Gan & Mira

#1 Simulation Practice

GOAL create 3:8 decoder

Solution implement decoder

in_2	in_1	in_0	out_7	out_6	out_5	out_4	out_3	out_2	out_1	out_0
0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

Apply OR of ANDs

↳ see that out_x is only 1 for one combination of in's

For each out_x write a AND rule

ex:

```
out[7] = in[2] & in[1] & in[0];
```

Add enable logic

↳ if ena on → normal
↳ if ena off → 0

} boolean truth table just another AND

ex:

```
out[7] = in[2] & in[1] & in[0] & ena;
```

Gah & Mira

#2 Conway's Game & Life – Cell Module

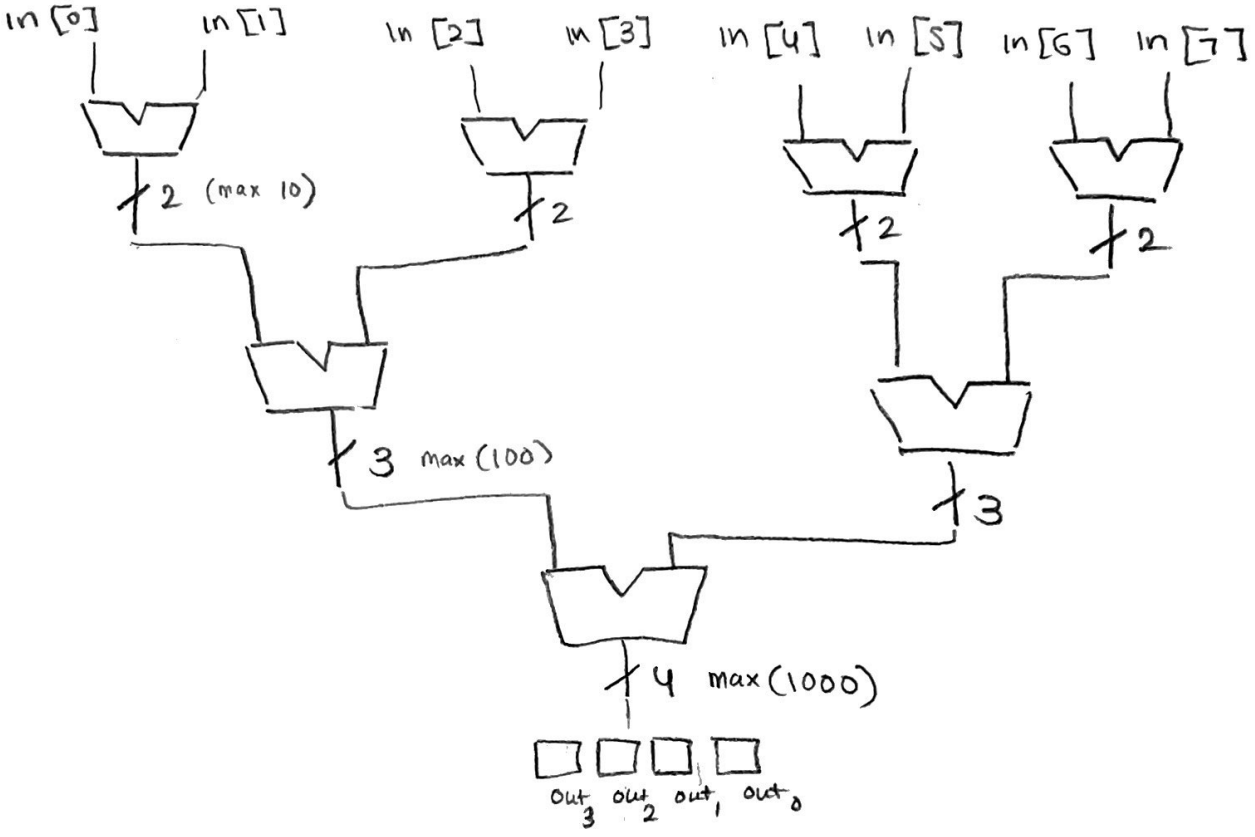
① implement adder

↳ use hierarchical design, as binary tree uses a minimum number of adders

↳ do not have to worry about overflow because most significant bit is the AND of previous location's bit (if most significant bit is ON then all other bits are off)

e.g. 1000 ✓ (all 8 neighbors on)

1111 × (invalid, impossible to have more than 8)



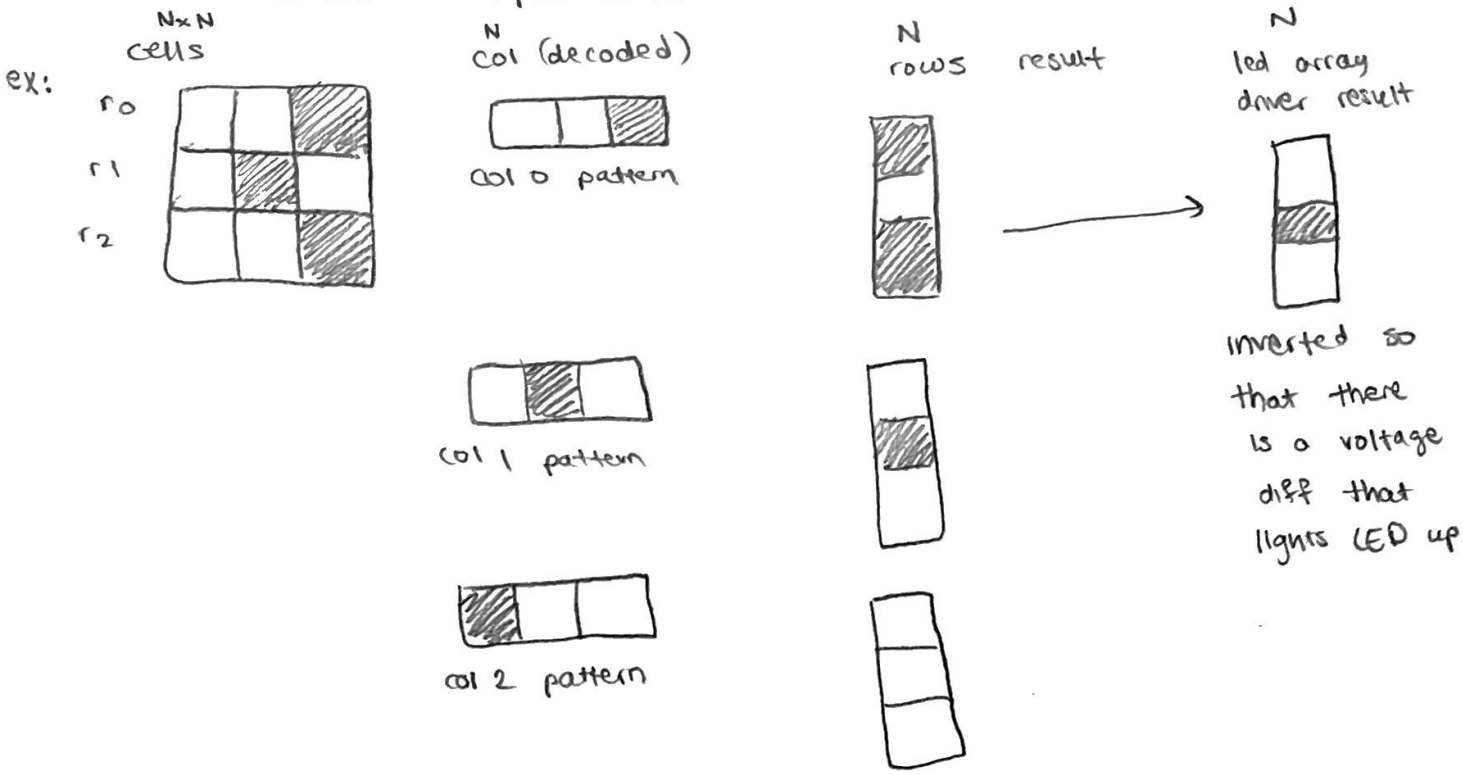
② logic to set new state based on old state & count of live neighbors

	out_3	out_2	out_1	out_0	state_0	state_1	
(3)	0	0	1	1	x	1	← 1 OR, write assign AND statement
(2)	0	0	1	0	1	1	← 1 OR, write assign AND statement
	all other cases					0	

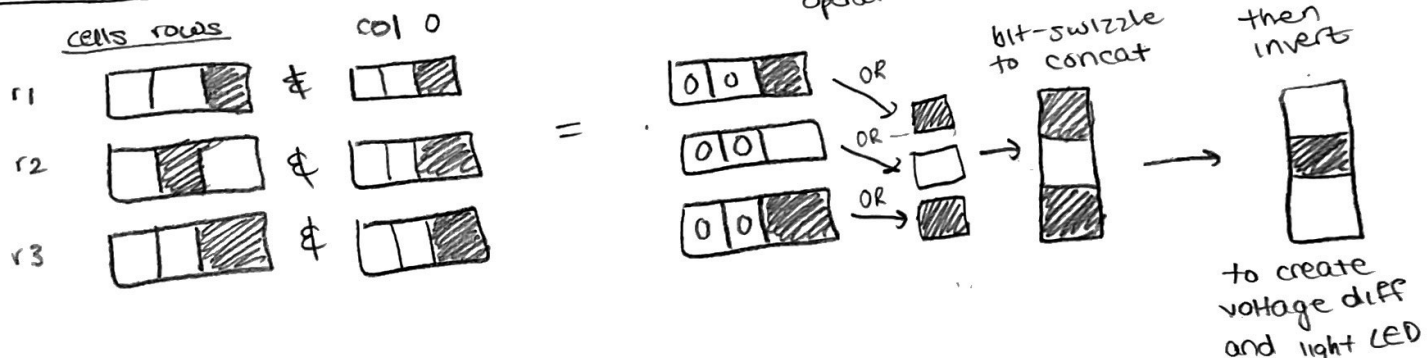
Gati & Mira

#4 LED Array Driver

① define behavior expectations



② pattern to calculate rows



③ use generator to handle all possible N

$$row[i] = NOT\left(\underset{bus}{OR}\left(cells\left[\begin{array}{c}corresponding\\ row\ of\ N\\ neighbors\end{array}\right] \ \& \ decoder\right)\right)$$

④ and add enable logic

```
assign rows[(N-1)-i] = N'(|(cells[(i+1)*N-1 : i*N] & x_decoded)) & ena;
```