PES UNIVERSITY

Department of Computer Science & Engineering
Microprocessor & Computer Architecture Lab

<u>Lab 3 Submission Format</u>

UE22CS251B

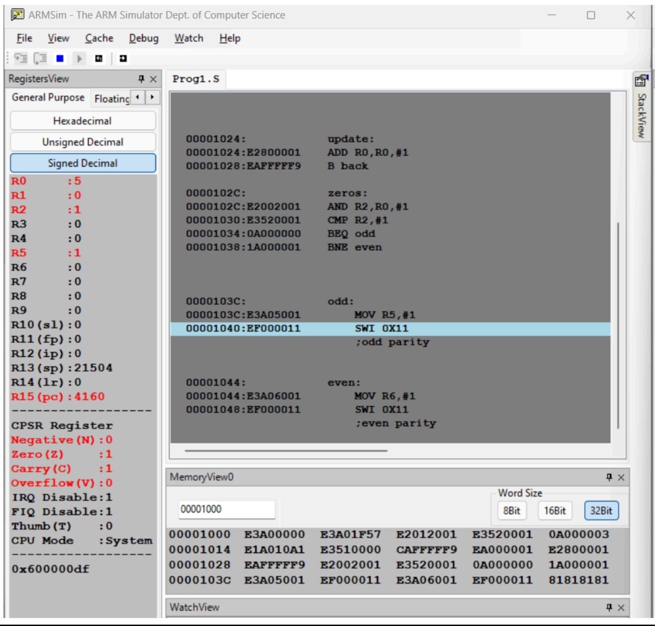
1 Write an ALP to check whether the given number has odd or even number of 1's (Even Parity and Odd Parity).

Program:

```
; Write an ALP to check whether the given number has odd or even number of 1's (Even Parity and Odd
Parity).


MOV R0,#0 ; counter

MOV R1,#0x5A ; input


testing:

AND R2,R1,#1

CMP R2,#1

BEQ update

back:

MOV R1,R1, LSR #1

CMP R1,#0

BGT testing

B zeros




update:

ADD R0,R0,#1

B back
```

```
zeros:

AND R2,R0,#1

CMP R2,#1

BEQ odd

BNE even




odd:

  MOV R5,#1

  SWI 0X11

  ;odd parity



even:

  MOV R6,#1

  SWI 0X11

  ;even parity
```

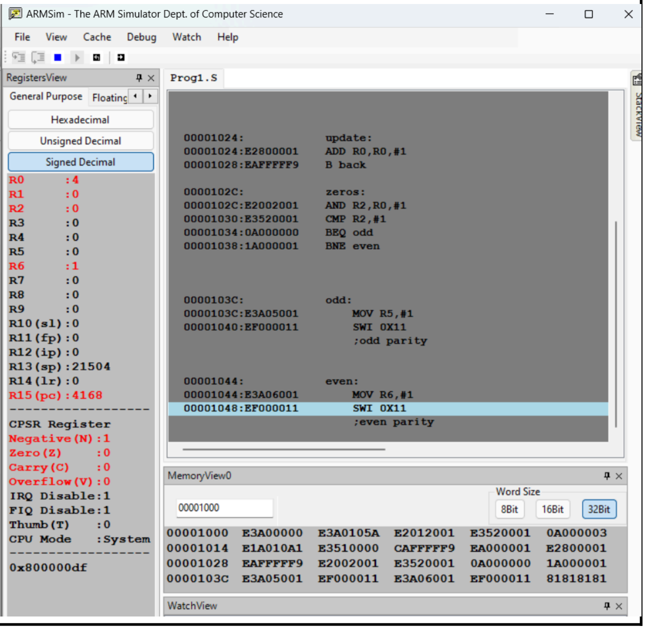
Output Screen Shot:



```
R0      :5
R1      :0
R2      :1
R3      :0
R4      :0
R5      :1
R6      :0
R7      :0
R8      :0
R9      :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):4160
------------------
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
```

```
00001024:             update:
00001024:E2800001     ADD R0,R0,#1
00001028:EAFFFFF9     B back

0000102C:             zeros:
0000102C:E2002001     AND R2,R0,#1
00001030:E3520001     CMP R2,#1
00001034:0A000000     BEQ odd
00001038:1A000001     BNE even



0000103C:             odd:
0000103C:E3A05001         MOV R5,#1
00001040:EF000011         SWI 0X11
                          ;odd parity


00001044:             even:
00001044:E3A06001         MOV R6,#1
00001048:EF000011         SWI 0X11
                          ;even parity
```

```
00001000
00001000  E3A00000  E3A01F57  E2012001  E3520001  0A000003
00001014  E1A010A1  E3510000  CAFFFFF9  EA000001  E2800001
00001028  EAFFFFF9  E2002001  E3520001  0A000000  1A000001
0000103C  E3A05001  EF000011  E3A06001  EF000011  81818181
```



```
R0      :4
R1      :0
R2      :0
R3      :0
R4      :0
R5      :0
R6      :1
R7      :0
R8      :0
R9      :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):4168
------------------
CPSR Register
Negative(N):1
Zero(Z)    :0
Carry(C)   :0
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T)   :0
CPU Mode   :System
------------------
0x800000df
```

```
00001024:             update:
00001024:E2800001     ADD R0,R0,#1
00001028:EAFFFFF9     B back

0000102C:             zeros:
0000102C:E2002001     AND R2,R0,#1
00001030:E3520001     CMP R2,#1
00001034:0A000000     BEQ odd
00001038:1A000001     BNE even



0000103C:             odd:
0000103C:E3A05001         MOV R5,#1
00001040:EF000011         SWI 0X11
                          ;odd parity


00001044:             even:
00001044:E3A06001         MOV R6,#1
00001048:EF000011         SWI 0X11
                          ;even parity
```

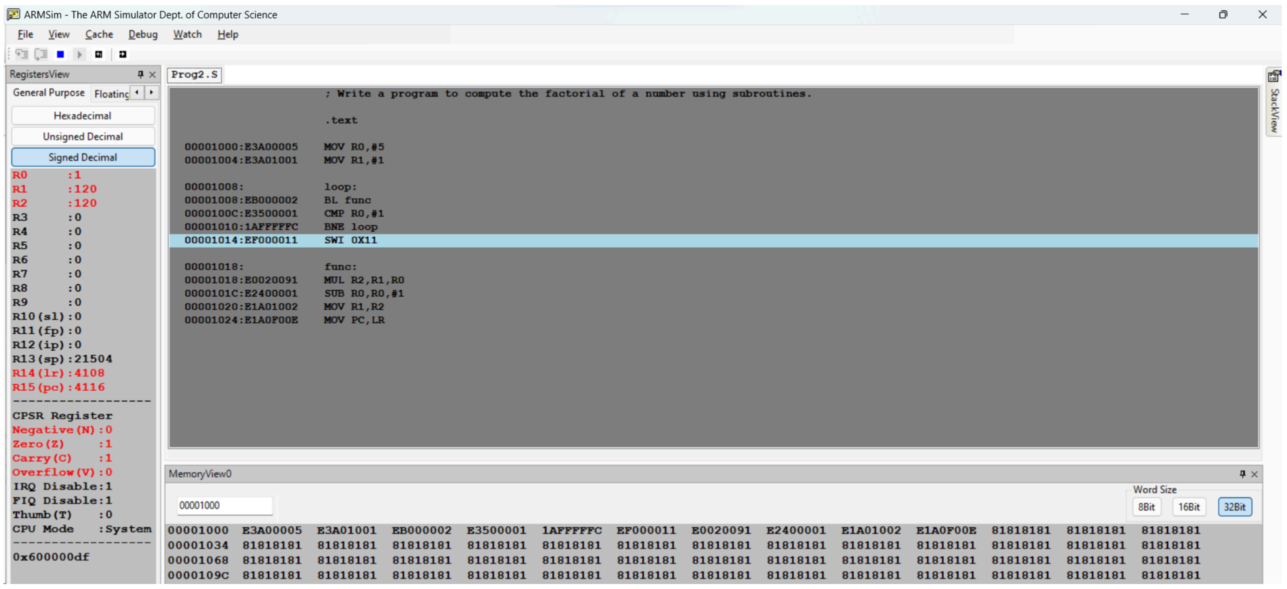
```
00001000
00001000  E3A00000  E3A0105A  E2012001  E3520001  0A000003
00001014  E1A010A1  E3510000  CAFFFFF9  EA000001  E2800001
00001028  EAFFFFF9  E2002001  E3520001  0A000000  1A000001
0000103C  E3A05001  EF000011  E3A06001  EF000011  81818181
```

2 Write a program to compute the factorial of a number using subroutines.

Program:

```
; Write a program to compute the factorial of a number using subroutines.

.text

MOV R0,#5
MOV R1,#1

loop:
BL func
CMP R0,#1
BNE loop
SWI 0X11

func:
MUL R2,R1,R0
SUB R0,R0,#1
MOV R1,R2
MOV PC,LR
```

Output Screen Shot:



```
                    ; Write a program to compute the factorial of a number using subroutines.

                    .text

00001000:E3A00005   MOV R0,#5
00001004:E3A01001   MOV R1,#1

00001008:           loop:
00001008:EB000002   BL func
0000100C:E3500001   CMP R0,#1
00001010:1AFFFFFC   BNE loop
00001014:EF000011   SWI 0X11

00001018:           func:
00001018:E0020091   MUL R2,R1,R0
0000101C:E2400001   SUB R0,R0,#1
00001020:E1A01002   MOV R1,R2
00001024:E1A0F00E   MOV PC,LR
```

```
R0       :1
R1       :120
R2       :120
R3       :0
R4       :0
R5       :0
R6       :0
R7       :0
R8       :0
R9       :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):4108
R15(pc):4116
------------------
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
```

```
00001000
00001000 E3A00005 E3A01001 EB000002 E3500001 1AFFFFFC EF000011 E0020091 E2400001 E1A01002 E1A0F00E 81818181 81818181 81818181
00001034 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
00001068 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
0000109C 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181 81818181
```

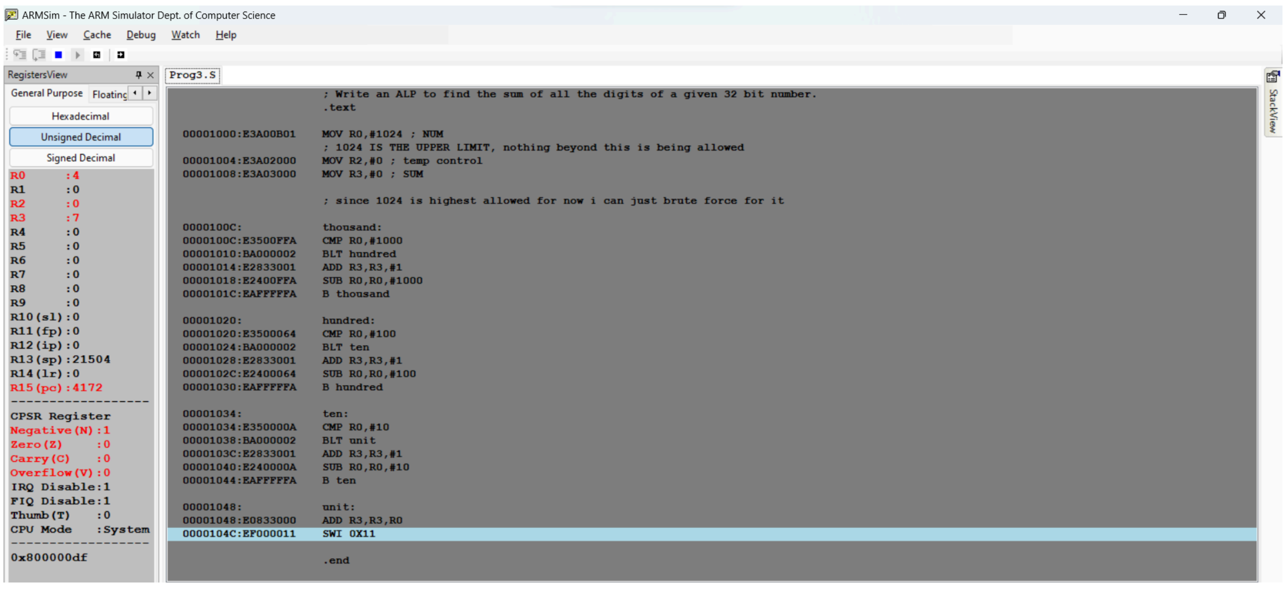
3 Write an ALP to find the sum of all the digits of a given 32 bit number.

Program:

```
; Write an ALP to find the sum of all the digits of a given 32 bit number.

.text


MOV R0,#1024 ; NUM

; 1024 IS THE UPPER LIMIT, nothing beyond this is being allowed

MOV R2,#0 ; temp control

MOV R3,#0 ; SUM


thousand:

CMP R0,#1000

BLT hundred

ADD R3,R3,#1

SUB R0,R0,#1000

B thousand


hundred:

CMP R0,#100

BLT ten
```

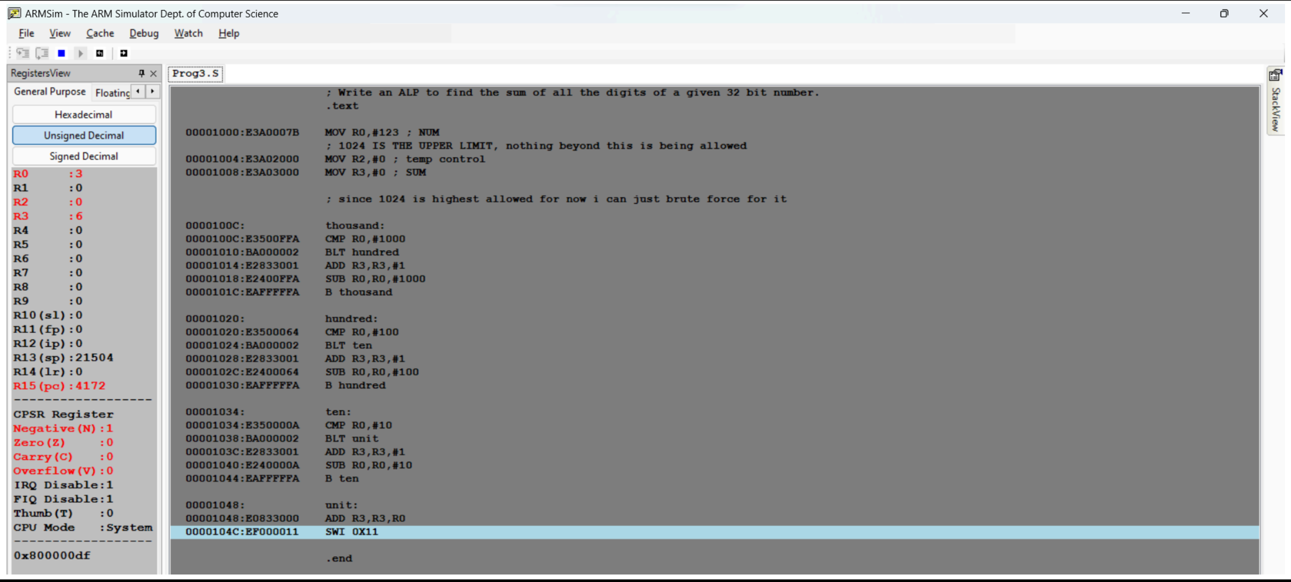
```
ADD R3,R3,#1

SUB R0,R0,#100

B hundred


ten:

CMP R0,#10

BLT unit

ADD R3,R3,#1

SUB R0,R0,#10

B ten


unit:

ADD R3,R3,R0

SWI 0X11


.end
```

Output Screen Shot:



```
R0       :4
R1       :0
R2       :0
R3       :7
R4       :0
R5       :0
R6       :0
R7       :0
R8       :0
R9       :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):4172
------------------
CPSR Register
Negative(N):1
Zero(Z)     :0
Carry(C)    :0
Overflow(V):0
IRQ Disable:1
FIQ Disable:1
Thumb(T)    :0
CPU Mode    :System
------------------
0x800000df
```

```
                    ; Write an ALP to find the sum of all the digits of a given 32 bit number.
                    .text

00001000:E3A00B01   MOV R0,#1024 ; NUM
                    ; 1024 IS THE UPPER LIMIT, nothing beyond this is being allowed
00001004:E3A02000   MOV R2,#0 ; temp control
00001008:E3A03000   MOV R3,#0 ; SUM

                    ; since 1024 is highest allowed for now i can just brute force for it

0000100C:           thousand:
0000100C:E3500FFA   CMP R0,#1000
00001010:BA000002   BLT hundred
00001014:E2833001   ADD R3,R3,#1
00001018:E2400FFA   SUB R0,R0,#1000
0000101C:EAFFFFFA   B thousand

00001020:           hundred:
00001020:E3500064   CMP R0,#100
00001024:BA000002   BLT ten
00001028:E2833001   ADD R3,R3,#1
0000102C:E2400064   SUB R0,R0,#100
00001030:EAFFFFFA   B hundred

00001034:           ten:
00001034:E350000A   CMP R0,#10
00001038:BA000002   BLT unit
0000103C:E2833001   ADD R3,R3,#1
00001040:E240000A   SUB R0,R0,#10
00001044:EAFFFFFA   B ten

00001048:           unit:
00001048:E0833000   ADD R3,R3,R0
0000104C:EF000011   SWI 0X11

                    .end
```



```
R0       :3
R1       :0
R2       :0
R3       :6
R4       :0
R5       :0
R6       :0
R7       :0
R8       :0
R9       :0
R10(sl)  :0
R11(fp)  :0
R12(ip)  :0
R13(sp)  :21504
R14(lr)  :0
R15(pc)  :4172
------------------
CPSR Register
Negative(N) :1
Zero(Z)     :0
Carry(C)    :0
Overflow(V) :0
IRQ Disable:1
FIQ Disable:1
Thumb(T)    :0
CPU Mode    :System
------------------
0x800000df
```

```
                        ; Write an ALP to find the sum of all the digits of a given 32 bit number.
                        .text

00001000:E3A0007B       MOV R0,#123 ; NUM
                        ; 1024 IS THE UPPER LIMIT, nothing beyond this is being allowed
00001004:E3A02000       MOV R2,#0 ; temp control
00001008:E3A03000       MOV R3,#0 ; SUM

                        ; since 1024 is highest allowed for now i can just brute force for it

0000100C:               thousand:
0000100C:E3500FFA       CMP R0,#1000
00001010:BA000002       BLT hundred
00001014:E2833001       ADD R3,R3,#1
00001018:E2400FFA       SUB R0,R0,#1000
0000101C:EAFFFFFA       B thousand

00001020:               hundred:
00001020:E3500064       CMP R0,#100
00001024:BA000002       BLT ten
00001028:E2833001       ADD R3,R3,#1
0000102C:E2400064       SUB R0,R0,#100
00001030:EAFFFFFA       B hundred

00001034:               ten:
00001034:E350000A       CMP R0,#10
00001038:BA000002       BLT unit
0000103C:E2833001       ADD R3,R3,#1
00001040:E240000A       SUB R0,R0,#10
00001044:EAFFFFFA       B ten

00001048:               unit:
00001048:E0833000       ADD R3,R3,R0
0000104C:EF000011       SWI 0X11

                        .end
```

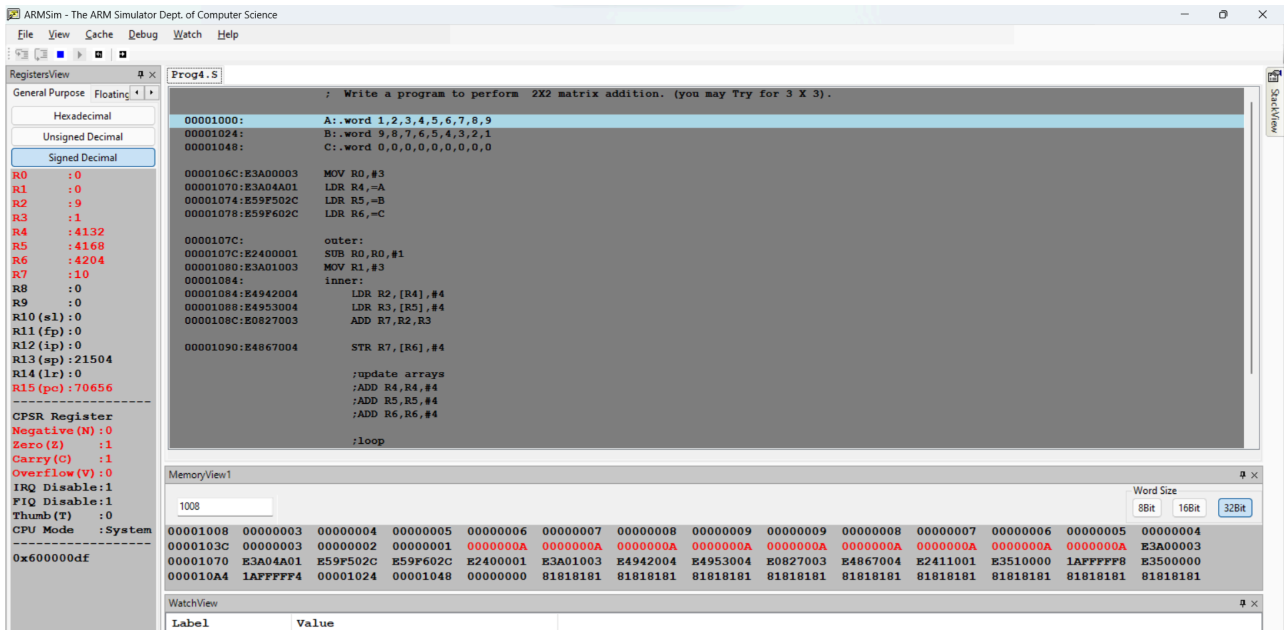
4 Write a program to perform 2X2 matrix addition. (you may Try for 3 X 3).

Program:

```
;  Write a program to perform  2X2 matrix addition. (you may Try for 3 X 3).


A:.word 1,2,3,4,5,6,7,8,9

B:.word 9,8,7,6,5,4,3,2,1

C:.word 0,0,0,0,0,0,0,0,0


MOV R0,#3

LDR R4,=A

LDR R5,=B

LDR R6,=C


outer:

SUB R0,R0,#1

MOV R1,#3

inner:

   LDR R2,[R4],#4

   LDR R3,[R5],#4

   ADD R7,R2,R3
```

```
    STR R7,[R6],#4

    ;update arrays
    ;ADD R4,R4,#4
    ;ADD R5,R5,#4
    ;ADD R6,R6,#4

    ;loop
    SUB R1,R1,#1
    CMP R1,#0
    BNE inner

CMP R0,#0
BNE outer
```

Output Screen Shot:



```
R0        :0
R1        :0
R2        :9
R3        :1
R4        :4132
R5        :4168
R6        :4204
R7        :10
R8        :0
R9        :0
R10(sl):0
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):70656
-------------------
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
```

```
                   ;  Write a program to perform  2X2 matrix addition. (you may Try for 3 X 3).

00001000:          A:.word 1,2,3,4,5,6,7,8,9
00001024:          B:.word 9,8,7,6,5,4,3,2,1
00001048:          C:.word 0,0,0,0,0,0,0,0,0

0000106C:E3A00003  MOV R0,#3
00001070:E3A04A01  LDR R4,=A
00001074:E59F502C  LDR R5,=B
00001078:E59F602C  LDR R6,=C

0000107C:          outer:
0000107C:E2400001  SUB R0,R0,#1
00001080:E3A01003  MOV R1,#3
00001084:          inner:
00001084:E4942004      LDR R2,[R4],#4
00001088:E4953004      LDR R3,[R5],#4
0000108C:E0827003      ADD R7,R2,R3

00001090:E4867004      STR R7,[R6],#4

                       ;update arrays
                       ;ADD R4,R4,#4
                       ;ADD R5,R5,#4
                       ;ADD R6,R6,#4

                       ;loop
```

```
1008
00001008  00000003  00000004  00000005  00000006  00000007  00000008  00000009  00000009  00000008  00000007  00000006  00000005  00000004
0000103C  00000003  00000002  00000001  0000000A  0000000A  0000000A  0000000A  0000000A  0000000A  0000000A  0000000A  0000000A  E3A00003
00001070  E3A04A01  E59F502C  E59F602C  E2400001  E3A01003  E4942004  E4953004  E0827003  E4867004  E2411001  E3510000  1AFFFFF8  E3500000
000010A4  1AFFFFF4  00001024  00001048  00000000  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181  81818181
```

```
Label            Value
```

5 Write a program to search for an element in an array using Linear search technique

Program:

; Write a program to search for an element in an array using Linear search technique

A:.word 10,24,26,27,28,19,20,69,70,67

MOV R0,#10 ; loop counter var

LDR R1,=A

MOV R2,#68 ; key

MOV R3,#0 ; 1 if found

loop:

LDR R4,[R1],#4

CMP R4,R2

BEQ found

SUB R0,R0,#1

CMP R0,#0

BNE loop

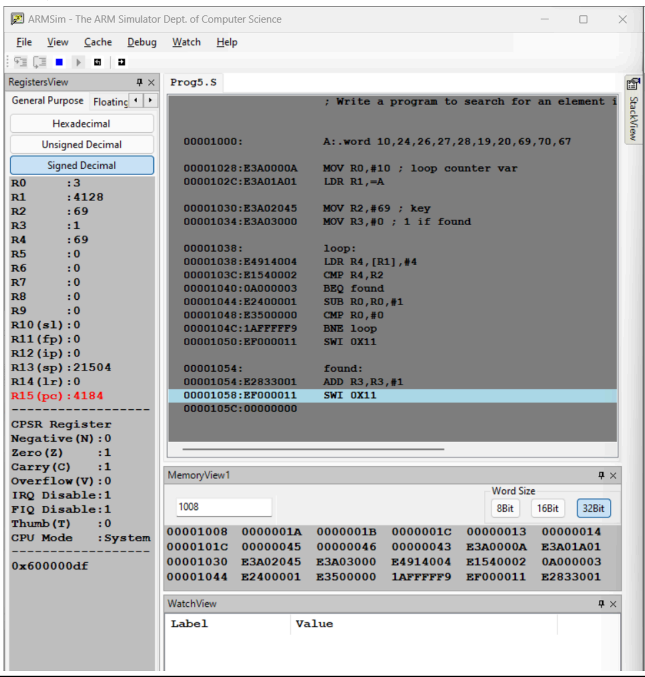
SWI 0X11

found:

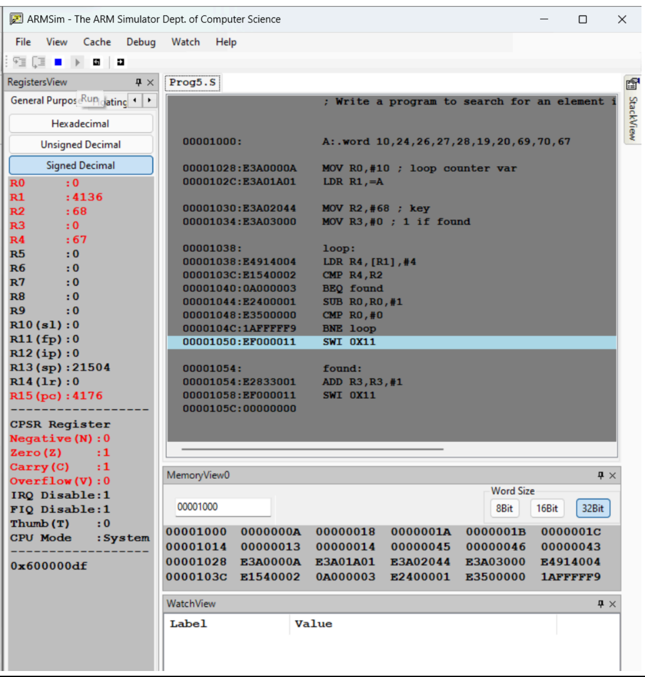
ADD R3,R3,#1

SWI 0X11

Output Screen Shot:



```
ARMSim - The ARM Simulator Dept. of Computer Science
RegistersView                 Prog5.S
General Purpose  Floating
Hexadecimal
Unsigned Decimal
Signed Decimal
R0       :3              ; Write a program to search for an element i
R1       :4128
R2       :69             00001000:             A:.word 10,24,26,27,28,19,20,69,70,67
R3       :1
R4       :69             00001028:E3A0000A     MOV R0,#10 ; loop counter var
R5       :0              0000102C:E3A01A01     LDR R1,=A
R6       :0
R7       :0              00001030:E3A02045     MOV R2,#69 ; key
R8       :0              00001034:E3A03000     MOV R3,#0 ; 1 if found
R9       :0
R10(sl):0                00001038:             loop:
R11(fp):0                00001038:E4914004     LDR R4,[R1],#4
R12(ip):0                0000103C:E1540002     CMP R4,R2
R13(sp):21504            00001040:0A000003     BEQ found
R14(lr):0                00001044:E2400001     SUB R0,R0,#1
R15(pc):4184             00001048:E3500000     CMP R0,#0
------------------       0000104C:1AFFFFF9     BNE loop
CPSR Register            00001050:EF000011     SWI 0X11
Negative(N):0
Zero(Z)    :1            00001054:             found:
Carry(C)   :1            00001054:E2833001     ADD R3,R3,#1
Overflow(V):0            00001058:EF000011     SWI 0X11
IRQ Disable:1            0000105C:00000000
FIQ Disable:1
Thumb(T)   :0
CPU Mode   :System
------------------
0x600000df

MemoryView1
1008                                   Word Size  8Bit  16Bit  32Bit
00001008  0000001A  0000001B  0000001C  00000013  00000014
0000101C  00000045  00000046  00000043  E3A0000A  E3A01A01
00001030  E3A02045  E3A03000  E4914004  E1540002  0A000003
00001044  E2400001  E3500000  1AFFFFF9  EF000011  E2833001

WatchView
Label        Value
```



```
ARMSim - The ARM Simulator Dept. of Computer Science
RegistersView                 Prog5.S
General Purpos Run lating
Hexadecimal
Unsigned Decimal
Signed Decimal
R0       :0              ; Write a program to search for an element i
R1       :4136
R2       :68             00001000:             A:.word 10,24,26,27,28,19,20,69,70,67
R3       :0
R4       :67             00001028:E3A0000A     MOV R0,#10 ; loop counter var
R5       :0              0000102C:E3A01A01     LDR R1,=A
R6       :0
R7       :0              00001030:E3A02044     MOV R2,#68 ; key
R8       :0              00001034:E3A03000     MOV R3,#0 ; 1 if found
R9       :0
R10(sl):0                00001038:             loop:
R11(fp):0                00001038:E4914004     LDR R4,[R1],#4
R12(ip):0                0000103C:E1540002     CMP R4,R2
R13(sp):21504            00001040:0A000003     BEQ found
R14(lr):0                00001044:E2400001     SUB R0,R0,#1
R15(pc):4176             00001048:E3500000     CMP R0,#0
------------------       0000104C:1AFFFFF9     BNE loop
CPSR Register            00001050:EF000011     SWI 0X11
Negative(N):0
Zero(Z)    :1            00001054:             found:
Carry(C)   :1            00001054:E2833001     ADD R3,R3,#1
Overflow(V):0            00001058:EF000011     SWI 0X11
IRQ Disable:1            0000105C:00000000
FIQ Disable:1
Thumb(T)   :0
CPU Mode   :System
------------------
0x600000df

MemoryView0
00001000                               Word Size  8Bit  16Bit  32Bit
00001000  0000000A  00000018  0000001A  0000001B  0000001C
00001014  00000013  00000014  00000045  00000046  00000043
00001028  E3A0000A  E3A01A01  E3A02044  E3A03000  E4914004
0000103C  E1540002  0A000003  E2400001  E3500000  1AFFFFF9

WatchView
Label        Value
```

6 Assignment Questions:

i) Write a program to search for an element in an array using binary search technique.

Program:

; Write a program to search for an element in an array using binary search technique.

; for binary search, i assume the given array is sorted

.DATA

A:.word 2,4,5,7,9,10,13,15,17,19 ; assuming these 10 elements

END:

.text

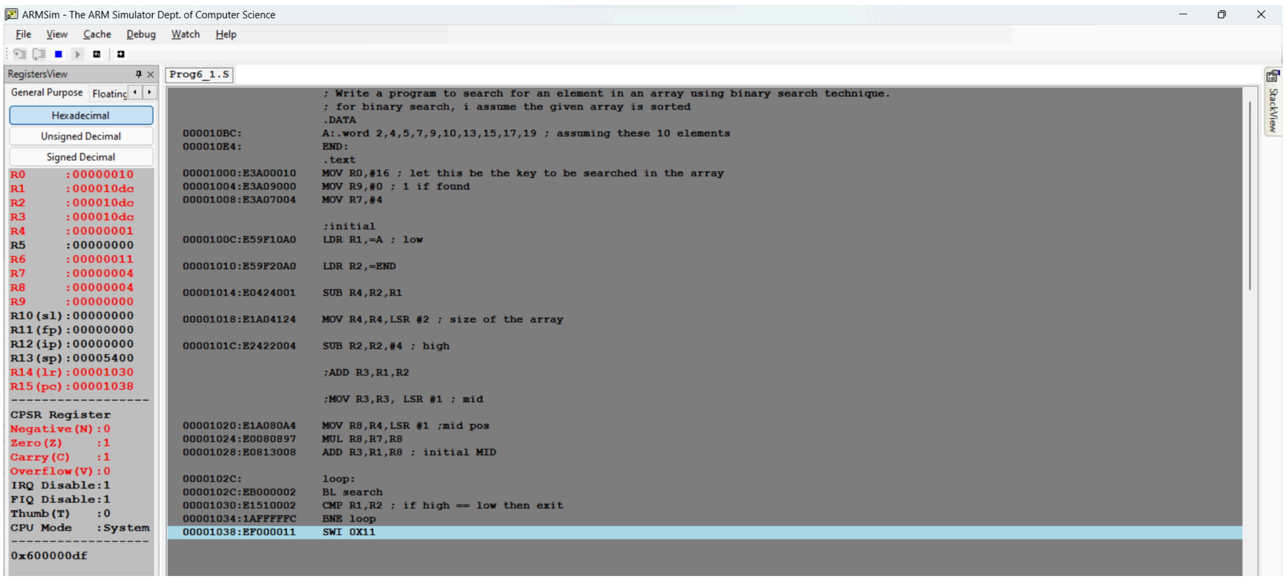
MOV R0,#15 ; let this be the key to be searched in the array

```
MOV R9,#0 ; 1 if found
MOV R7,#4

;initial
LDR R1,=A ; low

LDR R2,=END

SUB R4,R2,R1

MOV R4,R4,LSR #2 ; size of the array

SUB R2,R2,#4 ; high

;ADD R3,R1,R2

;MOV R3,R3, LSR #1 ; mid

MOV R8,R4,LSR #1 ;mid pos
MUL R8,R7,R8
ADD R3,R1,R8 ; initial MID

loop:
BL search
CMP R1,R2 ; if high == low then exit
BNE loop
SWI 0X11
```

```
search:
LDR R6,[R3]
CMP R0,R6
BEQ found
BLT lower
BGT higher
back:
MOV PC,LR

lower:
MOV R2,R3 ;UPDATE HIGH
; ADD R3,R2,R1
; MOV R3,R3,LSR #1 ;UPDATE MID
SUB R4,R2,R1
MOV R4,R4, LSR #2
CMP R4,#1
BEQ lowhelp

MOV R8,R4,LSR #1
MUL R8,R7,R8
ADD R3,R1,R8

B back



higher:
MOV R1,R3 ; UPDATE LOW
; ADD R3,R2,R1
; MOV R3,R3,LSR #1 ;UPDATE MID
SUB R4,R2,R1
```

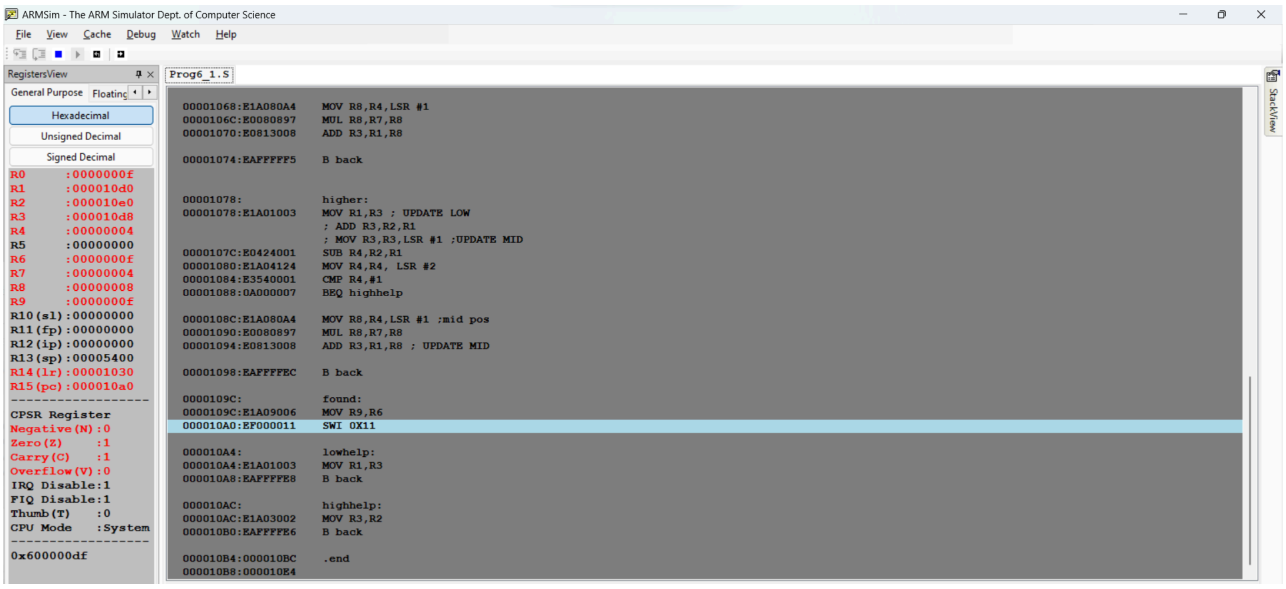
```
MOV R4,R4, LSR #2

CMP R4,#1

BEQ highhelp


MOV R8,R4,LSR #1 ;mid pos

MUL R8,R7,R8

ADD R3,R1,R8 ; UPDATE MID


B back


found:

MOV R9,R6

SWI 0X11


lowhelp:

MOV R1,R3

B back


highhelp:

MOV R3,R2

B back


.end
```

Output Screen Shot:



```
R0        :00000010
R1        :000010dc
R2        :000010dc
R3        :000010dc
R4        :00000001
R5        :00000000
R6        :00000011
R7        :00000004
R8        :00000004
R9        :00000000
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):00001030
R15(pc):00001038
------------------
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
```

```
                        ; Write a program to search for an element in an array using binary search technique.
                        ; for binary search, i assume the given array is sorted
                        .DATA
000010BC:               A:.word 2,4,5,7,9,10,13,15,17,19 ; assuming these 10 elements
000010E4:               END:
                        .text
00001000:E3A00010       MOV R0,#16 ; let this be the key to be searched in the array
00001004:E3A09000       MOV R9,#0 ; 1 if found
00001008:E3A07004       MOV R7,#4

                        ;initial
0000100C:E59F10A0       LDR R1,=A ; low

00001010:E59F20A0       LDR R2,=END

00001014:E0424001       SUB R4,R2,R1

00001018:E1A04124       MOV R4,R4,LSR #2 ; size of the array

0000101C:E2422004       SUB R2,R2,#4 ; high

                        ;ADD R3,R1,R2

                        ;MOV R3,R3, LSR #1 ; mid

00001020:E1A080A4       MOV R8,R4,LSR #1 ;mid pos
00001024:E0080897       MUL R8,R7,R8
00001028:E0813008       ADD R3,R1,R8 ; initial MID

0000102C:               loop:
0000102C:EB000002       BL search
00001030:E1510002       CMP R1,R2 ; if high == low then exit
00001034:1AFFFFFC       BNE loop
00001038:EF000011       SWI 0X11
```



```
R0        :0000000f
R1        :000010d0
R2        :000010e0
R3        :000010d8
R4        :00000004
R5        :00000000
R6        :0000000f
R7        :00000004
R8        :00000008
R9        :0000000f
R10(sl):00000000
R11(fp):00000000
R12(ip):00000000
R13(sp):00005400
R14(lr):00001030
R15(pc):000010a0
------------------
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
```

```
00001068:E1A080A4       MOV R8,R4,LSR #1
0000106C:E0080897       MUL R8,R7,R8
00001070:E0813008       ADD R3,R1,R8

00001074:EAFFFFF5       B back


00001078:               higher:
00001078:E1A01003       MOV R1,R3 ; UPDATE LOW
                        ; ADD R3,R2,R1
                        ; MOV R3,R3,LSR #1 ;UPDATE MID
0000107C:E0424001       SUB R4,R2,R1
00001080:E1A04124       MOV R4,R4, LSR #2
00001084:E3540001       CMP R4,#1
00001088:0A000007       BEQ highhelp

0000108C:E1A080A4       MOV R8,R4,LSR #1 ;mid pos
00001090:E0080897       MUL R8,R7,R8
00001094:E0813008       ADD R3,R1,R8 ; UPDATE MID

00001098:EAFFFFEC       B back

0000109C:               found:
0000109C:E1A09006       MOV R9,R6
000010A0:EF000011       SWI 0X11

000010A4:               lowhelp:
000010A4:E1A01003       MOV R1,R3
000010A8:EAFFFFE8       B back

000010AC:               highhelp:
000010AC:E1A03002       MOV R3,R2
000010B0:EAFFFFE6       B back

000010B4:000010BC       .end
000010B8:000010E4
```

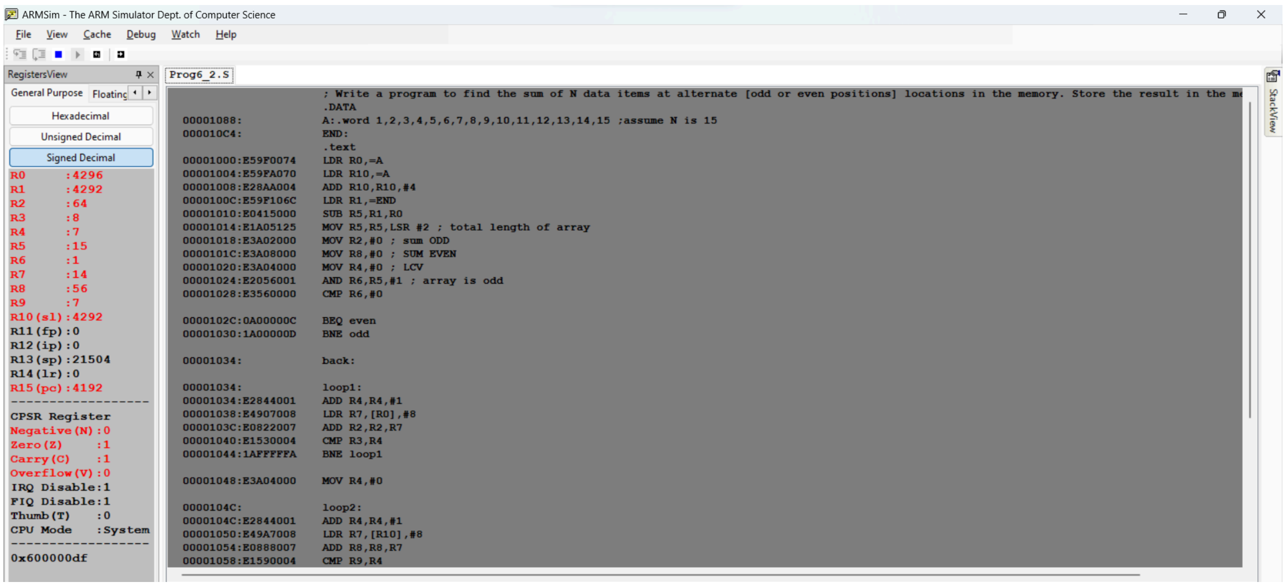
ii)Write a program to find the sum of N data items at alternate [odd or even positions] locations in the memory. Store the result in the memory location.

Program:

```
; Write a program to find the sum of N data items at alternate [odd or even positions] locations in the memory. Store the result in the memory location.

.DATA

A:.word 1,2,3,4,5,6,7,8,9,10,11,12,13,14 ;assume N is 14

END:

.text

LDR R0,=A

LDR R10,=A

ADD R10,R10,#4

LDR R1,=END

SUB R5,R1,R0

MOV R5,R5,LSR #2 ; total length of array

MOV R2,#0 ; sum ODD

MOV R8,#0 ; SUM EVEN

MOV R4,#0 ; LCV

AND R6,R5,#1 ; array is odd

CMP R6,#0


BEQ even

BNE odd


back:


loop1:

ADD R4,R4,#1

LDR R7,[R0],#8

ADD R2,R2,R7

CMP R3,R4
```

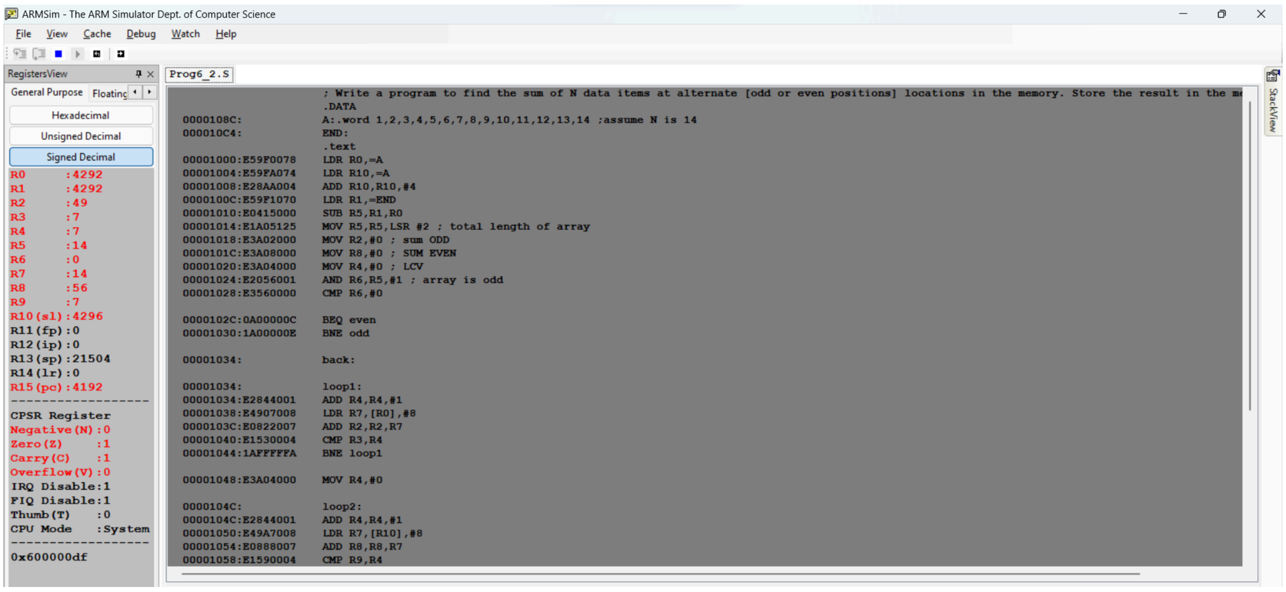
```
BNE loop1

MOV R4,#0

loop2:
ADD R4,R4,#1
LDR R7,[R10],#8
ADD R8,R8,R7
CMP R9,R4
BNE loop2
SWI 0X11

even:
MOV R3,R5,LSR #1
MOV R9,R3
B back

odd:
MOV R3,R5,LSR #1
MOV R9,R3
ADD R3,R3,#1
B back

.end
```

Output Screen Shot:



```
R0       :4296
R1       :4292
R2       :64
R3       :8
R4       :7
R5       :15
R6       :1
R7       :14
R8       :56
R9       :7
R10(sl):4292
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):4192
------------------
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
```

```
                    ; Write a program to find the sum of N data items at alternate [odd or even positions] locations in the memory. Store the result in the me
                    .DATA
00001088:           A:.word 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15 ;assume N is 15
000010C4:           END:
                    .text
00001000:E59F0074   LDR R0,=A
00001004:E59FA070   LDR R10,=A
00001008:E28AA004   ADD R10,R10,#4
0000100C:E59F106C   LDR R1,=END
00001010:E0415000   SUB R5,R1,R0
00001014:E1A05125   MOV R5,R5,LSR #2 ; total length of array
00001018:E3A02000   MOV R2,#0 ; sum ODD
0000101C:E3A08000   MOV R8,#0 ; SUM EVEN
00001020:E3A04000   MOV R4,#0 ; LCV
00001024:E2056001   AND R6,R5,#1 ; array is odd
00001028:E3560000   CMP R6,#0

0000102C:0A00000C   BEQ even
00001030:1A00000D   BNE odd

00001034:           back:

00001034:           loop1:
00001034:E2844001   ADD R4,R4,#1
00001038:E4907008   LDR R7,[R0],#8
0000103C:E0822007   ADD R2,R2,R7
00001040:E1530004   CMP R3,R4
00001044:1AFFFFFA   BNE loop1

00001048:E3A04000   MOV R4,#0

0000104C:           loop2:
0000104C:E2844001   ADD R4,R4,#1
00001050:E49A7008   LDR R7,[R10],#8
00001054:E0888007   ADD R8,R8,R7
00001058:E1590004   CMP R9,R4
```



```
R0       :4292
R1       :4292
R2       :49
R3       :7
R4       :7
R5       :14
R6       :0
R7       :14
R8       :56
R9       :7
R10(sl):4296
R11(fp):0
R12(ip):0
R13(sp):21504
R14(lr):0
R15(pc):4192
------------------
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
```

```
                    ; Write a program to find the sum of N data items at alternate [odd or even positions] locations in the memory. Store the result in the me
                    .DATA
0000108C:           A:.word 1,2,3,4,5,6,7,8,9,10,11,12,13,14 ;assume N is 14
000010C4:           END:
                    .text
00001000:E59F0078   LDR R0,=A
00001004:E59FA074   LDR R10,=A
00001008:E28AA004   ADD R10,R10,#4
0000100C:E59F1070   LDR R1,=END
00001010:E0415000   SUB R5,R1,R0
00001014:E1A05125   MOV R5,R5,LSR #2 ; total length of array
00001018:E3A02000   MOV R2,#0 ; sum ODD
0000101C:E3A08000   MOV R8,#0 ; SUM EVEN
00001020:E3A04000   MOV R4,#0 ; LCV
00001024:E2056001   AND R6,R5,#1 ; array is odd
00001028:E3560000   CMP R6,#0

0000102C:0A00000C   BEQ even
00001030:1A00000E   BNE odd

00001034:           back:

00001034:           loop1:
00001034:E2844001   ADD R4,R4,#1
00001038:E4907008   LDR R7,[R0],#8
0000103C:E0822007   ADD R2,R2,R7
00001040:E1530004   CMP R3,R4
00001044:1AFFFFFA   BNE loop1

00001048:E3A04000   MOV R4,#0

0000104C:           loop2:
0000104C:E2844001   ADD R4,R4,#1
00001050:E49A7008   LDR R7,[R10],#8
00001054:E0888007   ADD R8,R8,R7
00001058:E1590004   CMP R9,R4
```