Assembly Language for Intel-Based Computers, 5th Edition
Kip Irvine

Chapter 2: IA-32 Processor Architecture

Slides prepared by the author
Revision date: June 4, 2006

(c) Pearson Education, 2006-2007. All rights reserved. You may modify and copy this slide show for your personal use, or for use in the classroom, as long as this copyright statement, the author's name, and the title are not changed.
Chapter Overview

• Basic Microcomputer Design
• Instruction Execution Cycle
• Addressable Memory
• Registers
Basic Microcomputer Design

- clock synchronizes CPU operations
- control unit (CU) coordinates sequence of execution steps
- ALU performs arithmetic and bitwise processing
Instruction Execution Cycle

- Fetch
- Decode
- Fetch operands
- Execute
- Store output
Addressable Memory

- Protected mode
  - 4 GB
  - 32-bit address
  - MS Windows, Linux
- Real-address and Virtual-8086 modes
  - 1 MB space
  - 20-bit address
  - MS-DOS
Named storage locations inside the CPU, optimized for speed.

### 32-bit General-Purpose Registers

<table>
<thead>
<tr>
<th>EAX</th>
<th>EBP</th>
</tr>
</thead>
<tbody>
<tr>
<td>EBX</td>
<td>ESP</td>
</tr>
<tr>
<td>ECX</td>
<td>ESI</td>
</tr>
<tr>
<td>EDX</td>
<td>EDI</td>
</tr>
</tbody>
</table>

### 16-bit Segment Registers

<table>
<thead>
<tr>
<th>EFLAGS</th>
<th>CS</th>
<th>ES</th>
</tr>
</thead>
<tbody>
<tr>
<td>EIP</td>
<td>SS</td>
<td>FS</td>
</tr>
<tr>
<td></td>
<td>DS</td>
<td>GS</td>
</tr>
</tbody>
</table>
Accessing Parts of Registers

- Use 8-bit name, 16-bit name, or 32-bit name
- Applies to EAX, EBX, ECX, and EDX

<table>
<thead>
<tr>
<th>32-bit</th>
<th>16-bit</th>
<th>8-bit (high)</th>
<th>8-bit (low)</th>
</tr>
</thead>
<tbody>
<tr>
<td>EAX</td>
<td>AX</td>
<td>AH</td>
<td>AL</td>
</tr>
<tr>
<td>EBX</td>
<td>BX</td>
<td>BH</td>
<td>BL</td>
</tr>
<tr>
<td>ECX</td>
<td>CX</td>
<td>CH</td>
<td>CL</td>
</tr>
<tr>
<td>EDX</td>
<td>DX</td>
<td>DH</td>
<td>DL</td>
</tr>
</tbody>
</table>
Index and Base Registers

- Some registers have only a 16-bit name for their lower half:

<table>
<thead>
<tr>
<th>32-bit</th>
<th>16-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESI</td>
<td>SI</td>
</tr>
<tr>
<td>EDI</td>
<td>DI</td>
</tr>
<tr>
<td>EBP</td>
<td>BP</td>
</tr>
<tr>
<td>ESP</td>
<td>SP</td>
</tr>
</tbody>
</table>
Some Specialized Register Uses (1 of 2)

- General-Purpose
  - EAX – accumulator
  - ECX – loop counter
  - ESP – stack pointer
  - ESI, EDI – index registers (source, destination)
  - EBP – extended frame pointer (stack)

- Segment
  - CS – code segment
  - DS – data segment
  - SS – stack segment
  - ES, FS, GS - additional segments
Some Specialized Register Uses (2 of 2)

- **EIP** – instruction pointer
- **EFLAGS**
  - status and control flags
  - each flag is a single binary bit
Status Flags

- **Carry**
  - unsigned arithmetic out of range
- **Overflow**
  - signed arithmetic out of range
- **Sign**
  - result is negative
- **Zero**
  - result is zero
- **Auxiliary Carry**
  - carry from bit 3 to bit 4
- **Parity**
  - sum of 1 bits is an even number
Floating-Point, MMX, XMM Registers

- Eight 80-bit floating-point data registers
  - ST(0), ST(1), . . . , ST(7)
  - used for all floating-point arithmetic

- Eight 64-bit MMX registers
- Eight 128-bit XMM registers for single-instruction multiple-data (SIMD) operations