Instruction format

&

Addressing Modes Formats

D.Venkata Vara Prasad
SSN College of Engineering

Session Meta Data

Author	D.Venkata Vara Prasad
Version No	1.1
Release Date	25.01.2021
Reviewer	

Revision History

Date of Revision	Details	Version Number

Session Objectives

- To explain the various Instruction & addressing modes and formats.

Session Outcomes

- At the end of the session, students will be able to

- Understand the various addressing modes and formats.

Outline

- Implied
- Immediate
- Direct
- Indirect
- Register
- Register Indirect
- Displacement (Indexed)
- Stack

Instruction Format

Instruction Formats

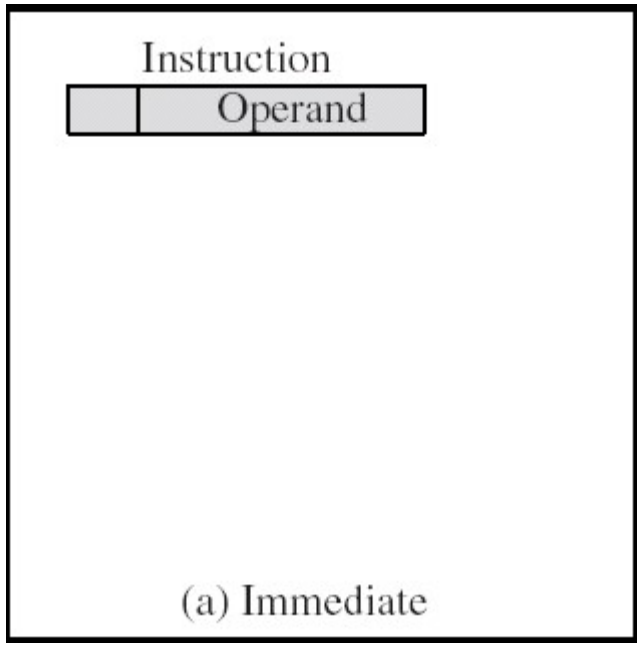
- Layout of bits in an instruction :
  - Includes opcode
  - Includes (implicit or explicit) operand(s)
  - Usually more than one instruction format in an instruction set
- Opcode: specifies type of operation to be executed
- Operand: data field

Instruction Length

- Affected by and affects:
  - Memory size
  - Memory organization
  - Bus structure
  - CPU complexity
  - CPU speed

Immediate Addressing

- Operand is part of instruction
- Operand = address field
- e.g. ADD AX, 5h
- LDA #5
  - Add 5 to contents of accumulator
  - 5 is operand
- No memory reference to fetch data
- Fast
- Limited range



Direct Addressing

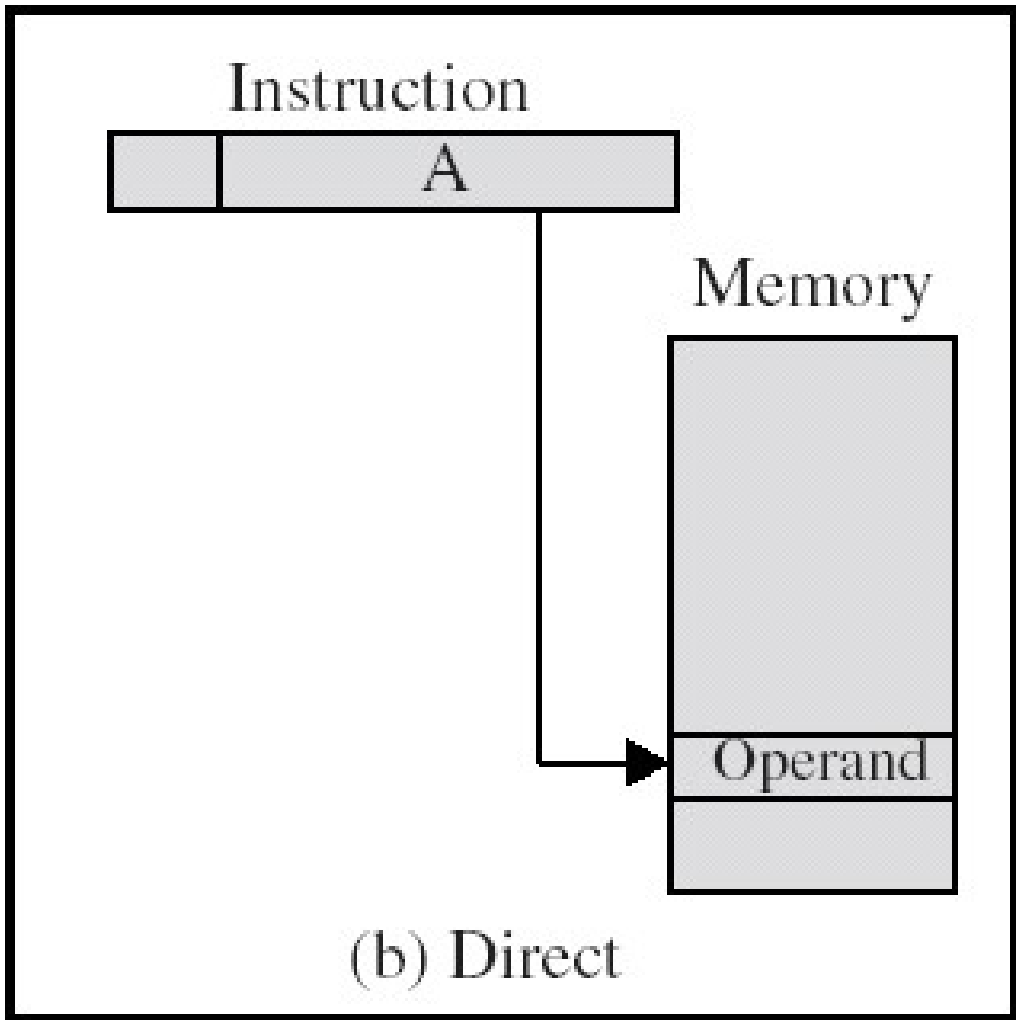
- Address field contains address of operand
- Effective address EA = address field (A)

ADD AX, value

Value DB 05h

  - Add contents of cell value to accumulator AX
  - Look in memory at address value for operand
- Single memory reference to access data
- No additional calculations to work out effective address
- Limited address space

Direct Addressing Diagram



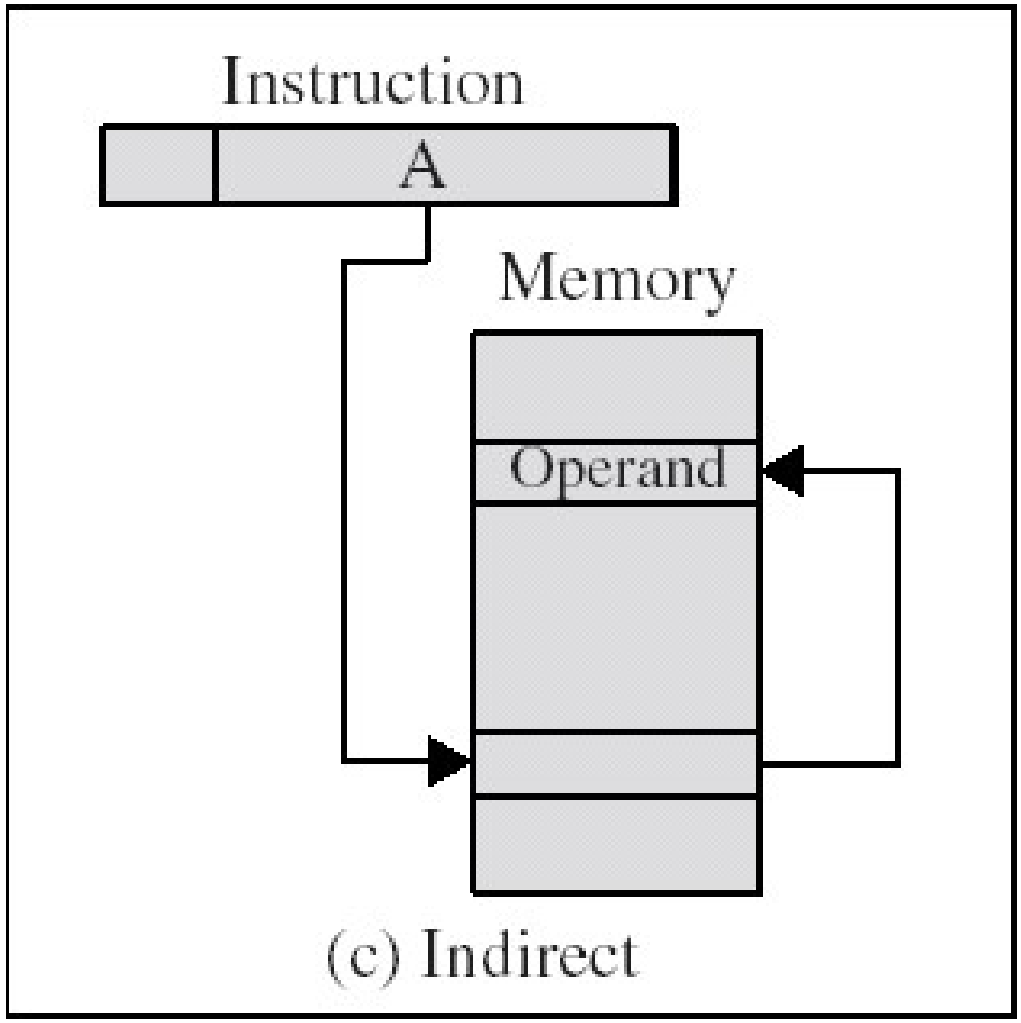
Indirect Addressing (1/2)

- Memory cell pointed to by address field contains the address of (pointer to) the operand
- EA =(A)
  - Look in A, find address (A) and look there for operand
- e.g. ADD AX, (A)
  - Add contents of cell pointed to by contents of A to accumulator

Indirect Addressing (2/2)

- Large address space
- 2^n where n = word length
- May be nested, multilevel, cascaded
  - e.g. EA = (((A)))
- Multiple memory accesses to find operand
- Hence slower

Indirect Addressing Diagram



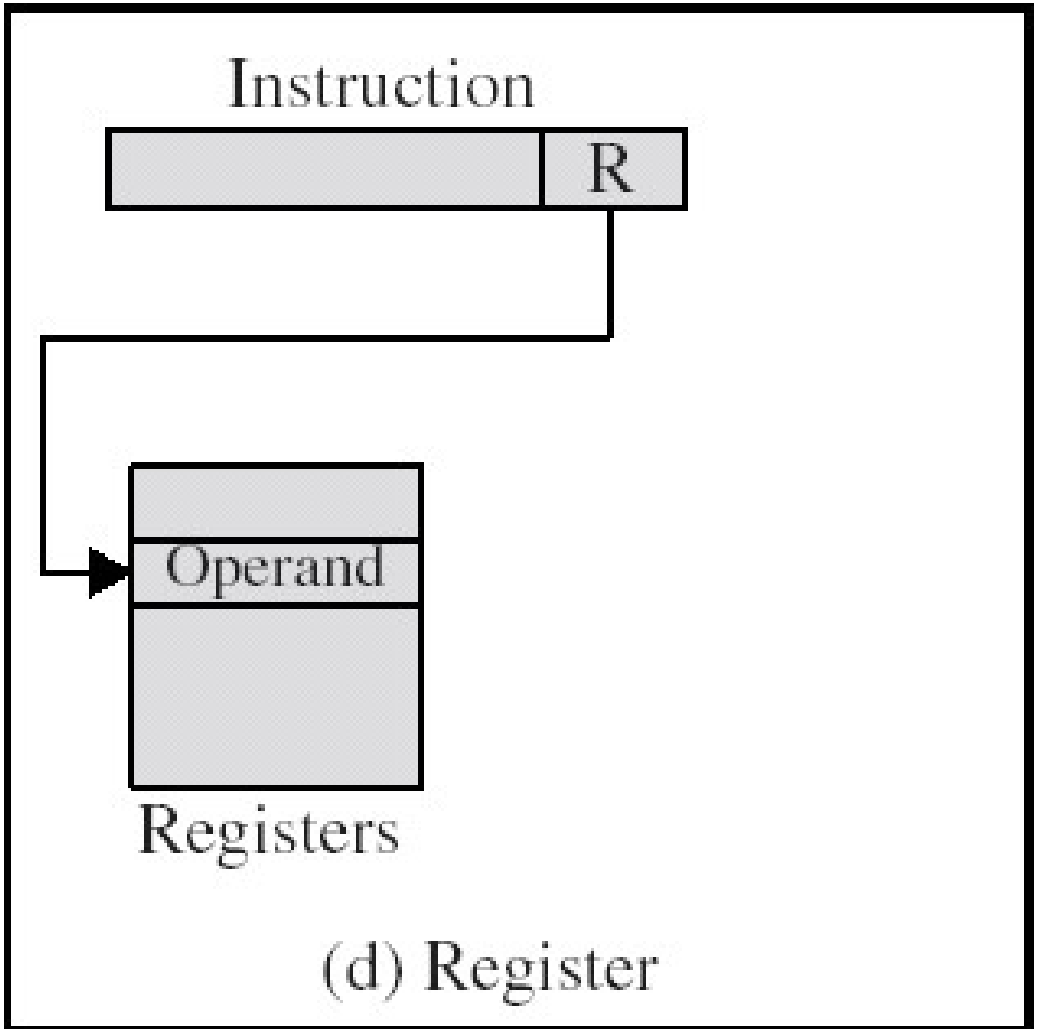
Register Addressing (1/2)

- Operand is held in register named in address filed
- EA = R
- Limited number of registers
- Very small address field needed
  - Shorter instructions
  - Faster instruction fetch
  - MOV AX, BX
  - ADD AX, BX

Register Addressing (2/2)

- No memory access
- Very fast execution
- Very limited address space
- Multiple registers helps performance
  - Requires good assembly programming or compiler writing
  - N.B. C programming
    - register int a;

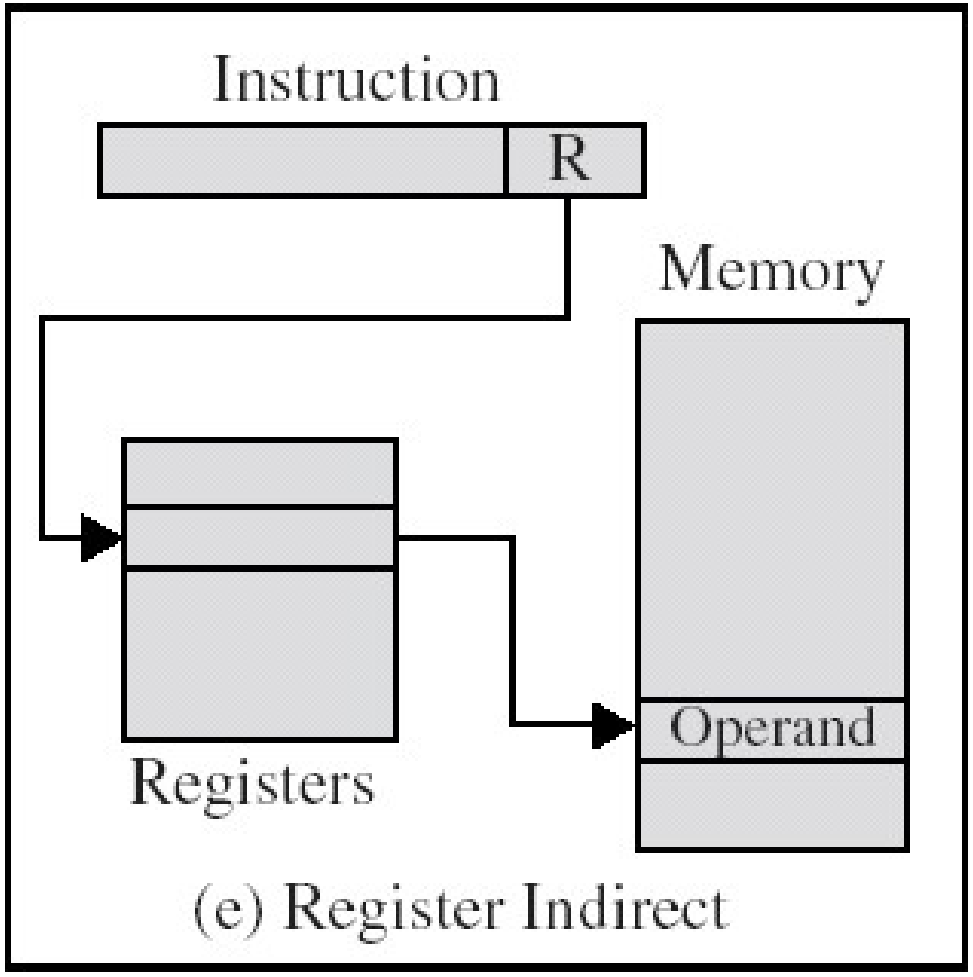
Register Addressing Diagram



Register Indirect Addressing

- indirect addressing
- EA = (R)
- Operand is in memory cell pointed to by contents of register R
- Large address space (2^n)
- One fewer memory access than indirect addressing

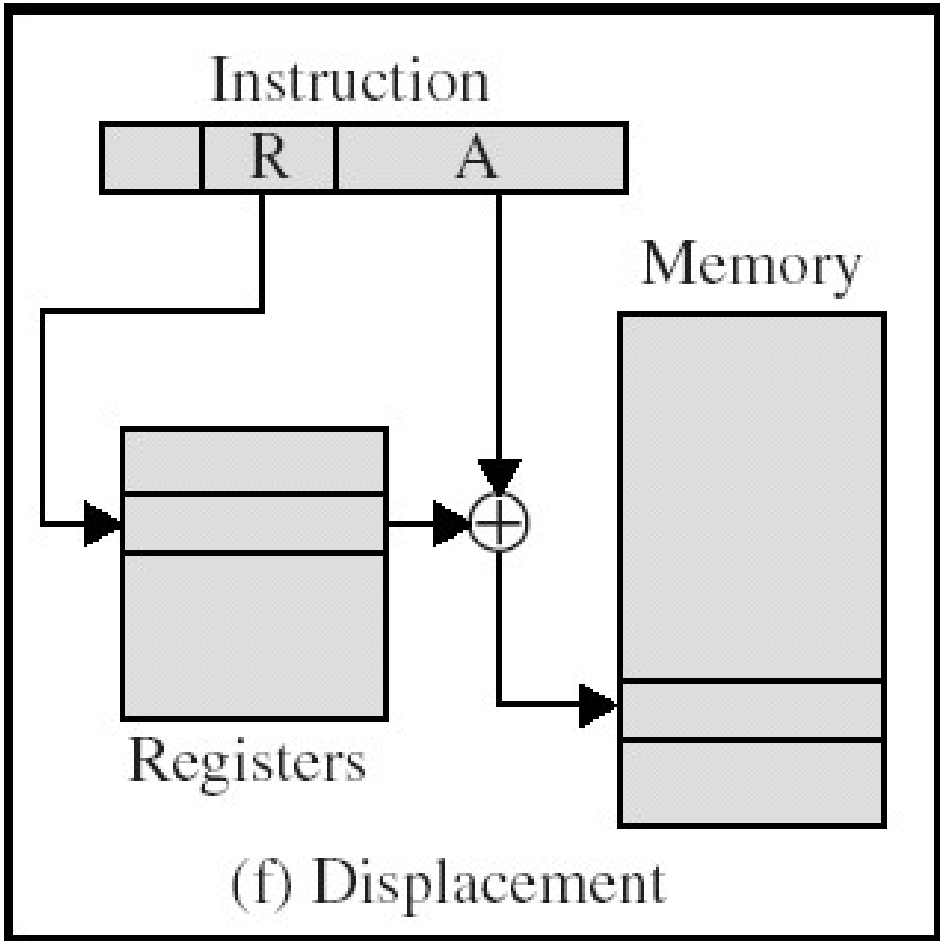
Register Indirect Addressing Diagram



Displacement Addressing

- EA = A + (R)
- Effective address=start address + displacement
- Effective address=Offset + (Segment Register)
- Use direct and register indirect
- Address field hold two values
  - A = base value
  - R = register that holds displacement
  - or vice versa

Displacement Addressing Diagram



Relative Addressing (PC-Relative)

- A version of displacement addressing
- R = Program counter, PC
- EA = A + (PC)
- i.e. get operand from A cells from current location pointed to by PC
- locality of reference & cache usage

Base-Register Addressing

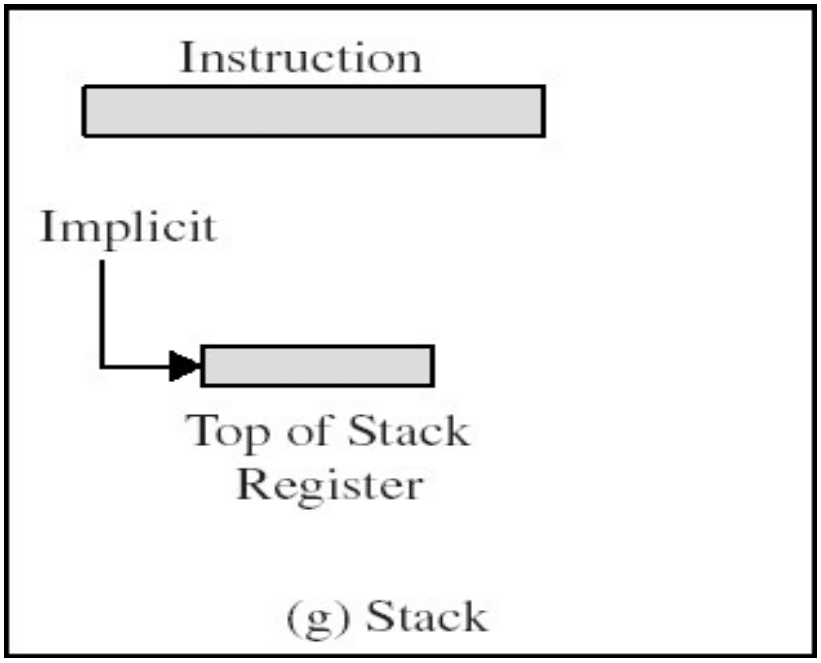
- A holds displacement
  - EA = (CS) + A
- R holds pointer to base address
- R may be explicit or implicit
- e.g. segment registers in 80x86

Indexed Addressing

- A = base
- R = displacement
  - EA = A + (R)
- Good for accessing arrays
  - EA = A + (R)
  - R++

Stack Addressing

- Operand is (implicitly) on top of stack
- e.g.
  - ADD Pop top two items from stack and add and push



Mode	Algorithm	Principal Advantage	Principal Disadvantage
Immediate	Operand = A	No memory reference	Limited operand magnitude
Direct	EA = A	Simple	Limited address space
Indirect	EA = (A)	Large address space	Multiple memory references
Register	EA = R	No memory reference	Limited address space
Register indirect	EA = (R)	Large address space	Extra memory reference
Displacement	EA = A + (R)	Flexibility	Complexity
Stack	EA = top of stack	No memory reference	Limited applicability

Pentium Addressing Modes

- Virtual or effective address is offset into segment
  - Starting address plus offset gives linear address
  - This goes through page translation if paging enabled
- 9 addressing modes available
  - Immediate
  - Register operand
  - Displacement
  - Base
  - Base with displacement
  - Scaled index with displacement
  - Base with index and displacement
  - Base scaled index with displacement
  - Relative

Summary

The various addressing modes and formats was studied.

References

1. David A. Patterson and John L. Hennessey, “Computer Organization and Design‟, Fifth edition, Morgan Kauffman / Elsevier, 2014.

2. V.Carl Hamacher, Zvonko G. Varanesic and Safat G. Zaky, “Computer Organisation“, VI edition, Mc Graw-Hill Inc, 2012.

Thank you