Renesas Synergy™プラットフォーム
DMAC HAL モジュールガイド

R11AN0098JU0101
Rev.1.01
2019.07.09

（注 1）本資料は英語版を翻訳した参考資料です。内容に相違がある場合には英語版を優先します。資料によっては英語版のバージョンが更新され、内容が変わっている場合があります。日本語版は、参考用としてご使用のうえ、最新および正式な内容については英語版のドキュメントを参照ください。

（注 2）本資料の第 6 章まで（要旨除く）の日本語訳は、「Synergy™ Software Package (SSP) v1.5.0 ユーザーズマニュアルモジュール概要編（参考資料）」の第 4 章「モジュールの概要」に掲載されていますのでそちらを参照ください。

要旨（Introduction）

本モジュールガイドは、ユーザが DMAC HAL モジュールを効果的に使用してシステムが開発できるようになることを目的としています。このモジュールガイドを習得することで、開発システムへのモジュールの追加とターゲットアプリケーション向けの正確な設定（configuration）ができ、さらに付属のアプリケーションプロジェクトコードを参照して、効率的なコード記述が行えるようになります。より詳細な API や、より高度なモジュール使用法を記述した他のアプリケーションプロジェクト例もルネサス WEB サイト（本書末尾の「参考文献」の項を参照）から入手でき、より複雑な設計に役立ちます。

ダイレクトメモリアクセスコントローラ（Direct Memory Access Controller、DMAC）HAL モジュールは、データ転送アプリケーション向けのハイレベル API（high-level API）で、`r_dmac` に実装されており、Synergy MCU 上にある DMAC 周辺回路（peripheral）を使用します。転送イベント（transfer event）が発生したときに CPU への通知が行われるように、ユーザ定義コールバックを作成することができます。

目次

このモジュールガイドで説明するアプリケーションプロジェクトを実際に使うことで、設計全体の手順を体験することができます。このプロジェクトは、このドキュメントの末尾にある「参考情報」章に掲載されているリンクにあります。ISDE でアプリケーションプロジェクトをインポートして開き、DMAC HAL モジュールに対応する設定項目を表示することができます。また、完成した設計で、DMAC HAL モジュール API を示すために使用しているコード `dmac_hal.c` を確認することもできます。

本アプリケーションプロジェクトは、DMAC HAL モジュール API の標準的な使用方法を示します。本アプリケーションプロジェクトは、DMAC チャネル 0、1、2 上に設定した 3 個の転送インタフェース（transfer interface）を使用するほか、複数の外部 IRQ 端子とタイマを組み合わせて使用し、データとトリガのソースを生成します。

転送ドライバ 0（Transfer Driver 0）は、配列（array）からデバイスの I/O ポートにデータを転送するように設定（configure）されています。選択した I/O ポートは、複数の LED の接続先である PORT6 です。データの転送状況は、複数の LED を通じて目視で確認できます。転送ドライバ 0（Transfer Driver 0）の起動元（activation source）は、非同期汎用タイマ（Asynchronous General Purpose Timer、AGT）の 1 つである AGT0 です。この AGT は、250 ms（ミリ秒）間隔で割り込みを生成するように設定済みです。

転送ドライバ 1（Transfer Driver 1）は、GPT カウンタの値（GPT Count Register 0）をソフトウェアバッファレジスタ（g_dest_data[ ]）にデータを転送します。転送ドライバ 1 のアクティブ化ソースは IRQ11 で、ユーザがプッシュボタン S4 を押下したときに外部 IRQ として生成されます。。

転送ドライバ 2（Transfer Driver 2）は、転送ドライバ 0 と同じアドレスの相互間でデータを転送するように設定済みです。ただし、転送ドライバ 0 が周辺回路（ハードウェア）によってトリガされるのに対し、転送ドライバ 2 はソフトウェアによってトリガされます。ユーザプッシュボタンである S5 が押下されたときに、ソフトウェアの呼び出しがトリガされます。プッシュボタン S5 の押し下げは IRQ10 を生成します。IRQ10 の割り込み要求（interrupt request、IRQ）は、IRQ10 のユーザコールバック関数内でソフトウェアアクティブ関数（software activation function）を呼び出します。セミホスト機能（semi-hosting）が有効な場合、転送ドライバ 1 により転送されたデータ g_dest_data[ ]の内容がターミナルに出力されます。

転送ドライバ 0 は、通常モード（Normal Mode）で動作し、60 の転送カウント（transfer count）に設定済みです。このモードでは、DMAC は指定された回数のデータを転送し、再アクティブ化されない場合は追加の転送を行いません。転送ドライバ 1 と転送ドライバ 2 は、反復モード（Repeat Mode）で動作し、転送カウントは 8 に設定済みです。反復モードで、DMAC が指定回数（この場合は 8）の転送を完了した時点で、DMAC は自らを自動的に再設定し、転送動作を繰り返します。その結果、DMAC は連続的に動作します。

表 5　このアプリケーションプロジェクトが使用するソフトウェアとハードウェアのリソース

リソース	リビジョン	説明
e² studio	5.3.1 またはそれ以降	統合ソリューション開発環境（ISDE）
SSP	1.2.0 またはそれ以降	Synergy ソフトウェアプラットフォーム
IAR EW for Renesas Synergy	7.71.2 またはそれ以降	IAR Embedded Workbench® for Renesas Synergy™
SSC	5.3.1 またはそれ以降	Synergy Standalone Configurator
SK-S7G2	v3.0, v3.1 またはそれ以降	スタータキット

以下の図に、このアプリケーションプロジェクトのシンプルなフローを示します。

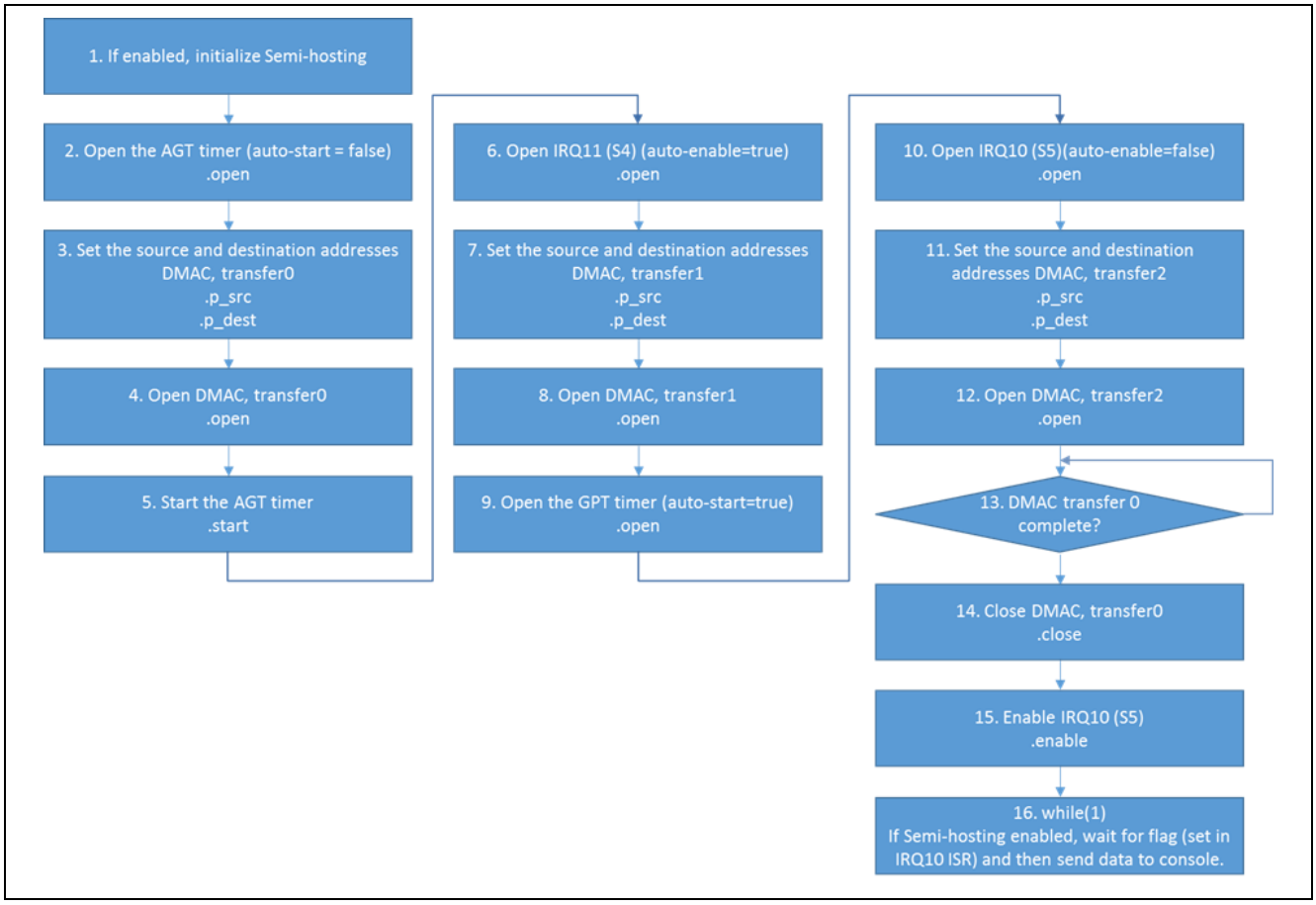


図 4　DMAC HAL モジュールのアプリケーションプロジェクトのフロー

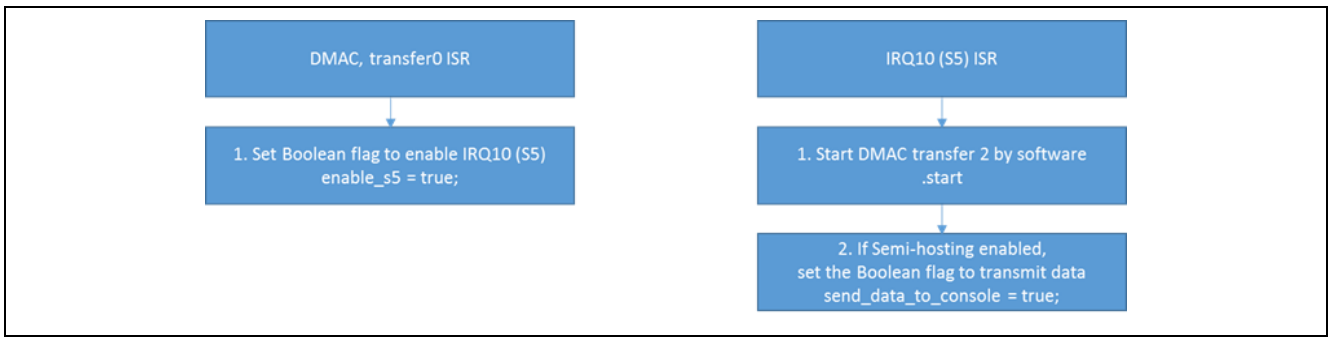


図 5　DMAC HAL モジュールのアプリケーションプロジェクトの ISR フロー

dmac_hal.c ファイルは、このプロジェクトを ISDE にインポートすると、プロジェクト内に配置されます。ISDE でこのファイルを開き、API の使い方のガイドを参照できます。

dmac_hal.c の最初のセクションは、複数のヘッダファイルを詳細に記述しています。最初のヘッダファイルである hal_data.h は、DMACtransfer（転送）と AGT、および汎用タイマ（General Purpose Timer、GPT）と割り込み要求（IRQ）の各インスタンス構造体（instance structure）を参照します。2 番目のヘッダファイルである dmac_hal.h は、プリプロセッサコマンド（pre-processor command）#define SEMI_HOSTING を追加または削除する方法で、セミホスト機能を有効または無効にすることができます。3 番目のヘッダファイルは、エディタがデバイスレジスタ位置を参照できるように用意されています。これらのレジスタの位置は、DMAC の transfer（転送）の転送元と転送先の各アドレスとして使用されます。

その後に、転送元と転送先の各配列や、ブール型変数（Boolean flag）など、このプロジェクトが使用するグローバル変数（global variable）の定義が続きます。次のセクションには、メインプログラム制御セクションに対応するエントリ関数（entry function）があります。この関数が有効で、GCC を使用する場合、セミホスト機能が有効になります。IAR Embedded Workbench® for Renesas Synergy™ を使用する場合、これらの動作は自動的に実施されます。次に AGT が開きます。この AGT は 1 秒ごとに割り込み（interrupt）が発生するように設定されています。設定パラメータ Auto Start（オートスタート）が false（論理的な「偽」）に設定されているため、AGT を開いた時点で AGT は開始されません。この AGT 割り込みは、DMAC の transfer0 をトリガします。

次に、DMAC の transfer0 が設定（set up）されます。この設定は注意深く検討する必要があります。各フィールドが用意されていても、コンフィギュレータ内で Destination Pointer（転送先ポインタ）と Source Pointer（転送元ポインタ）の設定することはできません。デフォルトでのこれらポインタの設定は NULL です。ただし、これらのポインタを NULL に設定し、Auto Enable（自動有効化）パラメータを True（論理的な「真」）に設定した場合、transfer（転送）を開いた時点でリターンコード（戻り値）は SSP_SUCCESS になりません。

ポインタを設定するには、次の方法のいずれかを使用します。

1. Destination Pointer（転送先ポインタ）と Source Pointer（転送元ポインタ）を NULL に設定する時点で、Auto Enable（自動有効化）を false に設定します。
   次の API コマンドを使用して、transfer（転送）を開きます。

```
g_transfer0.p_api->open(g_transfer0.p_ctrl, g_transfer0.p_cfg);
```

   transfer（転送）を開いた後、reset（リセット）API を使用して転送元と転送先の各アドレスを設定します。

```
g_transfer0.p_api->reset(g_transfer0.p_ctrl, SRC, DEST, COUNT);
```

2. 他の方法は、open API を使用する前に、転送元（source）と転送先（destination）の各設定パラメータを設定することです。例：

```
g_transfer0.p_cfg->p_info->p_src = (void *) &g_source_data;
g_transfer0.p_cfg->p_info->p_dest = (void *) &R_IOPORT6->PCNTR1;


ssp_err = g_transfer0.p_api->open(g_transfer0.p_ctrl, g_transfer0.p_cfg);
if (SSP_SUCCESS != ssp_err)
{
    while (1)
    {
    }
}
```

この方法により、Auto Enable（自動有効化）パラメータを true に設定することができます。

さらに、DMAC の transfer0 は、32 ビットの値を転送するように設定されます。NORMAL Mode（通常モード）で、cb_dmac0 という形で定義されているコールバックの中で、Source Address（転送元アドレス）はインクリメントし、Destination Address（転送先アドレス）は固定され、COUNT（カウント）の値は 60 になり、DMAC 割り込みは有効になります。

AGT と DMAC を transfer0 に設定した状態で、次は AGT を開始します。AGT は、DMAC をトリガする 1 秒ごとの割り込みを生成します。DMAC は g_source_data から R_IOPORT6->PCNTR1 にデータを転送します。転送元の配列（source array）は 32 ビットの値を 1 個保持しており、この値は I/O ポート制御レジスタ 1（I/O Port Control Register 1）の[2], [1], [0]ビットをセット（1 に設定）およびクリア（0 に設定）します。このポート制御レジスタ 1 は、ポート端子の I/O モードを制御します。モードを「出力」に設定した場合、出力値を設定することになります。ポート端子に複数の LED を接続した状態で、DMAC の transfer は、LED の状態を変化させます。

次に、IRQ11 をオープンにします。ユーザがプッシュボタン S4 を押した時点で、IRQ11 が生成されます。IRQ11 割り込みをトリガとして、DMAC の transfer1 が開始します。

次に、DMAC の transfer0 と同じ方法で DMAC の transfer1 を設定します。.open 関数を呼び出す前に、送信元と送信先の各アドレスを設定します。

DMAC の transfer1 は、いくつかの点で DMAC の transfer0 とは異なっていますが、最も異なるのはその構成モードです。DMAC の transfer1 は、REPEAT（反復）モードに設定されています。DMAC の transfer0 は、カウントが指定する回数分の転送を行います。このサンプルの場合は 60 回で、その後に動作を停止します。一方、DMAC の transfer1 の動作は、カウントによる指定回数にわたって転送を行い（このサンプルの場合は 8 回）、その後、DMAC は自動的に自らを再設定し、動作を再開します。したがって、DMAC は連続的に動作します。GPT カウンタレジスタ（GPT counter register）から転送先の配列 g_dest_data [ ]にデータを転送するように、DMAC の transfer1 を設定します。確実にデータを転送するために、GPT タイマを動作させる必要があります。DMAC transfer1 を設定した後、GPT を開きます。GPT の Auto Start（オートスタート）パラメータを true に設定します。

次に、IRQ10 をオープンにします。ユーザがプッシュボタン S5 を押すと、IRQ10 が生成されます。IRQ10 割り込みは DMAC の transfer2 にトリガをかけますが、その方法はソフトウェアアクティブ化 API（software activation API）の呼び出しによるものです。

次に、DMAC の transfer1 と同じ方法で DMAC の transfer2 を設定します。.open を呼び出す前に、転送元と転送先の各アドレスを設定します。

反復モードで使用するために DMAC の transfer2 をもう一度設定しますが、今回は他の 2 つの transfer と異なり、割り込みのアクティブ化ではなくソフトウェア呼び出しでトリガをかけます。転送元の配列 g_source_data から R_IOPORT6->PCNTR1 に 32 ビットデータを転送するように、DMAC の transfer2 を設定します。その結果、ユーザがプッシュボタン S5 を押下したときに、複数の LED の状態が変化します。

DMAC の transfer2 が発生するのは、DMAC の transfer0 が終了したときのみです。

IRQ10 で制御する転送動作が有効になるのは、DMAC の transfer0 による End（終了）割り込みが発生し、この状態を表すブール型フラグ（Boolean flag）をセットした場合のみです。

DMAC の transfer0 が完了した時点で、この transfer が閉じます。

アプリケーション関数の最後のセクションに while(1)ループがあります。このループは、ブール型フラグがセットされるまで待機します。このフラグがセットされた時点で、アプリケーションは g_dest_data[]の内容をコンソールに書き込みます。g_dest_data [ ]の内容は、S4 と DMAC の transfer1 による結果です。

最後のセクションには、複数のユーザコールバック関数があります。最初のコールバックは DMAC の transfer0 です。この呼び出しが実行されるのは、DMAC transfer0 が 60 回の転送を行い、ブール型フラグをセットしてアプリケーションコードへの通知を行ったときです。2 番目のコールバックは IRQ10(S5)であり、このコールバックはソフトウェア経由で DMAC の transfer2 にトリガをかけます。

注記:この説明は、Synergy ソフトウェアパッケージ内のデバッグコンソールで printf() を使用する方法をユーザが理解していることを想定しています。このような経験がない場合は、下記 WEB サイトの FAQ 2000008「Synergy ソフトウェアパッケージのデバッグコンソールで Printf_使用方法」という記事を参照してください。デバッグモードで変数ウォッチ機能を使用して結果を表示することもできます。
https://ja-support.renesas.com/knowledgeBase/17792531

このアプリケーションプロジェクトでは、ターゲットボードや MCU の物理プロパティ（physical property）に加え複数の重要なプロパティに対し、必要な操作をサポートするために設定しています。次の表に、この特定のプロジェクトで設定したそれらのプロパティの値を示します。また、実践的な演習として、このアプリケーションプロジェクトを開き、[Properties]（プロパティ）ウィンドウでこれらの設定を表示することもできます。

表 6　アプリケーションプロジェクトに対応する AGT HAL モジュールの設定項目

ISDE のプロパティ	設定値
Name（名前）	g_agt0
Channel（チャネル）	0
Mode（モード）	Periodic（周期的）
Period Value（期間の値）	250
Period Unit（期間の単位）	Milliseconds（ミリ秒）
Auto Start（オートスタート）	False
Count Source（カウント用クロックのソース）	LOCO

表 7　アプリケーションプロジェクトに対応する DMAC の transfer0 HAL モジュールの設定項目

ISDE のプロパティ	設定値
Name（名前）	g_transfer0
Channel（チャネル）	0
Mode（モード）	Normal（通常）
Transfer Size（転送サイズ）	4 Bytes（4 バイト）
Destination Address Mode（転送先アドレスモード）	Fixed（固定）
Source Address Mode（転送元アドレスモード）	Incremented（インクリメント）
Number of Transfers（転送の数）	60
Activation Source（アクティブ化ソース）	Event AGT0 INT
Auto Enable（自動有効化）	True
Callback（コールバック）	cb_dmac0
Interrupt Priority（割り込みの優先順位）	Priority 8（CM4: Valid, CM0+: invalid）（優先順位 8（CM4: 有効、CM0+: 無効））

表 8　アプリケーションプロジェクトに対応する IRQ11 HAL モジュールの設定項目

ISDE のプロパティ	設定値
Name（名前）	g_external_irq11_s4
Channel（チャネル）	11
Trigger（トリガ）	Falling（立ち下がり）
Callback（コールバック）	NULL
Interrupt Priority （割り込みの優先順位）	Priority 8（CM4: Valid, CM0+: invalid） （優先順位 8（CM4: 有効、CM0+: 無効））

表 9　アプリケーションプロジェクトに対応する DMAC の transfer1 HAL モジュールの設定項目

ISDE のプロパティ	設定値
Name（名前）	g_transfer1
Channel（チャネル）	1
Mode（モード）	Repeat（反復）
Transfer Size（転送サイズ）	4 Bytes（4 バイト）
Destination Address Mode （転送先アドレスモード）	Incremented（インクリメント）
Source Address Mode （転送元アドレスモード）	Fixed（固定）
Repeat Area（反復領域）	Destination（転送先）
Number of Transfers（転送の数）	8
Activation Source （アクティブ化ソース）	Event ICU IRQ11
Auto Enable（自動有効化）	True

表 10　アプリケーションプロジェクトに対応する GPT HAL モジュールの設定項目

ISDE のプロパティ	設定値
Name（名前）	g_gpt0
Channel（チャネル）	0
Mode（モード）	Periodic（周期的）
Period Value（期間の値）	0xffff
Period Unit（期間の単位）	Raw Counts（クロックをカウント）
Auto Start（オートスタート）	True

表 11　アプリケーションプロジェクトに対応する IRQ10 HAL モジュールの設定項目

ISDE のプロパティ	設定値
Name（名前）	g_external_irq10_s5
Channel（チャネル）	10
Trigger（トリガ）	Falling（立ち下がり）
Interrupt enabled after initialization （初期化の後に割り込みを有効にする）	False
Callback（コールバック）	Cb_irq10_s5
Interrupt Priority （割り込みの優先順位）	Priority 8（CM4: Valid, CM0+: invalid） （優先順位 8（CM4: 有効、CM0+: 無効））

表 12　アプリケーションプロジェクトに対応する DMAC の transfer2 HAL モジュールの設定項目

ISDE のプロパティ	設定値
Name（名前）	g_transfer2
Channel（チャネル）	2
Mode（モード）	Repeat（反復）
Transfer Size（転送サイズ）	4 Bytes（4 バイト）
Destination Address Mode（転送先アドレスモード）	Fixed（固定）
Source Address Mode（転送元アドレスモード）	Incremented（インクリメント）
Repeat Area（反復領域）	Source（転送元）
Number of Transfers（転送の数）	8
Activation Source（アクティブ化ソース）	Software Activation（ソフトウェアアクティブ化）
Auto Enable（自動有効化）	True

8. ターゲットアプリケーションに対応する DMAC HAL モジュールのカスタマイズ（Customizing the DMAC HAL Module for a Target Application）

このアプリケーションプロジェクトの設定項目は、このサンプルアプリケーションに合わせた値になっていますが、ユーザはターゲットアプリケーションに合わせて、DMAC HAL を容易に変更できます。たとえば、このサンプルアプリケーションでは、すべてのデータ転送を 32 ビットで実行しています。DMAC は、8 ビット、16 ビット、または 32 ビットのデータを転送するように設定できます。また、このサンプルアプリケーションは通常モードと反復モードの使用法を例示しているのに対し、DMAC はアクティブ化の際にブロック転送を行うように設定することも可能です。

9. DMAC HAL モジュールのアプリケーションプロジェクトの実行（Running the DMAC HAL Module Application Project）

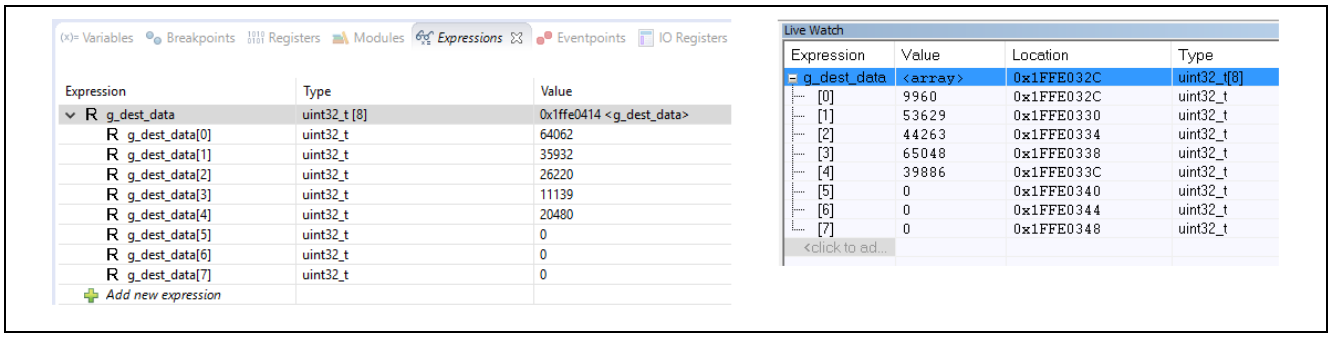
DMAC HAL モジュールのアプリケーションプロジェクトの動作を確認するために、ターゲットキットで ISDE にこのプロジェクトをインポートし、コンパイル（compile）してデバッグ（debug）を実行することができます。

新しいプロジェクト内で DMAC HAL モジュールアプリケーションを実装するには、ターゲットキットで定義、設定、ファイルの自動生成、コードの追加、コンパイル、デバッグを行うため、以下の手順に従います。

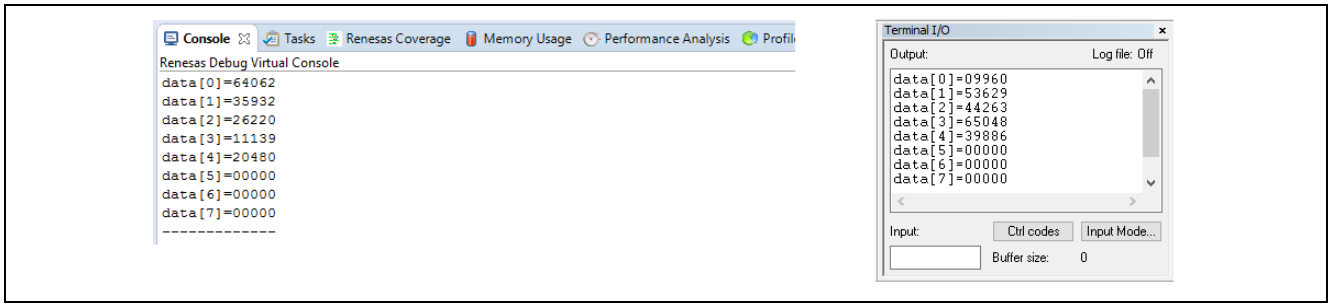
パッケージ付属のサンプルプロジェクトをインポートして、ビルドしてください。実行する手順については、『Synergy プロジェクトインポートガイド』（下記 WEB）を参照してください。

- 英語版：
  https://www.renesas.com/jp/ja/doc/products/renesas-synergy/apn/r11an0023eu0121-synergy-ssp-import-guide.pdf
- 日本語版（参考資料）：
  https://www.renesas.com/jp/ja/doc/products/renesas-synergy/apn/r11an0023ju0121-synergy-ssp-import-guide.pdf

このプロジェクトを実行すると、複数の LED が数秒間にわたってトグル（点灯と消灯を切り替え）します。S4 を押下すると、GPT カウンタの値が `g_dest_data[]` 配列に転送されます。図に示すように、e² studio の左側にある数式ウィンドウ、または IAR EW for Synergy の右側にあるウォッチウィンドウでこの配列を表示できます。



LED がトグルを停止した時点で、もう一度プッシュボタン S5 を押下すると LED の状態が変化します。配列の内容が e² studio（左側）または IAR EW for Renesas Synergy™（右側）のコンソールウィンドウに書き込まれます。



10. DMAC HAL モジュールのまとめ（DMAC HAL Module Conclusion）

このモジュールガイドは、サンプルプロジェクトでモジュールの選択、追加、設定、使用を行うために必要な背景となる情報全般を説明しました。従来の組み込みシステムでは、これらの手順を理解することに多くの時間を必要とし、また間違いが起こりやすい操作でした。Renesas Synergy プラットフォームにより、これら手順の所要時間が短くなり、設定項目の競合や、ローレベルドライバの誤った選択など、誤りが防止できるようになりました。アプリケーションプロジェクトで示したように、ハイレベル API を使用することで高いレベルの開発からスタートし、ローレベルドライバを作成するような従来の開発環境で必要とされる時間が不要になり、開発時間を短縮できます。

11. DMAC HAL モジュールの次の手順（DMAC HAL Module Next Steps）

シンプルな DMAC HAL モジュールのプロジェクトをマスターすれば、より複雑なサンプルをレビューできるようになります。DMAC HAL の使用方法を示す他のアプリケーションプロジェクトとアプリケーションノートは、このドキュメントの末尾にある「参考情報」の章に掲載されています。

12. DMAC HAL モジュールの参考情報（DMAC HAL Module Reference Information）

『SSP ユーザーズマニュアル』: SSP ディストリビューションパッケージの一部として HTML 形式が入手できるほか、Renesas Synergy™ WEBサイトのSSPページ
https://www.renesas.com/jp/ja/products/synergy/software/ssp.htmlから pdf を入手することもできます。

最新版の r_dmacモジュールの参考資料やリソースへのリンクは、以下の Synergy WEBサイトから入手できます。

https://www.renesas.com/jp/ja/products/synergy.html

Web サイトおよびサポート

サポート： https://synergygallery.renesas.com/support

テクニカルサポート：

- アメリカ： https://www.renesas.com/en-us/support/contact.html
- ヨーロッパ： https://www.renesas.com/en-eu/support/contact.html
- 日本： https://www.renesas.com/ja-jp/support/contact.html

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.01	2019.07.09		• 初版 • 英語版（R11AN0109EU0101, Rev.1.01, 2019.05.21）の巻頭と第 7 章以降を翻訳

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品、本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準：　コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等
   当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の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合せください。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、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。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、一切その責任を負いません。
9. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
10. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
12. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.4.0-1　2017.11)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。