



# Computer Systems

Week 9

## Overview

Name: Le Quang Hai

Student ID: 104175779

### Exercise 9.1.1

(a) Write a simple ARM lite assembly program that draws a single line of the same length across the second row (starting from the left-most column) in Low-res display mode.

The screenshot shows an ARM assembly simulator with three main panels: Program, Processor, and Memory.

**Program Panel:** Displays assembly code for drawing a red line. The code uses the `MOV` instruction to set the color to red and the `STR` instruction to draw pixels across the second row (address 0x00000000 to 0x00000050). The final instruction is `HALT`.

```

1| MOV R1, #.red
2| STR R1, .Pixel32
3| STR R1, .Pixel33
4| STR R1, .Pixel34
5| STR R1, .Pixel35
6| STR R1, .Pixel36
7| STR R1, .Pixel37
8| STR R1, .Pixel38
9| STR R1, .Pixel39
10| STR R1, .Pixel40
11| STR R1, .Pixel41
12| STR R1, .Pixel42
13| STR R1, .Pixel43
14| STR R1, .Pixel44
15| STR R1, .Pixel45
16| STR R1, .Pixel46
17| STR R1, .Pixel47
18| STR R1, .Pixel48
19| STR R1, .Pixel49
20| STR R1, .Pixel50
21| STR R1, .Pixel51
22| HALT

```

**Processor Panel:** Shows the current state of the processor. The PC (Program Counter) is 0x00000058. The Count register is 22. The Status bits are NZCV: 0000. The Input/Output panel shows "Program HALTED. STOP, LOAD or EDIT".

**Memory Panel:** Displays a memory dump. The address 0x00000000 is highlighted in orange, corresponding to the first pixel of the red line drawn by the program.

Address	0x0	0x4	0x8	0xc
0x00000000	0xe3a018ff	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c
0x00000001	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c
0x00000002	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c
0x00000003	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c
0x00000004	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c
0x00000005	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c	0xe50f1c8c
0x00000006	0x00000000	0x00000000	0x00000000	0x00000000
0x00000007	0x00000000	0x00000000	0x00000000	0x00000000
0x00000008	0x00000000	0x00000000	0x00000000	0x00000000
0x00000009	0x00000000	0x00000000	0x00000000	0x00000000
0x0000000a	0x00000000	0x00000000	0x00000000	0x00000000
0x0000000b	0x00000000	0x00000000	0x00000000	0x00000000
0x0000000c	0x00000000	0x00000000	0x00000000	0x00000000
0x0000000d	0x00000000	0x00000000	0x00000000	0x00000000
0x0000000e	0x00000000	0x00000000	0x00000000	0x00000000
0x0000000f	0x00000000	0x00000000	0x00000000	0x00000000
0x00000010	0x00000000	0x00000000	0x00000000	0x00000000
0x00000011	0x00000000	0x00000000	0x00000000	0x00000000
0x00000012	0x00000000	0x00000000	0x00000000	0x00000000
0x00000013	0x00000000	0x00000000	0x00000000	0x00000000
0x00000014	0x00000000	0x00000000	0x00000000	0x00000000
0x00000015	0x00000000	0x00000000	0x00000000	0x00000000
0x00000016	0x00000000	0x00000000	0x00000000	0x00000000
0x00000017	0x00000000	0x00000000	0x00000000	0x00000000
0x00000018	0x00000000	0x00000000	0x00000000	0x00000000
0x00000019	0x00000000	0x00000000	0x00000000	0x00000000
0x0000001a	0x00000000	0x00000000	0x00000000	0x00000000
0x0000001b	0x00000000	0x00000000	0x00000000	0x00000000
0x0000001c	0x00000000	0x00000000	0x00000000	0x00000000
0x0000001d	0x00000000	0x00000000	0x00000000	0x00000000
0x0000001e	0x00000000	0x00000000	0x00000000	0x00000000
0x0000001f	0x00000000	0x00000000	0x00000000	0x00000000

**(b)** Add to your assembly program code that draws a single line of the same length vertically, down the middle of the display in Low-res display mode

**Program**

```

1  MOV R1, #.red
2  STR R1, .Pixel16
3  STR R1, .Pixel48
4  STR R1, .Pixel80
5  STR R1, .Pixel112
6  STR R1, .Pixel144
7  STR R1, .Pixel176
8  STR R1, .Pixel208
9  STR R1, .Pixel240
10 STR R1, .Pixel272
11 STR R1, .Pixel304
12 STR R1, .Pixel336
13 STR R1, .Pixel368
14 STR R1, .Pixel400
15 STR R1, .Pixel432
16 STR R1, .Pixel464
17 STR R1, .Pixel496
18 STR R1, .Pixel528
19 STR R1, .Pixel560
20 STR R1, .Pixel592
21 STR R1, .Pixel624
22 HALT

```

**Processor**

PC: 0x00000058  
 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: 22  
 Current Instruction:   
 Status bits: NZCV 0000

**Input/Output**

Program HALTED. STOP, LOAD or EDIT

**Memory**

000	0x0	0x4	0x8	0xc
0x0000	0xe3a018ff	0xe50f1ccc	0xe50f1c50	0xe50f1bd4
0x0001	0xe50f1b58	0xe50f1adc	0xe50f1a60	0xe50f19e4
0x0002	0xe50f1968	0xe50f18ec	0xe50f1870	0xe50f17f4
0x0003	0xe50f1778	0xe50f16fc	0xe50f1680	0xe50f1604
0x0004	0xe50f1588	0xe50f150c	0xe50f1490	0xe50f1414
0x0005	0xe50f1398	0xe50f131c	0xe50f1220	0xe50f1124
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	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000000	0x00000000	0x00000000	0x00000000
0x0012	0x00000000	0x00000000	0x00000000	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

## Exercise 9.1.2

**Program**

```

1  MOV R1, #.PixelScreen // base address of the medium and high res pixel
2  MOV R2, #.red
3  MOV R3, #0
4  loop:
5  ADD R4, R1, R3 // calculate the byte offset (R1 + R3) for the next
6  STR R2, [R4] // color red
7  ADD R3, R3, #4
8  CMP R3, #80
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: 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

**Memory**

000	0x0	0x4	0x8	0xc
0x0000	0xe32d1000	0xe3a028ff	0xe3a03000	0xe0814003
0x0001	0xe5842000	0xe2833004	0xe3530050	0xbaffffff
0x0002	0xe1000070	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	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000000	0x00000000	0x00000000	0x00000000
0x0012	0x00000000	0x00000000	0x00000000	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

## Exercise 9.1.3

**(a)** Explain what specifically makes this code an example of indirect addressing ? How is it using indirect addressing to draw each pixel ?

**Ans:** Because we access and color red the cells via R4 instead of directly implementing it as in exercise 9.1.1, the R4 value increments every loop.

**(b)** Once you're confident you understand the code, modify the program so that it draws a line of the same length along the second row of the Mid-res display.

### Program

```

1|  MOV R1, #.PixelScreen // base address of the medium and high res pixel
2|  ADD R1, R1, #256
3|  MOV R2, #.red
4|  MOV R3, #0
5|  loop:
6|  ADD R4, R1, R3 // calculate the byte offset (R1 + R3) for the next
7|  STR R2, [R4] // color red
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: 0xffff314c  
R3: 0x00000050  
R2: 0x00ff0000  
R1: 0xffff3100  
R0: 0x00000000

Count: 105

Current Instruction:

Status bits: NZCV 0110

### Input/Output

Program HALTED. STOP, LOAD or EDIT

### Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe32d1000	0xe2811c01	0xe3a028ff	0xe3a03000
0x0001	0xe0814003	0xe5842000	0xe2833004	0xe3530a03
0x0002	0xbaffffff	0xe1000070	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	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000000	0x00000000	0x00000000	0x00000000
0x0012	0x00000000	0x00000000	0x00000000	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

(c) Further modify your program so that it also draws a line of the same length vertically down the middle of the display.

### Program

```

1|  MOV R1, #.PixelScreen // base address of the medium and high res pixel
2|  ADD R1, R1, #128
3|  MOV R2, #.red
4|  MOV R3, #0
5|  loop:
6|  ADD R4, R1, R3 // calculate the byte offset (R1 + R3) for the next
7|  STR R2, [R4] // color red
8|  ADD R3, R3, #256
9|  CMP R3, #12288
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: 0xffff5f80  
R3: 0x00003000  
R2: 0x00ff0000  
R1: 0xffff3080  
R0: 0x00000000

Count: 245

Current Instruction:

Status bits: NZCV 0110

### Input/Output

Program HALTED. STOP, LOAD or EDIT

### Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe32d1000	0xe2811080	0xe3a028ff	0xe3a03000
0x0001	0xe0814003	0xe5842000	0xe2833c01	0xe3530a03
0x0002	0xbaffffff	0xe1000070	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	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000000	0x00000000	0x00000000	0x00000000
0x0012	0x00000000	0x00000000	0x00000000	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

## Exercise 9.2.1

### Program

```

1|  MOV R1, #.PixelScreen
2|  MOV R2, #.red
3|  MOV R3, #0
4|  loop:
5|  STR R2, [R1 + R3]
6|  ADD R3, R3, #4
7|  CMP R3, #80
8|  BLT loop
9|  HALT

```

### 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: 0x00000050  
R2: 0x00ff0000  
R1: 0xffff3000  
R0: 0x00000000

Count: 84

Current Instruction:

Status bits: NZCV 0110

### Input/Output

Program HALTED. STOP, LOAD or EDIT

### Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe32d1000	0xe3a028ff	0xe3a03000	0xe7812003
0x0001	0xe2833004	0xe3530050	0xbaffffff	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	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000000	0x00000000	0x00000000	0x00000000
0x0012	0x00000000	0x00000000	0x00000000	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

## Exercise 9.2.2

### Program

```

1|  MOV R1, #.PixelScreen
2|  MOV R2, #.red
3|  MOV R3, #0
4|  MOV R4, #80
5| loop:
6|  STR R2, [R1 + R3]
7|  ADD R3, R3, #4
8|  CMP R3, R4
9|  BLT loop
10| // newline
11|  ADD R3, R3, #176
12|  ADD R4, R4, #256
13|  CMP R4, #12288
14|  BLT loop
15|  HALT

```

### Processor

PC: 0x00000034  
LR: 0x00000000  
SP: 0x00100000  
R12: 0x00000000  
R11: 0x00000000  
R10: 0x00000000  
R9: 0x00000000  
R8: 0x00000000  
R7: 0x00000000  
R6: 0x00000000  
R5: 0x00000000  
R4: 0x00003050  
R3: 0x00003000  
R2: 0x00ff0000  
R1: 0xfffff3000  
R0: 0x00000000

Count: 4037  
Current Instruction:   
Status bits: NZCV 0010

### Input/Output

Program HALTED. STOP, LOAD or EDIT

### Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe32d1000	0xe3a028ff	0xe3a03000	0xe3a04050
0x0001	0xe7812003	0xe2833004	0xe1530004	0xbaffffff
0x0002	0xe28330b0	0xe2844c01	0xe3540a03	0xbaffffff
0x0003	0xe1000070	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	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000000	0x00000000	0x00000000	0x00000000
0x0012	0x00000000	0x00000000	0x00000000	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

## Exercise 9.3.1 : 256 indicate the address of the (first element of the) array

### Program

```

1|  MOV R1, #5
2|  MOV R2, #256 // arr address
3|  MOV R4, #20
4|  ADD R3, R2, R4
5|  LDR R0, [R3]
6|  HALT
7|  .ALIGN 256
8| arrayLength: 10
9| arrayData: 9
10| 8
11| 7
12| 6
13| 5
14| 4
15| 3
16| 2
17| 1
18| 0

```

### Processor

PC: 0x00000018  
LR: 0x00000000  
SP: 0x00100000  
R12: 0x00000000  
R11: 0x00000000  
R10: 0x00000000  
R9: 0x00000000  
R8: 0x00000000  
R7: 0x00000000  
R6: 0x00000000  
R5: 0x00000000  
R4: 0x00000014  
R3: 0x00000114  
R2: 0x00000100  
R1: 0x00000005  
R0: 0x00000005

Count: 6  
Current Instruction:   
Status bits: NZCV 0000

### Input/Output

Program HALTED. STOP, LOAD or EDIT

### Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe3a01005	0xe3a02c01	0xe3a04014	0xe0623004
0x0001	0xe5930000	0xe1000070	0x00000000	0x00000000
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
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

## Exercise 9.3.2 & 9.3.3

### Program

```

1| // R0 = sum
2| // R1 = index
3| // R2 = address operand
4| // first element address
5|  MOV R3, #256
6|  ADD R3, R3, #4
7| loop:
8|  LDR R4, [R3 + R2]
9|  ADD R0, R0, R4
10|  ADD R1, R1, #1
11|  ADD R2, R2, #4
12|  CMP R2, #arrayLength
13|  BLT loop
14|  HALT
15|  .ALIGN 256
16| arrayLength: 10
17| arrayData: 9
18| 8
19| 7
20| 6
21| 5
22| 4
23| 3
24| 2
25| 1
26| 0

```

### Processor

PC: 0x00000024  
LR: 0x00000000  
SP: 0x00100000  
R12: 0x00000000  
R11: 0x00000000  
R10: 0x00000000  
R9: 0x00000000  
R8: 0x00000000  
R7: 0x00000000  
R6: 0x00000000  
R5: 0x00000000  
R4: 0x00000000  
R3: 0x00000104  
R2: 0x00000040  
R1: 0x00000040  
R0: 0x0000002d

Count: 387  
Current Instruction:   
Status bits: NZCV 0110

### Input/Output

Program HALTED. STOP, LOAD or EDIT

### Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe3a03c01	0xe2833004	0xe7934002	0xe0800004
0x0001	0xe2811001	0xe2822004	0xe3520c01	0xbaffffff
0x0002	0xe1000070	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
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000



## Exercise 9.4.1

Program

```
1| MOV R0, #arrayData
2| MOV R1, #newArr
3| MOV R4, #40
4| loop:
5| // read
6| LDR R5, [R0 + R3]
7| ADD R2, R2, #1
8| ADD R3, R3, #4
9| // write
10| SUB R12, R4, R3
11| STR R5, [R1 + R12]
12| // continue
13| CMP R2, #arrayLength
14| BLT loop
15| HALT
16| .ALIGN 256
17| arrayLength: 10
18| arrayData: 9
19| 8
20| 7
21| 6
22| 5
23| 4
24| 3
25| 2
26| 1
27| 0
28| newArr: 0
29| 0
30| 0
31| 0
32| 0
33| 0
```

Processor

PC: 0x00000020  
LR: 0x00000000  
SP: 0x00100000  
R12: 0xfffffdd0  
R10: 0x00000000  
R9: 0x00000000  
R8: 0x00000000  
R7: 0x00000000  
R6: 0x00000000  
R5: 0x00000000  
R4: 0x00000028  
R3: 0x00000058  
R2: 0x00000016  
R1: 0x0000012c  
R0: 0x00000104

Count: 155  
Current Instruction:   
Status bits: NZCV 1000

Input/Output  
Attempt to STR into the code area at line 11

Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe3a00f41	0xe3a01f4b	0xe3a04028	0xe7905003
0x0001	0xe2822001	0xe2833004	0xe044c003	0xe781500c
0x0002	0xe3520c01	0xbafffff8	0xe1000070	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	0x00000000	0x00000000	0x00000000	0x00000000
0x0011	0x00000006	0x00000005	0x00000004	0x00000003
0x0012	0x00000002	0x00000001	0x00000000	0x00000000
0x0013	0x00000001	0x00000000	0x00000000	0x00000000
0x0014	0x00000005	0x00000006	0x00000007	0x00000008
0x0015	0x00000009	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000

## Exercise 9.4.2

Program

```
1| MOV R0, #arrayData
2| MOV R12, #36
3| loop:
4| //read
5| SUB R4, R12, R3
6| LDR R5, [R0 + R3]
7| LDR R6, [R0 + R4]
8| STR R5, [R0 + R4]
9| STR R6, [R0 + R3]
10| //enter next element
11| ADD R2, R2, #1
12| ADD R3, R3, #4
13| //check if through arr mid ele
14| CMP R3, R4
15| BNE loop
16| HALT
17| .ALIGN 256
18| arrayLength: 10
19| arrayData: 9
20| 8
21| 7
22| 6
23| 5
24| 4
25| 3
26| 2
27| 1
28| 0
```

Processor

PC: 0x00000030  
LR: 0x00000000  
SP: 0x00100000  
R12: 0x00000024  
R11: 0x00000000  
R10: 0x00000000  
R9: 0x00000000  
R8: 0x00000000  
R7: 0x00000000  
R6: 0x00000004  
R5: 0x00000005  
R4: 0x00000014  
R3: 0x00000014  
R2: 0x00000005  
R1: 0x00000000  
R0: 0x00000104

Count: 48  
Current Instruction:   
Status bits: NZCV 0110

Input/Output  
Program HALTED. STOP, LOAD or EDIT

Memory

000	0x0	0x4	0x8	0xc
0x0000	0xe3a00f41	0xe3a0c024	0xe04c4003	0xe7905003
0x0001	0xe7906004	0xe7805004	0xe7806003	0xe2822001
0x0002	0xe2833004	0xe1530004	0x1afffff6	0xe1000070
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	0x00000003	0x00000004	0x00000001	0x00000002
0x0011	0x00000000	0x00000000	0x00000005	0x00000006
0x0012	0x00000007	0x00000008	0x00000009	0x00000000
0x0013	0x00000000	0x00000000	0x00000000	0x00000000
0x0014	0x00000000	0x00000000	0x00000000	0x00000000
0x0015	0x00000000	0x00000000	0x00000000	0x00000000
0x0016	0x00000000	0x00000000	0x00000000	0x00000000
0x0017	0x00000000	0x00000000	0x00000000	0x00000000
0x0018	0x00000000	0x00000000	0x00000000	0x00000000
0x0019	0x00000000	0x00000000	0x00000000	0x00000000
0x001a	0x00000000	0x00000000	0x00000000	0x00000000
0x001b	0x00000000	0x00000000	0x00000000	0x00000000
0x001c	0x00000000	0x00000000	0x00000000	0x00000000
0x001d	0x00000000	0x00000000	0x00000000	0x00000000
0x001e	0x00000000	0x00000000	0x00000000	0x00000000
0x001f	0x00000000	0x00000000	0x00000000	0x00000000