Name : SHAILESH PUPALWAR
Roll no. : EE20B100

ARM ASSEMBLY – COMPUTATIONS IN ARM

AIM :

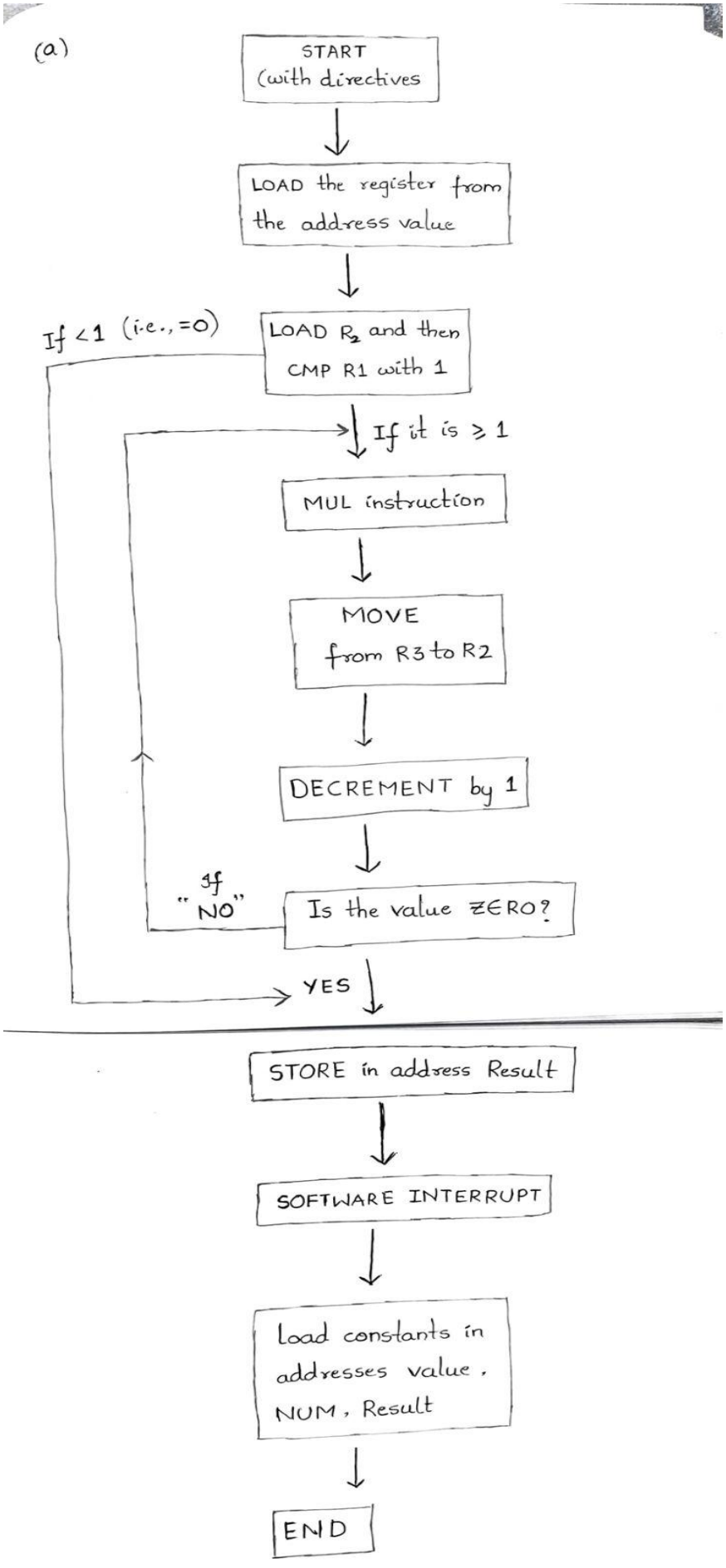
To (a) learn the architecture of ARM processors , (b) learn basics of ARM instruction set, in particular the ARM instructions pertaining to computations (c) go through example programs and (d) write assembly language programs for the given set of (computational) problems.

QUESTION-1 :

1. Compute the factorial of a given number using an ARM processor through assembly programming.

a) FLOW CHART :

(a)


START
(with directives
LOAD the register from
the address value
If <1 (i.e., =0)
LOAD R2 and then
CMP R1 with 1
If it is ≥ 1
MUL instruction
MOVE
from R3 to R2
DECREMENT by 1
If
"NO"
Is the value ZERO?
YES
STORE in address Result
SOFTWARE INTERRUPT
Load constants in
addresses value,
NUM, Result
END

CODE FOR QUESTION-1:

* factorial of a number

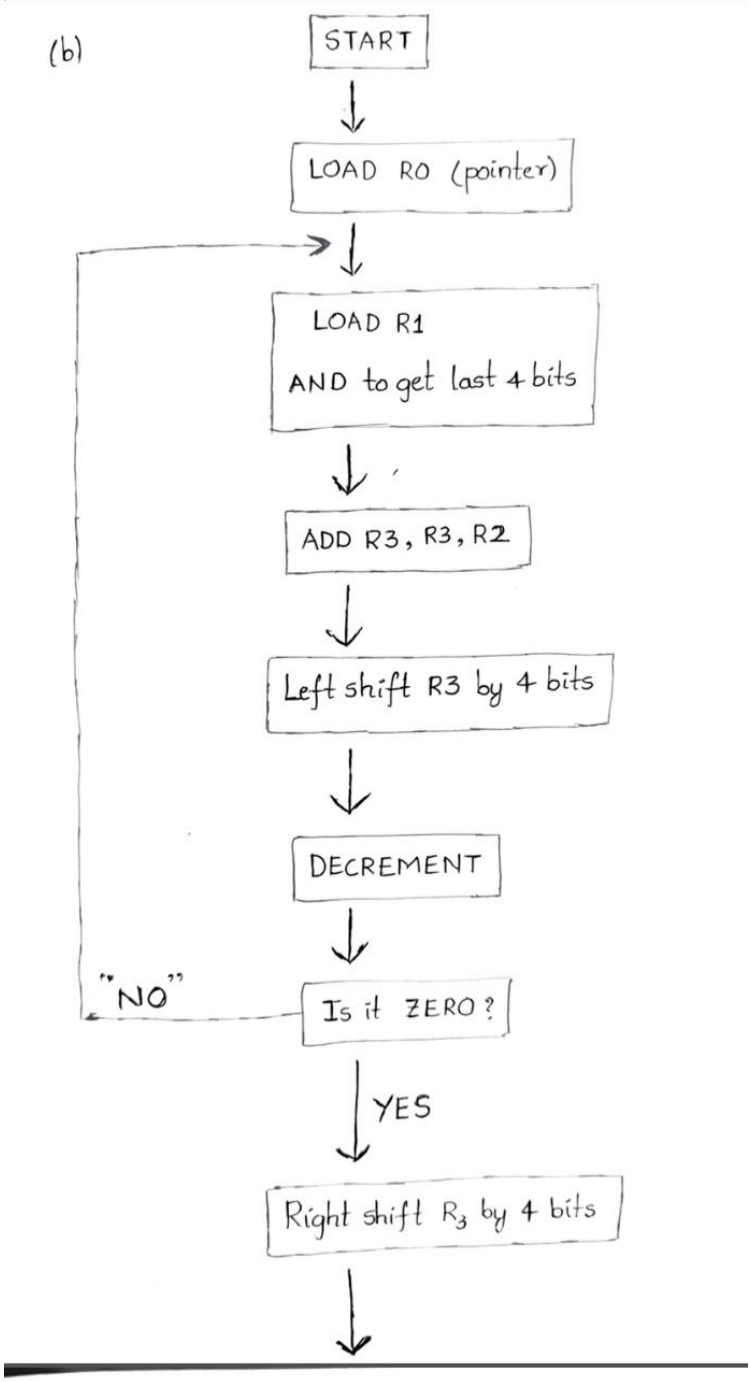
```
        TTL factorialnum
        AREA Program,CODE,READONLY
        ENTRY
Main
        LDR R2,NUM
        LDR R1,Value ; load the registers.
  CMP R1,#01
  BLT final

loop
        MUL R3,R2,R1 ; multiply values in R1,R2 and store them in R3
  MOV R2,R3 ;
  SUBS R1,R1,#1 ; subtract the value in R1 with the value in R4 i.e,1.
  BNE loop ; use branch instruction to end the loop.
final
        STR R2,Result ; now the final answer is stored at a new address.
  SWI &11

Value DCD &00 ; store constant values in corresponding addresses using DCD
directive.
NUM DCD &01
Result DCD 0 ;initialise the result value to zero.
  END
```

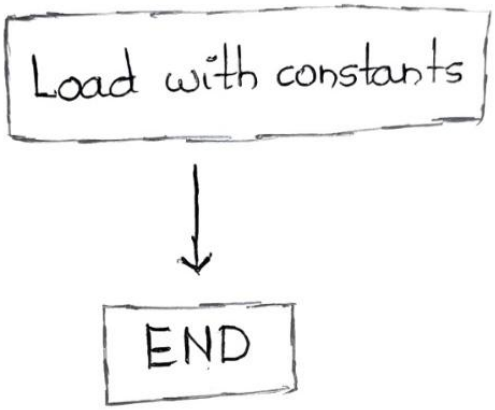
QUESTION-2 :

Combine the low four bits of each of the four consecutive bytes beginning at LIST into one 16-bit halfword. The value at LIST goes into the most significant nibble of the result. Store the result in the 32-bit variable RESULT.

b) : FLOWCHART :

(b)


START
LOAD R0 (pointer)
LOAD R1
AND to get last 4 bits
ADD R3, R3, R2
Left shift R3 by 4 bits
DECREMENT
"NO"
Is it ZERO?
YES
Right shift R3 by 4 bits


Load with constants
END

CODE FOR QUESTION-2 :

* 32-bit to 16-bit halfword

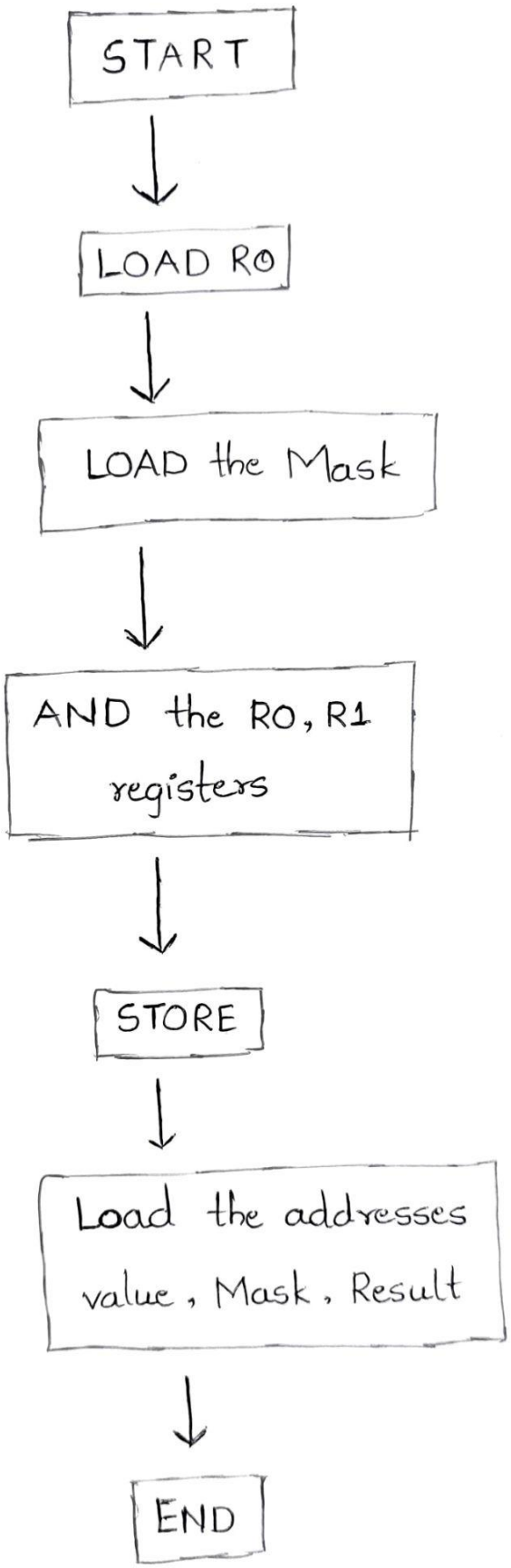
```
        TTL wordtohalfword
   AREA Program,CODE,READONLY
   ENTRY
Main
        LDR R0,LIST ;pointer to start list.
   LDR R1, [R0] ,#4 ;no of values of list is taken as the first element.
loop
        LDR R2, [R0], #4 ;
   AND R2, R2, #0xF ;masking to get the last 4 bits of a byte.
   ADD R3, R3, R2
   LSL R3, #04
   SUBS R1, R1, #1 ;like the decrement
   BNE loop
   LSR R3, #4 ;right shift by 4 bits to get the final value
   SWI &11

NUM DCD &4
        DCD &2A,&3D,&55,&0C4 ; 4 bytes taken

LIST DCD NUM
        END
```

QUESTION-3 :

Given a 32 bit number, identify whether it is an even or odd. (Your implementation should not involve division).

c) : FLOW CHART :

(c)


START
LOAD R0
LOAD the Mask
AND the R0, R1 registers
STORE
Load the addresses value, Mask, Result
END

CODE FOR QUESTION-3 :

* even or odd

```
        TTL evenodd
        AREA Program, CODE, READONLY
        ENTRY
Main
        LDR R0,Value ;load the 32-bit number to R0
        LDR R1,Mask ;load the bitmask
        AND R0,R0,R1 ;
        STR R0,Result ;if R0=0 it is even,if R0=1 it is odd
        SWI &11

Value DCD &F234
Mask DCD &001
Result DCD 0
        END
```