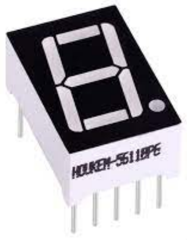
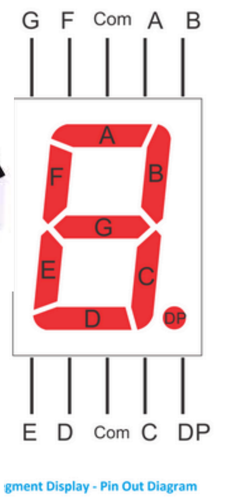
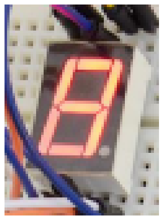
#12 Using the 1-digit, 7 segment tube module





The “tube” refers to the segment of light on the display module. This component has 7 tubes and one dot.

Different pins on the microcontroller are activated to end electricity to the tubes which lights them to create each individual number or letter. To make the number 8, for instance, all seven segments are lit. There are 10 pins total pins on the bottom of the component that fit nicely into a breadboard. 8 OUTPUT pins for each of the 7 segments, and the dot that indicates a decimal, and two Com, or common, pins that are to connect to ground (GND).

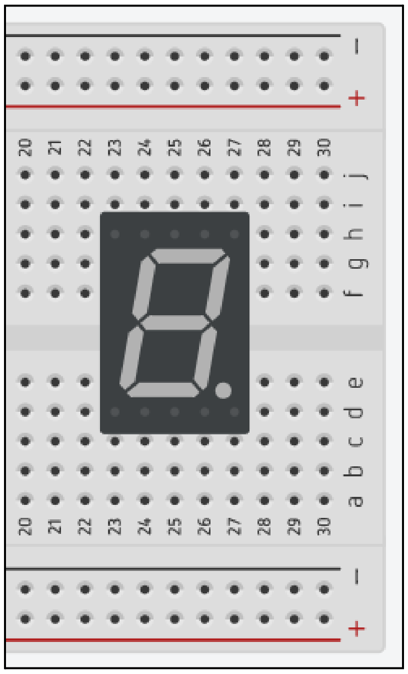


HELPFUL TO KNOW: Often, if there is a sequence of code within a program that will be repeated many times, coders will create ***functions***. Functions are small sequences of code set aside within the program. This way, a single line of code can “call out” that sequence of code. The sequence will run, and then the main code will resume.

In the code example we will be using, the ‘main’ code sequence is significantly shorter because single lines of code ‘call out’ to sequences of many lines of code that activate each of the numbers, then return back to the main code. Think about how you might use functions in other coding situations…

To wire this component you will need:

- 2 black male-to-male jumpers to connect the GND (ground pins)
- 8 male-to-male jumpers of different colors for the 8 output pins
- 8 220ohm resistors, one for each output pin
- Breadboard
- Micro:bit breakout/breadboard shield
- breadboard



Place the module on your breadboard so that 5 pins are in pinholes on one side of the breadboard and the other 5 pins are in pinholes on the other side - they should straddle the center/middle of the breadboard

See the next page for wiring instructions.

Wire as such:

Segment **g** pin to pin **16** (not 11)

NOTE: change pin11 to pin16 on the code and the wiring *(p11 controls button B on the v2 micro:bit! The wiring from the Keystudios directions was for the older version 1 micro:bit)*

Segment **f** pin to resistor to pin 12

Segment **a** pin to resistor to pin 13

Segment **b** pin to resistor to pin 14

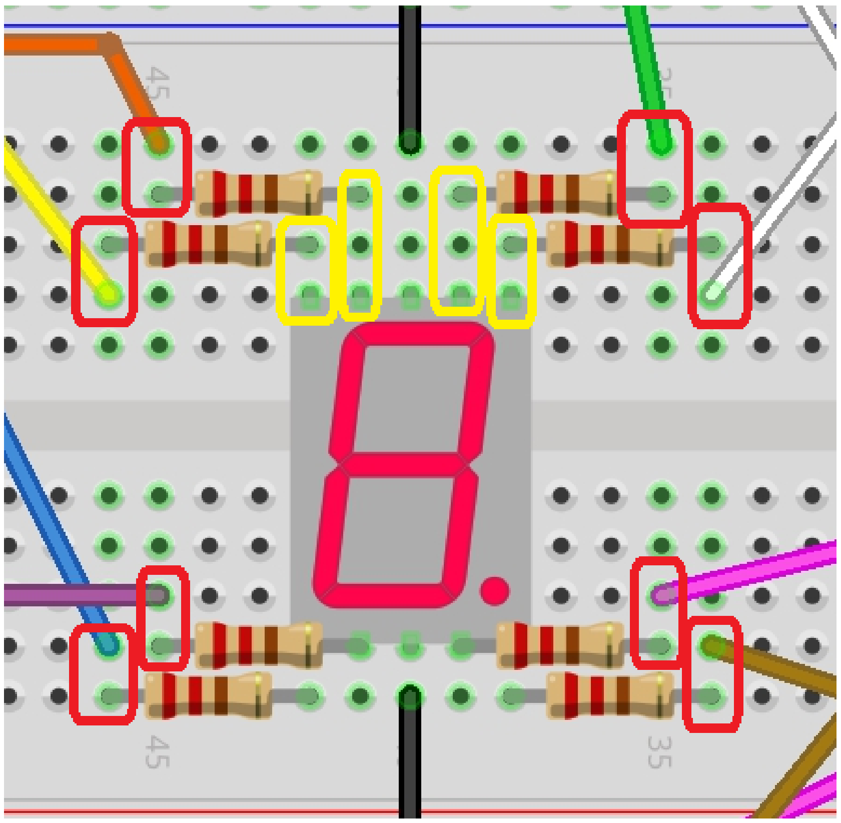
Segment **e** pin to resistor to pin 10

Segment **d** pin to resistor to pin 9

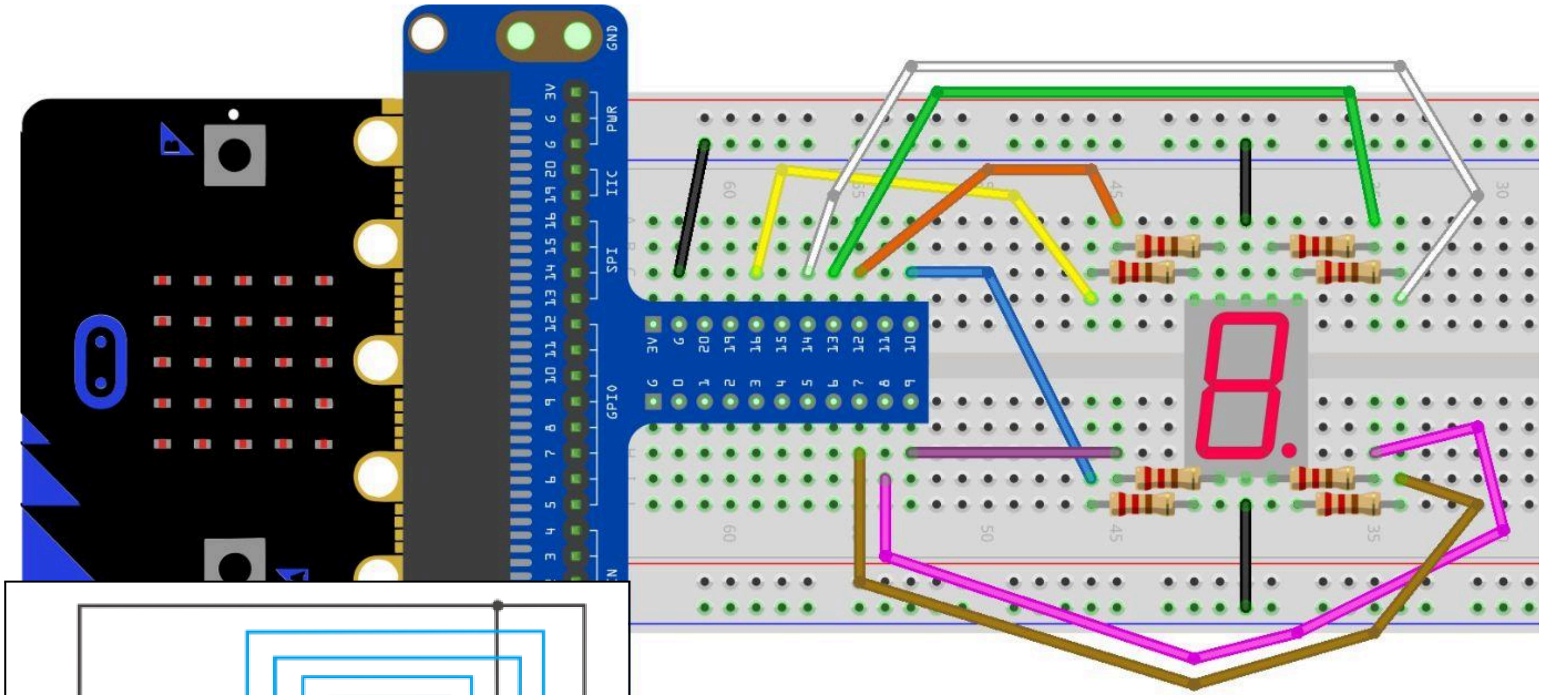
Segment **c** pin to resistor to pin 8

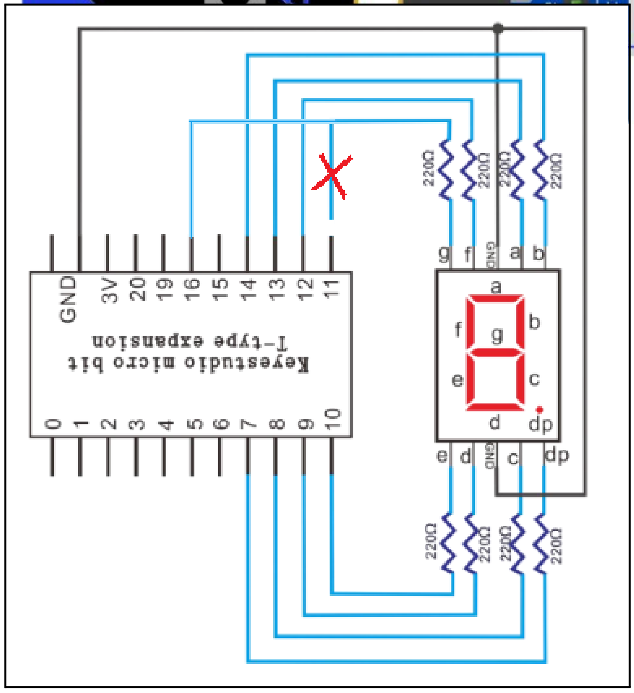
Segment **dp** *(decimal point)* pin to resistor to pin 7

Both top and bottom GND to ground rail



Be sure that each resistor is placed so that it is in the same row on the breadboard as the pin, and the jumper wires are placed in the same row of the OTHER end of the resistor.

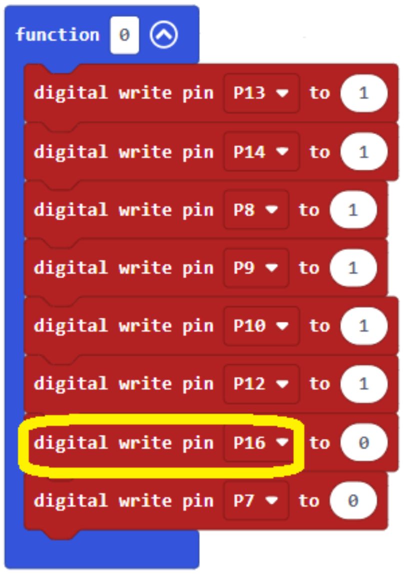




MakeCode

MakeCode code example can be found on page 257 of the [Keyestudios tutorial.](#)
*DON'T FORGET to **change p11 to p16 on the code** (p11 controls button B on the version 2 micro:bit! The wiring and code was for the older version 1.5 micro:bit)*

***Find the details about this component and the instructions *on page 257 in the keystudios inventors kit tutorials.*



```
function 0
    digital write pin P13 to 1
    digital write pin P14 to 1
    digital write pin P8 to 1
    digital write pin P9 to 1
    digital write pin P10 to 1
    digital write pin P12 to 1
    digital write pin P16 to 0
    digital write pin P7 to 0
```

This is the **MicroPython** code to confirm that you have wired the module successfully:

```
from microbit import *
display.off()

# Here are all the function code sequences
def off():
   pin13.write_digital(0)
   pin14.write_digital(0)
   pin8.write_digital(0)
   pin9.write_digital(0)
   pin10.write_digital(0)
   pin12.write_digital(0)
   pin16.write_digital(0)
   pin7.write_digital(0)
def test_all():
   pin13.write_digital(1)
   pin13.write_digital(1)
   pin14.write_digital(1)
   pin8.write_digital(1)
   pin9.write_digital(1)
   pin10.write_digital(1)
   pin12.write_digital(1)
   pin16.write_digital(1)
   pin7.write_digital(1)
def seven():
   pin13.write_digital(1)
   pin14.write_digital(1)
   pin8.write_digital(1)
   pin9.write_digital(0)
   pin10.write_digital(0)
```

# **def** = define. Any sequence of code indented under this is part of a "function". A function is a sequence of code that is referred to in another part of the code.

# There are 8 lines of code under this function/sequence. Each line turns one of the segments on or off. This code indented under "off():" turns each of the 'tubes' off. (each has a value of 0)

# This code indented under "test_all()" turns all tubes, and the decimal on. (each has a value of 1)

# This code indented under "seven()" turns on only the tubes needed to make the number 7 (the tubes not needed have a value of 0)

`    pin12.write_digital(0)`
`    pin16.write_digital(0)`
`    pin7.write_digital(0)`	

`def two():`
`    pin13.write_digital(1)`
`    pin14.write_digital(1)`
`    pin8.write_digital(0)`
`    pin9.write_digital(1)`
`    pin10.write_digital(1)`
`    pin12.write_digital(0)`
`    pin16.write_digital(1)`
`    pin7.write_digital(0)`
	# This code indented under “two()” turns on only the tubes needed to make the number 2 (the tubes not needed have a value of 0)

`def six():`
`    pin13.write_digital(1)`
`    pin14.write_digital(0)`
`    pin8.write_digital(1)`
`    pin9.write_digital(1)`
`    pin10.write_digital(1)`
`    pin12.write_digital(1)`
`    pin16.write_digital(1)`
`    pin7.write_digital(0)`
	# This code indented under “six()” turns on only the tubes needed to make the number 6 (the tubes not needed have a value of 0)

`def nine():`
`    pin13.write_digital(1)`
`    pin14.write_digital(1)`
`    pin8.write_digital(1)`
`    pin9.write_digital(1)`
`    pin10.write_digital(0)`
`    pin12.write_digital(1)`
`    pin16.write_digital(1)`
`    pin7.write_digital(0)`
	# This code indented under “nine()” turns on only the tubes needed to make the number 9 (the tubes not needed have a value of 0)

`def five():`
`    pin13.write_digital(1)`
`    pin14.write_digital(0)`
`    pin8.write_digital(1)`
`    pin9.write_digital(1)`
`    pin10.write_digital(0)`
`    pin12.write_digital(1)`
`    pin16.write_digital(1)`
`    pin7.write_digital(0)`
	# This code indented under “five()” turns on only the tubes needed to make the number 5 (the tubes not needed have a value of 0)

`def three():`
`    pin13.write_digital(1)`
`    pin14.write_digital(1)`
`    pin8.write_digital(1)`
`    pin9.write_digital(1)`
`    pin10.write_digital(0)`
`    pin12.write_digital(0)`
	# This code indented under “three()” turns on only the tubes needed to make the number 3 (the tubes not needed have a value of 0)

```
   pin16.write_digital(1)
   pin7.write_digital(0)
def eight():
   pin13.write_digital(1)
   pin14.write_digital(1)
   pin8.write_digital(1)
   pin9.write_digital(1)
   pin10.write_digital(1)
   pin12.write_digital(1)
   pin16.write_digital(1)
   pin7.write_digital(0)
```

# This code indented under “eight()” turns on only the tubes needed to make the number 8 (the tubes not needed have a value of 0)

```
def zero():
   pin13.write_digital(1)
   pin14.write_digital(1)
   pin8.write_digital(1)
   pin9.write_digital(1)
   pin10.write_digital(1)
   pin12.write_digital(1)
   pin16.write_digital(0)
   pin7.write_digital(0)
```

# This code indented under “zer0()” turns on only the tubes needed to make the number 0 (the tubes not needed have a value of 0)

```
def four():
   pin13.write_digital(0)
   pin14.write_digital(1)
   pin8.write_digital(1)
   pin9.write_digital(0)
   pin10.write_digital(0)
   pin12.write_digital(1)
   pin16.write_digital(1)
   pin7.write_digital(0)
```

# This code indented under “four()” turns on only the tubes needed to make the number 4 (the tubes not needed have a value of 0)

```
def one():
   pin13.write_digital(0)
   pin14.write_digital(1)
   pin8.write_digital(1)
   pin9.write_digital(0)
   pin10.write_digital(0)
   pin12.write_digital(0)
   pin16.write_digital(0)
   pin7.write_digital(0)
```

# This code indented under “one()” turns on only the tubes needed to make the number 1 (the tubes not needed have a value of 0)

```
def decimal():
   pin13.write_digital(0)
   pin14.write_digital(0)
   pin8.write_digital(0)
   pin9.write_digital(0)
   pin10.write_digital(0)
   pin12.write_digital(0)
   pin16.write_digital(0)
```

# This code indented under “decimal()” turns on only the tubes needed to make the decimal (the tubes all have a value of 0 except the dot)

```
   pin7.write_digital(1)
while True:
    if button_a.was_pressed():
        for i in range(4):
            decimal()
            sleep(500)
            off()
            sleep(100)
        zero()
        sleep(500)
        one()
        sleep(500)
        off()
        two()
        sleep(500)
        off()
        three()
        sleep(500)
        off()
        four()
        sleep(500)
        off()
        five()
        sleep(500)
        off()
        six()
        sleep(500)
        off()
        seven()
        sleep(500)
        off()
        eight()
        sleep(500)
        nine()
        sleep(500)
        off()
        decimal()
        sleep(500)
        off()
        test_all()
        sleep(500)
        off()
        sleep(3000)
```

# The “**while True**” begins the conditional “forever” loop - everything **indented** under it will be repeated “forever” (or until the electricity is eliminated). In this instance, the condition is: “when the A button is pressed”

# The “**for i in range()**” code begins a repeat loop, set at 4 times.

# The “**sleep()**” code delays the code from going to the next line of code. It is in milliseconds, 500 = ½ second.

# The **off()** code refers to a function found elsewhere in the code - it runs the code sequence defined as “off”, then returns to the next line in this code.

# The remainder of the code displays the numbers 0,1,2,3,4,5,6,7,8,and 9 in sequence with ½ second between each, then tests all tubes, and waits (sleeps) 3 seconds before beginning the forever loop again.

#Note there is an “off()” between each number. This is because just like we discovered in the LED Blink Code, if you turn it on, it stays on until you tell it to turn off.

#What do you think happens if we don’t include the “off()” between each number?