FPGA デザインへの仮想計器の追加

概要

チュートリアル
TU0135 (v1.0) 2008 年 5 月 17 日

このチュートリアルでは、FPGA デザインにおける仮想計器コンポーネントの利用について紹介します。その際、入力信号を直接監視したり、スクリプトを使用して出力信号を制御します。デザインのターゲットデバイスとしては、Desktop NanoBoard に挿入されたドータボード上の物理的な FPGA デバイスを使用します。

CUSTOM_INSTRUMENT コンポーネントは、FPGA デザイン中の信号を監視および制御するための、詳細なカスタマイズが可能な計器です。

計器の設定の一部として、独自の GUI (デザインがターゲットデバイスにプログラムされ、計器にアクセスするときに表示されるインターフェース) を作成することができます。標準的なコンポーネントと計器コントロールのパレットにより計器パネルをすばやく構築できると共に、コントロールに関連付けられているさまざまなプロパティを使用して微調整することができます。

回路上で計器に配線されている定義済みの IO 信号を、カスタム GUI のさまざまなコントロールに直接接続したり、必要に応じて IO を処理するための独自の DelphiScript コードを記述することができます。スクリプトは、計器がポーリングするときと、指定したイベントが発生したときに呼び出されます。

このチュートリアルでは、仮想計器コンポーネントを取り入れ、Desktop NanoBoard NB2DSK01 に挿入するドータボード上の FPGA にプログラムされる、単純なデザインを使用します。このデザインでは、Desktop NanoBoard に搭載されている以下のリソースを利用します。

- 8 ウェイ DIP スイッチ
- ユーザ LED
- DAUGHTER BD TEST/RESET ボタン

機能を説明するため、以下のように計器を設定します。

- さまざまな監視ベースのコントロールを使用して、DIP スイッチの現在の状態を表示します (すべて受信信号に直接接続します)。
- 監視ベースのコントロールを使用して、DAUGHTER BD TEST/RESET ボタンの現在の状態を表示します (受信信号に直接接続します)。
- スクリプトを使用してユーザ LED への出力を制御します。通常動作時は、計器パネル上のコントロールを使用して設定した値で LED への出力が決まります。この値は、以下の 2 つの方法で優先変更することができます。
  - DAUGHTER BD TEST/RESET ボタンを押した場合。優先設定が有効であることを示すメッセージが表示され、すべての LED が消灯します。
  - 計器パネル上で専用のボタンを押した場合。優先設定が有効であることを示す同様のメッセージが表示され、すべての LED が消灯します。

このチュートリアルで紹介するデザイン例 (図 1) は、Altium Designer をインストールしたフォルダの \Examples\NanoBoard Common\FPGA Hardware\Custom Instrument Design フォルダにあります。詳細を理解したり手順の一部を省略するには、この例を参照してください。

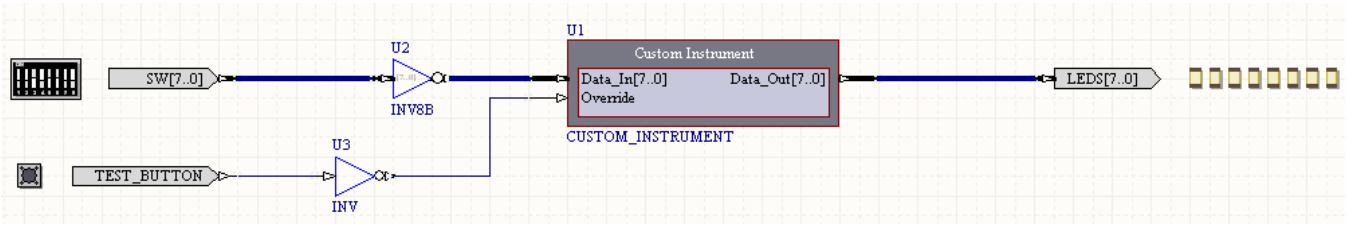


図 1. 仮想計器の機能を示すために使用する単純な FPGA デザイン

このチュートリアルでは、Altium Designer での FPGA のデザインの基礎に関する実際上の知識が前提となっています。このチュートリアルの前提条件ではありませんが、プロジェクトの作成、コンポーネントの配置、配線、設定、デザイン処理などの基本について知りたい場合は、『*TU0116 Getting Started with FPGA Design*』を参照してください。

FPGA ベンダツールに関する重要な注意事項

デザインを Desktop NanoBoard またはドータボードプラグイン上の物理デバイスにダウンロードする前に、コンピュータに適切なベンダツールをインストールしておく必要があります。これらのツールは、ターゲットデバイス用に FPGA デザインを配置および配線するために使用します。FPGA ベンダツールはシステムに付属しておらず、別途入手する必要があります。

さまざまなドータボードが Desktop NanoBoard で使用できます。これらのドータボード上の FPGA デバイスは、Web 上にあるそれぞれのベンダのダウンロード可能なツールや、これらのツールの製品版でサポートされています。選択したドータボードを使用するには、適切なツールをインストールする必要があります。

ベンダツールの詳細は、それぞれの FPGA ベンダの Web サイトを参照してください。

- *www.actel.com* にある Actel® Designer または Libero® IDE。ソフトウェアはダウンロード可能ですが、ライセンスが必要です。ライセンスについては Web サイトを参照してください。
- *www.altera.com* にある Altera® Quartus® II。Altera Quartus II Web Edition ソフトウェアは自由にダウンロードでき、ライセンスは不要です。
- *www.latticesemi.com* にある Lattice® ispLever®。ispLever Starter ソフトウェアはダウンロード可能ですが、ライセンスが必要です。ライセンスについては Web サイトを参照してください。
- *www.xilinx.com* にある Xilinx® ISE™。Xilinx ISE WebPACK は自由にダウンロードでき、ライセンスは不要です。

各ベンダのダウンロード可能なツールへのリンクは、当社の Web サイトの Vendor Resources エリア (*www.altium.com/Community/VendorResources*) にあります。このページには、Altium Designer から直接アクセスすることができます。**Devices** ビューがアクティブな状態で (**View » Devices View**)、メインの **Tools** メニューから **Vendor Tool Support** を選択します。

注記 :当社は、FPGA ベンダツールの技術サポートを提供していません。これらのツールのインストールについては、FPGA ベンダから提供されている情報を参照してください。

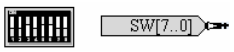
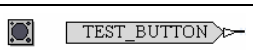
デザインの作成

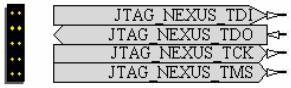
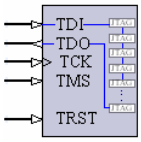
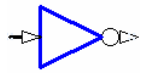
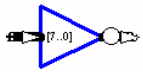
それでは、Altium Designer の FPGA プロジェクト内に単純な仮想計器デザインを作成します。

1. 新しい FPGA プロジェクトを作成し、`Custom_Instrument_Design.PrjFpg` という名前で新しいフォルダ `Custom Instrument Tutorial` に保存します。
2. 新しい回路図ドキュメントをこのプロジェクトに追加し、ファイルを `Custom_Instrument_Design.SchDoc` という名前で親プロジェクトと同じフォルダに保存します。

表 1 は、このデザインで使用する必要がある各種論理コンポーネントおよびデザインインターフェースコンポーネントと、それが含まれているソース統合ライブラリを示します。

表 1. 仮想計器デザイン回路図で必要なデザインコンポーネント

シンボル	コンポーネント名	内容	ライブラリ
	CUSTOM_INSTRUMENT	仮想計器	FPGA Instruments.IntLib
	DIPSWITCH	このコンポーネントは、Desktop NanoBoard 上の 8 ウェイ DIP スイッチとインターフェースします。これらのスイッチは、仮想計器への入力として使用します。	FPGA NB2DSK01 Port-Plugin.IntLib
	TEST_BUTTON	このコンポーネントは、Desktop NanoBoard 上の DAUGHTER BD TEST/RESET ボタンとインターフェースします。この信号は、仮想計器へのハードウェア優先設定入力として (反転して) 使用します。	FPGA NB2DSK01 Port-Plugin.IntLib
	LED	このコンポーネントは、Desktop NanoBoard 上のユーザ LED とインターフェースします。LED は、仮想計器の出力信号を視覚的に表	FPGA NB2DSK01 Port-Plugin.IntLib

		示するために使用します。	
	NEXUS_JTAG_CONNECTOR	このコンポーネントは、Soft JTAG チェーン信号 (NEXUS_TMS、NEXUS_TCK、NEXUS_TDI、NEXUS_TDO) とインターフェースし、基本的にチェーンをデザインに「持ち込み」ます。	FPGA NB2DSK01 Port-Plugin.IntLib
	NEXUS_JTAG_PORT	このコンポーネントは、Nexus 対応の仮想計器コンポーネントを Soft JTAG チェーンに「配線」するために使用します。	FPGA Generic.IntLib
	INV	反転器です。DAUGHTER BD TEST/RESET ボタンからのアクティブロー入力を反転するために使用します。	FPGA Generic.IntLib
	INV8B	8 ビット反転器、バスバージョンです。8 ウェイ DIP スイッチからのアクティブロー入力線を反転するために使用します。	FPGA Generic.IntLib

注記:アクティブロー入力を反転して計器に接続する理由は、スイッチをオンの位置に設定したときにライトが消灯するよりも点灯するほうが、監視目的では優れているためです。

3. CUSTOM_INSTRUMENT コンポーネントを回路図シートの中央に配置します。それを右クリックし、コンテキストメニューから **Configure** コマンドを選択し、*Custom Instrument Configuration* ダイアログを表示します。計器の設定については次のセクションで詳しく説明しますが、ここでは、配線のために以下の入力信号と出力信号を定義する必要があります。
   - デフォルトの入力信号の名前を、AIN[7..0] から Data_In[7..0] に変更します。
   - デフォルトの出力信号の名前を、AOUT[7..0] から Data_Out[7..0] に変更します。
   - Override という名前の別の入力信号を追加します。
4. 残りのコンポーネントを配置し、図 2 に示すようにデザインを配線します。デザイン内で該当するコンポーネントを必ず指定してください (たとえば、**Tools » Annotate Schematics Quietly** コマンドを使用します)。

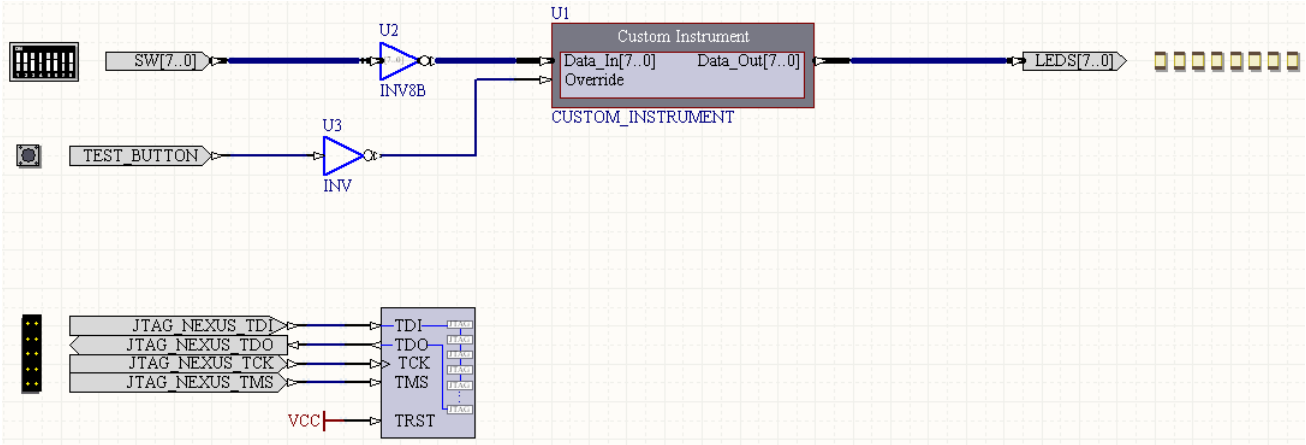


図 2. 最終的なデザイン - コンポーネントを配置し完全に配線したところ

5. 回路図とプロジェクトドキュメントを保存します。

これで基になるデザインが作成されました。次に、このチュートリアルの核心である、仮想計器自体の設定について説明します。

仮想計器の設定

仮想計器コンポーネントの設定は、*Custom Instrument Configuration* ダイアログで行います。このダイアログは、計器のインターフェース IO 信号を定義するために、前のセクションで簡単に使用しました。ここでは、計器を完全に設定し、単純なデザインプロジェクトの要件を満たすようにカスタマイズします。この設定では以下のことを行います。

- 設定ファイルとそれを Altium Designer で取得する方法の指定など、基本計器オプションを定義します。
- IO を監視および制御するためのさまざまな要素を使用してカスタム GUI を作成します。
- Desktop NanoBoard 上のユーザ LED への出力を制御するために使用する DelphiScript を記述します。

それでは始めましょう。

計器の基本オプションの定義

1. デザイン回路図面から、仮想計器のシンボルを右クリックし、コンテキストメニューから **Configure** を選択します。*Custom Instrument Configuration* ダイアログが表示され、デフォルトで **Signals** タブがアクティブになります (図 3)。

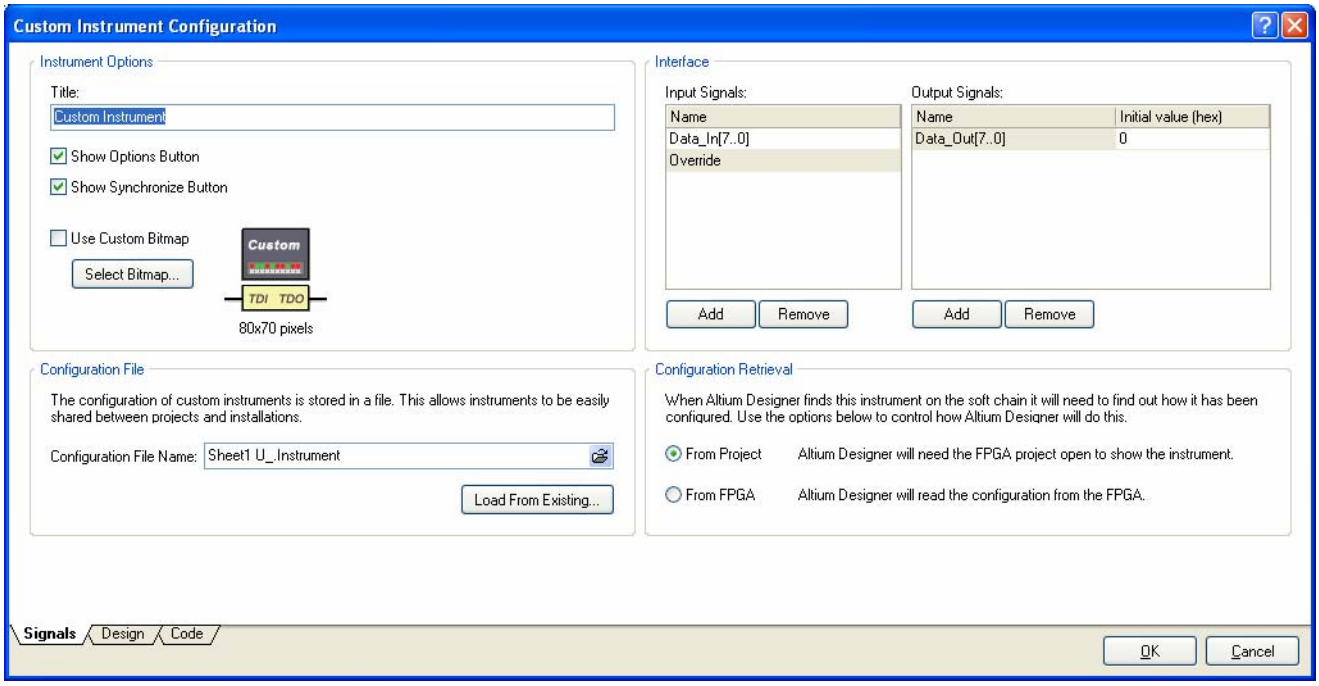


図 3. Signals タブ– 計器の基本設定オプションを表示

2. 計器のカスタマイズは、計器パネルの右下で使用されるタイトルにわたるため、タイトルを変更します。**Title** フィールドに `Example Instrument` と入力します。
3. パネル上に **Options** ボタンと **Synchronize** ボタンを表示するためにデフォルトの設定のままにします。
4. **Devices** ビューの Soft Devices JTAG チェーン内で計器を表すために使用するビットマップもカスタマイズすることができます。**Use Custom Bitmap** オプションを有効にし、**Select Bitmap** ボタンをクリックします。表示されるダイアログを使用して、Altium Designer がインストールされているフォルダの `\Examples\NanoBoard Common\FPGA Hardware\Custom Instrument Design` フォルダを参照し、`ExampleInstrumentIcon.bmp` ファイルを開きます。

タブの **Configuration File** 領域では、すべての設定情報が保存されるファイルを指定します。個別のファイル (`*.Instrument`) に設定情報を保存することで、カスタマイズされた計器を他のプロジェクトやインストールで使用することができます。自分好みの計器が使用できるときに、他の誰かの計器を使用する人はいません。

5. **Configuration File Name** フィールドの右にあるフォルダアイコンをクリックします。表示される *Save Configuration File To* ダイアログを使用して、設定ファイルを親プロジェクトと同じフォルダに `Custom_Instrument_Example.Instrument` として保存します。

Soft Devices JTAG チェーンで仮想計器が検出されたときに、Altium Designer が該当する設定情報にアクセスする方法は 2 つあります。

- **From Project** – 設定情報は、プロジェクトファイルと同じ場所にある、計器用の対応する `.Instrument` ファイルから取得されます。Devices ビューから計器を表示するためには、プロジェクトがオープンされている必要があります。

- **From FPGA** – 設定情報は、デザインと共に物理デバイスにダウンロードされ、ブロック RAM に保存されます。そこから直接取り出され、プロジェクトがオープンされている必要はありません。

6. このチュートリアルでは、計器の設定をターゲット物理 FPGA デバイスに保存します。タブの **Configuration Retrieval** 領域で **From FPGA** オプションを有効にします。

ここでの要件に合わせてダイアログの **Signals** タブを設定すると、図 4 に示すようになります。

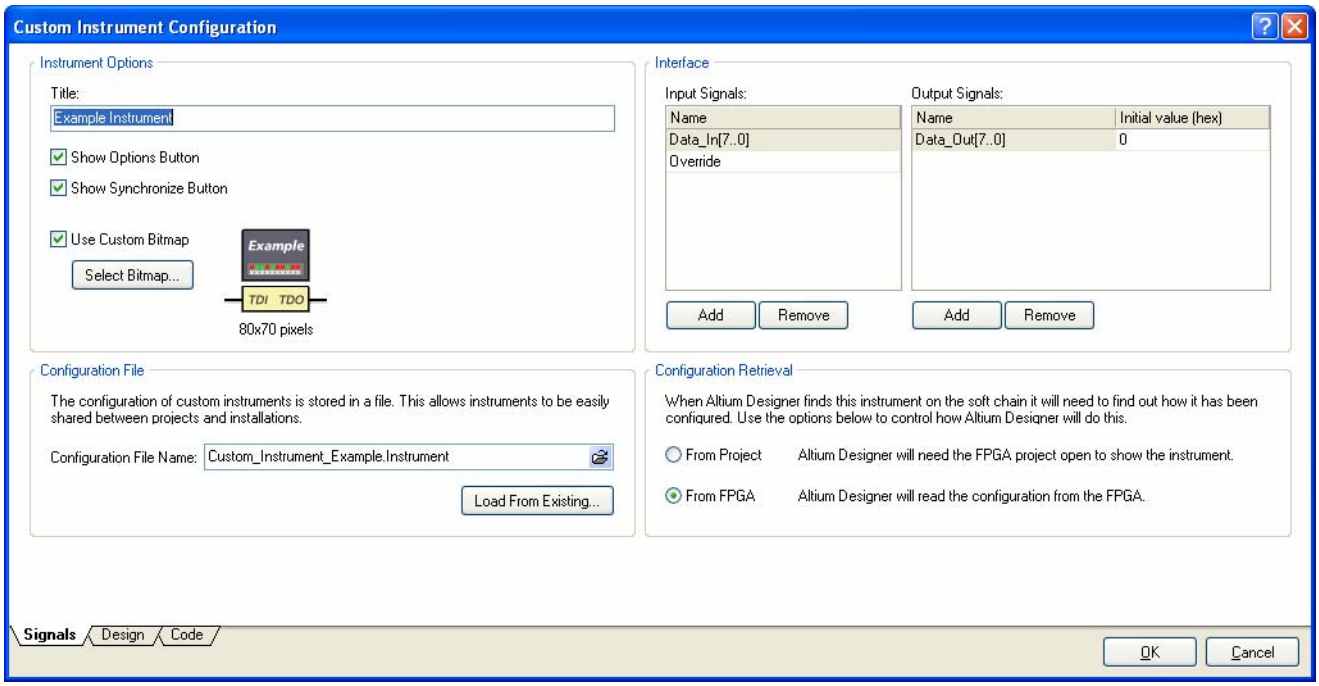


図 4. Signals タブ– デザインに合わせて完全に設定した状態

カスタム GUI の設計

次に GUI を作成します。GUI は、デザインをドータボードの FPGA にプログラムしたときにアクセスする、計器のカスタマイズされたパネルです。

1. *Custom Instrument Configuration* ダイアログの下部にある **Design** タブをクリックします。GUI を構築するための元となる「シェル」計器フォームが表示されます (図 5)。

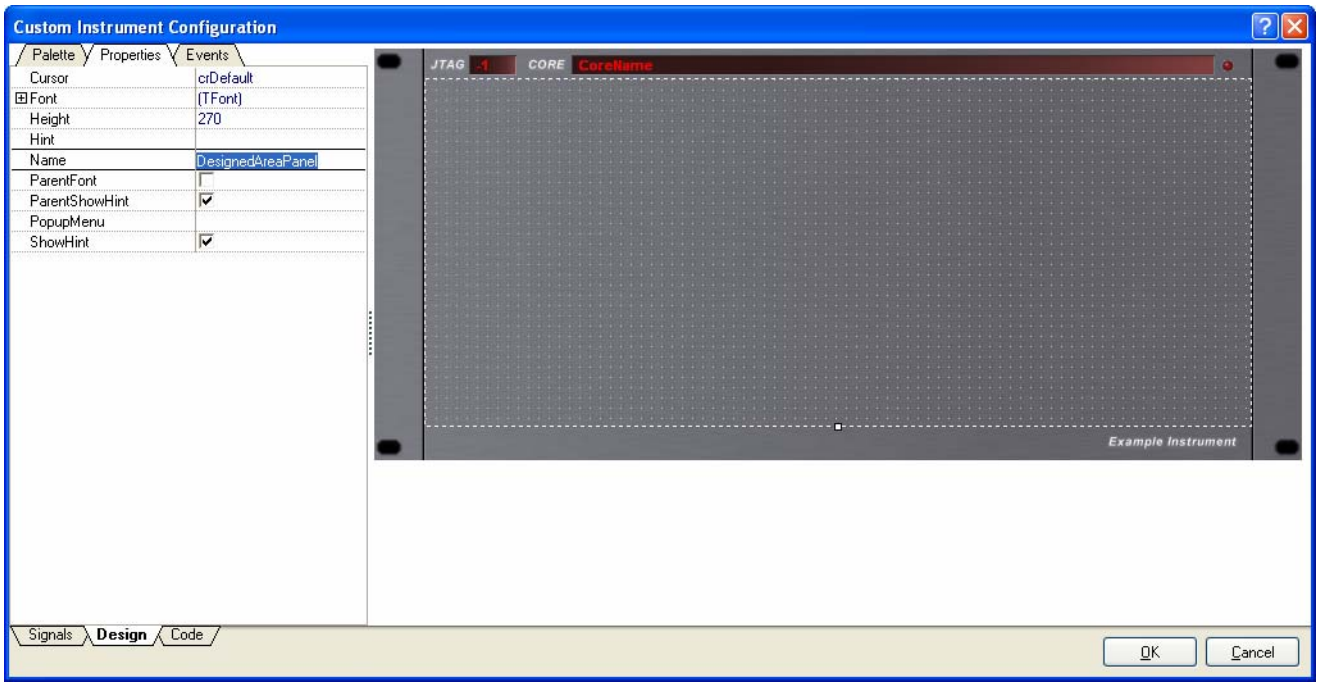


図 5. Design タブ– カスタム GUI を作成するためのキャンバスが表示される

フォームには、以下の 3 つのタブ付きパネルが関連付けられています。

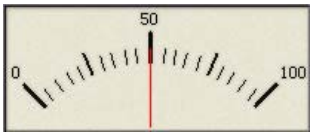
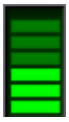
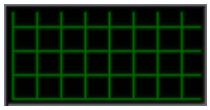
- **Palette** – フォームで使用するための、標準スクリプトコンポーネントと計器固有のコントロールがあります。
- **Properties** – フォーム中で現在選択されているコントロール (またはオブジェクト) のすべての属性が表示されます。
- **Events** – フォーム中で現在選択されているコントロール (またはオブジェクト) に該当するイベントの一覧が表示されます。このタブは、指定したイベントが発生したときに呼び出される、スクリプトのプロシージャまたは関数をコントロールに接続するために使用します。

フォーム中で現在選択されているオブジェクト `DesignedAreaPanel` は、フォームデザインの境界を表します。コントロールとオブジェクトはこの領域の中にしか配置できません。

2. さまざまなコントロールをパネルに追加する予定であるため、使用する領域を広げます。DesignedAreaPanel オブジェクトの **Height** プロパティを `270` から `352` に増やします。

表 2 に、カスタムパネル GUI で使用するさまざまな計器コントロールと、それらをフォームに配置するために使用する **Palette** パネルの **Instrument Controls** 領域内の対応するエントリを示します。

表 2. 仮想計器パネルで必要な計器コントロール

コントロール	コントロール名	配置するためにクリックするパレット上のシンボル	数
	**TInstrumentCaption**		**2**
	**TInstrumentGauge**		**1**
	**TInstrumentProgressBar**		**8**
	**TInstrumentLEDsPanel**		**2**
	**TInstrumentLEDDigits**		**1**
	**TInstrumentGraph**		**1**
	**TInstrumentNumericPanel**		**1**
	**TInstrumentLabel**		**2**
	**TInstrumentButton**		**1**

3. 図 6 に示すように、これらすべてのコントロールをフォームに配置します。この段階では、デフォルトコントロールに対する唯一の変更は、各 Progress Bar コントロールのサイズを小さくすることです。

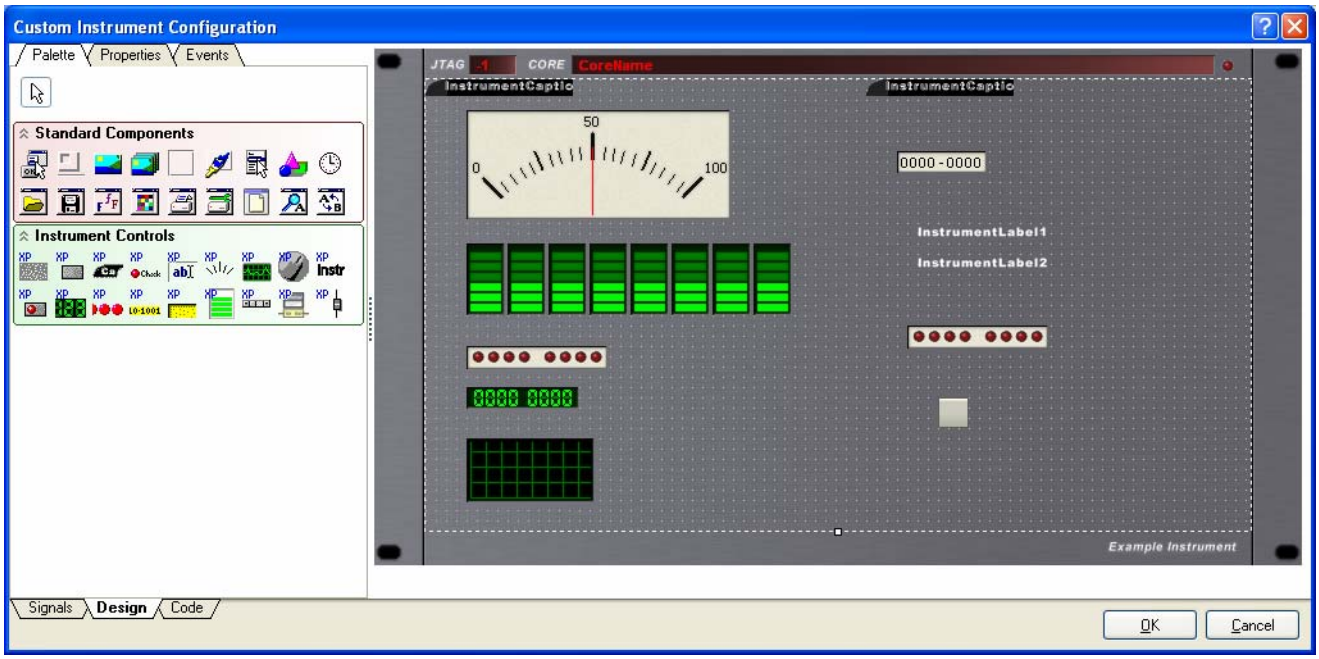


図 6. フォームに追加した初期コントロール

これで基本のコントロールが配置されたため、望ましい方法でパネルが表示されるようにプロパティを変更します。各コントロールを順に検討し、プロパティを調整します (**Properties** パネルから)。入力を監視するために使用するコントロールでは、コントロールに接続する信号を直接指定します。

4. 左上の Caption コントロールをクリックして選択します。以下のプロパティを変更します。
   - **AutoSize**: Disable
   - **Caption**: `DATA INPUT MONITORING` に変更
   - **Font » Color**: `CaptionText` に変更
   - **Font » Height**: `-11` に変更
   - **Font » Name**: `Tahoma` に変更
   - **Style » fsItalic**: Enable
   - **Width**: `296` に変更
5. Gauge コントロールをクリックして選択します。以下のプロパティを変更します。
   - **BigTicks » Color**: `Lime` に変更
   - **BigTicks » Step**: `32` に変更
   - **Color**: `Black` に変更
   - **Font » Color**: `White` に変更
   - **Max**: `256` に変更
   - **MediumTicks » Color**: `Silver` に変更
   - **MediumTicks » Step**: `16` に変更
   - **Needle » Kind**: `gnkTearDrop` に変更
   - **Numbers » Step**: `32` に変更
   - **Position**: `0` に変更
   - **SignalName**: ドロップダウンを使用して入力信号 `Data_In[7..0]` をこのコントロールに割り当て
   - **SmallTicks » Color**: `White` に変更
   - **SmallTicks » Step**: `8` に変更
6. Shift キーを押しながらクリックして、8 個の Progress Bar コントロールすべてを選択します。以下のプロパティを変更します。
   - **BarColor**: `Yellow` に変更
   - **BarInactiveColor**: `Gray` に変更

- **BarMiddleColor**: `Gray` に変更
- **ColorScheme**: 色を変更することで自動的に `icsCustom` に変わる
- **Max**: `1` に変更
- **Position**: `0` に変更

一番左の Progress Bar コントロールから始めて、各コントロールを順に選択し、**SignalName** プロパティを設定して、`Data_In[7..0]` 入力の対応する信号線を割り当てます。つまり、一番左のコントロールに `Data_In[7]` を設定し、次のコントロールに `Data_In[6]` を設定し、一番右のコントロールに `Data_In[0]` を設定します。

7. 左側の LED Panel コントロールをクリックして選択します。以下のプロパティを変更します。
   - **DigitsInGroup**: `8` に変更
   - **Metrics » EndSpace**: `1` に変更
   - **Metrics » Space**: `1` に変更
   - **Metrics » StartSpace**: `1` に変更
   - **SignalName**: ドロップダウンを使用して入力信号 `Data_In[7..0]` をこのコントロールに割り当て
8. LED Digits コントロールをクリックして選択します。以下のプロパティを変更します。
   - **DigitsInGroup**: `8` に変更
   - **Metrics » Space**: `6` に変更
   - **SignalName**: ドロップダウンを使用して入力信号 `Data_In[7..0]` をこのコントロールに割り当て
9. Graph コントロールをクリックして選択します。以下のプロパティを変更します。
   - **Color**: `InfoBk` に変更
   - **Grid » Visible**: Disable
   - **Height**: `64` に変更
   - **HighThreshold » Enabled**: Enable
   - **HighThreshold » Pen » Color**: `Teal` に変更
   - **HighThreshold » Value**: `255` に変更
   - **LowThreshold » Enabled**: Enable
   - **LowThreshold » Pen » Color**: `Teal` に変更
   - **Max**: `265` に変更
   - **Min**: `-10` に変更
   - **SignalForm » NormalSignal » Color**: `Red` に変更
   - **SignalName**: ドロップダウンを使用して入力信号 `Data_In[7..0]` をこのコントロールに割り当て
   - **Width**: `264` に変更
10. 右上の Caption コントロールをクリックして選択します。以下のプロパティを変更します。
    - **AutoSize**: Disable
    - **Caption**: `DATA OUTPUT CONTROL` に変更
    - **Font » Color**: `CaptionText` に変更
    - **Font » Height**: `-11` に変更
    - **Font » Name**: `Tahoma` に変更
    - **Style » fsItalic**: Enable
    - **Width**: `296` に変更
11. Numeric Panel コントロールをクリックして選択します。以下のプロパティを変更します。
    - **Name**: `Output_Data` に変更
    - **UseDefaultDigitClick**: Enable
12. InstrumentLabel1 コントロールをクリックして選択します。以下のプロパティを変更します。
    - **Caption**: `NB_Override` に変更
    - **Font » Color**: `Red` に変更
    - **Font » Height**: `-13` に変更
    - **Name**: `NB_Override` に変更

13. InstrumentLabel2 コントロールをクリックして選択します。以下のプロパティを変更します。
    - **Caption**: `Panel_Override` に変更
    - **Font » Color**: `Red` に変更
    - **Font » Height**: `-13` に変更
    - **Name**: `Panel_Override` に変更
14. 右側の LED Panel コントロールをクリックして選択します。以下のプロパティを変更します。
    - **Color**: `3DDkShadow` に変更
    - **Digits**: `1` に変更
    - **DigitsInGroup**: `1` に変更
    - **SignalName**: ドロップダウンを使用して入力信号 `Override` をこのコントロールに割り当て
    - **Style**: `icsGreen` に変更
15. Button コントロールをクリックして選択します。以下のプロパティを変更します。
    - **AllowAllUp**: Enable
    - **Caption**: `SOFTWARE OVERRIDE` と入力
    - **Font » Height**: `-13` に変更
    - **Style » fsBold**: Enable
    - **Style » fsItalic**: Enable
    - **GroupIndex**: `1` に変更
    - **Height**: `48` に変更
    - **Name**: `Panel_Override_Button` に変更
    - **Width**: `200` に変更

これらの変更を行うと、計器パネルフォームは図 7 のようになります。

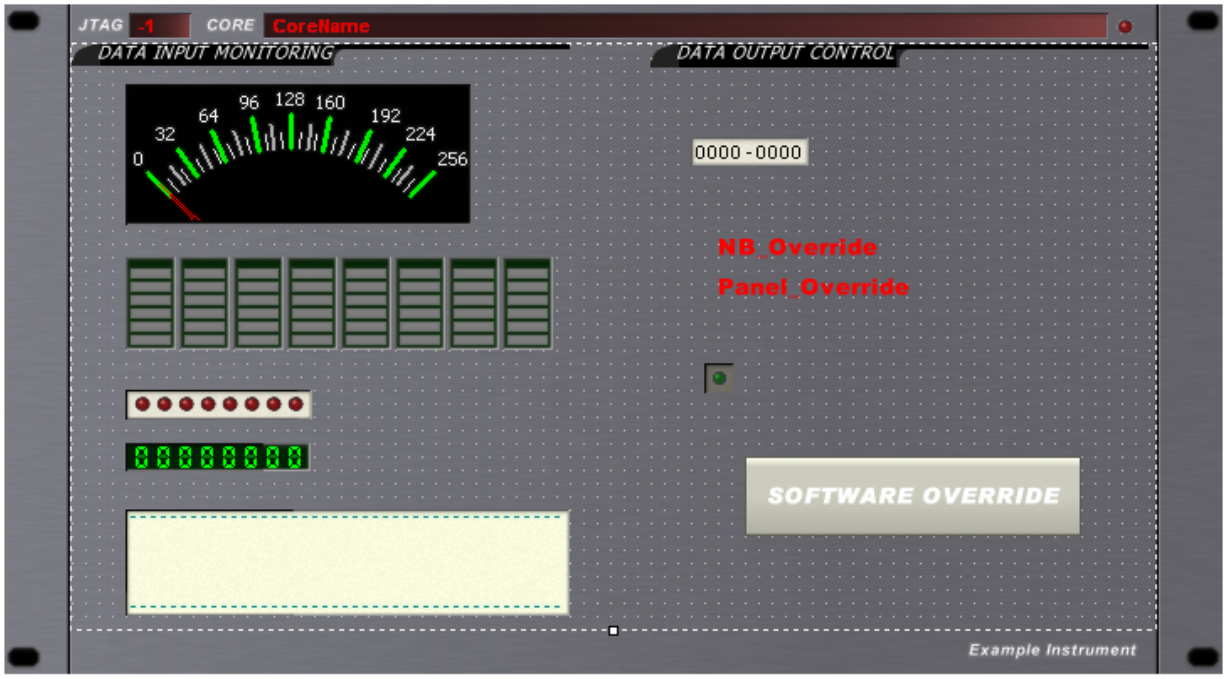


図 7. プロパティ変更後のパネルコントロール

説明用のラベルをいくつか追加して計器パネルの設計を終了します。High および Low しきい値用の Graph コントロールの横にラベルを追加し、パネル上のコントロールの動作を説明するさまざまなラベルを追加します。

16. 図 8 に示すキャプションと配置で InstrumentLabel コントロールを追加します。その際、**AutoSize** プロパティを無効にしてラベルのサイズを変更します。

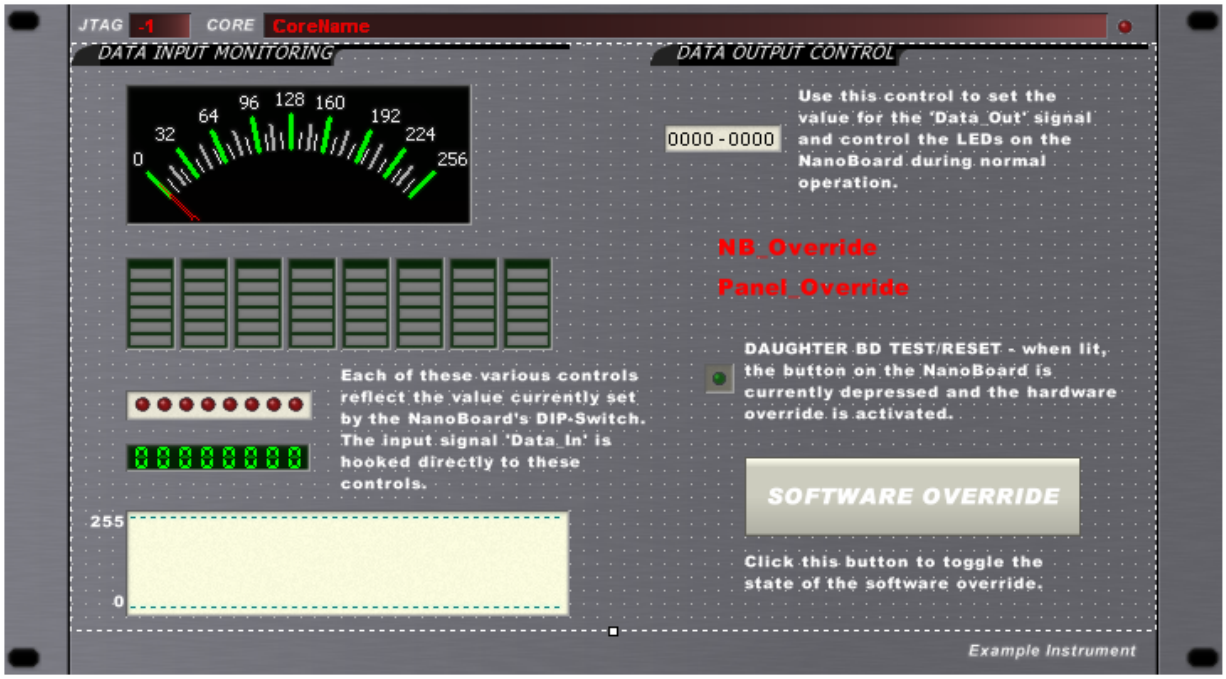


図 8. 説明用のラベルを追加して完成した計器パネルのデザイン

これでカスタムパネルのコントロールと外観に関する設計は終了です。仮想計器への入力信号を監視するために使用するコントロールでは、入力信号を該当するコントロールに直接割り当てました。しかし、データ出力では、Desktop NanoBoard 上のユーザ LED へのデータを制御する方法に関して若干の「知性」が必要です。このため、このデザインの要件を満たすためにスクリプトを使用します。

スクリプトの記述

スクリプト自体を説明する前に、仮想計器からのデータ出力に関して何をしたいのかを再確認します。

- 通常動作では、Numeric Panel コントロールで出力を制御する。
- Desktop NanoBoard 上の `DAUGHTER BD TEST/RESET` ボタンまたは計器パネルの `SOFTWARE OVERRIDE` ボタンが押されたら、Numeric Panel を優先設定する。値の出力はゼロとなり、どの優先設定が有効であるかをユーザに警告するため、対応するメッセージを計器パネルに表示する。

1. *Custom Instrument Configuration* ダイアログの下部にある **Code** タブをクリックします。スクリプトの内容を記述するための空白のページが表示されます (図 9)。

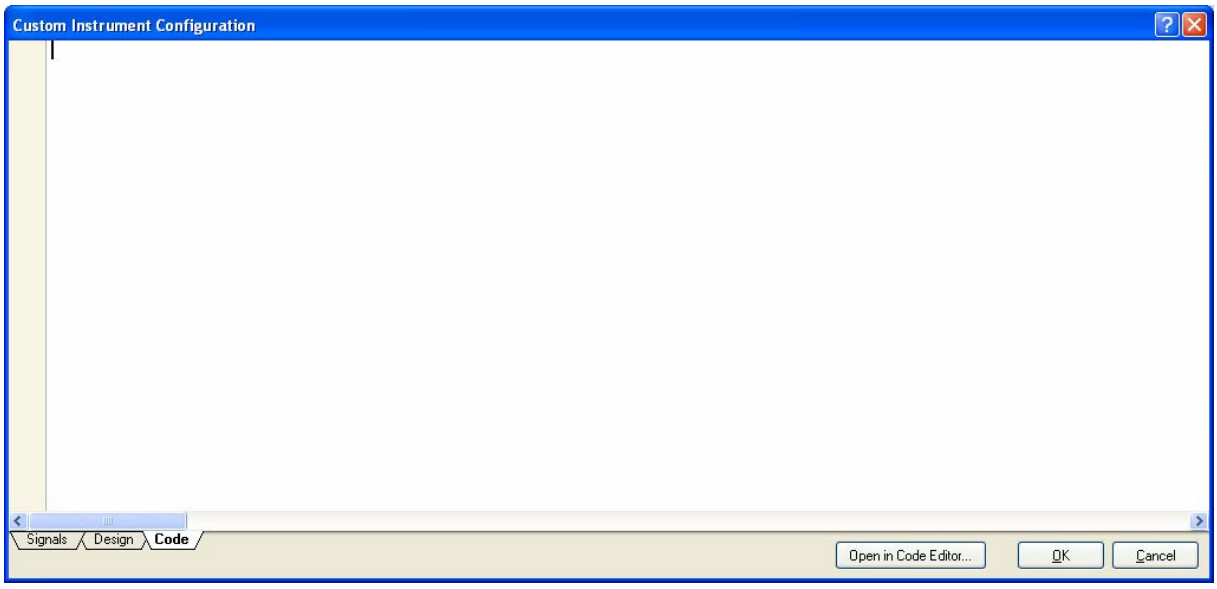


図 9. Code タブ – DelphiScript コードを作成するための領域

注記 :仮想計器用のスクリプトを記述する際にサポートされているのは DelphiScript だけです。Altium Designer 内のコード認識型のエディタでスクリプトを楽に記述するには、ダイアログの下部にある Open in Code Editor ボタンをクリックします。これにより、ダイアログの Code タブにはない、エディタで使用可能な構文ハイライトや関連するコードベースの機能を利用することができます。

Code Editor を使用してスクリプトを記述する場合、記述を終えたらファイルを保存するだけで、次回アクセスしたときにコードがダイアログに表示されます。

2. 以下のスクリプトを入力します。コメントはもちろん省略できますが、このチュートリアルの目的上、スクリプトを使用して何を実行するのかを明確にするため、常にコードにコメントを記述することをお勧めします。

```
//以下のプロシージャは、デザインしたパネルの領域内で読み込みイベントまたは書き込みイベントが発生したときに実行される
procedure TDesignedAreaPanel.DesignedAreaPanelReadWrite(Sender: TObject);
begin
//優先設定キャプションを初期化
     NB_Override.Caption := '';
     Panel_Override.Caption := '';
//SignalManager は、値を取得および設定するためのグローバル関数
     if  SignalManager.GetSignalByName('Override').Value = 1 then
//NB2DSK01 上のテストボタンが押されたら、関連するキャプションを表示し出力信号をゼロに設定
     begin
          NB_Override.Caption := 'Hardware Override Engaged!';
          SignalManager.GetSignalByName('Data_Out[7..0]').Value := 0;
     end
     else if Panel_Override_Button.Down then
//Software Override ボタンが押されたら、関連するキャプションを表示して出力信号をゼロに設定
     begin
          Panel_Override.Caption := 'Software Override Engaged!';
          SignalManager.GetSignalByName('Data_Out[7..0]').Value := 0;
     end
     else
//優先設定がない場合、/Output_Data というコントロールで現在定義されている値に出力を設定
     begin
          SignalManager.GetSignalByName('Data_Out[7..0]').Value := Output_Data.Value;
     end;
end;

//以下のプロシージャは、ユーザが Software Override ボタンをクリックしたときに実行される
procedure TDesignedAreaPanel.Panel_Override_ButtonClick(Sender: TObject);
begin
     Panel_Override_Button.Down := not Panel_Override_Button.Down;
end;
```

スクリプトを GUI に接続する

ここまでで、計器パネルを設計し、目的に合わせて出力を操作するための DelphiScript コードを作成しました。次に、パネル内で特定のイベントが発生したときに該当するプロシージャが呼び出されるように、スクリプトを GUI に接続する必要があります。

1. *Custom Instrument Configuration* ダイアログを開き、**Design** タブを選択します。
2. パネルの境界内のコントロール以外の場所をクリックし、DesignedAreaPanel オブジェクトを選択します。

3. **Events** パネルから、**OnReadWrite** イベントの右にあるフィールドの内部をクリックし、ドロップダウンを使用して `DesignedAreaPanelReadWrite` プロシージャを選択 (および割り当て) します。
4. SOFTWARE OVERRIDE Button コントロールをクリックして選択します。
5. **Events** パネルで、**OnClick** イベントの右にあるフィールドの内部をクリックし、ドロップダウンを使用して `Panel_Override_ButtonClick` プロシージャを選択 (および割り当て) します。

これで完了です。仮想計器の設定はすべて終わりました。次に、実際の動作を確認するため、物理 FPGA デバイスをデザインのターゲットとして指定しプログラムします。

物理 FPGA デバイスをターゲットとして指定する

ここまででデザインの作成段階は終了したため、使用する物理 FPGA デバイス (デザインのターゲットと、デザインを最終的にプログラムし実行するメディア) を指定する必要があります。このチュートリアルでは、Desktop NanoBoard NB2DSK01 に挿入する、3 コネクタ付きドータボード上の FPGA デバイスをターゲットに指定します。

プロジェクトを設定するための最も簡単な方法は、Altium Designer の自動設定機能を使用することです。この機能では、Desktop NanoBoard システム内に存在するハードウェアに基づいてプロジェクトが設定されます。それでは FPGA プロジェクトを設定します。

1. デザインのターゲットとなる、FPGA デバイスが搭載された 3 コネクタ付きドータボードが、NB2DSK01 マザーボードに挿入されていることを確認します。
2. 我々の単純な仮想計器デザインでは、プラグインペリフェラルボード上のリソースを使用しません。これらのボードは、マザーボードに接続したままでも、取り除いてもかまいません。
3. Desktop NanoBoard が USB (またはパラレル) 接続を経由して PC に接続され、電源が投入されていることを確認します。
4. プロジェクト (`Custom_Instrument_Design.PrjFpg`) に対して自動設定処理を実行します。

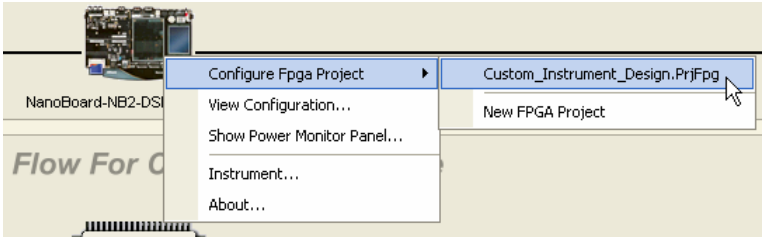


図 10. Devices ビューから直接自動設定

5. 設定が完了したら、すべてのソースファイルと親プロジェクトを保存します。

設定と制約の概念、およびデザインの移植性におけるそれらの役割の詳細は、記事『*AR0124 Design Portability, Configurations and Constraints*』を参照してください。

自動設定など、Desktop NanoBoard NB2DSK01 のコンストレインシステムの詳細は、『*AP0154 Understanding the Desktop NanoBoard NB2DSK01 Constraint System*』を参照してください。

デザインの処理

これでデザインをすべて作成し、ドータボード FPGA デバイスに対してターゲットを指定しましたので、デザインの処理に進みます。これは、ターゲットデバイスをプログラムするために使用する、必要な FPGA プログラミングファイルを最終的に作成することです。

1. Desktop NanoBoard が PC に接続され電源が投入されていることを確認します。
2. **Devices** ビューが Altium Designer のアクティブなビューになっていること、**Live** オプションが有効で **Connected** インジケータが緑色になっていることを確認します。
3. 物理デバイスに関連付けられているプロセスフローで、**Program FPGA** ボタン (フローの最終段階) をクリックします。デザインがコンパイルおよび合成され、ベンダツールが実行されます。最終的に FPGA プログラミングファイルが生成され、JTAG 経由でドータボード上の物理デバイスにダウンロードされます。

デザインがダウンロードされると、**Devices** ビューの物理デバイスの下にあるテキストが Reset から Programmed に変わります。ハードウェア側では、デザインが物理デバイスにロードされたことを示すため、ドータボードの Program LED が緑点灯します。

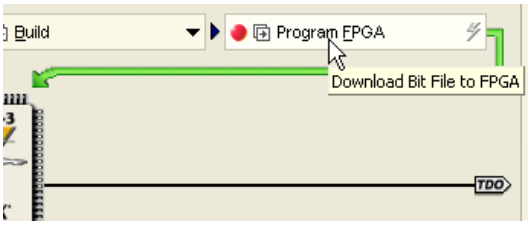


図 11. プログラミング処理の開始

プロセスフローと **Devices** ビューの詳細は、『*AP0103 Processing the Captured FPGA Design*』を参照してください。

カスタマイズされた計器パネルへのアクセス

これでデザインが物理 FPGA デバイスにプログラムされたため、計器のために作成したカスタマイズされた GUI にリアルタイムでアクセスすることができます。この手順はこのチュートリアルで最もおもしろいところです。これまで努力した結果を試してみることができるのです。

1. **Devices** ビューで、計器のためのカスタマイズされたアイコンが Soft Devices JTAG チェーンに表示されていることを確認します。これは、デザインが物理 FPGA デバイスにプログラムされると表示されます。計器のステータスが `Running` と表示されます。
2. 仮想計器のアイコンをダブルクリックします。関連付けられている計器パネル (作成したカスタム GUI) が **Instrument Rack – Soft Devices** パネルに表示されます。

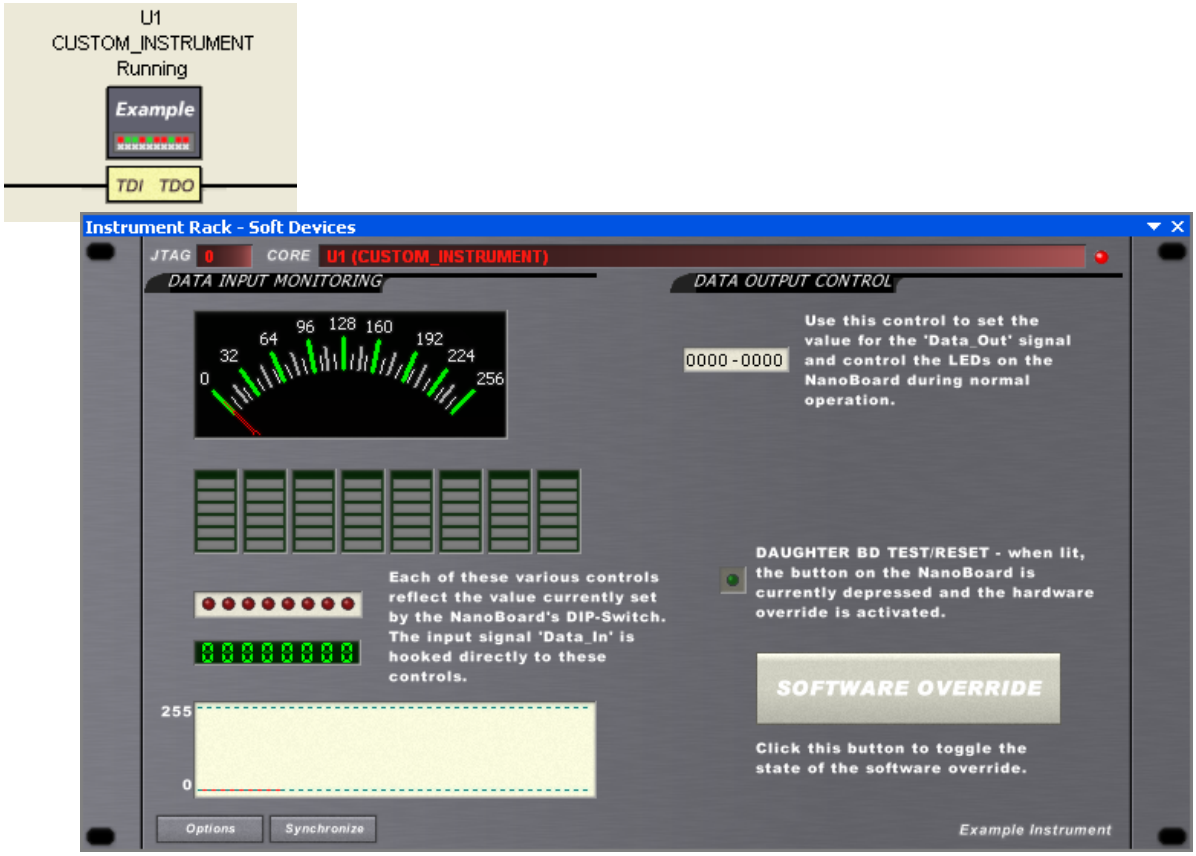


図 12. カスタマイズされた GUI へのアクセス

3. Desktop NanoBoard の DIP スイッチに関連付けられた各スイッチをオン/オフし、パネルに追加および定義した以下の各監視コントロールに同じ値が反映されることを確認します。
   - Gauge
   - Progress Bar
   - LED Panel
   - LED Digit
   - Graph
4. Numeric Panel コントロールを使用して、各ビットの ON と OFF を切り替えます。それに応じて Desktop NanoBoard 上の対応する LED が点灯または消灯することを確認します。
5. Numeric Panel コントロールの下位の桁を `1111` に設定し、NanoBoard 上のユーザ LED 3..0 を点灯させます。

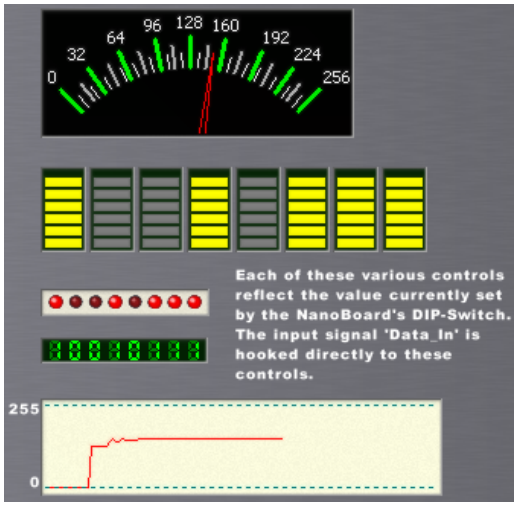


図 13. DIP スイッチからの入力値の監視

6. NanoBoard 上の DAUGHTER BD TEST/RESET ボタンを一瞬押します。以下のことを確認します。
   - ユーザ LED 3..0 が消灯すること
   - 関連付けられているハードウェア優先設定メッセージが計器パネルに表示されること
   - DAUGHTER BD TEST/RESET ボタンの状態を監視するために使用する計器パネル上の LED が点灯すること

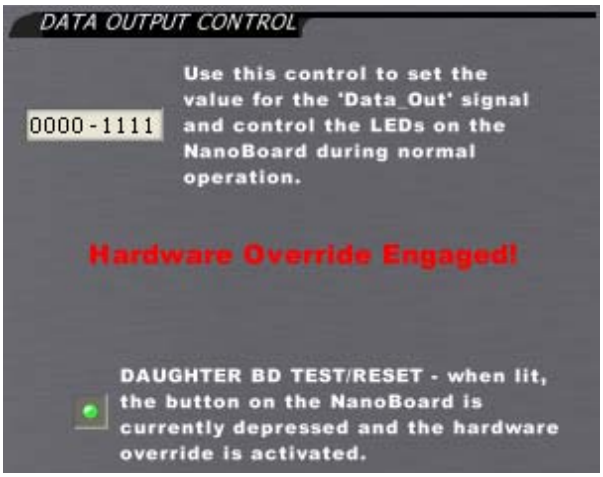


図14. ハードウェア優先設定の実行

図15. ソフトウェア優先設定の実行

7. 計器パネルの **SOFTWARE OVERRIDE** ボタンを押します。以下のことを確認します。
   - ユーザ LED 3..0 が消灯すること
   - 関連付けられているソフトウェア優先設定メッセージが計器パネルに表示されること
8. 再度 **SOFTWARE OVERRIDE** ボタンをクリックして優先設定を無効にし、ユーザ LEDS 3..0 が点灯することを確認します。

これでこのチュートリアルは終了です。ここでは、仮想計器で提供される機能と能力のごく一部を説明したに過ぎません。使用方法の基礎が理解できたので、この完全にカスタマイズ可能な計器によってデジタル IO の監視と制御で必要な機能が提供されるということを確信しながら、より複雑なデザインを使用することができます。

更新履歴

日付	バージョン番号	変更内容
2008 年 5 月 17 日	1.0	初リリース

ソフトウェア、ハードウェア、文書、および関連資料