Microcontrollers

- Arduino
- ESP32/ESP8266
- Raspberrys

The ideal microcontroller depends on whether your project requires high processing power, low-cost wireless connectivity, or high hardware durability.

A direct comparison of classic Arduino (e.g., Uno R3), ESP32/ESP8266, and the Raspberry Pi RP2040 (often called the Raspberry Pi Pico / Nano form factors).

Feature	Classic Arduino (Uno R3)	ESP8266 / ESP32	Raspberry Pi RP2040 (Pico)
Processor Type	8-bit AVR (Atmega328P)	32-bit Tensilica Single/Dual-Core	32-bit ARM Cortex-M0+ Dual-Core
Clock Speed	16 MHz	80 MHz (ESP8266) / 240 MHz (ESP32)	133 MHz (Overclockable)
SRAM (Memory)	2 KB	160 KB (ESP8266) / 520 KB (ESP32)	264 KB
Flash Storage	32 KB	1 MB to 16 MB	2 MB to 16 MB (External)
Operating Voltage	5V (Highly durable)	3.3V (Sensitive to 5V)	3.3V (Sensitive to 5V)
Native Wireless	None	Built-in Wi-Fi & Bluetooth (ESP32)	None (Requires Pico W variant)
Special Feature	Massive shield ecosystem	Hardware cryptographic acceleration	PIO (Programmable I/O) State Machines
Best Used For	Beginners & 5V legacy sensors	Smart Home & IoT cloud projects	High-speed processing & custom protocols
Primary Language	Arduino C++	Arduino C++, ESP-IDF (C)	MicroPython / CircuitPython, C/C++
Flashing Method	Serial (UART) via Bootloader	Serial (UART) with Auto-Reset	USB Mass Storage (UF2 Drag & Drop)
Recovery Mode	Hardware programmer (ISP)	Boot Pin (GPIO 0) to Ground	BOOTSEL Button on power-up
Interpreter Support	No (Compiled binary only)	Limited (Basic MicroPython)	Native / Built-in (Excellent for Python)
Execution Mode	Single-threaded (Sequential)	FreeRTOS (Multi-threaded)	Hardware Dual-Core (Symmetric)
OTA Updates	No (Requires special shields)	Native Wi-Fi OTA Flashing	No (Requires custom implementation)