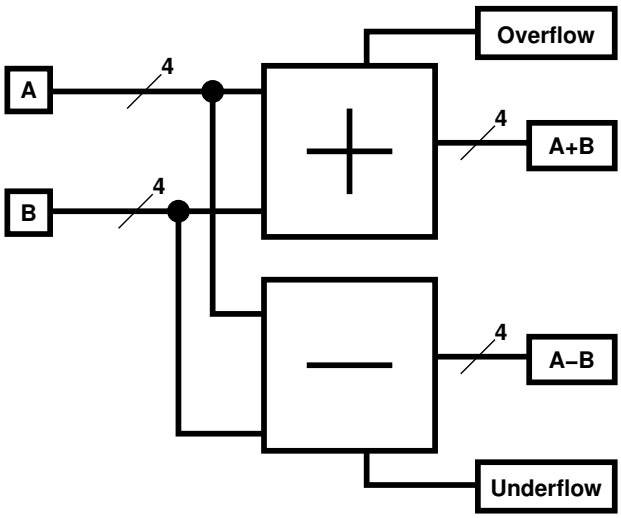


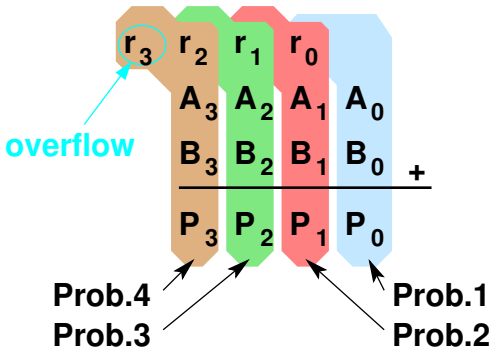
You are given an 8-input, 10-output circuit. `A` and `B` are numbers encoded in natural binary, using 4 bits: A_3, A_2, A_1, A_0 *and* B_3, B_2, B_1, B_0.
The outputs are also encoded in natural binary and defined as follows:
A+B= (P_3,P_2,P_1,P_0) *and* **A-B=** (M_3,M_2,M_1,M_0).
Finally, the overflow **O** and the underflow **U** outputs indicate occurrence of exceeding capacity. Your task is to design a gate-level schematic of the circuit.

Questions:

1. Draw a truth table of the adder.

   We would **never** manage to draw a truth table with $2^8 = 256$ rows!
   We shall use the *divide-and-conquer* method instead:

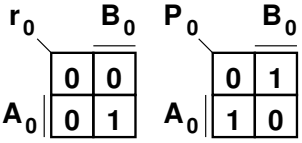
The "division" phase:



The first phase of the "conquest":

Prob.1:

A_0	B_0	r_0	P_0
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

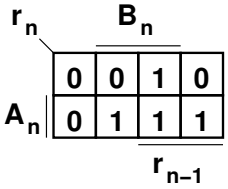


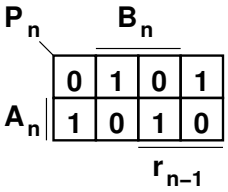
$r_0 = A_0 \cdot B_0$

$P_0 = A_0 \oplus B_0$

Prob.2:
Prob.3:
Prob.4:

r_{n-1}	A_n	B_n	r_n	P_n
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

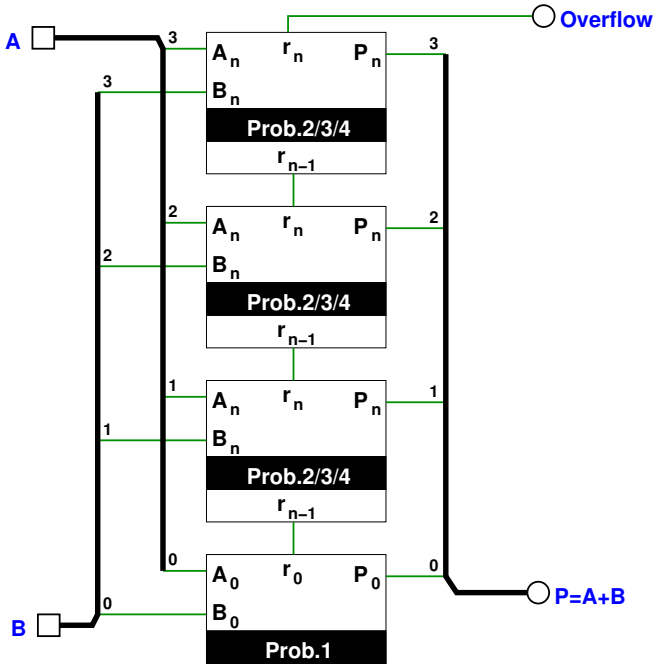


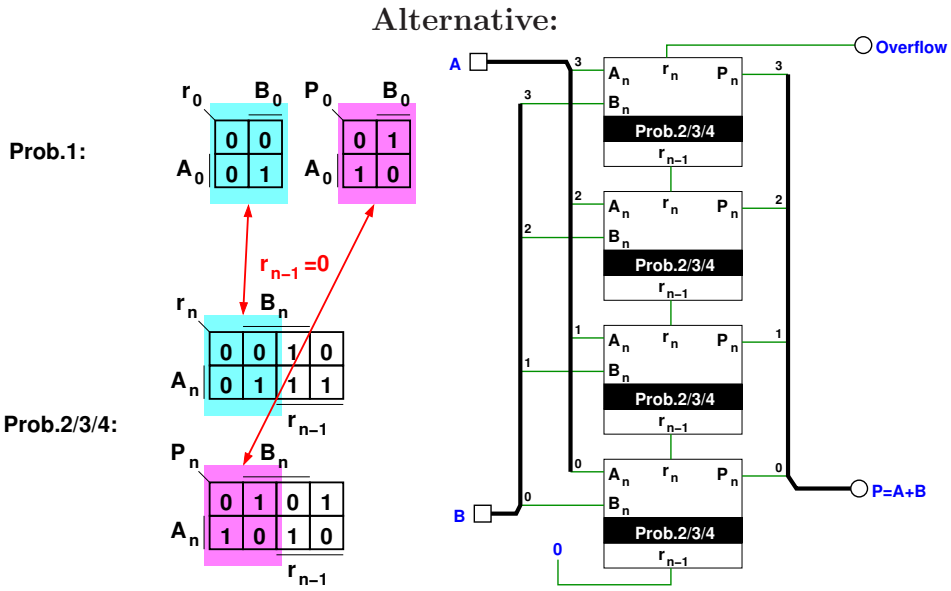


$r_n = A_n \cdot B_n + r_{n-1} \cdot A_n + r_{n-1} \cdot B_n$

$P_n = A_n \oplus B_n \oplus r_{n-1}$

The second phase of the "conquest":



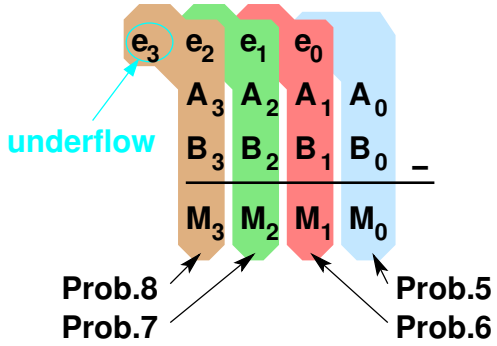


2. Using Logisim, design and verify a gate-level schematic of the adder.

   See the logisim file.

3. Draw a truth table of the subtractor.

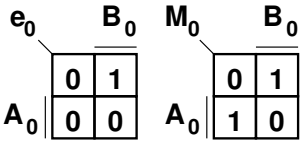
   Again we use the *divide-and-conquer* method:



The first phase of the "conquest":

Prob.5:

A_0	B_0	e_0	M_0
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

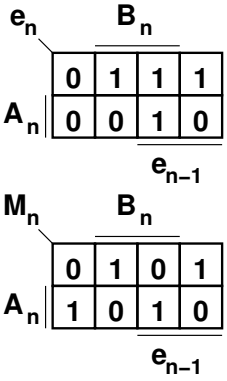


$e_0 = \overline{A_0} \cdot B_0$

$M_0 = A_0 \oplus B_0$

Prob.6:
Prob.7:
Prob.8:

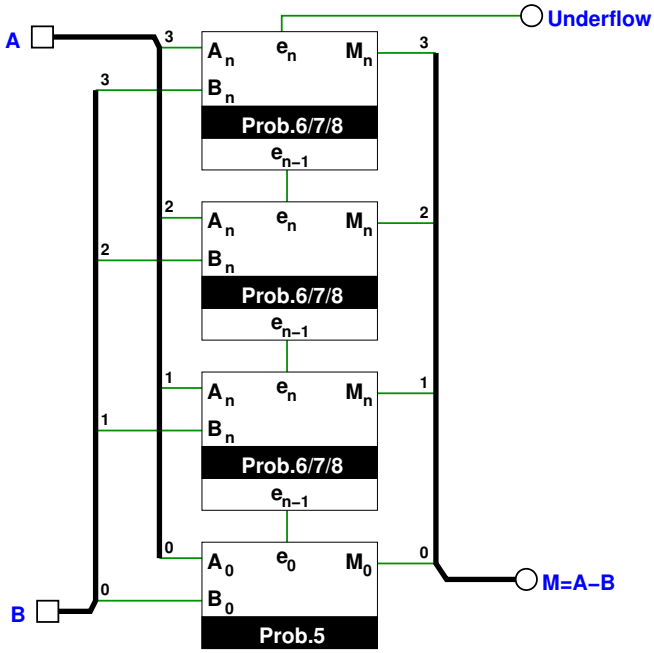
e_{n-1}	A_n	B_n	e_n	M_n
0	0	0	0	0
0	0	1	1	1
0	1	0	0	1
0	1	1	0	0
1	0	0	1	1
1	0	1	1	0
1	1	0	0	0
1	1	1	1	1

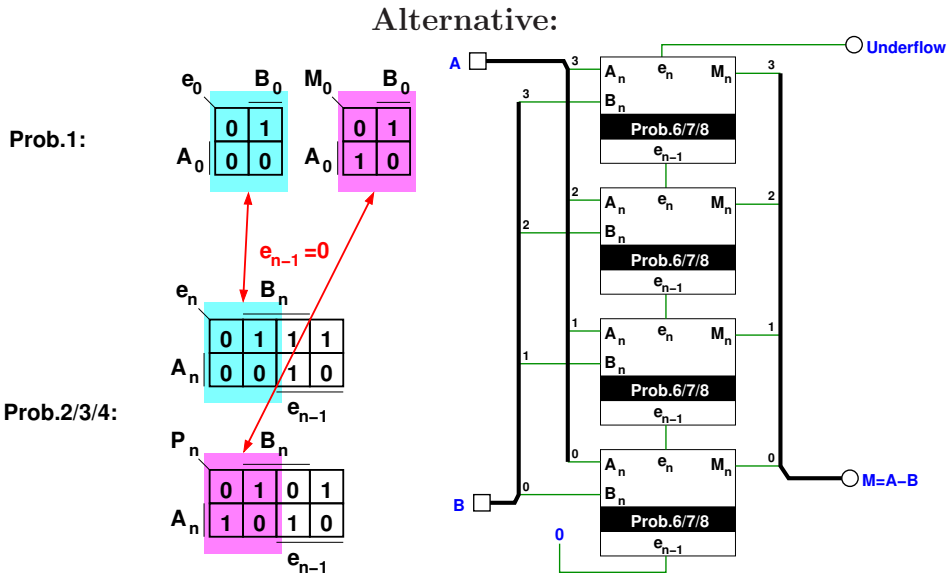


$e_n = \overline{A_n} \cdot B_n + e_{n-1} \cdot \overline{A_n} + e_{n-1} \cdot B_n$

$M_n = A_n \oplus B_n \oplus e_{n-1}$

The second phase of the "conquest":

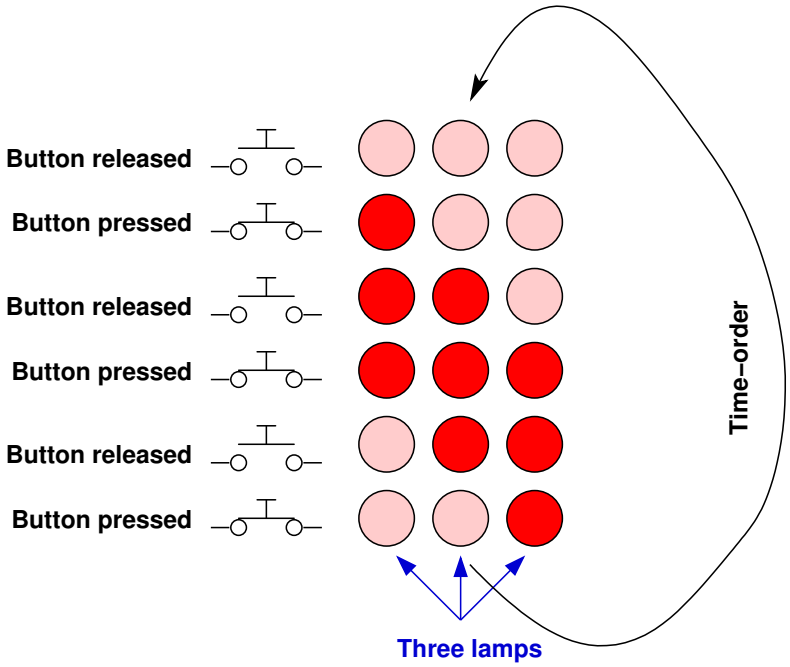




4. Using Logisim, design and verify a gate-level schematic of the subtractor.

   See the logisim file.

The purpose of this exercise is to design a controller circuit for a three-light-system. The state of the three lights changes on each press or release of the button, according to the scheme below:



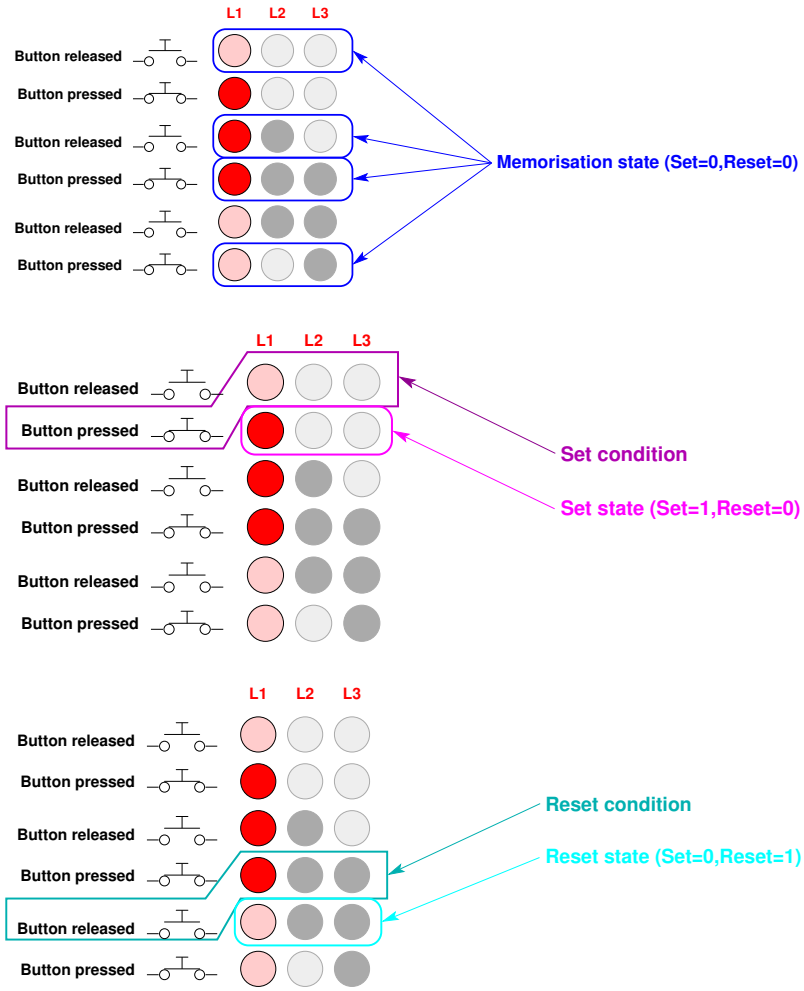
Tasks to complete:

5. Implement a system with the described functionality in Logisim. You are only allowed to use simple logic gates, SR-latches, LEDs, and buttons.

   For each LED, we have one SR-latch. Let us recall the functionality of an SR-latch:

Set	Reset	function
0	**0**	Keeping the prior state
0	**1**	LED off
1	**0**	LED on

   We can now observe the behavior of a single LED:



The states and transition conditions need to be determined for all three LEDs. For the solution, see the logisim file.

6. Download your design to the GECKO4Education board. What do you observe?