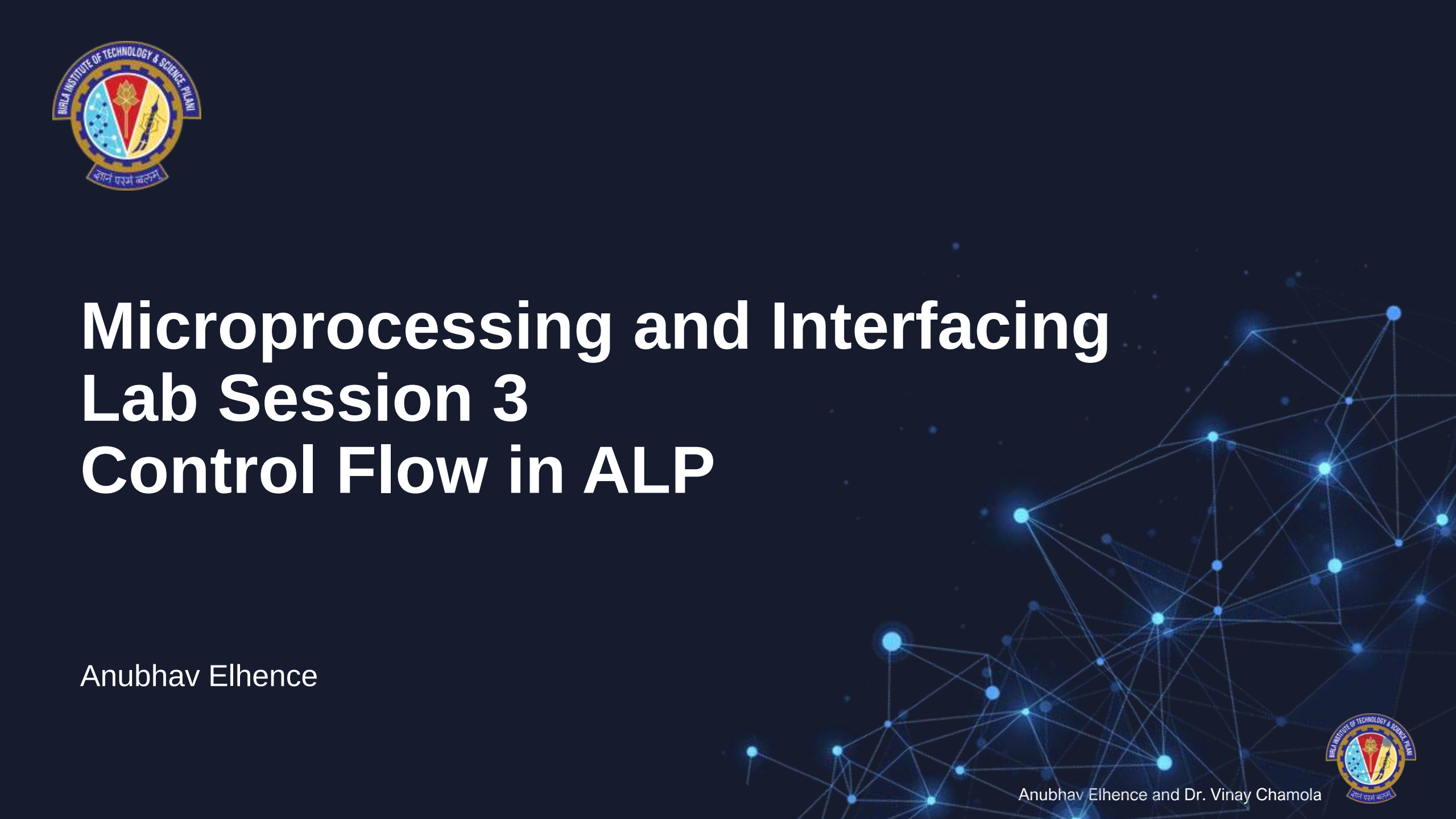

# Microprocessing and Interfacing
# Lab Session 3
# Control Flow in ALP

Anubhav Elhence

# Compare Instruction

- Compare instruction is a subtraction that changes only the Flag bits

- CMP Dest,Source
  - CMP CL, [BX]
  - CMP AX, 2000h
  - CMP [DI], CH

- CMP CX, BX
  - If CX = BX  CF = 0 | ZF = 1 | SF = 0
  - If CX < BX  CF = 1 | ZF = 0 | SF = 1
  - If CX > BX  CF = 0 | ZF = 0 | SF = 0

# To Check Results of CMP Logical

## CMP Logical

- JA/JNBE
- JAE/JNB/JNC
- JB/JC/JNAE
- JBE/JNA
- JE/JZ
- JNE/JNZ

## CMP Arithmetic

- JG/JNLE
- JGE/JNL
- JL/JNGE
- JLE/JNG
- JE/JZ
- JNE/JNZ

# Follow Along Example

- Write an ALP to copy 10 bytes of data from memory location src_data to memory location dst_data.

Given:

```
1    .model tiny
2    .data
     1 reference
3    src_str         db      03h,05h,06h,0abh,07h,08h,09h,10h,11h,12h
     1 reference
4    dst_str         db      10 dup(?)
```

- Pause the video and give it a try

- Let's look at a naive algorithm as Mem ⇔ Mem is not possible

```
1    .model tiny
2    .data
     2 references
3    src_str         db      03h,05h,06h,0abh,07h,08h,09h,10h,11h,12h
     2 references
4    dst_str         db      10 dup(?)
5    .code
6    .startup
7                    lea     si,src_str
8                    lea     di,dst_str
9                    mov     cx,0ah
10                   mov     ax,00h
11           x1:     mov     al,[si]
12                   mov     [di],al
13                   inc     si
14                   inc     di
15                   dec     cx
16                   cmp     cx,00h
17                   jne     x1
18   .exit
     1 reference
19   end
```

- Let's use a new instruction to do this

```
1    .model tiny
2    .data
     2 references
3    src_str         db      03h,05h,06h,0abh,07h,08h,09h,10h,11h,12h
     2 references
4    dst_str         db      10 dup(?)
5    .code
6    .startup
7                    lea     si,src_str
8                    lea     di,dst_str
9                    mov     cx,0ah
10             rep   movsb
11   .exit
     0 references
12   end
```

# Follow Along Example

- Write an ALP to find the greatest Signed number from a set of 10 bytes stored at location *array1* and store the greatest no. at location RESULT.

- Solution

```
1    .model tiny
2    .data
     1 reference
3    array1          db      91h,02h,083h,0ffh,075h,06h,047h,012h,076h
     2 references
4    RESULT             db      ?
5    .code
6    .startup
7                       lea     bx,RESULT
8                       lea     bx,array1
9                       mov     cl, 0ah
10                      mov     al,[bx]
11                      dec     cl
12                      inc     bx
13              X2:     CMP     al,[bx]
14                      JGE     SKIP
15                      MOV     al,[bx]
16              SKIP:   INC     bx
17                      dec     cl
18                      jnz     X2
19                      mov     RESULT,AL
20   .exit
     2 references
21   end
```

# Lab Task

- Task 1

  Write an ALP that will examine the contents of set of 10 bytes starting from location 'array1' for the presence of data '0Ah' and replace it with ASCII character 'E'.

```
1    .model tiny
2    .data
3    array1          db      91h,02h,083h,0Ah,075h,0Ah,047h,012h,076h,61h
```

# Lab Task

- Task 2

Write an ALP that will count the number of negative numbers in an array of 16-bit signed data stored from location 'array1'. The number of elements in array1 is present in location 'count'. The count of negative numbers must be stored in location 'NEG1'

```
1   .model tiny
2   .data
3   array1          db      ;fill this up yourself
4   count           db      ;since you have filled the above array,
5                           ;therefore you only know the count
6   NEG1            db      ?
```