

はじめに

MAX[®]II デバイスは、動作中のデバイスのプログラミングを可能にするリアルタイム ISP (In-System Programmability) 機能をサポートします。この機能を使用すると、システムの動作に影響を与えることなく、いつでも MAX II デバイスのインフィールドでのアップデートを実行できます。

また、MAX II デバイスを使用すれば、既存のデザインを新しいデザインに置き換えて、動作開始のタイミングを制御することができます。既存のデザインを新しいデザインに置き換えるのは、デバイスにパワー・サイクル（パワー・ダウンしてから再びパワー・アップする）を加えたときか、またはリアルタイム ISP の完了時に、特定の ISP 命令を実行して SRAM ダウンロード処理を開始するときのいずれかです。SRAM ダウンロード処理を実行すると、すべてのデバイス・レジスタの以前に保存された値がクリアされ、処理中はデフォルトにより I/O ピンがトライ・ステートになります。

このアプリケーション・ノートでは、デザイン例を示しながら、ISP 命令を実行して SRAM ダウンロードを行う方法、この処理の実行中に I/O ピンをフリーズさせる方法、新しいデザインに置き換える前にレジスタ・データの状態を復元する方法を説明します。



MAX II リアルタイム ISP 機能について詳しくは、「MAX II Device ハンドブック」の「MAX II デバイスのリアルタイム ISP および ISP Clamp」の章を参照してください。

SRAM ダウンロードの強制

パワー・サイクルなしで SRAM ダウンロードを強制するには、デバイスをプログラミング・モード (ISP モードまたは ISP Clamp モード) にしてから、すぐにプログラミング・モードを終了してユーザ・モードに戻す必要があります。デバイスがプログラミング・モードを終了すると (非リアルタイム ISP)、SRAM ダウンロードが開始されます。リアルタイム ISP のアップデートを実行する場合、新しいデザインは JTAG (Joint Test Action Group) コマンドが ISP エントリまたは ISP Clamp エントリとエグジットを実行した後にのみ SRAM にロードされます。SRAM ダウンロードには、デバイスの集積度によって t_{CONFIG} パワーアップ・タイミング仕様で規定されるのと同様 200 ~ 450 μs かかります。「MAX II デバイス・ハンドブック」の「DC & スイッチング特性」の章を参照)。

デバイスをプログラミング・モードにするか、またはユーザ・モードに戻すには、JTAG インタフェースを使用して適切な命令でシフトさせる必要があります。また、JTAG インタフェースを使用して、TAP (Test Access Port) コントローラ・ステート・マシンを制御することもできます。JTAG インタフェースでは、デバイスとの通信に JTAG ピン (TCK、TMS、TDI、および TDO) を使用する必要があります。SRAM ダウンロードを強制する最も簡単な方法は、必要な短い命令シーケンスを実行する STAPL (Jam Standard Test and Programming Language) コマンドの特別なセットを使用することです。このアプリケーション・ノートでは、Jam STAPL ファイルを使用してこれを行う方法を説明します。



MAX II JTAG インタフェースおよび TAP コントローラ・ステート・マシンについて詳しくは、「MAX II デバイス・ハンドブック」の「MAX II デバイスの IEEE 1149.1(JTAG) バウンダリ・スキャン・テスト」の章を参照してください。

SRAM ダウンロード時の I/O クランピング

MAX II ISP Clamp 機能により、デバイスが ISP モードのときに、I/O ピンを特定の状態にクランプすることができます。ピンを High または Low、あるいはトライ・ステートにクランプするか、またはデバイスが ISP に入る前にピンの状態をサンプリングし、デバイスが ISP モードになったときにそれらのピンをサンプリングした状態にクランプすることができます。

ISP Clamp 機能を使用して、デバイスが ISP モードに入るときの I/O ピンの状態をサンプリングおよびクランプすれば、ユーザ・モードと ISP 間でグリッチのない移行が可能です。SRAM ダウンロード強制手順の実行時に、これと同じプロセスを使用して I/O の状態を制御することができます。ISP_ENABLE_CLAMP 命令とその後の ISP_DISABLE 命令は、I/O ピンをバウンダリ・スキャン・レジスタの出力レジスタの値にクランプします。これと同じシーケンスによって、SRAM ダウンロードまたはリフレッシュが実行されます。SRAM ダウンロード完了時に、入力ピンが機能し始めても出力はクランプされたままです。この手順を使用して、SRAM ダウンロード処理中に I/O の状態を制御するだけでなく、デバイス内でレジスタ値を非パワーアップ状態に設定する信号を作成できます。以下のデザイン例では、この手順について説明します。



コンフィギュレーション中、出力ピンはデフォルトでは以下の設定になります。最大ドライブ電流、低速スルー・レート、およびウィーク・プルアップ抵抗がイネーブル。これらの設定は、ISP_ENABLE_CLAMP 命令を発行してから、ISP_DISABLE 命令を発行した t_{CONFIG} 時間後まで有効です。デザインで出力ピンに対して最小ドライブ電流および / または高速スルー・レートを使用する場合、SRAM ダウンロード処理中は出力ドライブ特性が異なります。また、SRAM ダウンロード処理中、トライ・ステート・ピンはウィーク・プルアップ抵抗でトライ・ステートになることにも注意してください。

レジスタ・データのキャプチャと保持

デフォルトでは、SRAM ダウンロード処理によってデバイス内のすべてのレジスタがクリアされます。レジスタ・データを保持する必要がある場合、SRAM ダウンロードが開始される直前に保存しておいて、ダウンロード完了後にリロードすることができます。

MAX II ユーザ・フラッシュ・メモリ (UFM) を使用すると、SRAM ダウンロードが開始される前にレジスタ・データを保存することができます。SRAM ダウンロード処理では、SRAM ダウンロード中に UFM はクリアされません。SRAM ダウンロードが終わると、UFM からデータを読み戻してレジスタをリロードすることができます。

UFM ストレージの代わりに、MAX II ISP Clamp 機能を使用して、出力レジスタのデータを保持することも可能です。ISP Clamp は、出力ピンを解放する前に入力ピンを解放して、デザインの動作をイネーブルします。入力ピンを使用して出力ピンのクランプ状態をサンプリングし、出力ピンを解放する前にレジスタをリロードすることができます。このアプリケーション・ノートでは、レジスタ・データを保持するための 2 つの方法を説明します。

UFM ストレージによるデザインのアップデート例

この SRAM ダウンロード例は、I/O ピンの状態を制御し UFM ブロックでレジスタ・データ保持する方法を説明しています。デザイン例は、以下の 2 つの部分で構成されています。

- Jam ファイル
- Quartus® II デザイン例

ユーザ独自の要求条件に合わせて、Jam ファイルとデザインを変更するか、またはこれをベースにして新しいデザインを作成することができます。

Jam ファイル

Jam ファイルには、デバイスにシフトされてデバイスを以下の特定のモードに設定する ISP および JTAG 命令が収められています。ユーザ・モード、ISP Clamp モード、またはリアルタイム ISP モードこのファイルは TAP コントローラ・ステート・マシンも制御します。

Quartus II ソフトウェアのプログラマは、このカスタム Jam ファイルを実行できません。Jam ファイルを実行するには、JTAG インタフェースを通してデバイスと通信するために、Jam STAPL Player が必要です。Jam STAPL Player は、www.altera.com からダウンロードできます。

Jam ファイルを実行するには、以下の手順を実行します。

- Jam ファイルのディレクトリのコマンドライン・プロンプトで、次のように入力します。:

```
jam -adownload download.jam
```

ここで、jam は Jam Player 実行モジュール、download は Jam STAPL アクション、download.jam は、命令を含む Jam ファイルです。

表 1 では、Jam ファイルの ISP および JTAG 命令を説明しています。

表 1. ISP および JTAG 命令		
命令名	命令コード	説明
SAMPLE_PRELOAD	00 0000 0101	デバイスが ISP Clamp モードに入ったときに、ピンをそれぞれの状態にクランプできるように I/O ピンの現在の状態をキャプチャします。
ISP_ENABLE_CLAMP	10 0011 0011	デバイスを ISP Clamp モードに設定します。
ISP_DISABLE	10 0000 0001	デバイスに ISP または ISP Clamp を終了させ SRAM ダウンロードを開始させます。
RT_ISP_ENABLE	01 1001 1001	デバイスをリアルタイム ISP モードに設定し、UFM リアルタイム ISP ビジー (rtpbusy) 信号をレジスタ・データを UFM に保存するためのトリガ信号としてアサートします。この信号は、デザインがデータをレジスタにリロードするためのインジケータとしても機能します。
RT_ISP_DISABLE	01 0110 0110	デバイスをリアルタイム ISP モードからユーザ・モードに復帰させます。

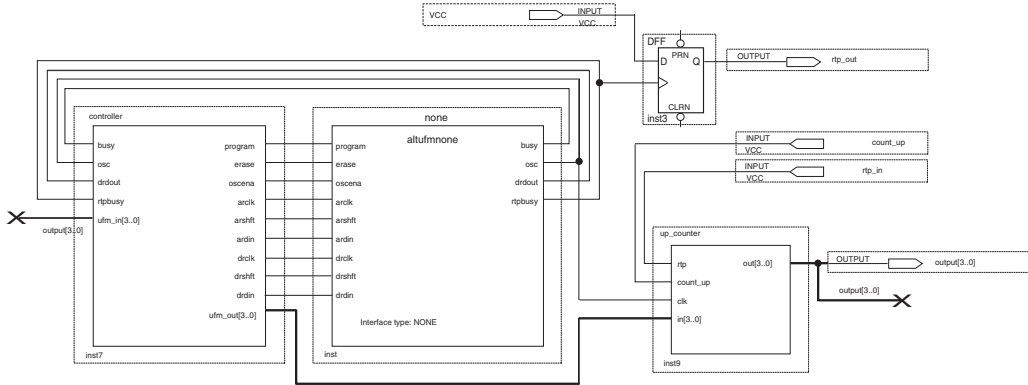
Quartus II デザイン例

Quartus II デザイン例は、以下の複数のモジュールで構成されています。

- コントローラ
- altufm_none メガファンクション
- 4 ビット・アップ・カウンタ
- D フリップ・フロップ

これらモジュールは、新しいアップデート・デザインおよび既存のデザインの一部です。図 1 に、Quartus II デザインのブロック図を示します。

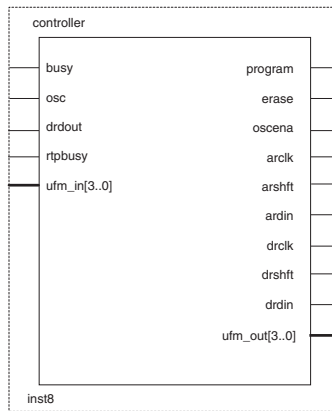
図 1. デザイン例



コントローラ

コントローラは、UFM ブロックとの間での読み出し、書き込み、および消去のすべてのインタフェース動作を扱います。コントローラが rtpbusy 信号を検出すると、SRAM ダウンロードを開始する前に、4 ビット・アップ・カウンタからのデータをキャプチャして UFM に保存します。4 ビット・アップ・カウンタは、レジスタ値がキャプチャおよび復元される同期ロジックの一例です。コントローラは、SRAM ダウンロード後にデータを UFM からカウンタに自動的に取り戻します。図 2 にコントローラ・モジュールを示します。

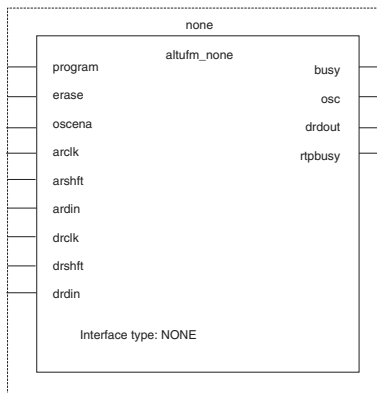
図 2. コントローラ・モジュール



altufm_none メガファンクション

altufm_none メガファンクション (図 3) は、MAX II UFM をインスタンス化し、ユーザ・ロジックおよび UFM ブロックをインタフェースします。

図 3. altufm_none メガファンクション



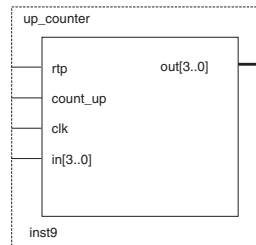
altufm_none メガファンクションについて詳しくは、「MAX II デバイス・ハンドブック」の「MAX II デバイスでのユーザ・フラッシュ・メモリの使用」の章を参照してください。

4 ビット・アップ・カウンタ

このデザインは 4 ビット・アップ・カウンタを使用して、カウンタの機能を中断したりデータを失うことなく、SRAM ダウンロード処理が実行される過程を示します。実際のデザインでは、SRAM ダウンロード処理中にデータを保持する必要がある他のレジスタを使用する場合もあります。

デフォルトでは、カウンタはパワーアップ時にすべて 0 になります。カウンタの値は、count_up ポートの立ち上がりエッジごとに 1 ずつインクリメントされます。rtp 入力ピンは、外部オフチップの D フリップ・フロップの出力ピンに接続されます。D フリップ・フロップの出力ピンは、カウンタが in[3..0] ポートを通して、UFM からデータをロードしなければならないタイミングを示します。出力信号を D フリップ・フロップから内部でカウンタの rtp ポートに配線してはなりません。図 4 に 4 ビット・アップ・カウンタを示します。

図 4. 4 ビット・アップ・カウンタ




D フリップ・フロップ

このデザインは、D フリップ・フロップを使用して `altufm_none` メガファンクションから `rtpbusy` 信号をキャプチャします。入力ピンを D フリップ・フロップの D 入力にドライブします。このピンは外部でデバイスの V_{CCIO} 電源に接続されています。D フリップ・フロップ・クロック入力は、内部で `altufm_none` メガファンクションの `rtpbusy` 信号に接続されます。フリップ・フロップの Q 出力は、外部で 4 ビット・アップカウンタの `rtp` ポートの入力ピンに接続される出力ピンにドライブされます。この信号は、UFM からのデータをカウンタにロードするタイミングを示します。

`rtp` 信号は、ノーマル・パワー・サイクルが発生した (ロジック 0) か、またはリアルタイム ISP アップデート発生した (ロジック 1) かを示すインジケータとして機能します。これにより、レジスタをパワーアップ・リセット状態にするか、またはプリセット / 非パワーアップ状態にするかを制御することができます。

デザインのアップデート手順

この項では、カウンタの出力ピンが解放されるまでの Jam ファイルの SRAM ダウンロード手順について説明します。

 実際のリアルタイム ISP は、SRAM ダウンロード処理を開始するより前に終了させる必要があります。カウンタ出力ピンには、処理全体を通してグリッチが発生しません。

1. デバイスは、トリガ信号を検出してレジスタ・データを UFM に保存します。

Jam ファイルは、`RT_ISP_ENABLE` 命令を発行して、デバイスをリアルタイム ISP モードにします。`altufm_none` メガファンクションからの `rtpbusy` 信号によって、コントローラは UFM セクタの 1 つを消去してから、レジスタのデータをその UFM セクタのアドレスの 1 つに保存します。コントローラ・モジュールの UFM アドレスを指定することができます。このデザインでは、そのアドレスの UFM セクタを消去しているだけです。

また、`rtppbusy` 信号によって、D フリップ・フロップの Q 出力からの出力ピンが High になります。デバイスは、SRAM ダウンロード処理が終了するまでこの出力ピンを High にクランプし、ついで他の出力ピンと共に解放します。この信号は 4 ビット・アップ・カウンタへのインジケータとして機能します。カウンタは SRAM ダウンロードの後、レジスタをリロードする前にコントローラ・モジュールからデータを読み出す必要があります。

2. デバイスはリアルタイム ISP モードを終了します。

Jam ファイルは `RT_ISP_DISABLE` 命令を発行します。

3. デバイスは I/O ピンの状態をサンプリングします。

Jam ファイルは、`SAMPLE_PRELOAD` 命令を発行し、I/O ピンの状態をキャプチャします (ピンの状態のキャプチャ中およびキャプチャ後にピンの状態が変化しないことを確認してください)。

4. デバイスは ISP Clamp モードに入ります。

Jam ファイルが `ISP_ENABLE_CLAMP` 命令を発行すると、デバイスは ISP Clamp モードに入ります。ISP Clamp モードに入ると、`SAMPLE_PRELOAD` 命令の実行中に I/O ピンがキャプチャした状態にクランプされます。

5. デバイスは ISP Clamp モードを終了します。

Jam ファイルが `ISP_DISABLE` 命令を発行すると、デバイスは ISP Clamp モードを終了します。ISP Clamp モードを終了すると、SRAM ダウンロードが実行されデバイス内のすべてのレジスタがクリアされます。新しいデザインは、SRAM ダウンロード後に機能し始めますが、すべての出力ピンはクランプされたままの状態です。コントローラ・モジュールは、UFM からデータを読み戻します。カウンタは D フリップ・フロップからの Q 出力信号を検出し、コントローラが UFM から読み出したデータをレジスタにロードします。

6. デバイスは出力ピンを解放します。

Jam ファイルが TAP コントローラ・ステート・マシンをリセット状態にすると、デバイスはすべての出力ピンを解放します。これはグリッチのない遷移で、SRAM ダウンロード前の値からカウンタ動作を再開できます。

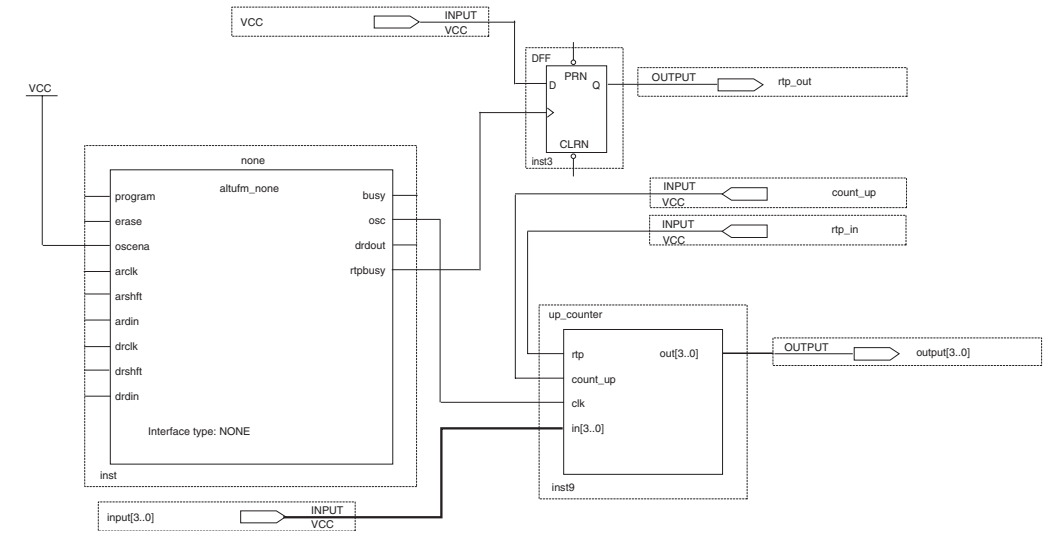
UFM 非使用 デザイン・ アップデート の代替手段

SRAM ダウンロード処理の前に UFM を使用してレジスタ・データを保存したくない場合は、既存のデザインを簡単に変更して代替させることができます。ここでは、2 つの UFM 非使用デザイン・アップデートの代替手段について説明します。

ISP Clampおよび入力ピンを使用したレジスタ・データの保持

UFM を使用してレジスタ・データを保存する代わりに、MAX II ISP Clamp 機能を使用してレジスタ・データを保持することができます。デバイスが ISP Clamp モードを終了すると、直ちにロジック・アレイのユーザ・デザインおよび入力ピンが機能し始めます。ただし、デバイスは TAP コントローラ・ステート・マシンがリセット状態になるまで出力ピンをクランプし続けます。これにより、クランプされた出力ピンをデバイスの入力ピンにドライブ・バックすることができます。要するに、クランプされた出力ピンを、それらの出力ピンをドライブするレジスタ・データのストレージとして使用しているわけです。図 5 にこのデザインの回路図を示します。

図 5. UFM を使用しないでレジスタの値を保持するためのデザイン



このデザインでは UFM への書き込みや UFM からの読み出しが不要なので、コントローラ・モジュールを使用しないため、LE リソースの消費量が少なくて済みます。ただし、`altufm_none` メガファンクションをインスタンス化して、`rtpbusy` 信号や `osc` クロックにアクセスする必要があります。`oscena` ポートを内部で `VCC` に接続して、常に `osc` クロックをイネーブルします。

ページ 5 の図 1 に示した以前のデザインのように、D フリップ・フロップ出力ピンを 4 ビット・アップカウンタ・モジュールの `rtp` ポートの入力ピンに接続します。D フリップ・フロップ入力ピンを外部でデバイスの `VCCIO` 電源に接続します。クロック入力は、内部で `altufm_none` メガファンクションからの `rtpbusy` 信号に接続されます。

このデザインではさらに4本の入力ピンを使用しています。カウンタ in[3..0] ポートを4本の入力ピンに接続し、これらのピンを外部で out[3..0] ポートの出力ピンに接続します。出力信号を内部で in[3..0] ポートに戻すように配線しないでください。多数のレジスタ用データを保存する必要がある場合は、多くの I/O ピンを使用するため、このデザインは適していないことがあります。

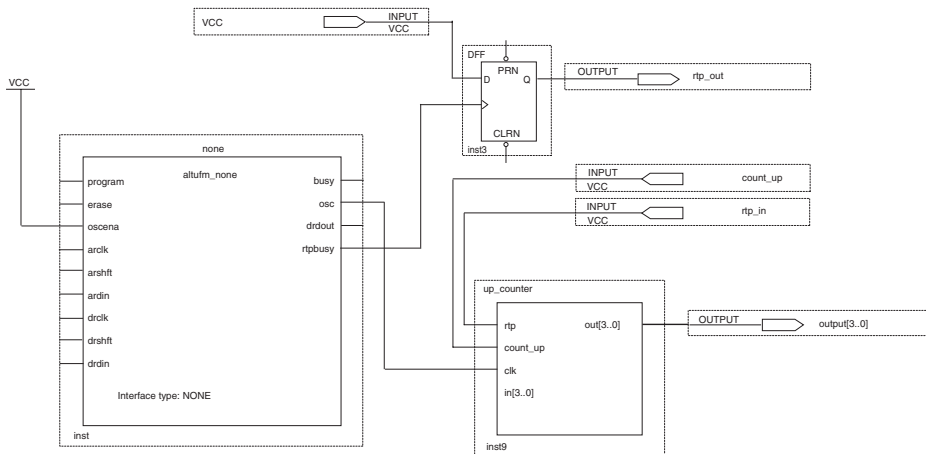


このデザインの Jam ファイルを変更する必要はありません。

SRAM ダウンロード後のレジスタへの指定値のロード

図 6 に示すデザインは、SRAM ダウンロードが終了するたびに、レジスタにプリセット値をロードします。カウンタ・モジュールを実装するためにデザインを変更する必要はほとんどありません。図 6 にこのデザインの回路図を示します。

図 6. レジスタに特定の値をロードするためのデザイン




プリセット値はカウンタ・モジュールで指定できます。`.v` ファイルに値を指定してデザインを再コンパイルすれば、カウンタ・モジュールの Verilog コードを変更することができます。カウンタには入力ポート in[3..0] は必要ありません。

カウンタ・モジュール用 Verilog ファイルの以下の部分に、プリセット値を入力することができます。

```
always @ (posedge clk)
begin
  if (count <= 999)
  begin
    count = count + 1;
```

```
if (count == 1000 && rtp == 0)
    out = 4'b0; //this is the default
                //register data when
                //powered up
else if (count == 1000 && rtp == 1)
    out = 4'b1111; //this is the preset data
                //to be loaded into the
                //registers after each
                //SRAM download
end
```

 このデザイン用の Jam ファイルを変更する必要はありません。

まとめ

このデザイン例は、リアルタイム ISP 後にパワー・サイクルなしで SRAM ダウンロードを実行するためのソリューションを提供します。このデザインはまた、SRAM ダウンロード処理を実行した後で、UFM または入力ピンを使用してデータを保持することにより、デバイス内のレジスタ・データの保持を可能にするソリューションも提供します。ISP Clamp 機能と併せて、MAX II デバイスではグリッチのない SRAM ダウンロード処理も提供されます。



〒 163-1332
東京都新宿区西新宿 6-5-1
新宿アイランドタワー 32F 私書箱 1594 号
TEL.03-3340-9480 FAX.03-3340-9487
<http://www.altera.co.jp>
E-mail:japan@altera.com

本社 Altera Corporation
101 Innovation Drive, San Jose, CA 95134
USA
TEL : (408) 544-7000
<http://www.altera.com>

Copyright © 2006 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.



I.S. EN ISO 9001

この資料は英語版を翻訳したもので、内容に相違が生じる場合には原文を優先します。こちらの日本語版は参考用としてご利用ください。設計の際には、最新の英語版で内容をご確認ください。