```
;/* PROGRAM TO FIND FACTORIAL OF A GIVEN NUMBER                                  */
;/* In this example we have taken n=7                                            */
;/* Check the result in R0/R3 register =13B0H (5040)                             */
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT       */

      AREA    FACTORIAL , CODE, READONLY

ENTRY                   ;Mark first instruction to execute

START

          MOV r0, #5                    ; STORE FACTORIAL NUMBER IN R0
          MOV r1,r0                     ; MOVE THE SAME NUMBER IN R1

FACT      SUBS r1, r1, #1               ; SUBTRACTION
          CMP r1, #1                    ; COMPARISON
          BEQ STOP
          MUL r3,r0,r1;                 ; MULTIPLICATION
          MOV       r0,r3                       ; Result
          BNE FACT                      ; BRANCH TO THE LOOP IF NOT EQUAL
STOP

          NOP
          NOP
          NOP

      END                       ;Mark end of file



;/* PROGRAM TO MULTIPLY TWO 16BIT NUMBERS                                        */
;/*   VALUE1: 1900H    (6400)          (IN R1)                                   */
;/*   VALUE2: 0C80H    (3200)    (IN R2)                                         */
;/*   RESULT: 1388000H(20480000) (IN R3)                                         */
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT       */


      AREA    MULTIPLY , CODE, READONLY

ENTRY                   ;Mark first instruction to execute

START

      MOV r1,#6400              ; STORE FIRST NUMBER IN R0
      MOV r2,#3200              ; STORE SECOND NUMBER IN R1
      MUL r3,r1,r2              ; MULTIPLICATION

      NOP
      NOP
      NOP

      END                ;Mark end of file
```

```
;/* PROGRAM TO ADD an array of 16BIT NUMBERS & STORE IN INTERNAL RAM
          */
;/*   ARRAY OF 6 NUMBERS      0X1111,0X2222,0X3333,0XAAAA,0XBBBB,0XCCCC
          */
;/* THE SUM IS 29997H THE RESULT CAN BE VIEWED IN LOCATION 0X40000000 & ALSO IN R0
*/
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT
          */


      AREA    ADDITION , CODE, READONLY

ENTRY                   ;Mark first instruction to execute

START
      MOV R5,#6                 ; INTIALISE COUNTER TO 6(i.e. N=6)
      MOV R0,#0                 ; INTIALISE SUM TO ZERO
      LDR R1,=VALUE1            ; LOADS THE ADDRESS OF FIRST VALUE
LOOP
      LDR R2,[R1],#2            ; WORD ALIGN T0 ARRAY ELEMENT
      LDR R3,MASK                     ; MASK TO GET 16 BIT
      AND R2,R2,R3              ; MASK MSB
      ADD R0,R0,R2              ; ADD THE ELEMENTS
      SUBS R5,R5,#1             ; DECREMENT COUNTER
      CMP R5,#0                 ;
      BNE LOOP                  ;   LOOK BACK TILL ARRAY ENDS

      LDR R4,=RESULT            ; LOADS THE ADDRESS OF RESULT
      STR R0,[R4]               ; STORES THE RESULT IN R1

      NOP
      NOP
      NOP

MASK DCD 0X0000FFFF               ;   MASK MSB

VALUE1 DCW      0X1111,0X2222,0X3333,0XAAAA,0XBBBB,0XCCCC    ;   ARRAY   OF   16   BIT
NUMBERS(N=6)



      AREA DATA2,DATA,READWRITE           ; TO STORE RESULT IN GIVEN ADDRESS
RESULT DCD 0X0

      END               ; Mark end of file



;/* PROGRAM TO DISASSEMBLE THE NUMBER                                          */
;/* CHECK THE RESULT IN R1,R4 & ALSO IN ADDRESS 40000000H                  */
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT     */
;/* PROGRAM TO ADD TWO      64BIT NUMBERS                                      */
;/*   VALUE1  0X1234E640 0X43210010 (R0,R1)                                */
;/* VALUE2    0X12348900 0X43212102 (R2,R3)                                */
;/*   RESULT  0X24696F40 0X86422112 (R5,R4)                                */
```

```
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT      */

      AREA    ADDITION , CODE, READONLY

ENTRY                       ;Mark first instruction to execute

START

      LDR R0,=0X1234E640      ;LOAD THE FIRST VALUE IN R0,R1
      LDR R1,=0X43210010
      LDR R2,=0X12348900      ;LOAD THE SECOND VALUE IN R2,R3
      LDR R3,=0X43212102
      ADDS R4,R1,R3           ;RESULT IS STORED IN R4,R5
      ADC R5,R0,R2

      NOP
      NOP
      NOP

      END                     ;Mark end of file


      AREA    DISASSEMBLE , CODE, READONLY

ENTRY                       ;Mark first instruction to execute

START

      LDR R1,VALUE                ; LOAD THE VALUE TO BE DISASSEMBLED
      LDR R2,MASK                 ; LOAD THE BITMASK
      MOV R3,R1,LSR#0X4           ; COPY JUST THE HIGH ORDER NIBBLE IN R3
      MOV R3,R3,LSL#0X8                 ; NOW LEFT SHIFT IT ONE BYTE
      AND R1,R1,R2                ; AND THE ORIGINAL NUMBER WITH BITMASK
      ADD R1,R1,R3                ; DISSASSEMBLED RESULT

      LDR R0,=RESULT              ; LOADS THE ADDRESS OF RESULT
      STR R1,[R0]                 ; STORES THE RESULT IN R1
      LDR R4,[R0]                 ; LOADS THE RESULT IN R4 FROM ADDRESS(40000000H)

      NOP
      NOP
      NOP


VALUE   DCD 0X00000024

MASK    DCD 0X0000000F

      AREA DATA1,DATA    ,READWRITE
RESULT  DCD 0X0                   ; RESULT STORED IN 40000000H

      END                     ;Mark end of file
```

```
;/* Assembly Program to find square of Number                                    */
;/* GIVEN NUMBER IS 6 (R1) THEN RESULT IS IN R3=24H(36)                          */
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT      */


      AREA    SQUARE , CODE, READONLY

ENTRY                   ;Mark first instruction to execute

START

      LDR R0, = TABLE1      ; Load start address of Lookup table
      LDR R1,= 2                  ; Load no whose square is to be find
      MOV R1, R1, LSL#0x2         ; Generate address corresponding to square of given no
      ADD R0, R0, R1              ; Load address of element in Lookup table
      LDR R3, [R0]            ; Get square of given no in R3

      NOP
      NOP
      NOP

;Lookup table contains Squares of nos from 0 to 10 (in hex)

TABLE1  DCD 0X00000000;           SQUARE OF 0=0
        DCD 0X00000001;           SQUARE OF 1=1
        DCD 0X00000004;           SQUARE OF 2=4
        DCD 0X00000009;           SQUARE OF 3=9
        DCD 0X00000010;           SQUARE OF 4=16
        DCD 0X00000019;           SQUARE OF 5=25
        DCD 0X00000024;           SQUARE OF 6=36
        DCD 0X00000031;           SQUARE OF 7=49
        DCD 0X00000040;           SQUARE OF 8=64
        DCD 0X00000051;           SQUARE OF 9=81
        DCD 0X00000064;           SQUARE OF 10=100

      END               ; Mark end of file




;/* PROGRAM TO FIND LARGEST NUMBER IN AN ARRAY & STORE IN INTERNAL RAM
*/
;/*   ARRAY OF 7 NUMBERS 0X44444444 ,0X22222222,0X11111111,0X33333333,0XAAAAAAAA
          */
;/*                                0X88888888 ,0X99999999                       */
;/* RESULT CAN BE VIEWED IN LOCATION 0X40000000 & ALSO IN R2                     */
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT
          */


      AREA    LARGEST , CODE, READONLY

ENTRY                   ;Mark first instruction to execute
```

```
START
      ;MOV R5,=VALUE1                    ; INTIALISE COUNTER TO 6(i.e. N=7)
       LDR R1,=VALUE1            ; LOADS THE ADDRESS OF FIRST VALUE
       MOV R5,[R1],#4
       LDR R2,[R1],#4            ; WORD ALIGN T0 ARRAY ELEMENT
LOOP
      LDR R4,[R1],#4             ; WORD ALIGN T0 ARRAY ELEMENT
      CMP R2,R4                  ; COMPARE NUMBERS
      BHI LOOP1                  ; IF THE FIRST NUMBER IS > THEN GOTO LOOP1

      MOV R2,R4                  ; IF THE FIRST NUMBER IS < THEN MOV CONTENT R4 TO R2
LOOP1
      SUBS R5,R5,#1              ; DECREMENT COUNTER
      CMP R5,#0                  ; COMPARE COUNTER TO 0
      BNE LOOP                   ; LOOP BACK TILL ARRAY ENDS

      LDR R4,=RESULT             ; LOADS THE ADDRESS OF RESULT
      STR R2,[R4]                ; STORES THE RESULT IN R2

      NOP
      NOP
      NOP


; ARRAY OF 32 BIT NUMBERS(N=7)

VALUE1
            DCD 0X00000000
            DCD 0X44444444             ;
            DCD0X22222222              ;
            DCD0X11111111              ;
            DCD0X33333333              ;
            DCD0X5AAAAAAA                    ;
            DCD0X88888888              ;
            DCD0X39999999              ;

            ;      AREA DATA1,DATA,READWRITE           ; TO STORE RESULT IN GIVEN ADDRESS

;RESULT1 DCD 0X0



      AREA DATA2,DATA,READWRITE           ; TO STORE RESULT IN GIVEN ADDRESS
RESULT DCD 0X0

      END                  ; Mark end of file


;/* PROGRAM TO FIND SMALLEST NUMBER IN AN ARRAY & STORE IN INTERNAL RAM
            */
;/*   ARRAY OF 7 NUMBERS 0X44444444 ,0X22222222,0X11111111,0X22222222,0XAAAAAAAA
            */
;/*                                    0X88888888 ,0X99999999                                    */
;/* RESULT CAN BE VIEWED IN LOCATION 0X40000000 & ALSO IN R2                                     */
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT
            */
```

```
    AREA    SMALLEST , CODE, READONLY

ENTRY                   ;Mark first instruction to execute

START
    MOV R5,#6           ; INTIALISE COUNTER TO 6(i.e. N=7)
    LDR R1,=VALUE1      ; LOADS THE ADDRESS OF FIRST VALUE
    LDR R2,[R1],#4      ; WORD ALIGN T0 ARRAY ELEMENT
LOOP
    LDR R4,[R1],#4      ; WORD ALIGN T0 ARRAY ELEMENT
    CMP R2,R4           ; COMPARE NUMBERS
    BLS LOOP1           ; IF THE FIRST NUMBER IS < THEN GOTO LOOP1

    MOV R2,R4           ; IF THE FIRST NUMBER IS > THEN MOV CONTENT R4 TO R2
LOOP1
    SUBS R5,R5,#1       ; DECREMENT COUNTER
    CMP R5,#0           ; COMPARE COUNTER TO 0
    BNE LOOP            ; LOOP BACK TILL ARRAY ENDS

    LDR R4,=RESULT      ; LOADS THE ADDRESS OF RESULT
    STR R2,[R4]         ; STORES THE RESULT IN R1

    NOP
    NOP
    NOP


; ARRAY OF 32 BIT NUMBERS(N=7)

VALUE1
        DCD 0X44444444      ;
        DCD0X22222222       ;
        DCD0X11111111       ;
        DCD0X22222222       ;
        DCD0XAAAAAAAA           ;
        DCD0X88888888       ;
        DCD0X99999999       ;


    AREA DATA2,DATA,READWRITE       ; TO STORE RESULT IN GIVEN ADDRESS
RESULT DCD 0X0

    END             ; Mark end of file


;/* Assembly Program to FIND LENGTH OF STRING                                  */
;/* CHECK THE RESULT IN R1 REGISTER                                           */
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT    */


    AREA    STRING1 , CODE, READONLY
```

```
ENTRY                   ;Mark first instruction to execute

START

    MOV R1, #0                  ; Counter for storing string length
    LDR R5, =TABLE1             ; LOAD THE ADDRESS OF TABLE1

LOOP

    LDRB R0, [R5],#1        ; Load first byte of String in R0
    CMP R0,#0                   ; Is It string terminator??
    BEQ STOP                    ; IF ITS EQUAL JUMP TO LOOP1
    ADD R1, R1, #1              ; Increament string length count by 1
    B LOOP                      ; BE IN A LOOP


STOP
    NOP
    NOP
    NOP

TABLE1 DCB " LPC2148 ALS BENGALURU ",0," SHETTY "; STRING LENGTH IS 17H (23 IN DECIMAL)

    END                     ;Mark end of file



;/* PROGRAM TO COUNT THE NUMBER OF ONES & ZEROS IN TWO CONSECUTIVE MEMORY LOCATIONS  */
;/*   WE TOOK TWO NUMBERS i.e. 0X11111111,0XAA55AA55  (R0)                          */
;/*   CHECK THE RESULT IN R2 FOR ONES & R3 FOR ZEROS
*/
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT
        */


  AREA   ONEZERO , CODE, READONLY

ENTRY                   ;Mark first instruction to execute

START

        MOV R2,#0               ; COUNTER FOR ONES
        MOV R3,#0               ; COUNTER FOR ZEROS
        MOV R7,#1               ; COUNTER TO GET TWO WORDS
        LDR R6,=VALUE           ; LOADS THE ADDRESS OF VALUE

LOOP    MOV R1,#16          ; 32 BITS COUNTER
        LDR R0,[R6],#4          ; GET THE 32 BIT VALUE

LOOP0   MOVS R0,R0,ROR #1       ; RIGHT SHIFT TO CHECK CARRY BIT (1's/0's)
        BHI ONES                ; IF CARRY BIT IS 1 GOTO ONES BRANCH OTHERWISE NEXT

ZEROS   ADD R3,R3,#1            ; IF CARRY BIT IS 0 THEN INCREMENT THE COUNTER BY 1(R3)
```

```
        B LOOP1                 ; BRANCH TO LOOP1

ONES    ADD R2,R2,#1            ; IF CARRY BIT IS 1 THEN INCREMENT THE COUNTER BY
1(R2)

LOOP1   SUBS R1,R1,#1           ; COUNTER VALUE DECREMENTED BY 1
        BNE LOOP0               ; IF NOT EQUAL GOTO TO LOOP0 CHECKS 32BIT

        SUBS R7,R7,#1           ; COUNTER VALUE DECREMENTED BY 1
        CMP R7,#0               ; COMPARE COUNTER R7 TO 0
        BNE LOOP                ; IF NOT EQUAL GOTO TO LOOP

    NOP
    NOP
    NOP


VALUE DCD 0X11111111        ;ONE    VALUES IN AN ARRAY

    END             ; Mark end of file



;/* PROGRAM TO SEARCH GIVEN NUMBER IN AN ARRAY    & GIVES THE POSITION          */
;/*    CHECK THE RESULT , IF IT IS FOUND THEN THE POSITION OF NUMBER IS IN R5      */
;/*    IF THE GIVEN NUMBER IS NOT FOUND R5 = 0                                    */
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT      */


    AREA    SEARCH1 , CODE, READONLY

ENTRY                   ;Mark first instruction to execute

START

        MOV R1,#0               ; INTIALISE COUNTER TO 1(N=6)
        LDR R2,=TABLE           ; LOADS THE ADDRESS OF FIRST VALUE
        MOV R5,#0               ; MAKE THE POSITION TO 0
LOOP
        LDR R4,[R2],#4          ; WORD ALIGN T0 ARRAY ELEMENT
        LDR R3,VALUE            ; VALUE TO BE FIND
        CMP R4,R3               ; COMPARE VALUE & STORED ARRAY
        BEQ FOUND               ; VALUE MATCHED WITH STORED ARRAY STOP
        SUB R1,R1,#1            ; DECREMENT COUNTER
        CMP R1,#7               ; IF GIVEN VALUE NOT FOUND IN ARRAY IT ENDS HERE
(NUM OF ELEMENTS)
        BEQ NOTFOUND            ; NOT FOUND
        BNE LOOP                ; LOOK BACK TILL ARRAY ENDS


FOUND   MOV R5,R1               ; POSITION OF SEARCHED ELEMENT IN ARRAY
        NOP
        NOP
        NOP
```

```
NOTFOUND
        NOP
        NOP

VALUE   DCD 0XAAAA1234              ; GIVEN NUMBER TO BE SEARCHED

TABLE   DCI  0X11110202,0X22220101,0XAAAA1234,0XABCD1234,0X1234BBBB,0XABCDCCCC;
ARRAY OF 32 BIT NUMBERS(N=6)


    END                 ; Mark end of file




;/* PROGRAM TO FIND HOW MANY NUMBERS ARE NEGATIVE IN AN ARRAY
        */
;/*  ARRAY OF 7 NUMBERS 0X12345678,0X8D489867,0X11111111,0X33333333,0XAAAAAAAA
        */
;/*                              0XE605546C ,0X99999999                          */
;/* RESULT CAN BE VIEWED   IN R2                                                 */
;/* SET A BREAKPOINT AT NOP INSTRUCTION,RUN THE PROGRAM & CHECK THE RESULT
        */


    AREA   NEGATIVE , CODE, READONLY

ENTRY                   ;Mark first instruction to execute

START
         MOV R5,#7              ; INTIALISE COUNTER TO 7(i.e. N=7)
         MOV R2,#0              ; COUNTER
         LDR R4,=VALUE          ; LOADS THE ADDRESS OF FIRST VALUE

LOOP
        LDR R1,[R4],#4          ; WORD ALIGN T0 ARRAY ELEMENT
        ANDS R1,R1,#1<<31       ; TO CHECK NEGATIVE NUMBER
        BHI FOUND               ; IF THE GIVEN NUMBER IS NEGATIVE GOTO FOUND
        B LOOP1                 ; IF THE GIVEN NUMBER IS   NOT NEGATIVE GOTO LOOP1

FOUND
        ADD R2,R2,#1            ; INCREMENT THE COUNTER (NEGATIVE NUMBER)
        B LOOP1                 ; GOTO LOOP1
LOOP1
        SUBS R5,R5,#1           ; DECREMENT COUNTER
        CMP R5,#0               ; COMPARE COUNTER TO 0
        BNE LOOP                ; LOOP BACK TILL ARRAY ENDS


    NOP
    NOP
    NOP
```

```
 ;ARRAY OF 32 BIT NUMBERS(N=7)

VALUE
        DCD0X12345678 ;
        DCD     0X8D489867    ;
        DCD     0X11111111    ;
        DCD     0X33333333    ;
        DCD0XE605546C     ;
        DCD0XAAAAAAAA     ;
        DCD0X99999999 ;

    END                 ; Mark end of file
```