SP12-CSE20 Discrete Mathematics
Homework 1 Solution

April 17, 2012

1 Boolean Functions and Computer Arithmetic

1.1 Problem 2.12

Find the 8-bit two's complement of 67_{10}.

We first convert it into binary format then calculate its two's complement. We iteratively divide this decimal number by 16 and set each digit (from lowest to highest) to be the remainder. For instance, $67_{10}/2 = 33$ and $67_{10} \bmod 2 = 1$, so the first bit is set to be 1, and we replace 67 with 33 for the following division and modulation operations. In the end the binary number is 1000011, which is only of 7 bits, so we add 0 to the high-end bit and make it to be 01000011. The two's complement is thus obtained by flip all the bits then add it by 1, which turns out to be 10111101.

1.2 Problem 2.13

Find the 8-bit two's complement of 108_{10}.

As above, we first convert it into binary format then calculate its two's complement. We iteratively divide this decimal number by 16 and set each digit (from lowest to highest) to be the remainder. For instance, originally for the first iteration, we have $108_{10}/2 = 54$ and $108_{10} \bmod 2 = 0$, so the first bit is set to be 10, and we replace 108 with 54 for the following division and modulation operations. In the end the binary number becomes 1101100, which is only of 7 bits, so we add 0 to the high-end bit and make it to be

01101100. The two's complement is thus obtained by flip all the bits then add it by 1, which turns out to be 10010100.

1.3 Problem 2.14

The number 10001001_2 is the 8-bit two's complement of a number k. What is the decimal representation of k?

For every n-bit number a, its two's complement, b, is calculated as $b = 2^n - a$. Notice that from this equation we can also obtain $a = 2^n - b$, thus a is also the two's complement of b, and it is a mutual relation. As a result, we can get the original binary number of k by flip all the bits of its two's complement and add by one, which is 01110111_2. The decimal representation turns out to be 119.

1.4 Problem 2.15

The number 10111010_2 is the 8-bit twos complement of a number k. What is the decimal representation of k?

Similar as above, for every n-bit number a, its two's complement, b, is calculated as $b = 2^n - a$. Notice that from this equation we can also obtain $a = 2^n - b$, thus a is also the two's complement of b, and it is a mutual relation. As a result, we can get the original binary number of k by flip all the bits of its two's complement and add by one, which is 01000110_2. The decimal representation turns out to be 70.

1.5 Problem 2.16

Using base-2 arithmetic, compute 79 - 43. Then compute it using 8-bit two's-complement registers. Remember to check for overflow.

We first convert the two numbers into 8-bit binary formats, using the method as denoted above. $79 \rightarrow 01001111_2$ and $43 \rightarrow 00101011_2$. Then since $79 > 43$, we take binary arithmetic operation on them and get the result as 00100100_2. In the other way, using two's complement, we first convert the negative number -43 to be 11010101_2 and take the addition operation with 01001111_2. Notice that any carry bit is seen as overflow and will be discarded, and after bitwise addition the result turns out to be also 00100100_2.

1.6 Problem 2.17

Using base-2 arithmetic, compute - 15 - 46. Then compute it using 8-bit two's-complement registers. Also compute 46 + 46 + 46, remember to check for overflow.

For base-2 arithmetic, we can first reformat the equation to be $-(15+46) = -1 \times (1111 + 101110) = -1 \times 111101 = -61_{10}$.

For two's complement to help us calculate the result. The 8-bit two's complement of (-15) and (-46) are 11110001_2 and 11010010_2, respectively, and the sum of the two is 11000011_2, which equals (-61_{10}).

$46+46+46 = 00101110_2+00101110_2+00101110_2 = 10001010_2 = -118_{10}$, which is incorrect, due to the overflow bit.

1.7 Problem 2.18

We have defined and learned how to use the idea of two's complement for n-bit binary numbers. What about "n-digit ten's complement" for base ten arithmetic? Define the appropriate notion of ten's complement and show, by example, how to compute with it in a way that is analogous to computing with two's complement.

We can define it in exactly the same way as we define two's complement for binary numbers. Therefore, we can define the range of n-digit base-10 10's complement to be $[-5 \times 10^{n-1}, 5 \times 10^{n-1} - 1]$.

For instance, we have two decimal digits, and it is a signed number system, so the range would be $[-50, 49]$. For each negative number, x, you just use $100 - x$ to represent it, and after arithmetic operation just filter out the overflow bit. E.g., you want to calculate 18 - 12, so we need ten's complement of (-12), which turns out to be $18 + (-12) = 18 + (10^2 - 12) = 18 + 88 = 106 = 6$ (the hundred bit is overflow thus filtered, since we allow two decimal digits) On the contrary, to calculate 12 - 18, it turns out to be $12 + (-18) = 12+(10^2-18) = 12+82 = 94$ (94 is a negative number since it is larger than 49 and two digits only allows $[-50, 49]$, so its actual value is $-(100 - 94) = -6$.