

Name : K. S. Mahammad Liyaz

Register number: 192212149

Subject: ECA4501-Embedded C Programming

Experiment Lab Outputs:





EXP 1

Aim: To Write a 8051 C program to multiply two 16 bit binary numbers.

Program:

```
#include <reg51.h>
void main ()
{
    while(1)
    {
        unsigned int num1, num2;
        unsigned long int product;
        num1=0x2222;
        num2=0xB BBB;
        product=(unsigned long int)num1*num2;
    }
}
```

Output:

Call Stack + Locals			
Name	Location/Value	Type	
 MAIN	C:0x082D		
 num1	0x2222	uint	
 num2	0xB BBB	uint	
 product	0x1907C4D6	ulong	

EXP 2

Aim: To Write a 8051 C program to find the sum of first 10 integer numbers.

Program:

```
#include <reg51.h>

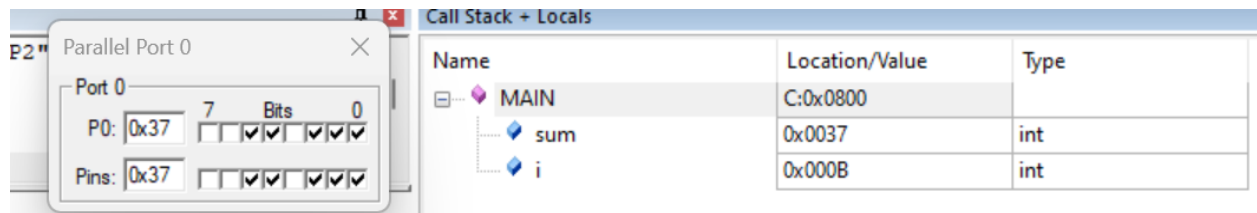
void main() {
    int sum = 0;
    int i;

    for(i = 1; i <= 10; i++) {
        sum += i;
    }

    P0=sum;

    while(1);
}
```

Output:



The screenshot displays two windows from a Keil IDE. On the left, the 'Parallel Port 0' window shows the P0 port value as 0x37 and the Pins value as 0x37, with a 7-bit display. On the right, the 'Call Stack + Locals' window shows the call stack with 'MAIN' at location C:0x0800. Below it, the local variables 'sum' and 'i' are listed with their respective memory addresses and types.

Name	Location/Value	Type
MAIN	C:0x0800	
sum	0x0037	int
i	0x000B	int

EXP 3

Aim: To write a 8051 C program to find factorials of a given number.

Program:

```
#include <reg51.h>
#include <stdio.h>

void main() {
    unsigned char num = 12;
    unsigned long factorial = 1;
    unsigned int i;

    for (i = 1; i <= num; i++) {
        factorial *= i;
    }

    P0=factorial;
    P1=(factorial & 0xff00)>>8;
    P2=(factorial & 0xff0000)>>16;
    P3=(factorial & 0xff000000)>>24;

    while (1);
}
```

Output:

The screenshot displays a 8051 microcontroller simulator interface. At the top, four windows show the status of Parallel Ports 0, 1, 2, and 3. Each window includes a 7-bit data bus (P0-P3) and an 8-bit pin status (Pins). The values shown are: P0: 0x00, P1: 0xFC, P2: 0x8C, and P3: 0x1C. Below these, a 'Call Stack + Locals' window is visible, showing the current state of the program. The stack frame for 'MAIN' is active, with the following variables and their values:

Name	Location/Value	Type
MAIN	C:0x0800	
num	0x0C	uchar
factorial	0x1C8CFC00	ulong
i	0x000D	uint

EXP 4

Aim: To write an 8051 Program to add an array of 16 bit numbers and store the 32 bit result in internal RAM.

Program:

```
#include <reg51.h>
#define ARRAY_SIZE 5

code unsigned int numbers[ARRAY_SIZE] = {1000, 2000, 3000, 4000, 5000};

unsigned long result; // 32-bit result




void main() {
    unsigned int i;
    unsigned long sum = 0;

    for (i = 0; i < ARRAY_SIZE; i++) {
        sum += numbers[i];
    }

    result = sum;

    while (1);
}
```

Output:

Call Stack + Locals		
Name	Location/Value	Type
 MAIN	C:0x0800	
└─  i	0x0005	uint
└─  sum	0x00003A98	ulong

EXP 5

Aim: To write a 8051 C program to find the square of a number (1 to 10) using look-up table.

Program:

```
#include <reg51.h>
void main() {
    unsigned char LUT[]={1,4,9,16,25,36,49,64,81,100};
    unsigned char num, square;
    for(num=1; num<11; num++)
    {
        square =LUT[num-1];
        P0=square;
    }
}
```

Output:

The screenshot displays the Keil uVision IDE interface. The 'Call Stack + Locals' window is open, showing the call stack with 'MAIN' at the top and 'LUT' below it. The 'LUT' array is expanded, showing its elements: [0] (0x01), [1] (0x04), [2] (0x09), [3] (0x10), [4] (0x19), [5] (0x24 'S'), [6] (0x31 '1'), [7] (0x40 '@'), [8] (0x51 'Q'), [9] (0x64 'd'), 'num' (0x0B), and 'square' (0x64 'd'). The 'Parallel Port 0' window is also open, showing the output of the program. The 'P0' register is set to 0x64, and the 'Pins' are set to 0x00. The 'Bits' field shows the output of the program, with the 7th bit (index 6) being 1, which corresponds to the value 49 in the LUT array.

Name	Location/Value	Type
MAIN	C:0x08F6	
LUT	D:0x08 "0x08\t0x08\$1@...	array[10] of uchar
[0]	0x01	uchar
[1]	0x04	uchar
[2]	0x09	uchar
[3]	0x10	uchar
[4]	0x19	uchar
[5]	0x24 'S'	uchar
[6]	0x31 '1'	uchar
[7]	0x40 '@'	uchar
[8]	0x51 'Q'	uchar
[9]	0x64 'd'	uchar
num	0x0B	uchar
square	0x64 'd'	uchar

Parallel Port 0

Port 0: 0x64 7 Bits 0

Pins: 0x00

EXP 6

Aim: To write a 8051 C Program to find the Largest and Smallest numbers in an array of numbers.

Program:

```
#include <reg51.h>

void main() {
    unsigned char numbers[] = {45, 34, 12, 56, 78, 23};
    unsigned char largest = numbers[0];
    unsigned char smallest = numbers[0];
    unsigned char i;

    for (i = 1; i < sizeof(numbers); i++) {
        if (numbers[i] > largest) {
            largest = numbers[i];
        }
        if (numbers[i] < smallest) {
            smallest = numbers[i];
        }
    }

    P0 = largest;
    P1 = smallest;

    while (1);
}
```

Output:

The screenshot displays the 'Call Stack + Locals' window in a Keil IDE, showing the state of the program's memory. The 'MAIN' function is active, and the 'numbers' array is visible with its elements: 45, 34, 12, 56, 78, 23. The 'largest' variable is 78 (0x4E) and the 'smallest' variable is 12 (0x0C). The 'i' variable is 0 (0x06). To the right, two windows show the output of the program: 'Parallel Port 0' displays 0x4E (78) and 'Parallel Port 1' displays 0x0C (12).

Name	Location/Value	Type
MAIN	C:0x08F6	
numbers	D:0x08 " - "\f8N□"	array[6] of uchar
[0]	0x2D '-'	uchar
[1]	0x22 ""	uchar
[2]	0x0C	uchar
[3]	0x38 '8'	uchar
[4]	0x4E 'N'	uchar
[5]	0x17	uchar
largest	0x4E 'N'	uchar
smallest	0x0C	uchar
i	0x06	uchar

Parallel Port 0

Port 0

P0: 0x4E 7 Bits 0

Pins: 0x4E

Parallel Port 1

Port 1

P1: 0x0C 7 Bits 0

Pins: 0x0C

EXP 7

Aim: To write a 8051 C Program to arrange a series of numbers in ascending and descending order locations.

Program:

(a) Ascending Order

```
#include<reg51.h>
void main()
{
    unsigned long array[]={0x33556666, 0xCCAADD00, 0x55998888, 0x77664444,
    0x11223344};
    unsigned long temp, i, j;

    for(i=0; i<5; j++)
    {
        for(j=0; j<5; i++)
        {
            if(array[j]>array[j+1])
            {
                temp=array[j+1];
                array[j+1]=array[j];
                array[j]=temp;
            }
        }
    }
}
```

(b) Descending Order

```
#include <reg51.h>

void main() {
    unsigned long array[] = {0x33556666, 0xCCAADD00, 0x55998888, 0x77664444,
    0x11223344};
    unsigned long temp;
    unsigned int i, j;

    for (i = 0; i < 4; i++) {
        for (j = 0; j < 4 - i; j++) {
```

```

        if (array[j] < array[j + 1]) {
            temp = array[j];
            array[j] = array[j + 1];
            array[j + 1] = temp;
        }
    }
}

while (1);
}

```

Output:

(a) Ascending Order:

Before Execution

Call Stack + Locals		
Name	Location/Value	Type
MAIN	C:0x092C	
array	D:0x08	array[5] of ulong
[0]	0x33556666	ulong
[1]	0xCCAADD00	ulong
[2]	0x55998888	ulong
[3]	0x77664444	ulong
[4]	0x11223344	ulong
temp	0x00000000	ulong
i	0x0000	uint
j	0x09E4	uint

After Execution

Call Stack + Locals		
Name	Location/Value	Type
MAIN	C:0x092C	
array	D:0x08	array[5] of ulong
[0]	0x11223344	ulong
[1]	0x33556666	ulong
[2]	0x55998888	ulong
[3]	0x77664444	ulong
[4]	0xCCAADD00	ulong
temp	0x33556666	ulong
i	0x0004	uint
j	0x0001	uint

(b) Descending order:

Before Execution

Call Stack + Locals		
Name	Location/Value	Type
MAIN	C:0x092C	
array	D:0x08	array[5] of ulong
[0]	0x33556666	ulong
[1]	0xCCAADD00	ulong
[2]	0x55998888	ulong
[3]	0x77664444	ulong
[4]	0x11223344	ulong
temp	0x00000000	ulong
i	0x0000	uint
j	0x09E4	uint

After Execution

Call Stack + Locals			x
Name	Location/Value	Type	
MAIN	C:0x092C		
array	D:0x08	array[5] of ulong	
[0]	0xCCAADD00	ulong	
[1]	0x77664444	ulong	
[2]	0x55998888	ulong	
[3]	0x33556666	ulong	
[4]	0x11223344	ulong	
temp	0x55998888	ulong	
i	0x0004	uint	
j	0x0001	uint	

EXP 8

Aim: To write a 8051 C program to count the number of ones and zeros in two consecutive in memory locations.

Program:

```
#include<reg51.h>
void main()
{
    unsigned char array[]={0x57, 0xfc};
    unsigned char i, ones, zeros;
    CY=0;

    for(i=0; i<8; i++)
    {
        array[0]>>=1;
        if(CY==1)ones++;
        else zeros++;
    }
    for(i=0; i<8; i++)
    {
        array[1]>>=1;
        if(CY==1)ones++;
        else zeros++;
    }
    P0=zeros;
    P1=ones;
    while(1);
}
```

Output:

The screenshot displays the Keil IDE interface. On the left, the 'Call Stack + Locals' window shows the 'MAIN' function with local variables: 'array' (array[2] of uchar), 'i' (uchar), 'ones' (uchar), and 'zeros' (uchar). The 'array' variable is expanded, showing 'array[0]' with value 0x00 and 'array[1]' with value 0x00. The 'i' variable has a value of 0x08, 'ones' has 0x0B, and 'zeros' has 0x05. On the right, there are two 'Parallel Port' windows. 'Parallel Port 0' shows 'Port 0' with value 0x05 and 'Pins' with value 0x05. 'Parallel Port 1' shows 'Port 1' with value 0x0B and 'Pins' with value 0x0B. Both windows have a bit field display showing bits 7 down to 0, with bits 0 and 1 checked in both.

Name	Location/Value	Type
MAIN	C:0x08F6	
array	D:0x08 ""	array[2] of uchar
array[0]	0x00	uchar
array[1]	0x00	uchar
i	0x08	uchar
ones	0x0B	uchar
zeros	0x05	uchar

Parallel Port 0

Port 0

P0: 0x05 7 Bits 0

Pins: 0x05

Parallel Port 1

Port 1

P1: 0x0B 7 Bits 0

Pins: 0x0B

EXP 9

Aim: To write a 8051 C program to scan a series of numbers to find how many are negative.

Program:

```
#include<reg51.h>
void main()
{
    unsigned long temp,
    array[]={0xff223344,0xaa336699,0x11223344,0x33445566,0x88aa3311};
    unsigned char i, pos, neg;
    CY=0;
    for(i=0; i<5; i++)
    {
        temp = array[i]<< 1;
        if(CY==1)neg++;
        else pos++;
        CY=0;
    }
    P0=neg;
    P1=pos;
    while(1);
}
```

Output:

The screenshot shows a debugger window titled "Call Stack + Locals". The left pane displays the call stack with the following entries:

Name	Location/Value	Type
MAIN	C:0x0915	
temp	0x11546622	ulong
array	D:0x0C	array[5] of ulong
array[0]	0xFF223344	ulong
array[1]	0xAA336699	ulong
array[2]	0x11223344	ulong
array[3]	0x33445566	ulong
array[4]	0x88AA3311	ulong
i	0x05	uchar
pos	0x02	uchar
neg	0x03	uchar

The right pane shows the Parallel Port I/O status for Port 0 and Port 1. Port 0 has P0: 0x03 and Pins: 0x03. Port 1 has P1: 0x02 and Pins: 0x02.

EXP 10

Aim: To write a 8051 C program to display a “Hello World” message in the UART serial window.

Program:

```
#include <reg51.h>
#include <stdio.h>

void main(void)
{
    SCON = 0x50;
    TMOD = 0x20;
    TH1 = 0xFD;
    TR1 = 1;
    TI = 1;
    while(1)
    {
        printf("Hello World !\n");
    }
}
```

Output:

A screenshot of a UART serial window titled "UART #1". The window has a blue header bar with a red close button in the top right corner. The main area is white and displays the text "Hello World !" repeated 15 times, one line per iteration. The text is in a monospaced font. A vertical scrollbar is visible on the right side of the window, and a horizontal scrollbar is at the bottom.

EXP 11

Aim: To write a 8051 C program to convert the hexadecimal data 0xFF to decimal and display the digits on ports P0, P1 and P2 (port window in simulator).

Program:

```
#include <reg51.h>
void main (void)
{
    unsigned char hexa=0xFF;
    unsigned char hundreds, tens, units;

    hexa=hexa/10;
    P0=B;
    units=B;
    hexa = hexa/10;
    hundreds=ACC;
    tens=B;
    P1=B;
    P2=ACC;
    while(1);
}
```

Output:

The screenshot displays the 'Call Stack + Locals' window of a 8051 simulator. The 'MAIN' function is active at memory location C:0x0800. The local variables are listed as follows:

Name	Location/Value	Type
hexa	0xFF 'y'	uchar
hundreds	0x02	uchar
tens	0x05	uchar
units	0x05	uchar

Below the call stack, three parallel port windows are visible:

- Parallel Port 0:** Port 0 value is 0x05, Pins are 0x05. The 7-bit display shows 05.
- Parallel Port 1:** Port 1 value is 0x05, Pins are 0x05. The 7-bit display shows 05.
- Parallel Port 2:** Port 2 value is 0x02, Pins are 0x02. The 7-bit display shows 02.