

LPC812M101 開発ボード マニュアル

株式会社日昇テクノロジー

http://www.csun.co.jp

info@csun.co.jp

作成・更新日 2013/09/22



copyright@2013



• 修正履歴

NO	バージョ	修正内容	修正日
	ン		
1	Ver1.0	新規作成	2013/09/22

※ この文書の情報は、文書を改善するため、事前の通知なく変更されることがあります。 最新版は弊社ホームページからご参照ください。「http://www.csun.co.jp」

※(株)日昇テクノロジーの書面による許可のない複製は、いかなる形態においても厳重に 禁じられています。



目次

第一章	概説	4
1.1.	チップファンクション	4
1.2.	ボードファンクション	5
第二章	Hardware layout and configuration	6
2.1.	Power supply	6
2.2.	Clock source	6
2.3.	Reset source	7
2.4.	USB-COM	7
2.5.	Potentiometer	7
2.6.	LED	7
2.7.	КЕҮ	7
第三章	開発ツール Keil の応用	8
3.1.	Keil コンパイル環境	8
3.1.1.	Keil コンパイル環境構築	8
3.1.2.	エミュレータ設定	8
3.2.	シリアルポートでプログラムダウンロード	10
第四章 サ	└ンプルソース説明	12
4.1.	lpcopen_periph_8xx.uvmpw プロジェクト管理ファイルのプログラム説明	12
4.1.1.	¥periph_blinky	
4.1.2.	¥periph_acmp	15
4.1.3.	¥periph_bod	15
4.1.4.	¥periph_clkout	16
4.1.5.	¥periph_crc	17
4.1.6.	¥periph_i2c	17
4.1.7.	¥periph_mrt	
4.1.8.	¥periph_spi	18
4.1.9.	¥periph_uart	19
4.1.10	0. ¥periph_uart_rom	19
4.1.1	1. ¥periph_wkt	20
4.1.12	2. ¥periph_wwdt	20
4.2.	lpcopen_freertos_8xx.uvmpw プロジェクト管理ファイルのプログラム説明	21
4.2.1.	¥freertos_blinky	21



第一章 概説

1.1. チップファンクション

ARM Cortex-MO+ processor

- running at frequencies of up to 30 MHz
- Nested Vectored Interrupt Controller (NVIC)
- System tick timer.
- Serial Wire Debug (SWD) and JTAG boundary scan modes supported
- Micro Trace Buffer (MTB) supported.

Memory:

- Up to 16 kB on-chip flash (ISP and IAP via on-chip bootloader software)
- Up to 4 kB SRAM

- ROM API support: Boot loader, USART drivers, I2C drivers, Power profiles Digital peripherals:

- High-speed GPIO interface with up to 18 General-Purpose I/O (GPIO) pins
- GPIO interrupt generation capability
- Switch matrix for flexible configuration of each $\mathrm{I}/\mathrm{0}$ pin function
- State Configurable Timer (SCT)
- Multiple-channel multi-rate timer (MRT)
- Self Wake-up Timer (WKT)
- CRC engine.
- Windowed Watchdog timer (WWDT)

Analog peripherals:

- Comparator with external voltage reference

Serial interfaces:

- 3 USART interfaces
- 2 SPI controllers
- I2C-bus interface
- Clock generation:
- 12 MHz internal RC Oscillator (IRC)
- Crystal Oscillator (SysOsc) with operating range of 1 MHz to 25 MHz
- Programmable watchdog oscillator with a frequency range of 9.4 kHz to 2.3 MHz
- 10 kHz low-power oscillator for the WKT
- PLL allows CPU operation up to the maximum CPU rate
- Clock output function with divider that can reflect various clocks

Power control:

- Integrated PMU (Power Management Unit)
- Reduced power modes: Sleep mode, Deep-sleep mode, Power-down mode, Deep power-down mode
- Power-On Reset (POR)
- Brownout detect
- Unique device serial number for identification

Single power supply



1.2. ボードファンクション

USB シリアル変換機能搭載、FlashMagic を介しプログラムをダウンロードサポート
RGB LED
1 つファンクションキー
1 つ変調抵抗
全ての I/0 を引き出し



低価格、高品質が不可能? 日昇テクノロジーなら可能にする



2.1. Power supply

開発ボードは MiniUSB インタフェースで給電する、同時に USB 仮想シリアルポートとしても使用する。

2.2. Clock source

開発ボードは12MHz外部水晶振動器をプロセッサのクロックリソースとする。

2.3. Reset source

開発ボードはローレベルリセット、リセットするには2つの方法がある:

二 ボードのS2ボタンを押す

□ JP1−P4

2.4. USB-COM

開発ボードは USB シリアルポート変換チップ CP2102 を使用し、プロセッサの USARTO と接続し、シリアル ポート機能を実現する。

2.5. Potentiometer

開発ボードに 10K の変調抵抗があり、PIO0_0と接続し、コンパレータを行う実験に使用する。

2.6. LED

開発ボードに1つの RGB 三色 LED があり、PIO0_7、PIO0_17、PIO0_16 と接続し、ローレベルで点灯する。

2.7. KEY

開発ボードには2つのキーがあり、RST はリセットキー、ISP はプロセッサが ISP モードに切り替えキー で、通常キーにも使用できる。



第三章 開発ツール Keil の応用

3.1. Keil コンパイル環境

3.1.1. Keil コンパイル環境構築

Keil 社の HP(http://www.keil.com/)から最新版(無償評価版)がダウンロードできます。 完全機能を使用するには License が別途で購入する必要がある。

3.1.2. エミュレータ設定

本開発ボードにエミュレータインタフェースがないため、エミュレータを使用するにはピン配列変換ケー ブルで下記の通りに接続する必要:

エミュレータの 20 ピンソケット	開発ボード JP1、JP2 ピン	プロセッサのピン名
7	JP1-7	PIO0_2_SWDIO
9	JP1-6	PIO0_3_SWCLK
15	JP1-4	PIO0_5_RESET
1 or 2	JP2-6	+3.3V
4、6、8、10、12、16、18、20 中	JP2-5	GND
の1ピン		



• * Options for Target*



● □下記のウィンドウ:



低価格、高品質が不可能?

日昇テクノロジーなら可能にする

Options for Target 'iflash_nxp_xpresso_812'						
Device Target Output Listing User C/C++ Asm Linker Debug Utilities						
○ Use Simulator Settings	● Use: ULINK2/ME Cortex Debugger ▼ Settings					
Limit Speed to Real-Time						
Load Application at Startup Run to main()	Load Application at Startup					
Initialization File:	Initialization File:					
Bestore Debug Session Settings	Bestore Debug Session Settings					
✓ ✓ Toolbox	Breakpoints Toolbox					
✓ Watch Windows & Performance Analyzer	Vatch Windows					
Memory Display	Memory Display					
CDU DU L. Deremeter	Driver DLL: Decemptor					
SARMCM3.DLL	SARMCM3.DLL					
1 1	1					
Dialog DI I Parameter	Dialog DI I Parameter					
DARMCM1.DLL -pCM0+	TARMCM1.DLL -pCM0+					
· · · · ·						
ОК Са	ncel Defaults Help					

* Utilities 選択、* Use Target Driver for Flash Programming* でエミュレータタイプ選択、ULINK2 を使用する場合、* ULINK2/Me Cortex Debugger* 選定; JLINK V8 を使用する場合、* J-LINK/J-Trace Cortex* 選定。.

Device Target Out	out Listing User	C/C++ As	m Linker C	Debug Utilities	
Configure Flash Menu Con	mmand Flash Programming		Configure Flash	Menu Command Driver for Flash Program	nming
C Use Externa C Use Externa C Use Externa C Use Externa Command: NULink C Signum S J-LINK / ULINK Pr Outink D SiLabs U ST-Link C CMSIS-D	ME Cortex Debugger ME Cortex Debugger CDI Systems JTAGjet I-Trace Cortex Deprecated Version) o Cortex Debugger Debugger DA Debugger Debugger AP Debugger	•	Init File: O Use Externa Command: Arguments:	J-LINK / J-1 race Cortex ULINK2/ME Cortex Deb Stellaris ICDI Signum Systems JTAG J-LINK / J-Trace Cortex ST-Link (Deprecated V ULINK Pro Cortex Debu NULink Debugger SiLabs UDA Debugger ST-Link Debugger CMSIS-DAP Debugger	ugger jet ersion) igger

● ` Debug`選択、Use リストでエミュレータ選択: ULINK2 を使用する場合、` ULINK2/Me Cortex
 Debugger`選定; JLINK V8 を使用する場合、` J-LINK/J-Trace Cortex`選定。設定完了後、エミュレータでプログラムをエミュレータできる。

User C/C++ Asm Linker Debug Utilities

	ULINK2/ME Cortex Debugger	•		J-LINK / J-Trace Cortex	•
	ULINK2/ME Cortex Debugger			RDI Interface Driver	
	RDI Interface Driver			Altera Blaster Cortex Debugger	
I Lood /	Altera Blaster Cortex Debugger			Stellaris ICDI	
l€ Loau	Stellaris ICDI			Signum Systems JTAGjet	
Initializatio	Signum Systems JTAGjet		Initializatio	J-LINK / J-Trace Cortex	
	J-LINK / J-Trace Cortex	-		ST-Link (Deprecated Version)	=
	ST-Link (Deprecated Version)			ULINK Pro Cortex Debugger	
	ULINK Pro Cortex Debugger			NULink Debugger	
Restore	NULink Debugger		Restore	SiLabs UDA Debugger	
R Bre	SiLabs UDA Debugger			ST-Link Debugger	
	ST-Link Debugger	$\overline{\mathbf{v}}$		CMSIS-DAP Debugger	$\overline{\mathbf{v}}$
- M/-		-		omolo bra bebuggei	_



低価格、高品質が不可能? 日昇テクノロジーなら可能にする

● Ďebug で Šettings をクリック、次のウィンドウで Debug →Port SWを選定
Device Target Output Listing User C/C++ Asm Linker Debug Utilities
○ Use Simulator Settings Image: ULINK2/ME Cortex Debugger Settings □ Limit Speed to Real-Time Image: ULINK2/ME Cortex Debugger Image: ULINK2/ME Cortex Debugger Image: ULINK2/ME Cortex Debugger
Cortex-M Target Driver Setup
Debug Trace Flash Download
ULINK USB - JTAG/SW Adapter
ULINK Version: ULINK2
Device Family: Cortex-M
Firmware Version: V2.01
SWJ Port SW V
Max Clock: IMHz
● `Output`→`Creat HEX File`有効、FlashMagic を介し HEX ファイルをダウンロードできる
Device Target Output Listing User C/C++ Asm
Select Folder for Objects
Create Executable: .\keil_output\iflash_nxp_xpres
✓ Debug Information
Create HEX File
✓ Browse Information
C Create Library: .\keil_output\iflash_nxp_xpresso_i

3.2. シリアルポートでプログラムダウンロード

- Tools ディレクトリにある FlashMagic. exe をインストール、仮想シリアルポートドライブ CP210x_VCP_Win_XP_ S2K3_Vista_7. exe をインストール。
- MiniUSB を介し、PC と開発ボードを接続、デバイスマネージャを開き、仮想シリアルポートの割り当てるポートを確認する。PC により、仮想ポートの COM は異なる。
 例 COM7:



- ISP を押しながら、RST キーを押して開発ボードをリセット、プロセッサが ISP モードに入る。
- FlashMagic を実行、Select でプロセッサを選定する。



低価格、高品質が不可能? 日昇テクノロジーなら可能にする



COM Port で PC と開発ボード通信 COM を選定(本実験は COM7)、ボーレート 19200~115200、Browse で HEX のプログラムを選定、Step 4 の^{*} Verify after program^{*} を有効、Start をクリック、プログラ ムを Flash にダウンロード。完了後、開発ボードの RST でリセット、プログラム実行する。

🐡 Flash Magic - NON PRODUCTION USE ON	LY 🗕 🗖 🗙						
File ISP Options Tools Help							
ا الله الله الله الله الله الله الله ال							
Step 1 - Communications	Step 2 - Erase						
Select LPC812M101FD20 1	Erase block 0 (0x000000-0x0003FF)						
Flash Bank:	Erase block 2 (0x000800-0x000BFF)						
COM Port: COM 7 2	Erase block 3 (UXUUUCUU-UXUUUFFF) Erase block 4 (0x001000-0x0013FF)						
Baud Rate: 115200 3	Erase block 5 (0x001400-0x0017FF)						
Interface: None (ISP) 🔹	Erase blocks used by Hex File						
Oscillator (MHz):							
Step 2. Hey File							
Hex File: E:\Mywork\ARM\LPC812\Code\LPCC	pen_v1.03\applications\lpct Browse						
Modified.生期八,五月11,2013,11.24	5						
Step 4 - Options	Step 5 - Start!						
Verify after programming 6	Start 7						
Gen block checksums							
Execute							
✓ Activate Flash Bank							
Technical on-line articles about 8051 and XA programming							
www.esacademy.com/faq/docs							
	1						



第四章 サンプルソース説明

¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph ¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥freertos

2 つのディレクトリはプロセッサの外部ドライブ例及び FreeRTOS の LED フラシュ例である。複数プロジェクトを管理するため、¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥kei1_uvision_projects ディレクトリに lpcopen_freertos_8xx.uvmpw 及び lpcopen_periph_8xx.uvmpw のプロジェクト管理ファイルを追加する。 この二つのプロジェクトをオーペンして全てのプロジェクトをコンパイル出来る。

vith • New folder		
Name	Date modified	Type
L periph_acmp	2013/5/14 22:07	File folder
📕 periph_blinky	2013/5/16 10:37	File folde
k periph_bod	2013/5/14 22:07	File folde
k periph_clkout	2013/5/14 22:07	File folde
L periph_crc	2013/5/14 22:07	File folde
k periph_i2c	2013/5/14 22:07	File folde
k periph_mrt	2013/5/14 22:07	File folde
k periph_sct	2013/5/14 22:07	File folde
L periph_spi	2013/5/14 22:07	File folde
k periph_uart	2013/5/14 22:07	File folde
L periph_uart_rom	2013/5/14 22:07	File folde
👢 periph_wkt	2013/5/14 22:07	File folde
L periph_wwdt	2013/5/14 22:07	File folde
-16 • Code • LPCOpen_v1.03 • app	lications + Ipc8xx + examp	ples I freerto
ith ▼ New folder		
Name	Date modified	Туре
👢 freertos_blinky	2013/5/11 11:04	File folder
L startup_code	2013/5/11 10:35	File folder

4.1. lpcopen_periph_8xx.uvmpw プロジェクト管理ファイルのプロ グラム説明

プロジェクト管理ファイルは¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥keil_uvision_projectsディレクトリにあり、lpcopen_periph_8xx はプロジェクト管理ファイル。

下記図の通り、WorkSpace ウィンドウで管理プロジェクトを確認できる、lib_lpc_board_8xx プロジェクトファイルは board_nxp_epresso_812. lib ファイルを生成し、lib ファイルは他のプロジェクトファイルが 必要なディバイスドライブの API 関数を提供する。なので他のプロジェクトを独立コンパイルするには、board_nxp_epresso_812. lib ファイルを確認する必要がある。



低価格、高品質が不可能? 日昇テクノロジーなら可能にする

Project	무 📧
🖃 🖷 WorkSpace	
lib_lpc_boards_8xx	
🖻 🚔 periph_blinky	
🗄 🗟 iflash_nxp_xpresso_812	
⊞ (src	
⊨	
🔤 🔝 board_nxp_xpres	sso_812.lib
🗄 💼 periph_acmp	
🖻 💼 periph_bod	
periph_clkout	
🗄 💼 periph_crc	
🔄 💼 periph_i2c	
🗄 💼 periph_mrt	
🗄 💼 periph_sct	
🗄 💼 periph_spi	
🖮 🧰 periph_uart	
periph_uart_rom	
🗉 💼 periph_wkt	
🗄 💼 periph_wwdt	

 lpcopen_periph_8xx.uvmpw プロジェクト管理ファイルにある全てのプロジェクトファイルを一回で 全部コンパイル出来、また単独コンパイルも出来る。Project->Batch Build でコンパイルプロジェク トを管理する。

Proj	ject	Flash	Debug	Peripherals	Tools	SVCS	Window	
	New μVision Project New Multi-Project Workspace Open Project Close Project							
	Exp Mar	ort nage						
	Select Device for Target 'nxp_xpresso_812' Remove Item							
Ň	Options							
	Clean 'lib_lpc_boards_8xx(nxp_xpresso_812)'							
	Build 'lib_lpc_boards_8xx(nxp_xpresso_812)'							
	Rebuild 'lib_lpc_boards_8xx(nxp_xpresso_812)'							
	Bate	ch Build						

 Deselect All、デフォルトクリア、編集プロジェクトを選定する。Build または Rebuild をクリック、 選択したファイルを同時にコンパイル出来る。



低価格、高品質が不可能? 日昇テクノロジーなら可能にする

Batch Build					
Select Project Targets:					
□ lib_lpc_boards_8xx		Build			
nxp_xpresso_812					
periph_acmp		Rebuild			
Iflash_nxp_xpresso_812		Cloan			
iflash nyn ynresso 812					
⊨ periph bod					
iflash_nxp_xpresso_812					
periph_clkout	=				
iflash_nxp_xpresso_812		Select All			
periph_crc		Selectruit			
iflash_nxp_xpresso_812		Deselect All			
iflash nyn ynresso 812					
E periph mrt					
iflash_nxp_xpresso_812					
⊨ periph_sct					
iflash_nxp_xpresso_812		Holp			
eriph_spi		Пеіћ			
iflash_nxp_xpresso_812	-	Close			
🗁 peripir_uart					

 通常は1つファイルを選択、単独コンパイルする。例 periph_blinky、右クリックし、Set as Active Project、プロジェクトファイルを青から黒に変化し、現在アクティブなプロジェクトとなる。



4.1.1. ¥periph_blinky



このプロジェクトは簡単な LED ドライブプログラム、システムタイマーを使用し、割り込みプログラムで 開発ボードの青い LED をフラシュさせる。

● 先ずは Board_Init() 関数でシステムタイマー、三つの LED 制御ピンを初期化する。次に SysTick_Config でシステムタイマーの割り込み周波数を設定する。システムタイマープログラムは Board_LED_Toggle(0)を呼び出して青い LED を値の反転動作し続ける。

```
void SysTick_Handler(void)
{
    Board_LED_Toggle(0);
}
int main(void)
{
    Board_Init();
    Board_LED_Set(0, false);
    /* Enable SysTick Timer */
```



}

株式会社日昇テクノロジー

```
SysTick_Config(SystemCoreClock / TICKRATE_HZ);
while (1) {
    __WFI();
}
return 0;
```

 生成した HEX ファイルは¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_ blinky¥kei1_output¥iflash_nxp_xpresso_812 ディレクトリに保存する。

4.1.2. ¥periph_acmp



本プロジェクトのプログラムはアナログコンパレータを実演する。プログラムは PIO0_0 をコンパレータ のプラス(+) 電圧入力に設定し、チップ内部の 0.8V バンドギャップ基準電圧は、コンパレータのマイナ ス(-) 電圧入力と設定し、PINO.15 はコンパレータの出力と設定する。変調抵抗の抵抗値を変化させて PIO0_0 の電圧を調整する。電圧 < 0.8V な場合開発ボードの青い LED 点灯、PINO.15 ピンは 0 を出力;電圧 > 0.8V な 場合、LED 消灯、PINO.15 ピンは 1 を出力。

● HEXファイルは

¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_acmp¥keil_output¥iflash_nx p_xpresso_812 ディレクトリ下に保存する。

注:プログラムを再ダウンロードする前に、変調抵抗を調整し、LEDを消灯させる必要がある。

4.1.3. ¥periph_bod

WorkSpace
 Iib_lpc_boards_8xx
 periph_blinky
 periph_acmp
 periph_bod
 Set as Active Project

低電圧検出の例である。開発ボードの電圧が低すぎな場合、BOD 割り込み生成、割り込みサービスプログラムは青い LED 値を反転動作する。

- プログラム実行後、開発ボードの USB ケーブルを抜いたら、青い LED が 1 回点灯する、開発ボードの 電源に大容量コンデンサを接続すると青い LED の点滅を確認できる。
- HEXファイルは

¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_bod¥keil_output¥iflash_nxp _xpresso_812 ディレクトリ下に保存する。



4.1.4. ¥periph_clkout



SYSCTL ドライバーを介し、プロセッサの異なるクロックリソースを通常ピンに引き出す実例である。

- プログラムは Chip_Clock_SetCLKOUTSource 関数で出力クロックを設定する。クロックリソースは下 記の通り:SYSCTL_CLKOUTSRC_IRC、SYSCTL_CLKOUTSRC_SYSOSC、SYSCTL_CLKOUTSRC_WDTOSC、 SYSCTL_CLKOUTSRC_MAINSYSCLK。
- プログラム実行後、開発ボードの青い LED が点滅する、オシロスコープで PIO0_1 ピンを測定、クロ ック出力を観察する。数秒後 LED は消灯し、PIO0_1 も出力波形を停止する



● HEXファイルは

¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_clkout¥keil_output¥iflash_ nxp_xpresso_812 ディレクトリ下に保存する。



低価格、高品質が不可能? 日昇テクノロジーなら可能にする

4.1.5. ¥periph_crc

🗆 🖷 🚾 Wo	orkSpace
÷… 🚞	lib_lpc_boards_8xx
÷	periph_blinky
÷… 🚞	periph_acmp
÷… 💼	periph_bod
÷… 💼	periph_clkout
÷… 🧰	p <u>eriph crc</u>
÷… 🧰	p Set as Active Project

チップ内部の CRC エンジンを使用し、8、16,32 ビットの CRC パリティチェックを行う実例である。 プログラムループは順序に8、16,32 ビットデータを CRC パリティチェックを行い、結果を別データ組に 保存し、奇数回数は正確動作、偶数回数は誤動作。最後に CRC パリティチェックの結果と予測結果をもう一 度 32 ビットのパリティチェックを行い、2 回の CRC パリティチェック結果を比較し、正確であれば青い LED 点灯、間違であれば LED 消灯。奇数回数は正確、偶数回数は間違い動作なのでプログラム実行後、青い LED は点滅し始める。

● HEXファイルは

¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_crc¥keil_output¥iflash_nxp _xpresso_812 ディレクトリ下に保存する。

4.1.6. ¥periph_i2c



I2C メインコントローラーとスレーブ機能を実現する例。2 つの開発ボードが必要とする。1 つの開発ボードはメインコントローラープログラムをダウンロードし、1 つの開発ボードはスレーブドライブをダウンロード。マクロ I2C_MASTER_MODE=1 メインコントローラープログラム実行、I2C_MASTER_MODE=0 スレーブプログラム実行する。

- テストする前に、2つの開発ボードの P0.10 (SDA) 同士、P0.11 (SCL) 同士を接続、2つの開発ボード の GND も接続する。1 つボードの SDA 及び SCL ピンに 4.7K のプルアップ抵抗と接続する必要。
- メインコントローラーのプログラムは先ずクエリモードで7-bit サブアドレスでスレーブディバイス プログラムの開発ボードに1バイトのデータを書き込み、読み取り動作を実行し、データを比較する。
- メインコントローラープログラムは割り込みモードで同じくデータを書き込み、読み取り、比較する。
- 最後にメインコントローラーは割り込みモードで 10-bit サブアドレス方式で書き/読み/比較する。
- 全ての動作が正しく実行すると、開発ボードの青い LED 点灯。
- HEX ファイルは ¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_i2c¥keil_output¥iflash_nxp



低価格、高品質が不可能? 日昇テクノロジーなら可能にする

_xpresso_812 ディレクトリ下に保存する。

4.1.7. ¥periph_mrt



マルチレートタイマーの例である。マルチレートタイマー(MRT)は4つのチャンネルの繰り返し割り込みタイマーを提供し、各チャンネルは独立の時間間隔を設定出来る。

- プログラムは Chip_MRT_GetIdleChannel で adle チャンネルを取得し、ランダム時間間隔を得られる。
 4 つのチャンネルはビジーな場合、青い LED 点灯。
- システムタイマーは緑 LED を 5 回点滅/秒と設定する。プログラム実行後、青い LED 点灯、緑 LED は 固定周波数で点滅し、赤 LED はランダムで点滅すると確認できる。
- HEX ファイルは ¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_mrt¥keil_output¥iflash_nxp _xpresso_812 ディレクトリ下に保存する。

4.1.8. ¥periph_spi



SPI クエリと割り込みの2つの動作モードがある。

- プログラムはループバック通信モードを有効し、1つの開発ボードで完成できる。ユーザーは spi.c
 の87~92行のマクロ定義を編集により、プログラムの動作モードを変更する。
- プログラムは8バイトのデータを書き込み、ループバック通信モードでバッファのデータと送信デー タを比較し、正確なら、青い LED 点灯。

● □HEX ファイルは ¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_spi¥keil_output¥iflash_nxp _xpresso_812 ディレクトリ下に保存する。



4.1.9. ¥periph_uart

÷	periph_sct
+)	periph_spi
÷… 🚞	periph_uart
±)	Set as Active Project

UART0 と UART1 間のデータ伝送プロセスであり、UART0 はクエリモードで、UART1 は割り込みモードを使用する。

- テストする前に TXD0 (P_0.4)と RXD1 (P_0.14)を接続し、RXD0 (P_0.0)と TXD1 (P_0.13)を接続。
- プログラムは UARTO から UART1 ヘデータ送信する、次に UART1 は受信データを UARTO に返信する。受送信データを比較、同じなら、青い LED 点灯。
- HEX ファイルは ¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_uart¥keil_output¥iflash_nx p_xpresso_812 ディレクトリ下に保存する。

4.1.10. ¥periph_uart_rom

periph_uart
 periph_uart_rom
 Set as Active Project

内部 ROM スペースにある UART API 関数の使い方を実演する。

開発ボードにプログラムをダウンロード後、ボーレートを115200、ハードウェアフロー制御なしと設定、RST キーでプロセッサリセット、ハイパターミナル上の表示情報は下記の通り:



 シリアルポートはクエリモードで入力文字を待ち、ハイパーターミナルで hello NXP を入力し、正確 な文字を入力すると、シリアルポートは割り込みモードに入り、送信した文字を受信する。x を入力 すると、テストを終了する。



低価格、高品質が不可能? 日昇テクノロジーなら可能にする



□HEXファイルは

¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_uart_rom¥keil_output¥iflash_nxp_ xpresso_812 ディレクトリ下に保存する。

4.1.11. ¥periph_wkt

periph_uart
 periph_uart_rom
 periph_wkt
 Set as Active Project

ウェイクアップ・タイマにより、プロセッサを低消費電力モードからウィックアップさせる。

プログラム実行後、開発ボードの青い LED 点滅、PIO0_6 ピンローにし、プロセッサは低消費電力モードに入る。PIO0_6 をハイにして、数秒後、ウェイクアップタイマーは動作モードに入り、青い LED 点滅、PIO0_6 ピンがローにするたびに、プロセッサは低消費電力モードを切り替える、4 つの低消費電力モードは SLEEP、DEEP_SLEEP、POWER_DOWN、DEEP_POWER_DOWN。

● □HEXファイルは Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_wkt¥keil_output¥iflash_nxp_ xpresso_812 ディレクトリ下に保存する。

4.1.12. ¥periph_wwdt



ウォッチドッグ割り込みトリガー機能を実演する。

● LPC812 プロセッサはウォッチドッグ機能があり、ユーザーは警報の時間をフィード時間内に設定し、



プログラムは警報時間内で、フィードしない場合、ウォッチドッグ警報がトリガーし、割り込みが生成する。

- プログラム実行後、割り込み発生する場合、青い LED は値を反転動作する。
- HEX ファイルは ¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥periph¥periph_wwdt¥keil_output¥iflash_nx p_xpresso_812 ディレクトリ下に保存する。

4.2. lpcopen_freertos_8xx.uvmpw プロジェクト管理ファイルのプ

ログラム説明

4.2.1. ¥freertos_blinky



FreeRTOS OS で LED 点滅を実現する。

- 3 つのタスクを作成する:1 つは青い LED 点滅、1 つは緑 LED 点滅、1 つは赤 LED 点滅を制御する。
- HEX ファイルは

¥Code¥LPCOpen_v1.03¥applications¥lpc8xx¥examples¥freertos¥freertos_blinky¥keil_output¥ifl ash_nxp_xpresso_812 ディレクトリ下に保存する。

以上。