

はじめに

メガファンクションは、パラメータ化可能で、Altera® デバイスのアーキテクチャに対して最適化されたベンダ固有の IP (Intellectual Property) ブロックです。アルテラは、より効果的なロジックの合成およびデバイスの実装を提供するメガファンクションのライブラリ (LPM (Library of Parameterized Modules) ファンクションおよび他のパラメータ化されたファンクションを含む) を提供します。

- 特定のメガファンクションについて詳しくは、「[資料：ユーザーガイド](#)」で該当するメガファンクションのユーザーガイドを参照してください。
- 使用可能なメガファンクションおよび LPM の最新リストは、Quartus® II Help のメガファンクションまたは LPM のセクションを参照してください。

概要

MegaWizard™ Plug-In Manager を使用することにより、デザイン・ファイルでインスタンス化されるカスタム・メガファンクション・バリエーションを含むデザイン・ファイルを作成または変更することができます。これらのカスタム・メガファンクション・バリエーションは、アルテラが提供するメガファンクションに基づいています。MegaWizard Plug-In Manager を使用して、Quartus II 開発ソフトウェア、EDA デザイン・エントリおよび合成ツールでデザインに使用するためのアルテラ・メガファンクション、LPM ファンクション、および IP ファンクションを作成できます。

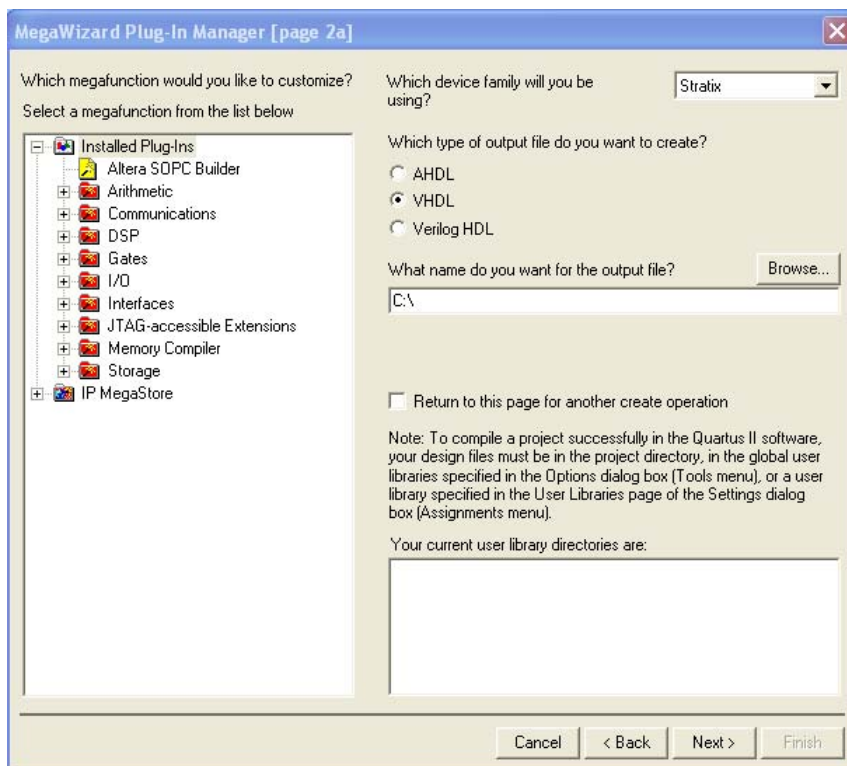
MegaWizard Plug-In Manager は、以下のいずれかの方法で開始します。

- Tools メニューの **MegaWizard Plug-In Manager** をクリックします。
- Block Editor を実行しているときは、Edit メニューの **Insert Symbol Block** をクリックするか、または Block Editor を右クリックして、**Insert** をポイントし、**Symbol as Block** をクリックします。Symbol ウィンドウの **MegaWizard Plug-In Manager** をクリックします。
- コマンド・プロンプトで次のコマンドを入力して、MegaWizard Plug-In Manager のスタンダードアロン・バージョンを起動します：

```
qmegawiz ←
```
- Windows で、スタート・メニューの **All Programs** を選択し、フォルダ内の Quartus II を選択します。そして、**Quartus II MegaWizard Plug-In Manager** をクリックします。

図 1 に、MegaWizard Plug-In Manager における使用可能なメガファンクションのカテゴリを示します。

図 1. MegaWizard Plug-In Manager における使用可能なメガファンクションのカテゴリ



サポートされるデバイス・ファミリ

以下のいずれかの方法を介して、特定のメガファンクションのサポートされるデバイス・ファミリを確認することができます。

- Quartus II Help ファイル — デバイス・ファミリ・サポートに関する最新の情報は、Quartus II Help ファイルの各メガファンクションのために個別に述べられています。
- MegaWizard Plug-In Manager:
 - MegaWizard Plug-In Manager のページ 2a で、**Which megafunction would you like to customize?** のリストから、メガファンクションを選択します。
そのメガファンクションのサポートされているデバイス・ファミリは、MegaWizard Plug-In Manager の同様なページの **Which device family will you be using?** のプルダウン・リストに自動的に表示されます。
 - MegaWizard Plug-In Manager のページ 2a で、**Which device family will you be using?** のプルダウン・リストから、デバイス・ファミリを選択します。
そのメガファンクションのサポートされているデバイス・ファミリは、MegaWizard Plug-In Manager の同様なページの **Which megafunction would you like to customize?** のプルダウン・リストに自動的に表示されます。サポートされていないデバイス・ファミリは、グレーに表示されています。

デザイン例 : Quartus II MegaWizard Plug-In Manager での新しいメガファンクションの作成

この例は、ALTCLKCTRL メガファンクションの新しいインスタンスを作成して、トップレベル・デザインの回路図の ALTPLL インスタンスにインスタンスを接続させるように、MegaWizard Plug-In Manager を使用します。アルテラのウェブサイトの「[資料 : ユーザーガイド](#)」セクションからこのデザインの例のプロジェクトアーカイブをダウンロードできます。

ALTCLKCTRL メガファンクションは、クロック信号を選択するクロック・バッファとして動作するクロック・コントロール・ブロックのメガファンクションです。4つのクロック信号があります。それらの信号はグローバル・クロック、リージョナル・クロック、デュアル・リージョナル・クロック、および外部クロック・パスです。

図2に、4つの入力クロック・ポート、出力イネーブル・ポート、2ビットのセレクトター・ポート、および出力ポートを装備する ALTCLKCTRL メガファンクションのブロック図を示します。

図 2. ALTCLKCTRL メガファンクションのブロック図

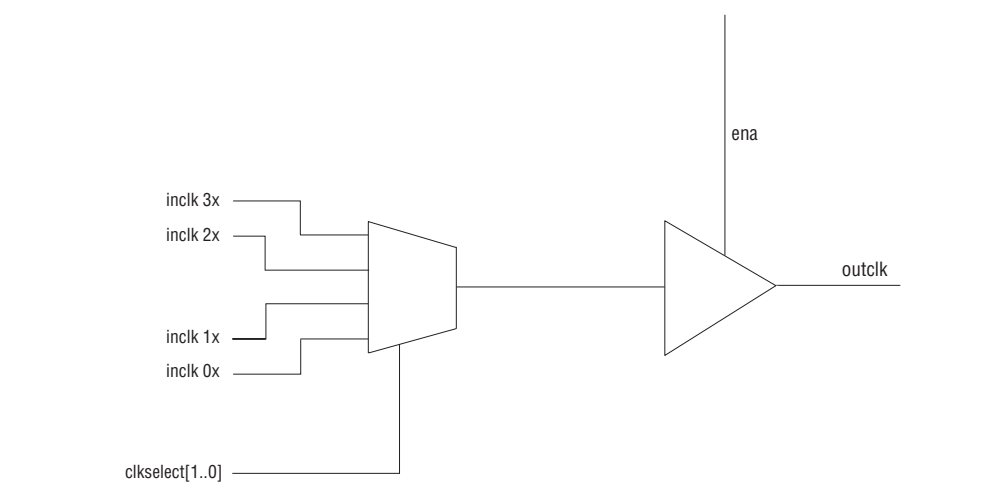



表1に、2ビットのセレクトター・ポートによるクロック・ソースの選択を示します。

表 1. クロック・ソースの選択

バイナリ値	シグナルの選択
<code>clkselect [1..0] = 00</code>	<code>inclk 0x</code>
<code>clkselect [1..0] = 01</code>	<code>inclk 1x</code>
<code>clkselect [1..0] = 10</code>	<code>inclk 2x</code>
<code>clkselect [1..0] = 11</code>	<code>inclk 3x</code>

 ALTCLKCTRL メガファンクションについて詳しくは、Quartus II Help の「[ALTCLKCTRL Megafunction User Guide](#)」を参照してください。

この例では、ALTPLL インスタンスは、PLL (Phase-Locked Loop) が 50 MHz および 200 MHz の 2 つの出力クロック信号を生成する入力クロックの 100MHz を使用して実装されています。ALTCLKCTRL インスタンスは、4 つのクロック入力ポート (ALTPLL の出力ポートに接続される 2 つの入力ポート、および専用クロック・ピンに接続される 2 つのポート) が含まれているクロック・セレクターを実装しています。この例では、Stratix III® の EP3SE50F484C2 デバイスを使用しています。

以下のステップでは、プロジェクトのアーカイブを抽出と復元するプロセス、またデザイン全体を完成するためのプロセスを解説します。

1. **mo_altclkctrl_DesignExample.zip** ファイルを解凍して Quartus II プロジェクトのアーカイブ (**mo_altclkctrl_ex.qar**) を抽出します。
2. Quartus II ソフトウェアで、**mo_altclkctrl_ex.qar** ファイルを開いて、そのアーカイブ・ファイルを作業ディレクトリに保存します。
3. トップレベル・デザインの回路図 (**mo_altclkctrl_ex.bdf**) を開きます。
4. 回路図の空白エリアをダブル・クリックします。
5. Symbol ウィンドウ内で **MegaWizard Plug-In Manager** ボタンをクリックします。MegaWizard Plug-In Manager のページ 1 が表示されます。
6. **Create a new custom megafunction variation** オプションを選択します。
7. **Next** をクリックします。MegaWizard Plug-In Manager のページ 2a が表示されます。
8. MegaWizard Plug-In Manager のページ 2a 以降で、[表 2](#) に示されているコンフィギュレーション設定を選択、または確認します。**Next** をクリックして、次のページを表示します。



MegaWizard Plug-In Manager で生成されるファイルの説明について詳しくは、[1-7 ページの「MegaWizard Plug-In Manager 生成されるファイル」](#)を参照してください。

表 2. MegaWizard Plug-In Manager ページのオプションと説明 (その 1)

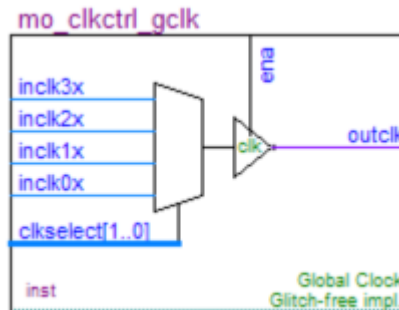
MegaWizard Plug-In Manager ページ	コンフィギュレーションの設定	値
2a	Which megafunction would you like to customize?	ALTCLKCTRL
	Which device family will you be using?	Stratix III
	Which type of output file do you want to create?	AHDL
	What name do you want for the output file?	'mo_clkctrl_gclk'
3	How do you want to use the altclkctrl?	For global clock
	How many clock inputs would you like?	4
	Create 'ena' port to enable or disable the clock network driven by this buffer	Selected
	How do you want to register the 'ena' port?	Falling edge of input clock
	Ensure glitch-free switchover implementation	Selected
4	Generate netlist	Selected

表 2. MegaWizard Plug-In Manager ページのオプションと説明 (その 2)

MegaWizard Plug-In Manager ページ	コンフィギュレーションの設定	値
5	Variation file	Selected
	AHDL Include file	Selected
	VHDL component declaration file	Selected
	Quartus II symbol file	Selected
	Instantiation template file	Selected
	Verilog HDL black-box file	Selected
	Synthesis area and timing estimation netlist	Selected

9. **Finish** をクリックします。mo_clkctrl_gclk モジュールが [図 3](#) に示されるように構築されます。

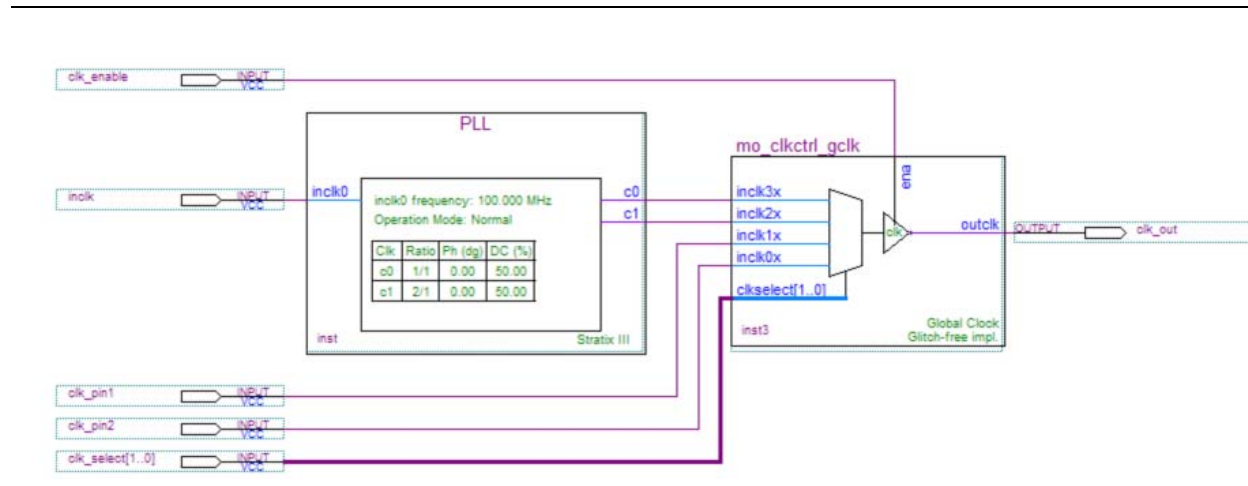
図 3.mo_clkctrl_gclk モジュール



10. Symbol ウィンドウで、**OK** をクリックします。

11. マウスを移動して、mo_clkctrl_gclk シンボルを altclkctrl_ex.bdf ファイルの既存のポートにアラインメントします。クリックしてシンボルを配置します。 [図 4](#) に示すように、これでデザイン・ファイルが完成しました。


図 4. 完全なデザイン回路図




12. File メニューの **Save** をクリックします。
13. フル・コンパイルを実行します。

シミュレーション・ライブラリ

生成されたデザイン・ファイルを適切にシミュレートするために、シミュレーション・モデル・ファイルをシミュレーション・ツールに含めなければなりません。ファイルの情報は、MegaWizard Plug-In Manager の **EDA** タブで調べられます。Quartus II シミュレーション・ライブラリは `<Quartus II installed path>\eda\sim_lib` に配置されます。

 VHDL でコード化されたデザインでは、シミュレーション・モデル・ライブラリ (`<library_name>.vhd` または `<library_name>_atoms.vhd`) は機能の説明またはアトム宣言が含まれています。そして、PACK ファイル (`<library_name>_pack.vhd`) または COMPONENT ファイル (`<library_name>_components.vhd`) と共にシミュレートする必要があります。

 LPM シミュレーション・モデルでは、VHDL でコード化されたデザインには **220model.vhd** および **220pack.vhd** ファイルを使用し、また Verilog HDL でコード化されたデザインには **220model.v** ファイルを使用してください。両方のファイルは、LPM バージョン 220 に基づいて作成されております (EIA-IS1031998 年 10 月)。

HDL コードでのメガファンクションのインスタンス化

以下の方法で、HDL コードでインスタンス化することによってメガファンクションを使用することができます。

- 1-6 ページの「*MegaWizard Plug-In Manager によるメガファンクションのインスタンス化*」。MegaWizard Plug-In Manager を使用して、ファンクションをパラメータ化すること、またラッパー・ファイルを作成することができます。
- 1-7 ページの「*その他の合成ツールのためのネットリストの生成*」。必要に応じてラッパーファイルの代わりに、ネットリスト・ファイルを作成することができます。
- 1-8 ページの「*ポートおよびパラメータ定義の使用でのメガファンクションのインスタンス化*」。HDL コードにファンクションを直接インスタンス化することができます。

MegaWizard Plug-In Manager によるメガファンクションのインスタンス化

このセクションで説明されるように MegaWizard Plug-In Manager を使用して、Quartus II GUI で HDL コードにインスタンス化できるメガファンクションを作成します。MegaWizard Plug-In Manager は、メガファンクションをカスタマイズとパラメータ化するには GUI を提供し、すべてのメガファンクションのパラメータが適切に設定されることを確認します。パラメータを設定し終わるとき、生成したいファイルを指定できます。MegaWizard Plug-In Manager は、正しいパラメータでメガファンクションをインスタンス化し、MegaWizard Plug-In Manager で選択した言語に応じて、Verilog-HDL (.v)、VHD (.vhd)、または AHDL (.tdf) のいずれかを使用したメガファンクション・バリエーション・ファイル (ラッパー・ファイル) を、他のサポートするファイルとともに生成します。

MegaWizard Plug-In Manager は、表 3 で記載される通り以下のファイルを作成するオプションを提供します。

表 3. MegaWizard Plug-In Manager 生成されるファイル

ファイル	説明
<output file>.v (1)	Verilog HDL バリエーション・ラッパー・ファイル Verilog HDL デザインのインスタンス化のためのメガファンクション・ラッパー・ファイルです。
<output file>.vhd (1)	VHDL バリエーション・ラッパー・ファイル VHDL デザインのインスタンス化のためのメガファンクション・ラッパー・ファイルです。
<output file>.tdf (1)	AHDL バリエーション・ラッパー・ファイル AHDL デザインのインスタンス化のためのメガファンクション・ラッパー・ファイルです。
<output file>.inc	AHDL インクルード・ファイル AHDL デザインに使用されます。
<output file>.cmp	コンポーネント宣言ファイル VHDL デザインに使用されます。
<output file>.bsf	ブロック・シンボル・ファイル 回路図のデザインに向き、Quartus II のブロック・デザイン・ファイルで使用されます (.bdf)。
<output file>_inst.v	インスタンス化のテンプレート このファイルは、メガファンクション・ラッパー・ファイルのモジュールの Verilog HDL インスタンス化のサンプルです。
<output file>_inst.vhd	VHDL インスタンス化のテンプレート メガファンクション・ラッパー・ファイルのエンティティの VHDL インスタンス化のサンプルです。
<output file>_inst.tdf	テキスト・デザイン・ファイルのインスタンス化のテンプレート メガファンクション・ラッパー・ファイルのサブデザインの AHDL インスタンス化のサンプルです。
<output file>_bb.v	ブラック・ボックス Verilog HDL モジュール宣言 サードパーティ合成ツール内にメガファンクションをブラック・ボックスとしてインスタンス化する際に使用できる Verilog HDL モジュール宣言ファイル (_bb.v) です。
<output file>_syn.v (2)	合成領域およびタイミング見積りネットリスト 合成領域およびタイミング見積りネットリストを生成するためのオプションをイネーブルする場合、MegaWizard Plug-In Manager は追加ネットリスト・ファイルを生成します。領域とタイミングの見積りを向上するのに特定のサードパーティの合成ツールで使用されるメガファンクション・ネットリストです。

表 3 の注:

- (1) MegaWizard Plug-In Manager は、ウィザードのメガファンクション選択のページでの出力ファイルに選択した言語に応じて、Verilog HDL、VHDL、または AHDL バリエーション・ラッパー・ファイルを生成します。
- (2) MegaWizard Plug-In Manager は、EDA ページ上のウィザードの Generate a synthesis area and timing estimation netlist オプションをオンにする場合にのみ、このファイルを生成します。選択された HDL 言語にかかわらずこのファイルは Verilog HDL 形式で生成されます。


その他の合成ツールのためのネットリストの生成

サードパーティの EDA 合成ツールを使用して、特定のメガファンクションを使用する場合 (Quartus II の統合された合成以外のツール)、ラッパー・ファイルの代わりに領域およびタイミング見積りをオプションで作成できます。

ネットリスト・ファイルは、Quartus II 開発ソフトウェアで使用される、カスタマイズされたロジックを表すものです。このファイルは、メガファンクションにおけるアーキテクチャ・エレメントの接続情報を提供しますが、真の機能を表していない場合があります（グレー・ボックス）。この情報により、特定のサードパーティ合成ツールは、レポート領域およびタイミング見積りを改善できます。更に、合成ツールは、タイミング情報を使用して、タイミング・ドリブン最適化に専念し、結果の品質を改善できます。


ネットリストを生成するには、MegaWizard Plug-In Manager の EDA ページ上の **Generate a synthesis area and timing estimation netlist** オプションをオンにします。ネットリスト・ファイルは <output file>_syn.v ファイルと呼ばれています。選択された HDL 言語にかかわらずこのファイルは Verilog HDL 形式で生成されます。合成にこのネットリストを使用する場合は、配置および配線のために Quartus II プロジェクトのメガファンクション・ラッパー・ファイル <output file>.v または <output file>.vhd を含める必要があります。


合成ツールは、このネットリストを生成するバックグラウンドで Quartus II をソフトウェア呼び出すことができるので、このオプションをオンにするための追加ステップを実行する必要はありません。

 合成ツールに領域およびタイミング見積りネットリストのサポートについて詳しくは、Quartus II ハンドブックの Volume 1 の *「Synthesis」* セクションでベンダのドキュメンテーションまたは適切な章を参照してください。


ポートおよびパラメータ定義の使用でのメガファンクションのインスタンス化

メガファンクションを呼び出して、そのパラメータを他のモジュール、コンポーネントまたはサブデザインの設定の場合と同じように設定することにより、メガファンクションを Verilog HDL、VHDL、または AHDL コードでインスタンス化できます。

 メガファンクション・ポートおよびパラメータの一覧については、Quartus II のヘルプの特定のメガファンクションを参照してください。Quartus II のヘルプは、各メガファンクションの VHDL コンポーネント宣言のサンプルと AHDL ファンクション・プロトタイプも提供します。

 アルテラは、PLL、トランシーバ、LVDS ドライバなどの複雑なメガファンクションに MegaWizard Plug-In Manager を使用することを強く勧めます。MegaWizard Plug-In Manager の使用方法について詳しくは、1-6 ページの *「MegaWizard Plug-In Manager によるメガファンクションのインスタンス化」* を参照してください。

例 1 および *例 2* に示すの Verilog HDL および VHDL コードのサンプルは、2:1 マルチプレクサに接続された入力のいずれかのトップレベルのモジュールで ALTFP_MULT をインスタンス化します。インスタンスのパラメータも同様にコード内で直接定義されています。インスタンスで正しいメガファンクション名が使用されるように、Quartus II のヘルプを参照してください。

 VHDL でメガファンクションインスタンス化するときに、適したライブラリを含めるようにしてください。

例 1. Verilog HDL での ALTFP_MULT のインスタンス化

```
module MF_top (a, b, sel, datab, clock, result);
    input [31:0] a, b, datab;
    input clock, sel;
    output [31:0] result;
    wire [31:0] wire_dataaa;

    assign wire_dataaa = (sel)? a : b;
    altfp_mult inst1 (.dataa(wire_dataaa), .datab(datab), .clock(clock),
.result(result));

    defparam
        inst1.pipeline = 11,
        inst1.width_exp = 8,
        inst1.width_man = 23,
        inst1.exception_handling = "no";
endmodule
```

例 2. VHDL での ALTFP_MULT のインスタンス化

```
library ieee;
use ieee.std_logic_1164.all;
library altera_mf;
use altera_mf.altera_mf_components.all;

entity MF_top is
    port (clock, sel : in std_logic;
        a, b, datab : in std_logic_vector(31 downto 0);
        result : out std_logic_vector(31 downto 0));
end entity;

architecture arch_MF_top of MF_top is
    signal wire_dataaa : std_logic_vector(31 downto 0);
begin

    wire_dataaa <= a when (sel = '1') else b;

    inst1 : altfp_mult
        generic map(
            pipeline => 11,
            width_exp => 8,
            width_man => 23,
            exception_handling => "no")
        port map (
            dataaa => wire_dataaa,
            datab => datab,
            clock => clock,
            result => result);
end arch_MF_top;
```

HDL コードからのメガファンクションの推測

Quartus II 合成機能などの合成ツールは、特定のタイプの HDL コードを認識し、メガファンクションが最適な結果をもたらす場合は、適切なメガファンクションを自動的に推測します。Quartus II ソフトウェアで、メガファンクションを明確にインスタンス化していない場合でも、デザインのコンパイル時にアルテラのメガファンクション・コードが使用されます。アルテラのデバイスのためにそれらが最適化されるので、Quartus II ソフトウェアはメガファンクションを推定します。したがって、

使用面積および性能は、一般的な HDL コードよりよいかもしれません。さらに、アルテラのアーキテクチャ独自の機能、例えば、メモリ、デジタル信号処理 (DSP) ブロック、シフト・レジスタなどにアクセスする場合は、必ずメガファンクションを使用しなければなりません。これらの機能は、基本的なロジック・エレメント (LE) と比較して性能の向上を提供しています。

例 3 と例 4 は、合成ツールで LPM_MULT または ALTMULT_ADD メガファンクションとして推測することができる符号なし乗算器および符号付き乗算器の Verilog HDL の例を示すコード・サンプルです。それぞれの例では、1 つの DSP ブロックの 9 ビット・エレメントに収まります。さらに、レジスタ・パッキングが起こる時には、レジスタのための追加のロジック・セルは必要ではありません。


例 3. Verilog HDL 符号なし乗算器

```
module unsigned_mult (out, a, b);
    output [15:0] out;
    input [7:0] a;
    input [7:0] b;
    assign out = a * b;
endmodule
```

例 4. 入力レジスタおよび出力レジスタとの Verilog HDL 符号付き乗算器 (パイプライン = 2)

```
module signed_mult (out, clk, a, b);
    output [15:0] out;
    input clk;
    input signed [7:0] a;
    input signed [7:0] b;
    reg signed [7:0] a_reg;
    reg signed [7:0] b_reg;
    reg signed [15:0] out;
    wire signed [15:0] mult_out;
    assign mult_out = a_reg * b_reg;
    always @ (posedge clk)
    begin
        a_reg <= a;
        b_reg <= b;
        out <= mult_out;
    end
endmodule
```

 サンプル・コードについて詳しくは、「Quartus II ハンドブック Volume 1」の「[Recommended HDL Coding Styles](#)」の章を参照してください。

 メガファンクション推測を制御するための合成オプションの詳細については、「Quartus II ハンドブック Volume 1」の「[Quartus II Integrated Synthesis](#)」を参照してください。

コンパイル後のメガファンクションの識別

Quartus II 開発ソフトウェアでのコンパイル中に、解析とエラーレクションが実行され、デザインの構造が構築されます。Project Navigator ウィンドウで、コンパイル階層を展開し、メガファンクションを名前検索すると、メガファンクションを見つけることができます。メガファンクション内のノード名を (Node Finder を使用して) 検索するには、**Look in** ダイアログ・ボックスで **Browse** をクリックし、**Hierarchy** ダイアログ・ボックスでメガファンクションを選択します。

新しい Quartus II ソフトウェア・バージョンにおけるメガファンクションの更新

Quartus II ソフトウェアのより新しいバージョンが使用される時には、MegaWizard 生成されたファイルをアップデートする必要はありませんが、特定の最新情報のメガファンクション（例えば、ALTGX トランシーバ・バリエーション）には、新しいソフトウェア・リリースにアップデートすることが必要です。Quartus II ソフトウェアでデザインをコンパイルするときにそのメガファンクションが更新されないと、コンパイル・エラーが発生します。このような場合では、デザインをコンパイルしようとする前に、ウィザードのラッパーを更新するよう促すリリース・ノートが表示されます。

結論

Quartus II 開発ソフトウェアは、加算器やカウンタなどの単純な演算ユニットから、最新 PLL (Phase-Locked Loop) ブロック、分周、およびメモリ構造に及ぶ、パラメータ化可能なメガファンクションを提供します。これらのメガファンクションはアルテラのデバイス用に性能が最適化されております。したがって、コーディング・プロセスを自動化して貴重な設計時間を節約するため、より効率的なロジック合成およびデバイス実装を可能にします。アルテラは、デザインの実装時にこれらの機能を使用することを推奨します。これによって、一貫してデザイン目標を達成することができます。

改訂履歴

以下の表に、このユーザーガイドの改訂履歴を示します。

日付	ドキュメント・バージョン	変更内容
2009 年 2 月	1.0	初版。

アルテラへのお問い合わせ

アルテラの製品に関する最新の情報については、次の表を参照してください。






お問い合わせ先 (注1)	お問い合わせ方法	アドレス
技術的なご質問	ウェブサイト	www.altera.co.jp/support
技術トレーニング	ウェブサイト	www.altera.co.jp/training
	電子メール	custrain@altera.com
アルテラの資料に関するお問い合わせ	電子メール	literature@altera.com
一般的なお問い合わせ ソフトウェア・ライセンスに関するお問い合わせ	電子メール	nacomp@altera.com
	電子メール	authorization@altera.com

注:

(1) 詳しくは、日本アルテラまたは販売代理店にお問い合わせください。

表記規則

本書では、以下の表に示す表記規則を使用しています。

書体	意味
太字かつ文頭が大文字	コマンド名をダイアログ・ボックス・タイトルを表します。例えば、Save As ダイアログ・ボックス
太字	ディレクトリ名、プロジェクト名、ディスク・ドライブ名、ファイル名、ファイルの拡張子、ダイアログ・ボックス・オプション、ソフトウェア・ユーティリティ名その他の GUI ラベルを表します。例： <code>\qdesigns directory, d: drive, and chiptrip.gdf file</code>
斜体かつ文頭が大文字	資料のタイトルを表します。例： <code>、 AN 519: Stratix IV Design Guidelines</code>
斜体	変数を表します。例えば、 <code>n + 1</code> 変数名は、山括弧 (<>) で囲んでいます。例： <code><file name></code> および <code><project name>.pof</code> ファイル
文頭が大文字	キーボード・キーおよびメニュー名を表します。例： <code>Delete</code> キー、および <code>Options</code> メニュー
「小見出しタイトル」	かぎ括弧は、資料内の小見出しおよび <code>Quartus II Help</code> トピックのタイトルを表します。例：「表記規則」
Courier フォント	信号、ポート、レジスタ、ビット、ブロック、およびプリミティブ名を表します。例： <code>data1, tdi, および input</code> 。アクティブ <code>Low</code> 信号は、サフィックス <code>n</code> で表示されています。例： <code>resetn</code> コマンドライン・コマンド、および表示されているとおりに入力する必要があるものを表します。例： <code>c:\qdesigns\tutorial\chiptrip.gdf</code> また、 <code>Report</code> ファイルなどの実際のファイルのセクション、ファイルの構成要素への参照（例： <code>AHDL</code> キーワードの <code>SUBDESIGN</code> ）、ロジック・ファンクション名（例： <code>TRI</code> ）も <code>Courier</code> フォントで表記されています。
1、2、3、および a、b、c.. など	手順など項目の順序が重要なものは、番号が付けられリスト形式で表記されています。
■ ■	箇条書きの黒点などは、項目の順序が重要ではないものに付いています。
	指差しマークは、要注意箇所を表しています。
	注意は、製品または作業中のデータに損傷を与えたり、破壊したりするおそれのある条件や状況に対して注意を促します。
	警告は、ユーザーに危害を与えるおそれのある条件や状況に対して注意を促します。
	矢印は、 <code>Enter</code> キーを押すよう指示しています。
	足跡マークは、詳細情報の参照先を示しています。



101 Innovation Drive
San Jose, CA 95134
www.altera.com
Technical Support
www.altera.com/support

Copyright © 2009 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