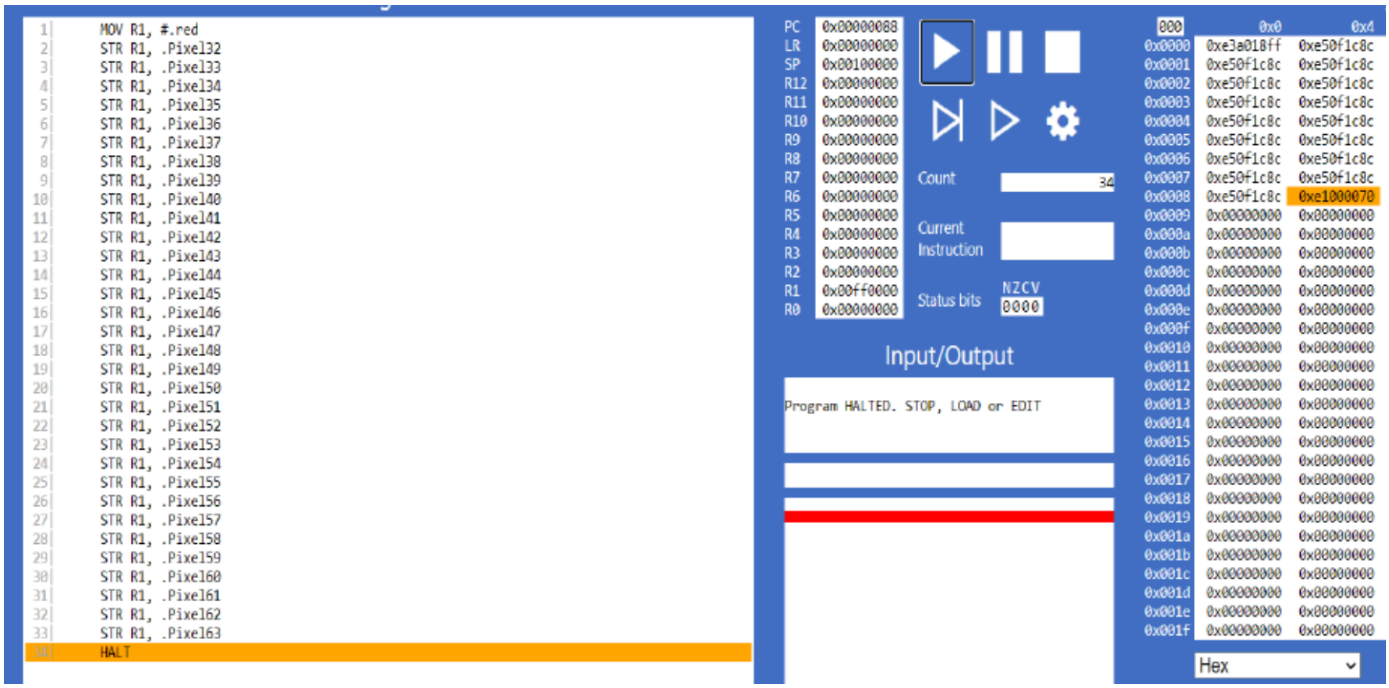
COS-10004 (Computer Systems)

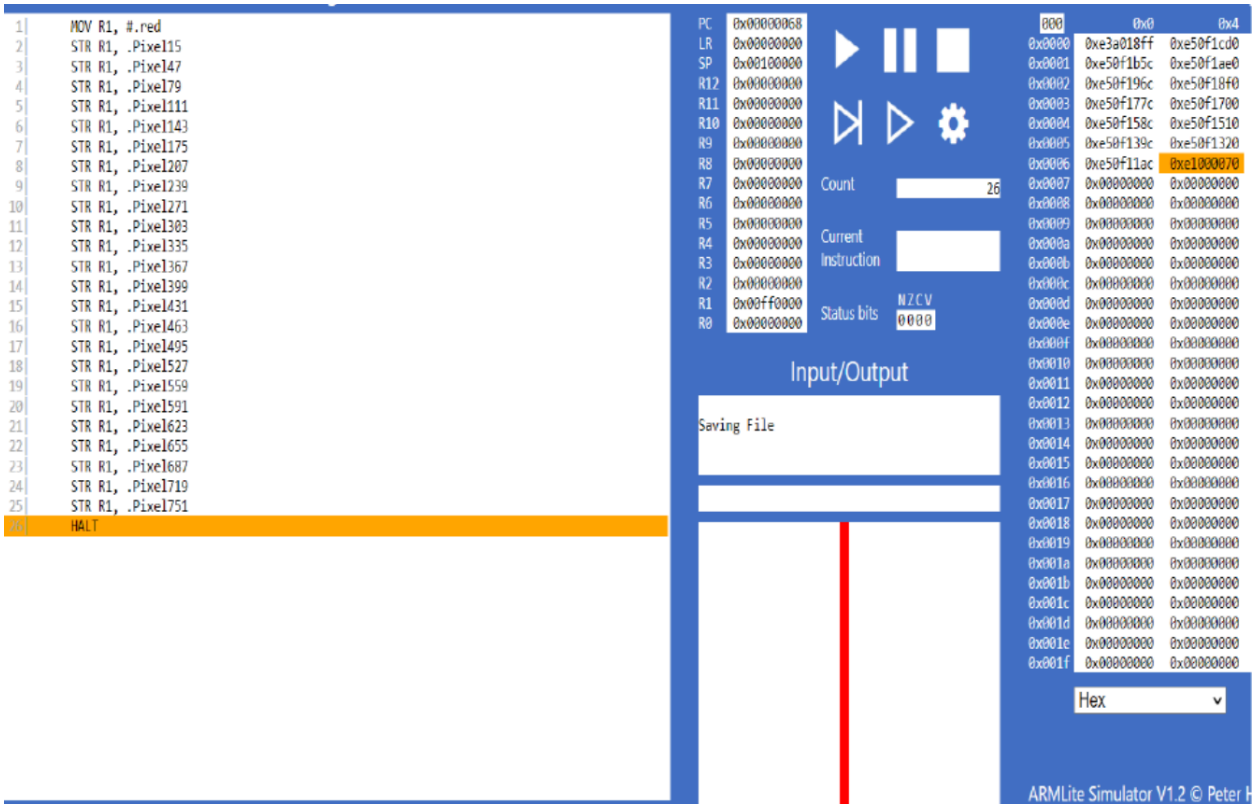
Name: Dave Nguyen (Nguyen Quang Anh).

ID: 104697710.

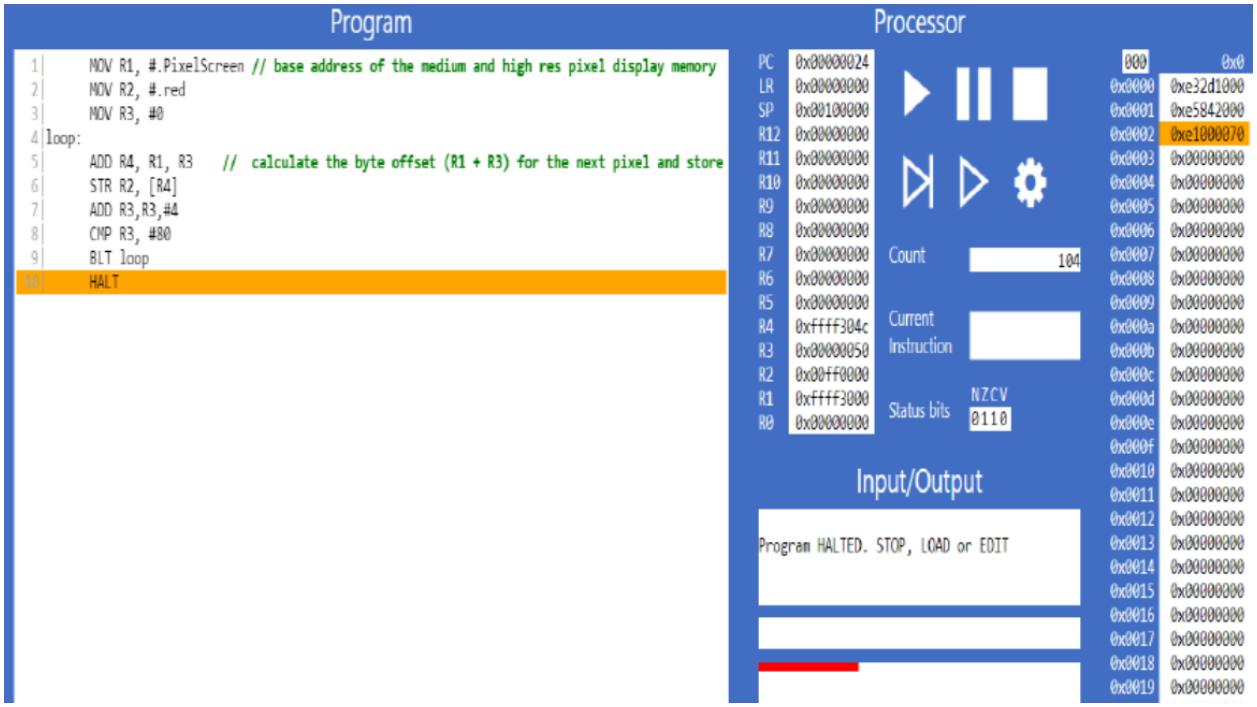
Q9.1.1 (a):



Q9.1.1 (b):


MOV R1, #.red
STR R1, .Pixel15
STR R1, .Pixel47
STR R1, .Pixel79
STR R1, .Pixel111
STR R1, .Pixel143
STR R1, .Pixel175
STR R1, .Pixel207
STR R1, .Pixel239
STR R1, .Pixel271
STR R1, .Pixel303
STR R1, .Pixel335
STR R1, .Pixel367
STR R1, .Pixel399
STR R1, .Pixel431
STR R1, .Pixel463
STR R1, .Pixel495
STR R1, .Pixel527
STR R1, .Pixel559
STR R1, .Pixel591
STR R1, .Pixel623
STR R1, .Pixel655
STR R1, .Pixel687
STR R1, .Pixel719
STR R1, .Pixel751
HALT
PC 0x00000068
LR 0x00000000
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000000
R6 0x00000000
R5 0x00000000
R4 0x00000000
R3 0x00000000
R2 0x00000000
R1 0x00ff0000
R0 0x00000000
Count 26
Current Instruction
Status bits NZCV 0000
Input/Output
Saving File
000 0x0 0x4
0x0000 0xe3a018ff 0xe50f1cd0
0x0001 0xe50f1b5c 0xe50f1ae0
0x0002 0xe50f196c 0xe50f18f0
0x0003 0xe50f177c 0xe50f1700
0x0004 0xe50f158c 0xe50f1510
0x0005 0xe50f139c 0xe50f1320
0x0006 0xe50f11ac 0xe1000070
0x0007 0x00000000 0x00000000
0x0008 0x00000000 0x00000000
0x0009 0x00000000 0x00000000
0x000a 0x00000000 0x00000000
0x000b 0x00000000 0x00000000
0x000c 0x00000000 0x00000000
0x000d 0x00000000 0x00000000
0x000e 0x00000000 0x00000000
0x000f 0x00000000 0x00000000
0x0010 0x00000000 0x00000000
0x0011 0x00000000 0x00000000
0x0012 0x00000000 0x00000000
0x0013 0x00000000 0x00000000
0x0014 0x00000000 0x00000000
0x0015 0x00000000 0x00000000
0x0016 0x00000000 0x00000000
0x0017 0x00000000 0x00000000
0x0018 0x00000000 0x00000000
0x0019 0x00000000 0x00000000
0x001a 0x00000000 0x00000000
0x001b 0x00000000 0x00000000
0x001c 0x00000000 0x00000000
0x001d 0x00000000 0x00000000
0x001e 0x00000000 0x00000000
0x001f 0x00000000 0x00000000
Hex
ARMLite Simulator V1.2 © Peter H

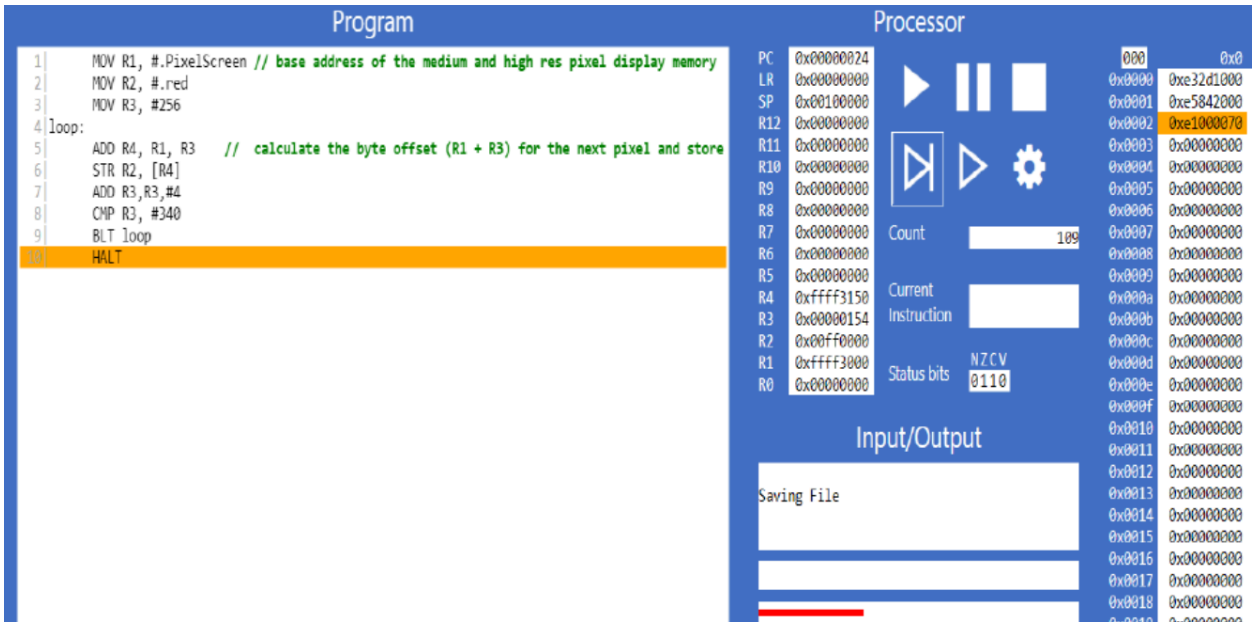
Q9.1.2:


Program
MOV R1, #.PixelScreen // base address of the medium and high res pixel display memory
MOV R2, #.red
MOV R3, #0
loop:
ADD R4, R1, R3 // calculate the byte offset (R1 + R3) for the next pixel and store
STR R2, [R4]
ADD R3,R3,#4
CMP R3, #80
BLT loop
HALT
Processor
PC 0x00000024
LR 0x00000000
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000000
R6 0x00000000
R5 0x00000000
R4 0xffff304c
R3 0x00000050
R2 0x00ff0000
R1 0xffff3000
R0 0x00000000
Count 104
Current Instruction
Status bits NZCV 0110
Input/Output
Program HALTED. STOP, LOAD or EDIT
000 0x0
0x0000 0xe32d1000
0x0001 0xe5842000
0x0002 0xe1000070
0x0003 0x00000000
0x0004 0x00000000
0x0005 0x00000000
0x0006 0x00000000
0x0007 0x00000000
0x0008 0x00000000
0x0009 0x00000000
0x000a 0x00000000
0x000b 0x00000000
0x000c 0x00000000
0x000d 0x00000000
0x000e 0x00000000
0x000f 0x00000000
0x0010 0x00000000
0x0011 0x00000000
0x0012 0x00000000
0x0013 0x00000000
0x0014 0x00000000
0x0015 0x00000000
0x0016 0x00000000
0x0017 0x00000000
0x0018 0x00000000
0x0019 0x00000000

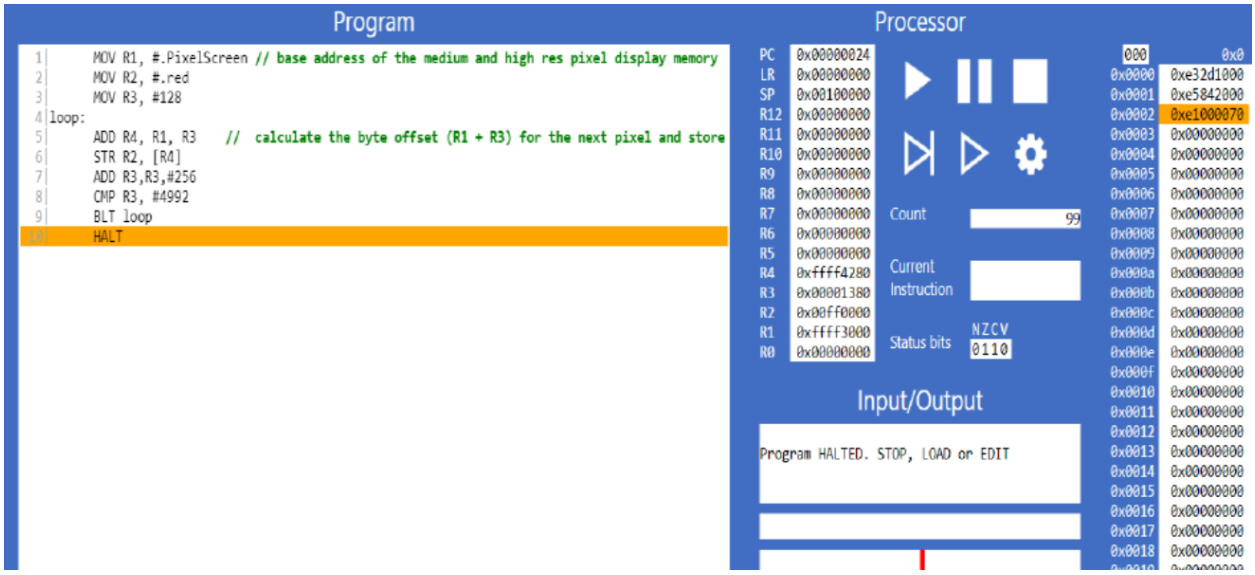
Q9.1.3 (a):

The above code uses indirect addressing since it may store the exact value of R2 in [R4], containing a memory location and its value. In the code, 4 bytes are added to the pixel each time until it reaches 80, and data is moved to the memory location (4 bytes, 32 bits). R1 is the base address; if we add a value in R3, it will be added to R1, which stores the address and may be used to create a new pointer to the subsequent pixel.The above code uses indirect addressing since it may store the exact value of R2 in [R4], containing a memory location and its value. In the code, 4 bytes are added to the pixel each time until it reaches 80, and data is moved to the memory location (4 bytes, 32 bits). R1 is the base address; if we add a value in R3, it will be added to R1, which stores the address and may be used to create a new pointer to the subsequent pixel.

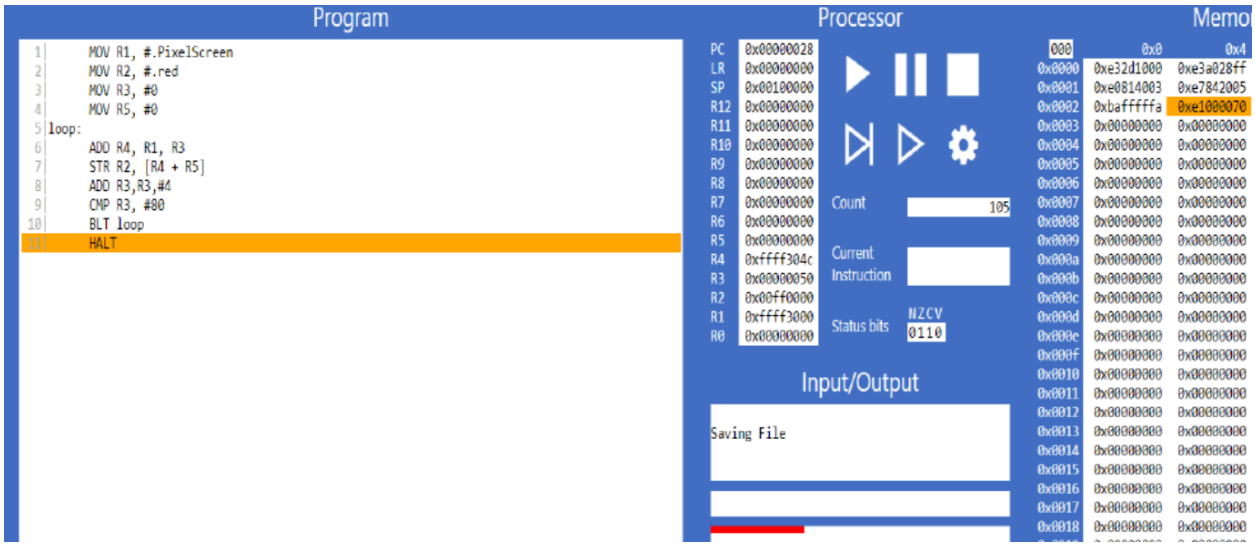
Q9.1.3 (b):



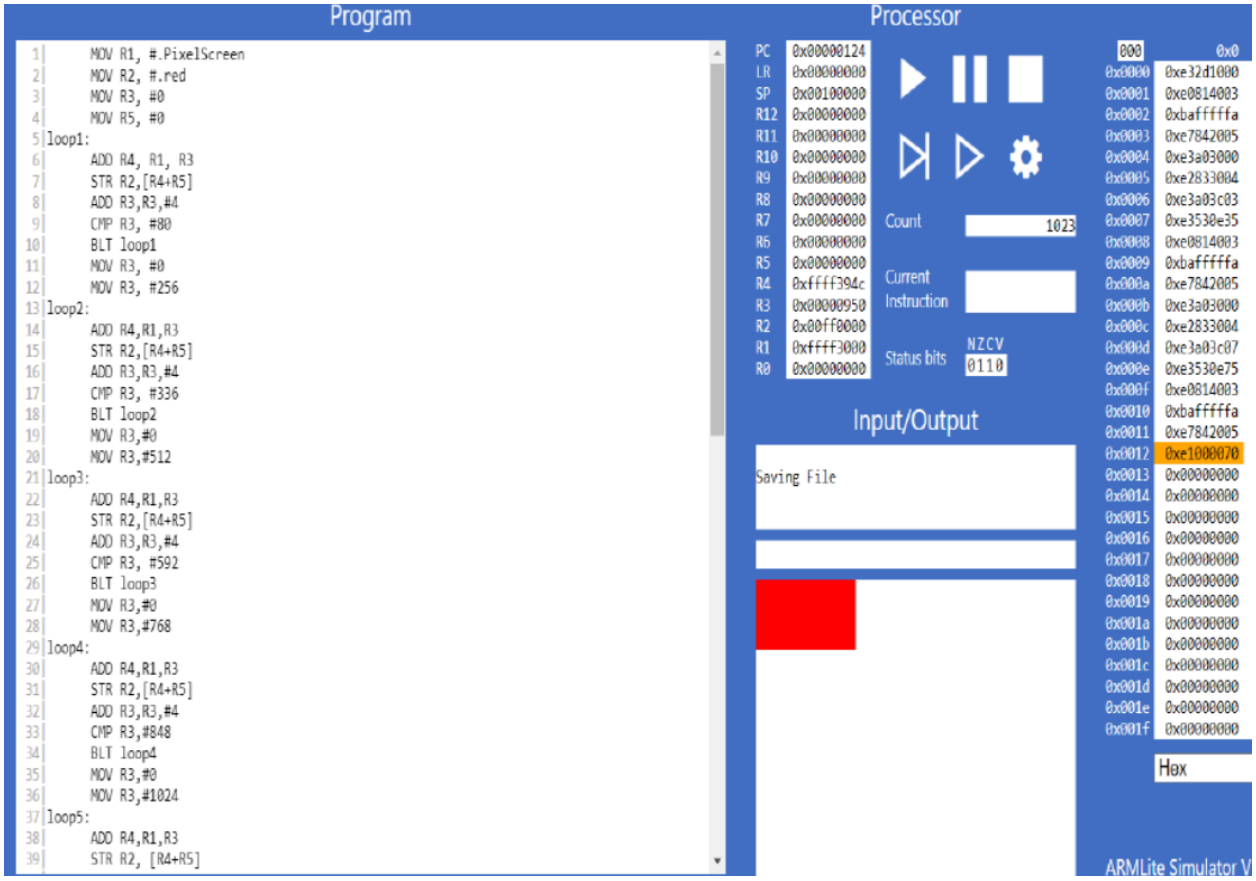
Q9.1.3 (c):


Program
1 MOV R1, #.PixelScreen // base address of the medium and high res pixel display memory
2 MOV R2, #.red
3 MOV R3, #128
4 loop:
5 ADD R4, R1, R3 // calculate the byte offset (R1 + R3) for the next pixel and store
6 STR R2, [R4]
7 ADD R3,R3,#256
8 CMP R3, #4992
9 BLT loop
10 HALT
Processor
PC 0x00000024
LR 0x00000000
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000000
R6 0x00000000
R5 0x00000000
R4 0xffff4280
R3 0x00001380
R2 0x00ff0000
R1 0xffff3000
R0 0x00000000
Count 99
Current Instruction
Status bits NZCV 0110
Input/Output
Program HALTED. STOP, LOAD or EDIT
000 0x0
0x0000 0xe32d1000
0x0001 0xe5842000
0x0002 0xe1000070
0x0003 0x00000000
0x0004 0x00000000
0x0005 0x00000000
0x0006 0x00000000
0x0007 0x00000000
0x0008 0x00000000
0x0009 0x00000000
0x000a 0x00000000
0x000b 0x00000000
0x000c 0x00000000
0x000d 0x00000000
0x000e 0x00000000
0x000f 0x00000000
0x0010 0x00000000
0x0011 0x00000000
0x0012 0x00000000
0x0013 0x00000000
0x0014 0x00000000
0x0015 0x00000000
0x0016 0x00000000
0x0017 0x00000000
0x0018 0x00000000

Q9.2.1:

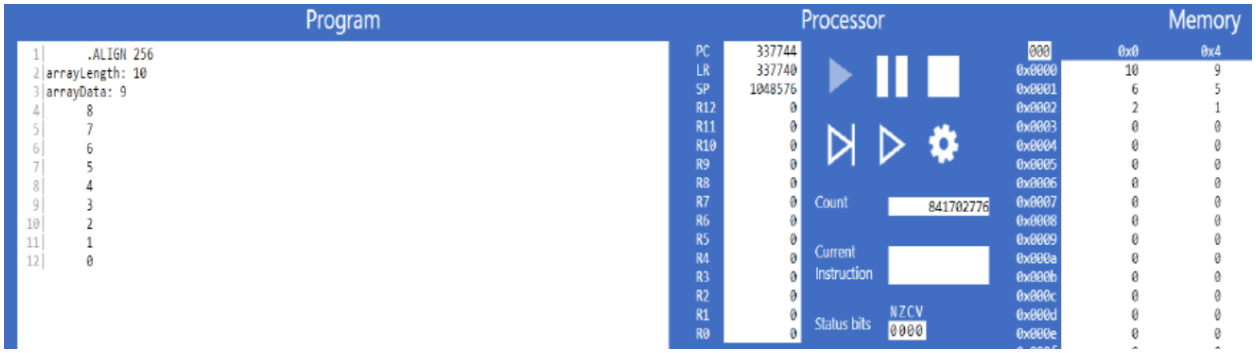

Program
1 MOV R1, #.PixelScreen
2 MOV R2, #.red
3 MOV R3, #0
4 MOV R5, #0
5 loop:
6 ADD R4, R1, R3
7 STR R2, [R4 + R5]
8 ADD R3,R3,#4
9 CMP R3, #80
10 BLT loop
11 HALT
Processor
PC 0x00000028
LR 0x00000000
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000000
R6 0x00000000
R5 0x00000000
R4 0xffff304c
R3 0x00000050
R2 0x00ff0000
R1 0xffff3000
R0 0x00000000
Count 105
Current Instruction
Status bits NZCV 0110
Input/Output
Saving File
Memo
000 0x0 0x4
0x0000 0xe32d1000 0xe3a028ff
0x0001 0xe0814003 0xe7842005
0x0002 0xbafffffa 0xe1000070
0x0003 0x00000000 0x00000000
0x0004 0x00000000 0x00000000
0x0005 0x00000000 0x00000000
0x0006 0x00000000 0x00000000
0x0007 0x00000000 0x00000000
0x0008 0x00000000 0x00000000
0x0009 0x00000000 0x00000000
0x000a 0x00000000 0x00000000
0x000b 0x00000000 0x00000000
0x000c 0x00000000 0x00000000
0x000d 0x00000000 0x00000000
0x000e 0x00000000 0x00000000
0x000f 0x00000000 0x00000000
0x0010 0x00000000 0x00000000
0x0011 0x00000000 0x00000000
0x0012 0x00000000 0x00000000
0x0013 0x00000000 0x00000000
0x0014 0x00000000 0x00000000
0x0015 0x00000000 0x00000000
0x0016 0x00000000 0x00000000
0x0017 0x00000000 0x00000000
0x0018 0x00000000 0x00000000

Q9.2.2:



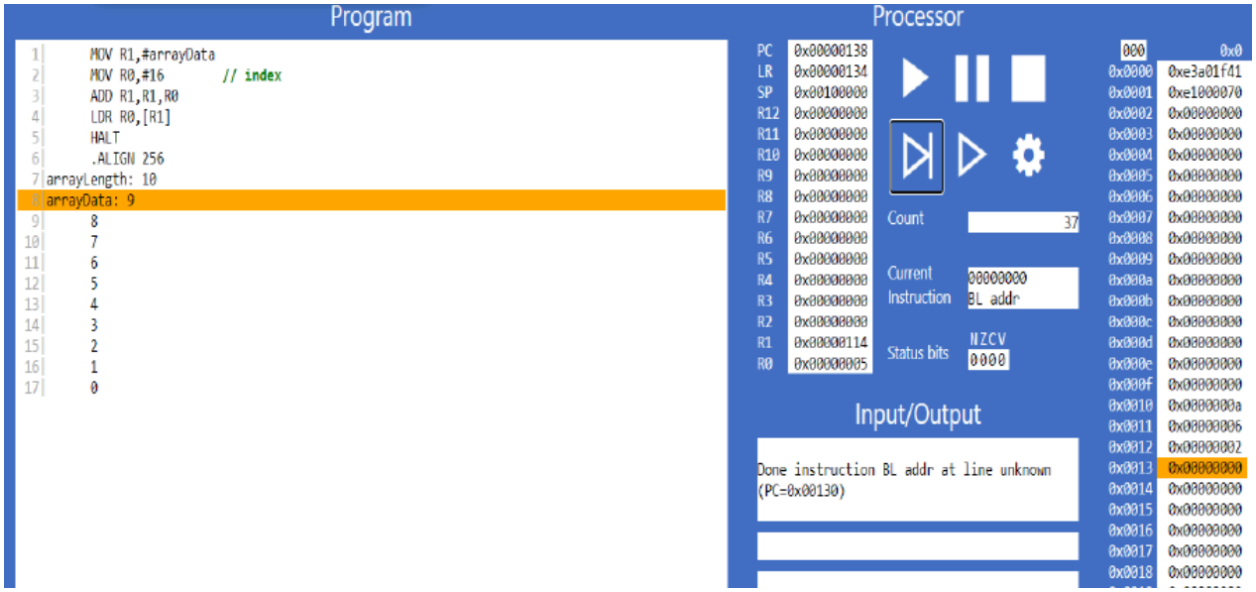

```
1       MOV R1, #.PixelScreen
2       MOV R2, #.red
3       MOV R3, #0
4       MOV R5, #0
5   loop1:
6       ADD R4, R1, R3
7       STR R2,[R4+R5]
8       ADD R3,R3,#4
9       CMP R3, #80
10      BLT loop1
11      MOV R3, #0
12      MOV R3, #256
13  loop2:
14      ADD R4,R1,R3
15      STR R2,[R4+R5]
16      ADD R3,R3,#4
17      CMP R3, #336
18      BLT loop2
19      MOV R3,#0
20      MOV R3,#512
21  loop3:
22      ADD R4,R1,R3
23      STR R2,[R4+R5]
24      ADD R3,R3,#4
25      CMP R3, #592
26      BLT loop3
27      MOV R3,#0
28      MOV R3,#768
29  loop4:
30      ADD R4,R1,R3
31      STR R2,[R4+R5]
32      ADD R3,R3,#4
33      CMP R3,#848
34      BLT loop4
35      MOV R3,#0
36      MOV R3,#1024
37  loop5:
38      ADD R4,R1,R3
39      STR R2, [R4+R5]
```

Q9.3.1 (a):

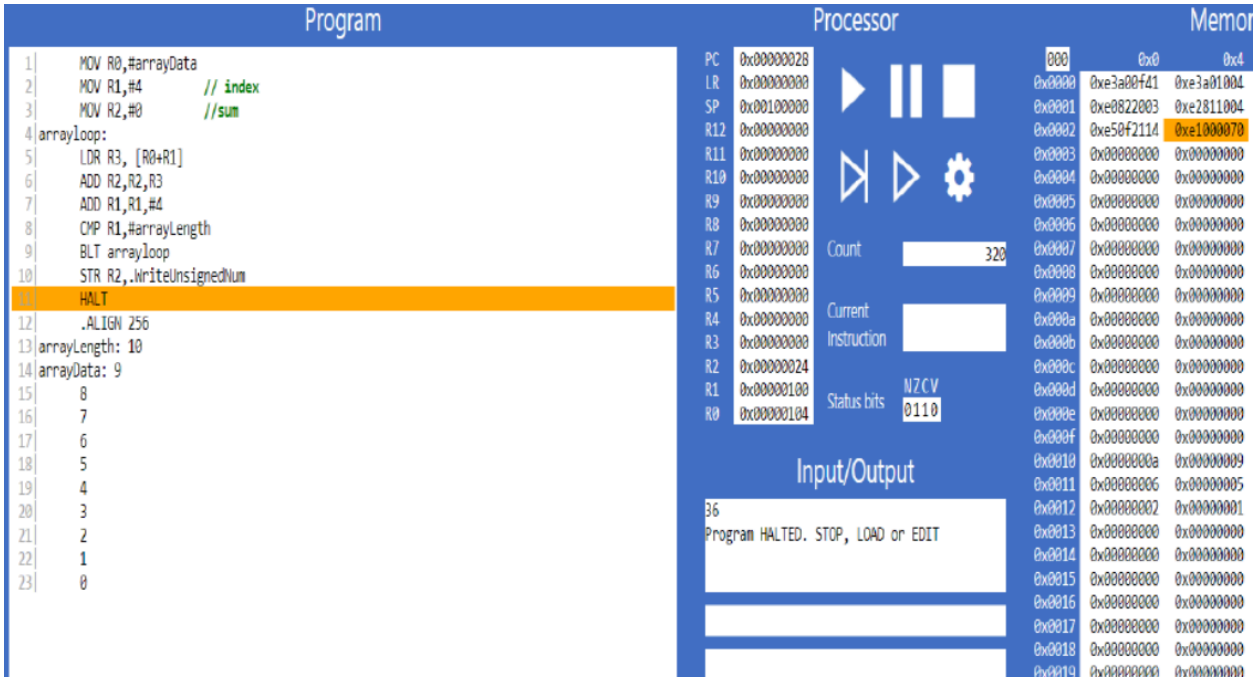



```
1       .ALIGN 256
2   arrayLength: 10
3   arrayData: 9
4       8
5       7
6       6
7       5
8       4
9       3
10      2
11      1
12      0
```

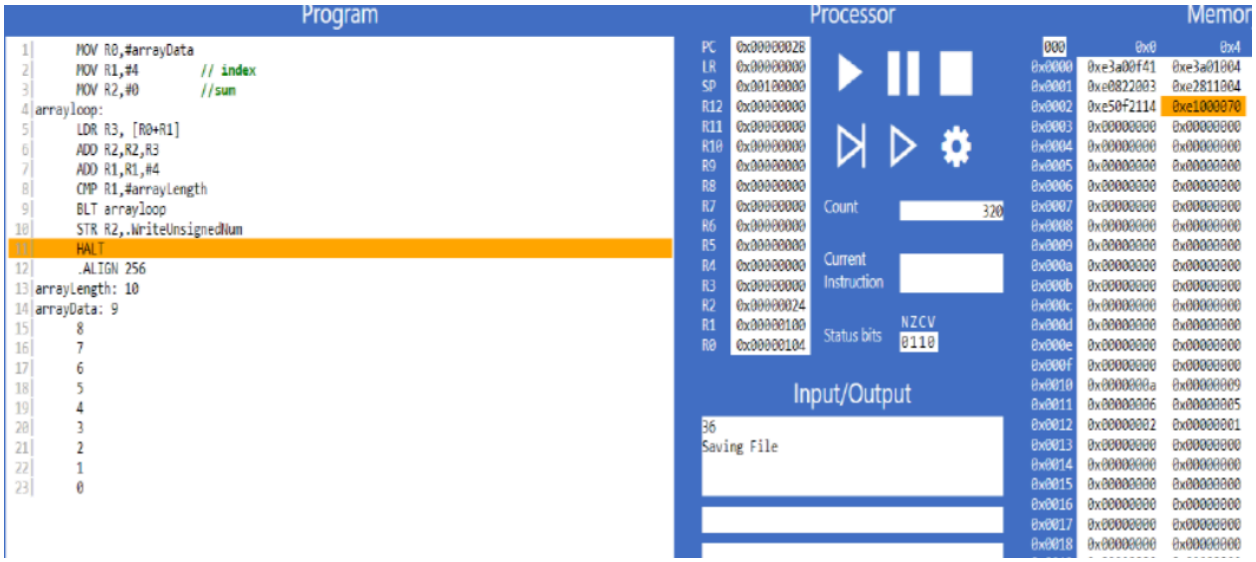
Q9.3.1 (b):


Program
1 MOV R1,#arrayData
2 MOV R0,#16 // index
3 ADD R1,R1,R0
4 LDR R0,[R1]
5 HALT
6 .ALIGN 256
7 arrayLength: 10
8 arrayData: 9
9 8
10 7
11 6
12 5
13 4
14 3
15 2
16 1
17 0
Processor
PC 0x00000138
LR 0x00000134
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000000
R6 0x00000000
R5 0x00000000
R4 0x00000000
R3 0x00000000
R2 0x00000000
R1 0x00000114
R0 0x00000005
Count 37
Current Instruction 00000000 BL addr
Status bits NZCV 0000
Input/Output
Done instruction BL addr at line unknown (PC=0x00130)
000 0x0
0x0000 0xe3a01f41
0x0001 0xe1000070
0x0002 0x00000000
0x0003 0x00000000
0x0004 0x00000000
0x0005 0x00000000
0x0006 0x00000000
0x0007 0x00000000
0x0008 0x00000000
0x0009 0x00000000
0x000a 0x00000000
0x000b 0x00000000
0x000c 0x00000000
0x000d 0x00000000
0x000e 0x00000000
0x000f 0x00000000
0x0010 0x0000000a
0x0011 0x00000006
0x0012 0x00000002
0x0013 0x00000000
0x0014 0x00000000
0x0015 0x00000000
0x0016 0x00000000
0x0017 0x00000000
0x0018 0x00000000

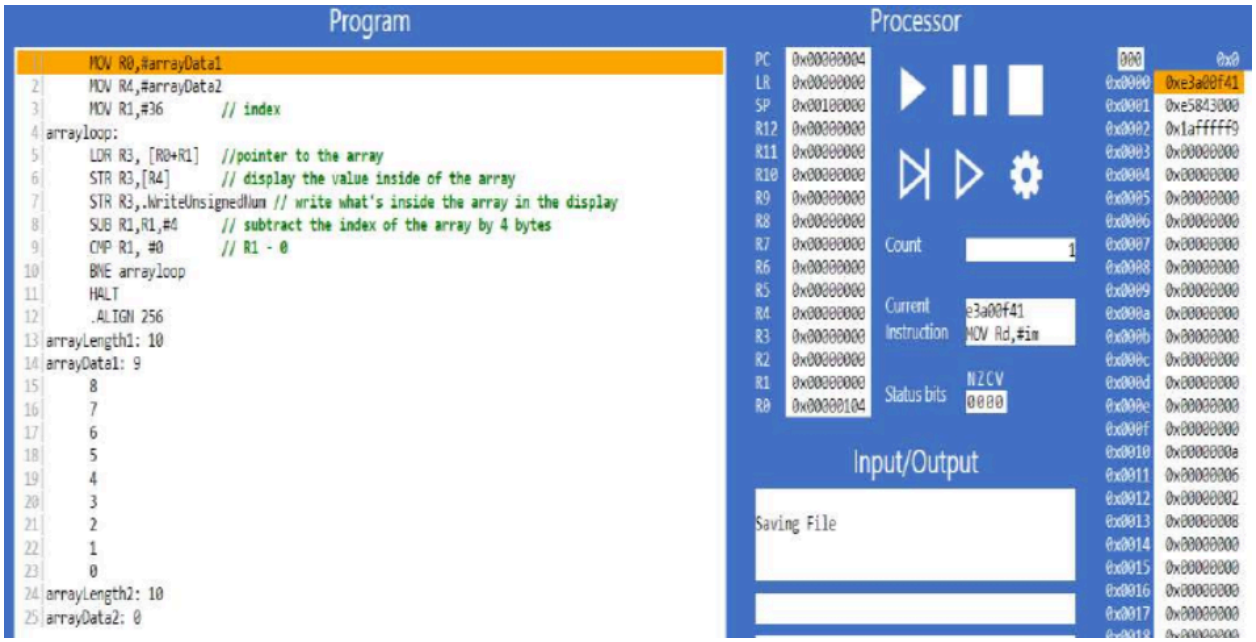
Q9.3.1 (c):


Program
1 MOV R0,#arrayData
2 MOV R1,#4 // index
3 MOV R2,#0 //sum
4 arrayloop:
5 LDR R3, [R0+R1]
6 ADD R2,R2,R3
7 ADD R1,R1,#4
8 CMP R1,#arrayLength
9 BLT arrayloop
10 STR R2,.WriteUnsignedNum
11 HALT
12 .ALIGN 256
13 arrayLength: 10
14 arrayData: 9
15 8
16 7
17 6
18 5
19 4
20 3
21 2
22 1
23 0
Processor
PC 0x00000028
LR 0x00000000
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000000
R6 0x00000000
R5 0x00000000
R4 0x00000000
R3 0x00000000
R2 0x00000024
R1 0x00000100
R0 0x00000104
Count 320
Current Instruction
Status bits NZCV 0110
Input/Output
36
Program HALTED. STOP, LOAD or EDIT
Memor
000 0x0 0x4
0x0000 0xe3a00f41 0xe3a01004
0x0001 0xe0822003 0xe2811004
0x0002 0xe50f2114 0xe1000070
0x0003 0x00000000 0x00000000
0x0004 0x00000000 0x00000000
0x0005 0x00000000 0x00000000
0x0006 0x00000000 0x00000000
0x0007 0x00000000 0x00000000
0x0008 0x00000000 0x00000000
0x0009 0x00000000 0x00000000
0x000a 0x00000000 0x00000000
0x000b 0x00000000 0x00000000
0x000c 0x00000000 0x00000000
0x000d 0x00000000 0x00000000
0x000e 0x00000000 0x00000000
0x000f 0x00000000 0x00000000
0x0010 0x0000000a 0x00000009
0x0011 0x00000006 0x00000005
0x0012 0x00000002 0x00000001
0x0013 0x00000000 0x00000000
0x0014 0x00000000 0x00000000
0x0015 0x00000000 0x00000000
0x0016 0x00000000 0x00000000
0x0017 0x00000000 0x00000000
0x0018 0x00000000 0x00000000

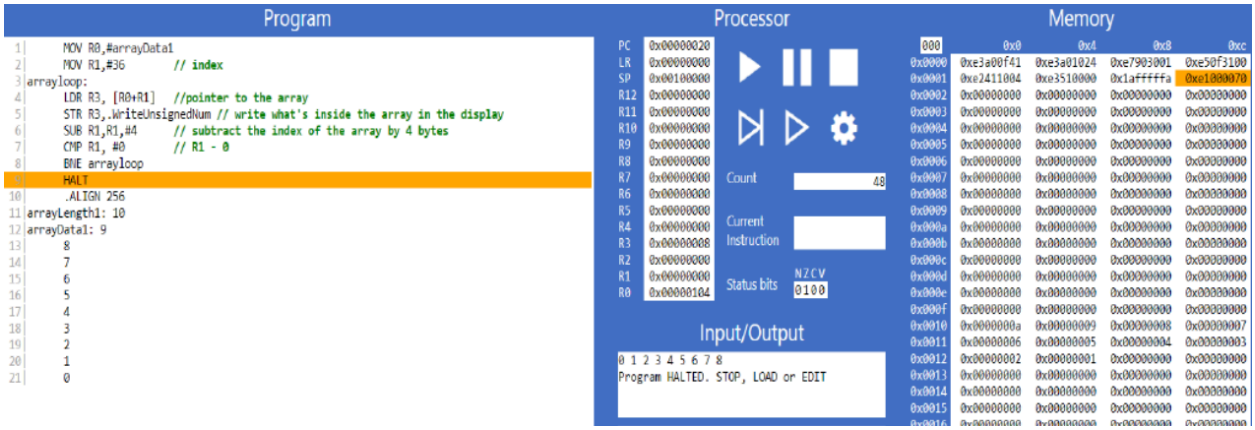
Q9.3.3:


Program
Processor
Memor
MOV R0,#arrayData
MOV R1,#4 // index
MOV R2,#0 //sum
arrayloop:
LDR R3, [R0+R1]
ADD R2,R2,R3
ADD R1,R1,#4
CMP R1,#arrayLength
BLT arrayloop
STR R2,.WriteUnsignedNum
HALT
.ALIGN 256
arrayLength: 10
arrayData: 9
8
7
6
5
4
3
2
1
0
PC 0x00000028
LR 0x00000000
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000000
R6 0x00000000
R5 0x00000000
R4 0x00000000
R3 0x00000000
R2 0x00000024
R1 0x00000100
R0 0x00000104
Count 320
Current Instruction
Status bits NZCV 0110
Input/Output
36
Saving File
000 0x0 0x4
0x0000 0xe3a00f41 0xe3a01004
0x0001 0xe0822003 0xe2811004
0x0002 0xe50f2114 0xe1000070
0x0003 0x00000000 0x00000000
0x0004 0x00000000 0x00000000
0x0005 0x00000000 0x00000000
0x0006 0x00000000 0x00000000
0x0007 0x00000000 0x00000000
0x0008 0x00000000 0x00000000
0x0009 0x00000000 0x00000000
0x000a 0x00000000 0x00000000
0x000b 0x00000000 0x00000000
0x000c 0x00000000 0x00000000
0x000d 0x00000000 0x00000000
0x000e 0x00000000 0x00000000
0x000f 0x00000000 0x00000000
0x0010 0x0000000a 0x00000009
0x0011 0x00000006 0x00000005
0x0012 0x00000002 0x00000001
0x0013 0x00000000 0x00000000
0x0014 0x00000000 0x00000000
0x0015 0x00000000 0x00000000
0x0016 0x00000000 0x00000000
0x0017 0x00000000 0x00000000
0x0018 0x00000000 0x00000000

Q9.4.1:


Program
Processor
MOV R0,#arrayData1
MOV R4,#arrayData2
MOV R1,#36 // index
arrayloop:
LDR R3, [R0+R1] //pointer to the array
STR R3,[R4] // display the value inside of the array
STR R3,.WriteUnsignedNum // write what's inside the array in the display
SUB R1,R1,#4 // subtract the index of the array by 4 bytes
CMP R1, #0 // R1 - 0
BNE arrayloop
HALT
.ALIGN 256
arrayLength1: 10
arrayData1: 9
8
7
6
5
4
3
2
1
0
arrayLength2: 10
arrayData2: 0
PC 0x00000004
LR 0x00000000
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000000
R6 0x00000000
R5 0x00000000
R4 0x00000000
R3 0x00000000
R2 0x00000000
R1 0x00000000
R0 0x00000104
Count 1
Current Instruction e3a00f41 MOV Rd,#im
Status bits NZCV 0000
Input/Output
Saving File
000 0x0
0x0000 0xe3a00f41
0x0001 0xe5843000
0x0002 0x1affffff9
0x0003 0x00000000
0x0004 0x00000000
0x0005 0x00000000
0x0006 0x00000000
0x0007 0x00000000
0x0008 0x00000000
0x0009 0x00000000
0x000a 0x00000000
0x000b 0x00000000
0x000c 0x00000000
0x000d 0x00000000
0x000e 0x00000000
0x000f 0x00000000
0x0010 0x0000000a
0x0011 0x00000006
0x0012 0x00000002
0x0013 0x00000008
0x0014 0x00000000
0x0015 0x00000000
0x0016 0x00000000
0x0017 0x00000000

Q9.4.2:


Program
1 MOV R0,#arrayData1
2 MOV R1,#36 // index
3 arrayloop:
4 LDR R3, [R0+R1] //pointer to the array
5 STR R3,.WriteUnsignedNum // write what's inside the array in the display
6 SUB R1,R1,#4 // subtract the index of the array by 4 bytes
7 CMP R1, #0 // R1 - 0
8 BNE arrayloop
9 HALT
10 .ALIGN 256
11 arrayLength1: 10
12 arrayData1: 9
13 8
14 7
15 6
16 5
17 4
18 3
19 2
20 1
21 0
Processor
PC 0x00000020
LR 0x00000000
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000000
R6 0x00000000
R5 0x00000000
R4 0x00000000
R3 0x00000008
R2 0x00000000
R1 0x00000000
R0 0x00000104
Count 48
Current Instruction
Status bits NZCV 0100
Input/Output
0 1 2 3 4 5 6 7 8
Program HALTED. STOP, LOAD or EDIT
Memory
000 0x0 0x4 0x8 0xc
0x0000 0xe3a00f41 0xe3a01024 0xe7903001 0xe50f3100
0x0001 0xe2411004 0xe3510000 0x1affffa 0xe1000070
0x0002 0x00000000 0x00000000 0x00000000 0x00000000
0x0003 0x00000000 0x00000000 0x00000000 0x00000000
0x0004 0x00000000 0x00000000 0x00000000 0x00000000
0x0005 0x00000000 0x00000000 0x00000000 0x00000000
0x0006 0x00000000 0x00000000 0x00000000 0x00000000
0x0007 0x00000000 0x00000000 0x00000000 0x00000000
0x0008 0x00000000 0x00000000 0x00000000 0x00000000
0x0009 0x00000000 0x00000000 0x00000000 0x00000000
0x000a 0x00000000 0x00000000 0x00000000 0x00000000
0x000b 0x00000000 0x00000000 0x00000000 0x00000000
0x000c 0x00000000 0x00000000 0x00000000 0x00000000
0x000d 0x00000000 0x00000000 0x00000000 0x00000000
0x000e 0x00000000 0x00000000 0x00000000 0x00000000
0x000f 0x00000000 0x00000000 0x00000000 0x00000000
0x0010 0x0000000a 0x00000009 0x00000008 0x00000007
0x0011 0x00000006 0x00000005 0x00000004 0x00000003
0x0012 0x00000002 0x00000001 0x00000000 0x00000000
0x0013 0x00000000 0x00000000 0x00000000 0x00000000
0x0014 0x00000000 0x00000000 0x00000000 0x00000000
0x0015 0x00000000 0x00000000 0x00000000 0x00000000