ARDUINO AND OTHER EMBEDDED SYSTEMS

Peter Vaughan

NSCC Online Learning

Table of Contents

Overview

The Arduino

Over the last few weeks, I have been developing a class for programming. My main objective of the class is to program an Arduino in the C++/C language. However, this class could be modified to other embedded systems based on what is available like an ESP32. The Arduino boards are usually available in many markets and versatile to understand basic programming. The IDE is open-source with many tutorials online for students to explore and learn.

The objective is to teach students basic transferable skills for any of their embedded systems projects. There is a lot of potential in this type of technology, which is growing regularly. I do not want to limit students or life-long learners on their potential journey.

My other documents for this class include a mock-up syllabus, a mock-up for weekly lessons and potential assignments as students progress through the course. There is potential for more advanced level programming class for the Arduino. These more advanced classes can still be affordable for supplies depending on how these classes are designed

Why Do I Compare the Arduino to the Raspberry Pi?

The reason why I did a document to include other embedded systems, especially the Raspberry Pi, is that people have options to developed their embedded systems skills. The Raspberry Pi is another popular embedded tool for hobbyist and students beyond the Arduino. Each embedded system can provide people with different tools and outcomes for more advanced projects. For example, one use of the Raspberry Pi is to use it as a server such Samba File Server or as part of Plex Media Server. However, the Arduino cannot do this due to lack of memory and no operating system.

The Arduino and Raspberry Pi are cheap learning tools for embedded systems and programming. Once a student buys the hardware, there are many open-source development tools and websites dedicated to helping anyone to learn. However, there is a lot of information there. This is why a class is important to teach the basics and to help students find their focus within programming and embedded systems.

Arduino VS Raspberry Pi

Summary

Arduino is ideal for beginners who want to learn electronics and embedded systems through hands-on projects like controlling sensors and motors. It uses a simple C++-based language and is great for real-time applications but lacks advanced computing capabilities. Raspberry Pi, on the other hand, is a full computer that supports multiple programming languages and is excellent for learning software development, Linux, and building complex applications. Also, Raspberry Pi can connect to hardware but is not as precise for real-time control. For comprehensive learning, combining both platforms allows users to leverage Arduino for hardware control and Raspberry Pi for processing and networking.

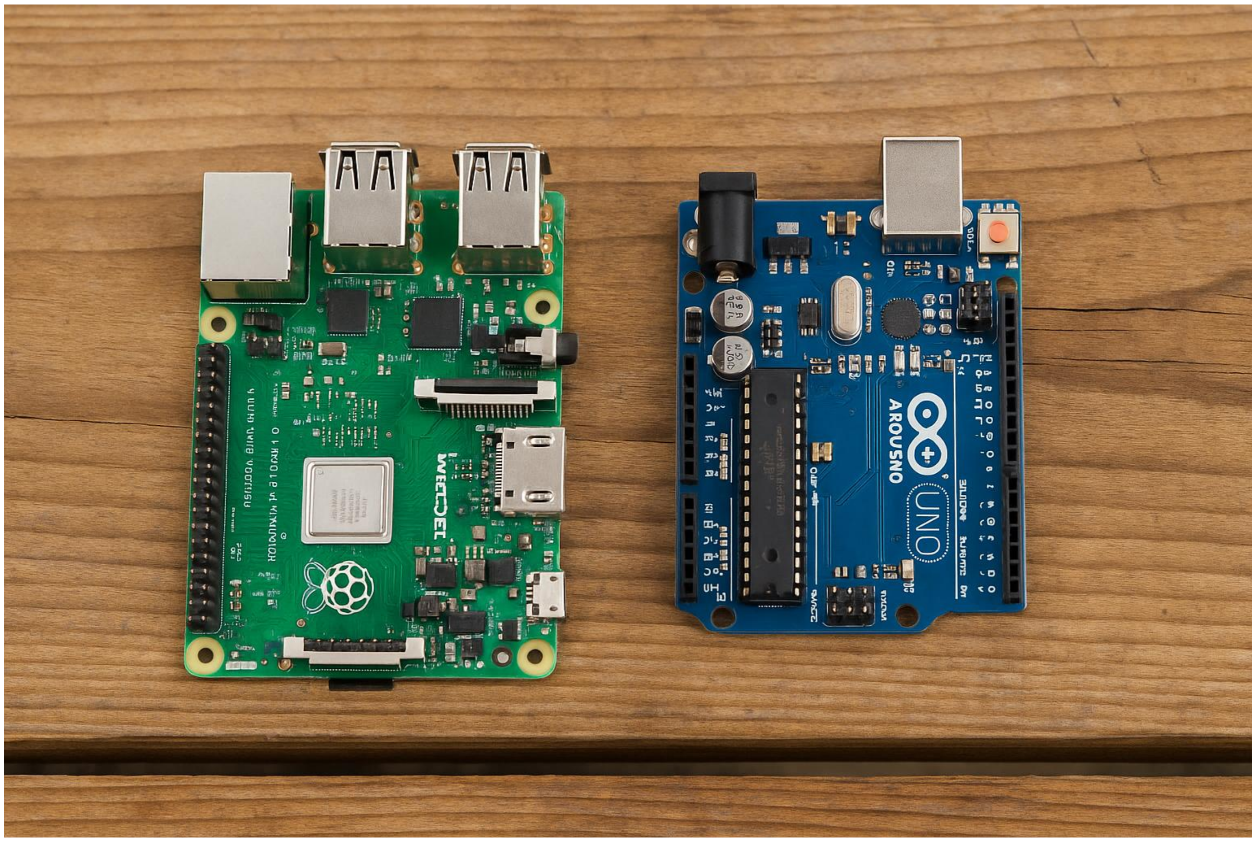

Figure 1 – Source: Vaughan (2025) - Raspberry Pi (left) and Arduino Uno (right)

Arduino

Arduinos are an open-source microcontroller platform used to build digital devices and interactive projects by reading inputs and controlling outputs. It is popular for beginners and hobbyists because it is easy to program and supports a wide range of sensors and components. The Arduino Uno is a very popular version of the Arduino. The images below show some of the Arduino boards.

Best for: Learning electronics, embedded systems, and basic programming.

Pros:

- **Great for hardware projects**: Directly controls sensors, motors, and lights.
- **Simple language (based on C++)**: Good for beginners in programming logic.
- **Real-time control**: Excellent for projects where precise timing is important (e.g., robots, weather stations).
- **Low cost**: Especially for clones or smaller boards like Arduino Nano.

Cons:

- **Limited computing power**: No OS, no multitasking.
- **Not good for advanced software projects**: No native support for Python, networking, or graphics.

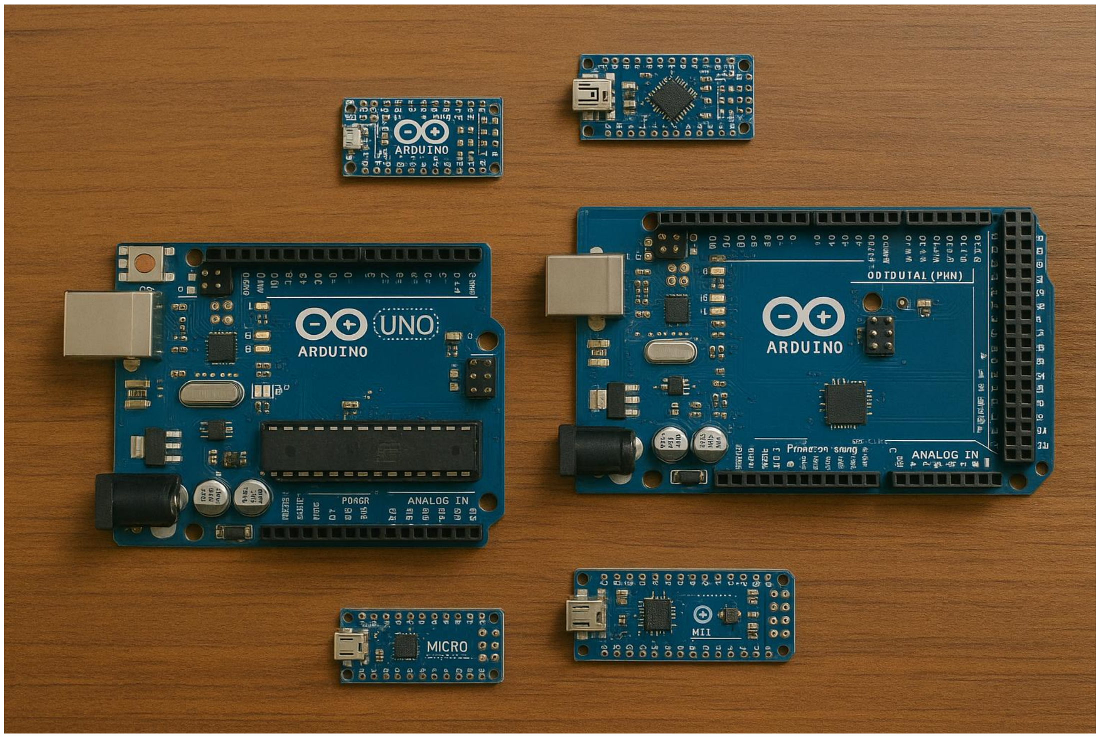

Figure 2 – Source: Vaughan (2025) - Various Arduinos

Some of the Many Arduino Components

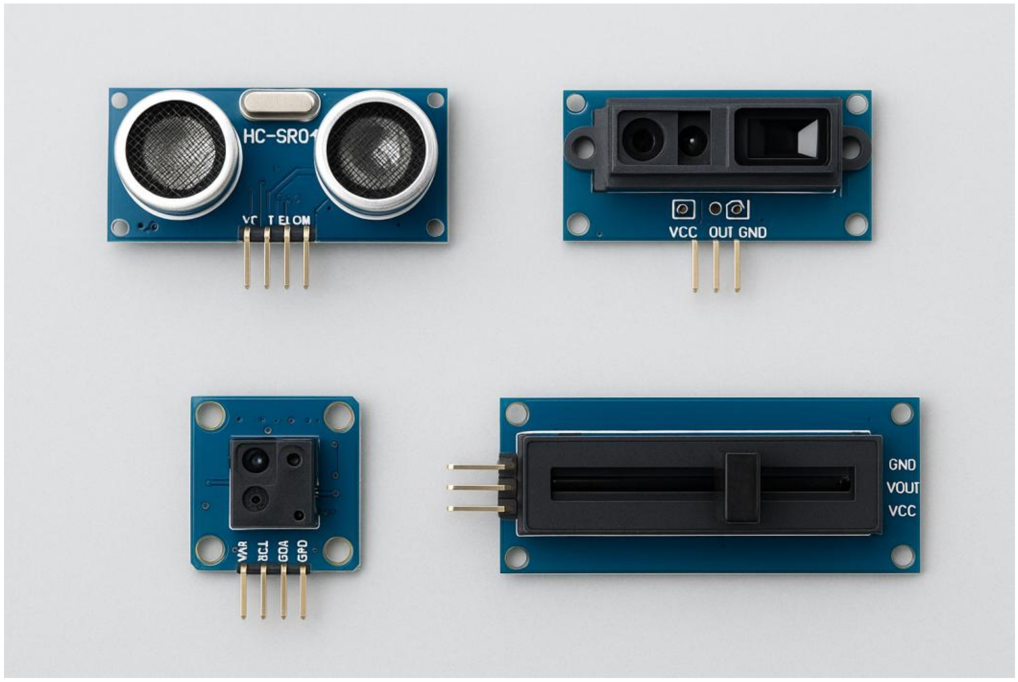

Figure 3

Source: Vaughan (2025)

Arduino Components

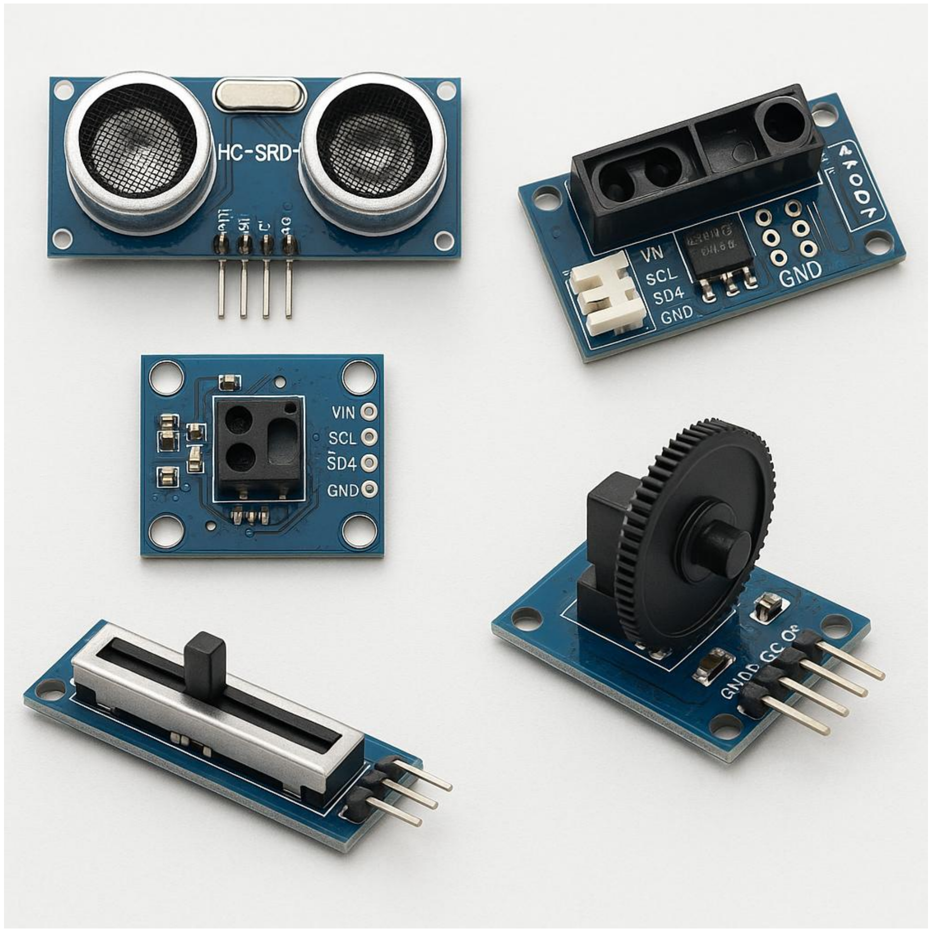

Figure 4

Source: Vaughan (2025)

Arduino Components

Raspberry Pi

A Raspberry Pi is a small, affordable computer that runs a full operating system, typically Linux. Also it is used for learning programming, building projects, and prototyping. It supports various programming languages. Also, it can handle tasks such as web browsing, coding, and connecting to electronics.

Best for: Learning general programming, computer science concepts, and working with Linux.

Pros:

- **Runs a full OS (usually Raspberry Pi OS)**: Like a mini computer.
- **Supports many languages**: Python, JavaScript, C++, etc.
- **Great for software development**: Web servers, games, AI, networking.
- **Can interface with other electronics** (though not as low-level as Arduino).

Cons:

- **More complex setup**: Needs keyboard, monitor, or SSH access.
- **Less ideal for precise hardware timing**: Not real-time like Arduino.

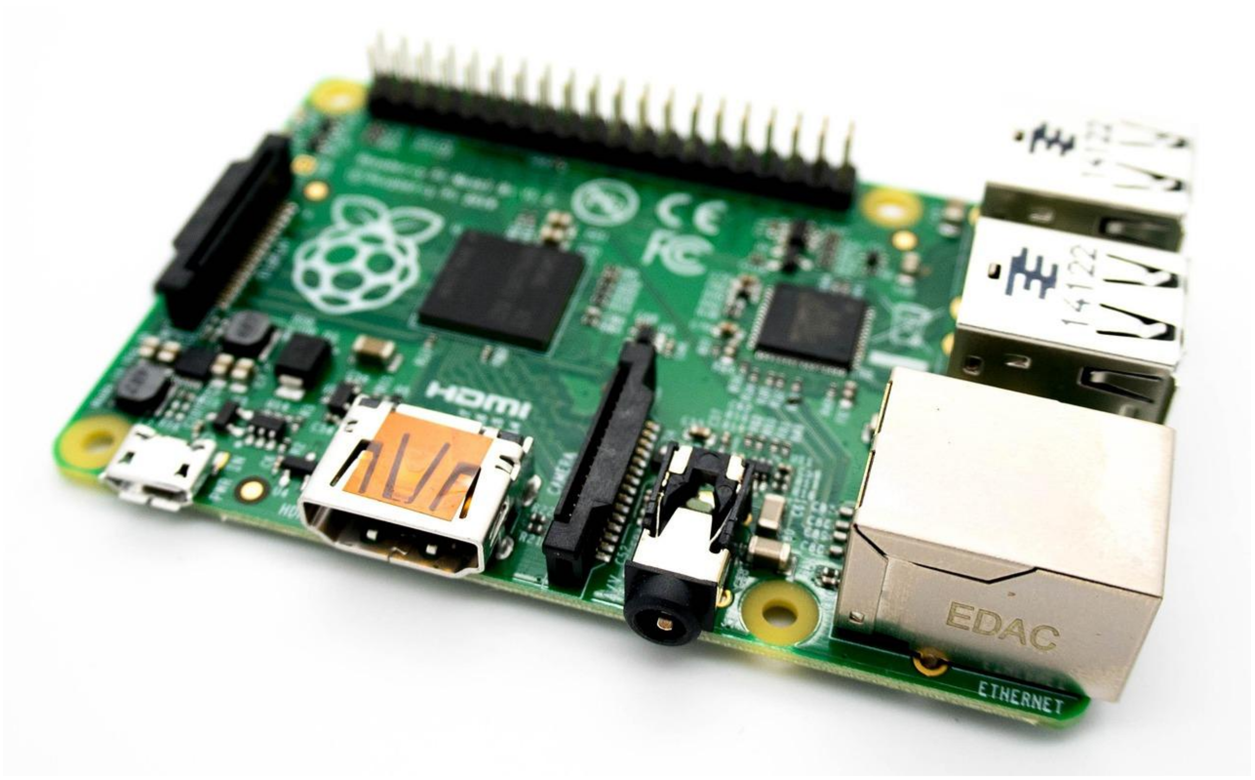

Figure 5 – Source: kevinpartner (2014) **-** Raspberry Pi

Comparison

Arduino and Raspberry Pi are both popular tools for learning electronics and coding, but they can serve different purposes. Arduino is a microcontroller ideal for controlling hardware like sensors and motors in real-time, using simple C/C++ code. Raspberry Pi is a full-fledged mini-computer capable of running a Linux-based OS, supporting complex software, and multitasking in languages like Python. Arduino is better suited for low-level, real-time control, while Raspberry Pi excels at high-level programming, networking, and multimedia tasks. Despite their differences, both can be used together in advanced projects where Arduino handles hardware control and Raspberry Pi manages processing and communication.

Goal	Recommended Device
Learning basic electronics & hardware control	Arduino
Learning software development, Python, Linux	Raspberry Pi
Building interactive IoT projects*	Use **both together** (Arduino as a sensor hub, Pi as the brain)

*IoT stands for Internet of Things. It refers to a network of physical devices—like sensors, appliances, or microcontrollers—that are connected to the internet to collect, share, and act on data.

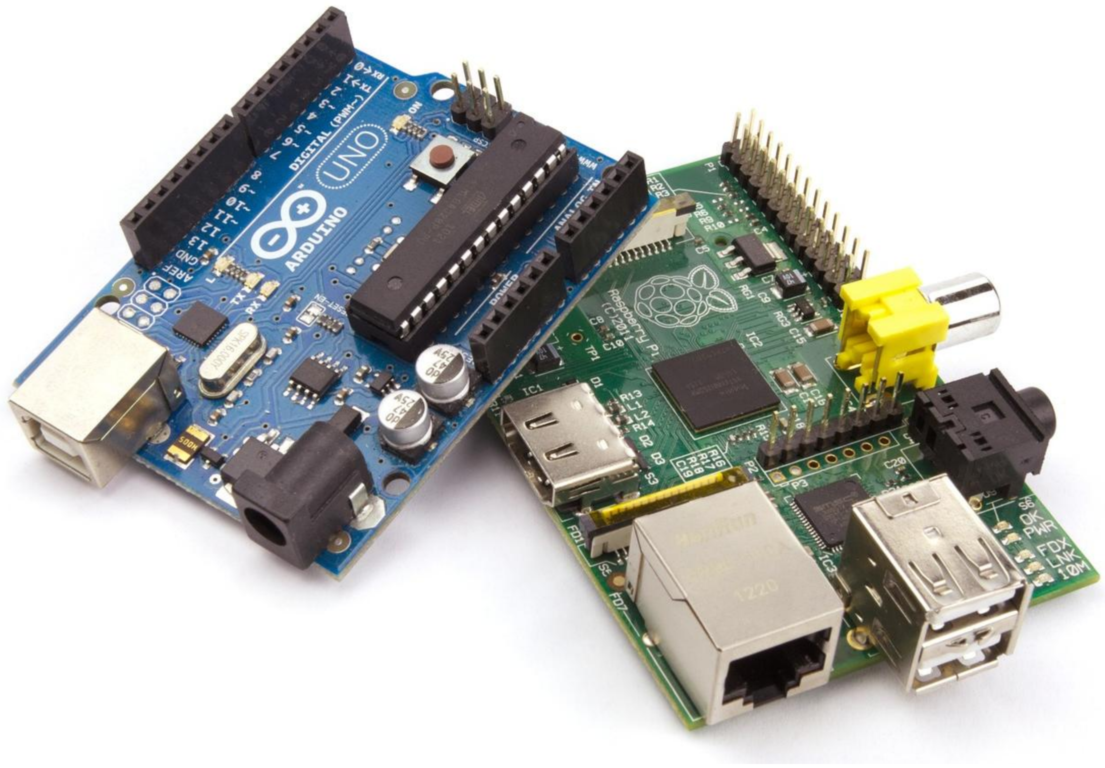

Figure 6 – Source: christoph1703 (2020) - Arduino (left) and Raspberry Pi (right)

Project Ideas

By combining the Arduino and a Raspberry Pi, this opens up powerful project possibilities by merging real-time sensor control with advanced computing and networking. A few popular ideas include smart home systems, security cameras, and automated gardens. One standout project is using an Arduino and Raspberry Pi together to monitor and control a 3D printer. The Raspberry Pi will be running OctoPrint and the Arduino managing motors and sensors. This setup provides real-time feedback, remote control, and a seamless printing experience—definitely my personal favorite!

1. Smart Home Automation System

Arduino controls sensors (temperature, motion, light) and relays for devices. Raspberry Pi hosts a dashboard for remote monitoring and control. Integrate with Home Assistant or MQTT for full IoT functionality.

2. Smart Security Camera with Sensor Triggers

Arduino triggers alerts from motion or door sensors. Raspberry Pi streams and records video when triggered. Add OpenCV for face detection or license plate recognition.

3. Automated Garden

Arduino monitors soil moisture and activates a water pump. Raspberry Pi logs data, runs a dashboard, and sends alerts. Schedule watering and integrate with weather forecasts. I have a similar concept with a sub-pump in my basement but there are currently not many options in the market.

4. Remote-Controlled Robot with Live Feed

Arduino drives motors and processes basic sensor input. Raspberry Pi provides camera vision and remote control via Wi-Fi. Use ROS or Flask for advanced robot features. Personally, I think ROS is fun to use!

5. Voice-Controlled Appliance Controller

Arduino controls relays to switch appliances on/off. Raspberry Pi processes voice commands using Google Assistant. Send voice-triggered commands to the Arduino over serial or MQTT.

6. Custom Game Console Controller

Arduino emulates a USB gamepad with physical buttons and joysticks. Raspberry Pi runs RetroPie or other emulators. Build a handheld or arcade-style case around it to give this project a finished look.

7. Industrial Sensor Dashboard

Arduino reads from vibration, pressure, or current sensors. Raspberry Pi stores data and visualizes it with Grafana. Receive real-time alerts via email or messaging apps.

8. Scientific Data Logger

Arduino captures high-frequency data from lab sensors. Raspberry Pi logs and analyzes data locally or uploads to the cloud. Schedule reports or visualizations with Python scripts.

9. IoT Door Access System

Arduino reads RFID/NFC tags and actuates a lock. Raspberry Pi manages user access via web interface. Combine with facial recognition for multi-factor authentication.

10. CNC/3D Printer Monitor (Figure 7)

Arduino controls various motors and reads endstops or sensors. Raspberry Pi runs OctoPrint for live monitoring and remote control. Add a camera or webcam for timelapse and print progress monitoring. This is my personal favourite.

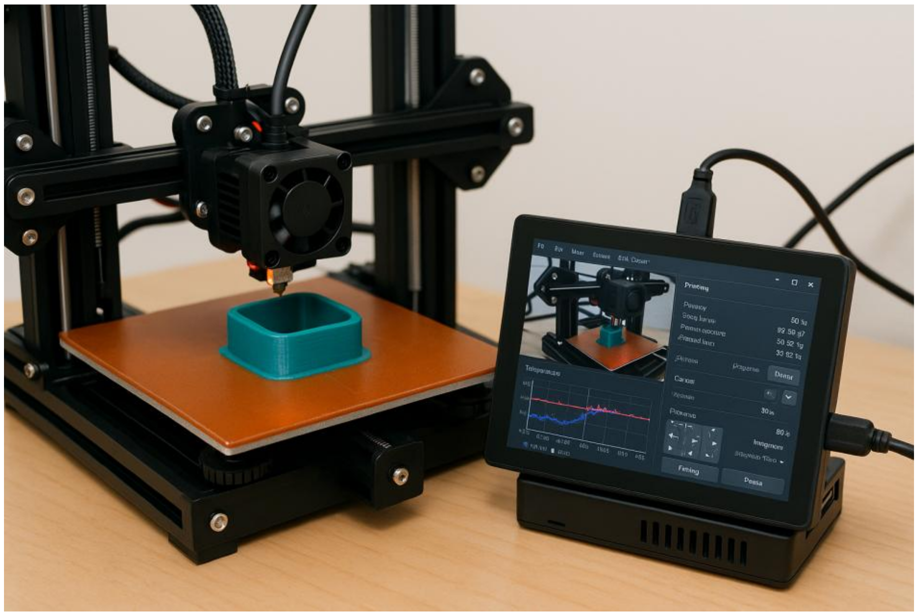

Figure 7

Source: Vaughan (2025)

A 3D Printer with a Raspberry Pi and a mini-monitor. The OctoPrint program is installed and running.

Embedded Systems for Beginners

Arduino and Raspberry Pi are popular platforms, but they are not the only choices for beginners exploring embedded systems. Alternatives like the **ESP32** offer built-in Wi-Fi and Bluetooth, making them ideal for wireless IoT projects. The **BBC micro:bit** is perfect for young learners, with simple block coding and built-in sensors. **Adafruit's Circuit Playground** and **Feather boards** are beginner-friendly and support CircuitPython, which is easier to learn than C++. For more advanced beginners, **STM32 boards** provide powerful microcontrollers with professional-grade features. Exploring these alternatives helps learners choose the right tool for their specific project goals and interests. Teachers should always encourage their students and let them explore some while learning.

1. Arduino

Arduino is a simple microcontroller platform great for learning electronics, sensors, and basic coding using C/C++. It's ideal for real-time control in projects like LED displays, robots, and environmental monitoring.

- **Best for:** Beginners, real-time control, hardware interfacing.
- **Why:** Easy to use, lots of sensors/modules, huge community.
- **Use Cases:** LED projects, sensors, robotics, automation.

2. Raspberry Pi

Raspberry Pi is a small computer that runs Linux and supports high-level programming languages like Python. It's perfect for projects involving networking, multimedia, or combining software and hardware.

- **Best for:** More complex software-based projects with networking or GUI.
- **Why:** Full Linux OS, can run Python, C++, etc.
- **Use Cases:** Web servers, media centers, smart home hubs, AI projects.

3. ESP32 / ESP8266

The ESP32 is a powerful, low-cost microcontroller with built-in Wi-Fi and Bluetooth. It's perfect for IoT projects and can be programmed with Arduino IDE or MicroPython.

- **Best for:** Wireless (Wi-Fi/Bluetooth) IoT projects.
- **Why:** More powerful than Arduino with built-in Wi-Fi, cheap.
- **Use Cases:** Smart home, remote sensors, weather stations, web APIs.

4. STM32

STM32 boards are powerful microcontrollers based on ARM Cortex-M processors, widely used in industrial and embedded applications. They offer advanced features like real-time processing, low power consumption, and extensive peripheral support, making them ideal for experienced makers and ambitious hobby projects. This is great for beginners if they are working with an experienced individual.

- **Best for:** Advanced hobbyists and low-power or performance-sensitive tasks.
- **Why:** ARM Cortex-M cores, real-time capabilities, widely used in industry.
- **Use Cases:** Drones, motor control, wearables, serious embedded learning.

5. BeagleBone Black

BeagleBone Black is a Linux-capable development board like the Raspberry Pi but with more I/O and real-time features. It's a solid choice for serious robotics or industrial applications.

- **Best for:** High-performance embedded Linux projects.
- **Why:** More I/O than Raspberry Pi, designed for developers.
- **Use Cases:** Industrial projects, robotics, real-time processing.

6. BBC Micro:bit

The BBC micro:bit is a beginner-friendly microcontroller designed for education, featuring built-in LEDs, buttons, sensors, and Bluetooth. It supports block-based coding, Python, and JavaScript, making it ideal for students and first-time coders.

- **Best for:** Kids and absolute beginners.
- **Why:** Block coding, built-in sensors, and easy-to-use environment.
- **Use Cases:** Educational projects, wearable tech, basic games.

7. Adafruit Feather & Circuit Playground

The **Adafruit Feather** is a compact, modular microcontroller board designed for portability, with versions that include Wi-Fi, Bluetooth, and other features for diverse project needs. It's great for battery-powered and wireless applications, and supports Arduino and CircuitPython programming.

The **Adafruit Circuit Playground Express** is an all-in-one beginner board with built-in LEDs, sensors, and touch inputs. It's ideal for learning through hands-on experiments and supports block coding, CircuitPython, and Arduino.

- **Best for:** Portable or beginner projects.
- **Why:** Built-in sensors, designed with makers in mind, supports CircuitPython.
- **Use Cases:** Wearables, quick prototyping, portable devices.

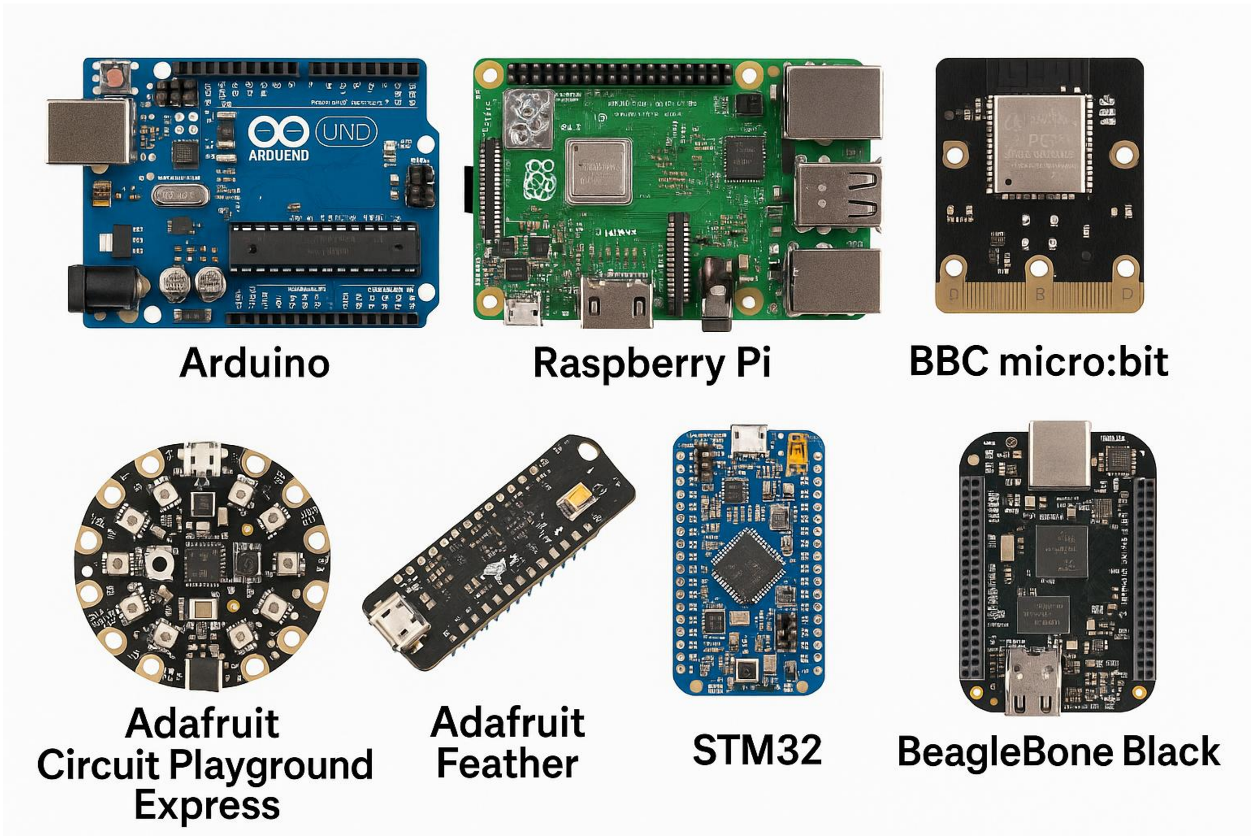

Figure 8 – Source: Vaughan (2025) – Beginner Embedded Systems

Beginner Embedded Systems Summary

There are many embedded systems that can be taught to anyone. Also, all of these embedded systems can work together by assigning them roles based on their strengths. For example, a Raspberry Pi can handle data processing, internet communication, and user interfaces, while an Arduino or STM32 manages real-time sensor or motor control. ESP32 boards can connect wirelessly to transmit data between devices or to the cloud. Systems like the BBC micro:bit or Circuit Playground can serve as input devices or learning tools within a larger connected network. Using standard protocols like I2C, UART, or MQTT, developers can integrate these boards into powerful, collaborative systems for smart homes, robotics, or educational ecosystems. Students are not limited on how they can learn about embedded systems and their interactions with other systems. There are a lot of industries that use embedded systems such as automotive, farming automation, education, medical equipment, retail (such as Point-of-Sale Systems), and security systems.

Additional Info

To learn more about the Arduino and Raspberry Pi, visit the GitHub repository, LinkedIn profile, or contact us via email. We are always looking for constructive feedback to add to the embedded systems for future learning.

Learning embedded systems might seem challenging at first, but every small step brings anyone closer to building something truly amazing. Do not worry if you understand everything right away—hands-on practice will teach you more than theory ever could. Mistakes are part of the process and often lead to the best learning moments. You are building skills that power real-world technology, from smart homes to robot. Stay curious, keep experimenting, and remember: even the most advanced engineers once started with blinking an LED.

I encourage you to learn the fundamentals of programming before diving into embedded systems. When I was learning, I learned about hardware before software. This made it difficult for me and my fellow classmates to fully develop an embedded system. This is why understanding basic concepts like variables, loops, and functions will make your exploration smoother and more enjoyable. To support your journey, I have included additional documentation and resources in my GitHub repository — feel free to explore and learn at your own pace! Developing this class has reminded me of my journey and my desire to help others in their journey.

GitHub Repo: https://github.com/Vaughan-Peter/ArduinoLearning

LinkedIn Profile: https://www.linkedin.com/in/peter-vaughan-997478239/

Contact E-mail: otherhalifaxprojects@gmail.com

References

christoph1703. (2020, January 08). *N/A*. Retrieved from Pixabay: https://pixabay.com/photos/arduino-raspberry-pi-electronic-4753005/

kevinpartner. (2014, December 18). Retrieved from Pixabay: https://pixabay.com/photos/raspberry-pi-computer-electronics-572481/

Vaughan, P. (2025, 05 28). Retrieved from Chat GPT: https://chatgpt.com/