Assembly Language Lecture Series:

x86-64 Addressing Modes

Sensei RL Uy, College of Computer Studies,
De La Salle University, Manila, Philippines

Copyright Notice

Talking Points

01 Addressing Modes

02 Register Addressing Mode

03 Immediate Addressing Mode

04 Memory Addressing Mode

Recall: Software Architecture (Memory)

```
section .data
varx db 0x12, 0x34, 0x56, 0x78, 0x9A, 0xBC, 0xDE, 0xF0

section .text
        mov rax, [varx]
        inc rax
```

Separate memory for data (known as **data segment**) and instruction (known as **code segment**).

Register RIP points to the next instruction to be executed

Data Memory

Label	Address	Data
	403017	F0
	403016	DE
	403015	BC
	403014	9A
	403013	78
	403012	56
	403011	34
varx	403010	12

Instruction Memory

Address	Data
	Etc.
4014E5	30
4014E4	10
4014E3	25
4014E2	04
4014E1	8B
4014E0	48

RIP	00	40	14	E0

Addressing Modes

Addressing modes: where the operands are located.

- **Three (3) locations:**
  - Internal registers (**Register addressing mode**)
  - Part of the instruction (**Immediate addressing mode**)
  - Memory (**Memory addressing mode**)

Register Addressing Mode

- The operand to be accessed is specified as **residing in the register**.
- Registers are located inside the CPU.
- **Syntax: register_name**
- Processing of data is **faster** in this addressing mode since there is no need to fetch the data externally (i.e., from the memory).

Register Addressing Mode

Example

MOV RAX, RBX

Q: What will RAX contain after execution?

Q: What will RBX contain after execution?

Initial Value

Register	Value
RAX	2401_FFFF_1234_ABCD
RBX	7FFF_FFFF_FFFF_FFFF
RCX	0000 0000 0000 0000
RDX	1234 5678 ABCD EF01

Register Addressing Mode

Example

MOV RAX, RBX

Q: What will RAX contain after execution?

A: 7FFFFFFFFFFFFFFF

Q: What will RBX contain after execution?

A: 7FFFFFFFFFFFFFFF

Initial Value

Register	Value
RAX	2401_FFFF_1234_ABCD
RBX	7FFF_FFFF_FFFF_FFFF
RCX	0000 0000 0000 0000
RDX	1234 5678 ABCD EF01

Register Addressing Mode

Example

MOV AH, DL

Q: What will RAX contain after execution?

Initial Value

Register	Value
RAX	2401_FFFF_1234_ABCD
RBX	7FFF_FFFF_FFFF_FFFF
RCX	0000 0000 0000 0000
RDX	1234 5678 ABCD EF01

Register Addressing Mode

Example

MOV AH, DL

Q: What will RAX contain after execution?

A: 2401FFFF123401CD

Initial Value

Register	Value
RAX	2401_FFFF_1234_ABCD
RBX	7FFF_FFFF_FFFF_FFFF
RCX	0000 0000 0000 0000
RDX	1234 5678 ABCD EF01

Register Addressing Mode

Example

MOV AL, SI

Q: What will RAX contain after execution?

Initial Value

Register	Value
RAX	2401_FFFF_1234_ABCD
RBX	7FFF_FFFF_FFFF_FFFF
RCX	0000 0000 0000 0000
RDX	1234 5678 ABCD EF01

Register Addressing Mode

Example

MOV AL, SI

Q: What will RAX contain after execution?

A: Syntax Error

Rule #1: No data size mismatch among operands.

Initial Value

Register	Value
RAX	2401_FFFF_1234_ABCD
RBX	7FFF_FFFF_FFFF_FFFF
RCX	0000 0000 0000 0000
RDX	1234 5678 ABCD EF01

Immediate Addressing Mode

- The operand is part of the instruction
- The **operand** is a **constant**
- The constant value depends on the **instruction** and the **destination operand**
  - **MOV** instruction can support (up to) 64-bit immediate operand
    MOV RAX 0x123456789ABCDEF1
  - Except if destination operand is **mem64**, immediate operand is limited to sign-extended 32-bit
    MOV qword [var1], 0x0000000012345678
  - **ADD** instruction can support (up to) 32-bit sign-extended immediate operand
    ADD RAX 0x0000000012345678
- **Syntax: constant value**

Immediate Addressing Mode

Example

MOV RAX, 0x123456789ABCDEF0

Q: What will RAX contain after execution?

Initial Value

Register	Value
RAX	2401_FFFF_1234_ABCD
RBX	7FFF_FFFF_FFFF_FFFF
RCX	0000 0000 0000 0000
RDX	1234 5678 ABCD EF01

Instruction Memory

Address	Data
4014E9	12
4014E8	34
4014E7	56
4014E6	78
4014E5	9A
4014E4	BC
4014E3	DE
4014E2	F0
4014E1	B8
4014E0	48

Operand is part of the instruction

Immediate Addressing Mode

Example

MOV RAX, 0x123456789ABCDEF0

Q: What will RAX contain after execution?

A: 123456789ABCDEF0

Initial Value

Register	Value
RAX	2401_FFFF_1234_ABCD
RBX	7FFF_FFFF_FFFF_FFFF
RCX	0000 0000 0000 0000
RDX	1234 5678 ABCD EF01

Instruction Memory

Address	Data
4014E9	12
4014E8	34
4014E7	56
4014E6	78
4014E5	9A
4014E4	BC
4014E3	DE
4014E2	F0
4014E1	B8
4014E0	48

Operand is part of the instruction

Immediate Addressing Mode

Example

MOV 123456789ABCDEF0, RAX

Q: What will RAX contain after execution?

Initial Value

Register	Value
RAX	2401_FFFF_1234_ABCD
RBX	7FFF_FFFF_FFFF_FFFF
RCX	0000 0000 0000 0000
RDX	1234 5678 ABCD EF01

Instruction Memory

Address	Data
4014E9	12
4014E8	34
4014E7	56
4014E6	78
4014E5	9A
4014E4	BC
4014E3	DE
4014E2	F0
4014E1	B8
4014E0	48

Operand is part of the instruction

Immediate Addressing Mode

Example

MOV RAX, 0x123456789ABCDEF0

Q: What will RAX contain after execution?

A: Syntax Error

Rule #2: Immediate addressing mode could only be used as source operand

Instruction Memory

Address	Data
4014E9	12
4014E8	34
4014E7	56
4014E6	78
4014E5	9A
4014E4	BC
4014E3	DE
4014E2	F0
4014E1	B8
4014E0	48

Operand is part of the instruction

Memory Addressing Mode

- The operand is in the **memory**.
- Needs to be fetched
- The offset part of a memory address can be specified as an address computation made up of the following components:
  - **Displacement**: an 8-,16-,32- or 64-bit value (can also be a variable name)
  - **Base**: the value in a general-purpose register
  - **Index**: the value in a general-purpose register
  - **Scale factor**: a value of 1, 2,4, or 8 that is multiplied by the index value

Memory Addressing Mode (32-bit mode)

$$\begin{bmatrix} EAX \\ EBX \\ ECX \\ EDX \\ ESI \\ EDI \\ ESP \\ EBP \end{bmatrix} + \left[\begin{pmatrix} EAX \\ EBX \\ ECX \\ EDX \\ ESI \\ EDI \\ EBP \end{pmatrix} * \begin{pmatrix} 1 \\ 2 \\ 4 \end{pmatrix} \right] + \begin{bmatrix} 8-bit \\ 16-bit \\ 32-bit \end{bmatrix}$$

Base Index Scale Displacement (can be variable)

Memory Addressing Mode (64-bit mode)

$$\begin{bmatrix} RAX \\ RBX \\ RCX \\ RDX \\ RSI \\ RDI \\ RSP \\ RBP \end{bmatrix} + \left[\begin{pmatrix} RAX \\ RBX \\ RCX \\ RDX \\ RSI \\ RDI \\ RBP \end{pmatrix} * \begin{pmatrix} 1 \\ 2 \\ 4 \\ 8 \end{pmatrix} \right] + \begin{bmatrix} 8-bit \\ 16-bit \\ 32-bit \\ 64-bit \end{bmatrix}$$

Base Index Scale Displacement (can be variable)

Memory Addressing Mode
(displacement/absolute)

```
section .data
var1 dq 0x123456789ABCDEF0
section .text
mov rbx, [0x403010]
; not recommended as the address is not known during compile time
```

```
section .data
var1 dq 0x123456789ABCDEF0
section .text
mov rbx, [var1]
; correct version!
```

Data Memory

Label	Address	Data
	403017	12
	403016	34
	403015	56
	403014	78
	403013	9A
	403012	BC
	403011	DE
varx	403010	F0

After execution:

RBX = 0x123456789ABCDEF0

Memory Addressing Mode
(displacement/absolute)

```
section .data
var1 dq 0x123456789ABCDEF0
section .text
lea rsi, [var1]
mov rbx, [rsi]
```

RSI →

Label	Address	Data
	403017	12
	403016	34
	403015	56
	403014	78
	403013	9A
	403012	BC
	403011	DE
varx	403010	F0

After execution:

RBX = 0x123456789ABCDEF0

Memory Addressing Mode
(based + displacement)

```
section .data
var1 dd 0x12345678
var2 dd 0xABCDEF10
section .text
mov rdx, 0x0000000000000001
mov bl, [var1+rdx];var1 is the base while rdx is the index
```

Note: Use as one-dimensional array

Label	Address	Data
	403017	AB
	403016	CD
	403015	EF
var2	403014	10
	403013	12
	403012	34
	403011	56
var1	403010	78

After execution:

BL = 0x56

Memory Addressing Mode
(based + index + displacement)

```
section .data
var1 dd 0x12345678
var2 dd 0xABCDEF10
section .text
mov rsi, 0x0000000000000001
mov rdi, 0x0000000000000003
mov ebx, [var1+rsi+rdi];var1 is the base while
rsi and rdi are the index
```

Note: Use as two-dimensional array

Label	Address	Data
	403017	AB
	403016	CD
	403015	EF
var2	403014	10
	403013	12
	403012	34
	403011	56
var1	403010	78

After execution:
EBX = 0xABCDEF10

Memory Addressing Mode
(scaled indexed byte)

```
section .data
var1 dd 0x12345678
var2 dd 0xABCDEF10
section .text
mov rdx, 0x0000000000000001
mov ebx, [var1+rdx*4]; var1 is the base while rdx is the index
```

Label	Address	Data
	403017	AB
	403016	CD
	403015	EF
var2	403014	10
	403013	12
	403012	34
	403011	56
var1	403010	78

Scaled indexed byte (SIB) addressing mode is viewed as a one-dimensional array with adjust to the data type size of a variable (e.g.,1-char, 2-short, 4-int, 8-long long int)

After execution:
EBX = 0xABCDEF10

Memory Addressing Mode

Rule#3: Memory to Memory Not Allowed!

MOV [var1], [var2]; syntax error

Label	Address	Data
	403017	AB
	403016	CD
	403015	EF
var2	403014	10
	403013	12
	403012	34
	403011	56
var1	403010	78

Memory Addressing Mode

Rule#4: If the operand is a combination of memory addressing mode and immediate addressing mode, the prefix byte, word, dword, qword should be used

Note: qword is up to a 32-bit sign-extended integer

Examples:

MOV byte [var1], 0x7F

MOV word [var1], 0x7FFF

MOV dword [var1], 0x7FFFFFFF

MOV qword [var1], 0x000000007FFFFFFF

INC byte [var1]

Label	Address	Data
	403017	AB
	403016	CD
	403015	EF
	403014	10
	403013	12
	403012	34
	403011	56
var1	403010	78

Memory Addressing Mode (Examples)

```
section .data
var1 db 0x12
var2 db 0x34
```

1.) Initialize memory var1 with a value of 5

- `MOV byte [var1], 0x05`

2.) Increment the value of memory var1 by one.

- `INC byte [var1]`

3.) Copy the value of var1 to var2.

- `MOV AL, [var1]`
- `MOV [var2], AL`

4.) Add the value of 5 to memory var1.

- `ADD byte [var1], 0x05`

5.) Add register AL with memory var1 and store the result in memory var1.

- `ADD [var1], AL`

6.) Add register AL with memory var1 and store the result in register AL

- `ADD AL, [var1]`

Exercises

Register	Value
RAX	0000 0000 2401 FFFF
RBX	0000 0000 0000 0002
RCX	0000 0000 0000 0004
RDX	1234 5678 0000 0000
RSI	0000 0000 0000 0002
RDI	0000 0000 0000 0004

1. MOV RBX, [ALPHA]
2. MOV RCX, [ALPHA+04]
3. MOV RAX, 0x1234567887654321
4. MOV AL, [ALPHA+RSI+RDI]
5. MOV BL,[ALPHA+RSI*2]
6. MOV DL, [ALPHA+RBX+RSI*4]
7. MOV DX,[ALPHA+RBX+RSI*4]
8. MOV [RCX], 0x12345678
9. MOV [RBX],[RCX]
10. MOV word [RSI], 0x12345678

Label	Addr	Data
	40301E	77
	40301D	66
	40301C	55
	40301B	44
	40301A	33
	403019	22
	403018	11
	403017	F0
	403016	DE
	403015	BC
	403014	9A
	403013	78
	403012	56
	403011	34
ALPHA	403010	12

Exercises

Register	Value
RAX	0000 0000 2401 FFFF
RBX	0000 0000 0000 0002
RCX	0000 0000 0000 0004
RDX	1234 5678 0000 0000
RSI	0000 0000 0000 0002
RDI	0000 0000 0000 0004

Label	Addr	Data
	40301E	77
	40301D	66
	40301C	55
	40301B	44
	40301A	33
	403019	22
	403018	11
	403017	F0
	403016	DE
	403015	BC
	403014	9A
	403013	78
	403012	56
	403011	34
ALPHA	403010	12

1. `MOV RBX, [ALPHA]`
2. `MOV RCX, [ALPHA+04]`
3. `MOV RAX, 0x1234567887654321`
4. `MOV AL, [ALPHA+RSI+RDI]`
5. `MOV BL,[ALPHA+RSI*2]`
6. `MOV DL, [ALPHA+RBX+RSI*4]`
7. `MOV DX,[ALPHA+RBX+RSI*4]`
8. `MOV [RCX], 0x12345678`
9. `MOV [RBX],[RCX]`
10. `MOV word [RSI], 0x12345678`

Answer:

1. RBX=F0DEBC9A78563412
2. RCX=44332211F0DEBC9A
3. RAX=1234567887654321
4. AL=DE
5. BL=9A
6. DL=33
7. DX=4433

8-10.) syntax error

Summary of Rules

Rule #1: No data size mismatch among operands

Rule #2: Immediate addressing mode could only be used as source operand

Rule #3: No memory-to-memory operations

Rule #4: If the operand is a combination of memory address mode & immediate addressing mode, the prefix byte, word, dword or qword should be used

Some restrictions:

1. *The ESP register cannot be used as an index register*
2. *Scale factor can only be used with index register*

Summary: Memory Addressing Mode

Possible Combinations:

1. **Displacement:** also known as absolute, direct, or static
    - Example: MOV RAX, [ALPHA]
2. **Base:** also known as register indirect
    - Example: MOV RAX, [RSI]
3. **Base+displacement:** one-dimensional array
    - Example: MOV RAX,[var1+RSI]
4. **Base+index+displacement:** two-dimensional array
    - Example: MOV RAX, [var1+RSI+RDI]

Summary: Memory Addressing Mode

Possible Combinations:

5. **(Index*scale)+Displacement (or scaled-index byte):** one-dimension array and the element size are either 1,2,4 or 8 bytes
    - Example: MOV RAX, [var1+RSI*4]
6. **Base+(index*scale)+Displacement (or scaled-index byte):** two-dimensional array and the element size are either 1,2,4 or 8 bytes
    - Example: MOV RAX, [var1+RBX+RDI*4]