Computer Science 240
Binary Operations
Assignment for Lab 2

Print this sheet and hand in completed exercises at the beginning of lab.

Perform addition on the following binary and hexadecimal numbers (assume **two's complement** format!). Indicate whether there is a carry-out or an overflow for each addition.

For the first 2 calculations, assume **16- bit representation.** Do the calculation using the binary values.

Then, convert the result to hexadecimal notation. To convert, divide the 16 binary digits of the result into groups of 4, and translate each group to the corresponding hexadecimal value. Note that if there is a carry-out, that is the 17th bit, and it is not used in result or in the hexadecimal translation!

Carry-Out? Overflow?

1. 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
+ 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Binary result :

Hexadecimal result:

Carry-Out? Overflow?

2. 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0
+ 0 1 1 1 1 1 1 1 1 0 0 1 1 0 0 1

Binary result :

Hexadecimal result:

Now, assume 32-bit representation, using hexadecimal notation, and specify result in hexadecimal.

Carry-Out? Overflow?

3. 0x A A F F 9 0 1 4
+ 0x A A E 3 C D 1 2

Hexadecimal result:

Carry-Out? Overflow?

4. 0x 7 F A A 3 2 7 8
+ 0x 6 0 2 4 C D 1 2

Hexadecimal result:

Examine the following C statements:

```
int x,y;

int sum = x + y;

int overflow = ((x > 0 && y > 0 && sum <= 0) || (x < 0 && y < 0 && sum >= 0));

long int carryout = (((long int)x + (long int)y) >> 32)&1;
```

5. Explain why the expressions for overflow and carryout will yield the correct result:

Use specific values for x and y (same as from exercises 3 and 4 above) to show that the correct results for overflow and carryout are performed for the given C expressions:

6. x = 0x A A F F 9 0 1 4
   y = 0x A A E 3 C D 1 2

7. x = 0x 7 F A A 3 2 7 8
   y = 0x 6 0 2 4 C D 1 2

NOTE: in the next exercises, you are asked to create a file containing a C program by using the *emacs* editor. If you copy and paste code from here to create your program, it may be necessary to re-type the quote marks to successfully compile the program.

8. If you did not complete exercises 8 – 9 from Lab 2, please complete them now, so that you get a chance to use the *emacs* editor and compile a file.

9. The following short C program accepts two integer (32-bit) hexadecimal numbers from the user as inputs and prints the sum, including the overflow and carryout:

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

void add(int x,int y) {
  int sum = x+y;
  printf("\n\nx + y  = 0x%x (decimal %d)",sum,sum);

  int overflow = ((x > 0 && y > 0 && sum <= 0)|| (x < 0 && y < 0 && sum >= 0));
  printf("\nOverflow = %d",overflow);

  long int carryout = (((long int)x + (long int)y) >> 32)&1;
  printf("\nCarryout = %ld\n",carryout);
}

int main()
{
   int x;
  int y;
  printf("Enter a hex value for x: 0x");
  scanf("%x",&x);
  printf("(decimal %d)",x);
  printf("\nEnter a hex value for y: 0x");
  scanf("%x",&y);
  printf("(decimal %d)",y);

  add(x,y);
  return 0;
}
```

In a Terminal, use emacs to create a file *sum.c* containing this program. Compile and run the program for the values from exercise 3 and 4. Verify your results.