SNS COLLEGE OF ENGINEERING

Kurumbapalayam (PO), Coimbatore – 641 107

Accredited by NAAC-UGC with 'A' Grade

Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai
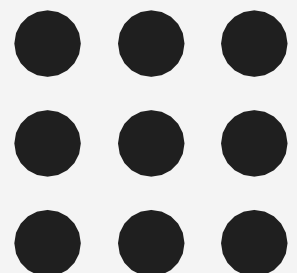
DEPARTMENT OF CSE (IoT)

COURSE NAME: 23ITT201 DIGITAL PRINCIPLES AND COMPUTER ORGANIZATION

II YEAR/ III SEM

Unit 3 : COMPUTER FUNDAMENTALS

Instruction and Instruction Sequencing
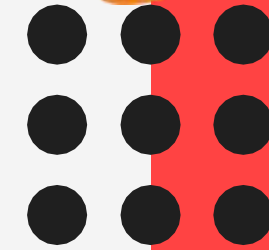
"Must-Perform" Operations

- Data transfers between the memory and the processor registers
- Arithmetic and logic operations on data
- Program sequencing and control
- I/O transfers

Register Transfer Notation

- Identify a location by a symbolic name standing for its hardware binary address (LOC, R0,DATAIN, …)
- Contents of a location are denoted by placing square brackets around the name of the location

$$R1 \leftarrow [LOC]$$
$$R3 \leftarrow [R1]+[R2]$$

- This type of notation is Register Transfer Notation (RTN)

23ITT201 / DP & CO / D.KAVITHA/AP/CSE(IoT) / Unit 3 / COMPUTER FUNDAMENTALS

Assembly Language Notation

- Assembly language(symbolic machine code) takes complete control over the system and its resources.
- Represent machine instructions and programs.
- Move LOC, R1
- Add R1, R2, R3

23ITT201 / DP & CO / D.KAVITHA/AP/CSE(IoT) / Unit 3 / COMPUTER FUNDAMENTALS
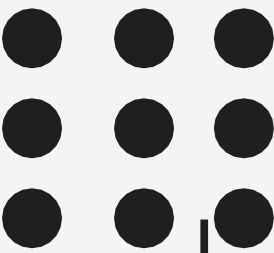
Instruction Formats

- Three-Address Instructions
  - Format: Operation Source1, Source2, Destination
  - ADD   R2, R3, R1   R1 ← [R2] + [R3]
- Two-Address Instructions
  - Format: Operation Source, Destination
  - ADD   R2, R1   R1 ← [R1] + [R2]
- One-Address Instructions
  - ADD   M   AC ← [AC] + [M]
  - Load   A
  - Store   A
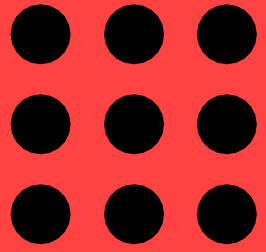
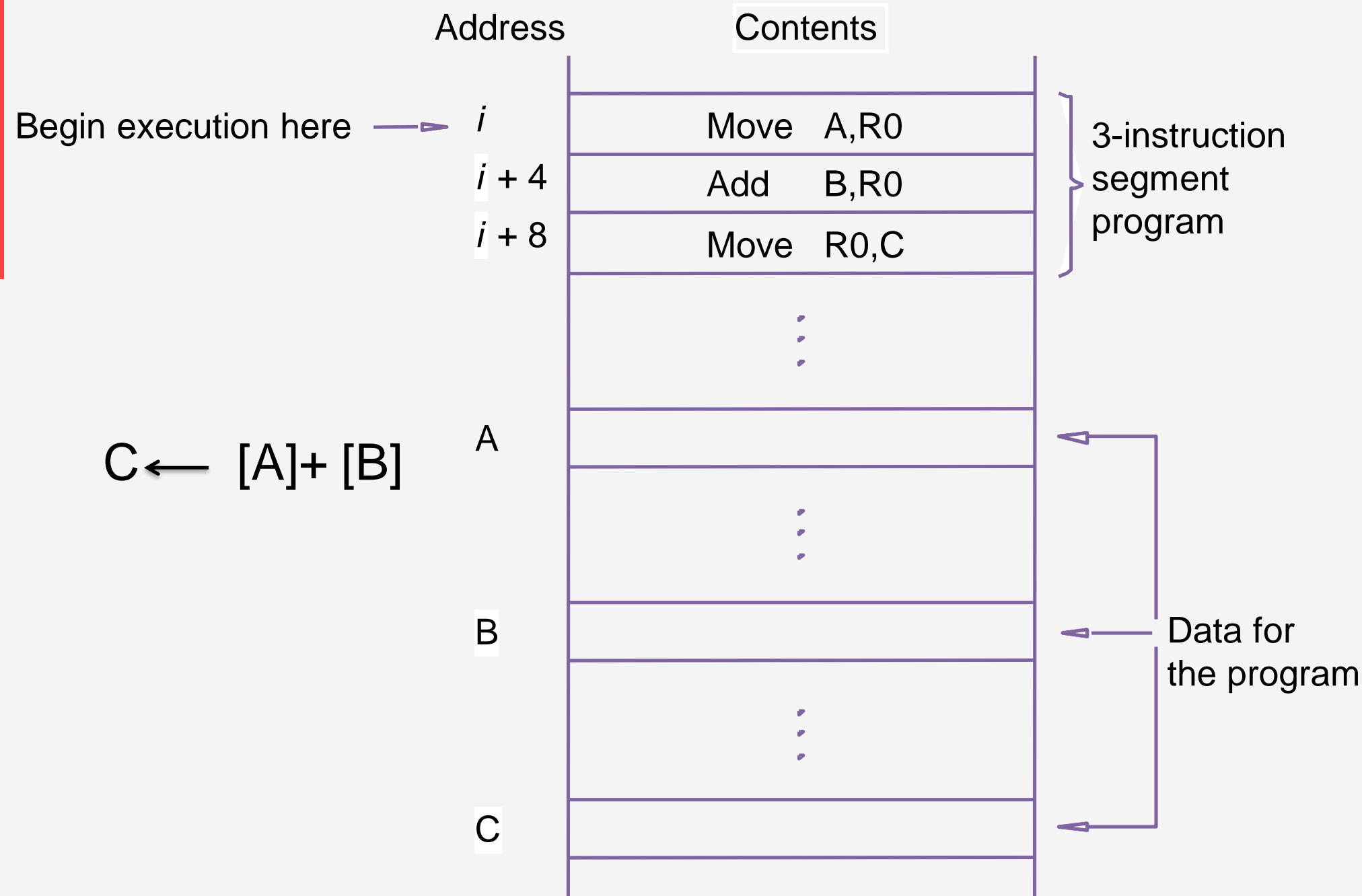# Instruction Formats

Example: Evaluate C = A+B using processor registers,

Move   A, R_i
Move   B, R_j
Add $R_{i,}$ R_j
Move $R_{j,}$ C

Instruction Execution and Straight-Line Sequencing

Address   Contents

Begin execution here ——→ *i*   Move   A,R0   ⎤ 3-instruction
                    i + 4   Add   B,R0   ⎬ segment
                    i + 8   Move   R0,C   ⎦ program

C ←—— [A]+ [B]

A   ←— ⎤

B   ←— ⎬ Data for
                                        ⎭ the program

C   ←—

Assumptions:
- One memory operand
  per instruction
- 32-bit word length
- Memory is byte
  addressable
- Full memory address
  can be directly specified
  in a single-word instruction

Two-phase procedure
-Instruction fetch
-Instruction execute

Processor control circuits use PC to fetch and execute instructions,
one at a time, in the order of increasing addresses **called straight-line
sequencing**

Branching

Address	Instruction	
i	Move	NUM1,R0
$i + 4$	Add	NUM2,R0
$i + 8$	Add	NUM3,R0
	⋮	
$i + 4n - 4$	Add	NUMn, R0
$i + 4n$	Move	R0, SUM
	⋮	
SUM		
NUM1		
NUM2		
	⋮	
NUM n		

A straight-line program for adding n numbers.

Branching

Move	N,R1
Clear	R0

LOOP
Determine address of "Next" number and add "Next" number to R0

Program loop

Decrement	R1
Branch>0	LOOP

Conditional branch

Move	R0,SUM

$\vdots$

SUM

N | n

NUM1
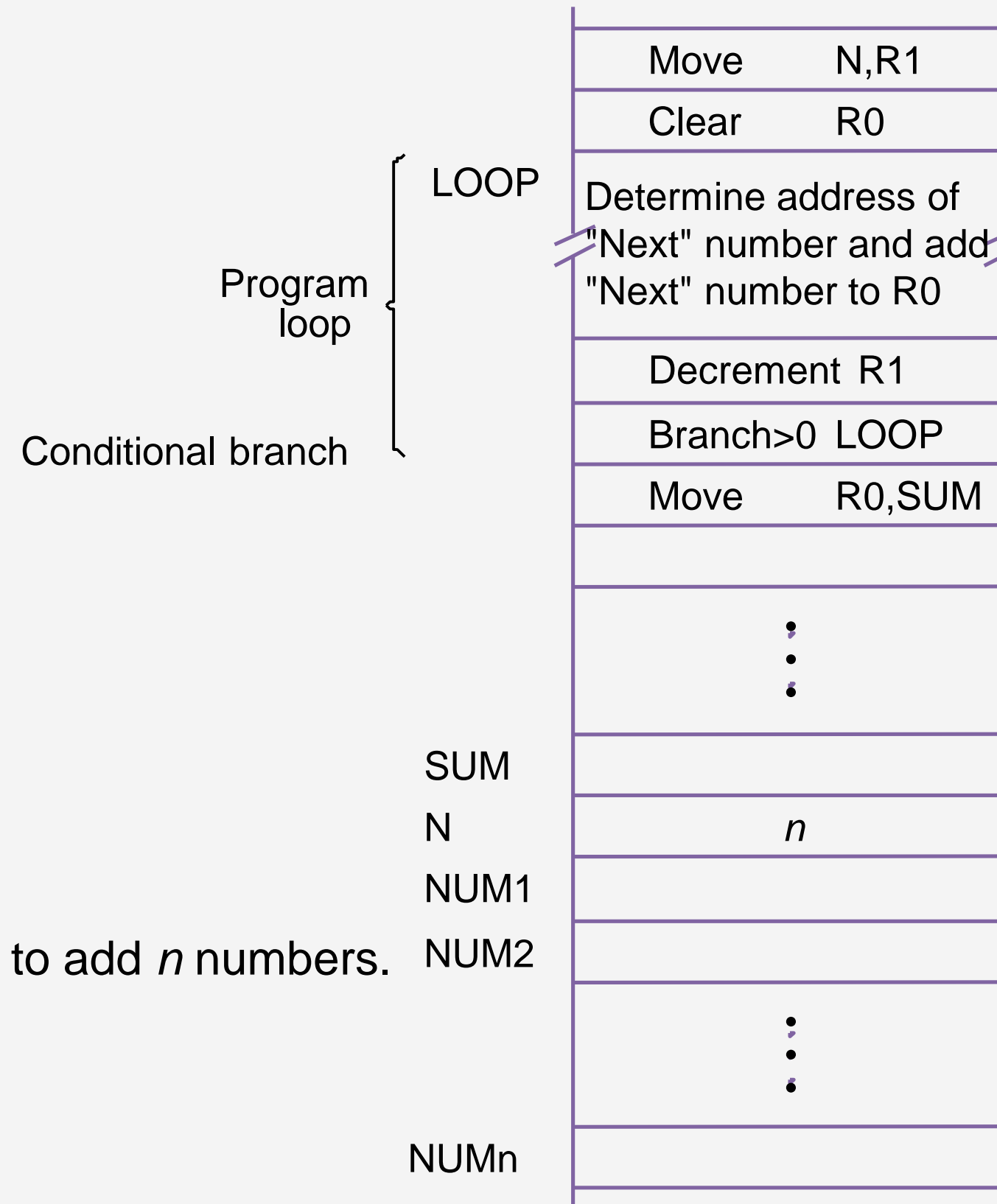
Using a loop to add n numbers.   NUM2

$\vdots$

NUMn
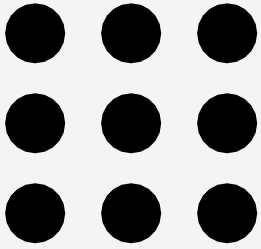
Condition Codes

- The processor keeps track of information about the results of various operations.
- This is accomplished by recording the required information in individual bits, called **Condition Code Flags.**
- Flags are grouped together in a special processor-register called the **condition code register (or status register).**

  - 4 commonly used flags
    - ✓ N (negative)
    - ✓ Z (zero)
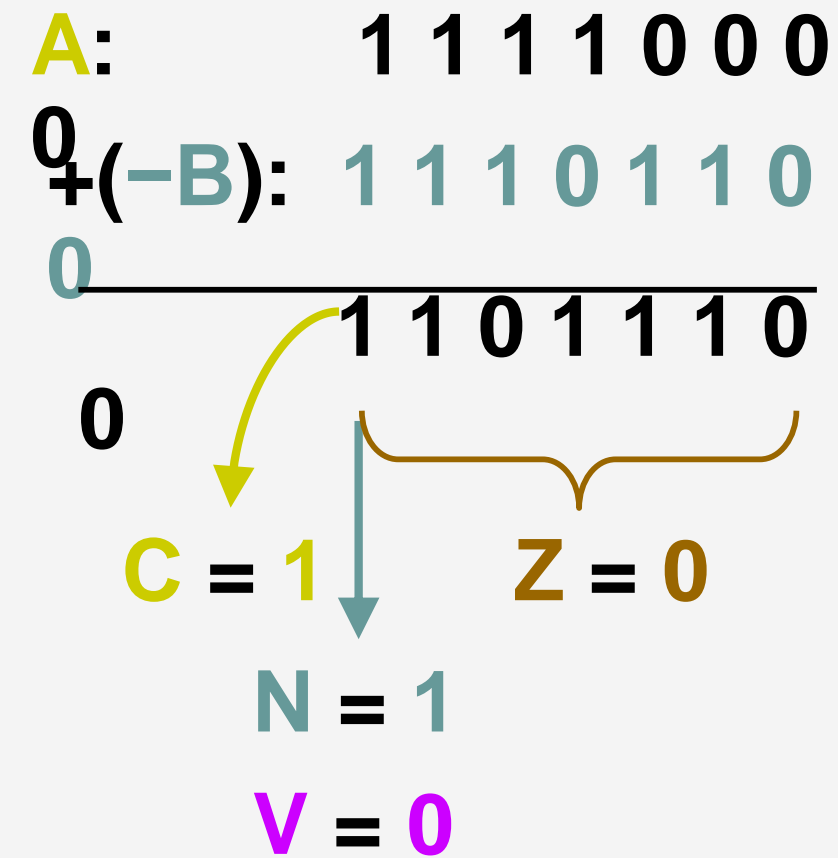   - ✓ V (overflow)
    - ✓ C (carry)
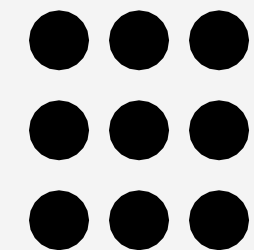
Conditional Branch Instructions

- Example:
  - A: 1 1 1 1 0 0 0 0
  - B: 0 0 0 1 0 1 0 0

A: 1 1 1 1 0 0 0 0

+(−B):   1 1 1 0 1 1 0 0

‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾

1 1 0 1 1 1 0 0

C = 1   Z = 0

N = 1

V = 0

Thank You