Error Handling

Error Handling Using the ESP-IDF

ESP-IDF Error Handling Brief Overview

- Overview of Error Handling
- Error Codes
- Converting Error Codes to Error Messages
- ESP_ERROR_CHECK Macro
- Error Handling Patterns

Error Handling Overview

- Espressif Documentation:

  https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/error-handling.html

- Recoverable Errors
  - Errors indicated by functions through return values (error codes).
- Unrecoverable (Fatal) Errors
  - Failed assertions (using assert macro and equivalent methods, see Assertions) and abort() calls.
  - CPU exceptions: access to protected regions of memory, illegal instruction, etc.
  - System level checks: watchdog timeout, cache access error, stack overflow, stack smashing, heap corruption, etc.

Error Codes

- Error Codes in Brief
  - Most ESP-IDF-specific functions use esp_err_t type to return error codes.
    - esp_err_t is a signed integer type.
    - Success (no error) is indicated with ESP_OK code, which is defined as zero.
  - Common error codes for generic failures (out of memory, timeout, invalid argument, etc.) are defined in esp_err.h file.
  - Various ESP-IDF header files define possible error codes using preprocessor defines. Usually these defines start with ESP_ERR_ prefix.

Converting Error Codes to Error Messages

- Conversion to Strings for Debug Logging
  - For each error code defined in ESP-IDF components, esp_err_t value can be converted to an error code name using esp_err_to_name() or esp_err_to_name_r() functions.

ESP_ERROR_CHECK Macro

- Similar to Assert…
  - ESP_ERROR_CHECK macro checks esp_err_t value rather than a bool condition.
    - If the argument of ESP_ERROR_CHECK is not equal ESP_OK, then an error message is printed on the console, and abort() is called.

Error message will typically look like this:

```
ESP_ERROR_CHECK failed: esp_err_t 0x107 (ESP_ERR_TIMEOUT) at 0x400d1fdf

file: "/Users/user/esp/example/main/main.c" line 20
func: app_main
expression: sdmmc_card_init(host, &card)

Backtrace: 0x40086e7c:0x3ffb4ff0 0x40087328:0x3ffb5010 0x400d1fdf:0x3ffb5030 0x400d0816:0x3ffb5050
```

Error Handling Patterns

- Strategies for Handling Errors
  - Documentation: https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/error-handling.html#error-handling-patterns
  - Attempt to Recover: Retry the call, reinitialize the driver in use, reset an external peripheral which is not responding.
  - Propagate the Error to the Caller: return the error codes that applies to the situation.
  - Convert to an Unrecoverable Error: e.g., using ESP_ERROR_CHECK.
    - Do not simply write ESP_ERROR_CHECK everywhere if your intention is to write production-ready code. Determine a strategy that suits your application needs!

Next Lesson