John Hubbard UCSC 30207 (p.1) 24 Jan 2015
Hw #2

Problem 2.6.1 – Part 1: Show that lutmask 8000 implements a 4-input AND gate.

ⓐ – The lutmask convention is: bit N of the lutmask is set if f(a, b, c, d) = 1 for the N^{th} entry of f's truth table. So N == 4'b $x_3 x_2 x_1 x_0$ where
$a = x_3$
$b = x_2$
$c = x_1$
$d = x_0$

A lutmask of 0x8000 means that $\overset{\text{(only)}}{\wedge}$ bit 15 is set, in a truth table that ranges from 0 to 15. That means:

$N = 15 = 4'b1111 = x_3 x_2 x_1 x_0$

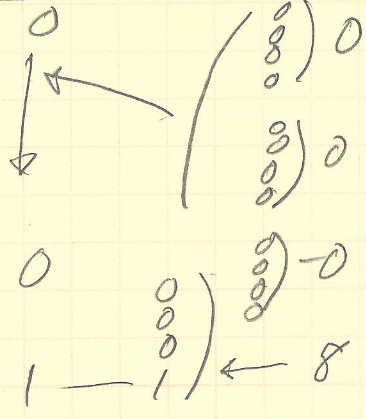
so
$$\left.\begin{array}{l} x_3 = 1 = a \\ x_2 = 1 = b \\ x_3 = 1 = c \\ x_4 = 1 = d \end{array}\right)$$
f is 1 only for abcd = 4b'1111, when is the same truth table as f = a·b·c·d

ⓑ – Another way of showing this is to show both truth tables:

a	b	c	d	f = abcd
0	0	0	0	0
0	0	0	1	↓
		⋮		
1	1	1	0	0
1	1	1	1	1

same ⟷

a	b	c	d	f = lutmask 0x8000
0	0	0	0	0
0	0	0	1	↓
0	0	1	0	
		⋮		
				0
1	1	1	1	1



John Hubbard UCSC 30207 (p. 2) 24 Jan 2015
HW #2

Problem 2.6.1 – Part 2: Implement a 4-input XOR gate using LUT4.

a b c d	f = a^b^c^d		
0 0 0 0	0		
0 0 0 1	1	6	← lutmask low bit
0 0 1 0	1		
0 0 1 1	0		
0 1 0 0	1		
0 1 0 1	0	9	
0 1 1 0	0		
0 1 1 1	1		
1 0 0 0	1		
1 0 0 1	0	9	
1 0 1 0	0		
1 0 1 1	1		
1 1 0 0	0		
1 1 0 1	1	6	← lutmask high bits
1 1 1 0	1		
1 1 1 1	0		

$\boxed{\text{lutmask} = 16'h\ 6996}$

= 0x6996

```
// Verilog:

module (f, a, b, c, d);
    input  a, b, c, d;
    output f;

    LUT4 #(16'h 6996) u(f, a, b, c, d);

endmodule
```

John Hubbard UCSC 30207 (p.3) 24Jan2015
HW #2

Problem 2.6.1 – Part 3: Implement this function using a LUT4:

$f = abcd + \bar{a}\bar{b}\bar{c}\bar{d} + a\bar{d} + \bar{a}b\bar{c} + \bar{b}d$
$+ \bar{a}b\bar{c} + ab\bar{c}d + bcd + \bar{a}c\bar{d}$

– Each term tells which bits are set in the lutmask

a b c d	f	
0 0 0 0	1	$\bar{a}\bar{b}\bar{c}\bar{d}$
0 0 0 1	1	$\bar{b}d$
0 0 1 0	1	$\bar{a}c\bar{d}$
0 0 1 1	1	$\bar{b}d$
0 1 0 0	1	$\bar{a}b\bar{c}$
0 1 0 1	1	$\bar{a}b\bar{c}$
0 1 1 0	1	$\bar{a}c\bar{d}$
0 1 1 1	1	bcd
1 0 0 0	1	$a\bar{d}$
1 0 0 1	1	$\bar{b}d$
1 0 1 0	1	$a\bar{d}$
1 0 1 1	1	$\bar{b}d$
1 1 0 0	1	$a\bar{d}$
1 1 0 1	1	$ab\bar{c}d$
1 1 1 0	1	$a\bar{d}$
1 1 1 1	1	$abcd$, bcd

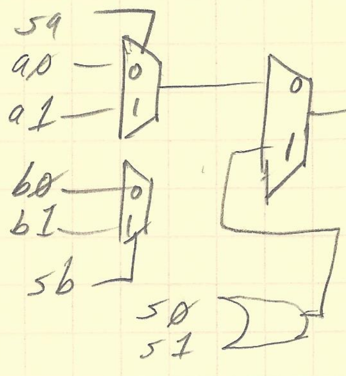
lutmask = 0x FFFF
= $\boxed{16'h FFFF}$

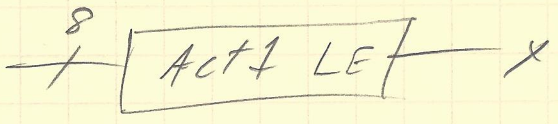
// Verilog:

```
module (f, a, b, c, d);
    input a, b, c, d;
     output f;

    LUT4 #(16'h FFFF) u(f, a, b, c, d);
endmodule

// or:   assign f = 1;
```

John Hubbard UCSC 30207 (p.4) 24 Jan 2015
HW #2

Problem 2.6.2 – Part 1: Implement all possible 2-input functions using Actel Act1 block:





	Function	a0	a1	sa	b0	b1	sb	s0	s1
1	xor(a,b)	0	1	b	1	0	b	a	0
2	and(a,b)	0	a	b	x	x	x	0	0
3	nand(a,b)	1	1	1	1	0	a	b	0
4	nor(a,b)	1	0	b	0	0	0	a	0
5	or(a,b)	0	0	0	1	1	1	a	b
6	xnor(a,b)	1	0	b	0	b	1	a	0
7	not(a)	1	0	a	x	x	x	0	0
8	not(b)	1	0	b	x	x	x	0	0
9	buf(a)	0	1	a	x	x	x	0	0
10	buf(b)	0	1	b	x	x	x	0	0
11	0 (contradiction)	0	0	0	0	0	0	0	0
12	a > b	a	0	b	x	x	x	0	0
13	a < b	b	0	a	x	x	x	0	0
14	a ≥ b	1	a	b	x	x	x	0	0
15	a ≤ b	1	b	a	x	x	x	0	0
16	1 (tautology)	1	1	1	1	1	1	1	1

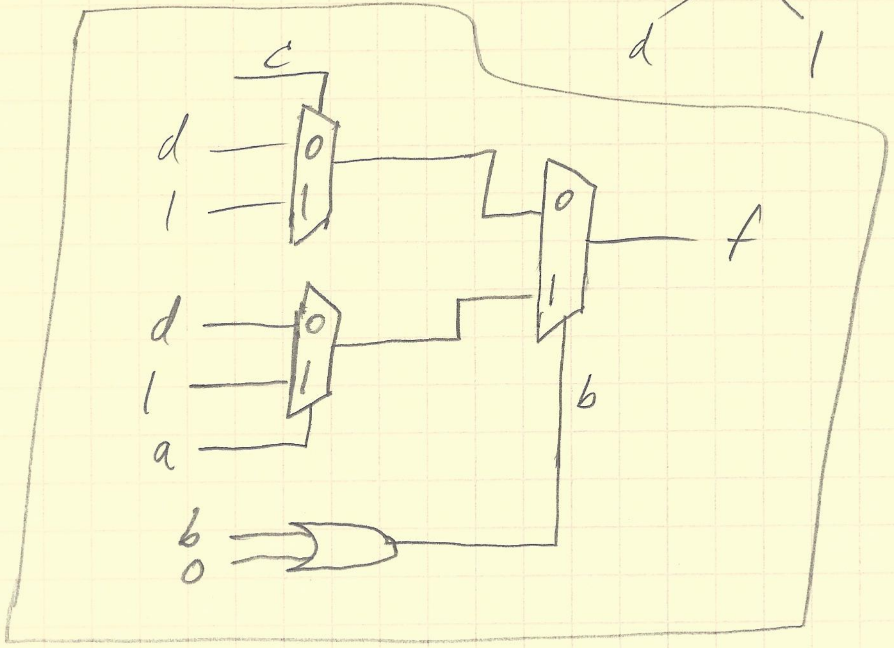
John Hubbard UCSC 30207 (p.5) 24 Jan 2015

HW #2

Problem 2.6.2 – Part 2: Implement this function using an Act 1 block:

$$f = ab + \bar{b}c + d$$

$$\begin{tikzcd}
 & & f \arrow[dl, dash, "b=0"'] \arrow[dr, dash, "b=1"] & & \\
 & c+d \arrow[dl, dash, "c=0"'] \arrow[d, dash, "c=1"] & & a+d \arrow[d, dash, "a=0"'] \arrow[dr, dash, "a=1"] & \\
d & 1 & & d & 1
\end{tikzcd}$$



c
d
1
0
1
d
1
0
1
a
0
1
b
f
b
0