

S1C31W74
周辺回路サンプル
ソフトウェアマニュアル

評価ボード・キット、開発ツールご使用上の注意事項

1. 本評価ボード・キット、開発ツールは、お客様での技術的評価、動作の確認および開発のみに用いられることを想定し設計されています。それらの技術評価・開発等の目的以外には使用しないで下さい。本品は、完成品に対する設計品質に適合していません。
2. 本評価ボード・キット、開発ツールは、電子エンジニア向けであり、消費者向け製品ではありません。お客様において、適切な使用と安全に配慮願います。弊社は、本品を用いることで発生する損害や火災に対し、いかなる責も負いかねます。通常の使用においても、異常がある場合は使用を中止して下さい。
3. 本評価ボード・キット、開発ツールに用いられる部品は、予告無く変更されることがあります。

本資料のご使用につきましては、次の点にご留意願います。

本資料の内容については、予告無く変更することがあります。

1. 本資料の一部、または全部を弊社に無断で転載、または、複製など他の目的に使用することは堅くお断りいたします。
2. 本資料に掲載される応用回路、プログラム、使用方法等はあくまでも参考情報であり、これらに起因する第三者の知的財産権およびその他の権利侵害あるいは損害の発生に対し、弊社はいかなる保証を行うものではありません。また、本資料によって第三者または弊社の知的財産権およびその他の権利の実施権の許諾を行うものではありません。
3. 特性値の数値の大小は、数直線上の大小関係で表しています。
4. 製品および弊社が提供する技術を輸出等するにあたっては「外国為替および外国貿易法」を遵守し、当該法令の定める手続きが必要です。大量破壊兵器の開発等およびその他の軍事用途に使用する目的をもって製品および弊社が提供する技術を費消、再販売または輸出等しないでください。
5. 本資料に掲載されている製品は、生命維持装置その他、きわめて高い信頼性が要求される用途を前提としていません。よって、弊社は本（当該）製品をこれらの用途に用いた場合のいかなる責任についても負いかねます。
6. 本資料に掲載されている会社名、商品名は、各社の商標または登録商標です。
ARM, Cortex, Keil および μ Vision は、ARM Limited(またはその子会社)の EU またはその他の国における登録商標です。IAR Systems, IAR Embedded Workbench, C-SPY, I-jet, IAR および IAR システムズのロゴタイプは、IAR Systems AB が所有権を有する商標または登録商標です。SEGGER および J-Link は、SEGGER Microcontroller GmbH & Co. KG の商標または登録商標です。All rights reserved.
"Reproduced with permission from ARM Limited. Copyright © ARM Limited"

目 次

1. 概要.....	1
1.1 動作環境.....	1
2. サンプルソフトウェアの操作.....	2
2.1 サンプルソフトウェアの構成.....	2
2.2 ターゲットダウンロード及び実行のための準備.....	4
2.2.1 ハードウェアの接続.....	4
2.2.2 UART 用 USB アダプタとの接続.....	4
2.2.3 I2C シリアル EEPROM (24FC512) との接続.....	6
2.3 IAR EWARM IDE サンプルソフトウェアの実行手順.....	7
2.3.1 ソフトウェアのセットアップ.....	7
2.3.2 ワークスペースのオープン.....	8
2.3.3 アクティブプロジェクトの選択.....	9
2.3.4 デバッグプローブの設定.....	10
2.3.5 フラッシュローダの設定.....	11
2.3.6 プロジェクトのビルド.....	12
2.3.7 プロジェクトのダウンロードおよびデバッグ.....	14
2.4 MDK-ARM (μVision) サンプルソフトウェアの実行手順.....	15
2.4.1 ソフトウェアのセットアップ.....	15
2.4.2 ワークスペースのオープン.....	16
2.4.3 アクティブプロジェクトの選択.....	17
2.4.4 デバッグプローブの設定.....	18
2.4.5 フラッシュローダの設定.....	20
2.4.6 プロジェクトのビルド.....	22
2.4.7 プロジェクトのダウンロードおよびデバッグ.....	23
3. サンプルソフトウェアの詳細.....	24
3.1 クロックジェネレータ(CLG).....	24
3.2 DMA コントローラ(DMAC).....	25
3.3 I2C(I2C).....	28
3.4 LCD ドライバ(LCD32B).....	29
3.5 入出力ポート(PPORT).....	31
3.6 同期式クワッドシリアルインタフェース(QSPI).....	32
3.7 同期式クワッドシリアルインタフェース with DMA (QSPI_DMA).....	34
3.8 同期式クワッドシリアルインタフェース マスタ(QSPI_MASTER).....	35
3.9 同期式クワッドシリアルインタフェース スレーブ(QSPI_SLAVE).....	38
3.10 IR リモートコントローラ(REMC2).....	40
3.11 R/F 変換器(RFC).....	41
3.12 リアルタイムクロック(RTCA).....	42
3.13 サウンドジェネレータ(SNDA).....	43
3.14 同期式シリアルインタフェース マスタ(SPIA_MASTER).....	44
3.15 同期式シリアルインタフェース スレーブ(SPIA_SLAVE).....	46
3.16 電源電圧検出回路(SVD2).....	47
3.17 16 ビットタイマ(T16).....	48

3.18 16 ビット PWM タイマ(T16B).....	49
3.19 UART(UART2)	50
3.20 USB CDC デバイス(USB_CDC).....	51
3.21 USB 2.0 FS デバイスコントローラ(USB_HID).....	52
3.22 USB MSC デバイス(USB_MSC)	54
3.23 ウォッチドッグタイマ(WDT2).....	55
3.24 フラッシュプログラミング	56
3.25 EEPROM ライブラリ (EEPROM).....	57
3.26 ブートローダ (BootLoader)	58
付録. サンプルプロジェクトのコードサイズ概要.....	59
改訂履歴表	60

1. 概要

本マニュアルは、S1C31W74マイクロコントローラ用サンプルソフトウェアの使用方法について記載しています。また、サンプルソフトウェア実行時の出力例も記載しています。

サンプルソフトウェアは、S1C31W74マイクロコントローラに搭載されている各種周辺回路の使い方の説明用に作成されており、S1C31W74周辺回路サンプルソフトウェアパッケージに含まれています。

S1C31W74マイクロコントローラの詳細については、S1C31W74テクニカルマニュアルを参照してください。

S1C31W74周辺回路サンプルソフトウェアパッケージの構成を以下に示します。

- 周辺回路ライブラリ
- 周辺回路の使用例を記述したサンプルソフトウェア
- S1C31W74 機種定義ファイルなどの機種別情報ファイル
- システムビューアデスクリプションファイル(svd ファイル)
- フラッシュローダ

周辺回路サンプルソフトウェアは、S1C31W74周辺回路を制御するための周辺回路ライブラリの使い方を示すことを目的としています。各サンプルソフトウェアは、個々の周辺回路の機能の実例を示しており、周辺回路の各種モードを実行します。

S1C31W74機種定義ファイルは、周辺回路制御レジスタのアドレス、アクセスサイズ、リード/ライトアクセスタイプを定義しています。個々のレジスタやそのビットフィールドは、S1C31W74機種定義ファイルに記述されたハードウェア情報を基にアクセスされます。周辺回路ライブラリは、周辺回路の初期化や周辺回路機能の管理など、周辺回路の機能を簡単に制御するための関数を提供します。

※1 本マニュアルに加えて、弊社Webサイトから入手できる“S1C31W74テクニカルマニュアル”および“S5U1C31W74T1マニュアル”も併せて参照してください。

※2 本マニュアルは、ver.3.00以降のサンプルソフトウェアパッケージを対象としています。

1.1 動作環境

S1C31W74サンプルソフトウェアを実行するためには、以下の機器が必要です。

- 評価ボード
 - S1C31W74 搭載 S5U1C31W74T1 評価ボード
 - S5U1C31001L1200 アダプタボード (Bridge Board Ver.2)
- デバッグプローブ *1
 - IAR Systems 社製 I-jet または SEGGER 社製 J-Link
- 統合開発環境
 - IAR Embedded Workbench for ARM® (IAR EWARM) または MDK-ARM®
- その他のデバイス (オプション)
 - UART 用 USB アダプタ
 - I2C シリアル EEPROM (Microchip 社製 EEPROM 24FC512)

*1: I-jet は IAR EWARM でのみ使用可能です。J-Link は IAR EWARM と MDK-ARM の両方で使用可能です。

2. サンプルソフトウェアの操作

2. サンプルソフトウェアの操作

2.1 サンプルソフトウェアの構成

S1C31W74サンプルソフトウェアは、S1C31W74の各周辺回路ハードウェアとのインタフェースを担う周辺回路ライブラリ関数のコール方法を実例で示しています。各周辺回路に対応するライブラリが、ソースファイル（“xxx.c”）とインクルードファイル（“xxx.h”）の構成で用意されています。インクルードファイルには、それぞれのライブラリで使用する定数、型、関数が定義されています。

S1C31W74サンプルソフトウェアは、ワークスペース“Examples”にまとめられています。ほとんどの場合、各周辺回路のサンプルソフトウェアは1つのサンプルプロジェクトとして提供されます。例えば、CLGプロジェクトは、CLG（クロックジェネレータ）を制御対象の周辺回路として作成されています。より複雑な周辺回路の場合は、その周辺回路の異なる側面を扱う複数のサンプルプロジェクトが用意されています。例えば、QSPIは複数のプロジェクト（接頭辞がQSPIのプロジェクト）が用意されています。図2.1.1にソフトウェアレイヤ間の関係を示します。

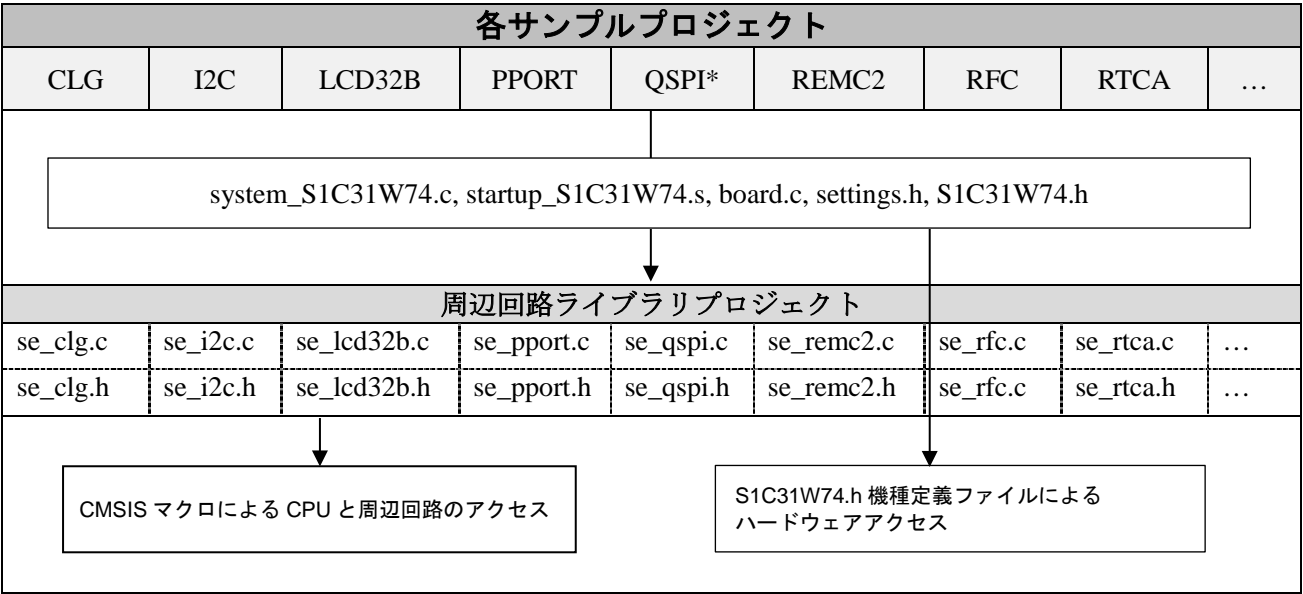


図 2.1.1 S1C31W74 サンプルソフトウェアワークスペース

注意:

- すべてのサンプルプロジェクトは、“CMSIS¥Device”フォルダにある IAR/Keil スタートアップファイルと IAR/Keil システムファイルを共有します。
- すべてのサンプルプロジェクトは、“board”フォルダにあるボード関連ファイルを共有します。
- board.c ファイルの BoardInit 関数に、起動時のシステム初期化処理が実装されています。
- BoardInit 関数は、SystemInit 関数からコールされます。この関数で、CPU 動作周波数、Sys Tick 値、優先順位などを設定します。
- SystemInit 関数は、IAR/Keil スタートアップファイルからコールされます。

2. サンプルソフトウェアの操作

settings.hファイルには、デフォルト設定の起動動作を変更するための定数定義がコメントアウトして記述されています。表2.1.1に、ボード機能設定用定数と設定内容を示します。

表2.1.1 S1C31W74ボード定義

設定用定数	定義時	未定義時
UART_PRINTF	標準ライブラリのprintf関数が、UARTコンソールに対して出力を行います。ただし、ハードウェアの追加が必要です。 MDK-ARM(μVision)の場合、この設定機能を定義する必要があります。	セミホスティングライブラリのprintf関数が、IAR EWARMターミナルウィンドウに対して出力を行います。
EXECUTE_ON_OSC3	BoardInit関数で、CPUの動作クロックを高精度のクロックに切り換えます。	CPUがデフォルトクロックで動作します。
IOSC_AUTOTRIMMING_ON	CPU動作クロックの高い精度を維持するために、BoardInit関数で、IOSCトリミングを実行します。	IOSCトリミングは実行されません。起動時間を短縮できます。
CACHE_ENABLED	Flashアクセス時の命令キャッシュが有効になります。デバッグターゲットでは無効です。	Flashアクセス時の命令キャッシュは無効です。
TICKLESS_ENABLED	SYSTICK割り込みが無効になります。	最後の起動時からのCPU動作時間の記録にSYSTICK割り込みが使用されます。
BOOT_LOADER	ブートローダーを実行します。	アプリケーションを実行します。
QSPI_MODE_SINGLE	QSPIの転送モードとしてシングルが選択されます。	QSPIの転送モードとしてデュアルまたはクワッドが選択されます。
SPIA_DMA	SPIA_MASTERおよびSPIA_SLAVE プロジェクトでDMAを使用します。	SPIA_MASTERおよびSPIA_SLAVE プロジェクトでDMAを使用しません。

注意:

- デフォルト設定ではすべてが未定義の状態です。
- 入出力にUARTコンソールを使用するには、settings.hファイル内のUART_PRINTFを定義している行をコメント解除してください。
- CPUのディープスリープ機能を使用する場合、settings.hファイル内のTICKLESS_ENABLEDを定義している行をコメント解除してください。

2. サンプルソフトウェアの操作

2.2 ターゲットダウンロード及び実行のための準備

2.2.1 ハードウェアの接続

サンプルソフトウェアの実行およびデバッグには、以下のハードウェアの使用を推奨します。

- S5U1C31W74T1 評価ボード
- S5U1C31001L1200 (Bridge Board Ver.2)
- デバッグプローブ (IAR Systems I-jet または SEGGER J-Link)

ハードウェア接続の詳細については、“S5U1C31W74T1 マニュアル” を参照してください。

2.2.2 UART用USBアダプタとの接続

UART 用 USB アダプタは、UART を使用するサンプルプログラムで使します。UART 用 USB アダプタを使用して S5U1C31W74T1 評価ボードと PC を接続することで、PC との UART 通信が可能になります。図 2.2.2.1 (写真)、図 2.2.2.2 (結線図) に UART 用 USB アダプタと S5U1C31W74T1 評価ボードの接続を示します。

UART 通信を行うためには、settings.h ファイルの UART_PRINTF の定義を有効にしてサンプルプログラムをビルドする必要があります (2.1 節および表 2.1.1 を参照)。また、PC 上でシリアル通信のターミナルソフトウェアを起動し、シリアルポートを設定する必要があります。表 2.2.2.1 にシリアルポートの設定値を示します。

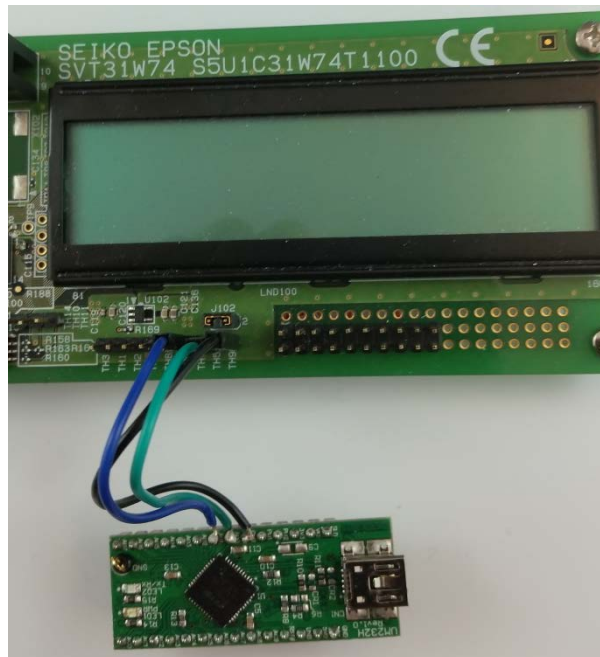


図2.2.2.1 S5U1C31W74T1評価ボードとUART用USBアダプタの接続例

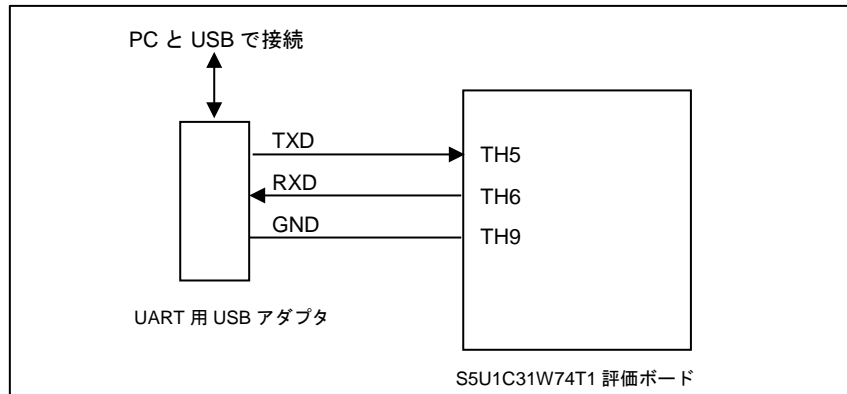


図2.2.2.2 S5U1C31W74T1評価ボードとUART用USBアダプタとの結線図

表 2.2.2.1 シリアル通信ターミナルソフトウェアのシリアルポート設定

パラメータ	設定値
ボーレート	115200 bps
データ	8 ビット
ストップビット	1 ビット
パリティ	None

S5U1C31W74T1 評価ボードの詳細については、“S5U1C31W74 マニュアル”を参照してください。

注意： 図 2.2.2.1 で使用している UART 用 USB アダプタは市販の製品であり、弊社より提供されません。必要に応じて用意して下さい。

2. サンプルソフトウェアの操作

2.2.3 I2CシリアルEEPROM (24FC512) との接続

I2C シリアル EEPROM (24FC512) は、I2C サンプルプログラムで使します (「3.3 I2C (I2C)」を参照)。
図 2.2.3.1 の写真および図 2.2.3.2 の結線図は、I2C シリアル EEPROM (24FC512) と S5U1C31W74T1 評価ボードの接続を示しています。

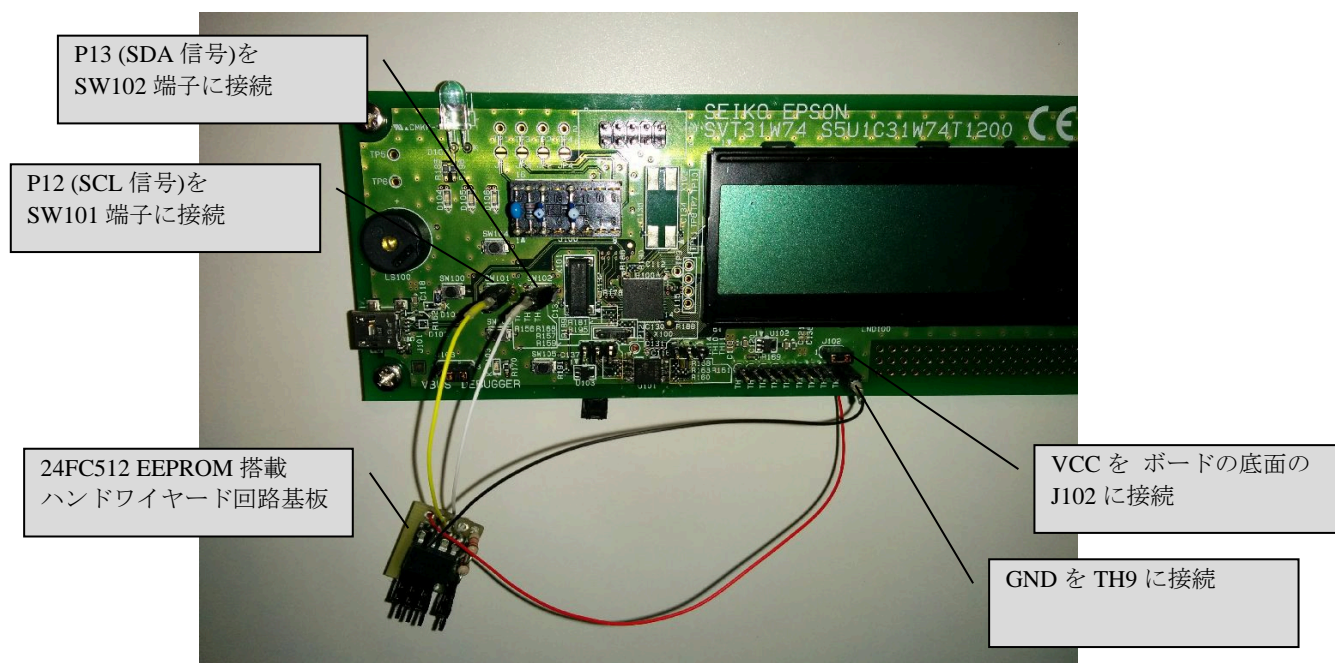


図2.2.3.1 S5U1C31W74T1評価ボードとI2CシリアルEEPROMの接続

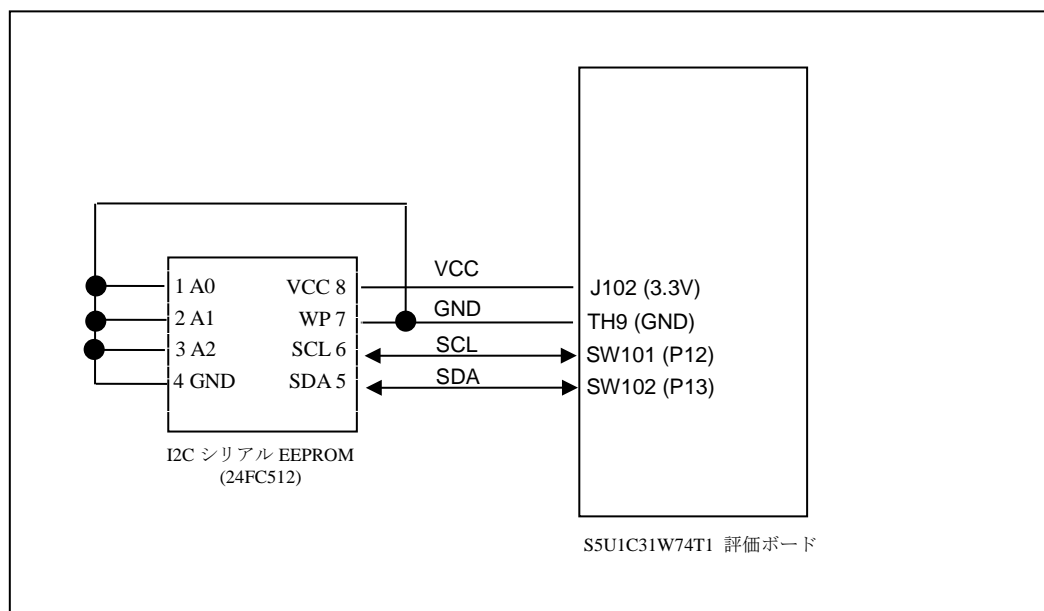


図2.2.3.2 S5U1C31W74T1評価ボードとI2CシリアルEEPROM間結線図

2.3 IAR EWARM IDEサンプルソフトウェアの実行手順

2.3.1 ソフトウェアのセットアップ

IAR EWARM上でサンプルソフトウェアを実行する場合、事前にサンプルソフトウェアのセットアップが必要です。サンプルソフトウェアのセットアップは、以下の手順で行います。

(1) サンプルソフトウェアのダウンロード

S1C31W74 周辺回路サンプルソフトウェアパッケージ (.exe) を弊社マイクロコントローラ Web サイトからダウンロードします。

(2) サンプルソフトウェアパッケージのインストール

起動中の他のすべてのプログラムを終了します。ダウンロードした実行形式のサンプルソフトウェアパッケージ (.exe) を管理者権限で実行してインストーラを起動します。インストーラ起動後に表示される「SOFTWARE LICENSE AGREEMENT」のライセンス条項を確認して、ライセンス条項に同意する場合は[I Agree]ボタンを選択します。続いて、所望のインストールフォルダを指定してインストールを行います。インストール終了後、セットアップユーティリティ「ToolchainSetup.exe」が自動的に起動するので [Next]ボタンを押します。次に、所望のバージョンの IAR EWARM をチェックボックスで選択して[Next]ボタンを押します。“Done” メッセージ表示後、[Finish]ボタンを押してインストールを完了します。

注: セットアップユーティリティは、機種別情報ファイル、svd ファイル、フラッシュローダを IAR EWARM のインストールフォルダへコピーします。

(3) IAR EWARM の再起動

セットアップが正常に完了したら IAR EWARM IDE を起動します。

本サンプルソフトウェアの評価に使用した IAR EWARM のバージョン、及び、セットアップユーティリティの詳細については、サンプルソフトウェアパッケージ内の“README_IAR.txt”を参照してください。

2. サンプルソフトウェアの操作

2.3.2 ワークスペースのオープン

サンプルソフトウェアパッケージに含まれる、すべてのサンプルソフトウェアプロジェクトは、“Examples” フォルダに1つに纏められて格納されています。これらのプロジェクトは、ビルド時に sePeripheralLibrary を必要とします。また、いくつかのプロジェクトでは、seGraphicsLibrary も必要とします。

すべてのサンプルソフトウェアプロジェクトを含むワークスペースを開くには、IAR EWARMメニューの [ファイル] > [開く] > [ワークスペース] をクリックし、“Examples¥WORKSPACE¥S5U1C31W74T1¥IAR” フォルダに移動して “Examples.eww” ファイルを選択します。

オプションとして、各サンプルソフトウェアプロジェクトのサブフォルダに、IARプロジェクトファイル (.ewp) とIARワークスペースファイル (.eww) を含む “board¥S5U1C31W74T1¥IAR” サブフォルダが用意されています。個々のサンプルソフトウェアプロジェクトのワークスペースを開くには、IAR EWARMメニューの [ファイル] > [開く] > [ワークスペース] をクリックし、そのプロジェクトの “board¥S5U1C31W74T1¥IAR” フォルダに移動してワークスペースファイル (.eww) を選択します。

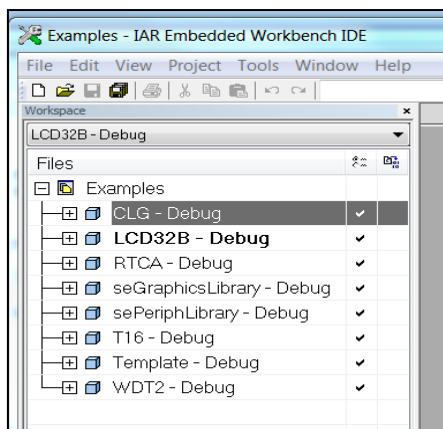


図 2.3.2.1 複数プロジェクトのワークスペースの例

2.3.3 アクティブプロジェクトの選択

ビルドおよびデバッグのターゲットとなるプロジェクト(アクティブプロジェクト)を選択します。IAR EWARMの「ワークスペース」ウィンドウからターゲットプロジェクトを選択し、右クリックで表示されるメニューから「アクティブに設定」を選択します。また、「ワークスペース」ウィンドウ上部のドロップダウンリストを使用することで、アクティブプロジェクトとビルド構成を同時に選択することができます(図2.3.3.1)。

サンプルソフトウェアの各プロジェクトには、以下に示す3種類のビルド構成が用意されています。

- Debug - 内蔵 RAM 上で実行されるプログラム ROM を生成します。
最適化レベルは“低”に設定されています。
- DebugFlash - 内蔵 Flash メモリ上で実行されるプログラム ROM を生成します。
最適化レベルは“低”に設定されています。
- ReleaseFlash - 内蔵 Flash メモリ上で実行されるプログラム ROM を生成します。
最適化レベルは“高”に設定されています。

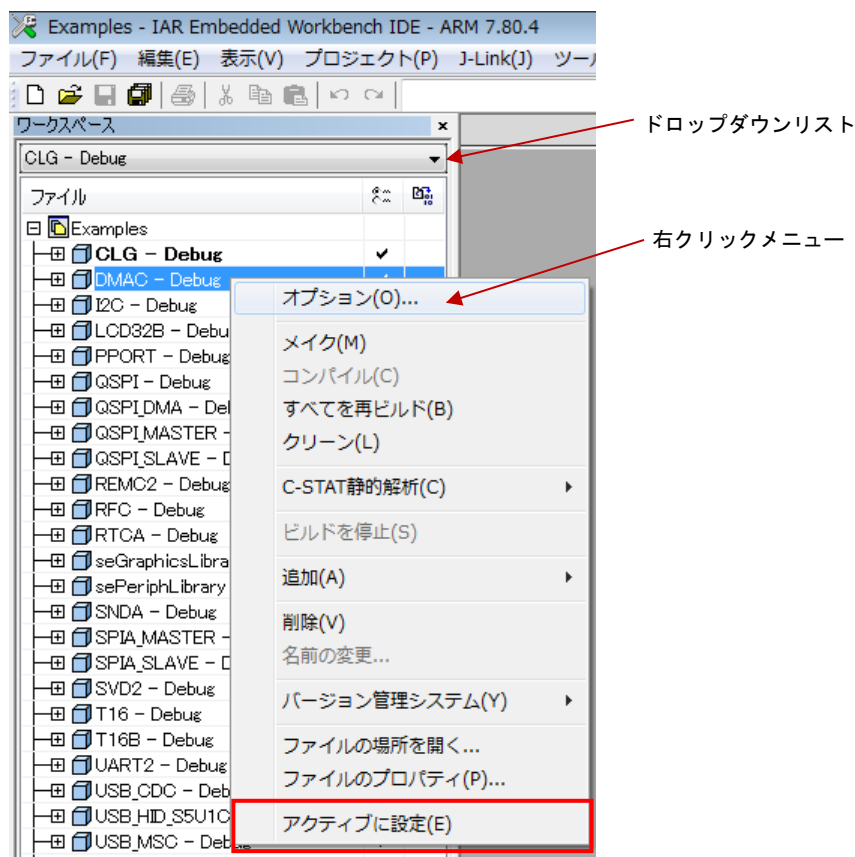


図 2.3.3.1 アクティブプロジェクトの設定

2. サンプルソフトウェアの操作

2.3.4 デバッグプローブの設定

ターゲットボードにデバッグプローブを接続してデバッグする場合、使用するデバッグプローブの種類に応じて、IAR EWARM 上でドライバの設定が必要です。

デバッグプローブの設定は、以下の手順で行います。

- (1) IAR EWARM メニューの [プロジェクト] > [オプション] を選択します。
- (2) 表示されたダイアログの [カテゴリ] リストから [デバッグ] を選択します (図 2.3.4.1)。
- (3) [設定] タブを選択し、[ドライバ] のドロップダウンリストから使用するデバッグプローブの種類を選択します。I-jet を使用する場合は “I-jet/JTAGjet”、J-Link を使用する場合は “J-Link/J-Trace” を選択してください (図 2.3.4.1)。

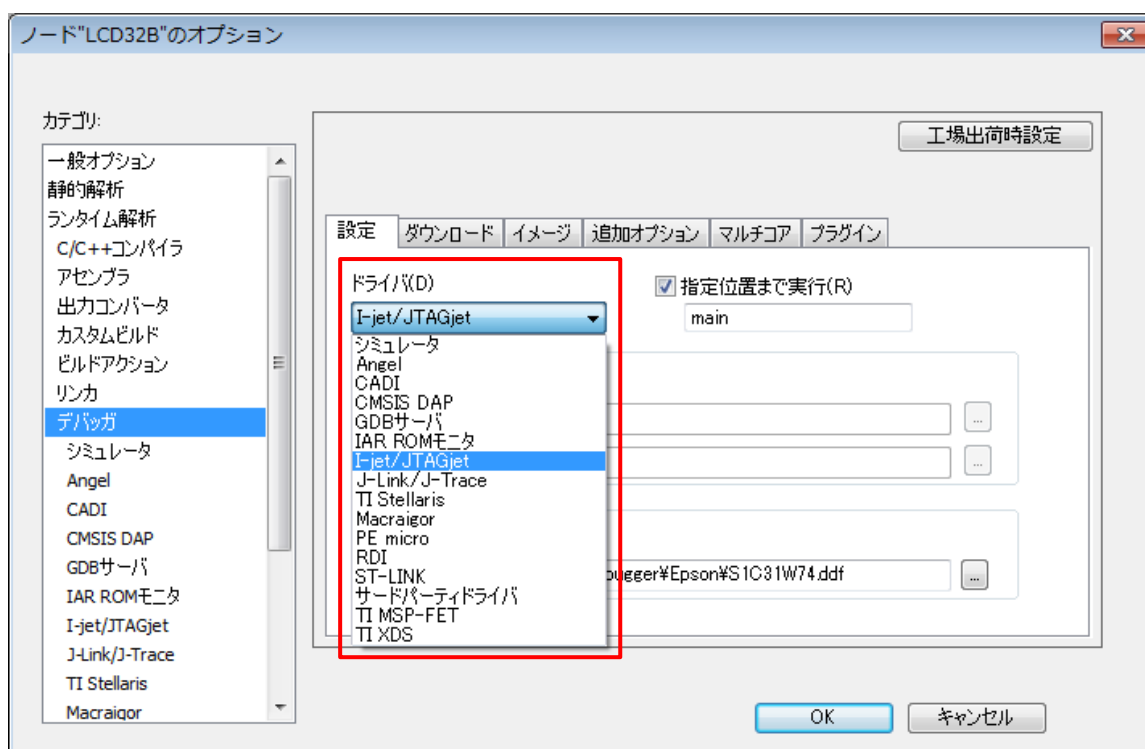


図 2.3.4.1 デバッグプローブの設定

2.3.5 フラッシュローダの設定

アクティブプロジェクトのビルド構成が DebugFlash または ReleaseFlash の場合、内蔵 Flash メモリにプログラムをロード（Flash プログラミング）するためのフラッシュローダを設定する必要があります。ビルド構成が Debug の場合は、内蔵 Flash メモリではなく内蔵 RAM にプログラムをロードするため、フラッシュローダの設定を解除する必要があります。

フラッシュローダの設定は、以下の手順で行います。

- ビルド構成が Debug の場合（RAM 上のプログラムを実行）
 - (1) IAR EWARM メニューの [プロジェクト] > [オプション] を選択します。
 - (2) 表示されたダイアログの [カテゴリ] リストから [デバッグ] を選択します (図 2.3.5.1)。
 - (3) [ダウンロード] タブを選択します (図 2.3.5.1)。
 - (4) [フラッシュローダを使用する(U)] のチェックを無効にします (図 2.3.5.1)。
- ビルド構成が DebugFlash または ReleaseFlash の場合（内蔵 Flash メモリのプログラムを実行）
 - (1) IAR EWARM メニューの [プロジェクト] > [オプション] を選択します。
 - (2) 表示されたダイアログの [カテゴリ] リストから [デバッグ] を選択します (図 2.3.5.1)。
 - (3) [ダウンロード] タブを選択します (図 2.3.5.1)。
 - (4) [フラッシュローダを使用する(U)] のチェックを有効にします (図 2.3.5.1)。
 - (5) [デフォルトの.board ファイルのオーバーライド] のチェックを有効にします (図 2.3.5.1)。
 - (6) [...] ボタンをクリックし、.board ファイルとして“S1C31W74_int.board”を選択します (図 2.3.5.1)。

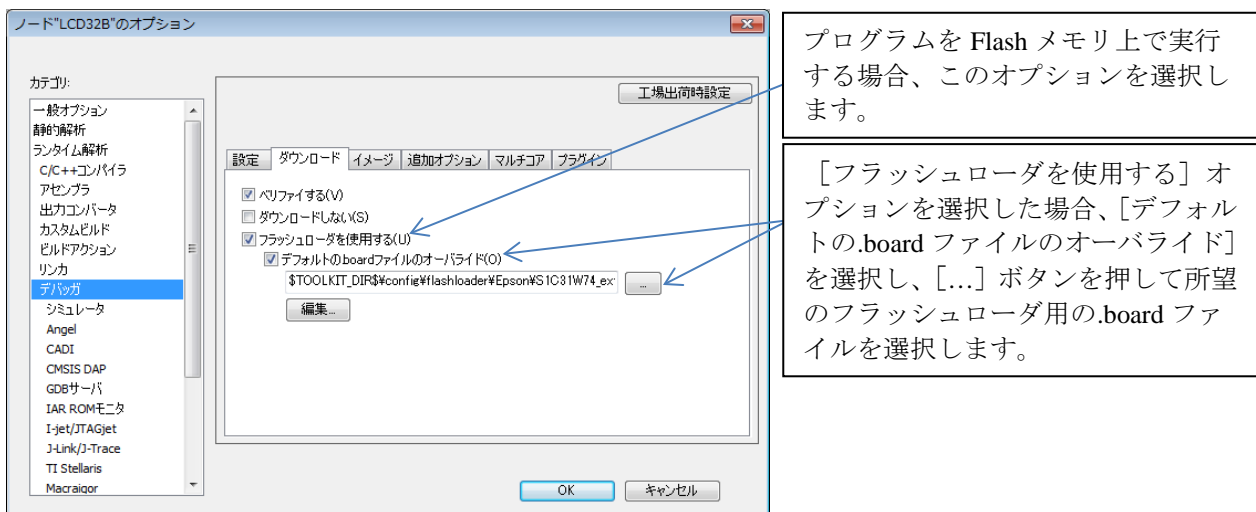


図 2.3.5.1 フラッシュローダの設定

2. サンプルソフトウェアの操作

2.3.6 プロジェクトのビルド

プロジェクトをビルドするには、IAR EWARM メニューの [プロジェクト] から [メイク]、[すべてを再ビルド] の2つのビルドコマンドのいずれかを選択します (図 2.3.6.1)。

注意： ビルド中にリンカエラーが発生する場合、sePeriphLibrary と seGraphicsLibrary の2つのライブラリがビルドされていない可能性があります。これらのライブラリをビルドしてから、再度、ターゲットプロジェクトをビルドしてください。このとき、ライブラリはターゲットプロジェクトと同じビルド構成を選択してください。例えば、“CLG-Debug”をビルドする場合、“sePeriphLibrary-Debug”、“seGraphicsLibrary-Debug”を選択してビルドしてください。

また、IAR EWARM メニューの [プロジェクト] > [バッチビルド] を選択することで、サンプルソフトウェアに含まれる、すべてのプロジェクトをまとめてビルドすることができます (図 2.3.6.1)。[バッチビルド] ダイアログの [Make] または [Rebuild All] ボタンのいずれかをクリックすると、指定されたビルド構成のプロジェクトがまとめてビルドされます (図 2.3.6.2)。

バッチビルドは、以下に示すビルド構成から選択して実行できます。

- all_Debug - ビルド構成が Debug のプロジェクトをまとめてビルド
- all_DebugFlash - ビルド構成が DebugFlash のプロジェクトをまとめてビルド
- all_ReleaseFlash - ビルド構成が ReleaseFlash のプロジェクトをまとめてビルド

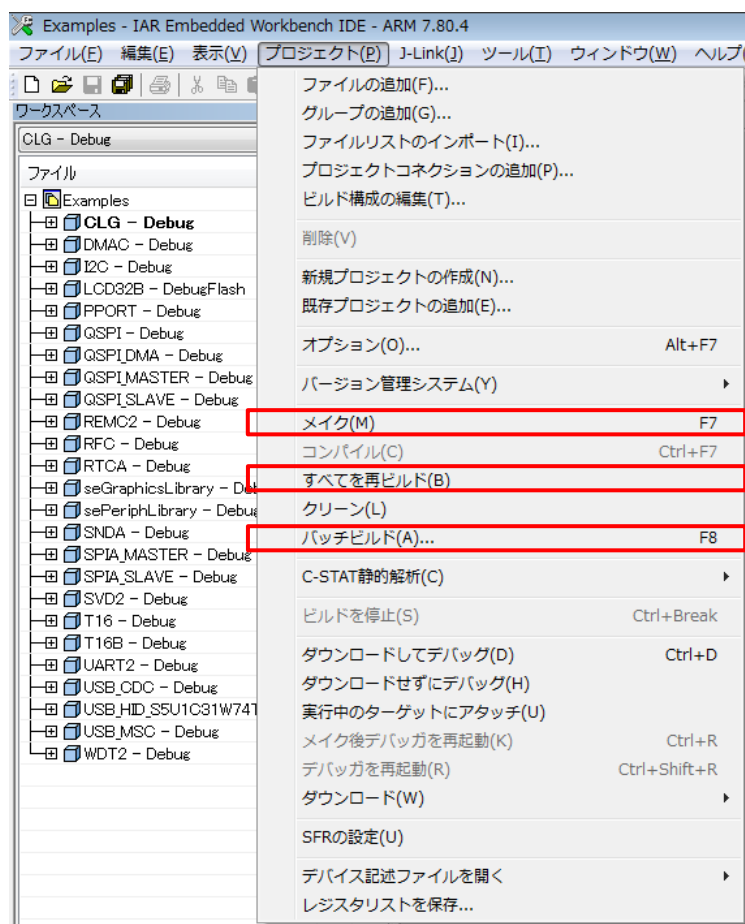


図 2.3.6.1 ビルドコマンド

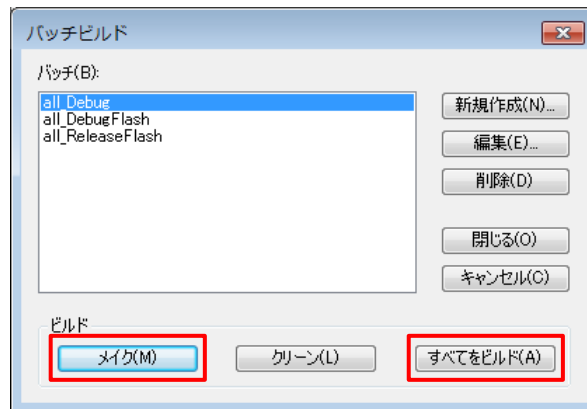


図 2.3.6.2 バッチビルド

2. サンプルソフトウェアの操作

2.3.7 プロジェクトのダウンロードおよびデバッグ

ビルドに成功したら、アクティブプロジェクトをターゲットボードにダウンロードします。アクティブプロジェクトをダウンロードするには、IAR EWARMメニューの [プロジェクト] > [ダウンロードしてデバッグ] を選択します (図2.3.7.1)。

プロジェクトがDebugビルドの場合、プログラムイメージは内蔵RAMにロードされてデバッグを開始します。プロジェクトがDebugFlashビルドまたはReleaseFlashビルドの場合、プログラムイメージは内蔵Flashメモリにロードされてデバッグを開始します。

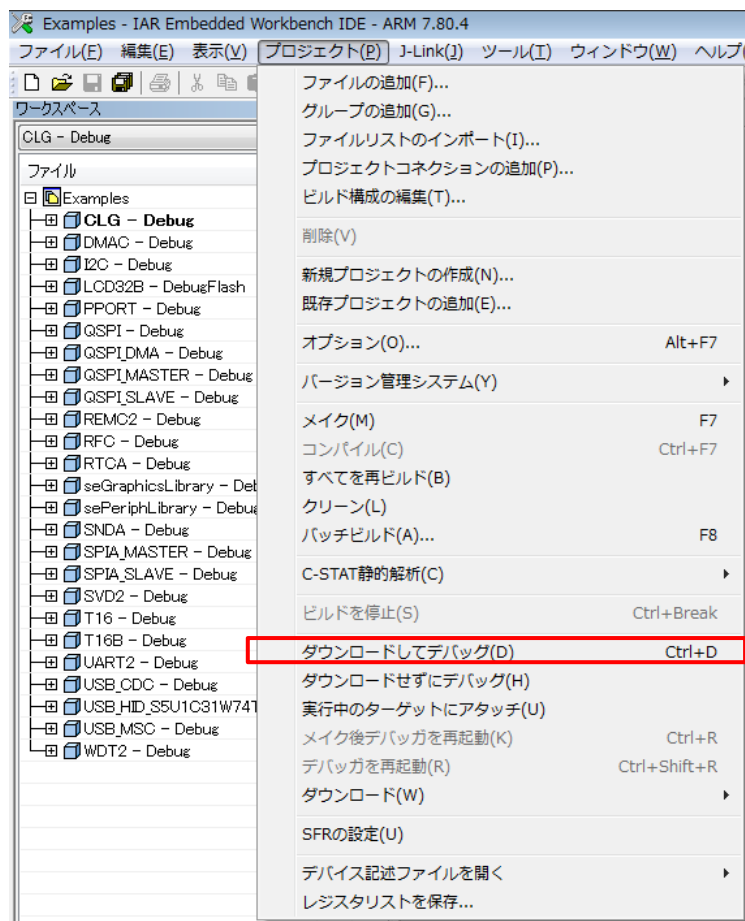


図 2.3.7.1 プロジェクトのダウンロードおよびデバッグ

2.4 MDK-ARM (μVision) サンプルソフトウェアの実行手順

2.4.1 ソフトウェアのセットアップ

MDK-ARM (μVision) 上でサンプルソフトウェアを実行する場合、事前にサンプルソフトウェアのセットアップが必要です。サンプルソフトウェアのセットアップは、以下の手順で行います。

(1) サンプルソフトウェアのダウンロード

S1C31W74 周辺回路サンプルソフトウェアパッケージ (.exe) を弊社マイクロコントローラ Web サイトからダウンロードします。

(2) サンプルソフトウェアパッケージのインストール

起動中の他のすべてのプログラムを終了します。ダウンロードした実行形式のサンプルソフトウェアパッケージ (.exe) を管理者権限で実行してインストーラを起動します。インストーラ起動後に表示される「SOFTWARE LICENSE AGREEMENT」のライセンス条項を確認して、ライセンス条項に同意する場合は[I Agree]ボタンを選択します。続いて、所望のインストールフォルダを指定してインストールを行います。インストール終了後、セットアップユーティリティ「ToolchainSetup.exe」が自動的に起動するので [Next]ボタンを押します。次に、所望のバージョンの μVision、デバッグプロンプをチェックボックスで選択して[Next]ボタンを押します。“Done” メッセージ表示後、[Finish]ボタンを押してインストールを完了します。

注: セットアップユーティリティは、フラッシュローダを μVision のインストールフォルダへコピーし、デバッグ設定ファイルを各サンプルソフトウェアプロジェクトのワークスペースフォルダへコピーします。

(3) μVision の起動

セットアップが正常に完了したら μVision を起動します。

本サンプルソフトウェアの評価に使用した MDK-ARM(μVision)のバージョン、及び、セットアップユーティリティの詳細については、サンプルソフトウェアパッケージ内の“README_KEIL.txt”を参照して下さい。

2. サンプルソフトウェアの操作

2.4.2 ワークスペースのオープン

サンプルソフトウェアパッケージに含まれる、すべてのサンプルソフトウェアプロジェクトは、“Examples” フォルダに1つに纏められて格納されています。これらのプロジェクトは、ビルド時に sePeripheralLibrary を必要とします。また、いくつかのプロジェクトでは、seGraphicsLibrary も必要とします。

すべてのサンプルソフトウェアプロジェクトを含むワークスペースを開くには、μVision メニューの [Project] > [Open Project...] をクリックし、“Examples\WORKSPACE\S5U1C31W74T1\ARM” フォルダに移動して “Examples.uvmpw” ファイルを選択します。

オプションとして、各サンプルソフトウェアプロジェクトのサブフォルダに、μVision プロジェクトファイル (.uvprojx) と μVision マルチプロジェクトワークスペースファイル (.uvmpw) を含む “board\S5U1C31W74T1\ARM” サブフォルダが用意されています。個々のサンプルソフトウェアプロジェクトのワークスペースを開くには、μVision メニューの [Project] > [Open Project...] をクリックし、そのプロジェクトの “board\S5U1C31W74T1\ARM” サブフォルダに移動してマルチプロジェクトワークスペースファイル (.uvmpw) を選択します。

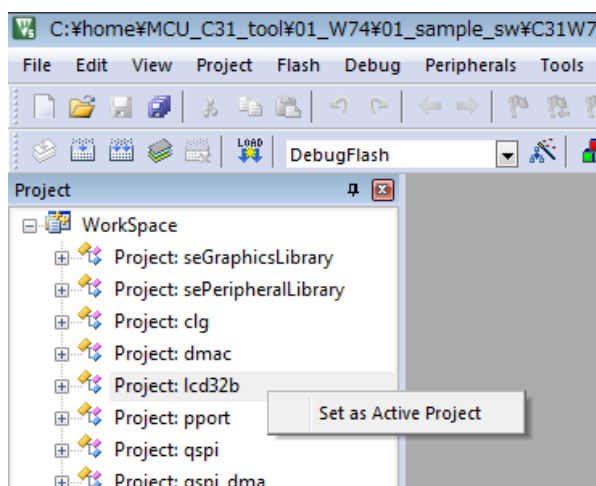


図 2.4.2.1 複数プロジェクトのワークスペースの例

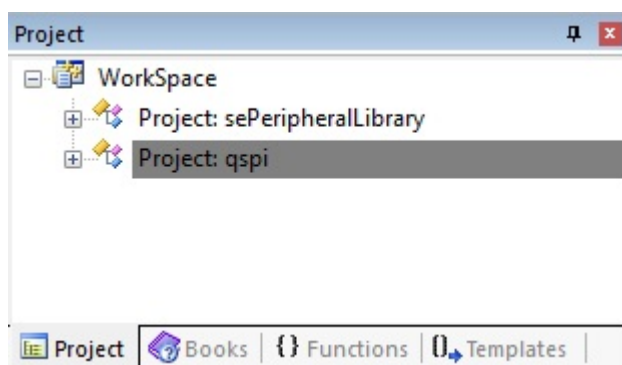


図 2.4.2.2 単体プロジェクトのワークスペースの例

2.4.3 アクティブプロジェクトの選択

ビルドおよびデバッグのターゲットとなるプロジェクト（アクティブプロジェクト）を選択します。
μVisionの [Project] ウィンドウ上からターゲットプロジェクトを選択し、右クリックで表示されるメニューから [Set as Active Project] を選択します（図2.4.3.1）。次に、ツールバー上のドロップダウンリストからビルド構成を選択します（図2.4.3.2）。

サンプルソフトウェアの各プロジェクトには、以下に示す2種類のビルド構成が用意されています。

- Debug - 内蔵 RAM 上で実行されるプログラム ROM を生成します。
- DebugFlash - 内蔵 Flash メモリ上で実行されるプログラム ROM を生成します。

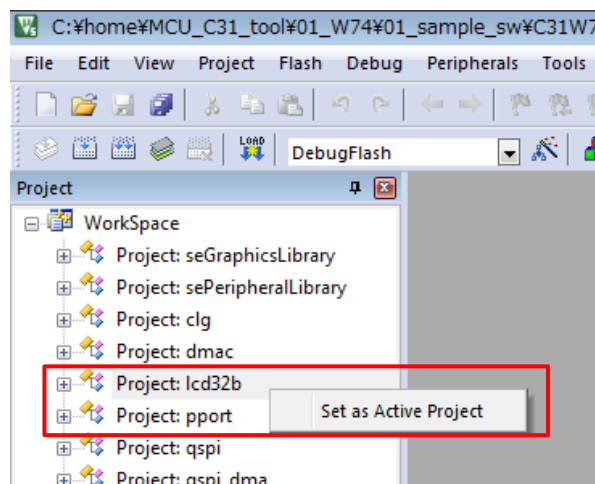


図 2.4.3.1 アクティブプロジェクトの設定

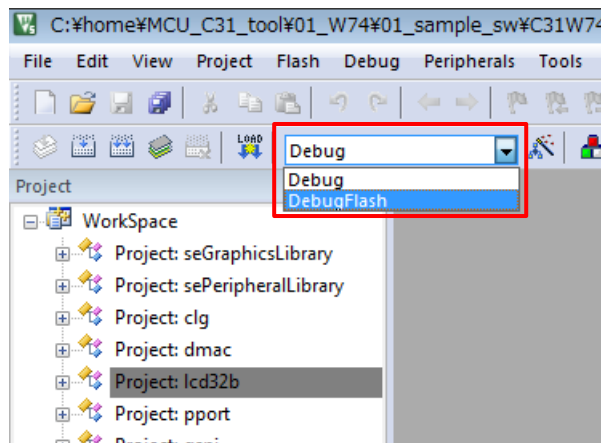


図 2.4.3.2 ビルド構成の選択

2. サンプルソフトウェアの操作

2.4.4 デバッグプローブの設定

ターゲットボードにデバッグプローブを接続してデバッグする場合、使用するデバッグプローブの種類に応じて、 μ Vision 上でドライバの設定が必要です。

デバッグプローブの設定は、以下の手順で行います。

- (1) μ Vision メニューの [Project] > [Options for {project} - Target '{build configuration}'] を選択します。
- (2) [Options for Target '{build configuration}'] ダイアログの [Debug] タブを選択します (図 2.4.4.1)。
- (3) [Use:] チェックボックスの右横にあるドロップダウンリストから [J-Link/J-TRACE Cortex] を選択します (図 2.4.4.1)。
- (4) 上記ドロップダウンリストの右横にある [Settings] ボタンを押下します (図 2.4.4.1)。
- (5) [Cortex JLink/JTrace Target Driver Setup] ダイアログの [Port] ドロップダウンリストから [SW] を選択します (図 2.4.4.1)。
- (6) 最後に[OK]ボタンを押下して、すべてのダイアログを閉じます。

注意：

- デバッグプローブの設定は、デバッグプローブ (J-Link) を PC に接続した状態で行ってください。
- 「2.4.1 ソフトウェアのセットアップ」に記載の setup-debugger-jlink.bat を実行済みの場合、サンプルソフトウェア内のすべてのプロジェクトに対してデバッグプローブ (J-Link) の設定が適用されています。ただし、デバッグプローブが正常に認識しない場合は、上記手順に従って設定を確認してください。

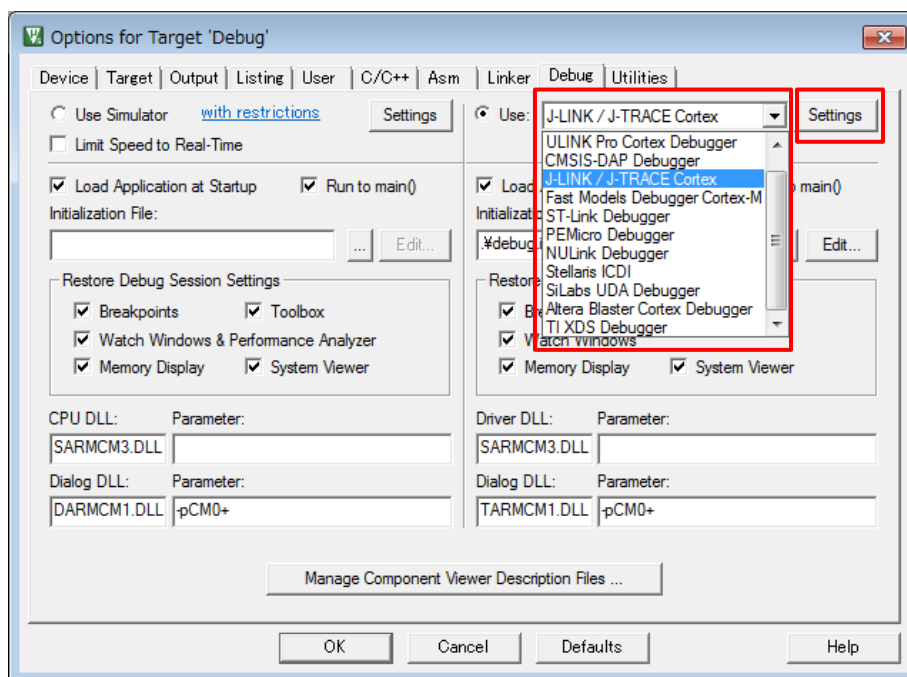


図 2.4.4.1 デバッグプローブの選択

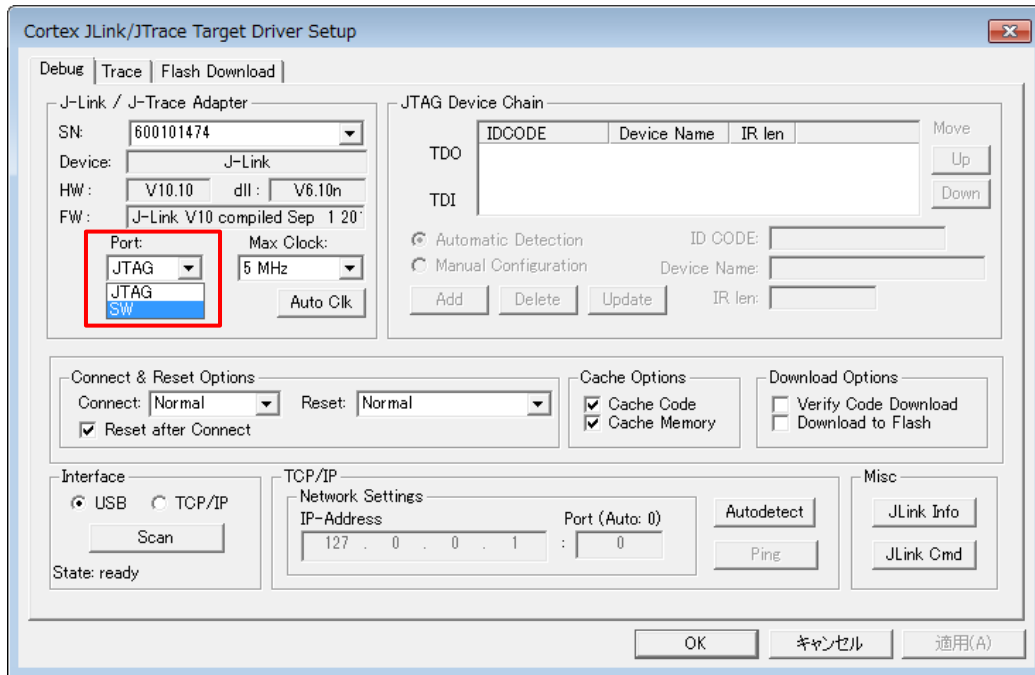


図 2.4.4.2 J-Link ドライバの設定

2. サンプルソフトウェアの操作

2.4.5 フラッシュローダの設定

アクティブプロジェクトのビルド構成が DebugFlash の場合、内蔵 Flash メモリにプログラムをロード (Flash プログラミング) するためにフラッシュローダを設定する必要があります。ビルド構成が Debug の場合は、内蔵 RAM にプログラムをロードして実行するためフラッシュローダの設定を解除する必要があります。

フラッシュローダの設定は、以下の手順で行います。

- ビルド構成が Debug の場合 (RAM 上のプログラムを実行)
 - (1) μVision メニューの [Project] > [Options for {project} - Target '{build configuration}'] を選択します。
 - (2) [Options for Target '{build configuration}'] ダイアログの [Debug] タブを選択します。
 - (3) [Initialize File] エディットボックス右横の [...] ボタンを押下し、debug.ini ファイルを選択します (図 2.4.5.1)。
 - (4) [Options for Target '{build configuration}'] ダイアログの [Utilities] タブを選択し、[Configure Flash Menu Command] 枠内の [Settings] ボタンを押下します。
 - (5) [Cortex JLink/JTrace Target Driver Setup]ダイアログの[Flash Download]タブを選択します。
 - (6) [Remove] ボタンを押下して、[Programming Algorithm]リスト上のフラッシュローダを全て削除します。
 - (7) [Download Function] 枠内の [Do not Erase]、[Reset and Run] のチェックを有効にし、それ以外のチェックを無効にします (図 2.4.5.2)。
- ビルド構成が DebugFlash の場合 (内蔵 Flash メモリのプログラムを実行)
 - (1) μVision メニューの [Project] > [Options for {project} - Target '{build configuration}'] を選択します。
 - (2) [Options for Target '{build configuration}'] ダイアログの [Debug] タブを選択します。
 - (3) [Initialize File] エディットボックスに何も設定せずに空白にします。
 - (4) [Options for Target '{build configuration}'] ダイアログの [Utilities] タブを選択し、[Configure Flash Menu Command] 枠内の [Settings] ボタンを押下します。
 - (5) [Cortex JLink/JTrace Target Driver Setup]ダイアログの[Flash Download]タブを選択します。
 - (6) [Remove] ボタンを押下して、[Programming Algorithm] リスト上のフラッシュローダを全て削除します。
 - (7) [Add] ボタンを押下して、[Add Flash Programming Algorithm] ダイアログを開きます。
 - (8) [Add Flash Programming Algorithm] ダイアログ上のリストからフラッシュローダ“S1C31W74int 512kB Flash”を選択し、[Add]ボタンを押下します。
 - (9) [Download Function] 枠内の [Erase Sectors]、[Program]、[Verify] のチェックを有効にし、それ以外のチェックを無効にします (図 2.4.5.3)。

注意： Debug ビルドを選択して実行する場合、プログラムは RAM にロードされるため内蔵 Flash メモリは更新されません。したがって、通常は Vector テーブルからロードされるプログラムカウンタおよびスタックレジスタの設定を、[Initialization File] オプションを使用して追加設定する必要があります。Debug ビルドを使用しているサンプルソフトウェアプロジェクトでは、debug.ini ファイルを使用して、これらの値を設定しています。DebugFlash ビルドのサンプルソフトウェアでは、[Initialization File] フィールドは何も設定せず空白のままにしてください。

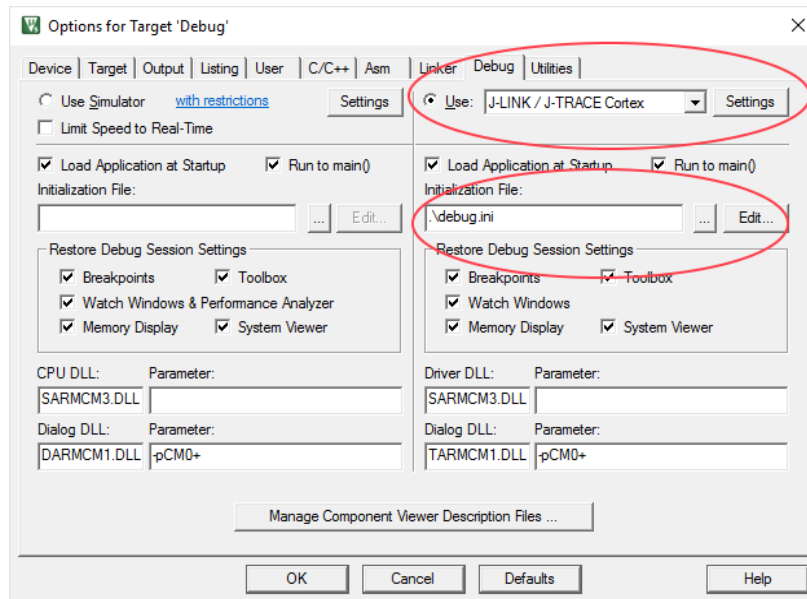


図 2.4.5.1 debug.ini ファイルの設定

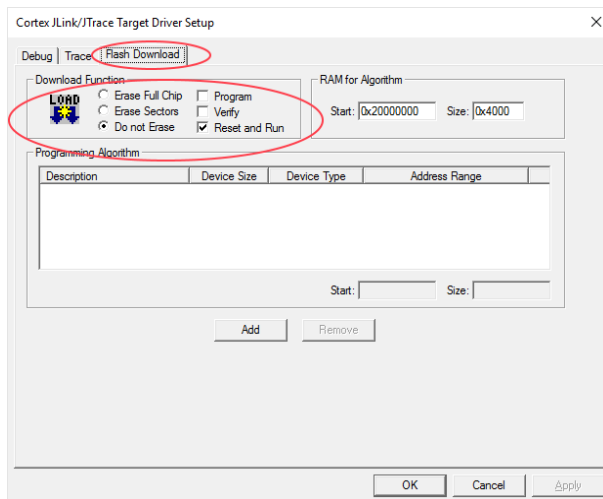


図 2.4.5.2 フラッシュローダの解除

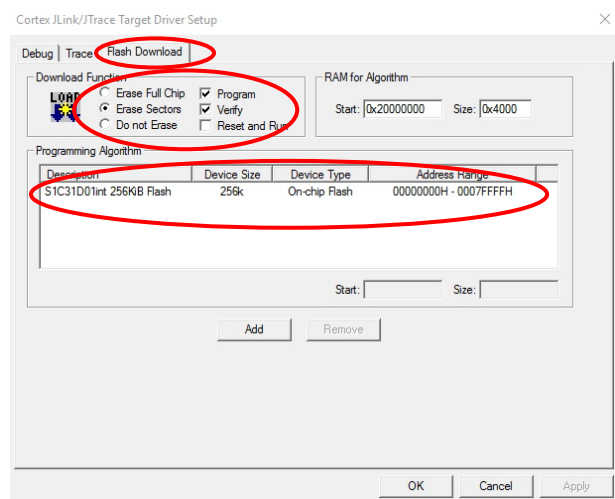


図 2.4.5.3 フラッシュローダの設定

2. サンプルソフトウェアの操作

2.4.6 プロジェクトのビルド

アクティブプロジェクトをビルドするには、μVision メニューの [Project] から [Build]、[Rebuild] の 2 つのビルドコマンドのいずれかを選択します (図 2.4.6.1)。

注意： ビルド中にリンカエラーが発生する場合は、sePeriphLibrary と seGraphicsLibrary の 2 つのライブラリプロジェクトがビルドされていない可能性があります。これらのライブラリプロジェクトをビルドしてから、再度、アクティブプロジェクトをビルドしてください。

また、μVision メニューの [Project] > [Batch Build] を選択することで、すべてのサンプルソフトウェアプロジェクトをまとめてビルドすることが可能です (図 2.4.6.1)。
[Batch Build] ダイアログの [Build]、[Rebuild] ボタンのいずれかをクリックすると、選択されたすべてのプロジェクトがビルドされます (図 2.4.6.2)。

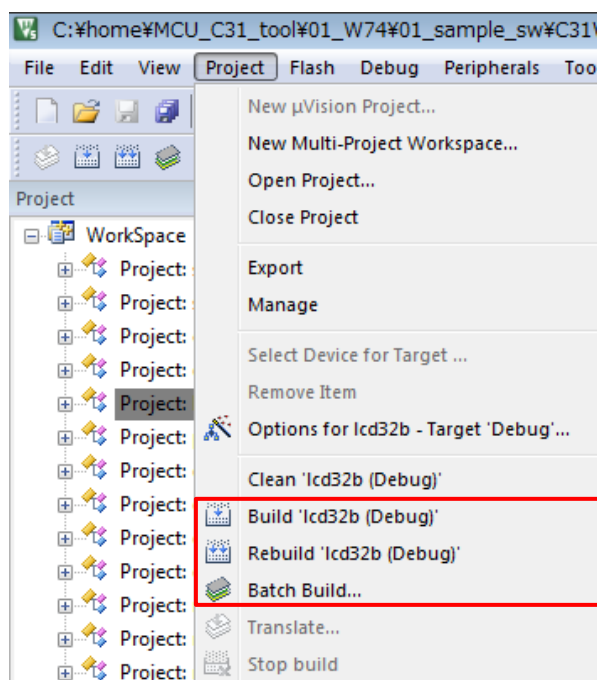


図 2.4.6.1 ビルドコマンド

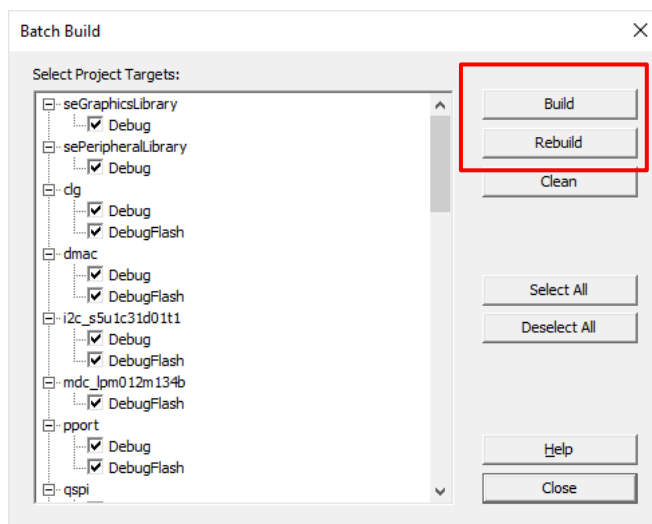


図 2.4.6.2 バッチビルド

2.4.7 プロジェクトのダウンロードおよびデバッグ

ビルドに成功したら、アクティブプロジェクトをターゲットボードにダウンロードします。アクティブプロジェクトをダウンロードするには、 μ Visionメニューの [Flash] > [Download] を選択します（図2.4.7.1）。プロジェクトがDebugビルドの場合、プログラムイメージは内蔵RAMにロードされます。DebugFlashビルドの場合、プログラムイメージは内蔵Flashメモリにロードされます。

ターゲットボードにダウンロードしたプログラムイメージのデバッグを開始するには、 μ Visionメニューの [Debug] > [Start/Stop Debug Session] を選択します（図2.4.7.2）。

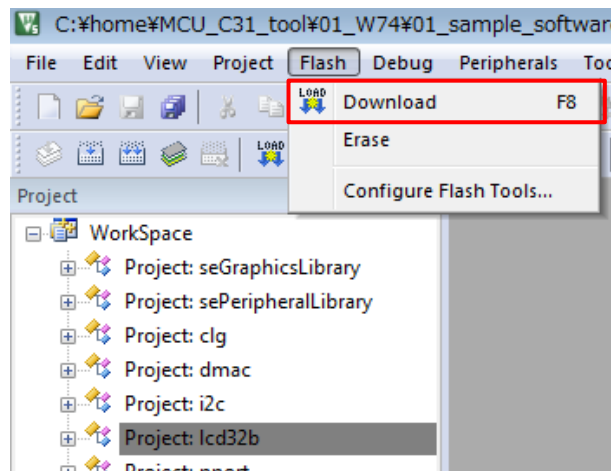


図 2.4.7.1 ダウンロード

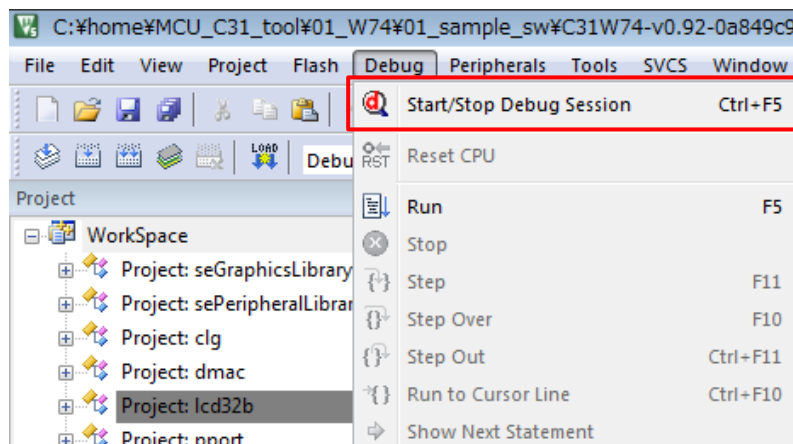


図 2.4.7.2 デバッグ

3. サンプルソフトウェアの詳細

3. サンプルソフトウェアの詳細

いくつかのサンプルプロジェクトでは、ハードウェアのセットアップが必要です。ハードウェアセットアップに関しては S5U1C31W74T1 のマニュアルも併せて参照してください。

3.1 クロックジェネレータ(CLG)

CLG は、クロックソースを制御し、CPU や周辺回路へのクロック供給を管理するクロックジェネレータです。

このサンプルプロジェクトでは、OSC の起動、ウェイクアップクロックやスリープモードの設定など、様々な CLG 機能を実行します。

動作

1. CLG を初期化します。
2. IOSC のオートトリミングを実行します。
3. 様々なクロックで実行して、SLEEP または HALT の状態を確認します。
4. 様々なクロックを実行して、ウェイクアップの状態を確認します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
CLG Initialization ok

CLG IOSC Auto-trimming ok

Halt (OSC3)      -actual- seCLG_OSC3
Wake up (OSC3)  -actual- seCLG_OSC3

Sleep (OSC1)     -actual- seCLG_OSC1
Wake up (OSC3)  -actual- seCLG_OSC3
Exit
```

3.2 DMAコントローラ(DMAC)

DMAC はメモリと周辺回路間のデータ転送に使用する回路です。

このサンプルプログラムでは、以下に示す様々なデータ転送を行います。

- DMAC を使用したメモリ間データ転送
- DMAC を使用したメモリから周辺回路へのデータ転送
- DMAC を使用した周辺回路からメモリへのデータ転送
- DMAC を使用した周辺回路からメモリへの DMA 転送およびメモリから周辺回路への DMA 転送の同時発生

DMAC を使用するデータ転送の他の例については、QSPI_DMA サンプルプログラムを参照してください。

ハードウェアセットアップ

UART DMA 転送デモを実行する場合は、UART 用 USB アダプタを使用して S5U1C31W74T1 評価ボードの UART と PC を接続してください。(図 2.2.2.1、図 2.2.2.2 参照)

動作

例 1: メモリ間 DMA 転送

- RAM 内のデータバッファの内容を、RAM 内の別のバッファにバイト単位で転送するように、DMAC Ch.0 を設定します。
- RAM 内のデータバッファの内容を、RAM 内の別のバッファにハーフワード単位で転送するように、DMAC Ch.1 を設定します。
- RAM 内のデータバッファの内容を、RAM 内の別のバッファにワード単位で転送するように、DMAC Ch.2 を設定します。
- ソフトウェアトリガにより転送を開始します。

このサンプルプログラムでは、

1. NVIC の DMAC 割り込みを無効にしています。
2. 使用する DMAC チャンネルの転送を有効にしています。
3. 転送元/転送先アドレスのインクリメントを有効にしています。
4. 転送は DMACSWREQ レジスタビットのセットにより開始します。
5. 転送終了時に DMA 転送完了割り込み要因が発生します。
6. 割り込み要因発生後、転送回数の読み出し値は 0 になります。
7. 割り込み要因発生後、DMA 転送完了割り込みフラグをクリアします。
8. 転送元と転送先のバッファの内容を比較し、全データが正常に転送されたことをチェックします。

例 2: メモリから周辺回路への DMA 転送

- RAM 内のデータバッファの内容を SNDA のデータレジスタに転送するように、DMAC Ch.0 を設定します。
- メロディ演奏するために SNDA を設定します。
- SNDA のトリガにより転送を開始します。

このサンプルプログラムでは、

1. NVIC の DMAC 割り込みを無効にしています。

3. サンプルソフトウェアの詳細

2. 使用する DMAC チャンネルの転送を有効にしています。
3. 使用する DMAC チャンネルに対する周辺回路からの DMA 転送要求マスクを解除しています。
4. 転送元アドレスのインクリメントを有効にしています。
5. 転送先アドレスのインクリメントを無効にしています。
6. SNDA のサウンドバッファエンプティをトリガとして転送を開始します。
7. 転送終了時に DMA 転送完了割り込み要因が発生します。
8. 割り込み要因発生後、転送回数の読み出し値は 0 になります。
9. 割り込み要因発生後、DMA 転送完了割り込みフラグをクリアします。
10. 演奏されるメロディにより、正常に転送されたことを確認します。

例 3: 周辺回路からメモリへの DMA 転送

- UART データレジスタの内容をメモリに転送するように、DMAC Ch.0 を設定します。
- UART のボーレートを 115,200bps に設定します。また、受信バッファフル要因によって DMA 転送を開始するように設定します。
- PC 上のターミナルプログラムから 8 文字を入力したことをトリガとして、データ転送を開始します。

このサンプルプログラムでは、

1. NVIC の DMAC 割り込みを無効にしています。
2. 使用する DMAC チャンネルの転送を有効にしています。
3. 使用する DMAC チャンネルに対する周辺回路からの DMA 転送要求マスクを解除しています。
4. 転送元アドレスのインクリメントを無効にしています。
5. 転送先アドレスのインクリメントを有効にしています。
6. 転送は UART2 の受信バッファが満杯になることにより開始します。
7. 転送終了時に DMA 転送完了割り込み要因が発生します。
8. 割り込み要因発生後、DMA 転送モードの読み出し値は 0x0(停止)になります。
9. 割り込み要因発生後、DMA 転送完了割り込みフラグをクリアします。
10. メモリから周辺回路への DMA 転送(例 2 と同様)を使用して、ターミナルウィンドウに受信文字列を返送します。
11. ターミナルウィンドウに表示された文字列を見て、正常に転送されたことを確認します。

例 4: 周辺回路からメモリへの DMA 転送およびメモリから周辺回路への DMA 転送の同時発生

- T16B Ch.0 のコンパレータ/キャプチャ回路 0 をコンパレータモードに設定します。
- T16B Ch.0 のコンパレータ/キャプチャ回路 1 をキャプチャモードに設定します。
- メモリデータを T16B Ch.0 のコンパレータ/キャプチャ回路 0 データレジスタに転送するように、DMAC Ch.0 を設定します。
- T16B Ch.0 のコンパレータ/キャプチャ回路 1 データレジスタの内容をメモリに転送するように、DMAC Ch.1 を設定します。

このサンプルプログラムでは、

1. NVIC の DMAC 割り込みを有効にしています。
2. 2 つの DMAC チャンネルの転送を有効にしています。
3. 使用する DMAC チャンネルに対する周辺回路からの DMA 転送要求マスクを解除しています。
4. メモリから周辺回路への転送は、T16B Ch.0 コンパレータ回路 0 からのコンペア割り込み要因により開始します。
5. 転送終了時に DMAC Ch.0 から DMA 転送完了割り込みが発生します。
6. 割り込み発生後、DMAC 割り込み処理ルーチンの中で、ソフトウェア完了フラグをセットします。

7. DMA 転送完了割り込みフラグをクリアします。
8. 周辺回路からメモリへの転送は、T16B Ch.0 キャプチャ回路 1 からのキャプチャ割り込み要因により開始します。
9. 転送終了時に DMAC Ch.1 から DMA 転送完了割り込みが発生します。
10. 割り込み発生後、DMAC 割り込み処理ルーチンの中で、ソフトウェア完了フラグをセットします。
11. DMA 転送完了割り込みフラグをクリアします。
12. 両方の転送が完了したことをソフトウェア完了フラグで確認します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
DMA Initialising
DMA Test: Memory
DMA Test: SNDA
DMA Test: UART2
-   Type 8 characters...
12345678 - 8 characters have been sent back to UART
DMA Test: T16B

Exit
```

3. サンプルソフトウェアの詳細

3.3 I2C(I2C)

I2Cは、I²Cバスインタフェースのサブセットです。

このサンプルプログラムでは、I2Cをマスタモードに設定してEEPROMとの通信を行います。このサンプルプログラムを実行するためには、S5U1C31W74T1評価ボードに接続するEEPROM 24FC512メモリを必要とします。

ハードウェアセットアップ

I2C サンプルプログラムをダウンロードした S5U1C31W74T1 評価ボードと EEPROM 24FC512 メモリを、図 3.3.1 に示すように接続します。

S1C31W74		24FC512	
[マスタ]		[スレーブ]	
P13	SDAn	-----><-----	SDA
P12	SCLn	----->>-----	SCL

図 3.3.1 マスタ-スレーブ間の結線

動作

このサンプルプログラムは、スレーブアドレス 0x50 で EEPROM を使用して I2C Ch.0 の通信を行います。

1. I2C Ch.0 をマスタモードに設定します。
2. EEPROM に ASCII 文字列を書き込みます。
3. EEPROM からデータを読み出し、書き込んだデータと比較します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
Testing I2C_0 module with EEPROM at 0x50
Write data: Test 0 1 2 4
Read data: Test 0 1 2 4
Status: OK
Exit
```


3.4 LCDドライバ(LCD32B)

LCD32B は LCD パネルを駆動する LCD ドライバです。

このサンプルプログラムでは、72x32 ドットマトリクス LCD ディスプレイに様々なパターンを表示させます。

動作

1. LCD32B を初期化します。
2. LCD パネルにいくつかのパターンを表示します。
3. コントラストを変えるデモを行います。
4. フォントを変えてテキストを表示します。
5. フレームバッファ割り込みを検出します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
External voltage mode selected
Set panel contrast high (dark)
Show all zeros, area0
Show all ones, area0
Show both ones and zeros, area0
Show checkboard pattern, area0
Set display inverse state NORMAL
Hit '+' to increase and '-' to decrease contrast, 'q' to exit
Set panel contrast 0
Set panel contrast 1
Set panel contrast 2
Set panel contrast 3
Set panel contrast 4
Set panel contrast 5
Set panel contrast 6
Set panel contrast 7
Set panel contrast 8
Set panel contrast 9
Set panel contrast 10
Set panel contrast 11
Set panel contrast 10
Set panel contrast 9
Set panel contrast 8
Set panel contrast 7
Set panel contrast 6
Set panel contrast 5
Set panel contrast 4
Set panel contrast 3
Set panel contrast 2
Set panel contrast 1
Set panel contrast 0
Set panel contrast 1
Set panel contrast 2
Set panel contrast 3
Set panel contrast 4
Set panel contrast 5
Set panel contrast 6
```

3. サンプルソフトウェアの詳細

```
LCD32B waiting for interrupt....  
CLK -- 0x0150  
CTL -- 0x0003  
TIM1 -- 0x011f  
TIM2 -- 0x0300  
PWR -- 0x8615  
DSP -- 0x0075  
COMC0 -- 0xffff  
COMC1 -- 0xffff  
INTF -- 0x0001  
INTE -- 0x0000  
TEST -- 0x0000  
Exit
```

3.5 入出力ポート(PPORT)

PPORT は S1C31W74 の入出力ポート (GPIO) を制御する回路です。

このサンプルプログラムでは、出力ポートとして設定した P01 端子と、入力ポートとして設定した P02 端子を接続し、P01 と P02 の接続をチェックするための様々なテストを実行します。

ハードウェアセットアップ

このサンプルプログラムを実行する場合は、P01 端子と P02 端子を接続してください。

(TH2) P01----->>-----P02 (TH3)

動作

1. OSC1 の発振を開始させ、システムクロックを IOSC から OSC1 に切り換えます。その後、IOSC を停止します。
2. 入出力ポートを初期化します。
3. P01 ポートを出力に設定し、LOW レベルを出力させます。
4. P02 ポートを入力に設定すると共に、LOW から HIGH への入力の変化により割り込みが発生するように設定します。
5. P01 ポートから HIGH レベルを出力させます。
6. P02 ポートの割り込みが発生後、P02 ポートの入力が HIGH レベルであることを確認します。
7. HIGH から LOW への入力の変化により割り込みが発生するように、P02 ポートを設定します。
8. P01 ポートから LOW レベルを出力させます。
9. P02 ポートの割り込みが発生後、P02 ポートの入力が LOW レベルであることを確認します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
P01 output state: Low
P02 input state: Low
P02 input waits for RISING edge interrupt
P01 output is going to HIGH...
Interrupt on P02 occurred
P01 output state: High
P02 input state: High

P01 output state: High
P02 input state: High
P02 input waits for FALLING edge interrupt
P01 output is going to LOW...
Interrupt on P02 occurred
P01 output state: Low
P02 input state: Low

Exit
```

3. サンプルソフトウェアの詳細

3.6 同期式クワッドシリアルインタフェース(QSPI)

QSPI は、マスタおよびスレーブの両モードに対応した同期式クワッドシリアルインタフェースです。

このサンプルプログラムでは、QSPI をマスタモードに設定して外部シリアルフラッシュメモリとの通信を行います。

ハードウェアセットアップ

このサンプルプログラムでは、S5U1C31W74T1 評価ボードに搭載された Micron フラッシュメモリを使用します。

動作

1. QSPI をマスタモードおよび以下の設定に初期化します。
 - データ長: 8 ビット
 - データフォーマット: MSB 先頭
 - クロックジェネレータとして 16 ビットタイマ(T16) Ch.2 を使用
2. クロック周波数を 10,000,000bps に設定します。
3. QSPI をシングルモードで起動します。
4. 外部フラッシュメモリのレジスタが正しく読み出せるかチェックします。
5. フラッシュ ID を読み出します。
6. ここまでのフラッシュ動作が正常に終了した場合は、3 つのモードを順次変更し以下の動作を行います。
 - クワッドモードに設定して、セクタ消去、セクタ書き込み、セクタ読み出しとデータ比較、の順で実行します。
 - デュアルモードに設定して、セクタ消去、セクタ書き込み、セクタ読み出しとデータ比較、の順で実行します。
 - シングルモードに設定して、セクタ消去、セクタ書き込み、セクタ読み出しとデータ比較、の順で実行します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
Get bus speed 77519
Set bus speed 10000000
Get bus speed 10000000
QSPI Start: OK
Read external flash register in SINGLE mode: OK
Manufacture ID: 20h, Device ID: ba18h

Set external flash in QUAD mode: OK
Set QSPI in QUAD mode.
Read external flash register in QUAD mode: OK
Erase flash sector in QUAD mode: OK
Program flash in QUAD mode: OK
Read flash in QUAD mode: OK
Compare R/W data in QUAD mode: OK

Set external flash in DUAL mode: OK
Set QSPI in DUAL mode.
Read external flash register in DUAL mode: OK
```

```
Erase flash sector in DUAL mode: OK
Program flash in DUAL mode: OK
Read flash in DUAL mode: OK
Compare R/W data in DUAL mode: OK

Set external flash in SINGLE mode: OK
Set QSPI in SINGLE mode.
Read external flash register in SINGLE mode: OK
Erase flash sector in SINGLE mode: OK
Program flash in SINGLE mode: OK
Read flash in SINGLE mode: OK
Compare R/W data in SINGLE mode: OK
Exit
```

3. サンプルソフトウェアの詳細

3.7 同期式クワッドシリアルインタフェース with DMA (QSPI_DMA)

QSPI は、マスタおよびスレーブの両モードに対応した同期式クワッドシリアルインタフェースです。

このサンプルプログラムでは、QSPI をマスタモードに設定し、DMA を使用して外部シリアルフラッシュメモリとの通信を行います。

ハードウェアセットアップ

このサンプルプログラムは、S5U1C31W74T1 評価ボード搭載の Micron フラッシュメモリを使用します。

動作

1. QSPI をマスタモードおよび以下の設定に初期化します。
 - データ長: 8 ビット
 - データフォーマット: MSB 先頭
 - クロックジェネレータとして 16 ビットタイマ(T16) Ch.2 を使用
2. クロック周波数を 10,000,000bps に設定します。
3. DMA 転送用のデータストラクチャを作成し、DMAC を初期化します。
4. QSPI をシングルモードで起動します。
5. 外部フラッシュメモリのレジスタが正しく読み出せるかチェックします。
6. フラッシュ ID を読み出します。
7. ここまでのフラッシュ動作が正常に終了した場合は、以下の動作を行います。
 - セクタ消去
 - レジスタアクセスモードで、DMA 転送を使用したセクタ書き込み
 - レジスタアクセスモードで、DMA 転送を使用したセクタ読み出し後ベリファイ
8. QSPI をメモリマップドアクセスモードに設定します。
9. メモリマップドアクセスモードで、DMA 転送を使用したセクタ読み出しを行い、更に書き込み値とのベリファイを行います。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
Set bus speed 10000000
QSPI Start: OK
Set external flash in SINGLE mode: OK
Erase flash sector in SINGLE mode: OK
Program flash using DMA in Register Access SINGLE mode: OK
Read flash using DMA in Register Access SINGLE mode: OK
Compare R/W data in Register Access SINGLE mode: OK
Set external flash in QUAD mode: OK
QSPI Start: OK
Read flash using DMA in Memory Mapped QUAD mode: OK
Compare R/W data using DMA in Memory Mapped QUAD mode: OK
Exit
```

3.8 同期式クワッドシリアルインタフェース マスタ(QSPI_MASTER)

QSPI は、マスタおよびスレーブの両モードに対応した同期式クワッドシリアルインタフェースです。
このサンプルプログラムでは、QSPI をマスタモードに設定して通信を行います。

ハードウェアセットアップ

1. S5U1C31W74T1 評価ボードには、シリアルフラッシュデバイスが搭載されており、QSPI インタフェースに接続されています。QSPI マスタ/スレーブのサンプルプログラムを実行するには、ボード上の R157、R158、R159、R160、R163、R168 の 0 オーム抵抗を取り除いて、シリアルフラッシュデバイスが QSPI インタフェースと絶縁されている必要があります。
注意： QSPI マスタ/スレーブサンプルプログラムを評価した後は、これらの 0Ω 抵抗の接続を元に戻してください。
2. QSPI マスタと QSPI スレーブサンプルプログラムをそれぞれインストールした 2 枚の S5U1C31W74T1 評価ボードを用意して、図 3.8.1 または図 3.8.2 に示す端子間を接続します。
3. 最初にスレーブ用サンプルプログラムを起動し、その後にマスタ用サンプルプログラムを起動してください。

デュアルまたはクワッドモード					
[マスタ]			[スレーブ]		
TH14	QSDIO _n 3	-----><-----	QSDIO _n 3	TH14	
TH13	QSDIO _n 2	-----><-----	QSDIO _n 2	TH13	
TH12	QSDIO _n 1	-----><-----	QSDIO _n 1	TH12	
TH11	QSDIO _n 0	-----><-----	QSDIO _n 0	TH11	
TH10	QSPICLK _n	----->>-----	QSPICLK _n	TH10	
TH15	#QSPISS _n	----->>-----	#QSPISS _n	TH15	
TH1	P00	----->>-----	P00	TH1	
TH9	GND	-----><-----	GND	TH9	

図 3.8.1 マスタ-スレーブ間の結線（デュアルまたはクワッドモード）

* THxについては、S5U1C31W74T1 マニュアル記載の回路図を参照してください。

3. サンプルソフトウェアの詳細

シングルモード					
[マスタ]			[スレーブ]		
TH11	QSDIO _n 0	----->>-----	QSDIO _n 1	TH12	
TH12	QSDIO _n 1	-----<<-----	QSDIO _n 0	TH11	
TH10	QSPICLK _n	----->>-----	QSPICLK _n	TH10	
TH15	#QSPISS _n	----->>-----	#QSPISS _n	TH15	
TH1	P00	----->>-----	P00	TH1	
TH9	GND	-----><-----	GND	TH9	

図 3.8.2 マスタ-スレーブ間の結線（シングルモード）

* THxについては、S5U1C31W74T1マニュアル記載の回路図を参照してください。

動作

1. QSPI をマスタモードで以下の設定に初期化します。
 - データ長: 8 ビット
 - データフォーマット: MSB 先頭
 - クロックジェネレータとして 16 ビットタイマ(T16) Ch.2 を使用
2. クロック周波数を 300,000bps に設定します。
3. スレーブへの Read/Write コマンドを設定するために P00 ポートを汎用出力に設定します。
4. “BUF_SIZE”で指定されるバイトサイズのランダムデータをスレーブに送信します。
5. スレーブから送られる“BUF_SIZE”バイトのデータを受信します。
6. 受信データと送信データを比較して終了します。

出力例

For Double/Quad Mode:

```
CPU clock- seCLG_IOSC (20000000)
Get bus speed 123456
Set bus speed 100000
Get bus speed 100000
Set QSPI_0 in DUAL mode.
- OK(1), NG(0)
- OK(2), NG(0)
- OK(3), NG(0)
- OK(4), NG(0)
- OK(5), NG(0)
- OK(6), NG(0)
- OK(7), NG(0)
- OK(8), NG(0)
- OK(9), NG(0)
- OK(10), NG(0)
Set QSPI_0 in QUAD mode.
```


- OK(11), NG(0)
- OK(12), NG(0)
- OK(13), NG(0)
- OK(14), NG(0)
- OK(15), NG(0)
- OK(16), NG(0)
- OK(17), NG(0)
- OK(18), NG(0)
- OK(19), NG(0)
- OK(20), NG(0)

For Single Mode:

```
CPU clock- seCLG_IOSC (20000000)
Get bus speed 123456
Set bus speed 100000
Get bus speed 100000
Set QSPI_0 in SINGLE mode.
- OK(1), NG(0)
- OK(2), NG(0)
- OK(3), NG(0)
- OK(4), NG(0)
- OK(5), NG(0)
- OK(6), NG(0)
- OK(7), NG(0)
- OK(8), NG(0)
- OK(9), NG(0)
- OK(10), NG(0)
```

3. サンプルソフトウェアの詳細

3.9 同期式クワッドシリアルインタフェース スレーブ(QSPI_SLAVE)

QSPI は、マスタおよびスレーブの両モードに対応した同期式クワッドシリアルインタフェースです。
このサンプルプログラムでは、QSPI をスレーブモードに設定して通信を行います。

ハードウェアセットアップ

1. S5U1C31W74T1 評価ボードには、シリアルフラッシュデバイスが搭載されており、QSPI インタフェースに接続されています。QSPI マスタ/スレーブのサンプルプログラムを実行するには、ボード上の R157、R158、R159、R160、R163、R168 の 0 オーム抵抗を取り除いて、シリアルフラッシュデバイスが QSPI インタフェースと絶縁されている必要があります。
注意： QSPI マスタ/スレーブサンプルプログラムを評価した後は、これらの 0Ω 抵抗の接続を元に戻してください。
2. QSPI マスタと QSPI スレーブサンプルプログラムをそれぞれインストールした 2 枚の S5U1C31W74T1 評価ボードを用意して、図 3.9.1 または図 3.9.2 に示す端子間を接続します。
3. 最初にスレーブ用サンプルプログラムを起動し、その後にマスタ用サンプルプログラムを起動してください。

デュアルまたはクワッドモード					
[マスタ]			[スレーブ]		
TH14	QSDIO _n 3	-----><-----	QSDIO _n 3	TH14	
TH13	QSDIO _n 2	-----><-----	QSDIO _n 2	TH13	
TH12	QSDIO _n 1	-----><-----	QSDIO _n 1	TH12	
TH11	QSDIO _n 0	-----><-----	QSDIO _n 0	TH11	
TH10	QSPICLK _n	----->>-----	QSPICLK _n	TH10	
TH15	#QSPISS _n	----->>-----	#QSPISS _n	TH15	
TH1	P00	----->>-----	P00	TH1	
TH9	GND	-----><-----	GND	TH9	

図 3.9.1 マスタ-スレーブ間の結線（デュアルまたはクワッドモード）

* TH_xについては、S5U1C31W74T1 マニュアル記載の回路図を参照してください。

シングルモード					
[マスタ]			[スレーブ]		
TH11	QSDIO _n 0	----->>-----	QSDIO _n 1	TH12	
TH12	QSDIO _n 1	-----<<-----	QSDIO _n 0	TH11	
TH10	QSPICLK _n	----->>-----	QSPICLK _n	TH10	
TH15	#QSPISS _n	----->>-----	#QSPISS _n	TH15	
TH1	P00	----->>-----	P00	TH1	
TH9	GND	-----><-----	GND	TH9	

図 3.9.2 マスタ-スレーブ間の結線（シングルモード）

* TH_xについては、S5U1C31W74T1マニュアル記載の回路図を参照してください。

動作

1. QSPI をスレーブモードおよび以下の設定に初期化します。
 - データ長: 8 ビット
 - データフォーマット: MSB 先頭
2. P00 ポートを汎用入力に設定し、通信開始検出用に立ち上がりエッジでの割り込みを許可します。
3. マスタから送られる、“BUF_SIZE”で指定されるバイトサイズのランダムデータを受信します。
4. “BUF_SIZE”バイトのデータをマスタに送信します。

出力例

なし

3. サンプルソフトウェアの詳細

3.10 IRリモートコントローラ(REMC2)

REMC2 は赤外線リモコン出力信号を生成する回路です。

このサンプルプログラムでは、REMC2 から様々な IR パルスを出力します。

ハードウェアセットアップ

このサンプルプログラムは、S5U1C31W74T1 評価ボード用です。 接続については、S5U1C31W74T1 のマニュアルを参照してください。

動作

このサンプルプログラムは、NEC フォーマットのリモコン信号を出力します。

1. REMC2 は、APLEN(データ信号のデューティ)と DBLEN(データ信号の周期)の 2 つのパラメータを使用してデータの波形を決定します。APLEN と DBLEN はそれぞれ 34 個のフレーム信号と 2 個のリピート信号を定義します。初期化の際に、この APLEN と DBLEN の 2 つによりリーダー、カスタムコード、リピート信号が設定されます。これらの信号は SW100～SW103 の押下により選択される全ての動作に共通です。データは SW100～SW103 の選択により変化し、スイッチ押下による割り込み発生時に設定されます。
2. 初期化処理では、更に SW100～SW103 の設定、REMO 出力のポートの割り当て、T16 Ch.0 の初期設定が行われます。この後、CPU は HALT モードになり、SW100～SW103 の押下による割り込みの発生を待ちます。
3. 割り込みが発生すると、データ信号を生成するための APLEN と DBLEN の値が設定されます。REMC2 はコンペアバッファを使用するため、初期化終了後に起動させます。また、T16 Ch.0 も起動させます。T16 Ch.0 は約 108ms 後に割り込みを発生します。
4. REMC2 からコンペア DB 割り込みが発生した時点で、続く APLEN と DBLEN の値をレジスタに書き込みます。最後の 1 つ前のデータを書き込む際は、REMC2 をワンショットモードに変更し、ひげ状の信号が出力されることを防ぎます。最後のデータ出力の後、REMC2 の動作を停止します。
5. T16 Ch.0 割り込みが発生した時点で、SW100～SW103 の同じスイッチが押されていないかチェックします。同じスイッチが押されていた場合は、REMC2 のレジスタ値をリピート波形が生成されるように設定し、REMC2 を初期化の後、再度起動します。同じスイッチが押されていない場合は、T16 Ch.0 の動作を停止します。

3.11 R/F変換器(RFC)

RFC は CR 発振方式の A/D 変換器(R/F 変換器)です。

このサンプルプログラムでは、RFC を動作させて計測を行います。

ハードウェアセットアップ

このサンプルプログラムを実行する前に、S5U1C31W74T1 評価ボードの J100 DIP ソケットの以下の端子にセンサ、抵抗およびキャパシタを接続してください。

SENA0 (P05) -----<抵抗性センサ(DCバイアス)>-----RFIN0 (P07)

REF0 (P06)----- <基準抵抗>-----RFIN0 (P07)

RFIN0 (P07) -----<基準抵抗および基準キャパシタ>---GND

動作

1. RFC を初期化します。
2. RFC を動作させ、計測カウンタの値を取得します。
3. 変換の状況とカウンタ値を表示します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
Starting R/F Conversion operation...
R/F Converting operation finished with OK status, count = 253430
Exit
```

3. サンプルソフトウェアの詳細

3.12 リアルタイムクロック(RTCA)

RTCA はパーペチュアルカレンダー機能を備えたリアルタイムクロックです。

このサンプルプログラムでは、時刻/日付設定、ストップウォッチ、アラーム、トリミングなどのリアルタイムクロックの様々な機能を実行します。

動作

1. RTCA を初期化します。
2. RTCA をスタートします。
3. 論理緩急補正値を算出します。
4. 日時を設定し、その値を読み出します。
5. アラームを設定し、CPU を SLEEP 状態にします。アラーム割り込みにより SLEEP 状態が解除されます。
6. 1秒タイマをスタートさせ、論理緩急を実行します。
7. ストップウォッチの動作を確認します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
RTCA TRM bits 0x43
RTCA Set hour 4 (PM), minute 17, second 0
RTCA Get hour 4 (PM), minute 17, second 0
RTCA Set year 15, month 2, day 5, Thursday
RTCA Get year, month, day. 2015, 2, 5, Thursday
RTCA Set Alarm hour 4 (PM), minute 17, second 5
RTCA waiting for alarm....
RTCA Alarm occurred : hour 4(PM), minute 17, second 5
RTCA Start Trimming
RTCA waiting for timer....
RTCA Trimming performed : hour 4(PM), minute 17, second 11
RTCA StopWatch (start) : hour 4(PM), minute 17, second 11
RTCA sleep 5 seconds waiting for Stop....
RTCA interrupts disabled for count reading
SysTick 5000
StopWatchHW: 5.14
Exit
```

3.13 サウンドジェネレータ(SNDA)

SNDA はメロディおよびブザー信号を発生するサウンドジェネレータです。

このサンプルプログラムでは、様々なブザーやメロディのサウンド出力を行います。

動作

1. OSC1 の発振を開始させ、SNDA を初期化します。
2. SNDA をノーマルブザーモードに設定し、5 秒間ブザー出力を行います。
3. SNDA をワンショットブザーモードに設定し、250ms のブザー出力を行います。
4. SNDA をメロディモードに設定し、サウンド出力を行います。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
Start SNDA in normal buzzer mode for 5 sec
Start SNDA in one-shot buzzer mode, 250ms
Start SNDA in melody mode
Start SNDA playing of 'Postoj Parovoz' music
Exit
```

3. サンプルソフトウェアの詳細

3.14 同期式シリアルインタフェース マスタ(SPIA_MASTER)

SPIA はマスタおよびスレーブの両モードに対応した同期式シリアルインタフェースです。
このサンプルプログラムでは、SPIA をマスタモードに設定してデータ通信を行います。

ハードウェアセットアップ

1. SPIA マスタと SPIA スレーブのサンプルプログラムをそれぞれインストールした 2 枚の S5U1C31W74T1 評価ボードを用意して、図 3.14.1 に示す端子間を接続します。
2. 最初にスレーブ用サンプルプログラムを起動し、その後にマスタ用サンプルプログラムを起動してください。

[マスタ]			[スレーブ]		
TH1	SDIn	-----<<-----	SDOn	TH2	
TH2	SDOn	----->>-----	SDIn	TH1	
TH3	SPICLK _n	----->>-----	SPICLK _n	TH3	
TH4	P03	----->>-----	#SPISS _n	TH4	
TH7	P27	----->>-----	P27	TH7	
TH9	GND	-----><-----	GND	TH9	

図 3.14.1 マスタ-スレーブ間の結線

* TH_xについては、S5U1C31W74T1 マニュアル記載の回路図を参照してください。

動作

1. SPIA をマスタモードおよび以下の設定に初期化します。
 - データ長: 8 ビット
 - データフォーマット: MSB 先頭
 - クロックジェネレータとして 16 ビットタイマ(T16) Ch.1 を使用
2. クロックを 1,000,000 に設定します。
3. スレーブへの Read/Write コマンドを設定するために P27 ポートを汎用出力に設定します。
4. “BUF_SIZE”で指定されるバイトサイズのランダムデータをスレーブに送信します。
5. スレーブから送られる“BUF_SIZE”バイトのデータを受信します。
6. 受信データと送信データを比較して終了します。

出力例

```
-CPU clock- seCLG_IOSCL (20000000)
Set bus speed 1000000
Get bus speed 1000000
OK(1), NG(0)
OK(2), NG(0)
```


OK(3), NG(0)
OK(4), NG(0)
OK(5), NG(0)
OK(6), NG(0)

3. サンプルソフトウェアの詳細

3.15 同期式シリアルインタフェース スレーブ(SPIA_SLAVE)

SPIA はマスタおよびスレーブの両モードに対応した同期式シリアルインタフェースです。

このサンプルプログラムで、SPIA をスレーブモードに設定してデータ通信を行います。

ハードウェアセットアップ

1. SPIA マスタと SPIA スレーブのサンプルプログラムをそれぞれインストールした 2 枚の S5U1C31W74T1 評価ボードを用意して、図 3.15.1 に示す端子間を接続します。
2. 最初にスレーブ用サンプルプログラムを起動し、その後にマスタ用サンプルプログラムを起動してください。

[マスタ]			[スレーブ]		
TH1	SDIn	-----<<-----	SDOn	TH2	
TH2	SDOn	----->>-----	SDIn	TH1	
TH3	SPICLK _n	----->>-----	SPICLK _n	TH3	
TH4	P03	----->>-----	#SPISS _n	TH4	
TH7	P27	----->>-----	P27	TH7	
TH9	GND	-----><-----	GND	TH9	

図 3.15.1 マスタ-スレーブ間の結線

* TH_xについては、S5U1C31W74T1 マニュアル記載の回路図を参照してください。

動作

1. SPIA をスレーブモードおよび以下の設定に初期化します。
 - データ長: 8 ビット
 - データフォーマット: MSB 先頭
2. マスタからの Read/Write コマンドを認識するために P27 ポートを汎用入力に設定します。
3. マスタからの“BUF_SIZE”で指定されるバイトサイズのランダムデータを受信します。
4. “BUF_SIZE”バイトのデータをマスタに送信します。

出力例

なし

3.16 電源電圧検出回路(SVD2)

SVD2 は VDD 端子に供給される電源電圧または外部端子に供給されている電源電圧を監視する電源電圧検出回路です。

このサンプルプログラムでは、VDD 電圧低下の検出を行います。

ハードウェアセットアップ

このサンプルプログラムを実行するには、S5U1C31W74T1 評価ボードの電源をデバッグプローブから供給されるように設定します。また、PC の USB ポートを、USB ケーブルを使用して S5U1C31W74T1 評価ボードの VDD に接続します。

動作

1. V_{DD} の電圧低下を監視するように SVD2 を設定します。
2. USB ケーブルの引き抜きを指示します。
3. USB ケーブルの引き抜きによる電圧低下を検出し、その結果を表示します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
SVD is configured to detect the voltage drop.
Please Disconnect USB cable.
Power voltage drop detected
Interrupt Low voltage occurred
IRQ handler called
Exit
```

3. サンプルソフトウェアの詳細

3.17 16 ビットタイマ(T16)

T16 は 16 ビットタイマです。

このサンプルプログラムでは、T16 タイマを定期的に中断し、5 秒間隔で発生した割り込みの数のカウントを行います。

動作

1. IOSC をタイマのクロックソースとして選択します。このサンプルプロジェクトでは、タイマクロックの精度を確保するため、IOSC クロックのオートトリミングを有効にしています。
2. T16 Ch.0 を初期化します。
3. T16 Ch.0 割り込みを設定します。
4. T16 割り込みをイネーブルにします。
5. 5 秒間に発生する割り込みの回数をカウントします。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
CLG IOSC Auto-trimming ok
IRQ interrupted: 1695 times during 5 sec
Exit
```

3.18 16ビットPWMタイマ(T16B)

T16B はコンパレータ/キャプチャ機能付きの 16 ビット PWM タイマです。

このサンプルプログラムでは、T16B の様々なコンパレータ/キャプチャ機能を実行します。

動作

1. T16B を以下のとおり設定し、タイマをスタートさせます。
 - モード: リピートアップカウントモード
 - MAX カウンタ値: 0x5000
 - コンパレータ/キャプチャ回路 0: コンパレータモード(コンペアバッファ: 0x2000)
 - コンパレータ/キャプチャ回路 1: キャプチャモード、割り込み回数設定(トリガ信号: LOW)
2. T16B をスタートさせ、CPU を HALT 状態にします。
3. T16B 割り込みにより CPU の HALT 状態が解除されます。
4. T16B 割り込みが発生するごとに、以下の割り込み回数をカウントします。
 - コンペア割り込み回数
 - キャプチャ割り込み回数
 - MAXカウンタ割り込み回数
 - カウンタゼロ割り込み回数
5. コンペア割り込みが発生した時点でキャプチャトリガ信号を HIGH に設定し直します。
6. MAX カウンタ割り込みが発生した時点でキャプチャトリガ信号を LOW に設定し直します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
Comparator Count = 0x2000
Timer Count = 0x0010
Max Count = 0x5000
Interrupt Max Count = 10
Interrupt Zero Count = 11
Interrupt Comparator Count = 10
Interrupt Capture Count = 20
- CaptureCountData[0] = 0x2007
- CaptureCountData[1] = 0x0005
- CaptureCountData[2] = 0x2007
- CaptureCountData[3] = 0x0005
- CaptureCountData[4] = 0x2007
- CaptureCountData[5] = 0x0005
- CaptureCountData[6] = 0x2007
- CaptureCountData[7] = 0x0005
- CaptureCountData[8] = 0x2007
- CaptureCountData[9] = 0x0005
- CaptureCountData[10] = 0x2007
- CaptureCountData[11] = 0x0005
- CaptureCountData[12] = 0x2007
- CaptureCountData[13] = 0x0005
- CaptureCountData[14] = 0x2007
- CaptureCountData[15] = 0x0005
- CaptureCountData[16] = 0x2007
- CaptureCountData[17] = 0x0005
- CaptureCountData[18] = 0x2007
- CaptureCountData[19] = 0x0005
Exit
```

3. サンプルソフトウェアの詳細

3.19 UART(UART2)

UART2 は非同期式シリアルインタフェースです。

このサンプルプログラムでは、PC と S5U1C31W74T1 評価ボードの間で UART 通信を行います。

ハードウェアセットアップ

以下の手順でハードウェアのセットアップを行います。

1. UART 用 USB アダプタを介して S5U1C31W74T1 評価ボードを PC に接続します (2.2.2 節、図 2.2.2.1、図 2.2.2.2 を参照)。
2. PC 上で UART コンソールとして動作するターミナルプログラムを起動します。
3. ターミナルプログラム上でシリアルポートの設定を行います (表 2.2.2.1)。

動作

1. UART2 を以下の設定で初期化します。
 - ボーレート: 115,200
 - データビット長: 8 ビット
 - ストップビット: 1 ビット
 - パリティ: なし
2. UART2 から ASCII 文字列を送信します。送信された文字列はターミナルプログラムのウィンドウに表示されます。
3. ターミナルプログラムからの受信に待機します。ターミナルプログラムで 8 文字を入力してください。
4. 入力された文字列をターミナルプログラムに送り返します。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
abcefgghiabcefgghi ... waiting for console input...
1234567812345678
Exit
```

3.20 USB CDCデバイス(USB_CDC)

USB は USB2.0 規格準拠の FS モードに対応した USB ターゲットデバイス用コントローラです。

このサンプルプログラムは、CDC-ACM (COM PORT) の実装例です。COM PORT から受信した文字をエコーバックします。

ハードウェアセットアップ

S5U1C31W74T1 評価ボードを USB ケーブルで PC に接続してください。

3. サンプルソフトウェアの詳細

3.21 USB 2.0 FSデバイスコントローラ(USB_HID)

USB は USB2.0 規格準拠の FS モードに対応した USB ターゲットデバイス用コントローラです。

このサンプルプログラムは、USB HID (Human Interface Device)クラスのデバイスの実装例です。このサンプルプログラムには、オープンソースの CMSIS USB スタックが使用されています。

HID クラスのデバイスは、HID インタフェースの標準と効率を遵守する一般要件を満たす必要があります。全ての HID デバイスは、コントロールエンドポイント(エンドポイント 0)、インタラプト IN エンドポイントを持ちます。また、多くの HID デバイスは、インタラプト OUT エンドポイントも使用します。すべての転送データは、その構造がレポートディスクリプタで定義されるレポートと呼ばれるフォーマットである必要があります。

ハードウェアセットアップ

S5U1C31W74T1 評価ボードを USB ケーブルで PC に接続してください。

動作

1. USB_HID サンプルプログラムを S5U1C31W74T1 評価ボードにダウンロードして実行します。
2. PC 上で Utility¥HidClient.exe をクリックして Epson HID Client ユーティリティを起動します。
3. Epson HID Client ウィンドウ上のプルダウンメニューから“EPSON CMSIS-HID”を選択します
4. Epson HID Client ウィンドウ上の“Out”フィールドのチェックボックス (LED0～LED7) をクリックして、ボード上のディスプレイに表示されるビット列が変化することを確認します。
5. ボード上の SW100 ボタン、または、SW101 ボタンを押して、HID Client ウィンドウ上の“IN”フィールドに“PRESSED”が表示されることを確認します。
6. PC によるウェイクアップを確認します。PC をスリープ状態に設定後、PC をウェイクアップして、HID Client が継続して操作できることを確認します。
7. リモートウェイクアップを確認します。Windows デバイスマネージャで電源管理を有効にして、PC をスリープ状態に設定します。ボード上の SW102 ボタンを押すと、PC がスリープから復帰します。
8. スヌーズを確認します。スリープモードを実行し SW103 ボタンを押してスヌーズを発動させます。

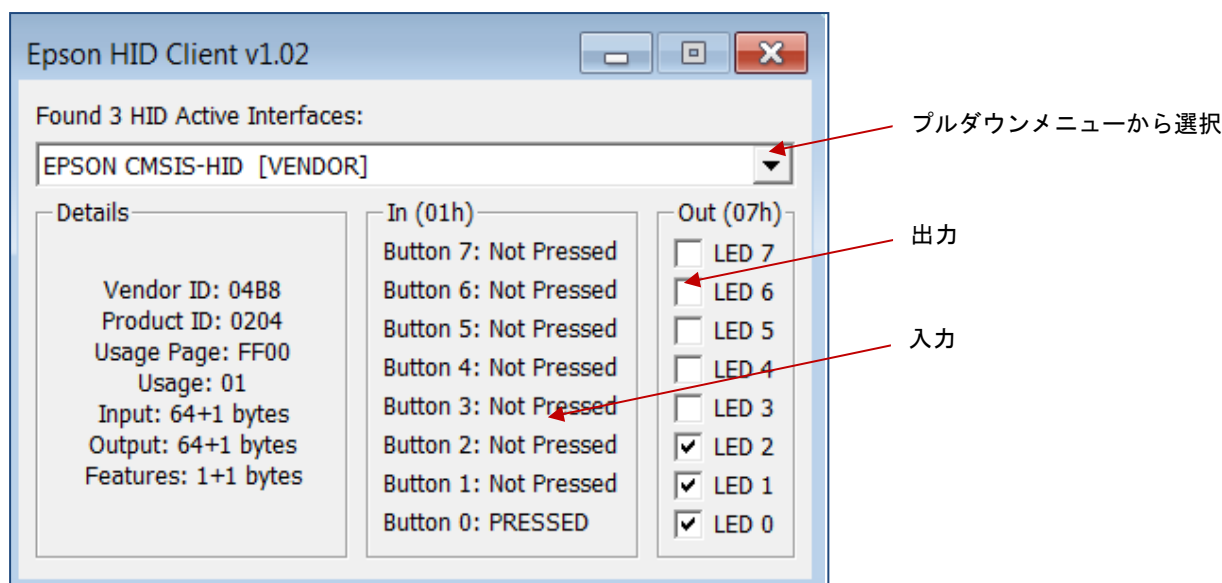


図3.21.1 Epson HID Clientユーティリティ

- LED0～LED7 をクリックすると対応するビットデータが S5U1C31W74T1 評価ボードのディスプレイに表示されます。
- S5U1C31W74T1 評価ボード上の SW100、SW101 を押すと、HID Client ウィンドウ上の“IN”フィールドに“PRESSED”が表示されます。
- リモートウェイクアップを確認します。デバイスマネージャで **HID** デバイスを選択し、“電源の管理”タブページの“このデバイスで、コンピュータのスタンバイ状態を解除できるようにする”を選択します。“電源の管理”については、Windows の文書を参照してください。
PC をスリープさせた後、S5U1C31W74T1 評価ボードの SW102 を押すと、PC がスリープから復帰します。

注意： Wakeup 機能は Windows 7 でのみ確認しています。

3. サンプルソフトウェアの詳細

3.22 USB MSCデバイス(USB_MSC)

USB は USB2.0 規格準拠の FS モードに対応した USB ターゲットデバイス用コントローラです。

このサンプルプログラムは、MSC デバイスの実装例です。この実装例では、S1C31W74 の内蔵 RAM を README.txt が格納された USB ストレージとして PC から認識されます。

ハードウェアセットアップ

S5U1C31W74T1 評価ボードを USB ケーブルで PC に接続してください。

出力例

```
-CPU clock- seCLG_IOSC (20000000)
Reset
Suspend
Resume
Reset
Reset
Configure
```

3.23 ウォッチドッグタイマ(WDT2)

WDT2 は問題の発生時にシステムを再起動させるウォッチドッグタイマです。

このサンプルプログラムでは、WDT2 を使用して NMI 割り込みとリセットの生成を行います。

動作

1. WDT2 を初期化します。
2. OSC1 を WDT2 のクロックソースとして設定し、OSC を起動してスリープモードで動作するように設定します。
3. NMI モードに設定し、NMI/リセット発生周期の経過による NMI を発生させます。
4. RESET after NMI モードに設定し、WDT をリセットして WDT リセットが発生しないことを確認します。
5. RESET after NMI モードに設定し、WDT をリセットせずに WDT リセットが発生させます。

出力例

```
CPU clock- seCLG_IOSC (20000000)
WDT Clock source = (1)
WDT Clock divider = (0)
WDT CLK = 250 Hz. CMP count = 1023
WDT 4 secs cycle is set
NMI interrupts due to comparator match in NMI Mode.
WDT NMI interrupts = 1
```

```
Watchdog timer was reset in Reset and NMI Mode.
Chip was alive after NMI
WDT Clock source = (1)
WDT Clock divider = (0)
WDT CLK = 250 Hz. CMP count = 250
WDT 1 secs cycle is set
WDT NMI interrupts = 5
```

```
Watchdog timer was not reset in Reset and NMI Mode.
Chip was reset on next comparator match.
WDT Clock source = (1)
WDT Clock divider = (0)
WDT CLK = 250 Hz. CMP count = 250
WDT 1 secs cycle is set
```

3. サンプルソフトウェアの詳細

3.24 フラッシュプログラミング

このサンプルプログラムは、seFlashLibrary を使用して内蔵 Flash メモリの消去／書き込みを行います。
seFlashLibrary は、CMSIS-Driver の Flash Interface 仕様に基づいて実装しています。

動作

1. GetInfo 関数をコールして、内蔵 Flash メモリの情報を取得します。
2. GetVersion 関数をコールして、seFlashLibrary のバージョンを取得します。
3. Initialize 関数をコールして、seFlashLibrary を初期化します。
4. EraseSector 関数をコールして、引数で指定されたセクタを消去します。
5. ProgramData 関数をコールして、ステップ 4 で指定されたセクタにデータを書き込みます。
6. ReadData 関数をコールして、ステップ 4 で指定されたセクタからデータを読み込み、その読み込みデータとステップ 5 の書き込みデータを比較します。
7. Uninitialize 関数をコールして、seFlashLibrary の初期化状態を解除します。

seFlashLibrary の使用上の注意

本ライブラリを使用する場合、以下の点に注意してください。

- 本ライブラリで提供される関数をコールする前に必ず割込み禁止に設定してください。
- 本ライブラリ実行時は、ライブラリを配置した領域を壊さないようにしてください。
- 本ライブラリを使用する際は、Flash メモリの書き換え可能回数に注意してください。Flash メモリの仕様については“S1C31W74 テクニカルマニュアル”を参照してください。
- 本ライブラリは 16 ビットタイマ (T16) ch.0 を使用しています。そのため、本ライブラリ実行後は、16 ビットタイマ (T16) ch.0 のレジスタ設定が変更されています。16 ビットタイマ(T16)ch.0 を使用しているプログラムで、本ライブラリを使用する場合は注意して下さい。
- 本ライブラリはシステムクロックを IOSC に切り換えて実行します。そのため、ライブラリ実行後は、クロックジェネレータ (CLG) のレジスタ設定が変更されています。必要に応じて、ライブラリ実行前に CLG のレジスタ設定をバックアップし、ライブラリ実行後に CLG のレジスタ設定を復元するなどの対策を取ってください。

3.25 EEPROMライブラリ (EEPROM)

このサンプルプログラムは、内蔵 Flash メモリで EEPROM 機能を実現するエミュレーションライブラリ (seEepromLibrary) を使用して、データの書き込み／読み込みを行います。seEepromLibrary は、EEPROM の各バイトに 256 バイトの内蔵 Flash メモリを使用します。0x60000 から 0x7FFFF の領域を EEPROM のエミュレーションに使用した場合、512 バイトの EEPROM に 100,000 回の読み書きを提供します。

seEepromLibrary は、CMSIS-Driver Flash Interface の仕様に基づいて実装しています。EEPROM ライブラリのヘッダーファイルは、CMSIS-Driver\Include\Driver_EEPROM.h にあります。seEepromLibrary を使用して初めてデータを書く場合は、0x60000 から 0x7FFFF の領域を一度だけ消去する必要があります。

パラメータ設定

EEPROM のエミュレーション領域を変更するにはリンカ設定ファイルを編集し、EEPROM を配置するセクションのメモリ領域を再定義してください。

IAR EWARM の場合は、"Examples/EEPROM/board/S5U1C31W74T1/IAR/config/S1C31W74_flash.icf" の "EEPROM1" のメモリ領域を再定義してください。

MDK-ARM (μ Vision) の場合は、"Examples /EEPROM/board/S5U1C31W74T1/ARM/eeprom_flash.sct" の "LR_IROM2" と "ER_IROM2" のメモリ領域を再定義してください。

EEPROM のサイズとリトライ回数を変更するには "Driver_EEPROM.h" を編集し、以下の定数を再定義してください。

```
#define CONFIG_EEPROM_SIZE_MAX (512)
#define CONFIG_RETRY_COUNT (4)
```

「CONFIG_EEPROM_SIZE_MAX」は EEPROM のサイズを示します。このサイズは 32/64/128/256/512 から選択できます。「CONFIG_RETRY_COUNT」は書き込みが失敗した時の書き込みリトライ回数を示します。書き込みリトライ回数を増やすと、書き込みルーチンの処理時間が長くなり、パフォーマンスが低下するおそれがありますので、数回程度で設定することを推奨します。

動作

1. GetInfo 関数をコールして、EEPROM の情報を取得します。
2. GetVersion 関数をコールして、EEPROM ドライバのバージョンを取得します。
3. ProgramData 関数をコールして、EEPROM にデータを書き込みます。
4. ReadData 関数をコールして、EEPROM からデータを読み込みます。

seFlashLibrary の使用上の注意

本ライブラリを使用する場合、以下の点に注意してください。

- 本ライブラリで提供される関数をコールする前に必ず割込み禁止に設定してください。
- 本ライブラリ実行時は、ライブラリを配置した領域を壊さないようにしてください。
- 本ライブラリは 16 ビットタイマ (T16) ch.0 を使用しています。そのため、本ライブラリ実行後は、16 ビットタイマ (T16) ch.0 のレジスタ設定が変更されています。16 ビットタイマ (T16) ch.0 を使用しているプログラムで、本ライブラリを使用する場合は注意して下さい。
- 本ライブラリはシステムクロックを IOSC に切り換えて実行します。そのため、ライブラリ実行後は、クロックジェネレータ (CLG) のレジスタ設定が変更されています。必要に応じて、ライブラリ実行前に CLG のレジスタ設定をバックアップし、ライブラリ実行後に CLG のレジスタ設定を復元するなどの対策を取ってください。

3. サンプルソフトウェアの詳細

3.26 ブートローダ (BootLoader)

このサンプルプログラムは、UART 通信により外部から送信されるプログラムのロードを行います。
本サンプルプログラムの詳細については、別紙「S1C31W74ブートローダマニュアル」を参照ください。

付録. サンプルプロジェクトのコードサイズ概要

表 付.1 に、本サンプルソフトウェアに含まれる各サンプルプロジェクトを IAR EWARM または MDK-ARM でビルドした場合のコードサイズの一覧を示します。

表 付.1各サンプルプロジェクトのコードサイズ

サンプルプロジェクト名	IAR EWARM (コードサイズ16KB以下)	MDK-ARM (コードサイズ32KB以下)
CLG	✓	✓
DMAC	✓	✓
Flash	✓	✓
I2C	✓	✓
LCD32B	✓	✓
PSPORT	✓	✓
QSPI	✓	✓
QSPI_DMA	✓	✓
QSPI_MASTER	✓	✓
QSPI_SLAVE	✓	✓
REMC2	✓	✓
RFC	✓	✓
RTCA	✓	✓
SNDA	✓	✓
SPIA_MASTER	✓	✓
SPIA_SLAVE	✓	✓
SVD2	✓	✓
T16	✓	✓
T16B	✓	✓
UART2	✓	✓
USB_CDC	✓	✓
USB_HID_S5U1C31W74T1	-	-
USB_MSC	-	-
WDT2	✓	✓

注: 統合開発環境のバージョンやビルドオプションの設定によって、コードサイズが表中のサイズを超える可能性があります。

改訂履歴表

[illegible]

セイコーエプソン株式会社

営業本部 デバイス営業部

東京 〒191-8501 東京都日野市日野421-8
TEL (042) 587-5313 (直通) FAX (042) 587-5116

大阪 〒530-6122 大阪市北区中之島3-3-23 中之島ダイビル22F
TEL (06) 7711-6770 (代表) FAX (06) 7711-6771

ドキュメントコード : 413276303
2017年 4月 作成
2019年 2月 改訂