Name: Yash R Umale Reg No: ECE18182

S. No.	Date	Title	Page No.	Teacher's Sign / Remarks
1. a)	13/1/21	Move 5 immediate values	1	
b)	13/1/21	Find 1's complement of nos.	2	
c)	13/1/21	Find 2's complement of nos.	3.	
2. a)	23/1/21	Addition of two 32 bit nos.	4	
b)	23/1/21	Addition with carry	5	
c)	23/1/21	Add 5 nos. with carry	6	
d)	27/1/21	Add two 64 bit nos.	7	
3. a)	27/1/21	Subtract two 32 bit nos.	8	
b)	27/1/21	Subtract two nos. with borrow	9	
c)	27/1/21	Subtract two 64 bit nos.	10.	
4. a)	3/2/21	Reverse - subtraction (32 bit)	11	
b)	3/2/21	Rev-sub with borrow	12	
c)	3/2/21	Rev-subtraction (64 bit)	13.	
5)	25/2/21	Addition with conditional loop	14, 15, 16.	
6)	4/3/21	Multiplication (32 bit)	17, 18, 19.	

5) (a)(i) Write an ALP to perform addition, subtraction of two 32-bit numbers using pre/post indexing.

ARM Code:

```
        AREA SBSC, CODE, READONLY
ENTRY
        LDR R0, =0x40000000
        LDR R1, [R0], #04
        LDR R2, [R0], #04
        ADDS R3, R1, R2
        STR R3, [R0]
UP B UP
        END
```

Results:

Obtained the sum of two 32-bit numbers using post-indexing with an ARM7 assembly language program.

(ii) Write an ALP to perform subtraction of two 32-bit numbers using ~~post~~ pre-indexing.

ARM Code:

```
    AREA SBSC, CODE, READONLY
ENTRY
    LDR R0, =0x40000000
    LDR R1, [R0, #04]
    LDR R2, [R0, #08]
    SUBS R3, R2, R1
    STR R3, [R0]
UP B UP
    END
```

Results:

Obtained the difference of two 32-bit numbers using pre-indexing with an ARM7 assembly language program.

5) (a) Output (Addition):

Registers	Values	
	Pre-execution	Post-execution
R0	0x00000000	0x40000008
R1	0x00000000	0x00000001
R2	0x00000000	0x00000006
R3	0x00000000	0x00000007
R4	0x00000000	0x00000000
R5	0x00000000	0x00000000
R6	0x00000000	0x00000000
⋮	⋮	⋮
R15	0x00000000	0x00000014
CPSR	0x000000d3	0x800000d3
N	0	0
Z	0	0
C	0	0
V	0	0
I	1	1
F	1	1
T	0	0
M	0x13	0x13

5)(a) Output (Subtraction)

Registers	Values	
	Pre-execution	Post-execution
R0	0x00000000	0x40000008
R1	0x00000000	0x00000001
R2	0x00000000	0x00000006
R3	0x00000000	0x00000005
⋮	⋮	⋮
R14	0x00000000	0x00000000
R15	0x00000000	0x00000014
CPSR	0x000000d3	0x200000d3
N	0	0
Z	0	0
C	0	1
V	0	0
I	1	1
F	1	1
T	0	0
M	0x13	0x13

5)(b) Write an ALP to perform an addition of five 32 bit numbers stored in memory using conditional loop and indexing.

ARM Code:

```
    AREA SBSC, CODE, READONLY
ENTRY
    LDR R2, =0x00000000
    LDR R3, =0x00000005
    LDR R5, =0x40000040
    LDR R0, =0x40000000
LC  LDR R1, [R0], #04
    ADDS R2, R2, R1
    ADDCS R6, R6, #1
    SUBS R3, R3, #1.
    BNE LC
    STR R2, [R5], #04
    STR R6, [R5]
UP  B UP
    END
```

Result:

Pre-indexing commands were used in looping of a program and to add 2 numbers.

5)(b) Output :

Registers	Values	
	Pre-execution	Post-execution
R0	0x00000000	0x40000014
R1	0x00000000	0x000 10005
R2	0x00000000	0x0005000f
R3	0x00000000	0x00000000
R4	0x00000000	0x00000000
R5	0x00000000	0x40000044
⋮	⋮	⋮
R14	0x00000000	0x00000000
R15	0x0000008c	0x0000002C
CPSR	0x000000d3	0x000000 d3
N	0	0
Z	0	1
C	0	1
V	0	0
I	1	1
F	1	1
T	0	0
M	0x13	0x13

6) (a) Write an ALP to perform multiplication of 32-bit numbers using MUL instruction.

ARM Code:

```
        AREA SBCS, CODE, READONLY
        ENTRY
        LDR R0, =0x40000000
        LDR R1, [R0], #04
        LDR R2, [R0], #04
        LDR R4, =0x40000030
        MUL R3, R1, R2
        STR R3, [R0]
UP      B UP
        END
```

Result:

Multiplication of two 32-bit numbers was performed using MUL instruction.

6)(a) Output:

Register	Value	
	Pre-execution	Post-execution
R0	0x00000000	0x40000008
R1	0x00000000	0x00000a43
R2	0x00000000	0x0000020a
R3	0x00000000	0x0014ec9e
R4	0x00000000	0x40000030
⋮	⋮	⋮
R12	0x00000000	0x00000000
R13	0x00000000	0x00000000
R14	0x00000000	0x00000000
R15	0x00000000	0x00000018
N	0	0
Z	0	0
C	0	0
⋮	⋮	⋮
M	0x13	0x13

6)(b) Output:

Registers	Values	
	Pre-execution	Post-execution
R0	0x00000000	0x40000008
R1	0x00000000	0x00000a01
R2	0x00000000	0x000c0b00
R3	0x00000000	0x00000000
R4	0x00000000	0x787a0b00
-1-	-x1x-	-x1x-
R14	0x00000000	0x00000000
R15	0x00000000	0x0000001c
CPSR	0x000000d3	0x000000d3
N	0	0
Z	0	0
C	0	0
V	0	0
I	1	1
F	1	1
T	0	0
M	0x13	0x13

6) (b) Write an ALP to perform multiplication of 32-bit numbers using UMULL instructions.

ARM Code:

```
        AREA SBCS, CODE, READONLY
        ENTRY
        LDR R0, =0x40000000
        LDR R5, =0x40000030
        LDR R1, [R0], #04
        LDR R2, [R0], #04
        UMULL R4, R3, R1, R2.
        STR R4, [R5], #04
        STR R3, [R5]
UP      B UP
        END.
```

Result:

Multiplication of two 32-bit numbers was performed using UMULL.

6) (c) Output:

Registers	Values	
	Pre-execution	Post-execution.
R0	0x00000000	0x40000028
R1	0x00000000	0x000d0013
R2	0x00000000	0x00003200
R3	0x00000000	0x00000000
R4	0x00000000	0x000000aa
R5	0x00000000	0x1c029c00
R6	0x00000000	0x00000000
R7	0x000000aa	0x40000044
x1x	-x1x-	-x1x-
R14	0x00000000	0x00000000
R15	0x00000008	0x00000030
CPSR	0x000000d3	0x600000d3
N	0	0
Z	0	1
C	0	1
V	0	0
I	1	1
F	1	1
T	0	0
M	0x13	0x13

6)(c) Write an ALP to perform multiply and accumulate operation of 5 32-bit numbers.

ARM Code:

```
        AREA SBSCS, CODE, READONLY
        ENTRY
        LDR R0, =0x40000000
        LDR R4, = 0x00000000
        LDR R5, = 0x00000000
        LDR R6, = 0x5
        LDR R7, =0x40000040

Lo      LDR R1, [R0], #04
        LDR R2, [R0], #04
        UMLAL R5, R4, R1, R2
        SUBS R6, R6, #1
        BNE Lo
        STR R4, [R7], #04
        STR R5, [R7]
UP      B UP
        END
```

Result: Multiplied 5 32-bit numbers using ALP.

Memory Block Outputs -

6) (a):

Memory Address: 0x40000000

0x40000000 : 43 0A 00 00

0x40000004 : 0A 02 00 00

0x40000008 : 9E EC 14 00

6(b):

Memory Address : 0x40000000

0x40000000 : 01 0A 00 00

0x40000004 : 00 0B 0C 00

0x40000030 : 00 0B 7A 7B

0x40000034 : 00 00 00 00

6)(c):

Memory Address : 0x40000000

0x40000000 : 00 0A 00 00

0x40000004 : 0B 32 00 00

0x40000008 : 00 14 0A 00

0x4000000C : 03 00 0C 00

0x40000010 : 00 04 AD 00

0x40000014 : EA 31 00 00

0x40000018 : 0A 00 32 00

0x4000001C : 00 42 00 00

0x40000020 : 13 00 0D 00

0x40000024 : 00 32 00 00