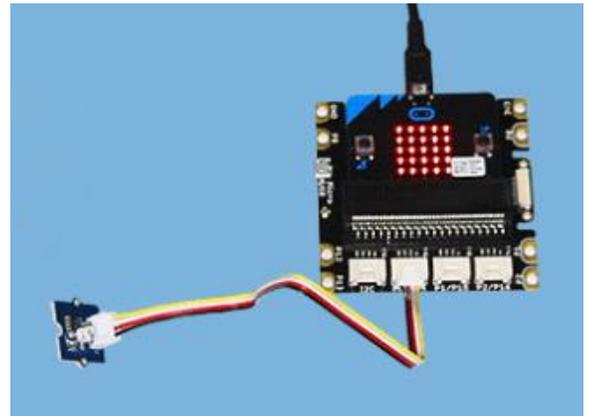


**Introduction:**

The micro:bit has a row of gold connections along the bottom edge for connecting electronic accessories. To make these connections easier there are many ‘expansion boards’ available that allow us to ‘plug in’ those accessories using standard cables. Two of these expansion boards are the **Grove shield** and the **Bitmaker expansion board**. The projects presented here will work with these and many other expansion boards. In addition to the expansion board, you will use some Grove accessories: an external LED (any *one* color), a light sensor, an ultrasonic ranger, and an external speaker.

These activities assume you have completed the *first* set of micro:bit activities.

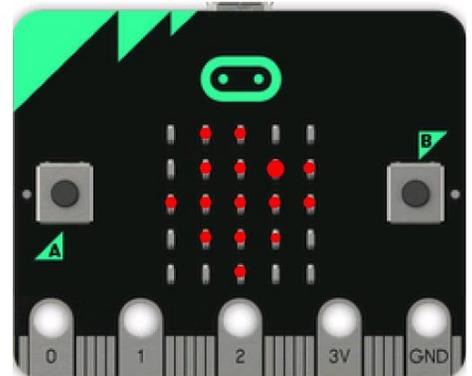
1. Before you begin, be sure that:
  - your **micro:bit** is connected to your TI-Nspire™ CX II
  - your **micro:bit** is inserted in the **expansion board**.  
(a Grove shield is shown here)
  - for this activity, you must have a Grove LED accessory and 4-wire connecting cable as shown here



2. Your micro:bit should look like this when it has power from the TI-Nspire™ graphing calculator:
 

Recall that the display on the **micro:bit** is showing the TI logo, an icon of the state of Texas with a bright spot near Dallas, the home of **Texas Instruments, Inc.**

This activity has two parts: an on/off switch and a temporary-on switch. There’s also a ‘Challenge’ at the end.

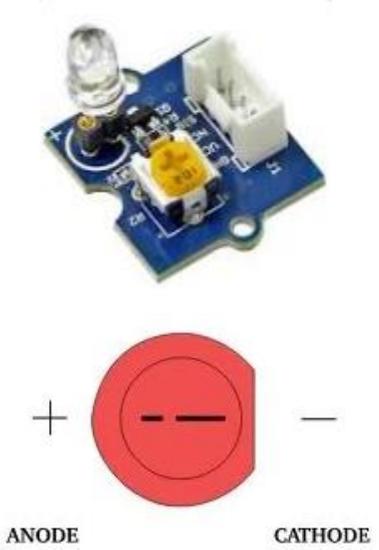


# 10 Minutes of Code: Python Modules

## MICRO:BIT: THE LIGHT SWITCH

### TI-NSPIRE™ CX II

- The Grove LED is a small circuit board with a white (or colored) LED on it. **Turn the yellow dial (a potentiometer) all the way to the right to get the maximum brightness.** Connect this LED board to the port labeled **P0 (pin0)** on the expansion board using the cable included.
  - Caution!** The LED can be removed from its socket on the circuit board. Be careful when re-inserting the wires properly. There is a positive (anode) wire and a negative (cathode) wire and the circuit board is labeled with '+' and '-' symbols. The negative wire on the LED is indicated by a 'flat spot' on the base of the LED as shown here.
- Inserting the LED incorrectly will cause it to not work but will not damage it.
- Push-button light switches like the one seen here were common in the first half of the 20<sup>th</sup> century (1900-1950's) when they were replaced with toggle switches. One button turned the circuit on while the other turned it off (and popped the other button out). These two projects will make similar switches using the micro:bit buttons.



### 6. Part 1: The 2-button on-off switch

Start a new TI-Nspire document and select **Add Python > New** to begin a new program (blank program), named **'switch1'**. In the Python Editor use **[menu] > More Modules > BBC micro:bit** to select the **import** statement at the top of the menu items:

**from microbit import \***

*Tip: This statement tests to see if the micro:bit is present. If the message 'micro:bit not connected' ever appears, just unplug the micro:bit and plug it in again (reset).*



# 10 Minutes of Code: Python Modules

## TI-NSPIRE™ CX II

7. Make a *while not escape* loop using the command found on  
[menu] > More Modules > BBC micro:bit > Commands >  
**while get\_key() != "esc":**



We also added comments and a `print()` statement to explain the program.

Note that `get_key()` is included in the `micro:bit` module. There's no need to import the `ti_system` module.

8. In the loop body, add two `if` statements, one for each micro:bit button using the two `_was_pressed()` functions:

◆◆ **if button\_a.was\_pressed():**



◆◆ **if button\_b.was\_pressed():**



These functions are found on

[menu] > More Modules > BBC micro:bit > Buttons under `button_a` and `button_b`.

One button will turn the LED on and the other will turn it off.

9. The LED lights up when it gets power from the micro:bit. It responds to power *levels* in the (decimal) range from 0 to 100. So...

To turn the LED on brightly, use the function:

**grove.set\_power(pin0, 100)**

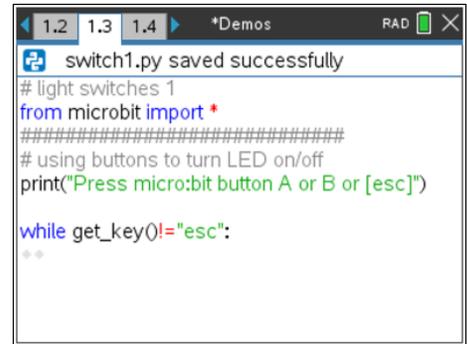
found under the **BBC micro:bit > Grove Devices > Output** menu .

**pin0** is a special micro:bit variable found on  
> **BBC micro:bit > I/O Pins.**

With your cursor on the first argument of the function, select **pin0** from the list (or you can simply type it in manually).

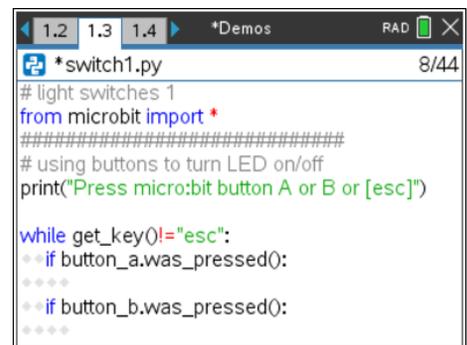
The second argument of the `grove.set_power( , )` function is the power level to send.

## MICRO:BIT: THE LIGHT SWITCH



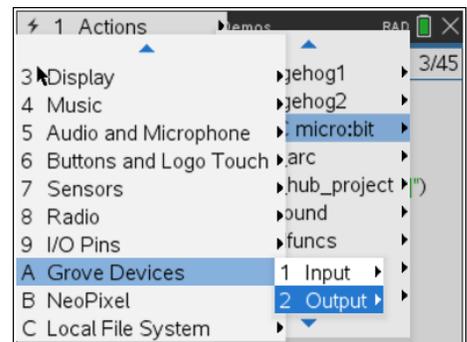
```
switch1.py saved successfully
# light switches 1
from microbit import *
#####
# using buttons to turn LED on/off
print("Press micro:bit button A or B or [esc]")

while get_key()!="esc":
  ..
```



```
*switch1.py 8/44
# light switches 1
from microbit import *
#####
# using buttons to turn LED on/off
print("Press micro:bit button A or B or [esc]")

while get_key()!="esc":
  ..if button_a.was_pressed():
  ..
  ..if button_b.was_pressed():
  ..
```



# 10 Minutes of Code: Python Modules

## TI-NSPIRE™ CX II

10. In a similar manner, turn the LED off when button B was pressed. Set the power level to 0 as shown here.

Test your program now to be sure the LED is working properly. Try other power levels for each button.

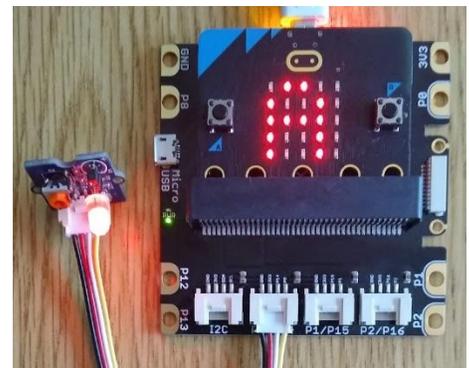
## MICRO:BIT: THE LIGHT SWITCH

```
1.2 1.3 1.4 *Demos RAD 12/45
*switch1.py
# using buttons to turn LED on/off
print("Press micro:bit button A or B or [esc]")

while get_key()!="esc":
    if button_a.was_pressed():
        grove.set_power(pin0, 100) # on
    if button_b.was_pressed():
        grove.set_power(pin0, 0) # off
```

11. Make one enhancement: display the letter of each button on the micro:bit display: Add `display.show("A")` when button A is pressed and a similar statement for button B.

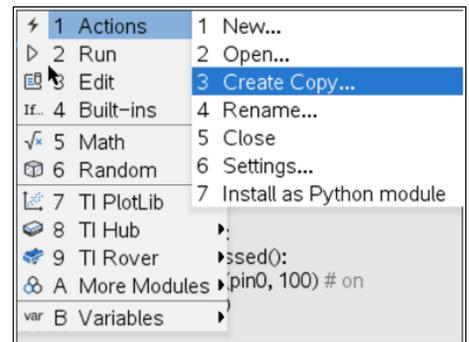
After the **while** loop ends (by pressing **[esc]**), turn the LED off and clear the display (or display the special TI logo image using `display.show(ti)` to return the micro:bit display to its original state).



## 12. Part 2: The ON button

This program will light the LED when button A is *held down* and it will be off when the button is released. Make a copy of your program from Part 1 of this activity. Use **[menu] > Actions > Create Copy...**

Give the new program a name (ours is **switch2.py**)



13. In the new program, modify the button A **if** statement: change **was** to **is**.

**if button\_a.is\_pressed():**

```
1.2 1.3 1.4 *Demos RAD 8/47
*switch2.py
# light switches 1
from microbit import *
#####
# using buttons to turn LED on/off
print("Press micro:bit button A or B or [esc]")

while get_key()!="esc":
    if button_a.is_pressed():
        grove.set_power(pin0, 100) # on
        display.show("A")
    if button_b.was_pressed():
        grove.set_power(pin0, 0) # off
```

# 10 Minutes of Code: Python Modules

## TI-NSPIRE™ CX II

14. We will not need the button B condition, but we will need to turn the LED off when button A is *not* pressed, so replace the **if button\_b...** statement with an **else:** clause for **if button\_a...** and turn off the LED in the **else:** block.

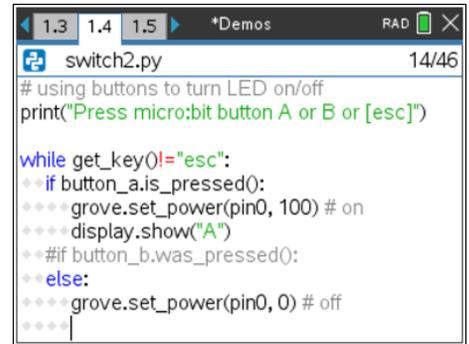
```
if button_a.is_pressed():
    grove.set_power(pin0, 100) # on
    display.show("A")
else:
    grove.set_power(pin0, 0) # off
    # if button_b.was_pressed():
```

*Note that if button\_b... is now commented and else: lines up with if !*  
If you displayed the letter “B” in the first project, change it to a space “ ” here so that nothing is displayed when no button is pressed.

15. **Challenge:** Make a single-button on-off switch. Can you modify the program again to turn the LED on when button A is pressed once and turn it off when button A is pressed again? That is, button A toggles the LED on and off by itself and does not need to be held down. Button B is not used at all. If you have micro:bit **version 2**, try using the pin\_logo (the gold oval above the 5x5 display in the image).

The display can show a ‘1’ when the LED is on and ‘0’ when it is off.

## MICRO:BIT: THE LIGHT SWITCH



```
switch2.py 14/46
# using buttons to turn LED on/off
print("Press micro:bit button A or B or [esc]")

while get_key()!="esc":
    if button_a.is_pressed():
        grove.set_power(pin0, 100) # on
        display.show("A")
    elif button_b.was_pressed():
        else:
            grove.set_power(pin0, 0) # off
```

