Course Code: ISL48 **Credit: 0:0:1**

Lab CIE Questions

Part A

1. a. ALP to add first 10 odd numbers. Store sum in register.

```
        AREA PROG2, CODE, READONLY
ENTRY
        MOV R1, #1
        MOV R0, #0
        MOV R2, #0
SUM
        ADD R0, R1
        ADD R1, R1, #2
        ADD R2, R2, #1
        CMP R2, #10
        BNE SUM
GO      B GO
        END
```

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

```
        AREA PROG3, CODE, READONLY
ENTRY
        MOV R0, #0
        MOV R1, #1
        MOV R2, #0
SUM
        BL SQU
        ADD R0, R0, R4
        ADD R1, R1, #1
        ADD R2, R2, #1
        CMP R2, #5
        BNE SUM
GO      B GO
SQU
        MUL R4, R1, R1
        MOV PC, LR
        END
```

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

```
        AREA PROG, CODE, READONLY
 ENTRY
        MOV R0, #0
        MOV R1, #0
        MOV R2, #2
```

```
          MOV R3, #0x40000000
  SUM
          ADD R0, R0, R2
          ADD R1, R1, #1
          ADD R2, R2, #2
          CMP R1, #5
          BNE SUM
          STR R0, [R3]
  GO B GO
          END
```

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

```
          AREA PROG7, CODE, READONLY
A RN 1
D RN 2
N RN 3
ENTRY
          MOV A,#1
          MOV D,#2
          MOV N,#10
          MOV R5,#0x40000000
LOOP
          MUL R6,A,D
          MOV A,R6
          STR A,[R5],#4
          SUBS N,N,#1
          BNE LOOP
  STOP B STOP
          END
```

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

```
          AREA PROG, CODE, READWRITE
NUM RN 1
NUMZERO RN 2
NUMONE RN 3
ENTRY
          MOV R5, #0x40000000
          MOV NUM, #0x000000BB
          MOV NUMZERO, #0
          MOV NUMONE, #0
LOOP
          LSRS NUM, #1
          ADDCC NUMZERO, #1
          ADDCS NUMONE, #1
          CMP NUM, #0
          BNE LOOP
  STR NUMZERO, [R5]
          STR NUMONE, [R5, #4]
GO        B GO
```

```
        END
```

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

```
        AREA PROG11,CODE,READONLY
ENTRY
        LDR R7,=TABLE
        MOV R0,#3
        LDRH R1,[R7]
BACKK
        LDRH R2,[R7,#2]!
        ADD R1,R1,R2
        SUBS R0,R0,#1
        BNE BACKK
        MOV R3,#4
        MOV R4,#0
        MOV R5,R1
BACKK1
        SUBS R5,R5,R3
        ADDPL R4,R4,#1
        BPL BACKK1
        ADDMI R5,R5,R3
GO      B GO
TABLE DCW 10,20,30,40
        END
```

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

```
        AREA PROG, CODE, READONLY
 N RN 1
 FACT RN 2
 ENTRY
        MOV N, #10
        MOV FACT, #1
 LOOP
        MUL FACT, N, FACT
        SUBS N, N, #1
        BNE LOOP
 GO B GO
        END
```

b. ALP to generate the first n Fibonacci numbers.

```
        AREA PROG, CODE, READONLY
N RN 1
ENTRY
        MOV N, #10
        LDR R5, =TABLE
        MOV R2, #0
        STRB R2, [R5], #1
        MOV R3, #1
        STRB R3, [R5], #1
        MOV R4, #2
LOOP
        ADD R6, R2, R3
        MOV R2, R3
        MOV R3, R6
        STRB R3, [R5], #1
        ADD R4, R4, #1
        CMP R4, N
        BNE LOOP
STOP    B STOP
TABLE SPACE 60
        END
```

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

```
        AREA SUM, CODE, READONLY
Q RN 3
R RN 4
RES RN 5
DV RN 1
DS RN 2
ENTRY
        MOV DV, #12
        MOV DS, #10
        MOV RES, #0
LOOP
        BL DIV
        ADD RES, R, RES
        CMP Q, #0
        MOV DV, Q
        BNE LOOP
STOP     B STOP
DIV
        MOV Q, #0
LOOP2
        SUBS DV, DV, DS
        ADDPL Q, Q, #1
        BPL LOOP2
        ADDMI R, DV, DS
        MOV PC, 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 NCR, CODE, READONLY
Q RN 3
REM RN 4
DV RN 1
DS RN 2
```

```
N RN 6
R RN 7
NUM RN 8
FACT RN 9
TEMP RN 10
ENTRY
        MOV N, #10 ;10C2
        MOV R, #2
        MOV NUM, N
        BL FACTORIAL
        MOV DV, FACT
        SUB NUM, N, R
        BL FACTORIAL
        MOV TEMP, FACT
        MOV NUM, R
        BL FACTORIAL
        MUL TEMP, FACT, TEMP
        MOV DS, TEMP
        BL DIV
STOP    B STOP
FACTORIAL
        MOV FACT, #1
LOOP1
        MUL FACT, NUM, FACT
        SUBS NUM, NUM, #1
        BNE LOOP1
        MOV PC, LR
DIV
        MOV Q, #0
LOOP2
        SUBS DV, DV, DS
        ADDPL Q, Q, #1
        BPL LOOP2
        ADDMI REM, DV, DS
        MOV PC, 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 NPR, CODE, READONLY
Q RN 3
REM RN 4
DV RN 1
DS RN 2
N RN 6
R RN 7
```

```
NUM RN 8
FACT RN 9
ENTRY
            MOV N, #10 ;10P2
            MOV R, #2
            MOV NUM, N
            BL FACTORIAL
            MOV DV, FACT
            SUB NUM, N, R
            BL FACTORIAL
            MOV DS, FACT
            BL DIV
STOP     B STOP
FACTORIAL
            MOV FACT, #1
LOOP1
            MUL FACT, NUM, FACT
            SUBS NUM, NUM, #1
            BNE LOOP1
            MOV PC, LR
DIV
            MOV Q, #0
LOOP2
            SUBS DV, DV, DS
            ADDPL Q, Q, #1
            BPL LOOP2
            ADDMI REM, DV, DS
            MOV PC, 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);
int main()
{
IODIR0 = 0x00000001;
while(1){
  IOSET0 = 0x00000001;
  delay(500);
  IOCLR0 = 0x00000001;
  delay(500);
}
}

void delay(int n)
```

```
{
int i =0;
for(i = 0;i<n;i++);
}
```

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

```
#include<LPC214x.H>

void wait(){
        T0TCR = 1;                  //timer control register bit0- enable
        while(T0TC != T0MR1);
}

int main() {
        T0MR1 = 0x1234;             //match register1 = terminal count
        T0MCR = 0x10;                       //match control register - b4:reset
        while(1) {
                IODIR0 = 0xFFFFFFFF;
                //IOPIN0 = ~IOPIN0;
                IOSET0 = 0xFFFFFFFF;
                wait();
                IOCLR0 = 0xFFFFFFFF;
                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);    // set System Clock to run at 48MHz (PLL with 12MHz crystal
input)
        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\DrvGPIO.h"
#include "Driver\DrvUART.h"

int main(void)
{
        UNLOCKREG(); //UNLOCK REGISTER FOR PROGRAMMING
        DrvSYS_Open(48000000); //set System clock to run at 48MHz
        LOCKREG(); //LOCK register from programming
        DrvGPIO_Open(E_GPB, 11, E_IO_OUTPUT); //intial GPIO pin GPB11 for controlling buzzer
        while(1){
        DrvGPIO_ClrBit(E_GPB,11); //GPB11 = 0 to turn on buzzer
        DrvSYS_Delay(100000); //delay
        DrvGPIO_SetBit(E_GPB,11); //GPB11 = 1 to turn off buzzer
        DrvSYS_Delay(100000); //delay
        }
}
```