

EP1000

Embedded Systems 1

Embedded Systems

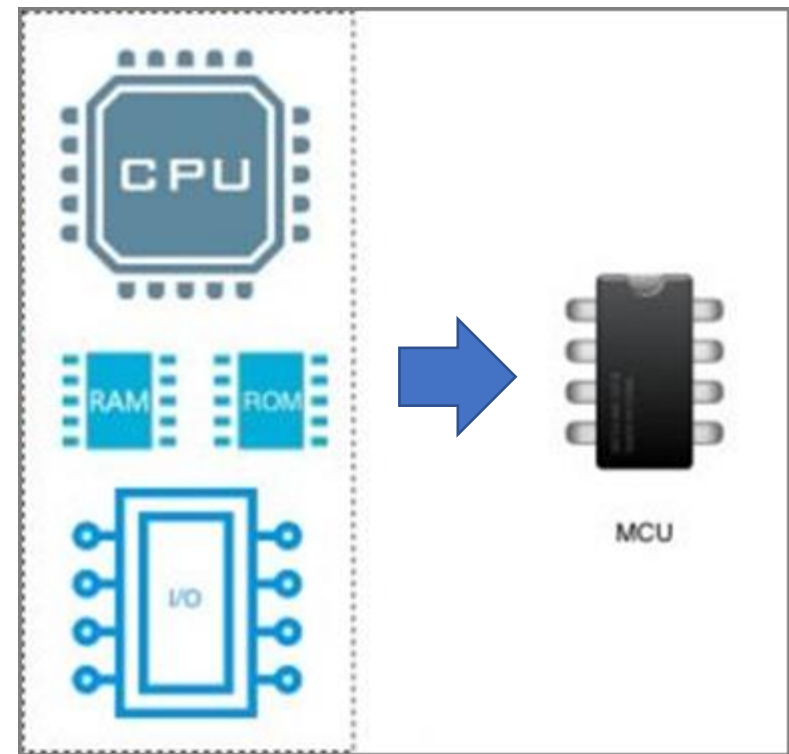
- A dedicated computer system with a dedicated function within a larger system to perform a specific task. E.g. smart TVs, ovens
- After development with a microcontroller, the circuit could be reduced to a single IC with support devices.



AudioQuest Dragonfly audio Digital-Analog Converter

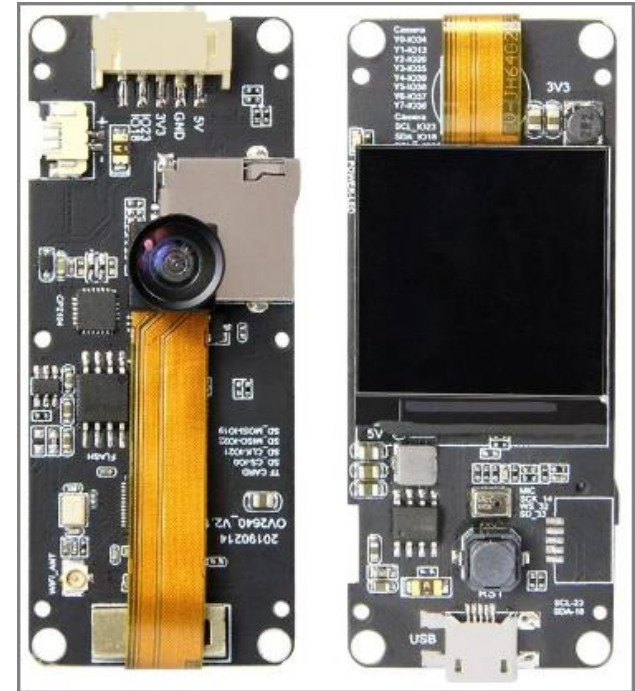
Microcontroller Systems

- Combines
 - Microprocessor
 - Memory (RAM, ROM)
 - Input Output
 - Digital
 - Analog
 - Interface protocols
 - Timing
 - Other peripherals
- Into a **Single** chip



Common Microcontrollers

- Atmel
 - ATTiny [45](#), [44](#), 412, [1614](#)
 - ATmega [328](#), 3u24
- ARM
 - D11C, D21E
- Xtensa
 - [ESP8266](#), [ESP32](#)
 - Integrated RF, Bluetooth, networking



The Arduino Embedded System

- An Arduino Embedded System comprises of:
 - Software & Software Tools
 - Integrated Development system (IDE)
 - Arduino programming language (based on Processing)
 - Development & Debugging Tools
 - Software libraries
 - Hardware
 - ATmel processor boards (and others)
 - Shields (add-on modules)
 - Sensors, actuators, peripherals
 - Open Source Platform

Arduino Systems

- A hardware and software company based in Italy
- Initial development – software (based on Processing) and later hardware boards (Atmel based)
- Produces and markets “official” boards: Uno, Due, Leonardo, Diecimila, Mega, Nano
- Software and hardware is **open source**.

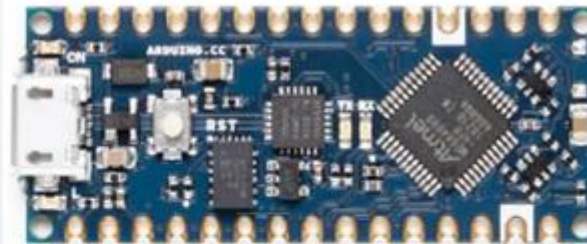


Why use Arduino Systems

- Open source **libraries** & **support**
- **Inexpensive**, lots of clones, 3rd party manufacturers
- Simple, clear programming environment using GUI
- Cross-platform (Windows, MacOSx, Linux)



UNO (atmEGA328)



Nano (atmEGA328)



Digispark
(ATTiny85)

Arduino Uno

- Most common microcontroller board to begin Arduino projects.
- Uses a ATmel Atmega328P processor with a separate programmable interface using another Atmel processor and USB.
- Has sockets for interfacing and power.
- So popular that it is called Arduino.
(Please don't make this mistake!)



“Arduino” or “Uno”

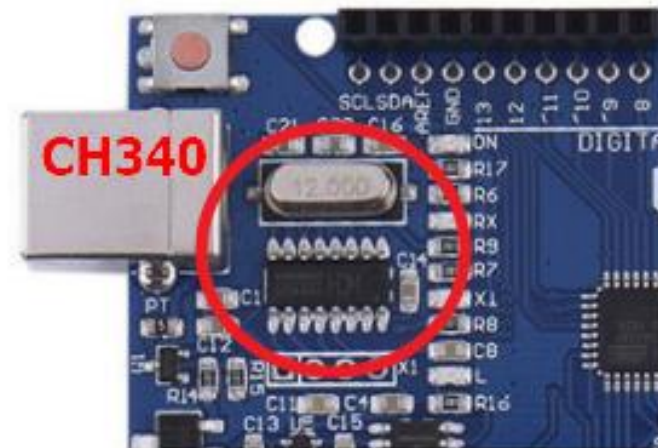
Variations of the Uno

- Being open-source, there are many variations.
- Programming and usage are basically the same with some minor variations.
- All boards use the **ATMega328P** processor (may be in different formats)
- All boards have the same I/O pins
- Difference is in **\$ (cost)**

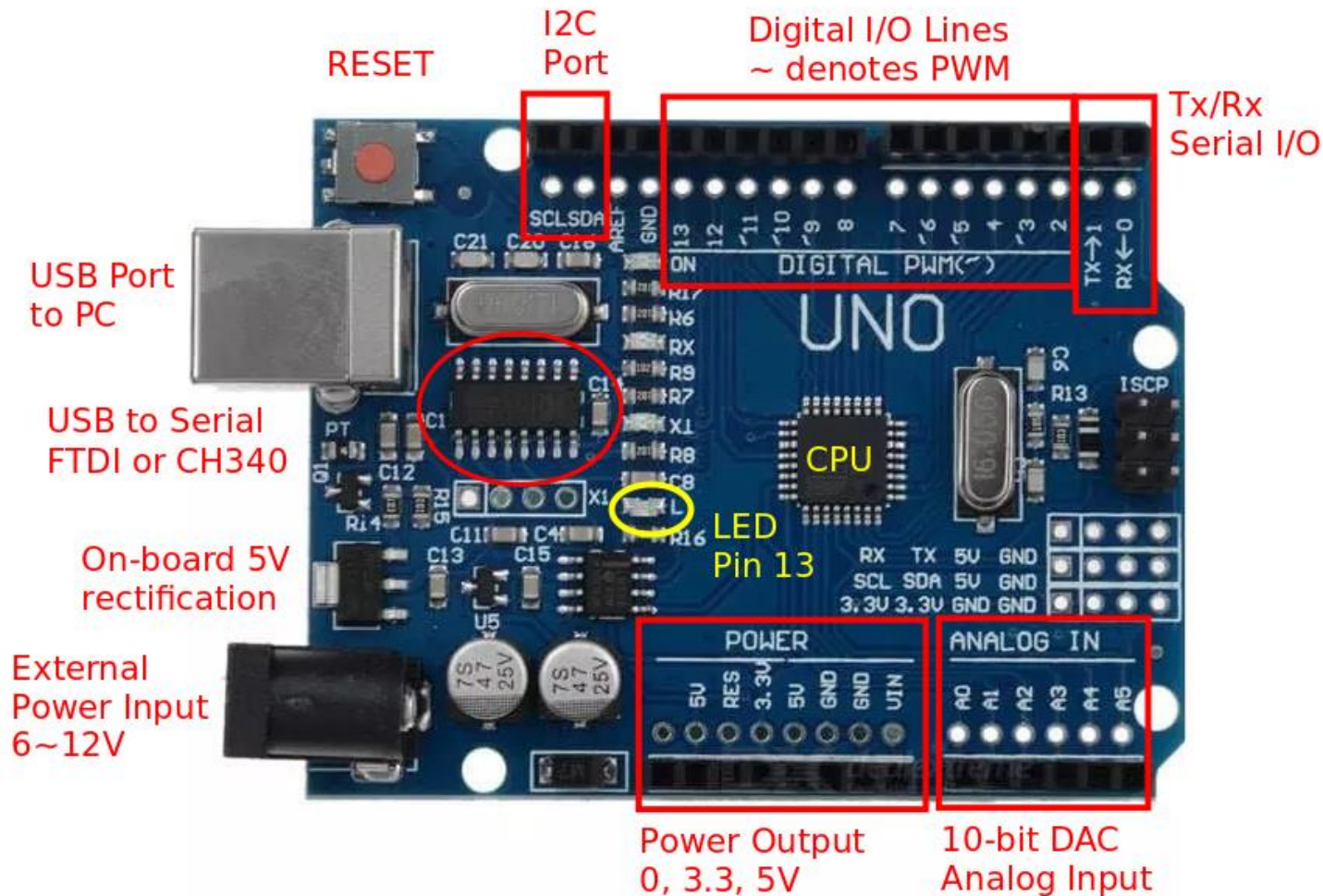


Main Uno board difference

- Expensive boards uses the **FTDI** chip. For USB China clones uses the **CH340** chip
- China clones need to install the [CH340 drivers](#).
- Lots of tutorials from Google



Uno Board Features



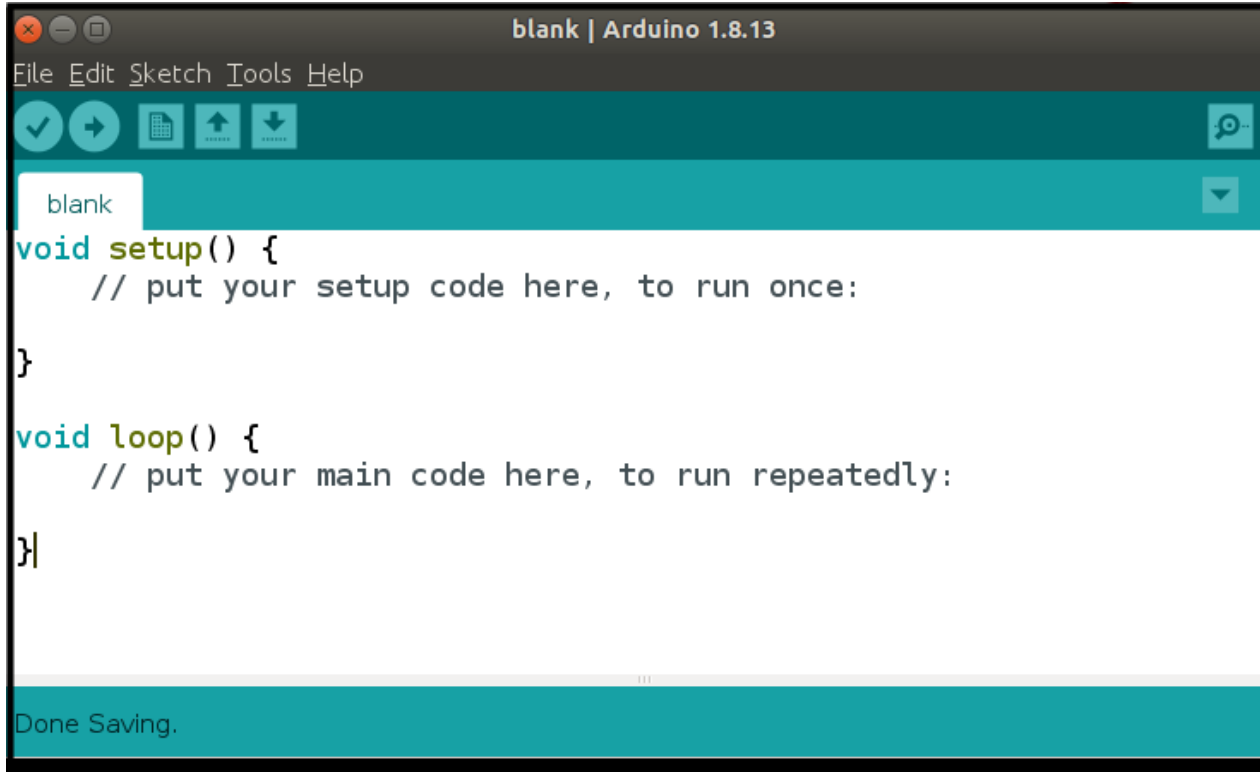
Uno Board Features

- Power
 - Can be powered from USB port ($<0.5A$)
 - External power from DC jack 6V ~ 12V
 - Power outputs: 0 (GND), 3.3V, 5.0V (up to 0.3A), Vref (5V)
- Digital Input/Output pins
 - Can be configured as Input, Output or Pull-Up Inputs
 - Has Pulse-Width-Modulation on ~ indicated Pins
 - Built-in LED on Pin13
- Analog Inputs
 - 10-bit Analog inputs
- Others
 - Serial I/O interface
 - I2C interface
 - SPI interface

Programming the Uno

- Requires an Integrated Development Environment:
 - Download from [Arduino.CC](https://www.arduino.cc)
 - Suggest using v2.0 BETA because of features
 - Syntax highlighting
 - Auto Assist in typing keywords (CTRL-SPACEBAR)
- Arduino programs are called **sketches**.
 - In text
 - Have extension **.ino**

Programming: Getting Started



The screenshot shows the Arduino IDE window titled "blank | Arduino 1.8.13". The menu bar includes "File", "Edit", "Sketch", "Tools", and "Help". The toolbar contains icons for checking, running, uploading, and downloading. The main text area shows the following code:

```
blank
void setup() {
  // put your setup code here, to run once:
}

void loop() {
  // put your main code here, to run repeatedly:
}
```

A status bar at the bottom indicates "Done Saving."

- Saved as sketches (extension .ino)
- Follows most characteristics of a C++ syntax/structure
- Always has 2 functions:
 - `setup()` – code is only executed once
 - `loop()` – code is continuously looped

setup()

- Executed only **ONCE** after each powerup or reset of the UNO.
- UNO is automatically reset after each successful uploading of sketch
- Contains
 - initialization code
 - initialization of variables
 - setup and configuration of I/O ports
 - setup of other interfaces
- Tip: Use identifiers to name your I/O pins, it makes programming and code recognitions much easier

loop()

- The loop() function is executed **after** the setup() code completes.
- Loops **infinitely** executing code within the loop() function.
- Place your code/program within this function, there is **NO** stopping this code

Coding Tips

- Arduino code is based on C++ programming language (actually [Processing](#))
- Follow good C++ programming habits:
 - use comments
 - indent your code
 - use UPPERCASE to denote constants or defines
 - name your variables intelligently

Digital Input/Output

- ATmega328 has 14 digital I/O Pins
 - labelled pin0 to pin13
 - Pin0(Tx), Pin1(Rx) are assigned as Serial I/O
 - Pin3, Pin5, Pin6, Pin9, Pin10 and Pin11 have PWM functionality
- Digital values
 - 0 (0 V, GND, ON, LOW)
 - 1 (5 V, Vcc, OFF, HIGH - typically > 3.3 V)
- Some pins are **multifunctional**, i.e. have different functions depending on how they are initialized.
 - inputs (default)
 - inputs with pull-up resistors
 - outputs
 - Pulse Width Modulation outputs

Digital Input/Output

- Arduino provides [useful functions and libraries](#) simplifying these operations:
 - `pinMode()` - initialises the pin
 - `digitalRead()` - reads/inputs a digital value
 - `digitalWrite()` - outputs a digital value
 - `analogWrite()` - outputs a PWM

Digital Output

- Let's blink (turn ON/OFF) the on-board LED
 - `pinMode()` initialises pin13 to be an **OUTPUT** pin
 - `digitalWrite()` outputs a 0 or a 1 alternately
 - delay of 1 second to allow us to view the result



```
blank | Arduino 1.8.13
File Edit Sketch Tools Help
[Icons] [Search]
blank §

#define LED 13      // give a name to the LED pin

void setup() {
  pinMode(LED, OUTPUT);
}

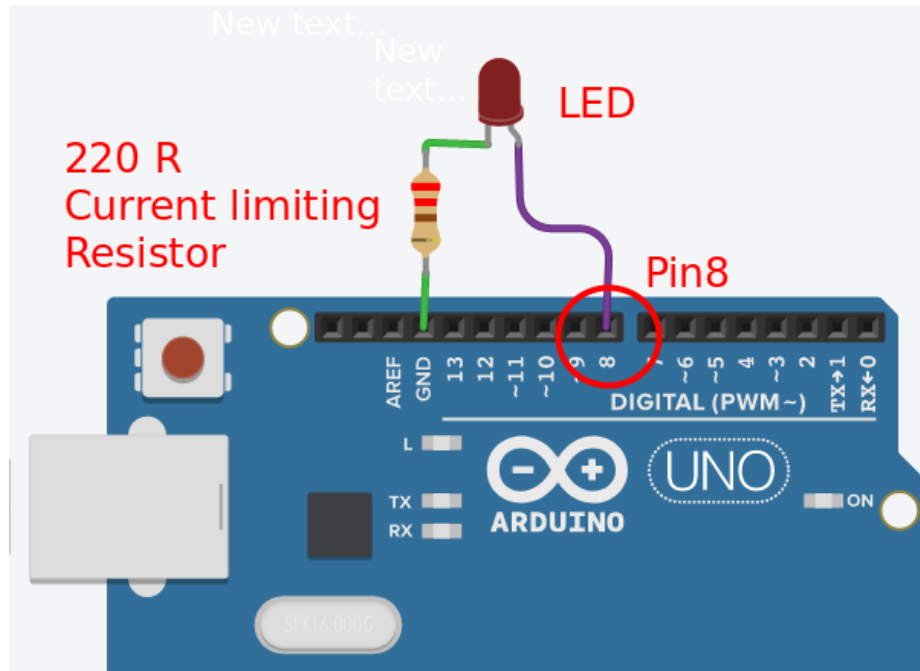
void loop() {
  digitalWrite(LED, LOW);
  delay(1000);
  digitalWrite(LED, HIGH);
  delay(1000);
}

Done Saving.
```

pinMode(pin, MODE)

- Configures specified pin to behave either as input or output.
- Modes available
 - INPUT
digital input mode (high-impedance states)
 - INPUT_PULLUP
digital input mode with internal 20K-50K ohm pull-up resistor
 - OUTPUT
digital output mode
able to source up to 40mA per pin, total of 200mA per chip

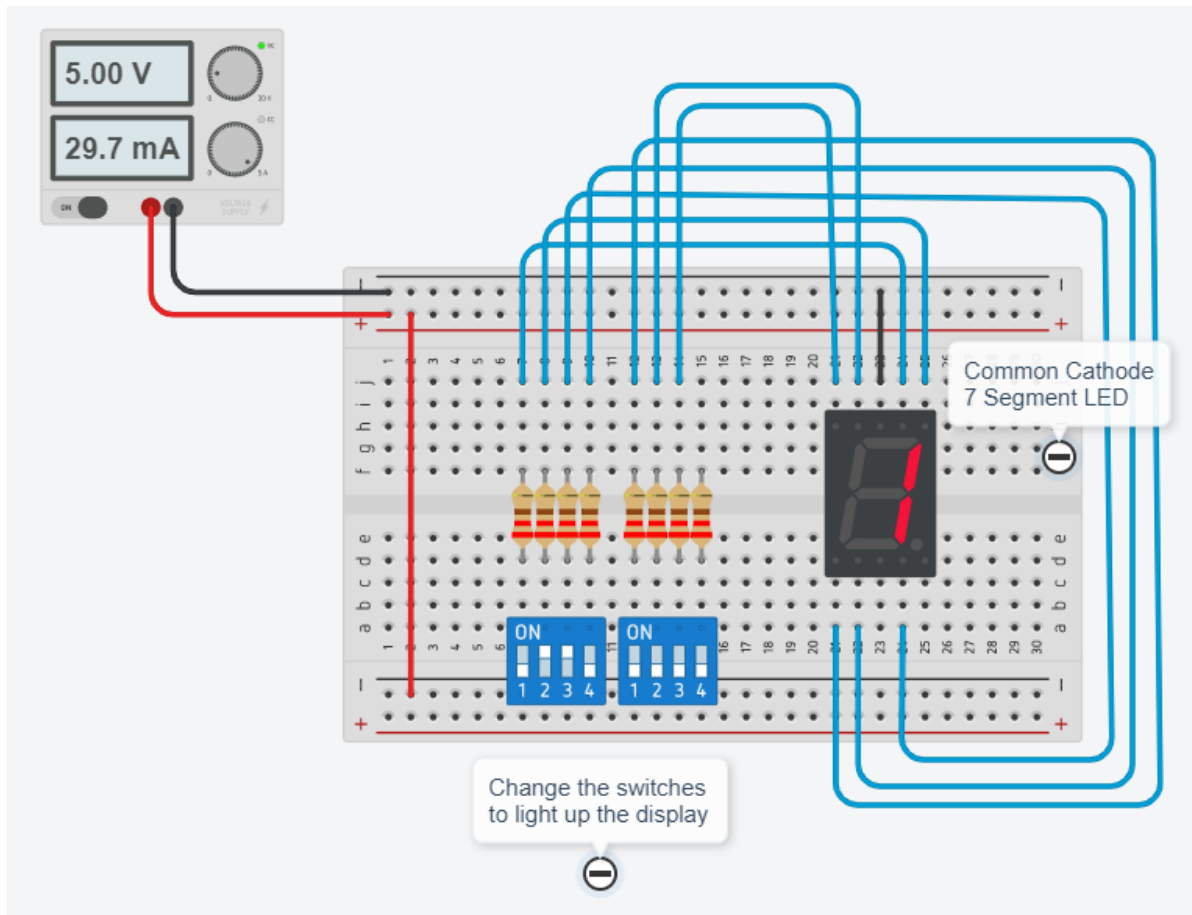
Using an External LED



- Let's blink an external LED
 - need to **WIRE-up** the circuit
 - requires a **current-limiting** resistor
 - change the pin to **8**
 - Compile, Upload, Execute!

What is the purpose of the resistor?
 Which pins can be used as output pins?
 Can you add additional leds?

Exercise: 7 Segment LED Timer



- 7 segment LED
 - Made up of 7 separate LEDs
 - Combined in segments, forms digits
- Change the switches to see the effect

Tinkercad: CC 7 Seg LED

Assignment: A countdown timer

1. Experiment with the 7-segment circuit
2. Convert the circuit which uses an Uno to control the 7-segment CC display
3. Write a program that counts from 0 to 9 continuously

[Why I should use Arduino IDE 2.0?](#)

[Why can't I use a single resistor at the CC pin?](#)

Review: This lesson

- Review the notes (Introduction to Arduino Systems)
- Install Arduino IDE (ver 2)
- Use TinkerCAD, play with Uno + single LED
- Use TinkerCAD, open Uno CC 7 Seg display
 - Check out codes of digits 0..9
 - Replace switches with UNO
 - Program UNO to display 0..9 continuously, 1 second between digits

EP1000

Embedded Systems 1

End