Chapter 11
Interrupt Controller

Topics
- Interrupt vector table
- Interrupt service routine
- Categories of interrupts
  - Hardware interrupts
  - Software interrupts
- 8259 Interfacing
- 8259 programming

8088/8086 Interrupts

- An interrupt is an external event which informs the CPU that a device needs its service.
- There are 256 interrupts (types): INT 00, INT 01, ..., INT FF in the 8088/8086.
- When executes an interrupt, microprocessor automatically saves the flag register, the instruction pointer, and the code segment register on the stack, and goes to a fixed memory location.

Interrupt Vector Table

<table>
<thead>
<tr>
<th>INT number</th>
<th>Physical address</th>
<th>Logical address</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT 00</td>
<td>00000</td>
<td>0000:0000</td>
</tr>
<tr>
<td>INT 01</td>
<td>00004</td>
<td>0000:0004</td>
</tr>
<tr>
<td>INT 02</td>
<td>00008</td>
<td>0000:0008</td>
</tr>
<tr>
<td>INT 03</td>
<td>0000C</td>
<td>0000:000C</td>
</tr>
<tr>
<td>INT 04</td>
<td>00010</td>
<td>0000:0010</td>
</tr>
<tr>
<td>INT 05</td>
<td>00014</td>
<td>0000:0014</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>INT FF</td>
<td>003FC</td>
<td>0000:03FC</td>
</tr>
</tbody>
</table>

- Interrupt goes to the memory location that is four times the value of the interrupt number.
- For example, INT 03 will go to address \(4 \times 3 = 12 = 0000_{16}\).
Interrupt Service Routine (ISR)

- There is a program associated with every interrupt.
- When an interrupt is invoked, a CPU runs a program for a service → an interrupt service routine (ISR).
- The address of the interrupt service routine is shown in the interrupt vector table.
- Four bytes of memory are allocated for every interrupt.
- The memory space of 1024 bytes (256x4=1024) are set aside for the interrupt vector table.

Intel’s List of Designated Interrupts for the 8088/8086

<table>
<thead>
<tr>
<th>CS</th>
<th>IP</th>
<th>INT FF</th>
</tr>
</thead>
<tbody>
<tr>
<td>0003FC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>00010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0000C</td>
<td></td>
<td>INT 04 Signed number overflow</td>
</tr>
<tr>
<td>00008</td>
<td></td>
<td>INT 03 Breakpoint</td>
</tr>
<tr>
<td>00004</td>
<td></td>
<td>INT 02 NMI</td>
</tr>
<tr>
<td>00000</td>
<td></td>
<td>INT 01 Single step</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INT 00 Divide error</td>
</tr>
</tbody>
</table>

CALL vs INT

<table>
<thead>
<tr>
<th>CALL FAR</th>
<th>INT nn</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jump to any location within the 1 MByte address range.</td>
<td>Go to a fixed memory location in the interrupt vector table.</td>
</tr>
<tr>
<td>Locate in the sequence of instructions in the program.</td>
<td>Externally activated hardware interrupt can come at any time.</td>
</tr>
<tr>
<td>Cannot be masked (disabled).</td>
<td>Can be masked.</td>
</tr>
<tr>
<td>Save CS:IP of the next instruction on the stack.</td>
<td>Save CS:IP of the next instruction and flag register on the stack.</td>
</tr>
<tr>
<td>Use RETF in the last instruction of its routine. RETF pops CS, IP off the stack.</td>
<td>Use IRET in the last instruction of its routine. IRET pops CS, IP, and the flag register.</td>
</tr>
</tbody>
</table>

Hardware Interrupts

- Three pins, in the 80x86, are associated with hardware interrupts:
  - INTR (interrupt request):
    - an input signal into the CPU which can be masked (ignored) and unmasked through CLI (disabled) and STI (enabled).
    - No specific location in the vector table assigned for INTR.
    - It uses any INT nn: space that has not been assigned.
    - 8259 chip connects to INTR to expand # of hardware interrupts up to 64.
  - NMI (nonmaskable interrupt): INT 02
    - an input signal into the CPU that cannot be masked and unmasked using software instructions.
    - CPU will go to memory location 0008 to get the address (CS:IP) of the interrupt service routine associated with NMI.
  - INTA (interrupt acknowledge)
    - An output of the microprocessor to signal the external circuitry that the interrupt request has been acknowledged and to prepare to put its interrupt type number on the data bus.
**Software Interrupts**

- INT nn is invoked software (sequence of code)
- **Examples**
  - DOS INT 21H, BIOS INT 10H.
  - INT 00 (divide error)
  - INT 01 (single step)
  - INT 03 (breakpoint)
  - INT 04 (signed number overflow)

**Processing Interrupts**

1. The flag register (2-byte register) is pushed onto the stack and SP is decremented by two.
2. IF (interrupt enable flag) and TF (trap flag) are cleared → mask (ignore) other interrupt requests from the INTR and disable single-step.
3. The current CS:IP are pushed onto the stack → SP is decremented by 4.
4. The INT number is multiplied by 4 to get the physical address of the location of CS:IP of the interrupt service routine.
5. CPU starts to fetch and execute of the interrupt service routine from the new CS:IP.
6. CPU executes the last instruction (IRET) and get IP, CS, and flag register back from the stack.
7. CPU continues executing the code where it left off.

**8259 Programmable Interrupt Controller**

80x86 can use INTR and INTA pins to expand the number of interrupts.

**Block Diagram of the 8259A**

Block diagram showing the pin connections and interrupt inputs.
Internal Architecture of the 8259A

Initialization Command Word (ICW1, 2)

Note: Buffered mode is used in systems which require buffering on the data bus. The 8259’s G7/EN pin goes low when its data bus output is enabled; therefore, G7/EN can be used to enable a transceiver/buffer for the data bus output.

Operation Command Word (OCW)
Source of Hardware Interrupts

Sources of NMI