Computer Architecture and Programming Languages Course: CO20-320241
Jacobs University Bremen 7 September, 2018
Romelda Blaceri

Homework 4

Problem 4.1

Solution:

We want the output X to be 1, therefore we need all the three inputs of the AND gate to be 1. One of these inputs is C directly, so C=1. For the XNOR gate ($\overline{B \oplus C}$) to be 1 , since we have C=1, then B must also be 1 ($\overline{1 \oplus 1} = 1$). Using the same logic for the XOR gate ($A \oplus B$), since B is 1, A should be 0 for $0 \oplus 1$ to give 1. As a result, the input condition has to be $A = 0, B = 1, C = 1$.

Problem 4.2

Solution:

a) The truth table for the circuit:

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

b) From the truth table, we get the following expression:

$\overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot \overline{B} \cdot C + \overline{A} \cdot B \cdot C + A \cdot \overline{B} \cdot \overline{C} + A \cdot B \cdot C$
$= \overline{A} \cdot \overline{B} \cdot (\overline{C} + C) + BC \cdot (\overline{A} + A) + A \cdot \overline{B} \cdot \overline{C}$
$= \overline{A} \cdot \overline{B} + B \cdot C + A \cdot \overline{B} \cdot \overline{C}$

The simplified expression using distributivity, commutativity and complement, is: $Y = \overline{A} \cdot \overline{B} + B \cdot C + A \cdot \overline{B} \cdot \overline{C}$

Problem 4.3

Solution:

a) We find the unsigned binary representation of +27:

$27_{10} \rightarrow$ binary
$27/2 = 13 + \mathbf{1}$
$13/2 = 6 + \mathbf{1}$
$6/2 = 3 + \mathbf{0}$
$3/2 = 1 + \mathbf{1}$
$1/2 = 0 + \mathbf{1}$

The resulting binary number is: 11011_2

After finding the binary representation, we add leading zeros until 7 bits are completed, and keeping in mind that the first bit should be 0 since it is a positive number, we add the 8th bit also. Therefore, the binary representation of +27 using 2's complement is:

$$0001\ 1011_2$$

b) Binary representation of unsigned number:

$66_{10} \rightarrow$ binary
$66/2 = 33 + \mathbf{0}$
$33/2 = 16 + \mathbf{1}$
$16/2 = 8 + \mathbf{0}$
$8/2 = 4 + \mathbf{0}$
$4/2 = 2 + \mathbf{0}$
$2/2 = 1 + \mathbf{0}$
$1/2 = 0 + \mathbf{1}$

The resulting binary number is: 1000010_2

Since there are already 7 bits, we just have to add the sign bit, which would be 0 since the number is positive. The resulting binary representation is:

$$0100\ 0010_2$$

c) Unsigned number $|-18| = 18$:

$18_{10} \rightarrow$ binary
$18/2 = 9 + \mathbf{0}$
$9/2 = 4 + \mathbf{1}$
$4/2 = 2 + \mathbf{0}$
$2/2 = 1 + \mathbf{0}$
$1/2 = 0 + \mathbf{1}$

The resulting binary number: 10010_2

We add three leading zeros: $0001\ 0010_2$

We invert the bits: $1110\ 1101_2$

We add one to the resulting binary and get the final representation as:

$$1110\ 1110_2$$

d) Using same procedure as in the previous examples:

$127_{10} \rightarrow$ binary
$127/2 = 63 + \mathbf{1}$
$63/2 = 31 + \mathbf{1}$
$31/2 = 15 + \mathbf{1}$
$15/2 = 7 + \mathbf{1}$
$7/2 = 3 + \mathbf{1}$
$3/2 = 1 + \mathbf{1}$
$1/2 = 0 + \mathbf{1}$

The resulting binary number: 1111111_2

We add the remaining bit, which will be 0 since it's the sign bit, so the final representation would be:

$$0111\ 1111_2$$

e) Using the result from part d: $0111\ 1111_2$

We invert the bits: $1000\ 0000_2$

Lastly, we add 1 to the resulting binary, and the final representation is:

$$1000\ 0001_2$$

f) $128_{10} \rightarrow$ binary
$128/2 = 64 + \mathbf{0}$
$64/2 = 32 + \mathbf{0}$
$32/2 = 16 + \mathbf{0}$
$16/2 = 8 + \mathbf{0}$
$8/2 = 4 + \mathbf{0}$
$4/2 = 2 + \mathbf{0}$
$2/2 = 1 + \mathbf{0}$
$1/2 = 0 + \mathbf{1}$

Binary representation: 10000000_2

We invert the number: 01111111_2

We add 1: 10000000_2

It is mentioned even in the 6th exercise, -128 is the last negative number that can be expressed in 8 bit representation. Positive 128 can't, that's why we have the same value when we first converted unsigned 128 to binary and after we calculated for -128.

g) $131_{10} \rightarrow$ binary
$131/2 = 65 + \mathbf{1}$
$65/2 = 32 + \mathbf{1}$
$32/2 = 16 + \mathbf{0}$
$16/2 = 8 + \mathbf{0}$
$8/2 = 4 + \mathbf{0}$
$4/2 = 2 + \mathbf{0}$
$2/2 = 1 + \mathbf{0}$
$1/2 = 0 + \mathbf{1}$

The resulting binary number: 10000011_2

We see that the result occupies 8 bits already, so we can't actually talk about a sign bit in this number. Therefore we cannot represent +131 in the required 8 bit format. And again, as it will be shown in exercise 6, the biggest positive number that can be expressed in 8 bit representation is 127 (01111111).

h) $7_{10} \rightarrow$ binary
$7/2 = 3 + \mathbf{1}$
$3/2 = 1 + \mathbf{1}$
$1/2 = 0 + \mathbf{1}$

The resulting binary number is: 111_2

The leading zeros are added: 00000111_2

We invert the bits: 11111000_2

After adding one, the final result is: $1111\ 1001_2$

** Clarified notation: $n/2 = m + \textbf{remainder}$ in the sense: $2m + \textbf{remainder} = n$

Problem 4.4
Solution:

a) The sign bit is 0, so the number is positive. Therefore, the conversion will be as usual:

$$\overset{7}{0}\overset{6}{0}\overset{5}{0}\overset{4}{1}\overset{3}{1}\overset{2}{0}\overset{1}{0}\overset{0}{0}{}_2 \rightarrow 0 \cdot 2^7 + 0 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 =$$
$$= 16 + 8 = +24_{10}$$

b) The sign bit is 1, so the number is negative. Therefore, we perform the following conversion:
- We subtract 1 from the value: $11110101 - 1 = 11110100$
- We invert the bits: 0000 1011

Now the conversion continues normally:

$$\overset{76543210}{00001011_2} \rightarrow 0\cdot 2^7 + 0\cdot 2^6 + 0\cdot 2^5 + 0\cdot 2^4 + 1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 =$$
$$= 1 + 2 + 8 = 11 \rightarrow -11_{10}$$

c) Checking the sign bit, this number is positive, so we perform the normal conversion:

$$01011011_2 \rightarrow 0\cdot 2^7 + 1\cdot 2^6 + 0\cdot 2^5 + 1\cdot 2^4 + 1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 =$$
$$= 64 + 16 + 8 + 2 + 1 = +91_{10}$$

d) This number is negative (sign bit is 1). Same procedure as in point b will be used:

** $10110110 - 1 = 10110101$
** Inversion: 01001010
** Normal conversion:

$$01001010_2 \rightarrow 0\cdot 2^7 + 1\cdot 2^6 + 0\cdot 2^5 + 0\cdot 2^4 + 1\cdot 2^3 + 0\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0 =$$
$$= 64 + 8 + 2 = 74 \rightarrow -74_{10}$$

e) This number is negative as the sign bit is 1. Therefore, we have:

** $11111111 - 1 = 11111110$
** Inversion: 00000001
** Normal conversion:

$$00000001_2 \rightarrow 0\cdot 2^7 + 0\cdot 2^6 + 0\cdot 2^5 + 0\cdot 2^4 + 0\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 1\cdot 2^0 =$$
$$= 1 \rightarrow -1_{10}$$

f) This number is positive (sign bit is 0).

$$01101111_2 \rightarrow 0\cdot 2^7 + 1\cdot 2^6 + 1\cdot 2^5 + 0\cdot 2^4 + 1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 =$$
$$= 64 + 32 + 8 + 4 + 2 + 1 = +111_{10}$$

g) We have a negative number since sign bit is 1:

** $10000001 - 1 = 10000000$
** Inversion: 01111111
** Normal conversion to binary:

$$01111111_2 \rightarrow 0\cdot 2^7 + 1\cdot 2^6 + 1\cdot 2^5 + 1\cdot 2^4 + 1\cdot 2^3 + 1\cdot 2^2 + 1\cdot 2^1 + 1\cdot 2^0 =$$
$$= 64 + 32 + 16 + 8 + 4 + 2 + 1 = 127 \rightarrow -127_{10}$$

h) Negative number case again:

** $10000000 - 1 = 01111111$
** Inversion: 10000000
** Normal conversion:

$$10000000_2 \rightarrow 1\cdot 2^7 + 0\cdot 2^6 + 0\cdot 2^5 + 0\cdot 2^4 + 0\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0 =$$
$$= 128 \rightarrow -128_{10}$$

Problem 4.5

Solution:

a) When adding 27 and 36, $7 + 6 > 9$, which resultn in a carry. Therefore, we have to add 6 (representation in BCD is 0110) to the result:

27	0010	0111
+36	+ 0011	0110
63	0101	1101
	+	0110
	0110	0011

As a result we get the correct BCD representation of 63.

b) When we add 73 and 29, both additions of 3 and 9, as well as of 7 and 2, create a carry, so we have to add 6 (0110_{BCD}) to both the left counterpart and the right counterpart of the initial result:

73		0111	0011
+29		+ 0010	1001
102		1001	1100
		+ 0110	0110
	0001	0000	0010

In the end we get the exact BCD representation of 102.

Problem 4.6

Solution:

a) An m-bit unsigned number represents all numbers in the range 0 to $2^m - 1$. In our case, the range of 8-bit unsigned binary numbers is from 0 to 255_{10}. We derive that formula from the fact that the last bit represents 2^7 and the sum of the remaining bits when they are set to 1 is $2^7 - 1$. The total value: $2^7 + 2^7 - 1 = 2^8 - 1 = 256 - 1 = 255_{10}$.

b) According to 2's complement representation we're using mostly, we have the sign bit and then 7 bits to represent the range, which is from -128 to 127, because $2^7 - 1 = 127$ for the positive part. As for the negative side, same logic is used but since 0 is considered among the positive numbers in this case, we add a number. We could also calculate this in the following way: biggest positive 8-bit number would be 01111111 (0 - sign bit, all the other bits = 1) = 127. As for the negative numbers, we already made the calculations in problem 4.4 for 10000000 = -128 (since the number is inverted at first, in order for its unsigned value to be the biggest we need 7 1s that turned 0, and the first bit would be 1 since it's the sign bit).

c) Same logic as in part a: $2^{11} - 1 = 2048 - 1 = 2047$. Therefore, the range is from 0 to 2047.

d) Same logic as in part b: from -2^{10} to $2^{10}-1$ which is from -1024 (10000000000) to 1023 (01111111111).

e) Same logic again: -2^{15} to $2^{15} - 1$ which is from -32768 (1000000000000000) to 32767 (0111111111111111).