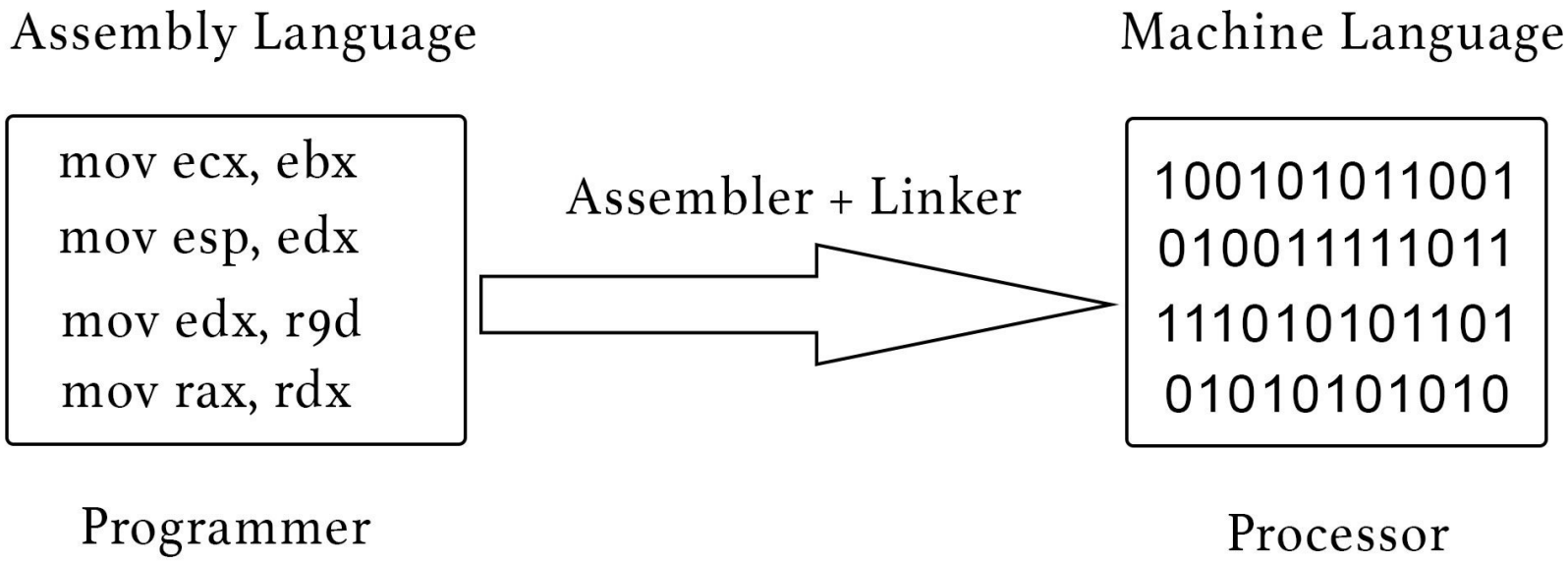
电子科技大学
格拉斯哥学院
Glasgow College, UESTC

UESTC4019: Real-Time Computer Systems and Architecture

Lecture 12
Instruction Sets – Characteristics and Function (Part-1)

Assembly Code vs Machine Code



Machine Instruction Characteristics

- The operation of the processor is determined by the instructions it executes, referred to as machine instructions or computer instructions
- The collection of different instructions that the processor can execute is referred to as the processor's instruction set
- Each instruction must contain the information required by the processor for execution

Elements of a Machine Instruction (1 of 2)

- Operation code (opcode)
  - Specifies the operation to be performed. The operation is specified by a binary code, known as the operation code, or opcode

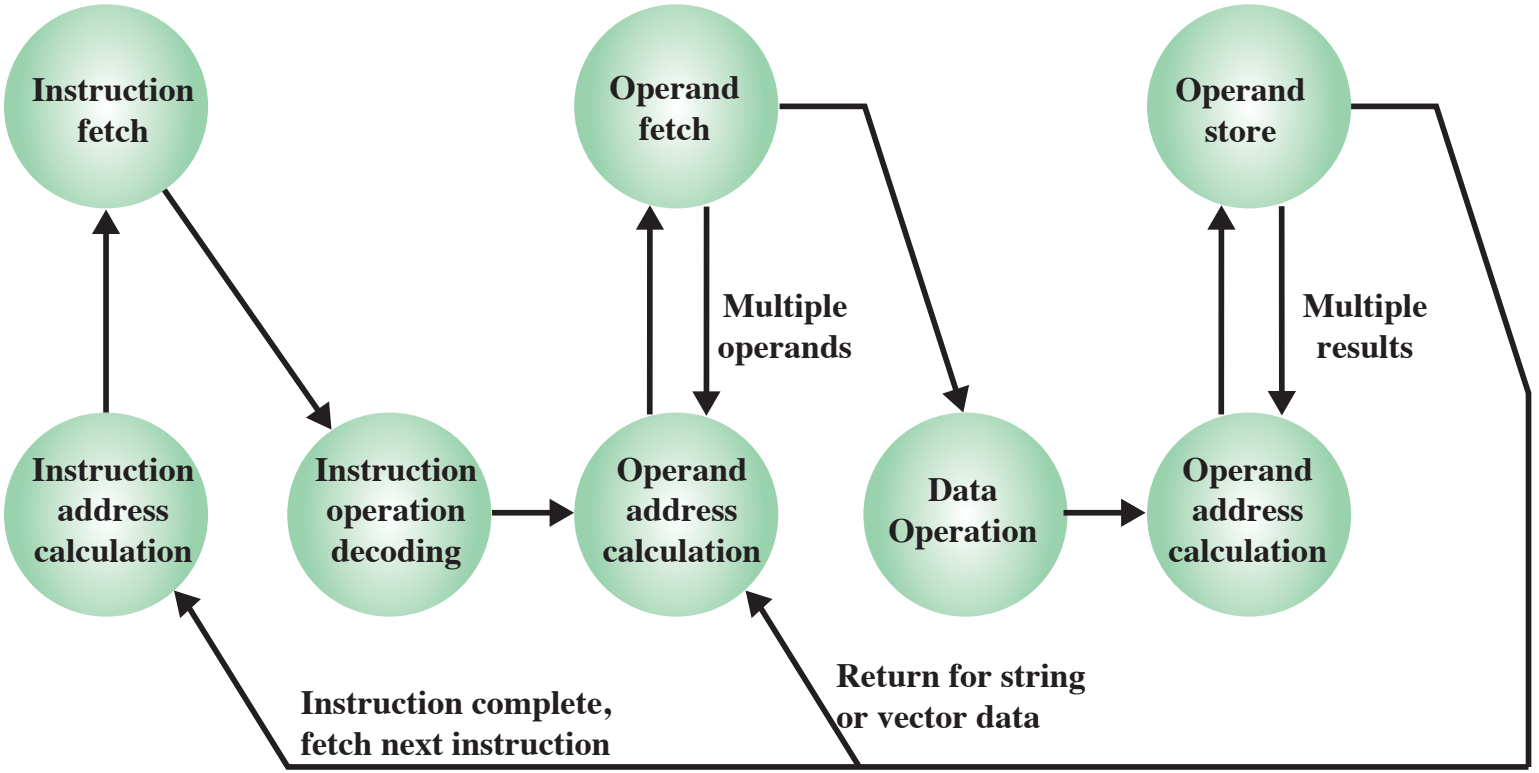
- Source operand reference
  - The operation may involve one or more source operands, that is, operands that are inputs for the operation

Elements of a Machine Instruction (2 of 2)

- Result operand reference
  - The operation may produce a result

- Next instruction reference
  - This tells the processor where to fetch the next instruction after the execution of this instruction is complete

Instruction Cycle State Diagram



Source and Result Operands (1 of 2)

- Main or virtual memory
  - As with next instruction references, the main or virtual memory address must be supplied

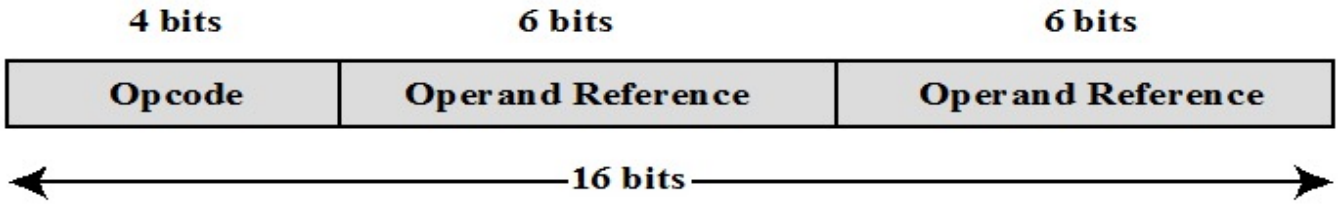
- I/O device
  - The instruction must specify the I/O module and device for the operation. If memory-mapped I/O is used, this is just another main or virtual memory address

Source and Result Operands (2 of 2)

- Processor register
  - A processor contains one or more registers that may be referenced by machine instructions.
  - If more than one register exists each register is assigned a unique name or number and the instruction must contain the number of the desired register
- Immediate
  - The value of the operand is contained in a field in the instruction being executed

Instruction Representation (1 of 3)

- Within the computer each instruction is represented by a sequence of bits
- The instruction is divided into fields, corresponding to the constituent elements of the instruction



A Simple Instruction Format

Instruction Representation (2 of 3)

- Opcodes are represented by abbreviations called mnemonics
- Examples include:
  - ADD Add
  - SUB Subtract
  - MUL Multiply
  - DIV Divide
  - LOAD Load data from memory
  - STOR Store data to memory

Instruction Representation (3 of 3)

- Operands are also represented symbolically
- Each symbolic opcode has a fixed binary representation
- The programmer specifies the location of each symbolic operand

Instruction Types (1 of 2)

- Data processing
  - Arithmetic instructions provide computational capabilities for processing numeric data
  - Logic (Boolean) instructions operate on the bits of a word as bits rather than as numbers, thus they provide capabilities for processing any other type of data the user may wish to employ
- Data storage
  - Movement of data into or out of register and or memory locations

Instruction Types (2 of 2)

- Data movement
  - I/O instructions are needed to transfer programs and data into memory and the results of computations back out to the user
- Control
  - Test instructions are used to test the value of a data word or the status of a computation
  - Branch instructions are used to branch to a different set of instructions depending on the decision made

Utilization of Instruction Addresses (Nonbranching Instructions)

Number of Addresses	Symbolic Representation	Interpretation
3	OP A,B,C	$A \leftarrow B\ OP\ C$
2	OP A,B	$A \leftarrow A\ OP\ B$
1	OP A	$AC \leftarrow AC\ OP\ A$
0	OP	$T \leftarrow (T-1)\ OP\ T$

- AC=accumulator
- T= top of stack
- (T-1)=second element of stack
- A,B,C=memory or register locations

Example Program to Execute

Instruction	Comment
SUB	$Y \leftarrow A - B$
MPY	$T \leftarrow D \times E$
ADD	$T \leftarrow T + C$
DIV	$Y \leftarrow Y \div T$

(a) Three-address instructions

Instruction	Comment
MOVE	$Y \leftarrow A$
SUB	$Y \leftarrow Y - B$
MOVE	$T \leftarrow D$
MPY	$T \leftarrow T \times E$
ADD	$T \leftarrow T + C$
DIV	$Y \leftarrow Y \div T$

(b) Two-address instructions

$$Y = \frac{A - B}{C + (D \times E)}$$

Instruction	Comment
LOAD	$AC \leftarrow D$
MPY	$AC \leftarrow AC \times E$
ADD	$AC \leftarrow AC + C$
STOR	$Y \leftarrow AC$
LOAD	$AC \leftarrow A$
SUB	$AC \leftarrow AC - B$
DIV	$AC \leftarrow AC \div Y$
STOR	$Y \leftarrow AC$

(c) One-address instructions

Example from Last Year Exam

Compare zero-, one-, two-, and three-address machines by writing programs to compute.

$$X = (A + B * C)/(D - E * F)$$

for each of the four machines given in Table. The instructions available for use are as follows:

0 Address	1 Address	2 Address	3 Address
PUSH M	LOAD M	MOVE (X ← Y)	MOVE (X ← Y)
POP M	STORE M	ADD (X ← X + Y)	ADD (X ← Y + Z)
ADD	ADD M	SUB (X ← X − Y)	SUB (X ← Y − Z)
SUB	SUB M	MUL (X ← X × Y)	MUL (X ← Y × Z)
MUL	MUL M	DIV (X ← X/Y)	DIV (X ← Y/Z)
DIV	DIV M		

0 Address	1 Address	2 Address	3 Address
PUSH M	LOAD M	MOVE (X ← Y)	MOVE (X ← Y)
POP M	STORE M	ADD (X ← X + Y)	ADD (X ← Y + Z)
ADD	ADD M	SUB (X ← X − Y)	SUB (X ← Y − Z)
SUB	SUB M	MUL (X ← X × Y)	MUL (X ← Y × Z)
MUL	MUL M	DIV (X ← X/Y)	DIV (X ← Y/Z)
DIV	DIV M		

Instruction Set Design (1 of 2)

- Instruction set is very complex because it affects so many aspects of the computer system
- Defines many of the functions performed by the processor
- Programmer's means of controlling the processor
- Fundamental design issues:
  - Operation repertoire
    - How many and which operations to provide and how complex operations should be
  - Data types
    - The various types of data upon which operations are performed

Instruction Set Design (2 of 2)

- Instruction format
  - Instruction length in bits, number of addresses, size of various fields, etc.
- Registers
  - Number of processor registers that can be referenced by instructions and their use
- Addressing
  - The mode or modes by which the address of an operand is specified

Types of Operands

- The most important general categories of data are
  - Addresses
  - Numbers
  - Characters
  - Logical Data

Numbers (1 of 2)

- All machine languages include numeric data types
- Numbers stored in a computer are limited:
  - Limit to the magnitude of numbers representable on a machine
  - In the case of floating-point numbers, a limit to their precision
  - Three types of numerical data are common in computers:
    - Binary integer or binary fixed point
    - Binary floating point
    - Decimal

Numbers (2 of 2)

- Packed decimal
- Each decimal digit is represented by a 4-bit code with two digits stored per byte
- To form numbers 4-bit codes are strung together, usually in multiples of 8 bits

Characters

- A common form of data is text or character strings
- Textual data in character form cannot be easily stored or transmitted by data processing and communications systems because they are designed for binary data
- Most commonly used character code is the International Reference Alphabet (IRA)
  - Referred to in the United States as the American Standard Code for Information Interchange (ASCII)
  - Another code used to encode characters is the Extended Binary Coded Decimal Interchange Code (EBCDIC)
  - EBCDIC is used on IBM mainframes

Logical Data

- An n-bit unit consisting of *n* 1-bit items of data, each item having the value 0 or 1
- Two advantages to bit-oriented view:
  - Memory can be used most efficiently for storing an array of Boolean or binary data items in which each item can take on only the values 1 (true) and 0 (false)
  - To manipulate the bits of a data item
    - If floating-point operations are implemented in software, we need to be able to shift significant bits in some operations
    - To convert from IRA to packed decimal, we need to extract the rightmost 4 bits of each byte