

# Lab 1 Report

## Computer Organization

### Fall 2025

Alexandria University  
Faculty of Engineering  
Ahmed Sherif Abdelmonem Abdelhameed  
ID: 23010197

December 6, 2025

## Problem One: Code Analysis

### Given Code:

```
1 ORG 18dh
2 MOV AX, 1
3 MOV CX, data
4 START_LOOP:
5 MUL CX
6 LOOP START_LOOP
7 data DW 0005h
8 END
```

Listing 1: Factorial Calculation

### Functionality:

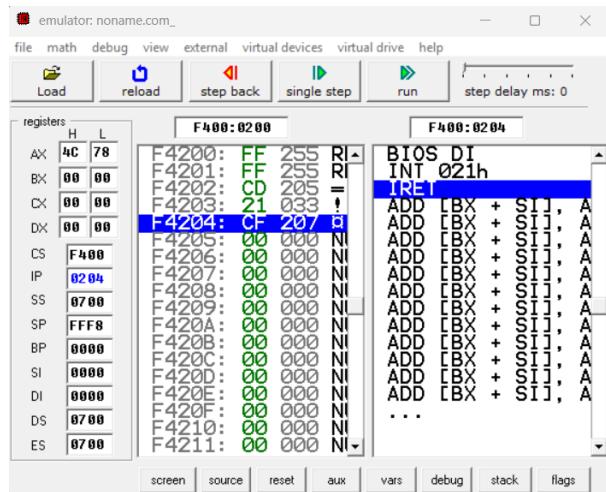
This program calculates the factorial of the value stored in **data** (5). It uses a loop that multiplies AX by CX, decrementing CX each iteration until CX becomes 0.

### Test Case:

- **Input:** **data** = 0005h (5 decimal)
- **Expected Output:** **AX** = 0078h (120 decimal = 5!)

### Register Values After Execution:

- **AX** = 0078h (120 decimal = 5!)
- **CX** = 0000h (loop ends when CX reaches 0)



## Problem Two: Range Sum Calculator

### Assembly Code:

```

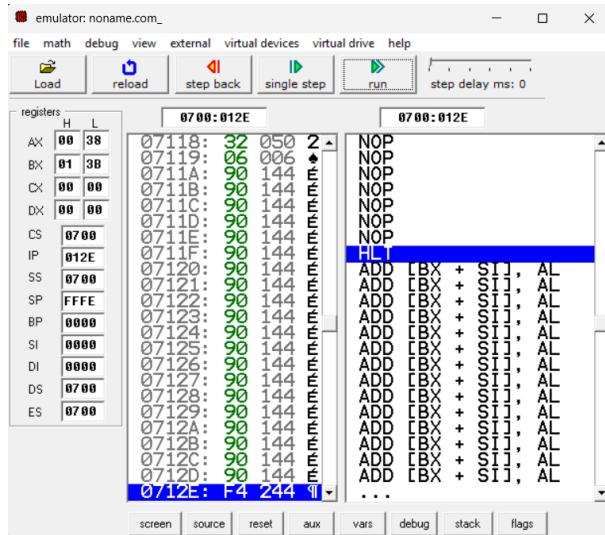
1 ORG 100h
2 MOV CX, 0
3 MOV CL, Range
4 MOV AL, Start
5
6 MOV BL, 0
7 MOV BH, 0
8
9 START_LOOP:
10    ADD BL, AL
11
12    JNC CONTINUE
13    INC BH
14
15 CONTINUE:
16    INC AL
17    LOOP START_LOOP
18
19 Start DB 50
20 Range DB 6
21 END

```

Listing 2: Range Sum Calculator

### Test Case:

- **Input:** Start = 50, Range = 6
- **Calculations:**  $50 + 51 + 52 + 53 + 54 + 55 = 315$
- **Expected Output:** BX = 013Bh (BH = 01h, BL = 3Bh = 315 decimal)



### Verification:

- Result in BX: 013Bh (315 decimal)
- No overflow occurs: BH increments only if BL exceeds 255
- Final check:  $50+51+52+53+54+55 = 315$

## Problem Three: 10-Byte Addition with Carry

### Assembly Code:

```

1 ORG 100h
2
3 NUM1 DB 0C7h, 96h, 47h, 4Eh, 46h, 82h, 0Fh, 22h, 0BDh, 0CFh
4 NUM2 DB 0Dh, 0EFh, 07h, 6Dh, 0BAh, 7Ch, 1Eh, 6Bh, 00h, 40h
5
6 ORG 500h
7 RESULT DB 11 DUP(0)
8
9 ORG 100h
10 MOV SI, OFFSET NUM1
11 MOV DI, OFFSET NUM2
12 MOV BX, 500h
13 MOV CX, 10
14 CLC
15
16 ADD_LOOP:
17     MOV AL, [SI]
18     ADC AL, [DI]
19     MOV [BX], AL
20     INC SI
21     INC DI
22     INC BX

```

```
23  LOOP ADD_LOOP  
24  
25      MOV AL, 0  
26      ADC AL, 0  
27      MOV [BX], AL  
28      HLT
```

Listing 3: 10-Byte Addition with Carry

### Test Case:

#### Input Numbers (LSB first):

- **NUM1:** C7 96 47 4E 46 82 0F 22 BD CF
- **NUM2:** 0D EF 07 6D BA 7C 1E 6B 00 40

#### Expected Result:

- **Sum:** 0F BD 8D 2D FF 00 BB 4F 85 D4
- **Final Carry:** 01

### Memory Output at 500h:

Address	Data
0500h:	0Fh
0501h:	BDh
0502h:	8Dh
0503h:	2Dh
0504h:	FFh
0505h:	00h
0506h:	BBh
0507h:	4Fh
0508h:	85h
0509h:	D4h
050Ah:	01h (final carry)

### Verification:

- **Addition performed correctly:** Each byte added with carry from previous
- **Carry propagation handled:** Final carry stored at 50Ah
- **Result matches example:**

## Conclusion

All three programs were successfully implemented and tested in emu8086. Problem One calculates factorials, Problem Two sums ranges with overflow handling, and Problem Three performs 10-byte addition with carry storage. Each program behaves as expected for the given test cases.

The screenshot shows a CPU emulator interface with the title "emulator: p3.com". The menu bar includes file, math, debug, view, external, virtual devices, virtual drive, help, and a toolbar with Load, reload, step back, single step, run, and step delay ms: 0.

The registers window shows the following values:

	H	L
AX	3B	00
BX	05	0A
CX	08	00
DX	1E	7C
CS	0700	
IP	013D	
SS	0700	
SP	0000	
BP	00CF	
SI	000A	
DI	0114	
DS	0700	
ES	0000	

The assembly code window displays the following instructions:

```
0712C: 8A 138 ē     MOV AL, [SI]
0712D: 04 004       ADC AL, [DI]
0712E: 12 018       MOV [BX], AL
0712F: 05 005       INC SI
07130: 88 136       INC DI
07131: 07 007       INC BX
07132: 46 070       LOOP 012Ch
07133: 47 071       MOV AL, 00h
07134: 43 067       ADC AL, 00h
07135: E2 226       MOV [BX], AL
07136: F5 245       HLT
07137: B0 176       NOP
07138: 00 000       NOP
07139: 14 020       NOP
0713A: 00 000       NOP
0713B: 88 136       NOP
0713C: 07 007       NOP
0713D: F4 244       NOP
0713E: 90 144       NOP
0713F: 90 144       NOP
07140: 90 144       NOP
07141: 90 144       NOP
07142: 90 144       ...
```

At the bottom of the assembly window, there are tabs for screen, source, reset, aux, vars, debug, stack, and flags.