Faculty of Engineering and Technology

Electrical and Computer Engineering Department

Assembly Assignment

Prepared by:

Mohammad Abu-Shelbaia 1200198

Instructor:

Dr.Abualseoud Hanani

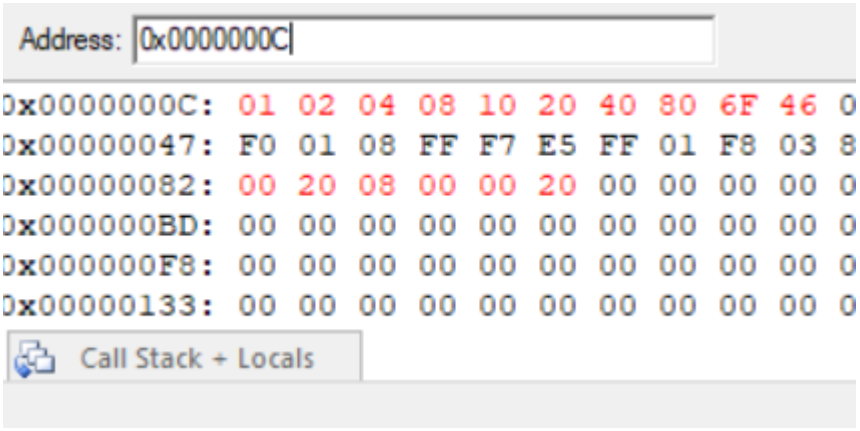
Section: 1

Date:

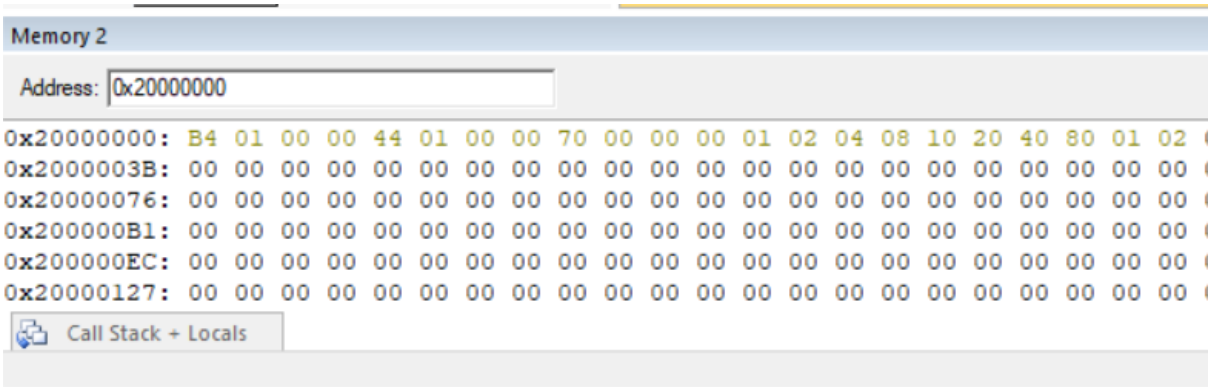
23rd June 2022

Solution

Using the array: 1, 2, 4, 8, 16, 32, 64, 128, 111, 70 (B)
The output array should be: 01, 02, 04, 08, 10, 20, 40, 80, 01, 02 (H)
Even Sum: 324 in decimal, 144 in hexadecimal.
Odd Sum:112 in decimal, 70 in hexadecimal.
Sum: 436 in decimal, 1B4 in hexadecimal.
Array in memory:

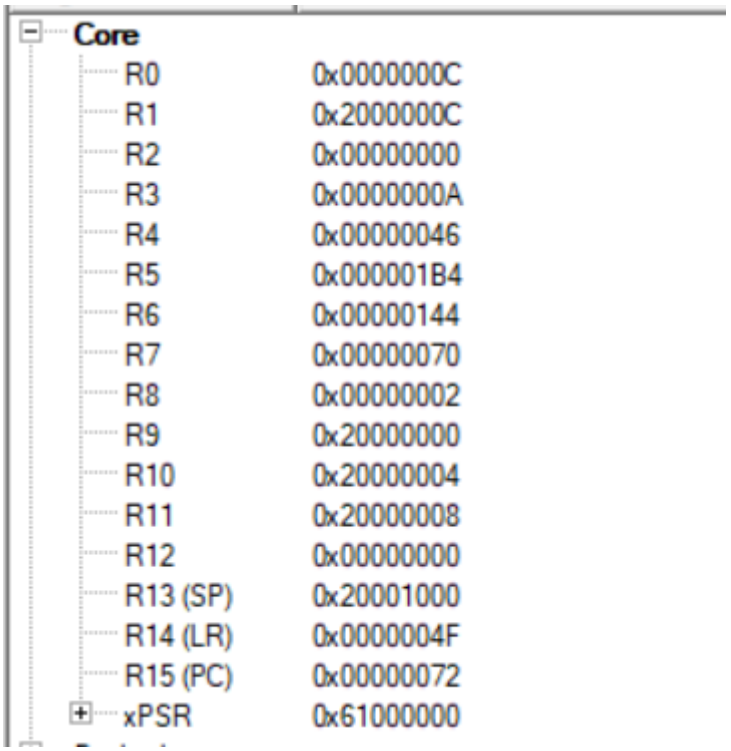


Sum, even, odd and array in memory:



Address 20000000 is the sum = 000001B4H
Address 20000004 is the even sum = 00000144H
Address 20000008 is the odd sum = 00000070H
Address 2000000C is the output array = 01, 02, 04, 08, 10, 20, 40, 80, 01, 02 (H)

Registers:


Core
R0 0x0000000C
R1 0x2000000C
R2 0x00000000
R3 0x0000000A
R4 0x00000046
R5 0x000001B4
R6 0x00000144
R7 0x00000070
R8 0x00000002
R9 0x20000000
R10 0x20000004
R11 0x20000008
R12 0x00000000
R13 (SP) 0x20001000
R14 (LR) 0x0000004F
R15 (PC) 0x00000072
xPSR 0x61000000

Register Table

```
;+----------+-----------------------------+
;| Register | Purpose                     |
;+----------+-----------------------------+
;| R0       | Input Array Address         |
;+----------+-----------------------------+
;| R1       | Output Array Address        |
;+----------+-----------------------------+
;| R2       | Stores Length "Loop"        |
;+----------+-----------------------------+
;| R3       | Array Displacement          |
;+----------+-----------------------------+
;| R4       | Loading Elements from R0    |
;+----------+-----------------------------+
;| R5       | Summation Accumulator       |
;+----------+-----------------------------+
;| R6       | Even Accumulator            |
;+----------+-----------------------------+
;| R7       | Odd Accumulator             |
;+----------+-----------------------------+
;| R8       | POW2 Output                 |
;+----------+-----------------------------+
;| R9       | Summation Address in memory |
;+----------+-----------------------------+
;| R10      | Even Address in Memory      |
;+----------+-----------------------------+
;| R11      | Odd Address in Memory       |
;+----------+-----------------------------+
```

CODE

```
      PRESERVE8
            THUMB
            AREA RESET, DATA, READONLY
            EXPORT __Vectors
__Vectors
            DCD 0x20001000
            DCD Reset_Handler
            ALIGN
Length DCD 10
      ALIGN
Array DCB   1, 2, 4, 8, 16, 32, 64, 128, 111, 70
            AREA MYRAM, DATA, READWRITE
SUM DCD 0
EVEN DCD 0
ODD DCD 0
newArray    space 10
            AREA MYCODE, CODE, READONLY
            ENTRY
            EXPORT Reset_Handler
POW2  PROC
            CMP R4, #0              ; IF THE NUMBER IS 0
            BXEQ LR                 ; return (1) if the number is 0
            TST R4, R8              ; R8 is a bit mask starting from 1
and shifting by 1 bit each time until reaching the first rightmost set
bit
            LSLEQ R8, #1      ; Shift the mask by 1 bit
            BEQ POW2                ; loop until the condition is false
            BX LR                   ; return after finishing
            ENDP
Reset_Handler
            LDR R0, =Array          ; Array Address
            LDR R1, =newArray ; newArray Address
            LDRB R2, Length         ; Length
            MOV R3, #0              ; Array displacement
FOR_START
            LDRB R4, [R0, R3]       ; READING BIT BY BIT FROM THE ARRAY
            ADD R5, R4              ; Adding numbers to the sum
Accumulator register
            TST R4, #1              ; Testing to see if the number is
even or not
            ADDEQ R6, R4            ; Adding even numbers to even
Accumulator register
            ADDNE R7, R4            ; Adding odd numbers to odd
Accumulator register
```

```
            MOV R8, #1              ; Bit mask which we will use as an
input along with the value we stored in r4
            BL POW2                 ; BRANCH LINKING TO PROC
            STRB R8, [R1, R3] ; Storing result of POW2 back into the new
array
            ADD  R3, #1             ; Incrementing displacement register
            SUBS R2, #1             ; Decrement the counter
            BNE FOR_START           ; RELOOP until the counter is 0
            LDR R9, =SUM ; storing sum pointer in register
            STR R5, [R9]            ; storing sum in memory
            LDR R10,=EVEN           ; storing even pointer in register
            STR R6, [R10]           ; storing even in memory
            LDR R11, =ODD           ; storing odd pointer in register
            STR R7, [R11]           ; storing odd in memory
STOP
            B STOP
            END
```

POW2

Code explanation:
In this code, we used a bit mask to find the first bit set in the input given, starting the bit mask from 0000 0001 and applying it to the input integer using TST command, which changes the zero-flag, we check if the zero-flag is set we shift the mask to the left by 1 which is equivalent to multiplying it by 2 and recall the function, and we keep going until the zero-flag is clear we exit the function and store the value of the mask, and to avoid infinite loops when we give 0 as an input we exit immediately and store the initial value of the mask which is 1.