CMPE 310
Systems Design and Programming

L14: Chapter 12 – Interrupts

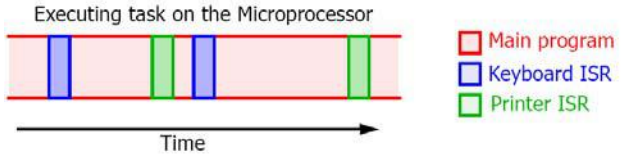
UMBC
AN HONORS UNIVERSITY IN MARYLAND

L14 Objectives

* Explain how the x86 PC executes interrupts by using the interrupt vector table and interrupt routines
* Describe the difference between hardware & software interrupts

Interrupts

* Interrupt processing is an alternative to polling.



* The Intel microprocessors support hardware interrupts through:
  * Two pins that allow interrupt requests (INTR and NMI)
  * One pin ($\overline{\text{INTA}}$), that acknowledges the interrupt requested through INTR.
* And software interrupts through instructions:
  * INT, INTO, INT 3 and BOUND
* Control is provided through
  * IF and TF flag bits
  * IRET (or IRETD in the 80386, 80486, Pentium – Pentium 4)

Interrupt Vector Table

080H	32-255 User defined	
	14-31 Reserved	
040H	Coprocessor error	16
03CH	Unassigned	15
038H	Page fault	14
034H	General protection	13
030H	Stack seg overrun	12
02CH	Segment not present	11
028H	Invalid task state seg	10
024H	Coproc seg overrun	9
020H	Double fault	8
01CH	Coprocessor not avail	7
018H	Undefined Opcode	6
014H	Bound	5
010H	Overflow (INTO)	4
00CH	1-byte breakpoint	3
008H	NMI pin	2
004H	Single-step	1
000H	Divide error	0

The interrupt vector table is located in the first 1024 bytes of memory at addresses 000000H through 0003FFH.

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

Interrupt Vector Table

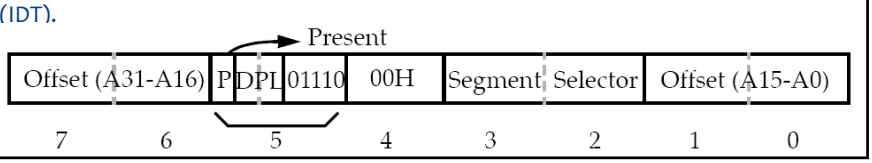
- INT (single step) and INT3 (breakpoint) behave in a similar way.
  - INT n:
    - Microprocessor saves the FR & CS:IP
    - Calls ISR located at a fixed memory location (n *4).
    - The INT instruction is a two bytes instruction, opcode plus n.
- BOUND and INTO are both conditional.
  - BOUND:
    - **BOUND *AX, DATA*** ***;Compares AX with DATA***
    - AX is compared with DATA and DATA+1, if less than an interrupt occurs.
    - AX is compared with DATA+2 and DATA+3, if greater than an interrupt occurs.
  - INTO:
    - Checks the overflow flag (OF). If OF=1, the ISR is called.
- IRET removes 6 bytes from the stack, 2 for IP, 2 for CS and 2 for FLAGS.

Real Mode Interrupts

- After the execution of each instruction, the microprocessor determines whether an interrupt is active by checking, in order:
  1. Other instruction executions
  2. Single-step
  3. NMI
  4. Coprocessor segment overrun
  5. INTR
  6. INT instructions
- If one or more of these conditions are present, then:
  1. FLAGS is pushed onto the stack
  2. Both the interrupt (IF) and trap (TF) flags are cleared, which disables the INTR pin and the trap or single-step feature.
  3. The CS and IP are pushed onto the stack.
  4. The interrupt vector contents are fetched and loaded into CS and IP and execution resumes in the ISR.
  5. On IRET, CS, IP and FLAGS are popped.
     - IF and TF are set to the state prior to the interrupt.

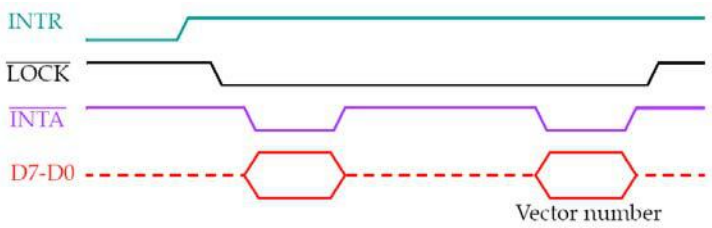
Real and Protected Mode Interrupts

- The return address (CS:IP) is pushed onto the stack during the interrupt.
- The return address can point to:
  - The next instruction.
  - The offending (current) instruction.
    - The latter case occurs for interrupts 0, 5, 6, 7, 8, 10, 11, 12 and 13.
      - This makes it possible to try the instruction again in case of error.
- Protected Mode:
  - The same interrupt assignments are made and the same sequence of operations occurs in protected mode but the interrupt table is different.
  - Instead, uses 256 interrupt descriptors stored in the interrupt descriptor table (IDT).



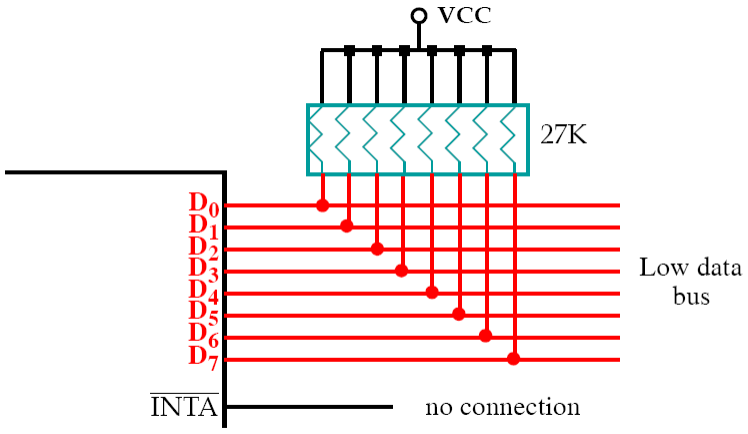
Hardware Interrupts

- The INTR pin must be externally decoded to select a vector.
  - Any vector is possible, but the interrupt vectors between 20H and FFH are usually used (Intel reserves vectors between 00H and 1FH).
- $\overline{\text{INTA}}$ is an output of the microprocessor to signal the external decoder to place the interrupt number on data bus connections D7-D0.
- The INTR pin is set by an external device (8259A) and cleared in the ISR.
  - The input is automatically disabled by the microprocessor once it is recognized and re-enabled by IRET or IRETD instruction.
  - Timing diagram of the handshake.

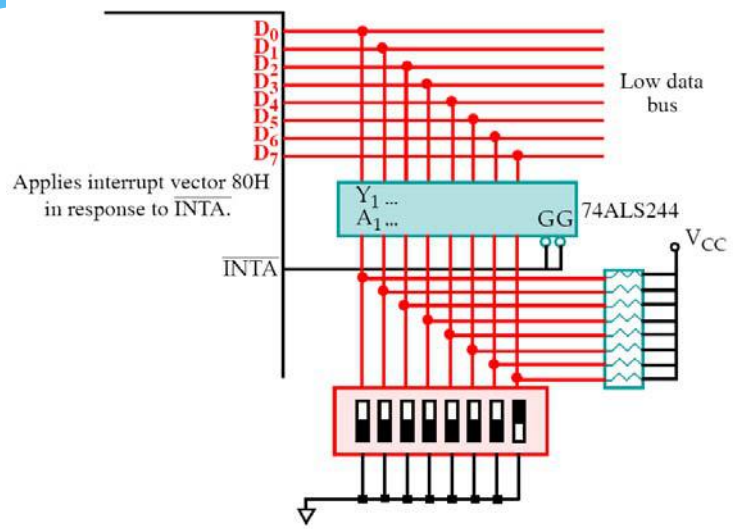


Hardware Interrupts

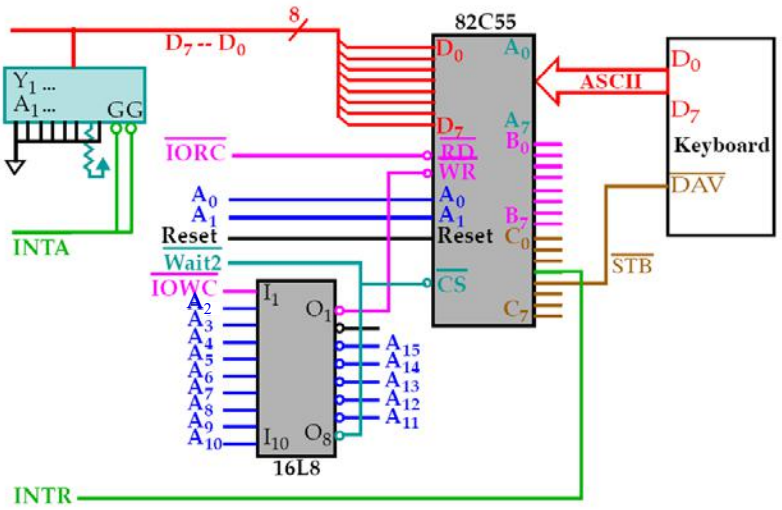
* Simplest method of generating an interrupt vector:
* Always generates interrupt vector FFH in response to INTR.



Tri-state Buffer for Generating the Interrupt Vector



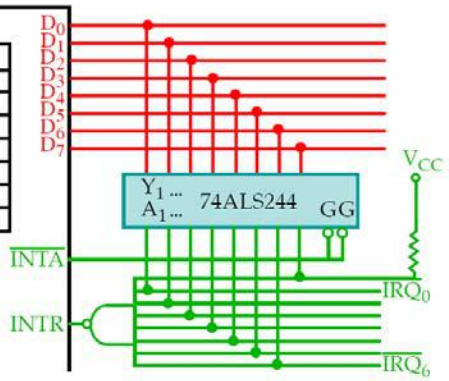
An Example 82C55 Keyboard Interrupt Configuration



Handling more than 1 IRQ

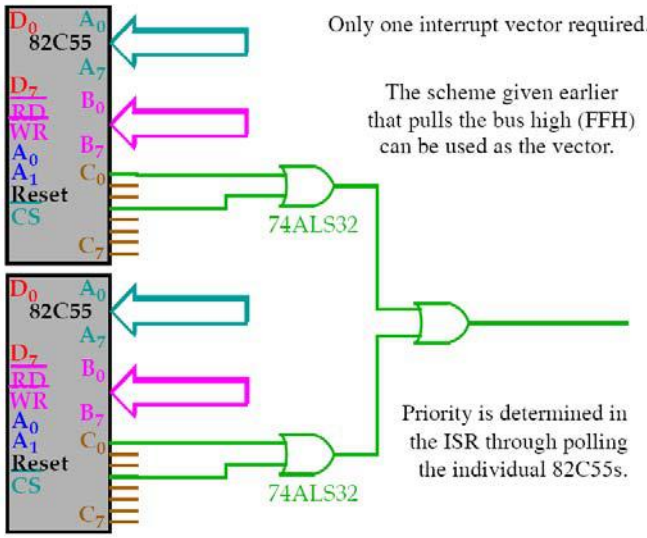
IRQs

6	5	4	3	2	1	0	Vect
1	1	1	1	1	1	0	FEH
1	1	1	1	1	0	1	FDH
1	1	1	1	0	1	1	FBH
1	1	1	0	1	1	1	F7H
1	1	0	1	1	1	1	EFH
1	0	1	1	1	1	1	DFH
0	1	1	1	1	1	1	BFH



- If any of $\overline{IRQx}$ goes low, the output of the NAND goes high requesting an interrupt through INTR pin.
- Note that if more than one IRQ goes low, a unique interrupt vector is generated and an interrupt priority needs to be defined.
  - The entire top half of the vector table & its 128 interrupt vectors must be expanded to accommodate all possible conditions of these seven IRQs

Daisy-Chained Mechanism for Multiple IRQs



Only one interrupt vector required.

The scheme given earlier that pulls the bus high (FFH) can be used as the vector.

Priority is determined in the ISR through polling the individual 82C55s.

Next time

* Programmable Interrupt Controller

STOP