IAT 884 Workshop

Micro-Controllers: Digital Output

Microcontroller Basics

I/O Board

An I/O board is a device that acts as a conduit between various electronic devices. I/O boards generally utilize a microprocessor to analyze and transmit data packets between the attached devices.

Microprocessor

A microprocessor is a silicon chip that contains a CPU. Microprocessors control the logic of almost all digital devices, from clock radios to fuel-injection systems for automobiles.

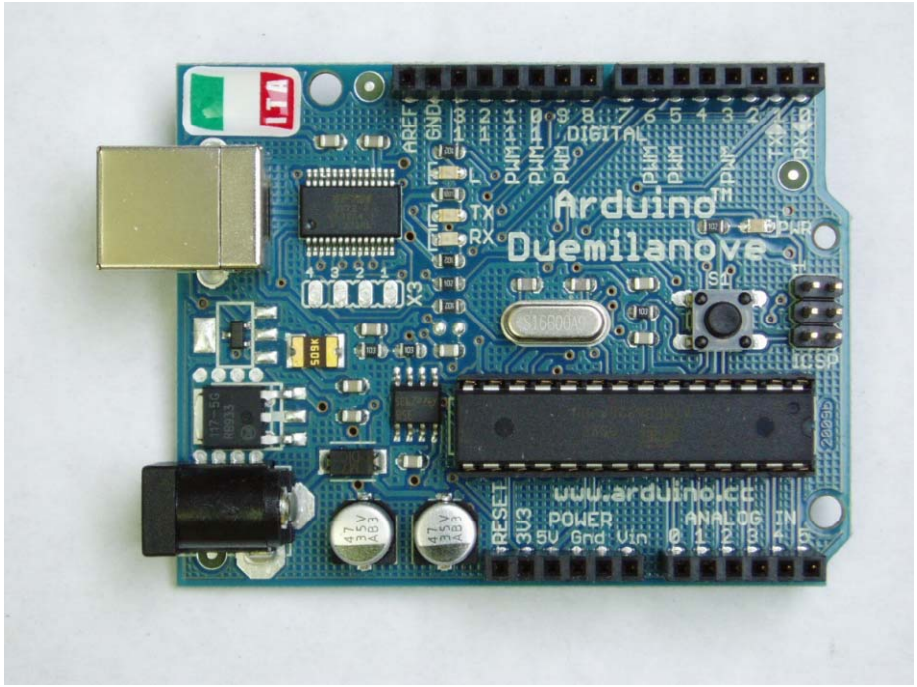
Microcontroller

A microcontroller is a low power consumption, self sufficient microprocessor. They typically integrate read/write memory, ROM memory, and EEPROM for permanent data storage.

The Arduino Board

A cheap, robust I/O board based on the ATmega168 chip.

The Arduino board can function connected to a computer (in Serial communication mode) or as a stand-alone CPU that can drive a hardware application.



Arduino Duemilanove

MicroPRocessor Atmega 328

Operating Voltage 5V

Input Voltage (recommended) 7-12 V

Digital I/O Pins 14 (of which 6 provide PWM output)

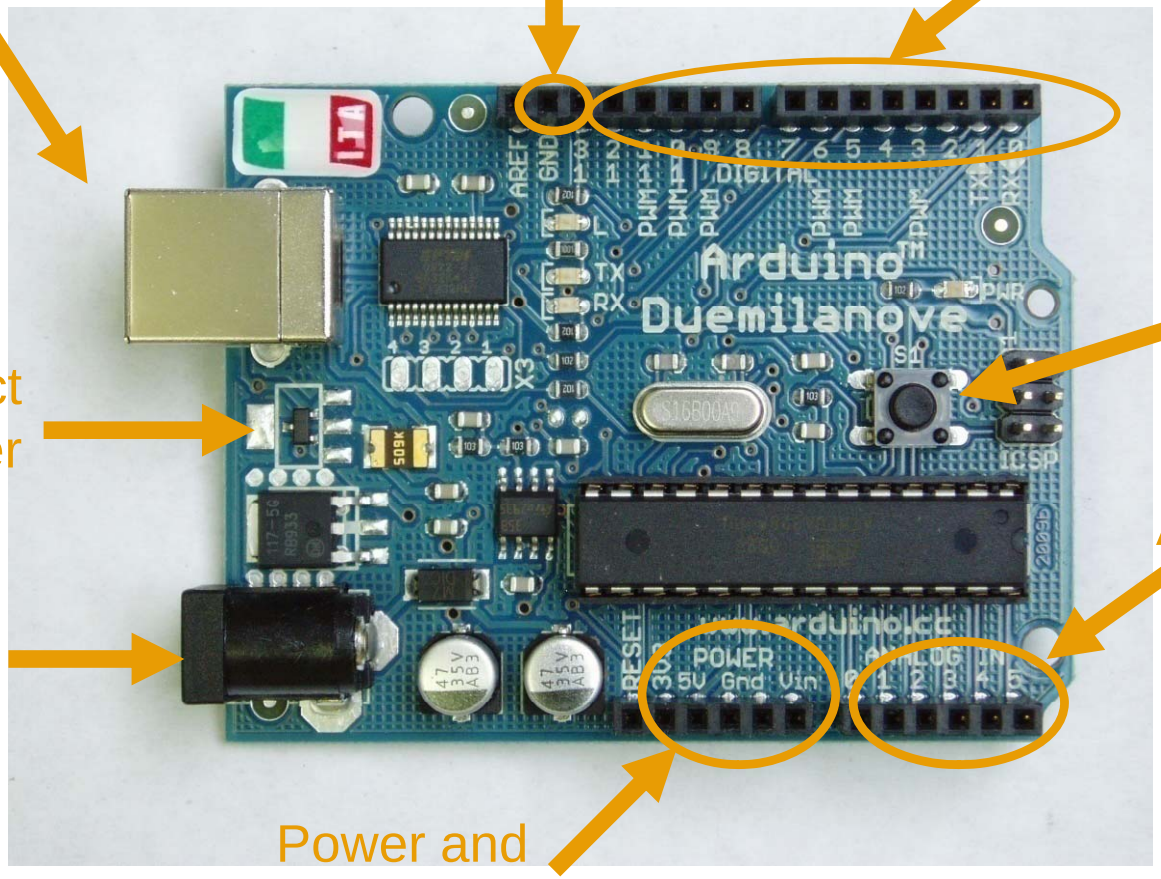
Analog Input Pins 6

DC Current per I/O Pin 40 mA

Flash Memory32KB (of which 2 KB used by bootloader)

Clock Speed 16 MHz

The Arduino Duemilanove



USB Plug
(Power and Communication)

Ground Pin

Digital I/O Pins

Reset Button

Power Select
Jumper

External Power
(7-12 Volts)

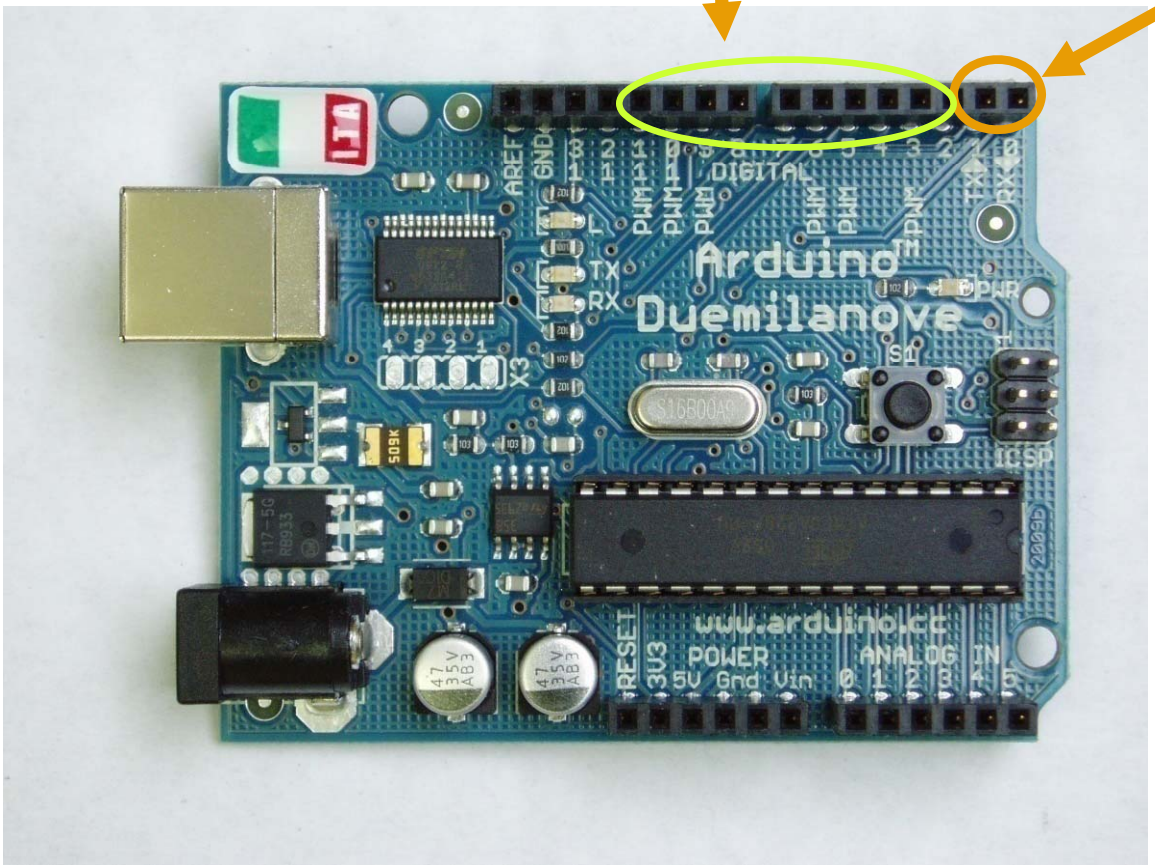
Analog In Pins

Power and
Ground Pins

The Arduino Duemilanove

PWM Pins:
3,5,6,9,10,11

Pins 0 and 1
Reserved for serial communication



Microcontroller Output

What is Microcontroller output used for?

Output from the Arduino can replace the use of a battery for powering electronic circuits

Unlike a battery which has a preset output, the Arduino can be programmed to send out a specific amount of voltage.

The Arduino can output from 0 – 5 volts

What makes using a microcontroller particularly useful is the ability to change the output voltage over time or in response to an action.

For example, the Arduino can be programmed to slowly boost the brightness of an LED or the speed of a DC Motor by increasing the voltage being sent to them.

Microcontroller Output

There are two types of output that the Arduino is capable of:

Digital and **Analog**

Digital works like a switch. It is either:
ON (sending voltage) or
OFF (not sending voltage)

Analog allows for setting the amount of voltage as a specific value from 0 – 5 volts

This is an important distinction since it affects the way that information will be displayed to your user.

Digital output is good for alerts: Is something on? Is there danger?

Analog Output is good for conveying continuous and subtle information: How much? Volume, Speed of rotation, levels.

Digital Out

Binary is **0** or **1**
For digital signals we use **high** or **low.**

High = 5v
Low = 0v

When you set a digital pin to high it will begin sending out 5 volts until it has its state changed to low.

When you set a digital pin to low it will stop sending voltage until it has its state changed to high.

There are 14 dedicated digital out pins on the Arduino Duemilanove. You can additionally use the 6 analog in pins as digital output pins if necessary. These are references as digital pins14-19.

Structure of an Arduino Program

```
Global Variables;

void setup()
 {
     Setup environment variables;
     Setup Pins for digital/analog output or input;
 }

void loop()
{
      This is the stuff that creates the interaction;
      Generally you will either be checking the state of a sensor
      Or changing the state of an attached device (LED, Motor, etc)
      This is also where you call other functions;
}

void function (var 1, var 2)
{
      Do something;
}
```

Using Digital Out

These three lines of code show how you use digital output on the Arduino

First Step

Before you can send anything out you need to tell the Arduino which physical pin you will be using and whether it will sending Analog or Digital out.

This command sets up physical pin 2 for digital output. It always goes in the setup section of the code:

```
pinMode(2, OUTPUT);
```

Second Step

The following two commands actually change the amount of voltage being sent out the pin. This code goes in the loop section.

Set the pin to output 5v: `digitalWrite(2, HIGH);`
Set the pin to output 0v: `digitalWrite(2, LOW);`

Digital Out

EXAMPLE:

```
int ledPin = 13;                        // LED connected to digital pin 13

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

void loop()
{
        digitalWrite(ledPin, HIGH);   // sets the LED on
        delay(1000);                  // waits for a second
        digitalWrite(ledPin, LOW);    // sets the LED off
        delay(1000);                  // waits for a second
}
```

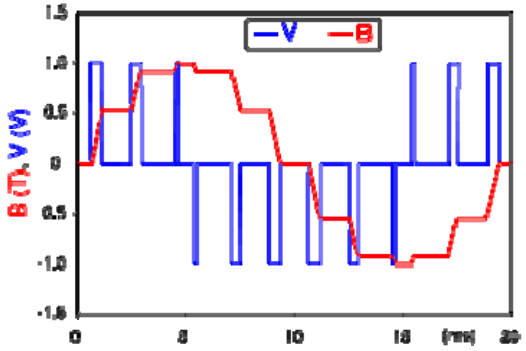
Analog Out

Pulse Width Modulation (PWM)

What's the difference between an analog and digital signal?

Digital output = Discrete (On/Off)
Analog output = Continuous (Any value from 0v – 5v)

PWM allows us to generate an signal that for our purposes is identical to analog.



PWM modulates the duty cycle of a square wave to control the amount of power sent out to a load. On the Diecimila the output signal is at approximately 490 Hz.

Analog Out

You do not need to set PINMODE for analog output pins.

analogWrite(pin_number, value)

Pin number must be one of the PWM pins (3, 5, 6, 9, 10, 11)
Value is between 0 -255 (0v – 5v)

EXAMPLE:

```
void setup()
{
        //nothing needs to be here
}

void loop()
 {
        //send out ~2.5v on Pin 9
        analogWrite (9, 150);
}
```

Analog Out

This code will fluctuate the amount of voltage
(From 0 – 5 volts) being sent an LED on Pin 3

EXAMPLE:

```
int ledPin = 3;

void setup()
{                    //nothing needs to be here
}

void loop()
 {
      for(i=0;i<255;i++);
      {
                  analogWrite(ledPin, i);
                  delay(10);
      }
}
```

Analog Out

Example:

Cycle forward and backward continuously through all the values from 0-255 and output them on pin 3.

```
int ledPin = 3;                    //Set output pin to write to
int val = 0;                       //variable to store the current value to write
boolean up = true;                 //tracks weather the count is up or down

void setup()
{
}

void loop()
{
   if(up){                         //if the count is upward
   val++;                          //increase the value of val
   if(val == 255){                 //if we have reached the max then…
        up = false;                // start counting downward
   }
  }
  else{                            //if we are not counting upwards
       val--;                      //decrease the value of val
       if(val == 0){               //if val is at its minimum value
            up = true;             //start counting upwards
       }
  }
    analogWrite(ledPin, val);      //write the value to the output pin
}
```

Debugging

Serial.print() and Serial.println()

Send data out over the serial communication channel.

This is the primary way to send characters and numbers out from the Arduino board. The output gets printed at the bottom of the screen when the “serial monitor” is activated. Because serial communication is used, the data sent is available to any application capable of retrieving serial data communications. So they also also works as a way of sharing data between the Arduino board and Max/MSP, Processing, Flash and other programming environments.

Serial.println is the same as Serial.print() except it appends both a carriage return character(ASCII 13) and a newline character (ASCII 10).

Syntax:

```
Serial.begin(9600);                    //initiates serial communications
Serial.println(currentValue, DEC);     //Prints number between 0-255
Serial.println(currentValue, BYTE);    //Prints an ASCII character
Serial.println(str);                   //Prints a string
```

Debugging

EXAMPLE:

```
int ledPin = 3;

void setup()
{
Serial.begin(9600);
}

void loop()
 {
      for(int i=0;i<255;i++)
      {
            analogWrite(ledPin, i);
            Serial.println(i);
            delay(10);
      }
}
```

OUTPUT:

```
1
2
3
4
5
6
7
8
9
10
11
…
255
```