Eclipse Plugin for ESP-IDF
• Introduction
• ESP8266 & ESP32
• Espressif Hardware & Software
• IoT Applications
• ESP-IDF
• ESP-IDF Eclipse Plugin
• Demo
• Future Plans
• Q&A
Who are we?

Espressif Systems

Fabless semiconductor company based in Shanghai, China, with offices in China, India, and Europe.
Designs connected MCUs, targeting IoT applications.
Widely known for designing low-cost and low entry barrier Wi-Fi solutions.
Espressif Hardware & Software

SoCs
- ESP32
- ESP32-S2 (soon)
- ESP8266, ESP8285

Modules
- ESP32 Modules
- ESP8266 Modules

Development Boards
- ESP32-DevKit-C
- ESP-WROVER-KIT
- ESP8266-DevKit-C

SDKs
Bundle of utilities and device-level APIs for ESP8266 and ESP32
• Processor
  • Xtensa® Dual-Core 32-bit LX6
  • Operating at 160 or 240 MHz
  • Ultra low power (ULP) co-processor

• Memory
  • 520 KiB SRAM
  • 4MB external flash

• Wireless connectivity
  • Wi-Fi: 802.11 b/g/n
  • Bluetooth: v4.2 BR/EDR and BLE
<table>
<thead>
<tr>
<th>Specifications</th>
<th>ESP8266</th>
<th>ESP32</th>
<th>ESP32-S2</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MCU</strong></td>
<td>Single core Xtensa CPU up to 160MHz clock</td>
<td>Dual core Xtensa CPU up to 240MHz clock</td>
<td>Single core Xtensa CPU up to 240MHz clock</td>
</tr>
<tr>
<td><strong>Internal Memory (IRAM+DRAM)</strong></td>
<td>160kB</td>
<td>520kB</td>
<td>320kB</td>
</tr>
<tr>
<td><strong>Connectivity</strong></td>
<td>Wi-Fi 802.11bgn (HT20)</td>
<td>Wi-Fi 802.11bgn (HT40)</td>
<td>Wi-Fi 802.11bgn (HT40)</td>
</tr>
<tr>
<td><strong>External Memory</strong></td>
<td>Up to 16MB SPI flash, (1MB executable XIP)</td>
<td>Up to 16MB SPI flash, (4MB XIP), Up to 4MB SPI RAM</td>
<td>Up to 1GB SPI flash, (7MB XIP, 4MB rodata), Up to 10MB SPI RAM</td>
</tr>
<tr>
<td><strong>Peripherals</strong></td>
<td>GPIO, SPI, UART, I2S, ADC</td>
<td>GPIO, SPI, UART, I2C, I2S, PWM, ADC, DAC, RMT, CAN, SD host/slave, low-power coprocessor</td>
<td>GPIO, SPI, UART, I2C, I2S, PWM, ADC, DAC, RMT, USB, low-power coprocessor</td>
</tr>
<tr>
<td><strong>Hardware Security Features</strong></td>
<td>SHA2, RSA, AES accelerators, Transparent flash decryption, Secure boot 1024 bit OTP</td>
<td>SHA2, RSA, AES, HMAC accelerators, Transparent flash/ RAM en/decryption (AES-XTS) RSA-PSS Secure boot 4096 bit OTP</td>
<td>SHA2, RSA, AES, HMAC accelerators, Transparent flash/ RAM en/decryption (AES-XTS) RSA-PSS Secure boot 4096 bit OTP</td>
</tr>
</tbody>
</table>
Applications

- Consumer Electronics
  - Appliances
  - Smart home
  - Toys
- Industrial devices
  - Bridges, gateways
  - Data collection
  - Embedded controllers
ESP IoT Development Framework (ESP-IDF) is the official development framework for the ESP32 series of chips.

Source code: [https://github.com/espressif/esp-idf/](https://github.com/espressif/esp-idf/) (Apache 2.0 license)

SoC vendors have to provide...

- Startup code, bootloader, linker scripts, register definitions
- Peripheral drivers
  - RTOS
  - Network stack
  - Cryptography, TLS stack
    - Filesystems
    - Protocols (HTTP, MQTT, CoAP,…, Cloud connectors)
  - POSIX abstractions
    - Build system
    - Profiling, tracing, debugging tools
    - IDE
SDKs: custom code

- Startup code, bootloader, linker scripts, register definitions
- Peripheral drivers
  - RTOS
  - Network stack
  - Cryptography, TLS stack
    - Filesystems
    - Protocols (HTTP, MQTT, CoAP,…, Cloud connectors)
  - POSIX abstractions
    - Build system
    - Profiling, tracing, debugging tools
    - IDE
Vendor SDK choices

- Library packaging format: CMSIS-Packs, mBed & Arduino libraries, etc.
- Build system can deal with libraries and dependencies: CMake, Scons, Meson...
- Compile-time configuration support: CMSIS Packs, Kconfig
- Command line tools (build system, configuration tools, flashing the boards)
- Integration with IDEs
ESP-IDF Components

ESP-IDF

Components
- FreeRTOS
- WiFi Stack
- LwIP
- Drivers
  (etc)

Third party components
- Cloud
- Drivers
- Protocols

Build system

Tools

Application

Components
- Business logic
- Glue code
- Drivers

Makefiles

Configuration
Component Structure

**Typical component**
- include/
- Source files `.c`, `.cpp`, `.S`
- `CMakeLists.txt`
- `Kconfig`

**Wrapper component**
- Third-party library
- Platform layer
- `CMakeLists.txt`
- `Kconfig`

**Fully custom component**
- Custom directory structure
- `CMakeLists.txt`
- `Kconfig`
Requirements for developing ESP32 Applications

**Hardware:**
- An ESP32 board
- USB cable - USB A / micro USB B
- Computer running Windows, Linux, or macOS

**Software:**
- Toolchain to compile code for ESP32
- Build tools - CMake and Ninja to build
- ESP-IDF
- Editor - Eclipse IDE
IDF Eclipse Plugins aiming to provide better tooling capabilities, which simplifies and enhances standard Eclipse CDT for developing and debugging ESP32 IoT applications.
GitHub page:
https://github.com/espressif/idf-eclipse-plugin

Update Site:
https://dl.espressif.com/dl/idf-eclipse-plugin/updates/latest/
Current Release: 1.0.0-beta
Challenges we are trying to solve

• Toolchain configuration
• Environment variables Configuration
• Installing required ESP-IDF Tools
• End-to-End workflow
• Everything from Eclipse IDE
• Official support from Espressif
IDF Eclipse Plugin Features

- Installing ESP-IDF Tools
- Creating a new CMake IDF Project
- Create Launch Target
- Compiling the Project
- Flashing the Project
- Viewing Serial Output
IDF Eclipse Plugin Features

• sdkconfig Editor
• CMake Editor [https://github.com/15knots/cmakeed](https://github.com/15knots/cmakeed)
• Configuring Build Environment Variables (auto-configured)
• Configuring Core Build Toolchain and CMake Toolchain (auto-configured)
• Configuring the flash arguments (auto-configured)
• ESP-IDF Templates selection during project creation
• Import an existing project
GDB Hardware Debugging Plugins
• Part of Eclipse CDT Package

Open OCD Debugging Plugins
Debugging with JTAG

- xtensa-esp32-elf-gdb: on-chip debugger
- OpenOCD: USB
- JTAG adapter
- ESP32

Application Loading and Monitoring
- make: compile / build
- esptool.py: flash / monitor
- USB / UART interface
- Single USB connection on ESP32-WROVER-KIT
IDF Eclipse Plugin Prerequisites

• **Java 8** and above
• **Python 3.5** and above
• **Eclipse 2018-12 CDT** and above
• **ESP-IDF 4.0** and above
Demo
Future Plans

- Keep improving the Eclipse support for ESP-IDF
- Automate all the configuration part
- Support Multiple Chip targets
- JTAG Debugger customization for ESP boards
- Evaluating Core Dump and GDB Stub Debugging
- Static Code analyzer
ESP-IDF Learning Resources

ESP-IDF Eclipse Plugin
https://github.com/espressif/idf-eclipse-plugin

ESP-IDF
https://github.com/espressif/esp-idf/

ESP-IDF Official documentation

ESP32 Forum
https://esp32.com
EVALUATE THE SESSIONS

Sign in and vote using the conference app or eclipsecon.org