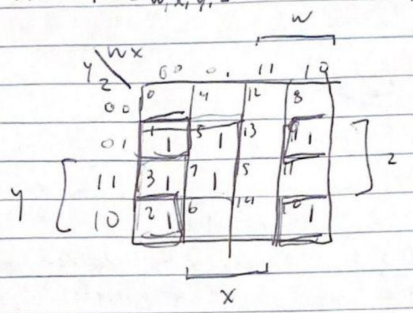
total: 18.5+p3

Logic Design HW #6

Tyler Youk

1) K Maps

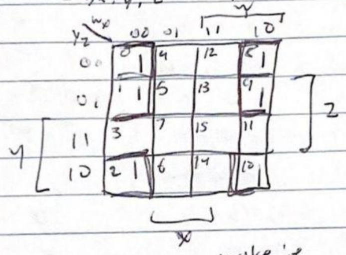
1. $F=\Sigma_{w,x,y,z}(1,2,3,5,7,9,10)$



$F=[w' * z]$
$+[x' * y * z']$
$+[x' * z * y']$

3/3

$F=\Sigma_{w,x,y,z}(0,1,2,8,9,10)$

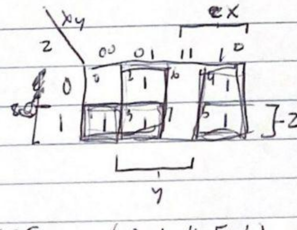


make it simple

$F=$
$+[x' * z' * y]$
$+[x' * y']$

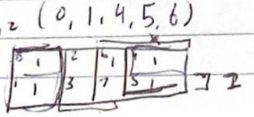
-0.5: You missed the wrap aro

2. $F=\Sigma_{x,y,z}(1,2,3,4,5)$



$F=[x' * z]$
$+[x' \cdot y]$
$+[x \cdot y']$

$F=\Sigma_{x,y,z}(0,1,4,5,6)$



$F=$ $+[x * z']$ $+[y']$

ECSE HW # 6

3.3.4 K maps

3. $F = \Sigma_{w,x,y,z}(5, 6, 7, 10, 11, 13, 14, 15)$

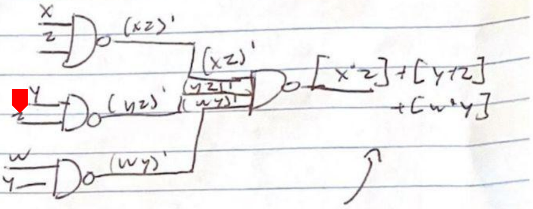
yz \ wx	00	01	11	10
00	0	4	12	8
01	1	5 1	13 1	9
11	3	7 1	15 1	11 1
10	2	6 1	14 1	10 1

W — X — Y — Z

~~F = [z * x]~~
~~+ [x * w * z]~~
~~+ [x * z * y']~~
~~+ [x * z * y]~~
~~+ [x * z * w]~~
~~+~~

$F = [x \cdot z]$
$+ [y \cdot x]$
$+ [w \cdot y]$

3 - 2 input NAND gate
1 - 3 input NAND gate



$[x \cdot z] + [y \cdot z]$
$+ [w \cdot y]$

-1.5

$\left[[x \cdot z]' \cdot [y \cdot z]' \cdot [w \cdot y]' \right]'$

4. 2 2-input NOR gates
1 3-input NOR gates

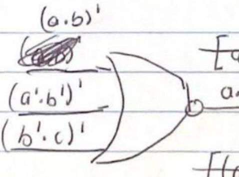
$P = a \cdot b + a' \cdot b' + b' \cdot c$

$[x + y + z]'$

~~[[a·b] + [a'·b'] + [b'·c]]'~~
~~[[a·b]' · [a'·b']' · [b'·c]']'~~
~~[a·b] + [a'+b'] + [b'·c]~~

~~[a'+b'] · [a'+b'] · [b'+c]~~

[On next page]



~~[a'+b']' + [b]' + [a+c']'~~
a·b + a'·b' + b'·c

~~[(a·b)' · (a'·b')' · (b'·c)']'~~
~~(a'+b') · (a+b) · (b+c')~~

~~[a'+b']'~~ ~~[a'+b']'~~ ~~(a'+b') · [b · [a+c']]~~
~~+[b]' + [a+c']'~~ ~~+ [b · [a+c']]'~~ ~~[a'+b'] · [b' + [a+c']']'~~

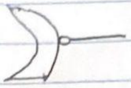
or $(x+y)'$

$x'\cdot y'$

3/6 incorrect circuit/fails to meet criteria

4. 2 2-input NOR gate

1 3-input NOR gate



$(x+y+z)'$

$x'\cdot y'\cdot z'$

$$F = a\cdot b + a'\cdot b' + b'\cdot c$$

$$F = (a'+b')' + (a+b)' + (b+c')'$$

$$\left[(a'+b')\cdot(a+b)\cdot(b+c')\right]'$$

$$\left[[a'+b']\cdot b\cdot[a+c']\right]$$

$$\left[[a'\cdot b] \cancel{+ [b'\cdot b]} \cdot [a+c']\right]'$$

$$\left[[a'\cdot b]^{0}\cdot[a+c']\right]'$$

~~$[a'\cdot b]' + [a+c']'$~~

~~$[a+b'] + [a'+c]$~~

use

$$\left[[a'\cdot b]\cdot[a+c']\right]'$$

$$\left[[a+b']'\cdot[a+c']\right]'$$

$0\cdot 0=0$ $1\cdot 0=0$

$1\cdot 1=1$ $1+0=1$

$1+1=1$

$0+0=0$

Truth table

a	b	c	F	
0	0	0	$0+1+0=1$	$x+y+z$
0	0	1	$0+1+1=1$	$x+y+z'$
0	1	0	$0+0+0=0$	$x+y'+z$
0	1	1	$0+0+0=0$	$x+y'+z'$
1	0	0	$0+0+0=0$	$x'+y+z$
1	0	1	$0+0+1=1$	$x'+y+z'$
1	1	0	$1+0+0=1$	$x'+y'+z$
1	1	1	$1+0+0=1$	$x'+y'+z'$

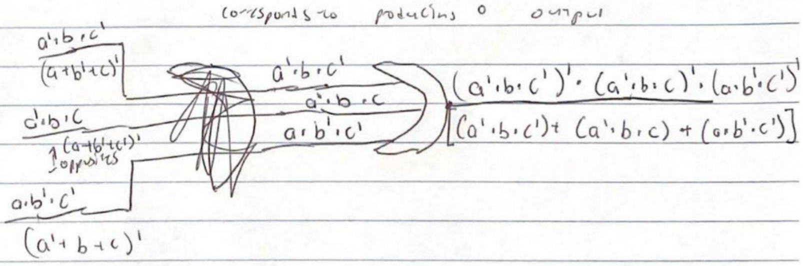
use maxterms then ~~take~~ use demorgan on individual

$$F = (x+y'+z)\cdot(x+y'+z')\cdot(x'+y+z)$$

$$F = (x'\cdot y\cdot z')'\cdot(x'\cdot y\cdot z)'\cdot(x\cdot y'\cdot z')$$

$$= (a'\cdot b\cdot c')'\cdot(a'\cdot b\cdot c)'\cdot(a\cdot b'\cdot c')'$$

corresponds to producing 0 output



5. 3 innut logic circuit

produce a 1-output when more ones than 0

001 produces 0-output

110 produced 1-output

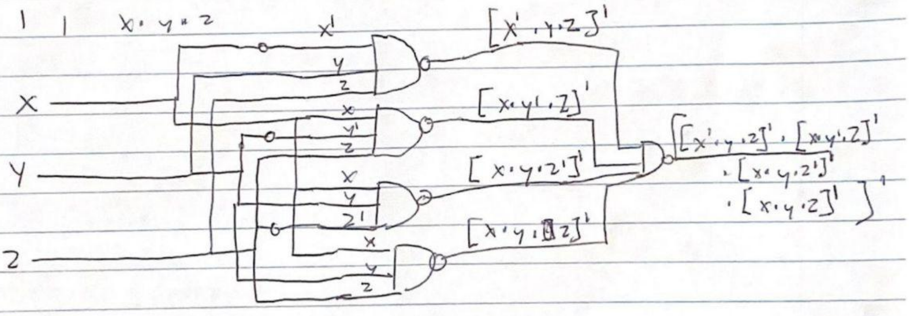
~~a b c~~ minsum - for 1s

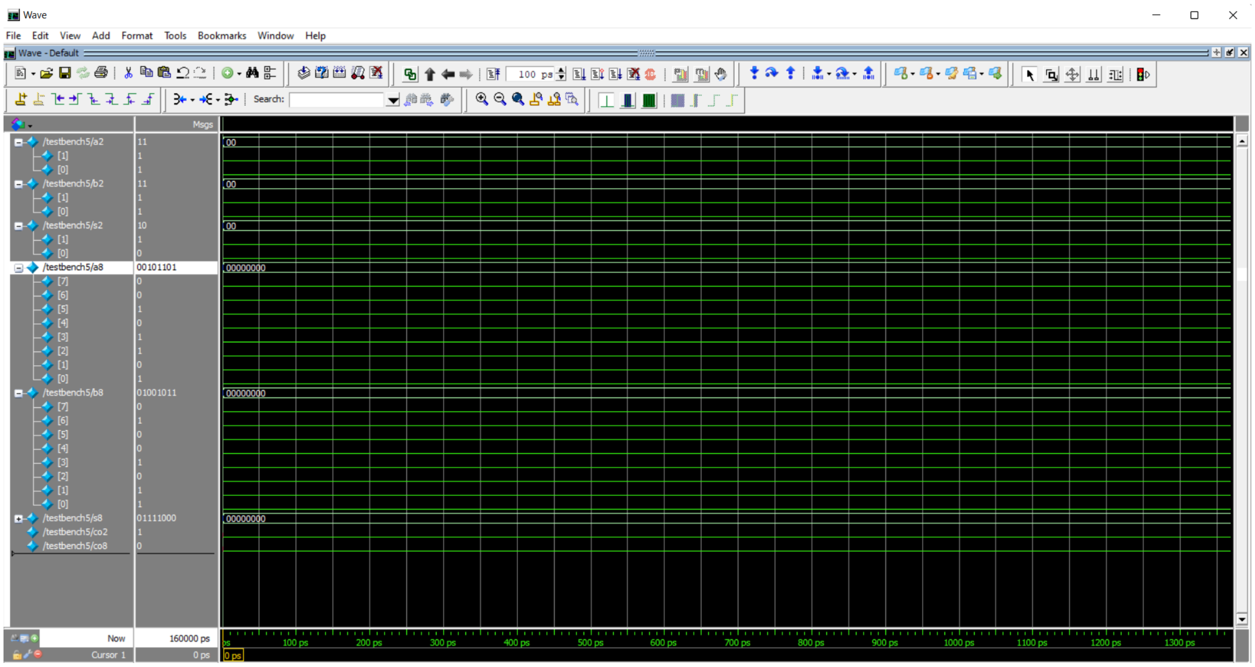
4/6 missing K-map: incorrect simplification

x	y	z	F	
0	0	0	0	$x' \cdot y' \cdot z'$
0	0	1	0	$x' \cdot y' \cdot z$
0	1	0	0	$x' \cdot y \cdot z'$
0	1	1	1	$x' \cdot y \cdot z$
1	0	0	0	$x \cdot y' \cdot z'$
1	0	1	1	$x \cdot y' \cdot z$
1	1	0	1	$x \cdot y \cdot z'$
1	1	1	1	$x \cdot y \cdot z$

$[X' \cdot y \cdot Z] + [x \cdot y' \cdot Z] + [x \cdot y \cdot z'] + [x \cdot y \cdot z]$

only use NAND gates





```
/testbench5/a2      11        00
    [1]             1
    [0]             1
/testbench5/b2      11        00
    [1]             1
    [0]             1
/testbench5/s2      10        00
    [1]             1
    [0]             0
/testbench5/a8      00101101  00000000
    [7]             0
    [6]             0
    [5]             1
    [4]             0
    [3]             1
    [2]             1
    [1]             0
    [0]             1
/testbench5/b8      01001011  00000000
    [7]             0
    [6]             1
    [5]             0
    [4]             0
    [3]             1
    [2]             0
    [1]             1
    [0]             1
/testbench5/s8      01111000  00000000
/testbench5/co2     1
/testbench5/co8     0

Now                 160000 ps
Cursor 1            0 ps      0 ps
```

Code:

```
//Combining rc_adder_slice and rc_adder4 due to compiling issues

module cla_adder # (

parameter N = 8
) (
	input logic [N-1:0] a, b,
	input logic c_in,
	output logic [N-1:0] s,
	output logic c_out
);

	logic [N-1:0] p, g;
	logic [N:0] c;

	assign p = a ^ b;
	assign g = a & b;

for(genvar i = 0; i <= N; i++) begin

if (i == 0)
	assign c[i] = c_in;
else
```

```
        assign c[i] = g[i - 1] | p[i-1] & c[i-1];
end

        assign s = c ^ p;
        assign c_out = c[N];

endmodule



// Using testbenchHWLab6 name due to issues with testbench_lab3 naming errors

`timescale 1ns/10ps
module testbenchHWLab6 ();

        logic [1:0] a2, b2, s2;
        logic [7:0] a8, b8, s8;
        logic co2, co8;

//Instantiation of cla adder of 2 bits
cla_adder #(.N(2)) UUT2 (
.a(a2),
.b(b2),
.c_in(1'b0),
.s(s2),
.c_out(co2)
);

//Instantiation of cla adder of 8 bits
cla_adder #(.N(8)) UUT8 (
.a(a8),
.b(b8),
.c_in(1'b0),
.s(s8),
.c_out(co8)
);

        initial begin

                a2 = 0;
                forever
                        #10 a2++;
        end
```

```
    initial begin

        a8 = 0;
        forever
            #10 a8+=3;
    end


    initial begin

        b2 = 0;
        forever
            #40 b2++;
    end


    initial begin

        b8 = 0;
        forever
            #10 b8+=5;
    end

    initial
        # 160 $finish();

endmodule
```