

EEE-201 Chapter 1

In the [Arduino 1](#) class, we covered the topic of [pulse width modulation](#) (PWM); we will revisit that concept in this chapter and see how it is used to control servos. We'll be covering the following concepts:

- pulse width, voltage and period
- duty cycle
- servo control

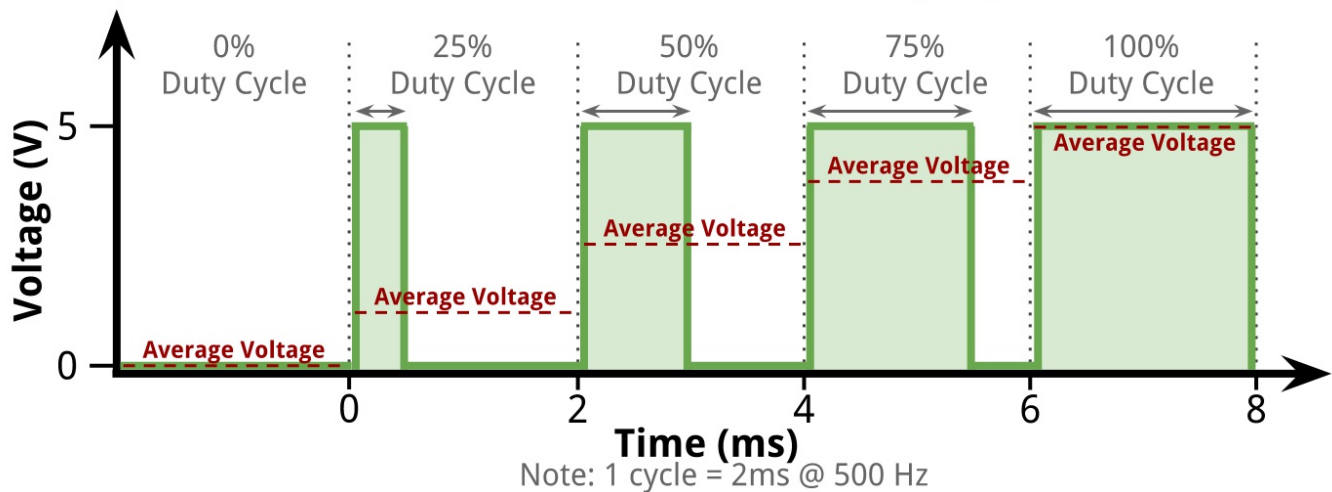
Table of contents

- [Part 1 - pulse width modulation](#)
- [Part 2 - controlling a servo](#)

Part 1 - pulse width modulation

We [previously learned](#) that PWM is a method of using a digital output pin (which can only be `LOW` or `HIGH`) and turning it into a quasi-analog output pin which can have almost any value between 0V to 5V; the figure below shows how this is achieved:

Pulse Width Modulation Duty Cycles



A PWM is a signal (square wave) made up of a repetitive pattern pulses or cycles. As the name implies, a square wave as a square pattern with a voltage being either **LOW** (0V) or **HIGH** (5V in this case). The start/end of the pulse period is designated by the voltage changing from 0V to 5V. So, in the example above, we can see that the pulse period is set to 2ms (or 0.002s) - that is, every 2ms the signal repeats itself.

These pulses are often designated in units of frequency which is just the inverse of the pulse period in seconds - in this case, $1/0.002s = 500 \text{ Hz}$.

The amount of time the signal is at **HIGH** (or at 5V in this case) defines the duty cycle or pulse width; for example, if the pulse is **HIGH** for 1ms, the duty cycle equates to 50% (1ms/2ms). At a 50% duty the cycle, the average voltage during the entire cycle will be 50% of the pulse voltage ($5V * 50\% = 2.5V$). In this way, by adjusting the pulse width, we can adjust the voltage on the output pin. We had used this technique in EEE-105 to control the brightness of a LED, but in this class we will use the same technique to control a servo!

Part 2 - controlling a servo

We can use a signal generator to show how to use PWM to control a

servo. The majority of hobby servos requires a 50Hz signal (0.02s or 20ms) with a pulse width in the range of 0.5-2.5ms; if you do that math, that equates to a duty cycle range of 2.5-12.5%.

We will use the [Rigol DG1022A](#) signal generator to generate a square wave using the `Pulse` functionality. Note that we can't use the `Square` function on the signal generator because the smallest duty cycle it allows is 20%, which is out of the range of the servo - so instead, we use the `Pulse` function.

By hooking the servo up to the signal, we can see that by adjusting the duty cycle, the position of the servo also changes. It's important to note here that the pulse width can only be changed at a frequency of 50Hz; if it is changed more rapidly, you can get erratic behavior in your servo.