

Addressing Modes and Syntax Examples

- Immediate
 - E.g., Add R1, 9;
- Direct
 - E.g., ADD R1, (M[9]);
- Register
 - E.g., ADD R1, R2;
- Register direct
 - E.g., ADD R1, (R2);
- Register indexed
 - E.g., ADD R1, R2, (M[9]);

Operand Notation	Addressing Mode
V	I, immediate: V is an immediate input operand, a 2's complement number.
(V)	D, direct: V is a memory address and (V) indicates the content of memory address V (i.e., M[V]).
R	R, register: Indicates an input data register source or a destination register or both
R, (V)	X, indexed: V is a memory address and R + V is the address of the next data item in memory (i.e., M[R + V]).

TABLE 8.1 Examples of Addressing Modes

Immediate E.g., Add R1, 9;
// $R1 \leftarrow R1 + 9$

○Direct E.g., ADD R1, (M[9]);

$R1 \leftarrow R1 + M[9]$

○Register E.g., ADD R1, R2;

$R1 \leftarrow R1 + R2$

○Register direct E.g., ADD R1, (R2);

$R1 \leftarrow R1 + M[2]$

○Register indexed E.g., ADD R1, R2, (M[9]);

$R1 \leftarrow R1 + M[R2 + 9]$

$R1 = 5;$

$R2 = 10;$

$M[19] = 50$

$R1 = 5 + 50 = 55$

$R1 = 55;$

Stack ISA - Example of assembly program: $A = B * (C + D)$

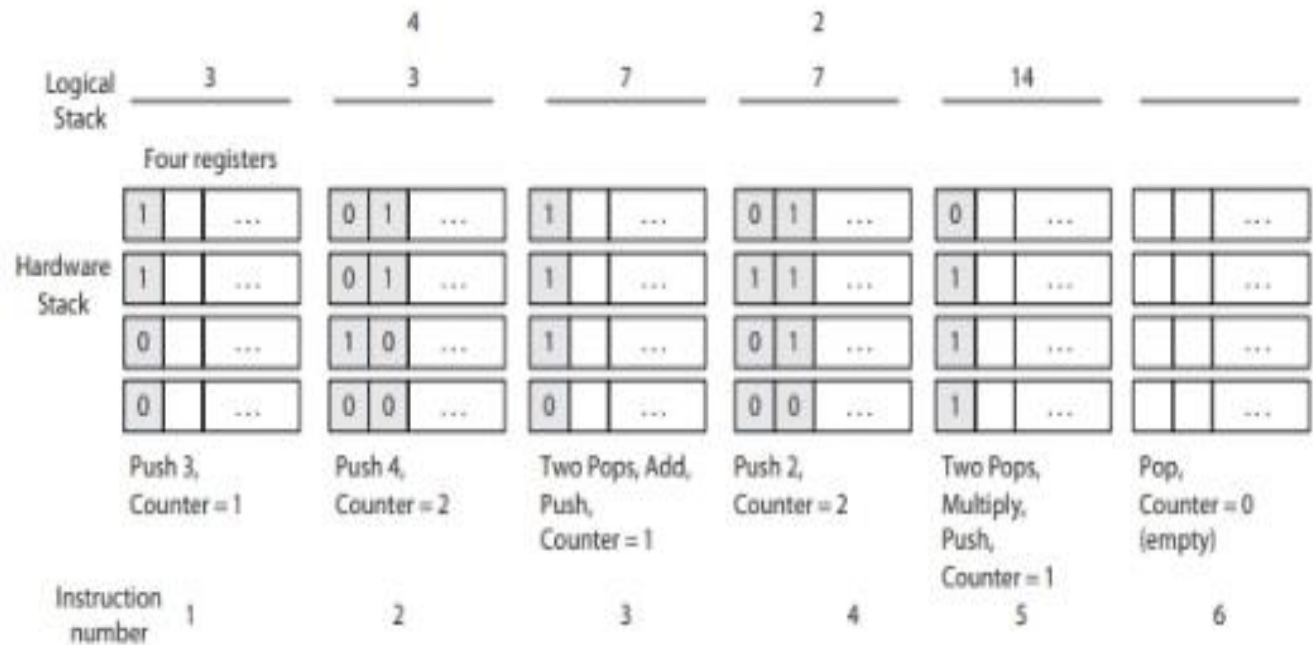


FIGURE 8.4 An illustration of stack content when computing the reverse polish notation $C D + B * = A$; it is assumed that $(B) = 2$, $(C) = 3$, and $(D) = 4$.

Instruction
number

- 1: PUSH (C) //stack $\leftarrow M[C]$
- 2: PUSH (D) //stack $\leftarrow M[D]$
- 3: ADD //stack $\leftarrow (C) + (D)$, values popped, added,
//result pushed
- 4: PUSH (B) //stack $\leftarrow M[B]$
- 5: MUL //stack $\leftarrow ((C) + (D)) * (B)$, values popped, added,
//result pushed
- 6: POP (A) //M[A] $\leftarrow (((C) + (D)) * (B))$, value is popped
//and stored in memory

Acc ISA - Example of assembly program: $A = B * (C + D)$

1: LD (C) // $ACC \leftarrow M[C]$

2: ADD (D) // $ACC \leftarrow ACC + M[D] \text{ (} C + D \text{)}$

3: MUL (B) // $ACC \leftarrow ACC * M[B]$

4: ST (A) // $M[A] \leftarrow ACC$

CISC-ISA: Example of assembly program: $A = B * (C + D)$

$B = 4; C = 5; D = 10$

The value in R1 after execution of instruction No. 1 is 5.

The value of R1 after execution of instruction No. 3 is 60

1. LD R1, (C) // $R1 \leftarrow M[C]$
2. ADD R1, (D) // $R1 \leftarrow R1 + M[D]$
3. MUL R1, (B) // $R1 \leftarrow R1 * M[B]$
4. ST (A), R1 // $M[A] \leftarrow R1$

RISC-ISA: Example of assembly program: $A = B * (C + D)$

```
1. LD    R1,    (C) // R1 ← M[C]
2. LD    R2,    (D) // R2 ← M[D]
3. ADD   R3,    R1, R2 // R3 ← R1 + R2
4. LD    R4,    (B) // R4 ← M[B]
5. MUL   R5,    R3, R4 // R5 ← R3 * R4
6. ST    (A),    R5    // M[A] ← R5
```

Program level Translation:

Now, here is an example of a real C If-Then-Else:

```
if (x == 10)
{
    x = 0;
}
else
{
    x++;
}
```

Which gets translated into the following assembly/machine code:

X = 5; 0x 05 = 5 in decimal;

X = 0xA which is equal decimal 10.

Mov eax, \$x

Cmp eax, 0x0A ; 0x0A = 10

Jne else

Mov eax, 0

Jmp end

Else:

Inc eax

End;

Mov \$x, eax