

## Experiment 2: Data overflow

### Objective:

1. To understand the concept of data overflow

### Theory:

Overflow occurs when an arithmetic operation or a memory allocation exceeds the capacity of the data type or storage that is available. There are specific conditions under which overflow happens, depending on the type of operation and the data type being used.

### Condition for overflow:

```
((AS==BS) &(AS==RS) || (AS!=BS))  
{  
NO overflow, display result;  
}
```

### Algorithm:

- i. Start
- ii. Observe carry into the sign bit position & carry out the sign bit position
- iii. If the two carry aren't equal, overflow should be detected
- iv. If the two carry are applied to an X-OR gate overflow will be detected when output of gate is 1.
- v. Stop

### Source Code:

```
#include <stdio.h>  
int main()  
{  
    int num1, num2, result, sign1, sign2, signResult;  
    printf("****Compiled by Jonash Chataut****\n");  
    printf("Enter two 4-bit numbers (0 to 15): ");  
    scanf("%d%d", &num1, &num2);  
    result = num1 + num2;  
    // Extract the 4th bit (sign bit) of each number and the result
```

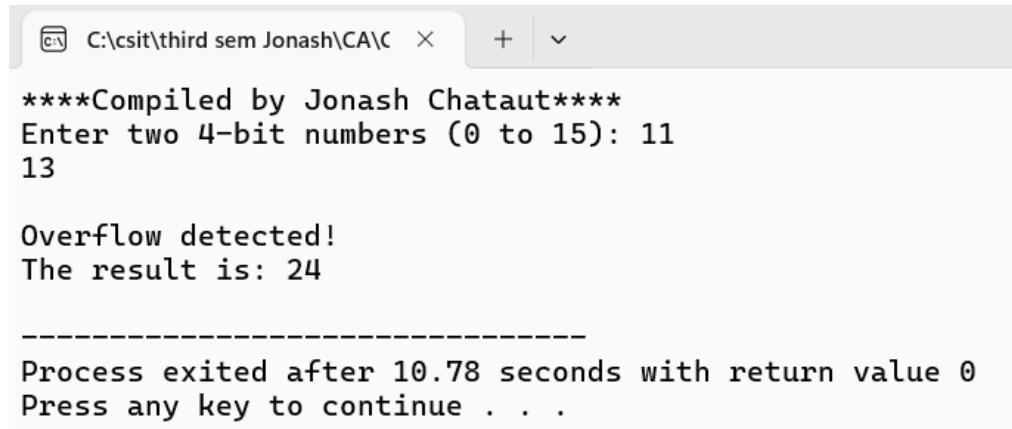
```
sign1 = (num1 >> 3) & 1; // 4th bit of num1
sign2 = (num2 >> 3) & 1; // 4th bit of num2
signResult = (result >> 4) & 1; // 5th bit of result (overflow bit)

// Check for overflow

if (signResult == 1)
{
    printf("\nOverflow detected!\nThe result is: %d\n", result);
}
else
{
    printf("\nNo overflow.\nResult = %d\n", result);
}

return 0;
}
```

**Output:**



```
****Compiled by Jonash Chataut****
Enter two 4-bit numbers (0 to 15): 11
13

Overflow detected!
The result is: 24

-----
Process exited after 10.78 seconds with return value 0
Press any key to continue . . .
```