or more of the Interrupt request lines(IR0) to IR7) are raised high, setting the corresponding IRR bits.
2. The priority resolver checks 3 registers: The IRR for interrupt requests, the IMR for masking bits, the ISR for interrupt request being served.
3. It resolves the priority and sets the INT high when appropriate.

8259 Interrupt Procedure II

4. 8086 acknowledges the INT and responds with an $\overline{INTA}$ pulse.
5. 8259 uses first INTA pulse to do some activity related to priority mode and in the second INTA pulse it outputs interrupt type on data bus.
6. Also on receiving INTA, the highest priority ISR bit is set and corresponding IRR bit is reset.
7. After the interrupt is properly serviced by 8086, the ISR bit remains set until an appropriate EOI command (End of Interrupt) is issued at end of interrupt subroutine.

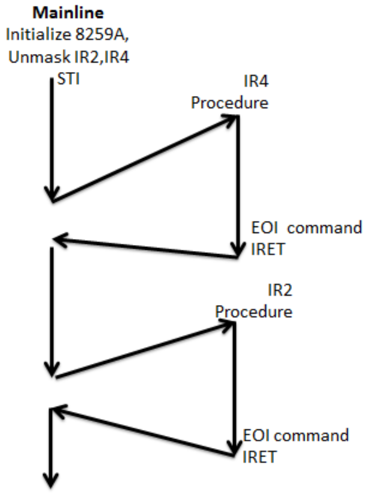
8259 and 8086 program flow

If IR4 is follwed by IR2 and INTR enabled in IR4 procedure

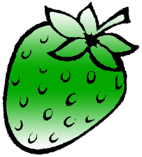


8259 and 8086 program flow

If IR4 is follwed by IR2 and INTR not enabled in IR4 procedure



STRAWBERRY



/strawberrydevelopers

/strawberry_app

For more visit:

Strawberrydevelopers.weebly.com