



**Faculty of Engineering & Technology  
Electrical & Computer Engineering Department**

**Computer Organization and Microprocessor  
ENCS2380  
Assembly Assignment**

---

**Prepared by:**

Maha Maher Mali

1200746

**Instructor:** Dr. Abualseoud Hanani

**Section:** 1

**Date:** 23/6/2022

## Contents

Procedure.....	4
Declare an array.....	4
Code.....	4
Register .....	5
Memory .....	5
Sum of all elements.....	6
code.....	6
Register .....	7
Memory .....	7
Sum of even elements.....	8
Code.....	8
Register .....	9
Memory .....	9
Find the largest power .....	10
Code.....	10
Register .....	11
Memory .....	12

## Figures

Figure1:code for part A .....	4
Figure2:Register for part A .....	5
Figure3:Memory for part A.....	5
Figure4: for part B .....	6
Figure5:Register for part B .....	7
Figure6:Memory for part B .....	7
Figure7:code for part C .....	8
Figure8:register for part C .....	9
Figure9:Memory for part C .....	9
Figure10:code for part d .....	10
Figure11:register for part d .....	11
Figure 12:memory for part d .....	12

## Procedure

### Declare an array

Declare an array of at least 10 8-bit unsigned integer numbers in the memory with initial values.

e.g. 34, 56, 27, 156, 200, 68, 128, 235, 17, 45

#### Code

```
1      AREA RESET, DATA, READONLY
2      EXPORT __Vectors
3
4      __Vectors DCD 0x20001000
5              DCD Reset_Handler
6
7      ALIGN
8      AREA MYRAM, DATA, READWRITE
9      SUM DCD 0
10     AREA MYCODE, CODE, READONLY
11     ENTRY
12     EXPORT Reset_Handler
13 Reset_Handler
14
15     LDR R1,=Array; R1 pointer to the array
16     MOV R2,#10;STORE ARRAY IN R2 THE NUMBER 10 MEANS THE ELEMENT IN ARRAY IS 10
17
18 loop LDR R0,[R1],#8;STORE 8 BYTE FROM R1 TO R0
19     SUB R2,R2,#01; SUBTRACT (R2=R2-#01)
20     CMP R2,#00;COMPERE IF R2=0 TO EXIST THE LOOP
21
22 Array DCB 34, 56, 27, 156, 200, 68, 128,235, 17, 45; declare the array
23
24 here B here
25 end
```

Figure1:code for part A

## Register

Register	Value
<b>Core</b>	
R0	0x9C1B3822
R1	0x00000020
R2	0x00000009
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20001000
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x00000022
xPSR	0x21000000

Figure2: Register for part A

## Memory

Memory 2																
Address: 0x016																
0x00000016:	00	2A	22	38	1B	9C	C8	44	80	EB	11	2D	FE	E7	18	00
0x0000002B:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x00000040:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x00000055:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x0000006A:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x0000007F:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x00000094:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x000000A9:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x000000BE:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x000000D3:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x000000E8:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x000000FD:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Figure3: Memory for part A

## Sum of all elements

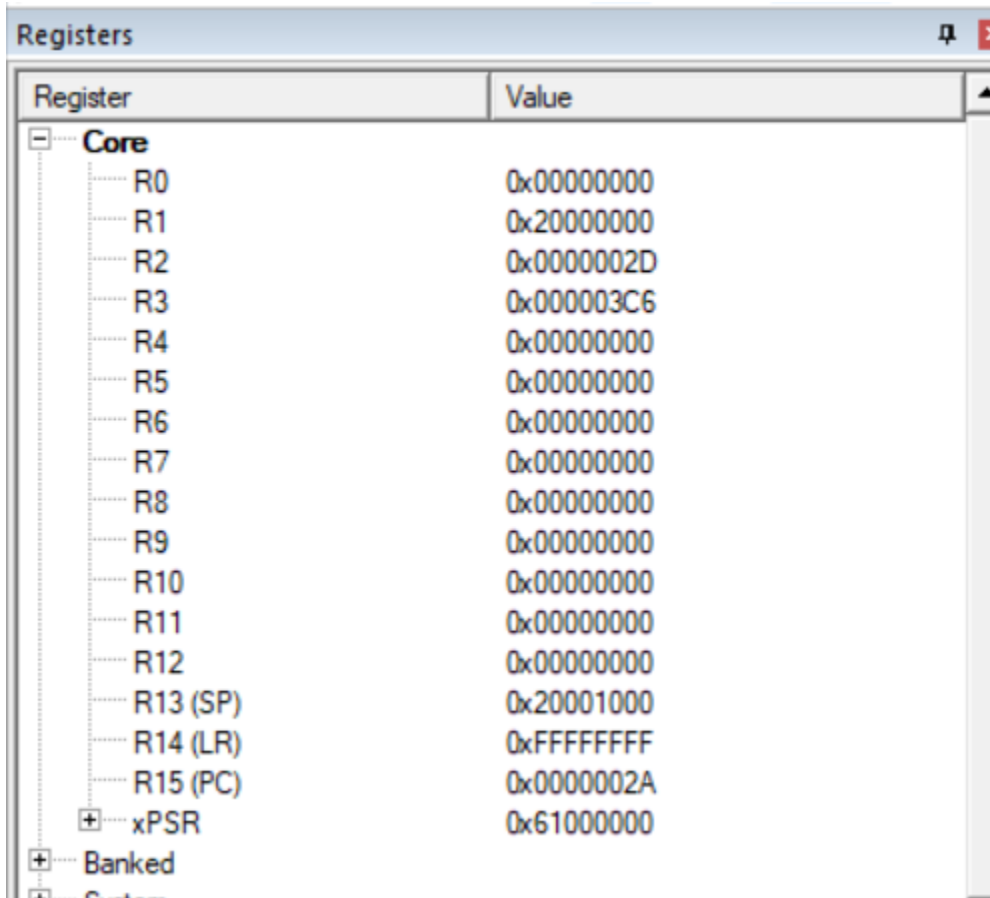
Find the sum of all elements of the array and store it in the memory, e.g. variable SUM.

code

```
1
2 AREA RESET, DATA, READONLY;NAME:MAHA MAHER MALI/ID=1200746
3 EXPORT __Vectors
4 __Vectors DCD 0x20001000
5          DCD Reset_Handler
6 align
7 area myRam, data, readwrite
8 sum DCD 0
9 area mycode, code, readonly
10 ENTRY
11 EXPORT Reset_Handler
12 Reset_Handler
13
14     LDR R1, =Array
15     MOV R0,#10
16     LDR R1,=Array
17     LDR R3,=0;R3
18     LDR R2,=0;R2
19 loopl LDRB R2,[R1],#1
20     ADD R3,R3,R2
21     SUB R0,R0,#1 SUBTRACT (R0=R0-#01)
22     CMP R0,#00 COMPARE IF R0=00 TO EXIST THE LOOP
23     BNE loopl
24     LDR R1,sum;LOAD SUM TP R1
25     STR R3 ,[R1] ;STORE R1 IN R3
26 here B here
27 Array DCB 34, 56, 27, 156, 200, 68, 128,235, 17, 45
28     end
```

Figure4: for part B

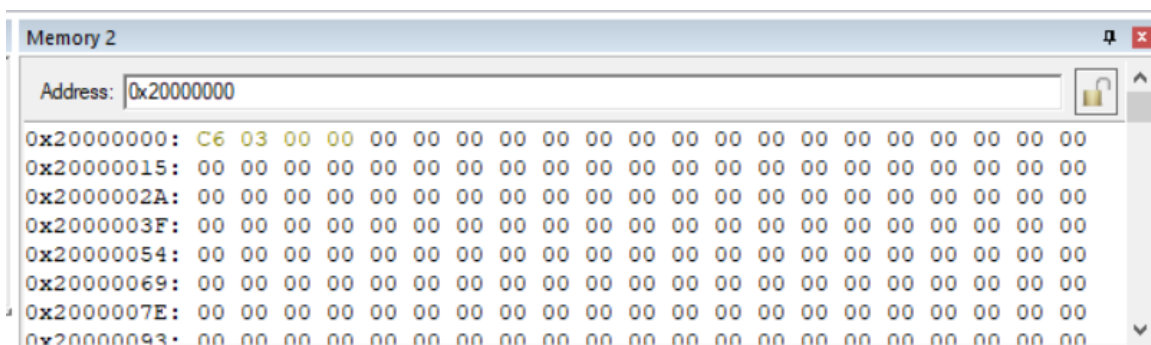
## Register



Register	Value
<b>Core</b>	
R0	0x00000000
R1	0x20000000
R2	0x0000002D
R3	0x000003C6
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20001000
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x0000002A
xPSR	0x61000000
<b>Banked</b>	
<b>System</b>	

Figure5:Register for part B

## Memory



Memory 2	
Address:	0x20000000
0x20000000:	C6 03 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000015:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x2000002A:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x2000003F:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000054:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000069:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x2000007E:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000093:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Figure6:Memory for part B

## Sum of even elements

find the sum of the even numbers in this array and store it in the memory, e.g. variable EVEN

Code

```
1      AREA RESET, DATA, READONLY;NAME:MAHA MAHER MALI/ID:1200746
2      EXPORT __Vectors
3      __Vectors DCD 0x20001000
4          DCD Reset_Handler
5      align
6      area myRam, data, readwrite
7      SUM DCD 0
8      EVEN DCD 0
9      area mycode, code, readonly
10     ENTRY
11     EXPORT Reset_Handler
12     Reset_Handler
13         LDR R1, =Array
14         MOV R0, #10
15         LDR R1, =Array
16         LDR R3, =0
17         LDR R2, =0
18     loop1 LDRB R2, [R1], #1
19         ADD R3, R3, R2
20         SUB R0, R0, #1
21         TST R2, #1
22         ADDEQ R5, R2
23         CMP R0, #00
24         BNE loop1
25         LDR R1, =SUM;LOAD R1 TO VARAIBLE SUM
26         STR R3, [R1];STORE R1 IN R3
27
28         LDR R1, =EVEN ;LOAD R1 TO VARAIBLE EVEN
29         STR R5, [R1];STORE R1 IN R5
30     here B here
31     Array DCB 34, 56, 27, 156, 200, 68, 128, 235, 17, 45
32
33     end
```

Figure7:code for part C



## Register

Register	Value
<b>Core</b>	
R0	0x00000000
R1	0x20000004
R2	0x0000002D
R3	0x000003C6
R4	0x00000000
R5	0x00000282
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20001000
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x00000036
+ xPSR	0x61000000

Figure8:register for part C

## Memory

Memory 2	
Address:	0x20000004
0x20000004:	82 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000019:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x2000002E:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000043:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000058:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x2000006D:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000082:	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Figure9:Memory for part C

Find the largest power

**d)** Find the largest power of 2 divisor that divides into a number exactly for each element in the array and store it in another array in the memory. You have to use a procedure (function), **POW2**, which takes an integer as an input parameter and return its largest power of 2. For example, POW(52) would return 4, where POW(56) would return 8, and so on.

Code

```

1      AREA RESET, DATA, READONLY;NAME:MAHA MAHER MALI/ID:1200746
2      EXPORT __Vectors
3      __Vectors DCD 0x20001000
4          DCD Reset_Handler
5      align
6      area myRam, data, readwrite
7      SUM DCD 0
8      EVEN DCD 0
9      Array2 DCB 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
10     area mycode, code, readonly
11     ENTRY
12     EXPORT Reset_Handler
13     POW2 PROC;DEFINE FUNCTION
14     TST R7,R2
15     LSLEQ R7,#1
16     BEQ POW2
17     BX LR
18     ENDP;END OF FUNCTION
19     Reset_Handler
20     LDR R1, =Array
21     MOV R0,#10
22     LDR R8,=Array2
23     LDR R1,=Array
24     LDR R3,=0
25     LDR R2,=0
26     loop1 LDRB R2,[R1],#1
27     ADD R3,R3,R2
28     SUB R0,R0,#1
29     TST R2, #1
30     MOV R7 ,#1
31     BL POW2
32
33     STRB R7, [R8],#1
34     ADDEQ R5,R2
35     CMP R0,#00
36     BNE loop1
37     LDR R1,=SUM
38     STR R3 , [R1]
39     LDR R1,=EVEN
40     STR R5 , [R1]
41     here B here
42     Array DCB 34, 56, 27, 156, 200, 68, 128,235, 17, 45
43     end

```

Figure10:code for part d

## Register

Register	Value
<b>Core</b>	
R0	0x00000009
R1	0x00000053
R2	0x00000022
R3	0x00000022
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000001
R8	0x20000008
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20001000
R14 (LR)	0x0000003D
R15 (PC)	0x00000008
+ xPSR	0x41000000
+ Banked	
+ System	
- Internal	
Mode	Thread
Privileged	Privileged

Figure11:register for part d

## Memory

Memory 2																			
Address:		0x20000008																	
0x20000008:	02	08	01	04	08	04	80	01	01	01	00	00	00	00	00	00	00	00	00
0x2000001D:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x20000032:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x20000047:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x2000005C:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x20000071:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x20000086:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x2000009B:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x200000B0:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x200000C5:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x200000DA:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x200000EF:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x20000104:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x20000119:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
0x2000012E:	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Figure 12:memory for part d