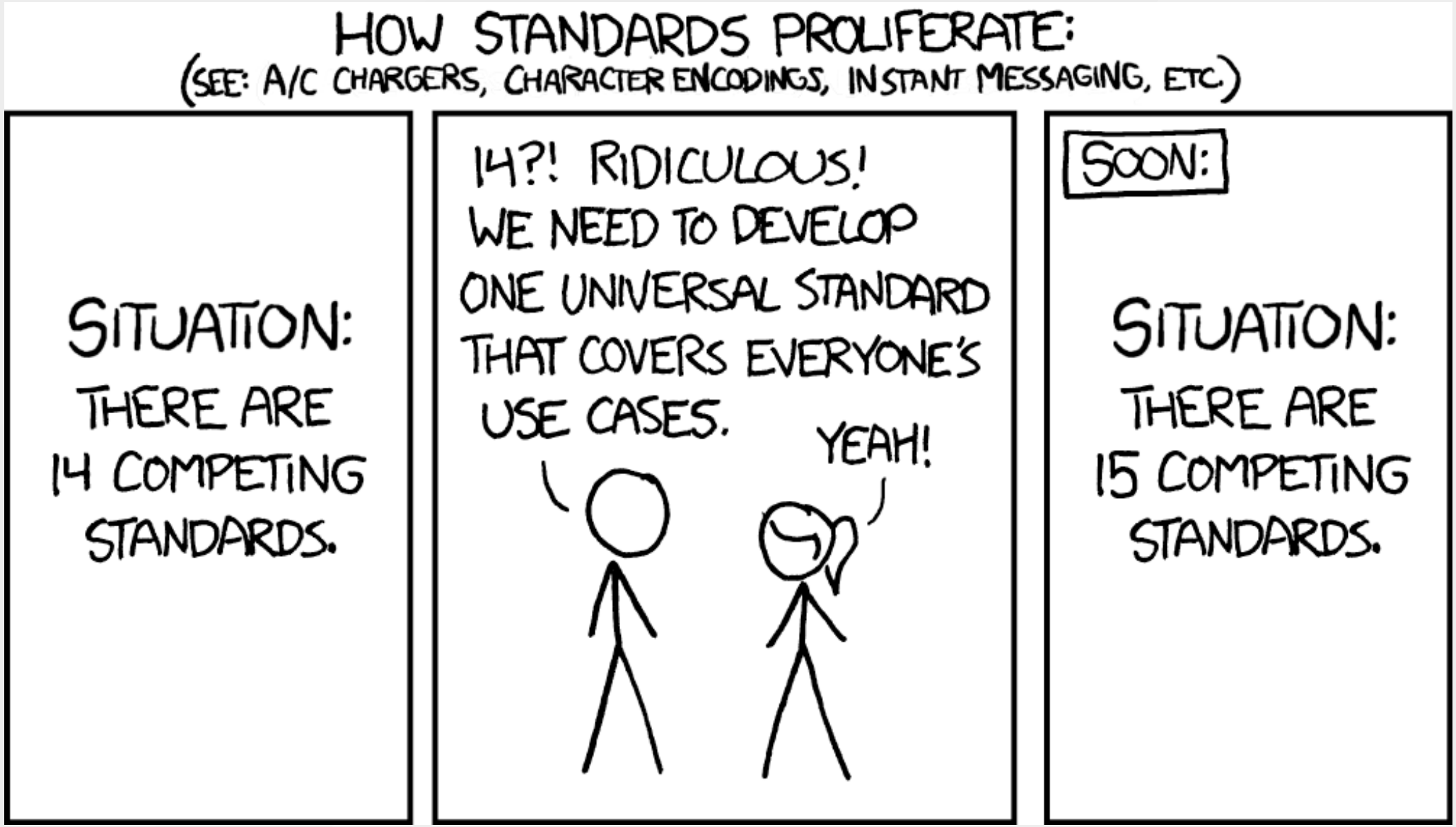
EEF - Engineero Embedded Framework

One framework to rule them all


HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)
SITUATION: THERE ARE 14 COMPETING STANDARDS.
14?! RIDICULOUS! WE NEED TO DEVELOP ONE UNIVERSAL STANDARD THAT COVERS EVERYONE'S USE CASES.
YEAH!
SOON:
SITUATION: THERE ARE 15 COMPETING STANDARDS.

EEF - Engineero Embedded Framework

One framework to ~~rule~~ wrap them all

What is EEF, the Engineero Embedded Framework?

- C framework for embedded applications with abstracted vendor HAL's and (RT)OS functions
- Simply build an application for a wide range of MCU's and OS'es
  - Currently supported are stm32f1, stm32f2, NXP's RT1170 and RP2040
  - Currently only FreeRTOS support

Why EEF?

- Internal project to keep us happy
- No fully hardware independent framework
  - Ignore the other frameworks like CMSIS for a second here 😜
- Unified build system
  - Everything in CMake
- With devcontainer support: IDE independent
  - You should love VSCode though

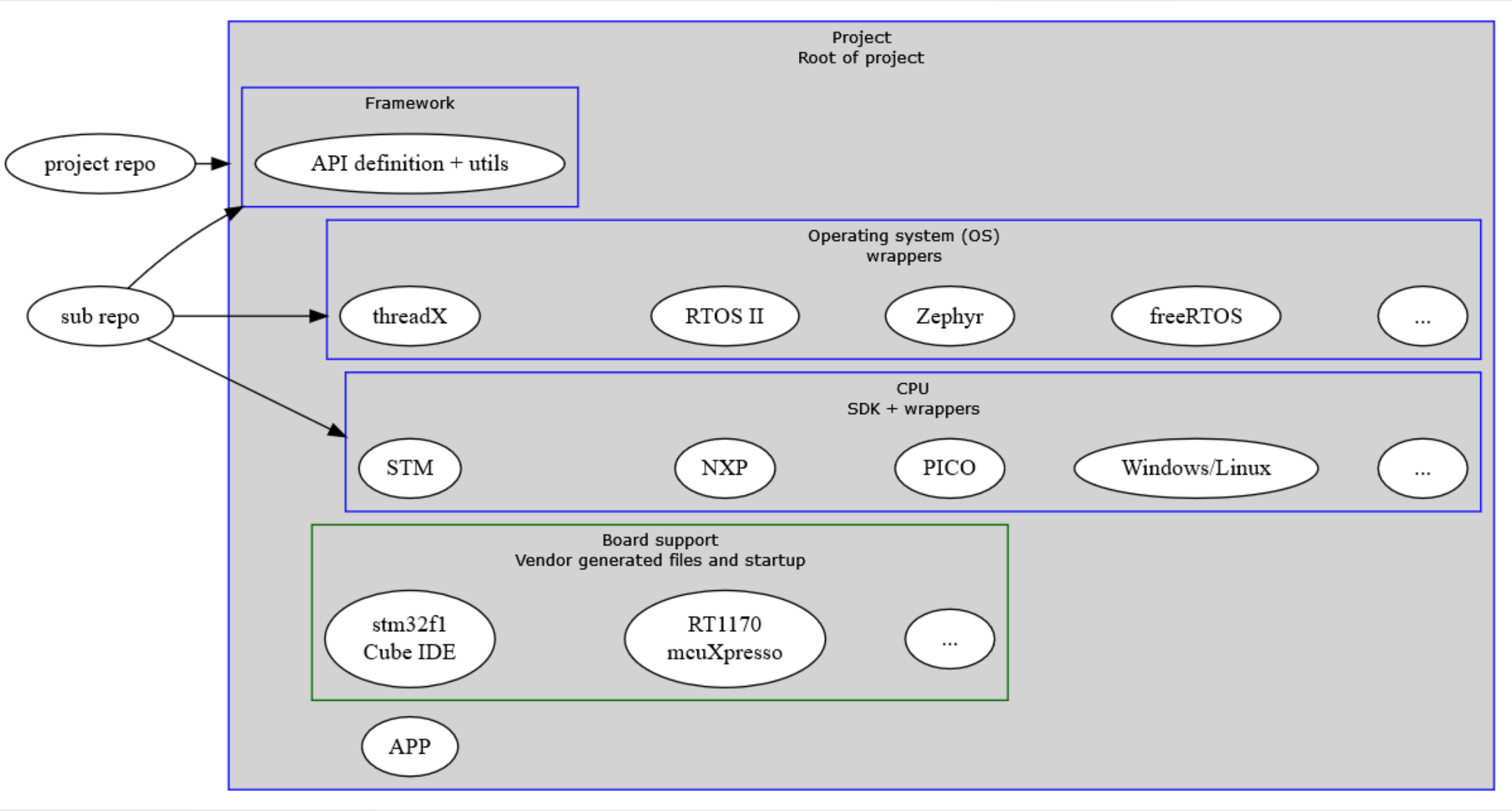
```c
void appThread(void *arg){
    for(;;){
        EEF_gpioSetDigital(LED, 1);
        EEF_delayMs(500);
        EEF_gpioSetDigital(LED, 0);
        EEF_delayMs(100);
    }
}

void EEF_Start(){
    EEF_threadInitialize();

    int appThreadID;
    EEF_threadCreate("app", appThread, NULL, 256, 1, &appThreadID);
}
```

Difficulties faced and overcome

- Vendor provided libraries ≠ buildsystem independent
- Unified API with different underlying vendor HAL's
  - API broad enough but not too broad
- General project organisation with vendor SDK's with different needs

General setup

- Repo for general utilities and API for OS/CPU
- Repo per OS with wrapper implementations for the OS API
- Repo per CPU with wrapper implementations for the CPU API
  - May contain vendor delivered SDK with HAL
- Main project contains BSP's with startup and configuration files
  - Vendor tool generated
  - Mapping between EEF and vendor SDK

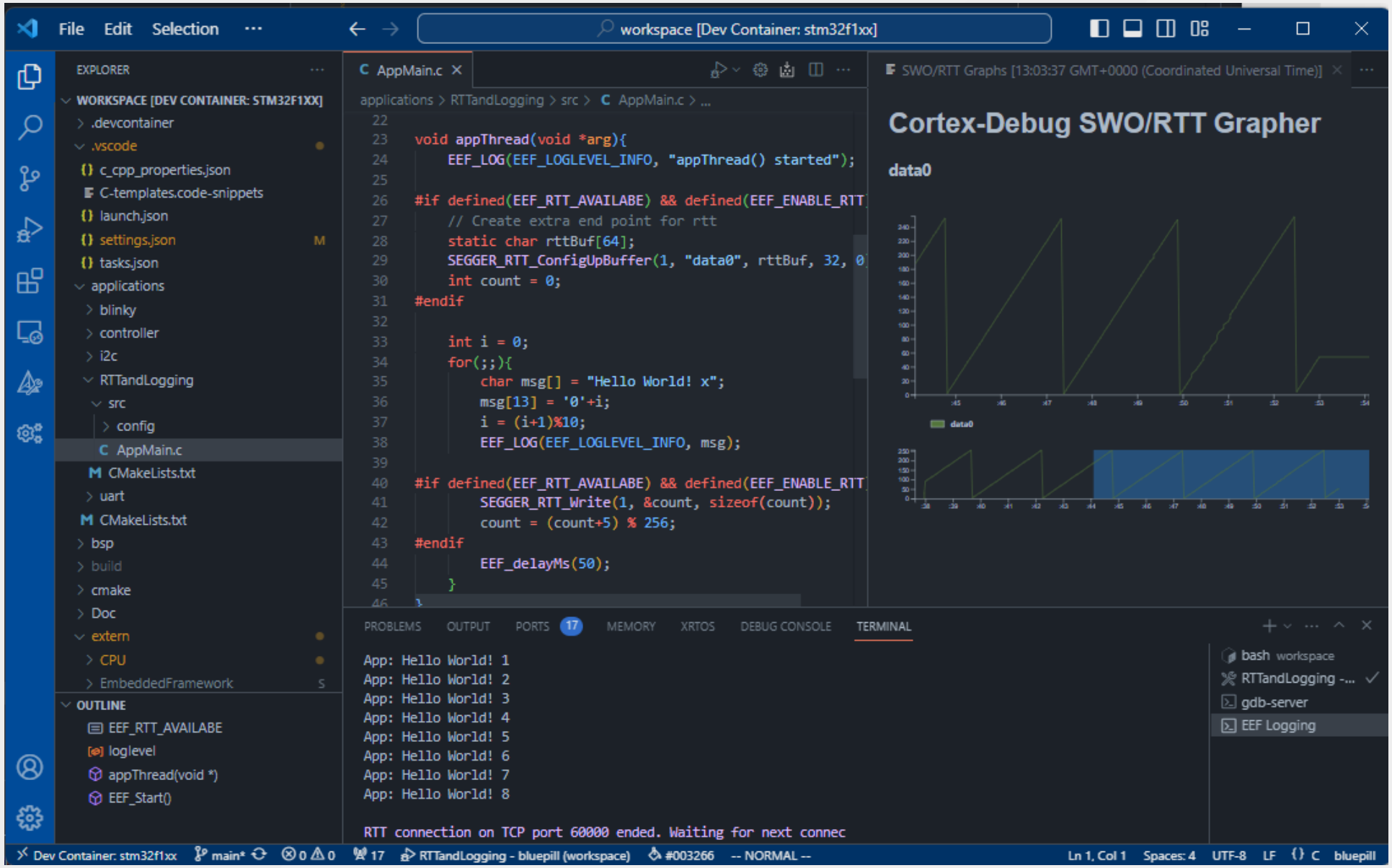

Project
Root of project
Framework
API definition + utils
project repo
sub repo
Operating system (OS)
wrappers
threadX
RTOS II
Zephyr
freeRTOS
...
CPU
SDK + wrappers
STM
NXP
PICO
Windows/Linux
...
Board support
Vendor generated files and startup
stm32f1
Cube IDE
RT1170
mcuXpresso
...
APP

Give it a go at

Github

Give it a go

- Clone examples repo (on WSL)
- Open in VSCode and enter STM32F1xx devcontainer
- Play around with blinky and RTTandLogging
  - RTTandLogging -> example for VSCode + devcontainers


File Edit Selection
workspace [Dev Container: stm32f1xx]
EXPLORER
WORKSPACE [DEV CONTAINER: STM32F1XX]
.devcontainer
.vscode
c_cpp_properties.json
C-templates.code-snippets
launch.json
settings.json
tasks.json
applications
blinky
controller
i2c
RTTandLogging
src
config
AppMain.c
CMakeLists.txt
uart
CMakeLists.txt
bsp
build
cmake
Doc
extern
CPU
EmbeddedFramework
OUTLINE
EEF_RTT_AVAILABE
loglevel
appThread(void *)
EEF_Start()
AppMain.c
applications > RTTandLogging > src > AppMain.c > ...
void appThread(void *arg){
EEF_LOG(EEF_LOGLEVEL_INFO, "appThread() started");
#if defined(EEF_RTT_AVAILABE) && defined(EEF_ENABLE_RTT
// Create extra end point for rtt
static char rttBuf[64];
SEGGER_RTT_ConfigUpBuffer(1, "data0", rttBuf, 32, 0
int count = 0;
#endif
int i = 0;
for(;;){
char msg[] = "Hello World! x";
msg[13] = '0'+i;
i = (i+1)%10;
EEF_LOG(EEF_LOGLEVEL_INFO, msg);
#if defined(EEF_RTT_AVAILABE) && defined(EEF_ENABLE_RTT
SEGGER_RTT_Write(1, &count, sizeof(count));
count = (count+5) % 256;
#endif
EEF_delayMs(50);
}
SWO/RTT Graphs [13:03:37 GMT+0000 (Coordinated Universal Time)]
Cortex-Debug SWO/RTT Grapher
data0
data0
PROBLEMS OUTPUT PORTS 17 MEMORY XRTOS DEBUG CONSOLE TERMINAL
App: Hello World! 1
App: Hello World! 2
App: Hello World! 3
App: Hello World! 4
App: Hello World! 5
App: Hello World! 6
App: Hello World! 7
App: Hello World! 8
RTT connection on TCP port 60000 ended. Waiting for next connec
bash workspace
RTTandLogging -...
gdb-server
EEF Logging
Dev Container: stm32f1xx main* 0 0 17 RTTandLogging - bluepill (workspace) #003266 -- NORMAL -- Ln 1, Col 1 Spaces: 4 UTF-8 LF C bluepill