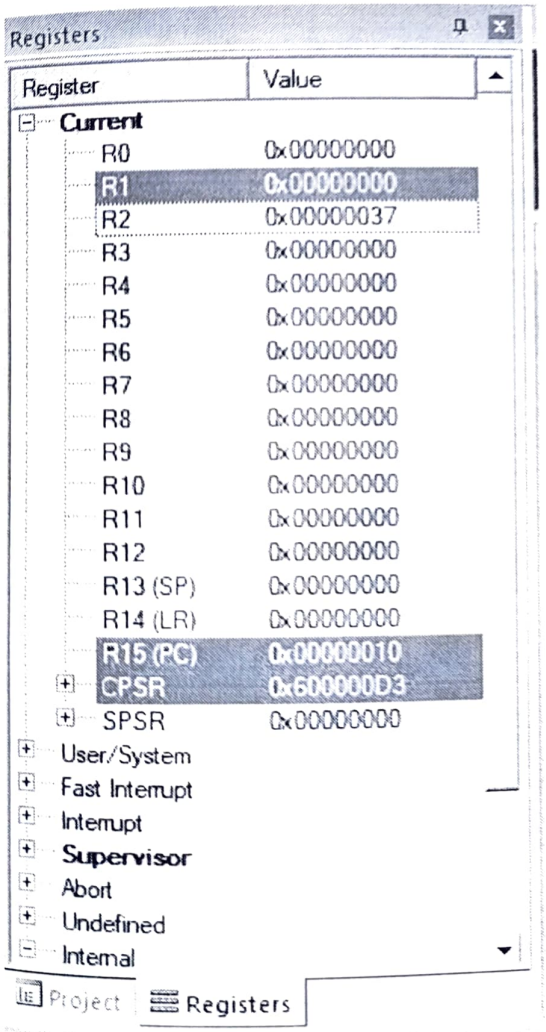
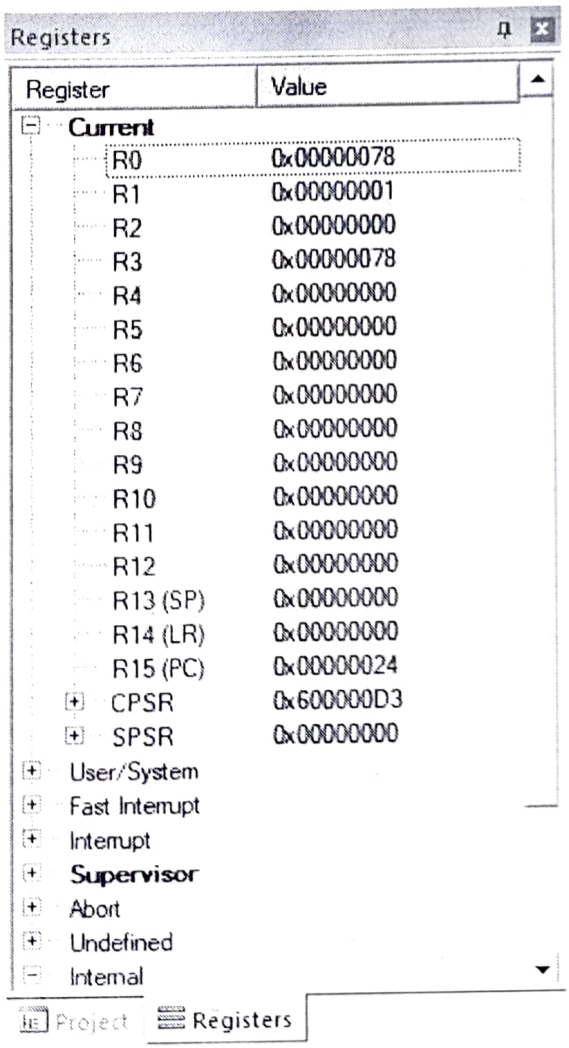
1.Write a program to find the sum of first 10 integer numbers.

LABEL FIELD	MNEMONIC FIELD	COMMENTS FILED
	AREA INTSUM, CODE, READONLY	
	ENTRY	; Mark first instruction to execute
	MOV R1,#10	; LOAD 10 TO REGISTER
	MOV R2,#0	; EMPTY R2 REGISTER TO STORE RESULT
LOOP	ADD R2,R2,R1	; ADD THE CONTERNT OF R1 WITH RESULT AT R2
	SUBS R1,#0X01	; DECREMENT R1 BY 1
	BNE LOOP	; REPEAT TILL R1 GOES TO ZERO
HERE	B HERE	
	END	



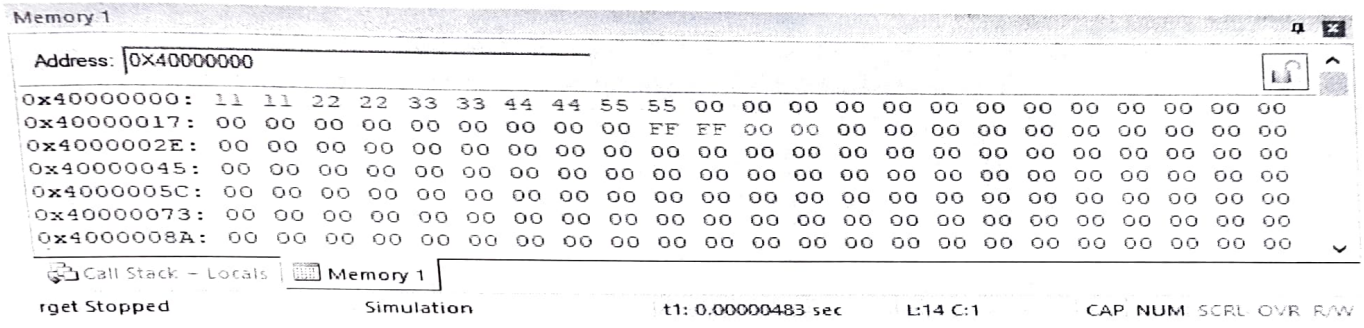
2.Write a program to find factorial of a number.

LABEL FIELD	MNEMONIC FIELD	COMMENTS FILED
	AREA FACTORIAL,CODE, READONLY	
	ENTRY	; MARK FIRST INSTRUCTION TO EXECUTE
	MOV R0, #5	; STORE FACTORIAL NUMBER IN R0
	MOV R1,R0	; MOVE THE SAME NUMBER IN R1
FACT	SUBS R1, R1, #1	; SUBTRACTION
	CMP R1, #1	; COMPARISON
	BEQ STOP	
	MUL R3,R0,R1;	; MULTIPLICATION
	MOV R0,R3	; RESULT
	BNE FACT	; BRANCH TO THE LOOP IF NOT EQUAL
STOP	NOP	
HERE	B HERE	
	END	; MARK END OF FILE



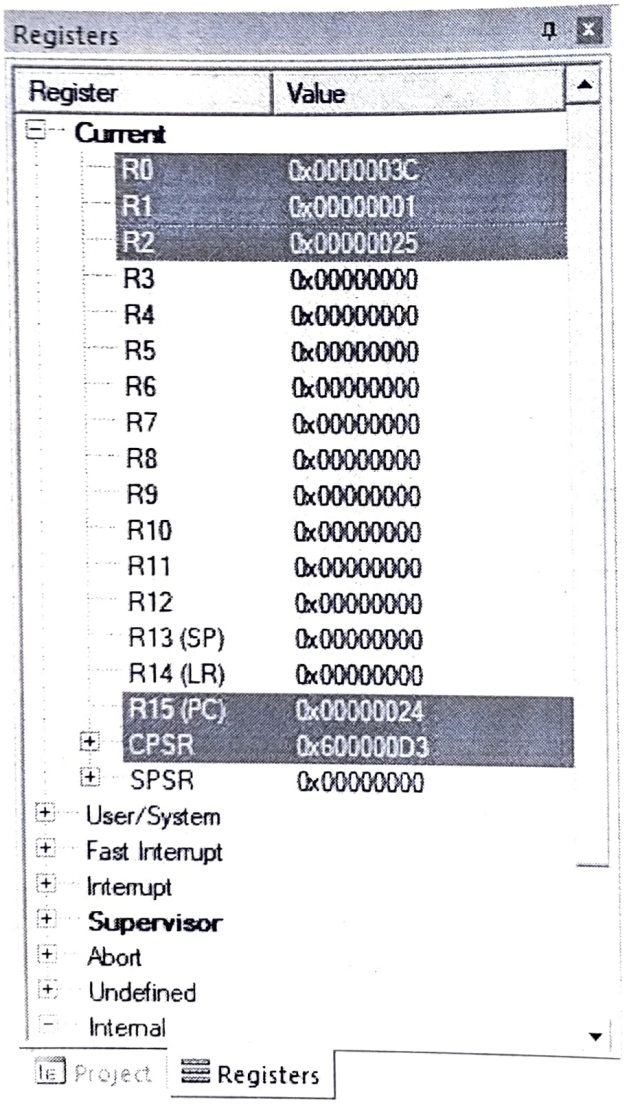
3.Write a program to add an array of 16 bit numbers and store the 32 bit result in internal RAM

LABEL FIELD	MNEMONIC FIELD	COMMENTS FILED
	AREA FACTORIAL,CODE, READONLY	
	ENTRY	; MARK FIRST INSTRUCTION TO EXECUTE
	MOV R1,#05	; COUNTER BIT FOR 5 16BIT ADDITION
	SUB R1,#01	; DECREMENTED BY 1 BECAUSE WE ADD ONLY 4 TIME
	MOV R0,#0X40000000	R0 POINTING TO 0X40000000 MEMORY LOCATION
	LDRH R2,[R0]	; LODING HALF WORD POINTED BT RO TO R2
UP	ADD R0,R0,#2	; MEMORY POINTER INCREMENTED BY 2
	LDRH R3,[R0]	; SECOND 16BIT NUMBER IS LOADED TO R3
	ADD R2,R2,R3	; ADDITION IS DONE
	SUBS R1,#01	; DECREMENTS COUNTER BIT FOR NUMBER OF ADDITION
	BNE UP	; IF COUNTER≠0 THE EXECUTION JUMPS TO THE LABEL 'UP'
	MOV R0,#0X40000020	; MEMORY LOCATION WHERE RESULT SHOULD BE SAVED
	STR R2,[R0]	; STORING OF RESULT
HERE	B HERE	
	END	; MARK END OF FILE



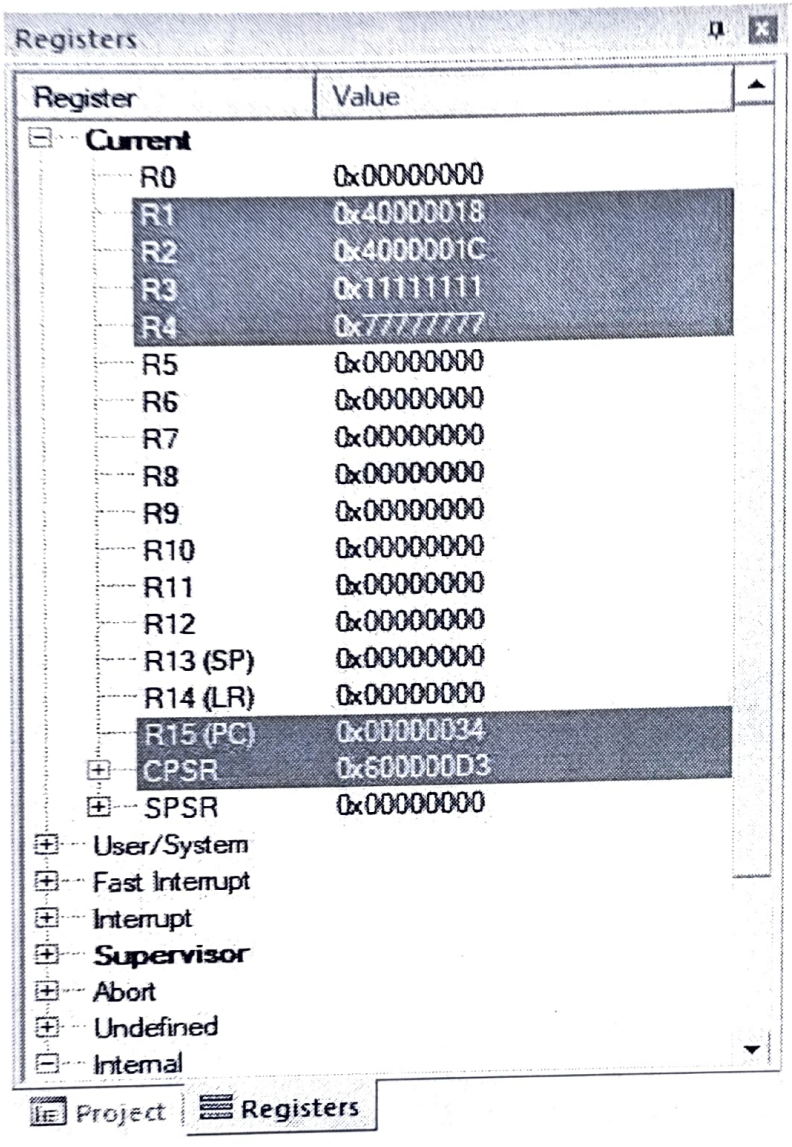
4. Write a program to find the square of a number (1 to 10) using look-up table.

LABEL FIELD	MNEMONIC FIELD	COMMENTS FILED
	AREA SQUARE,CODE, READONLY	
	ENTRY	; MARK FIRST INSTRUCTION TO EXECUTE
	LDR R0, =DATATABLE	; Load start address of Lookup table
	LDR R1,VALUE	; Load no whose square is to be find
	CMP R1,#1	; check whether it is one
	BEQ LOOP1	; if one then return the first address value as square(1)=1
LOOP2	ADD R0,R0,#4	; if not increment the address location
	SUBS R1,R1,#1	; decrement the value of the number
	CMP R1,#1	; check whether it is 1, idea here is to check for the location
	BNE LOOP2	; if not one then go to loop2 else for loop1
LOOP1	LDR R2,[R0]	; store the result in r2
HERE	B HERE	; infinite loop
	END	; MARK END OF FILE



5. Write a program to find the largest/smallest number in an array of 32 numbers

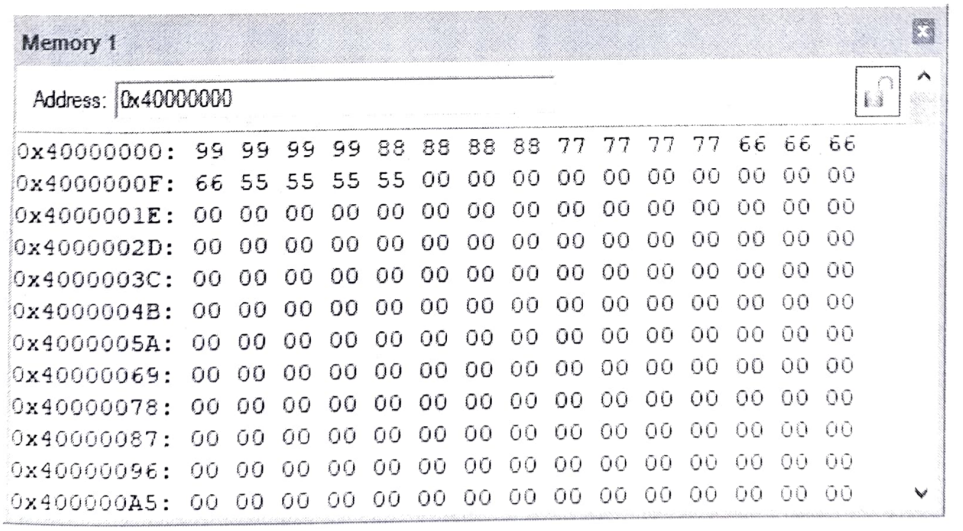
LABEL FIELD	MNEMONIC FIELD	COMMENT FIELD
	AREA LAR_SMAL, READONLY	
	ENTRY	
	MOV R5,#06	; COUNTER VALUE E.G 7 NUMBERS
	MOV R1,#0X40000000	; START OF THE DATA MEMORY
	MOV R2,#0X4000001C	; RESULT LOCATION
	LDR R3,[R1]	; GET THE FIRST DATA
	MOV R2,#0X4000001C	; RESULT LOCATION
	LDR R3,[R1]	; GET THE FIRST DATA
LOOP	ADD R1,R1,#04	; MEMORY POINTER UPDATED TO FETCH 2ND DATA
	LDR R4,[R1]	; GET SECOND NUMBER
	CMP R3,R4	; COMPARE BOTH NUMBERS
	BLS LOOP1	;BHI → for large; IF 1ST> 2ND THAN LOOP1
	MOV R3,R4	
LOOP1	SUBS R5,R5,#01	; DECREMENT THE COUNTER
	CMP R5,#00	
	BNE LOOP	
	STR R3,[R2]	
STOP	B STOP	
	END	


Registers
Register
Value
Current
R0 0x00000000
R1 0x40000018
R2 0x4000001C
R3 0x11111111
R4 0x77777777
R5 0x00000000
R6 0x00000000
R7 0x00000000
R8 0x00000000
R9 0x00000000
R10 0x00000000
R11 0x00000000
R12 0x00000000
R13 (SP) 0x00000000
R14 (LR) 0x00000000
R15 (PC) 0x00000034
CPSR 0x600D00D3
SPSR 0x00000000
User/System
Fast Interrupt
Interrupt
Supervisor
Abort
Undefined
Internal
Project
Registers

6.Write a program to arrange a series of 32 bit numbers in ascending/ descending order

LABEL FIELD	MNEMONIC FIELD	COMMENT FIELD
	AREA ASCENDING , CODE, READONLY	
	ENTRY	
	MOV R0,#05	; OUTER LOOP
OUTTERLOOP	MOV R5,#0X40000000	; DATA ADDRESS
	ADD R6,R5,#4	; INC TO CMP WITH NEXT DATA
	MOV R3,#4	; INNER LOOP
INNERLOOP	LDR R1,[R5]	; GET 1ST DATA
	LDR R2,[R6]	; GET 2ND DATA
	CMP R1,R2	; COMPARE 2 NO'S
	BLO LOOP3	; IF 1>2 THEN NO NEED TO EXCHANGE; BHI
	MOV R4,R2	; IF 1<2 THEN EXCHANGE
	MOV R2,R1	
	MOV R1,R4	
LOOP3	STR R1,[R5]	
	STR R2,[R6]	
	ADD R5,R5,#04	; INC 4 TIMES TO GET NEXT DATA FOR CMP
	ADD R6,R6,#04	
	SUBS R3,R3,#01	; DECREMENT INNER LOOP
	BNE INNERLOOP	
	SUBS R0,R0,#1	
	BNE OUTTERLOOP	; DECREMENT OUTTER LOOP
STOP	B STOP	
	END	

Before sorting for ascending



7. Write a program to count the number of ones and zeros in two consecutive memory locations

LABEL FIELD	MNEMONIC FIELD	COMMENT FIELD
	AREA ONEZERO , CODE, READONLY	
	ENTRY	;MARK FIRST INSTRUCTION TO EXECUTE
	MOV R2,#0	; COUNTER FOR ONES
	MOV R3,#0	; COUNTER FOR ZEROS
	MOV R6,#0X00000002	; LOADS THE VALUE
	MOV R1,#32	; 32 BITS COUNTER
	MOV R0,R6	; GET THE 32 BIT VALUE
	MOV R0,R6	; GET THE 32 BIT VALUE
LOOP0	MOVS R0,R0,ROR #1	; RIGHT SHIFT TO CHECK CARRY BIT (1'S/0'S)
	BHI ONES	; IF C=1 GOTO ONES BRANCH OTHERWISE NEXT
ZEROS	ADD R3,R3,#1	; IF C= 0 THEN INCREMENT THE COUNTER BY 1(R3)
	B LOOP1	; BRANCH TO LOOP1
ONES	ADD R2,R2,#1	; IF C=1 THEN INCREMENT THE COUNTER BY 1(R2)
LOOP1	SUBS R1,R1,#1	; COUNTER VALUE DECREMENTED BY 1
	BNE LOOP0	; IF NOT EQUAL GOTO TO LOOP0 CHECKS 32BIT
STOP	B STOP	
	END	

8.Write an ARM assembly program that checks if a 32-bit number is a palindrome. Assume that the input is available in r 3. The program should set r 4 to 1 if it is a palindrome, otherwise r 4 should have 0. A palindrome is a number which is the same when read from both sides. For example, 1001 is a 4 bit palindrome.

```
mov R0,1044480
mov R3,0
mov R4,31
movu R2,0x0001
mov R5,0
andu R1,R0,0xffff
.loop:
and R3,R2,R1
lsl R2,R2,1
lsl R3,R3,R4
add R5,R5,R3
sub R4,R4,2
cmp R4,0
bgt .loop
lsr R0,R0,16
lsr R5,R5,16
sub R0,R5,R0
cmp R0,0
beq .palin
mov R0,0
b .exit
.palin:
mov R0,1
.exit:
```