Name(s)_____ Period _____ Date _____

Activity Guide - Flippy Do Part 1

Directions

Use your Flippy Do to answer the questions.

All 4-Bit Numbers: Fill in the binary equivalents for the decimal numbers below. We've started the first three for you.

Binary: 4-bit number	Decimal
0000	0
0001	1
0010	2
0011	**3**
0100	**4**
0101	**5**
0110	**6**
0111	**7**

Binary: 4-bit number	Decimal
1000	**8**
1001	**9**
1010	**10**
1011	**11**
1100	**12**
1101	**13**
1110	**14**
1111	**15**

What do you notice when you compare the odd numbers with the even numbers? What might explain this?

The odd numbers have a 1 in the farthest right most slot. This is because

Binary Numbers with exactly one 1: Complete the chart with all 8-bit binary numbers that have exactly one 1. We've done the first two for you.

Binary: 8-bit number (with exactly one *1*)	Decimal
0000 0001	1
0000 0010	2
0000 0100	**4**
0000 1000	**8**

Binary: 8-bit number (with exactly one *1*)	Decimal
0001 0000	**16**
0010 0000	**32**
0100 0000	**64**
1000 0000	**128**

What do you notice about the decimal equivalents above?
They are all even and they are powers of 2. It is base 2^odd power.
2^3 2^1 etc...

Conversion Practice: Find the equivalent binary or decimal numbers below.

Binary	Decimal
100	4
101	5
1101	13
0001 1111	31
0010 0000	32
1010 1010	170
1111 1111	255

Binary	Decimal
00000101	5
00010001	17
00111111	63
01000000	64
01111111	127
00000001 00000000	256
00000010 00000001	513

.

When you add a zero to the right of a decimal number, it multiplies its value by 10 (For example, “15” becomes “150”). What similar result happens to the value of a binary number when you add a zero on the right? (For example, “11” would become “110”).

You multiply its value by 2 since it is base 2.

Do the binary numbers “0011” and “000011” have the same value or different values? Explain.

They have the same value on paper because it doesnt change its value when you add a zero in front of the number. When you add a number in from you add 0 to its value.

$(0011)_2 = (0 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = (3)_{10}$
$(000011)_2 = (0 \times 2^5) + (0 \times 2^4) + (0 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = (3)_{10}$

Would two bits be enough to assign a unique binary number to each vowel in the English language? Explain.

There are 5 vowels in the English Language:
A, E, I, O, U.
If we assign a decimal number to each:
0 A
1 E
2 I
3 O
4 U
comparison to binary:
0 A 00000000

1 E 00000010
2 I 00000010
3 O 00000011
4 U 00000100

You would have to have 3 bits to fit all of the vowels not 2. Since 4: 0000100 is 3 bits of “information” long, (not including the leading zeros).

How many bits would you need if you wanted to count up to 1000?
$1111101000_2 = 1000_{10}$
$1111101000_2 = 10$ bits