

Department of Computer Science & Engineering Microprocessor & Computer Architecture Lab

Lab 5 Submission Format

UE22CS251B

|  |  |
| --- | --- |
| 1 | Write an ALP to multiply 2 matrices.  Program:  .data  matrixA: .word 1, 2, 3, 4  .word 5, 6, 7, 8  .word 9, 10, 11, 12  .word 13, 14, 15, 16  matrixB: .word 2, 0, 1, 3  .word 1, 2, 0, 1  .word 0, 1, 2, 0  .word 3, 0, 1, 2  resultMatrix: .space 16  .text  .global \_start  \_start:  MOV r0, #4  MOV r1, #4  MOV r2, #4  BL matrix\_multiply B \_exit  matrix\_multiply:  PUSH {r4-r10, lr}  LDR r4, =matrixA LDR r5, =matrixB  LDR r6, =resultMatrix  MOV r7, #0 |

|  |  |
| --- | --- |
|  | outer\_loop:  MOV r8, #0  inner\_loop:  MOV r9, #0  MOV r10, #0  sum\_loop:  LDR r11, [r4, r7, LSL #2] LDR r12, [r5, r8, LSL #2] MUL r11, r11, r12  ADD r9, r9, r11  ADD r10, r10, #1 CMP r10, r2  B*NE* sum\_loop  STR r9, [r6, r7, LSL #2] ADD r8, r8, #1  CMP r8, r1  B*NE* inner\_loop  ADD r7, r7, #1 CMP r7, r0  B*NE* outer\_loop POP {r4-r10, pc}  \_exit:  MOV r7, #1 SWI 0x11  Screenshot: |

|  |  |
| --- | --- |
|  |  |
| 2 | Write an ALP using conditional ARM instructions to sort an array of numbers using Bubble Sort Algorithm.  Program:  .data  array: .word 8, 3, 1, 5, 4, 2, 7, 6  size: .word 8  .text  .global \_start  \_start:  LDR r0, =array LDR r1, =size  BL bubble\_sort B \_exit  bubble\_sort:  PUSH {r4-r9, lr}  LDR r4, [r1]  SUBS r4, r4, #1  outer\_loop:  MOV r5, #0 |

|  |  |
| --- | --- |
|  | inner\_loop:  LDR r6, [r0, r5, LSL #2] LDR r7, [r0, r5, LSL #2]  ADD r8, r5, #1  LDR r9, [r0, r8, LSL #2]  CMP r6, r9  B*LE* no\_swap  STR r9, [r0, r5, LSL #2] STR r7, [r0, r8, LSL #2]  no\_swap:  ADD r5, r5, #1 CMP r5, r4  B*LT* inner\_loop  SUBS r4, r4, #1 CMP r4, #0  B*GT* outer\_loop POP {r4-r9, pc}  \_exit:  MOV r7, #1 SWI 0x11  Screenshot: |

|  |  |
| --- | --- |
|  |  |
| 3 | **Assignment:**  i)  Write a program to swap the first and last character of a given string. Example:  Input: ‘dog’ Output: ‘god’  Program:  .data  string: .asciz "dog"  .text  .global \_start  \_start:  LDR r0, =string  BL swap\_first\_last B \_exit  swap\_first\_last: PUSH {r4-r6, lr}  LDRB r4, [r0] LDR r5, =1  ADD r6, r0, r5  LDRB r6, [r6] |

|  |  |
| --- | --- |
|  | STRB r6, [r0] STRB r4, [r0, r5]  POP {r4-r6, pc}  \_exit:  MOV r7, #1 SWI 0x11  Screenshot:    ii)  Given a c Code convert it in its equivalent Arm Code.  **a)x = (a + b) - c;**  Program:  .data  a: .word 10  b: .word 5  c: .word 3  x: .word 0  .text  .global \_start  \_start:  LDR r1, =a |

|  |  |
| --- | --- |
|  | LDR r2, =b LDR r3, =c  LDR r4, [r1] LDR r5, [r2] LDR r6, [r3]  ADD r7, r4, r5 SUB r8, r7, r6  STR r8, [r0]  B \_exit  \_exit:  MOV r7, #1 SWI 0x11  Screenshot:    b)  **z = (a << 2) |(b & 15);**  Program:  .data  a: .word 7  b: .word 12  z: .word 0 |

|  |  |
| --- | --- |
|  | .text  .global \_start  \_start:  LDR r1, =a LDR r2, [r1]  LDR r3, =b LDR r4, [r3]  LSL r2, r2, #2 AND r5, r4, #15 ORR r6, r2, r5  LDR r7, =z STR r6, [r7]  B \_exit  \_exit:  MOV r7, #1 SWI 0x11  Screenshot: |