Introduction to IA-32
IA-32 Processors

- Evolutionary design
  - Starting in 1978 with 8086
  - Added more features as time goes on
  - Still support old features, although obsolete
  - Totally dominate computer market

- Complex Instruction Set Computer (CISC)
  - Many different instructions with many different formats
  - Hard to match performance of Reduced Instruction Set Computers (RISC)
  - But, Intel has done just that!
# IA-32 History

- **Evolution with backward compatibility**

<table>
<thead>
<tr>
<th>Year</th>
<th>Processor</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1978</td>
<td>8086</td>
<td>x86 is born</td>
</tr>
<tr>
<td>1980</td>
<td>8087</td>
<td>x87 is born</td>
</tr>
<tr>
<td>1985</td>
<td>80386</td>
<td>“IA-32”</td>
</tr>
<tr>
<td>1995</td>
<td>Pentium Pro</td>
<td>PAE</td>
</tr>
<tr>
<td>1997</td>
<td>Pentium MMX</td>
<td>MMX</td>
</tr>
<tr>
<td>1999</td>
<td>Pentium III</td>
<td>SSE</td>
</tr>
<tr>
<td>2000</td>
<td>Pentium 4</td>
<td>SSE2</td>
</tr>
<tr>
<td>2004</td>
<td>Pentium 4 Prescott</td>
<td>SSE3, Intel 64</td>
</tr>
<tr>
<td>2005</td>
<td>Pentium 4 662</td>
<td>Intel VT</td>
</tr>
<tr>
<td>2006</td>
<td>Core 2</td>
<td>SSSE3</td>
</tr>
<tr>
<td>2008</td>
<td>Core 2 Penryn</td>
<td>SSE4.1</td>
</tr>
<tr>
<td>2008</td>
<td>Core i7</td>
<td>SSE4.2</td>
</tr>
</tbody>
</table>
## Basic Execution Environment

### Application Programming Registers

<table>
<thead>
<tr>
<th>General-purpose registers</th>
<th>Segment registers</th>
<th>Control registers</th>
</tr>
</thead>
<tbody>
<tr>
<td>EAX 31</td>
<td>CS 15</td>
<td>CR0 31</td>
</tr>
<tr>
<td>EBX 30</td>
<td>DS 14</td>
<td>CR1 30</td>
</tr>
<tr>
<td>ECX 29</td>
<td>SS 13</td>
<td>CR2 29</td>
</tr>
<tr>
<td>EDX 28</td>
<td>ES 12</td>
<td>CR3 28</td>
</tr>
<tr>
<td>EBP 27</td>
<td>EDX 11</td>
<td>CR4 27</td>
</tr>
<tr>
<td>ESI 26</td>
<td>EBP 10</td>
<td></td>
</tr>
<tr>
<td>EDI 25</td>
<td>ESI 9</td>
<td></td>
</tr>
<tr>
<td>ESP 24</td>
<td>EDI 8</td>
<td></td>
</tr>
<tr>
<td>EIP 31</td>
<td>SI 7</td>
<td></td>
</tr>
<tr>
<td>Eflags 30</td>
<td>DI 6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SP 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### System Table Registers

- GDTR
  - linear base address
  - table limit
- IDTR
  - linear base address
  - table limit

### System Segment Registers

- TR
  - seg. selector
- LDTR
  - seg. selector
General-Purpose Registers

- **EAX, EBX, ECX, EDX, ESI, EDI, ESP, EBP**
  - Some instructions assume that pointers in certain registers are relative to specific segments.
  - Many instructions assign specific registers to hold operands

<table>
<thead>
<tr>
<th>Register</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EAX</td>
<td>Accumulator for operands and results data</td>
</tr>
<tr>
<td>EBX</td>
<td>Pointer to data in the DS segment</td>
</tr>
<tr>
<td>ECX</td>
<td>Counter for string and loop operations</td>
</tr>
<tr>
<td>EDX</td>
<td>I/O pointer</td>
</tr>
<tr>
<td>ESI</td>
<td>Pointer to data in the segment pointed to by the DS register; Source pointer for string operations</td>
</tr>
<tr>
<td>EDI</td>
<td>Pointer to data in the segment pointed to by the ES register; Destination pointer for string operations</td>
</tr>
<tr>
<td>ESP</td>
<td>Stack pointer (in the SS segment)</td>
</tr>
<tr>
<td>EBP</td>
<td>Pointer to data on the stack (in the SS segment)</td>
</tr>
</tbody>
</table>
EFLAGS Register (1)

- ID Flag (ID)
- Virtual Interrupt Pending (VIP)
- Virtual Interrupt Flag (VIF)
- Alignment Check (AC)
- Virtual-8086 Mode (VM)
- Resume Flag (RF)
- Nested Task (NT)
- I/O Privilege Level (IOPL)
- Overflow Flag (OF)
- Direction Flag (DF)
- Interrupt Enable Flag (IF)
- Trap Flag (TF)
- Sign Flag (SF)
- Zero Flag (ZF)
- Auxiliary Carry Flag (AF)
- Parity Flag (PF)
- Carry Flag (CF)

S Indicates a Status Flag
C Indicates a Control Flag
X Indicates a System Flag

Reserved bit positions. DO NOT USE.
Always set to values previously read.
Status flags

**CF (Carry):** set if an arithmetic operation generates a carry or a borrow; indicates an overflow condition for unsigned-integer arithmetic.

**PF (Parity):** set if the least-significant byte of the result contains an even number of 1 bits

**AF (Adjust):** set if an arithmetic operation generates a carry or a borrow out of bit 3 of the result; used in binary-coded decimal (BCD) arithmetic

**ZF (Zero):** set if the result is zero

**SF (Sign):** set equal to the most-significant bit of the result

**OF (Overflow):** set if the integer result is too large a positive number or too small a negative number to fit in the destination operand; indicates an overflow condition for signed-integer arithmetic.

**DF (Direction):** setting the DF causes the string instructions to auto-decrement; set and cleared by STD/CLD instructions
Instruction Pointer

- **EIP Register**
  - Contains the offset in the current code segment for the next instruction to be executed.
    - Advanced from one instruction boundary to the next in straightline code, or
    - Moved ahead or backwards by instructions such as JMP, Jcc, CALL, RET, and IRET.
  - Cannot be accessed directly by software
    - EIP is controlled implicitly by control transfer instructions, interrupts, and exceptions
  - Because of instruction prefetching, an instruction address read from the bus does not match the value in the EIP register.
Assembly Characteristics (1)

- **Minimal data types**
  - "Integer" data of 1, 2, 4, or 8 bytes
    - Data values
    - Addresses (untyped pointers)
  - Floating point data of 4, 8, or 10 bytes
  - No aggregate types such as arrays or structures
    - Just contiguously allocated bytes in memory
  - (cf.) In IA-32, a "word" means 16-bit data
Assembly Characteristics (2)

- **Primitive operations**
  - Perform an arithmetic/logical function on register or memory data
  - Transfer data between memory and register
    - Load data from memory into register
    - Store register data into memory
  - Transfer control
    - Unconditional jumps
    - Conditional branches
    - Procedure calls and returns
IA-32 Reference

- Intel 64 and IA-32 Architectures Software Developer’s Manual
  - Volume 1: Basic Architecture
  - Volume 2A, 2B: Instruction Set Reference
  - Volume 3A, 3B: System Programming Guide

- Available online: