iocafe

open source IoT/IO device communication

testing development tools and secure communications

– Visual Studio Code, Platform IO, Arduino libs, TLS, GDB, Esp-Prog/JTAG, and ESP32 –

190915, updated 15.9.2019/pekka

A Lot of acronyms in the title. There are many options how to do micro-controller development, and this is one combination. The goal here was to evaluate the development environment and to ensure that TLS works from ESP32.

Hardware

The microcontroller development board: MELIFE ESP32 Development Board (from Amazon).

External USB/JTAG debugger: ESP-Prog (from Grid Connect).

posts

Installing NASA Core Flight System and OpenSatKit to Virtual Machine

PlatformIO is great

Democar project

ESP32: 1 MHz clock signal to output pin

Configuring micro-controller WiFi with Android phone's LED flash

IOCOM in a simple multiplayer game?

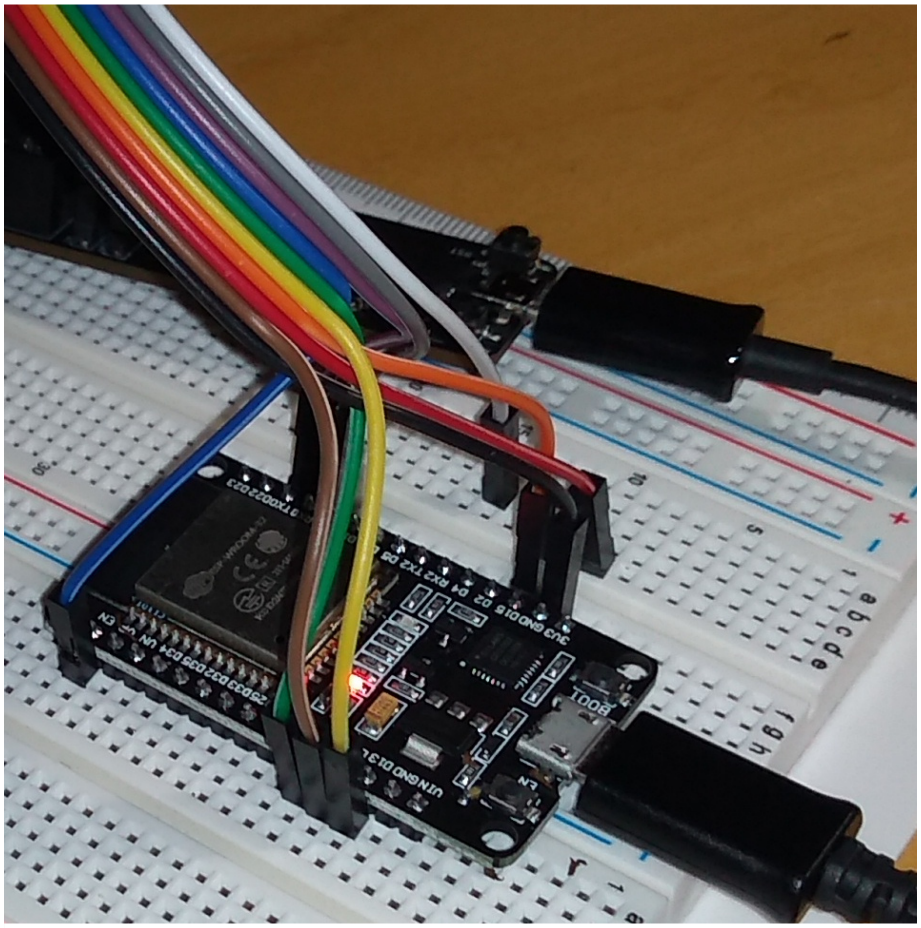
iocom secure network topology

testing development tools and secure communications

comments

Marco Roda on Configuring micro-controller WiFi with Android phone's LED flash

Get Your Microcontroller Online at the Speed of Light – UpMyTech on



ESP32 development board and ESP-Prog on background

It is a bit of effort to find correct pins to connect. Initially, I powered the ESP32 trough Esp-Prog. This didn't provide enough current, causing "brownouts" at ESP when Wifi started up. Solved by disconnecting red +3.3V wire between ESP32 and Esp-Proj, and connecting power to ESP32 with it's own USB connector. I am unsure how will this connect the grounds of the two USB cables together? Anyhow after this hardware worked fine.

ESP-idf – the ESP32 support and development

The first thing needed to develop for ESP32 is esp-idf by Espressif, the chipmaker. It is solid and good, and can be used to build, load and monitor ESP32 code as it. But it is not a full-fledged development environment with a debugger, other components are needed. If one wants a GUI, choices are Eclipse or Visual Studio code. I vote for Visual Studio Code, even it is

Configuring micro-controller WiFi with Android phone's LED flash

pekkalehtikoski on Configuring micro-controller WiFi with Android phone's LED flash

pekkalehtikoski on Configuring micro-controller WiFi with Android phone's LED flash

Get Your Microcontroller Online at the Speed of Light | 3d print ......errori ed esperienze, on Configuring micro-controller WiFi with Android phone's LED flash

archived

July 2020

May 2020

April 2020

March 2020

December 2019

September 2019

tags

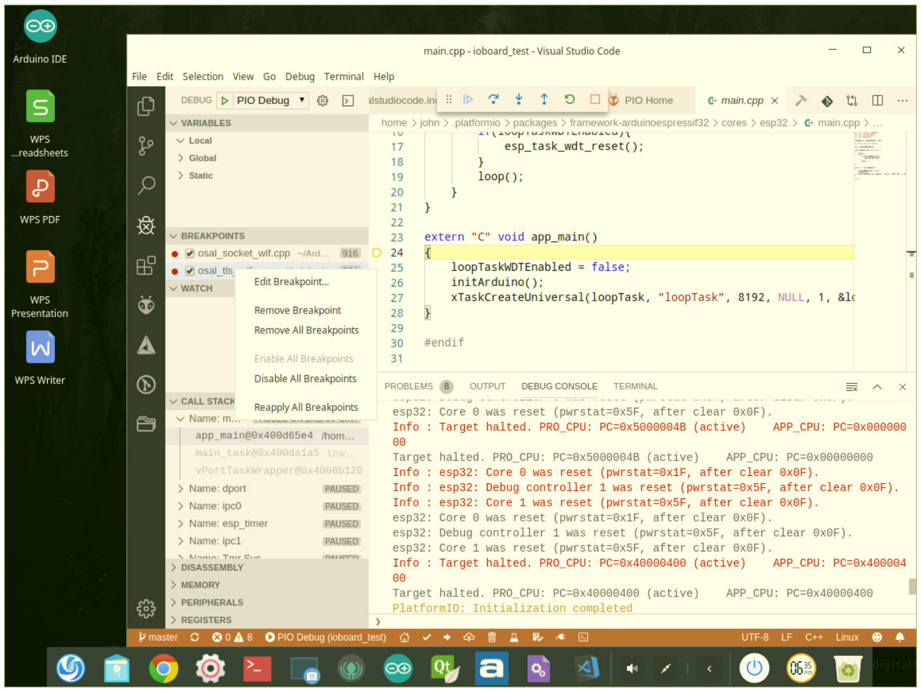
cFS core Flight System Cosmos cubesat democar esp32 espprog io-domain iocafe iocom mac NASA OBC OpenSatKit platformio programming flash release 2020 satellite secure network topology tls virtualbox virtual machine visual studio code tricky to set up it is still easier than Eclipse and works with Platform IO

Arduino libraries – nice to use and pain to debug

This is a love/hate relationship. I find Arduino libraries easiest way to write micro-controller code which is at least “about portable” from micro-controller to another. Way of distributing libraries as .zip files containing sources is also good, and the best part is that the application stays small, simple and readable. Hardware dependent code is not bundled in.

But then. It is an environment which has not given any taught on debugging, which is painful to put it nicely.

Options that have worked for me are using gdb -tui from the command line (hackaday) or using Visual Studio Code + Platform IO + GDB. Both have their own tricks and are not easy to set up or use, but get the job done. As a preference I think Visual Studio Code is better of the two, still, it is difficult to get to work if one is not intimately familiar with the Visual Studio Code and Platform IO configuration.



```
            esp_task_wdt_reset();
        }
        loop();
    }
}

extern "C" void app_main()
{
    loopTaskWDTEnabled = false;
    initArduino();
    xTaskCreateUniversal(loopTask, "loopTask", 8192, NULL, 1, &lo
}

#endif
```

```
esp32: Core 0 was reset (pwrstat=0x5F, after clear 0x0F).
Info : Target halted. PRO_CPU: PC=0x5000004B (active)    APP_CPU: PC=0x000000
00
Target halted. PRO_CPU: PC=0x5000004B (active)    APP_CPU: PC=0x00000000
Info : esp32: Core 0 was reset (pwrstat=0x1F, after clear 0x0F).
Info : esp32: Debug controller 1 was reset (pwrstat=0x5F, after clear 0x0F).
Info : esp32: Core 1 was reset (pwrstat=0x5F, after clear 0x0F).
esp32: Core 0 was reset (pwrstat=0x1F, after clear 0x0F).
esp32: Debug controller 1 was reset (pwrstat=0x5F, after clear 0x0F).
esp32: Core 1 was reset (pwrstat=0x5F, after clear 0x0F).
Info : Target halted. PRO_CPU: PC=0x40000400 (active)    APP_CPU: PC=0x400004
00
Target halted. PRO_CPU: PC=0x40000400 (active)    APP_CPU: PC=0x40000400
PlatformIO: Initialization completed
```

Visual Studio Code/Platform IO debugger on Deepin virtual machine

TLS – secure transport

The environment comes with Mbed TLS by default. It worked easily > I was happy. The IOCOM library traffic flows nicely from ESP32 to Linux or Windows (or to anything else) trough secure socket. I have still not implemented device identification. So even the communication is encrypted, device and control computer cannot be certain with whom they communicate. This is on to-do list.

vmware virtual machines and other easier choices

Setting up a micro-controller development environment like this one takes quite a bit of time and Googling. There are many setting, software version requirements, etc, which need to be sorted out. This makes me think that a good way for a new person to get started is to download a ready virtual machine for the specific setup. Even if one wants to set up a native development environment, the virtual machine can be used as a working reference. Then some tools are relatively easy, especially Arduino IDE is simple and easy (without debugging), no need for the virtual machine. Atollic True Studio for SMT32 chips is also easy but tends to bundle much HW dependent code in.

When you get ESP32 make sure that it has USB connector which can be used for debugging (get ESP-WROVER-KIT), no need to fiddle with wires and pins. See Espressif for debugging info.

Comment and corrections welcome

This text is based on my experiences and can be improved. Comments, proposals, and corrections are very welcome.

Tagged on: esp32 espprog platformio visual studio code

pekkalehtikoski September 15, 2019

iocom library development and testing No Comments

← running the virtual machine on Mac

iocom secure network topology →

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Save my name, email, and website in this browser for the next time I comment.

Post Comment