カスタムマイコン設計トライアル

～MAX® 10 FPGA を使用した組み込み設計を体感～

演習マニュアル

(MAX® 10 開発キット編)

2020 年 10 月 Rev.1

カスタムマイコン設計トライアル　演習マニュアル

(MAX 10　開発キット編)

目次

本書をお読みになる前に

この資料の内容は 2020 年 10 月現在のものです。

この資料で紹介しているソフトウェアやハードウェア、操作手順などは、指定バージョンやデバイス等以外でも共通のものもありますが、一部については共通にならないものもありますので、ご注意ください。

文書中の記号

ⓘ **Note**	補足情報などを記載しています。
Ⓟ **Point**	重要なポイントを記載しています。
参考	理解を深めるため、参考となる資料やサイトを紹介しています。
⚠ **注記**	この資料の中では具体的には触れませんが、必要となる知識や情報を記載しています。
⃠ **禁止**	注意点や、してはいけないことを記載しています。

文中の表記

下線	クリックする事で、資料中の別の章や、外部のサイトにジャンプします。
太字斜体	画面の操作をする際の、メニューやウィンドウなどに表示されている文字を示しています。
xxxxxxx↵	入力するコマンド文字列を示しています。
網掛け	使用するツールを示しています。

はじめに

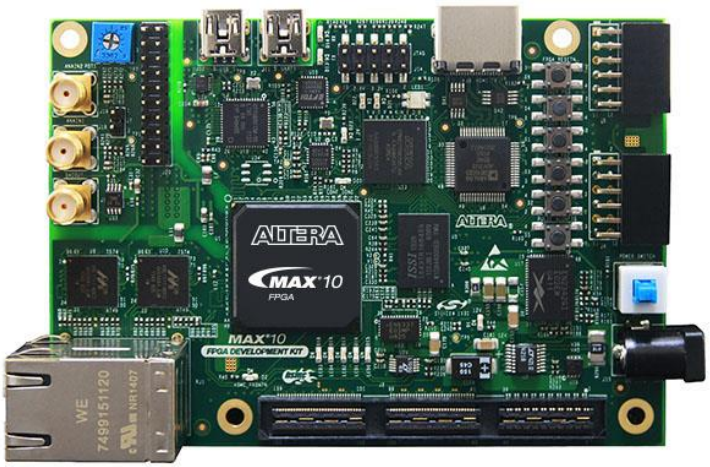
この資料は、「カスタムマイコン設計トライアル」受講者向けの演習マニュアルです。

すべての演習を進めるにあたり、各演習でそれぞれのステップの作業を漏れなく実行する必要があります。そのため、進行経過が確認できるように各ステップの先頭に(____)を設けています。そのステップの作業が完了したら、そこへチェック・マークをつけるようにしてください。

演習で使用する開発環境

この演習を実施するためには、以下の環境が必要です。

【表 1-1】演習実施に必要な開発環境

No.	環境	概要
1	パソコン	Quartus® Prime が動作するスペックを保有するパソコンをご用意ください。 ・サポート OS 情報は、Table 1: Operating System Support をご覧ください。 ・搭載するメモリ容量の情報は、使用するバージョンの Table 2: Release Notes and readme.txt Files をご覧ください。
2	Quartus Prime Standard Edition または Quartus Prime Lite Edition	開発ソフトウェアをインストールしてください。 各種ソフトウェアのダウンロードおよびインストール方法は、以下の Web ページをご参照ください。 ・インテル Quartus Prime 開発ソフトウェアおよび ModelSim - Intel FPGA Edition のダウンロード方法 ・インテル Quartus Prime 開発ソフトウェアおよび ModelSim - Intel FPGA Edition のインストール方法
3	Nios® II Software Build Tools for Eclipse	
4	開発ボード	このチュートリアルに対応した開発 ボードは以下のとおりです。 ・MAX® 10 FPGA 開発キット ：10M50DAF484C6GES 

No.	環境	概要
5	インテル FPGA ダウンロード・ケーブル II 用ドライバー	MAX 10 開発キットは、オンボード インテル ダウンロード・ケーブル II 対応 です。 ダウンロード・ケーブルをはじめて利用するパソコン をご利用の際は 、事前にドライバ ー をインストールしてください。 インストール方法は、以下の Web ページをご参照ください。 ・ USB-Blaster™ II のドライバをインストールしてみよう
6	演習データ	このマニュアルを入手した Web ページ からダウンロードしてください。ファイルの解凍先は任意です。 ⚠ **注記:** 解凍先は、全角やスペースの含まれないフォルダーパスを指定してください。 **なお このマニュアルは、演習データの展開先を下記フォルダーとして作成しています。** ***C:/intelFPGA_prj/***

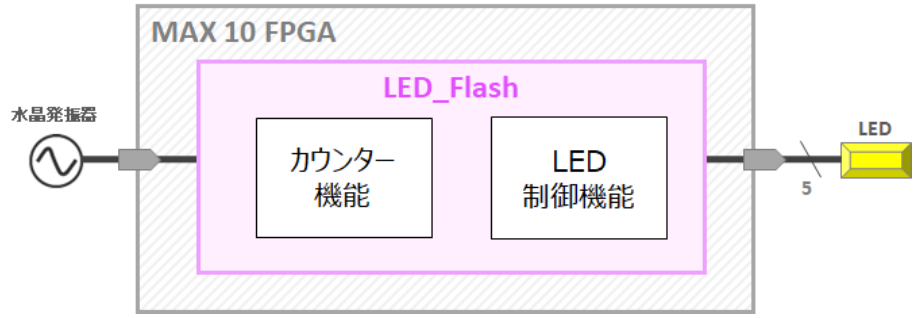
演習 1.　FPGA　設計フローを体験

インテル　Quartus Prime　開発ソフトウェア　(以下、Quartus Prime) を使用し、MAX 10 FPGA　の開発フローを体験しましょう。

- **演習で使用する回路の概要**

この演習では、VHDL　もしくは　Verilog HDL　で　すでに記述されたデザイン　(デジタル論理回路) を使用します。

デザインのブロックイメージは、以下のとおりです。



- **論理回路の動作**

ボード上の 5　個の　LED　が 1　秒間隔で点滅します。

- **主な作業内容**
  - プロジェクト作成
  - デザイン確認
  - ピン配置設定の確認
  - コンパイル実行
  - FPGA　へダウンロード

1. プロジェクト作成

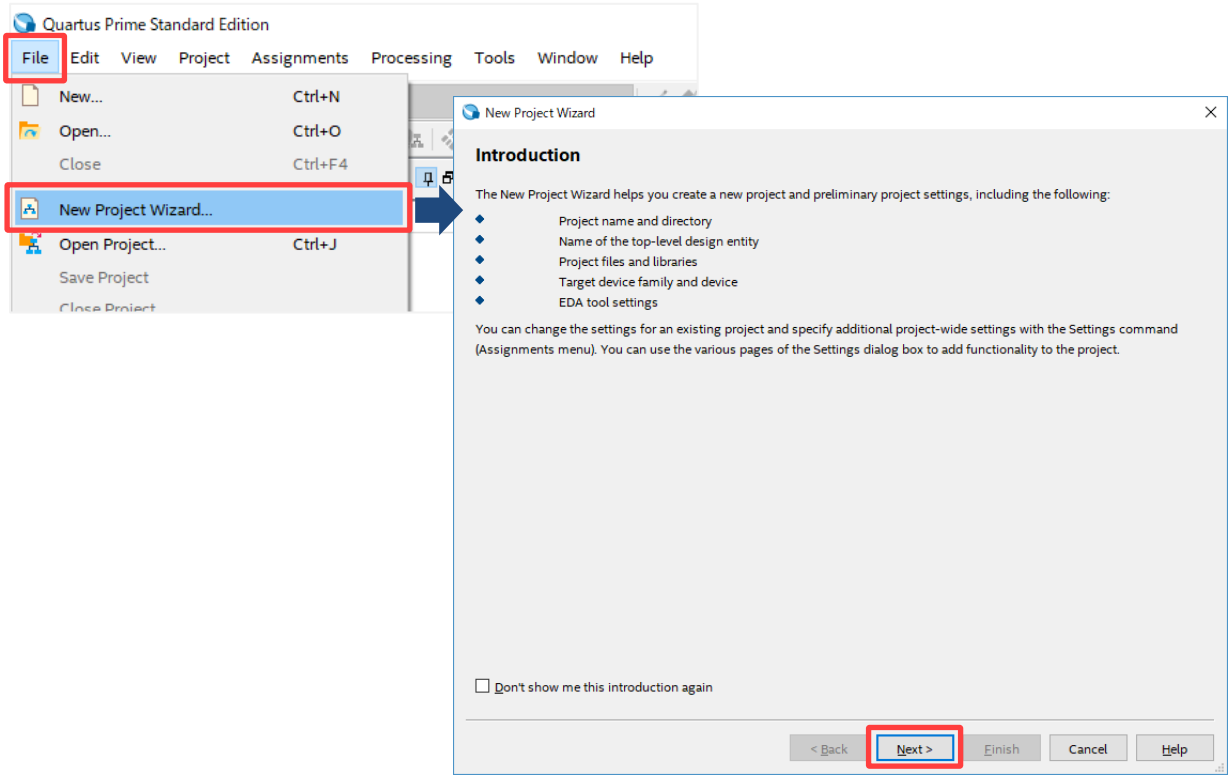
以下の手順に沿って Quartus Prime のプロジェクトを作成します。

____1. Quartus Prime を起動します。

Windows OS の場合は、***[スタート]*** ➤ ***Intel FPGA <version_build> Standard Edition*** ➤ ***Quartus (Quartus Prime <version>)*** をクリック、もしくはデスクトップに生成した Quartus Prime のショートカット・アイコンをダブルクリックしてください。

Linux OS の場合は、ターミナル上で quartus コマンドを実行し、起動させます。

____2. Quartus Prime のメニューから ***File*** ➤ ***New Project Wizard*** を選択してウィンドウを開き、***Introduction*** ページで ***[Next]*** ボタンをクリックします。



【図 1-1-1】 New Project Wizard / Introduction ページ

____3. ***New Project Wizard： Directory、Name、Top-Level Entity ページ*** において、作業フォルダー、プロジェクト名、最上位階層のエンティティ名を入力します。【図 1-1-2】を参考に、以下の手順に従ってください。

① 上段に、作業フォルダーを指定します。

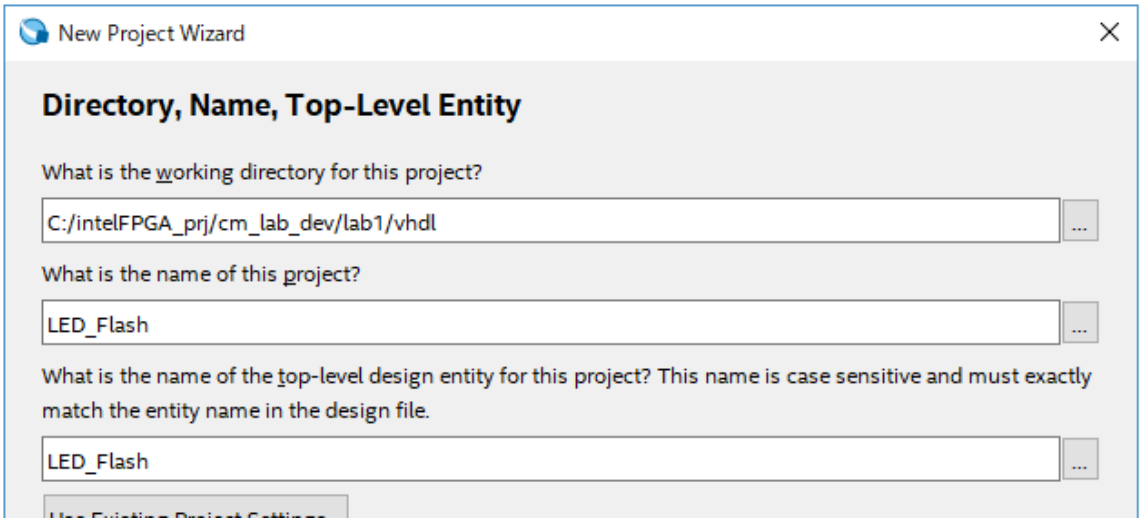
右端のボタン（ブラウズボタン）をクリックし、演習 1 用の作業フォルダーを指定します。

/cm_lab_dev/lab1 フォルダーには、VHDL 用（vhdl フォルダー）と Verilog HDL 用（Verilog フォルダー）の作業フォルダーを用意しています。いずれかを選択し、***[フォルダーの選択]*** ボタンをクリックします。

② 中段に、プロジェクト名を入力します。この演習では ***LED_Flash*** と入力します。

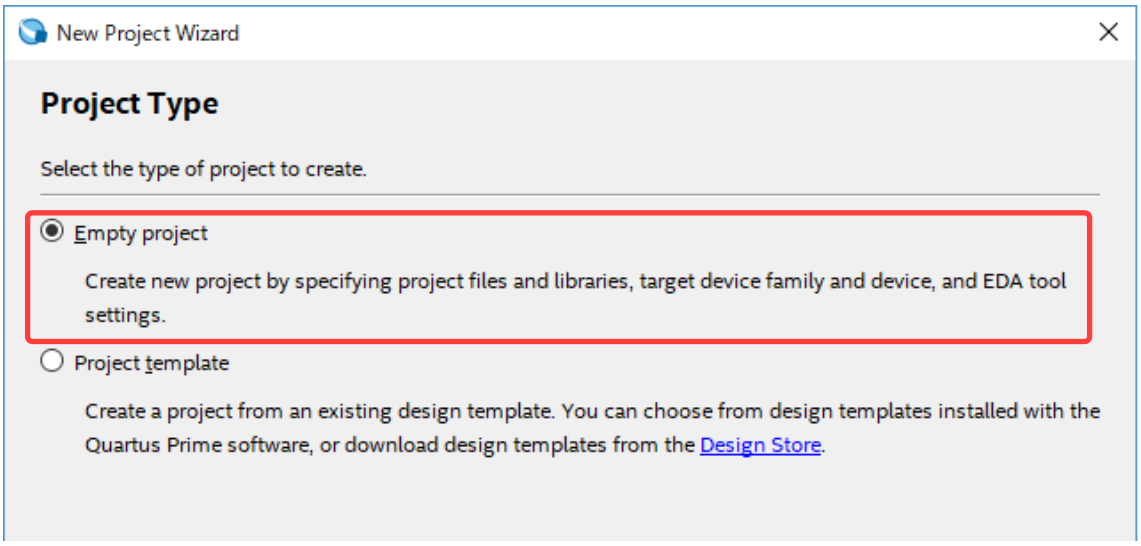
③ 下段に、最上位階層のエンティティ名を入力します。この演習では ***LED_Flash*** と入力ます。

④ ***[Next]*** ボタンをクリックします。



【図 1-1-2】 New Project Wizard： Directory、Name、Top-Level Entity ページ (VHDL フォルダーを選択した場合)

____4. ***New Project Wizard： Project Type ページ*** において、***Empty project*** を選択後、***[Next]*** ボタンをクリックします。



【図 1-1-3】 Project Type ページ

____5. ***New Project Wizard： Add Files ページ*** において、回路を構成する既存のデザインファイルを選択します。

File name 欄右端のボタン (ブラウズボタン) をクリックし、【表 1-1-1】を参考に、作業フォルダー内にあるデザインファイルを選択して ***[開く]*** ボタンをクリックします。

【表 1-1-1】プロジェクトに登録するデザインファイル

選択した言語	ファイル名
VHDL	LED_Flash.vhd
Verilog HDL	LED_Flash.v

ウィンドウ内の一覧に、LED_Flash.vhd もしくは LED_Flash.v が追加されたことを確認し、***[Next]*** ボタンをクリックします。

____6. ***New Project Wizard： Family, Device & Board Settings*** **ページ** において、開発ボードに搭載されているターゲット・デバイスの型番を選択します。

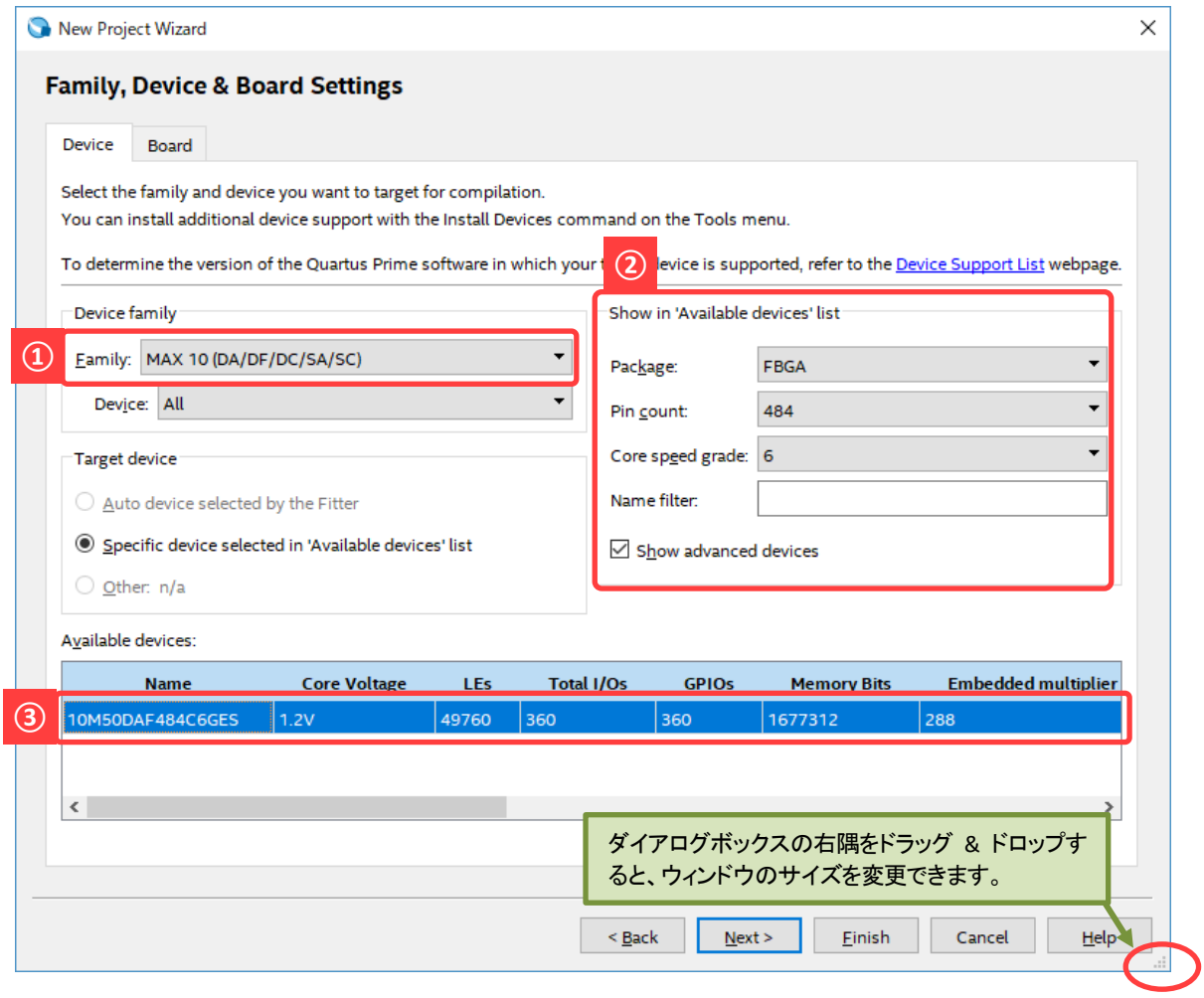
- ターゲット・デバイスの型番： 10M50DAF484C6GES

【図 1-1-4】を参考に、以下の手順に従ってください。

① ***Family*** 欄において、***MAX 10 FPGA (DA/DF/DC/SA/SF/SC)*** を選択します。

② ***Show in 'Available devi ces' list*** セクションで、型番に応じた条件を指定します。

③ ***Available devices*** 欄から ***10M50DAF484C6GES*** をハイライト選択します。



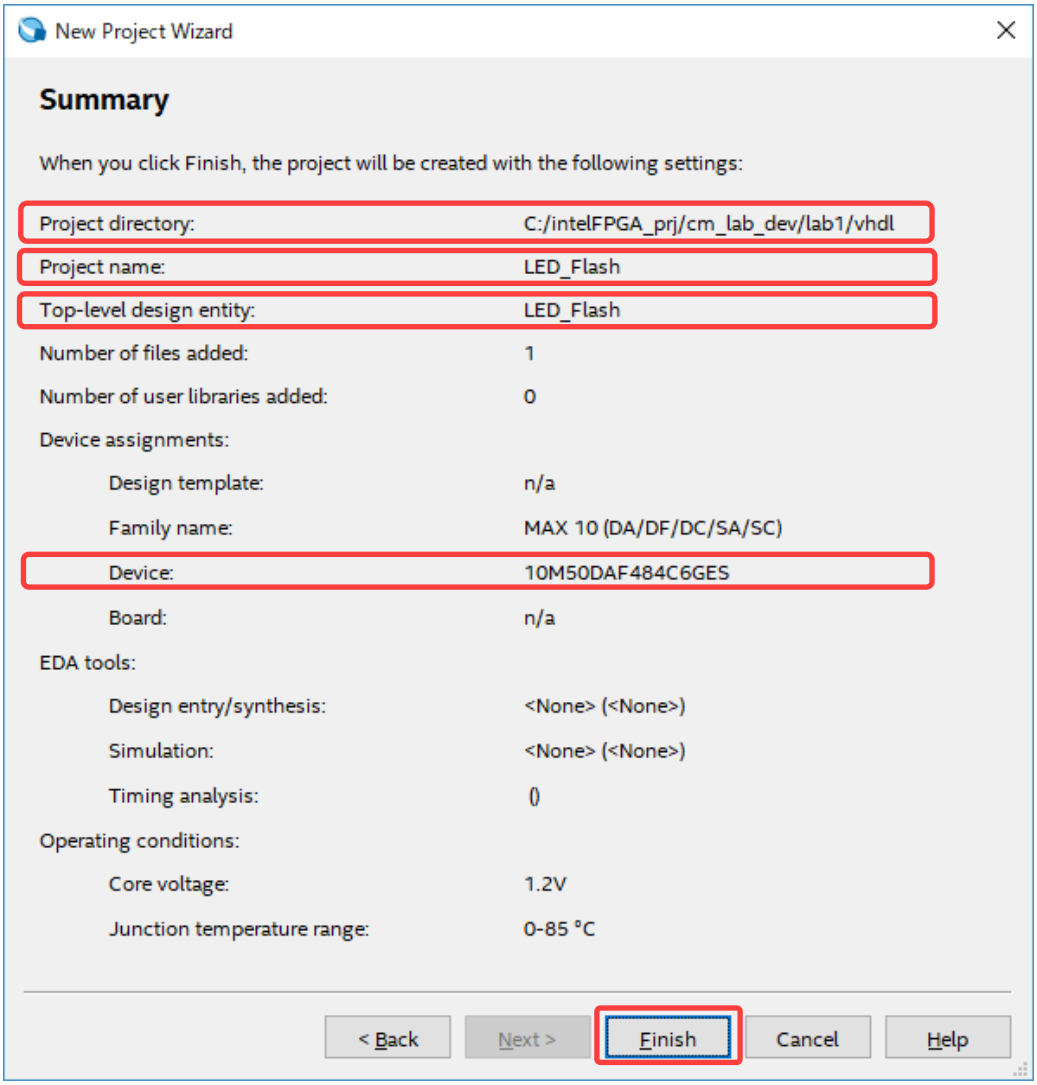
【図 1-1-4】Family, Device & Board Settings ページ

④ ***[Next]*** ボタンをクリックします。

____7. ***New Project Wizard : EDA Tool Settings ページ*** は、この演習では使用しないため ***[Next]*** ボタンで次へ進みます。

____8. ***New Project Wizard : Summary ページ*** で、設定した内容を再確認します。

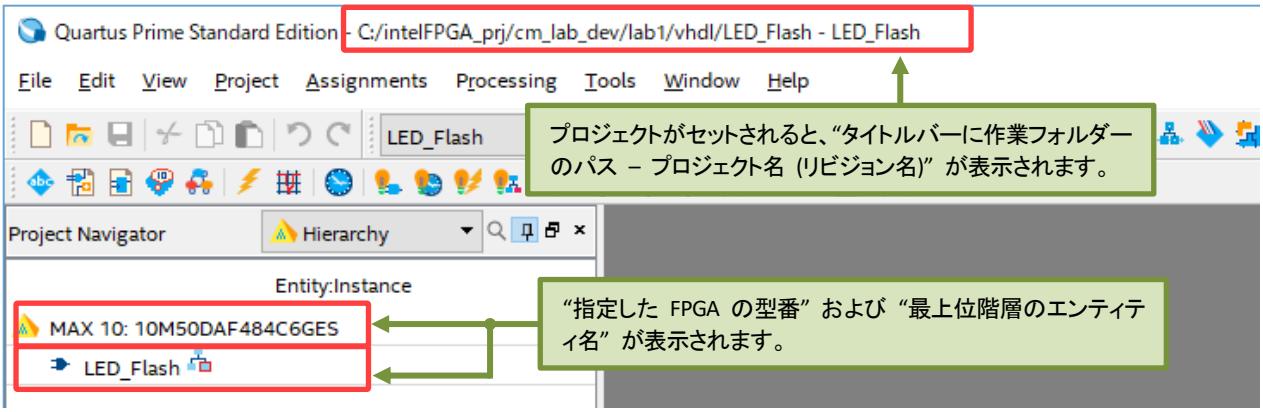
- Project directory (作業フォルダーのパス)
- Project name (プロジェクト名)
- Top-level design entity (最上位階層のエンティティ名)
- Device (Device assignments 内) (デバイス型番)



【図 1-1-5】 Summary ページ (例: VHDL)

問題が無ければ ***[Finish]*** ボタンをクリックします。

以上で、プロジェクト作成が完了しました。



【図 1-1-6】 プロジェクトをセットした後の Quartus Prime 画面 (例: VHDL)

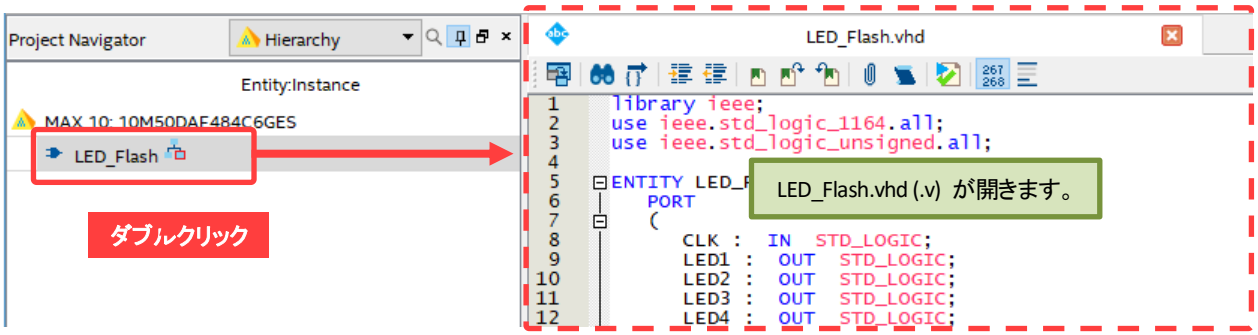
2. デザイン確認

プロジェクトに追加した LED_Flash.vhd もしくは LED_Flash.v ファイルのコードを確認します。ファイルを開いてみましょう。

このファイルは、本紙 6 ページ“演習で使用する回路の概要”を VHDL / Verilog HDL で記述したものです。

今回 デザインを編集する必要はありません。

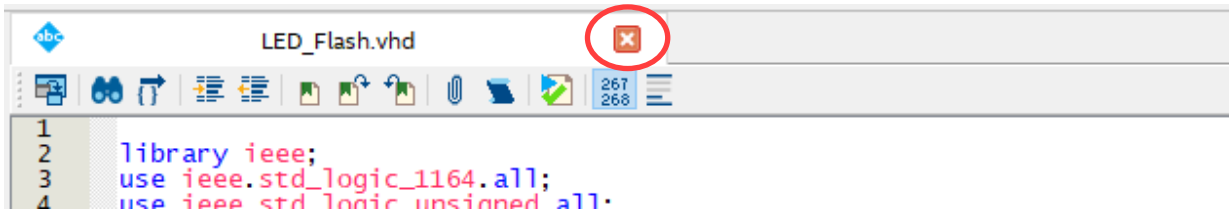
____1. ***Project Navigator ウィンドウ*** (Quartus Prime 画面左上) の LED_Flash をダブルクリックしてください。最上位階層のデザインファイルが開きます。



【図 1-2-1】 デザインファイルを開く (例: VHDL)

> ⓘ Note:
> Quartus Prime のメニューから ***File*** ➤ ***Open*** を選択し、ファイルを指定して開く方法もあります。

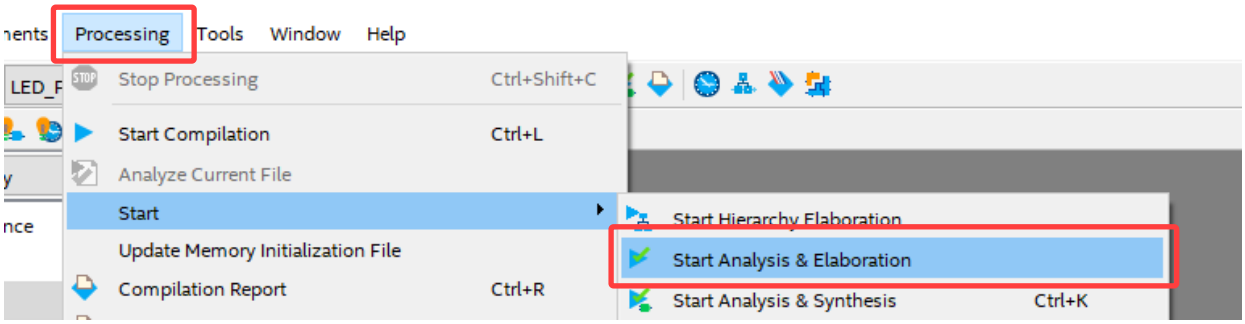
コードを確認したら、ワーキングシートのタブ右端にある *[X]* ボタンをクリックし、ファイルを閉じます。



【図 1-2-2】 デザインファイルを閉じる

____ 2. デザインを初期チェックするため、Quartus Prime のメニューから ***Processing*** ➤ ***Start*** ➤ ***Start Analysis & Elaboration*** を実行します。

今回は完成したデザインを使用しているため、エラーが発生することなく Analysis & Elaboration のステップが終了するはずです。



【図 1-2-3】 Start Analysis & Elaboration

以上で、デザインの確認は終了です。

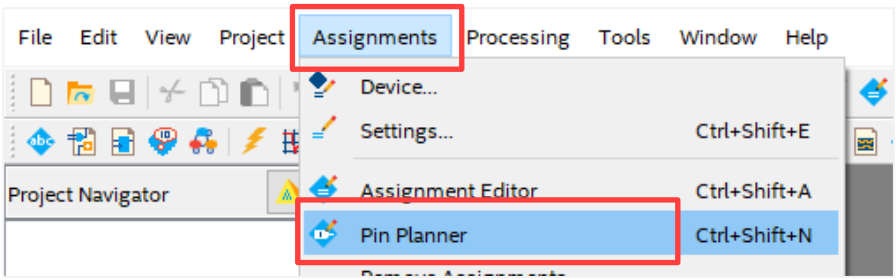
3. ピン配置設定の確認

デザインが完成したら、コンパイルを実行します。

実際の設計では、Quartus Prime でコンパイルを実行する前にいくつかの設定を行う必要がありますが、今回の演習ではあらかじめ設定された環境が提供されています。

ここでは、ユーザー設定のうち ピン配置設定がどのようになっているのかを確認してみましょう。

____ 1. Quartus Prime のメニューから ***Assignments*** ➤ ***Pin Planner*** を選択し、Pin Planer を起動します。

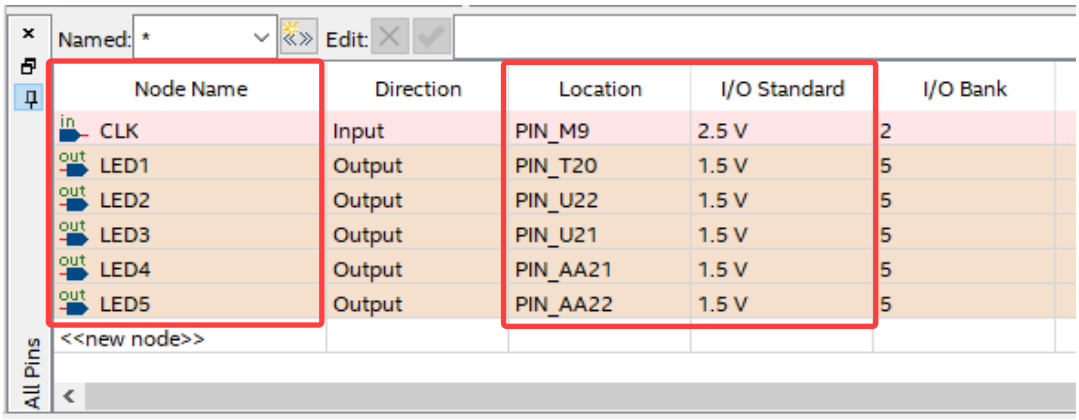


【図 1-3-1】 Pin Planner

____ 2. Pin Planner の ***All Pins リスト*** (ウィンドウ下部) に、設定済みのピンの割り当て (ピンアサイン) が表示されています。

今回使用している FPGA のピンは、ボード上の水晶発振器からのクロック信号を入力する CLK と、ボード上の LED をドライブするための出力 LED 5 本のみです。

各ピンに対して、Location 項にピン番号、I/O Standard 項に I/O 規格が、開発ボードに合わせた内容であらかじめ入力されていることが確認できます。



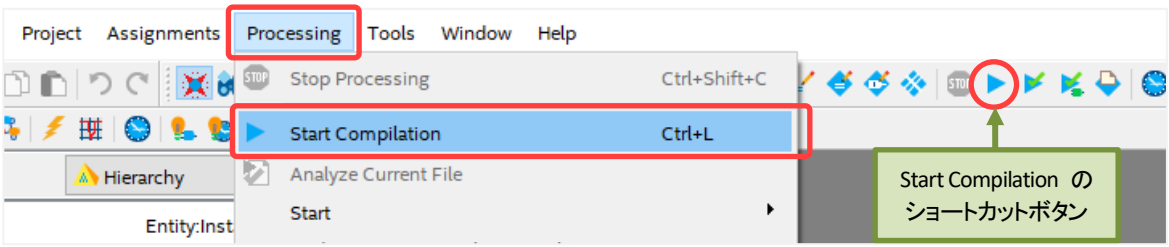
【図 1-3-2】 All Pins リスト (Pin Planner)

____ 3. ピンアサイン設定が確認できたら、Pin Planner 上のメニューから ***File*** ➤ ***Close*** を選択し、Pin Planner を閉じます。

4. コンパイル実行

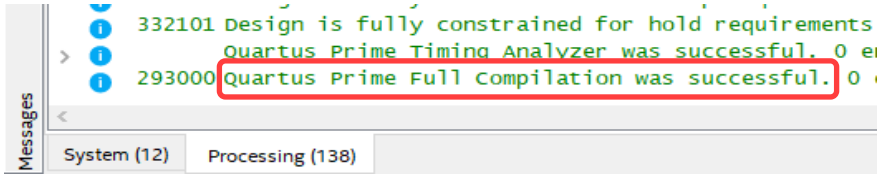
コンパイルを実行し、デザインの論理合成と配置配線などを行います。

____ 1. Quartus Prime のメニューから **Processing ➤ Start Compilation** をクリックし、コンパイルを実行します。



【図 1-4-1】 コンパイル実行

____ 2. Messages ウィンドウに、Quartus Prime Full Compilation was successful. のインフォメーション・メッセージが表示されたら、エラーが発生することなくコンパイルが終了しています

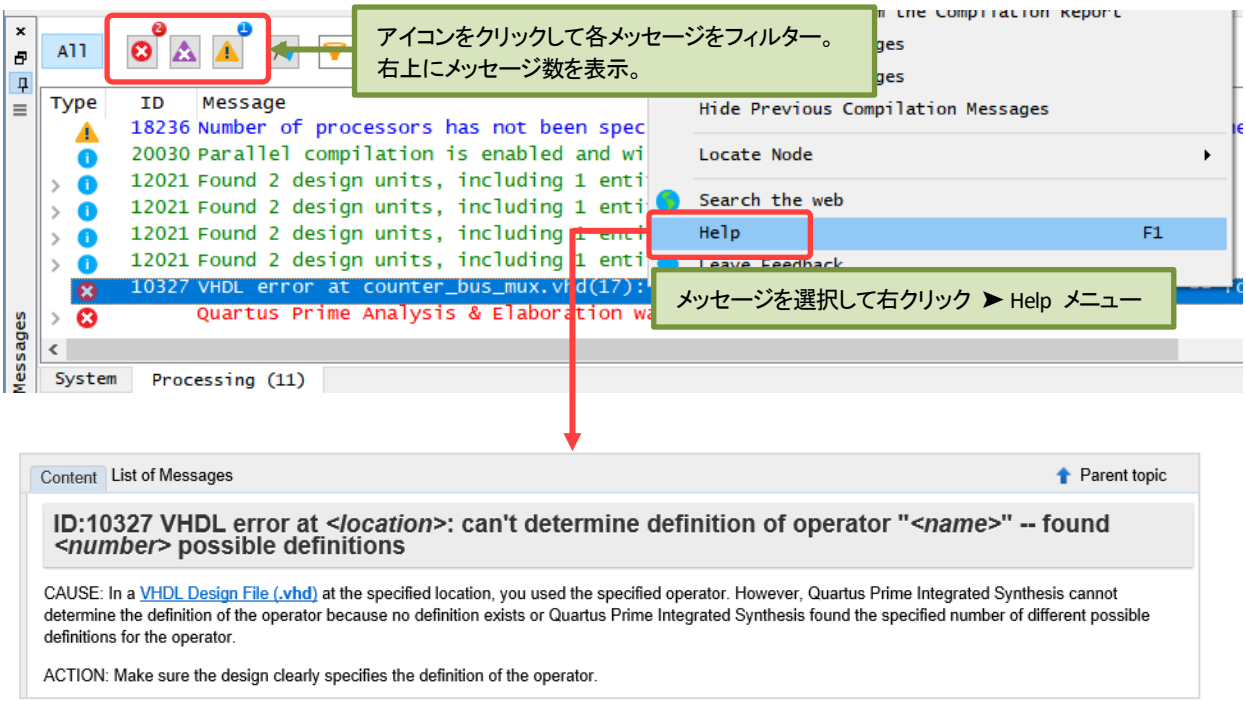


【図 1-4-2】 Messages ウィンドウ

この演習はエラーが発生することなくコンパイルが完了するはずです。もしエラーが発生した場合は、解消しないと次のステップには進めません。回避するためにエラーの原因を追究し修正する必要があります。

また、エラー・メッセージの他、ワーニング・メッセージが発生する場合もあります。ワーニングは解消しなくても次のステップへ進めますが、必ず内容を確認し、その内容を回避すべきか無視できるのかをユーザーが判断してください。

各メッセージ内容の詳細を確認するには、ヘルプ機能の活用が有効です（【図 1-4-3】参照)。 ヘルプには、メッセージが発生した要因や回避するためのヒントが掲載されています。その情報からエラーやワーニングを回避してください。



【図 1-4-3】 Quartus Prime Help の活用

以上で コンパイルは完了です。

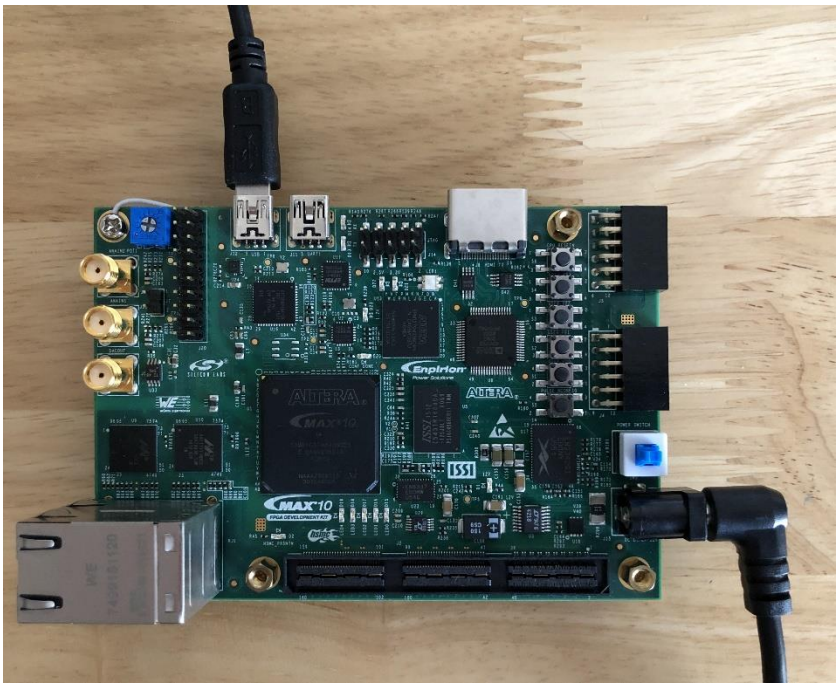
5. FPGA へダウンロード

コンパイルにより生成されたデザインのデータを、ボード上の FPGA にダウンロードし、動作を確認します。

> ⚠ **注記:**
> 実際の開発では、コンパイル後に Quartus Prime の Timing Analyzer によるタイミング検証を行い、期待どおりの動作が実現できるかを検証します。期待するタイミングを満足できることが確認できたら、ボード上のデバイスへデータを書き込みます。
> 今回の演習ではタイミング検証を省略していますが、自身の開発時は必ずタイミング検証を行った上でデバイスへの書き込みを行ってください。

> ⓘ Note:
>
> MAX 10 開発キットは、FPGA へデータをダウンロードする際 ボードに組み込まれたインテル FPGA ダウンロード ・ ケーブル II (旧称 USB Blaster™ II) 回路を使用します。パソコンとボードの接続は、キットに付属している mini-USB ケーブルで行います。パソコンとボードが通信するためには、インテル FPGA ダウンロード・ケーブル II 用のドライバーをインストールする必要があります。インストール方法は、【表 1-1】No. 5 をご覧ください。

____ 1. キット付属の電源コネクタを接続し、ボードの電源が OFF であることを確認してください。その後、付属の mini-USB ケーブルを用いてボードとパソコンを接続します。

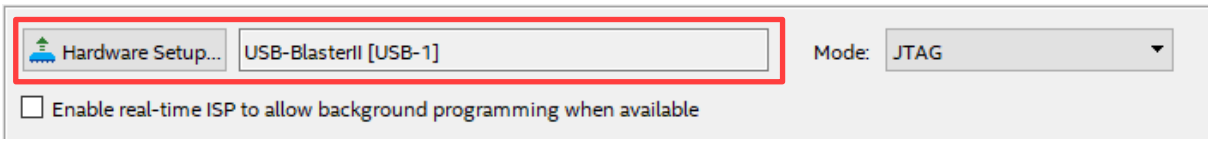


【図 1-5-1】 パソコンととボードを接続

____ 2. 開発ボードへ電源を供給します。

____ 3. Quartus Prime のメニューから ***Tools*** ➤ ***Programmer*** を選択し、Programmer を起動します。

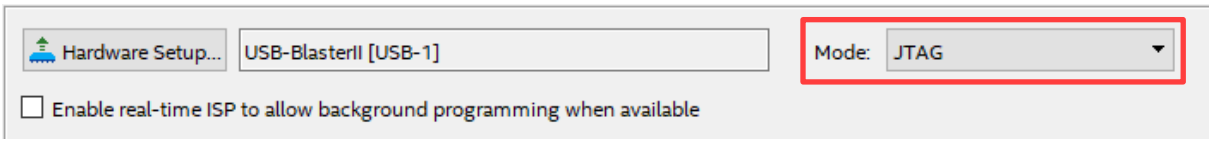
____ 4. Programmer の ***[Hardware Setup]*** ボタン 右横の欄が、***USB-Blaster II*** と表示されていることを確認します。



【図 1-5-2】 Hardware Setup

もし No Hardware 表示の場合には、***[Hardware Setup]*** ボタン をクリックし、***Hardware Settings タブ*** の ***Currently selected hardware*** 項のプルダウン・リストから、***USB-Blaster II*** を選択し、***[Close]*** ボタンをクリックしてください。

____5. Programmer の ***Mode*** プルダウン・リストから ***JTAG*** を選択します。



【図 1-5-3】 Mode 選択

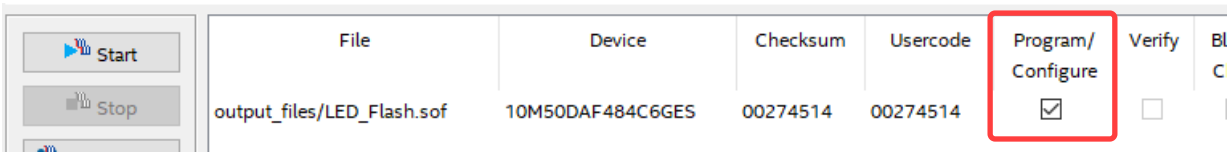
____6. Programmer の ***Files*** 欄に、ダウンロードするファイル ***LED_Flash.sof*** が選択されていることを確認します。

<none> となっている場合には、***[Add File]*** ボタン をクリックし、起動したファイルブラウザから ***LED_Flash.sof*** を選択します。(作業フォルダー内 ***output_files フォルダー*** に生成されています。)

【図 1-5-4】 ダウンロードする sof ファイルを指定

____7. プログラミング・オプションを選択します。

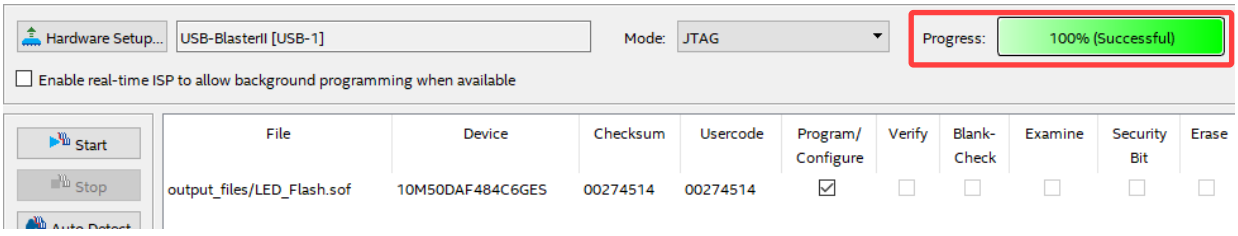
今回の演習では、FPGA の SRAM 領域にデータを転送するため、***Program/Configure*** にチェックを入れてください。



【図 1-5-5】 プログラミング・オプションを選択

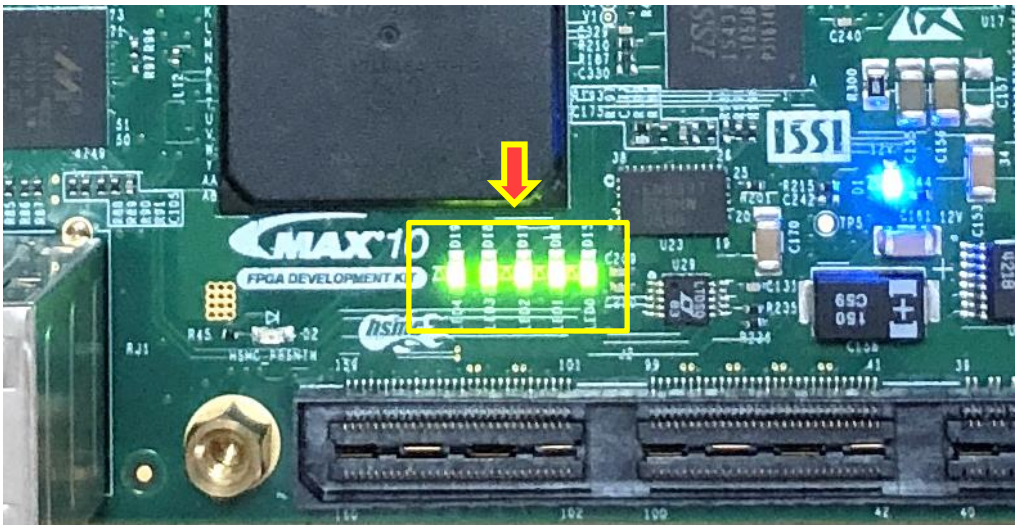
____8. Programmer の ***[Start]*** ボタン をクリックし、データの書き込みを開始します。

Progress バーが 100% になったら書き込み完了です。Messages ウィンドウには Successful のインフォメーションが表示されます



【図 1-5-6】 書き込み完了

開発ボード上の LED (5 個) が 1 秒間隔で点滅していれば、正しく回路データがダウンロードされています。



【図 1-5-7】 開発ボード上の LED の位置

____ 9. FPGA の動作を確認後、ボードの電源を OFF にします。その後、再び接続し、ボードの電源を投入してください。先ほど書き込んだ MAX 10 の動作は消去され、データを書き込む前の状態に戻っていることが確認できます。

> ⓘ Note:
>
> MAX 10 内部の SRAM 領域にデータを書き込んだ場合は 電源 OFF 時にデータを保持することができません。MAX 10 内部の不揮発性メモリ領域 (CRAM) にデータを書き込む際は、pof ファイルを使用します。(この演習では実施しません。)

以上で、演習 1 の作業は終了です。

※ 時間に余裕がある場合には、以下のオプション演習を実施してみましょう。

実施しない場合には、Quartus Prime のメニューから ***File*** ➤ ***Close Project*** を選択し、演習 1 のプロジェクトを終了してください。その後、開発ボードの電源を OFF にしてください。

＜オプション演習＞

演習 1 で作成したプロジェクトのデザインを編集し、LED の点滅間隔を半分にしてみましょう。

作業内容は以下のとおりです。

- デザインを編集
- コンパイル実行
- FPGA へダウンロード

もし Quartus Prime の GUI を閉じてしまった場合は、再度 Quartus Prime を起動し、Quartus Prime のメニューから ***File*** ➤ ***Open Project*** を選択し、作業フォルダーから ***LED_Flash.qpf*** を開いて 演習 1 で作成したプロジェクト環境を開きます。

____ 1. ***Project Navigator ウィンドウ*** (Quartus Prime 画面左上) において ***LED_Flash*** をダブルクリックし、デザインファイルを開きます。

____ 2. 下記のようにコードを編集してください。

① VHDL の場合: 32 行目

if (div_cntr2 = X"2FA") then ➔ if (div_cntr2 = X"17E") then

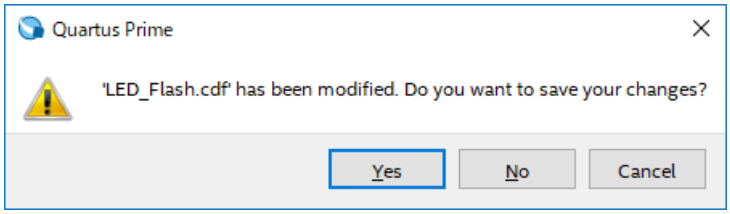
② Verilog HDL の場合: 26 行目

if (div_cntr2 == 762) ➔ if (div_cntr2 == 382)

____ 3. Quartus Prime のメニューから ***File*** ➤ ***Save*** で編集したファイルを保存します。

____ 4. Quartus Prime のメニューから ***Processing*** ➤ ***Start Compilation*** を選択し、コンパイルを実行します。

※ Programmer ウィンドウを開いたままコンパイルを実行すると、下記のメッセージ・ウィンドウが表示される場合があります。ここでは ***No*** を選択してメッセージ・ウィンドウを閉じます。



【図 1-5-8】 メッセージ

____ 5. 再コンパイルにより生成された LED_Flash.sof を MAX 10 FPGA にダウンロードします。

操作手順は、14 ページの 5. FPGA へダウンロード を参考にしてください。

LED の点滅間隔は短くなりましたか?

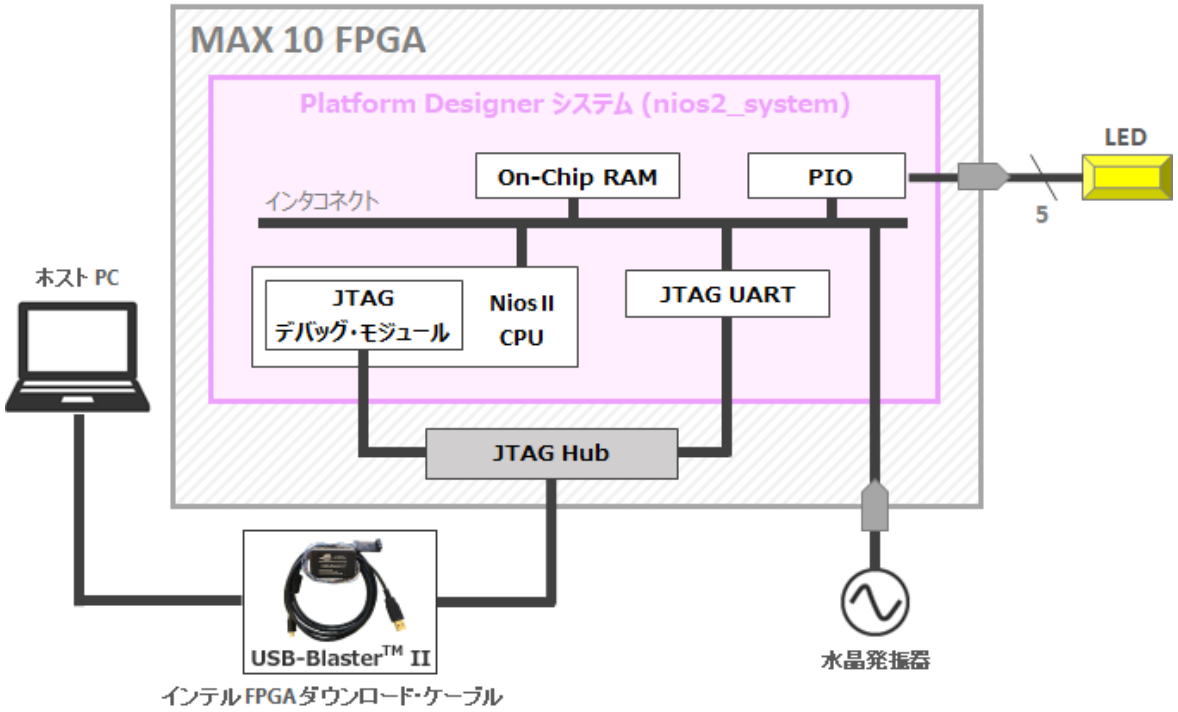
すべての作業が終了したら、Quartus Prime のメニューから ***File*** ➤ ***Close Project*** を選択し、演習 1 のプロジェクトを終了してください。その後、開発ボードの電源を OFF にしてください。

以上で、演習 1 の作業はすべて終了です。

演習 2.　FPGA を使用したカスタムマイコンの作成

MAX 10 FPGA に Nios® II プロセッサー、JTAG UART、PIO、On-Chip メモリ を実装し、Nios II にサンプルの C 言語ソースコードを実行させ LED を制御させる簡単なシステムを構築しましょう。

- **演習で構築するシステムの概要**



JTAG UART というインテル FPGA 特有のシリアル・ポートを利用し、ホストとのシリアル通信をインテル FPGA ダウンロード・ケーブル II を使用して、JTAG ポート経由で行うことができます。これにより、ソフトウェア・プログラムで printf() 関数を使用したコンソール出力を JTAG 経由で行うことが可能となります。JTAG デバッグ・モジュール経由でのデバッグについても、インテル FPGA ダウンロード・ケーブル II を介して行われます。

今回の演習では、PIO (Parallel I/O) を経由して、外部の LED を点滅させるソフトウェア・プログラムを作成しますが、これらの周辺ペリフェラルは Nios II プロセッサー用内部バスである プラットフォーム・デザイナー・インタコネクトで接続されます。

上図の組み込みシステムは、システム名 nios2_system というブロックとして FPGA 内に実装されます。

- 主な作業内容
  - プラットフォーム・デザイナーでシステム・モジュールを作成
  - ハードウェア・デザイン作成
  - ハードウェア・デザインを FPGA へダウンロード
  - ソフトウェアを実行

1. プロジェクトをセット

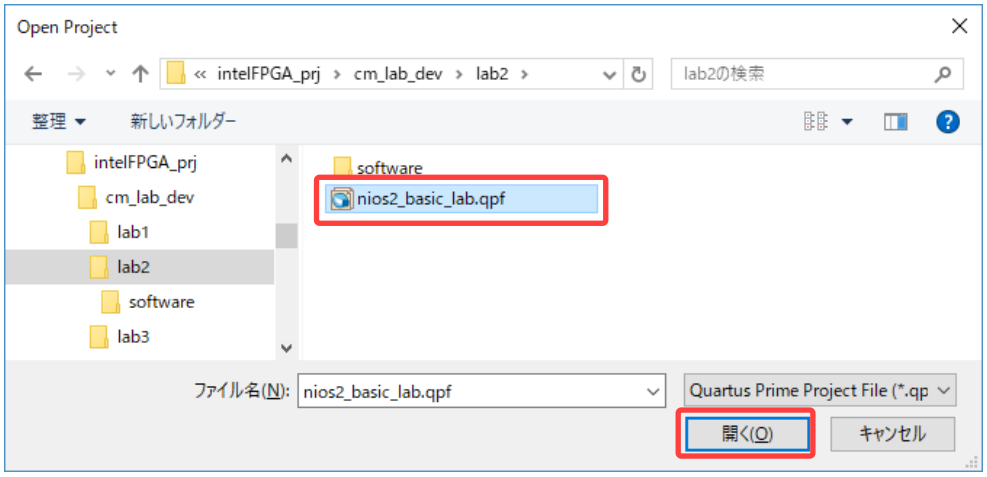
以下の手順に沿って Quartus Prime でシステム・モジュールを作成します。

この演習では、すでにベースとなる Quartus Prime プロジェクト (nios2_basic_lab) が作成されています。既存のプロジェクトにデザインを追加することで、ハードウェア用のプロジェクトを完成させます。Nios II プロセッサーの開発を行う際は、必ずハードウェア用のプロジェクトを作成する必要があります。

演習 1 終了後に Quartus Prime の GUI を閉じてしまった場合は、Quartus Prime を起動してください。

____ 1. Quartus Prime のメニューから ***File*** ➤ ***Open Project*** を選択し、以下の既存プロジェクト・ファイルを選択して ***[開く]*** ボタンをクリックします。

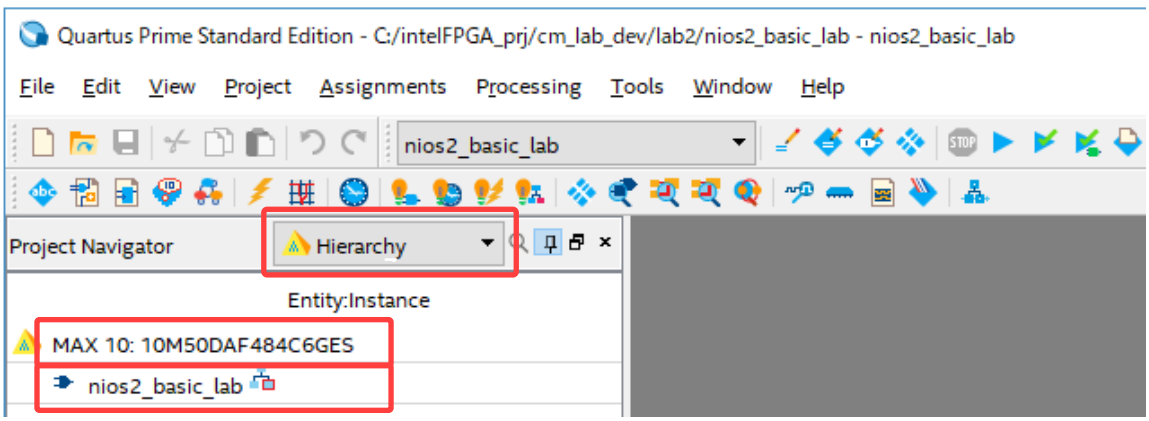
C:/intelFPGA_prj/cm_lab_dev/lab2/nios2_basic_lab.qpf



【図 2-1-1】 Open Project

____ 2. Quartus Prime の画面左上に位置する Project Navigator ウィンドウの ***Hierarchy ビュー*** において、以下 2 点を確認してください。

- **nios2_basic_lab** が登録されている
- ターゲットの FPGA 型番が **10M50DAF484C6GES** である



【図 2-1-2】 Hierarchy ビュー (Project Navigator ウィンドウ)

2. プラットフォーム・デザイナーでシステム・モジュールを作成

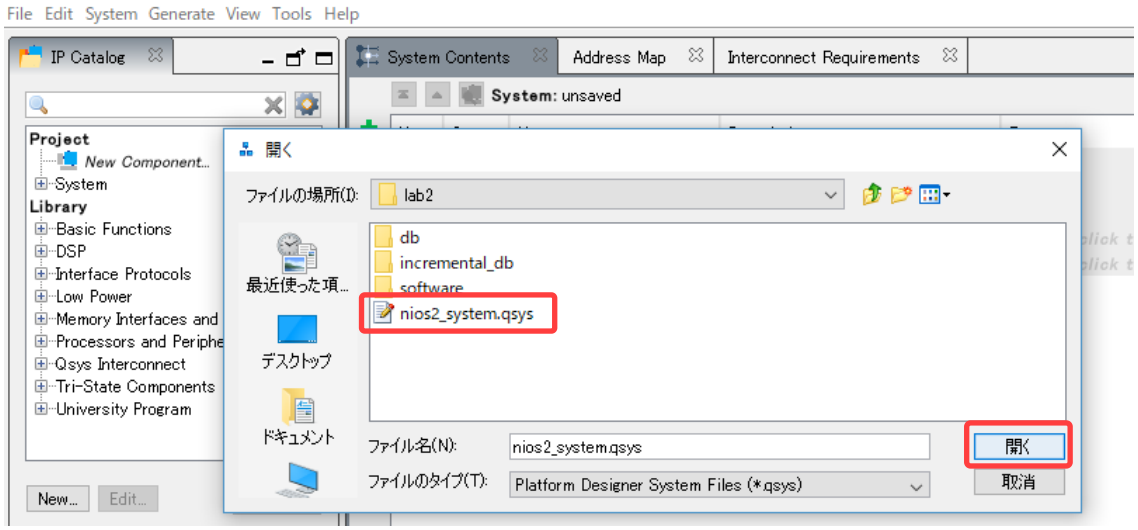
Quartus Prime に付属するシステム統合ツール プラットフォーム・デザイナー (Platform Designer) を使用して、Nios II プロセッサーを含むシステム・モジュールを作成します。

2-1. プラットフォーム・デザイナーを起動

____1. Quartus Prime のメニューから ***Tools*** ➤ ***Platform Designer*** を選択します。

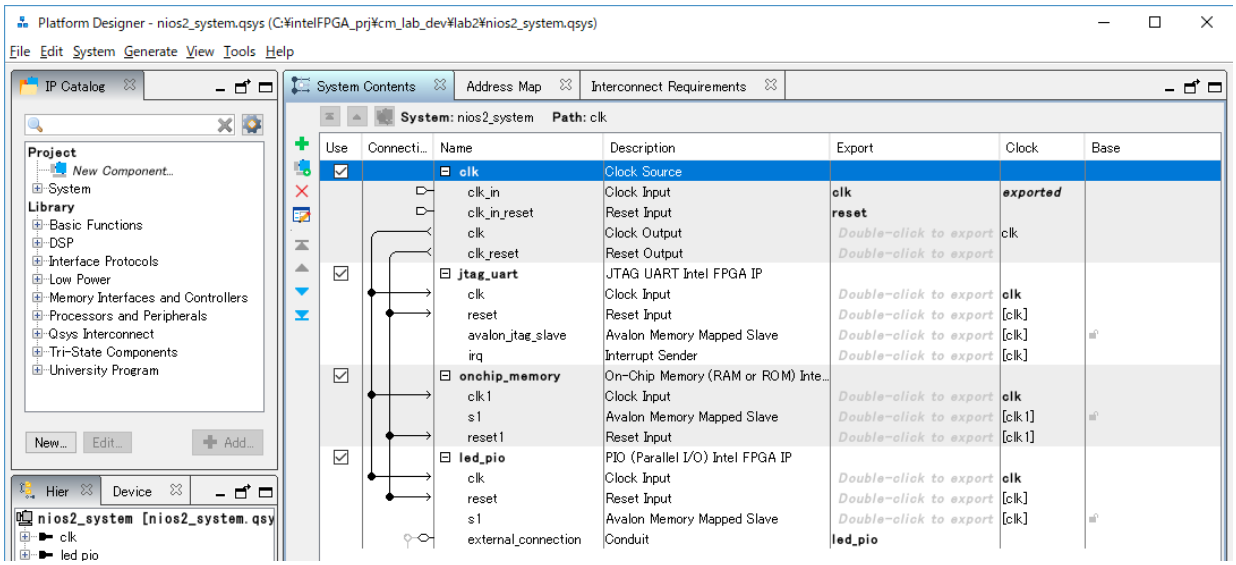
____2. プラットフォーム・デザイナーが起動し、ファイルの選択画面が表示されます。

この演習には、途中まで作成したシステム・モジュールがあらかじめ用意されています。作業フォルダーから ***nios2_system.qsys*** を選択し、***[開く]*** ボタンをクリックします。



【図 2-2-1】 Platform Designer で .qsys を選択

作成途中のシステム・モジュールが復元されます。



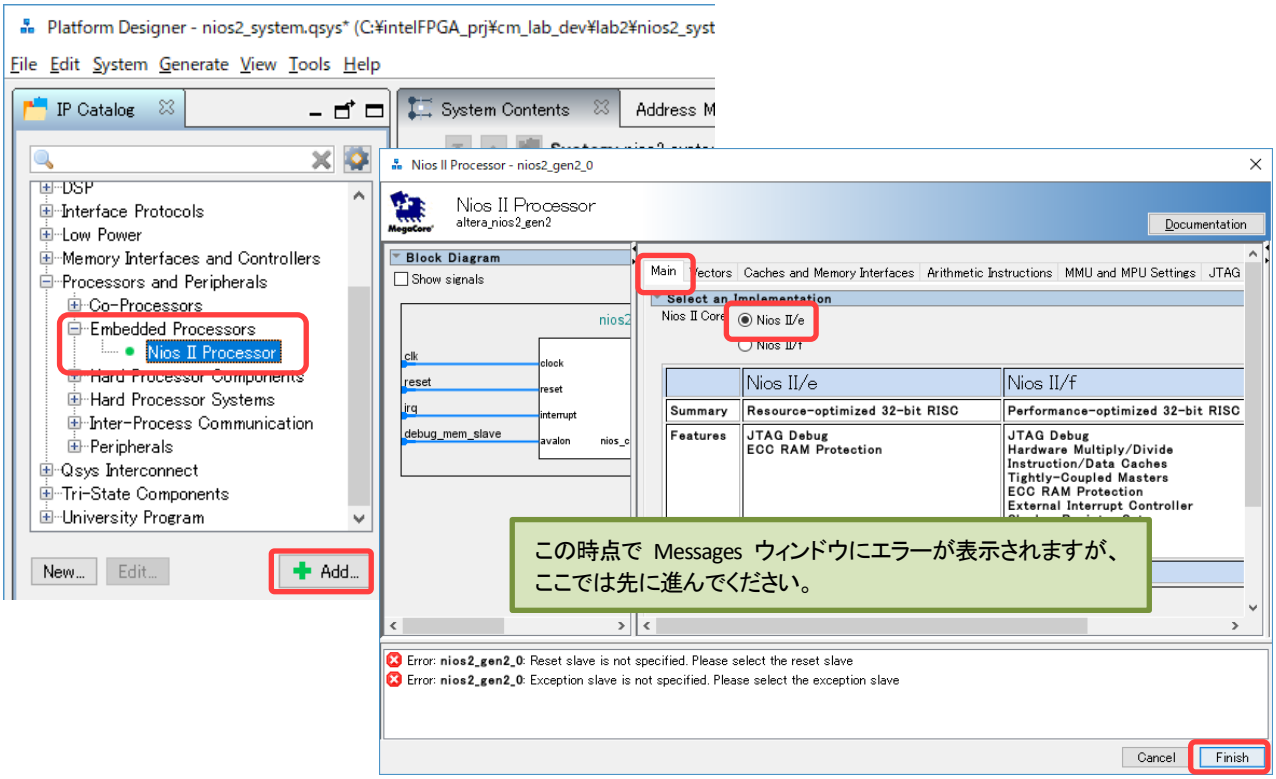
【図 2-2-2】 Platform Designer で .qsys を復元

2-2. Nios II プロセッサーを追加

システム・モジュールに Nios II プロセッサーを追加します。

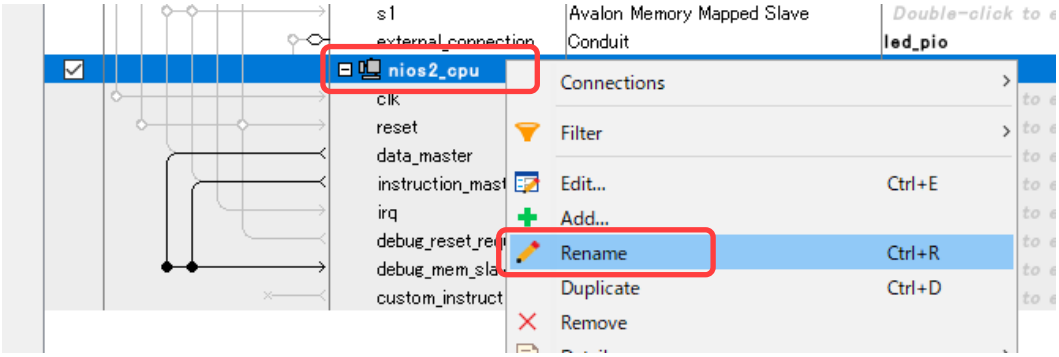
____ 1. プラットフォーム・デザイナーの IP Catalog ウィンドウ内 Library から、***Processors and Peripherals*** ➤ ***Embedded Processors*** ➤ ***Nios II Processor*** を選択し、***[Add]*** ボタンをクリックします。

Nios II プロセッサーのパラメーター設定画面が表示されます。***Main*** タブにおいて ***Nios II/e*** を選択し、***[Finish]*** ボタンをクリックします。



【図 2-2-3】 Nios II プロセッサーを追加

____ 2. プラットフォーム・デザイナーの ***System Contents*** タブにおいて、Nios II Processor の **nios2_gen2_0** (Name 項目) をマウスで右クリック ➤ ***Rename*** を選択し、***nios2_cpu*** に変更します。



【図 2-2-4】 nios2_cpu へリネーム

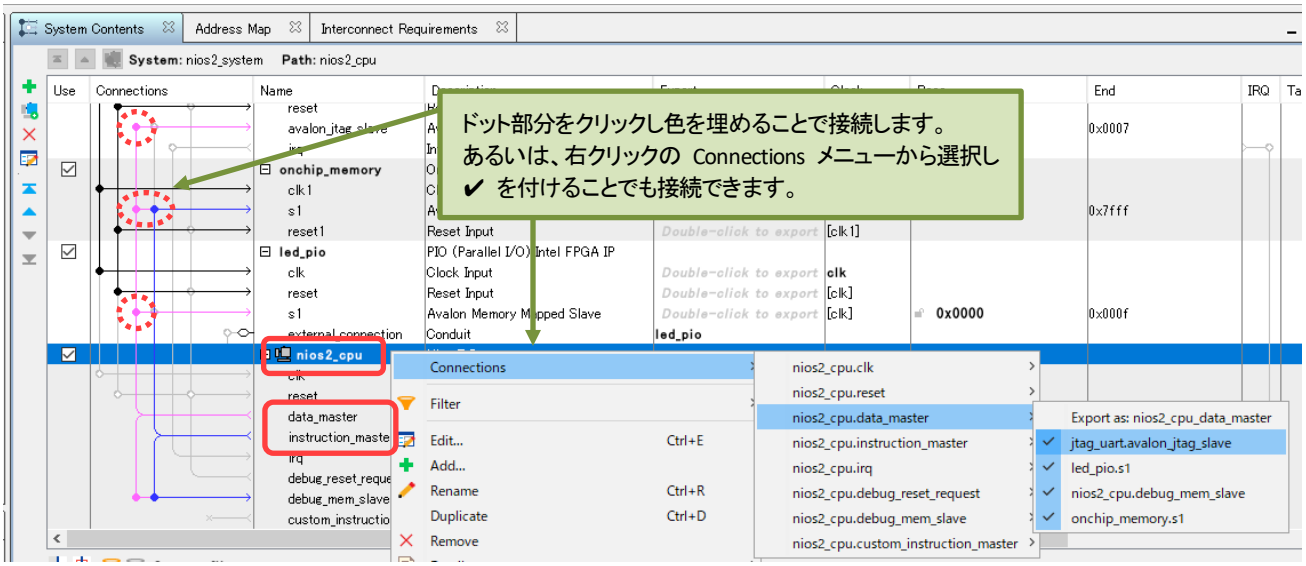
2-3. Nios II プロセッサーのリセットベクタと例外ベクタを設定

____ 1. Nios II プロセッサーのリセットベクタおよび例外ベクタに FPGA 内部のオンチップ・メモリを指定するため、Nios II プロセッサーとオンチップ・メモリを接続します。

プラットフォーム・デザイナーの ***System Contents*** タブにおいて、【表 2-2-1】および【図 2-2-5】を参考に、Nios II プロセッサーのデータ・マスターとインストラクション・マスターを各モジュールに接続します。

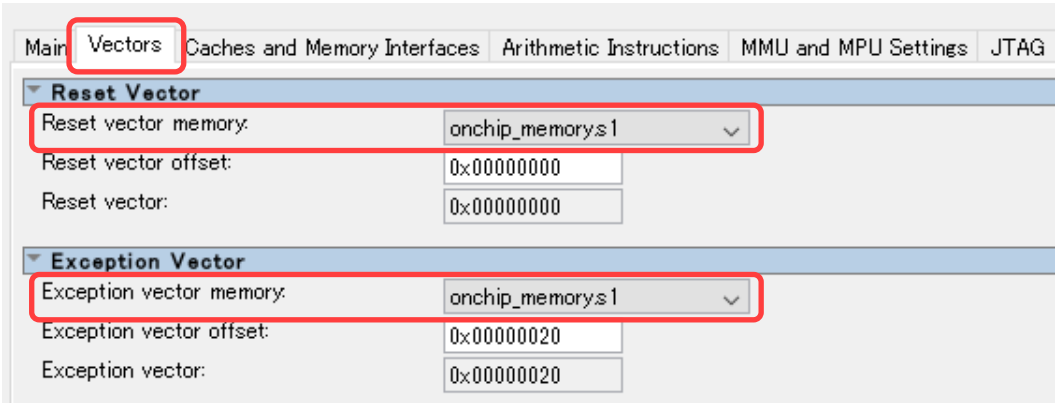
【表 2-2-1】 Nios II とオンチップ・メモリの接続

Name	ポート名	接続先
nios2_cpu	data_master	jtag_uart.avalon_jtag_slave
		onchip_memory.s1
		led_pio.s1
	instraction_master	onchip_memory.s1



【図 2-2-5】 nios2_cpu へリネーム

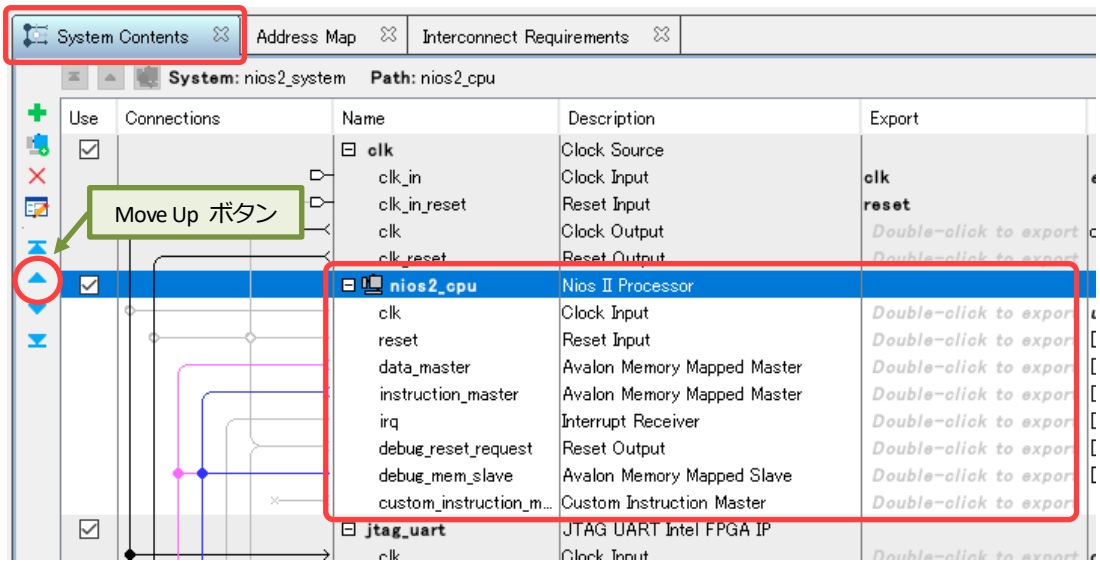
____ 2. nios2_cpu をマウスで右クリック ➤ ***Edit*** を選択し、再び Nios II プロセッサーの GUI 画面を起動します。***Vectors*** タブに切り替え、***Reset vector memory*** および ***Exception vector memory*** を下図のとおり変更し、***[Finish]*** ボタンをクリックします。



【図 2-2-6】 Nios II プロセッサーの Vectors タブ

2-4. クロックおよびリセットを接続

____ 1. プラットフォーム・デザイナーの System Contents タブにおいて nios2_cpu を選択し、見やすい位置に移動するため、ツールバーの [Move Up] ボタンを 3 回クリックして、clk モジュールの下へ配置します。



【図 2-2-7】 Nios II プロセッサーの配置変更

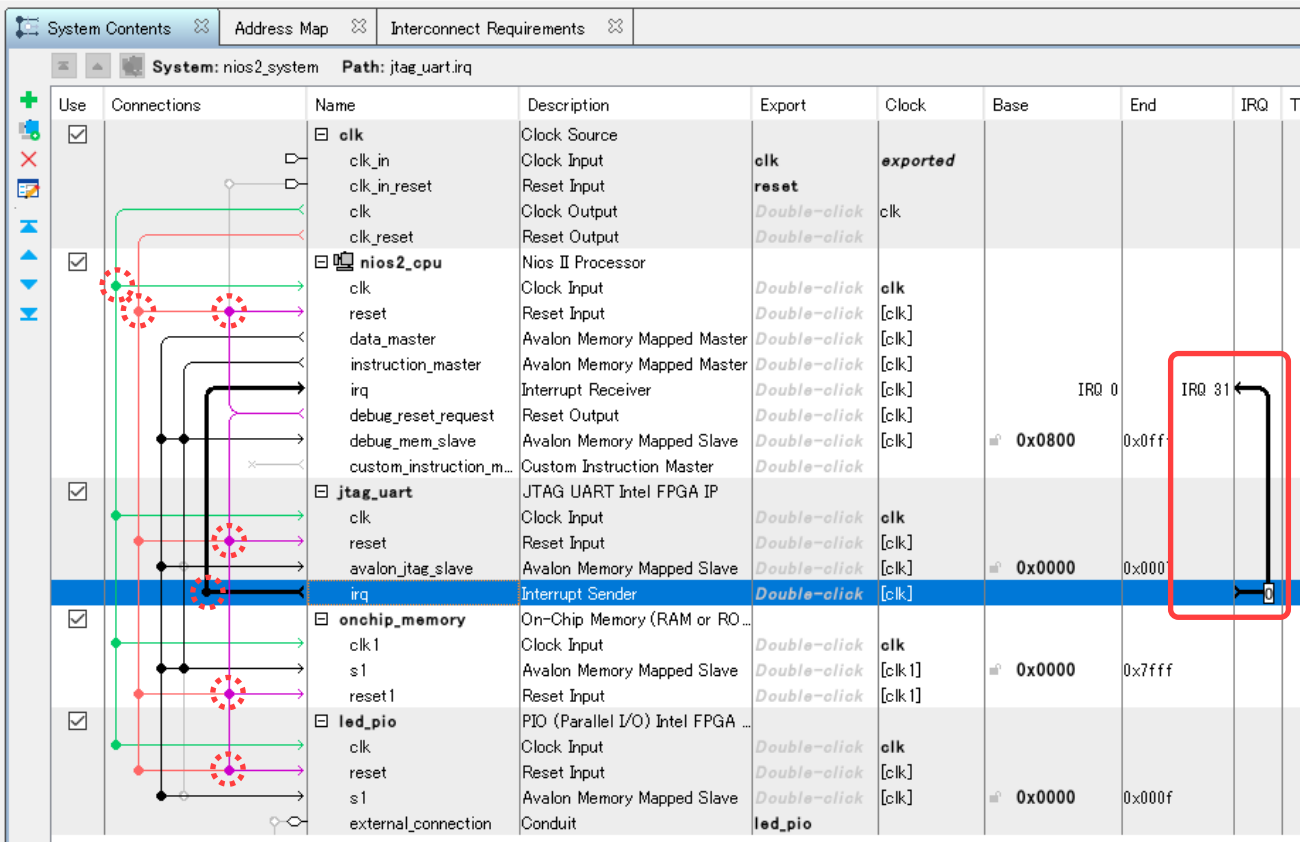
____ 2. Connections 項目において、【表 2-2-2】および【図 2-2-8】を参考に各モジュールのクロックとリセット接続します。

【表 2-2-1】 クロックとリセットの接続

Name	ポート名	接続先
clk	clk	nios2_cpu.clk
	clk_reset	nios2_cpu.reset
nios2_cpu	debug_reset_request	jtag_uart.reset
		led_pio.reset
		nios2_cpu.reset
		onchip_memory.reset1

____ 3. jtag_uart から Nios II プロセッサーへの割り込み接続を設定します。

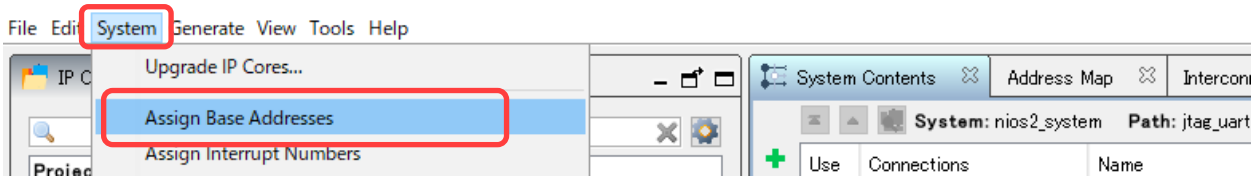
IRQ 列にある jtag_uart の白丸部分をクリックします。同時に Connections 項目の jtag_uart の irq が nios2_cpu の irq に接続されることを確認してください。(【図 2-2-8】参照)



【図 2-2-8】 クロックとリセットの接続 / 割り込みの接続

2-5. ベースアドレスを設定

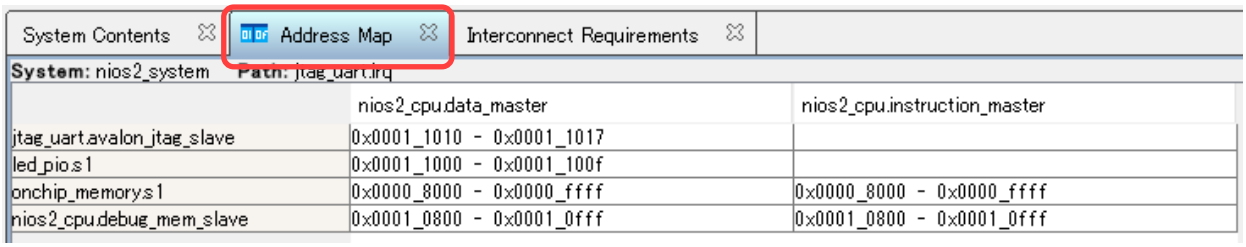
____1. プラットフォーム・デザイナーのメニューから ***System*** ➤ ***Assign Base Address*** を選択し、ベースアドレスを自動で適切な値に設定します。



【図 2-2-9】 Assign Base Address

____2. プラットフォーム・デザイナーの Address Map タブに切り替え、ベースアドレスの一覧を確認します。

表示される順序やアドレスが【図 2-2-10】と異なっていても、特に問題ありません。

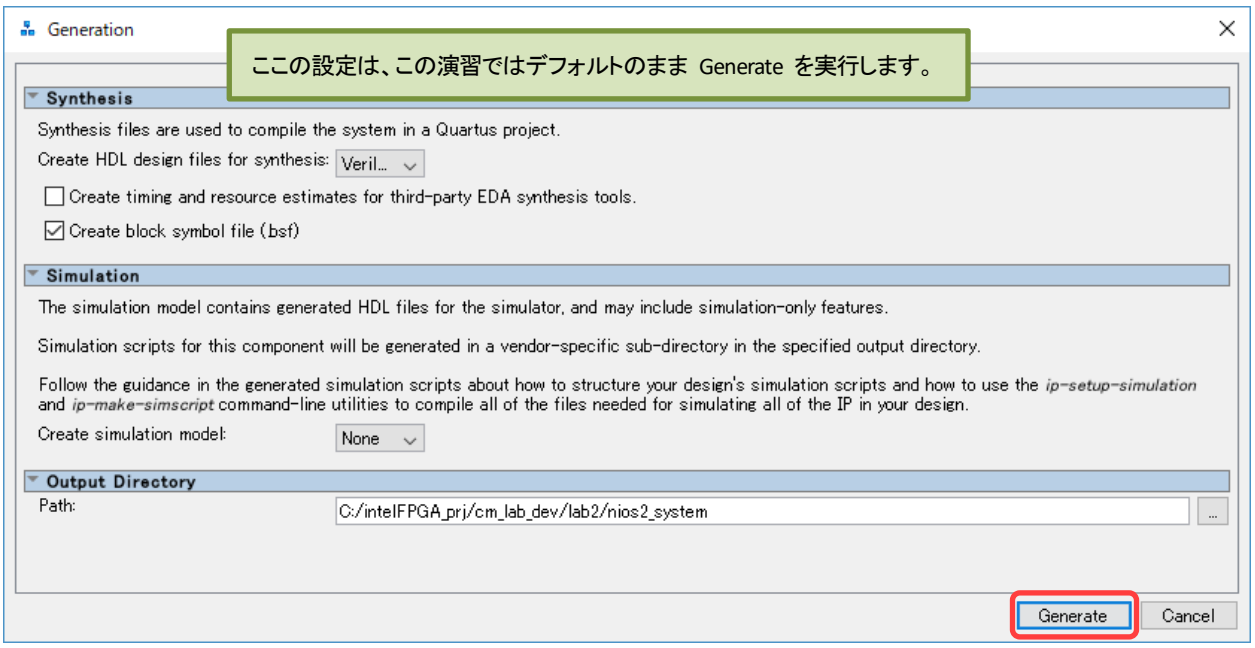


【図 2-2-10】 Address Map タブ

____3. プラットフォーム・デザイナーのメニューから ***File*** ➤ ***Save*** を選択し、nios2_system.qsys ファイルを保存します。保存が完了したら、Save System ウィンドウを ***[Close]*** ボタンで閉じます。

2-6. システム・モジュールを生成

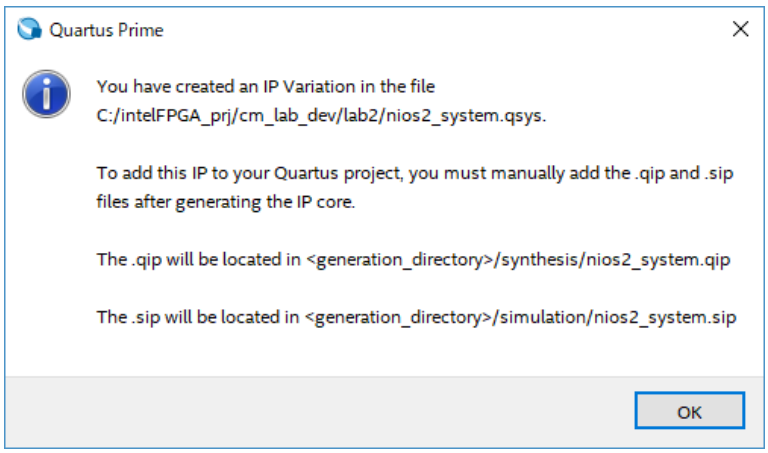
____ 1. プラットフォーム・デザイナーのメニューから ***Generate*** ➤ ***Generate HDL*** を選択すると、Generation ウィンドウが表示されます。右下にある ***[Generate]*** ボタンをクリックし、システムを生成します。



【図 2-2-11】 Generation ウィンドウ

____ 2. Generate ウィンドウのメッセージ・ボックスに “Generate: completed successfully.” と表示されれば、システム・モジュールが正常に生成されました。Generate ウィンドウを ***[Close]*** ボタンをクリックして閉じます。

____3. プラットフォーム・デザイナーのメニューから ***File*** ➤ ***Exit*** を選択し、プラットフォーム・デザイナーを閉じます。以下のウィンドウが表示されたら ***[OK]*** ボタンをクリックして閉じます。



【図 2-2-12】 Generation ウィンドウ

3. ハードウェア・デザインの作成

プラットフォーム・デザイナーで生成したシステム・モジュールを最上位階層に配置して、ハードウェア・デザインを作成します。

3-1. システム・モジュールをインスタンスする

____ 1. Quartus Prime のメニューから ***File*** ➤ ***Open*** を選択し、あらかじめ用意された最上位階層のデザインファイルを開きます。

フォーマットは、以下 2 種類の言語が用意されていますので、自分の設計手法である言語ファイルいずれかを選択し、開いてください。

【表 2-3-1】 クロックとリセットの接続

言語	選択するファイル名
VHDL	nios2_basic_lab.vhd
Verilog HDL	nios2_basic_lab.v

____ 2. nios2_basic_lab.vhd (.v) は、あらかじめ nios2_system モジュールをインスタンスする記述が途中まで行われています。ポートを接続する箇所が未完成のため、【表 2-3-2】従い 点線枠部分を編集してください。

【表 2-3-2】 編集箇所と編集内容

VHDL の場合 (28 行目)

編集前

```
u0 : nios2_system
PORT MAP ( clk_clk        => CONNECTED_TO_clk_clk,          -- clk.clk
           led_pio_export => CONNECTED_TO_led_pio_export,   -- led_pio.export
           reset_reset_n  => CONNECTED_TO_reset_reset_n     -- reset.reset_n
         );
```

編集後

```
u0 : nios2_system
PORT MAP ( clk_clk        => osc_clk,
           led_pio_export => tmp,
           reset_reset_n  => reset_n
         );
```

Verilog HDL の場合 (15 行目)

編集前

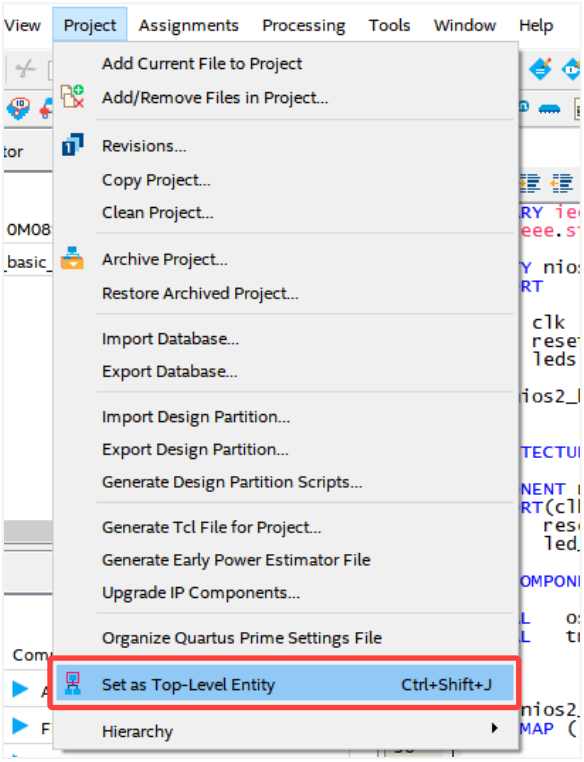
```
nios2_system u0 (
       .clk_clk        (<connected-to-clk_clk>),        // clk.clk
       .led_pio_export (<connected-to-led_pio_export>), // led_pio.export
       .reset_reset_n  (<connected-to-reset_reset_n>)   // reset.reset_n
       );
```

編集後

```
nios2_system u0 (
       .clk_clk        (osc_clk),
       .led_pio_export (tmp),
       .reset_reset_n  (reset_n)
       );
```

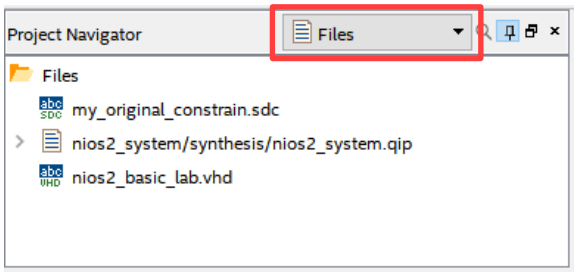
____3. 編集終了後、Quartus Prime のメニューから ***File*** ➤ ***Save*** を選択し、デザインファイルを保存します。

____4. nios2_basic_lab.vhd (.v) が開いている状態で、Quartus Prime のメニューから ***Project*** ➤ ***Set as Top-Level Entity*** を実行します。この操作で、nios2_basic_lab.vhd (.v) はこのプロジェクトの最上位デザインとして定義され、プロジェクトに登録されます。

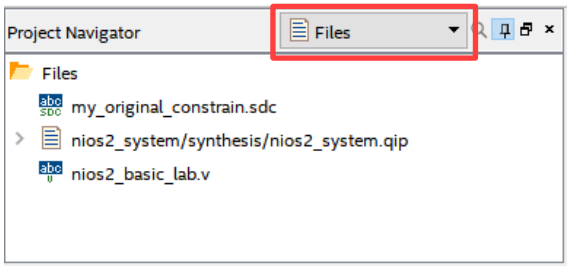


【図 2-3-1】 Set as Top-Level Entity

____5. ***Project Navigator ウィンドウ*** (Quartus Prime 画面左上) のプルダウンを ***Files ビュー*** に切り替え、下記ファイルが登録されていることを確認してください。



【図 2-3-2】 Project Navigator > Files (VHDL)



【図 2-3-3】 Project Navigator > Files (Verilog HDL)

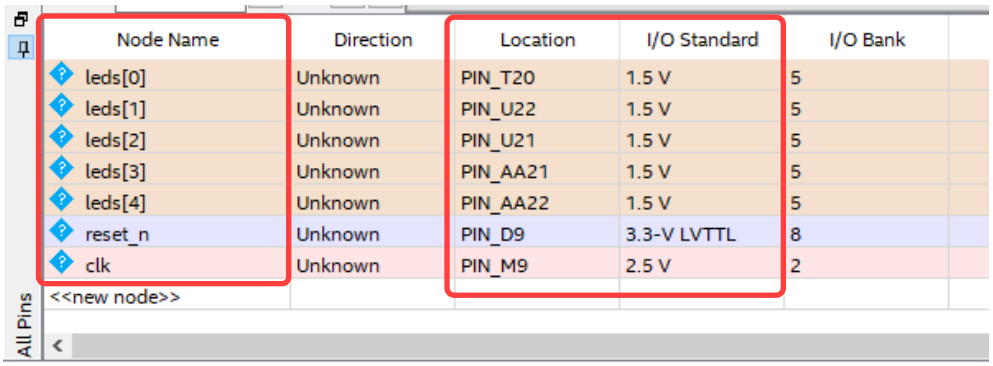
確認後は、Project Navigator ウィンドウのビューを Hierarchy に戻してください。

3-2. FPGA の外部端子をレイアウト（ピンアサイン）する

作成したデザイン上の信号を FPGA を外部端子へアサインするため、ピン番号を指定します。

なお、この演習ではすでにピンアサインが実施されています。どのように設定されているかを確認しましょう。

____ 1. Quartus Prime のメニューから ***Assignments*** ➤ ***Pin Planer*** を起動し、***All Pins リスト*** の枠内が【図 2-3-4】のとおり設定されていることを確認してください。



Node Name	Direction	Location	I/O Standard	I/O Bank
leds[0]	Unknown	PIN_T20	1.5 V	5
leds[1]	Unknown	PIN_U22	1.5 V	5
leds[2]	Unknown	PIN_U21	1.5 V	5
leds[3]	Unknown	PIN_AA21	1.5 V	5
leds[4]	Unknown	PIN_AA22	1.5 V	5
reset_n	Unknown	PIN_D9	3.3-V LVTTL	8
clk	Unknown	PIN_M9	2.5 V	2
<<new node>>				

All Pins

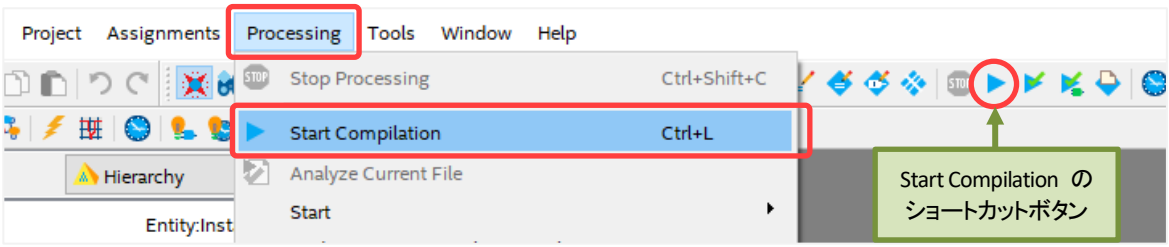
【図 2-3-4】 All Pins リスト（Pin Planner）

____ 2. 問題がなければ、Pin Planner のメニューから ***File*** ➤ ***Close*** により Pin Planner を閉じます。

3-3. ハードウェア・デザインをコンパイル

完成したハードウェア・デザインをコンパイルします。

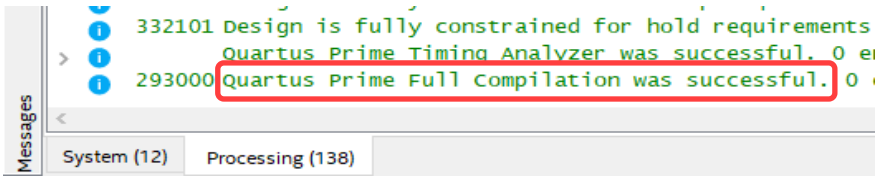
____ 1. Quartus Prime のメニューから ***Processing*** ➤ ***Start Compilation*** をクリックし、コンパイルを開始します。



【図 2-3-5】 コンパイル実行

コンパイルには多少の時間を要します。

____ 2. Messages ウィンドウに、Quartus Prime Full Compilation was successful. のインフォメーション・メッセージが表示されたら、エラーが発生することなくコンパイルが終了しています



【図 2-3-6】 Messages ウィンドウ

この演習はエラーが発生することなくコンパイルが完了するはずです。もしエラーが発生した場合は、解消しないと次のステップには進めません。回避するためにエラーの原因を追究し修正する必要があります。

また、エラー・メッセージの他、ワーニング・メッセージが発生する場合もあります。ワーニングは解消しなくても次のステップへ進めますが、必ず内容を確認し、その内容を回避すべきか無視できるのかをユーザーが判断してください。

各メッセージ内容の詳細を確認するには、ヘルプ機能を活用してください。 (14 ページ【図 1-4-3】参照)

4. ハードウェア・デザインを FPGA へダウンロード

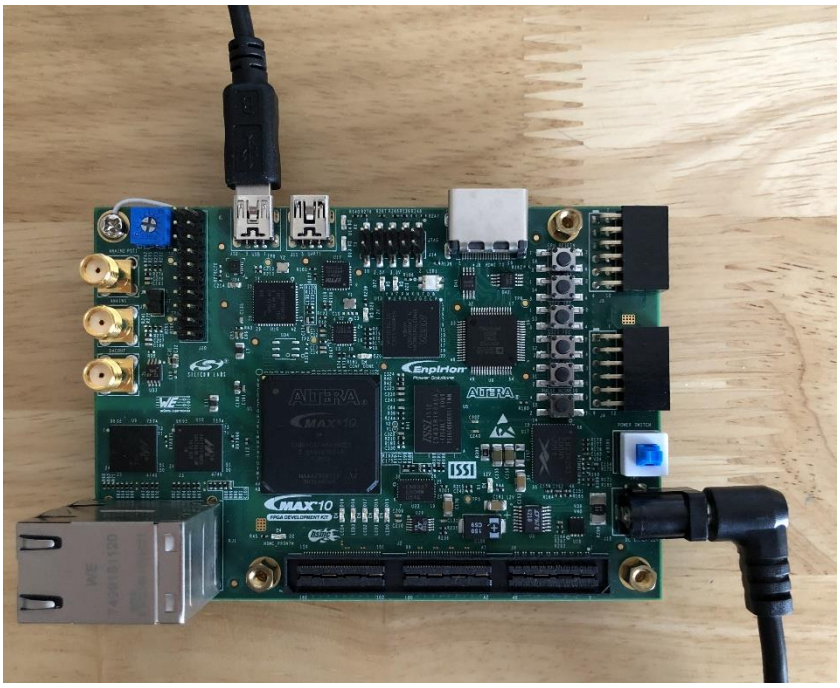
作成したハードウェア・デザインのデータを MAX 10 FPGA にダウンロードします。

> ⚠ **注記:**
> 実際の開発では、コンパイル後に Quartus Prime の Timing Analyzer によるタイミング検証を行い、期待どおりの動作が実現できるかを検証します。期待するタイミングを満足できることが確認できたら、ボード上のデバイスへデータを書き込みます。
> 今回の演習ではタイミング検証を省略していますが、自身の開発時は必ずタイミング検証を行った上でデバイスへの書き込みを行ってください。

> ⓘ Note:
> MAX 10 開発キットは、FPGA へデータをダウンロードする際 ボードに組み込まれたインテル FPGA ダウンロード・ケーブル II (旧称 USB Blaster™ II) 回路を使用します。パソコンとボードの接続は、キットに付属している mini-USB ケーブルで行います。パソコンとボードが通信するためには、インテル FPGA ダウンロード・ケーブル II 用のドライバーをインストールする必要があります。インストール方法は、【表 1-1】No. 5 をご覧ください。

____ 1. キット付属の電源コネクタを接続し、ボードの電源が OFF であることを確認してください。その後、付属の mini-USB ケーブルを用いてボードとパソコンを接続します。 (【図 2-4-1】参照)

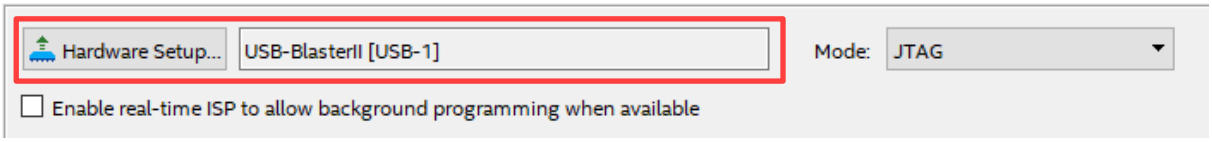
____ 2. 開発ボードへ電源を供給します。



【図 2-4-1】 パソコンととボードを接続

____3. Quartus Prime のメニューから ***Tools*** ➤ ***Programmer*** を選択し、Programmer を起動します。

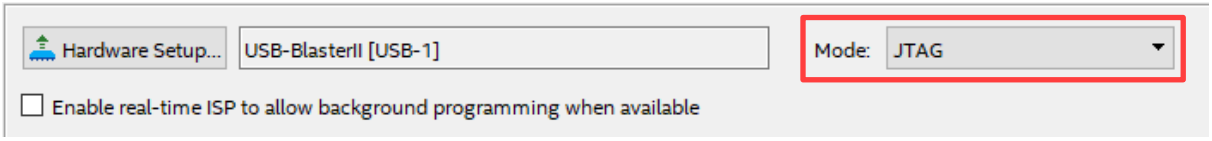
____4. Programmer の ***[Hardware Setup]*** ボタン 右横の欄が、***USB-Blaster II*** と表示されていることを確認します。



【図 2-4-2】 Hardware Setup

もし No Hardware 表示の場合には、***[Hardware Setup]*** ボタン をクリックし、***Hardware Settings タブ*** の ***Currently selected hardware*** 項のプルダウン・リストから、***USB-Blaster II*** を選択し、***[Close]*** ボタンをクリックしてください。

____5. Programmer の ***Mode*** プルダウン・リストから ***JTAG*** を選択します。



【図 2-4-3】 Mode

____ 6. Programmer の ***Files*** 欄に、ダウンロードするファイル ***nios2_basic_lab.sof*** が選択されていることを確認します。(【図 2-4-4】参照)

<none> となっている場合には、***[Add File]*** ボタン をクリックし、起動したファイルブラウザから nios2_basic_lab.sof を選択します。(作業フォルダー内 ***output_files フォルダー*** に生成されています。)

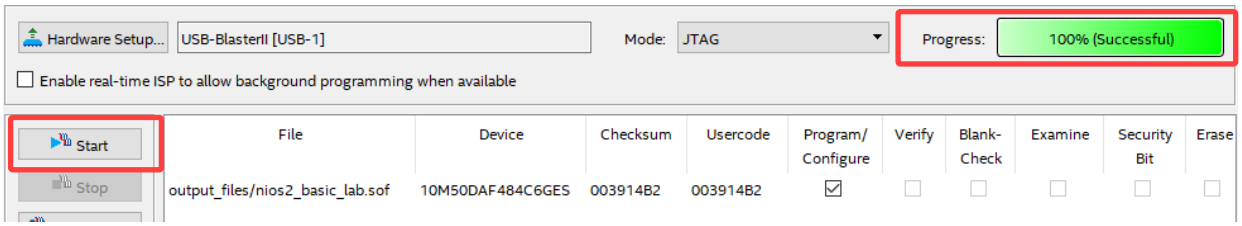
____7. プログラミング・オプションを選択します。

今回の演習では、FPGA の SRAM 領域にデータを転送するため、***Program/Configure*** にチェックを入れてください。

【図 2-4-4】ダウンロードする sof ファイルを指定

____8. Programmer の ***[Start]*** ボタン をクリックし、データの書き込みを開始します。

Progress バーが 100% になったら書き込み完了です。Messages ウィンドウには Successful のインフォメーションが表示されます



【図 2-4-5】書き込み完了

ハードウェア・デザインのデータは FPGA に転送されましたが、FPGA 内部の Nios II プロセッサーが実行するためのプログラムがまだありません。そのため、開発ボードの LED は消灯したままです。

続いて、Nios II プロセッサーが実行するソフトウェアの環境を構築し、実行しましょう。

5. ソフトウェアを実行

Nios II Software Build Tools (以下 Nios II SBT) は、Nios II プロセッサー向けの統合開発環境です。

今回の演習では、Nios II SBT を使用して、C ソースコードのビルド、およびターゲット・プログラムの開発ボードへのダウンロードまでを実施します。使用する C ソースコードは、Nios II プロセッサーから PIO を経由して、開発ボードの LED を点滅させる簡単なプログラムを用います。

5-1. Nios II SBT を起動

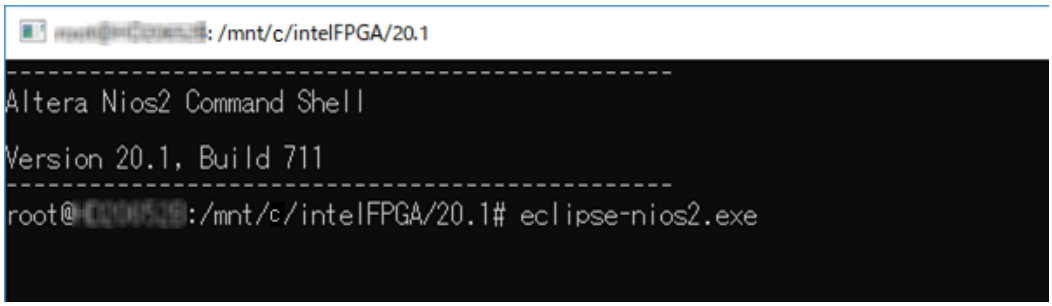
____1. Quartus Prime のメニューから ***Tools ➤ Nios II Software Build Tools for Eclipse*** をクリックして起動します。

ⓘ Note:

もし この起動方法で、GUI が表示されない、あるいは ソフトウェア・プロジェクト作成時にエラー (Failed to execute: wsl ./create-this-app –no-make) が発生した場合は、以下のフローにより Nios II SBT を起動してください。

<Windows 10 の場合>

1. Windows 10 の [スタート] ➤ ***Intel FPGA <version_build> Standard Edition ➤ Nios II Command Shell (Quartus Prime <version>)*** を右クリック選択し、ショートカット・アイコンのファイルの場所を開きます。
2. ***Nios II Command Shell (Quartus Prime <version>)*** を [管理者として実行] し、Nios II Command Shell を起動します。
3. `eclipse-nios2.exe` コマンドを実行して起動します。



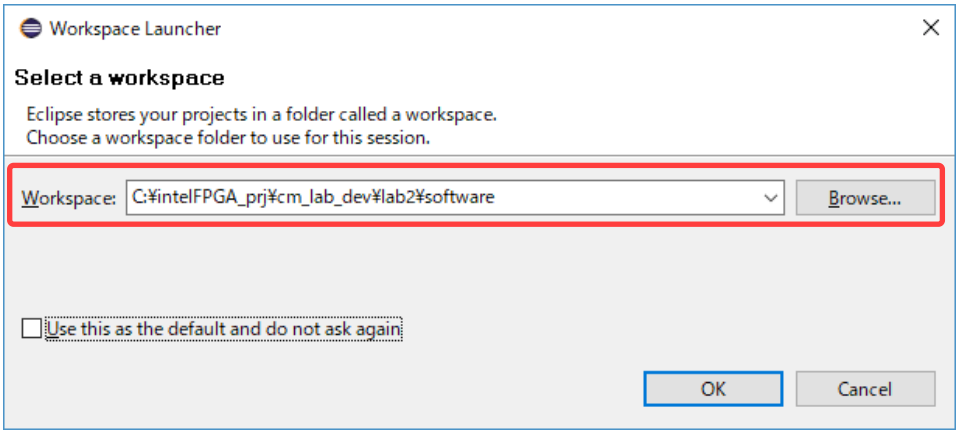
<Linux OS の場合>

1. ターミナル上で下記コマンドにより Nios II Command Shell を起動します。

   `<Nios II EDS install path>/nios2_command_shell.sh`

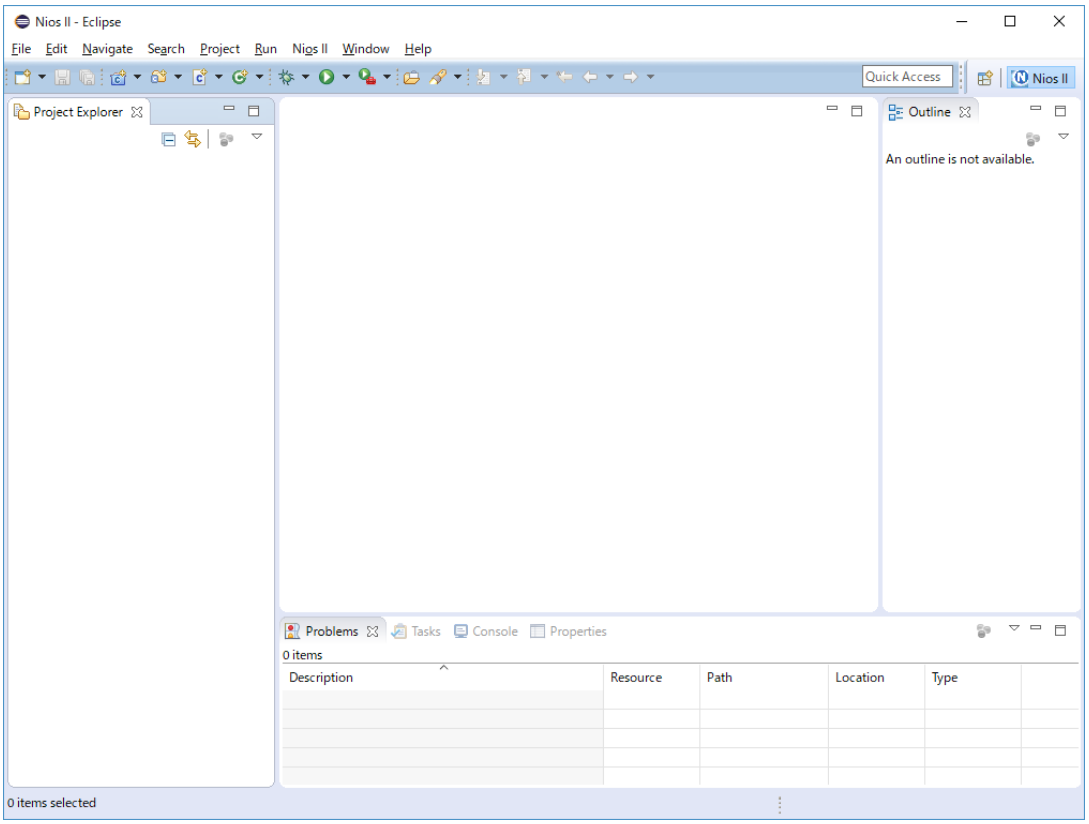
2. `eclipse-nios2` をタイプして起動します。

____ 2. Workspace Launcher が起動します。 ***[Browse] ボタン*** をクリックし Workspace に演習 2 用の Quartus Prime プロジェクト・フォルダー下の ***software*** を指定して ***[OK] ボタン*** をクリックします。



【図 2-5-1】 Workspace Launcher

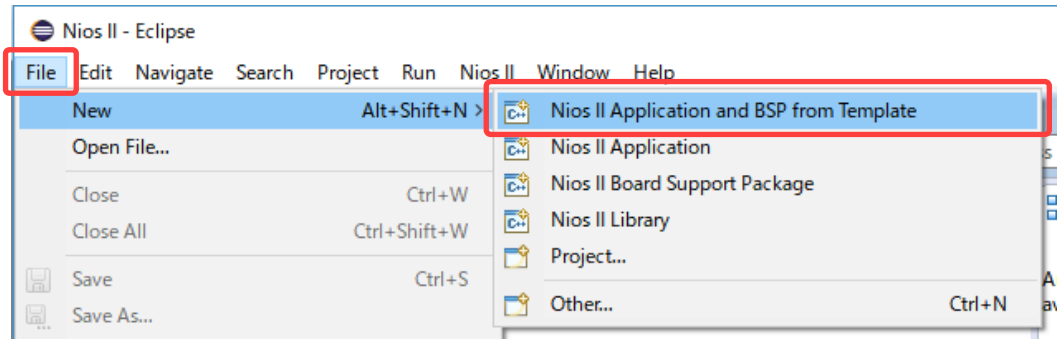
Nios II SBT が起動します。



【図 2-5-2】 Nios II SBT

5-2. ソフトウェア・プロジェクトを作成

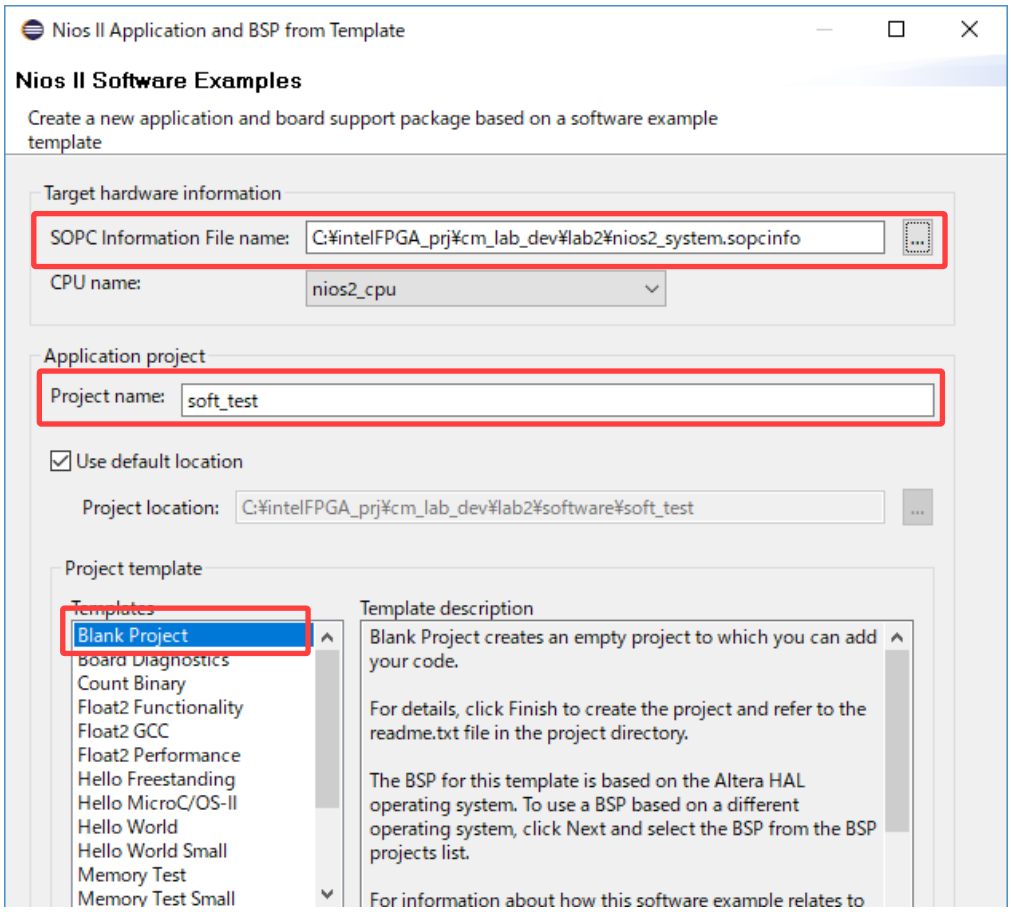
____1. Nios SBT 上のメニューから ***File*** ➤ ***New*** ➤ ***Nios II Application and BSP from Template*** をクリックし起動します。



【図 2-5-3】 Nios II Application and BSP from Template を起動

____2. プラットフォーム・デザイナーで作成したシステムのハードウェア情報が記述されたシステム定義ファイルを指定します。

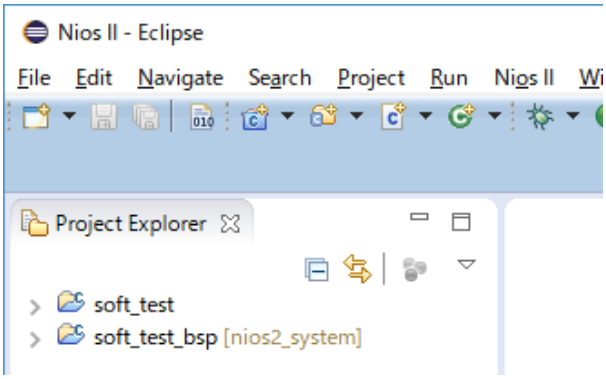
- Target hardware information エリアの ***SOPC Information File name*** 欄に、ハードウェア・デザインを作成した lab2 フォルダー内の ***nios2_system.sopcinfo*** を選択します。
- Application project エリアの ***Project name*** 欄に、***soft_test*** と入力します。
- Templates 枠内 から ***Blank Project*** を選択します。



【図 2-5-4】 Nios II Application and BSP from Template

設定後、***[Finish] ボタン*** をクリックします。

____ 3. 新しいソフトウェア・プロジェクトが作成され、Nios II SBT の Project Explorer タブ内に ***soft_test*** と ***soft_test_bsp*** が追加されます。

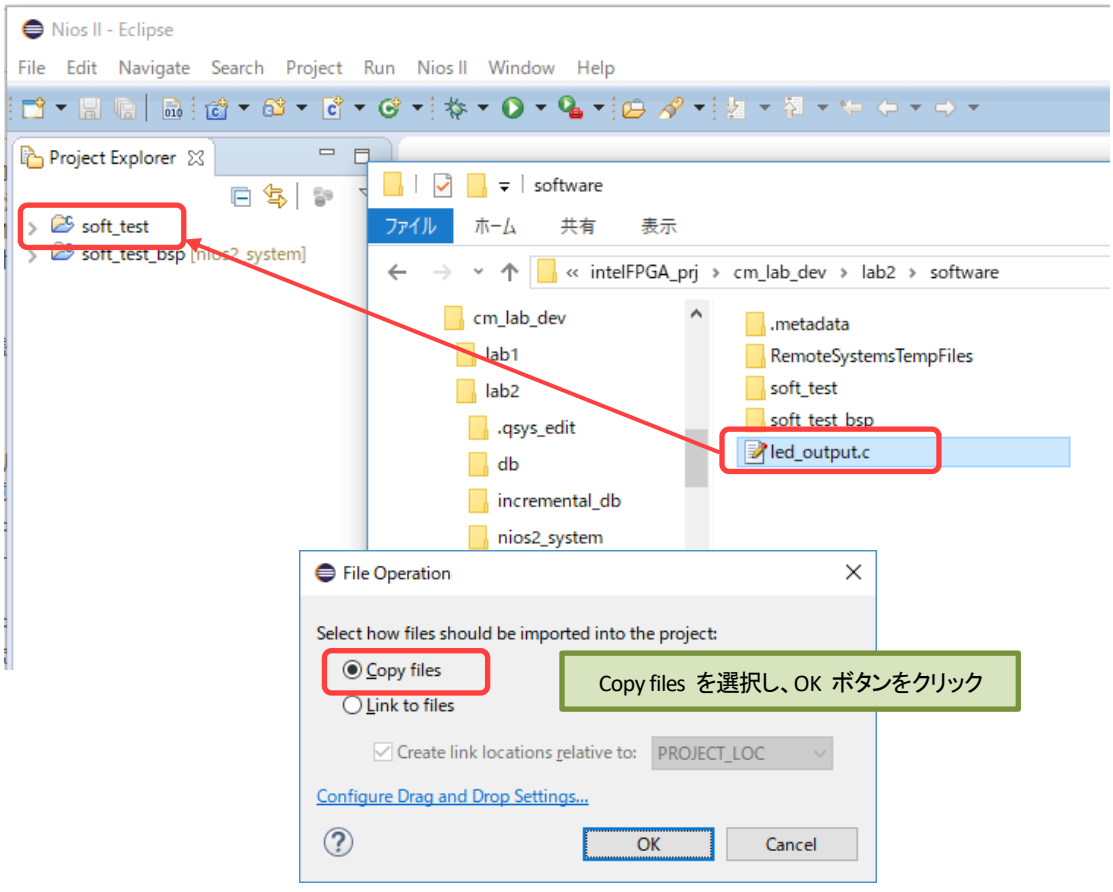


【図 2-5-5】ソフトウェア・プロジェクト作成

5-3. C ソースコードのインポート

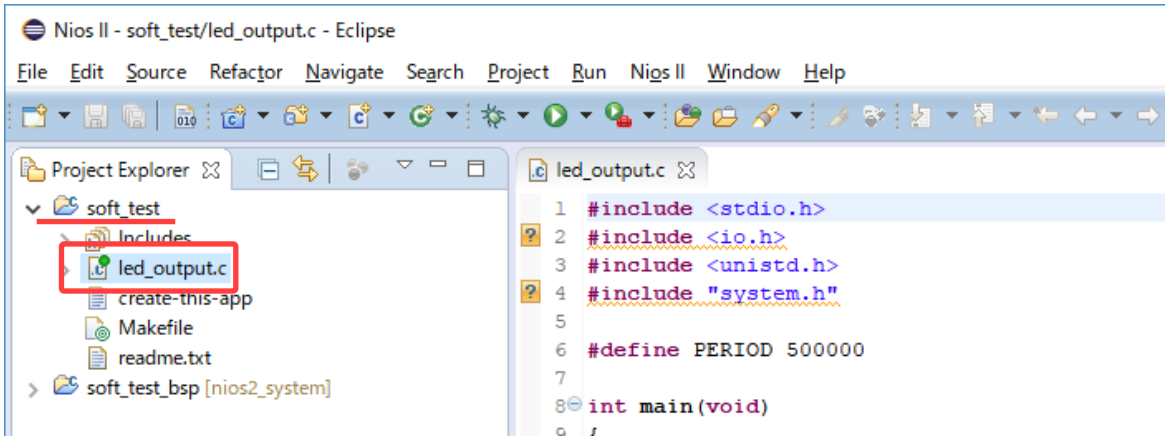
あらかじめ用意された C 言語のソースファイルをソフトウェア・プロジェクトにインポートします。

____1. Windows エクスプローラを開き、この演習 2 フォルダー下の ***software*** フォルダーにある ***led_output.c*** ファイルを、Nios II SBT の Project Explorer タブ内にある ***soft_test*** フォルダーにドラッグ&ドロップします。



【図 2-5-6】C ソースコードをプロジェクトへコピー

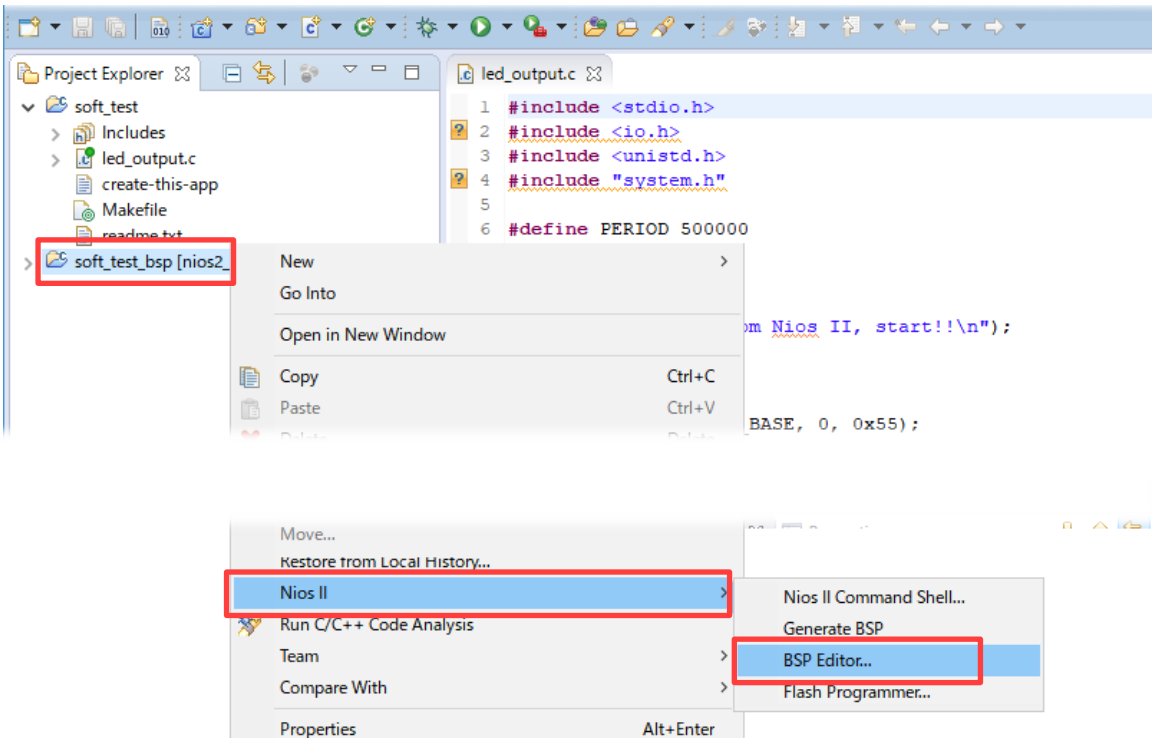
____ 2. Nios II SBT の Project Explorer タブ内にある soft_test フォルダーを展開し、led_output.c が soft_test フォルダーにインポートされていることを確認してください。led_output.c をダブルクリックすると、C ソースコードが表示されます。



【図 2-5-7】 led_output.c

5-4. ソフトウェア・プロジェクトのビルド

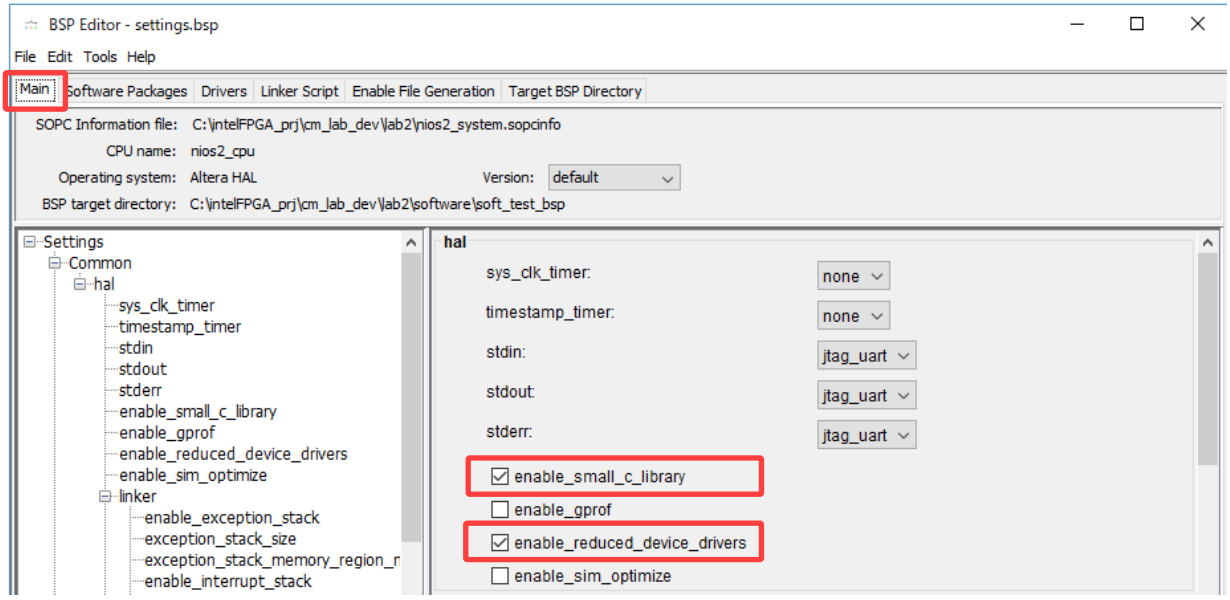
____ 1. Nios II SBT の Project Explorer タブ内にある ***soft_test_bsp*** フォルダーを右クリックで選択 ➤ ***Nios II*** ➤ ***BSP Editor*** を選択し、システムの設定を確認します。



【図 2-5-8】 BSP Editor を起動

____ 2. BSP Editor ウィンドウの ***Main タブ*** において、下記オプションを有効 (✓) にします。その後、BSP Editor ウィンドウ右下に位置する ***[Generate] ボタン*** をクリックし、***[Exit] ボタン*** で BSP Editor ウィンドウを閉じます。

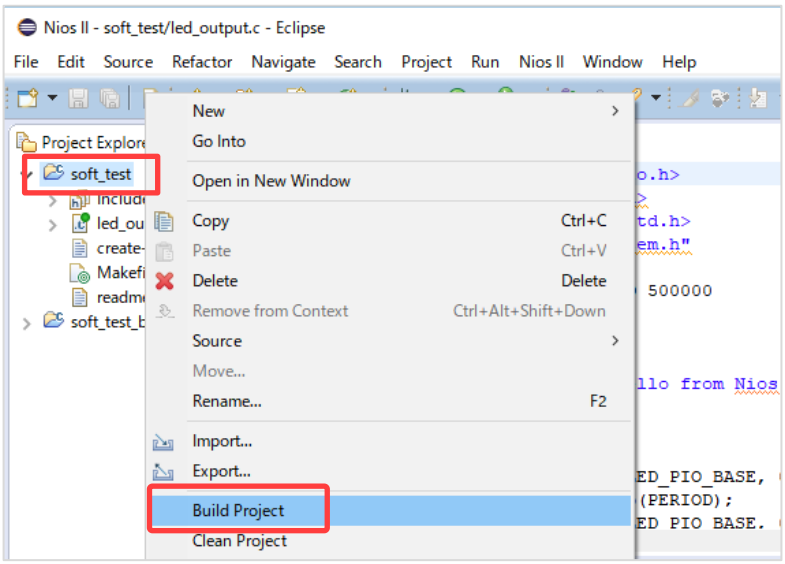
- enable_small_c_library = On
- enable_reduced_device_drivers = On



【図 2-5-9】 BSP Editor

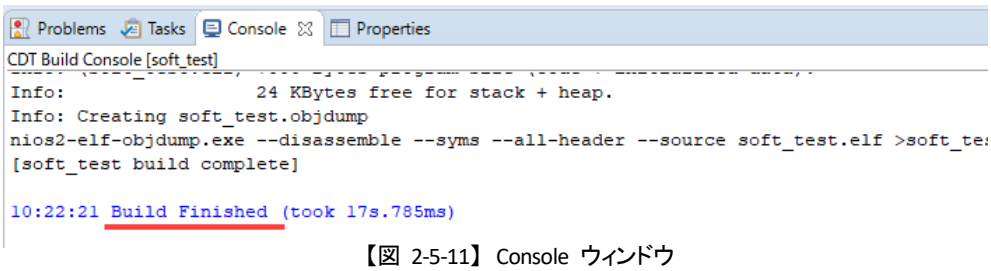
____3. ソフトウェアをビルド (Build) します。

Nios II SBT の Project Explorer タブ内にあるアプリケーション･プロジェクトのフォルダー (soft_test) を選択し、右クリック ➤ ***Build Project*** をクリックします。



【図 2-5-10】 プロジェクトをビルド

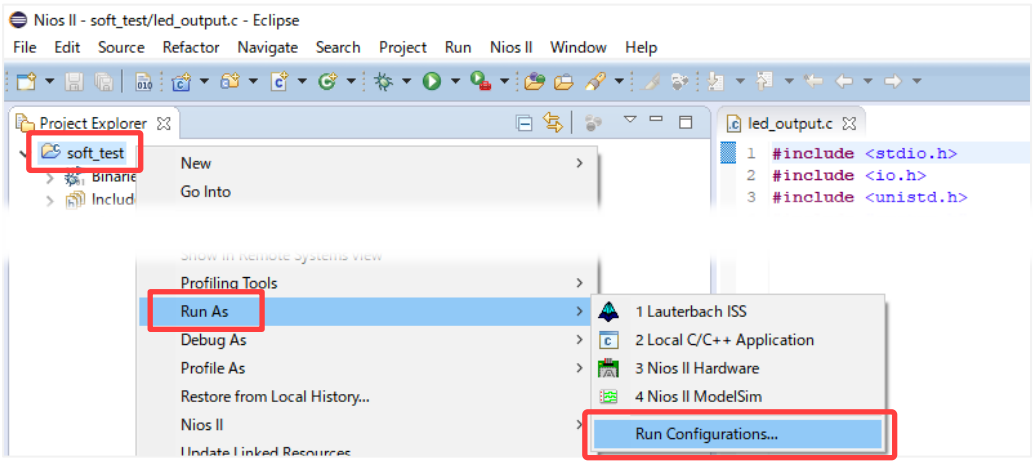
____4. エラーなくビルドが完了することを確認します。



【図 2-5-11】 Console ウィンドウ

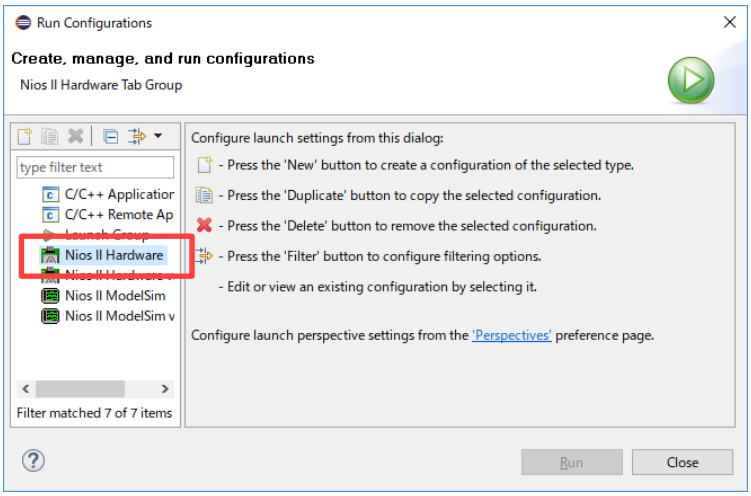
5-5. ソフトウェア・プログラムを実行

____1. Nios II SBT の Project Explorer タブ内の ***soft_test フォルダー*** を右クリック ➤ ***Run As*** ➤ ***Run Configurations*** をクリックします。



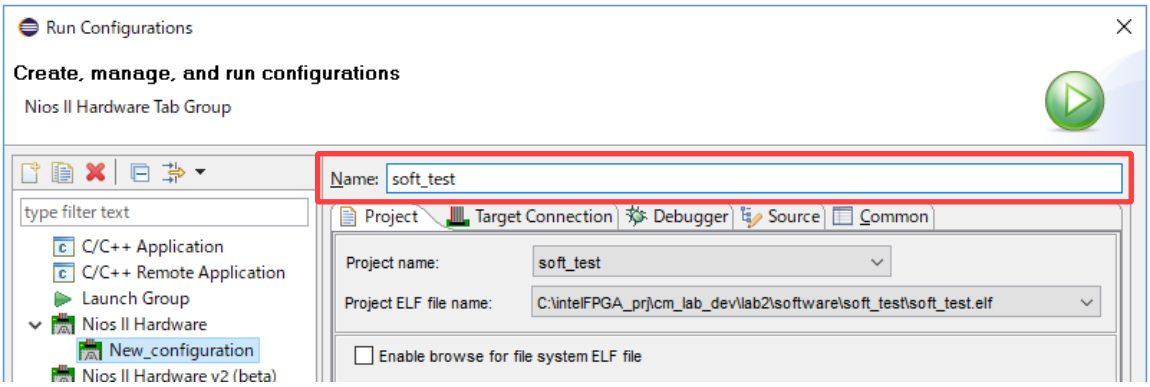
【図 2-5-12】Run Configurations

____2. Run Configurations ウィンドウのアイテムから ***Nios II Hardware*** を選択してダブルクリックします。



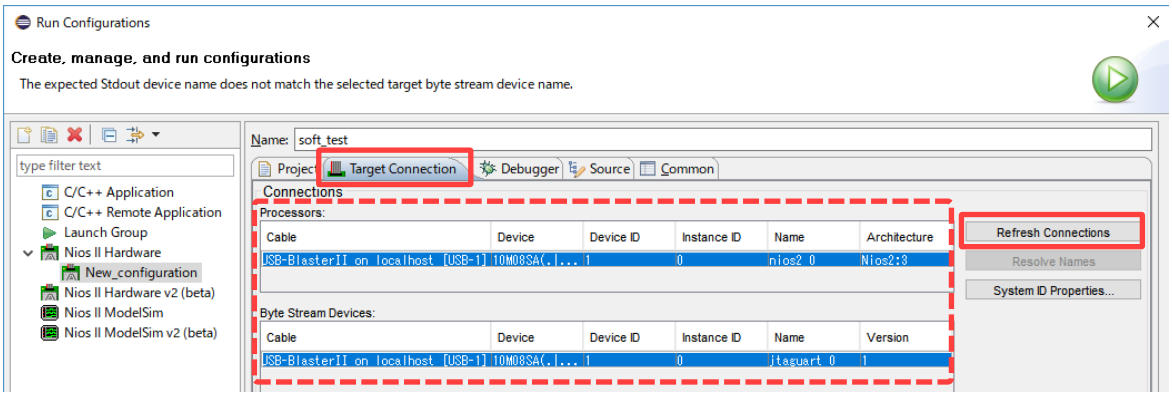
【図 2-5-13】 Run Configurations

____3. Run Configurations ウィンドウの ***Name 欄*** において、デフォルトの New_configuration から任意の名称に変更します。この演習では ***soft_test*** と入力してください。



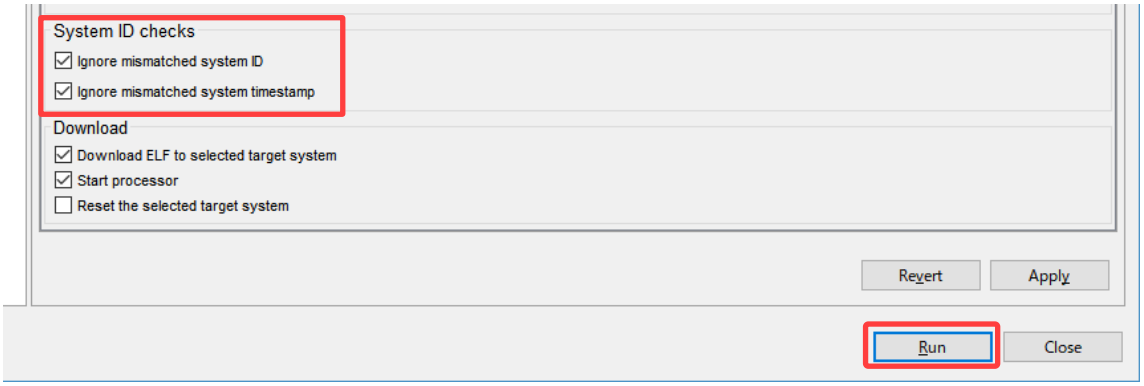
【図 2-5-14】 Name 欄を編集

____4. Run Configurations ウィンドウの ***Target Connection タブ*** を選択し、右上の ***[Refresh Connections] ボタン*** をクリックします。【図 2-5-15】のように USB-Blaster II が検出されるのを確認します。



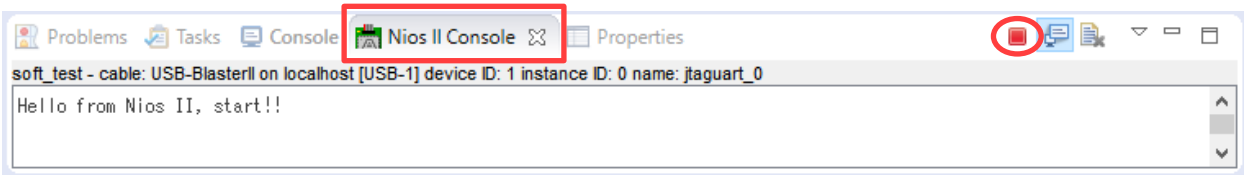
【図 2-5-15】 接続を確認

System ID checks 欄 の 2 つのオプションを有効にし、***[Run] ボタン*** をクリックします。



【図 2-5-16】 System ID checks のオプション

____5. Nios II SBT の Nios II Console ウィンドウに、C 言語ソースコード上に記載した printf 関数の出力キャラクタ “Hello from Nios II, start!!” が確認できます。



【図 2-5-17】 Nios II Console ウィンドウ

同時に、開発ボード上の LED の点滅を確認してください。これにより、Nios II が正しくソフトウェア・プログラムを実行したことがわかります。

LED の点滅を確認したら、Nios II Console ウィンドウ右上の赤いボタンをクリックしてターミナルを終了します。

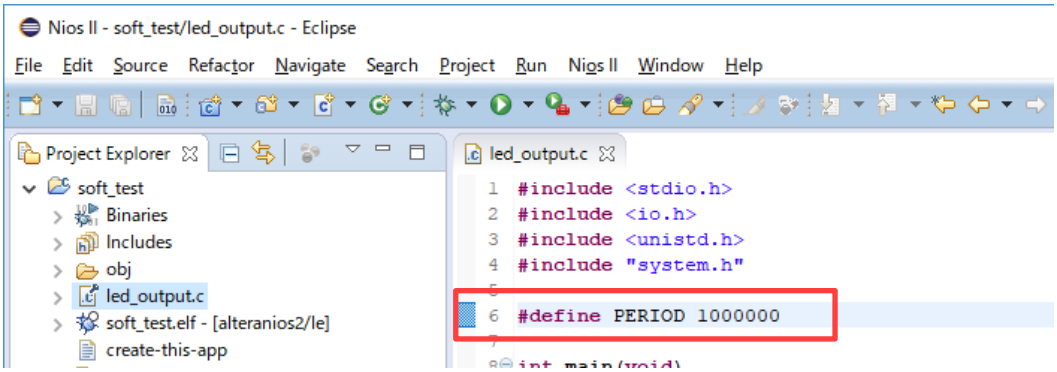
5-6. ソフトウェア・プログラムを変更して実行

C 言語のコードを編集して、プログラムを変更してみましょう。

____1. soft_test フォルダー下の led_output.c をダブルクリックし、C 言語ソースコードを開きます。

____2. led_output.c の 6 行目に ***PERIOD*** として定義されている値を ***500000 から 1000000 に変更*** します。

Nios II SBT のメニューから ***File ➤ Save*** を選択し、C 言語ソースコードを保存します。



【図 2-5-18】 led_output.c を編集

____3. アプリケーション･プロジェクトのフォルダー (soft_test) を右クリックで選択 ➤ ***Build Project*** をクリックし、プロジェクトをビルドします。

____4. ビルドが完了したら、soft_test プロジェクトのフォルダーを右クリック ➤ ***Run As*** ➤ ***Nios II Hardware*** をクリックし、プログラムを実行します。

____5. Nios II SBT の Nios II Console ウィンドウに、“Hello from Nios II, start!!” の文字が確認でき、同時に開発ボード上の LED の点滅する速度が変化していることを確認してください。

____6. LED の点滅を確認したら、Nios II Console ウィンドウ右上の赤いボタンをクリックしてターミナルを終了します。

____7. Nios II SBT のメニューから ***File ➤ Exit*** で、Nios II SBT を閉じます。

すべての作業が終了したら、Quartus Prime のメニューから ***File ➤ Close Project*** により、演習 2 のプロジェクトを終了してください。

また、開発ボードの電源を OFF にしてください。

ⓘ Note:

この演習では、ハードウェア･イメージおよび Nios II のブート･イメージを 共に MAX 10 FPGA の SRAM 領域にダウンロードしているので、開発ボードの電源を切ると FPGA 内のデータは消去されます。そのため製品として成立させるためには、ハードウェア･イメージは MAX 10 の Flash メモリ領域 (CFM) にプログラムし、Nios II のブート･イメージは MAX 10 の内部あるいは外部に用意した不揮発性メモリに格納する必要があります。

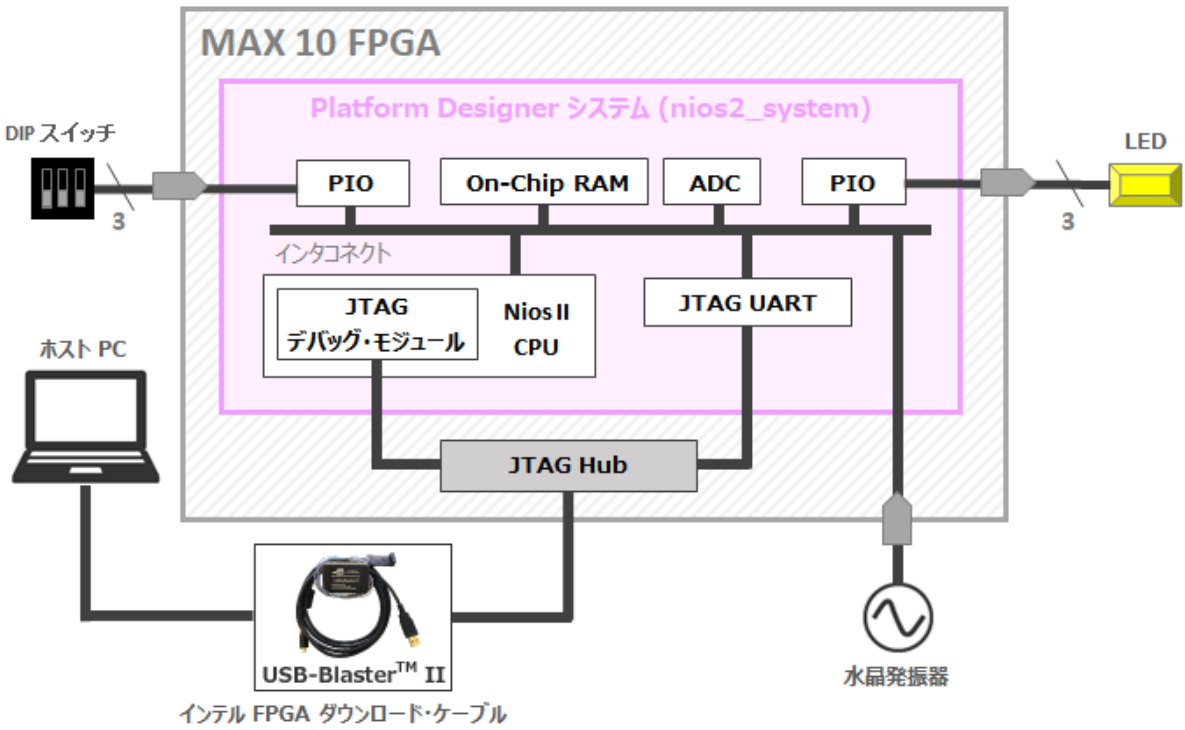
※ これらの作業は今回の演習では行いません。別コースのセミナを受講されるか、メーカーのドキュメントや弊社の Web コンテンツをご覧ください。

以上で、演習 2 の作業はすべて終了です。

演習 3.　内蔵 ADC を使用して温度測定

この演習では、MAX 10 FPGA に Nios II プロセッサー、JTAG UART、PIO、On-Chip メモリ、ADC を実装し、サンプルの C 言語ソースコードの動作にしたがって MAX 10 FPGA 内蔵の温度センサからアナログ信号を ADC でデジタル信号に変換してプロセッサー内で演算処理し、結果を表示するデザインを作成します。

- **演習で構築するシステムの概要**



これらの周辺ペリフェラルは Nios II プロセッサー用内部バスである プラットフォーム・デザイナーのインタコネクトで接続されます。

上図の組み込みシステムは、システム名 nios2_system というブロックとして FPGA 内に実装されます。

- 主な作業内容
  - プラットフォーム・デザイナーでシステム・モジュールを作成
  - ハードウェア・デザイン作成
  - ハードウェア・デザインを FPGA へダウンロード
  - ソフトウェアを実行

1. プロジェクトをセット

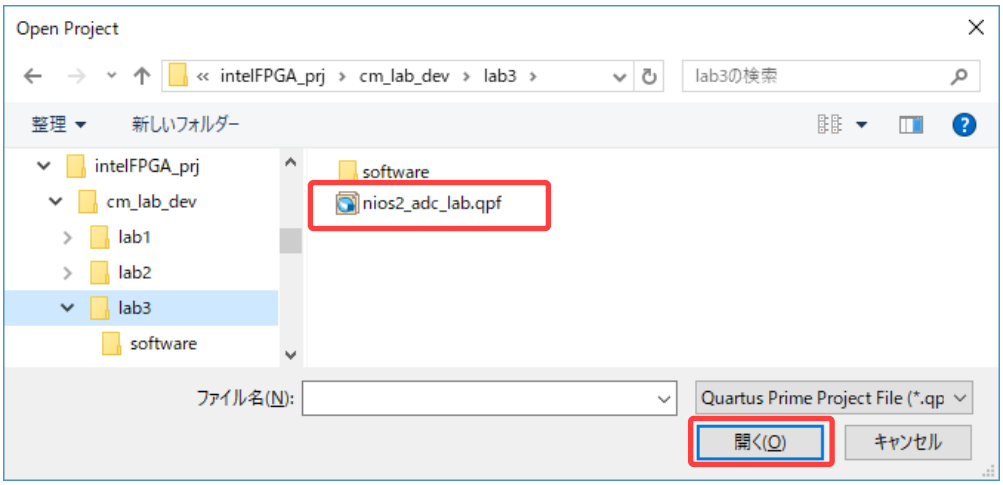
以下の手順に沿って Quartus Prime でシステム・モジュールを作成します。

この演習では、すでにベースとなる Quartus Prime プロジェクト (nios2_adc_lab) が作成されています。既存のプロジェクトにデザインを追加することで、ハードウェア用のプロジェクトを完成させます。Nios II プロセッサーの開発を行う際は、必ずハードウェア用のプロジェクトを作成する必要があります。

演習 2 終了後に Quartus Prime の GUI を閉じてしまった場合は、Quartus Prime を起動してください。

____ 1. Quartus Prime のメニューから ***File*** ➤ ***Open Project*** を選択し、以下の既存プロジェクト・ファイルを選択して ***[開く]*** ボタンをクリックします。

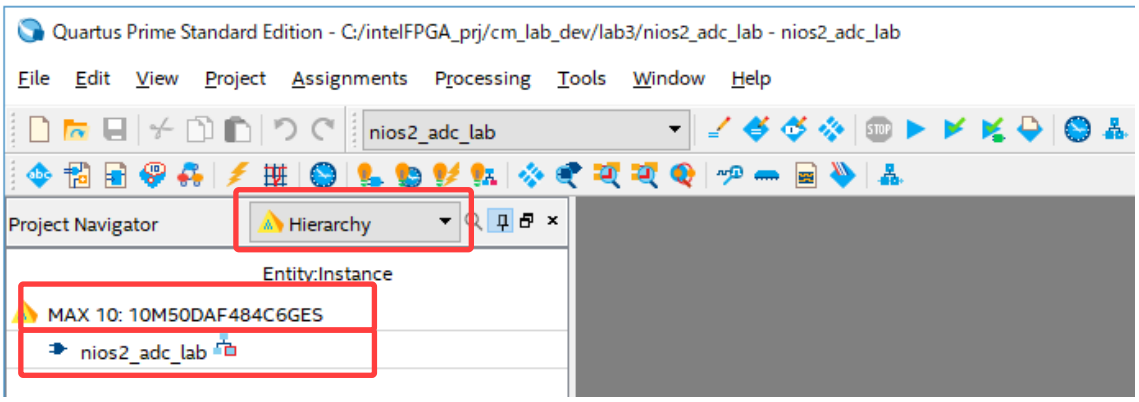
C:/intelFPGA_prj/cm_lab_dev/lab3/nios2_adc_lab.qpf



【図 3-1-1】 Open Project

____ 2. Quartus Prime の画面左上に位置する Project Navigator ウィンドウの ***Hierarchy ビュー*** において、以下 2 点を確認してください。

- **nios2_adc_lab** が登録されている
- ターゲットの FPGA 型番が **10M50DAF484C6GES** である



【図 3-1-2】 Hierarchy ビュー (Project Navigator ウィンドウ)

2. プラットフォーム・デザイナーでシステム・モジュールを作成

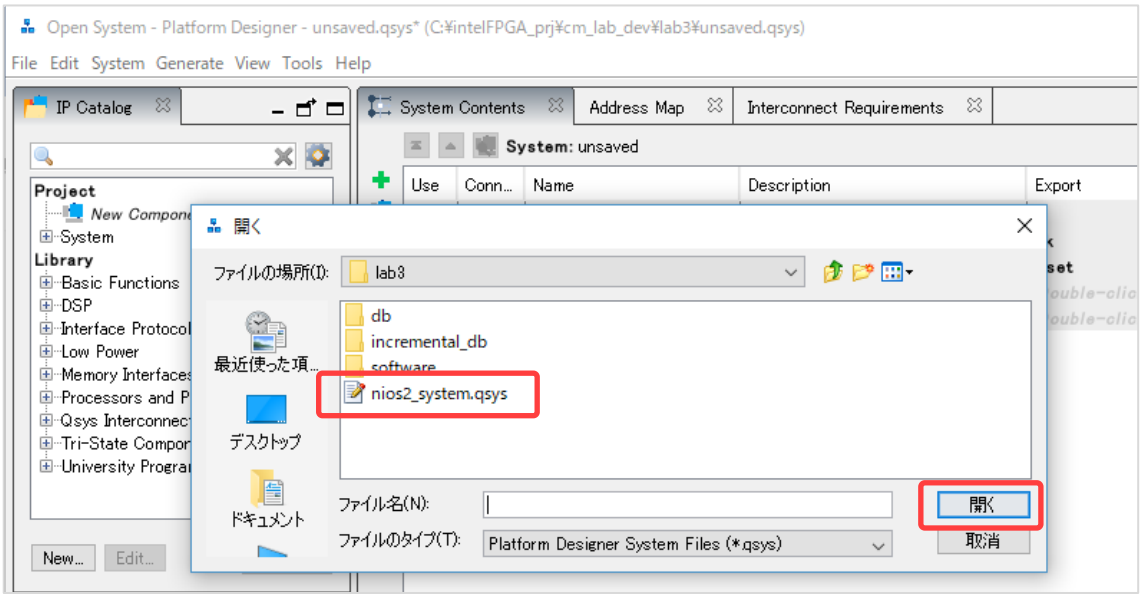
Quartus Prime に付属するシステム統合ツール プラットフォーム・デザイナー (Platform Designer) を使用して、Nios II プロセッサーを含むシステム・モジュールを作成します。

2-1. プラットフォーム・デザイナーを起動

____1. Quartus Prime のメニューから ***Tools*** ➤ ***Platform Designer*** を選択します。

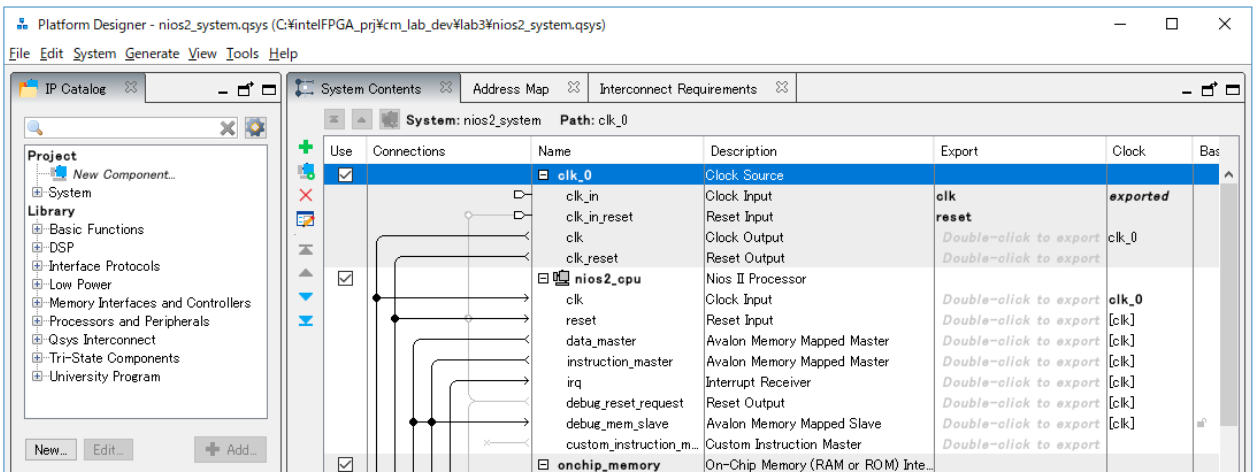
____2. プラットフォーム・デザイナーが起動し、ファイルの選択画面が表示されます。

この演習には、途中まで作成したシステム・モジュールがあらかじめ用意されています。作業フォルダーから ***nios2_system.qsys*** を選択し、***[開く]*** ボタンをクリックします。



【図 3-2-1】 Platform Designer で .qsys を選択

作成途中のシステム・モジュールが復元されます。

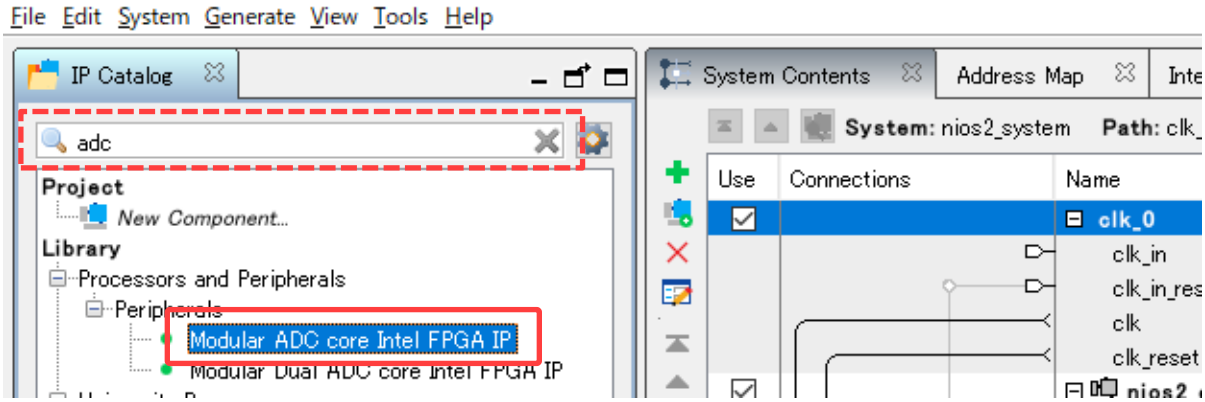


【図 3-2-2】 Platform Designer で .qsys を復元

2-2. ADC コアを追加

____1. プラットフォーム・デザイナーの IP Catalog ウィンドウにある検索バーに、***adc*** とキーワードを入力します。

フィルター検索された Library から、***Modular ADC core Intel FPGA IP*** を選択し、***[Add]*** ボタンをクリックします。

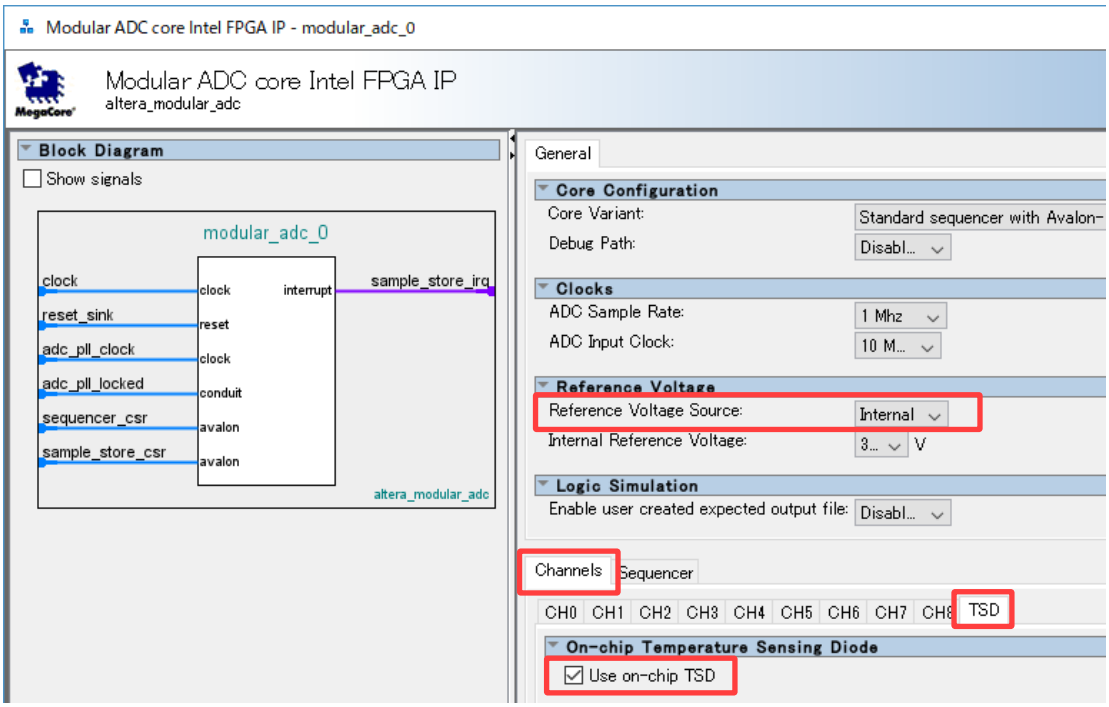


【図 3-2-3】 IP Catalog から ADC IP モジュールを検索

____2. Modular ADC core Intel FPGA IP の ***General タブ*** において、【表 3-2-1】および【表 3-2-2】のとおりにパラメーターを設定します。

【表 3-2-1】 Modular ADC core Intel FPGA IP のパラメーター設定 (1)

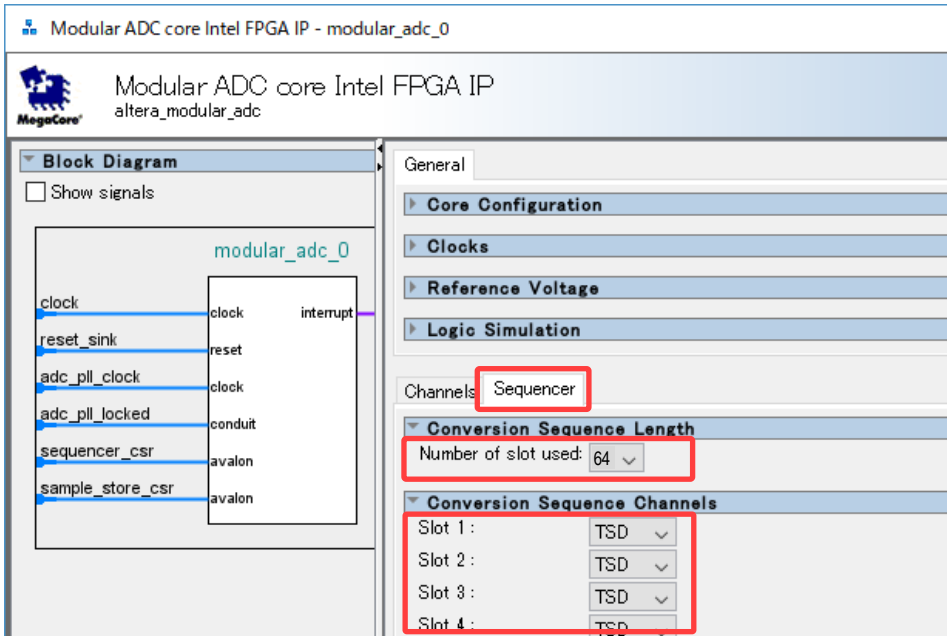
項目	設定内容
Reference Voltage > Reference Voltage Source	Internal
Channels タブ > TSD タブ	Use on-chip TSD オプション = ✓ (On)



【図 3-2-4】 Modular ADC core Intel FPGA IP のパラメーター設定 (1)

【表 3-2-2】 Modular ADC core Intel FPGA IP のパラメーター設定 (2)

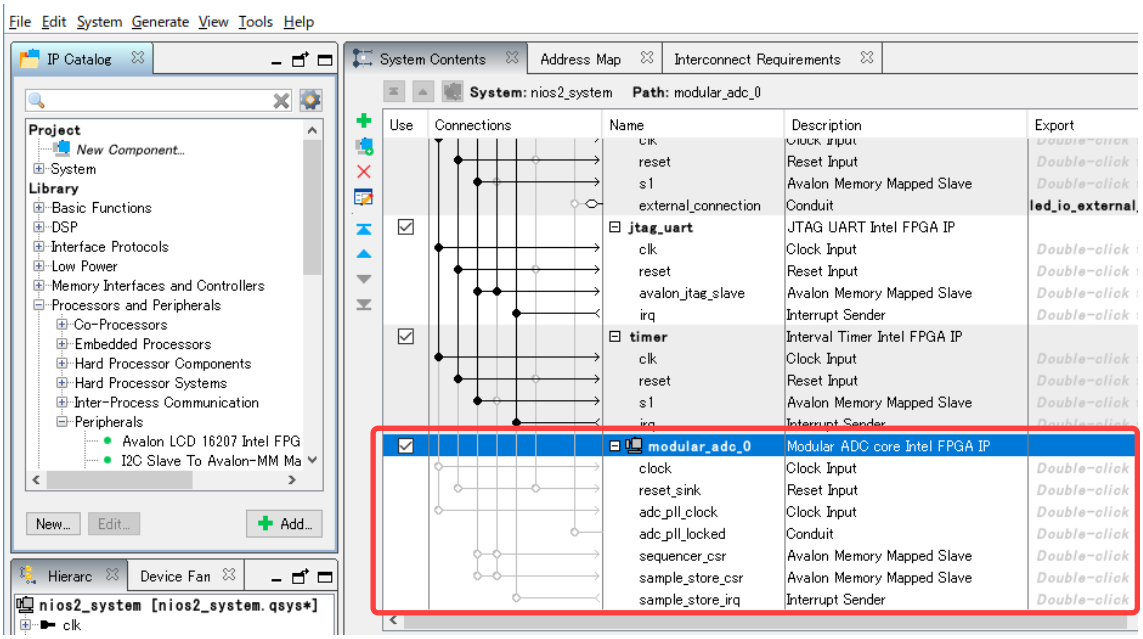
項目	設定内容
Sequencer タブ > Conversion Sequence Length > Number of slot used	64
Slot 1 ～ Slot 64	TSD



【図 3-2-5】 Modular ADC core Intel FPGA IP のパラメーター設定 (2)

____3. Modular ADC core Intel FPGA IP 画面の右下に位置する ***[Finish] ボタン*** をクリックし、ADC IP のパラメーター設定ウィンドウを閉じます。

プラットフォーム・デザイナーの System Console 内に Modular ADC core Intel FPGA IP が追加されました。

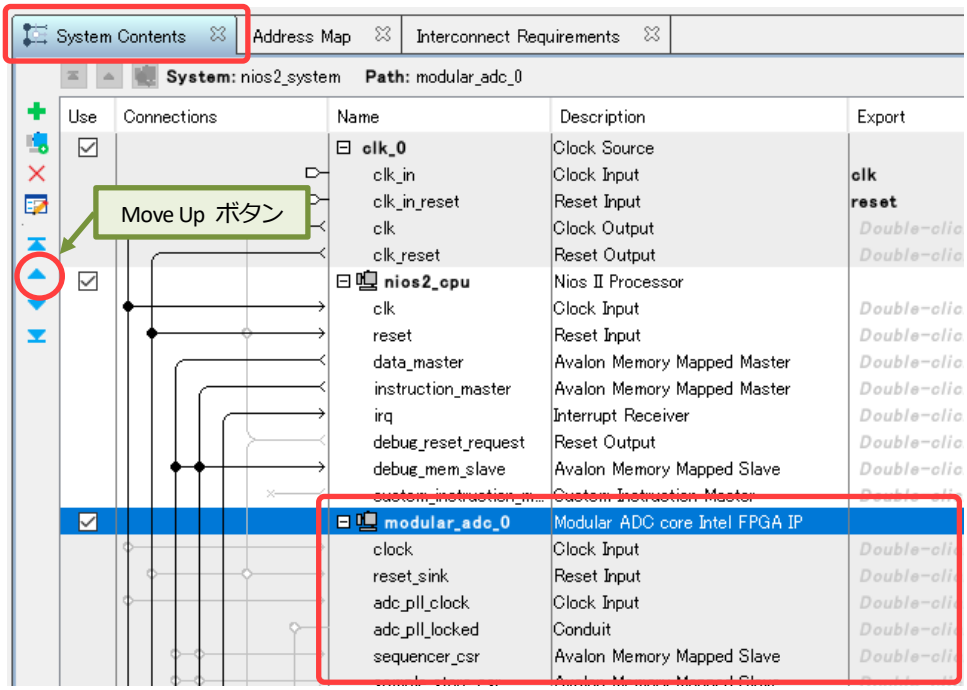


【図 3-2-6】 nios2_system に Modular ADC core Intel FPGA IP が追加

2-3. ADC コアを Nios II プロセッサーに接続

____1. プラットフォーム・デザイナーの ***System Contents*** タブにおいて、modular_adc_0 を見やすい位置にするため、ツールバーの ***[Move Up] ボタン*** を 5 回クリックし、nios2_cpu の下に移動します。

※ 画面を見やすくするための操作です。必須ではありません。

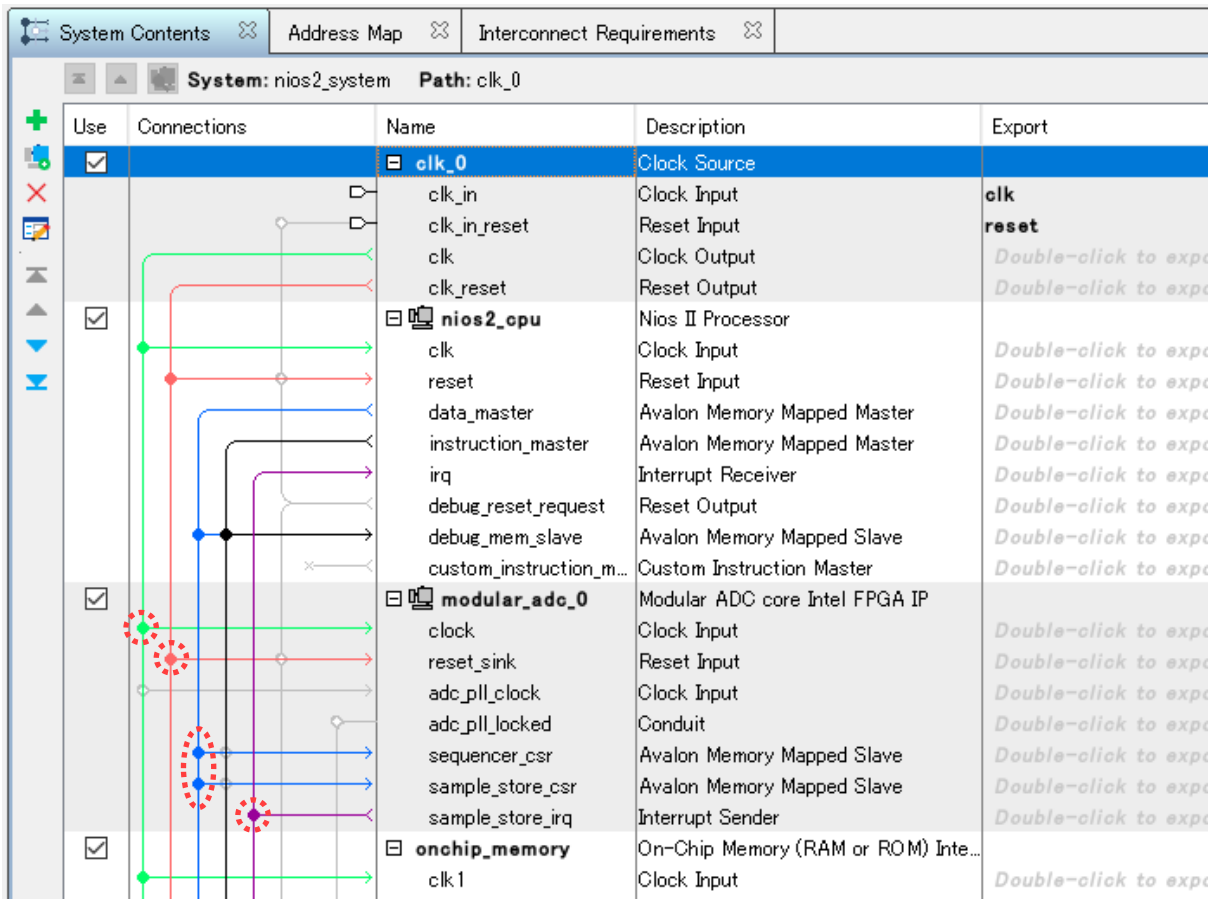


【図 3-2-7】 modular_adc_0 の配置変更

____ 2. Connections の項目において、【表 3-2-3】および【図 3-2-8】を参考に ADC コアにクロックおよびリセットを接続し、Nios II プロセッサーと接続します。

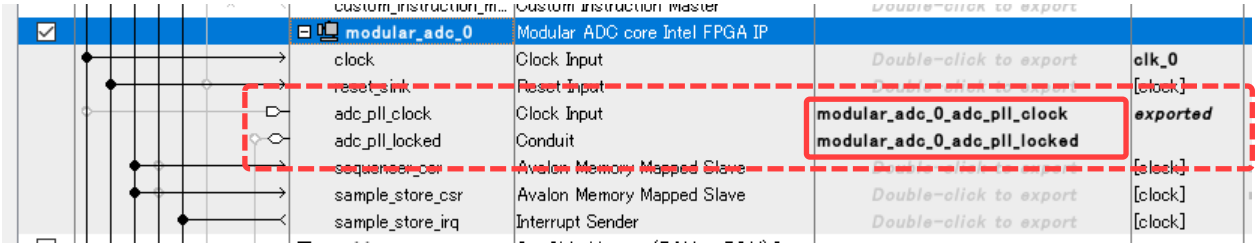
【表 3-2-3】 クロックとリセットの接続

Name	ポート名	接続先
modular_adc_0	clock	clk_0.clk
	reset_sink	clk_0.clk_reset
	sequencer_csr	nios2_cpu.data_master
	sample_store_csr	nios2_cpu.data_master
	sample_store_irq	nios2_cpu.irq



【図 3-2-8】 クロック、リセット、Nios II と接続

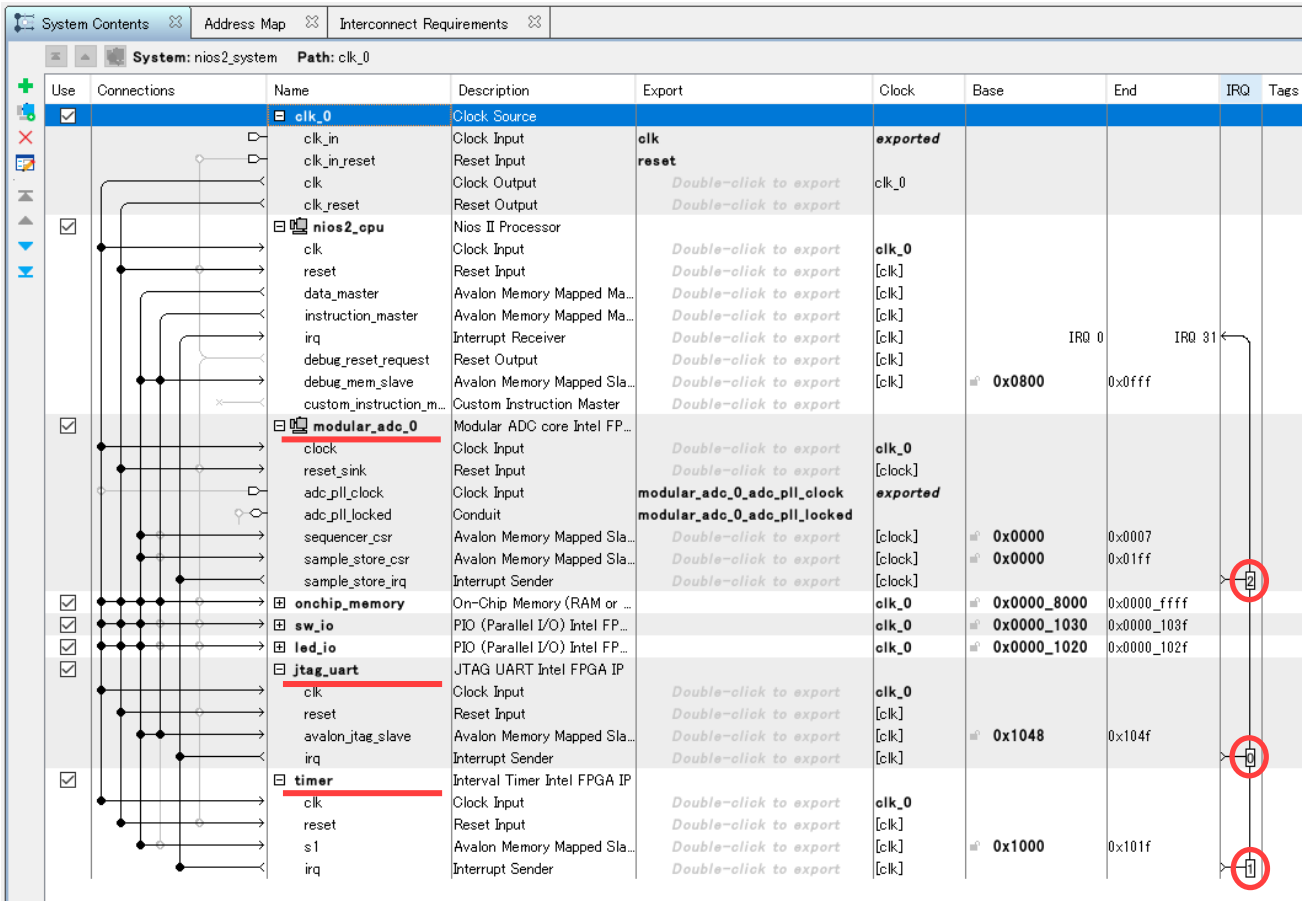
____ 3. ADC コアに対して外部からクロックを供給するため、adc_pll_clock と adc_pll_locked ポートの Export カラムをダブルクリックし、ポート名を表示させます。



【図 3-2-9】 PLL と接続するためのポートを作成

____ 4. jtag_uart から Nios II プロセッサーへの割り込み接続を設定します。

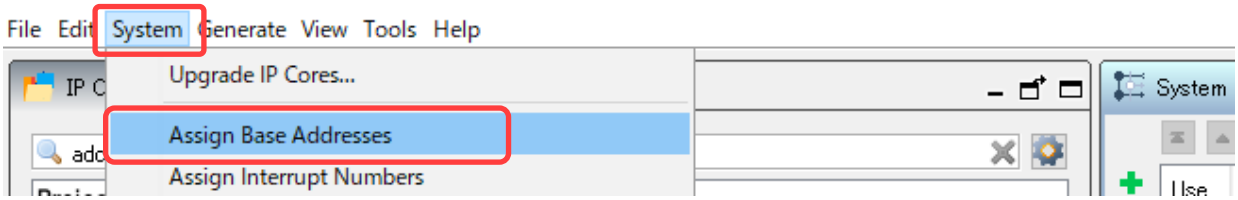
IRQ カラムの接続の交点において、jtag_uart に [0]、timer に [1]、modular_adc_0 に [2] と設定されていることを確認します。(【図 3-2-10】参照)



【図 3-2-10】 割り込みの設定

2-4. ベースアドレスを設定

____1. プラットフォーム・デザイナーのメニューから ***System ➤ Assign Base Address*** を選択し、ベースアドレスを自動で適切な値に設定します。



【図 3-2-11】 Assign Base Address

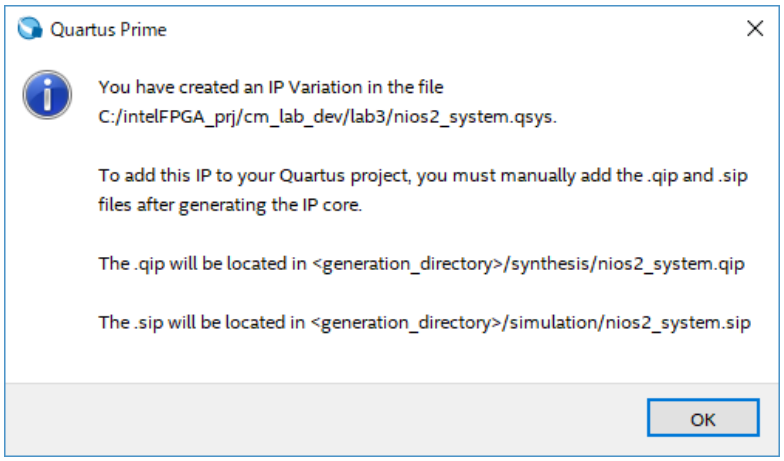
____2. プラットフォーム・デザイナーのメニューから ***File ➤ Save*** を選択し、nios2_system.qsys ファイルを保存します。保存が完了したら、Save System ウィンドウを ***[Close]*** ボタンで閉じます。

2-5. システム・モジュールを生成

____ 1. プラットフォーム・デザイナーのメニューから ***Generate*** ➤ ***Generate HDL*** を選択すると、Generation ウィンドウが表示されます。設定項目がありますが、この演習ではデフォルトのまま、右下にある ***[Generate]*** ボタンをクリックし、システムを生成します。

____ 2. Generate ウィンドウのメッセージ・ボックスに “Generate: completed successfully.” と表示されれば、システム・モジュールが正常に生成されました。Generate ウィンドウを ***[Close]*** ボタンをクリックして閉じます。

____ 3. プラットフォーム・デザイナーのメニューから ***File*** ➤ ***Exit*** を選択し、プラットフォーム・デザイナーを閉じます。以下のウィンドウが表示されたら ***[OK]*** ボタンをクリックして閉じます。



【図 3-2-12】 Generation ウィンドウ

3. ハードウェア・デザインの作成

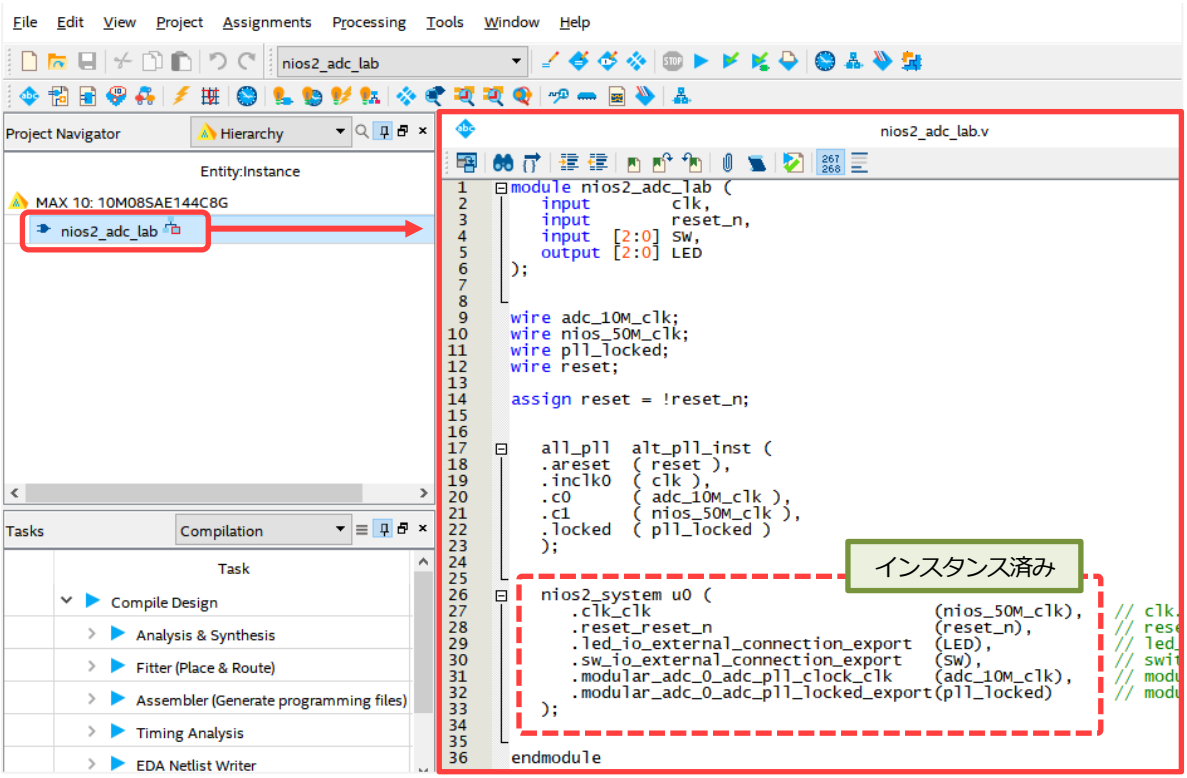
プラットフォーム・デザイナーで生成したシステム・モジュールを最上位階層に配置して、ハードウェア・デザインを作成します。

この演習では、最上位階層デザインファイルがすでに用意され、システム・モジュールのインスタンスも完了しています。

3-1. ファイルの確認

____ 1. Quartus Prime の画面左上に位置する Project Navigator ウィンドウの Hierarchy ビュー内に表示されている nios2_adc_lab をダブルクリックし、nios2_adc_lab.v が表示されることを確認してください。(【図 3-1-1】参照)

コードを確認後、nios2_adc_lab.v タブ右端にある ***[X] ボタン*** をクリックし、ファイルを閉じます。



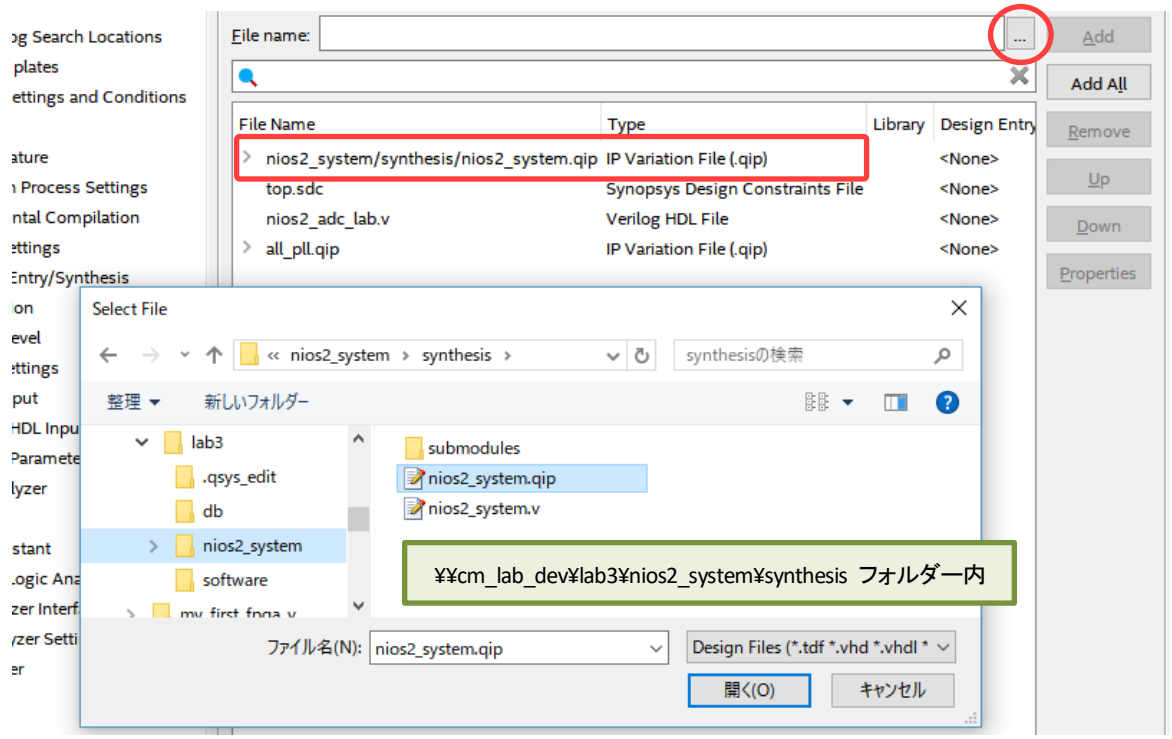
【図 3-3-1】 nios2_adc_lab.v

____ 2. 先ほど作成したシステム・モジュール nios2_system を現在のプロジェクトに登録します。

Quartus Prime のメニューから **Project ➤ *Add/Remove Files in Project*** を選択します。***File name*** 欄の右端にあるブラウズボタンをクリックし、Select File ウィンドウにおいて ***¥lab3¥nios2_system¥synthesis*** 内にある ***nios2_system.qip*** を選択します。(【図 3-3-2】参照)

このプロジェクトに必要なその他のデザインファイルは、登録済みです。

nios2_system.qip が登録されたことを確認後、Settings ダイアログボックスを ***[OK] ボタン***で 閉じます。



【図 3-3-2】 nios2_systemqip を選択

3-2. FPGA の外部端子をレイアウト (ピンアサイン) する

作成したデザイン上の信号を FPGA を外部端子へアサインするため、ピン番号を指定します。

なお、この演習ではすでにピンアサインが実施されています。どのように設定されているかを確認しましょう。

____ 1. Quartus Prime のメニューから ***Assignments*** ➤ ***Pin Planer*** を起動し、***All Pins リスト*** の枠内が【図 3-3-3】のとおり設定されていることを確認してください。

Node Name	Direction	Location	I/O Standard	I/O Bank
leds[0]	Unknown	PIN_T20	1.5 V	5
leds[1]	Unknown	PIN_U22	1.5 V	5
leds[2]	Unknown	PIN_U21	1.5 V	5
leds[3]	Unknown	PIN_AA21	1.5 V	5
leds[4]	Unknown	PIN_AA22	1.5 V	5
reset_n	Unknown	PIN_D9	3.3-V LVTTL	8
clk	Unknown	PIN_M9	2.5 V	2
<<new node>>				

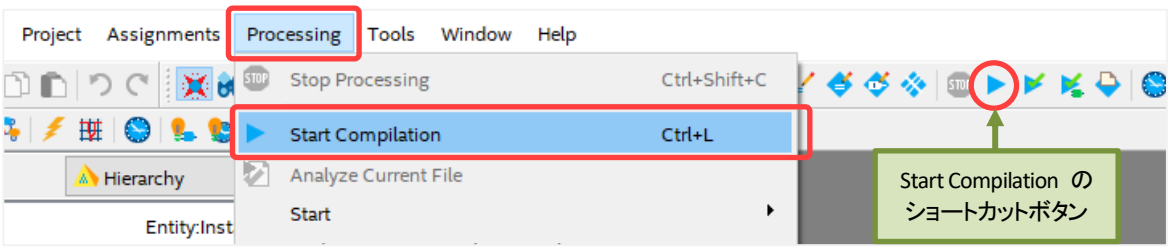
All Pins

【図 3-3-3】 All Pins リスト (Pin Planner)

____ 2. 問題がなければ、Pin Planner のメニューから ***File*** ➤ ***Close*** により Pin Planner を閉じます。

3-3. ハードウェア・デザインをコンパイル

____1. Quartus Prime のメニューから ***Processing*** ➤ ***Start Compilation*** をクリックし、コンパイルを開始します。



【図 2-3-5】コンパイル実行

コンパイルには多少の時間を要します。

____2. Messages ウィンドウに、Quartus Prime Full Compilation was successful. のインフォメーション・メッセージが表示されたら、エラーが発生することなくコンパイルが終了しています

この演習はエラーが発生することなくコンパイルが完了するはずです。もしエラーが発生した場合は、解消しないと次のステップには進めません。回避するためにエラーの原因を追究し修正する必要があります。

また、エラー・メッセージの他、ワーニング・メッセージが発生する場合もあります。ワーニングは解消しなくても次のステップへ進めますが、必ず内容を確認し、その内容を回避すべきか無視できるのかをユーザーが判断してください。

各メッセージ内容の詳細を確認するには、ヘルプ機能を活用してください。 (14 ページ【図 1-4-3】参照)

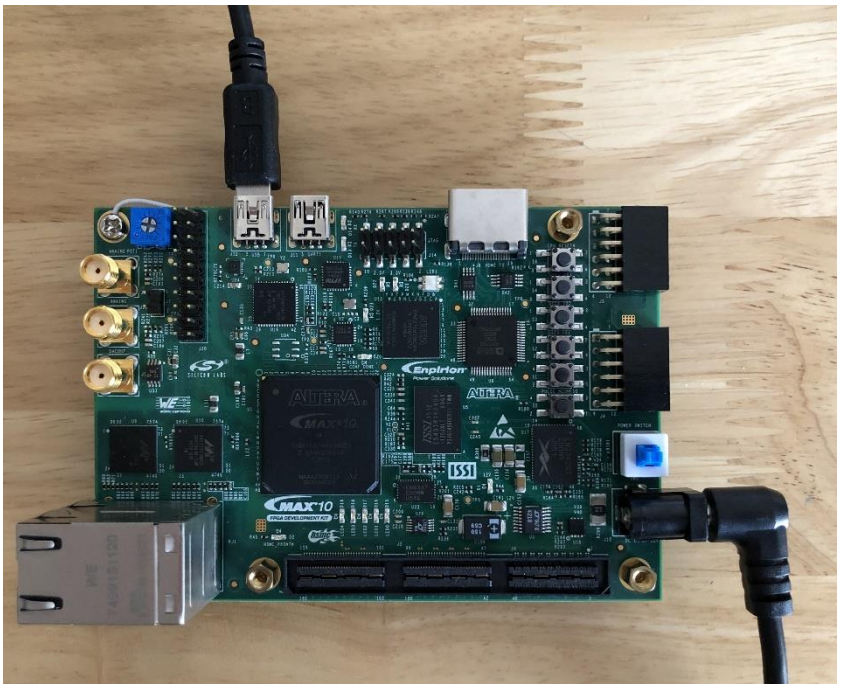
4.　ハードウェア・デザインを FPGA へダウンロード

作成したハードウェア・デザインのデータを MAX 10 FPGA にダウンロードします。

> ⚠ **注記:**
> 実際の開発では、コンパイル後に Quartus Prime の Timing Analyzer によるタイミング検証を行い、期待どおりの動作が実現できるかを検証します。期待するタイミングを満足できることが確認できたら、ボード上のデバイスへデータを書き込みます。
> 今回の演習ではタイミング検証を省略していますが、自身の開発時は必ずタイミング検証を行った上でデバイスへの書き込みを行ってください。

> ⓘ Note:
> MAX 10 開発キットは、FPGA へデータをダウンロードする際 ボードに組み込まれたインテル FPGA ダウンロード ・ ケーブル II (旧称 USB Blaster™ II) 回路を使用します。パソコンとボードの接続は、キットに付属している mini-USB ケーブルで行います。パソコンとボードが通信するためには、インテル FPGA ダウンロード・ケーブル II 用のドライバーをインストールする必要があります。インストール方法は、【表 1-1】No. 5 をご覧ください。

____ 1. キット付属の電源コネクタを接続し、ボードの電源が OFF であることを確認してください。その後、付属の mini-USB ケーブルを用いてボードとパソコンを接続します。 (【図 2-4-1】参照)

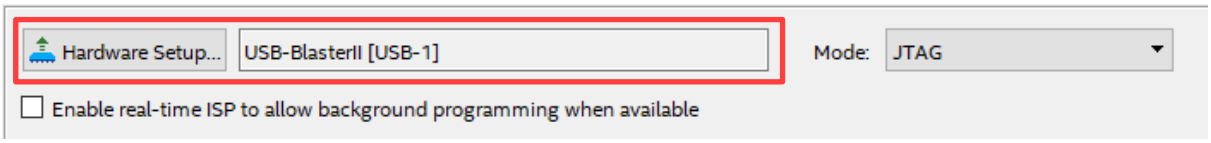


【図 3-4-1】 パソコンととボードを接続

____ 2. 開発ボードへ電源を供給します。

____3. Quartus Prime のメニューから ***Tools*** ➤ ***Programmer*** を選択し、Programmer を起動します。

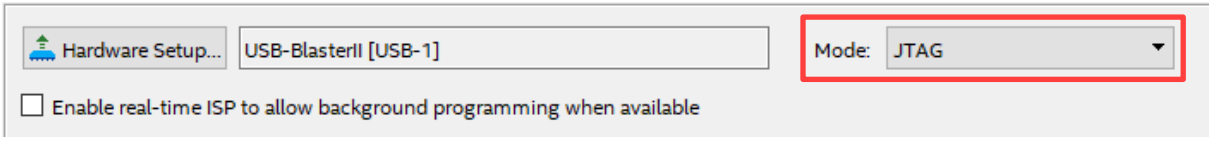
____4. Programmer の ***[Hardware Setup]*** ボタン 右横の欄が、***USB-Blaster II*** と表示されていることを確認します。



【図 3-4-2】 Hardware Setup

もし No Hardware 表示の場合には、***[Hardware Setup]*** ボタン をクリックし、***Hardware Settings タブ*** の ***Currently selected hardware*** 項のプルダウン・リストから、***USB-Blaster II*** を選択し、***[Close]*** ボタンをクリックしてください。

____5. Programmer の ***Mode*** プルダウン・リストから ***JTAG*** を選択します。



【図 3-4-3】 Mode

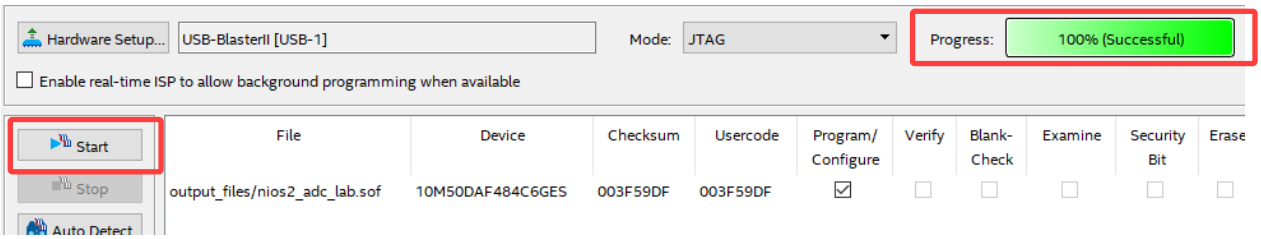
____ 6. Programmer の ***Files*** 欄に、ダウンロードするファイル ***nios2_adc_lab.sof*** が選択されていることを確認します。

<none> となっている場合には、***[Add File]*** ボタン をクリックし、起動したファイルブラウザから nios2_adc_lab.sof を選択します。(作業フォルダー内 ***output_files フォルダー*** に生成されています。)

____7. プログラミング・オプションの ***Program/Configure*** にチェックを入れます。

____8. Programmer の ***[Start]*** ボタン をクリックし、データの書き込みを開始します。

Progress バーが 100% になったら書き込み完了です。Messages ウィンドウには Successful のインフォメーションが表示されます



【図 3-4-4】 書き込み完了

ハードウェア・デザインのデータは FPGA に転送されましたが、FPGA 内部の Nios II プロセッサーが実行するためのプログラムがまだありません。

続いて、Nios II プロセッサーが実行するソフトウェアの環境を構築し、実行しましょう。

5. ソフトウェアを実行

この演習では、Nios II SBT を使用して、C ソースコードのビルド、およびターゲット・プログラムの開発ボードへのダウンロードまでを実施します。

5-1. Nios II SBT を起動

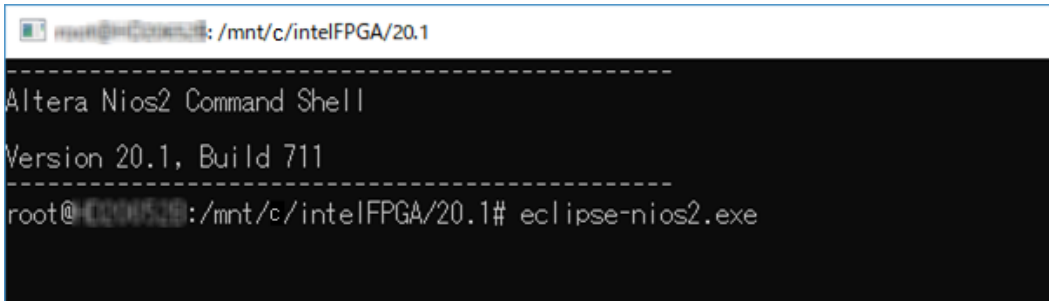
____1. Quartus Prime のメニューから ***Tools ➤ Nios II Software Build Tools for Eclipse*** をクリックして起動します。

ⓘ Note:

もし この起動方法で、GUI が表示されない、あるいは ソフトウェア・プロジェクト作成時にエラー (Failed to execute: wsl ./create-this-app –no-make) が発生した場合は、以下のフローにより Nios II SBT を起動してください。

<Windows 10 の場合>

1. Windows 10 の [スタート] ➤ ***Intel FPGA <version_build> Standard Edition ➤ Nios II Command Shell (Quartus Prime <version>)*** を右クリック選択し、ショートカット・アイコンのファイルの場所を開きます。
2. ***Nios II Command Shell (Quartus Prime <version>)*** を [管理者として実行] し、Nios II Command Shell を起動します。
3. `eclipse-nios2.exe` コマンドを実行して起動します。



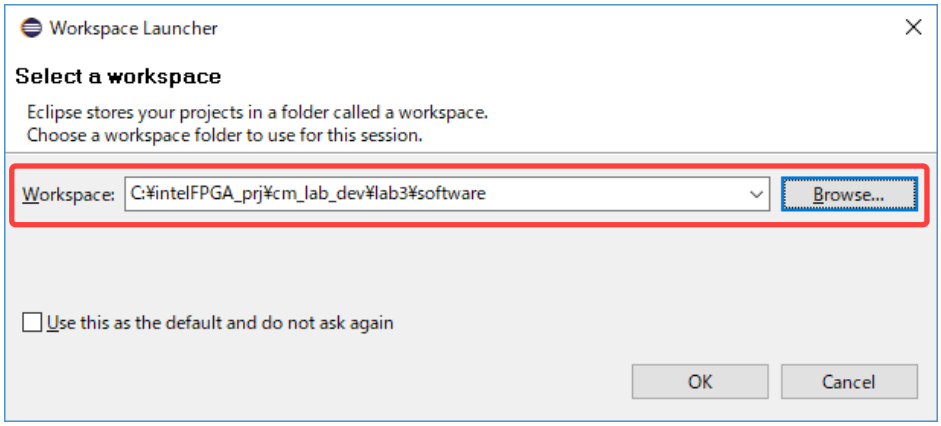
<Linux OS の場合>

1. ターミナル上で下記コマンドにより Nios II Command Shell を起動します。

   `<Nios II EDS install path>/nios2_command_shell.sh`

2. `eclipse-nios2` をタイプして起動します。

____ 2. Workspace Launcher が起動します。 ***[Browse] ボタン*** をクリックし Workspace に演習 3 用の Quartus Prime プロジェクト・フォルダー下の ***software*** を指定して ***[OK] ボタン*** をクリックします。

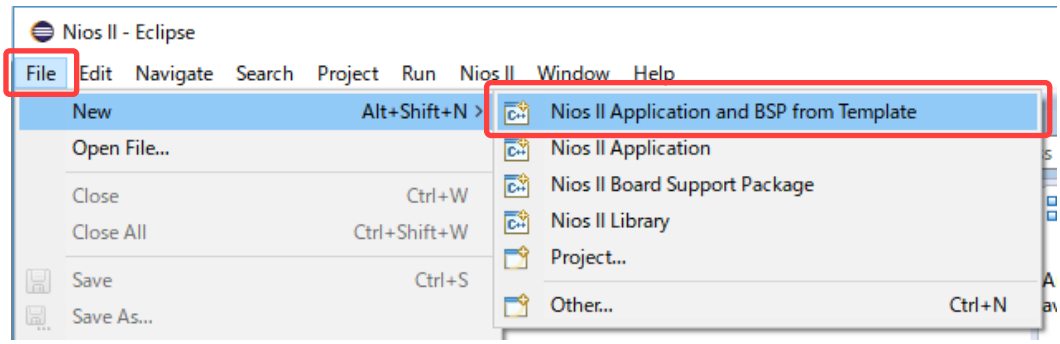


【図 3-5-1】 Workspace Launcher

Nios II SBT が起動します。

5-2. ソフトウェア・プロジェクトを作成

____ 1. Nios SBT 上のメニューから ***File ➤ New ➤ Nios II Application and BSP from Template*** をクリックし起動します。

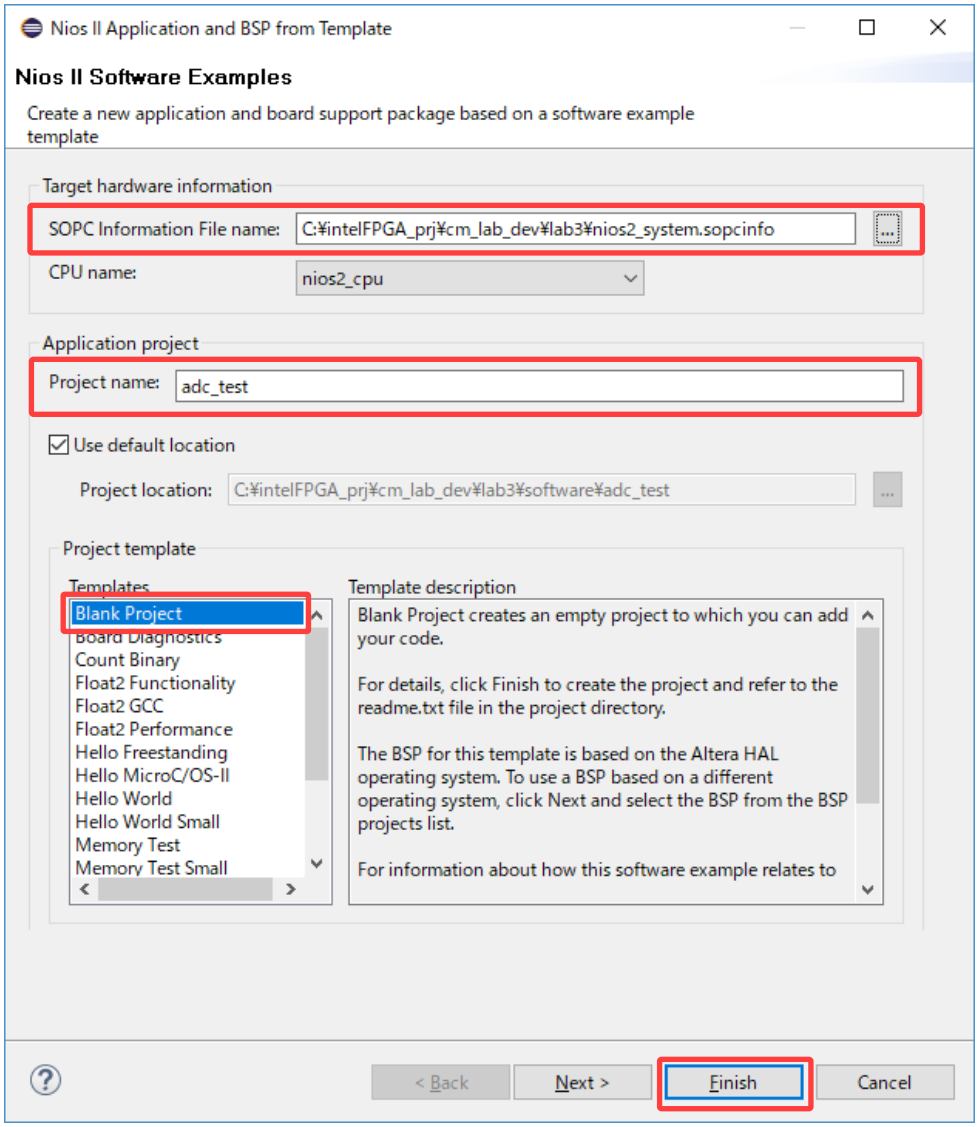


【図 3-5-2】 Nios II Application and BSP from Template を起動

____ 2. プラットフォーム・デザイナーで作成したシステムのハードウェア情報が記述されたシステム定義ファイルを指定します。(【図 3-5-3】参照)

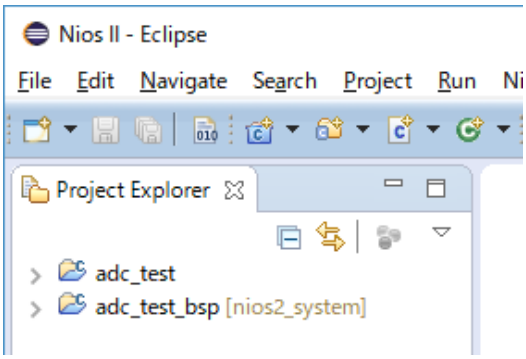
- Target hardware information エリアの ***SOPC Information File name*** 欄に、ハードウェア・デザインを作成した lab3 フォルダー内の ***nios2_system.sopcinfo*** を選択します。
- Application project エリアの ***Project name*** 欄に、***adc_test*** と入力します。
- Templates 枠内 から ***Blank Project*** を選択します。

設定後、***[Finish] ボタン*** をクリックします。



【図 3-5-3】 Nios II Application and BSP from Template

____ 3. 新しいソフトウェア・プロジェクトが作成され、Nios II SBT の Project Explorer タブ内に ***adc_test*** と ***adc_test_bsp*** が追加されます。

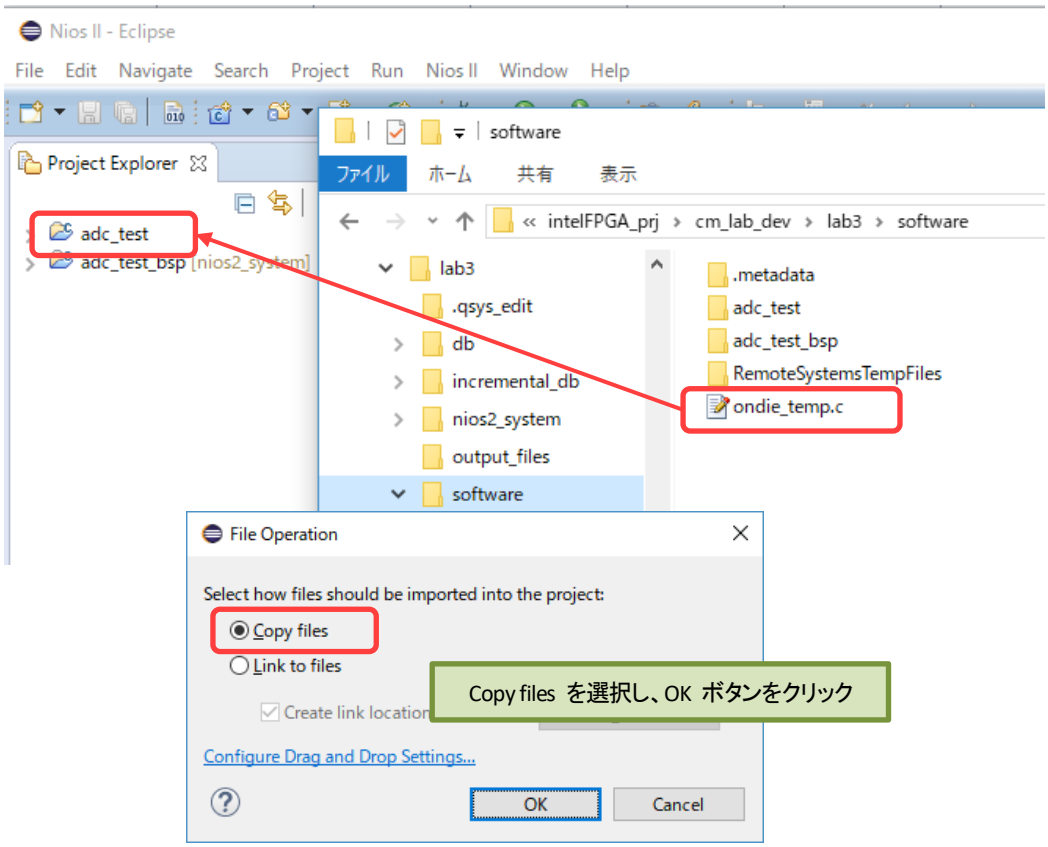


【図 3-5-4】 ソフトウェア・プロジェクト作成

5-3. C ソースコードのインポート

あらかじめ用意された C 言語のソースファイルをソフトウェア・プロジェクトにインポートします。

____ 1. Windows エクスプローラを開き、この演習 3 用フォルダー下の ***software*** フォルダーにある ***ondie_temp.c*** ファイルを、Nios II SBT の Project Explorer タブ内にある ***adc_test*** フォルダーにドラッグ&ドロップします。

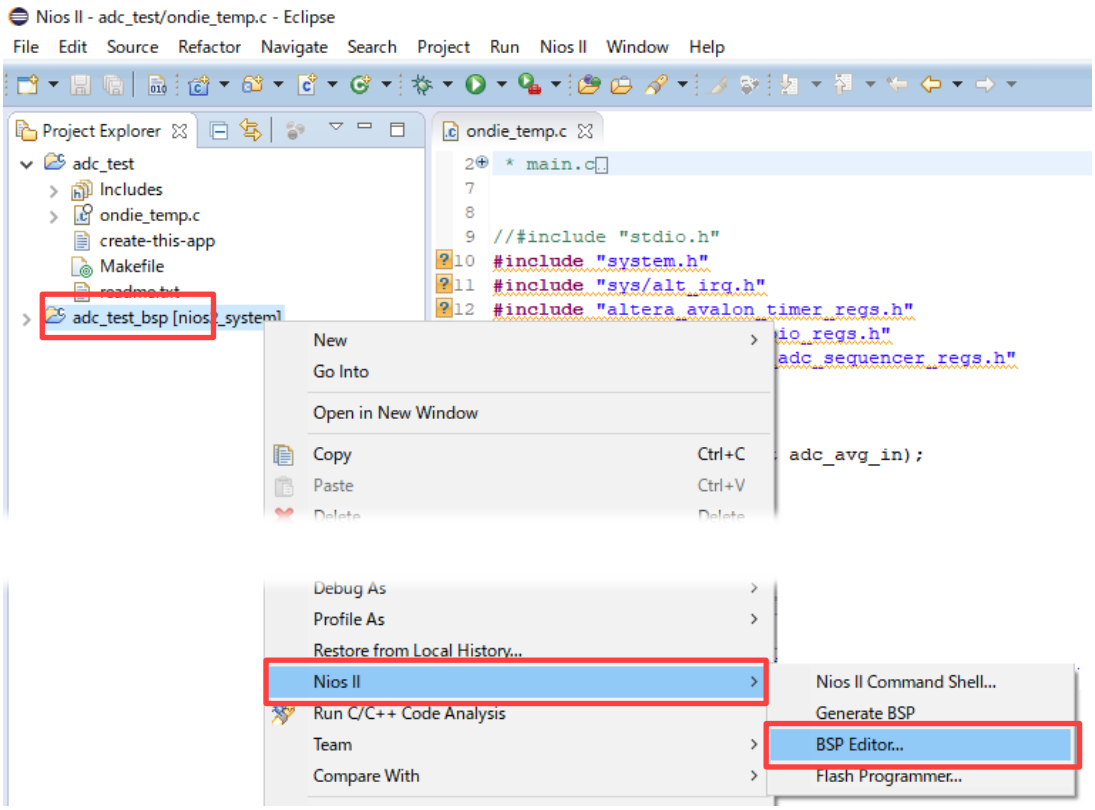


【図 3-5-5】 C ソースコードをプロジェクトへコピー

____ 2. Nios II SBT の Project Explorer タブ内にある adc_test フォルダーを展開し、ondie_temp.c が adc_test フォルダーにインポートされていることを確認してください。ondie_temp.c をダブルクリックすると、C ソースコードが表示されます。

5-4. ソフトウェア・プロジェクトのビルド

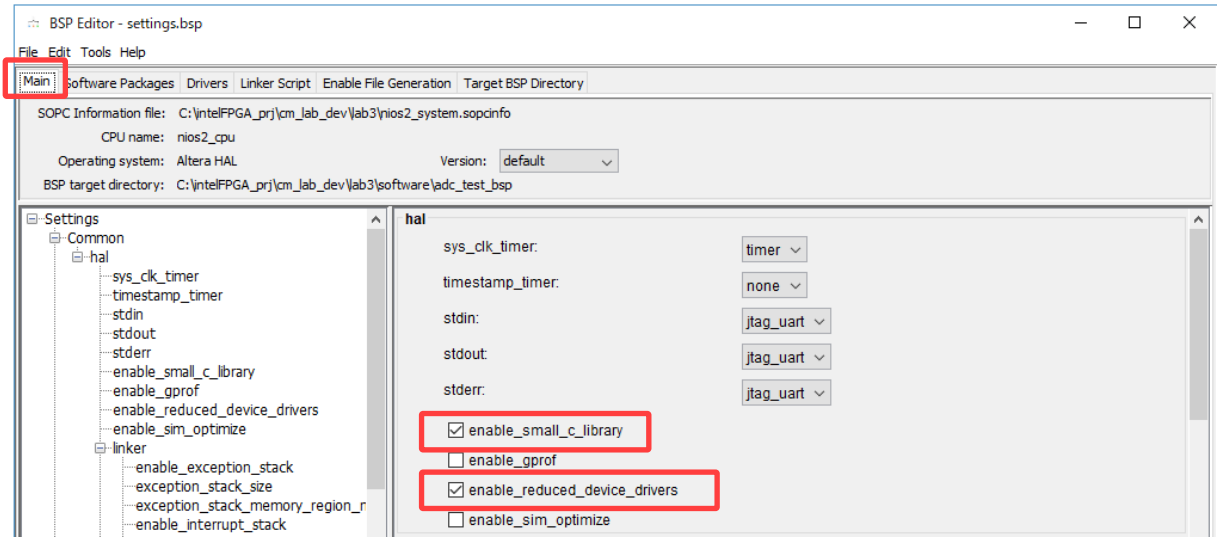
____ 1. Nios II SBT の Project Explorer タブ内にある ***adc_test_bsp*** フォルダーを右クリックで選択 ➤ ***Nios II*** ➤ ***BSP Editor*** を選択し、システムの設定を確認します。(【図 3-5-6】参照)



【図 3-5-6】 BSP Editor を起動

____2. BSP Editor ウィンドウの ***Main タブ*** において、下記オプションを有効 (✓) にします。

- enable_small_c_library = On
- enable_reduced_device_drivers = On



【図 3-5-7】 BSP Editor

その後、BSP Editor ウィンドウ右下に位置する ***[Generate] ボタン*** をクリックし、***[Exit] ボタン*** で BSP Editor ウィンドウを閉じます。

____3. ソフトウェアをビルド (Build) します。

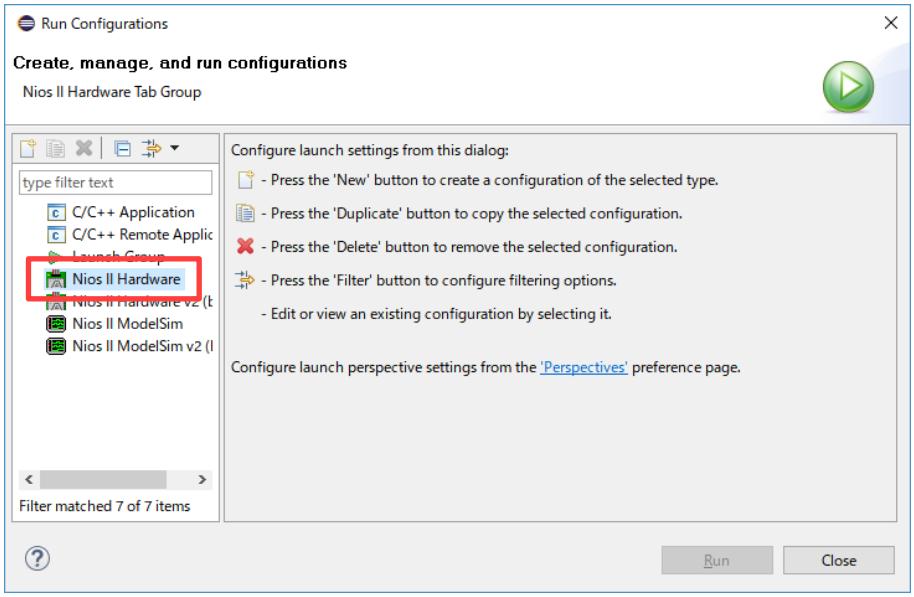
Nios II SBT の Project Explorer タブ内にあるアプリケーション･プロジェクトのフォルダー (adc_test) を選択し、右クリック ➤ ***Build Project*** をクリックします。

エラーなくビルドが完了することを確認します。

5-5. ソフトウェア･プログラムを実行

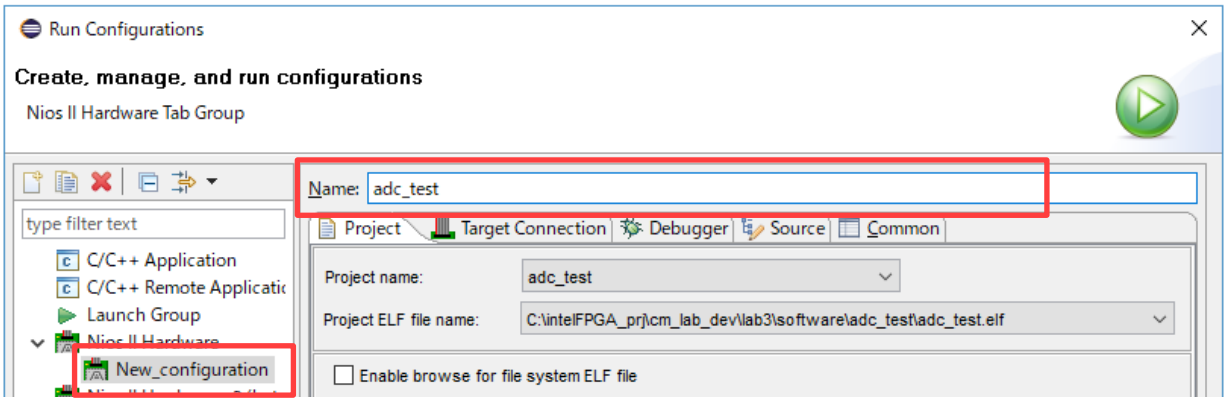
____1. Nios II SBT の Project Explorer タブ内の ***adc_test フォルダー*** を右クリック ➤ ***Run As*** ➤ ***Run Configurations*** をクリックします。

____2. Run Configurations ウィンドウのアイテムから ***Nios II Hardware*** を選択してダブルクリックします。



【図 3-5-8】 Run Configurations

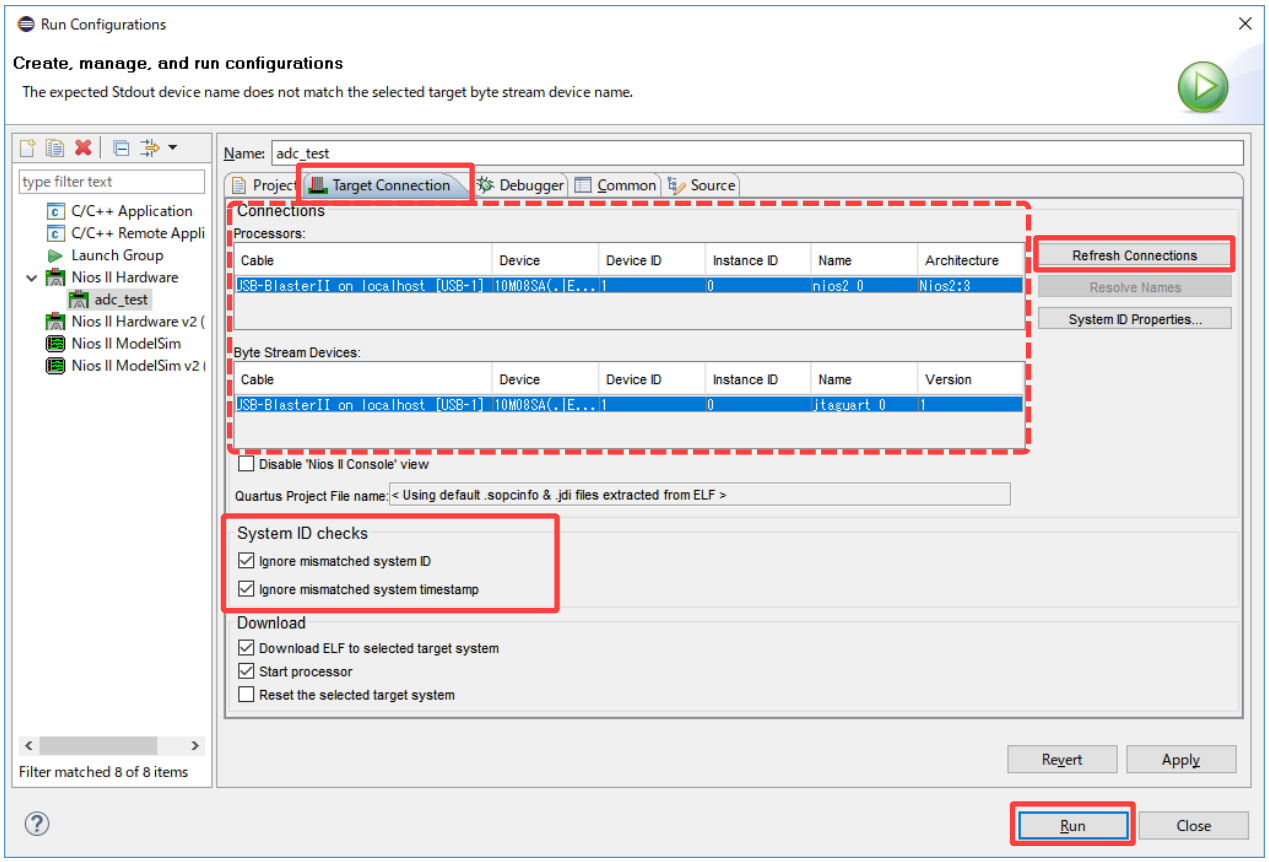
____ 3. Run Configurations ウィンドウの ***Name 欄*** において、デフォルトの New_configuration から任意の名称に変更します。この演習では ***adc_test*** と入力してください。



【図 3-5-9】 Name 欄を編集

____4. Run Configurations ウィンドウの ***Target Connection タブ*** を選択し、右上の ***[Refresh Connections] ボタン*** をクリックします。【図 2-5-15】のように USB-Blaster II が検出されるのを確認します。

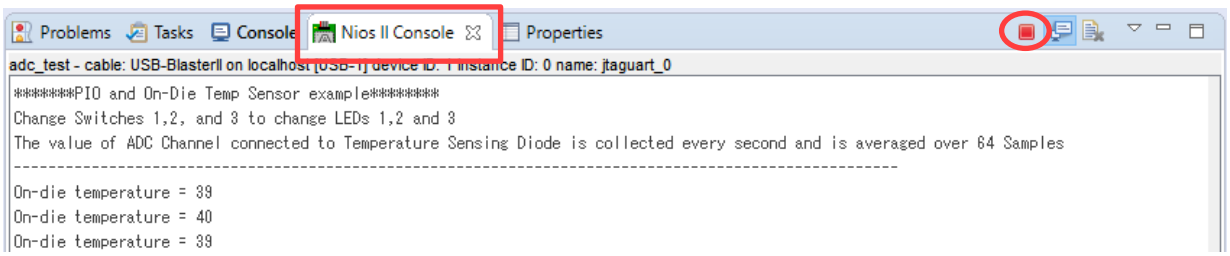
System ID checks 欄 の 2 つのオプションを有効にし、***[Run] ボタン*** をクリックします。



【図 3-5-10】 Run Configurations ダイアログボックス

____5. Nios II SBT の Nios II Console ウィンドウに、C 言語ソースコード上に記載した printf 関数の出力キャラクタが確認できます。同時に、開発ボード上の MAX 10 内蔵の ADC が取得した温度が表示されています。

確認したら、Nios II Console ウィンドウ右上の赤いボタンをクリックしてターミナルを終了します。



【図 3-5-11】 Nios II Console ウィンドウ

____6. Nios II SBT のメニューから ***File ➤ Exit*** で、Nios II SBT を閉じます。

すべての作業が終了したら、Quartus Prime のメニューから ***File*** ➤ ***Close Project*** により、演習 3 のプロジェクトを終了してください。

また、開発ボードの電源を OFF にしてください。

ⓘ Note:

この演習では、ハードウェア・イメージおよび Nios II のブート・イメージを 共に MAX 10 FPGA の SRAM 領域にダウンロードしているので、開発ボードの電源を切ると FPGA 内のデータは消去されます。そのため製品として成立させるためには、ハードウェア・イメージは MAX 10 の Flash メモリ領域 (CFM) にプログラムし、Nios II のブート・イメージは MAX 10 の内部あるいは外部に用意した不揮発性メモリに格納する必要があります。

※　これらの作業は今回の演習では行いません。別コースのセミナを受講されるか、メーカーのドキュメントや弊社の Web コンテンツをご覧ください。

以上で、演習 3 の作業はすべて終了です。

演習はすべて終了しました。お疲れ様でした。