UART 用
自己診断プログラム
アプリケーションノート

概要

本アプリケーションノートは、UART 用の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、UART のシリアル通信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

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、UART 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)をベースに記載しております。
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
シリアル通信テスト	UART ポートの内部ループバックを設定し、送信データと受信データの一致を確認します。

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
UART_Sample	UART を利用してループバックテストを行い、結果を LED 表示します。	safety_UART_loopback()

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、UART 用自己診断プログラムです。
以下のテストは、TMPM4K 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. シリアル通信テスト

UART ポートの内部ループバックを設定し、送信データと受信データの一致を確認します。

このライブラリ関数では、使用する UART ポートを ID で区別し、また割り込みハンドラの呼び出しを同じ ID で行います。使用スピードは 115200bps 固定です。SysTick 割り込みを利用してデータ長から計算される時間からタイムアウトを検出します。

safety_UART_loopback 関数では割り込みを使用します。

ソースファイル: safety_uart.c
ヘッダファイル: safety_uart.h
使用ライブラリ: txz_cg.c/.h, txz_gpio.c/.h, txz_hal.c/.h, txz_uart.c/.h

関数名	
bool safety_UART_loopback(SafetyUartID id, uint32_t start, uint32_t length)	
入力パラメータ	
SafetyUartID id	使用する UART を示す enum 値は下記のいずれかです。 SAFETY_UART0_ID = 100 SAFETY_UART1_ID = 101 SAFETY_UART2_ID = 102 SAFETY_UART3_ID = 103
uint32_t start	使用するデータの先頭アドレス
uint32_t length	使用するデータの長さ(バイト単位)
出力パラメータ	
なし	-
戻り値	
bool	結果(true: 成功、false: 失敗=パラメータエラー、結果相違、タイムアウトなど)

※本サンプルプログラムは、「SAFETY_UART2_ID」を使用しています。
※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。

本テスト開発で使用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: テスト失敗

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 UART チャネルに対して適切なポート初期化と割り込みハンドラの実装を行う必要があります。このテストではテスト対象の UART を実際の通信にも使う場合を想定しており、safety_UART_loopback 関数呼び出し中のみ割り込みフックをかけるための safety_UART_hook_rx() / safety_UART_hook_tx() / safety_UART_hook_err() の 3 つの割り込みハンドラを実装します。

1. 初期化関数の実装

UART ポートごとの bsp.c の初期化状況は以下のようになります。UART0 以外を使用する場合はポートの初期化を行う必要があります。

UART ポート	bsp.c でのポート初期化
UART0	あり
UART1	なし
UART2	なし
UART3	なし

初期化は以下のように行います。(以下は UART1 の例)

```
// 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. 割り込みハンドラ(フック)の実装

送信/受信/エラーの 3 種類の割り込みに対してそれぞれフックを実装します。

UART ポート	使用する割り込み番号	bsp.c での既存の実装
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)

いずれの UART チャネルも bsp.c で main.c 内の関数呼び出しに変換されています。そこで、呼び出し先の割り込みハンドラに以下のようにフック呼び出しを実装します。このとき、safety_UART_loopback へ設定したのと同じ ID 引数を渡します。

Project¥Examples¥Safety¥UART_Sample¥src¥main.c より

```
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
}
```

※同様に TX 割り込みおよび ERR 割り込みも実装します。

さらに、safety_UART_loopback ではタイムアウトを検出するため SysTick 割り込みを利用しています。「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。

4. SysTick タイマ割り込みの実装

ライブラリ関数 safety_UART_loopback ()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_initialize()関数、または main()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_tick() 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
UART	txz_uart.c
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

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エア、ソフトウエ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、お客様のハードウエア・ソフトウエア・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。