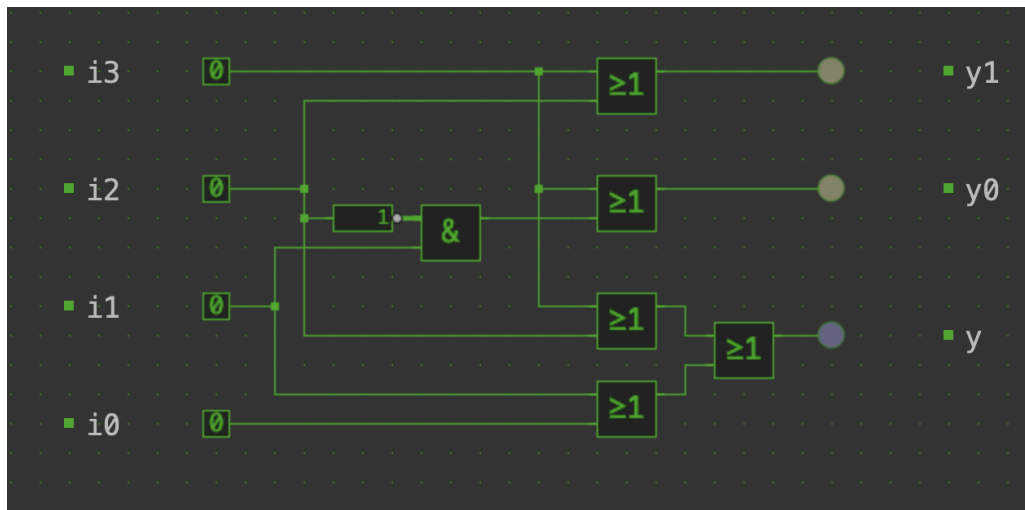


ICS 2022 Problem Sheet #8

Problem 8.1: digital circuit analysis

(1+1+2 = 4 points)

You are given the following digital circuit. The circuit may as well be found online at <http://simulator.io/board/pu8qlKwg1J/3> (but there is no guarantee that it persists).



- Write down the truth table defining the outputs y_0 , y_1 , and y .
- Write down short boolean expressions defining y_0 , y_1 , and y .
- Describe in your own words what the circuit is doing and how it might be used.

Problem 8.2: dice display

(2+1+1 = 4 points)

Too many students waiting inside the coffee bar to obtain drinks and snacks was found to be problematic and as a consequence the number of people waiting to be served got limited to seven. You got the task to create a display showing how many students are inside and you decided to build a display out of light emitting diodes (LEDs) that can be powered by a very tiny solar panel. Your display resembles the form of a dice with LEDs positioned as follows:

a b c
 d
 e f g

The numbers 0 to 7 are displayed as follows (a star indicates a LED producing light, a circle indices an LED currently off).

o o o	o o o	o o *	o o *
o	*	o	*
o o o	o o o	* o o	* o o
* o *	* o *	* * *	* * *
o	*	o	*
* o *	* o *	* * *	* * *

Your display is driven by three input lines x_2, x_1, x_0 indicating a binary number.

- Write a truth table defining the necessary boolean functions.
- Provide (simple) boolean expressions for the boolean functions.
- Create a digital circuit using <https://simulator.io/>.

Submit an image of your digital circuit and a link resolving to your digital circuit.

Problem 8.3: *decimal to binary and binary to decimal (haskell)* (1+1 = 2 points)

Implement a functions to convert decimal numbers into binary notation and back.

- Implement a function `dtob :: Int -> String` converting a non-negative integer number into a `String` (consisting of the characters '0' and '1') representing the integer number as a binary number. It is not necessary to handle negative integers in a meaningful way.
- Implement a function `btod :: String -> Int` converting a `String` (consisting of the characters '0' and '1') representing a binary number into the corresponding non-negative integer number. It is not necessary to handle unexpected strings in a meaningful way.

Submit your Haskell code as a plain text file. Below is a template file with a few unit test cases.

```
1  module Main (main) where
2
3  import Test.HUnit
4
5  -- | Convert a non-negative integer number into a String providing a
6  --    binary representation of the number.
7  dtob :: Int -> String
8  dtob _ = undefined
9
10 -- | Convert a String representing a non-negative integer number as a
11 --    binary number into an integer number.
12 btod :: String -> Int
13 btod _ = undefined
14
15 -- Below are some test cases...
16
17 dtobTests = TestList [ dtob 0 ~?= "0"
18                        , dtob 1 ~?= "1"
19                        , dtob 2 ~?= "10"
20                        , dtob 127 ~?= "1111111"
21                        , dtob 12345 ~?= "11000000111001"
22                        ]
23
24 btodTests = TestList [ btod "0" ~?= 0
25                       , btod "1" ~?= 1
26                       , btod "10" ~?= 2
27                       , btod "1111111" ~?= 127
28                       , btod "11000000111001" ~?= 12345
29                       ]
30
31 main = runTestTT $ TestList [ dtobTests, btodTests ]
```