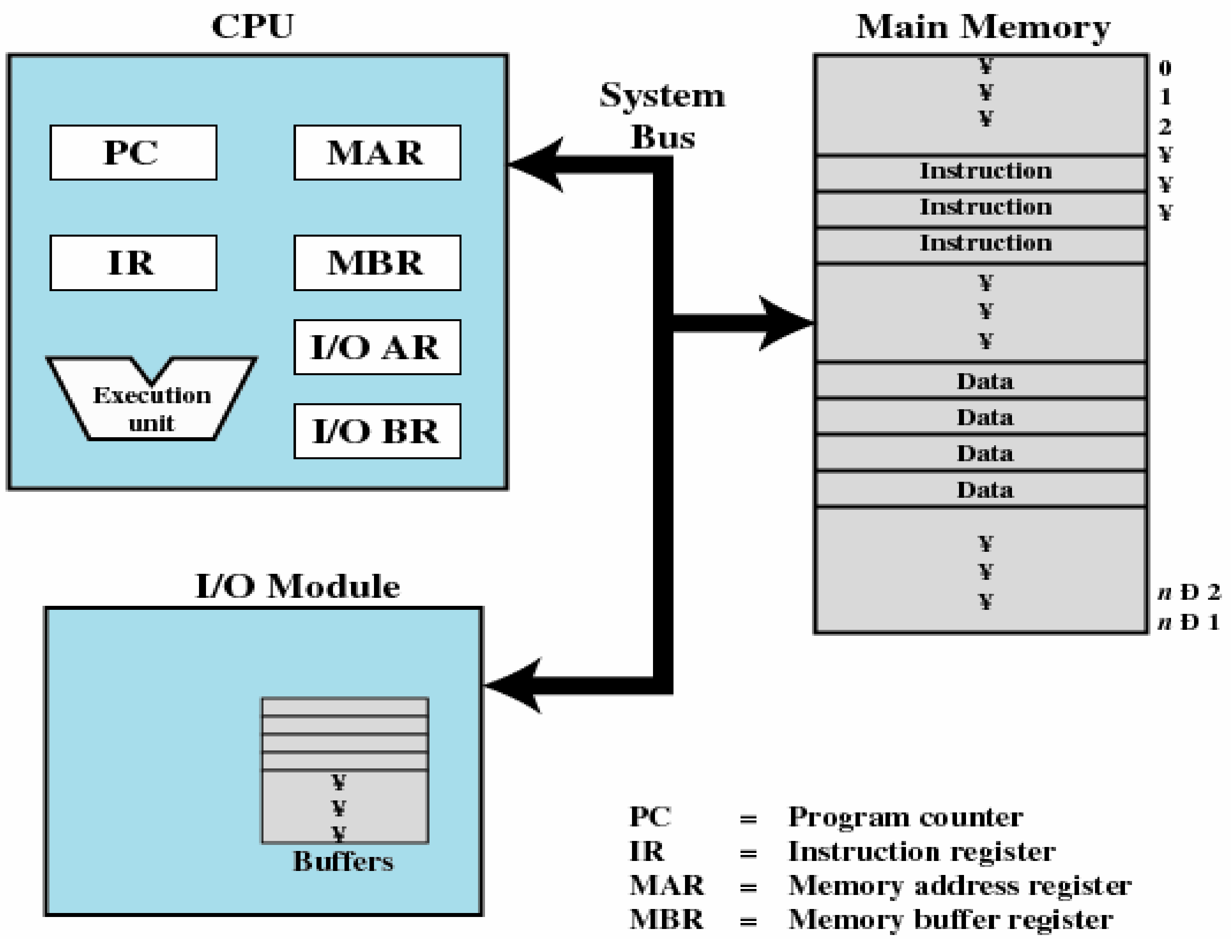
CGS 3763: Operating System Concepts
Spring 2006

Chapter 2 – Hardware – Part 1

Instructor : Mark Llewellyn
markl@cs.ucf.edu
CSB 242, 823-2790
http://www.cs.ucf.edu/courses/cgs3763/spr2006

School of Electrical Engineering and Computer Science
University of Central Florida

Top-Level Computer Components



PC	=	Program counter
IR	=	Instruction register
MAR	=	Memory address register
MBR	=	Memory buffer register
I/O AR	=	Input/output address register
I/O BR	=	Input/output buffer register

Processor Registers

- User-visible registers
  - Enable programmer to minimize main-memory references by optimizing register use
- Control and status registers
  - Used by processor to control operating of the processor
  - Used by privileged operating-system routines to control the execution of programs

User-Visible Registers

- May be referenced by machine language
- Available to all programs - application programs and system programs
- Types of registers
  - Data
  - Address
    - Index
    - Segment pointer
    - Stack pointer

User-Visible Registers

- Address Registers
  - Index
    - Involves adding an index to a base value to get an address
  - Segment pointer
    - When memory is divided into segments, memory is referenced by a segment and an offset
  - Stack pointer
    - Points to top of stack

Control and Status Registers

- Program Counter (PC)
  - Contains the address of an instruction to be fetched
- Instruction Register (IR)
  - Contains the instruction most recently fetched
- Program Status Word (PSW)
  - Condition codes
  - Interrupt enable/disable
  - Supervisor/user mode

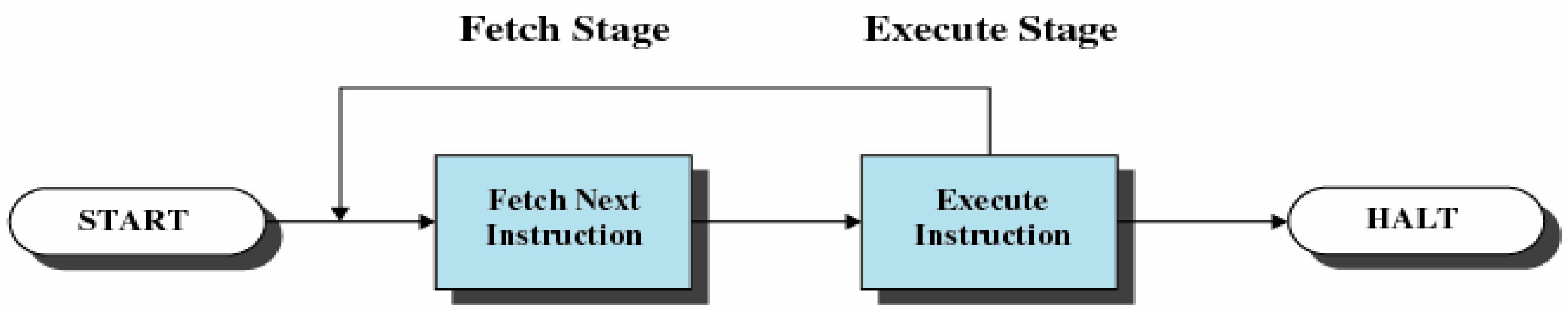
Control and Status Registers

- Condition Codes or Flags
  - Bits set by the processor hardware as a result of operations
  - Examples
    - Positive result
    - Negative result
    - Zero
    - Overflow

Instruction Execution

- Two steps (Fetch/Decode & Execute)
  - Processor reads instructions from memory
    - Fetches into the MBR (MDR)
  - Processor executes each instruction

Instruction Cycle



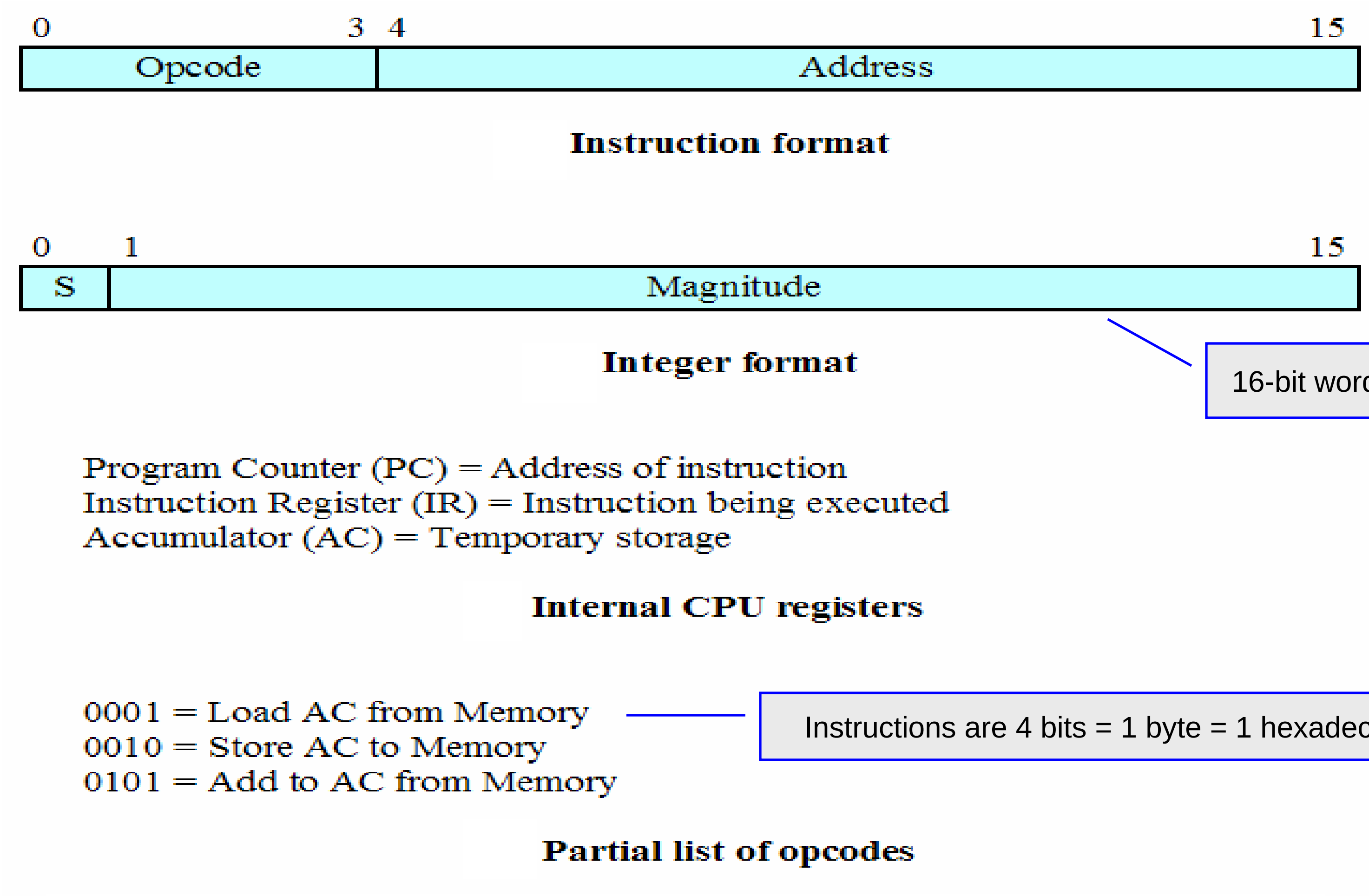
Instruction Fetch and Execute

- The processor fetches the instruction from memory
- Program counter (PC) holds address of the instruction to be fetched next
- Program counter is incremented after each fetch

Instruction Register

- Fetched instruction is placed in the instruction register
- Categories
  - Processor-memory
    - Transfer data between processor and memory
  - Processor-I/O
    - Data transferred to or from a peripheral device
  - Data processing
    - Arithmetic or logic operation on data
  - Control
    - Alter sequence of execution

A Hypothetical Machine



Instruction format

Integer format

16-bit word = 4 bytes

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

Internal CPU registers

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

Instructions are 4 bits = 1 byte = 1 hexadecimal digit

Partial list of opcodes

An Aside on Number Systems

Binary = base 2, digits are 0, 1

Decimal = base 10, digits are 0,1,2,3,4,5,6,7,8,9

Hexadecimal = base 16, digits are 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

1 **bi**nary digi**t** = 1 bit. 1 bit can represent either 0 or 1 (e.g. on or off)

Positional notation is represented via digit position as a power of 2.

$(x + 2^n) + \ldots + 2^5 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0$ (recall that $x^0 = 1$ and $0^x = 0$)

So, 2 bits can represent up to base 4 (digits 0,1,2,3), since

$(0 + 2^1) + (0 + 2^0) = 0$ $(0 + 2^1) + (1 + 2^0) = 1$

$(1 + 2^1) + (0 + 2^0) = 2$ $(1 + 2^1) + (1 + 2^0) = 3$

An Aside on Number Systems

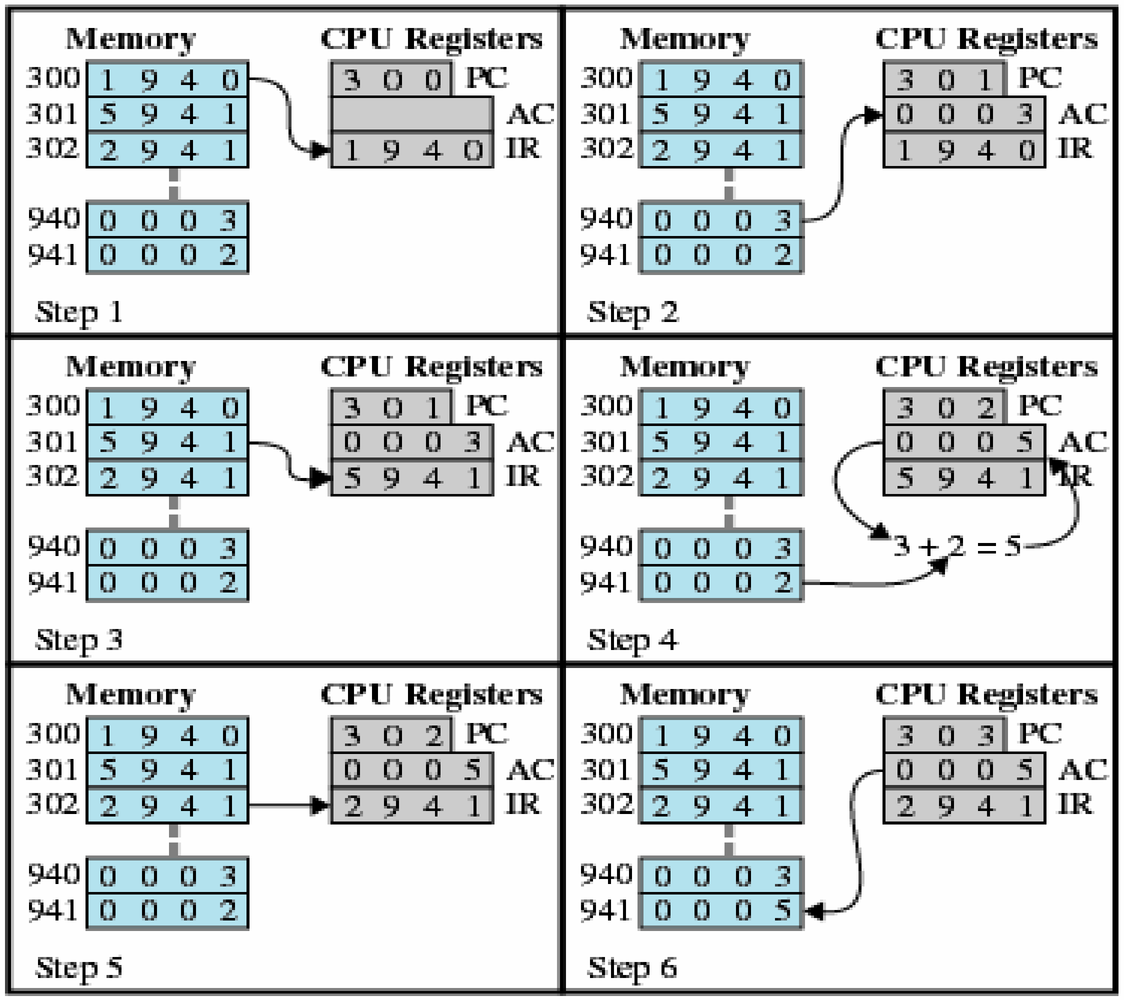
HEX	DECIMAL	OCTAL	BINARY
0	0	0	0000
1	1	1	0001
2	2	2	0010
3	3	3	0011
4	4	4	0100
5	5	5	0101
6	6	6	0110
7	7	7	0111
8	8	10	1000
9	9	11	1001
A	10	12	1010
B	11	13	1011
C	12	14	1100
D	13	15	1101
E	14	16	1110
F	15	17	1111

For hexadecimal, 4 bits are required to represent 1 hex digit.

The hex number 345 in binary would be written as 1101000101 (leading zeros are dropped).

$$\underbrace{1\ 1}_{= 3_{16}}\ \underbrace{0\ 1\ 0\ 0}_{= 4_{16}}\ \underbrace{0\ 1\ 0\ 1}_{= 5_{16}}$$

Example of Program Execution



Explanation of Example Program Execution

1. The PC contains 300, the address of the first instruction. This instruction (with value 1940 in hexadecimal) is loaded into the instruction register (IR) and the program counter (PC) is incremented. Note that both the MAR and MDR are used in this step but are not shown in the previous slide.

2. The first 4 bits (first hex digit) in the IR indicates that the accumulator register (AC) is to be loaded from memory. Decode of opcode indicates 0001 which represents the load accumulator from memory instruction. The remaining 12 bits (3 hex digits) represents the memory address where the operand (the value to be loaded into the accumulator) is located. This address is 940 (also in hex).

Explanation of Example Program Execution

3. The next instruction (hex 5941), is fetched from memory location 301 and the PC is incremented. The decode of this instruction indicates an instruction (opcode 0101) that adds the value in the AC to a value obtained from the memory location specified in the operand of the instruction as (hex 941).

4. The old contents of the AC and the contents of memory location 941 are added together and the result is stored in the AC.

5. The next instruction (hex 2941), is fetched from memory location 302 and the PC is incremented. The decode of this instruction indicates an instruction (opcode 0010) that stores the contents of the AC into the memory location specified in the operand of the instruction as (hex 941).

6. The contents of the AC are stored in memory location 941.