Lab 3 exercises

June 24, 2025

1. Add the provided source/header files to the project as described below:
    1. Copy the files/folders to the project root
    2. Open `armgcc\CMakeLists.txt`
    3. Look for:

```
add_executable(${MCUX_SDK_PROJECT_NAME}
"${ProjDirPath}/../main.c"
"${ProjDirPath}/../peripherals.c"
[...]
)
```

        1. Add the newly introduced files to the build system (e.g: `"${ProjDirPath}/../src/memcopy_functions.c"`)
2. Test the `check_prime` function implementation (declaration available at: `src\functions_under_test.h`, implementation available at: `src\check_prime.c`).
    a. Identify test scenarios.
    b. Implement a test function named `test_check_prime` using this source file: `tests\test_check_prime.c`. The test function should run multiple test scenarios and provide a descriptive return code and message (using `error_message`) in case of failure. The Function Under Test (FUT) will be called inside this test function.
3. Test the `memcopy_n` function implementations. To do this, you may need to implement some helper functions (since these can be used in multiple different testing scenarios and will make the testing code more readable).
    1. Implement `add_fencing`, `check_fencing` and `check_result` functions, inside `test_utils.c`. A short description of these functionalities is provided inside `test_utils.h`.
    2. Implement test_memcopy. This test function should be able to receive a function reference as an input parameter and call any of the 3 memcopy functions: `memcopy_1`, `memcopy_2`, `memcopy_3`, available at `src\memcopy_functions.c`. These functions' implementations were obfuscated to create a "closed-box testing" experience.
        - Suggestion: use the `test_params` buffer to make the test function more configurable (you can then send different input parameters to test different test vectors).
        - Use Hercules to send data for: test selection, input parameters. Anything can be changed/updated inside this project **BUT** the `check_prime` and `memcopy_n` implementations.