EE 109 Unit 10 - Pulse Width Modulation

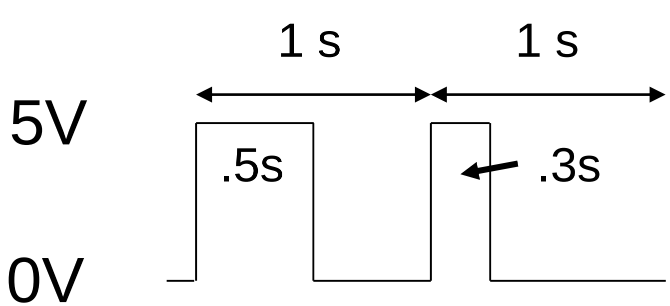
Power

- Recall (or learn) that Power is a measure of:
  - Energy per unit time
- In an electronic circuit, P = I * V
  - Power = Current & Voltage (each may be varying w/ time)
- A circuit that draws a constant 2 mA of current at a constant 5V would consume 10 mW
- Since voltage and current may change rapidly, it is often helpful to calculate the average power

$$P = \frac{1}{T}\int_0^T P(t)dt$$

- Just sum the total power and divide by the total time

I = 1A



Average Power
= (1*5*.8)/2 = 2W

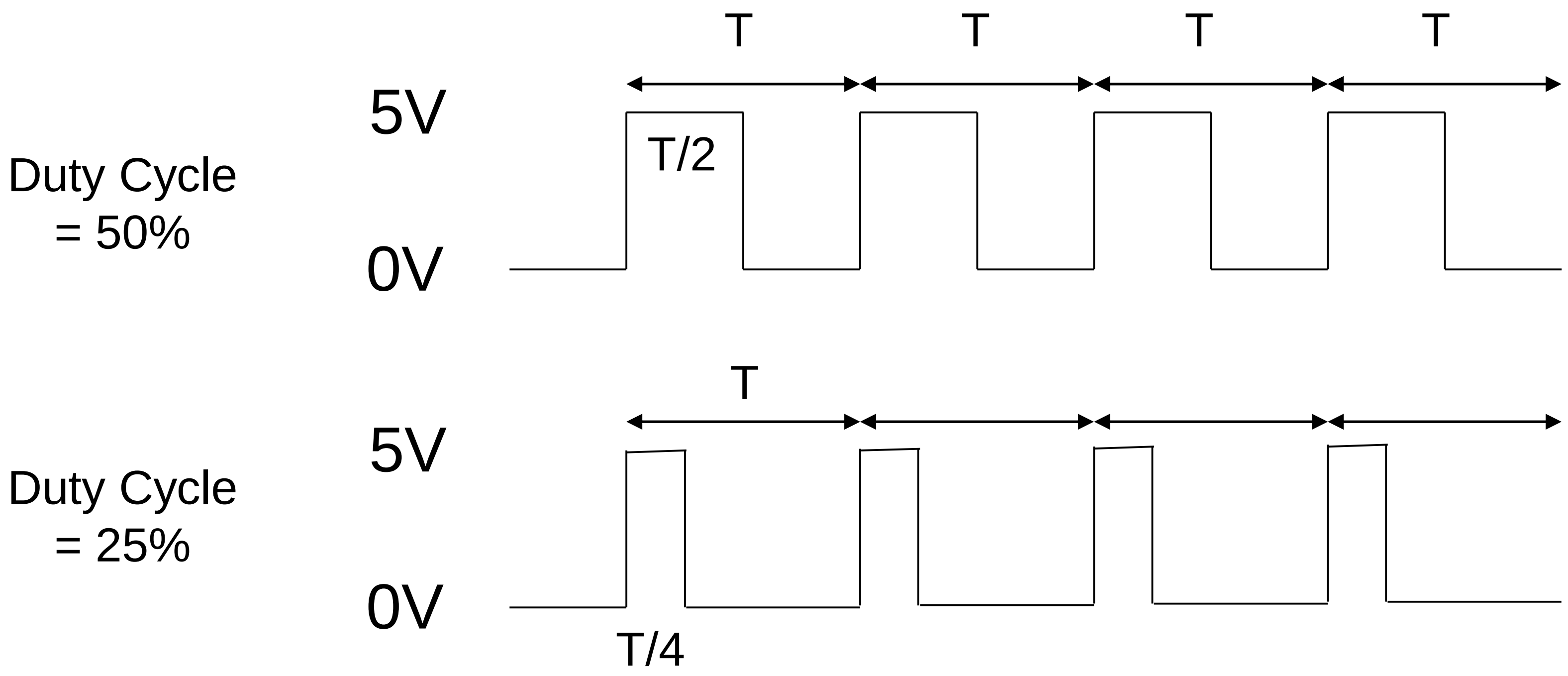
Output Devices

- What do the following have in common?
  - Servo motor that can rotate to any angle w/in 180 degrees
  - Light dimmer
  - Oven or microwave with various power levels
- They are controlled by Pulse Width Modulation (PWM)
  - Usually a 3-pin interface: Power (Vcc), GND, PWM Signal

Duty Cycle

- A pulse is just a short window of time when a signal is 'on'
- We could repeat the pulse at some regular period, T
- We define the duty cycle as

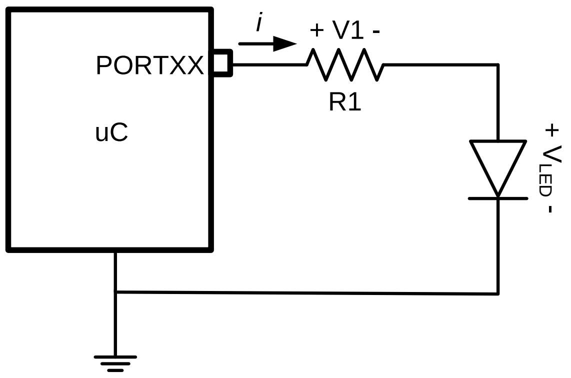
Duty Cycle % = (ON Time / T) * 100

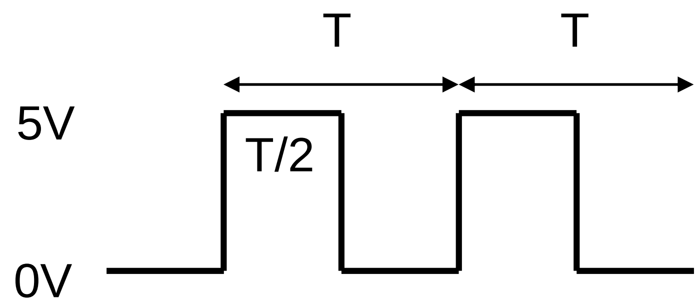


Power & Duty Cycle

- When we light up an LED we often just turn a PORTxx output 'on' and leave it 'on'
  - This supplies the maximum power possible to the LED
- We could pulse the output at some duty cycle (say 50%) at a fast rate
  - Fast so that the human eye can't detect it flashing
  - Average power would be ½ the original always 'on' power
  - Result would be a 'dimmer' LED glow

PORTxx 'on' constantly





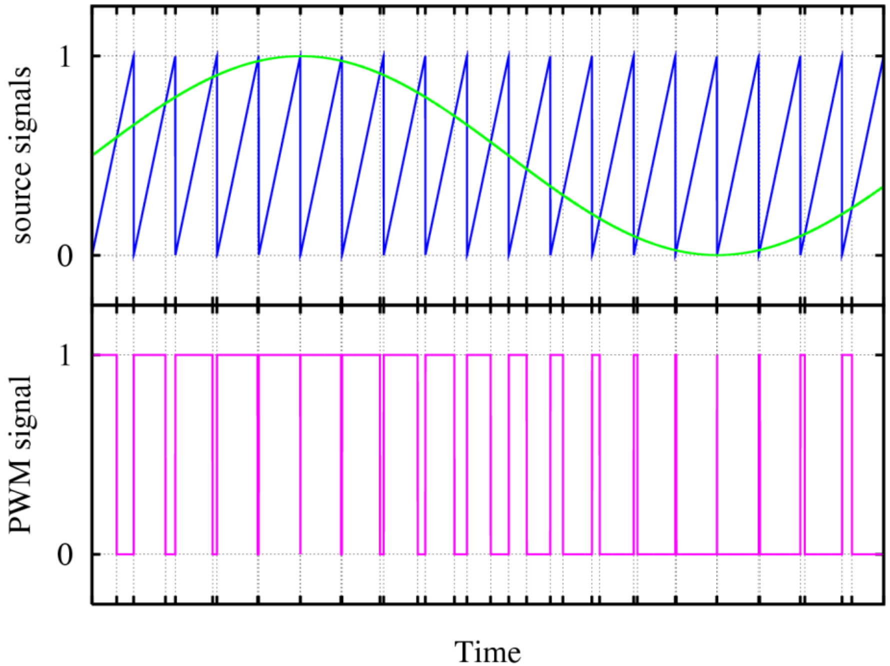
PORTxx 'on' 50% of time

In-Class Activity

- Write a program with a loop that turns on the LED (PORT B5) for x milliseconds and then turns it off for 100-x milliseconds
  - Initially set x = 100
  - Now set x = 50
  - Now set x = 20
  - Now set x = 10
  - Now set x = 2
- Notice result may be non-linear
- A similar tactic is used in your microwave oven when you want to cook something at 80%, 70%, etc. power.

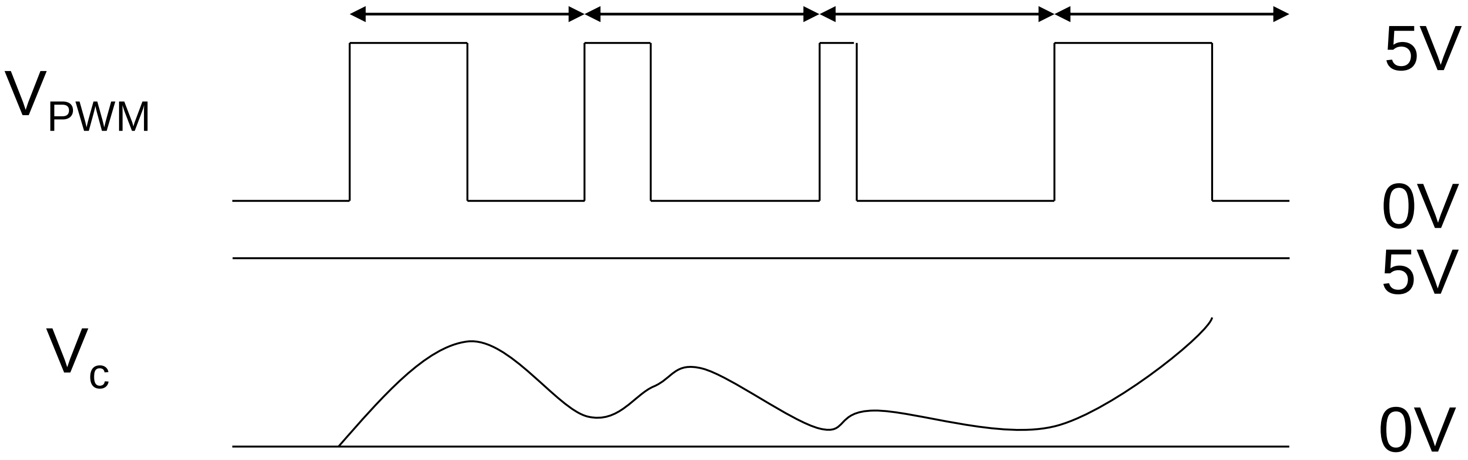
PWM

- Modulation refers to changing a value based on some signal (i.e. changing one signal based on another)
- Pulse width modulation refers to modifying the width of a pulse based on another signal
- It can be used to transform one signal into another
  - Example below of sine wave represented as pulses w/ different widths
- Or it can just be used to alter average power as in the last activity



Simple Digital-To-Analog

- Connecting a PWM output to a resistor-capacitor circuit as shown causes the voltage at Vc to "integrate" the digital PWM signal (charge the capacitor)
  - Analogy: Imagine you have a leaky bucket (i.e. capacitor) and you want to produce a variable level (i.e. analog voltage, Vc) of water by only turning the hose (digital output) on or off

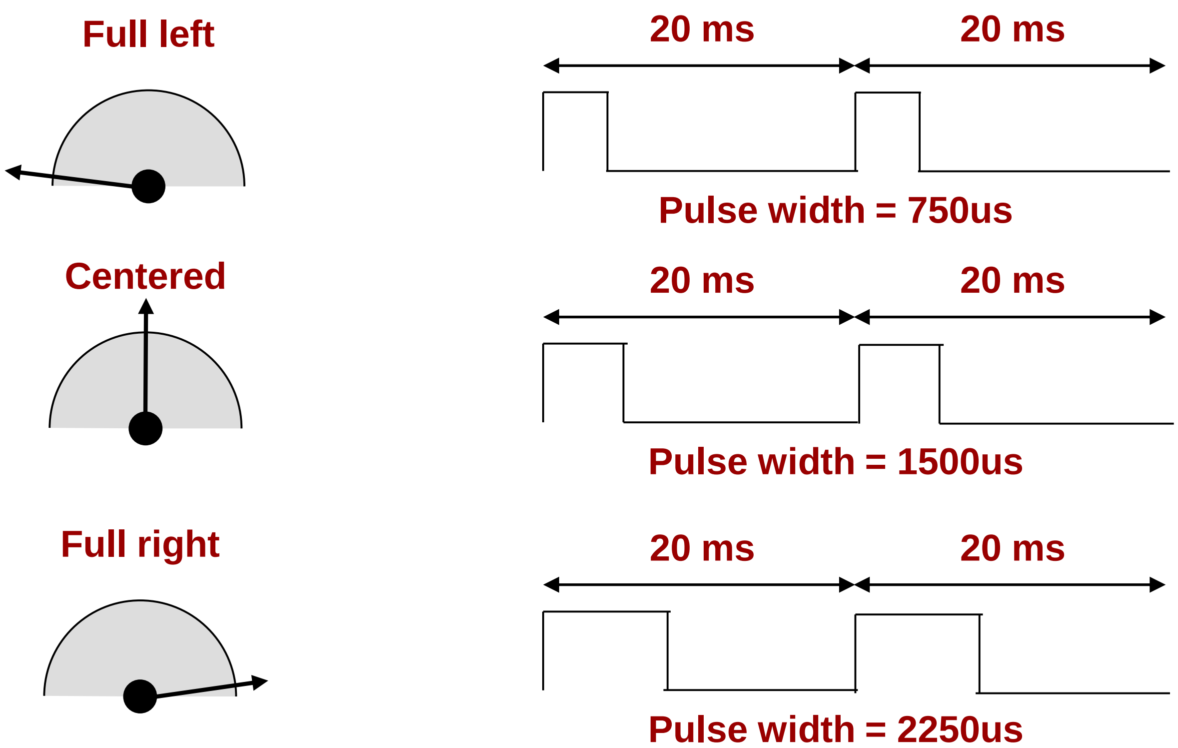


Servo Motors

- Many embedded systems use servo motors to move or rotate mechanical devices
- Most servo motors use some form of pulse width modulation to control the direction and speed of their rotation
- 2 Kinds
  - Standard servo motors: can only rotate through a certain arc (usually 180 degrees)
  - Continuous: can keep spinning round and round while pulses are provided

Standard Servo Motor

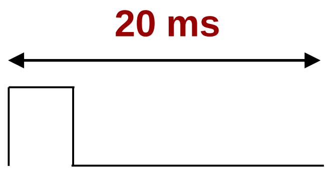
- Pulse width determines angle (position) of servo motor
- Must continue to give pulses for the duration of time it takes to rotate to the desired position
- No pulses = stay put



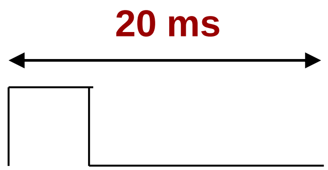
Do an Internet search for Standard Servo Motors & try to find the appropriate pulse width for each position

Continuous Servo Motors

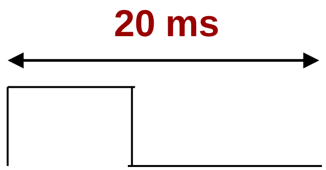
- Pulse width determines speed & direction of rotation
- Controlled via PWM (Pulse Width Modulation)
  - Short pulse = Rotate one direction
  - Medium pulse = Stop
  - Long pulse = Rotate other direction



Pulse Width = 1000 us = Full Speed Clockwise



Pulse Width = 1500 us = Stopped



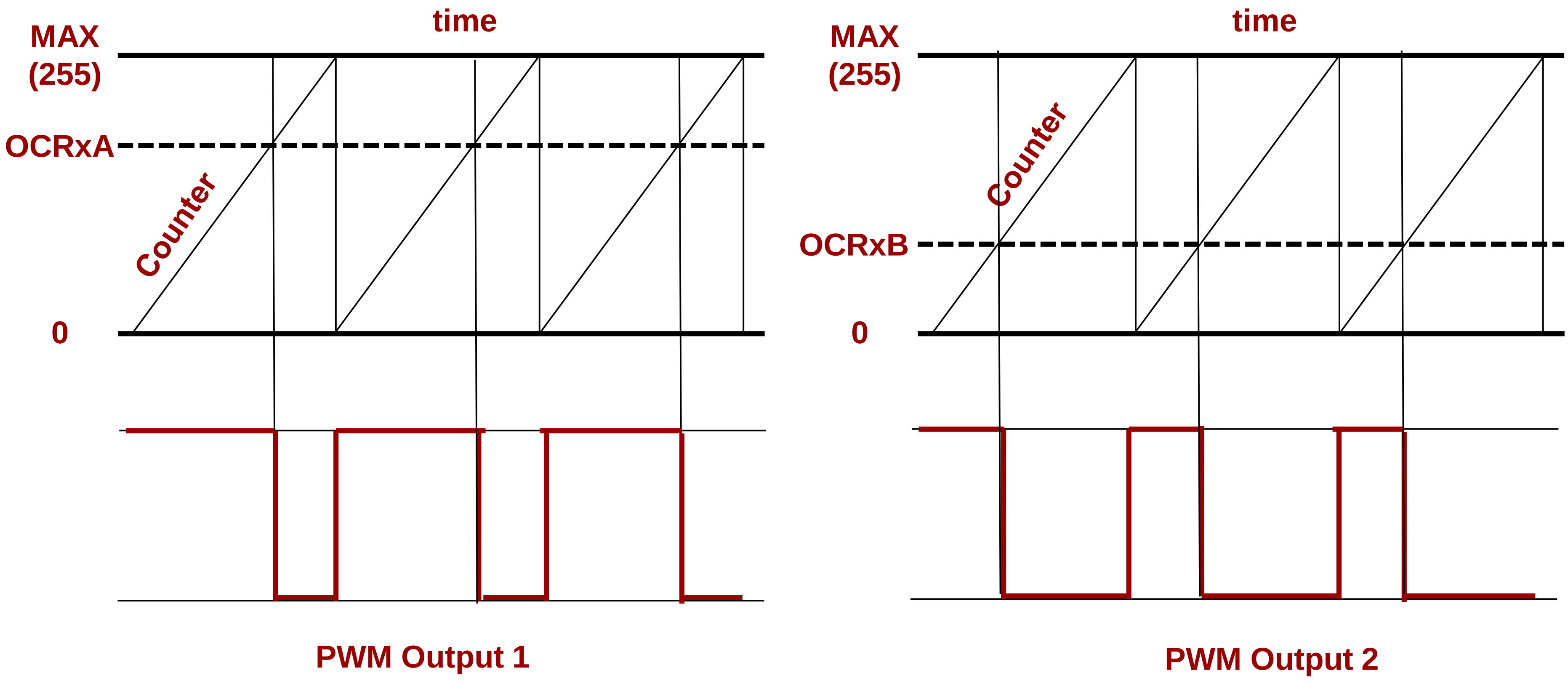
Pulse Width = 2000 us = Full Speed Counter-Clockwise

Implementing PWM

- Can use delays or timers to make your own pulse signals
- Most microcontrollers have hardware to automatically generate PWM signals based on the contents of some control registers
- Many microcontrollers use the Timers to also serve as PWM signals
  - Recall the timer module gave us a counter that would increment until it hit some 'modulus' (MAX) count which would cause it to restart and also generate an interrupt

Using Timers for PWM

- For PWM we can use that counter to just count 0 to some MAX count making the:
  - PWM output = '1' while the count < threshold (OCRxx) and
  - PWM output = '0' when the count >= threshold (OCRxx)



PWM Control Registers

- In this slide packet we will use the 8-bit Timer/Counter0 rather than the 16-bit Timer/Counter1
- Refer to Timer Slides w/ following additions
- Set WGM0[2:0] bits for Fast PWM mode as opposed to CTC
- Timer/Counter0 can produce two PWM outputs on Arduino pins D5 and D6, each with its own threshold value, so you need to pick which one you want to use
  - Bits COM0A[1:0] and threshold register OCRA control operation of output D6 (PORTD6)
  - Bits COM0B[1:0] and threshold register OCRB control operation of output D5 (PORTD5)

See datasheet, textbook or other documentation for further explanation

PWM Control Registers

- Set WGM bits for PWM mode [usually Fast PWM mode] as opposed to CTC
- Pick COM0?[1:0] for desired waveform
- Still need to pick a prescaler to slow down the clock
- Set OCRA or OCRB to the desired threshold which will effectively control the duty cycle of the PWM output

COM0 A1	COM0 A0	COM0 B1	COM0 B0	-	-	WGM 01	WGM 00

TCCR0A Reg.
Timer/Counter0 Control Register

FOC 0A	FOC 0B	-	-	WGM 02	CS02	CS01	CS00

TCCR0B Reg.
Timer/Counter0 Control Register

CS0 [2:0]	Prescaler
010	Clk / 8
011	Clk / 64
100	Clk / 256
101	Clk / 1024

COM0?1, COM0?0	Output Compare pin (assume WGM02=0)
00	Don't use Pin
01	Don't use Pin
10	Set Pin on CTR=0x00, Clear pin on match=OCR?
11	Clear Pin on CTR=0x00, Set pin on match=OCR?

WGM01, WGM00	WGM02=0	WGM02=1 (Ignore)
00	Normal (Counter)	Unused
01	Phase Correct PWM	Phase Correct PWM (Top=OCRA)
10	CTC (Timer)	Unused
11	**Fast PWM (Top=255)**	Fast PWM (Top=OCRA)

Exercise

- Try to use PWM to make your LED glow at various brightness levels similar to what you did earlier with normal digital I/O