Clock Frequency/Interrupt Self-diagnosis Program Application Note

Outlines

This application note describes the self-diagnosis program to check that the error detection of the clock frequency and the interrupts operate correctly. The self-diagnosis program in this document is a library to check the clock frequency and the interrupts.
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

1. Preface

This application note is the manual for the self-diagnosis program using the clock frequency and the interrupts.
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
Clock frequency test	Using the oscillation frequency detector (OFD) and the reference clock input (IHOSC2), the frequency of the system clock (fc) or the external high-speed oscillator (f_{EHOSC}) is checked for a constant time. After the check time elapses, the clock frequency test returns the result whether the OFD detects a clock error or not.
Interrupt test	Using the timer interrupt, it is checked that the interrupt processes are executed and complete correctly with an expected frequency. When no interrupts occur or the interrupt occurs much more times than the designed ones, the corresponding error is notified.

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
CLOCKIRQ_Sample	The clock frequency test and the interrupt test are done for each constant time. It is checked that the clock frequency is in the proper range, or the interrupt occurrence frequency is correct. The result is shown with the LED lighting.	safety_CLOCK() safety_Interrupt()

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 clock frequency.
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. Clock Frequency Test

Using the oscillation frequency detector (OFD) and the reference clock (IHOSC2), the clock frequency of the system clock (fc) or the external high-speed oscillator (f_{EHOSC}) is checked for a constant time (100 ms).
After the check time elapses, the OFD returns the result whether a clock error has occurred or not.

The tested clock can be selected from any of fc clock or f_{EHOSC} clock.
When the reference clock (IHOSC2) is invalid, it is validated automatically during the test. And after the test completes, it is set to the original state (valid or invalid). In the case of the TMPM4K Group(1), the reference clock frequency is 10 MHz. The calculation example for the “Maximum Detection Frequency (high_clock)” and “Minimum Detection Frequency (low_clock)” values using the reference clock frequency is shown in “Oscillation Frequency Detection” of Reference Manual.

Source file: safety_clock.c
Header file: safety_clock.h
Used library: txz_cg.c/.h, txz_gpio.c/.h, txz_ha.c/.h, and txz_t32a.c/.h

Function name	
bool safety_clock(int clock_id, int high_clock, int low_clock)	
Input parameter	
int clock_id	Selection of the tested clock. 0: fc 1: f_{EHOSC} Any other value is considered as an error.
int high_clock	Maximum Detection Frequency setting value (12 bit) A value of less than 0x1000 is considered as an error.
int low_clock	Minimum Detection Frequency setting value (12 bit) The value of 0x1000 or more is considered as an error.
Output parameter	
None.	-
Return value	
bool	Test result (true: success, false: failure)

* 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.

Note: This test uses the SysTick interrupt to measure the test time. “Implementation of SysTick Timer Interrupt” in this document should be referred, and the hal_get_tick() function supported by SysTick interrupt should be implemented.

3.2. Interrupt Test

This is a test of the interrupt whose factor is the timer. It is considered as an error that no interrupts, much less interrupts than an expected count, or too many interrupts occur.

The measurement of the test time is done using the busy loop. Using the loop counter value of the time result which has been measured, the interrupt which occurs every 100 µs is counted for around 100-ms. In the normal case, around 1000 interrupts occur. When the interrupt count is in the range of 500 to 2000, it should be OK. Otherwise, it is considered as an error. It is also an error that no interrupts occur.

This test is supposed to be done before the set-up of the first interrupt process at the start-up of the system.

Source file: safety_interrupt.c
Header file: safety_interrupt.h
Used library: txz_cg.c/.h, txz_gpio.c/.h, txz_hal.c/.h, and txz_t32a.c/.h

Function name	
bool safety_Interrupt(int timer_id, uint32_t *result_word)	
Input parameter	
int timer_id	Channel number of the used T32A timer (0 to 5).
Output parameter	
uint32_t *result_word	Memory buffer which stores the result. bit0: Timer interrupt has not occurred. bit1: The count of the timer interrupt is too small. bit2: The count of the timer interrupt is too large.
Return value	
bool	Test result (true: success, false: failure)

* 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.

Note: When this library function is used, the implementation of the call of safety_interrupt_irq_handler() is necessary in the appropriate interrupt handler for the T32A timer channel which is used for the test.

The following table shows the implementation of the interrupts for the T32A timer in the current start-up code and bsp.c.

T32A timer channel	Used interrupt number	Current implementation in bsp.c
0	INTT32A0AC_IRQn (= 47)	irq_timer (BSP_TIMER_1MS)
1	INTT32A1AC_IRQn (= 53)	None.
2	INTT32A2AC_IRQn (= 59)	irq_timer (BSP_TIMER_1S);
3	INTT32A3AC_IRQn (= 65)	None.
4	INTT32A4AC_IRQn (= 71)	None.
5	INTT32A5AC_IRQn (= 77)	None.

In the table, the channel0 and channel2 implement the forward function of the timer interrupt request in bsp.c.
An example of the implementation in main.c is as follows;
When the timer ID = 0, BSP_TIMER_1MS is used. And the ID is 2, BSP_TIMER_1S is used.

Implementation in main.c (In the case of Timer channel (ID) = 2)

```
// #define INTERRUPT_TIMER_ID           BSP_TIMER_1MS     // = 0
#define INTERRUPT_TIMER_ID           BSP_TIMER_1S      // = 1

void irq_timer(BSPTimer timer)
{
      // timer IRQ ID (any number)
      //assigned by "bsp.c" or other values not used in "bsp.c"
      switch (timer)
      {
      case INTERRUPT_TIMER_ID:
            safety_interrupt_irq_handler();
            break;
      default:
            /* no processing */
            break;
      }
}
```

For the channels 1, 3, 4, and 5, the call of safety_interrupt_irq_handler() should be added to the empty interrupt handler in bsp.c. For example, when the channel3 is used, the following modification should be done.

Modification in Project\Examples\BSP\TMPM4K4A_eval\src\bsp.c

```
void INTT32A3AC_IRQHandler(void)
{
          safety_interrupt_irq_handler();    // T32a CH 3
}
```

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: Interrupt test fails.
LED3 (PJ4) lighting: Clock frequency test fails.

4. Implementation of SysTick Timer Interrupt

This library functions safety_CLOCK() 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 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
GPIO	txz_gpio.c
SysTick interrupt	txz_hal.c
Clock generator	txz_cg.c
Timer	txz_t32a.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	2019-08-28	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.**