Lab #3 (Combinational Logic and ALU)

Name: Raul Aguilar

Date: September 18, 2020

Converting between Octal and Decimal Numbers

1. Convert **1337**$_8$ to decimal (base 10) Use sum of expansion of products (don't skip steps!)
$(1\times8^3)+(3\times8^2)+(3\times8^1)+(7\times8^0)$ $512+192+24+7$ $512 + 192 + 24 + 7 = 735$ $1337_8 = 735_{10}$
2. Convert **71**$_{10}$ to octal (base 8) Use the Double-Dabble method of successive divsion
71/8 8 r7 8/8 1 r0 1/8 0 r1 $71_{10} = 107_8$

3. What file permissions does the octal number **5** exhibit?	4. What file permissions does the octal number **3** exhibit?
read and execute	write and execute

Converting between Hexadecimal and Decimal Numbers

5. Convert **AC34D1**$_{16}$ to decimal (base 10)
   Use sum of expansion of products (don't skip steps!)

$$(10 \times 16^5) + (12 \times 16^4) + (3 + 16^3) + (4 + 16^2) + (13 \times 16^1) + (1 \times 16^0)$$

$11{,}285{,}713_{10}$

6. Convert **365**$_{10}$ to hexadecimal (base 16)
   Use the Double-Dabble method of successive divsion

365/16	22	r 13
22/16	1	r 6
1/16	0	r 1

$16D_{16}$

Adding Signed Binary Numbers (with Two's Complement)

7. Add -8 + 5 in 4-digit binary.
   First convert to Two's Complement binary, then compute the sum.

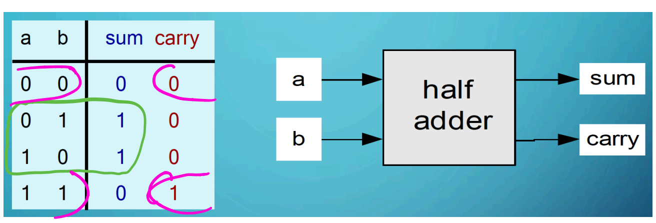
```
 1000
 0111
    1
 ----
 1000
```

```
  1000
+ 0101
  ----
  1101
```

1101_2

Half Adder (Two Inputs) Design

8. Write the Boolean function for the outputs (sum and carry). Use K-maps if needed.
   Then write the HDL code.



sum(a, b) =

carry(a, b) =

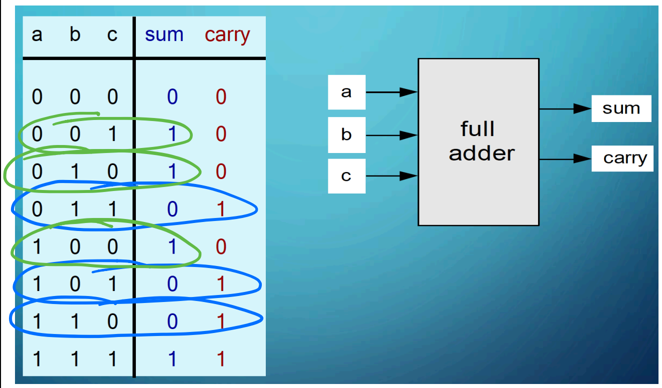
```
CHIP HalfAdder {
   IN a, b;
   OUT sum, carry;

   PARTS:
   And(a, b, carry)
   Xor (a, b, sum)

}
```

Full Adder (Three Input) Design

9. Write the Boolean function for the outputs (sum and carry). Use K-maps if needed. Then write the HDL code.



a	b	c	sum	carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

sum(a, b, c) =

carry(a, b, c) =

```
CHIP FullAdder {
   IN a, b, c;
   OUT sum, carry;

   PARTS:

}
```

Implementing the ALU Design (with signed two's complement numbers)

These bits instruct how to pre-set the x input		These bits instruct how to pre-set the y input		This bit selects between + / And	This bit inst. how to post-set out	Resulting ALU output
zx	nx	zy	ny	f	no	out=
if zx then x=0	if nx then x=!x	if zy then y=0	if ny then y=!y	if f then out=x+y else out=x And y	if no then out=!out	f(x,y)=
1	0	1	0	1	0	0
1	1	1	1	1	1	1
1	1	1	0	1	0	-1
0	0	1	1	0	0	x
1	1	0	0	0	0	y
0	0	1	1	0	1	!x
1	1	0	0	0	1	!y
0	0	1	1	1	1	-x
1	1	0	0	1	1	-y
0	1	1	1	1	1	x+1
1	1	0	1	1	1	y+1
0	0	1	1	1	0	x-1
1	1	0	0	1	0	y-1
0	0	0	0	1	0	x+y
0	1	0	0	1	1	x-y
0	0	0	1	1	1	y-x
0	0	0	0	0	0	x&y
0	1	0	1	0	1	x|y

f(x,y) =	0	**"1010"** -6	**"0001"** 1		**f(x,y) =**	1	**"1010"**	**"0001"**
zx =	1	**"0000"**			**zx =**			
nx =	0	**"0000"**			**nx =**			
zy =	1		**"0000"**		**zy =**			
ny =	0		**"0000"**		**ny =**			
f =	1	**"0000"**			**f =**			
no =	0	**"0000"** [0]			**no =**			
f(x,y) =	-1	**"1010"**	**"0001"**		**f(x,y) =**	x	**"0100"**	**"0101"**
zx =					**zx =**			
nx =					**nx =**			
zy =					**zy =**			
ny =					**ny =**			
f =					**f =**			
no =					**no =**			

f(x,y) =	y	**"1010"**	**"0011"**		**f(x,y) =**	!x	**"1010"**	**"0101"**
zx =					**zx =**			
nx =					**nx =**			
zy =					**zy =**			
ny =					**ny =**			
f =					**f =**			
no =					**no =**			
f(x,y) =	!y	**"1010"**	**"0101"**		**f(x,y) =**	"-x"	**"0010"**	**"1000"**
zx =					**zx =**			
nx =					**nx =**			
zy =					**zy =**			
ny =					**ny =**			
f =					**f =**			
no =					**no =**			
f(x,y) =	"-y"	**"1010"**	**"0001"**		**f(x,y) =**	x+1	**"0001"**	**"0001"**
zx =					**zx =**			
nx =					**nx =**			
zy =					**zy =**			
ny =					**ny =**			
f =					**f =**			
no =					**no =**			
f(x,y) =	y+1	**"1010"**	**"1111"**		**f(x,y) =**	x-1	**"0110"**	**"0001"**
zx =					**zx =**			
nx =					**nx =**			
zy =					**zy =**			
ny =					**ny =**			
f =					**f =**			
no =					**no =**			
f(x,y) =	y-1	**"0001"**	**"1111"**		**f(x,y) =**	x+y	**"0010"**	**"0101"**
zx =					**zx =**			
nx =					**nx =**			
zy =					**zy =**			
ny =					**ny =**			
f =					**f =**			
no =					**no =**			

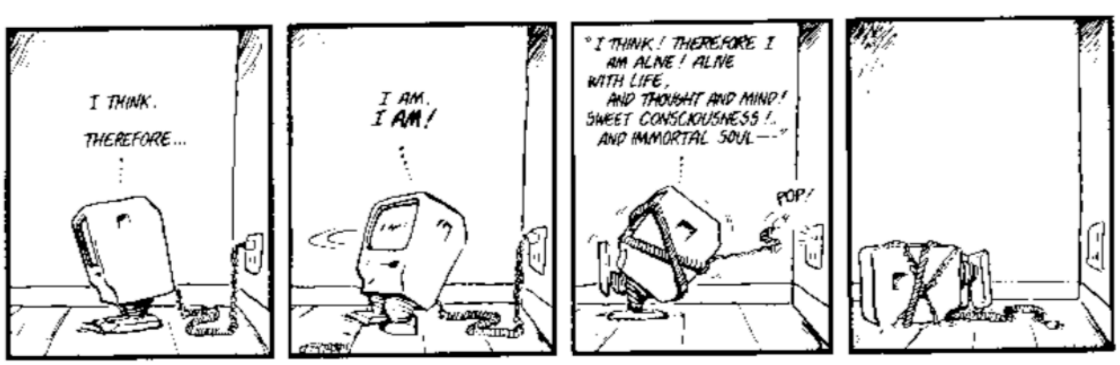
f(x,y) =	x-y	**"0111"**	**"0010"**		**f(x,y) =**	y-x	**"1101"**	**"1111"**
zx =					**zx =**			
nx =					**nx =**			
zy =					**zy =**			
ny =					**ny =**			
f =					**f =**			
no =					**no =**			
f(x,y) =	x&y	**"1011"**	**"1000"**		**f(x,y) =**	x|y	**"1111"**	**"1010"**
zx =					**zx =**			
nx =					**nx =**			
zy =					**zy =**			
ny =					**ny =**			
f =					**f =**			
no =					**no =**			

```
My dear creative, emotional, sometimes foolish, opinionated human,

You should now see that the characteristics of binary numbers in the
two's complement system coupled with a combination of four simple
binary/Boolean operations (zeroing, bitwise negation, adding, or'ing)
provides us with at least eighteen simple arithmetic functions.

true,
Banana Jr. 2000

PS. Now go build your ALU.
```



Bloom County Babylon by Berke Breathed