This week's highlights

- Define multiple ways for representing numbers
- Compute the ranges of numbers that can be represented using a given definition
- Represent negative integers in multiple ways
- Perform arithmetic operations on integers using multiple representations
- Relate algorithms for integer operations to bitwise boolean operations
- List the truth tables and meanings for conjunction, disjunction, exclusive or.
- Correctly use XOR and bit shifts
- Relate boolean operations to applications in combinatorial circuits.

Lecture videos

Week 2 Day 1 YouTube playlist

Week 2 Day 2 YouTube playlist

Week 2 Day 3 YouTube playlist

example finding the binary expansion $(103)_7$

Def: $(103)_7 = 1\cdot 7^2 + 0\cdot 7 + 3\cdot 7^0$
$= 49 + 0 + 3 = 52$

number is 52.

what binary expansion does it have?

Alg 1: write 52 as sum of powers of 2, starting with biggest

$$52 = 32 + \underbrace{16 + \overbrace{4}^{4}}_{20}$$

$$= 2^5 + 2^4 + 0\cdot 2^3 + 1\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0$$

$$= (110100)_2$$

alternately:

Alg 2

$$\underbrace{\_\ \_\ \_}_{52 \text{ div } 2}\ \overset{52 \bmod 2}{\_}$$

$a_0 = 52 \bmod 2 = 0$ remaining value $q = 52 \text{ div } 2 = 26$

$a_1 = 26 \bmod 2 = 0$ update $q = 26 \text{ div } 2 = 13$

$a_2 = 13 \bmod 2 = 1$ update $q = 13 \text{ div } 2 = 6$

$$13 = 2\boxed{6} + \underset{r}{1}$$

Monday January 11

Definition (Rosen p. 246) For b an integer greater than 1 and n a positive integer, the **base b expansion of n** is

$$(a_{k-1} \cdots a_1 a_0)_b$$

b^0 place
ones place

k-1st column   0th column

where k is a positive integer, $a_0, a_1, \ldots, a_{k-1}$ are nonnegative integers less than b, $a_{k-1} \neq 0$, and

$$n = a_{k-1}b^{k-1} + \cdots + a_1 b + a_0$$

coeff of 1.

* example $n = 17$ $b = 3$
$17 = 9 + 6 + 2 = 1 \cdot 3^2 + 2 \cdot 3^1 + 2 \cdot 3^0 = (122)_3$

* example $(1)_2 = 1 = (1)_8$

* example $(20)_{10} = 16 + 4 = 2^4 + 2^2 = (10100)_2$

* example $(142)_{16} = 1 \cdot 16^2 + 4 \cdot 16^1 + 2 \cdot 16^0 = 256 + 64 + 2 = 322$

* example $(35)_8 = 3 \cdot 8^1 + 5 \cdot 8^0 = 24 + 5 = 29$

* example $(1D)_{16} = 1 \cdot 16^1 + 13 = 16 + 13 = 29$

Algorithm for converting from base b_1 expansion to base b_2 expansion:

Input: number in base b_1

1. Use definition of base expansion to express number in decimal
2. Use Alg1 to convert to expansion in base b_2, output this expansion.

Definition For b an integer greater than 1, w a positive integer, and n a nonnegative integer $n < b^w$, the **base b fixed-width w expansion of n** is

$$(a_{w-1} \cdots a_1 a_0)_{b,w}$$

NOTATION: b is base
w is width

where $a_0, a_1, \ldots, a_{w-1}$ are nonnegative integers less than b and

new: no constraint

$$n = a_{w-1}b^{w-1} + \cdots + a_1 b + a_0$$

same as before

Decimal $b = 10$	Binary $b = 2$	Binary fixed-width 10 $b = 2$, $w = 10$	Binary fixed-width 7 $b = 2$, $w = 7$	Binary fixed-width 4 $b = 2$, $w = 4$
$(20)_{10}$	$(10100)_2$	$(0000010100)_{2,10}$	$(0010100)_{2,7}$	impossible

Definition For b an integer greater than 1, w a positive integer, w' a positive integer, and x a real number the **base b fixed-width expansion of x with integer part width w and fractional part width w'** is $(\underbrace{a_{w-1}\cdots a_1 a_0}_{\text{integer part}}\underset{\text{radix point}}{.}\underbrace{c_1\cdots c_{w'}}_{\text{fractional part}})_{b,w,w'}$ where $a_0, a_1, \ldots, a_{w-1}, c_1, \ldots, c_{w'}$ are nonnegative integers less than b and

$$x \geq a_{w-1}b^{w-1} + \cdots + a_1 b + a_0 + c_1 b^{-1} + \cdots + c_{w'} b^{-w'}$$

and

$$x < a_{w-1}b^{w-1} + \cdots + a_1 b + a_0 + c_1 b^{-1} + \cdots + (c_{w'}+1) b^{-w'}$$

GIVES APPROXIMATION!

3.75 in fixed-width binary, integer part width 2, fractional part width 8	$(\underset{2}{1}\,\underset{1}{1}\,.\,\underset{\frac{1}{2}}{1}\,\underset{\frac{1}{4}}{1}\,\underset{\frac{1}{8}}{0}\,\underset{\frac{1}{16}}{0}\,\underset{\frac{1}{32}}{0}\,\underset{\frac{1}{64}}{0}\,\underset{\frac{1}{128}}{0}\,\underset{\frac{1}{256}}{0})_{2,2,8}$
0.1 in fixed-width binary, integer part width 2, fractional part width 8	$(00.00011001)_{2,2,8}$

$0.1 < 0.5$

$0.1 < 0.25$

$0.0625 < 0.1 < 0.125$

Example

12.375 in fixed width binary integer part width ~~4~~ 5 fractional part width 5.

$(01100.0110\_)_{2,5,5}$

$12 = 8 + 4 = 2^3 + 2^2 + 0\cdot 2^1 + 0 \cdot 2^0$

$= (1100)_2$

$2^{-1} = \frac{1}{2} = 0.5$ $\quad 2^{-2} = \frac{1}{4} = 0.25$ $\quad 2^{-3} = \frac{1}{8} = 0.125$ $\quad 2^{-4}$ $\quad 2^{-5}$

$2^{-2} + 2^{-3} = 0.375$

```
welcome $jshell
|  Welcome to JShell -- Version 10.0.1
|  For an introduction type: /help intro

jshell> 0.1
$1 ==>

jshell> 0.2
$2 ==>

jshell> 0.1 + 0.2
$3 ==>

jshell> Math.sqrt(2)
$4 ==>

jshell> Math.sqrt(2)*Math.sqrt(2)
$5 ==>

jshell>
```

Note: Java uses floating point, not fixed width representation, but similar rounding errors appear in both.

Wednesday January 13

Reminders of definitions:

base b expansion of n	**base b fixed-width w expansion of n**
For b an integer greater than 1 and n a positive integer, the **base b expansion of n** is $(a_{k-1} \cdots a_1 a_0)_b$ where k is a positive integer, $a_0, a_1, \ldots, a_{k-1}$ are nonnegative integers less than b, $a_{k-1} \neq 0$, and $n = a_{k-1}b^{k-1} + \cdots + a_1 b + a_0$	For b an integer greater than 1, w a positive integer, and n a nonnegative integer with $n < b^w$, the **base b fixed-width w expansion of n** is $(a_{w-1} \cdots a_1 a_0)_{b,w}$ where $a_0, a_1, \ldots, a_{w-1}$ are nonnegative integers less than b and $n = a_{w-1}b^{w-1} + \cdots + a_1 b + a_0$

Representing negative integers in binary: Fix a positive integer width for the representation w, $w > 1$.

	To represent a positive integer n	To represent a negative integer $-n$
Sign-magnitude	$[0a_{w-2} \cdots a_0]_{s,w}$, where $n = (a_{w-2} \cdots a_0)_{2,w-1}$ ↑ signbit Example $n = 17$, $w = 7$: $17 = 16+1 = (10001)_2 = (010001)_{2,6}$ $17 = [0\ 0\ 1\ 0\ 0\ 0\ 1]_{s,7}$	$[1a_{w-2} \cdots a_0]_{s,w}$, where $n = (a_{w-2} \cdots a_0)_{2,w-1}$ ↑ sign bit Example $-n = -17$, $w = 7$: $17 = 16+1 = (10001)_2 = (010001)_{2,6}$ $-17 = [1\ 0\ 1\ 0\ 0\ 0\ 1]_{s,7}$
2s complement	$[0a_{w-2} \cdots a_0]_{2c,w}$, where $n = (a_{w-2} \cdots a_0)_{2,w-1}$ Example $n = 17$, $w = 7$: $17 = (010001)_{2,6}$ $17 = [0\ 0\ 1\ 0\ 0\ 0\ 1]_{2c,7}$	$[1a_{w-2} \cdots a_0]_{2c,w}$, where $2^{w-1} - n = (a_{w-2} \cdots a_0)_{2,w-1}$ Example $-n = -17$, $w = 7$: $2^6 - 17 = 64 - 17 = 47 = 32 + 8 + 4 + \ \ + 1$ $= (101111)_{2,6}$ $-17 = [1\ 1\ 0\ 1\ 1\ 1\ 1]_{2c,7}$
Extra example: 1s complement	$[0a_{w-2} \cdots a_0]_{1c,w}$, where $n = (a_{w-2} \cdots a_0)_{2,w-1}$ Example $n = 17$, $w = 7$:	$[1\bar{a}_{w-2} \cdots \bar{a}_0]_{1c,w}$, where $n = (a_{w-2} \cdots a_0)_{2,w-1}$ and we define $\bar{0} = 1$ and $\bar{1} = 0$. Example $-n = -17$, $w = 7$:

Trick: $-17 = -2^6 + ?$

$= -64 + 32 + 8 + 4 + 2 + 1$ so $-17 = [1\ 1\ 0\ 1\ 1\ 1\ 1]_{2c,7}$

$-2^6\ \ 2^5\ \ 2^4\ \ 2^3\ \ 2^2\ \ 2^1\ \ 2^0$

Trick: $17 = (0010001)_{2,7}$ Flipbits 1101110 Add1 1101111 matches

Representing 0:

Sign magnitude $[00000]_{s,5} = 0 = [10000]_{s,5}$

2s complement $[000]_{2c,3} = 0$ $\quad [1111]_{2c,4} =$

$[1000]_{2c,4} =$

$$[000]_{2c,3} = 0\cdot(-2^2) + 0\cdot 2^1 + 0\cdot 2^0$$
$$= 0(-4) + 0(2) + 0(1)$$
$$= 0$$

base

$$\left(\begin{array}{ccccccc} 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 3^6 & 3^5 & 3^4 & 3^3 & 3^2 & 3^1 & 3^0 \end{array}\right)_3 = \text{pos \# - - -}$$

$$\left(\begin{array}{ccccccc} 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \end{array}\right)_2 = \text{pos \# - -}$$

$$\begin{array}{ccccccc} & & 16 & & 4 & 2 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 \end{array}_{s,7} = -23$$

width

$$\begin{array}{ccccccc} -64 & & 16 & & 4 & 2 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 \end{array}_{2c,7} = -64+16+4+2+1$$
$$= -41$$

$$\begin{array}{cccc} -8 & 4 & 2 & 1 \\ 1 & 1 & 1 & 1 \end{array}_{2c,4} = 1\cdot(-8) + 1(4) + 1(2) + 1(1)$$
$$= -8+4+2+1 = -1$$

$$[1000]_{2c,4} = 1(-8) + 0(4) + 0(2) + 0(1)$$
$$= -8$$

$$[0111]_{2c,4} = 0(-8) + 1(4) + 1(2) + 1(1)$$
$$= 7$$

$$\left[\overset{\text{sign}}{1}\ \underset{4}{1}\ \underset{2}{1}\ \underset{1}{1}\right]_{s,4} = -(4+2+1)$$

$$= -7$$

$$[1000]_{s,4} = (-1)(0 \cdot 4 + 0 \cdot 2 + 0 \cdot 1)$$

$$= (-1)(0) = 0$$

$$[0111]_{s,4} = (+1)(4+2+1) = +7$$

Fixed-width addition: adding one bit at time, using the usual column-by-column and carry arithmetic, and dropping the carry from the leftmost column so the result is the same width as the summands. *Does this give the right value for the sum?*

+1 carry

$$
\begin{array}{r}
\begin{array}{cccccc} 32 & 16 & 8 & 4 & 2 & 1 \end{array} \\
(1\ 1\ 0\ 1\ 0\ 0)_{2,6} \\
+(0\ 0\ 0\ 1\ 0\ 1)_{2,6} \\
\hline
1\ 1\ 1\ 0\ 0\ 1
\end{array}
$$

fixed width calculations (unsigned)

$32+16+4=52$

$4+1=5$

$57=32+16+8+1$
$=(111001)_{2,6}$

Matches!

$$
\begin{array}{r}
\begin{array}{cccccc} +/- & 16 & 8 & 4 & 2 & 1 \end{array} \\
[1\ 1\ 0\ 1\ 0\ 0]_{s,6} \\
+[0\ 0\ 0\ 1\ 0\ 1]_{s,6} \\
\hline
\end{array}
$$

sign magnitude calculations

$16+4=20$

Top summand is -20

$4+1=5$

Bottom summand is 5

$-15=[101111]_{s,6}$

$-25=[111001]_{s,6}$

Not right

$$
\begin{array}{r}
\begin{array}{cccccc} -32 & 16 & 8 & 4 & 2 & 1 \end{array} \\
[1\ 1\ 0\ 1\ 0\ 0]_{2c,6} \\
+[0\ 0\ 0\ 1\ 0\ 1]_{2c,6} \\
\hline
\end{array}
$$

2s complement calculations

$-32+16+4=-12$

Top summand is -12

$4+1=5$

Bottom summand is 5

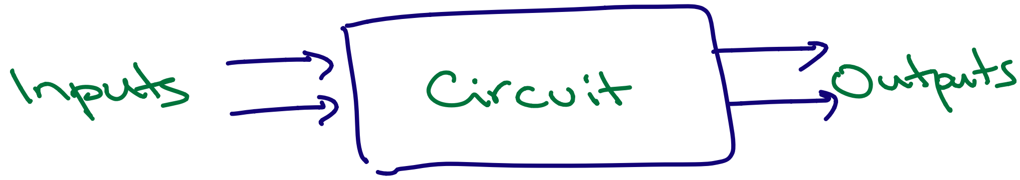
$-7=-32+16+8+1$
$=[111001]_{2c,6}$

Matches!

Extra example

$$
\begin{array}{r}
(1\ 1\ 0\ 1\ 0\ 0)_{2,6} \\
\times(0\ 0\ 0\ 1\ 0\ 1)_{2,6} \\
\hline
\end{array}
\qquad
\begin{array}{r}
[1\ 1\ 0\ 1\ 0\ 0]_{s,6} \\
\times[0\ 0\ 0\ 1\ 0\ 1]_{s,6} \\
\hline
\end{array}
\qquad
\begin{array}{r}
[1\ 1\ 0\ 1\ 0\ 0]_{2c,6} \\
\times[0\ 0\ 0\ 1\ 0\ 1]_{2c,6} \\
\hline
\end{array}
$$

Friday January 15

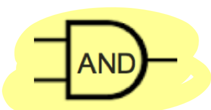


Definition tables for gates:

Input		Output
x	y	x AND y
1	1	1
1	0	0
0	1	0
0	0	0

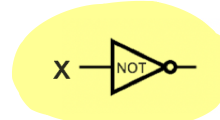
Input		Output
x	y	x XOR y
1	1	0
1	0	1
0	1	1
0	0	0

Input	Output
x	NOT x
1	0
0	1



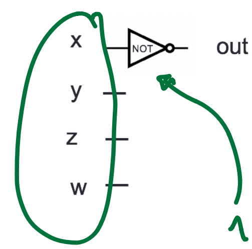
$\wedge$

$\oplus$



$\neg$ inverter

Example digital circuit:



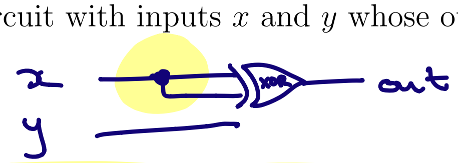
Output when $x = 1, y = 0, z = 0, w = 1$ is 0

Output when $x = 1, y = 1, z = 1, w = 1$ is 0

Output when $x = 0, y = 0, z = 0, w = 1$ is 1

output value only depends on value of input x for this example

Draw a logic circuit with inputs x and y whose output is always 0. *Can you use exactly 1 gate?*



use exactly 1 gate, an XOR gate

Fixed-width addition: adding one bit at time, using the usual column-by-column and carry arithmetic, and dropping the carry from the leftmost column so the result is the same width as the summands. In many cases, this gives representation of the correct value for the sum when we interpret the summands in fixed-width binary or in 2s complement.

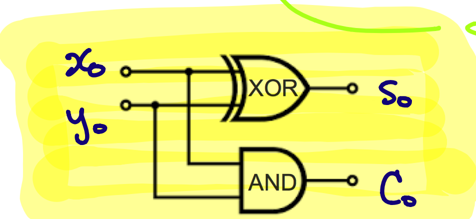
For single column:

Definition table for one column addition

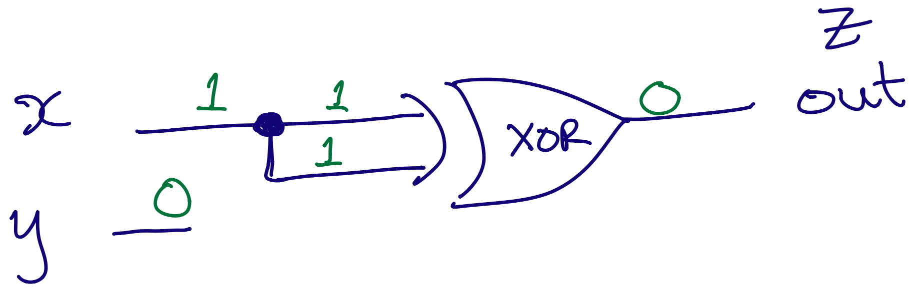
Input		Output	
x_0	y_0	s_0	c_0
1	1	0	1
1	0	1	0
0	1	1	0
0	0	0	0

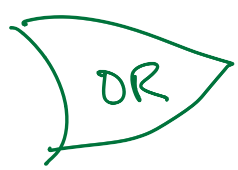
carry bit

sum bit

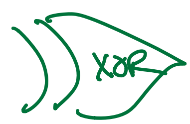


This circuit is called a half adder

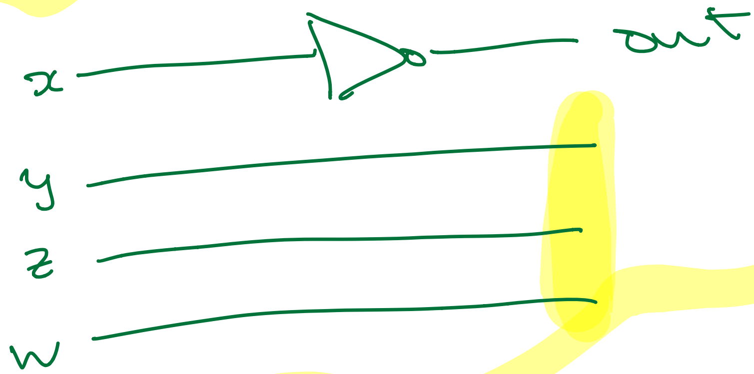




not
same
as



x	y	x OR y	x XOR y
1	1	1	0
1	0	1	1
0	1	1	1
0	0	0	0
		$\vee$	$\oplus$



The output out is set to 1 exactly when $(wzyx)_{2,4}$ is an even integer and set to zero otherwise

b/c x is rightmost bit

out is set to 1 exactly when x is 0 and out is set to 0 when x is 1.

Two cases:

x is 0

$(wzyx)_{2,4}$

$= (wzy0)_{2,4}$

$= w2^3 + z2^2 + y2^1$

$= 2\,(4w + 2z + y)$ even

x is 1

$(wzyx)_{2,4}$

$= 2\,\underline{(4w+2z+y)} + 1$

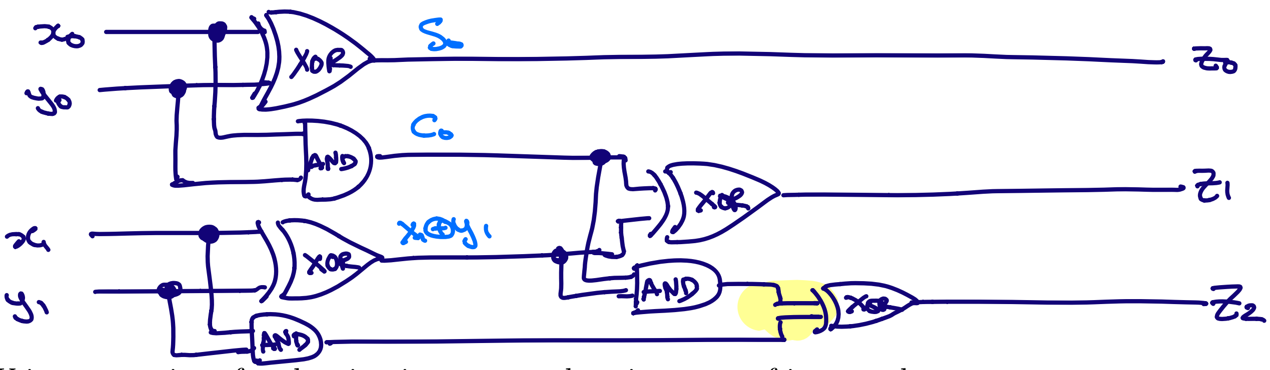
odd

Draw a logic circuit that implements fixed-width 2 binary addition:

- Inputs x_0, y_0, x_1, y_1 represent $(x_1x_0)_{2,2}$ and $(y_1y_0)_{2,2}$
- Outputs z_0, z_1, z_2 represent $(z_2z_1z_0)_{2,3} = (x_1x_0)_{2,2} + (y_1y_0)_{2,2}$ (may require up to width 3)

max input values: 3, 3
$3+3=6=(110)_2$

First approach: half-adder for each column, then combine carry from right column with sum of left column



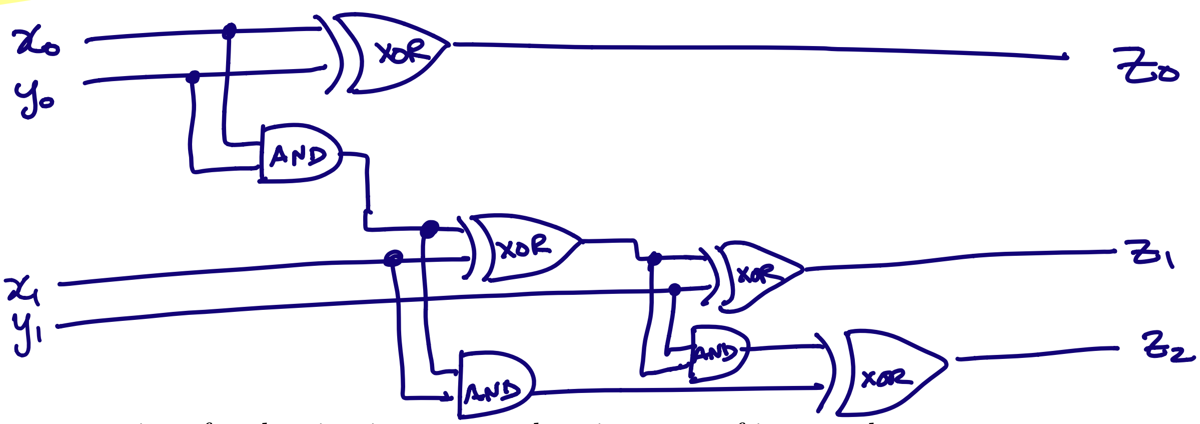
Write expressions for the circuit output values in terms of input values:

$z_0 =$ $x_0 \oplus y_0$

$z_1 =$ $(x_1 \oplus y_1) \oplus C_0 = (x_1 \oplus y_1) \oplus (x_0 \wedge y_0)$

$z_2 =$ $C_1 \oplus ((x_1 \oplus y_1) \wedge C_0)$

Second approach: for middle column, first add carry from right column to x_1, then add result to y_1



Write expressions for the circuit output values in terms of input values:

$z_0 =$ $x_0 \oplus y_0$

$z_1 =$ $(C_0 \oplus x_1) \oplus y_1 = ((x_0 \wedge y_0) \oplus x_1) \oplus y_1$

$z_2 =$ $(C_0 \wedge x_1) \oplus ((C_0 \oplus x_1) \wedge y_1)$

Extra example Describe how to generalize this addition circuit for larger width inputs.

Review quiz questions

1. Recall the definitions from class for number representations for **base b expansion of n**, **base b fixed-width w expansion of n**, and **base b fixed-width expansion of x with integer part width w and fractional part width w'**.

   For example, the base 2 (binary) expansion of 4 is $(100)_2$ and the base 2 (binary) fixed-width 8 expansion of 4 is $(00000100)_{2,8}$ and the base 2 (binary) fixed-width expansion of 4 with integer part width 3 and fractional part width 2 of 4 is $(100.00)_{2,3,2}$

   Compute the listed expansions. Enter your number using the notation for base expansions with parentheses but without subscripts. For example, if your answer were $(100)_{2,3}$ you would type `(100)2,3` into Gradescope.

   (a) Give the binary (base 2) expansion of the number whose octal (base 8) expansion is

   $$(371)_8$$

   (b) Give the decimal (base 10) expansion of the number whose octal (base 8) expansion is

   $$(371)_8$$

   (c) Give the octal (base 8) fixed-width 3 expansion of $(9)_{10}$?

   (d) Give the ternary (base 3) fixed-width 8 expansion of $(9)_{10}$?

   (e) Give the hexadecimal (base 16) fixed-width 6 expansion of $(16711935)_{10}$?[1]

   (f) Give the hexadecimal (base 16) fixed-width 4 expansion of

   $$(1011\ 1010\ 1001\ 0000)_2$$

   Note: the spaces between each group of 4 bits above are for your convenience only. How might they help your calculations?

   (g) Give the binary fixed width expansion of 0.125 with integer part width 2 and fractional part width 4.

   (h) Give the binary fixed width expansion of 0.1 with integer part width 2 and fractional part width 3.

2. Select all and only the correct choices below.

   (a) Suppose you were told that the positive integer n_1 has the property that n_1 **div** $2 = 0$. Which of the following can you conclude?

      i. n_1 has a binary (base 2) expansion
      ii. n_1 has a ternary (base 3) expansion
      iii. n_1 has a hexadecimal (base 16) expansion
      iv. n_1 has a base 2 fixed-width 1 expansion
      v. n_1 has a base 2 fixed-width 20 expansion

   (b) Suppose you were told that the positive integer n_2 has the property that n_2 **mod** $4 = 0$. Which of the following can you conclude?

[1] This matches a frequent debugging task – sometimes a program will show a number formatted as a base 10 integer that is much better understood with another representation.

i. the leftmost symbol in the binary (base 2) expansion of n_2 is 1
ii. the leftmost symbol in the base 4 expansion of n_2 is 1
iii. the rightmost symbol in the base 4 expansion of n_2 is 0
iv. the rightmost symbol in the octal (base 8) expansion of n_2 is 0

3. Recall the definitions of signed integer representations from class: sign-magnitude and 2s complement.

 (a) Give the 2s complement width 6 representation of the number represented in binary fixed-width 6 representation as $(001011)_{2,6}$.
 (b) Give the 2s complement width 4 representation of the number represented in sign-magnitude width 4 as $[1111]_{s,4}$.
 (c) Give the sign magnitude width 4 representation of the number represented in 2s complement width 4 as $[1111]_{2c,4}$.
 (d) In binary fixed-width addition (adding one bit at time, using the usual column-by-column and carry arithmetic, and ignoring the carry from the leftmost column), we get:

$$\begin{array}{rl} 1110 & \text{first summand} \\ +0100 & \text{second summand} \\ \hline 0010 & \text{result} \end{array}$$

 Select all and only the true statements below:

 i. When interpreting each of the summands and the result in binary fixed-width 4, the result represents the actual value of the sum of the summands.
 ii. When interpreting each of the summands and the sum in sign-magnitude width 4, the result represents the actual value of the sum of the summands.
 iii. When interpreting each of the summands and the sum in 2s complement width 4, the result represents the actual value of the sum of the summands.

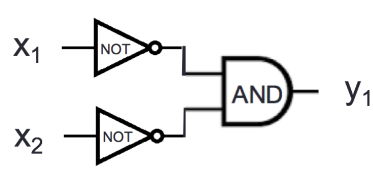
 (e) In binary fixed-width addition (adding one bit at time, using the usual column-by-column and carry arithmetic, and ignoring the carry from the leftmost column), we get:

$$\begin{array}{rl} 0110 & \text{first summand} \\ +0111 & \text{second summand} \\ \hline 1101 & \text{result} \end{array}$$

 Select all and only the true statements below:

 i. When interpreting each of the summands and the result in binary fixed-width 4, the result represents the actual value of the sum of the summands.
 ii. When interpreting each of the summands and the sum in sign-magnitude width 4, the result represents the actual value of the sum of the summands.
 iii. When interpreting each of the summands and the sum in 2s complement width 4, the result represents the actual value of the sum of the summands.

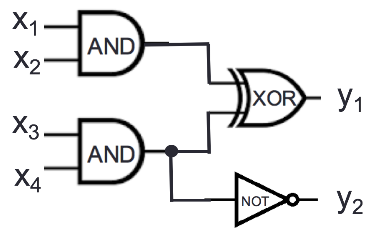
4. (a) Consider the logic circuit



Calculate the value of the output of this circuit (y_1) for each of the following settings(s) of input values.

i. $x_1 = 1$, $x_2 = 1$
ii. $x_1 = 1$, $x_2 = 0$
iii. $x_1 = 0$, $x_2 = 1$
iv. $x_1 = 0$, $x_2 = 0$

(b) Consider the logic circuit



For which of the following settings(s) of input values is the output $y_1 = 0$, $y_2 = 1$. (Select all and only those that apply.)

i. $x_1 = 0$, $x_2 = 0$, $x_3 = 0$, and $x_4 = 0$
ii. $x_1 = 1$, $x_2 = 1$, $x_3 = 1$, and $x_4 = 1$
iii. $x_1 = 1$, $x_2 = 0$, $x_3 = 0$, and $x_4 = 1$
iv. $x_1 = 0$, $x_2 = 0$, $x_3 = 1$, and $x_4 = 1$