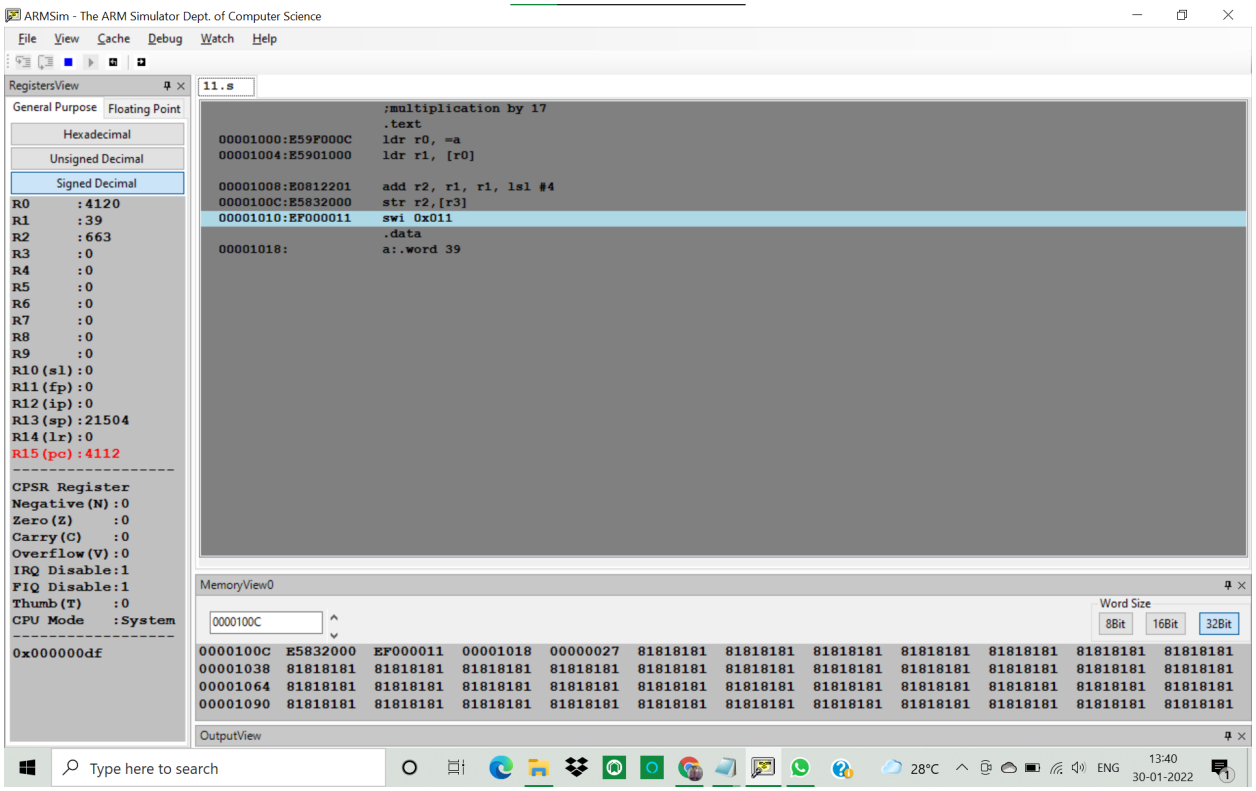
Name : yuvaraj dc
Srn : pes1ug20cs521
Sec i, 4th sem

---

4. Write a program in ARM7TDMI–ISA to find the product of two 32–bit numbers using barrel shifter.

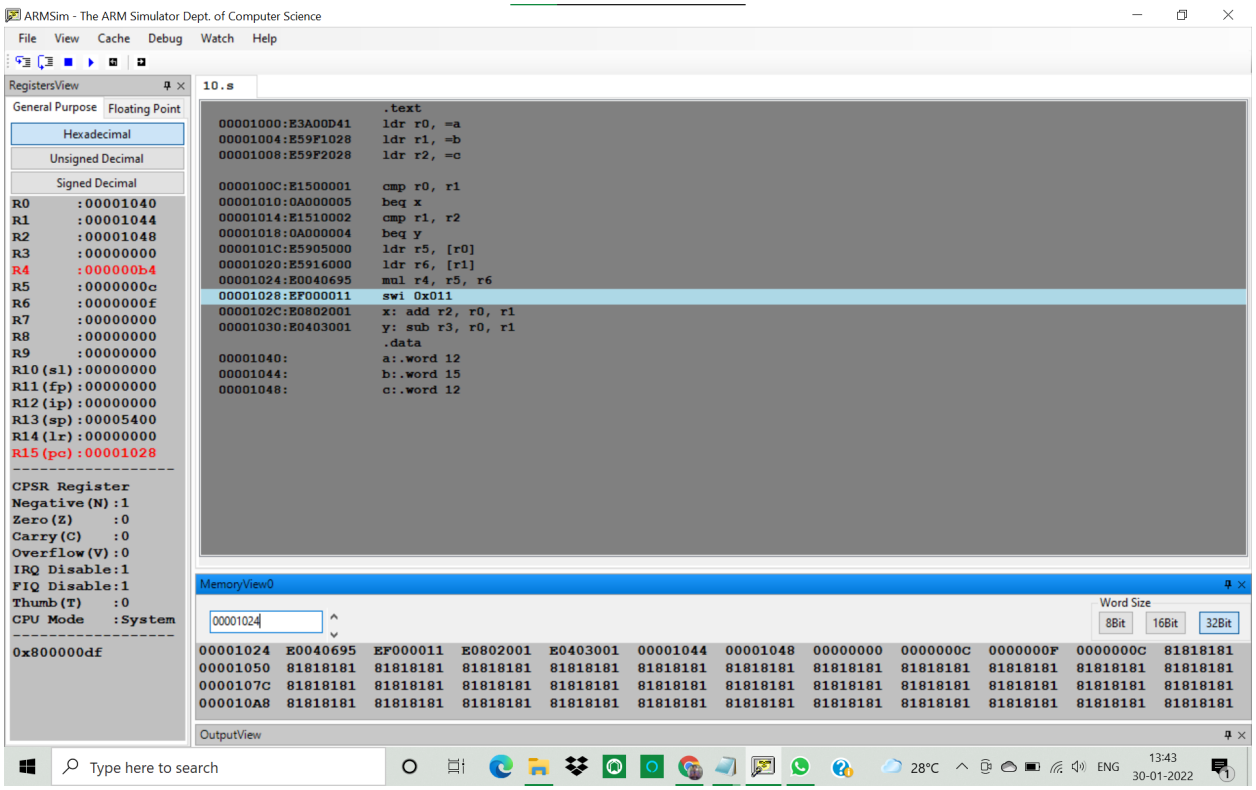
Code

```
;multiplication by 17
.text
ldr r0, =a
ldr r1, [r0]

add r2, r1, r1, lsl #4
str r2,[r3]
swi 0x011
.data
a:.word 39
```



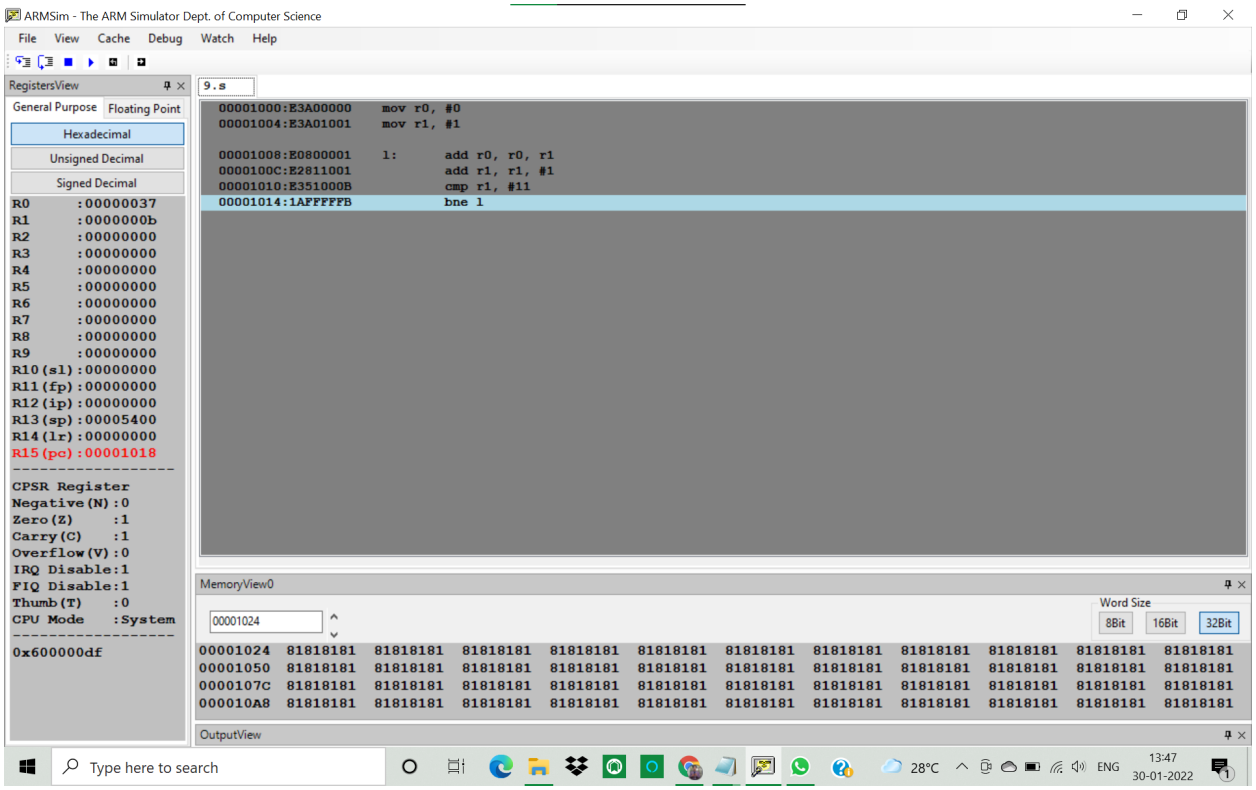
5. Convert the following statement in C language into an ALP using ARM7TDMI – ISA. IF([A]==[B]) then C=[A]+[B]; ELSE IF ([B]==[C]) D=[A]-[B]; ELSE E=[A]*[B] Where A,B, C, D & E are memory locations.

```
.text
ldr r0, =a
ldr r1, =b
ldr r2, =c

cmp r0, r1
beq x
cmp r1, r2
beq y
ldr r5, [r0]
ldr r6, [r1]
mul r4, r5, r6
swi 0x011
x: add r2, r0, r1
y: sub r3, r0, r1
.data
a:.word 12
b:.word 15
c:.word 12
```



3. Write a program in ARM7TDMI–ISA to find the sum of N natural numbers. Store the result in the memory location.

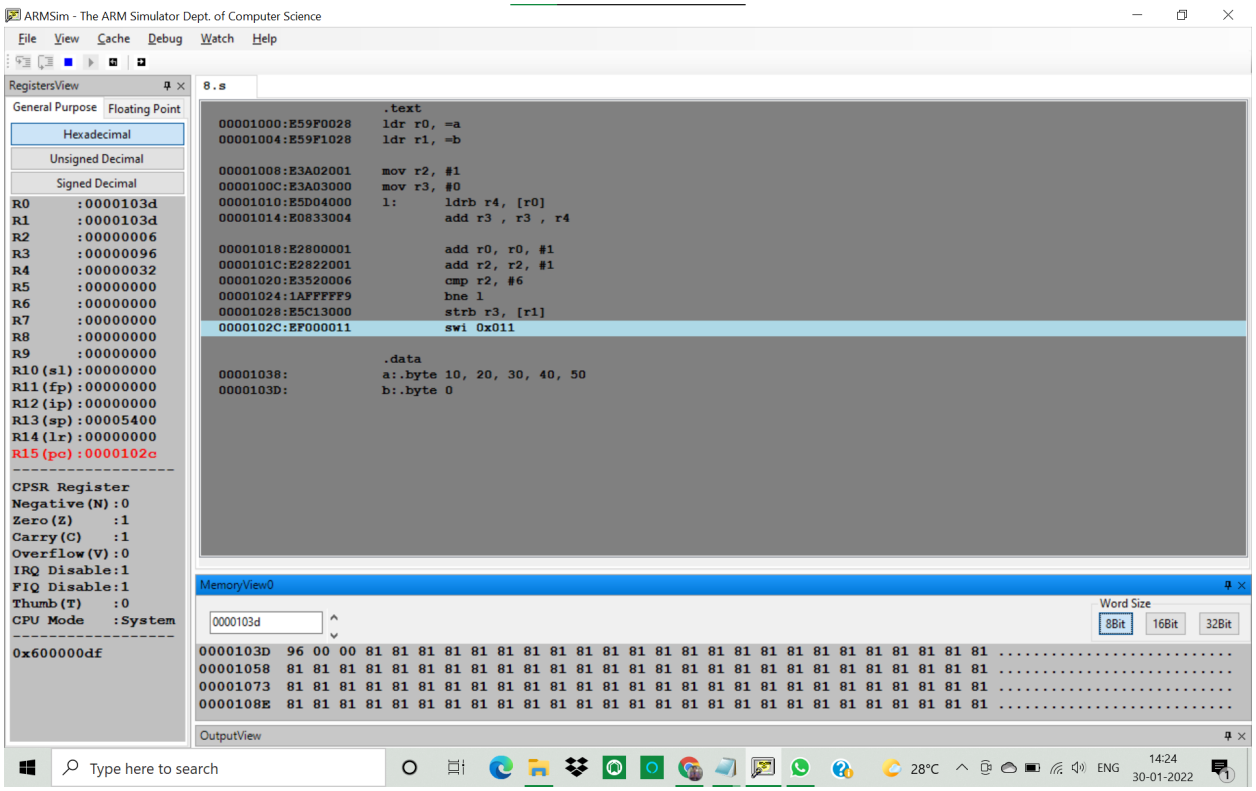
```
mov r0, #0
mov r1, #1

l:      add r0, r0, r1
        add r1, r1, #1
        cmp r1, #11
        bne l
```



2. Write a program in ARM7TDMI–ISA to find the sum of N data items in the memory. Store the result in the memory location. a) Use Full word (.word directive) b) Use Half word(.Hword directive) c) Use Byte wise (.Byte directive)

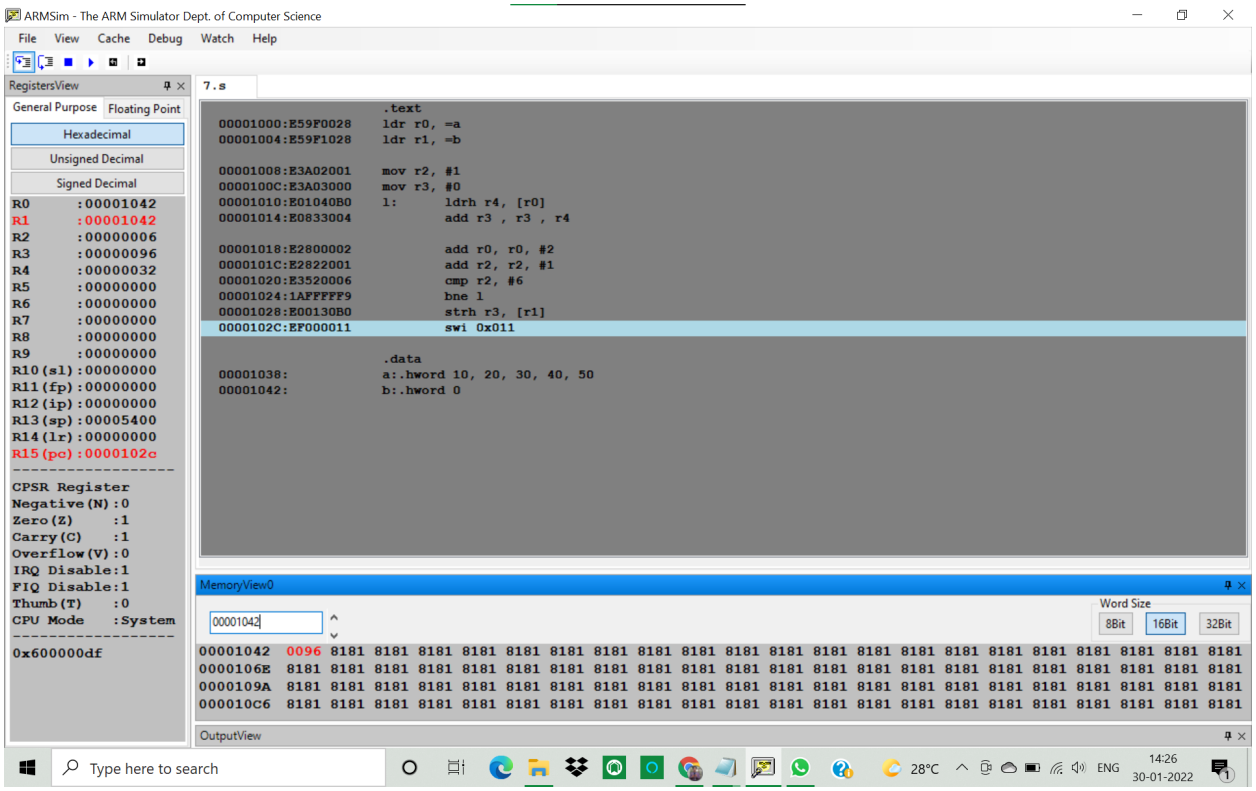
```
.text
ldr r0, =a
ldr r1, =b

mov r2, #1
mov r3, #0
l:      ldrb r4, [r0]
        add r3 , r3 , r4

        add r0, r0, #1
        add r2, r2, #1
        cmp r2, #6
        bne l
        strb r3, [r1]
        swi 0x011
```

```
.data
a:.byte 10, 20, 30, 40, 50
b:.byte 0
```

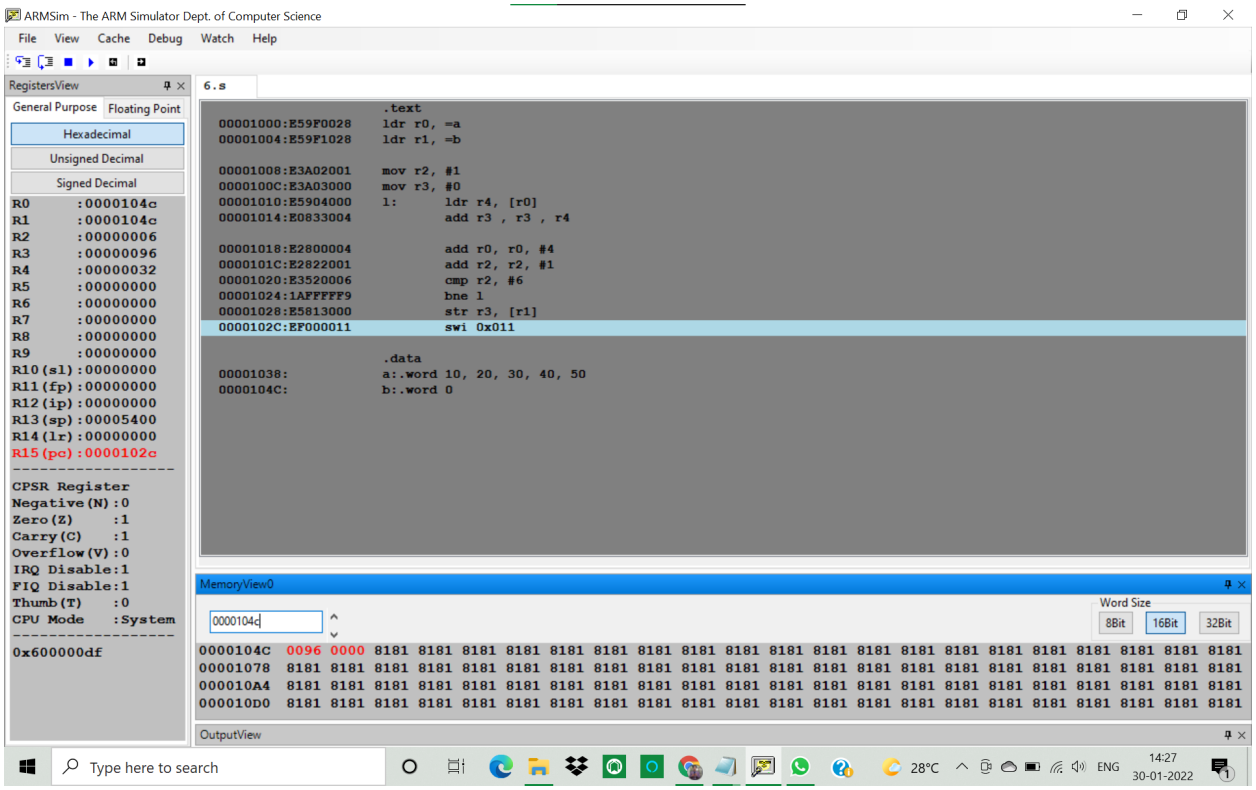



```
.text
ldr r0, =a
ldr r1, =b

mov r2, #1
mov r3, #0
l:      ldrh r4, [r0]
        add r3 , r3 , r4

        add r0, r0, #2
        add r2, r2, #1
        cmp r2, #6
        bne l
        strh r3, [r1]
        swi 0x011

.data
a:.hword 10, 20, 30, 40, 50
```

b:.hword 0



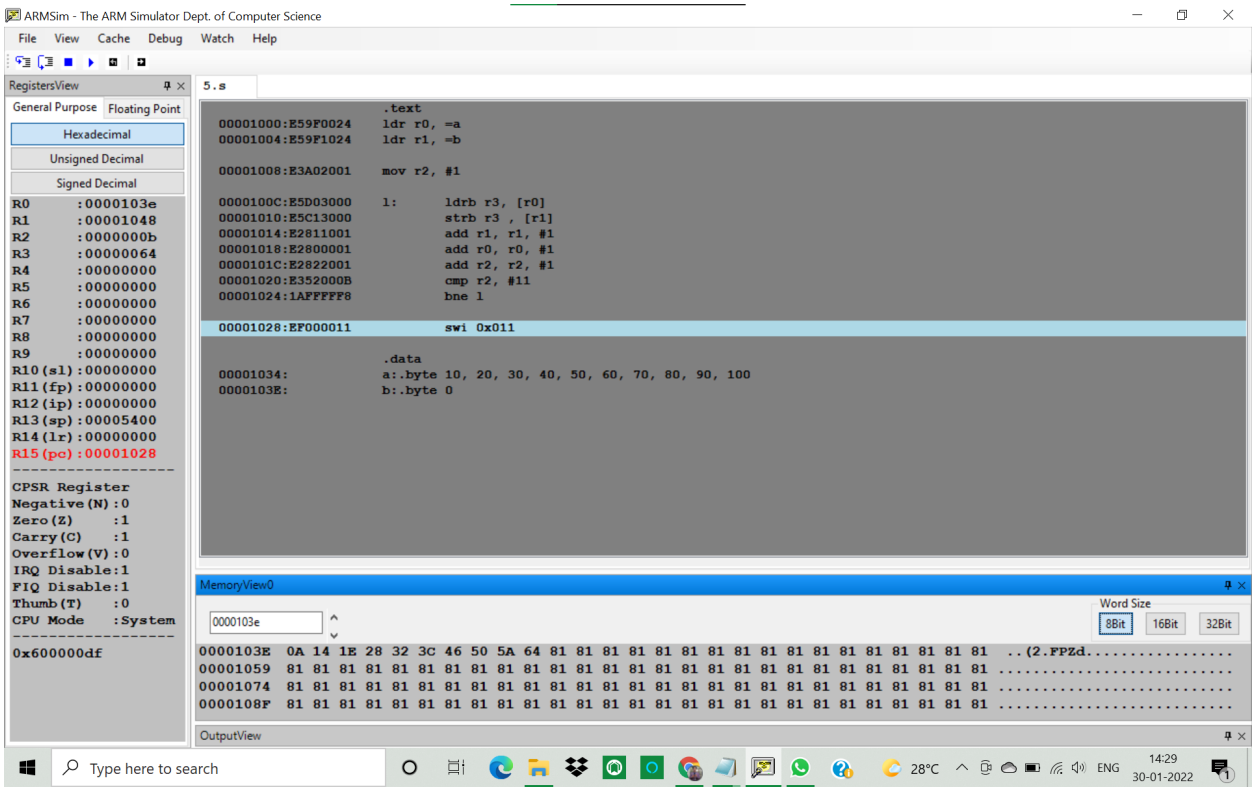

```
.text
ldr r0, =a
ldr r1, =b

mov r2, #1
mov r3, #0
l:      ldr r4, [r0]
        add r3 , r3 , r4

        add r0, r0, #4
        add r2, r2, #1
        cmp r2, #6
        bne l
        str r3, [r1]
        swi 0x011

.data
a:.word 10, 20, 30, 40, 50
b:.word 0
```



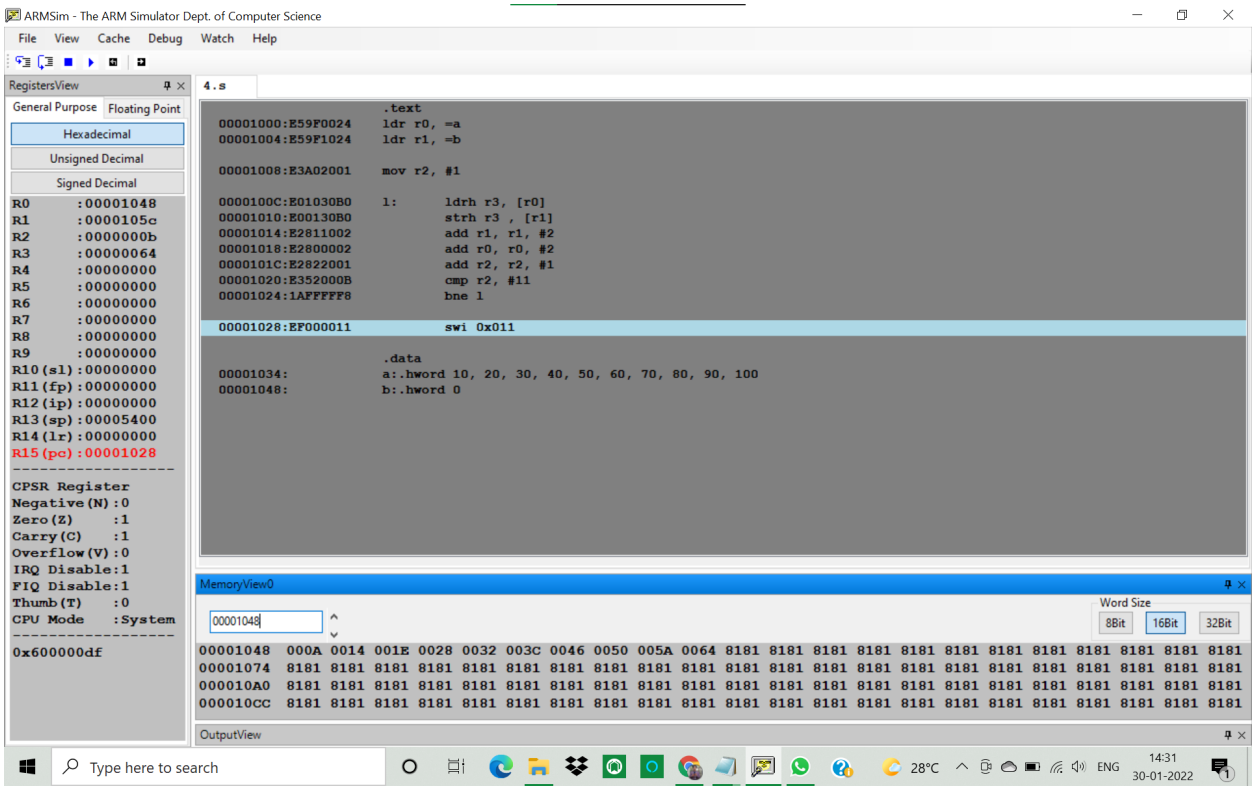
1. Write a program in ARM7TDMI–ISA to copy a block of N data items from Location A to Location B. a) Use Full word (.word directive) b) Use Half word(.Hword directive) c) Use Byte wise (.Byte directive)

```
.text
ldr r0, =a
ldr r1, =b

mov r2, #1

l:      ldrb r3, [r0]
        strb r3 , [r1]
        add r1, r1, #1
        add r0, r0, #1
        add r2, r2, #1
        cmp r2, #11
        bne l

        swi 0x011

.data
a:.byte 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
```

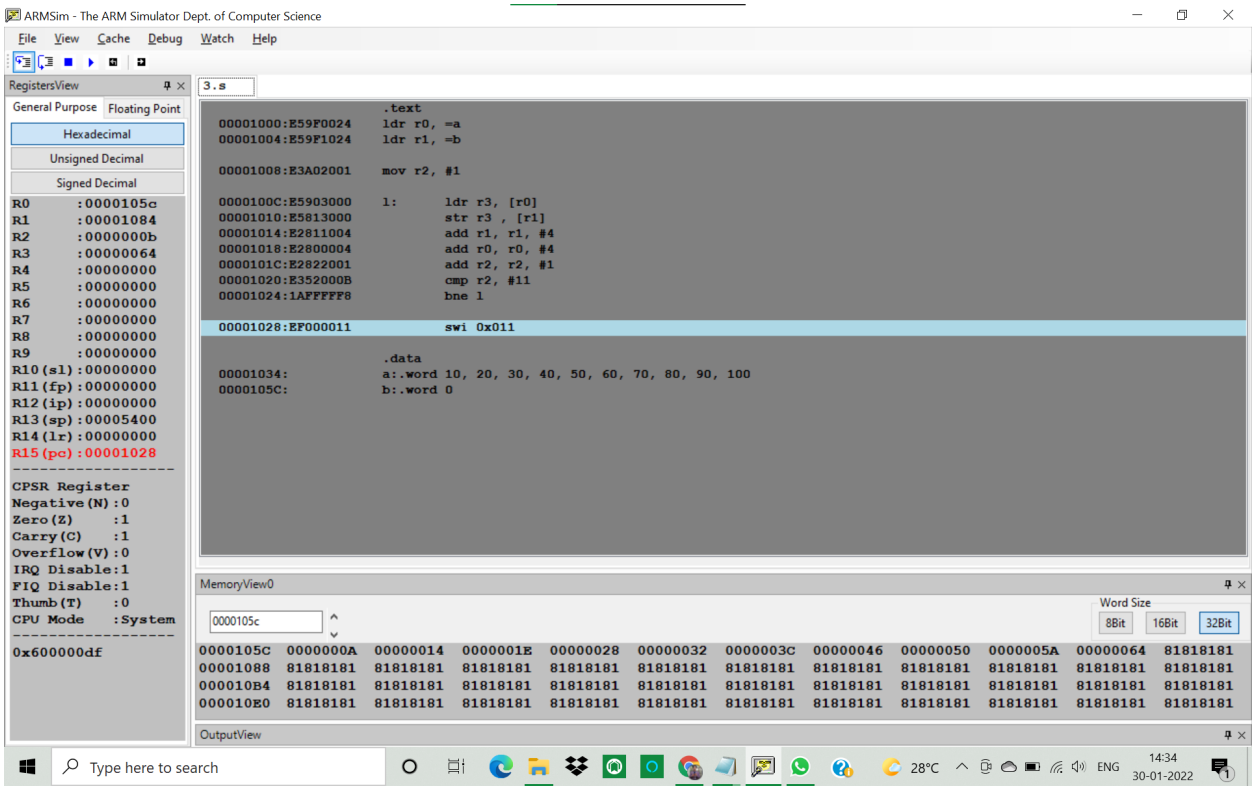
b:.byte 0




```
.text
ldr r0, =a
ldr r1, =b

mov r2, #1

l:      ldrh r3, [r0]
        strh r3 , [r1]
        add r1, r1, #2
        add r0, r0, #2
        add r2, r2, #1
        cmp r2, #11
        bne l

        swi 0x011

.data
a:.hword 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
b:.hword 0
```




```
.text
ldr r0, =a
ldr r1, =b

mov r2, #1

l:      ldr r3, [r0]
        str r3 , [r1]
        add r1, r1, #4
        add r0, r0, #4
        add r2, r2, #1
        cmp r2, #11
        bne l

        swi 0x011

.data
a:.word 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
b:.word 0
```


ARMSim - The ARM Simulator Dept. of Computer Science
File View Cache Debug Watch Help
RegistersView
General Purpose
Floating Point
Hexadecimal
Unsigned Decimal
Signed Decimal
R0 :0000105c
R1 :00001084
R2 :0000000b
R3 :00000064
R4 :00000000
R5 :00000000
R6 :00000000
R7 :00000000
R8 :00000000
R9 :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):00000000
R15(pc):00001028
CPSR Register
Negative(N):0
Zero(Z) :1
Carry(C) :1
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T) :0
CPU Mode :System
0x600000df
3.s
.text
00001000:E59F0024 ldr r0, =a
00001004:E59F1024 ldr r1, =b
00001008:E3A02001 mov r2, #1
0000100C:E5903000 l: ldr r3, [r0]
00001010:E5813000 str r3 , [r1]
00001014:E2811004 add r1, r1, #4
00001018:E2800004 add r0, r0, #4
0000101C:E2822001 add r2, r2, #1
00001020:E352000B cmp r2, #11
00001024:1AFFFFF8 bne l
00001028:EF000011 swi 0x011
.data
00001034: a:.word 10, 20, 30, 40, 50, 60, 70, 80, 90, 100
0000105C: b:.word 0
MemoryView0
0000105c
Word Size
8Bit
16Bit
32Bit
0000105C 0000000A 00000014 0000001E 00000028 00000032 0000003C 00000046 00000050 0000005A 00000064 81818181
00001088 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010B4 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
000010E0 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
OutputView
Type here to search
28°C
ENG
14:34
30-01-2022