- addition / subtraction
- carry / overflow
- negative #s!
- Logic Gates
- noise Margin
- Skills Review due ~~Sat~~ Next LSN
- Fill in the blank exercise!

Decimal	Binary	Hexadecimal
2006	0111 1101 0110	7D6
109	1101101	6D
57,324	1101 1111 1110 1100	DFEC

Decimal	Binary	Hex
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Addition

Carries → 1 1 1
1234
+6789
8032

1 1 ← carries, even in Binary
10111
+00110
11101

Subtraction

~~10010~~
~~-01101~~
~~01~~

```
 0 2 2 0 2
 1 0 0 1 0
-0 1 1 0 1
----------
 0 0 1 0 1
```

what happens with this?

```
 1100
-1101
```
⇒ negative #

Negative #s in Binary:

1) Signed Magnitude

7: 0111

-7: 1111

Problems?

- you have to be told if it is signed/unsigned
- Two representations for one binary #
- Addition doesn't work!

```
 0101     5
+0110     6
-----    --
 1011    11
```

-3_{10} in signed magnitude

This is why we don't use signed magnitude in computers

old odometers on vehicles had 6 digits

999,999
000 000 ← rolls over
000001

⇒ this is similar to 2s compliment

2) 2s compliment
   a) flip all bits
   2) add one

Find 2s complement f/ -2?

```
       0010
flip → 1101
     +    1
     ------
       1110
```

1101 → -3
1110 → -2
1111 → -1
0000 → 0
0001 → 1
0010 → 2

Unsigned Binary Ranges

N-Bits: $0 \rightarrow 2^N - 1$

$b=4 \Rightarrow 0 \rightarrow 15$

signed magnitude

$-(2^{b-1}-1) \rightarrow (2^{b-1}-1)$

$\Rightarrow -7 \rightarrow 7$

2s Complement:

$-(2^{b-1}) \rightarrow 2^{b-1}-1$

$\Rightarrow -8 \rightarrow +7$

Find 5-bit 2s Complement of -6?

00110

flip → 11001

\+ 1

11010

Convert to decimal from 2s comp?

11011 - Do the same thing

flip → 00100

\+ 1 ← add 1

-00101 $\Rightarrow -5_{10}$

↑ add negative sign

Given a Binary value you need to be told whether it is signed, unsigned, or 2s comp.

This is why computers use 2s Complement!

Using 4-Bit system.

$$\begin{array}{rr} -5 & 1011 \\ +\ -6 & 1010 \\ \hline -11 & 10101 \end{array} \Rightarrow 5 \text{ not } -11?$$

Since it is a 4-Bit system we have an overflow

overflow
Not Carry

some computers have Carry & overflow flags

What if we have a 16-Bit system

$$\begin{array}{r} 1111\ 1111\ 1111\ 1011 \\ 1111\ 1111\ 1111\ 1010 \\ \hline 1\ 1111\ 1111\ 1111\ 0101 \end{array}$$

add 1s in the front (i.e. sign extention)

This extra 1-bit is actually a carry not overflow

what if they were +5 +6?

add 0s in front (i.e. sign extention)

$$\begin{array}{r} 0000\ 0101 \\ 0000\ 0110 \\ \hline 0000\ 1011 \end{array}$$

Sign extentions
Add 1s (if negative)
add 0s (if positive)

Not Gate

$Y = \overline{A}$

A	Y
0	1
1	0

Buffer

$Y = A$

A	Y
0	0
1	1

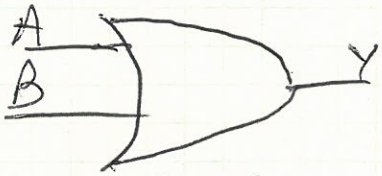
More Gates

AND

$Y = AB$

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

OR



$Y = A + B$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

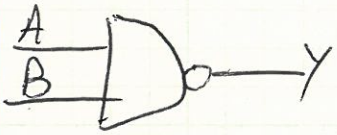
Pac Man analogy in Book

XOR

$Y = A \oplus B$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

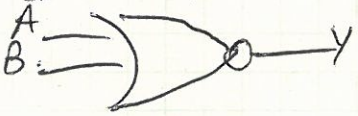
NAND – can be used to make any other gates



$Y = \overline{AB}$

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

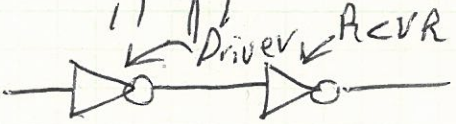
NOR $Y = \overline{A+B}$



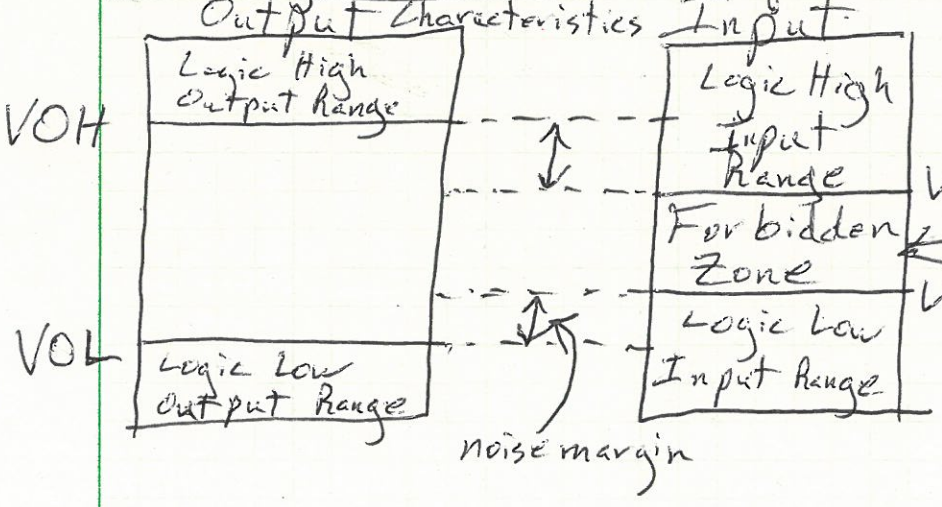
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

XNOR $Y = \overline{A \oplus B}$

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1



Voltage levels:



Output voltage is unknown in the Forbidden Zone (Gambled)