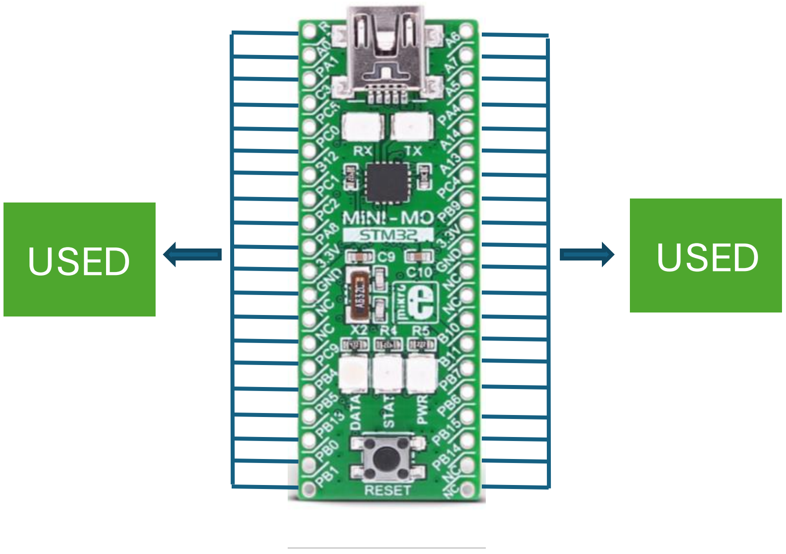
What to do when you run out of MCU pins



AZMAN BAKHTIAR
Embedded Developer

Complex embedded projects often need more I/Os than one MCU provides

Fully utilized GPIOs = can’t connect more sensors, buttons, displays

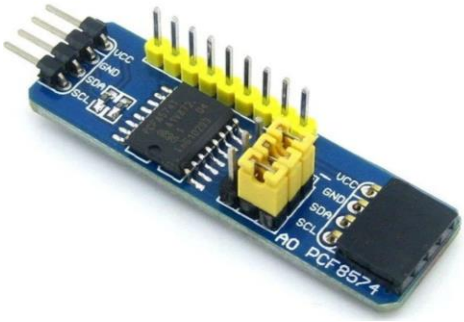
Common in projects with multiple modules (Wi-Fi, BLE, SD, TFT, etc.)

Solutions must balance cost, complexity, and performance

AZMAN BAKHTIAR
Embedded Developer

Hardware Methods to Expand I/O

- **GPIO Expanders** (e.g., PCF8574, MCP23S17): Add 8–16 GPIOs via I2C/SPI
- **Multiplexers/Demux** (e.g., 74HC4067): Scan many inputs using few pins
- **Shift Registers** (e.g., 74HC595): Control many outputs with 3 wires
- **Matrix Wiring:** 4x4 keypad = 16 buttons using 8 pins



AZMAN BAKHTIAR
Embedded Developer

Architectural Solutions

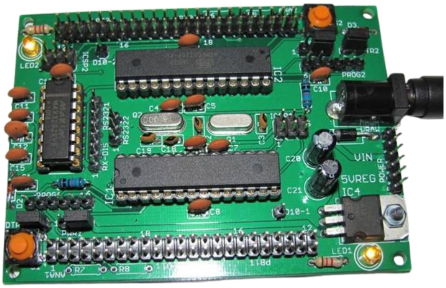
Use a Secondary MCU: Offload UI or communication tasks

Split logic: One MCU handles sensors, the other handles outputs

Use SPI/I2C/UART to connect MCUs — define a clear protocol

Modular Design: Move I/O-hungry features to a daughterboard

Adds flexibility, reduces overloading a single chip



AZMAN BAKHTIAR
Embedded Developer

Smart Design Practices

- Choose higher pin-count MCUs (STM32F4, H7 series in LQFP-100/144)
- Leave extra pins routed to test points during design
- Use multifunction pins (e.g., analog/digital/shared SPI lines)
- Disable unused modules/peripherals to free GPIOs
- Plan expansion headers (I^2C/SPI) early for future add-ons

AZMAN BAKHTIAR
Embedded Developer

AZMAN BAKHTIAR
Embedded Developer

Reach Out for Free IoT & Embedded Consultation

Reshare if You Found it Helpful