



The CPU and Memory

- Reading: Chapter 7



Reference

- USB
 - en.wikipedia.org/wiki/Universal_Serial_Bus
- 64 bit Intel Architecture



Intro

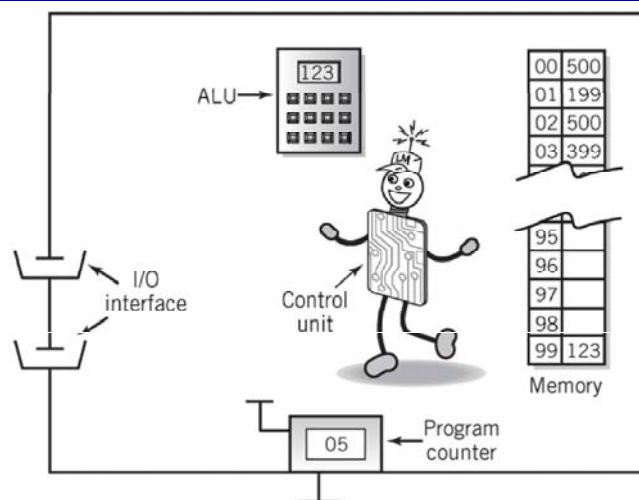
- Last session – reviewed actions of Little Man Computer (LMC)
- Compare architectural approaches of real computers with that of LMC
- Consider performance expectation based on components and architecture (other performance projection techniques later in the course)
- Specific examples next session

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-3



The Little Man Computer



Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-4



CPU: Major Components

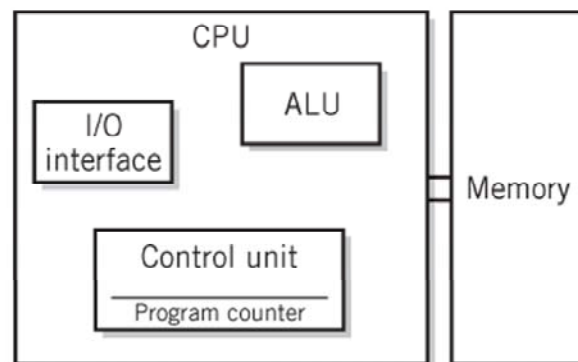
- **ALU** (arithmetic logic unit)
 - Performs calculations and comparisons
- **CU** (control unit)
 - Performs fetch/execute cycle
 - Accesses program instructions and issues commands to the ALU
 - Moves data to and from CPU registers and other hardware components
 - Subcomponents:
 - *Memory management unit*: supervises fetching instructions and data from memory
 - *I/O Interface*: sometimes combined with memory management unit as *Bus Interface Unit*

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-5



System Block Diagram



Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-6



Concept of Registers

Distance equates to time in a computer

Tradeoff of access time vs. capacity

- Small, *permanent* storage locations within the CPU used for a particular purpose
- Manipulated directly by the Control Unit
- Wired for *specific function*
- Size in bits or bytes (not in MB like memory)
- Can hold data, an address or an instruction
- Registers in the LMC – calculator and location counter

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-7



Registers

- Use of Registers
 - Scratchpad for currently executing program
 - Holds data needed quickly or frequently
 - Stores information about status of CPU and currently executing program
 - Address of next program instruction
 - Signals from external devices
- General Purpose Registers
 - *User-visible registers*
 - Hold intermediate results or data values, e.g., loop counters
 - Equivalent to LMC's calculator
 - Typically several dozen in current CPUs

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-8



Special-Purpose Registers

- *Program Count Register (PC)*
 - Also called instruction pointer
- *Instruction Register (IR)*
 - Stores instruction fetched from memory
- *Memory Address Register (MAR)*
- *Memory Data Register (MDR)*
- *Status Registers*
 - Status of CPU and currently executing program
 - *Flags* (one bit Boolean variable) to track condition like arithmetic carry and overflow, power failure, internal computer error

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-9



Register Operations

- Stores values from other locations (registers and memory)
- Addition and subtraction
- Shift or rotate data
- Test contents for conditions such as zero or positive

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-10



Operation of Memory

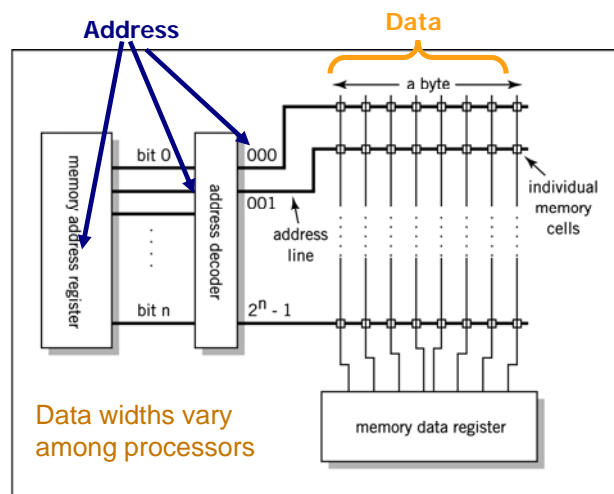
- Each memory location has a unique address
- Address from an instruction is copied to the MAR which finds the location in memory
- CPU determines if it is a store or retrieval
- Transfer takes place between the MDR and memory
- MDR is a two way register

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-11



Relationship between MAR, MDR and Memory

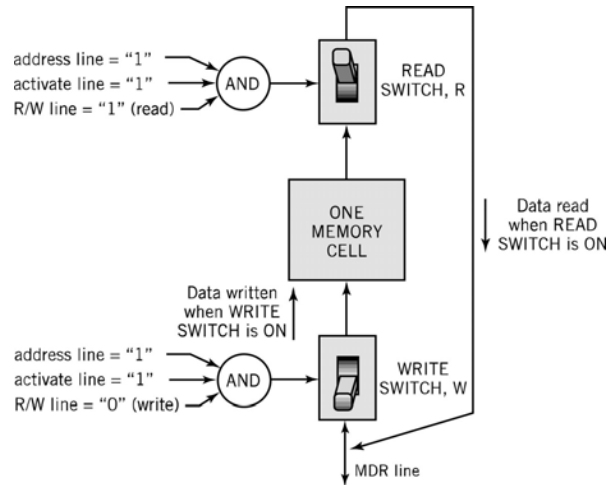


Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-12



Individual Memory Cell



Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-13



Memory Capacity

Primarily determined by two factors

1. Number of bits in the MAR
 - LMC = 100 (00 to 99)
 - 2^K where K = width of the register in bits
2. Size of the address portion of the instruction
 - 4 bits allows 16 locations
 - 8 bits allows 256 locations
 - 32 bits allows 4,294,967,296 or 4 GB

Compare this to the LMC

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-14



RAM: Random Access Memory

- *DRAM (Dynamic RAM)*
 - Most common, cheap, less electrical power, less heat, smaller space
 - Volatile: must be refreshed (recharged with power) 1000's of times each second
- *SRAM (static RAM)*
 - Faster and more expensive than DRAM
 - Volatile
 - Small amounts are often used in *cache memory* for high-speed memory access

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-15



Nonvolatile Memory

- *ROM*
 - Read-only Memory
 - Holds software that is not expected to change over the life of the system
- *EEPROM*
 - Electrically Erasable Programmable ROM
- *Flash Memory*
 - Faster than disks but more expensive
 - Uses hot carrier injection to store bits of data
 - Slow rewrite time compared to RAM
 - Useful for nonvolatile portable computer storage

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-16



Fetch-Execute Cycle

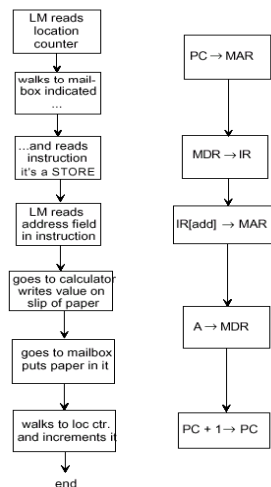
- Two-cycle process because both instructions and data are in memory
- *Fetch*
 - Decode or find instruction, load from memory into register and signal ALU
- *Execute*
 - Performs operation that instruction requires
 - Move/transform data

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-17



LMC vs. CPU Fetch and Execute Cycle



Notation: PC ⇒ MAR indicates movement of data (e.g., program counter to memory address register)

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-18



Load Fetch/Execute Cycle

1. PC \rightarrow MAR Transfer the address from the PC to the MAR
2. MDR \rightarrow IR Transfer the instruction to the IR (instruction register)
3. IR[address] \rightarrow MAR Address portion of the instruction loaded in MAR
4. MDR \rightarrow A Actual data copied into the accumulator
5. PC + 1 \rightarrow PC Program Counter incremented

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-19



Store Fetch/Execute Cycle

1. PC \rightarrow MAR Transfer the address from the PC to the MAR
2. MDR \rightarrow IR Transfer the instruction to the IR
3. IR[address] \rightarrow MAR Address portion of the instruction loaded in MAR
4. A \rightarrow MDR* Accumulator copies data into MDR
5. PC + 1 \rightarrow PC Program Counter incremented

*Notice how Step #4 differs for LOAD and STORE

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-20



ADD Fetch/Execute Cycle

- | | |
|----------------------------------|--|
| 1. PC \rightarrow MAR | Transfer the address from the PC to the MAR |
| 2. MDR \rightarrow IR | Transfer the instruction to the IR |
| 3. IR[address] \rightarrow MAR | Address portion of the instruction loaded in MAR |
| 4. A + MDR \rightarrow A | Contents of MDR added to contents of accumulator |
| 5. PC + 1 \rightarrow PC | Program Counter incremented |

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-21



Bus

- The physical connection that makes it possible to transfer data from one location in the computer system to another
- Group of electrical or optical conductors for carrying signals from one location to another
 - Wires or conductors printed on a circuit board
 - *Line*: each conductor in the bus
- 4 kinds of signals
 1. Data
 2. Addressing
 3. Control signals
 4. Power (sometimes)

Bus protocol covers the way components communicate over the bus

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-22



Bus Characteristics

- Number of separate conductors
- Data width in bits carried simultaneously
- Addressing capacity
- Lines on the bus are for a single type of signal or shared
- Throughput - data transfer rate (bps)
- Distance between two endpoints
- Number and type of attachments supported
- Type of control required
- Defined purpose
- Features and capabilities

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-23



Bus Categorizations

- Parallel vs. serial buses
- Direction of transmission
 - Simplex – unidirectional
 - Half duplex – bidirectional, one direction at a time
 - Full duplex – bidirectional simultaneously
- Method of interconnection
 - Point-to-point – single source to single destination
 - Cables – point-to-point buses that connect to an external device
 - Multipoint bus – also broadcast bus or multidrop bus
 - Connect multiple points to one another

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-24



Parallel vs. Serial Buses

- Serial
 - 1 bit transmitted at a time
 - Single data line pair and a few control lines
 - For many applications, throughput is higher than for parallel because of the lack of electrical interference
- Parallel
 - High throughput because all bits of a word are transmitted simultaneously
 - Expensive and require a lot of space
 - Subject to radio-generated electrical interference which limits their speed and length
 - Generally used for short distances such as CPU buses and on computer motherboards

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-25



Example - USB



- Universal Serial Bus
- Current generation – USB 2.0 (2000)
- Specification for communications between devices and a host
- Highlights
 - 5 meter max cable length
 - 5 volt DC power
 - 1-bit serial Note the bandwidth notation (bps vs. Bps)
 - 480 Mbps maximum bandwidth per controller
 - Controller directed (no requests from devices)

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

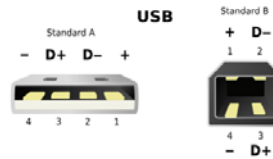
7-26



USB Connectors



- A, B, mini (falling out of use), and micro



Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-27



USB 3.0

- Higher transfer rate (5 Gbs)
- Device initiated communications
- Products becoming available during 2010

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-28



USB Alternatives

- FireWire
 - faster than USB 2.0 in actual use
 - Designed for audio/video applications
- 802.3af – Power over Ethernet (PoE)
 - Longer cable length, greater power
 - Useful for VoIP, security cameras, etc.

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-29



Classification of Instructions

- Data Movement (load, store)
 - Most common, greatest flexibility
 - Involve memory and registers
 - What's the size of a *word*? 16? 32? 64 bits or variable?
- Arithmetic
 - Operators + - / * ^
 - Integers and floating point

Word size is older terminology, and not particularly useful to describe modern computers
- Boolean Logic
 - Often includes at least **AND**, **XOR**, and **NOT**
- Single operand manipulation instructions
 - Negating, decrementing, incrementing, set to 0

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-30



Example

- What is the boolean AND, XOR, and OR of the following 8-bit binary numbers?

10110001

01100000

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-31



More Instruction Classifications

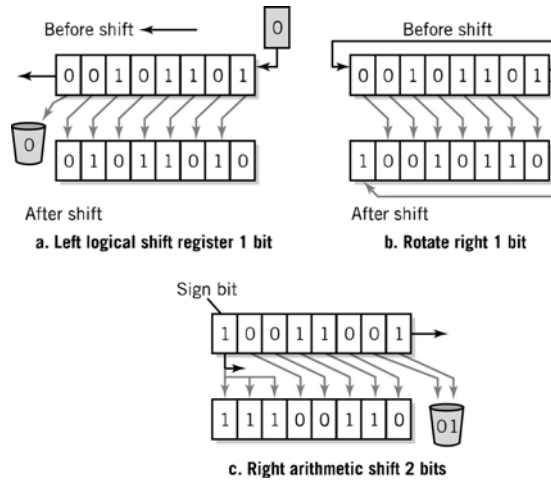
- Bit manipulation instructions
 - Flags to test for conditions
- Shift and rotate
- Program control
- Stack instructions
- Multiple data instructions
- I/O and machine control

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-32



Register Shifts and Rotates



Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

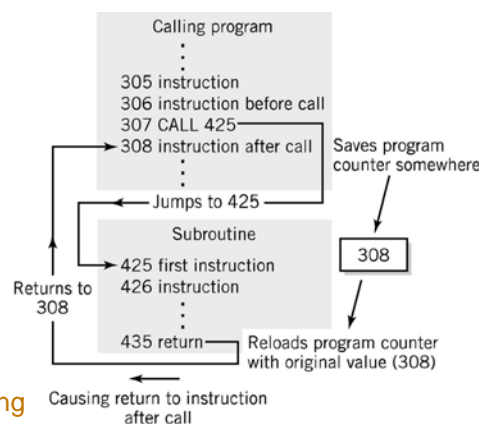
7-33



Program Control Instructions

- Program control
 - Jump and branch
 - Subroutine call and return

Some computers include instructions targeted to applications or programming languages



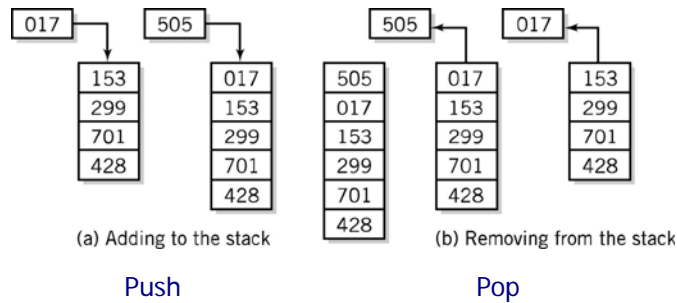
Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-34



Stack Instructions

- Stack instructions
 - LIFO method for organizing information
 - Items removed in the reverse order from that in which they are added

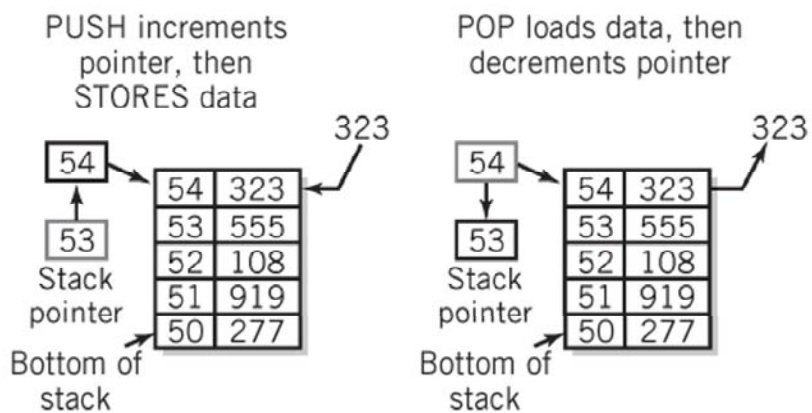


Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-35



Block of Memory as a Stack



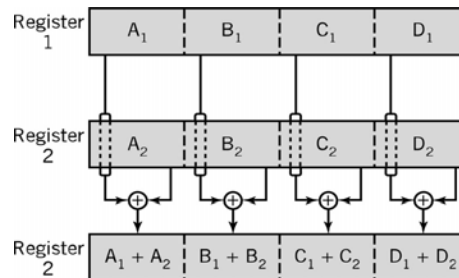
Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-36



Multiple Data Instructions

- Perform a single operation on multiple pieces of data simultaneously
 - SIMD: Single Instruction, Multiple Data
 - Commonly used in multimedia, *vector* and array processing applications



Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-37



Instruction Elements

- OPCODE: task
 - Source OPERAND(s)
 - Result OPERAND
- } Addresses
- Location of data (register, memory)
 - Explicit: included in instruction
 - Implicit: default assumed

OPCODE	Source OPERAND	Result OPERAND
--------	----------------	----------------

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-38



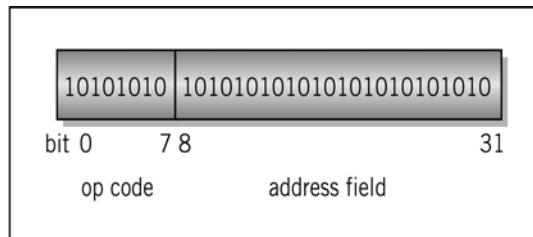
Instruction Format

- *Machine-specific* template that specifies
 - Length of the op code
 - Number of operands
 - Length of operands

More complex instruction formats use multiple address fields

Simple 32-bit Instruction Format

Number of bits for the op code corresponds to # of instructions



Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-39



Instructions

- Instruction
 - Direction given to a computer
 - Causes electrical or optical signals to be sent through specific circuits for processing
- Instruction set
 - Design defines functions performed by the processor
 - Differentiates computer architecture by the
 - Number of instructions
 - Complexity of operations performed by individual instructions
 - Data types supported
 - Format (layout, fixed vs. variable length)
 - Use of registers
 - Addressing (size, modes)

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-40



Instruction Word Size

- Fixed vs. variable size
 - Pipelining has mostly eliminated variable instruction size architectures Pipelining is covered in the next chapter
- Most current architectures use byte multiples (32-bit or 64-bit) for addressing, data paths, etc.
- Addressing Modes
 - Direct - Mode used by the LMC
 - Register Deferred
 - Also immediate, indirect, indexed

Copyright 2010-2011 John Wiley & Sons, Inc. & Robert F. Kelly

7-41