

## analogWrite()

### Description

Writes an analog value ([PWM wave](#)) to a pin. Can be used to light a LED at varying brightnesses or drive a motor at various speeds. After a call to `analogWrite()`, the pin will generate a steady square wave of the specified duty cycle until the next call to `analogWrite()` (or a call to `digitalRead()` or `digitalWrite()` on the same pin). The frequency of the PWM signal on most pins is approximately 490 Hz. On the Uno and similar boards, pins 5 and 6 have a frequency of approximately 980 Hz. Pins 3 and 11 on the Leonardo also run at 980 Hz.

On most Arduino boards (those with the ATmega168 or ATmega328), this function works on pins 3, 5, 6, 9, 10, and 11. On the Arduino Mega, it works on pins 2 - 13 and 44 - 46. Older Arduino boards with an ATmega8 only support `analogWrite()` on pins 9, 10, and 11.

The Arduino Due supports `analogWrite()` on pins 2 through 13, plus pins DAC0 and DAC1. Unlike the PWM pins, DAC0 and DAC1 are Digital to Analog converters, and act as true analog outputs.

You do not need to call `pinMode()` to set the pin as an output before calling `analogWrite()`.

The *analogWrite* function has nothing to do with the analog pins or the *analogRead* function.

### Syntax

```
analogWrite(pin, value)
```

### Parameters

pin: the pin to write to.

value: the duty cycle: between 0 (always off) and 255 (always on).

### Returns

nothing

### Notes and Known Issues

The PWM outputs generated on pins 5 and 6 will have higher-than-expected duty cycles. This is because of interactions with the `millis()` and `delay()` functions, which share the same internal timer used to generate those PWM outputs. This will be noticed mostly on low duty-cycle settings (e.g 0 - 10) and may

result in a value of 0 not fully turning off the output on pins 5 and 6.

### Example

Sets the output to the LED proportional to the value read from the potentiometer.

```
int ledPin = 9;    // LED connected to digital pin 9
int analogPin = 3; // potentiometer connected to analog pin 3
int val = 0;      // variable to store the read value

void setup()
{
  pinMode(ledPin, OUTPUT); // sets the pin as output
}

void loop()
{
  val = analogRead(analogPin); // read the input pin

  analogWrite(ledPin, val / 4); // analogRead values go from 0 to 1023,
  analogWrite values from 0 to 255
}
```

[\[Get Code\]](#)

### See also

[analogRead\(\)](#)

[analogWriteResolution\(\)](#)

[Tutorial: PWM](#)

[Reference Home](#)