Software Release Notes for

TUSB3410 WDF-USB2UART Driver v6.7.1

Document Revision: 1.0
Date: Sep 21, 2011

1. Details of Release

1.1. Contents of Release:

1. Source code (usbuart drivers)
2. Binaries for Windows Win8 32-bit and 64-bit Release
3. Both 32-bit and 64-bit versions are WQHL certified.
4. TI_WDF_USBUART_SINGLE_DRIVER_TI_V6.7.1.0.zip

1.2. Features/Modules added:

- Fix for HCK's "Concurrent Hardware and Operating System (CHAOS) Test (Certification)" test failure.

2. Installation Instructions

2.1. Pre Installation Requisites:

1. Windows Operating System (Win8) with latest SP
2. TUSB3410 Hardware
3. USB-UART driver binaries

2.2. Installation Procedure:

1. Extract the Zip file to appropriate folder.
2. Copy all the Setup files (Executable) to a specific folder.
3. Attach the TUSB4310 device and proceed with enumeration process.
5. Run the setup.exe file and complete the installation process.

2.3. Post Installation Procedure:

Open device manager and verify if driver is loaded and running.

3. Assumptions, dependencies and constraints

-NA-

4. Build Instructions

1. Use latest Windows Driver Kit (Currently, WDK 7600.16385.1 is the latest version)
2. Open the build from Start Menu→Programs→Windows Driver Kits→WDK 7600.16385.1→Build Environments. Select required operating system environment.
3. Browse to the path where source code is available.

4. Type the command “build –cz” to build the driver.
5. A .sys file is generated in the source code path with appropriate OS name as folder name.

5. Known Defects / Bugs / Issues

-NA-

6. Special Comments, if any

1. **Root cause of Problem**: The HCK CHAOS test sending the dummy read/write requests and followed by calling the cancellation routine for cancelling the same request as it is a dummy request. The driver marks the request as pending and sends status as pending after receiving the request from the framework. There is a read polling thread which tries to complete the pending request. This read polling thread starts executes the request and before calling the port read function it saves some of the request pointers in the local variables and it releases the lock. After returning from the port read function it again acquires the lock. In between i.e. after releasing and acquiring the lock, the framework calls the cancellation routine for cancelling the same request as it is a dummy and it cancelled. But after acquiring the lock, we are accessing the Irp associated with the request that had already been cancelled and this result in a BSOD.

   Resolution: Included a condition, whether the request is still present or cancelled, before accessing the Irp associated with that request in read and write function paths.

   Files modified: UmpComPort.c

   Functions added/modified: SerialReadRxBuffer() and SerialWriteTxBuffer().

NOTE: Removed a few conditions, which were added in the earlier release, in timer functions. These conditions were added to send the status success in case of timer is not initialized for a request. Due to these condition checks, a customer device was not installed properly. After removing these conditions, the customer device installed successfully.

7. Exceptions / Waivers

-NA-

Note: Specify NA if anything is not applicable

Appendix A: Test-Cases

-NA-

Appendix B: Test-Log / Report

-NA-