**Develop Firmware for an IO board**

**Objective**

Our main goal is developing firmware for a I/O controller which is connected to another device such PC or other microcontroller via UART communication protocol to handle some configurable digital and analog I/O. The firmware should provide both runtime configurability and efficient data handling.

**Key Requirements**

1. **Digital I/O (DIO)**
   * **Quantity**: 8 configurable digital I/O pins
   * **Modes**:
     + Configurable as input or output at runtime.
     + **Output Options**:
       - **Regular Mode**: Standard digital output.
       - **Blink Mode**: Output toggles with adjustable frequency ranging from 1 Hz to 1 kHz.
2. **Analog Output (AO)**
   * **Quantity**: 2 analog output channels
   * **Specifications**:
     + Resolution: 10-bit (0–1023).
     + Output Voltage Range: 0–5 V.
3. **Analog Input (AI)**
   * **Quantity**: 2 analog input channels
   * **Specifications**:
     + Resolution: 10-bit (0–1023).
     + Input Voltage Range: 0–5 V.
4. **UART Communication**
   * **Primary UART**:
     + Serves as a communication interface with the main MCU for reading/writing I/O states and configurations.
     + Must be fully configurable.
   * **Secondary UART**:
     + Reserved for debugging purposes, allowing insights into firmware operation and status.
5. **Error Handling**
   * Implement robust error detection and handling mechanisms to ensure stable and reliable firmware operation.
   * Requirements:
     + **I/O Error Detection**: Identify and respond to errors in DIO, AO, and AI (e.g., configuration mismatches, out-of-range values).
     + **UART Communication Errors**: Detect UART communication failures, framing, and data corruption, with mechanisms for automatic retries or fallback modes.
     + **Configuration Errors**: Verify non-volatile configurations at startup and handle invalid settings gracefully by reverting to default values or notifying via the debug UART.
     + **Status Reporting**: Use the debug UART to report any detected errors, system status updates, or recovery actions.

**What I Have Chosen According to the Requirements**

Since the requirements file does not specify a particular microcontroller or development environment, I have selected the **ESP32-WROOM-32** microcontroller, as it is readily available to me. For firmware development, I am using the **ESP-IDF** framework in **VS Code**, with the current version being **ESP-IDF v5.2.3**.

I chose the **ESP32 Dev Module** for this application because it is **low-cost**, suitable for **low-power applications**, and supports an **input voltage range of 0V to 3.3V**. It features **18 ADC channels** distributed across two ADC units:

* **ADC1:** 8 channels (**GPIOs 32-39**)
* **ADC2:** 10 channels (**GPIOs 0, 2, 4, 12-15, 25-27**)

The ADC supports a **12-bit resolution**, while the module also includes **two DAC channels**:

* **DAC1 (GPIO25)**
* **DAC2 (GPIO26)**

Both DAC channels support an **8-bit resolution** with an **output voltage range of 0V to 3.3V**. As non-volatile memory, I am using the ESP32's internal flash memory to store necessary data.

Since I am using the ESP32 as the main microcontroller (due to availability), there are some **limitations** compared to our actual requirements, particularly in terms of **DIO and AIO voltage range** and **bit resolutions**. However, as this is an **example firmware development test**, I am proceeding with this setup while considering these limitations. Alternatively, I could select a different microcontroller, such as an **STM32**, to better meet the actual requirements.

**Resources**

To develop the firmware, I collected the necessary configurations instruction such as digital and analog gpio configuration, UART configuration, error handling and others for ESP-IDF framework from the Espressif documentation and GitHub resources. I have attached here the documentations link below.

* 1. ESP-IDF documentation: <https://docs.espressif.com/projects/esp-idf/en/v5.2.3/esp32>
  2. GitHub: <https://github.com/espressif/esp-idf>