|  |
| --- |
| **Binary Numbers** |

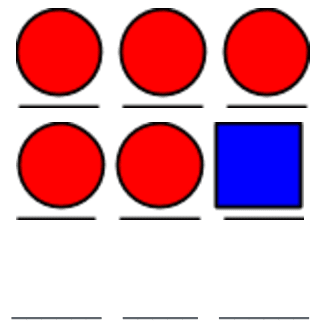
|  |
| --- |
| **Your Tasks (Mark these off as you go)** |
| Brainstorm: How many three place patterns can you make with a circle and square  Watch the Circle-Triangle-Square to Binary video  Get acquainted with the virtual Flippy-Do  Use your Flippy-Do to determine all the possible combinations of a binary number for a given number of bits  Determine the base 10 value of all the 8-bit binary numbers with exactly one *1*  Practice with conversions  Complete the reflection questions  Receive credit for this lab guide |

* **Brainstorm: How many three place patterns can you make with a circle and square**

In the previous lesson you created 27 different 3-place patterns out of circles, triangles and squares, and tried to define a system of rules to generate all of the patterns.

What if you only had a circle and square? With only a circle and square, how many 3-place patterns are there? A few are started below. How many are there total?

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

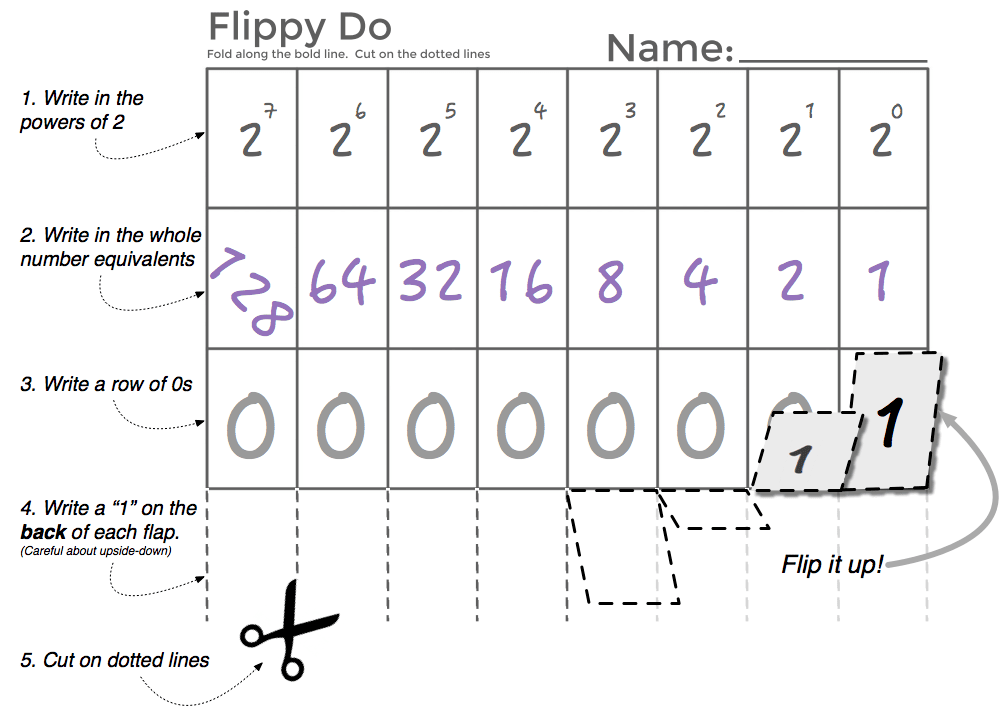


* **Watch the Circle-Triangle-Square to Binary Video**

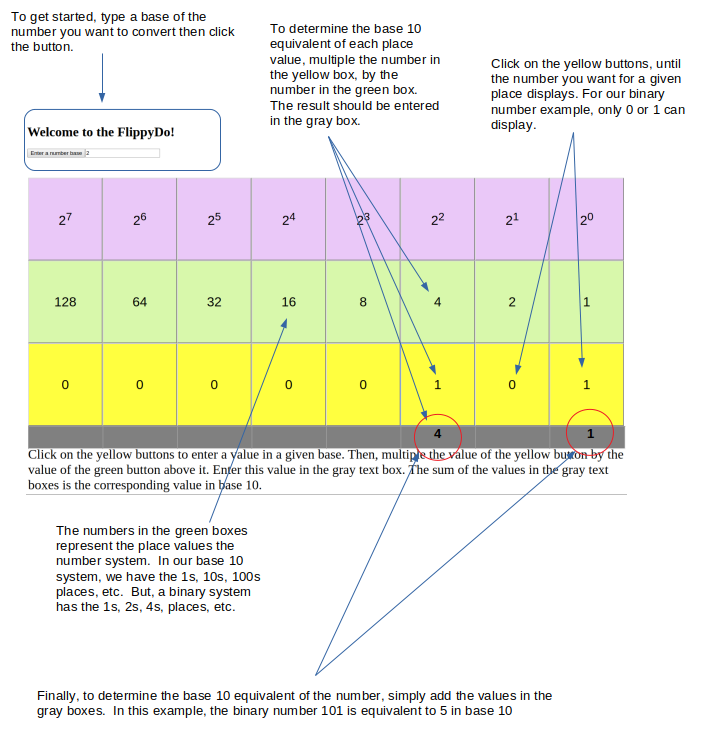
To help with the transition from circle-triangle-square to binary check out the following video,

|  |
| --- |
|  |
| <https://www.youtube.com/watch?v=91HLBUjCHbs> |

* **Get acquainted with the virtual Flippy-Do**

****A Flippy-Do is a useful tool for figuring out the decimal equivalent of a base 2 number. An example of a paper Flippy-Do is illustrated to the right.

In this lab we will be using a virtual Flippy-Do! This virtual Flippy Do will allow you to convert any number from base 2 thru 10 to its decimal equivalent. How to use the virtual Flippy Do is illustrated below,

****

To play with the virtual Flippy-Do follow the link below,

<https://timberlinecs.github.io/FlippyDo>

* **Use your Flippy-Do to determine all the possible combinations of a binary number for a given number of bits**

Navigate to the Flippy Do, type the number 2 for base-2 and click the button. You can enter binary numbers by clicking on the yellow buttons. If each yellow button represents a bit, or a binary number. How many possible combinations are there for each number of bits. Refer to the Flippy-Do to figure this out.

|  |  |  |  |
| --- | --- | --- | --- |
| **2 bits** | **3 bits** | **4 bits** | **5 bits** |
|  |  |  |  |

Write down all the possible combinations of 1s and 0s for a 3 bit system. For each combination, indicate the base-10 equivalent

|  |  |
| --- | --- |
| **Combination of 1s and 0s** | **Base 10 equivalent** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

* **Determine the base 10 value of all the 8-bit binary numbers with exactly one 1**

The table below contains *every* 8-bit number that has exactly one *1* in it. Write down the decimal equivalent next to each one. Do you notice a pattern?

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

* **Practice with conversions**

Using your own binary skills (aided by the flippy do) fill in the decimal and binary equivalents below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **What’s the Decimal Number?** | |  | **What’s the Binary Number?** | |
| **Binary** | **Decimal** |  | **Binary** | **Decimal** |
| **100** |  |  |  | **5** |
| **101** |  |  |  | **17** |
| **1101** |  |  |  | **63** |
| **0001 1111** |  |  |  | **64** |
| **0010 0000** |  |  |  | **127** |
| **1010 1010** |  |  |  | **256\*** |
| **1111 1111** |  |  |  | **513\*** |
| **NOTE**: a short binary number like **101** is assumed to have leading 0s for all the other bits, like: **00000101.**  Typically large binary numbers are grouped in 4-bit chunks to improve readability, for example: **0110 0101 1010** | |  | **\*NOTE: 256 and 513** exceed the capacity of the flippy-do but you can work it logically following what you know about patterns with binary numbers. | |

* **Complete the reflection questions**

There is a simple pattern for determining if a binary number is odd. What is it and why does this pattern occur?

|  |
| --- |
|  |

How many bits would you need if you wanted to have the ability to count up to 1000?

|  |
| --- |
|  |

How high could you count in binary if you used all 10 of your fingers as bits? (finger up means 1, finger down means 0)

|  |
| --- |
|  |

* **Receive Credit for this lab guide**

Submit this portion of the lab to Pluska to receive credit for the lab guide.