# Stack

# Covering topics

- Overview
- Stack
- PUSH and PUSHF, POP and POPF instructions
- A stack application

# Overview

- Stack segment of a program is used for temporary storage of data and addresses, so we will study how to manipulate with stack.
- What are POP and PUSH instructions?

# The Stack

- A stack is one-dimensional data structure.
- Items can be add and remove from one end; that is, it is processed in a LIFO manner.
- The most recent item addition is called top of the stack
- A program must set aside a block of memory to hold up the stack. As
  .STACK 100h
- When program is assembled and loaded into memory, SS contains segment number of the stack segment. In above statement the SP is initialized to 100h. Which represent the empty stack position. When the stack is not empty SP contains the offset address of the top of the segment.

# PUSH and PUSHF

- When we wants to add a new word into the stack we push it on using following syntax:

  PUSH SOURCE

  where SOURCE is a register or a memory word.

For example

PUSH AX this line when executed causes following:

1=> SP is decreased by 2

2=> A copy of the AX contents is moved into the memory location specified by the SS:SP, while the source is unchanged.

- PUSHF has no operand and pushes the FLAGS register contents into the stack.

# POP and POPF

- To remove the top item from the stack, we POP it using syntax:

  POP destination

  where destination is a 16 bit register (except IP) or memory word.

  For Example:

  POP BX when executed cause;

  1=> The contents of SS:SP(top of the stack) is moved into the destination.

  2=> SP is increased by 2
- The POPF pops the top of the stack into the FLAGS register.
- There is no effect of PUSH,PUSHF,POP,POPF on the FLAGS.

# A Stack Application

- Stack behaves in LEFO manner, the order that comes off the stack is the reverse of the order the entered in. We will demonstrate the same concept using following program that read a sequence of characters and display it in reverse order.

```
.MODEL  SMALL
.STACK  100h
.DATA
   MSG  DB "Enter a sequence of
character press enter to end reading",
0DH,0AH,  "$"
.CODE
MAIN PROC
MOV AX,@DATA
MOV DS,AX
MOV AH,9
LEA DX,MSG
INT 21h
XOR CX,CX   ;initialize cx=0
MOV AH,1
INT 21h
WHILE:
 CMP    AL,0DH
 JE     END_WHILE
 PUSH   AX
 INC    CX
 INT    21h
 JMP    WHILE
END_WHILE:
MOV     AH,2
MOV     DL,0DH
INT     21h
MOV     DL,0AH
INT     21h
JCXZ    EXIT    ;jump if no character reads
TOP:
   POP     DX
   INT     21h
   LOOP    TOP

EXIT:
   MOV     AH,04CH
   INT     21H
MAIN ENDP
END MAIN
```

# References


- Assembly Language Programming and Organization of the IBM PC (Ytha Yu, Charles Marut)