SEICHE 2022
Basic Arduino Programming

Instructor: Paul Frommeyer

www.paulfrommeyer.com

Corporate Sponsor: DXC Technology

YOUR INSTRUCTOR

Career

- Computer Infrastructure Engineer (30+ yrs)
  - Apple Computer – Sony – General Magic
  - Cisco Systems – HP/HPE/DXC [current]
- Linux/Unix Sysadmin (20+ years)
- C developer (professionally just entry level)
- BS Computer Science, BS Philosophy

Personal

- Ubergeek-at-large
- Computers, electronics, woodworking
  - Electronics since 6th grade
  - Computers since 7th grade

“Lesson” 1 – Intro and Setup

Part A

- Introduction to class format
- Overview of lesson plan
- Presentation format (monitor, camera, screen, whiteboard)
- Inventory of USB Drives

Part B

- Installation of device drivers (Windows)
- Installation of Arduino IDE

Class Format and Goals
“Detailed understanding of basic C language programming”

Goals

- IDE software installation
- Installing and configuring new boards
- Installing and using software libraries
- Loading software onto a board
- Accessing and using example programs
- Understand how computers represent numbers and letters
- Arduino C++ programming basics
- Using an LED matrix display

To Be Avoided

- Detailed electronics knowledge and physical wiring of components
- Processor internals and microcontroller internal architecture
- Extensive math, computer science, or EE

The Maker and Hacker Approach

The maker culture is a contemporary subculture representing a technology-based extension of DIY culture that intersects with hardware-oriented parts of hacker culture and revels in the creation of new devices as well as tinkering with existing ones. The maker culture in general supports open-source hardware. Typical interests enjoyed by the maker culture include engineering-oriented pursuits such as electronics, robotics, 3-D printing, and the use of computer numeric control tools, as well as more traditional activities such as metalworking, woodworking, and, mainly, its predecessor, traditional arts and crafts.

The subculture stresses a cut-and-paste approach to standardized hobbyist technologies, and encourages cookbook re-use of designs published on websites and maker-oriented publications.[1][2] There is a strong focus on using and learning practical skills and applying them to reference designs.

– *Wikipedia*

The hacker culture is a subculture of individuals who enjoy the intellectual challenge of creatively overcoming the limitations of software systems to achieve novel and clever outcomes.[1] The act of engaging in activities (such as programming or other media[2]) in a spirit of playfulness and exploration is termed hacking. However, the defining characteristic of a hacker is not the activities performed themselves (e.g. programming), but how it is done[3] and whether it is exciting and meaningful.[2]

…

Richard Stallman explains about hackers who program:

> ***What they had in common was mainly love of excellence and programming. They wanted to make their programs that they used be as good as they could. They also wanted to make them do neat things. They wanted to be able to do something in a more exciting way than anyone believed possible and show "Look how wonderful this is. I bet you didn't believe this could be done."[7]***

Hackers from this subculture tend to emphatically differentiate themselves from what they pejoratively call "crackers"; those who are generally referred to by media and members of the general public using the term "hacker", and whose primary focus—be it to malign or for malevolent purposes—lies in exploiting weaknesses in computer security.[8]

– *Wikipedia*

Lesson Plan Overview

Lesson 1 – Intro and Setup
[may require 2 classes]

- Introduction to class format
- Overview of lesson plan
- Presentation format (monitor, camera, screen, whiteboard)
- Review of microcontrollers and types of boards
- SEICHE LED display architecture
  - ESP8266 pinout
  - High level architecture

Lesson 2 – Laptop operation review – Windows and Linux

- Inventory of USB drives
- Installation of Arduino IDE software
- Installation of CH340/ESP8266 serial port drivers (Windows only)
- Control panel/settings location
- Home directories and folder hierarchy
- Arduino file locations
- Search functions
- (Windows) Device Manager
- (Linux) Konsole
- Copying flash drive contents [critical]
- Open questions and issues

Lesson 3 – IDE essentials

- Starting the Arduino IDE
- Basic Arduino sketch (program) structure
- Loading example sketches
- Loading and configuring new boards
- Connecting boards
- Identifying the microcontroller serial port
  - Linux
  - Windows
- Libraries
- Uploading a sketch
- The serial port monitor
- Printing to the serial port monitor
- A note on brace formatting

Lesson 4/5 – The Binary Number System (may take 2-3 lessons)

- Numerals vs numbers
- Review: the base 10 system and digit place values
- New: the base 2 system and digit place values
- Counting to 1000
- Bits and bytes and nybbles
- Binary addition and subtraction
- Formatting printed output
- Shifting and exponents

Lesson 6/7 – Integers and Arithmetic Operators (may take 2 lessons)

- **The integer data type**
- **Variables and the assignment operator**
- **Autoincrement and autodecrement**
- **Arithmetic Operators**
- **Assigning arithmetic results to a variable**
- **Bytes as data types**

Lesson 8 – Communicating with Sensors

- **Reading a potentiometer**
- **Reading a button**
- **Initializing I2C sensors**
- **Reading I2C sensors**

Lesson 9 – Characters and Arrays

- The hexadecimal number system
- Representing characters as numbers
- The ASCII character code
- The char data type
- Strings and character arrays

Lesson 10 – Comparison Operators

- ==, =>, <=
- strcmp
- Binary comparisons
- Uploading a sketch to the microcontroller

Lesson 11 – Control Structures

- if-then-else
- while
- for-next

Lesson 12 – Programming the LED matrix

- Initializing the SPI interface
- Controlling display brightness
- Lighting and clearing a single pixel
- Displaying text on the LED matrix
- Default fonts
- Nested for-next loops

USB Flash Drive Inventory

DOCUMENTATION – Arduino Reference

- Programming with Arduino.pdf
- Arduino Cookbook-2ndEdition.pdf
- Arduino-For-Beginners-REV2.pdf
- IntroArduinoBook-AlanSmith.pdf
- IntroductionToArduino-Book-AlanGSmith.pdf
- Make_Getting_Started_with_Arduino_3E.pdf

DOCUMENTATION – Electronics Reference

- the-original-guide-to-boards-2021.pdf
- Basic Electronics-Semiconductors.pdf
- Grobs Basic Electronics 2010.pdf
- Instructables-Basic-Electronics.pdf
- Intro to Electronics-Noisemantra.pdf
- Make Electronics 2nd Edition by Charles Platt.pdf
- SPIE-TT107-PracticalElectronicsforOpticalDesignandEngineering-Chapter1.pdf

DOCUMENTATION – ESP8266

- ESP8266 pinout diagram

DOCUMENTATION – Boards Guides

- Original Boards Guides from 2019-2022

DOCUMENTATION – Class Information

- SEICHE-BasicArduinoProgramming-Lesson1-V1.pdf

Critical Documents

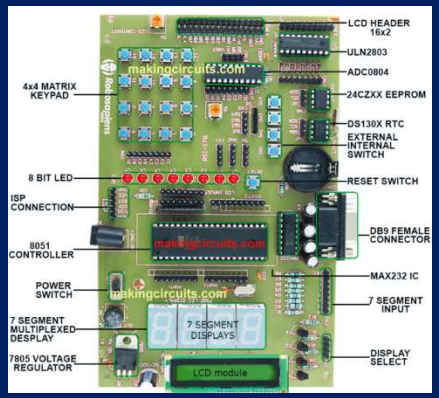
- Make Getting Started With Arduino – Use this for self study and back reference
- Arduino Cookbook 2nd Edition – Use this for reference and advanced study
- Make Electronics 2nd Edition – Use this for self study and back reference

RETURN FLASH DRIVES AT END OF EACH LESSON

- I'll load new content each week, at least that week's lesson but also sketches

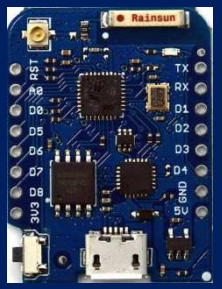
Microcontroller Evolution

- Originally were microprocessors with separate memory, I/O
- As chip transistor density increased, it become possible to integrate all components onto a single System on a Chip, or SoC
- SoCs evolved from embedded systems to user/hobbyist-accessible
- An SoC microcontroller made in a user-accessible format is called a microcontroller board or just “board” for short
- All boards offer General Purpose I/O (GPIO) pins which do a variety of things
- First and most famous user accessible microcontroller is the Arduino Uno
- We now have hundreds of microcontroller boards to choose from
- See https://makingcircuits.com/blog/types-of-microcontroller-boards-and-their-applications/ for details and examples



A vintage embedded microcontroller

Arduino Uno



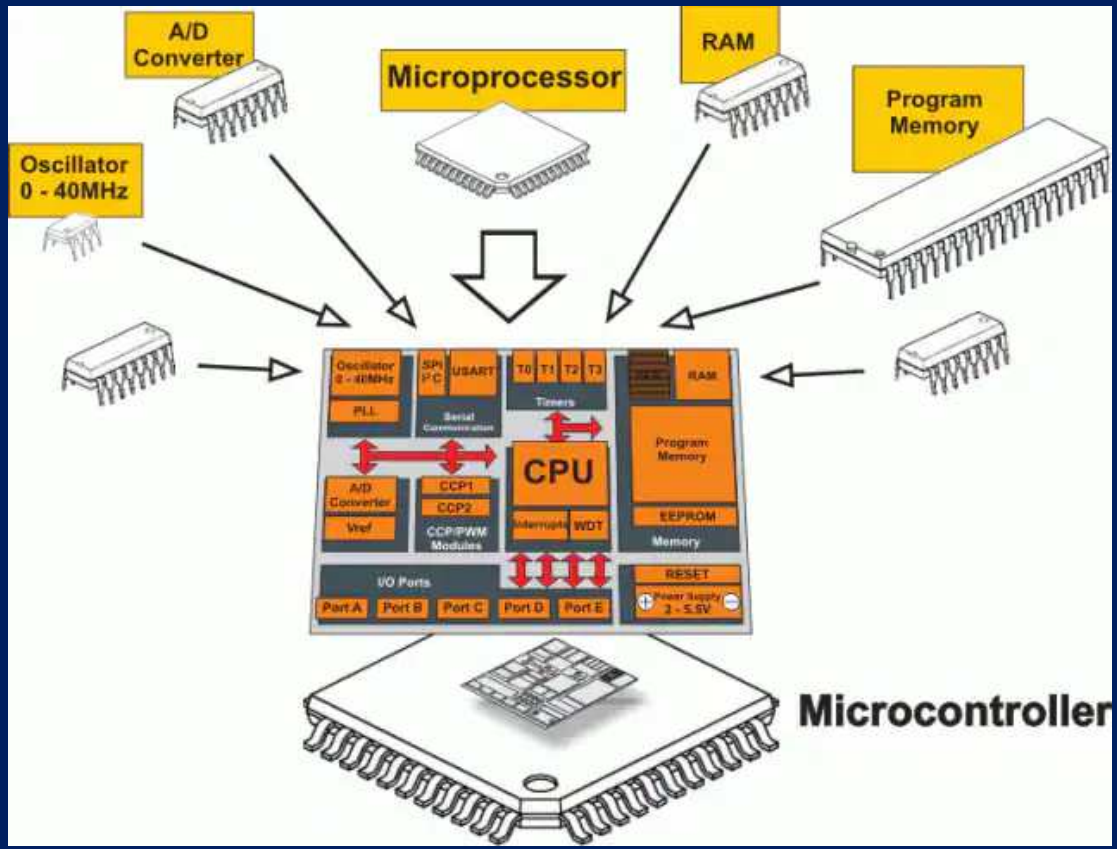
LOLIN WeMos D1 Mini

The Make/Digi-Key Boards Guide

Generic Microcontroller Architecture

All microcontrollers use a similar architecture with similar parts. There are unique features for each processor and each board, yet they all have basic concepts in common

- Central Processing Unit (CPU)
- Running program memory (Dynamic RAM)
- Stored program memory (Flash)
- I/O ports and signal converters
- Clock source and timers
- Power regulator



Microcontroller Differences

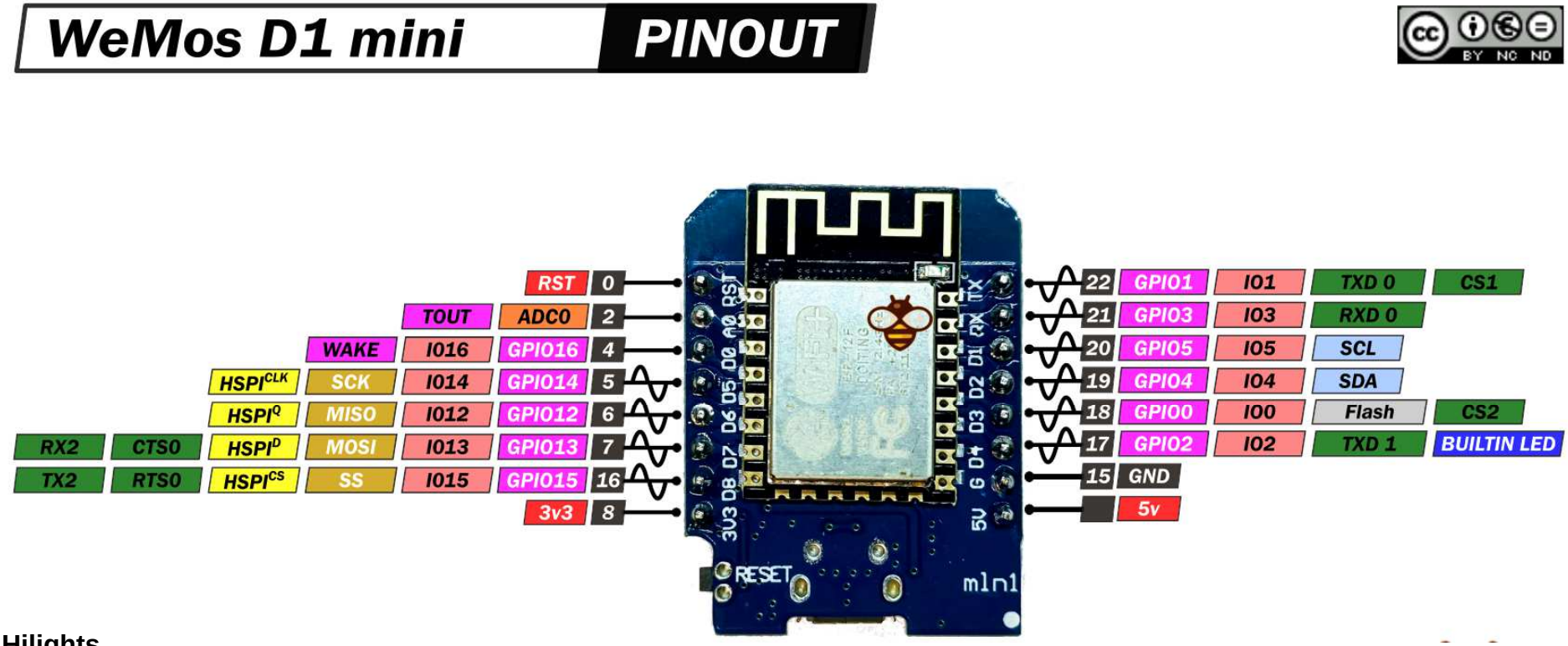
In addition to unique features to numerous to cover, there are some things that will be different between microprocessors. These are cost/benefit tradeoffs.

- Size of dynamic program memory (DRAM)
- Size of offline program storage (Flash memory)
- Speed of the processor (base clock speed, e.g., 120Mhz)
- Addressing and data sizes the processor can handle, e.g., 8-bit vs 32-bit
- Number of General Purpose I/O connections
- Number and resolution of analog ports (ADC)
- Number and type of serial connections available
- Number of ports capable of pulse width modulation (PWM) output
- How the microcontroller is flashed with new software
- Integrated WiFi and/or Bluetooth
- Integrated USB connectivity (some boards don’t have this!)

More subtle differences

- The number and type of internal timers available
- Which ports can act as interrupt inputs
- How the internal port configuration registers work

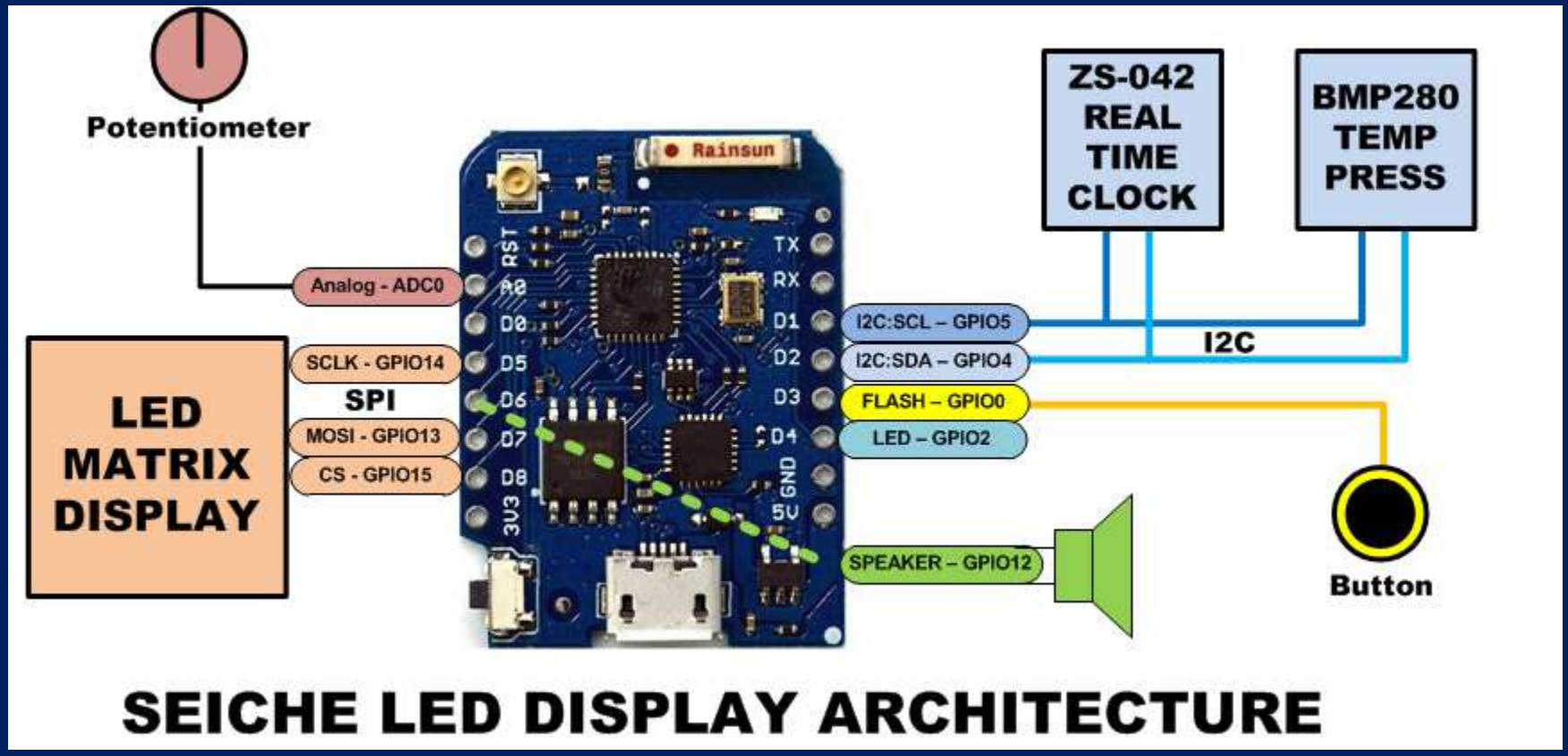
Our Microcontroller: The WeMos D1 Mini



Hilights

- **11 digital IO: all are interrupt and pwm capable (except D0/GPIO16)**
- **1 analog input (3.2V max input): A0/ADC0**
- **Micro USB or Type-C USB Port (clones usually have micro USB)**
- **Two SPI interfaces (one is used for on-board flash memory), one I2C interface, two serial ports**
- **Built-in WiFi (client or standalone access point modes) and Bluetooth**
- **Compatible with MicroPython, Arduino, NodeMCU**
- **Uses the CH340 USB-to-serial driver (installation usually needed on Windows)**
- **Extremely low cost (approx $3.00 US on Amazon; one of the two least expensive components in your kits)**

SEICHE LED Display Architecture



- Red MAX7219 8x32 LED matrix display (SPI)
- ZS-042 real-time clock module (I2C)
- BMP280 Temperature and Pressure Sensor (I2C)
- Piezoelectric speaker (PWM)
- 10KΩ Potentiometer (Analog-to-Digital Converter)
- Button (Pullup and interrupt)

Formal End of Lesson 1

HOMEWORK!

- **Getting Started With Arduino – Chapter 3**
- **Arduino Cookbook – Chapter 1**

In next week's exciting episode

- USB drive distribution and contents
- Installing CH340 drivers
- Installing the Arduino IDE