8085 Architecture &
Its Assembly language programming

Dr A Sahu
Dept of Computer Science & Engineering
IIT Guwahati

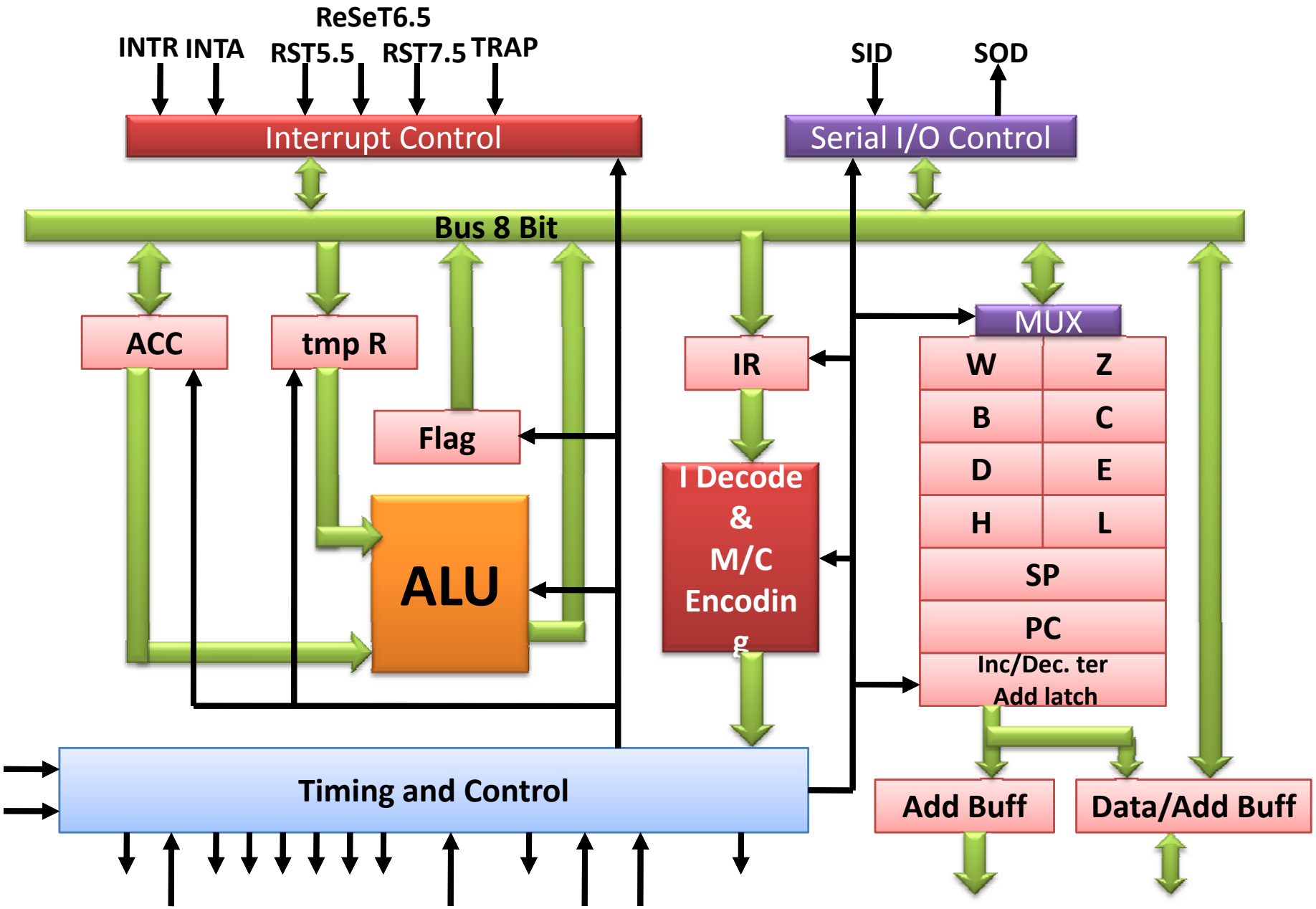
Outline

- 8085 Era and Features
- 8085
  - Block diagram (Data Path)
  - Bus Structure
  - Register Structure
- Instruction Set of 8085
- Sample program of 8085
- Simulator & Kit for 8085

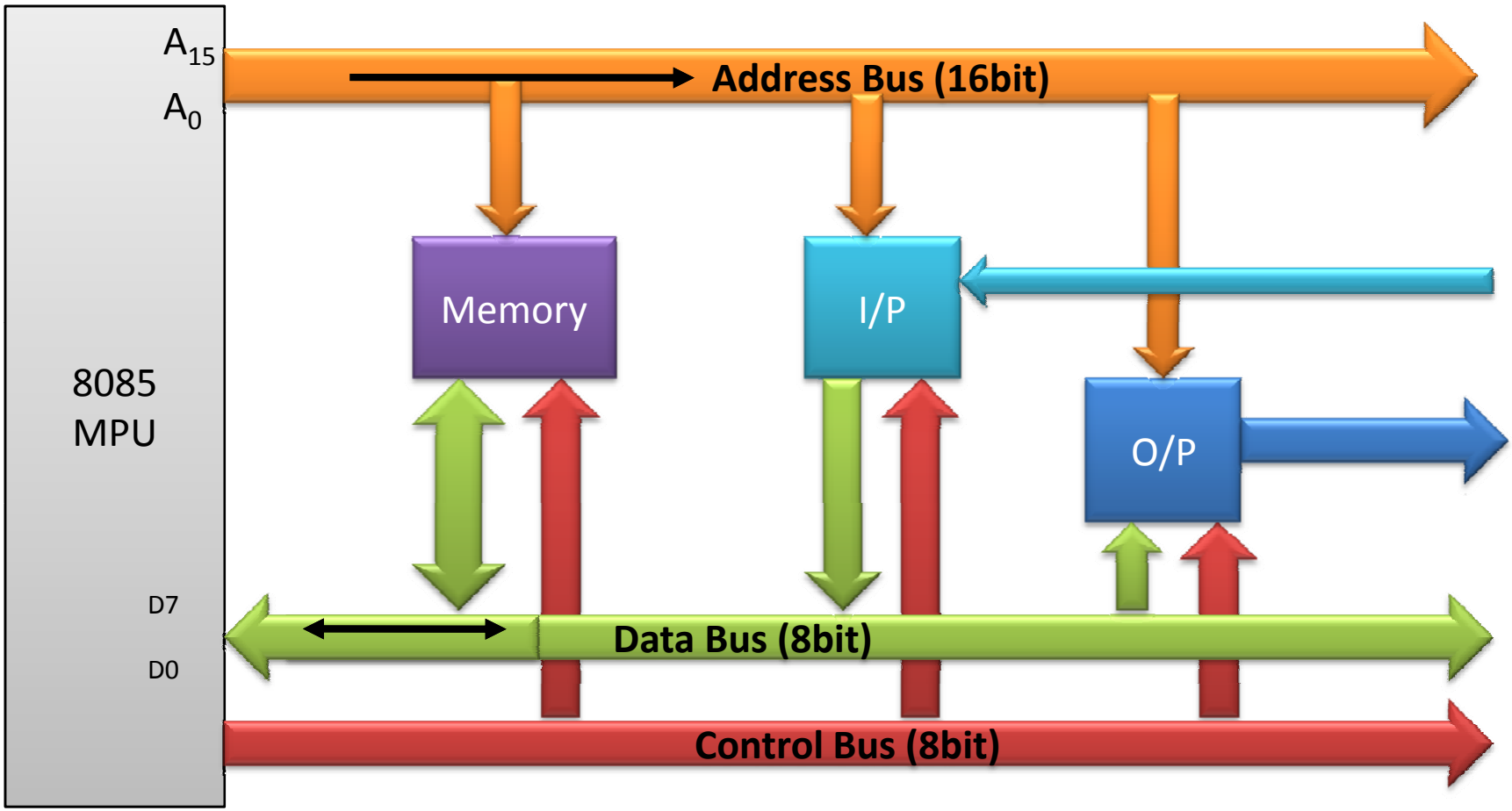
8085 Microprocessor

- 8 Bit CPU
- 3-6Mhz
- Simpler design: Single Cycle CPU
- ISA = Pre x86 design (Semi CISC)
- 40 Pin Dual line Package
- 16 bit address
- 6 registers: B, C, D, E, H,L
- Accumulator 8 bit

8085 Microprocessor Architecture



The 8085 Bus Structure

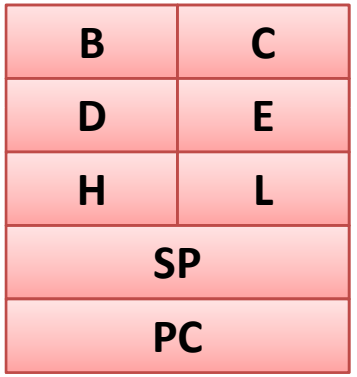


8085 Bus Structure

- Address Bus : Consists of 16 address lines: $A_0 – A_{15}$
  - Address locations: 0000 (hex) – FFFF (hex)
  - Can access 64K (= 2^{16}) bytes of memory, each byte has 8 bits
  - Can access 64K $\times$ 8 bits of memory
  - Use memory to map I/O, Same instructions to use for accessing I/O devices and memory
- Data Bus : Consists of 8 data lines: $D_0 – D_7$
  - Operates in bidirectional mode
  - The data bits are sent from the MPU to I/O & vice versa
  - Data range: 00 (hex) – FF (hex)
- Control Bus:
  - Consists of various lines carrying the control signals such as read / write enable, flag bits

8085 Registers

- Registers:
  - Six general purpose 8-bit registers: B, C, D, E, H,L
  - Combined as register pairs to perform 16-bit operations: BC, DE, HL
  - Registers are programmable (load, move, etc.)
- Stack Pointer (SP)
- Accumulator & Flag Register
  - (Zero, Sign, Carry, Parity, AuxCarry)
- Program Counter (PC)
  - Contains the memory address (16 bits) of the instruction that will be executed in the next step.



How instruction executed

- All instructions (of a program) are stored in memory.
- To run a program, the individual instructions must be read from the memory in sequence, and executed.
  - Program counter puts the 16-bit memory address of the instruction on the address bus
  - Control unit sends the Memory Read Enable signal to access the memory
  - The 8-bit instruction stored in memory is placed on the data bus and transferred to the instruction decoder
  - Instruction is decoded and executed

Instruction Set of 8085

- Arithmetic Operations
  - add, sub, inr/dcr
- Logical operation
  - and, or, xor, rotate, compare, complement
- Branch operation
  - Jump, call, return
- Data transfer/Copy/Memory operation/IO
  - MOV, MVI, LD, ST, OUT

Copy/Mem/IO operation

- MVI R, 8 bit *// load immediate data*
- MOV R1, R2 *// Example MOV B, A*
- MOV R M *// Copy to R from 0(HL Reg) Mem*
- MOV M R *// Copy from R to 0(HL Reg) Mem*

- LDA 16 bit *// load A from 0(16bit)*
- STA 16 bit *// Store A to 0(16bit)*
- LDAX Rp *// load A from 0(Rp), Rp=RegPair*
- STAX Rp *// Store A to 0(Rp)*
- LXI Rp 16bit *// load immediate to Rp*

- IN 8bit *// Accept data to A from port 0(8bit)*
- OUT 8 bit *// Send data of A to port 0(8bit)*

Arithmetic Operation

- ADD R *// Add A = A + B.reg*
- ADI 8bit *// Add A= A + 8bit*
- ADD M *// Add A=A + 0(HL)*

- SUB R *// Sub A = A -B.reg*
- SUI 8bit *// Sub A= A - 8bit*
- SUB M *// Sub A=A - 0(HL)*

- INR R *// R = R+1*
- INR M *// 0(HL)=0(HL)+1*
- DCR R *// R = R-1*
- DCR M *// 0(HL)=0(HL)-1*
- INX Rp *// Rp=Rp+1*
- DCX Rp *// Rp=Rp-1*

Other Operations

- Logic operations
  - ANA R ANI 8bit ANA M
  - ORA, ORI, XRA, XRI
  - CMP R // compare with R with ACC
  - CPI 8bit // compare 8 bit with ACC
- Branch operations
  - JMP 16bit, CALL 16 bit
  - JZ 16bit, JNZ 16bit, JC 16bit, JNC 16 bit
  - RET
- Machine Control operations
  - HLT, NOP, POP, PUSH

Assumption

- RAM Memory is interfaced
- Instructions are stored in memory
- One I/O display port is interfaced to display data of ACC

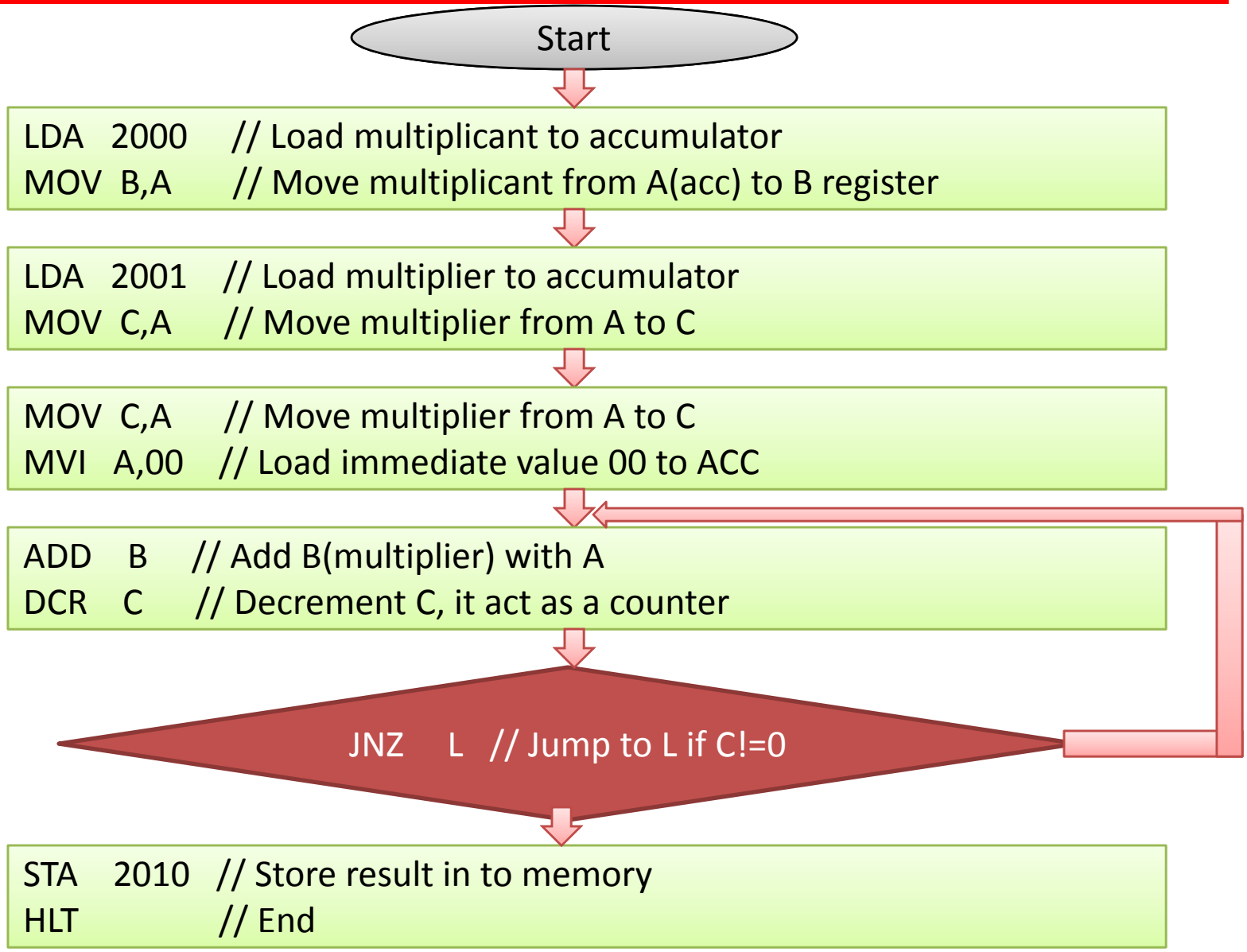
Simple Assembly Program

```
MVI   A, 24H      // load Reg  ACC with 24H
MVI   B , 56H     // load Reg B with 56H
ADD   B           // ACC= ACC+B
OUT 01H           // Display ACC contents on port 01H
HALT              // End the program
```

Result: 7A (All are in Hex)

DAA operation for Decimal Adjust A+6=10H

Flowchart to multiply two number



Code to multiply two number

```
    LDA   2000     // Load multiplicant to accumulator
    MOV  B,A       // Move multiplicant from A(acc) to B register
    LDA   2001     // Load multiplier to accumulator
    MOV  C,A       // Move multiplier from A to C
    MVI   A,00     // Load immediate value 00 to a
L:  ADD    B       // Add B(multiplier) with A
    DCR    C       // Decrement C, it act as a counter
    JNZ    L       // Jump to L if C reaches 0
    STA    2010    // Store result in to memory
    HLT            // End
```

Factorial of a Program

```
        LXI SP, 27FFH ; Initialize stack pointer
        LDA 2200H ; Get the number
        CPI 02H ; Check if number is greater than 1
        JC LAST
        MVI D, 00H ; Load number as a result
        MOV E, A
        DCR A
        MOV C,A ; Load counter one less than number
        CALL FACTO ; Call subroutine FACTO
        XCHG ; Get the result in HL  // HL with DE
        SHLD 2201H ; Store result in the memory   // store HL at 0(16bit)
        JMP END
LAST:   LXI H, 000lH ; Store result = 01
END:    SHLD 2201H
        HLT
```

Sub Routine for FACTORIAL

```
FACTO:   LXI H, 0000H
         MOV B, C ; Load counter
BACK:    DAD D   // double add ; HL=HL+DE
         DCR B
         JNZ BACK ; Multiply by successive addition
         XCHG ; Store result in DE // HL with DE
         DCR C ; Decrement counter
         CNZ FACTO ; Call subroutine FACTO
         RET ; Return to main program
```

8085 Simulator & Kit

- 8085 Simulator is available
  - Course website
- 8085 Kit is available in HW Lab (CS422)
  - First test the program on Simulator and then go for the HW
  - Sometime Kit have Driver, IDE and Assembler