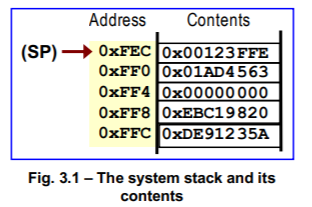
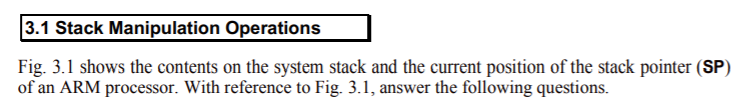
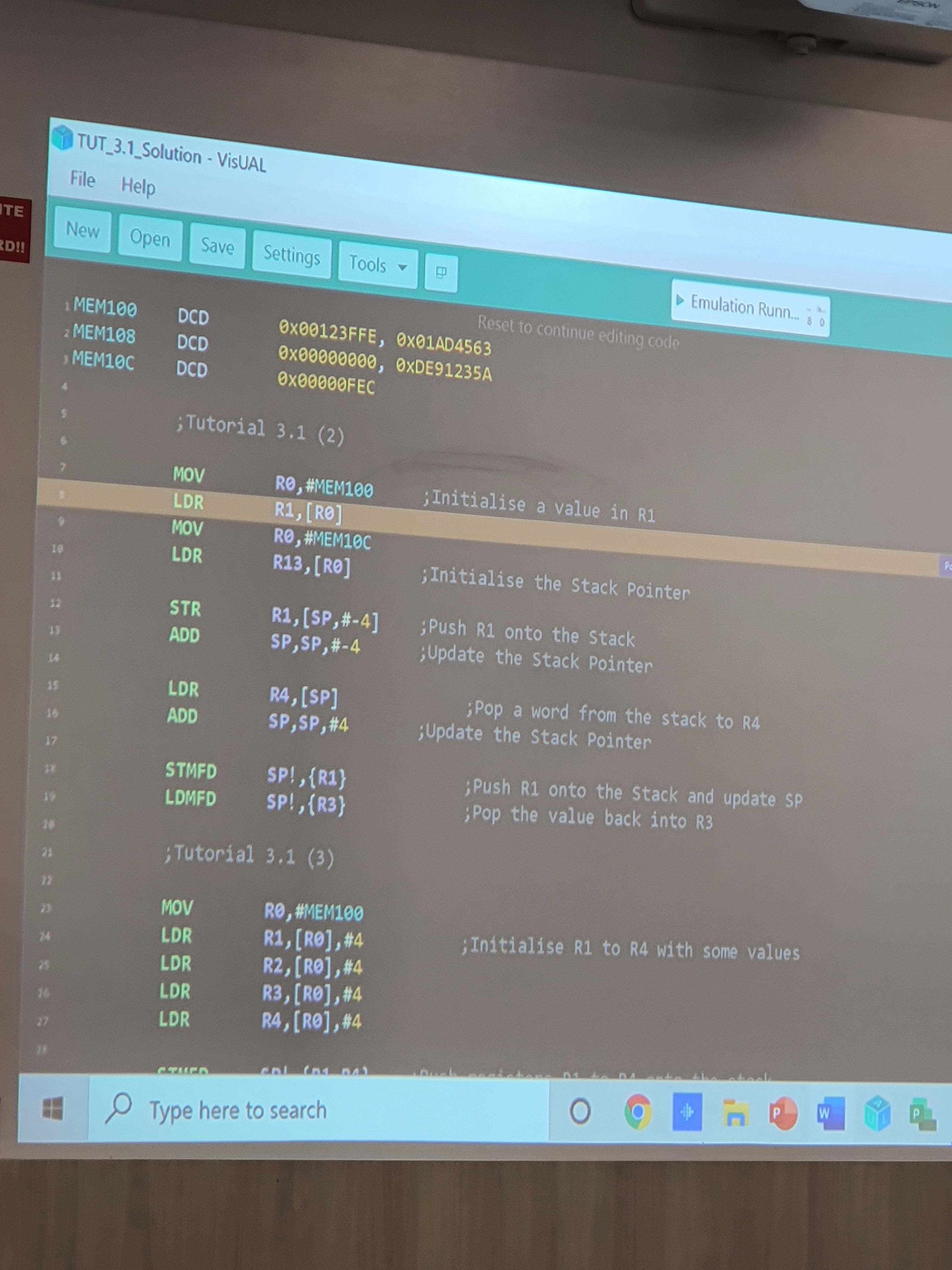
CZ1106 Tutorial THREE

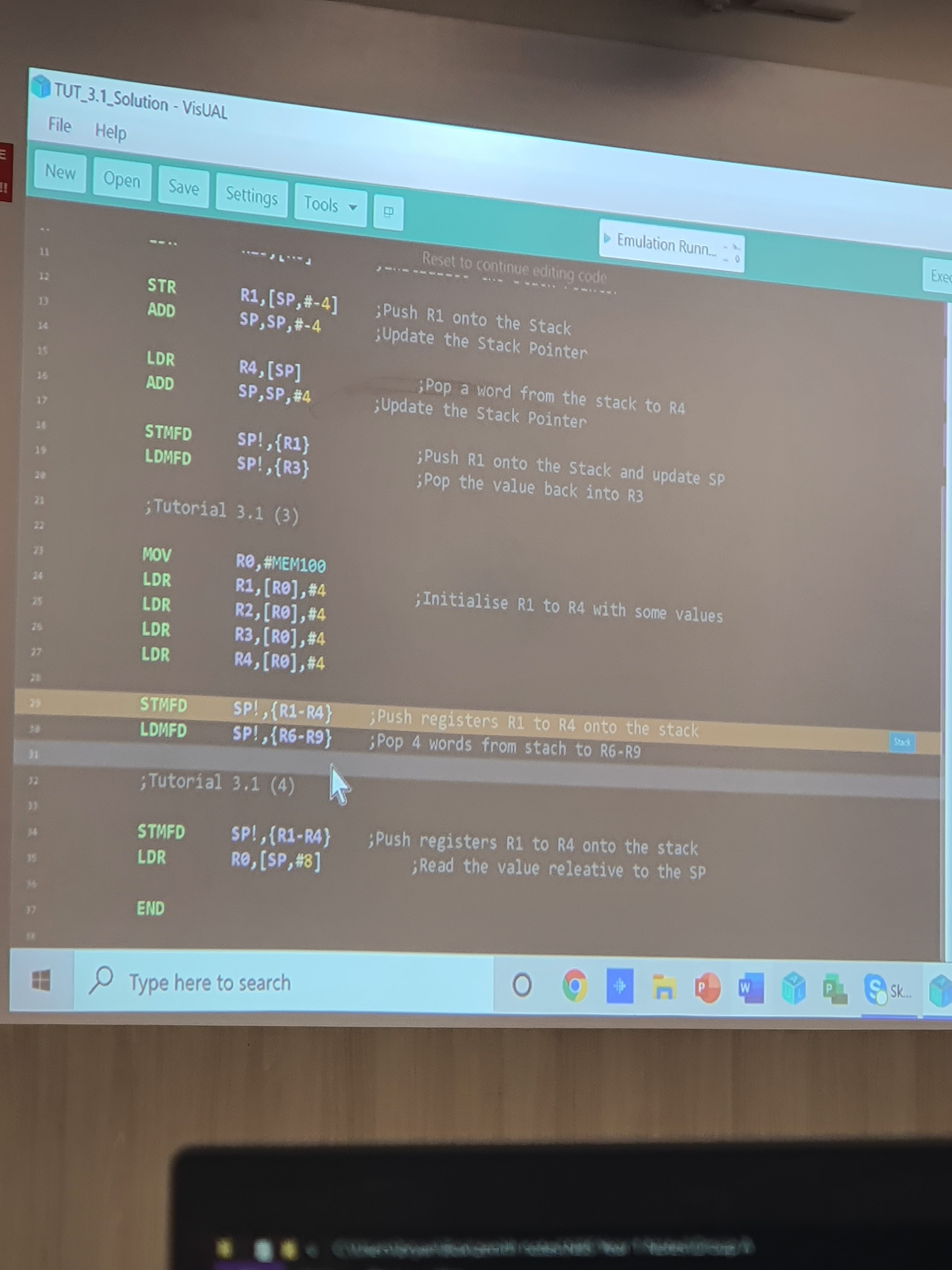
MODULAR PROGRAMMING





Stack pointer contains memory address. 0xFEC. ✅Stack Pointer points to the last item on the stack by storing its memory address.



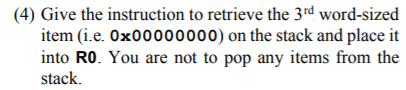




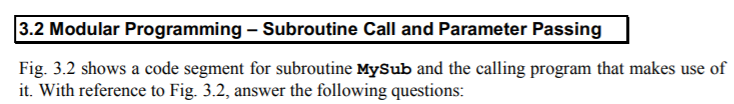
STR R1, [SP] ❌system stack --> need to move the SP after pushing! STR R1, [SP, #-4]!. STMFD SP!, {R1}

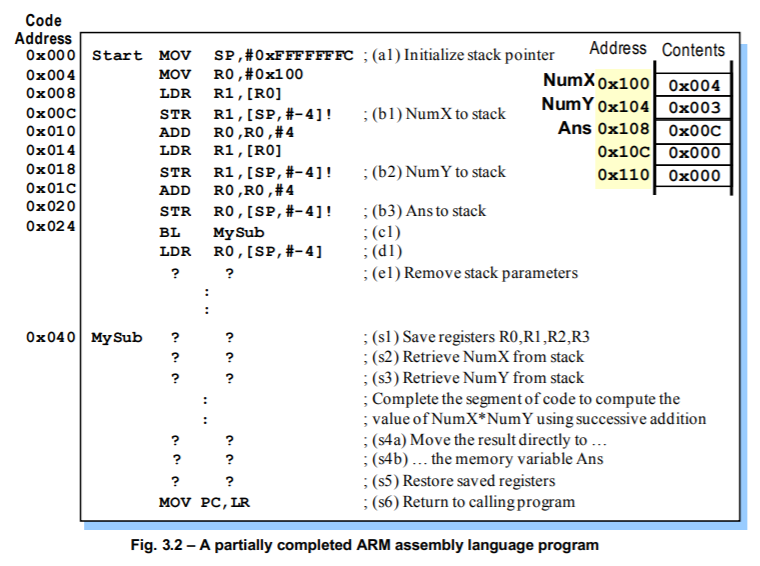


LDMFD SP!, {R6-R9} ✅



LDR R0, [SP, #-8] ❌remember: goes to higher memory address! LDR R0, [SP, #8]

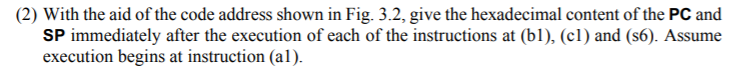






NumX and NumY are by value. They are being loaded into the register R1 by accessing their respective memory addresses (0x100 and 0x104) using LDR. Their literal values are then pushed onto the stack from register R1 using STR. ✅

Ans is by reference. Its memory address of 0x108 is being passed into the stack. ✅



(b1):

PC: 0x014  
SP: 0xFFFF FFF8 ✅

(c1):

PC: 0x040

SP: 0xFFFF FFF0 ✅

(s6):

PC: 0x028

SP: 0xFFFF FFF0 ✅



The content in 0xFFFF FFEC ❌R0 = value of R3 (can be 0)



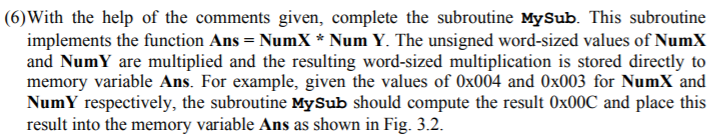
ADD SP, SP, #12 ✅



STR LR, [SP, #-4]! ;push link register into stack

BL MySub

LDR LR, [SP], #4 ;pop link register from stack ❌



MySub STMFD SP!, {R0-R3} ;(s1) save registers R0, R1, R2, R3

;R3

;R2

;R1

;R0

;[Ans]

;NumY

;NumX

LDR R0, [SP, #24] ;(s2) retrieve NumX from stack

LDR R1, [SP, #20] ;(s3) retrieve NumY from stack

;compute NumX\*NumY using successive addition

;add R0 to R3 for R1 times

MOV R3, #0

loop ADD R3, R3, R0

SUBS R1, R1, #1

BNE loop

;(s4a) move the result directly to

;(s4b) the memory variable Ans

LDR R2, [SP, #16]

STR R3, [R2]

;restore saved registers

LDMFD SP!, {R0-R3}

Mov PC,LR ;(s6) return to calling program

