The Stack and Introduction to procedures

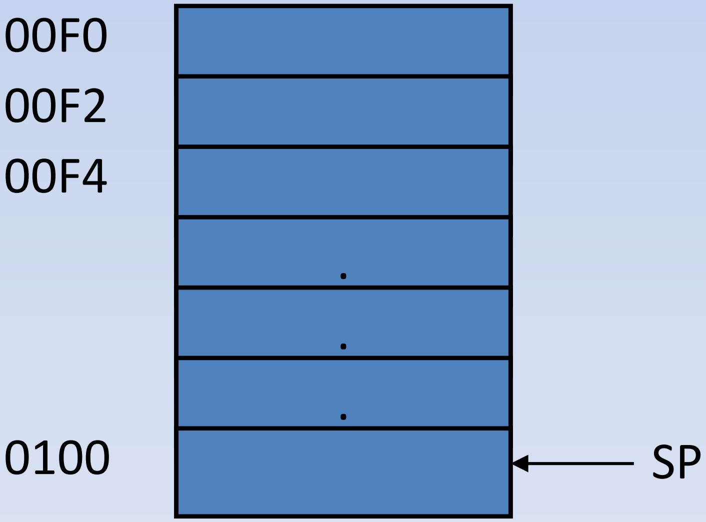
Reference: Assembly Language Programming and Organization of the IBM PC – Charles Marut – Chapter 8

The Stack

- A stack is a one dimensional data structure.
- Items are added and removed from one end of the structure in a last in first out manner.
- The most recent addition to the stack is called the **top of the stack**.
- Stack instructions are PUSH, PUSHF, POP and POPF.
- There is no effect of stack instructions on flags.

Stack Instructions

- PUSH source
- Source is a 16 bit register or memory word.
- PUSH AL > Illegal



SP=0100

AX=1234

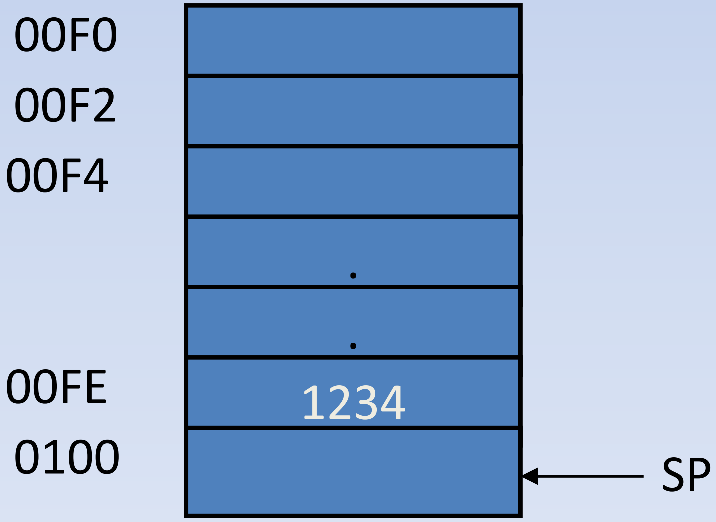
Stack Instructions

The stack grows towards the beginning of the memory

- PUSH AX

➢ SP is decreased by 2

➢ A copy of the source content is moved to the address specified by SS:SP. The source is unchanged.



SP=00FE

AX=1234

Initially SP contains the offset address of the memory location immediately following the stack segment

Stack Instructions

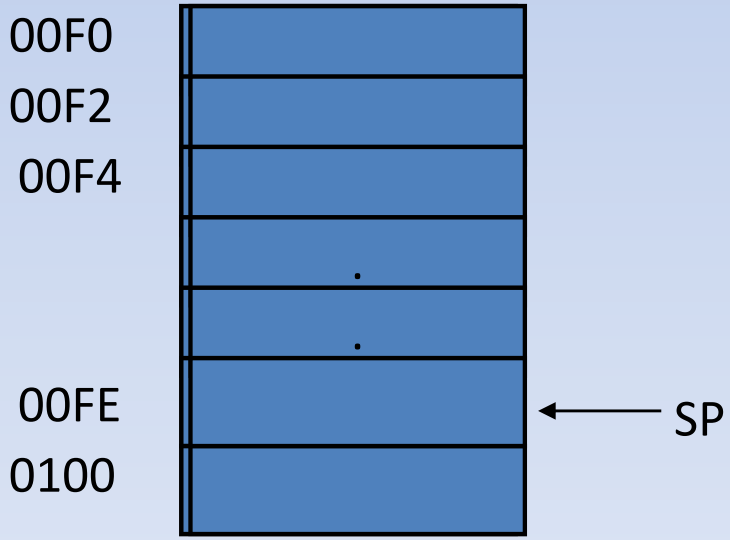
- PUSHF
- Pushes the contents of the flag register onto the stack.
- POP destination
- Destination is a 16 bit register (except IP) or memory word.

Stack Instructions

- POP BX

➢ The content of SS:SP (top of the stack) is moved to the destination

➢ SP is increased by 2



00F0

00F2

00F4

.

.

00FE

0100

SP

SP=0100

BX=1234

- POPF pops the top of the stack into the flags register.

Procedures

- name PROC type

```
      ;body of the procedure
      RET
  name ENDP
```

- Type (near or far) is optional
- Near: the statement that calls the procedure is in the same segment as the procedure itself.
- Far: the statement that calls the procedure is in a different segment.

Procedures

- The RET instruction causes control to transfer back to the calling procedure.
- Every procedure should have a RET someplace.
- Usually it is the last statement in the procedure.

CALL and RET

- To invoke a procedure, the CALL instruction is used.
- There are two kinds of procedure calls, direct and indirect.
- CALL name (direct)
- CALL address expression (indirect)
- Address expression specifies a register or memory location containing the address of a procedure.

Executing a CALL Instruction

- The return address of the calling program is saved on the stack. This is the offset of the next instruction after the CALL statement.
- IP gets the offset address of the first instruction of the procedure. This transfers control to the procedure.

Before CALL

CODE SEGMENT

OFFSET Address	Instruction
	MAIN PROC
0010H	CALL PROC1
0012H	NEXT INSTRUCTION
	....
	PROC PROC
0200H	FIRST INSTRUCTION
0300H	RET

IP

STACK SEGMENT

OFFSET Address	Instruction
00FCH	
00FEH	
0100H	

SP

AFTER CALL

CODE SEGMENT

	OFFSET Address	Instruction
		MAIN PROC
	0010H	CALL PROC1
	0012H	NEXT INSTRUCTION
		……….
		PROC PROC
IP →	0200H	FIRST INSTRUCTION
	0300H	RET

STACK SEGMENT

	OFFSET Address	Instruction
	00FCH	
SP →	00FEH	0012
	0100H	

Executing a RET Instruction

- RET pop value
- The integer argument pop value is optional.
- For a NEAR procedure, execution of RET causes the stack to be popped into IP.
- If a pop value N is specified, it is added to SP and thus has the effect of removing N additional bytes from the stack.
- CS:IP now contains the segment:offset of the return address and control returns to the calling program.

BEFORE RET

CODE SEGMENT

OFFSET Address	Instruction
	MAIN PROC
0010H	CALL PROC1
0012H	NEXT INSTRUCTION
	……….
	PROC PROC
0200H	FIRST INSTRUCTION
0300H	RET

IP

STACK SEGMENT

SP

OFFSET Address	Instruction
00FCH	
00FEH	0012
0100H	

AFTER RET

CODE SEGMENT

OFFSET Address	Instruction
	MAIN PROC
0010H	CALL PROC1
0012H	NEXT INSTRUCTION
	……….
	PROC PROC
0200H	FIRST INSTRUCTION
0300H	RET

IP

STACK SEGMENT

OFFSET Address	Instruction
00FCH	
00FEH	
0100H	

SP

Example: Write an assembly code that will count the number of 1’s in the content of AX register.

```
MOV AX,3H  ; Value to be counted
CALL PROC1
HLT
PROC1 PROC
MOV BX, 0 ; Initialize Counter
START: CMP AX, 0 ; Check for ending
          JZ END
          TEST AX, 1 ; Check for 1
           JZ BELOW
            INC BX ; Increment count
BELOW: SHR AX, 1
             JMP START
END: RET
PROC1 ENDP
```

Thanks…….