

- Indirect addressing – memory address is stored in register

```
movl (%ebx), %eax      %eax ← M[%ebx]
```

- It is often called register indirect
- Note similarity with base-displacement, with displacement of zero

```
movl 0x0(%ebx), %eax  %eax ← M[0+%ebx]
```

- Indexed addressing – memory address is formed by adding two registers (one called “base”, the other called “index”)

```
addl (%ebx,%esi),%ecx  %ecx ← %ecx + M[%ebx+%esi]
```

- Scaled indexed addressing – index register can be scaled by 1, 2, 4, or 8

```
movl $0x7f, (%ebx,%edi,2)  M[%ebx + 2×%edi] ← 0x7f
```

- Full form – literal (direct), base register, index register, scale are all allowed

```
movl 0x50(%ebx,%esi,4), %eax
      %eax ← M[0x50 + %ebx + 4×%esi]
```

- Form commonly output by disassemblers:

```
addl name(,1), %ecx
```

Practice Problems

Address	Value	Register	Value
0x100	0xff	%eax	0x100
0x104	0xab	%ecx	0x1
0x108	0x13	%edx	0x3

Operand	Value	Addr. mode
(%eax)		
0x4(%eax)		
0x1(%eax,%edx)		

Practice Problems

Address	Value	Register	Value
0x100	0xff	%eax	0x100
0x104	0xab	%ecx	0x1
0x108	0x13	%edx	0x3
0x10c	0x11		

Operand	Value	Effective Addr.
0xfc(,%ecx,4)		
(%eax,%edx,4)		

Operand Sizes

<op>l	32-bit operand
<op>w	16-bit operand
<op>b	8-bit operand

General Form

label: opcode operands comment

- Label is optional
- Comments use `/* comment */` notation
 - Many other assemblers use `; comment`
 - Or some other notation, e.g., `# comment`
- Pseudo-operations are commands to assembler
 - `.text` means instructions follow