Name : K. S. Mahammad Liyaz
Register number: 192212149
Subject: ECA4501-Embedded C Programming

Experiment Lab Outputs:

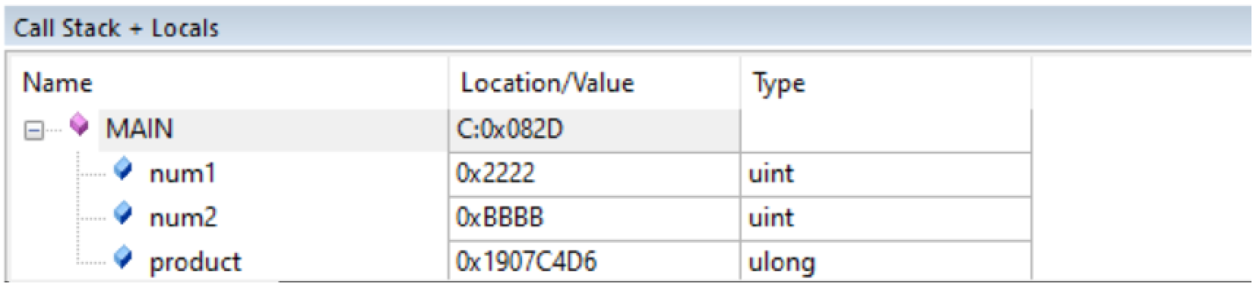
EXP 1

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=0xBBBB;
        product=(unsigned long int)num1*num2;
        }
}
```

Output:



Call Stack + Locals

Name	Location/Value	Type
MAIN	C:0x082D	
num1	0x2222	uint
num2	0xBBBB	uint
product	0x1907C4D6	ulong

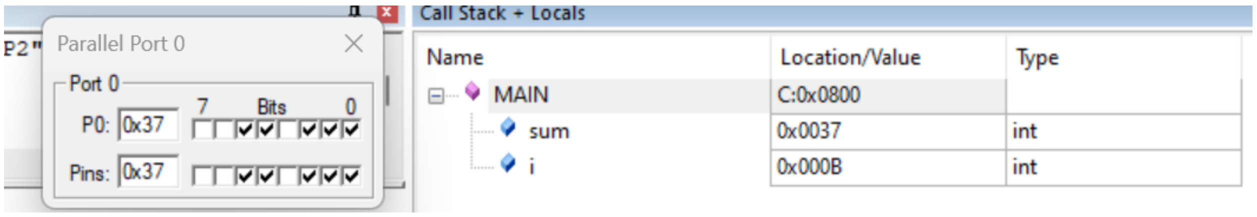
EXP 2

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:



```
P0: 0x37
Pins: 0x37
```

Call Stack + Locals

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

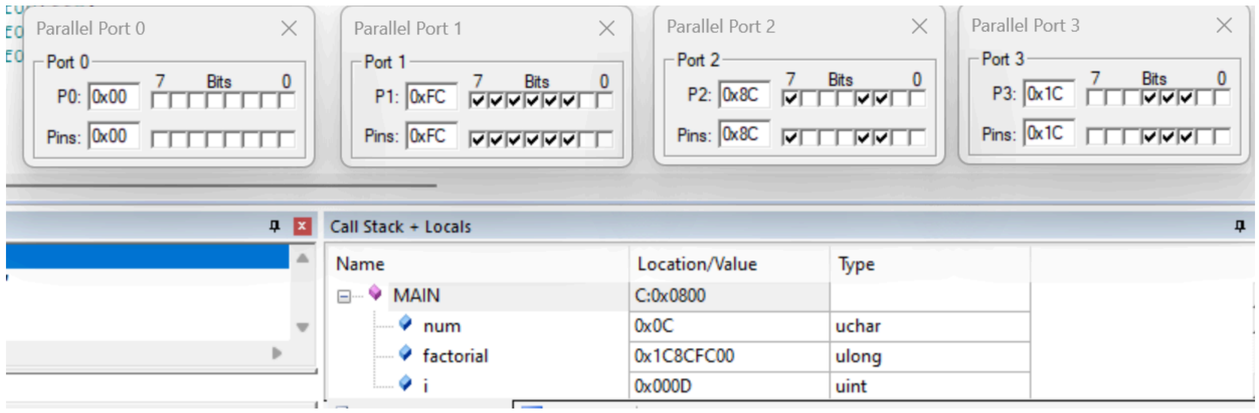
EXP 3

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:



```
Parallel Port 0          Parallel Port 1          Parallel Port 2          Parallel Port 3
Port 0                   Port 1                   Port 2                   Port 3
P0:   0x00               P1:   0xFC               P2:   0x8C               P3:   0x1C
Pins: 0x00               Pins: 0xFC               Pins: 0x8C               Pins: 0x1C
```

Call Stack + Locals

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

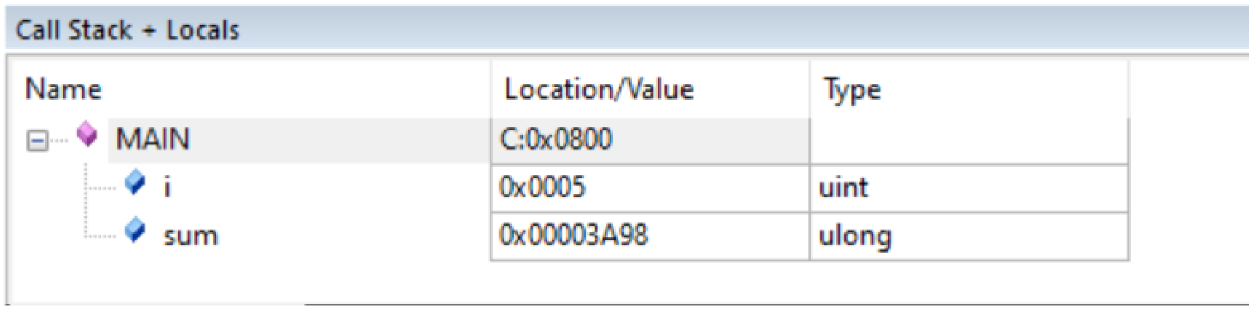
EXP 4

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

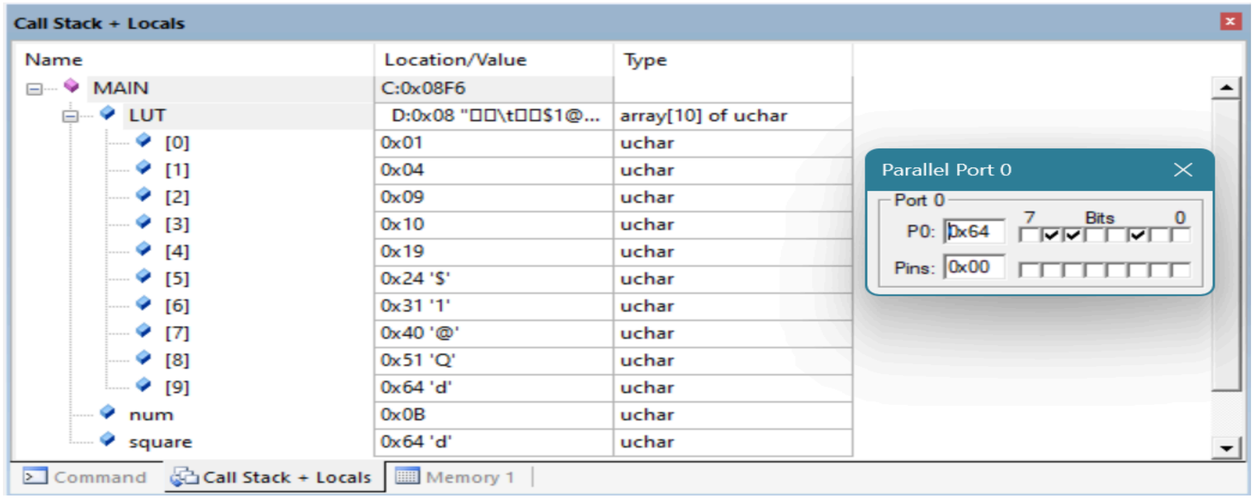
EXP 5

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:



Call Stack + Locals

Name	Location/Value	Type
MAIN	C:0x08F6	
LUT	D:0x08 "□□\t□□$1@...	array[10] of uchar
[0]	0x01	uchar
[1]	0x04	uchar
[2]	0x09	uchar
[3]	0x10	uchar
[4]	0x19	uchar
[5]	0x24 '$'	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
        7   Bits   0
P0:   0x64   ☐✓✓☐☐✓☐☐
Pins: 0x00   ☐☐☐☐☐☐☐☐
```

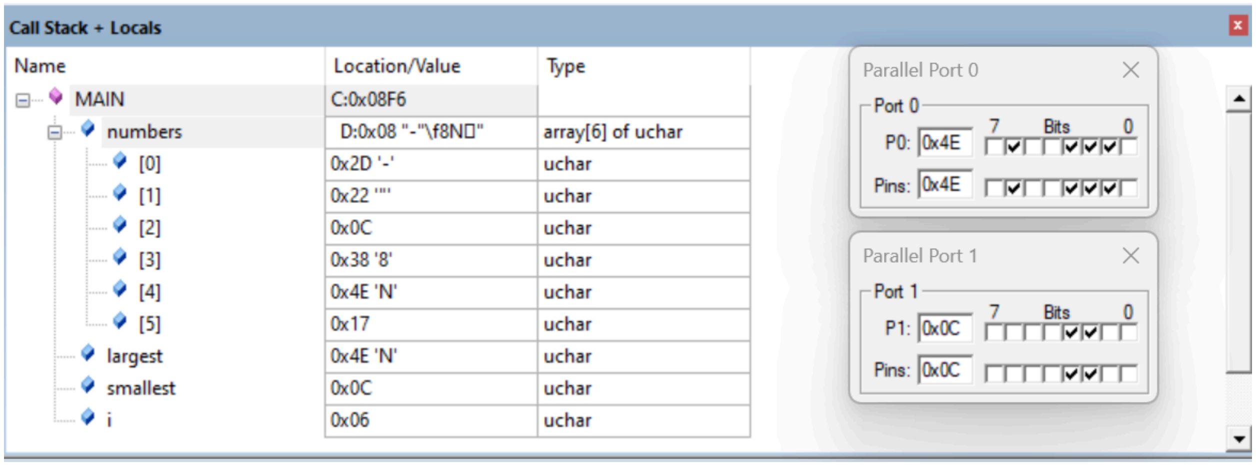
EXP 6

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:



Call Stack + Locals

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

		7							0
P0:	0x4E	☐	✓	☐	☐	✓	✓	✓	☐
Pins:	0x4E	☐	✓	☐	☐	✓	✓	✓	☐

Parallel Port 1

Port 1

		7							0
P1:	0x0C	☐	☐	☐	☐	✓	✓	☐	☐
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

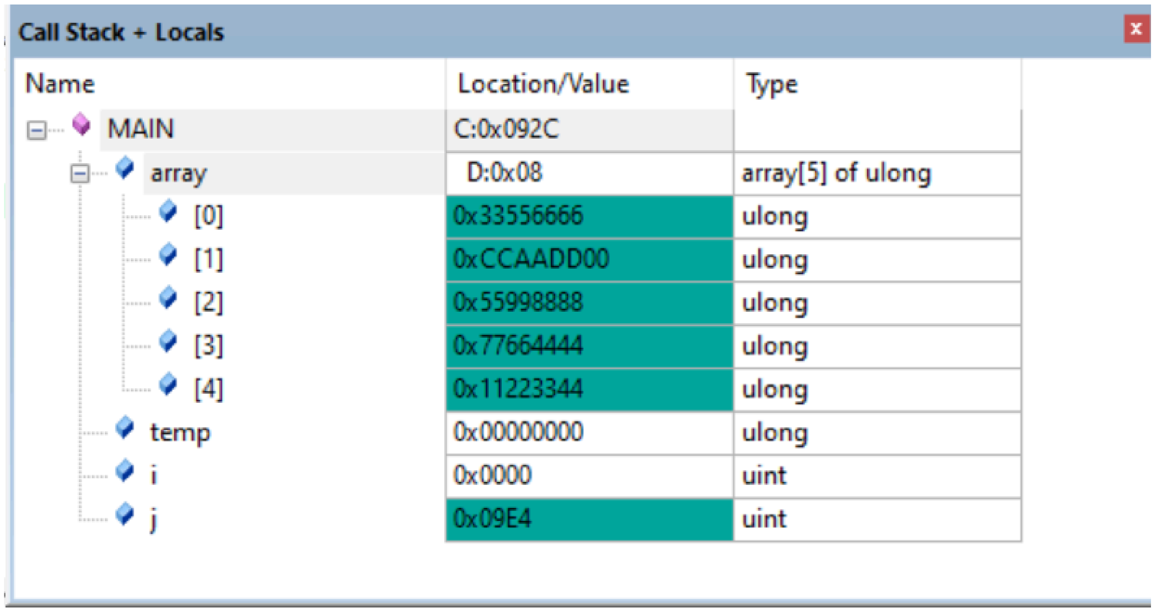
```
#include <reg51.h>

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:

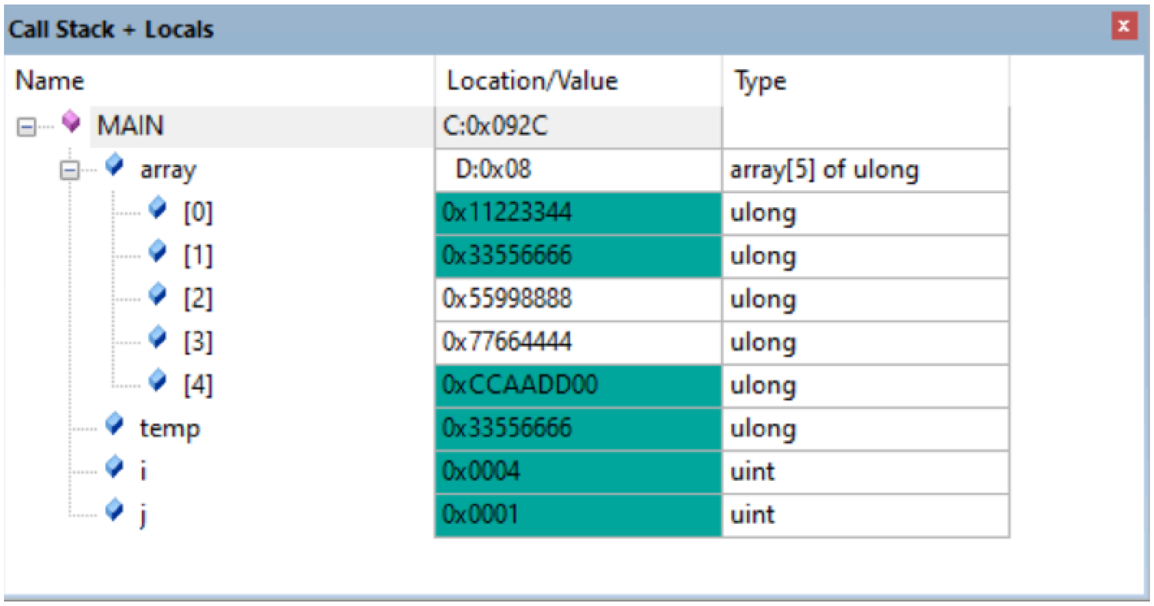
Before Execution



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

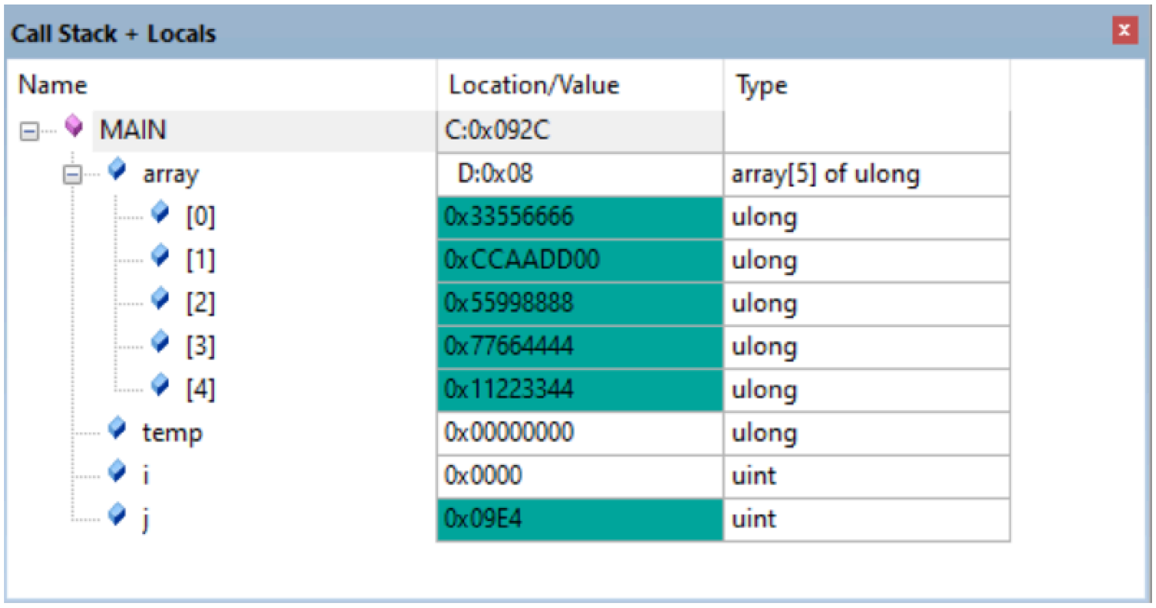


Call Stack + Locals

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) <u>Descending order</u>:

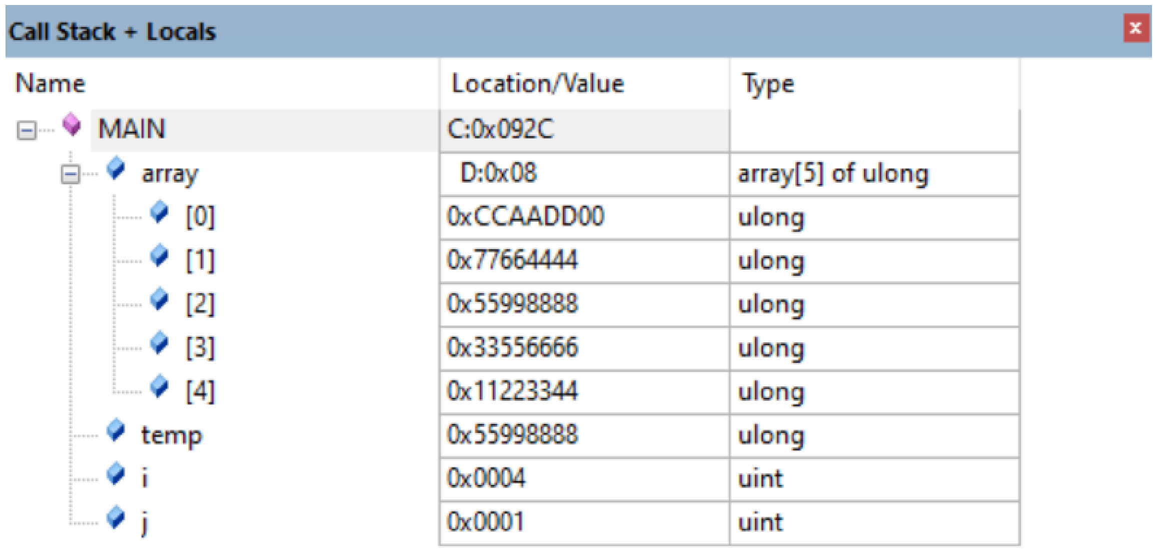
Before Execution



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



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

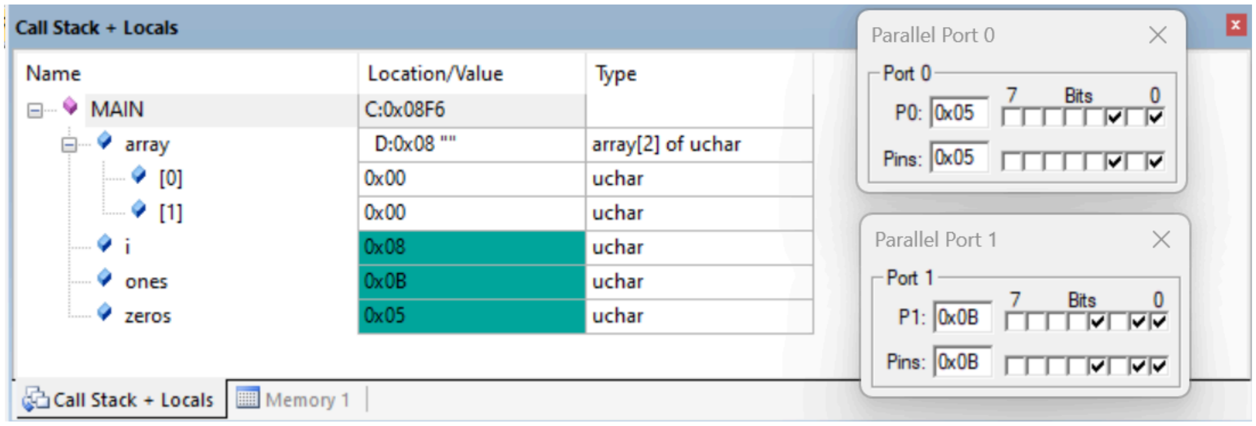
EXP 8

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:



Call Stack + Locals

Name	Location/Value	Type
MAIN	C:0x08F6	
array	D:0x08 ""	array[2] of uchar
[0]	0x00	uchar
[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

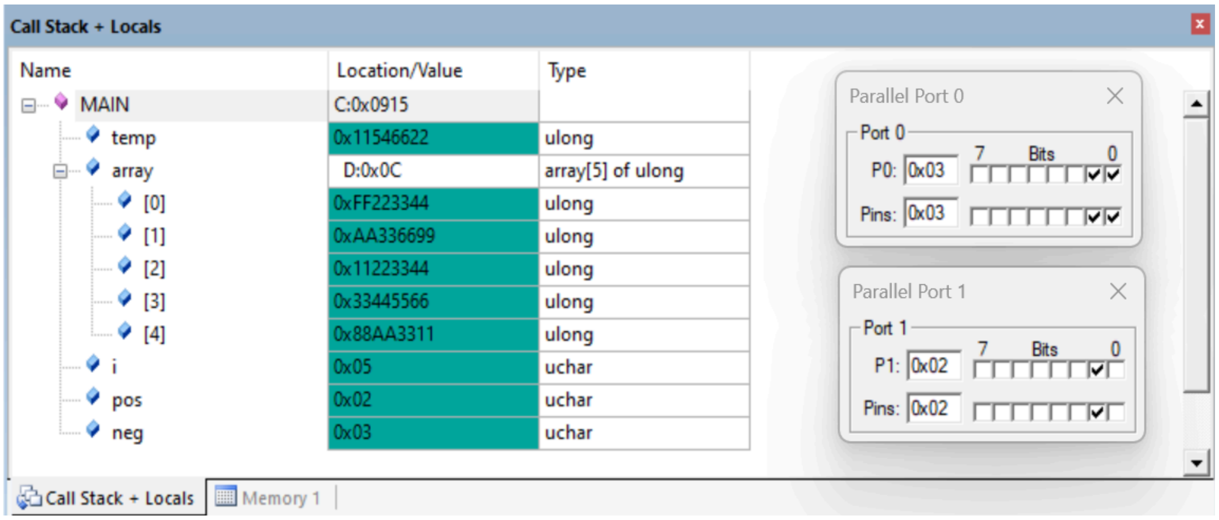
EXP 9

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:



Call Stack + Locals

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

Parallel Port 0
Port 0
P0: 0x03
Pins: 0x03

Parallel Port 1
Port 1
P1: 0x02
Pins: 0x02

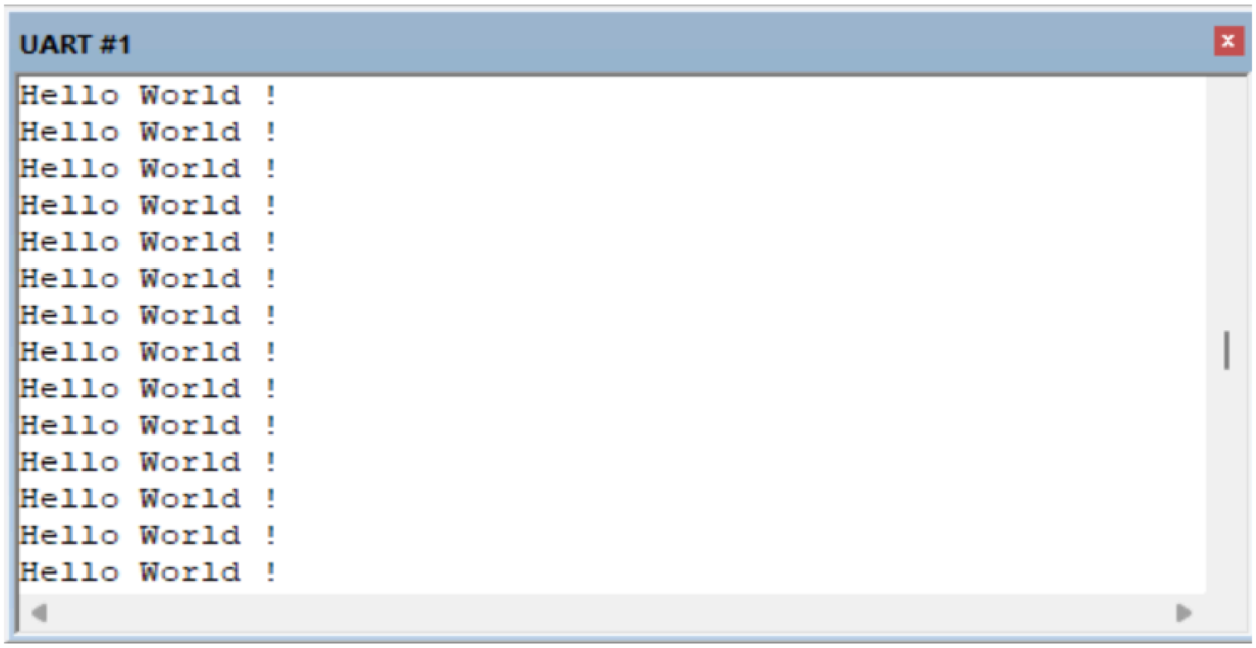
EXP 10

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:



```
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
Hello World !
```

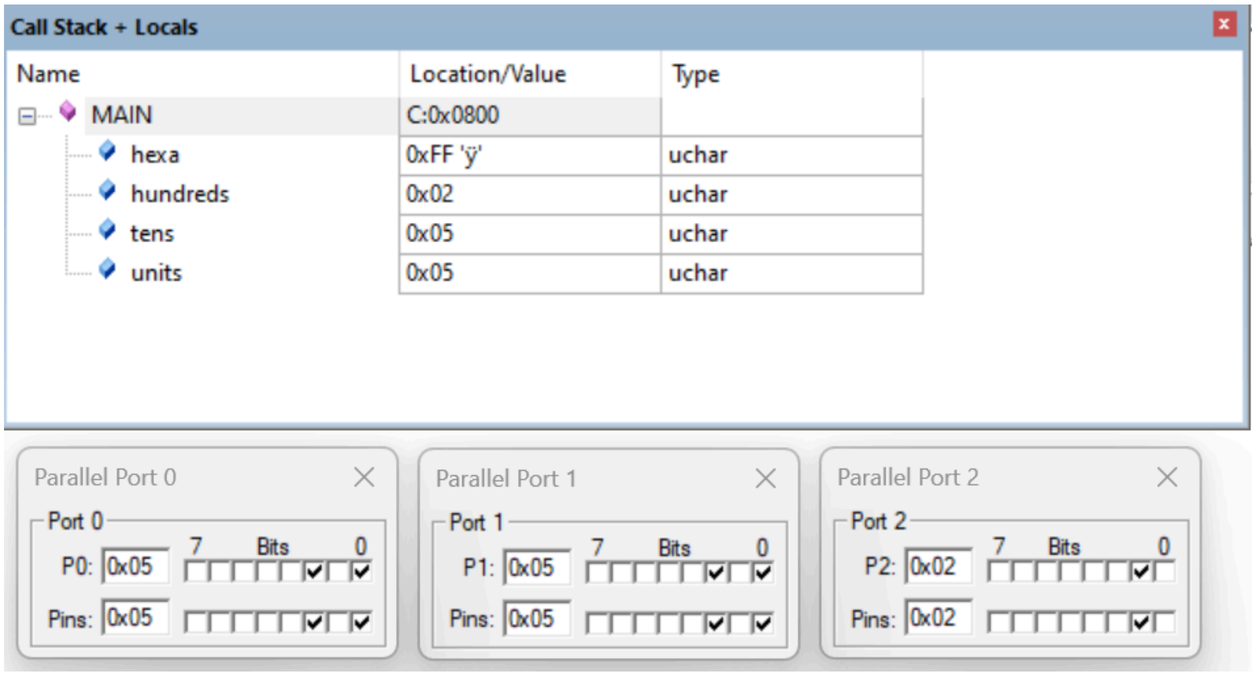
EXP 11

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:



Call Stack + Locals

Name	Location/Value	Type
MAIN	C:0x0800	
hexa	0xFF 'ÿ'	uchar
hundreds	0x02	uchar
tens	0x05	uchar
units	0x05	uchar

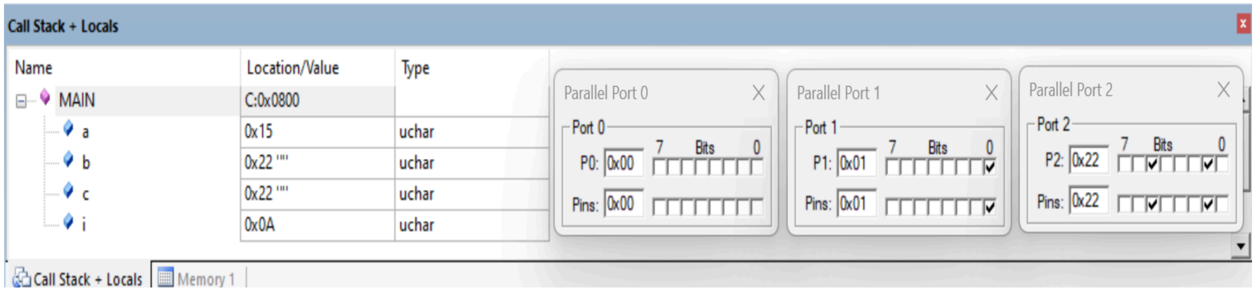
EXP 12

Aim: To Write a 8051 C program to generate and print the first 10 numbers in the Fibonacci sequence.

Program:

```
#include <reg51.h>
void delay()
        {
  unsigned int i;
  for (i = 0; i < 30000; i++);
        }
void main()
        {
  unsigned char a = 0, b = 1, c;
  unsigned char i;

  P0 = a;
  delay();
  P1 = b;
  delay();

  for (i = 2; i < 10; i++)
              {
    c = a + b;
    P2 = c;
    delay();

    a = b;
    b = c;
  }

  while (1);
}
```

Output:


Call Stack + Locals

Name	Location/Value	Type
MAIN	C:0x0800	
a	0x15	uchar
b	0x22 '"'	uchar
c	0x22 '"'	uchar
i	0x0A	uchar

```
Parallel Port 0
Port 0          7   Bits   0
P0:   0x00
Pins: 0x00

Parallel Port 1
Port 1          7   Bits   0
P1:   0x01
Pins: 0x01

Parallel Port 2
Port 2          7   Bits   0
P2:   0x22
Pins: 0x22
```

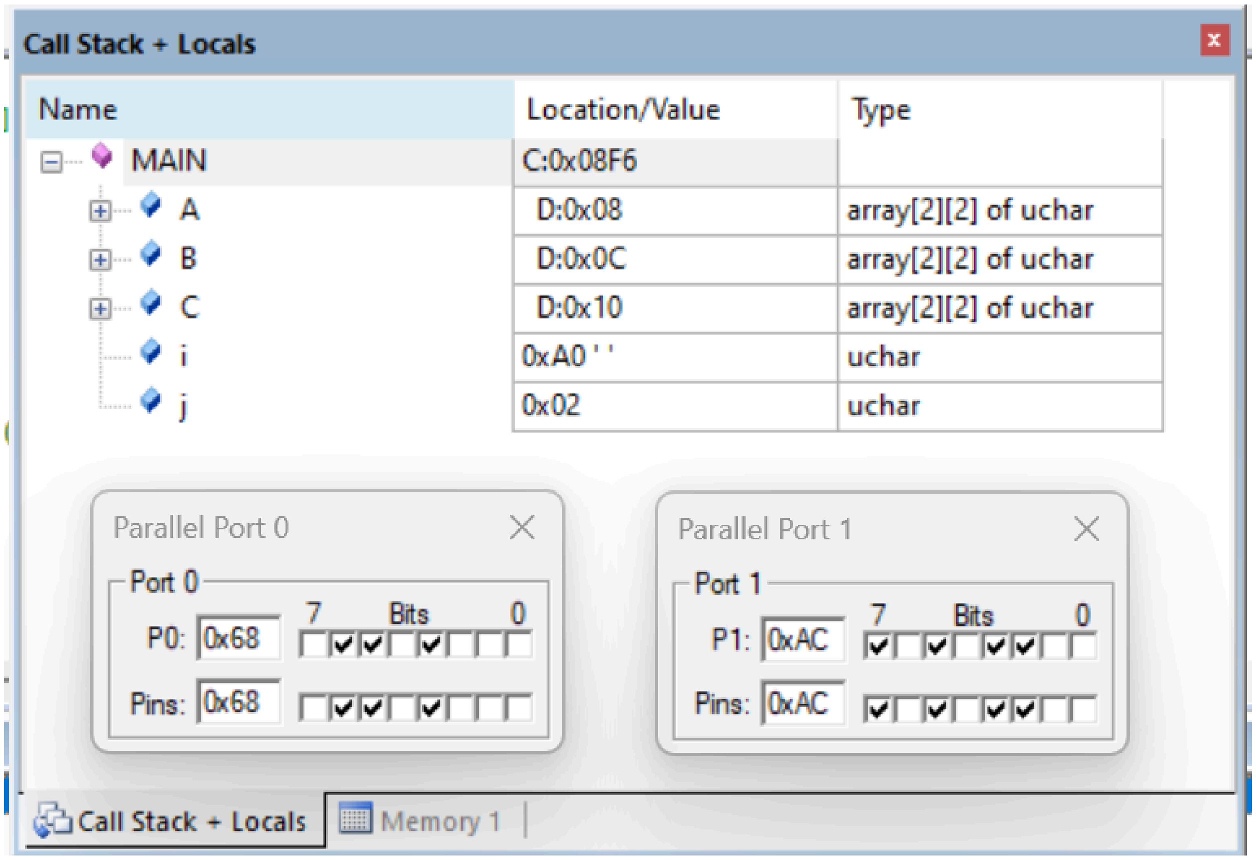
EXP 13

Aim: To Write a 8051 C program to perform matrix addition of two 2×2 matrices.

Program:

```
#include <reg51.h>

void main() {
   unsigned char A[2][2] = {{1, 2}, {3, 4}};
   unsigned char B[2][2] = {{5, 6}, {7, 8}};
   unsigned char C[2][2];
   unsigned char i, j;


   for (i = 0; i < 2; i++) {
      for (j = 0; j < 2; j++) {
         C[i][j] = A[i][j] + B[i][j];
      }
   }


   P0 = (C[0][0] << 4) | (C[0][1] & 0x0F);
   P1 = (C[1][0] << 4) | (C[1][1] & 0x0F);
   while (1);
}
```

Output:



Name	Location/Value	Type
MAIN	C:0x08F6	
A	D:0x08	array[2][2] of uchar
B	D:0x0C	array[2][2] of uchar
C	D:0x10	array[2][2] of uchar
i	0xA0 ' '	uchar
j	0x02	uchar

```
P0: 0x68    Bits 7..0: ☐ ✓ ✓ ☐ ✓ ☐ ☐ ☐
Pins: 0x68  Bits 7..0: ☐ ✓ ✓ ☐ ✓ ☐ ☐ ☐

P1: 0xAC    Bits 7..0: ✓ ☐ ✓ ☐ ✓ ✓ ☐ ☐
Pins: 0xAC  Bits 7..0: ✓ ☐ ✓ ☐ ✓ ✓ ☐ ☐
```

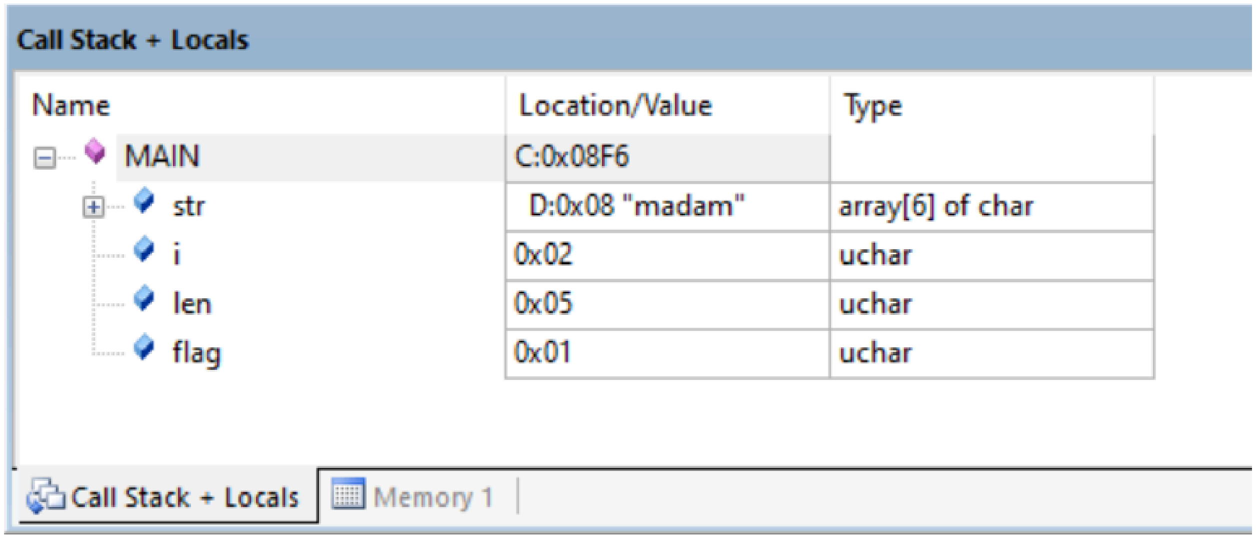
<u>EXP 14</u>

Aim: To Write a C program to check if a given string is a palindrome.

Program:

```
#include <reg51.h>
void main() {
 char str[] = "madam";
 unsigned char i, len = 0, flag = 1;
 while (str[len] != '\0') {
 len++;
 }
 for (i = 0; i < len / 2; i++) {
 if (str[i] != str[len - 1 - i]) {
 flag = 0;
 break;
 }
 }
while (1);
}
```

Output:



Call Stack + Locals

Name	Location/Value	Type
MAIN	C:0x08F6	
str	D:0x08 "madam"	array[6] of char
i	0x02	uchar
len	0x05	uchar
flag	0x01	uchar

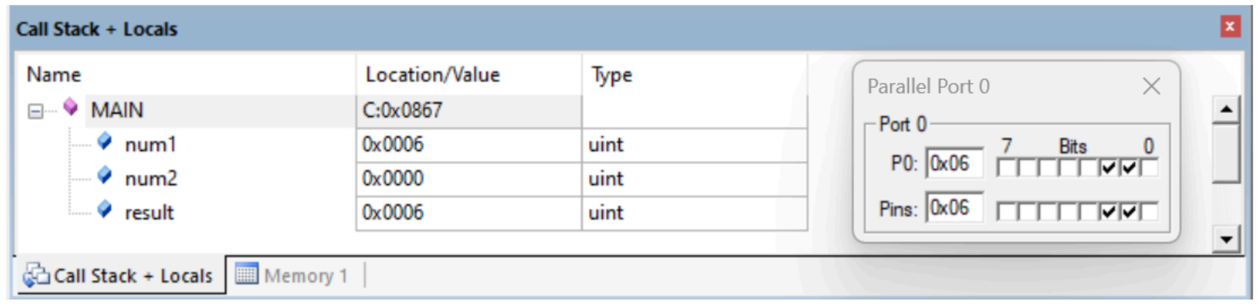
EXP 15

Aim: To Write a 8051 C program to calculate the greatest common divisor (GCD) of two integers.

Program:

```
#include <reg51.h>
unsigned int gcd(unsigned int a, unsigned int b) {
    while (b != 0) {
        unsigned int temp = b;
        b = a % b;
        a = temp;
    }
    return a;
}

void main() {
    unsigned int num1 = 48, num2 = 18;
    unsigned int result;

    result = gcd(num1, num2);
    P0 = result;

    while (1);
}
```

Output:



Call Stack + Locals

```
Name            Location/Value   Type
MAIN            C:0x0867
  num1          0x0006           uint
  num2          0x0000           uint
  result        0x0006           uint
```

Parallel Port 0

Port 0

P0: 0x06

Pins: 0x06

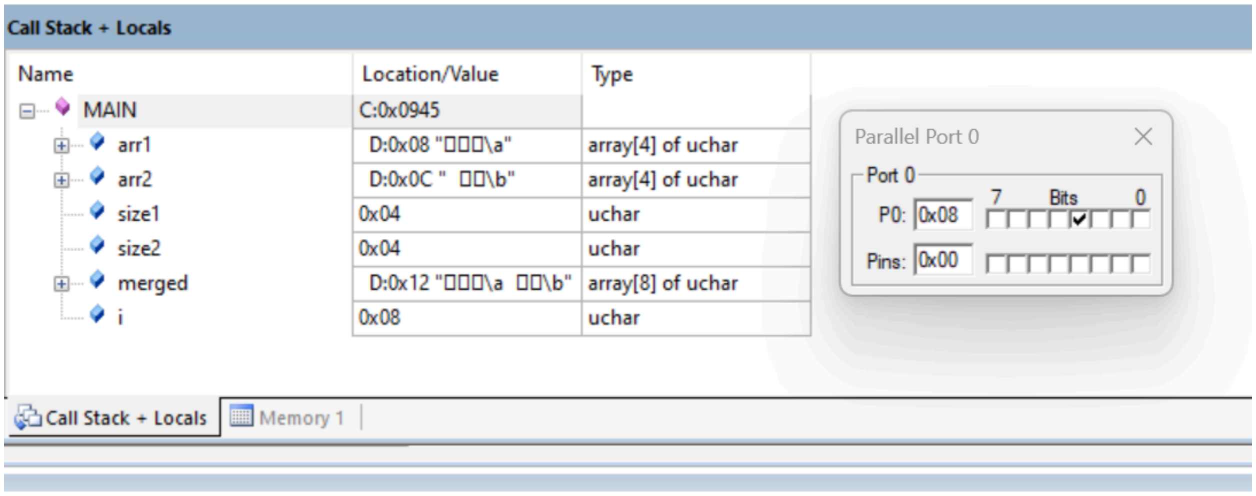
EXP 16

Aim: To Write a 8051 C program to calculate the greatest common divisor (GCD) of two integers.

Program:

```
#include <reg51.h>

void merge_arrays(unsigned char arr1[], unsigned char size1, unsigned char
arr2[], unsigned char size2, unsigned char merged[]) {
    unsigned char i, j;


    for (i = 0; i < size1; i++) {
        merged[i] = arr1[i];
    }


    for (j = 0; j < size2; j++) {
        merged[size1 + j] = arr2[j];
    }
}

void main() {
    unsigned char arr1[] = {1, 3, 5, 7};
    unsigned char arr2[] = {2, 4, 6, 8};
    unsigned char size1 = sizeof(arr1) / sizeof(arr1[0]);
    unsigned char size2 = sizeof(arr2) / sizeof(arr2[0]);
    unsigned char merged[8];
    unsigned char i;


    merge_arrays(arr1, size1, arr2, size2, merged);


    for (i = 0; i < (size1 + size2); i++) {
        P0 = merged[i];
    }

    while (1);
}
```

Output:



Call Stack + Locals

Name	Location/Value	Type
MAIN	C:0x0945	
arr1	D:0x08 "□□□\a"	array[4] of uchar
arr2	D:0x0C " □□\b"	array[4] of uchar
size1	0x04	uchar
size2	0x04	uchar
merged	D:0x12 "□□□\a □□\b"	array[8] of uchar
i	0x08	uchar

Parallel Port 0

Port 0

P0: 0x08

Pins: 0x00

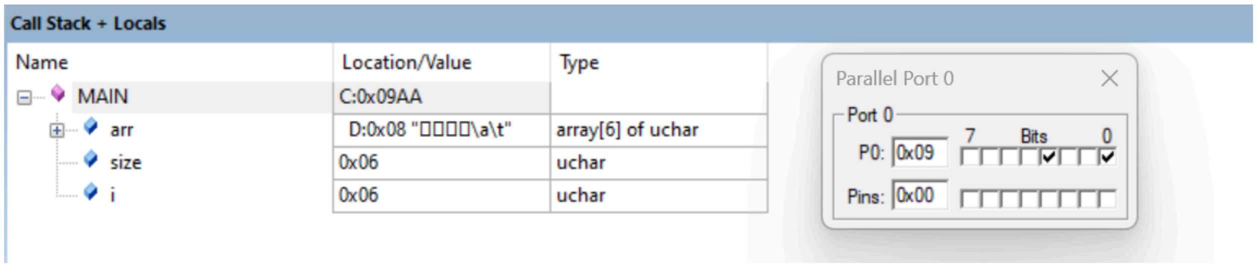
EXP 17

Aim: To Write a 8051 C program to sort an array of integers in ascending order.

Program:

```
#include <reg51.h>

void bubble_sort(unsigned char arr[], unsigned char size) {
    unsigned char i, j, temp;
    for (i = 0; i < size - 1; i++) {
        for (j = 0; j < size - 1 - i; j++) {
            if (arr[j] > arr[j + 1]) {
                // Swap elements
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

void main() {
    unsigned char arr[] = {9, 4, 7, 1, 3, 6};
    unsigned char size = sizeof(arr) / sizeof(arr[0]);
    unsigned char i;

    bubble_sort(arr, size);


    for (i = 0; i < size; i++) {
        P0 = arr[i];
    }

    while (1);
}
```

Output:



Call Stack + Locals

Name	Location/Value	Type
MAIN	C:0x09AA	
arr	D:0x08 "□□□□\a\t"	array[6] of uchar
size	0x06	uchar
i	0x06	uchar

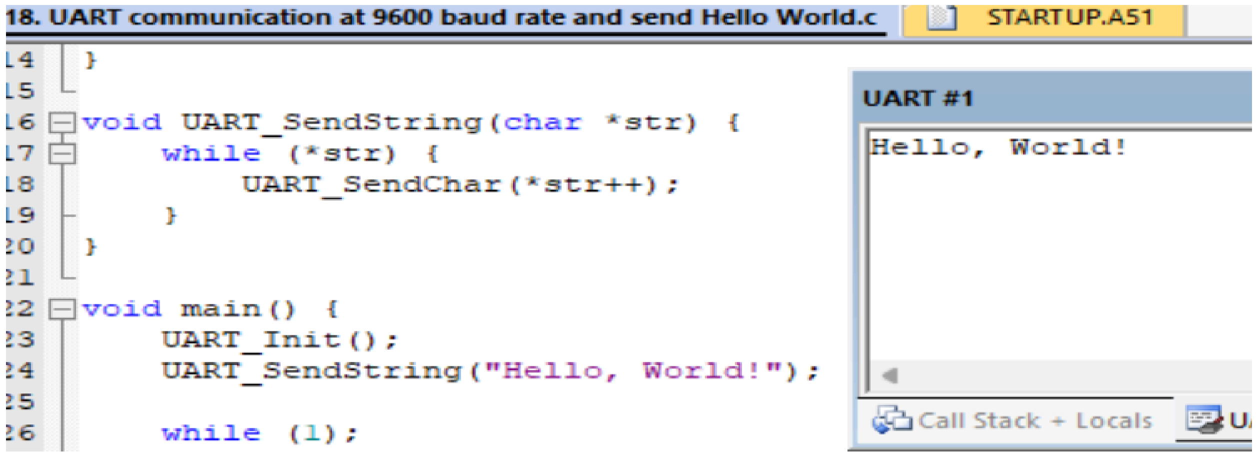
EXP 18

Aim: To Write a 8051 C program to initialize UART communication at 9600 baud rate and send "Hello, World!" (simulation mode).

Program:

```
#include <reg51.h>

void UART_Init() {
   TMOD = 0x20;
   TH1 = 0xFD;
   SCON = 0x50;
   TR1 = 1;
}

void UART_SendChar(char c) {
   SBUF = c;
   while (TI == 0);
   TI = 0;
}

void UART_SendString(char *str) {
   while (*str) {
      UART_SendChar(*str++);
   }
}

void main() {
   UART_Init();
   UART_SendString("Hello, World!");

   while (1);
}
```

Output:



18. UARTcommunication at 9600 baud rate and send Hello World.c   STARTUP.A51

```
}

void UART_SendString(char *str) {
    while (*str) {
        UART_SendChar(*str++);
    }
}

void main() {
    UART_Init();
    UART_SendString("Hello, World!");

    while (1);
```

UART #1

```
Hello, World!
```

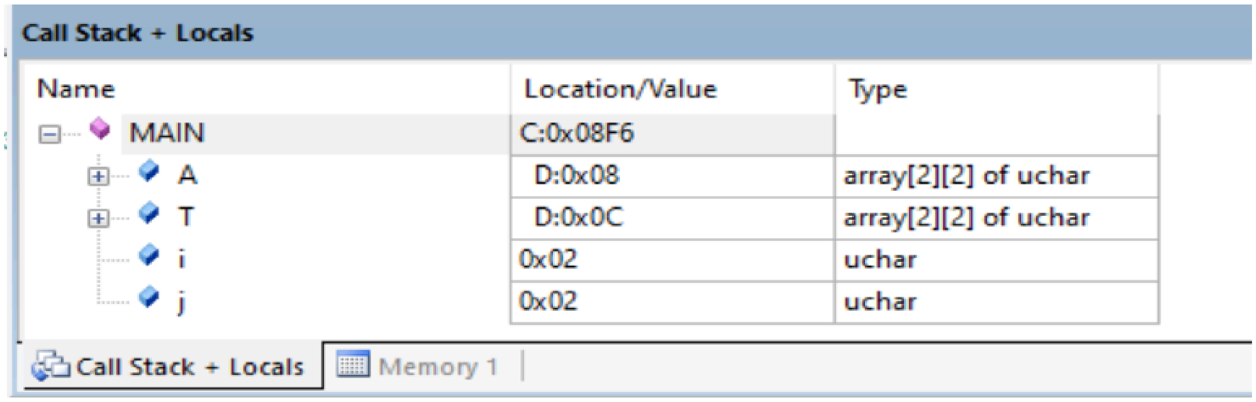
EXP 19

Aim: To Write a C program to find the transpose of a matrix.

Program:

```
#include <reg51.h>
void main() {
 unsigned char A[2][2] = {{1, 2}, {3, 4}};
 unsigned char T[2][2];
 unsigned char i, j;
 for (i = 0; i < 2; i++) {
 for (j = 0; j < 2; j++) {
 T[j][i] = A[i][j];
 }
 }
 }
```

Output:



Call Stack + Locals

Name	Location/Value	Type
MAIN	C:0x08F6	
A	D:0x08	array[2][2] of uchar
T	D:0x0C	array[2][2] of uchar
i	0x02	uchar
j	0x02	uchar

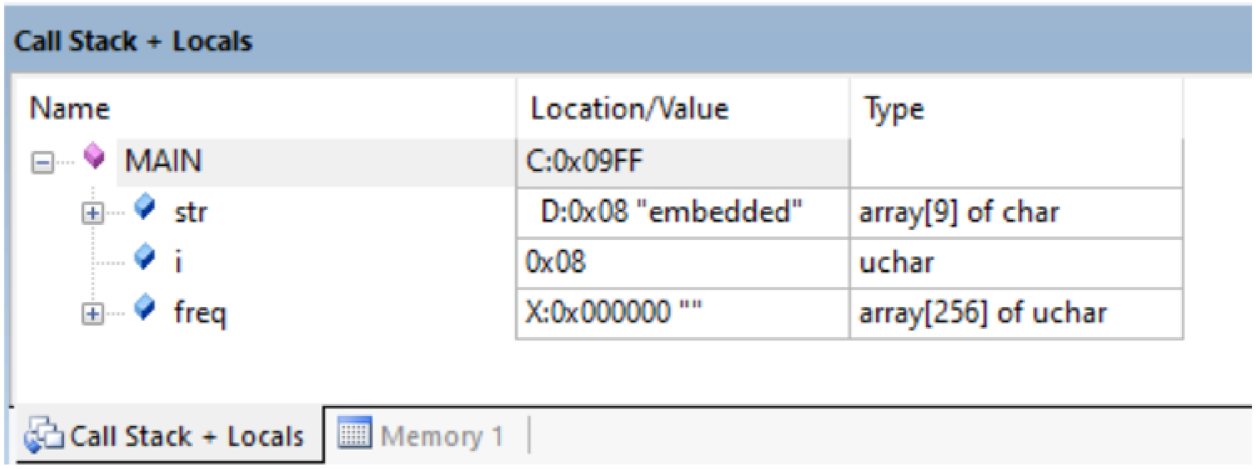
EXP 20

Aim: To Write a C program to count the frequency of each character in a given string.

Program:

```
#include <reg51.h>
void main() {
 char str[] = "embedded";
 unsigned char i;
 unsigned char xdata freq[256] = {0}; // Use external RAM (xdata)
 for (i = 0; str[i] != '\0'; i++) {
 freq[str[i]]++;
 }
 while (1);
}
```

Output:



Name	Location/Value	Type
MAIN	C:0x09FF	
str	D:0x08 "embedded"	array[9] of char
i	0x08	uchar
freq	X:0x000000 ""	array[256] of uchar