gradeup

Prep Smart. Score Better.

Computer Organization and Architecture

Programming the basic computer Part-2

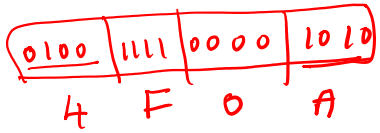
ABOUT ME : MURALIKRISHNA BUKKASAMUDRAM

- MTech with 20 years of Experience in Teaching GATE and Engineering colleges
- IIT NPTEL Course topper in Theory of computation with 96 %
- IGIP Certified (Certification on International Engineering educator)
- GATE Qualified
- Trained more than 50 Thousand students across the country
- Area of Expertise : TOC,OS,COA,CN,DLD

Programming the basic computer Part-2

AND 000
ADD 001
LDA 010

Assembly Programming



$(4F0A)_{16}$

0x 4F0A

Machine Language
- binary
- hexadecimal
- machine code or object code

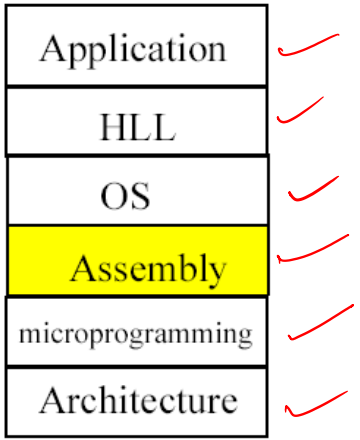
Assembly Language
- mnemonics
- assembler

High-Level Language
- Pascal, Basic, C, C#, JAVA
- compiler

MOV
PUSH
MUL
DIV

Motivations

- Why do you learn assembly language?



Programming the basic computer Part-2

MOV R1, R2 ; R1 ← R2

Assembly Instruction format

General format

mnemonic	operand(s)	;comments

```
MOV        destination,source    ;copy source operand to destination
```

Example:

```
MOV    DX,CX
```

8086

8085

Example 2:

```
MOV    CL,55H
MOV    DL,CL
MOV    AH,DL
MOV    AL,AH
MOV    BH,CL
MOV    CH,BH
```

AH	AL
BH	BL
CH	CL
DH	DL

55H

Programming the basic computer Part-2

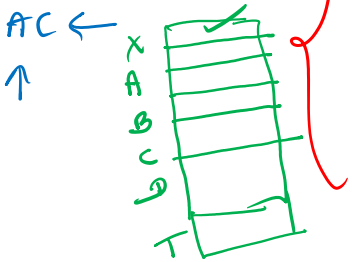
Assembly Language instruction consist of four fields

[label:] mnemonic [operands] [;comment]

- Labels
  See rules
- mnemonic, operands
  MOV AX, 6764
- comment
  ; this is a sample program

Programming the basic computer Part-2

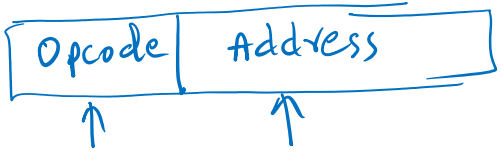
Accumulator



(1) three Address instructions } GPA
(2) Two Address instructions
(3) One Address instructions (single Add. Single AC)
(4) Zero-Address instruction. (opcode)

Stack organization

ONE-Address instructions



D_0 000 – AND
001 – ADD
010 – LDA
011 – STA
100 – BUN
101 – BSA
D_6 110 – ISZ

$X = (A+B) * (C+D)$

```
LDA  A ; AC ← M[A]
ADD  B ; AC ← AC + M[B]
STA  T ; M[T] ← AC
LDA  C ; AC ← M[C]
ADD  D ; AC ← AC + M[D]
MUL  T ; AC ← AC * M[T]
STA  X ; M[X] ← AC
```

Programming the basic computer Part-2

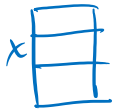
$X = (A + B) * (C + D)$

General Register's organization

Opcode	oprd1	oprd2	oprd3

Opcode	oprd1	oprd2

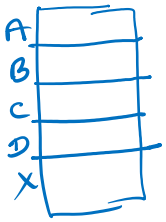
ADD X, Y



Three-Address instructions

```
ADD R1, A, B   ; R1 ← M[A] + M[B]
ADD R2, C, D   ; R2 ← M[C] + M[D]
MUL X, R1, R2  ; M[X] ← R1 * R2
```

Programming the basic computer Part-2



Two - Address instructions

$X = (A+B) * (C+D)$

```
MOV R1, A   ; R1 ← M[A]
ADD R1, B   ; R1 ← R1 + M[B]
MOV R2, C   ; R2 ← M[C]
ADD R2, D   ; R2 ← R2 + M[D]
MUL R1, R2  ; R1 ← R1 * R2
MOV X, R1   ; M[X] ← R1
```

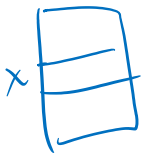
Zero - Address instructions (stack-organized computer)

$X = (A+B) * (C+D)$

RPN (Reverse Polish Notation) | Post fix

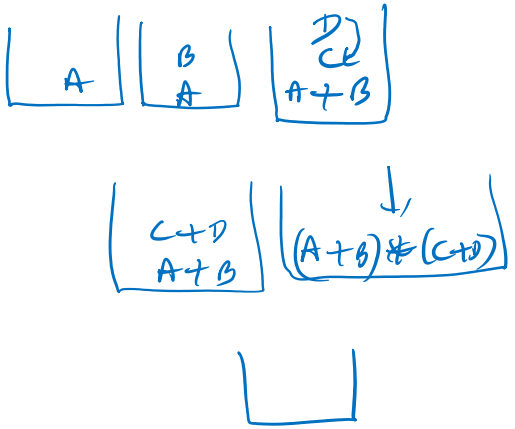
$AB + CD + *$

Programming the basic computer Part-2



AB + CD + *

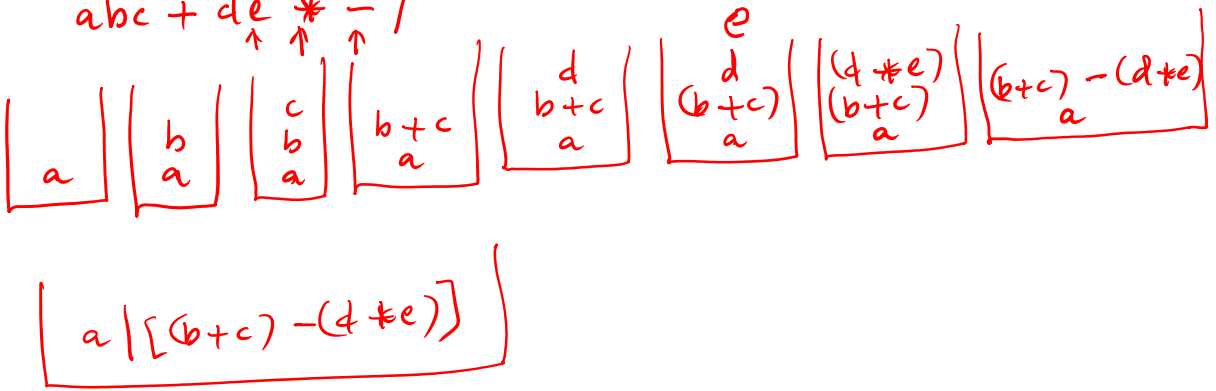
```
Push A ;  TOS ← M[A]
Push B ;  TOS ← M[B]
ADD    ;  TOS ← A+B
Push C ;  TOS ← M[C]
Push D ;  TOS ← M[D]
ADD    ;  TOS ← C+D
MUL    ;  TOS ← (A+B)*(C+D)
POP X  ;  M[X] ← TOS
```



Programming the basic computer Part-2

RPN : abc + de * - /

Q). what are the min no. of registers required to perform the operation ?

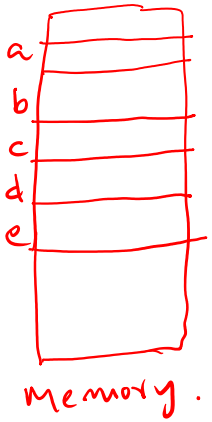


→ Assume that a Computer evaluates the following expression using two Address instruction using GPR's.

$$(a-b) + (e-(c+d))$$ ✓

Assume, a, b, c, d, and e are initially in memory. We can perform the opn, only when both operands are in registers, No, intermediate results can be stored in memory.

Programming the basic computer Part-2

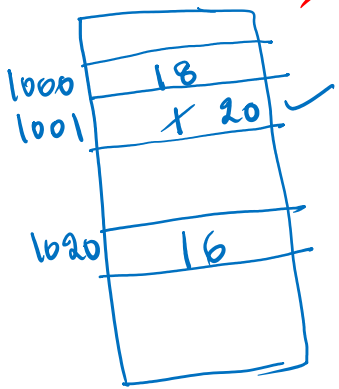


$$[(a-b)+(e-(c+d))]$$

3 Registers

```
Load  R1, a   ; R1 ← M[a]
Load  R2, b   ; R2 ← M[b]
Sub   R1, R2  ; R1 ← A1 − R2
Load  R2, c   ; R2 ← M[c]
Load  R3, d   ; R3 ← M[d]
Add   R2, R3  ; R2 ← R2 + R3
Load  R3, e   ; R3 ← M[e]
Sub   R3, R2  ; R3 ← R3 − R2
Add   R1, R3  ; R1 ← R1 + R3
```

Programming the basic computer Part-2



→ The Memory Locations 1000, 1001, and 1020 have data values 18, 1 and 16 respectively before the following Program is executed.

```
MOVI   Rs, 1          ; Move immediate
LOAD   Rd, 1000(Rs)   ; Load from memory
ADDI   Rd, 1000       ; Add immediate
STOREI 0(Rd), 20      ; Store immediate
```

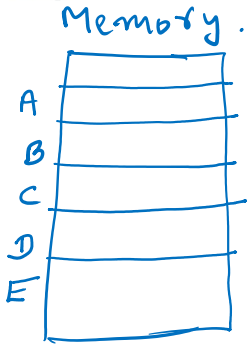
1000 + (Rs)
1000 + 1

which of the statements below is TRUE, after the Program is executed ?

(A) Mem. Location 1000 has value 20
(B) " 1020 " 20
(C) " 1021 " 20
(D) " 1001 " 20

Rs = 1
Rd = 1
1 + 1000
1001
Rd = 1001
0 + 1001 = 1001

Programming the basic computer Part-2



→ Evaluate the following Expression using, ONE Address, Two Address, three Address and Zero Address instructions.

AB + C * DE - |

$X = [(A+B) * C] | (D-E)$

BODMAS

ONE - Address

```
LDA  D   ; AC ← M[D]
SUB  E   ; AC ← AC - M[E]
STA  T   ; M[T] ← AC
LDA  A   ; AC ← M[A]
ADD  B   ; AC ← AC + M[B]
MUL  C   ; AC ← AC * M[C]
DIV  T   ; AC ← AC | M[T]
STA  X   ; M[X] ← AC.
```

```
Push A ;
Push B ;
ADD    ;
Push C ;
MUL    ;
```

```
Push D ;
Push E ;
SUB    ;
DIV    ;
Pop X  ;
```