1.a. ALP to add first 10 odd numbers. Store sum in register.
b. ALP to compute sum of squares of 5 numbers starting from 1. Write and use procedure SQU. Store sum in register

A].

```
        AREA ODDNO10,CODE,READONLY
ENTRY
        MOV R1,#1
        MOV R2,#9
        MOV R3,#1
BACK ADD R3,R3,#2
        ADD R1,R1,R3
        SUBS R2,R2,#1
        BNE BACK
GO B GO
        END
```

B].

```
        AREA SQSUM,CODE,READONLY
ENTRY
        MOV R1,#0
        MOV R2,#1
LOOP BL SQU
        ADD R1,R1,R4
        ADD R2,R2,#1
        CMP R2,#6
        BNE LOOP
GO B GO
SQU MUL R4,R2,R2
        MOV PC,LR
        END
```

2. a. ALP to add the first n even numbers. Store the result in a memory location.

b. ALP to generate a geometric progression with a limit n. Display the results in memory.

A]

```
        AREA ESUM,CODE,READONLY
N RN 1
RES RN 2
ENUM RN 3
ENTRY  MOV N,#5
```

```
        MOV RES,#0
        MOV ENUM,#2
        MOV R4,#0X40000000
LOOP ADD RES,RES,ENUM
     ADD ENUM,ENUM,#2
        SUBS N,N,#1
        BNE LOOP
         STR RES,[R4]
GO B GO
        END
```

B]

```
        AREA GP,CODE,READONLY
 A RN 1
 D RN 2
 N RN 3
 ENTRY
        MOV A,#1
        MOV D,#2
        MOV N,#10
        MOV R4,#0X40000000
 LOOP MUL R6,A,D
         MOV A,R6
         STR A,[R4]
         SUBS N,N,#1
         BNE LOOP
 GO B GO
        END
```

3]. a. ALP to count the number of zeroes and ones in a binary number.

b. ALP to find the average of ten 16-bit numbers stored in memory.

A]

```
        AREA BZO,CODE,READONLY
NUM RN 1
NONE RN 2
NZERO RN 3
ENTRY  LDR NUM,=0xB
```

```
        MOV NONE,#0
        MOV NZERO,#0
        MOV R5,#0X40000000
LOOP    LSRS NUM,#1
        ADDCS NONE,#1
        ADDCC NZERO,#1
        CMP NUM,#0
        BNE LOOP
        STR NONE,[R5]
        STR NZERO,[R5,#4]
GO B GO
        END
```

B]

```
        AREA BIT,CODE,READONLY
ENTRY  LDR R7,=TABLE
        MOV R0,#9
        LDRH R1,[R7]
BACK LDRH R2,[R7,#2]!
        ADD R1,R1,R2
        SUBS R0,R0,#1
        BNE BACK
        MOV R3,#10
        MOV R4,#0
        MOV R5,R1
BACK1 SUBS R5,R5,R3
        ADDPL R4,R4,#1
        BPL BACK1
        ADDMI R5,R5,R3
GO B GO
TABLE DCD 1000,2564,8936,344,5667,908,786,654,9761,456
        END
```

4]. a. ALP to find the factorial of a number.

b. ALP to generate the first n Fibonacci numbers.

A]

```
        AREA FCT,CODE,READONLY
N RN 1
FACT RN 2
ENTRY
        MOV N,#5
        MOV FACT,#1
LOOP    MUL FACT,N,FACT
        SUBS N,N,#1
        BNE LOOP
GO B GO
        END
B]
        AREA PROG13,CODE,READONLY
ENTRY
        MOV R1,#1
        LDR R2,=TABLE
        LDR R3,=NUMFIBONACCI
        LDRB R6,[R3]
        STRB R1,[R2],#1
        MOV R3,#0
        MOV R4,#0
        MOV R5,#1
        SUB R6,R6,#1
BACKK ADD R4,R3,R1
        STRB R4,[R2],#1
        MOV R3,R1
        MOV R1,R4
        ADD R5,R5,#1
        CMP R5,R6
        BLS BACKK
GO B GO
NUMFIBONACCI DCB 0x0A
AREA NUMBER,DATA,READWRITE
TABLE SPACE 60
END
```

5]. ALP to find the sum of digits of a number.

```
        AREA SUMDIG,CODE,READONLY
DIVD RN 1
DIVS RN 2
Q RN 3
R RN 4
RES RN 5
ENTRY  LDR DIVD,=12345
        MOV RES,#0
        MOV DIVS,#10
LOOP BL DIV
        ADD RES,RES,R
        CMP Q,#0
        MOV DIVD,Q
        BNE LOOP
GO B GO
DIV MOV Q,#0
LOOP1   SUBS DIVD,DIVD,DIVS
         ADDPL Q,Q,#1
        BPL LOOP1
        ADDMI R,DIVD,DIVS
        BX LR
        END
```

6] ALP to select a set of r objects from a set of n objects without considering the order of elements in a selection using combination method

```
  AREA COMB,CODE,READONLY
DIVD RN 1
DIVS RN 2
Q RN 3
REM RN 4
N RN 9
R RN 10
```

```
NDR RN 11
ENTRY  MOV N,#6
       MOV R,#3
       SUBS NDR,N,R
       MOV DIVD,N
       BL FACT
       MOV N,DIVS
       MOV DIVD,R
       BL FACT
       MOV R,DIVS
       MOV DIVD,NDR
       BL FACT
       MOV NDR,DIVS
       MUL DIVS,R,NDR
       MOV DIVD,N
       BL DIV
STOP B STOP
FACT MOV DIVS,#1
LOOP1 MUL DIVS,DIVD,DIVS
   SUBS DIVD,DIVD,#1
   BNE LOOP1
   BX LR
DIV MOV Q,#0
LOOP2 SUBS DIVD,DIVD,DIVS
   ADDPL Q,Q,#1
   BPL LOOP2
   ADDMI REM,DIVD,DIVS
   BX LR
   END
```

7]. **ALP to select a set of r objects from a set of n objects considering the order of elements in an arrangement using permutation method.**

```
       AREA COMB,CODE,READONLY
DIVD RN 1
```

```
DIVS RN 2
Q RN 3
REM RN 4
N RN 9
R RN 10
NDR RN 11
ENTRY MOV N,#6
        MOV R,#3
        SUBS NDR,N,R
        MOV DIVD,N
        BL FACT
        MOV N,DIVS
        MOV DIVD,NDR
        BL FACT
        MOV NDR,DIVS
        MOV DIVS,NDR
        MOV DIVD,N
        BL DIV
STOP B STOP
FACT MOV DIVS,#1
LOOP1 MUL DIVS,DIVD,DIVS
        SUBS DIVD,DIVD,#1
         BNE LOOP1
         BX LR
DIV MOV Q,#0
LOOP2 SUBS DIVD,DIVD,DIVS
        ADDPL Q,Q,#1
        BPL LOOP2
        ADDMI R,DIVD,DIVS
        BX LR
         END
```

PART -B

1. **C program to toggle the lowest pin of Port 0 with a delay between the two**

states. Observe and record the waveform obtained using the Logic Analyzer in the Keil simulator.

```
#include<lpc214x.h>
void delay(int n)
{
int i;
for(i=0;i<n;i++);
}
int main()
{
IODIR0=0X2;
while(1)
{
IOSET0=0X2;
delay(500);
IOCLR0=0X2;
delay(500);
}
}
```

2] C program to generate a square wave using Timer0 in the interrupt mode

```
#include<LPC214x.H>

void wait()
{
    T0TCR = 0x1;
    while(T0TC != T0MR0);
}

int main() {
    IODIR0 = 0xFFFFFFFF;
    T0MR0= 0x1234;          //match register1 = terminal count
```

```
    T0MCR = 0x4;                    //match control register - b4:reset
    while(1) {

        //IOPIN0 = ~IOPIN0;
        IOSET0 = 0x1;
        wait();
        IOCLR0 = 0x1;
        wait();
    }
}
```

3] Write a C program to Interface NuMicro MCU Learning Board to Light a RGB LED connected to port A12-14.

```
#include <stdio.h>
#include "NUC1xx.h"
#include "Driver\DrvGPIO.h"
#include "Driver\DrvUART.h"
#include "Driver\DrvSYS.h"

// Initial GPIO pins (GPA 12,13,14) to Output mode
void Init_LED()
{
    // initialize GPIO pins
    DrvGPIO_Open(E_GPA, 12, E_IO_OUTPUT); // GPA12 pin set to output mode
    DrvGPIO_Open(E_GPA, 13, E_IO_OUTPUT); // GPA13 pin set to output mode
    DrvGPIO_Open(E_GPA, 14, E_IO_OUTPUT); // GPA14 pin set to output mode
    // set GPIO pins output Hi to disable LEDs
    DrvGPIO_SetBit(E_GPA, 12); // GPA12 pin output Hi to turn off Blue  LED
    DrvGPIO_SetBit(E_GPA, 13); // GPA13 pin output Hi to turn off Green LED
    DrvGPIO_SetBit(E_GPA, 14); // GPA14 pin output Hi to turn off Red   LED
```

```
}

int main (void)
{
    UNLOCKREG();                    // unlock register for programming
   DrvSYS_Open(48000000);
    LOCKREG();                              // lock register from programming

    Init_LED();

    while (1)
    {
    // GPA12 = Blue,  0 : on, 1 : off
    // GPA13 = Green, 0 : on, 1 : off
    // GPA14 = Red,   0 : on, 1 : off

    // set RGBled to Blue
   DrvGPIO_ClrBit(E_GPA,12); // GPA12 = Blue,  0 : on, 1 : off
   DrvGPIO_SetBit(E_GPA,13);
   DrvGPIO_SetBit(E_GPA,14);
    DrvSYS_Delay(1000000);

    // set RGBled to Green
   DrvGPIO_SetBit(E_GPA,12);
   DrvGPIO_ClrBit(E_GPA,13); // GPA13 = Green, 0 : on, 1 : off
   DrvGPIO_SetBit(E_GPA,14);
    DrvSYS_Delay(1000000);

    // set RGBled to Red
   DrvGPIO_SetBit(E_GPA,12);
   DrvGPIO_SetBit(E_GPA,13);
   DrvGPIO_ClrBit(E_GPA,14); // GPA14 = Red,   0 : on, 1 : off
    DrvSYS_Delay(1000000);
```

```
    // set RGBled to off
   DrvGPIO_SetBit(E_GPA,12); // GPA12 = Blue,  0 : on, 1 : off
   DrvGPIO_SetBit(E_GPA,13); // GPA13 = Green, 0 : on, 1 : off
   DrvGPIO_SetBit(E_GPA,14); // GPA14 = Red,   0 : on, 1 : off
    DrvSYS_Delay(1000000);

    }
}
```

4] Write a C program to Interface NuMicro MCU Learning Board to beep a buzzer connected to port B11.

```
#include<stdio.h>
#include "NUC1xx.h"
#include "Driver\DrvSYS.h"
#include "Driver\DrvGPIO.h"
#include "Driver\DrvADC.h"
int main(void)
{
        UNLOCKREG();
    DrvSYS_Open(48000000);
        LOCKREG();
        DrvGPIO_Open(E_GPB, 11,E_IO_OUTPUT);
        while(1)
        {
                DrvGPIO_ClrBit(E_GPB,11);
                DrvSYS_Delay(1000);
                DrvGPIO_SetBit(E_GPB,11);
                DrvSYS_Delay(100000);
        }
}
```