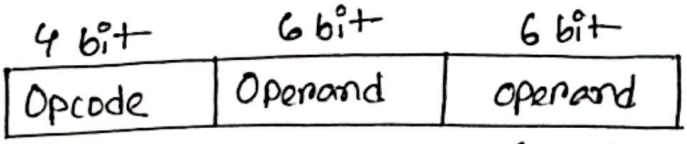
CHAP-10 Instruction Set : Characteristics & function

☐ Elements of instruction

① OP Code – Do this
② Source Operand – To this
③ Result Operand – Put the result here
④ Next instruction –

☐ Source & Result operand can be

① Main or Virtual memory
② CPU Registers
③ Immediate
④ I/O device



simple instruction format

☐ Instruction Cycle state Diagram

chap 03 note

☐ Instruction Representation

→ In machine code each instruction has a unique bit pattern.
→ It is difficult for programmers to deal with binary representation
→ Opcodes are represented by abbreviations, called mnemonics, that indicate the operation

ADD → add
SUB → subtract
MUL → x
DIV → ÷
LOAD → Load data from memory

☐ Instruction types

1. Data processing : Arithmetic and Logic instructions
2. Data storage : Movement of data into or out of reg. or memory location
3. Data movement : I/O instructions
4. Control : Test and branch instruction

☐ Numbers of Addresses

→ 3 address – a = b+c
- operand 1, Operand 2, Result
- Need very long word to hold everything

→ 2 address – a = a+b, Operand and Result
- Reduce length of instruction
- Requires some extra work

→ 1 address – Implicit second address
- Usually Accumulators

→ 0 Address – all addresses implicit
- Uses a stack
- PUSH a
  PUSH b
  ADD
  POP C

C = a+b

- Mode address in instruction
  - → More complex instructions
  - → More register
  - → fewer instruction per program
- Fewer address
  - → Less complex
  - → More instruction per program
  - → ~~fat~~ faster fetch/execution of instruction
- Types of operand
  1. address
  2. Numbers (Integer/floating point)
  3. Characters (ASCII etc)
  4. Logical Data
- Types of Operation:
  1. Data transfer
  2. Arithmetic
  3. Logical
  4. Conversion
  5. I/O
  6. System Control
  7. Transfer of control

☐ Data transfers

- Source, Destination, Amount of Data must be specified

☐ Arithmetic

- Add, SUB, MUL, DIV
- signed integers, floating-point, Packed Decimal
  → May include:
  increment (A++)
  decrement (A--)
  Negate (-A)

☐ Conversion

Binary to Decimal

☐ Input/Output

- isolated programmed I/O
- memory-mapped I/O
- DMA

☐ Transfer of Control

→ Branch – branch to x if result is zero

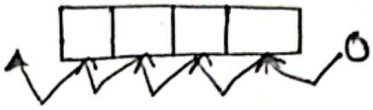
→ Skip – increment & skip if 0

→ Subroutine call - interrupt call

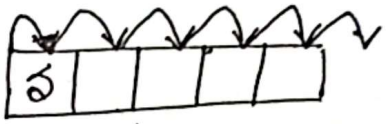
☐ Logical

- AND, OR, NOT
- logical right shift

logical left shift



- Arithmetic left shift



Arithmetic right shift

☐ System control

→ reserved for operating system

→ only when CPU in privileged state