UART
Self-diagnosis Program
Application Note

Outlines

This application note describes the self-diagnosis program for the UART.
The self-diagnosis program in this document is the library which is used to check the UART serial communication.
This library supports the sample program with the peripheral driver of TXZ series enclosed, and should be used after it is overwritten to the sample program.

Table of Contents

Arm, Cortex and Keil are registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

All other company names, product, and service names mentioned herein may be trademarks or registered trademarks of respective companies.

1. Preface

This application note describes the self-diagnosis program which uses the UART.
This library code should be used after it is added (overwritten) to the published sample program.

The explanation of this document uses the TMPM4K Group (1).
When the program is applied to a product, some appropriate modification may be necessary depending on the specifications of the product.

This sample program has been developed and evaluated under the conditions in the operation confirmation environment in “Self-diagnosis Program Application Note – Basic Setting”, and by using the TMPM4K4A 1.0.0 sample program and the reference manual released in February in 2019.

2. Outline of Self-diagnosis Test Library

This self-diagnosis test library has been checked on the evaluation board.

The following self-test function is supported.

Name	Description
Serial communication test	The internal loop-back is set in the UART port. And it is checked that a transmission data and its reception data are identical.

2.1. Self-diagnosis Sample Project

In the “Project\Examples\Safety” folder, the following sample project for the self-diagnosis library is placed.

Project name	Operation	Utilized self-diagnosis library function
UART_Sample	A loop-back test is done using the UART. The result is shown with the LED lighting.	safety_UART_loopback()

3. Details of Self-diagnosis Test Library

This section describes the details of the self-diagnosis test library function.
This sample program is a self-diagnosis program for the UART.
The following setting is an example when a product in the TMPM4K group (1) is used.

The source code (.c file) of the self-diagnosis library is in the “Libraries\Safety\src” folder, and the header file (.h file) is in the “Libraries\Safety\inc” folder.

3.1. Serial Communication Test

An internal loop-back is set for the UART port. And it is checked that a transmission data and its reception data are identical.
This library function distinguishes a used UART port with its ID. The call of its interrupt handler uses the same ID. The transmission rate is fixed to 115200 bps. The timeout is detected with the time length which is calculated with the data length utilizing the SysTick interrupt.

The “safety_UART_loopback” function uses the interrupt.

Source file: safety_uart.c
Header file: safety_uart.h
Used library: txz_cg.c/.h, txz_gpio.c/.h, txz_hal.c/.h, and txz_uart.c/.h

Function name	
bool safety_UART_loopback(SafetyUartID id, uint32_t start, uint32_t length)	
Input parameter	

SafetyUartID id	The enum value indicating the UART to use is one of the following.
	SAFETY_UART0_ID = 100
	SAFETY_UART1_ID = 101
	SAFETY_UART2_ID = 102
	SAFETY_UART3_ID = 103

uint32_t start	Start address of the used data
uint32_t length	Used data length (Byte unit)
Output parameter	
None.	-
Return value	
bool	Result (true: success, false: failure = a parameter error, result difference, a timeout, and others)

* This sample program used “SAFETY_UART2_ID”.
* If the return value cannot be confirmed for several seconds, it is supposed the test is not executed correctly. The process for the test failure should be done.
When, however, the terminal I/O output display is used, it takes several seconds to complete the display. The judgment of the test failure should be done by checking the display.

The test result is shown with the LEDs after all the tests finish on the evaluation board which was used to develop this test program.
LED1 (PJ0) lighting: All tests are successful.
LED2 (PJ2) lighting: The test fails.

Note:
An appropriate port initialization for the tested UART channel and the implementation of the interrupt handler are necessary to use this library function.
The test target UART is supposed to be used for real communication in this test. The following three interrupt handlers should be implemented; safety_UART_hook_rx () / safety_UART_hook_tx () / safety_UART_hook_err () to hook the interrupt only while the safety_UART_loopback function is called.

1. Implementation of Initialization function

The initialization status of the bsp.c in each UART port is shown in the following table. When a UART other than UART0 is used, its port should be initialized.

UART port	Port initialization in bsp.c
UART0	Implementation
UART1	None.
UART2	None.
UART3	None.

The initialization is done as follows (an example for the UART1);

```
        // setup A0/A1 for UART1
        // PA0/PA1 default use in "bsp.c" is SPI1
        gpio_func(p_gpio, GPIO_PORT_A, 0, GPIO_PA0_UT1RXD, GPIO_PIN_INPUT);
        gpio_func(p_gpio, GPIO_PORT_A, 1, GPIO_PA1_UT1TXDA, GPIO_PIN_OUTPUT);
```

2. Implementation of Interrupt handler (hook)

The hooks of the transmission, the reception, and the error interrupts should be implemented, respectively.

UART port	Used interrupt number	Current implementation in bsp.c

UART0	INTSC0RX_IRQn (= 31)
	INTSC0TX_IRQn (= 32)
	INTSC0ERR_IRQn (= 33)
		irq_usb_uart_rx(BSP_USB_UART_0)
		irq_usb_uart_tx(BSP_USB_UART_0)
		irq_usb_uart_err(BSP_USB_UART_0)

UART1	INTSC1RX_IRQn (= 34)
	INTSC1TX_IRQn (= 35)
	INTSC1ERR_IRQn (= 36)
		irq_sflash_rx(BSP_SFLASH_1)
		irq_sflash_tx(BSP_SFLASH_1)
		irq_sflash_ex(BSP_SFLASH_1)

UART2	INTSC2RX_IRQn (= 37)
	INTSC2TX_IRQn (= 38)
	INTSC2ERR_IRQn (= 39)
		irq_sflash_rx(BSP_SFLASH_2)
		irq_sflash_tx(BSP_SFLASH_2)
		irq_sflash_ex(BSP_SFLASH_2)

UART3	INTSC3RX_IRQn (= 40)
	INTSC3TX_IRQn (= 41)
	INTSC3ERR_IRQn (= 42)
		irq_sflash_rx(BSP_SFLASH_3)
		irq_sflash_tx(BSP_SFLASH_3)
		irq_sflash_ex(BSP_SFLASH_3)

Every UART channel is converted in a call function in the main.c by the bsp.c.

So, a call of the hook should be implemented in the interrupt hander of the call destination, as follows. The same ID argument as set in safety_UART_loopback is transferred.

From Project\Examples\Safety\UART_Sample\src\main.c

```
void irq_usb_uart_rx(BSPUsbUart uart)
{
   // UART0 use this IRQ for RX
#if SAFETY_UART_CH==SAFETY_UART0
     safety_UART_hook_rx(SAFETY_UART0_ID);
#endif
}
```

* The transmission (TX) and the error (ERR) interrupts should be implemented as well.

The timeout is detected using the SysTick interrupt in the safety_UART_;loopback. Referring to “Implementation of SysTick Timer Interrupt”, the support of the hal_get_tick() function by the SysTick interrupt should be implemented.

4. Implementation of SysTick Timer Interrupt

This library functions safety_UART_loopback() use the txz_hal.c and the SysTick timer to measure time.

In these functions, it is necessary that the program implementation should be done correctly to measure the time length by the SysTick interrupt before the actual call is done.
* This setting has been implemented to this sample program.

The two followings should be implemented in order to use the SysTick timer.

1. Timer interrupt handler setting

The interrupt handler should be set in the main.c, as follows;

```
void irq_systick(void)
{
        hal_inc_tick();
}
```

2. SysTick interrupt start process

The start of the interrupt should be set at an appropriate location in the application_initialize() function or the main() function, as follows;

```
        {
            // start SysTick IRQ
            const uint32_t period = 80000;    // 80MHz / 1000Hz = 1msec SysTick

            (void)SysTick_Config(period);       /* systick interrupt cycle setting */

            // SysTick IRQ started
        }
```

Implementing the above, the hal_get_tick() function supplied by the txz_hal.c can acquire the count value in the units of 1 ms.

5. List of Used Drivers

This test library uses the driver and the code in the project of the TMPM4KxA_v1.0.0 version.

CMSIS library

Category	Source file name
Start-up	startup_TMPM4K4A.s
System (Clock setting and others)	system_TMPM4KxA.c

Periph_driver

Category	Source file name
UART	txz_uart.c
GPIO	txz_gpio.c
SysTick interrupt	txz_hal.c
Clock generator	txz_cg.c

In the Project examples

Category	Source file name
BSP (Evaluation board support)	bsp.c
LED output	bsp_led.c

6. Reference Document

For development, refer to the following documents.

- Datasheet of each product
- Reference Manual
- Self-diagnosis Program Application Note - Basic Setting
- ARM® Cortex®-M4 Processor technical Reference Manual
- ARMv7-M Architecture Reference Manual

7. Revision History

Revision	Date	Description
1.0	2019-08-27	First release

RESTRICTIONS ON PRODUCT USE

Toshiba Corporation and its subsidiaries and affiliates are collectively referred to as “TOSHIBA”.
Hardware, software and systems described in this document are collectively referred to as “Product”.

- TOSHIBA reserves the right to make changes to the information in this document and related Product without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA's written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating all operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS' PRODUCT DESIGN OR APPLICATIONS.**
- **PRODUCT IS NEITHER INTENDED NOR WARRANTED FOR USE IN EQUIPMENTS OR SYSTEMS THAT REQUIRE EXTRAORDINARILY HIGH LEVELS OF QUALITY AND/OR RELIABILITY, AND/OR A MALFUNCTION OR FAILURE OF WHICH MAY CAUSE LOSS OF HUMAN LIFE, BODILY INJURY, SERIOUS PROPERTY DAMAGE AND/OR SERIOUS PUBLIC IMPACT ("UNINTENDED USE").** Except for specific applications as expressly stated in this document, Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, lifesaving and/or life supporting medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, and devices related to power plant. **IF YOU USE PRODUCT FOR UNINTENDED USE, TOSHIBA ASSUMES NO LIABILITY FOR PRODUCT.** For details, please contact your TOSHIBA sales representative or contact us via our website.
- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.
- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.
- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the applicable export laws and regulations including, without limitation, the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.
- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. **TOSHIBA ASSUMES NO LIABILITY FOR DAMAGES OR LOSSES OCCURRING AS A RESULT OF NONCOMPLIANCE WITH APPLICABLE LAWS AND REGULATIONS.**

TOSHIBA ELECTRONIC DEVICES & STORAGE CORPORATION
https://toshiba.semicon-storage.com/