**COSC2406 Assignment 2**

*Michael Buffone, September 22, 2019*

1. The five steps in the instruction execution cycle along with their descriptions are:
   1. Fetch the instruction [REQUIRED]
      1. The CPU fetches the instruction from an area of memory called the instruction queue. It then increments the instruction pointer.
   2. Decode [REQUIRED]
      1. The CPU decodes the instruction by looking at its binary bit pattern, which can possibly contain operands within the instruction.
   3. Fetch the operands
      1. If there are operands within the instruction, the CPU fetches them from registers and memory which can require address calculations.
   4. Execute [REQUIRED]
      1. The CPU executes the instruction utilizing any operands it fetched.
      2. It also updates the status flags like “Zero”, “Carry”, and “Overflow”.
   5. Store the result
      1. If an output operand was part of the instruction, the CPU stores the result of its execution in the operand.
2. When a processor switches from one task to another, the memory location of the tasks must be preserved
3. The x86 processor has four modes of operation which are:
   1. Protected mode
      1. The native state of the processor where all instructions and features are available to be used.
      2. Programs are given segments, and the processor prevents programs from referencing memory outside their assigned segments.
   2. Real-address mode
      1. Has the programming environment of an early Intel processor with a few added features (ex. The ability to switch into other modes).
      2. This mode is useful if a program requires direct access to system memory and hardware devices.
   3. System management mode
      1. Provides an operating system with a mechanism for implementing functions like power management and system security.
      2. Usually implemented by computer manufacturers who customize the processor for a specific system setup.
   4. Virtual-8086 mode (Sub mode)
      1. An operating system can execute multiple virtual-8086 sessions at the same time, with the intent of protecting other running programs if a program crashes or attempts to write data into the system memory area.
4. When running in 64-bit programming mode, 16-bit real mode or virtual-8086 mode aren’t supported.
5. The 64-bit x86-64 processor can support 256 terabytes of RAM.
6. 32-bit general purpose registers:

|  |  |
| --- | --- |
| **EAX** (Extended accumulator register)  -Used by multiplication and division instructions | **EBP** (Extended frame pointer)  -Used by high-level languages to reference function parameters and local variables on the stack |
| **EBX** | **ESP** (Extended stack pointer)  -Addresses data on the stack, and is rarely used for arithmetic or data transfer |
| **ECX**  -Used by the CPU for loop counting | **ESI** (Extended source index)  -Used by high-speed memory transfer instructions |
| **EDX** | **EDI** (Extended destination index)  -Used by high-speed memory transfer instructions |

1. The **carry flag** is set when the result of an unsigned arithmetic operation is too large to fit into the destination.
2. The **overflow flag** is set when the result of a signed arithmetic operation is either too large or too small to fit into the destination.
3. Four additional CPU status flags besides the carry and overflow flag are:
   1. Sign flag
      1. Set when the result of an arithmetic or logical operation generates a negative result.
   2. Zero flag
      1. Set when the result of an arithmetic or logical operation generates a result of zero.
   3. Auxiliary carry flag
      1. Set when an arithmetic operation causes a carry from bit 3 to bit 4 in an 8-bit operand.
   4. Parity flag
      1. Set if the least-significant byte in the result contains an even number of 1 bits. (used for error checking when data may be altered or corrupted)