Tests

Pieter P

Compiling the library and the unit tests

It's crucial to compile your code regularly while you're writing code. It'll uncover problems that your IDE doesn't know about.

Running CMake

Use the CMake Tools extension to configure the project using CMake. (`CTRL+SHIFT+P`, “CMake: Configure”)

To be able to generate the coverage report, use the `Clear Build Directory and Run CMake` task. (`CTRL+P`, “task Clear Build Directory and Run CMake”)

Compiling

To compile the Control Surface library and its unit tests, you can use the `CTRL+SHIFT+B` shortcut (build).
Alternatively, you can press `CTRL+P`, type "task", space, and then "Build All".

Running the tests

To run the tests, press `CTRL+Shift+P`, type "Tasks: Run Test Task". There should be three options.

- Run Tests (gtest): builds any files that were modified, runs the tests, and prints the output to the console.
- Run Tests: equivalent to Run Tests (gtest), but it only prints the test results to the console, not the output (print statements) of the tests. It is slower because it runs each test in isolation
- Build & Test & Build examples: the same as Run Tests (gtest), and after running the tests, it also compiles all Arduino examples for one specific platform, specified in `.vscode/tasks.json`. This can take a significant amount of time.

Before committing, Build & Test & Build examples has to pass.

Compiling using a terminal

If you're not using VSCode, you can compile everything from a terminal.
Simply navigate to the `Control-Surface/build` directory, and invoke CMake.
You can speed up the compilation by compiling multiple files in parallel using the `-j` option. `nproc` returns the number of CPUs available on your machine.

```
$ cmake ..
$ cmake --build . -j$(nproc)
```

Running the tests using a terminal

Open a terminal in the `Control-Surface/build` directory, compile everything (see previous paragraph) and then run the tests:

```
$ ctest
```

To compile the Arduino examples, open a terminal in the `Control-Surface/tools` directory, and run:

```
$ python3 arduino-example-builder.py 'AVR USB'
```

You can replace `'AVR USB'` with any platform specified in `test/examples-board-fqbns.yaml`.