

クロック周波数/割り込み用 自己診断プログラム アプリケーションノート

概要

このアプリケーションノートは、クロック周波数の異常の検知と、割り込みが正常に働くかを確認する自己診断プログラムを説明した資料です。

本資料で説明されている自己診断プログラムは、クロック周波数と割り込みを確認するためのライブラリです。

TXZ シリーズのペリフェラルドライバが同梱されたサンプルプログラムに対応しています。
ライブラリはサンプルプログラムに上書きして使用します。

目次

概要	1
目次	2
1. はじめに.....	4
2. 自己診断テストライブラリ概要	5
2.1. 自己診断サンプルプロジェクト	5
3. 自己診断テストライブラリ詳細	6
3.1. クロック周波数テスト	6
3.2. 割り込みテスト	7
4. SysTick タイマ割り込みの実装	9
5. 使用ドライバー一覧.....	10
6. 参考資料.....	11
7. 改訂履歴.....	12
製品取り扱い上のお願い.....	13

Arm、Cortex および Keil は、Arm Limited（またはその子会社）の米国およびその他の国における登録商標です。

この資料に記載されている社名・商品名・サービス名などは、それぞれ各社が商標として使用している場合があります。

1. はじめに

本アプリケーションノートは、クロック周波数と割り込みを使用した自己診断プログラムの説明資料です。

本ライブラリコードは、公開中のサンプルプログラムに追加(上書き)して使用します。

本資料の説明は、TMPM4K グループ(1)を使用した例です。
各製品への展開は、製品仕様に合わせて変更をお願いします。

本サンプルプログラムは、「自己診断プログラムアプリケーションノート 基本設定」の動作確認環境の条件、TMPM4KxA v1.0.0 のサンプルプログラム、2019年2月時点のリファレンスマニュアルを使用して開発/評価を実施しています。

2. 自己診断テストライブラリ概要

本自己診断テストライブラリは、Evaluation Board 上で動作確認を行っています。

以下のセルフテスト機能を提供します。

名称	テスト内容
クロック周波数テスト	周波数検知回路(OFD)と基準クロック入力(IHOSC2)を用いて、システムクロック(fc)あるいは外部高速発振器(f _{EHOSC})のクロック周波数が異常かどうかを一定時間チェックします。 チェック時間経過後、OFD でクロック異常が発生したかどうかを返します。
割り込みテスト	タイマ割り込みを利用し、一定時間経過内に期待される回数の割り込み処理が発生し、正常に割り込み処理が終了することを確認します。 割り込みが一度も発生しなかったり、設計よりも非常に多くの割り込みが発生したりした場合にはエラーとします。

2.1. 自己診断サンプルプロジェクト

Project¥Examples¥Safety フォルダ以下には、以下の自己診断ライブラリ用サンプルプロジェクトが配置されます。

プロジェクト名	動作概要	利用する自己診断ライブラリ関数
CLOCKIRQ_Sample	クロック周波数テストと割り込みテストをそれぞれ一定時間行い、クロック周波数範囲が適切か、また割り込み発生回数適切かを判断し LED 表示します。	safety_CLOCK() safety_Interrupt()

3. 自己診断テストライブラリ詳細

自己診断テストライブラリ機能の詳細を記載します。
本サンプルプログラムは、クロック周波数用自己診断プログラムです。
以下のテストは、TPPM4K グループ(1)を使用した場合の設定例です。

自己診断ライブラリのソースコード (.c ファイル) は Libraries¥Safety¥src フォルダに、ヘッダファイル (.h ファイル) は Libraries¥Safety¥inc に存在します。

3.1. クロック周波数テスト

周波数検知回路 (OFD) と基準クロック (IHOSC2) を用いて、システムクロック (fc) または外部高速発振器 (fEHOSC) のクロック周波数が異常かどうかを一定時間 (100ms) チェックします。

テスト対象のクロックは fc または fEHOSC のいずれから選択可能です。

基準クロック (IHOSC2) が無効の場合、テスト中に自動的に有効にされ、テスト後はもとの設定 (有効または無効) に戻します。TPPM4K グループ(1)の場合、基準クロックの周波数は 10MHz です。この値を用いて上限異常検知 (high_clock) および下限異常検知 (low_clock) の値を計算する例が、リファレンスマニュアルの「周波数検知回路 (OFD)」に記載されています。

ソースファイル: safety_clock.c

ヘッダファイル: safety_clock.h

使用ライブラリ: txz_cg.c/.h, txz_gpio.c/.h, txz_ha.c/.h, txz_t32a.c/.h

関数名	
bool safety_clock(int clock_id, int high_clock, int low_clock)	
入力パラメータ	
int clock_id	テストするクロック入力の種別 0 : fc 1 : fEHOSC これ以外の値の場合エラーとします。
int high_clock	上限異常検知の設定値 (12bit) 0x1000 以上の値の場合はエラーとします。
int low_clock	下限異常検知の設定値(12bit) 0x1000 以上の値の場合はエラーとします。
出力パラメータ	
なし	-
戻り値	
bool	テスト結果 (true: 成功、false: 失敗)

※戻り値が数秒間確認できない場合は、上記関数以外の要因によりテストが正常に機能していないことが考えられるのでテスト失敗として処理してください。

ただし、ターミナル I/O 出力表示を使用しているときに表示終了まで数秒かかりますので、テスト失敗判断は表示を見ながら判定してください。

注意: このテストではテスト時間を計測するために SysTick 割り込みを使用します。本ドキュメントの「SysTick タイマ割り込みの実装」を参照し、SysTick 割り込みによる hal_get_tick() 関数のサポートを実装してください。

3.2. 割り込みテスト

タイマを割り込み源とした割り込みテストを行います。テスト中に一度も割り込みがかからないか、想定したより割り込み回数が少な過ぎる、または多過ぎるときにエラーとします。

テスト時間の計測はビジータイムで行います。あらかじめ時間計測した結果のループカウンタ値を用いて、約 100ms の間 100 マイクロ秒間隔で発生する割り込み回数をカウントします。標準では約 1000 回の割り込みが発生するはずですが、500 回以上 2000 回以下の割り込み発生回数が検出された場合 OK とし、これを外れるか、または 1 回も発生しなかった場合はエラーとします。

本テストはシステム起動時に最初の割り込み処理のセットアップ前に実行されることを想定しています。

ソースファイル: `safety_interrupt.c`

ヘッダファイル: `safety_interrupt.h`

使用ライブラリ: `txz_cg.c/h`, `txz_gpio.c/h`, `txz_hal.c/h`, `txz_t32a.c/h`

関数名	
<code>bool safety_Interrupt(int timer_id, uint32_t *result_word)</code>	
入力パラメータ	
<code>int timer_id</code>	使用する T32A タイマチャンネル番号 (0-5)
出力パラメータ	
<code>uint32_t *result_word</code>	結果を格納するメモリバッファ bit0: タイマ割り込み発生しない bit1: タイマ割り込みの回数が少な過ぎる bit2: タイマ割り込みの回数が多過ぎる
戻り値	
<code>bool</code>	テスト結果 (true: 成功、false: 失敗)

※戻り値が数秒間確認できない場合は、上記関数以外の要因によりテストが正常に機能していないことが考えられるのでテスト失敗として処理してください。

ただし、ターミナル I/O 出力表示を使用しているときに表示終了まで数秒かかりますので、テスト失敗判断は表示を見ながら判定してください。

注意: このライブラリ関数を利用する際には、テストに使用する T32A タイマチャンネルに対して適切な割り込みハンドラから `safety_interrupt_irq_handler()` を呼び出すように実装する必要があります。

現在のスタートアップコードおよび `bsp.c` で T32A タイマがどのように割り込み実装されているかは以下のようにまとめられます。

T32A タイマチャンネル	使用する割り込み番号	bsp.c での既存の実装
0	<code>INTT32A0AC_IRQn (= 47)</code>	<code>irq_timer (BSP_TIMER_1MS)</code>
1	<code>INTT32A1AC_IRQn (= 53)</code>	なし
2	<code>INTT32A2AC_IRQn (= 59)</code>	<code>irq_timer (BSP_TIMER_1S);</code>
3	<code>INTT32A3AC_IRQn (= 65)</code>	なし
4	<code>INTT32A4AC_IRQn (= 71)</code>	なし
5	<code>INTT32A5AC_IRQn (= 77)</code>	なし

上記の表のうち、チャンネル 0 と 2 は `bsp.c` 内にタイマ割り込み要求をフォワードする実装が用意されているため、(例えば) `main.c` 内に以下のように実装します。

タイマ ID == 0 の場合、`BSP_TIMER_1MS` を使用。2 の場合は `BSP_TIMER_1S` を使用します。

main.c で以下を実装 (タイマチャンネル (ID) == 2 を使用する例)

```
// #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;
    }
}
```

チャンネル 1/3/4/5 の場合は、bsp.c 内の空の割り込みハンドラに safety_interrupt_irq_handler() 呼び出しを追加します。例えばチャンネル 3 を利用する場合は以下のように変更します。

Project¥Examples¥BSP¥TMPM4K4A_eval¥src¥bsp.c 内を変更

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

本テスト開発で使用了した評価ボードでは、全テストが終了したらテスト結果を LED で表示します。

LED1 (PJ0) 点灯: 全テスト成功

LED2 (PJ2) 点灯: 割り込みテスト失敗

LED3 (PJ4) 点灯: クロック周波数テスト失敗

4. SysTick タイマ割り込みの実装

ライブラリ関数 `safety_CLOCK0` は時間計測を行うために、`txz_hal.c` および SysTick タイマを利用します。これらの関数では 呼び出し前に SysTick 割り込みによる時間計測が正しく動作するようにプログラム実装する必要があります。

※サンプルプログラムでは本設定は実装されています。

SysTick タイマを利用するには以下の 2 つの実装を行います。

1. タイマ割り込みハンドラの設定

`main.c` 内で以下のように割り込みハンドラを設定します。

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

2. SysTick 割り込みの開始処理

`application_initialize0` 関数、または `main0` 関数内の適切な箇所で、以下のように割り込み開始します。

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

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

    // SysTick IRQ started
}
```

以上を実装することにより、`txz_hal.c` で提供される `hal_get_tick0` 関数が 1ms 単位のカウント値を取得できるようになります。

5. 使用ドライバー一覧

このテストライブラリではTMPM4KxA v1.0.0バージョンのドライバおよびプロジェクト内のコードを利用しています。

CMSIS ライブラリ

カテゴリ	ソースファイル名
スタートアップ	startup_TMPM4K4A.s
システム(クロック設定など)	system_TMPM4KxA.c

Periph_driver

カテゴリ	ソースファイル名
GPIO	txz_gpio.c
SysTick 割り込み	txz_hal.c
クロックジェネレータ	txz_cg.c
タイマ	txz_t32a.c

Project Examples 内

カテゴリ	ソースファイル名
BSP (評価ボードサポート)	bsp.c
LED 出力	bsp_led.c

6. 参考資料

以下の資料を使用して開発を実施しています。

- ・ データシート
- ・ リファレンスマニュアル
- ・ 自己診断プログラムアプリケーションノート基本設定
- ・ ARM® Cortex®-M4 Processor technical Reference Manual
- ・ ARMv7-M Architecture Reference Manual

7. 改訂履歴

Revision	Date	Description
1.0	2019-08-29	初版

製品取り扱い上のお願い

株式会社東芝およびその子会社ならびに関係会社を以下「当社」といいます。

本資料に掲載されているハードウェア、ソフトウェアおよびシステムを以下「本製品」といいます。

- 本製品に関する情報等、本資料の掲載内容は、技術の進歩などにより予告なしに変更されることがあります。
- 文書による当社の事前の承諾なしに本資料の転載複製を禁じます。また、文書による当社の事前の承諾を得て本資料を転載複製する場合でも、記載内容に一切変更を加えたり、削除したりしないでください。
- 当社は品質、信頼性の向上に努めていますが、半導体・ストレージ製品は一般に誤作動または故障する場合があります。本製品をご使用頂く場合は、本製品の誤作動や故障により生命・身体・財産が侵害されることのないように、お客様の責任において、お客様のハードウェア・ソフトウェア・システムに必要な安全設計を行うことをお願いします。なお、設計および使用に際しては、本製品に関する最新の情報（本資料、仕様書、データシート、アプリケーションノート、半導体信頼性ハンドブックなど）および本製品が使用される機器の取扱説明書、操作説明書などをご確認の上、これに従ってください。また、上記資料などに記載の製品データ、図、表などに示す技術的な内容、プログラム、アルゴリズムその他応用回路例などの情報を使用する場合は、お客様の製品単独およびシステム全体で十分に評価し、お客様の責任において適用可否を判断してください。
- 本製品は、特別に高い品質・信頼性が要求され、またはその故障や誤作動が生命・身体に危害を及ぼす恐れ、膨大な財産損害を引き起こす恐れ、もしくは社会に深刻な影響を及ぼす恐れのある機器（以下“特定用途”という）に使用されることは意図されていませんし、保証もされていません。特定用途には原子力関連機器、航空・宇宙機器、医療機器（ヘルスケア除く）、車載・輸送機器、列車・船舶機器、交通信号機器、燃焼・爆発制御機器、各種安全関連機器、昇降機器、発電関連機器などが含まれますが、本資料に個別に記載する用途は除きます。特定用途に使用された場合には、当社は一切の責任を負いません。なお、詳細は当社営業窓口まで、または当社 Web サイトのお問い合わせフォームからお問い合わせください。
- 本製品を分解、解析、リバースエンジニアリング、改造、改変、翻案、複製等しないでください。
- 本製品を、国内外の法令、規則及び命令により、製造、使用、販売を禁止されている製品に使用することはできません。
- 本資料に掲載してある技術情報は、製品の代表的動作・応用を説明するためのもので、その使用に際して当社及び第三者の知的財産権その他の権利に対する保証または実施権の許諾を行うものではありません。
- 別途、書面による契約またはお客様と当社が合意した仕様書がない限り、当社は、本製品および技術情報に関して、明示的にも黙示的にも一切の保証（機能動作の保証、商品性の保証、特定目的への合致の保証、情報の正確性の保証、第三者の権利の非侵害保証を含むがこれに限らない。）をしておりません。
- 本製品、または本資料に掲載されている技術情報を、大量破壊兵器の開発等の目的、軍事利用の目的、あるいはその他軍事用途の目的で使用しないでください。また、輸出に際しては、「外国為替及び外国貿易法」、「米国輸出管理規則」等、適用ある輸出関連法令を遵守し、それらの定めるところにより必要な手続を行ってください。
- 本製品の RoHS 適合性など、詳細につきましては製品個別に必ず当社営業窓口までお問い合わせください。本製品のご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用ある環境関連法令を十分調査の上、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は一切の責任を負いかねます。