Pulse-Width Modulation: Simulating variable DC output

ME 120
Mechanical and Materials Engineering
Portland State University
http://web.cecs.pdx.edu/~me120
Motivation

- Arduino Uno boards do not have arbitrary voltage output, a.k.a. analog output
- Pulse-Width Modulation (PWM) is a common technique for supplying variable power to “slow” electrical devices such as LEDs and DC motors
- PWM is easy to implement and greatly extends the range of control applications with microcontrollers in general and Arduinos in particular
PWM is a variable width pulse train

- The frequency of pulses is fixed
- The width of the pulse is variable

The ratio $\tau_0/\tau_c$ is called the duty cycle
PWM can act as a variable voltage

- If a PWM signal is supplied to a “slow” device, the effective power delivered is proportional to the duty cycle

\[
\text{Power} \approx \text{duty cycle} \times \text{max power}
\]

- Examples of “slow” devices
  - LED: because our eyes are slow
  - DC motors: because of inertia and inductive energy storage

- “Slow” means that the frequency of the PWM pulse train is much faster than the response time of the device
analogWrite(...) produces variable Duty cycle

\[ V_s = 5V \]

\[ \frac{\tau}{\tau_c} = 0.25 \]

\[ \frac{\tau}{\tau_c} = 0.50 \]

\[ \frac{\tau}{\tau_c} = 0.75 \]

http://arduino.cc/en/Reference/AnalogWrite
http://arduino.cc/en/Tutorial/PWM
Arduino Uno Pins 3, 5, 6, 9, 10, 11 for PWM

- The ~ before the pin number indicates PWM capability

http://arduino.cc/en/Reference/AnalogWrite
http://arduino.cc/en/Tutorial/PWM
PWM output is

\texttt{analogWrite( pin, dutyCycle )}

- \texttt{pin} = one of 3, 5, 6, 9, 10, 11
- \texttt{dutyCycle} is an unsigned 8-bit value
  - \( 0 \leq \texttt{dutyCycle} \leq 255 \)

```c
int PWM_pin = 5;    // Digital I/O pin must have PWM capability
                    // Pins 3, 5, 6, 9, 10, 11 on Arduino Uno can do PWM
void setup() {
    pinMode(PWM_pin, OUTPUT);  // Configure I/O pin for high current output
}

void loop() {
    int duty = 127;            // Duty cycle must be in range 0 <= duty <= 255
    analogWrite(PWM_pin, duty); // Adjust duty cycle of output pin
}
```

http://arduino.cc/en/Reference/AnalogWrite
http://arduino.cc/en/Tutorial/PWM
Example: PWM control of LED brightness

- Connect a potentiometer to an Analog Pin
- Connect an LED circuit to a Digital Pin with PWM
//   File:  LED_dimmer.ino
//
//   Use a potentiometer to control the brightness of an LED.
//   Voltage supplied to the LED is a PWM signal w/ variable duty cycle.

int LED_pin = 10;    //  Digital I/O pin must have PWM capability
                      //  Pins 3, 5, 6, 9, 10, 11 on Arduino Uno can do PWM

void setup() {
  pinMode(LED_pin, OUTPUT);  // Configure I/O pin for high current output
  Serial.begin(9600);        // Open serial monitor for diagnostic messages
}

void loop() {

  int duty, pot_reading, pot_pin=A1;

  pot_reading = analogRead(pot_pin);         //  Get potentiometer setting
  duty = map(pot_reading, 0, 1023, 0, 255);  //  map 0-1023 to 0-255
  duty = constrain(duty, 0, 255);            //  Make sure 0 <= duty <= 255
  analogWrite(LED_pin, duty);                //  Adjust duty cycle of output pin

  //  -- Print potentiometer reading and scaled delay as a diagnostic
  //    Printing values does not affect operation of the code.
  Serial.print("Potentiometer = ");
  Serial.print(pot_reading);
  Serial.print("   Duty cycle = ");
  Serial.println(duty);
}