**Day 5 Notes**

Yes, a \*\*byte\*\* variable is perfect for storing the value of a 3-position DIP switch in binary! A `byte` can hold values from `0` to `255` (or `00000000` to `11111111` in binary), which is more than enough for this setup. With a 3-position DIP switch, you’re working with values from `000` (0 in decimal) to `111` (7 in decimal), which fits comfortably within a `byte`.

### Here’s how to use a byte to store and read the DIP switch state:

1. \*\*Declare a `byte` Variable\*\*:

```cpp

byte switchState = 0;

```

2. \*\*Read Each Switch and Assign Values\*\*:

- Assign each DIP switch to a pin, then read each pin and combine their values to store a binary number in `switchState`.

- For example, if you have your DIP switches connected to pins 2, 3, and 4, you can shift each input into place.

3. \*\*Example Code for a Binary Counter\*\*:

```cpp

byte switchState = 0;

void setup() {

pinMode(2, INPUT);

pinMode(3, INPUT);

pinMode(4, INPUT);

}

void loop() {

// Read each switch and build the binary number

switchState = (digitalRead(2) << 2) | (digitalRead(3) << 1) | digitalRead(4);

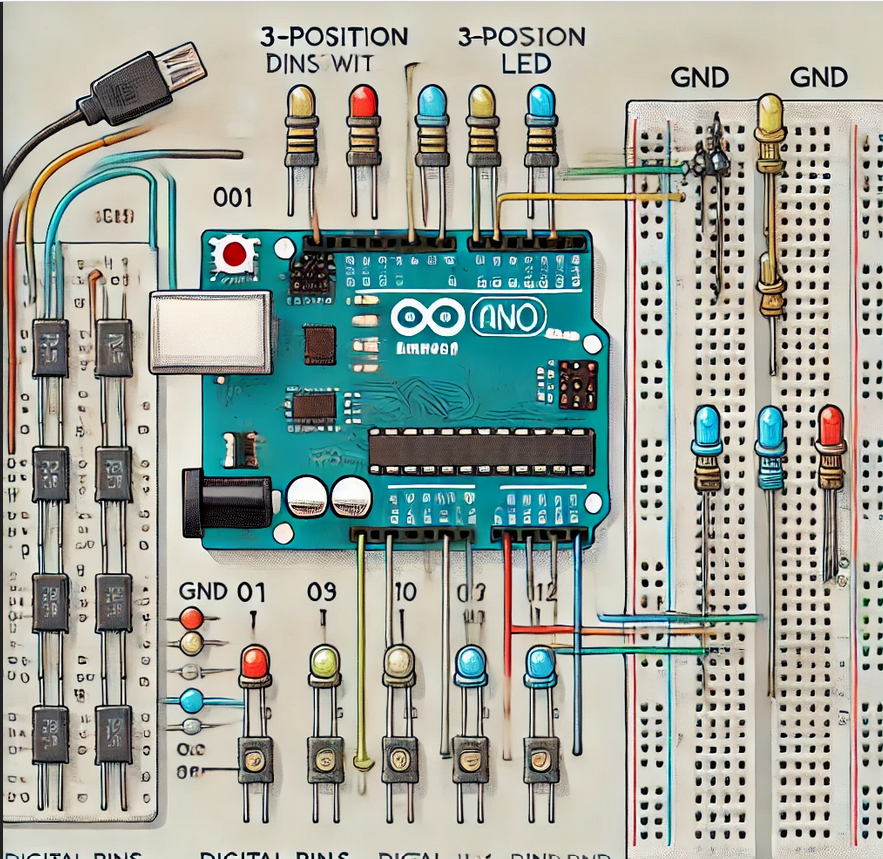
// Now, switchState holds the binary value of the DIP switch

// Use switchState to control LEDs or other outputs

}

```

Each switch position is read as either `0` or `1` and shifted into place to create a binary number representing all three switches. This allows you to control LEDs in a binary count sequence or create different patterns. Let me know if you want to explore this further or see it in action with some LEDs!



*~~Here’s the visual circuit diagram you requested! The setup includes a 3-position DIP switch connected to digital pins 2, 3, and 4 on the HERO board, each with a 10k Ohm pull-down resistor to GND. Three LEDs are connected to pins 9, 10, and 11 with 220 Ohm resistors.~~*

Reuse Board from Day 4 instead

A circuit board with wires

Description automatically generated

### Code Example

This code reads the DIP switch position and turns on LEDs based on the binary values:

```cpp

byte switchState = 0;

void setup() {

pinMode(2, INPUT);

pinMode(3, INPUT);

pinMode(4, INPUT);

pinMode(9, OUTPUT);

pinMode(10, OUTPUT);

pinMode(11, OUTPUT);

}

void loop() {

// Read the DIP switch state and create a binary number

switchState = (digitalRead(2) << 2) | (digitalRead(3) << 1) | digitalRead(4);

// Turn LEDs on or off based on the switchState value

digitalWrite(9, switchState & 0b001 ? HIGH : LOW); // LED on pin 9

digitalWrite(10, switchState & 0b010 ? HIGH : LOW); // LED on pin 10

digitalWrite(11, switchState & 0b100 ? HIGH : LOW); // LED on pin 11

}

```

Each switch position corresponds to binary numbers from `000` to `111` (0 to 7), selectively controlling which LEDs turn on based on the DIP switch configuration. Enjoy experimenting, and let me know how it goes!

That works, but the circuit you gave me was pretty cluttered and I already had one built so I just reused the one fro Day 4. Had to change the output pins from 9, 10, and 11 to 10, 11, 12 to match the circuit I had built.

Then the LEDs were reversed left to right compared to the switches from my viewing point. So, I switched the line of code

switchState = (digitalRead(2) << 2) | (digitalRead(3) << 1) | digitalRead(4);

to

switchState = (digitalRead(4) << 2) | (digitalRead(3) << 1) | digitalRead(2);

Now it works the way I wanted it to.