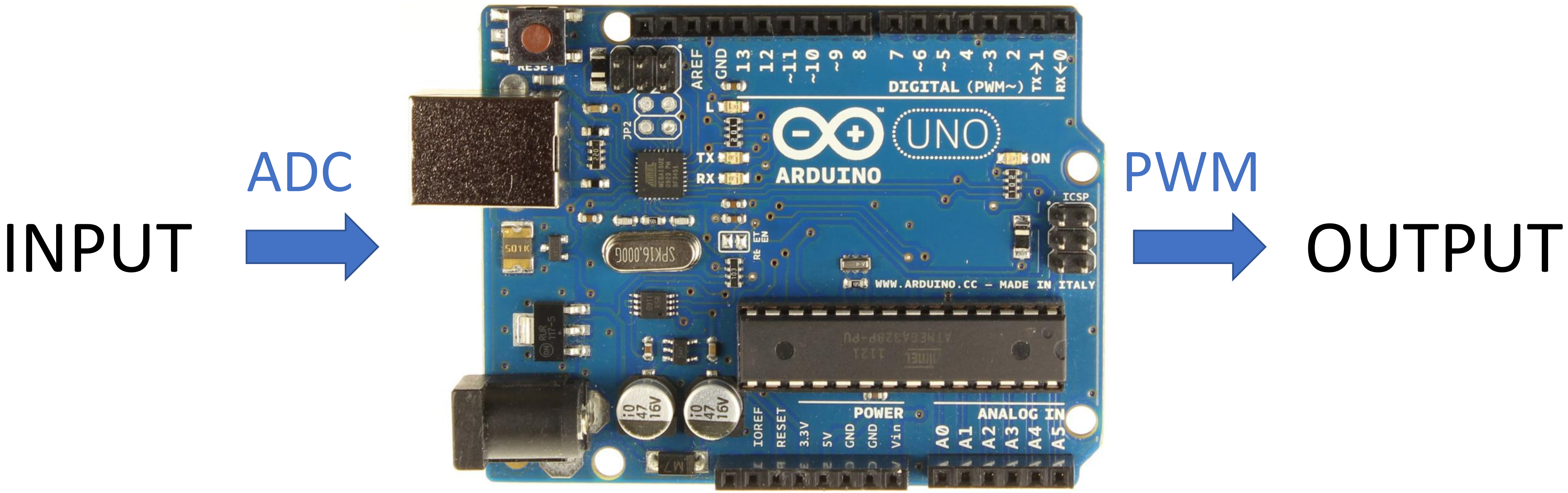


INPUT

ADC

PWM

OUTPUT

ADC and PWM

Wheeler HS Fall 2019

Foundations of Engineering

ADC –
Analog to Digital Conversion

- **Analog** – continuous “real-world” signal
- **Digital** – discrete (incremented) “computer” signal

- Example of ADC: a digital scale
  - An object may way any amount i.e. 100 lbs, 100.1 lbs, 100.00001 lbs, 100.00000001111 lbs etc.
  - However, a digital scale is limited in the numbers it can read: 100 lbs, 100.1 lbs, 100.2 lbs, 100.3 lbs etc

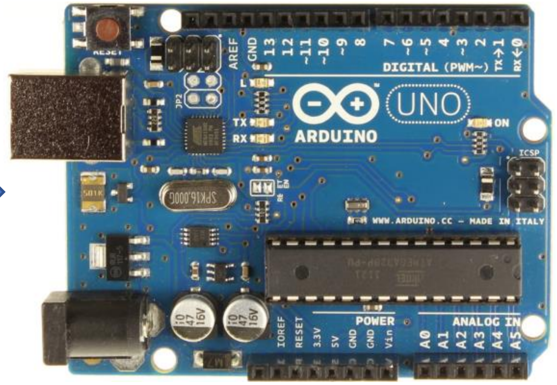
ADC – Why we're learning it

- How a microcontroller "reads" inputs from the real world

Analog

Digital

ADC of temperature

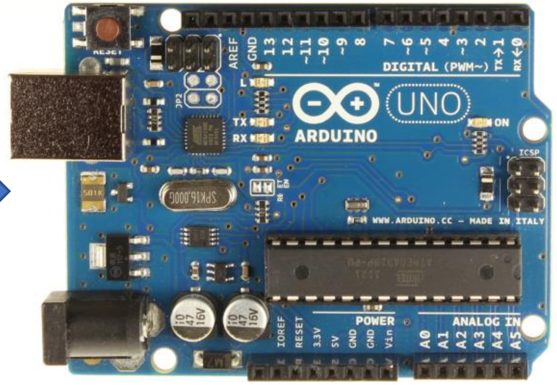


Digital reading of 0-1023

1023 = highest possible temp. reading

0 = lowest possible temp. reading

ADC of voltage



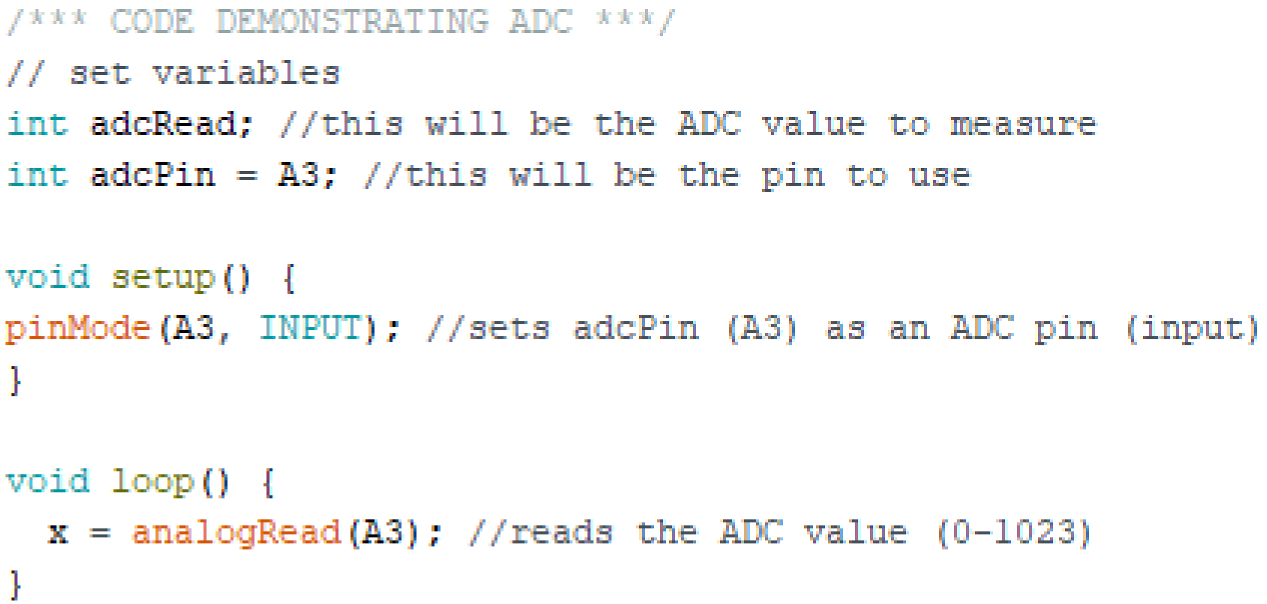
Digital reading of 0-1023

1023 = highest possible voltage reading

0 = lowest possible voltage reading

C/C++ Code for ADC on an Arduino

1. Initialize the ADC
   - pinMode(*adcPin,* INPUT)
2. READ the ADC signal
   - analogRead(adcPin);
   - google “Arduino analogRead” for documentation on this function
3. Connect the ADC input pin to A1-A3

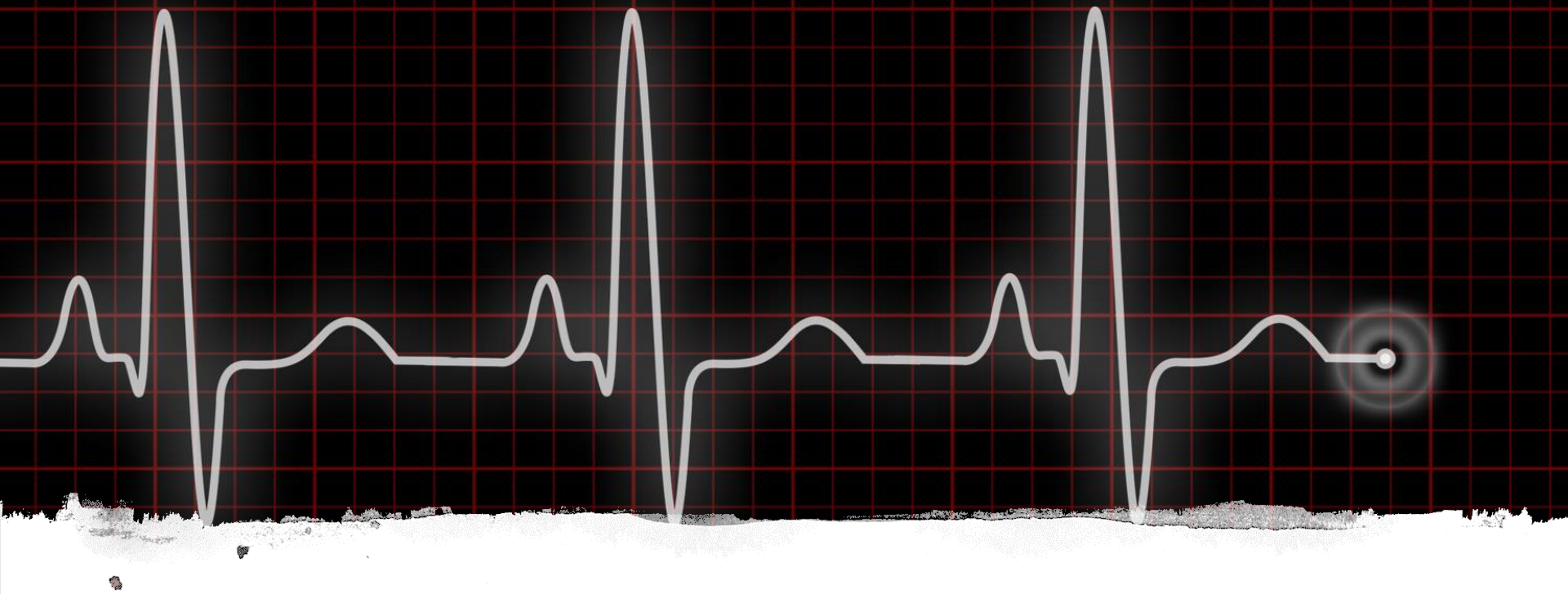


```
/*** CODE DEMONSTRATING ADC ***/
// set variables
int adcRead; //this will be the ADC value to measure
int adcPin = A3; //this will be the pin to use

void setup() {
pinMode(A3, INPUT); //sets adcPin (A3) as an ADC pin (input)
}

void loop() {
  x = analogRead(A3); //reads the ADC value (0-1023)
}
```

Possible ADC pins:
A0-A5

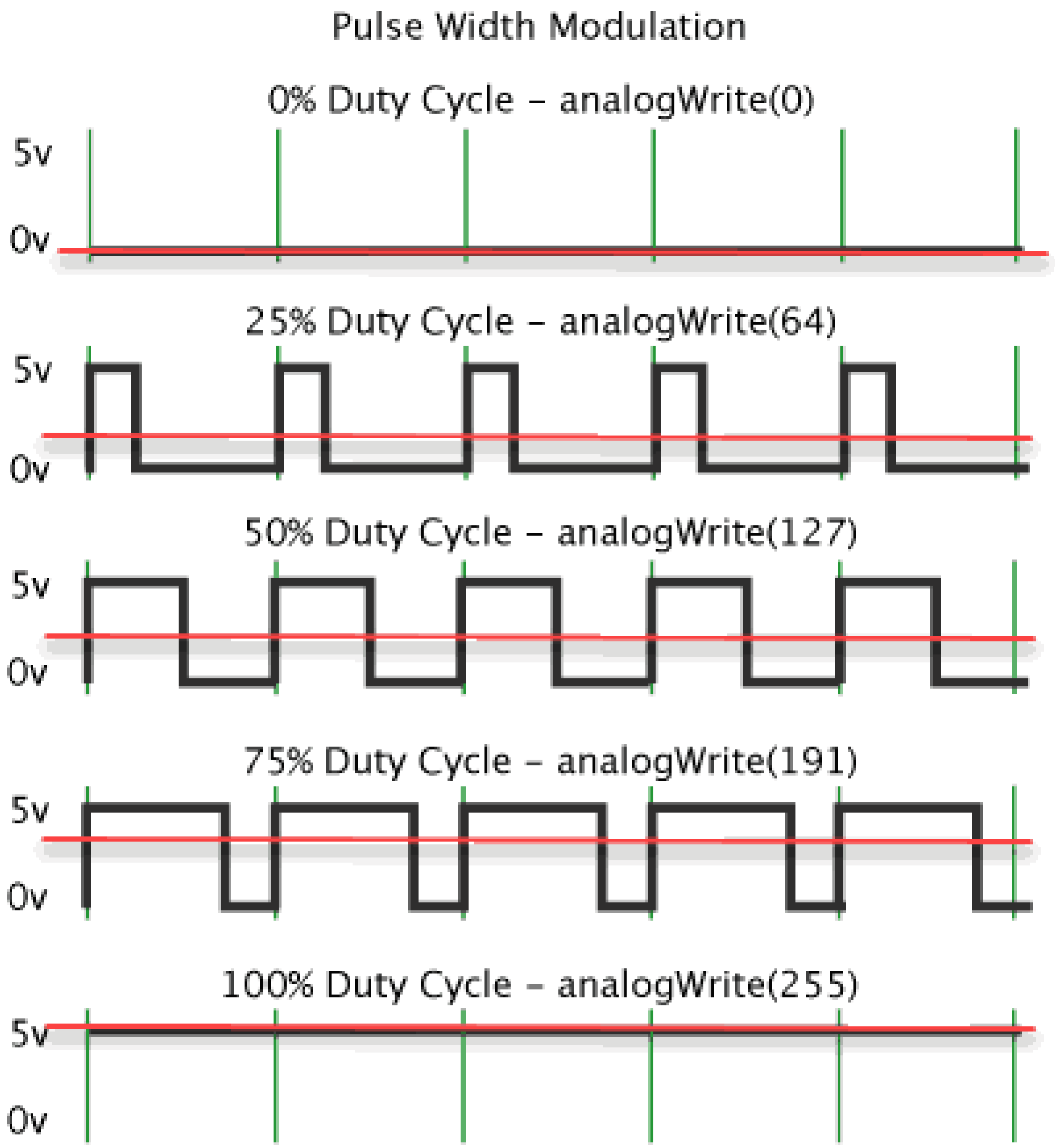


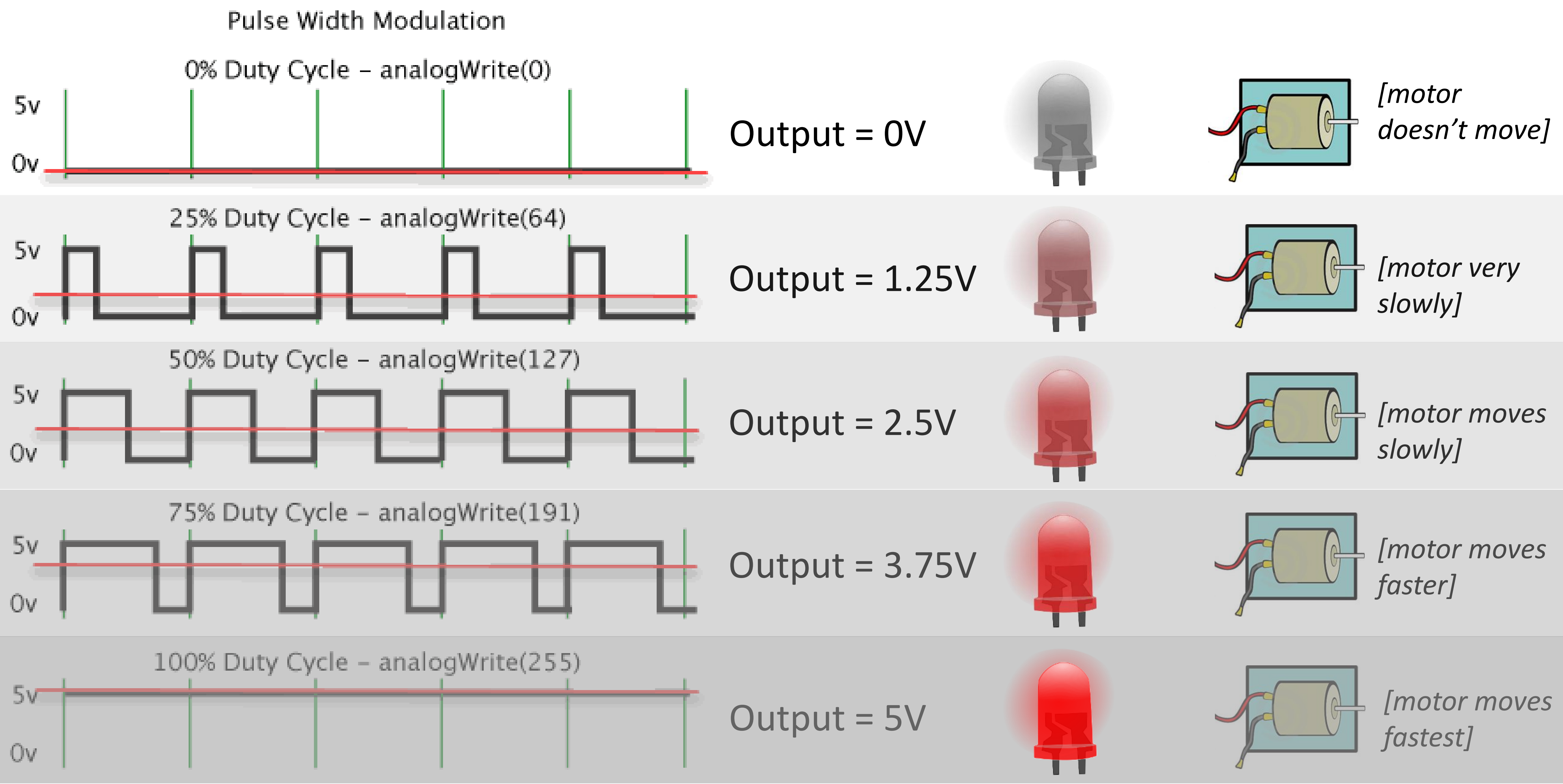
PWM –
Pulse Width Modulation

- Pulse
- Width
- Modulation

PWM – Why We're learning it

- Microcontrollers pins can only output HIGH (5 V) or LOW (0V)
- PWM allows us to choose a specific output value (voltage)


Pulse Width Modulation
0% Duty Cycle – analogWrite(0)
5v
0v
25% Duty Cycle – analogWrite(64)
5v
0v
50% Duty Cycle – analogWrite(127)
5v
0v
75% Duty Cycle – analogWrite(191)
5v
0v
100% Duty Cycle – analogWrite(255)
5v
0v


Pulse Width Modulation
0% Duty Cycle – analogWrite(0)
25% Duty Cycle – analogWrite(64)
50% Duty Cycle – analogWrite(127)
75% Duty Cycle – analogWrite(191)
100% Duty Cycle – analogWrite(255)
5v
0v

Output = 0V

[motor doesn't move]

Output = 1.25V

[motor very slowly]

Output = 2.5V

[motor moves slowly]

Output = 3.75V

[motor moves faster]

Output = 5V

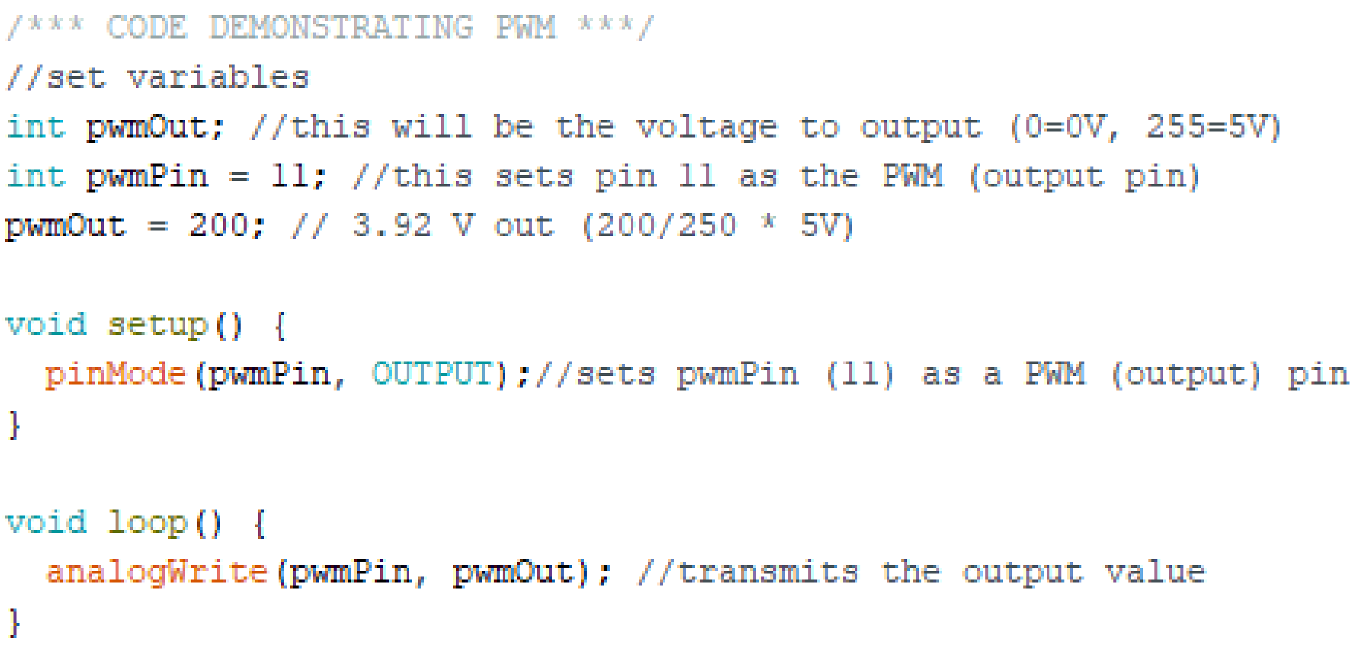
[motor moves fastest]

255 = highest PWM voltage

0 = lowest highest PWM voltage

C/C++ Code for PWM

1. Initialize the PWM
   - pinMode(*inputPin,* OUTPUT);
2. Transmit the PWM signal
   - analogWrite(pwmPin);
   - google “Arduino analogWrite” for documentation on this function
3. Connect the following wires/pins:
   1. ground
   2. one of the PWM pins (1-13)



```
/*** CODE DEMONSTRATING PWM ***/
//set variables
int pwmOut; //this will be the voltage to output (0=0V, 255=5V)
int pwmPin = 11; //this sets pin 11 as the PWM (output pin)
pwmOut = 200; // 3.92 V out (200/250 * 5V)

void setup() {
  pinMode(pwmPin, OUTPUT);//sets pwmPin (11) as a PWM (output) pin
}

void loop() {
  analogWrite(pwmPin, pwmOut); //transmits the output value
}
```

Possible PWM pins:
Digital pins 3, 5, 6,
9, 10, 11

The meaning of 255 and 1023

- The range of ADC values is 0-1023 and PWM is 0-255
  - Why do you think this is?

For you to do:

- Challenge #1 (PWM): Use the Arduino's PWM function to input to control the brightness of an LED through the Serial Monitor Window
- Bonus Challenge #1 (ADC): Use the Arduino's ADC functions to print a digital reading from a power supply on the Serial Monitor Window
  - WARNING: THE ARDUINO CAN ONLY INPUT A MAXIMUM VOLTAGE OF 5 VOTLS. ANYTHING BEYOND THIS WILL BURN THE CIRCUITRY
- Bonus Challenge #2: put the ADC and PWM together; vary the speed of a motor by altering the "knob" (potentiometer) on the power supply