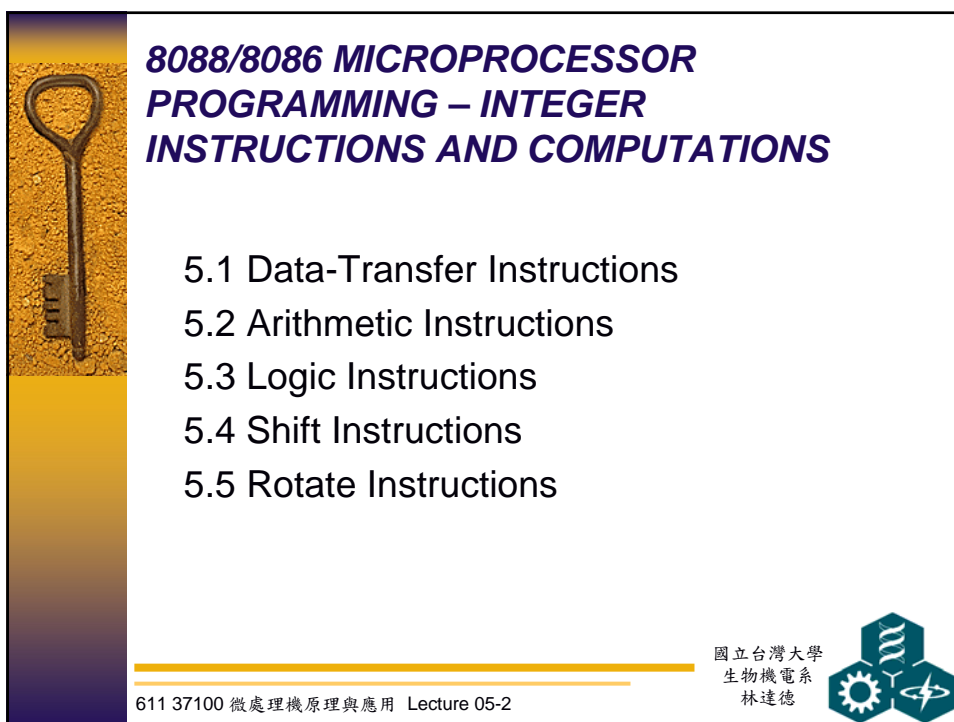




***8088/8086 MICROPROCESSOR
PROGRAMMING – INTEGER
INSTRUCTIONS AND
COMPUTATIONS***



***8088/8086 MICROPROCESSOR
PROGRAMMING – INTEGER
INSTRUCTIONS AND COMPUTATIONS***

- 5.1 Data-Transfer Instructions
- 5.2 Arithmetic Instructions
- 5.3 Logic Instructions
- 5.4 Shift Instructions
- 5.5 Rotate Instructions





5.1 Data-Transfer Instructions

- The data-transfer functions provide the ability to move data either between its internal registers or between an internal register and a storage location in memory.
- The data-transfer functions include
 - ❖ MOV (Move byte or word)
 - ❖ XCHG (Exchange byte or word)
 - ❖ XLAT (Translate byte)
 - ❖ LEA (Load effective address)
 - ❖ LDS (Load data segment)
 - ❖ LES (Load extra segment)

611 37100 微處理機原理與應用 Lecture 05-3

國立台灣大學
生物機電系
林達德



5.1 Data-Transfer Instructions

■ The MOVE Instruction

The move (MOV) instruction is used to transfer a byte or a word of data from a source operand to a destination operand.

Mnemonic	Meaning	Format	Operation	Flags affected
MOV	Move	MOV D, S	(S) → (D)	None

e.g. MOV DX, CS

 MOV [SUM], AX

611 37100 微處理機原理與應用 Lecture 05-4

國立台灣大學
生物機電系
林達德



5.1 Data-Transfer Instructions

■ The MOVE Instruction

Note that the MOV instruction cannot transfer data directly between external memory.

Destination	Source
Memory	Accumulator
Accumulator	Memory
Register	Register
Register	Memory
Memory	Register
Register	Immediate
Memory	Immediate
Seg-reg	Reg16
Seg-reg	Mem16
Reg16	Seg-reg
Memory	Seg-reg

Allowed operands for MOV instruction

611 37100 微處理機原理與應用 Lecture 05-5

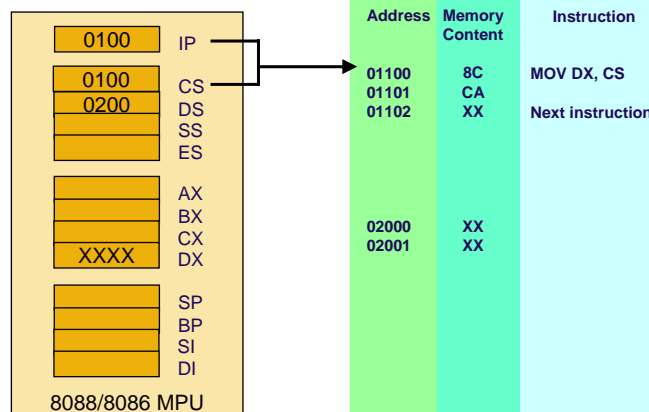
國立台灣大學
生物機電系
林達德



5.1 Data-Transfer Instructions

■ The MOVE Instruction

MOV DX, CS



Before execution

611 37100 微處理機原理與應用 Lecture 05-6

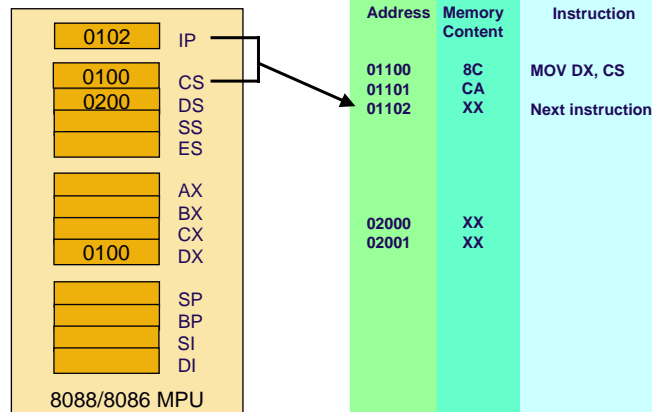
國立台灣大學
生物機電系
林達德



5.1 Data-Transfer Instructions

■ The MOVE Instruction

MOV DX, CS



After execution

611 37100 微處理機原理與應用 Lecture 05-7

國立台灣大學
生物機電系
林達德



5.1 Data-Transfer Instructions

EXAMPLE

What is the effect of executing the instruction

MOV CX, [SOURCE_MEM]

Where SOURCE_MEM equal to 20_{16} is a memory location offset relative to the current data segment starting at $1A000_{16}$.

Solution:

$$((DS)0 + 20_{16}) \rightarrow (CL)$$

$$((DS)0 + 20_{16} + 1_{16}) \rightarrow (CH)$$

Therefore CL is loaded with the contents held at memory address

$$1A000_{16} + 20_{16} = 1A020_{16}$$

and CH is loaded with the contents of memory address

$$1A000_{16} + 20_{16} + 1_{16} = 1A021_{16}$$

611 37100 微處理機原理與應用 Lecture 05-8

國立台灣大學
生物機電系
林達德





5.1 Data-Transfer Instructions

EXAMPLE

Use the DEBUG to verify the previous example.

Solution:

```

CONSOLE MODE - debug
-r
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=0B35 ES=0B35 SS=0B35 CS=0B35 IP=0100 NV UP EI PL NZ NA PO NC
0B35:0100 E375 JXZ 0177
-a
0B35:0100 MOV CX, [20]
0B35:0104
-r DS
DS 0B35
:1A00
-E 20 55 AA
-T
AX=0000 BX=0000 CX=AA55 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=1A00 ES=0B35 SS=0B35 CS=0B35 IP=0104 NV UP EI PL NZ NA PO NC
0B35:0104 A0D396 MOV AL,[96D3] DS:96D3=00

```



5.1 Data-Transfer Instructions

■ The XCHG Instruction

The exchange (XCHG) instruction can be used to swap data between two general-purpose registers or between a general-purpose register and a storage location in memory.

Mnemonic	Meaning	Format	Operation	Flags affected
XCHG	Exchange	XCHG D, S	(D) ↔ (S)	None

e.g. XCHG AX, DX

Destination	Source
Accumulator	Reg16
Memory	Register
Register	Register
Register	Memory

Allowed operands for XCHG instruction



5.1 Data-Transfer Instructions

EXAMPLE

What is the result of executing the following instruction?

XCHG [SUM], BX

Where $SUM = 1234_{16}$, $(DS) = 1200_{16}$

Solution:

$((DS)0 + SUM) \leftrightarrow (BX)$

$PA = 12000_{16} + 1234_{16} = 13234_{16}$

Execution of the instruction performs the following 16-bit swap:

$(13234_{16}) \leftrightarrow (BL)$

$(13235_{16}) \leftrightarrow (BH)$

So we get $(BX) = 00FF_{16}$

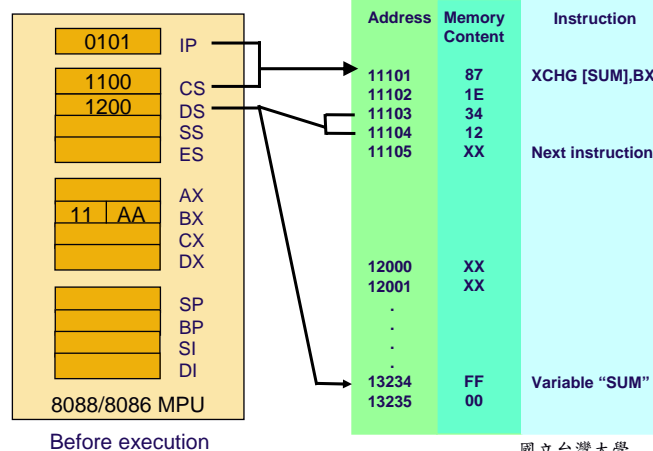
$(SUM) = 11AA_{16}$



5.1 Data-Transfer Instructions

■ The XCHG Instruction

XCHG [SUM], BX



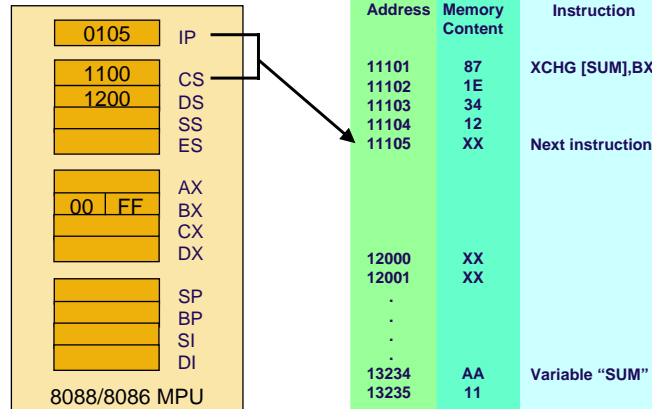
Before execution



5.1 Data-Transfer Instructions

■ The XCHG Instruction

XCHG [SUM], BX



After execution

611 37100 微處理機原理與應用 Lecture 05-13

國立台灣大學
生物機電系
林達德



5.1 Data-Transfer Instructions

EXAMPLE

Use the DEBUG program to verify the previous example.

Solution:

```

CONSOLE MODE - DEBUG
--R
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=0B35 ES=0B35 SS=0B35 CS=0B35 IP=0100 NV UP EI PL NZ NA PO NC
0B35:0100 8B0E2000 MOV CX,[0020] DS:0020=FFF
--A 1100:101
1100:0101 XCHG [1234], BX
1100:0105
--R BX
BX 0000
:11AA
--R DS
DS 0B35
:1200
--R CS
CS 0B35
:1100
--R IP
IP 0100
:101
--
    
```

611 37100 微處理機原理與應用 Lecture 05-14

國立台灣大學
生物機電系
林達德



5.1 Data-Transfer Instructions

EXAMPLE

Use the DEBUG program to verify the previous example.

Solution:

```

CONSOLE MODE
-R
AX=0000 BX=11AA CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=1200 ES=0B35 SS=0B35 CS=1100 IP=0101 NV UP EI PL NZ NA PO NC
1100:0101 871E3412 XCHG BX,[1234] DS:1234=11AA
-E 1234 FF 00
-U 101 104
1100:0101 871E3412 XCHG BX,[1234]
-T
AX=0000 BX=00FF CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=1200 ES=0B35 SS=0B35 CS=1100 IP=0105 NV UP EI PL NZ NA PO NC
1100:0105 0000 ADD [BX+SI],AL DS:00FF=00
-D 1234 1235
1200:1230 AA 11
-Q
C:\>
    
```



5.1 Data-Transfer Instructions

■ The XLAT Instruction

The translate (XLAT) instruction is used to simplify implementation of the lookup-table operation. Execution of the XLAT replaces the contents of AL by the contents of the accessed lookup-table location.

Mnemonic	Meaning	Format	Operation	Flags affected
XLAT	Translate	XLAT	$((AL)+(BX)+(DS)0) \rightarrow (AL)$	None

e.g.

$$\begin{aligned}
 PA &= (DS)0 + (BX) + (AL) \\
 &= 03000_{16} + 0100_{16} + 0D_{16} = 0310D_{16} \\
 (0310D_{16}) &\rightarrow (AL)
 \end{aligned}$$



5.1 Data-Transfer Instructions

■ The LEA, LDS, and LES Instructions

The LEA, LDS, LES instructions provide the ability to manipulate memory addresses by loading either a 16-bit offset address into a general-purpose register or a register together with a segment address into either DS or ES.

Mnemonic	Meaning	Format	Operation	Flags affected
LEA	Load effective address	LEA Reg16, EA	EA →(Reg16)	None
LDS	Load register and DS	LDS Reg16, Mem32	(Mem32) →(Reg16) (Mem32+2) →(DS)	None
LES	Load register and ES	LES Reg16, Mem32	(Mem32) →(Reg16) (Mem32+2) →(ES)	None

e.g. LEA SI, [DI+BX+5H]

611 37100 微處理機原理與應用 Lecture 05-17

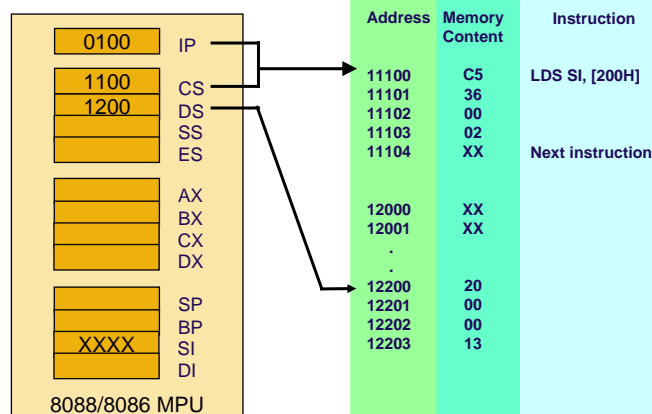
國立台灣大學
生物機電系
林達德



5.1 Data-Transfer Instructions

■ The LEA, LDS, and LES Instructions

LDS SI, [200H]



Before execution

611 37100 微處理機原理與應用 Lecture 05-18

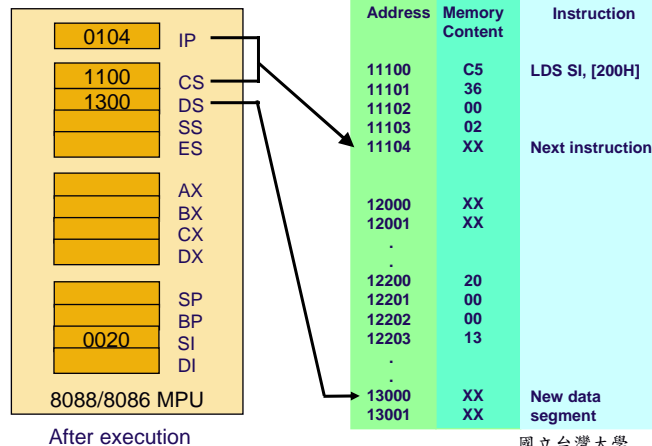
國立台灣大學
生物機電系
林達德



5.1 Data-Transfer Instructions

■ The LEA, LDS, and LES Instructions

LDS SI, [200H]



After execution

國立台灣大學
生物機電系
林達德



611 37100 微處理機原理與應用 Lecture 05-19

5.1 Data-Transfer Instructions

EXAMPLE

Verify the following instruction using DEBUG program.

LDS SI, [200H]

```

CONSOLE MODE - DEBUG
C:\>DEBUG
-R IP
IP 0100
.
-R CS
CS 0B37
:1100
-R DS
DS 0B37
:1200
-R SI
SI 0000
.
-A CS:100
1100:0100 LDS SI, [200]
1100:0104
-E 200 20 00 00 13
-T
AX=0000 BX=0000 CX=0000 DX=0000 SP=FFEE BP=0000 SI=0020 DI=0000
DS=1300 ES=0B37 SS=0B37 CS=1100 IP=0104 NV UP EI PL NZ NA PO NC
1100:0104 0000 ADD [BX+SI],AL DS:0020=00
    
```

國立台灣大學
生物機電系
林達德



611 37100 微處理機原理與應用 Lecture 05-20



5.1 Data-Transfer Instructions

EXAMPLE

Initializing the internal registers of the 8088 from a table in memory.

Solution:

```
MOV AX, [INIT_TABLE]
MOV SS, AX
LDS SI, [INIT_TABLE+02H]
LES DI, [INIT_TABLE+06H]
MOV AX, [INIT_TABLE+0AH]
MOV BX, [INIT_TABLE+0CH]
MOV CX, [INIT_TABLE+0EH]
MOV DX, [INIT_TABLE+10H]
```

611 37100 微處理機原理與應用 Lecture 05-21

國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

■ The arithmetic instructions include

- ❖ Addition
- ❖ Subtraction
- ❖ Multiplication
- ❖ Division

■ Data formats

- ❖ Unsigned binary bytes
- ❖ Signed binary bytes
- ❖ Unsigned binary words
- ❖ Signed binary words
- ❖ Unpacked decimal bytes
- ❖ Packed decimal bytes
- ❖ ASCII numbers

611 37100 微處理機原理與應用 Lecture 05-22

國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

ADDITION	
ADD	Add byte or word
ADC	Add byte or word with carry
INC	Increment byte or word by 1
AAA	ASCII adjust for addition
DAA	Decimal adjust for addition
SUBTRACTION	
SUB	Subtract byte or word
SBB	Subtract byte or word with borrow
DEC	Decrement byte or word by 1
NEG	Negate byte or word
AAS	ASCII adjust for subtraction
DAS	Decimal adjust for subtraction
MULTIPLICATION	
MUL	Multiply byte or word unsigned
IMUL	Integer multiply byte or word
AAM	ASCII adjust for multiply
DIVISION	
DIV	Divide byte or word unsigned
IDIV	Integer divide byte or word
AAD	ASCII adjust for division
CBW	Convert byte to word
CWD	Convert word to doubleword

611 37100 微處理機原理與應用 Lecture 05-23

國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

■ Addition Instructions: ADD, ADC, INC, AAA, DAA

Mnemonic	Meaning	Format	Operation	Flags affected
ADD	Addition	ADD D, S	$(S) + (D) \rightarrow (D)$ Carry $\rightarrow (CF)$	OF, SF, ZF, AF, PF, CF
ADC	Add with carry	ADC D, S	$(S) + (D) + (CF) \rightarrow (D)$ Carry $\rightarrow (CF)$	OF, SF, ZF, AF, PF, CF
INC	Increment by 1	INC D	$(D) + 1 \rightarrow (D)$	OF, SF, ZF, AF, PF
AAA	ASCII adjust for addition	AAA		AF, CF OF, SF, ZF, PF undefined
DAA	Decimal adjust for addition	DAA		SF, ZF, AF, PF, CF, OF undefined

611 37100 微處理機原理與應用 Lecture 05-24

國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

- Addition Instructions: ADD, ADC, INC, AAA, DAA

Destination	Source
Register	Register
Register	Memory
Memory	Register
Register	Immediate
Memory	Immediate
Accumulator	Immediate

Allowed operands for ADD and ADC instructions

Destination
Reg16
Reg8
Memory

Allowed operands for INC instruction

611 37100 微處理機原理與應用 Lecture 05-25

國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

EXAMPLE

Assume that the AX and BX registers contain 1100_{16} and $0ABC_{16}$, respectively. What is the result of executing the instruction ADD AX, BX?

Solution:

$$(BX)+(AX)= 0ABC_{16} + 1100_{16}=1BBC_{16}$$

The sum ends up in destination register AX. That is

$$(AX) = 1BBC_{16}$$

611 37100 微處理機原理與應用 Lecture 05-26

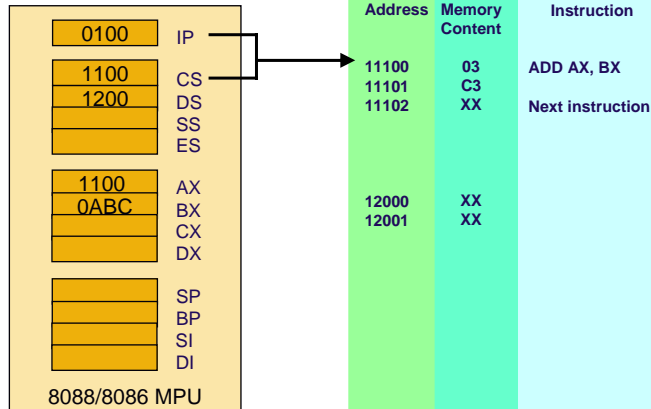
國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

- Addition Instructions: ADD, ADC, INC, AAA, DAA

ADD AX, BX



Before execution

611 37100 微處理機原理與應用 Lecture 05-27

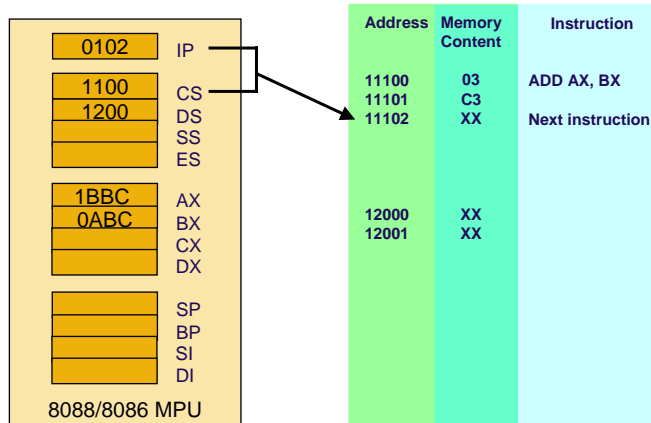
國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

- Addition Instructions: ADD, ADC, INC, AAA, DAA

ADD AX, BX



After execution

611 37100 微處理機原理與應用 Lecture 05-28

國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

EXAMPLE

Verify the previous example using DEBUG program.

Solution:

```
CONSOLE MODE - DEBUG
-A 1100:0100
1100:0100 ADD AX, BX
1100:0102
-R AX
AX 0000
:1100
-R BX
BX 0000
:0ABC
-U 1100:100 100
1100:0100 01D8 ADD AX, BX
-T =1100:100
AX=1B8C BX=0ABC CX=0000 DX=0000 SP=FFEE BP=0000 SI=0020 DI=0000
DS=1300 ES=0837 SS=0B37 CS=1100 IP=0102 NV UP EI PL NZ NA PO NC
1100:0102 0002 ADD [BP+SI], AL SS:0020=FF
```



5.2 Arithmetic Instructions

EXAMPLE

The original contents of AX, BL, word-size memory location SUM, and carry flag (CF) are 1234_{16} , AB_{16} , $00CD_{16}$, and 0_{16} , respectively. Describe the results of executing the following sequence of instruction?

```
ADD AX, [SUM]
ADC BL, 05H
INC WORD PTR [SUM]
```

Solution:

$$(AX) \leftarrow (AX) + (SUM) = 1234_{16} + 00CD_{16} = 1301_{16}$$

$$(BL) \leftarrow (BL) + \text{imm8} + (CF) = AB_{16} + 5_{16} + 0_{16} = B0_{16}$$

$$(SUM) \leftarrow (SUM) + 1_{16} = 00CD_{16} + 1_{16} = 00CE_{16}$$



5.2 Arithmetic Instructions

EXAMPLE

What is the result of executing the following instruction sequence?

```
ADD AL, BL
```

```
AAA
```

Assuming that AL contains 32_{16} (ASCII code for 2) and BL contains 34_{16} (ASCII code 4), and that AH has been cleared.

Solution:

$$(AL) \leftarrow (AL) + (BL) = 32_{16} + 34_{16} = 66_{16}$$

The result after the AAA instruction is

$$(AL) = 06_{16}$$

$$(AH) = 00_{16}$$

with both AF and CF remain cleared



5.2 Arithmetic Instructions

EXAMPLE

Perform a 32-bit binary add operation on the contents of the processor's register.

Solution:

$$(DX, CX) \leftarrow (DX, CX) + (BX, AX)$$

$$(DX, CX) = FEDCBA98_{16}$$

$$(BX, AX) = 01234567_{16}$$

```
MOV DX, 0FEDCH
```

```
MOV CX, 0BA98H
```

```
MOV BX, 01234H
```

```
MOV AX, 04567H
```

```
ADD CX, AX
```

```
ADC DX, BX
```

; Add with carry





5.2 Arithmetic Instructions

- Subtraction Instructions:
SUB, SBB, DEC, AAS, DAS, and NEG

Mnemonic	Meaning	Format	Operation	Flags affected
SUB	Subtract	SUB D, S	(D)-(S)→(D) Borrow →(CF)	OF, SF, ZF, AF, PF, CF
SBB	Subtract with borrow	SBB D, S	(D)-(S)-(CF)→(D)	OF, SF, ZF, AF, PF, CF
DEC	Decrement by 1	DEC D	(D)-1→(D)	OF, SF, ZF, AF, PF
NEG	Negate	NEG D	0-(D) →(D) 1 →(CF)	OF, SF, ZF, AF, PF, CF
DAS	Decimal adjust for subtraction	DAS		SF, ZF, AF, PF, CF, OF undefined
AAS	ASCII adjust for subtraction	AAS		AF, CF OF, SF, ZF, PF undefined

611 37100 微處理機原理與應用 Lecture 05-33

國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

- Subtraction Instructions:
SUB, SBB, DEC, AAS, DAS, and NEG

Destination	Source
Register	Register
Register	Memory
Memory	Register
Register	Immediate
Memory	Immediate
Accumulator	Immediate

Allowed operands for SUB and SBB instructions

Destination
Reg16
Reg8
Memory

Allowed operands for DEC instruction

Destination
Register
Memory

Allowed operands for NEG instruction

611 37100 微處理機原理與應用 Lecture 05-34

國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

EXAMPLE

Assuming that the contents of register BX and CX are 1234_{16} and 0123_{16} , respectively, and the carry flag is 0, what is the result of executing the instruction SBB BX, CX?

Solution:

$$(BX)-(CX)-(CF) \rightarrow (BX)$$

We get

$$\begin{aligned}(BX) &= 1234_{16} - 0123_{16} - 0_{16} \\ &= 1111_{16}\end{aligned}$$

the carry flag remains cleared.



5.2 Arithmetic Instructions

EXAMPLE

Verify the previous example using DEBUG program.

Solution:

```
CONSOLE MODE - DEBUG
-R BX
BX 0000
:1234
-R CX
CX 0000
:0123
-R F
NV UP EI PL NZ NA PO NC -
+A
0B37:0100 SBB BX,CX
0B37:0102
-R
AX=0000 BX=1234 CX=0123 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=0B37 ES=0B37 SS=0B37 CS=0B37 IP=0100 NV UP EI PL NZ NA PO NC
0B37:0100 19CB SBB BX,CX
-T
AX=0000 BX=1111 CX=0123 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=0B37 ES=0B37 SS=0B37 CS=0B37 IP=0102 NV UP EI PL NZ NA PO NC
0B37:0102 01ACEB78 ADD [SI+78EB],BP DS:78EB=0000
```





5.2 Arithmetic Instructions

EXAMPLE

Assuming that the register BX contains $003A_{16}$, what is the result of executing the following instruction?

NEG BX

Solution:

$$\begin{aligned} (BX) &= 0000_{16} - (BX) = 0000_{16} + 2\text{'s complement of } 003A_{16} \\ &= 0000_{16} + \text{FFC6}_{16} \\ &= \text{FFC6}_{16} \end{aligned}$$

Since no carry is generated in this add operation, the carry flag is complemented to give

$$(CF) = 1$$

611 37100 微處理機原理與應用 Lecture 05-37

國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

EXAMPLE

Verify the previous example using DEBUG program.

Solution:

```
CONSOLE MODE
-R BX
BX 0000
: 3A
-A
0B37:0100 NEG BX
0B37:0102
-R BX
BX 003A
-U 100 101
0B37:0100 F7DB NEG BX
-T
AX=0000 BX=FFC6 CX=0000 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=0B37 ES=0B37 SS=0B37 CS=0B37 IP=0102 NV UP EI NG NZ AC PE CY
0B37:0102 01ACEB78 ADD [SI+78EB],BP DS:78EB=0000
-Q
```

611 37100 微處理機原理與應用 Lecture 05-38

國立台灣大學
生物機電系
林達德



5.2 Arithmetic Instructions

EXAMPLE

Perform a 32-bit binary subtraction for variable X and Y.

Solution:

```

MOV SI, 200H      ; Initialize pointer for X
MOV DI, 100H      ; Initialize pointer for Y
MOV AX, [SI]      ; Subtract LS words
SUB AX, [DI]
MOV [SI], AX      ; Save the LS word of result
MOV AX, [SI]+2    ; Subtract MS words
SBB AX, [DI]+2
MOV [SI]+2, AX    ; Save the MS word of result
    
```



5.2 Arithmetic Instructions

■ Multiplication Instructions:

MUL, DIV, IMUL, IDIV, AAM, AAD, CBW, and CWD

Mnemonic	Meaning	Format	Operation	Flags affected
MUL	Multiply (unsigned)	MUL S	(AL)·(S8)→(AX) (AX)·(S16)→(DX)(AX)	OF, CF SF, ZF, AF, PF undefined
DIV	Division (unsigned)	DIV S	(1)Q((AX)/(S8))→(AL) R((AX)/(S8))→(AH) (2)Q((DX,AX)/(S16))→(AX) R((DX,AX)/(S16))→(DX) If Q is FF ₁₆ in case (1) or FFFF ₁₆ in case (2), then type 0 interrupt occurs	OF, SF, ZF, AF, PF, CF undefined
IMUL	Integer multiply (signed)	IMUL S	(AL)·(S8)→(AX) (AX)·(S16)→(DX)(AX)	OF, CF SF, ZF, AF, PF undefined
IDIV	Integer divide (signed)	IDIV S	(1)Q((AX)/(S8))→(AL) R((AX)/(S8))→(AH) (2)Q((DX,AX)/(S16))→(AX) R((DX,AX)/(S16))→(DX) If Q is positive and exceeds 7FFF ₁₆ or if Q is negative and become less than 8001 ₁₆ , then type 0 interrupt occurs	OF, SF, ZF, AF, PF, CF undefined



5.2 Arithmetic Instructions

■ Multiplication Instructions:

MUL, DIV, IMUL, IDIV, AAM, AAD, CBW, and CWD

Mnemonic	Meaning	Format	Operation	Flags affected
AAM	Adjust AL for multiplication	AAM	Q((AL)/10)→(AH) R((AL)/10)→(AL)	SF,ZF,PF OF,AF,CF undefined
AAD	Adjust AX for division	AAD	(AH)·10+(AL)→(AL) 00→(AH)	SF,ZF,PF OF,AF,CF undefined
CBW	Convert byte to word	CBW	(MSB of AL)→(All bits of AH)	None
CWD	Convert word to double word	CWD	(MSB of AX)→(All bits of DX)	None

Destination
Reg8
Reg16
Mem8
Mem16

Allowed operands

國立台灣大學
生物機電系
林達德



611 37100 微處理機原理與應用 Lecture 05-41

5.2 Arithmetic Instructions

EXAMPLE

The 2's-complement signed data contents of AL are -1 and that of CL are -2. What result is produced in AX by executing the following instruction?

MUL CL and **IMUL CL**

Solution:

$$(AL) = -1 \text{ (as 2's complement)} = 11111111_2 = FF_{16}$$

$$(CL) = -2 \text{ (as 2's complement)} = 11111110_2 = FE_{16}$$

Executing the MUL instruction gives

$$(AX) = 11111111_2 \times 11111110_2 = 1111110100000010_2 = FD02_{16}$$

Executing the IMUL instruction gives

$$(AX) = -1_{16} \times -2_{16} = 2_{16} = 0002_{16}$$

國立台灣大學
生物機電系
林達德



611 37100 微處理機原理與應用 Lecture 05-42



5.2 Arithmetic Instructions

EXAMPLE

Verify the previous example using DEBUG program.

Solution:

```

CONSOLE MODE - DEBUG
-
-R AX
AX 0000
:FF
-R CX
CX 0000
:FE
-A
0B35:0100 MUL CL
0B35:0102
-R AX
AX 00FF
:
-R CX
CX 00FE
:
-T
AX=FD02 BX=0000 CX=00FE DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=0B35 ES=0B35 SS=0B35 CS=0B35 IP=0102 OV UP EI PL NZ NA PO CY
0B35:0102 1850A0 SBB [BX+SI-60],DL DS:FFA0=00

```



5.2 Arithmetic Instructions

EXAMPLE

What is the result of executing the following instructions?

MOV AL, 0A1H

CBW

CWD

Solution:

$$(AL) = A1_{16} = 10100001_2$$

Executing the CBW instruction extends the MSB of AL

$$(AH) = 11111111_2 = FF_{16} \quad \text{or} \quad (AX) = 1111111110100001_2$$

Executing the CWD instruction, we get

$$(DX) = 1111111111111111_2 = FFFF_{16}$$

That is, $(AX) = FFA1_{16}$ $(DX) = FFFF_{16}$





5.3 Logic Instructions

■ The logic instructions include

- ❖ AND
- ❖ OR
- ❖ XOR (Exclusive-OR)
- ❖ NOT

Mnemonic	Meaning	Format	Operation	Flags affected
AND	Logical AND	AND D, S	$(S) \cdot (D) \rightarrow (D)$	OF, SF, ZF, PF, CF AF undefined
OR	Logical Inclusive-OR	OR D, S	$(S) \vee (D) \rightarrow (D)$	OF, SF, ZF, PF, CF AF undefined
XOR	Logical exclusive-OR	XOR D, S	$(S) \oplus (D) \rightarrow (D)$	OF, SF, ZF, PF, CF AF undefined
NOT	Logical NOT	NOT D	$(\text{NOT } D) \rightarrow (D)$	None



5.3 Logic Instructions

■ Logic instructions : AND, OR, XOR, NOT

Destination	Source
Register	Register
Register	Memory
Memory	Register
Register	Immediate
Memory	Immediate
Accumulator	Immediate

Allowed operands for AND, OR, and XOR instructions

Destination
Register
Memory

Allowed operands for NOT instruction



5.3 Logic Instructions

EXAMPLE

Describe the results of executing the following instructions?

```
MOV AL, 01010101B
AND AL, 00011111B
OR AL, 11000000B
XOR AL, 00001111B
NOT AL
```

Solution:

$$(AL) = 01010101_2 \cdot 00011111_2 = 00010101_2 = 15_{16}$$

Executing the OR instruction, we get

$$(AL) = 00010101_2 + 11000000_2 = 11010101_2 = D5_{16}$$

Executing the XOR instruction, we get

$$(AL) = 11010101_2 \oplus 00001111_2 = 11011010_2 = DA_{16}$$

Executing the NOT instruction, we get

$$(AL) = (\text{NOT})11011010_2 = 00100101_2 = 25_{16}$$

611 37100 微處理機原理與應用 Lecture 05-47

國立台灣大學
生物機電系
林達德



5.3 Logic Instructions

EXAMPLE

Masking and setting bits in a register.

Solution:

Mask off the upper 12 bits of the word of data in AX

```
AND AX, 000F16
```

Setting B₄ of the byte at the offset address CONTROL_FLAGS

```
MOV AL, [CONTROL_FLAGS]
```

```
OR AL, 10H
```

```
MOV [CONTROL_FLAGS], AL
```

Executing the above instructions, we get

$$(AL) = \text{XXXXXXXX}_2 + 00010000_2 = \text{XXX1XXXX}_2$$

611 37100 微處理機原理與應用 Lecture 05-48

國立台灣大學
生物機電系
林達德





5.4 Shift Instructions

■ Shift instructions: SHL, SHR, SAL, SAR

Mnemonic	Meaning	Format	Operation	Flags affected
SAL/SHL	Shift arithmetic left / Shift logical left	SAL D, Count SHL D, Count	Shift the (D) left by the number of bit positions equal to Count and fill the vacated bits positions on the right with zeros	CF, PF, SF, Z AF undefined OF undefined if count ≠1
SHR	Shift logical right	SHR D, Count	Shift the (D) right by the number of bit positions equal to Count and fill the vacated bits positions on the left with zeros	CF, PF, SF, Z AF undefined OF undefined if count ≠1
SAR	Shift arithmetic right	SAR D, Count	Shift the (D) right by the number of bit positions equal to Count and fill the vacated bits positions on the left with the original most significant bits	CF, PF, SF, Z AF undefined OF undefined if count ≠1

611 37100 微處理機原理與應用 Lecture 05-49

國立台灣大學
生物機電系
林達德



5.4 Shift Instructions

■ Shift instructions: SHL, SHR, SAL, SAR

Destination	Count
Register	1
Register	CL
Memory	1
Memory	CL

Allowed operands for shift instructions

611 37100 微處理機原理與應用 Lecture 05-50

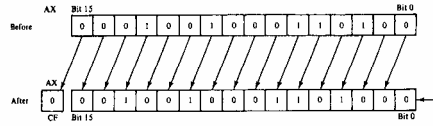
國立台灣大學
生物機電系
林達德



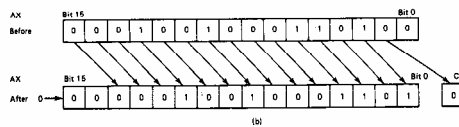


5.4 Shift Instructions

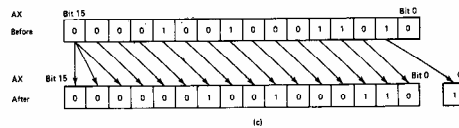
■ Shift instructions: SHL, SHR, SAL, SAR



SHL AX, 1



SHR AX, CL



SAR AX, CL

Figure 5-27 Results of executing SHL AX,1. (b) Results of executing SHR AX,CL. (c) Results of executing SAR AX,CL.



5.4 Shift Instructions

EXAMPLE

Assume that CL contains 02_{16} and AX contains $091A_{16}$. Determine the new contents of AX and the carry flag after the instruction SAR AX, CL is executed.

Solution:

$$(AX) = 0000001001000110_2 = 0246_{16}$$

and the carry flag is $(CF) = 1_2$





5.4 Shift Instructions

EXAMPLE

Verify the previous example using DEBUG program.

Solution:

```
CONSOLE MODE - DEBUG
-
-A
0B35:0104 SAR AX, CL
0B35:0106
-R AX
AX FD02
:091A
-R CX
CX 00FE
:2
-R F
OV UP EI PL NZ NA PO CY -
-T
AX=0246 BX=0000 CX=0002 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=0B35 ES=0B35 SS=0B35 CS=0B35 IP=0104 NV UP EI PL NZ NA PO CY
0B35:0104 D3F8 SAR AX,CL
```



5.4 Shift Instructions

EXAMPLE

Isolate the bit B_3 of the byte at the offset address CONTROL_FLAGS.

Solution:

```
MOV AL, [CONTROL_FLAGS]
MOV CL, 04H
SHR AL, CL
```

Executing the instructions, we get

$$(AL) = 0000B_7B_6B_5B_4$$

and

$$(CF) = B_3$$





5.5 Rotate Instructions

■ Rotate instructions: ROL, ROR, RCL, RCR

Mnemonic	Meaning	Format	Operation	Flags affected
ROL	Rotate left	ROL D, Count	Rotate the (D) left by the number of bit positions equal to Count. Each bit shifted out from the leftmost bit goes back into the rightmost bit position.	CF OF undefined if count $\neq 1$
ROR	Rotate right	ROR D, Count	Rotate the (D) right by the number of bit positions equal to Count. Each bit shifted out from the rightmost bit goes back into the leftmost bit position.	CF OF undefined if count $\neq 1$
RCL	Rotate left through carry	RCL D, Count	Same as ROL except carry is attached to (D) for rotation.	CF OF undefined if count $\neq 1$
RCR	Rotate right through carry	RCR D, Count	Same as ROR except carry is attached to (D) for rotation.	CF OF undefined if count $\neq 1$

611 37100 微處理機原理與應用 Lecture 05-55

國立台灣大學
生物機電系
林達德



5.5 Rotate Instructions

■ Rotate instructions: ROL, ROR, RCL, RCR

Destination	Count
Register	1
Register	CL
Memory	1
Memory	CL

Allowed operands for rotate instructions

611 37100 微處理機原理與應用 Lecture 05-56

國立台灣大學
生物機電系
林達德





5.5 Rotate Instructions

■ Rotate instructions: ROL, ROR, RCL, RCR

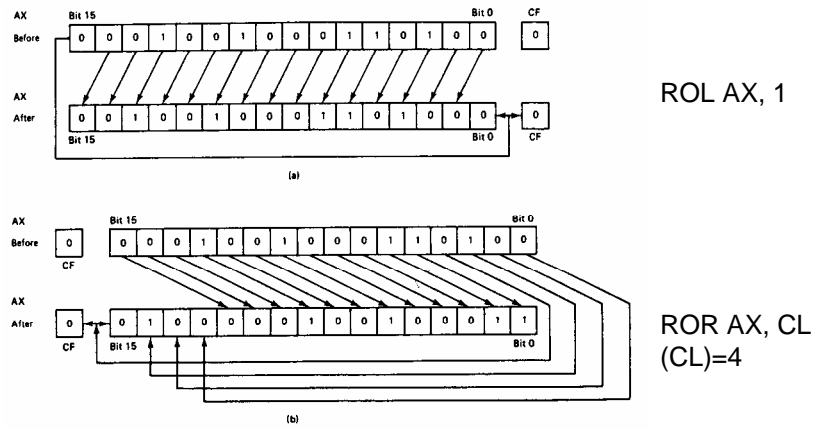


Figure 5-30 Results of executing ROL AX,1. (b) Results of executing ROR AX,CL with (CL)=4.



5.5 Rotate Instructions

■ Rotate instructions: ROL, ROR, RCL, RCR

For RCL, RCR, the bits are rotate through the carry flag

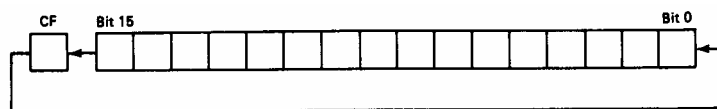


Figure 5-31 Rotation caused by execution of the RCL instruction.



5.5 Rotate Instructions

EXAMPLE

What is the result in BX and CF after execution of the following instructions?

RCR BX, CL

Assume that, prior to execution of the instruction, (CL)=04₁₆, (BX)=1234₁₆, and (CF)=0

Solution:

The original contents of BX are

$$(BX) = 0001001000110100_2 = 1234_{16}$$

Execution of the RCR command causes a 4-bit rotate right through carry to take place on the data in BX, the results are

$$(BX) = 1000000100100011_2 = 8123_{16}$$

$$(CF) = 0_2$$

611 37100 微處理機原理與應用 Lecture 05-59

國立台灣大學
生物機電系
林達德



5.5 Rotate Instructions

EXAMPLE

Verify the previous example using DEBUG program.

Solution:

```
CONSOLE MODE - DEBUG
-A
0B35:0100 RCR BX,CL
0B35:0102
-R BX
BX 0000
:1234
-R CX
CX 0000
:4
-R F
NV UP EI PL NZ NA PO NC -
-T
AX=0000 BX=8123 CX=0004 DX=0000 SP=FFEE BP=0000 SI=0000 DI=0000
DS=0B35 ES=0B35 SS=0B35 CS=0B35 IP=0102 OV UP EI PL NZ NA PO NC
0B35:0102 D3F8 SAR AX,CL
```

611 37100 微處理機原理與應用 Lecture 05-60

國立台灣大學
生物機電系
林達德





5.5 Rotate Instructions

EXAMPLE

Disassembly and addition of 2 hexadecimal digits stored as a byte in memory.

Solution:

```
MOV AL, [HEX_DIGITS]
MOV BL, AL
MOV CL, 04H
ROR BL, CL
AND AL, 0FH
AND BL, 0FH
ADD AL, BL
```

