Lec 13

November 17, 2022

ARM 7 - Instruction SET IV

(Stack Related operations)

Push

- STMFA SP! {R0 - R4}
  STMFA R13! {R0 - R4}
- STMFD r13! {r0 - r4}
- STMEA r13! {r0 - r4}
- STMED r13! {r0 - r4}

POP

- LDMFA r13! {r0 - r4}
- LDMFD r13! {r0 - r4}
- LDMEA r13! {r0 - r4}
- LDMED r13! {r0 - r4}

Push operation is keeping something in the new stack. The stack is in the memory. So Push operation is keeping something in the memory. In short Push is a store operation.

On the hand Pop is getting/retriving something from the stack. or we can directly say. Pop operation is getting something from the memory and that is called load operation.

Push → store (ST)

Pop → Load. (LD)

'M' stands for multiple.

ARM7 allows multiple push/pop operation at a time.

we can push one register or two register value or many values as our requirement.

ST → store (push)

LD → load (pop)

M → multiple

if we push value we write

STM -- SP! {R0}

if we push multiple register values we write

STM -- SP! {R0, R1, R3, R7}

or STM -- SP! {R0, R1}

or STM -- SP! {R0, R7, R3}

or STM -- SP! {R0, R1, R2, R3, R4}

↓

or STM -- SP! {R0-R4}

The address of the top of stack is stored in a SP register. ! indicates that SP or R13 register will give the address.

There are ~~16 regis~~ 16 general purpose registers. Out of which (R0 to R15)

R13 → SP

R14 → LR

R15 → PC.

we can write SP! or R13!

Both are ways are same with different way of writing.

'F' stands for Full

'E' stands for Empty

'A' stands for ascending

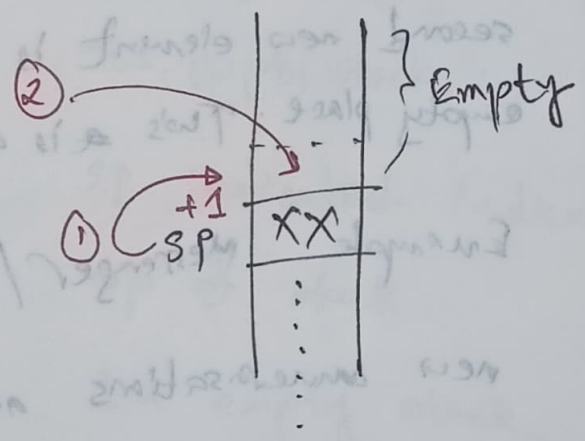
'D' Stands for Descending

There are two types of stack operation in processors.

8086 → sp gets decremented in ~~after~~ push operation.

~~808~~ 8051 → sp gets incremented in push operation.

STMFA



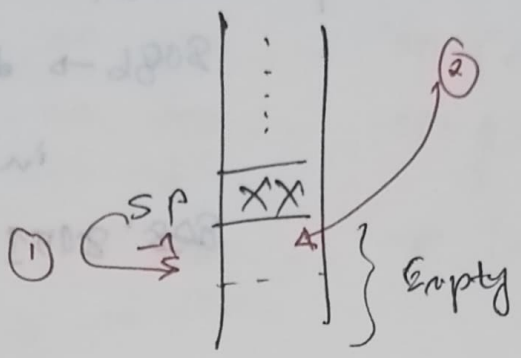
sp indicates the top of ~~the~~ stack.

first sp gets incremented.

second new element is stored in empty place. This is an ascending stack.

Example: when new messages comes in our mobile it is stored ~~step of~~ top of row. So this is ascending stack.

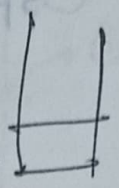
STMFD



first sp gets decremented.
second new element is stored in the empty place. This is a descending stack.

Example: Messenger / whatsapp chat new conversations are shown in last of the row.

sp starts from zero location.



when we push any value to stack the value will be stored from location 1.
So there is wasted one location.

There are two type of stack
full and empty.

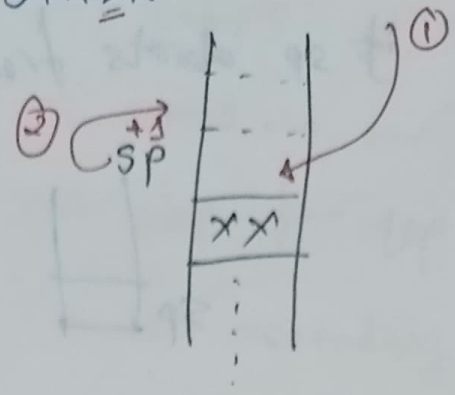
Ex The full one we have already been taught. [STMFA / STMFD] sp → top of stack.

The empty stack is
sp ~~indicated~~ indicates above the top of stack.

In empty stack the new element is stored first in sp location. Then sp gets incremented or decremented.

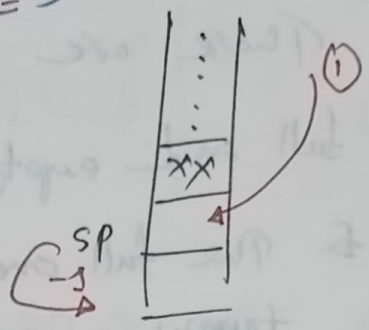
Using this empty method, sp
the first location of the stack is not wasted.

STMEA



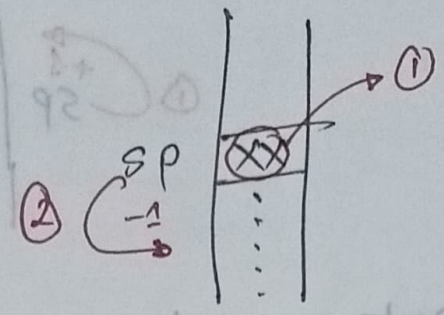
first, new element is stored in sp location.
second sp gets incremented.

STMED



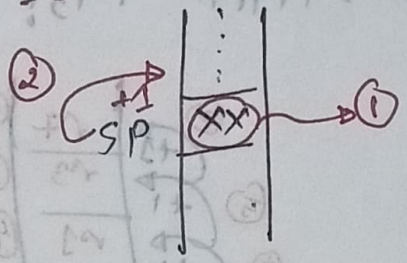
first, new element is stored in sp location
second, sp gets decremented.

Pop operation # LDMFA



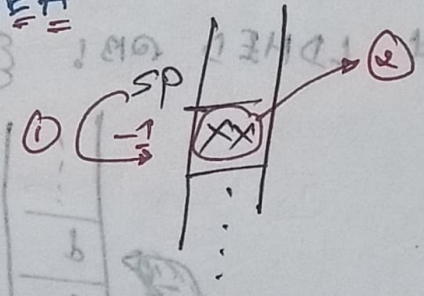
first, top element will be retrived.
second sp gets decremented.

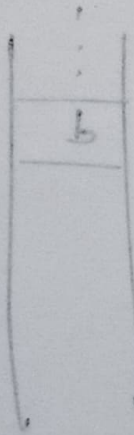
LDMFD



first, top element will be retrived.
second, sp will get incremented.

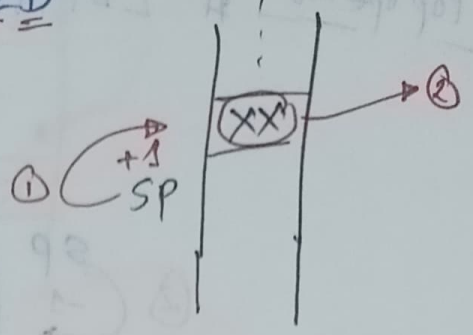
LDMEA





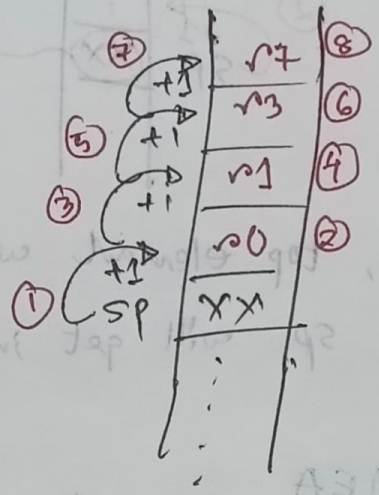
first, sp gets decremented.
second, top element will be retrived.

LDMED



first, sp gets incremented.
second, top element will be retrived.

STMFA r13! {r0, r1, r3, r7}



LDMED r13! {r0, r2, r4}