CSCI 4717/5717
Computer Architecture

Topic: Single Processor Architecture
Reading: Stallings, Sections 3.1 through 3.3

Review of Three Key Concepts of von Neumann Architecture

- Data and instructions in single read-write memory
- Memory contents are addressable by location regardless of whether content is data or instruction
- Execution of code is sequential from one instruction to the next unless a jump is encountered

Program Concept

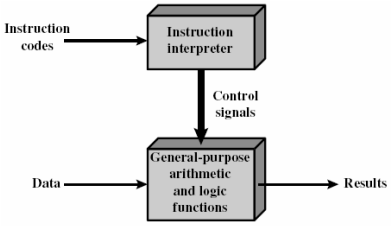
- Just about any function can be realized with hardwired logic components (calculator)
- Hardwired systems, however, are inflexible
- General purpose hardware can do different tasks, given correct control signals
- Instead of re-wiring, supply a new set of control signals

How Can We Create a Program?

- Each step activates a set of control signals to control general purpose logic
- Each step is an arithmetic or logical operation
- For each operation, a different set of control signals is needed
- Program equals the sequence of steps
- Programming is no longer a case of rewiring

Instruction Interpreter

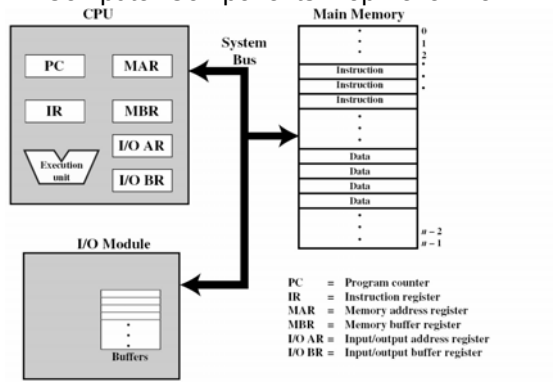
Now we need a device to accept instruction codes and turn them into control signals for the arithmetic and logic hardware.



Encoding Instructions

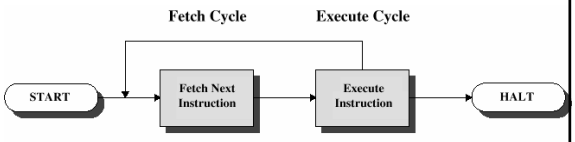
- Unique binary patterns identify operation to be performed.
- Examples:
  - Simple addition machine in Figure 3.4 on page 61 of textbook
  - X86 Encoding – http://webster.cs.ucr.edu/AoA/DOS/ch03/CH03-3.html#HEADING3-102

Computer Components: Top-Level View



Simplified 2-Step Instruction Cycle

- Instruction cycle is not the same thing as a clock cycle
- Two steps:
  - Fetch
  - Execute



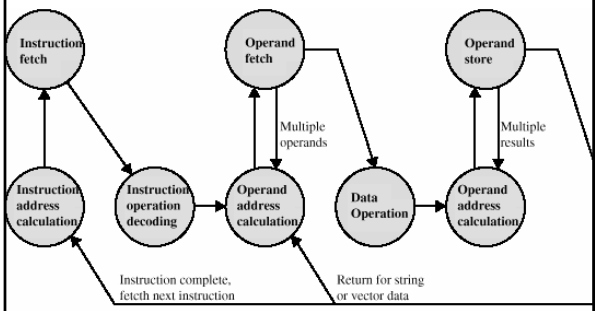
Fetch Cycle

- Program Counter (PC) holds address of next instruction to fetch
- Processor fetches instruction from memory location pointed to by PC
- Increment PC unless instructed otherwise
- Instruction loaded into Instruction Register (IR)
- Processor interprets instruction and performs required actions

Execute Cycle

- Processor-memory
  - data transfer between CPU and main memory
- Processor I/O
  - Data transfer between CPU and I/O module
- Data processing
  - Some arithmetic or logical operation on data
- Control
  - Alteration of sequence of operations, e.g. jump
- Combination of above

Instruction Cycle State Diagram



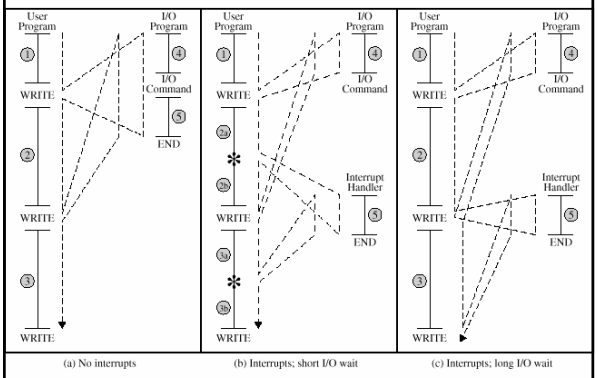
Interrupts

- No special code is needed in main code
- Interrupt Service Routines (ISR) handle condition
- Interrupts may be disabled; pending interrupts serviced as soon as interrupts are enabled again
  - Global enabling – affects all maskable interrupts
  - Local enabling – affects individual interrupts

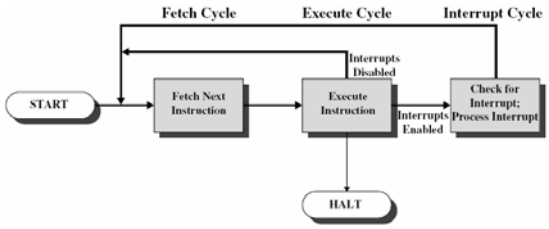
Types of Interrupts

- Program – Something that occurs as a result of program execution such as illegal instructions, arithmetic overflow, divide by zero, or memory handling error
- Timer – Generated by one of the processor's internal timers so that the processor can perform some time-scheduled task
- I/O – Generated by an I/O controller to request service from the processor such as keyboard, mouse, NIC, disk drive
- Hardware failure – signifies some error condition with the hardware

Program Flow Control



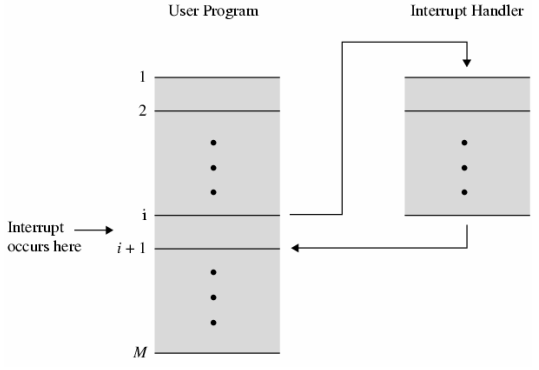
Interrupt Cycle



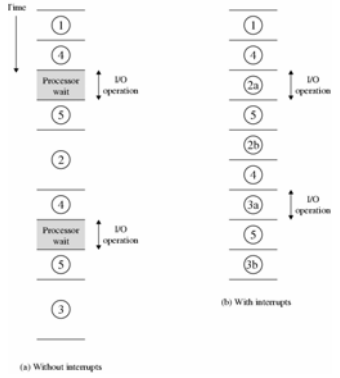
Interrupt Cycle (continued)

- Added to instruction cycle
- Processor checks for interrupt
- If no interrupt, fetch next instruction
- If interrupt pending:
  - Suspend execution of current program
  - Save context on stack (typically registers, PC, flags, etc.)
  - Set PC to start address of interrupt handler routine
  - Process interrupt
  - Restore context and continue interrupted program

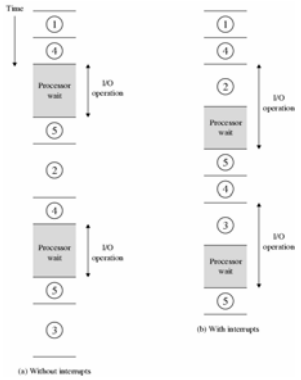
Transfer of Control via Interrupts



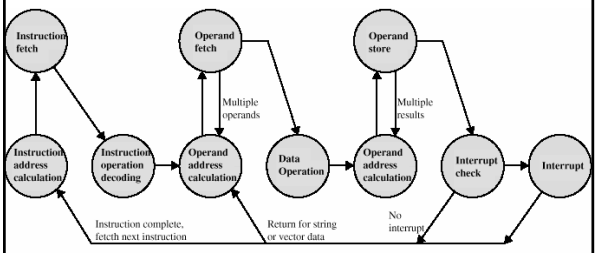
Program Timing -- Short I/O Wait



Program Timing -- Long I/O Wait



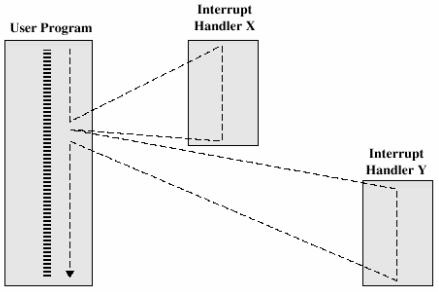
Instruction Cycle (with Interrupts) - State Diagram



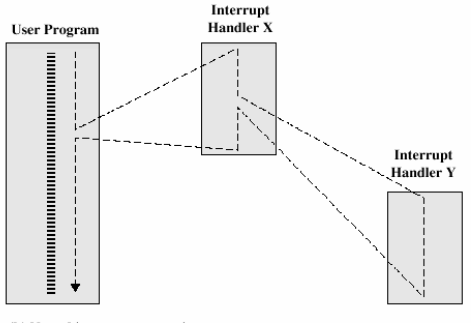
Multiple Interrupts

- Disable interrupts
  - Processor can ignore further interrupts whilst processing one interrupt or interrupts may be nested
  - Ignored interrupts remain pending and are checked after first interrupt has been processed
- Define priorities
  - Low priority interrupts can be interrupted by higher priority interrupts
  - When higher priority interrupt has been processed, processor returns to previous interrupt

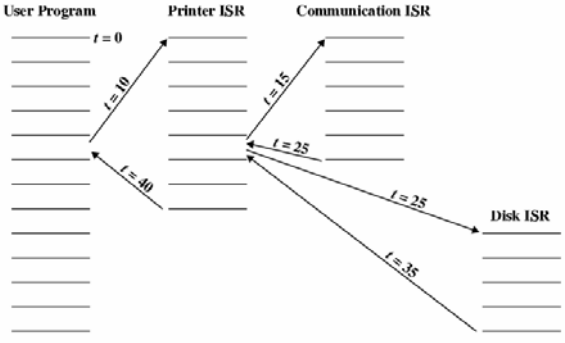
Multiple Interrupts - Sequential



Multiple Interrupts – Nested



Time Sequence of Multiple Interrupts



I/O Modules

- I/O modules occasionally require attention, usually in the form of a data transfer
- Processor can simply transfer data back and forth with the device as if it were memory
- Alternatively, processor can grant I/O module permission to write directly to memory – Direct Memory Access (DMA) – Interrupt occurs when DMA is complete