🔧 UDP Server Example, debug print missing

1. Problem Phenomenon (The issue):
    - The user is not seeing the expected debug print statements on the serial terminal when running the `w5x00_udp_server.c` example on a W55RP20-EVB-PICO board, despite enabling the `_LOOPBACK_DEBUG_` macro.
2. Cause (The cause of the issue):
    - The `_LOOPBACK_DEBUG_` macro is intended to print messages only when there is a send or receive error, not for mirroring received UDP packets to the UART port as the user assumed.
3. Solution (Concrete code changes, configuration changes, step-by-step actions, etc., that actually solved the problem):
    - To change the UART or USB message output settings, modify the options in the `CMakeLists.txt` file located at `WIZnet-PICO-C\examples\udp\udp_server\` by setting the desired output to 1:

      ```
      pico_enable_stdio_usb(${TARGET_NAME} 1)
      pico_enable_stdio_uart(${TARGET_NAME} 0)
      ```

    - This configuration enables USB output and disables UART output. Adjust these settings according to the desired output method.

View GitHub Issue

🔧 Failed compilation on Linux Evironment

1. Problem Phenomenon (The issue):
    - When running the `make` command, an error occurs during the linking stage of the `w5x00_tftp_client.elf` executable. The error reports undefined references to the function `save_data` in the `tftp.c` file, specifically at lines 440 and 423.
2. Cause (The cause of the issue):
    - The cause of the issue is likely missing or incorrect definitions for the `save_data` function in the `tftp.c` file, leading to undefined reference errors during the linking process.
3. Solution (Concrete code changes, configuration changes, step-by-step actions, etc., that actually solved the problem):
    - Apply a patch to the project to resolve the undefined reference errors. Follow these steps:
        1. Navigate to the `ioLibrary_Driver` directory within the project:

           ```
           cd libraries/ioLibrary_Driver
           ```

2. Apply the patch using the `git apply` command:

```
git apply ../../patches/0002_iolibrary_driver_tftp.patch
```

- For additional information or troubleshooting, refer to the provided README document.
- If the issue persists, further support may be necessary.

View GitHub Issue

🔧 How to add correctly the mbedtls project as submodule

1. Problem Phenomenon (The issue):
   - The user is trying to add the mbedtls library as a submodule using a Git command. However, when configuring with CMake, some files are missing, and the user is unable to resolve the issue by updating within the mbedtls folder.
2. Cause (The cause of the issue):
   - The issue arises because the user is not correctly adding the mbedtls library as a submodule and is possibly not configuring the path correctly in the CMakeLists.txt file. Additionally, there might be a manual inclusion of a configuration file that is not needed in the specific version being used.
3. Solution (Concrete code changes, configuration changes, step-by-step actions, etc., that actually solved the problem):
   - Update the path of the mbedtls library in the **WIZnet-PICO-C/CMakeLists.txt** file (line 80) by setting the correct directory:

```
set(MBEDTLS_DIR ${CMAKE_SOURCE_DIR}/libraries/mbedtls)
```

   - To add a new version of mbedtls as a submodule, follow these steps:
     1. Clone the specific version needed (replace 3.6.2 with the desired version):

```
git clone --branch mbedtls-3.6.2 --depth 1
https://github.com/Mbed-TLS/mbedtls.git libraries/mbedtls
```

     2. Navigate to the mbedtls directory:

```
cd libraries/mbedtls
```

     3. Initialize and update the submodules:

```
git submodule update --init --recursive
```

- If the warning `#warning "Do not include mbedtls/check_config.h manually!"` is encountered, comment out the following line in the **WIZnet-PICO-C/port/mbedtls/inc/ssl_config.h** file at line 32:

```
// #include "mbedtls/check_config.h"
```

- These steps ensure that the correct version of mbedtls is used and configured properly, and unnecessary warnings are avoided. If further issues arise, provide the complete execution log for additional assistance.

View GitHub Issue

🔧 Issue: W55RP20-EVB-PICO Firmware Not Booting After Building

1. **Problem Phenomenon (The issue):**
   - After building and flashing the firmware for the W55RP20-EVB-PICO board using the specified repository, the board fails to boot up. The firmware file (`firmware.uf2`) is generated successfully, but the board does not operate as expected once flashed.
2. **Cause (The cause of the issue):**
   - The issue arises because the repository in question uses Pico-SDK version 1.5.0, which does not support the necessary clock initialization changes that are required for the board to function correctly. There is a known patch (`0001_pico_sdk_clocks.patch`) that addresses similar issues, but it is designed to work with Pico-SDK 2.0.0. Consequently, the lack of support for the required changes in version 1.5.0 leads to the board's failure to boot.
3. **Solution (Concrete code changes, configuration changes, step-by-step actions, etc., that actually solved the problem):**
   - Two potential solutions are suggested:
     1. Update the repository to use Pico-SDK 2.0.0, which is compatible with the existing patch and should resolve the clock initialization issue.
     2. Develop and provide a new patch that works with Pico-SDK 1.5.0 to address the clock initialization issue.
   - Additionally, there is an ongoing pull request for MicroPython W55RP20-EVB-PICO that supports Pico-SDK 2.0.0, which can be referred to for guidance. However, caution is advised as this PR is still in progress and may be subject to further changes.

View GitHub Issue