Hetalben Patel
Computer Architecture - Summer 2017
Professor Brian Russell

Numeric Base Conversion and Calculator

---

calc.c ReadMe

Design

This program does not account for decimal inputs or outputs over what would fit into a 32 bit int, however it does arbitrary size for all other bases
The program calc takes in four arguments. The first argument is the operation (+ or -) to be done on two numbers. The second and third arguments are the numbers (having base: binary, decimal, octal or hexadecimal) in a string format. And the last argument is the base of the output. After all the variables are declared, the program runs an error check to make sure that the number of arguments is correct. Then, it goes through the arguments using a switch statement and assigns the value to the respective variable. Before assigning the value to num1 and num2, the program calls the functions first_number and second_number respectively. Both these functions check whether there is '-' sign or not and it creates a char array which holds only the digits of the arguments. The program consists of four different functions for the different bases that convert the given ASCII string of any base to an integer array which consists of only 1s and 0s. After converting both num1 and num2 to an array of 1s and 0s, the program calls either the addition or subtraction function based on the operation from the argument. If the argument is a negative number, the program calls 'fliping_bits' function, before executing add or subtract function, which would give the two's complement of the inputted number. Then it performs the add/subtract function. After adding or subtracting the numbers, the last thing left to do is to convert the array to the given output base. The program consists of four different function for the four bases that would convert the added/subtracted array back to the respective output base and print the final answer.

Challenges

- Writing and handling the conversion functions took a lot longer than expected
- Figuring out the segmentation fault errors within the code.

Analysis

This program is expected to run in linear time. Majority of the code depends on manipulation of the array. Therefore, it should take no longer than linear time. The amount of space used will also be linear.

Format Interpretation

---

format.c ReadMe

Design

The program format takes in two arguments. The first argument is a sequence of 32 bits consisting of 0s and 1s. The second argument gives the type (int or float) the sequence is to be interpreted as. The program first runs an error check to make sure that there are correct number of arguments. After that it take the input bit sequence and bit shifts it into a number union. Then it checks whether the type given is int or float. If it is an integer, it passes the bit sequence to binary_to_int function which would convert it to an integer which then gets passed to int_to_ASCII function which would convert it to a string. If the type is float, the bit sequence gets passes to floatToASCII function provided by the professor.