Binary Calculator

What You Need:

1 x Crazy Circuits Bit Board
1 x micro:bit
1 x 7 Segment Display
4 x Jumper Wires
4 x Slide Switch
1 x Jumbo Pushbutton
1 x LEGO Baseplate
Misc. LEGO Pieces
1/8" Maker Tape

How it Works:

There are four switches that can be set to an **on** or **off** postion. The on position is equal to 1, and the off position is equal to 0.

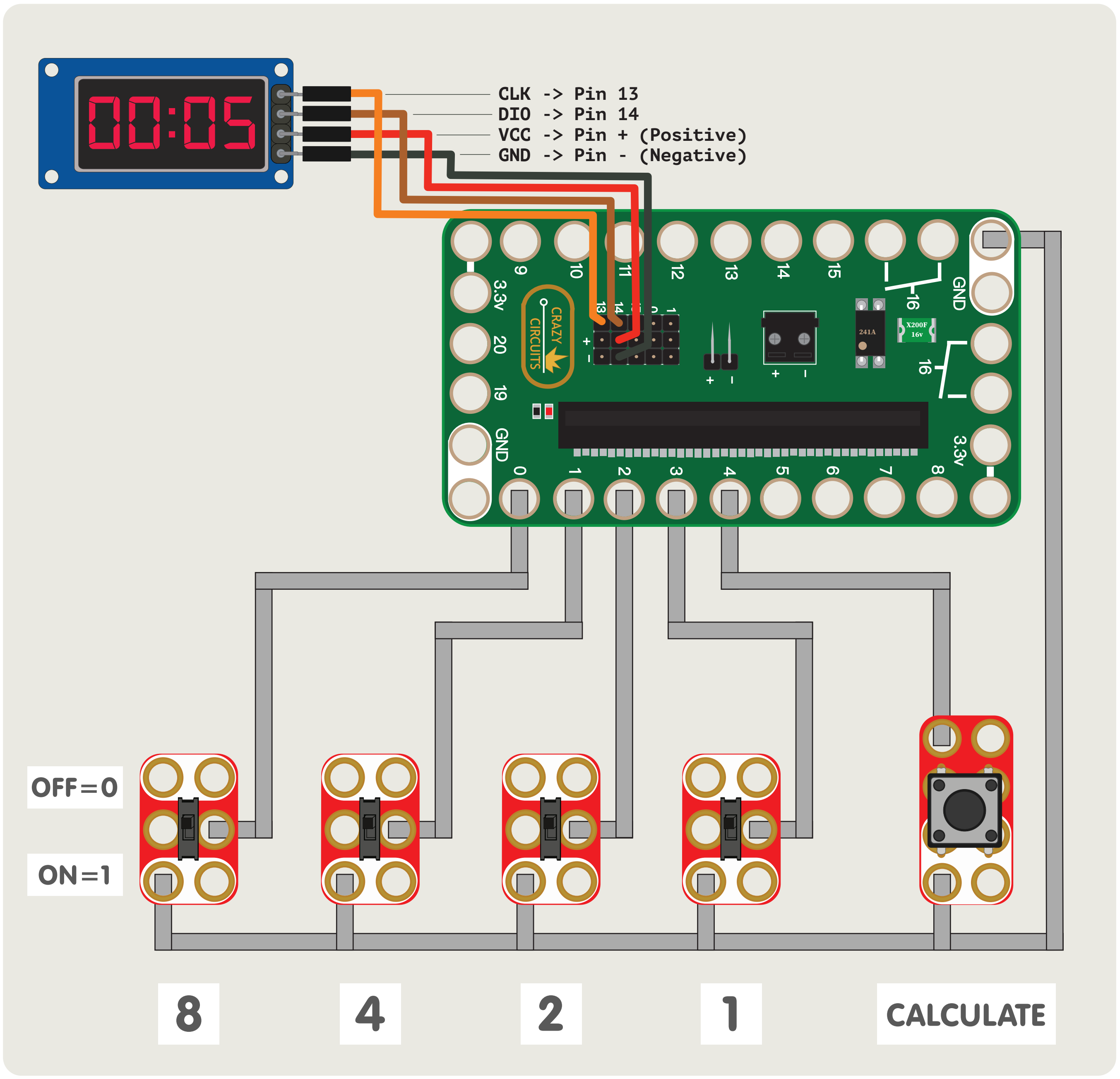
After setting each switch into a position to equal 1 or 0 you can press the pushbutton to calculate the value.

With all four switches in the off position, the binary number 0000 would translate to 0 in decimal.

When a switch is in the on position you add the value for that switch (8, 4, 2, or 1) to the total which will be shown when you press the CALCULATE button.

If the switches are **off, on, off, on** while reading left to right your binary number would be 0101 and would add **0 + 4 + 0 + 1** for a total of **5**.

Each time you press the **CALCUATE** button it resets the number to zero, and then adds up the results of the switches and shows it on the 4 digit display connected to the Bit Board.



For more fun projects and educational guides visit us at BrownDogGadgets.com

Binary Numbering System

To the right is a chart showing how you would translate a binary number into a decimal number.

You can get any value between 0 and 15 with just four binary digits. (This is a **4 bit register**.)

1, 2, 4, and 8 are the only numbers we need to accomplish our counting. If we need 3 we just add 1 and 2. If we need 7 we add 4 and 2 and 1, and so on up to 15.

If this is a 4 bit register, what would an 8 bit register look like? To start with, you would have 8 digits instead of 4 and you would be able to count much higher of course.

Here is our **8 bit register**. It has 8 "slots" that we can toggle on/off to create our decimal number. (And yes, each "slot" holds one **bit** which is why this is referred to as **8 bit**.)

128	64	32	16	8	4	2	1
0	0	0	0	0	0	0	0

If you examine the numbers from right to left you'll notice that the next digit is always one more than the total we can get using the previous numbers combined.

For instance, 1 + 2 = 3, so we don't need a 3 but we do need a 4 because we can't count to 4 just using 1 and 2. Likewise, 4 + 2 + 1 = 7, so the next value we need is 8. This continutes on along the string of numbers. 64 + 32 + 16 + 8 + 4 + 2 + 1 = 127. If we add 127 and 128 we get 255. Now, if you're familiar with 8 bit and know that it's related to the number 256, that's because we need to count zero! (A count from 0 to 255 gives us 256 values because when computers count they count 0 as a valid value.)

These bits are part of a **base 2 numbers system**. While base 10 numbers look like 10, 100, 1,000, 10,000, etc. Our base 2 numbers look like 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, etc.

With our 4 bits, that means 2 to the power of 4 (or **2 x 2 x 2 x 2**) which equals 16, and the highest value we can count to with our 8 bit register. (Remember, we're including zero, so 0 to 15 is really 16 values.)

The term 'bit' is short for 'binary digit'

Binary	Decimal
8 4 2 1	
0 0 0 0	0
0 0 0 1	1
0 0 1 0	2
0 0 1 1	3
0 1 0 0	4
0 1 0 1	5
0 1 1 0	6
0 1 1 1	7
1 0 0 0	8
1 0 0 1	9
1 0 1 0	10
1 0 1 1	11
1 1 0 0	12
1 1 0 1	13
1 1 1 0	14
1 1 1 1	15

4 bit register

Binary Calculator Code

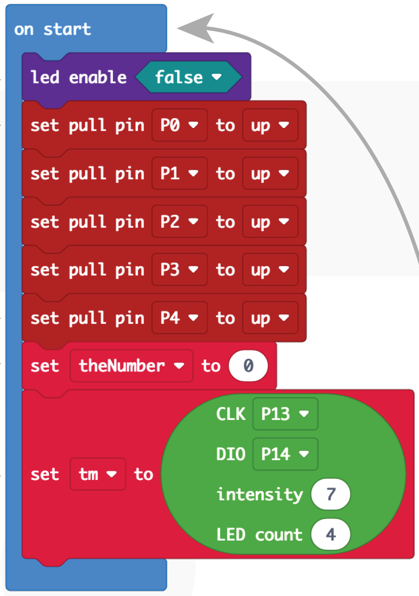
Disable the built-in LED matrix

For each of the pins we connect to for the four switches and the pushbutton we need to set them as “up” pins so that when we turn “on” the switch, or push the button, the are pulled “down” to ground by completing the connection

We have a variable called **theNumber** which will hold the value we get when we add our registers together

We have a variable called **tm** which represents the 7 Segment Display we are using to display the decimal number

We have the 7 Segment Display connected to Pin 13 and Pin 14, so we specify that here



```
on start
    led enable false
    set pull pin P0 to up
    set pull pin P1 to up
    set pull pin P2 to up
    set pull pin P3 to up
    set pull pin P4 to up
    set theNumber to 0
    set tm to  CLK P13
               DIO P14
               intensity 7
               LED count 4
```

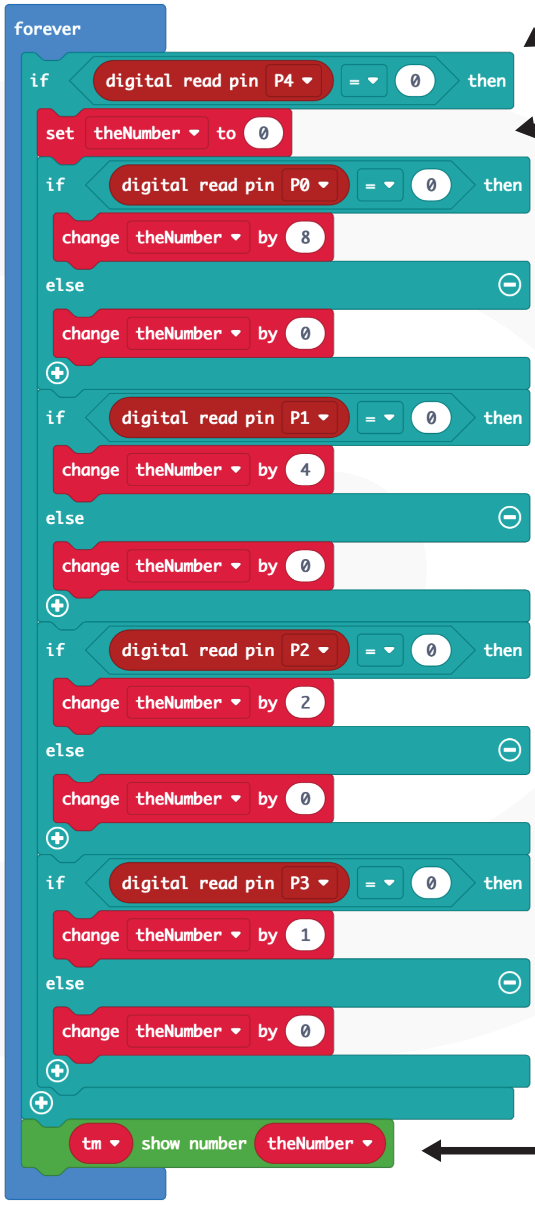
The **start** section of the code runs just once when our program begins (typically this is when the micro:bit is powered on) so it contains a lot of **setup** statements.

The **forever** section runs in a **loop** for as long as the micro:bit is turned on and receiving power.

In other systems the **start** section is referred to as the **setup** and the **forever** is referred to as the **loop**. They may have different names, but they serve the same purpose.

Let’s take a look at the code for the Binary Calculator so we can see what is happening!

https://makecode.microbit.org/_avbceA0oHcob



```
forever
    if digital read pin P4 = 0 then
        set theNumber to 0
        if digital read pin P0 = 0 then
            change theNumber by 8
        else
            change theNumber by 0
        if digital read pin P1 = 0 then
            change theNumber by 4
        else
            change theNumber by 0
        if digital read pin P2 = 0 then
            change theNumber by 2
        else
            change theNumber by 0
        if digital read pin P3 = 0 then
            change theNumber by 1
        else
            change theNumber by 0
    tm show number theNumber
```

Our pushbutton is connected to Pin 4, so we don’t do anything until the button is pressed

We set our **theNumber** variable to 0 each time the button is pressed

We have four if/then statements to evaluate each of the four switches

In each case, if the switch is in the **on** position (and it is pulled down and equals 0) we then add to **theNumber** to increase the value

From left to right the first switch represents 8, the second 4, the third 2, and the fourth 1

For each if/them statement if the switch is in the **off** position then the **else** part of the if/then statement is triggered and the value is increased by 0 (which means it does not increase in value)

We constantly display the value for **theNumber** on the 7 Segment Display, which only changes when we press the button