Embedded Code Generation Tutorial

Introduction to PLECS STM32 Code Generation

PLECS STM32のコード生成の紹介

Tutorial Version 1.0

KESCO KEISOKU ENGINEERING SYSTEM
計測エンジニアリングシステム株式会社
https://kesco.co.jp

はじめに

このチュートリアルでは、PLECS CoderとSTM32 Target Support Packageを使用してSTM32マイクロコントローラ(Micro Controller Unit: MCU)を使用する方法を学習します。

このチュートリアルを始める前に、次のものを用意するかインストールしておいてください:

- PLECS StandaloneまたはPLECS Blocksetのいずれかの有効なライセンスが必要です。無料トライアルライセンスはPlexim社のウェブサイト(https://www.plexim.com/trial)からお申し込みいただけます。
- PLECS Coderのライセンス。無料トライアルをご希望の場合は、アドオン製品"PCCT: PLECS Coder Trial"を選択してください。
- STM32 Target Support Package(TSP): STM32 TSPのインストールと使用開始に関するステップバイステップの説明ビデオ[1]に従うか、STM32 Target Support Packageユーザマニュアル[2]のクイックスタートを参照してください。
- STM32 NUCLEO-G474REボード[3]を1台(以下"G474RE"または"STM32 MCU"または"STM32ボード"と表記)。
- ジャンパ線。
- 演習の各段階で、自分のモデルと比較できる添付ファイルがあることを確認してください。

演習1: LEDを点滅させる

最初の演習では、G474REのポートAのピン5(PA5)に接続された緑色のLED "LD2"を点滅させる簡単なモデルを作成します。LEDは0.5秒間点灯し、その後0.5秒間消灯します。

課題1.1: PLECSモデルの作成

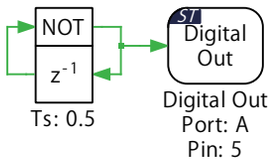
あなたのタスク:

1. PLECSを開き、新しいモデルファイルを作成します。PLECSライブラリブラウザからサブシステムブロックをメイン回路図にドラッグアンドドロップし、"LED Blinker"というラベルを付けます。
2. "LED Blinker"サブシステムを右クリックし、**サブシステム** -> **実行の設定...**選択します。設定ウィンドウで、**コード生成機能の有効化**チェックボックスをオンにし、**適用**をクリック後、OKボタンを押します。これにより、サブシステムが不可分なサブシステム(サブシステム内のすべてのコンポーネントがグループとして一緒に実行される)に変換され、PLECSモデル全体をビルドすることなく、サブシステムのみをターゲットデバイス上にビルドできるようになります。
3. この"LED Blinker"サブシステム内の既存のコンポーネントをすべて削除し(デフォルトでは信号ポートが含まれています)、**遅れ要素**ブロックをその回路図にドラッグアンドドロップし、**サンプル時間**パラメータを`0.5`に設定します。次に、**論理演算子**ブロックをドラッグアンドドロップし、**演算子**パラメータで`NOT`を選択します。
4. PLECSライブラリブラウザのSTM32 Targetから、Digital Outブロックをドラッグアンドドロップします。Portを`A`、Pin numberを`5`に設定します。
5. 次に、図1のように、これら3つのブロックを接続します。

課題1.2: PLECSでオフラインシミュレーションを検証

"LED Blinker"サブシステムをMCUのターゲット固有のコードに直接変換する前に、PLECSモデルをコンピュータ上でオフラインでシミュレーションすることができます。

図1: G474REのLEDを点滅させる回路図



あなたのタスク: 図2に示すように、トップレベルの回路図にPLECSスコープを配置し、Digital Outポートに接続してシミュレーションを開始します(Ctrl + T)。図3に示すように、1Hz、50%デューティー比のPWM波形が表示されます。

図2: トップレベルの回路図

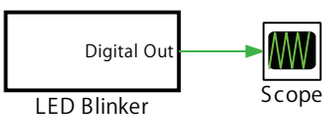
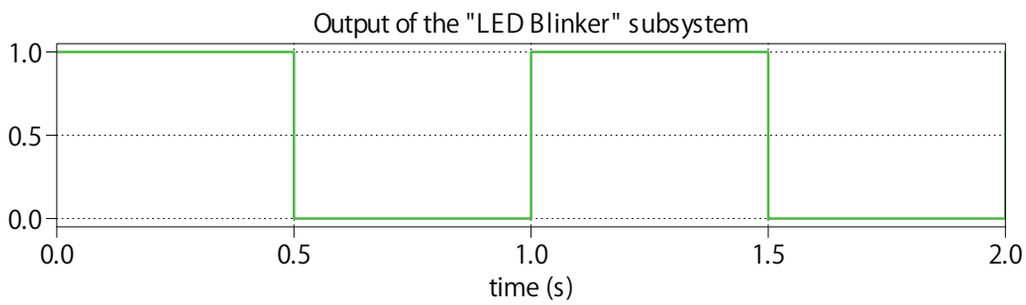


図3: PLECS上で"LED Blinker"サブシステムをオフラインで実行した結果



課題1.3: Coderオプションの構成

あなたのタスク:

1. PLECS回路図のメニューから**Coder** -> **Coderオプション...**を選択します。Coder Optionsウィンドウの**システム**リストから"LED Blinker"を選択します。
2. 次に、**タスク**タブで、**離散化ステップサイズ**を`1e-4`に設定します。このパラメータは、制御ブロックの連続状態変数を離散化するために使用する基本サンプル時間を指定します。次に、**基本設定**タブの**浮動小数点書式**で`float`を選択します。
3. **ターゲット**タブで、ターゲットのリストから`STM32G4x`を選択します。次に、サブタブの**General**で、**Chip**から`G474RE`を選択します。
4. その他のパラメータはデフォルト値のままにし、任意の場所にモデルを保存します。

課題1.4: STM32 MCUのフラッシュ

このPLECSモデルをSTM32 G474RE MCUのターゲット固有のコードに変換します。

あなたのタスク:

1 目的のMCUをUSBケーブルを介してホストコンピューターに接続します。

2 PLECSから MCUターゲットを直接デプロイするには: **Coder Options**ウィンドウの**ターゲット**タブで、ドロップダウンメニューから必要な**Programming interface**を選択し、**ビルド**をクリックします。デフォルトのプログラミングインタフェースは`OpenOCD`です。

正しくプログラムされていれば、STM32ボード上のLEDが点滅します。

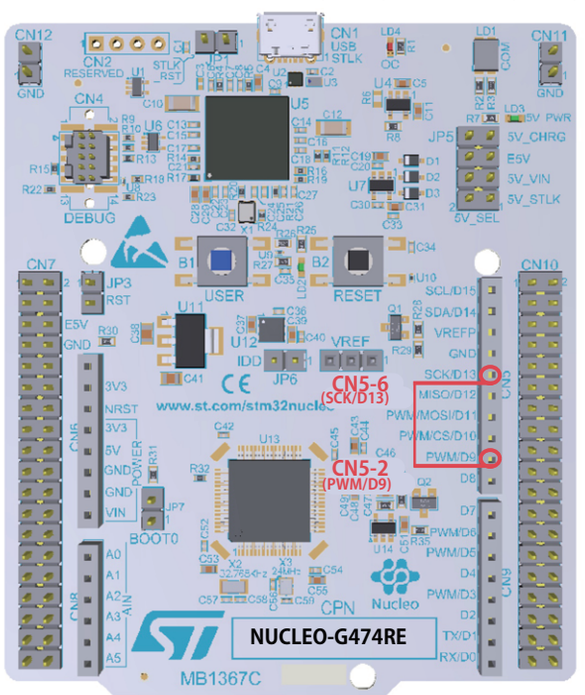
この段階では、モデルとシミュレーション結果は`stm32_intro_1.plecs`と同じになっているはずです。

演習2: PWM出力の設定

この演習では、演習1と同じLED "LD2"を、PWMブロックを使用して制御します。PWMブロックは、1つ以上のPWMリソース上に単一または補完的なPWMペアを生成します。このブロックへの入力として変調インデックスを指定する必要があります。

G474REのLED "LD2"(PA5、CN5-6またはCN10-11の"SCK/D13"と表示)は、STM32 MCUでPWM信号を生成するために使用する高機能タイマ(TIM)に接続していません[4]。そのため、まず利用可能なPWM出力(PC7、CN5-2またはCN10-19の"PWM/D9"というラベル)にPWM信号を生成し、図4に示すように、ジャンパ線を使用してLEDに接続します。

図4: ジャンパ線を使用してPWM出力(CN5-2)をLED "LD2"(CN5-6)に接続

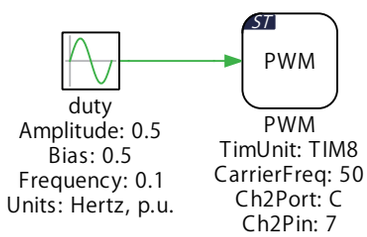


課題2.1: PLECSモデルの修正

あなたのタスク:

1. 演習1と同じPLECSモデルファイル内に、2つ目のサブシステムブロックを配置し、"LED Dimmer"という名前を付けます。再度、内部の既存のコンポーネントを削除し、サブシステムを右クリックし、**サブシステム** -> **実行の設定...**選択します。設定ウィンドウで、**コード生成機能の有効化**チェックボックスをオンにして、コード生成を有効にします。
2. STM32 Targetライブラリから、PWMブロックを"LED Dimmer"サブシステムにドラッグアンド ドロップします。
3. [4]のI/O割り当て表から、PC7がTIM8_CH2に接続されていることがわかります。そのため、PWMのブロックパラメータウィンドウの**全般**タブで、**TIM unit**に`TIM8`を選択し、**Carrier frequency**を`50`Hzに設定します。
4. **Channel 2**タブで、**Mode**を`Single output`に選択し、**Port**を`C`、**Pin**を`7`に設定します。
5. その他のチャネルはすべて`Disable`にします。その他のパラメータはすべてデフォルト値のままにしておきます。
6. 次に、図5のように、PWMブロックを**振幅**`0.5`、**バイアス**`0.5`、**周波数**`0.1`Hz、**周波数と位相の単位**がヘルツ(p.u.)の正弦波信号に接続します。

図5: PWMブロックを使用してG474REのLEDを調光する回路図



課題2.2: PLECSでオフラインシミュレーションを検証

オフラインシミュレーション中、PWMブロックは通常のPWM生成ブロックとして動作します。PWMブロックを右クリックし、**サブシステム** -> **サブシステムのモデル表示**を選択して、オフラインの"回路図"の実装を確認してください。

あなたのタスク: PLECSスコープの**ファイル**メニュー -> **スコープパラメータ**で**プロット数**を増やすか、プロットウィンドウを右クリックして**プロットを下に挿入**して、トップレベル回路図のスコープにプロットを追加し、新しく作成したスコープのポートをPWM出力ポートに接続します(図6を参照)。新しいシミュレーションを開始すると、図7のようにPWM 波形が表示されます。変調波形により、LED の明るさが徐々に増減します。

図6: トップレベルの回路図

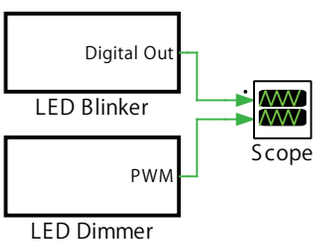
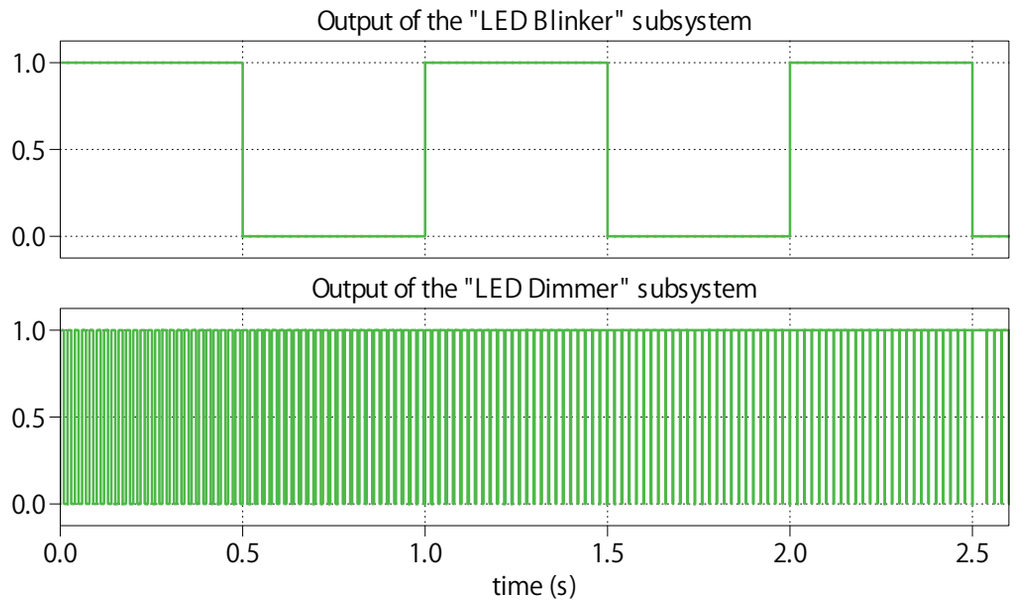


図7: PLECSにおける"LED Dimmer"サブシステムのオフラインテスト結果



課題2.3: 必要なハードウェアを接続

あなたのタスク: ジャンパ線を使用して、LED "LD2"に対応するピンPA5(CN5-6またはCN10-11の"SCK/D13"のラベル)をPWM出力に対応するピンPC7(CN5-2またはCN10-19の"PWM/D9"のラベル)に接続します(図4を参照)。

課題2.4: Coderオプションを構成し、STM32 MCUをフラッシュさせる

あなたのタスク: Coder Optionsウィンドウに戻り、左側のシステムリストから"LED Dimmer"を選択します。次に、コード生成機能の有効化を設定し、演習1で説明したようにSTM32 MCUをフラッシュします。

正しくプログラムされていれば、STM32ボード上のLEDの明るさが徐々に増減するはずです。

この段階では、モデルとシミュレーション結果は`stm32_intro_2.plecs`と同じになっているはずです。

オプション: PWMブロックのデューティー比入力やCarrier Frequencyを変更し、LEDの明るさへの影響を調べます。

演習3: 外部モードへの接続

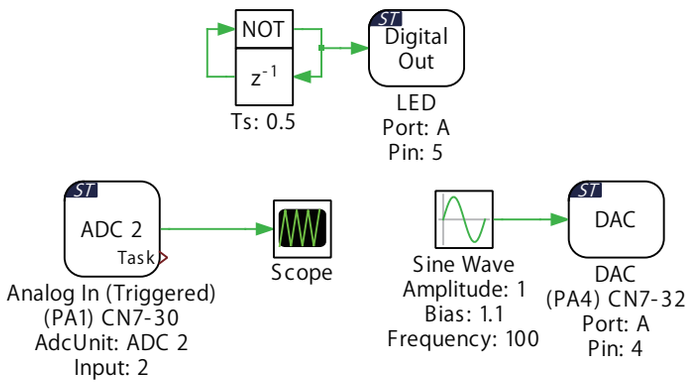
この演習では、外部モードに接続する方法を学習します。生成したコードはSTM32ターゲット上で実行され、外部モードを使用してPLECSアプリケーションのPLECSスコープをリアルタイム波形で更新できます。

課題3.1: PLECSモデルの修正

ここで、図8に示すように、**STM32 Target**ライブラリのDACブロックとAnalog In (Triggered)ブロックを使用して、単純なループバックモデルを作成します。

ジャンパ線を介してDACブロックをアナログ入力に外部接続すると、生成されたアナログ信号はAnalog In (Triggered)ブロックを介して感知し、モデル環境で使用できるようになります。オプションで、ADCブロックとDACブロックのパラメータウィンドウを介して、各チャネルのスケーリング係数とオフセット係数を設定できます。

図8: Analog Loopbackサブシステムの回路図



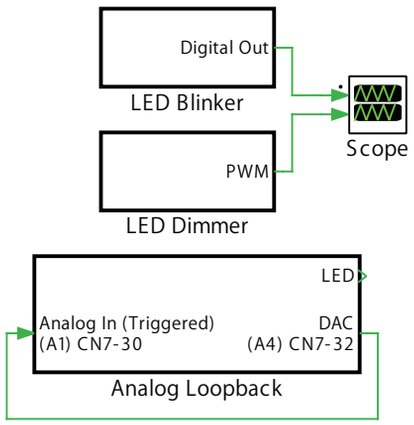
あなたのタスク:

1. 演習2と同じPLECSモデルファイルを使用して、"LED Blinker"サブシステムのコピーを作成し、名前を"Analog Loopback"に変更します。
2. LEDを点滅させる回路はそのままにして、この"Analog Loopback"サブシステム内で、**STM32 Target**ライブラリからDACブロックとAnalog In (Triggered)ブロックをドラッグアンドドロップします。
3. DACブロックのパラメータ:
   - **DAC selection**で**Port**を`A`、**Pin number**を`4`に設定します。
   - 図8のように、DACブロックを**振幅**`1`、**バイアス**`1.1`、**周波数**`100`Hz、**周波数と位相の単位**がヘルツ (p.u.)の正弦波信号に接続します。
4. Analog In (Triggered)ブロックのパラメータ:
   - **ADC unit** を`ADC2`として選択し、**Analog input channel**を`2`に設定します。
   - **Trigger source** を`Determine automatically`に設定します。
5. 上記で定義されていない他のすべてのパラメータはデフォルト値のままにします。
6. 図8のように、Analog In (Triggered)ブロックの出力をPLECSスコープに接続します。

課題3.2: PLECSでオフラインシミュレーションを検証

あなたのタスク: 最上位レベルの回路図で、図9のように、DAC出力ポートをAnalog In (Triggered)入力ポートに接続します。次に、新しいシミュレーションを開始すると、"Analog Loopback"サブシステムのPLECSスコープ内に適切な正弦波形が表示されます。

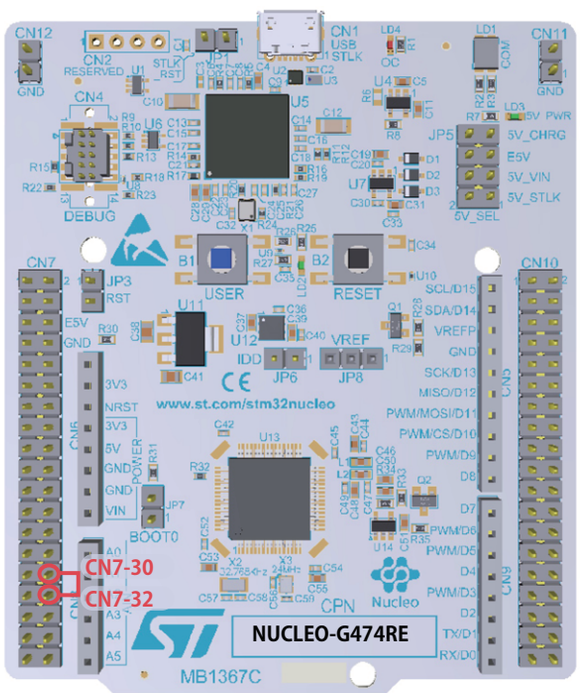
図9: トップレベルの回路図



課題3.3: 必要なハードウェアを接続

あなたのタスク: ジャンパ線を使用してピンPA4(CN7-32、DAC)をPA1(CN7-30、Analog In)に接続します(図10を参照)。

図10: ジャンパ線を使用してピンCN7-32をCN7-30に接続



課題3.4: Coderオプションを構成し、STM32 MCUをフラッシュさせる

あなたのタスク:

1 **Coder Options**ウィンドウで、左側のシステム リストの下にある"Analog Loopback"を選択します。次に、演習1で説明したように、コード生成機能の有効化を設定します。

2 **Coder Options**ウィンドウで**ターゲット**タブの**External Mode**サブタブに移動します。**外部モード**の通信オプションとして`JTAG`を選択します。**Target buffer size**は、外部モードの信号をバッファリングするために割り当てるターゲットメモリの量を指定します。`1000`のままにしておきます。

3 **ビルド**をクリックして、MCUをビルドしてプログラムします。

課題3.5: 外部モードに接続

生成したコードはSTM32ターゲット上で実行され、外部モードを使用してPLECSアプリケーションのPLECSスコープをリアルタイム波形で更新できます。

あなたのタスク:

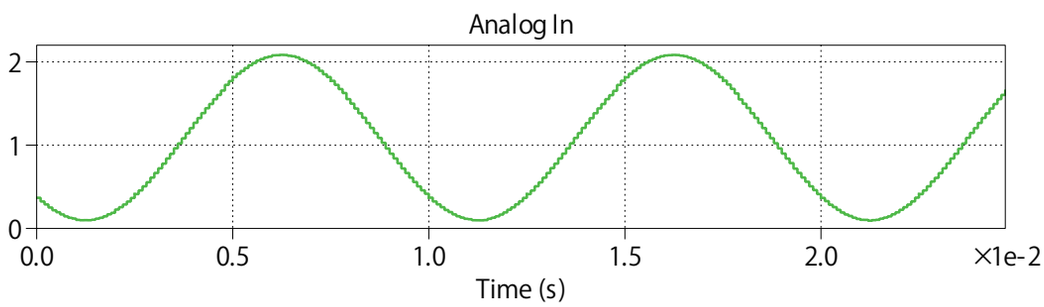
1 まず、**Coder Options**ウィンドウの左側にある**システム**リストから、目的のMCUを選択します。

2 次に、**外部モード**タブで アイコンをクリックして、組み込みターゲットを編集します。**デバイス種類**として`Serial over GDB`を選択し、**デバイス名**として`localhost`と入力します。

3 次に、**接続**をクリックし、**自動トリガを有効化**をクリックしてサブシステムのスコープでテスト結果を観察します。

この段階では、モデルとシミュレーション結果は`stm32_intro_3.plecs`と同じになっているはずです。キャプチャされたリアルタイムPWMおよびADC波形は、図11に示すものと同じになるはずです。

図11: Nucleo-G474REでキャプチャしたリアルタイムADC信号



演習4: パラメータのインライン化

モデルをビルドする前に、コンポーネントを**Coder Options**ウィンドウの**パラメータのインライン化**タブにある**例外**リストに追加すると、ターゲットデバイスの特定のパラメータ値をリアルタイムで変更できます。

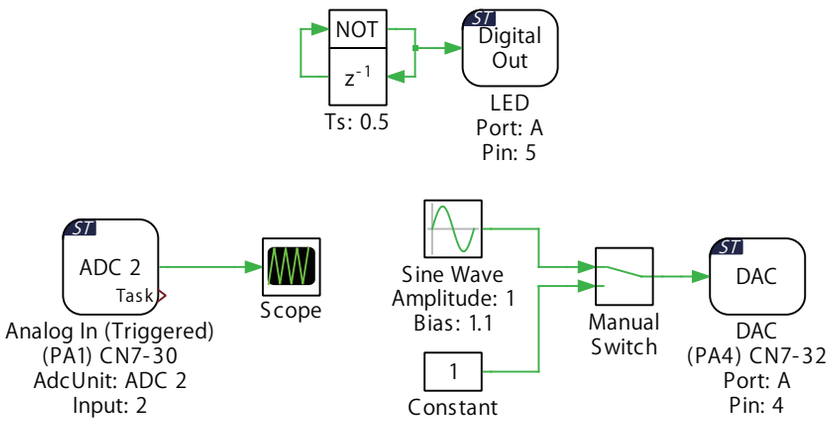
この演習では、いくつかのパラメータを調整します。つまり、これらのパラメータは、外部モードを介してターゲットデバイスに接続したときに変更できます。パラメータの変更は、有効になるとすぐにPLECSスコープのトレースに反映されます。

課題4.1: PLECSモデルの修正

あなたのタスク:

1. 演習3の"Analog Loopback"サブシステムが外部モードに接続されている場合は、切断します。
2. 演習3と同じPLECSモデルファイルを使用して、"Analog Loopback"に**定数**ブロックと**手動切替スイッチ**ブロックを追加します(図12を参照)。

図12: 編集したAnalog Loopbackサブシステムの回路図



課題4.2: パラメータのインライン化の設定、STM32 MCUのフラッシュおよび外部モードへの接続

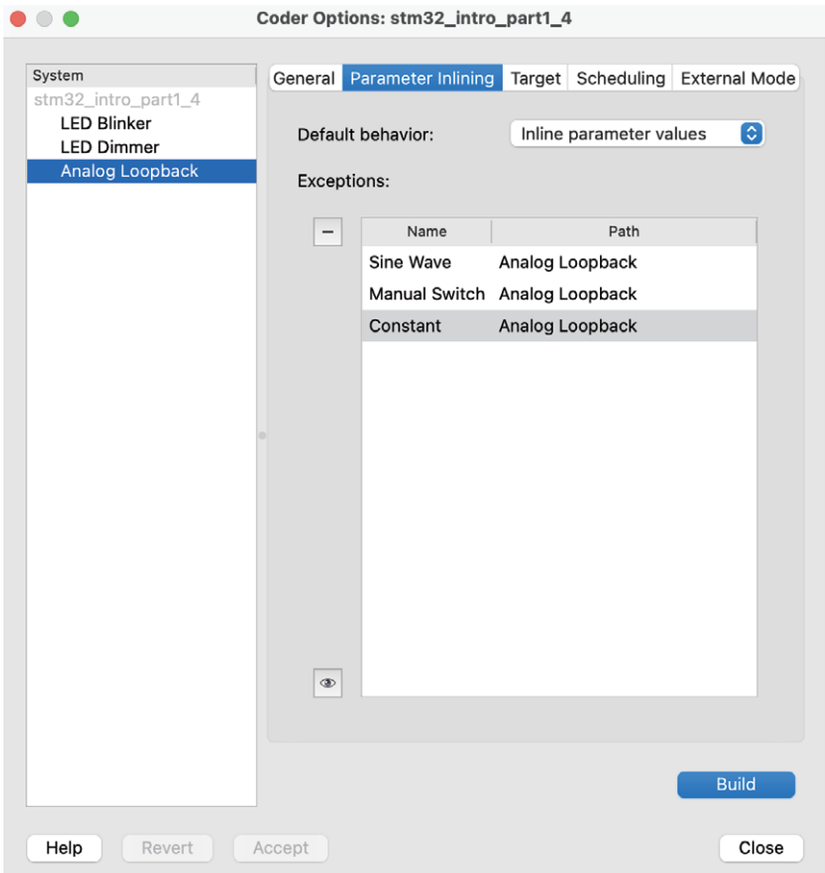
あなたのタスク:

1. Coder Optionsウィンドウの**パラメータのインライン化**タブに移動し、図13に示すように、"正弦波信号"、"定数"、"手動切替スイッチ"ブロックを**例外**リストウィンドウにドラッグアンドドロップします。
2. **デフォルト設定**でインラインパラメータを選択します。
3. STM32 MCU上に"Analog Loopback"サブシステムを**ビルド**し、演習3の指示に従って外部モードに**接続**します。
4. これで、DACブロックから生成される正弦波を定数値に簡単に切り替えられるようになり、また、定数の値、正弦波信号の**振幅**と**バイアス**を変更して、PLECSスコープでこれらの変化をリアルタイムで観察できるようになります。

この段階では、モデルとシミュレーション結果は`stm32_intro_4.plecs`と同じになっているはずです。

オプション: STM32 MCUは0～3.3Vの値のみを送信または受信できます。この範囲外の値は切り捨てられます。DACブロックとAnalog Inブロックのスケーリングとオフセットパラメータを使用して、0V未満または3.3Vを超える値を送受信してみます。

図13: パラメータのインライン化の構成



参考文献

[1]Getting Started with the STM32 Target Support Package:
https://www.plexim.com/support/videos/coder-stm32-getting-started

[2]STM32 Target Support User Manual: https://plexim.com/sites/default/files/stm32manual.pdf
日本語訳: https://adv-auto.co.jp/products/plexim/manual.html

[3]STMicroelectronics, NUCLEO-G474RE: https://www.st.com/en/evaluation-tools/nucleo-g474re.html

[4]STM32G4 Nucleo-64 boards (MB1367) User manual:
https://www.st.com/resource/en/user_manual/um2505-stm32g4-nucleo64-boards-mb1367-stmicroelectronics.pdf

改訂履歴:

Tutorial Version 1.0　　初版

Plexim

Pleximへの連絡方法:

☎ +41 44 533 51 00　Phone

+41 44 533 51 01　Fax

✉ Plexim GmbH　Mail
Technoparkstrasse 1
8005 Zurich
Switzerland

@ info@plexim.com　Email

http://www.plexim.com　Web

KESCO KEISOKU ENGINEERING SYSTEM
計測エンジニアリングシステム株式会社
https://kesco.co.jp

Embedded Code Generation Tutorial