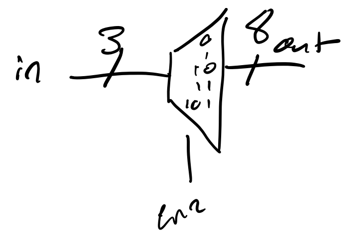
Lab 1: Conways Game of Life Notes

3 to 8 Decoder

en2		input [2]	input [1]	input [0]	output [7]	[6]	[5]	[4]	[3]	[2]	[1]	[0]
0			x					0				
1	0	0	0	0							0	1
	1	0	0	1						0	1	0
	2	0	1	0					0	1	0	
	3	0	1	1				0	1	0		
	4	1	0	0			0	1	0			
		1	0	1		0	1	0				
		1	1	0	0	1	0					
		1	1	1	1	0						

$out[7] = en2 \cdot in[2] \cdot in[1] \cdot in[0]$



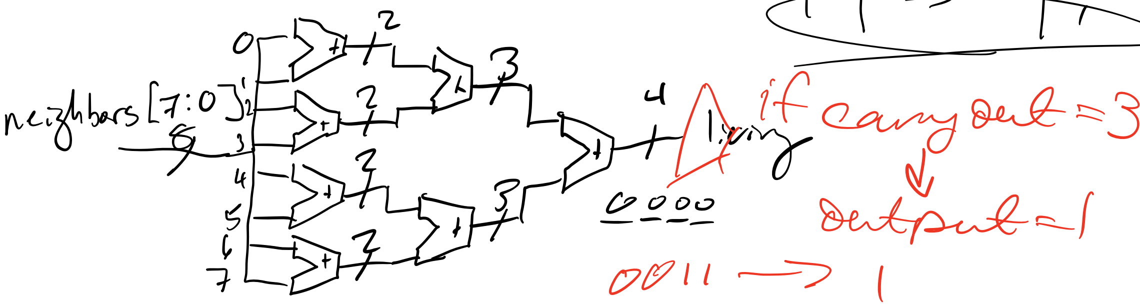
Conways Game

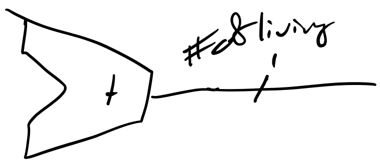
1. Any live cell w/ 2-3 living neighbors survives (becomes 1'b1 on posedge clk)
2. Any dead cell w/ 3 neighbors becomes alive (")
3. All others die/stay dead (1'b0 at pos clk edge)

2	3	4
1	me	5
0	7	6

neighbors[7:0] → ? → living

living	neighbors	living
0	3	1
1	2	1
1	3	1





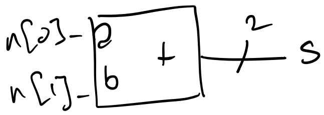
states transition table

Current State living	Inputs neighbors# (0-7)	Next State
L0	≤2	L0
L0	3	L1
L0	≥4	L0
L1	≤1	L0
L1	2-3	L1
L1	≥4	L0
X	≥4	L0
X	≤1	L0
X	3	L1
L1	2	L1
L0	2	L0

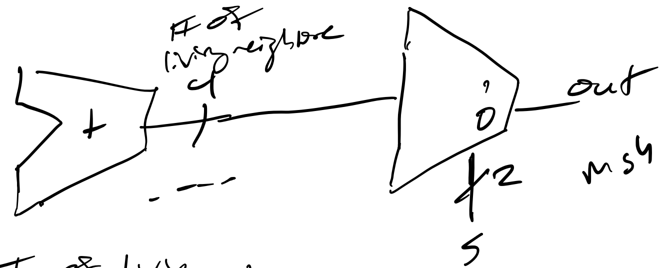
States: Living

L0 : DEAD 0

L1 : Living 1



Cin	a	b	s	Carry out



msb

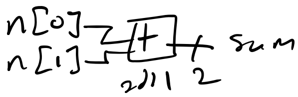
# of living neighbors	out

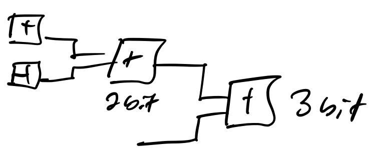
Lsb

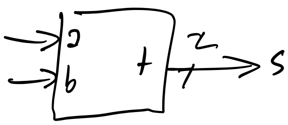
0011

0

Adder



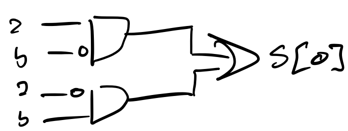




b a	s(d)	s(b)
00	0	00
01	1	01
10	1	10
11	2	11

01

10

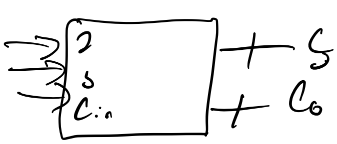


↳ aka carry output

```
always_comb begin
  //S[0]=(~a&b)|(~b&a);
  S[0] = a ^ b;
  S[1] = a & b;
end
```

$\overline{Cin}\ (A \oplus B)$

$\overline{Cin}\ (A \text{ XOR } B)$

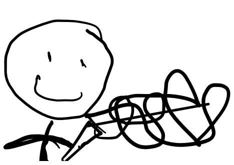


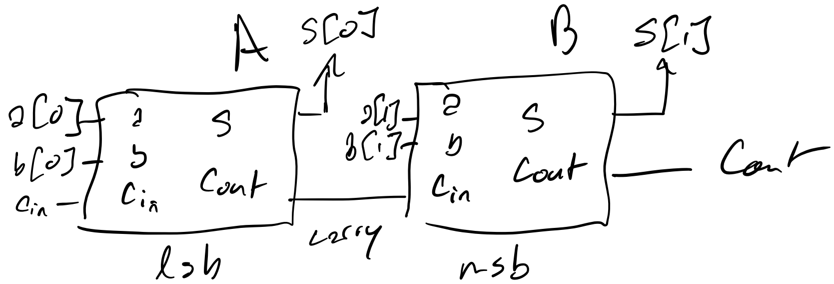
Cin	A	B	Co	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$S = \overline{Cin}\,\overline{A}B + \overline{Cin}\,A\overline{B} + Cin\,\overline{A}\,\overline{B} + Cin\,AB$

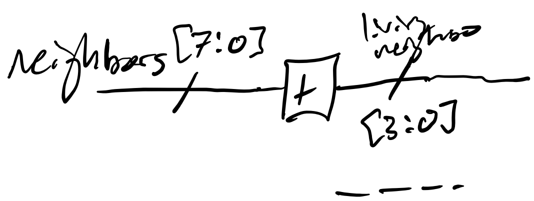
$Cin\ (A \text{ XNOR } B)$

$Cin\ (\overline{X \oplus Y})$





* Only use logic if you are creating a gate

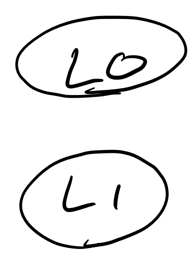


Current State living	Inputs neighbors# (0-7)	Next state
LO	≤2	LO
LO	3	L1
LO	≥4	LO
L1	≤1	LO
L1	2-3	L1
L1	≥4	LO
X	≥4	LO
X	≤1	LO
X	3	L1
L1	2	L1
LO	2	LO

States	Living	
LO	DEAD	0
L1	Living	1

2: 0010
3: 0011

2 living → li



Next State logic

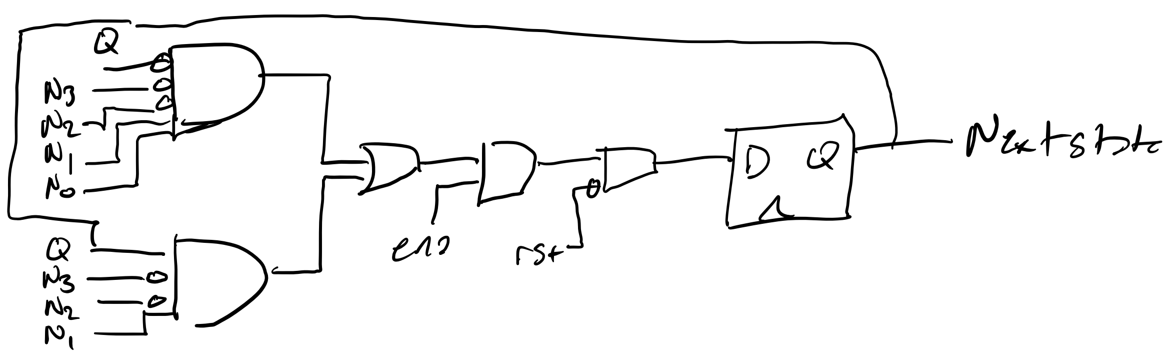
rst	ena	living [0]	neighbors [3:0] $N_3 N_2 N_1 N_0$	Next state
0	1	0	0000	0
			0001	0
			0010	0
			0011	1
			01xx	0
			1xxx	0
0	1	1	0000	0
			0001	0
			001x	1
			01xx	0
			1xxx	0
0	0	0 1	X	0 1
1	x	x	x	0

$D_0 = \overline{living}\,\overline{N_3}\,\overline{N_2}\,N_1 N_0\, ENA$
$+ living\,\overline{N_3}\,\overline{N_2}\,N_1\, ENA$

Living = Q

$D_0 = \overline{Q}\,\overline{N_3}\,\overline{N_2}\,N_1 N_0$
$+ Q\,\overline{N_3}\,\overline{N_2}\,N_1$

ENA (living



$V_F = 2.0V$ @ 20mA

$R = \frac{V}{I} = \frac{1.3V}{20mA} = 65\,\Omega$ Min

$65 \times 10 = 650\,\Omega$

$1K\,\Omega$

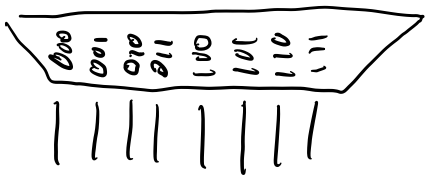
$V_T = V_F + V_R$

$V_R = 3.3V - 2.0V = 1.3V$

borked_cells_0[49:52] = 0

$[51:49] = 111 = 7_{10}$

$[48:0] = 0$



0 - 0

clog2(5)

x[2:0]

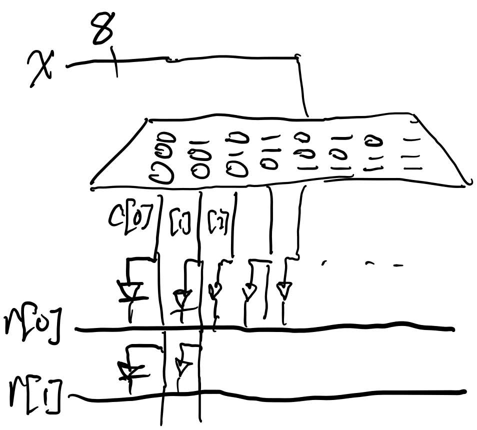
Power col high based on x_decoded

0 1 0000000

↳ power on col[1]

cells[24:0] represent the live status of each cell.

When x has 1, want led on → col = 1 row = 0

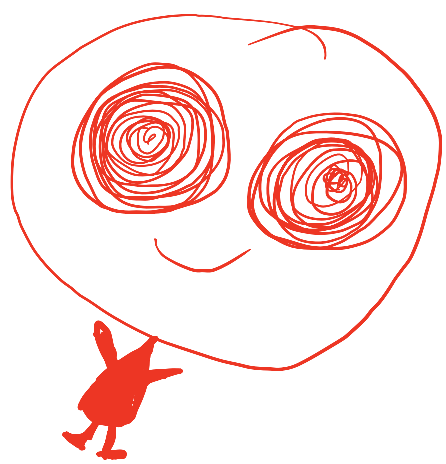


Inputs = cols & cells

out = rows

	c[0]	c[1]	c[2]
r[0]	0 (0 0)	1 (0 1)	2 (0 2)
r[1]	3 (1 0)	4 (1 1)	5 (1 2)
r[2]	6 (2 0)	7 (2 1)	8 (2 2)

2	6	7	8
1	3	4	5
0	0	1	2
	0	1	2



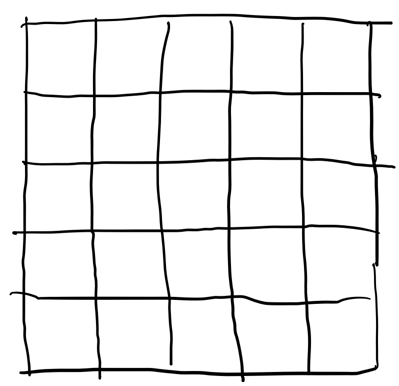
Inputs cols 0	1	2	cells 0	1	2	3	4	5	6	7	8	Outputs rows 0	1	2
0	0	1	x	x	0	x	x	0	x	x	1	0	0	1
			x	x	0	x	x	1	x	x	0	0	1	0
			x	x	0	x	x	1	x	x	1	0	1	1
			x	x	1	x	x	0	x	x	0	1	0	0
			x	x	1	x	x	0	x	x	1	1	0	1
			x	x	1	x	x	1	x	x	0	1	1	0
			x	x	1	x	x	1	x	x	1	1	1	1
0	1	0	x	0	x		0	x		1	x	0	0	1
				0			1			0		0	1	0
				0			1			1		0	1	1
				1			0			0		1	0	0
				1			0			1		1	0	1
				1			1			0		1	1	0
				1			1			1		1	1	1
1	0	0	0	x		0	x		1	x		0	0	1
			0			1			0			0	1	0
			0			1			1			0	1	1
			1			0			0			1	0	0
			1			0			1			1	0	1
			1			1			0			1	1	0
			1			1			1			1	1	1

-> ~rows after

r[0] = ~((c[2] & cell[2]) |
(c[1] & cell[1]) |
(c[0] & cell[0]));

r[1] = ~((c[2] & cell[5]) |
(c[1] & cell[4]) |
(c[0] & cell[3]));

r[0] = ~((c[2] & cell[8]) |
(c[1] & cell[7]) |
(c[0] & cell[6]));



x	cells	rows	cols
%2d	0x%0h	%b	%b

(0, 0) on

100	000	000
000	000	000
000	000	000

3x3

i = 0

j = 0

for x=0 x<3 x++

x = 0

cells = 0

Cell [0] = 1

Cells = 1 0 0
0 0 0
0 0 0

col=1 & row=0

print 1

if col = row, 0

x_decode [0] = 1 rest 0

rows [0] = 0 rest 1

col [0] = 1 rest 0

L0 = B[0] XOR B[1]

L1 = B[0] & B[1]

B[0]	B[1]	L[0]	L[1]
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

7	56	57	58	59	60	61	62	63
6	48							
5	40							
4	32							
3	24							
2	16							
1	8							
0	0	1	2	3	4	5	6	7
	0	1	2	3	4	5	6	7