

## COS-10004 (Computer Systems)

**Name:** Dave Nguyen (Nguyen Quang Anh).

**ID:** 104697710.

**Q9.1.1 (a):**

The screenshot displays a computer system simulator interface. On the left, a list of assembly instructions is shown, with line 34 highlighted in orange:

```
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 STR R1, .Pixel52
23 STR R1, .Pixel53
24 STR R1, .Pixel54
25 STR R1, .Pixel55
26 STR R1, .Pixel56
27 STR R1, .Pixel57
28 STR R1, .Pixel58
29 STR R1, .Pixel59
30 STR R1, .Pixel60
31 STR R1, .Pixel61
32 STR R1, .Pixel62
33 STR R1, .Pixel63
34 HALT
```

On the right, the hardware status is displayed:

- PC:** 0x00000088
- 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

Control and status indicators include:

- Count:** 32
- Current Instruction:** (empty)
- Status bits:** NZCV 0000
- Input/Output:** Program HALTED. STOP, LOAD or EDIT

On the far right, a memory dump shows hexadecimal values for addresses 0x0000 to 0x001f. Address 0x0007 contains the value 0xe1000070, which is highlighted in orange.

**Q9.1.1 (b):**

```

1|  MOV R1, #.red
2|  STR R1, .Pixel15
3|  STR R1, .Pixel147
4|  STR R1, .Pixel179
5|  STR R1, .Pixel111
6|  STR R1, .Pixel143
7|  STR R1, .Pixel175
8|  STR R1, .Pixel107
9|  STR R1, .Pixel139
10| STR R1, .Pixel1271
11| STR R1, .Pixel1303
12| STR R1, .Pixel1335
13| STR R1, .Pixel1367
14| STR R1, .Pixel1399
15| STR R1, .Pixel1431
16| STR R1, .Pixel1463
17| STR R1, .Pixel1495
18| STR R1, .Pixel1527
19| STR R1, .Pixel1559
20| STR R1, .Pixel1591
21| STR R1, .Pixel1623
22| STR R1, .Pixel1655
23| STR R1, .Pixel1687
24| STR R1, .Pixel1719
25| STR R1, .Pixel1751
26| 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

```

1|  MOV R1, #.PixelScreen // base address of the medium and high res pixel display memory
2|  MOV R2, #.red
3|  MOV R3, #0
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, #4
8|  CMP R3, #80
9|  BLT loop
10| HALT

```

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):

The screenshot displays a program execution environment with two main panels: 'Program' and 'Processor'.

**Program Panel:** Shows assembly code with line numbers 1 through 10. Line 10, 'HLT', is highlighted in orange.

```
1 MOV R1, #.PixelScreen // base address of the medium and high res pixel display memory
2 MOV R2, #.red
3 MOV R3, #256
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, #4
8   CMP R3, #340
9   BLT loop
10  HLT
```

**Processor Panel:** Displays the state of the processor registers and control units.

- Registers:** A list of registers from PC to R0. R1 is highlighted in orange with the value 0xffff3000.
- Control Units:** Includes a 'Count' unit set to 100, a 'Current Instruction' unit, and 'Status bits' set to 0110.
- Input/Output:** A section labeled 'Saving File' with a text input field.
- Memory Dump:** A table on the right showing memory addresses and their corresponding values. Address 0xe1000070 is highlighted in orange.

Address	Value
0x00000000	0x0
0x00000001	0xe32d1000
0x00000002	0xe5842000
0x00000003	0xe1000070
0x00000004	0x00000000
0x00000005	0x00000000
0x00000006	0x00000000
0x00000007	0x00000000
0x00000008	0x00000000
0x00000009	0x00000000
0x0000000a	0x00000000
0x0000000b	0x00000000
0x0000000c	0x00000000
0x0000000d	0x00000000
0x0000000e	0x00000000
0x0000000f	0x00000000
0x00000010	0x00000000
0x00000011	0x00000000
0x00000012	0x00000000
0x00000013	0x00000000
0x00000014	0x00000000
0x00000015	0x00000000
0x00000016	0x00000000
0x00000017	0x00000000
0x00000018	0x00000000

### 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: N Z C V

Status bits: 0110

### Input/Output

Program HALTED. STOP, LOAD or EDIT

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: N Z C V

Status bits: 0110

### Input/Output

Saving File

### Memo

0x000	0xe32d1000	0xe3a028ff
0x001	0xe0814003	0xe7842005
0x002	0xbafffffa	0xe1000070
0x003	0x00000000	0x00000000
0x004	0x00000000	0x00000000
0x005	0x00000000	0x00000000
0x006	0x00000000	0x00000000
0x007	0x00000000	0x00000000
0x008	0x00000000	0x00000000
0x009	0x00000000	0x00000000
0x00a	0x00000000	0x00000000
0x00b	0x00000000	0x00000000
0x00c	0x00000000	0x00000000
0x00d	0x00000000	0x00000000
0x00e	0x00000000	0x00000000
0x00f	0x00000000	0x00000000
0x010	0x00000000	0x00000000
0x011	0x00000000	0x00000000
0x012	0x00000000	0x00000000
0x013	0x00000000	0x00000000
0x014	0x00000000	0x00000000
0x015	0x00000000	0x00000000
0x016	0x00000000	0x00000000
0x017	0x00000000	0x00000000
0x018	0x00000000	0x00000000

Q9.2.2:



### 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: BL addr

Status bits: NZCV 0000

Input/Output: Done instruction BL addr at line unknown (PC=0x00130)

000	0x0
0x000	0xe3a01f41
0x001	0xe1000070
0x002	0x00000000
0x003	0x00000000
0x004	0x00000000
0x005	0x00000000
0x006	0x00000000
0x007	0x00000000
0x008	0x00000000
0x009	0x00000000
0x00a	0x00000000
0x00b	0x00000000
0x00c	0x00000000
0x00d	0x00000000
0x00e	0x00000000
0x00f	0x00000000
0x010	0x0000000a
0x011	0x00000006
0x012	0x00000002
0x013	0x00000000
0x014	0x00000000
0x015	0x00000000
0x016	0x00000000
0x017	0x00000000
0x018	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	0x00000010
R0	0x00000104

Count: 320

Current Instruction:

Status bits: NZCV 0110

Input/Output: 36 Program HALTED. STOP, LOAD or EDIT

### Memor

000	0x0	0x4
0x000	0xe3a00f41	0xe3a01004
0x001	0xe0822003	0xe2811004
0x002	0x50f2114	0xe1000070
0x003	0x00000000	0x00000000
0x004	0x00000000	0x00000000
0x005	0x00000000	0x00000000
0x006	0x00000000	0x00000000
0x007	0x00000000	0x00000000
0x008	0x00000000	0x00000000
0x009	0x00000000	0x00000000
0x00a	0x00000000	0x00000000
0x00b	0x00000000	0x00000000
0x00c	0x00000000	0x00000000
0x00d	0x00000000	0x00000000
0x00e	0x00000000	0x00000000
0x00f	0x00000000	0x00000000
0x010	0x0000000a	0x00000009
0x011	0x00000006	0x00000005
0x012	0x00000002	0x00000001
0x013	0x00000000	0x00000000
0x014	0x00000000	0x00000000
0x015	0x00000000	0x00000000
0x016	0x00000000	0x00000000
0x017	0x00000000	0x00000000
0x018	0x00000000	0x00000000
0x019	0x00000000	0x00000000

Q9.3.3:



### 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: NZCV

Status bits: 0110

### Input/Output

86

Saving File

### Memory

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

```

1  MOV R0,#arrayData1
2  MOV R4,#arrayData2
3  MOV R1,#36          // index
4  arrayloop:
5      LDR R3, [R0+R1] //pointer to the array
6      STR R3,[R4]     // display the value inside of the array
7      STR R3,WriteUnsignedNum // write what's inside the array in the display
8      SUB R1,R1,#4    // subtract the index of the array by 4 bytes
9      CMP R1, #0      // R1 - 0
10     BNE arrayloop
11     HALT
12     .ALIGN 256
13     arrayLength1: 10
14     arrayData1: 9
15         8
16         7
17         6
18         5
19         4
20         3
21         2
22         1
23         0
24     arrayLength2: 10
25     arrayData2: 0

```

### Processor

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: MOV Rd,#im

Status bits: NZCV

### Input/Output

Saving File

### Memory

000	0x0
0x0000	0xe3a00f41
0x0001	0xe5843000
0x0002	0x1afffff9
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

Q9.4.2:

### 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

0x00000000

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	0xe0351000	0xc1affffff	0xe01000070
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