جَامعة المَنارة
MANARA UNIVERSITY

كلية الهندسة المعلوماتية
بنيان حواسيب 2
الفصل الأول 2024-2025
المحاضرة الثالثة

د كندة أبو قاسم

مجموعة التعليمات Instruction Set

- Overview of assembly language instructions

- Assembly language statements
- Data allocation
- Where are the operands?
  - Addressing modes
    - Register
    - Immediate
    - Direct
    - Indirect
- Data transfer instructions
  - **mov**, **xchg**, and **xlat**
  - Ambiguous moves

للـ 8086 6 أنماط للتعليمات.

1. تعليمات نقل البيانات **Data Transfer Instructions**
2. التعليمات الحسابية **Arithmetic Instructions**
3. التعليمات المنطقية **Logical Instructions**
4. تعليمات معالجة السلاسل **String manipulation Instructions**
5. تعليمات التحكم بالعملية **Process Control Instructions**
6. تعليمات التحكم بالنقل **Control Transfer Instructions**

Addressing mode

- Operands required by an operation can be specified in a variety of ways
- A few basic ways are:
  - operand is in a register
    - register addressing mode
  - operand is in the instruction itself
    - immediate addressing mode
  - operand is in the memory
    - variety of addressing modes
      - ✈direct and indirect addressing modes
  - operand is at an I/O port

Register Addressing Mode

- Most efficient way of specifying an operand
  - » operand is in an internal register

Examples

```
mov     EAX,EBX
mov     BX,CX
```

- The **mov** instruction

  mov destination,source

  copies data from **source** to **destination**

جَامعة المَنارة

Direct Addressing Mode (cont'd)

Examples

```
mov     AL,[response]
```

» Assembler replaces **response** by its effective address (i.e., its offset value from the symbol table)

```
mov     [table1],56
```

» **table1** is declared as

```
table1   TIMES  20  DW  0
```

» Since the assembler replaces **table1** by its effective address, this instruction refers to the first element of **table1**

– In C, it is equivalent to

```
table1[0] = 56
```

Immediate Addressing Mode

* Data is part of the instruction
  - » operand is located in the code segment along with the instruction
  - » Efficient as no separate operand fetch is needed
  - » Typically used to specify a constant

Example

```
mov     AL,75
```

* This instruction uses register addressing mode for specifying the *destination* and immediate addressing mode to specify the *source*

Indirect Addressing Mode

- The offset is specified indirectly via a register
  - Sometimes called *register indirect* addressing mode
  - For 16-bit addressing, the offset value can be in one of the three registers: BX, SI, or DI
  - For 32-bit addressing, all 32-bit registers can be used

Example

```
mov     AX,[EBX]
```

- Square brackets [ ] are used to indicate that EBX is holding an offset value
  - » EBX contains a pointer to the operand, not the operand itself

Direct Addressing Mode (cont'd)

- Problem with direct addressing
  - Useful only to specify simple variables
  - Causes serious problems in addressing data types such as arrays
    - » As an example, consider adding elements of an array
      - – Direct addressing does not facilitate using a loop structure to iterate through the array
      - – We have to write an instruction to add each element of the array
- Indirect addressing mode remedies this problem

مجموعة التعليمات Instruction Set

1. تعليمات نقل البيانات

تستخدم تعليمات هذا النمط لنقل البيانات /العناوين من وإلى المسجلات ومن وإلى مواقع الذاكرة ومن وإلى بوابات I/O.
Instructions that are used to transfer data/ address in to registers, memory locations and I/O ports.

المستهدف بشكل عام هو حدين two operands: حدّي المصدر Source والهدف نبفس الحجم same size.

المصدر Source: مسجل أو موقع ذاكرة أو بيانات فورية immediate data.
الهدف Destination: مسجل أو موقع ذاكرة.

بحجم بايت أو كلمة byte or a word.

يمكننا نقل بيانات بحجم 8-bit إلى مسجلات بحجم ومواقع ذاكرة بنفس الحجم والبيانات بحجم 16-bit إلى مسجلات وموقع ذاكرة بنفس الحجم.

تعريف المعطيات يمكن تعريف المعطيات في لغة التجميع

تعرف المعطيات في مقطع data segment
يمكن أن تعرف اما DB byte
أو كلمة من Word 16 bit
من 32 bit double Word

- Fivfor initialized data

```
DB   Define Byte e define directives      ;allocates 1 byte
DW   Define Word          ;allocates 2 bytes
DD   Define Doubleword    ;allocates 4 bytes
DQ   Define Quadword      ;allocates 8 bytes
DT   Define Ten bytes     ;allocates 10 bytes
```

Examples

```
sorted     DB    'y'
value      DW    25159
Total      DD    542803535
float1     DD    1.234
```

تعريف مصفوفة أحادية البعد من 8 عناصر

- Multiple definitions can be cumbersome to initialize data structures such as arrays

Example

To declare and initialize an integer array of 8 elements

```
marks   DW    0,0,0,0,0,0,0,0
```

- What if we want to declare and initialize to zero an array of 200 elements?
  - There is a better way of doing this than repeating zero 200 times in the above statement
    - » Assembler provides a directive to do this (DUP directive)

* Examples
  - » Previous marks array

```
marks   DW    0,0,0,0,0,0,0,0
```

can be compactly declared as

```
marks   TIMES   8   DW   0
```

Symbol Table

* Assembler builds a symbol table so we can refer to the allocated storage space by the associated label

Example

```
.DATA
value     DW   0
sum       DD   0
marks     DW   10 DUP (?)
message   DB   'The grade is:',0
char1     DB   ?
```

name	offset
value	0
sum	2
marks	6
message	26
char1	40

تعليمات نقل البيانات
Data Transfer Instructions

Mnemonics: MOV, XCHG, PUSH, POP, IN, OUT …

* The format is

  mov destination,source

  - » Copies the value from **source** to **destination**
  - » **source** is not altered as a result of copying
  - » Both operands should be of same size
  - » **source** and **destination** cannot both be in memory

MOV reg2/ mem, reg1/ mem	
MOV reg2, reg1	(reg2) ← (reg1)
MOV mem, reg1	(mem) ← (reg1)
MOV reg2, mem	(reg2) ← (mem)

تعليمة Mov
نقل من مسجل الى مسجل أخر
نقل من مسجل الى ذاكرة
نقل من ذاكرة الى مسجل

MOV reg/ mem, data	
MOV reg, data	(reg) ← data
MOV mem, data	(mem) ← data

تعليمة Mov
نقل معطيات الى مسجل أخر
نقل معطيات الى موقع ذاكرة

XCHG reg2/ mem, reg1	
XCHG reg2, reg1	(reg2) ↔ (reg1)
XCHG mem, reg1	(mem) ↔ (reg1)

تعليمة تبديل XCHG
تبديل محنوى مسجل بمحنوى مسجل أخر
تبديل محتوى مسجل بمحنوى موقع ذاكرة

جَامعة المَنارة
MANARA UNIVERSITY

The mov instruction

* Five types of operand combinations are allowed:

Instruction type	Example
`mov register,register`	`mov DX,CX`
`mov register,immediate`	`mov BL,100`
`mov register,memory`	`mov EBX,[count]`
`mov memory,register`	`mov [count],ESI`
`mov memory,immediate`	`mov [count],23`

جَامعة المَنارة
MANARA UNIVERSITY

مثال :
حدد الأخطاء في البرنامج التالي

```
.data
bVal   BYTE   100
bVal2  BYTE   ?
wVal   WORD   2
dVal   DWORD  5
.code
    mov ds,45
    mov esi,wVal
    mov eip,dVal
    mov 25,bVal
    mov bVal2,bVal
```

immediate move to DS not permitted
size mismatch عدم تطابق الحجم

لا يمكن أن يكون المسجل هدف IP or eip
immediate value cannot be destination

memory-to-memory move not permitted
لا يُسمح بنقل البيانات من ذاكرة إلى ذاكرة