CS430 Computer Architecture

Spring 2015

Chapter 3
A Top-Level View of Computer Function and Interconnection

- Skipping Section 2.6 Performance Assessment until needed
- Computer Components & Function (Section 3.1 & 3.2) on pp.66-84
- Good problems to work: 3.1, 3.2, 3.3

Computer Components
Top-level View

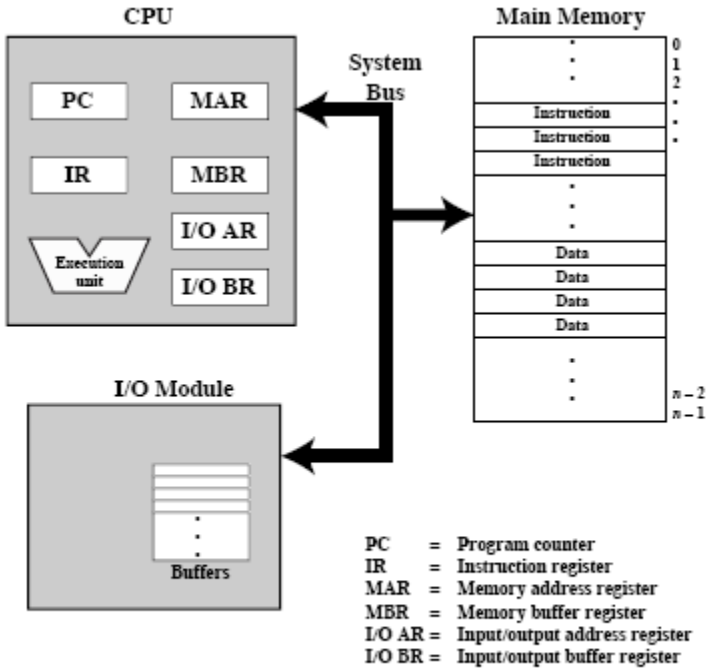


Figure 3.2 Computer Components: Top-Level View

Basic Computer Function

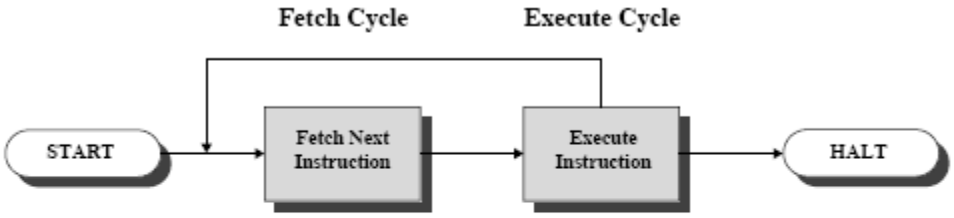


Figure 3.3 Basic Instruction Cycle

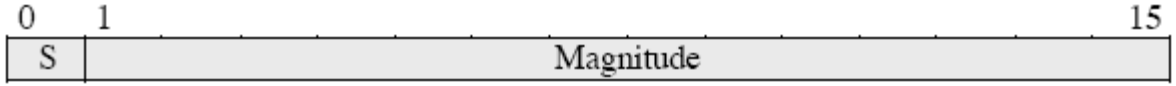
Problem

- Explain an instruction fetch using the components of Figure 3.2

Hypothetical Computer

- Consider a hypothetical computer with a single accumulator (register) where both instructions and data are 16-bits long.

(a) Instruction format



(b) Integer format

Program Counter (PC) = Address of instruction
Instruction Register (IR) = Instruction being executed
Accumulator (AC) = Temporary storage

(c) Internal CPU registers

0001 = Load AC from Memory
0010 = Store AC to Memory
0101 = Add to AC from Memory

(d) Partial list of opcodes

Figure 3.4 Characteristics of a Hypothetical Machine

Problem

1. How many possible instructions can this hypothetical computer have? Why?

2. Give the hexadecimal representation of the number 15 and -15.

Program Trace

- Program fragment adds contents at address 940 to contents at address 941 with result at 941
- 0001=Load AC from MEM
- 0010=Store AC to MEM
- 0101=Add to AC from MEM

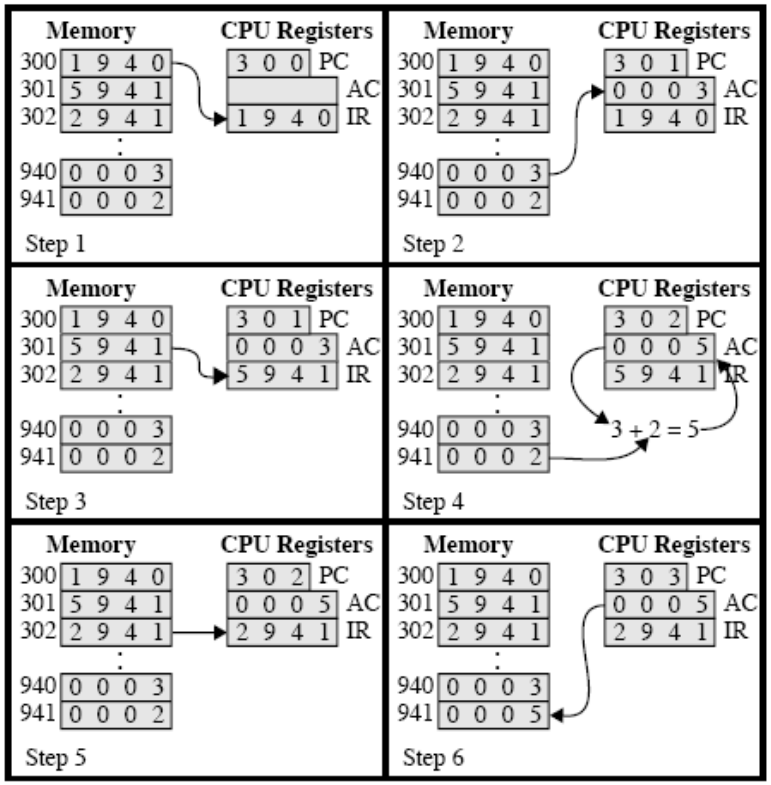


Figure 3.5 Example of Program Execution (contents of memory and registers in hexadecimal)

Instruction Cycle

- Let's break down the instruction cycle as follows:

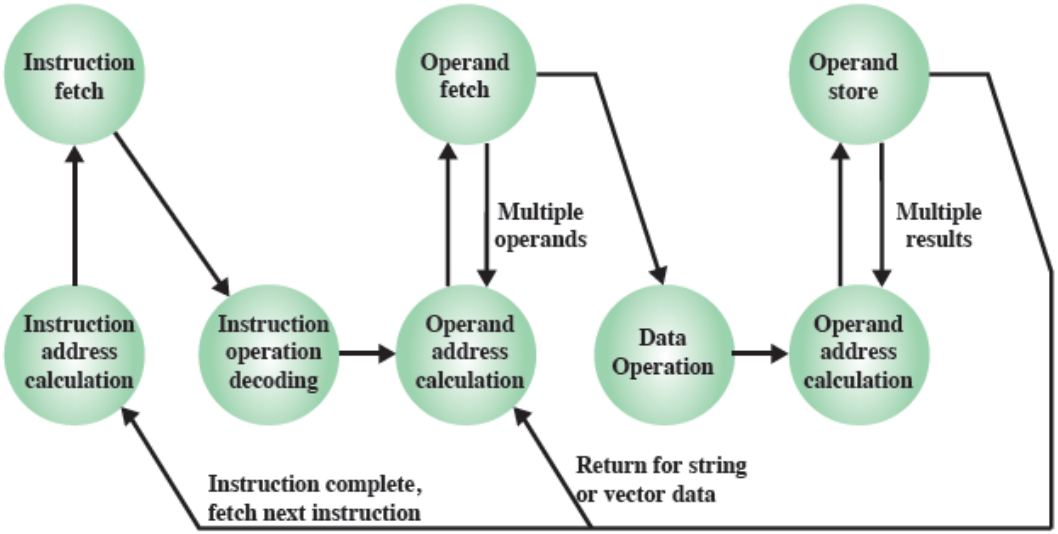


Figure 3.6 Instruction Cycle State Diagram

Interrupt

- Interrupt - is an external request for service. In particular, an interrupt causes the microprocessor to stop executing the current procedure (saving the status) and continue on with the routine specified by the interrupt. When the interrupt has been fully serviced, control returns to the previously executing routine.
- Two types of interrupts exist:
  1. maskable - depending on the status of the interrupt flag, this interrupt can be ignored by the hardware.
  2. nonmaskable - must be acknowledged by the hardware independent of the interrupt flag.

Classes of Interrupts

- Program - generated by some instruction execution
  1. division by zero
  2. attempt to execute illegal opcode
  3. reference outside a user's memory space
- Timer - generated by a timer within the processor for OS
- I/O - generated by an I/O controller to signal normal completion
- Hardware Failure - generated by a power failure or memory parity error

Modified Instruction Cycle

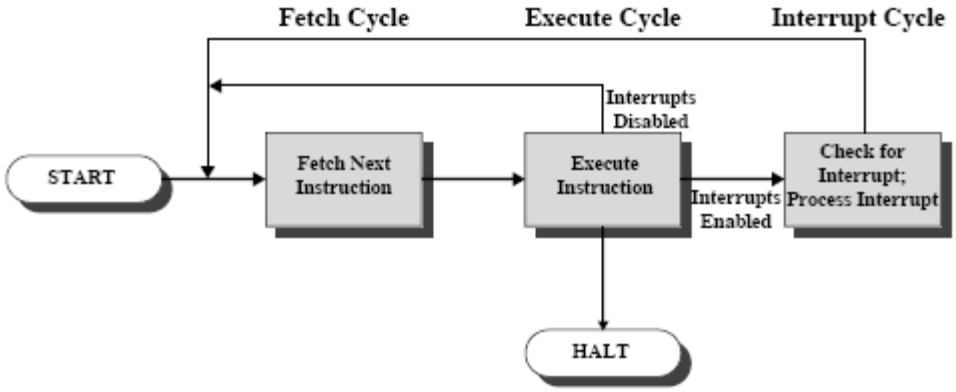


Figure 3.9 Instruction Cycle with Interrupts

Flow of Control

- In figure 3.8, the instruction completes at location i and then any pending interrupts are checked for by the processor. If an interrupt is found, as in the case of figure 3.8, the next instruction at i+1 is saved and execution continues at the interrupt handler. Once the interrupt handler has completed, control returns to the instruction at location i+1.

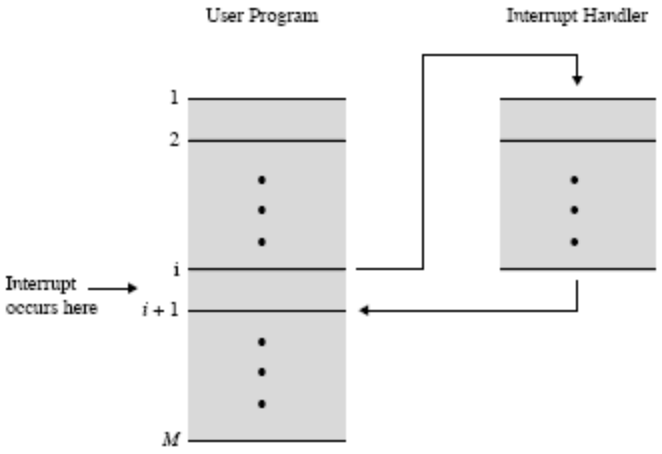


Figure 3.8 Transfer of Control via Interrupts

Flow of Control

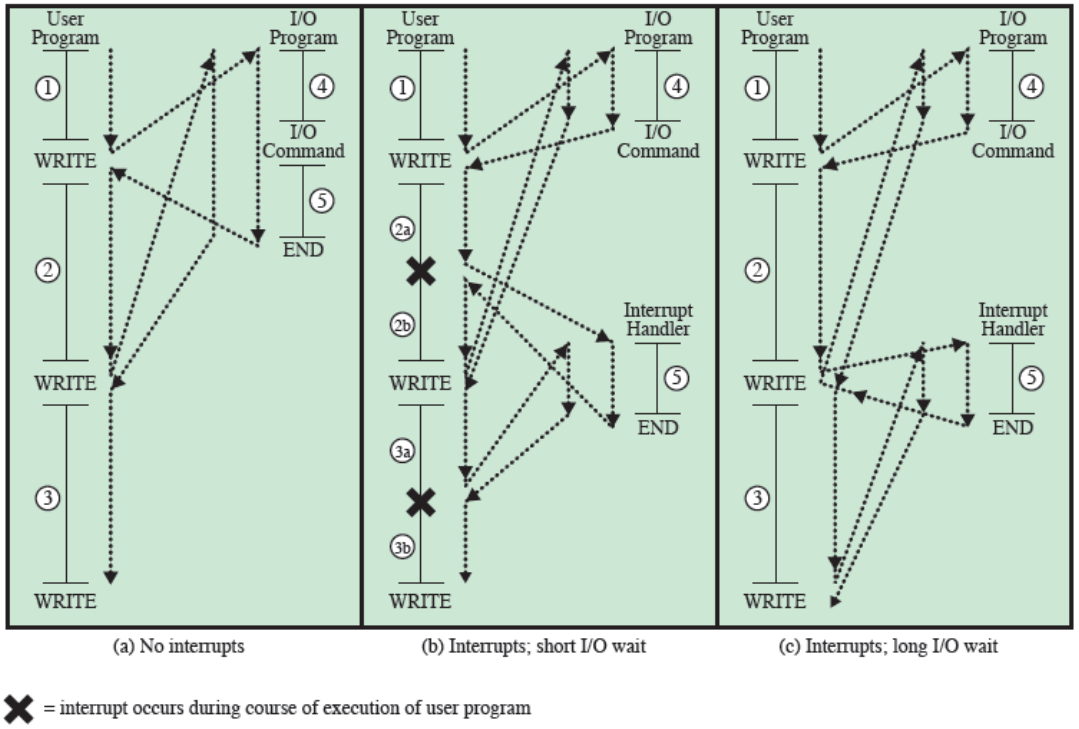


Figure 3.7 Program Flow of Control Without and With Interrupts

Time Savings using Interrupts

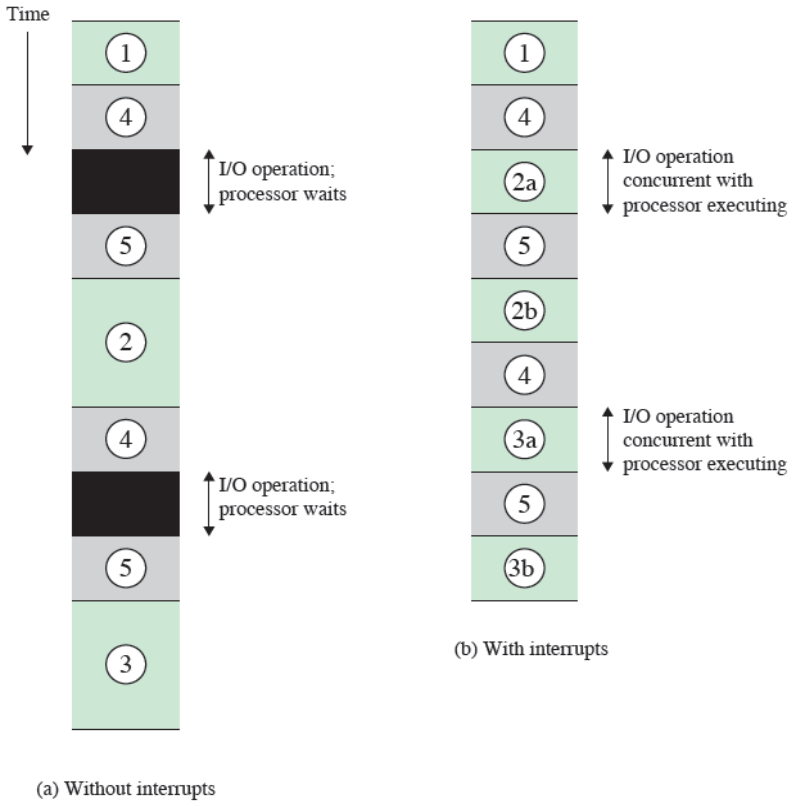


Figure 3.10 Program Timing: Short I/O Wait

Time Savings using Interrupts

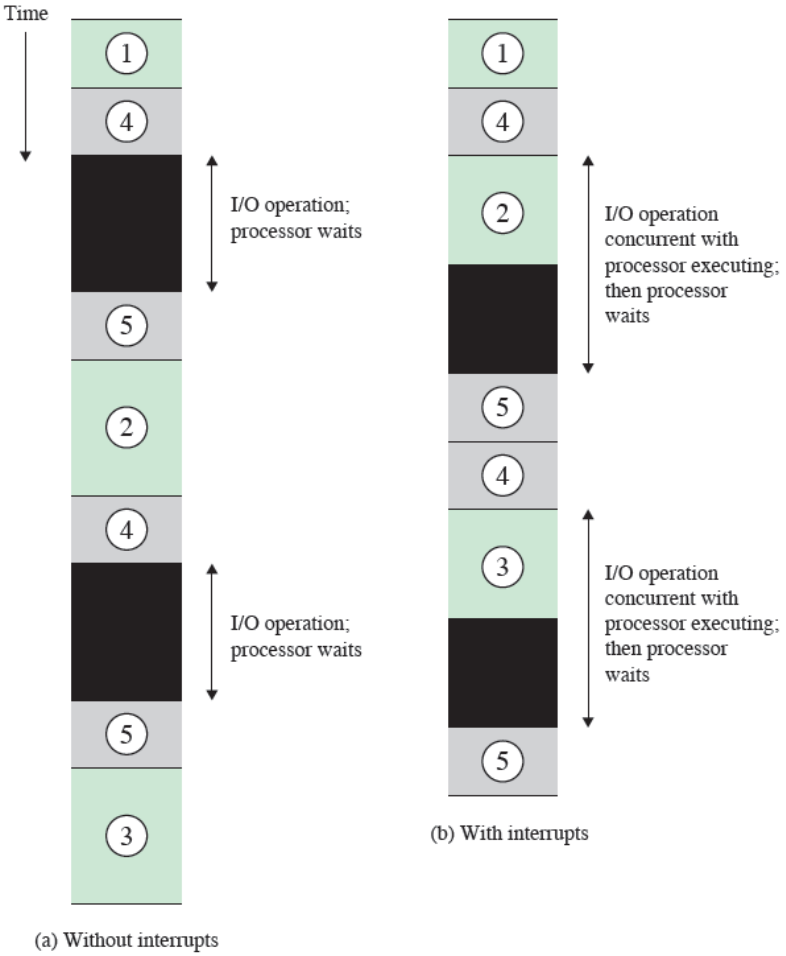


Figure 3.11 Program Timing: Long I/O Wait

Instruction Cycle with Interrupts

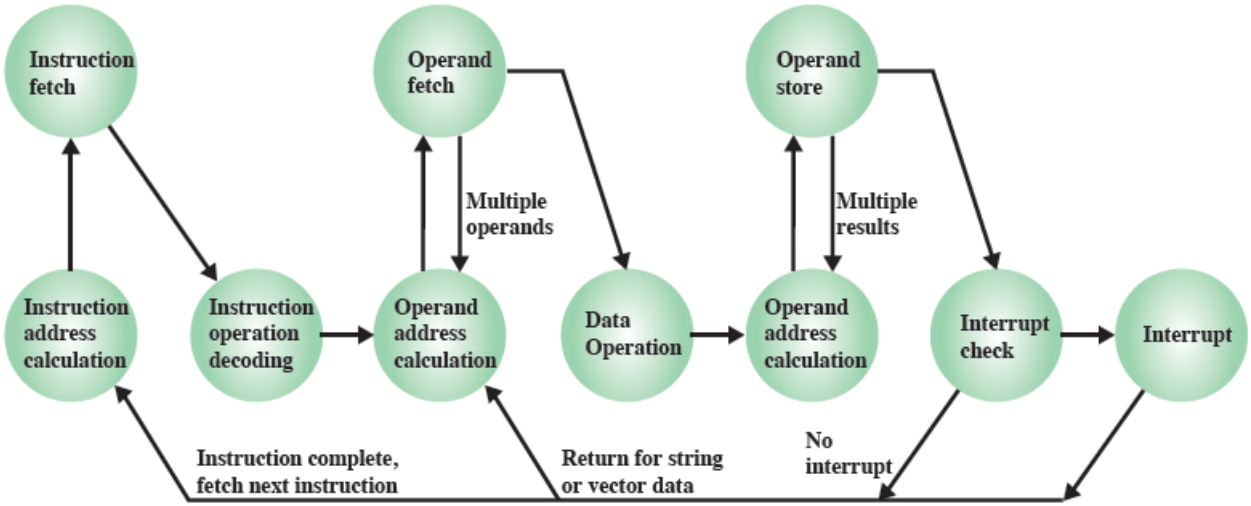


Figure 3.12 Instruction Cycle State Diagram, With Interrupts