

Q1) Assume a 32-bit number in 40000004H. Add nibble4 and nibble0 and store the result in 4000000CH.

**SOURCE CODE:-**

---

;Assume a 32-bit number in 40000004H.

;Add nibble4 and nibble0 and store the result in 4000000CH.

AREA NIBBLE\_ADD, CODE, READONLY

ENTRY

MAIN

LDR R0, VALUE ; Load the Address of the Value

LDR R1, [R0] ; Load the content of R0 into R1

MOV R2, #0X0000000F ; Move masking value(mask1) for selecting Nibble0

MOV R3, #0X000F0000 ; Move masking value(mask2) for Nibble4

AND R4, R1, R2 ; AND the original number with mask1 to select Nibble0 value

AND R5, R1, R3 ; AND the original number with mask2 to select Nibble4 value

LSR R5, R5, #16 ; LogicalShiftLeft R5 register content to move it to LowestNibble

ADD R6, R4, R5 ; Add both the nibble values and Store in R6

LDR R0, RESULT

STR R6, [R0] ; Load the value of result to mentioned address location

SVC &11

VALUE DCD &40000004

RESULT DCD &4000000C

END

---

```

1 ;Consider an array of number present from 40000000H. Add only if the numbers are positive.
2 ;40000000H has the count of the array.
3 ;A number is said to be negative if its MSB bit is set to 1.
4 ;Registration Number - 211039020
5     AREA ADD_ARRAY, CODE, READONLY
6     ENTRY
7 MAIN
8     LDR R0, TABLE      ; In the table first value is count followed by array elements
9     LDR R2, [R0]        ; Load the count
10    EOR R3, R3, R3      ; Clear Register R3 for storing sum of positive elements
11 LOOP
12    CMP R2, #0          ; Compare count with Zero
13    BEQ DONE           ; Branch to DONE if count is zero
14    LDR R1, [R0, #4]!   ; Load the array elements to R1 register
15    CMP R1, #0          ; Compare R1 register content with 0
16    BMI LOOP1          ; If the number is negative then Branch to Loop1 and decrement the count
17                        ; BMI - Branch if minus
18    ADD R3, R3, R1      ; It checks the Negative Flag if it is set Branch to Loop1
19    SUB R2, R2, #1      ; If the number is positive Add it with R3 register content
20    B LOOP              ; Decrement the count
21 LOOP1
22    SUB R2, R2, #1
23    CMP R2, #0
24    BEQ DONE
25    BNE LOOP
26 DONE LDR R4, RESULT
27     STR R3, [R4]        ; Load the Result in R3 to desired location mentioned below
28 STOP B STOP
29
30 TABLE DCD &40000000
31 RESULT DCD &4000003C
32     END

```

OUTPUT:-

The screenshot displays the Keil uVision IDE interface. On the left, the 'Registers' window shows the current state of registers R0 through R15, CPSR, and SPSR. Register R0 contains 0x40000000, R1 contains 0x21435678, R2 contains 0x0000000F, R3 contains 0x000F0000, R4 contains 0x00000008, R5 contains 0x00000003, and R6 contains 0x0000000B. The 'PC' register shows 0x00000008.

The main window shows the assembly code for the program. The code is for a program that adds the first 4 bytes of an array starting at 40000000H. The code is as follows:

```

1 ;Assume a 32-bit number in 40000004H.
2 ;Add nibble4 and nibble0 and store the result in 4000000CH.
3 ;Registration Number - 211039020
4     AREA NIBBLE_ADD, CODE, READONLY
5     ENTRY
6 MAIN
7     LDR R0, VALUE ;Load the Address of the Value
8     LDR R1, [R0] ; Load the content of R0 into R1
9     MOV R2, #0X0000000F ;Move masking value(mask1) for selecting Nibble0
10    MOV R3, #0X000F0000 ;Move masking value(mask2) for Nibble4
11    AND R4, R1, R2 ;AND mask1 to select Nibble0 value
12    AND R5, R1, R3 ;AND mask2 to select Nibble4 value
13    LSR R5, R5, #16 ;Shift content to move it to LowestNibble
14    ADD R6, R4, R5 ;Add Store in R6
15    LDR R0, RESULT
16    STR R6, [R0] ;Load mentioned address location
17    SVC #11
18
19 VALUE DCD &40000004
20 RESULT DCD &4000000C
21     END

```

The 'Memory 1' window is open, showing the memory contents at addresses 0x40000000 through 0x4000001C. The contents are as follows:

Address	Value
0x40000000	00 00 00 00
0x40000004	78 56 43 21
0x40000008	00 00 00 00
0x4000000C	0B 00 00 00
0x40000010	00 00 00 00
0x40000014	00 00 00 00
0x40000018	00 00 00 00
0x4000001C	00 00 00 00

The result of the addition is shown as 0B 00 00 00 at address 0x4000000C.

Q2) Consider an array of number present from 40000000H. Add only if the numbers are positive. 40000000H has the count of the array.

#### SOURCE CODE:-

---

;Consider an array of number present from 40000000H. Add only if the numbers ;are positive.

;40000000H has the count of the array.

;A number is said to be negative if its MSB bit is set to 1.

;Registration Number - 211039020

AREA ADD\_ARRAY,CODE,READONLY

ENTRY

MAIN

LDR R0,TABLE; In the table first value is count followed by array elements

LDR R2,[R0];Load the count

EOR R3,R3,R3;Clear Register R3 for storing sum of positive elements

LOOP

CMP R2,#0 ; Compare count with Zero

BEQ DONE

LDR R1,[R0,#4]! ;Load the array elements to R1 register

CMP R1,#0 ;Compare R1 register content with 0

BMI LOOP1 ;If the number is negative then Branch to Loop1 and

;decrement the count

;BMI - Branch if minus ; It checks the Negative Flag if it is set Branch to Loop1

ADD R3,R3,R1 ;If the number is positive Add it with R3 register content

SUB R2,R2,#1 ;Decrement the count

B LOOP

LOOP1

SUB R2,R2,#1

CMP R2,#0

BEQ DONE

BNE LOOP

DONE

LDR R4,RESULT

STR R3,[R4] ;Load the Result in R3 to desired location

STOP B STOP

TABLE DCD &40000000

RESULT DCD &4000003C

END

```
1 ;Consider an array of number present from 40000000H. Add only if the numbers are positive.
2 ;40000000H has the count of the array.
3 ;A number is said to be negative if its MSB bit is set to 1.
4 ;Registration Number - 211039020
5 AREA ADD_ARRAY, CODE, READONLY
6 ENTRY
7 MAIN
8     LDR R0, TABLE      ; In the table first value is count followed by array elements
9     LDR R2, [R0]         ; Load the count
10    EOR R3, R3, R3       ; Clear Register R3 for storing sum of positive elements
11 LOOP    CMP R2, #0      ; Compare count with Zero
12        BEQ DONE
13        LDR R1, [R0, #4] ; Load the array elements to R1 register
14        CMP R1, #0       ; Compare R1 register content with 0
15        BMI LOOP1        ; If the number is negative then Branch to Loop1 and decrement the count
16        ; BMI - Branch if minus
17        ; It checks the Negative Flag if it is set Branch to Loop1
18        ADD R3, R3, R1    ; If the number is positive Add it with R3 register content
19        SUB R2, R2, #1    ; Decrement the count
20        B LOOP
21 LOOP1   SUB R2, R2, #1
22        CMP R2, #0
23        BEQ DONE
24        BNE LOOP
26 DONE   LDR R4, RESULT
27        STR R3, [R4]      ; Load the Result in R3 to desired location mentioned below
28 STOP B STOP
29
30 TABLE DCD &40000000
31 RESULT DCD &4000003C
32 END
```

## OUTPUT:-

The screenshot displays an ARM assembly simulator interface. On the left, the 'Registers' window shows the current state of registers R0 through R15, with R0, R1, and R4 highlighted. The main window shows the assembly code for a program named 'add\_arrays'. The code includes comments and instructions for loading, comparing, and adding array elements. On the right, the 'Memory 1' window shows the memory contents at addresses 0x40000000 through 0x4000004C. A callout points to the memory at 0x40000004, which contains the value 06 00 00 00, labeled as 'Negative Numbers'. Another callout points to the memory at 0x4000003C, which contains the value 0D 00 00 00, labeled as 'Result'.

**Registers**

Register	Value
R0	0x40000014
R1	0x00000006
R2	0x00000000
R3	0x00000000
R4	0x4000003C
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13	0x00000000
R14	0x00000000
R15	0x00000000
SPSR	0x00000000
PC	0x00000044

**add\_arrays**

```
4 ;Registration Number - 211039020
5 AREA ADD_ARRAY, CODE, READONLY
6 ENTRY
7 MAIN
8     LDR R0, TABLE; In the table first value is count follow
9     LDR R2, [R0]; Load the count
10    EOR R3, R3, R3; Clear Register R3 for storing sum of pos
11    LOOP
12    CMP R2, #0 ; Compare count with Zero
13    BEQ DONE
14    LDR R1, [R0, #4]; Load the array elements to R1 register
15    CMP R1, #0 ; Compare R1 register content with 0
16    BMI LOOP1 ; If the number is negative then Branch to 1
17    ADD R3, R3, R1 ; If the number is positive Add it with R
18    SUB R2, R2, #1 ; Decrement the count
19    B LOOP
20    LOOP1
21    SUB R2, R2, #1
22    CMP R2, #0
23    BEQ DONE
24    BNE LOOP
25    DONE LDR R4, RESULT
26    STR R3, [R4]; Load the Result in R3 to desired location
27    STOP B STOP
28
29 TABLE DCD 40000000
30 RESULT DCD 4000003C
31 END
```

**Memory 1**

Address	Value
0x40000000	05 00 00 00
0x40000004	06 00 00 00
0x40000008	04 00 00 00
0x4000000C	03 00 00 00
0x40000010	08 00 00 00
0x40000014	06 00 00 00
0x40000018	05 00 00 00
0x4000001C	00 00 00 00
0x40000020	00 00 00 00
0x40000024	00 00 00 00
0x40000028	00 00 00 00
0x4000002C	00 00 00 00
0x40000030	00 00 00 00
0x40000034	00 00 00 00
0x40000038	00 00 00 00
0x4000003C	0D 00 00 00
0x40000040	00 00 00 00
0x40000044	00 00 00 00
0x40000048	00 00 00 00
0x4000004C	00 00 00 00

Negative Numbers

Result