CSE 331/EEE 332 (Microprocessor Interfacing & Embedded System Lab)

Lab 07 : Memory Segmentation – Stack
Instructions: PUSH, POP
Procedure
Lab Instructor : Rokeya Siddiqua

Topics to be covered in class today:

- Memory segmentation – Stack
- Instructions: PUSH, POP

The stack holds data temporarily and stores return addresses used by procedures. Stack memory is *LIFO* (last-in, first-out) memory which describes the way data are stored and removed from the stack. Data are placed on the stack with a **PUSH** instruction; removed with a **POP** instruction. Stack memory is maintained by two registers:
segment register (SS)
pointer/offset register (SP)

PUSH Operation:
Whenever a *WORD* of data is pushed onto the stack, the high-order 8 bits are placed in the location addressed by SP – 1. The low-order 8 bits are placed in the location addressed by SP – 2. The SP is decremented by 2 so the next word is stored in the next available stack location.

POP Operation:
When data are popped from the stack, the low-order 8 bits are removed from the location addressed by SP. The higher order 8 bits are removed from location addressed by SP+1. Finally, the SP register is incremented by 2.

Note: PUSH and POP store or retrieve WORD of data—never BYTE—in 8086.

Initializing the Stack: .Stack 100H

Three forms of the PUSH and POP instructions:

1. *Register addressing* allows contents of any 16-bit register to transfer to & from the stack.
2. *Memory-addressing* PUSH and POP instructions store contents of a 16- or 32 bit memory location on the stack or stack data into a memory location.
3. *Immediate addressing* allows immediate data to be pushed onto the stack, but not popped off the stack.

Task: Write an ASM code containing 3 Procedures named *main*, *addTwoNumbers* and *subTwoNumbers*. The " *addTwoNumbers*" and " *subTwoNumbers*" procedures perform addition and subtraction operations on two numbers obtained from the stack and push their results back onto the stack. In the *main* procedure, two values are pushed onto the stack. Subsequently, the *main* procedure calls the " *addTwoNumbers*" and " *subTwoNumbers*" procedures. Finally, the *main* procedure retrieves the two results from the stack and stores them in the AX and BX registers.