Name: _____

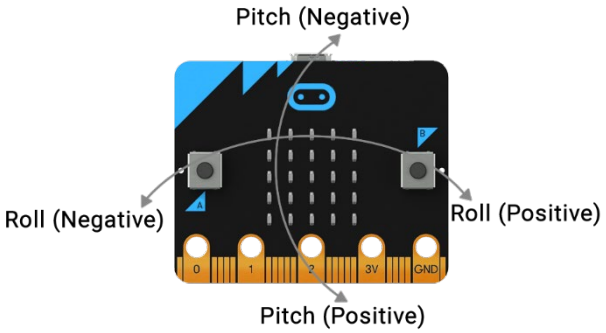
Tilt that LED!

Professor Bob Brown – Kennesaw State University
Bob.Brown@Kennesaw.edu

Pitch and Roll

Remember the first thing we did with the Micro:Bit? Part of the pre-loaded program was a game. You could "roll" an LED by tilting the Micro:Bit left, right, up, and down, to try to make the moving LED hit a target. It's time for us to do something like that! We'll begin a little slowly.

The Micro:Bit contains a part called an *accelerometer* that can measure changes in position. The *roll* of the Micro:Bit tells how much is it tilted, left or right. Tilting the left side downward gives a negative value for roll. Tilting the left side upward gives a positive value. If the Micro:Bit is level, the value is zero.



The *pitch* tells how much the Micro:Bit is tilted up or down. Tilting the top of the Micro:Bit downward gives a negative value for pitch. Tilting the top upward gives a positive value. If the Micro:Bit is level, the value of pitch is zero.

Both pitch and roll are given in degrees, but at the moment we are only interested in left/right and down/up, so only in negative or positive.

Rolling Sideways

To start, we will make an LED "roll" left or right along the X-axis to show which way the Micro:Bit is tilted. When the position reaches the edge of the LED field, we will stop until the LED is "rolled back" by tilting the other way.

Rolling Sideways Algorithm

Start a new program called *roll*.

Remember that the LEDs are numbered 0 to 4 along each axis.

We need two variables, *x* to show which LED on the X-axis to turn on, and *roll*, to hold the measurement from the accelerometer. We will make a *y* variable as well, but for now we will just set it to 2, the middle row of LEDs.

On start, set *x* to 2, set *y* to 2, plot *x*-*y*, and pause one second for humans to see what happened.

Forever

- unplot *x,y* — Turn off whatever LED was on before
- set *roll* to rotation(roll) — "rotation" is in the More part of the Input bin.
- if *roll* < 0 — Negative, so tilted left
    - Set *x* to *x* – 1 — "roll" the LED left
    - If *x* < 0 set *x* to 0 — We are at the left edge
- else if *roll* > 0 — Positive, so tilted right
    - set *x* to *x* + 1
    - if *x* > 4 set *x* to 4 — We are at the right edge
- plot *x,y* — Turn on new LED
- pause 200 ms — For humans to catch up to computer

Code your program and run it on the simulator. You can tilt the simulated Micro:Bit with the mouse. Try it and see.

To run this on a real Micro:Bit, you could set that last pause to 100 ms because you can do a better job of controlling the tilt with your hand.

Pitching Up and Down

Start a new program called *pitch*. Write your own algorithm using the one above as a pattern. *Hint:* This time we will keep *x* constant at 2 and change the value of *y*. You will need to change rotation from "roll" to "pitch."

Be sure to write your algorithm out before you start coding. Test your program. Can you "roll" the LED up and down?

Going Further – Make a Game!

Can you combine the two algorithms so that you can roll the LED to any spot?

~~Can you set up a "target" LED at a place other than 2,2, then have the player try to roll the moving LED to hit the target. Use new variables for the target location, perhaps *targetx* and *targety*. Remember that you have a *pick random* operator in the Math bin. What will you do if the two random numbers you pick are 2, 2? (That would place the target at the same place as the rolling LED, and *that* would be bad.)~~

~~How will you detect when the moving LED hits the target? (*Hint:* Can you check whether *x* and *y* are equal to *targetx* and *targety*? How?~~

~~What should happen when the moving LED hits the target?~~

We'll do that next week, OK?