



Department of Computer Science & Engineering
Microprocessor & Computer Architecture Lab
SRN:PES2UG22CS546
Lab 3 Submission Format

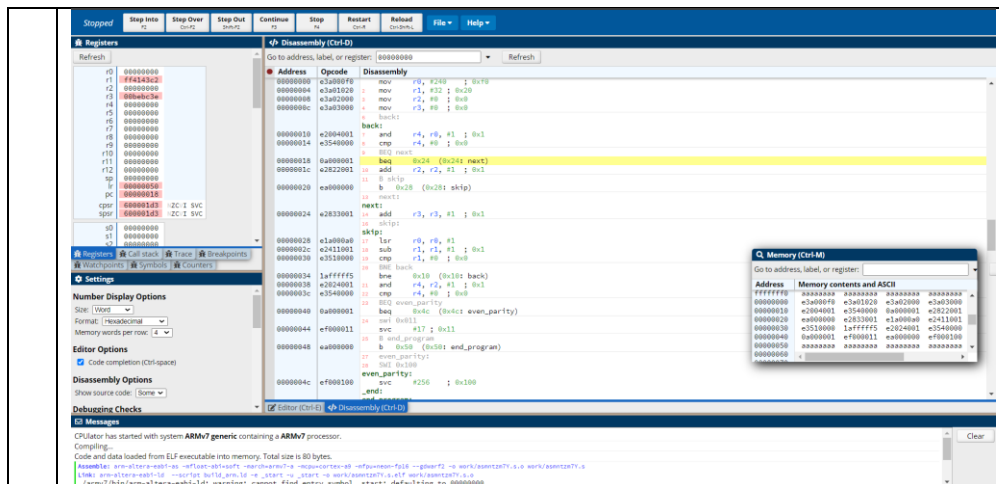
UE22CS251B

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

Program:

```
1  MOV R0, #0x00F0
2  MOV R1, #32
3  MOV R2, #0
4  MOV R3, #0
5
6  back:
7  AND R4,R0,#1
8  CMP R4,#0
9  BEQ next
10 ADD R2, R2, #1
11 B skip
12
13 next:
14 ADD R3, R3, #1
15
16 skip:
17 LSR R0,R0,#1
18 SUB R1, R1, #1
19 CMP R1, #0
20 BNE back
21 AND R4,R2,#1
22 CMP R4,#0
23 BEQ even_parity
24 swi 0x011
25 B end_program
26
27 even_parity:
28 SWI 0x100
29
30 end_program:
31
```

Output Screen Shot:



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

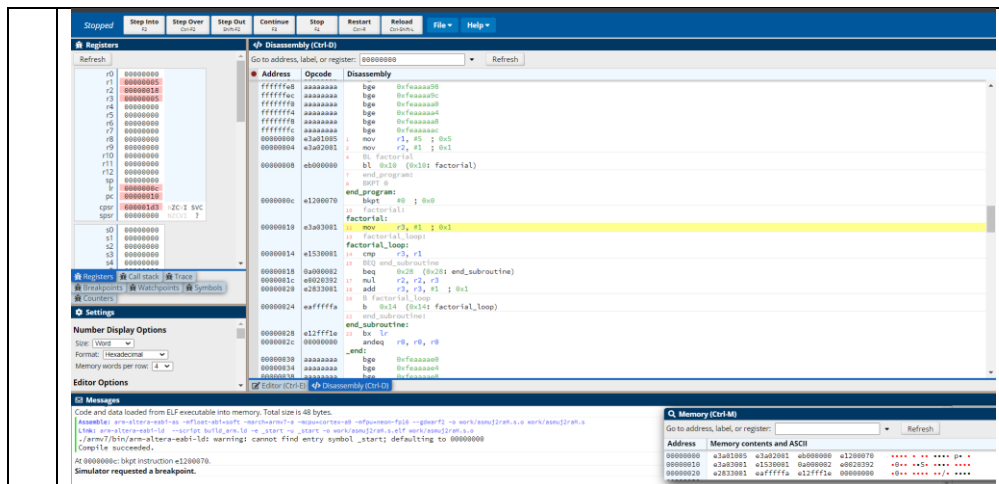
Program:

```

W3_P2.S
1      MOV R1, #5
2      MOV R2, #1
3
4      BL factorial
5
6
7      end_program:
8      BKPT 0
9
10     factorial:
11         MOV R3, #1
12
13     factorial_loop:
14         CMP R3, R1
15         BEQ end_subroutine
16
17         MUL R2, R2, R3
18         ADD R3, R3, #1
19
20         B factorial_loop
21
22     end_subroutine:
23         BX LR
24

```

Output Screen Shot:



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

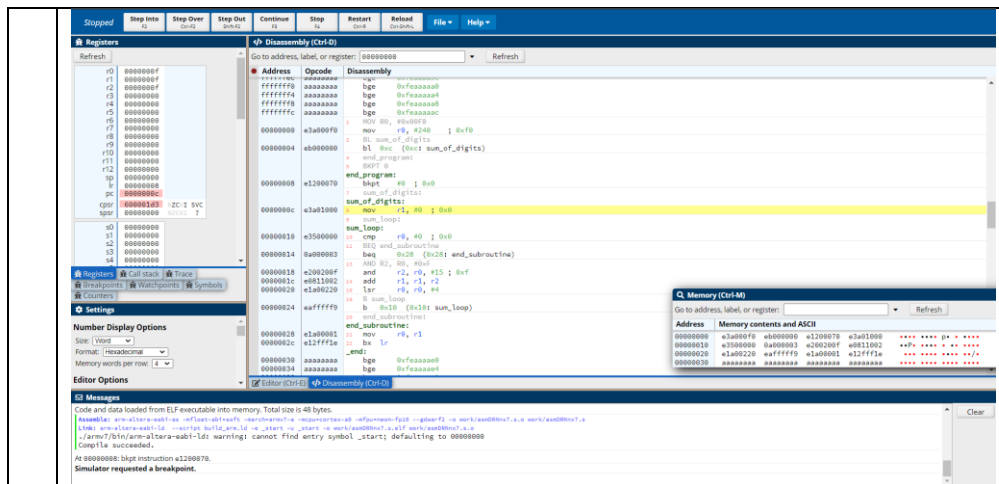
Program:

```

W3_P3.s
1  MOV R0, #0x00F0
2  BL sum_of_digits
3
4  end_program:
5      BKPT 0
6
7  sum_of_digits:
8      MOV R1, #0
9  sum_loop:
10     CMP R0, #0
11     BEQ end_subroutine
12
13     AND R2, R0, #0xF
14     ADD R1, R1, R2
15
16     LSR R0, R0, #4
17
18     B sum_loop
19
20 end_subroutine:
21     MOV R0, R1
22     BX LR
23

```

Output Screen Shot:



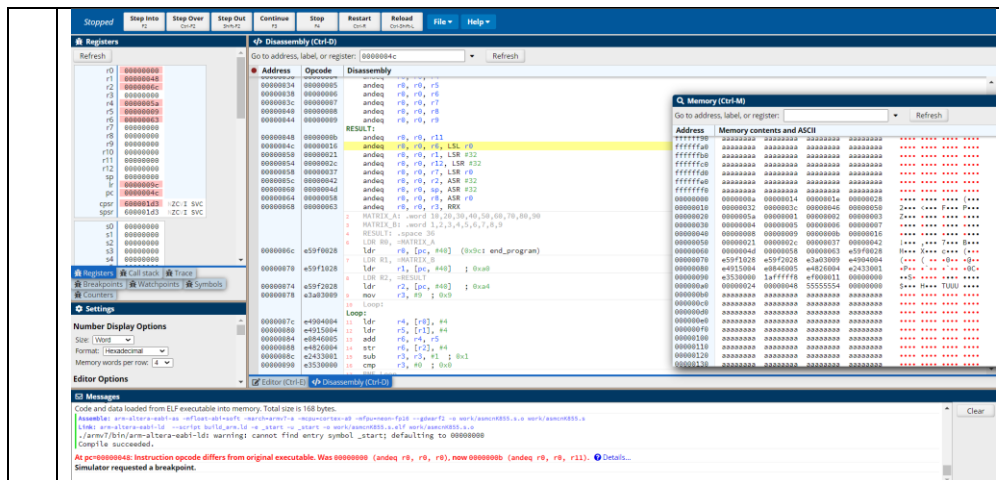
- 4 Write a program to perform 2X2 matrix addition. (you may Try for 3 X 3).
Program:

```

W3_P4.s
1
2     MATRIX_A: .word 10,20,30,40,50,60,70,80,90
3     MATRIX_B: .word 1,2,3,4,5,6,7,8,9
4     RESULT: .space 36
5
6     LDR R0, =MATRIX_A
7     LDR R1, =MATRIX_B
8     LDR R2, =RESULT
9     MOV R3, #9
10    Loop:
11    LDR R4, [R0], #4
12    LDR R5, [R1], #4
13    ADD R6, R4, R5
14    STR R6, [R2], #4
15    SUB R3, R3, #1
16    CMP R3, #0
17    BNE Loop
18    SWI 0x011
19    end_program:

```

Output Screen Shot:



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

Program:

```

W3_P5.s
1  A: .word 5,10,15,20,25,30,35,40,45,50
2  LDR R0, =A
3  MOV R1, #10
4  MOV R3, #10
5  MOV R2, #0
6
7  LOOP:
8      LDR R4, [R0], #4
9      CMP R4, R3
10     BEQ KEYFOUND
11     SUB R1, R1, #1
12     CMP R1, #0
13     BNE LOOP
14     B NOTFOUND
15
16 KEYFOUND:
17     MOV R2, #1
18
19 NOTFOUND:
20     SWI 0x11
21     .end
22

```

Output Screen Shot:

The screenshot shows a debugger interface with the following components:

- Registers:** A list of registers (r0-r15) with their current values. r0 is 00000000, r1 is 00000000, r2 is 00000000, r3 is 00000000, r4 is 00000000, r5 is 00000000, r6 is 00000000, r7 is 00000000, r8 is 00000000, r9 is 00000000, r10 is 00000000, r11 is 00000000, r12 is 00000000, r13 is 00000000, r14 is 00000000, r15 is 00000000.
- Disassembly:** A list of instructions with their addresses and opcodes. The first instruction is 'andeq r0, r0, #0' at address 00000000. Other instructions include 'ldr r0, [pc, #4]', 'mov r1, #0', 'mov r2, #0', 'ldr r0, [pc, #4]', 'mov r1, #0', 'mov r2, #0', 'ldr r0, [pc, #4]', 'mov r1, #0', 'mov r2, #0'.
- Memory:** A window showing memory contents and ASCII values. It displays a large block of memory starting at address 00000000.
- Messages:** A window showing messages from the debugger. It displays a message: 'Code and data loaded from ELF executable into memory. Total size is 96 bytes.'

6 Assignment Questions:

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

The screenshot shows a debugger interface with the following components:

- Registers:** A list of registers (r0-r15) with their current values. r0 is 00000000, r1 is 00000000, r2 is 00000000, r3 is 00000000, r4 is 00000000, r5 is 00000000, r6 is 00000000, r7 is 00000000, r8 is 00000000, r9 is 00000000, r10 is 00000000, r11 is 00000000, r12 is 00000000, r13 is 00000000, r14 is 00000000, r15 is 00000000.
- Disassembly:** A list of instructions with their addresses and opcodes. The first instruction is 'ldr r0, [pc, #4]' at address 00000000. Other instructions include 'mov r1, #0', 'mov r2, #0', 'ldr r0, [pc, #4]', 'mov r1, #0', 'mov r2, #0', 'ldr r0, [pc, #4]', 'mov r1, #0', 'mov r2, #0'.
- Memory:** A window showing memory contents and ASCII values. It displays a large block of memory starting at address 00000000.
- Messages:** A window showing messages from the debugger. It displays a message: 'Code and data loaded from ELF executable into memory. Total size is 112 bytes.'

Program:

```

W3_P6.s
1  A: .BYTE 1,2,3,4,5,6,7,8,9,10
2  LDR R0, =A
3  MOV R1, #5
4  MOV R2, #-1
5  MOV R3, #0
6  MOV R4, #9
7  MOV R8, #0
8  L:
9  CMP R4, R3
10 BMI EXIT
11 ADD R5, R3, R4
12 ADD R6, R8, R5, LSR #1
13 LDRB R7, [R0,R6]
14 CMP R7, R1
15 BEQ FOUND
16 BMI LESSER
17 SUB R6, R6, #1
18 MOV R4, R6
19 B L
20 FOUND:
21 MOV R2, R6
22 SWI 0x011
23 LESSER:
24 ADD R6, R6, #1
25 MOV R3, R6
26 B L
27 EXIT:
28 SWI 0x011

```

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.

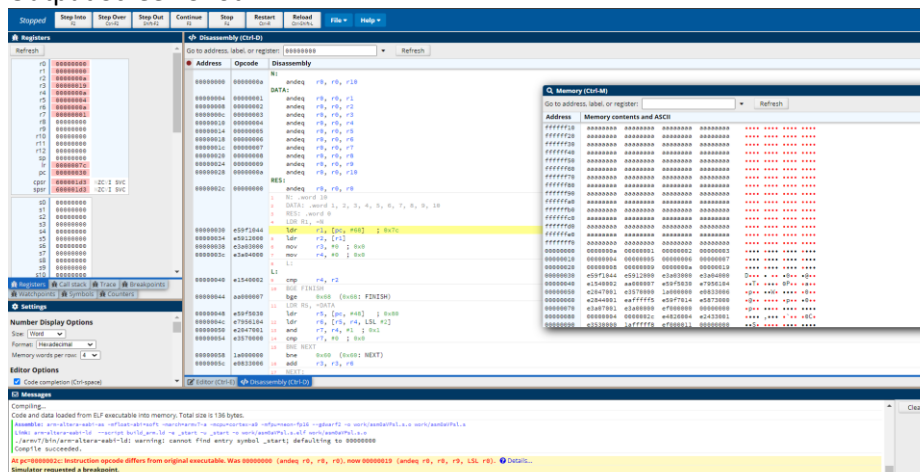
Program:

```

W3_P6_1.s
1  N: .word 10
2  DATA: .word 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
3  RES: .word 0
4  LDR R1, =N
5  LDR R2, [R1]
6  MOV R3, #0
7  MOV R4, #0
8  L:
9  CMP R4, R2
10 BGE FINISH
11 LDR R5, =DATA
12 LDR R6, [R5, R4, LSL #2]
13 AND R7, R4, #1
14 CMP R7, #0
15 BNE NEXT
16 ADD R3, R3, R6
17 NEXT:
18 ADD R4, R4, #1
19 B L
20 FINISH:
21 LDR R7, =RES
22 STR R3, [R7]
23 MOV R7, #1
24 MOV R0, #0
25 SWI 0

```

Output Screen Shot:



Note:

- Link to upload the file:
 - Will be provided by the respective Theory Teacher
- Upload PDF only.

- Save your file with your SRN _ Name