MAX II デバイスの UFM の使用

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。

2007 年 12 月 ver 1.0 Application Note 489

はじめに

このアプリケーション・ノートでは、不揮発性情報の保存について説明します。大部分の CPLD では、不揮発性情報のストレージを実現するためにシリアル EEPROM を使用していますが、MAX® II CPLD はユーザ・フラッシュ・メモリ（UFM）を提供する唯一の CPLD です。この UFM を使用すると、8 K ビットまでの不揮発性情報を保存することができます。UFM は書き込みの他に、シリアル・インタフェース、パラレル・インタフェース、その他のプロトコルもサポートしています。このアプリケーション・ノートでは、UFM に対する情報の効率良い保存方法および読み出し方法と、I²C プロトコルを使用して、MAX II CPLD の UFM に対するインタフェースとアクセスの方法について説明します。

ユーザ・フラッシュ・メモリ

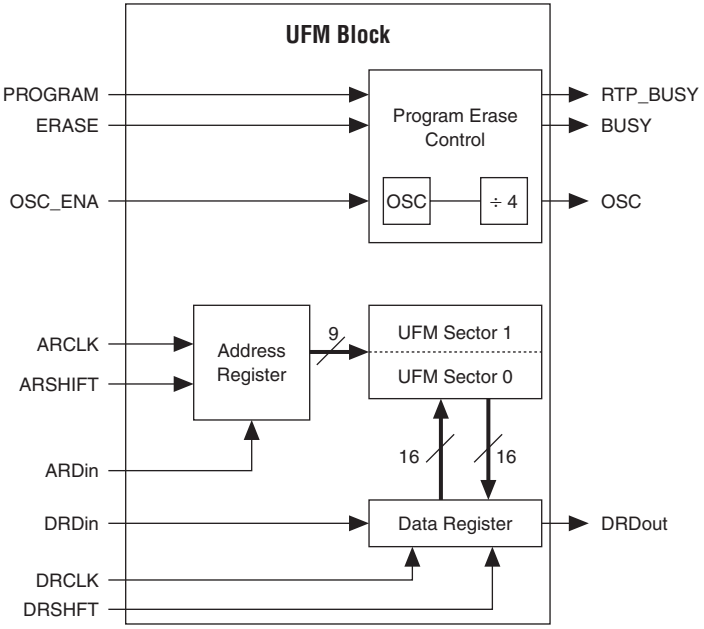
次に、UFM のアプリケーション例と機能を示します。

- UFM を使用すると、ASSP またはプロセッサの設定ビットや、ボード製造時の電子ID情報などの重要な不揮発性情報の保存、またはプロセッサのパワーオン時に LCD に表示する情報の保存を行うことができます。
- 機能としては、シリアル・コミュニケーション・インタフェース（SCI）、シリアル・ペリフェラル・インタフェース（SPI）、I²C（Inter-Integrated Circuit）、Microwire などのパラレルおよびシリアル・インタフェースおよびその他の独自プロトコルなどがあります。MAX II デバイスは、既製品の EEPROM デバイスより柔軟性の優れたインタフェースを提供します。
- UFM はオシレータを内蔵しているため、外付けクロック回路のために必要とされるスペースとコストが不要になります。

MAX II CPLD を使用すると、ロジック・デバイスとメモリ・デバイスをデザイン・ボードに組み込むことができるため、チップ間遅延、ボード・スペース、トータル・システム・コストを削減することができます。

UFM は、各 4 K ビットの 2 つのセクションに分けることができます。アドレス・レジスタは、データ読み書きの対象となる UFM メモリ・ロケーションのアドレスを指定します。データ・レジスタは、UFM に対して読み書きするデータを保持します。書き込み消去コントロール・ブロックは、UFM の書き込みまたは消去に使用され、また内蔵オシレータのイネーブルにも使用されます。図 1 に、UFM のブロック図を示します。

図 1.　ユーザ・フラッシュ・メモリ（UFM）のブロック図



MAX II デバイスの UFM の使用

UFMは、セクタ0とセクタ1の2つのセクタに分割されます。各セクタは4,096 ビットで、それぞれ 000h ～ 0FFh と 100h ～ 1FFh のアドレス範囲を持っています。アドレス・ロケーションは 9 ビットでアクセスされ、各アドレス・ロケーションは 8 ビットのデータを保存することができます。

Read/Stream Read、Program、Erase の各命令を使用して、UFM 内のデータをロードおよび変更することができます。Read/Stream Read 命令は、アドレス・レジスタで指定されたロケーションの値を読み出すときに使用されます。アドレス・レジスタをインクリメントすることにより、連続メモリ・ロケーションを読み出すことができます（Stream Read）。Program 命令は UFM へデータを書き込むときに、Erase 命令は UFM の値を変更するときに、それぞれ使用します。ただし、1 つのアドレス・ロケーションを消去することはできません。この命令は、UFM（A2A1A0 = 111）全体の消去、または送信されたアドレスの MSB で指定される UFM のセクタの消去を行います。

また、UFM は、イネーブル・コントロール付きオシレータも内蔵しています。この信号はロジック・アレイ・ブロック全体に供給可能で、さらに ARCLK と DRCLK へ接続することもできます。

このデザインでは、I²C シリアル・インタフェースを使用して、UFM に対してデータを読み書きしています。次に、I²C プロトコルについてまとめます。

- 2 本の双方向バス・ラインを使用します。SDA ラインはアドレスとデータの転送に使用され、SCL ラインは I²C クロックとして使用されます。両ラインは、使用されないとき、プルアップ抵抗により High レベルに維持されます。
- 通信は、SCL が High のときの SDA ラインの High から Low への変化でスタートます。
- その後に、スレーブのアドレスが SDA で送信されます。データ転送は、スレーブによりアドレスがアクノリッジされたときに開始されます。送信データは、クロックが High の間 SDA ライン上で安定している必要があります。
- 通信のストップ条件は、SCL が High のときの SDA ラインの Low から High への変化となります。

表 1 に、UFM ブロックの信号を示します。

表 1. UFM ブロックで使用される信号の概要 注 (1)

信号	説明
DRDin	DRCLK パルス毎に、データをデータ・レジスタへシフト入力します。
DRCLK	DRDin から DRDout へのデータのシフト、および UFM からデータ・レジスタへのデータのパラレル・ロードを制御します。
DRSHFT	High: DRDin から LSB をシフト入力、DRDout へ MSB をシフト出力。 Low: データを UFM からデータ・レジスタへラッチ。
ARDin	メモリ・ロケーション・アドレスを保存するためのシリアル入力です。
ARCLK	メモリ・ロケーション・アドレスのシフトとアドレス・レジスタへ入力するアドレスのインクリメントを制御します。
ARSHFT	High: ARDin からアドレス・レジスタへアドレスをシリアル・シフト。 Low: アドレス・レジスタ内のアドレスをインクリメント。
PROGRAM	この信号の立ち上がりエッジで、データをデータ・レジスタからアドレス・レジスタで指定されるメモリ・ロケーションへ書き込みます。
ERASE	この信号の立ち上がりエッジで、アドレス・レジスタの MSB で指定されるメモリ・セクタを消去します。
OSC_ENA	UFM の内蔵オシレータをイネーブルするときに使用する信号です。
DRDout	データ・レジスタの出力。MSB が最初に取得されます。
BUSY	メモリが program 命令または erase 命令のためビジーであることを表示します。
RTP_BUSY	リアルタイム ISP 機能をイネーブルした場合に必要です。

表 1 の注：

(1) 詳細は、「MAX II デバイス・ハンドブック」の「MAX II デバイスにおけるユーザ・フラッシュ・メモリの使用」の章の「UFM インタフェース信号」の表を参照してください。

フラッシュ・メモリ・メガファンクションは、UFM をインスタンス化する際に使用されます。このメガファンクションを使用すると、インタフェース、所要メモリ・サイズ、UFM 内のデータ書き込み保護オプション、内蔵オシレータをイネーブルするオプションを選択し、さらに CPLD のポートへ配線し、また UFM スレーブ・アドレスの先頭 4 ビット（ビット A6 ～ A3）も設定することができます。さらに、ボード上の残りのスレーブ・アドレス 3 ビット（A2 ～ A0）を書込むオプションも使用することができます。

Quartus II ソフトウェア内での UFM メガファンクションのインスタンス化

UFMメガファンクションをインスタンス化するには、次のステップを実行します。

1. Quartus II ソフトウェアを使用して内蔵オシレータをインスタンス化するプロジェクトを開きます。
2. Tools メニューの **MegaWizard Plug-In Manager** をクリックします。**MegaWizard Plug-In Manager [page 1]** ダイアログ・ボックスが表示されます。
3. **Create a new custom megafunction variation** を選択し、**Next** をクリックします。**MegaWizard Plug-In Manager [page 2a]** ダイアログ・ボックスが表示されます。表 2 に、**MegaWizard Plug-In Manager [page 2a]** ダイアログ・ボックス内のオプションと設定を示します。

表 2. MegaWizard Plug-In Manager [page 2a] のオプション

オプション	設定
Which device family will you be using?	**MAX II** を選択します。
Which megafunction would you like to customize?	「+」アイコンをクリックして **Memory Compiler** を展開して、**Flash Memory** を選択します。
Which type of output file do you want to create?	**AHDL**、**VHDL**、または **Verilog HDL** を選択します。
What name do you want for the output file?	ファイル名を入力します。

4. **Next** をクリックします。**MegaWizard Plug-In Manager ALTUFM [page 3 of 5]** ダイアログ・ボックスが表示されます。表 3 に、ALTUFM ウィザード・ページ 3 内のオプションと設定を示します。

表 3. MegaWizard Plug-In Manager [page 3 of 5] のオプション

オプション	設定	その他のオプション
What is the interface protocol?	None	Use arclkena port (clock enable for arclk)
		Use drclkena port (clock enable for drclk)
	Parallel	What is the width of the address bus?
		What is the width of the data bus?
		Use the 'osc' (oscillator) output port
	Serial Peripheral Interface (SPI)	Base mode (uses 8 bit address and data)
		Extended mode (uses 16 bit address and data)
		Use the 'osc' (oscillator) output port
	I²C	What is the MSB of the device address (in binary)?
		What is the size of the memory?
		Use the 'osc' (oscillator) output port
What is the access mode for the user flash memory?	Read/Write または Read Only	—

5. **Next** をクリックします。

- **I2C** を選択すると、ALTUFM ウィザード・ページ 4 が表示されます。表 4 に、ALTUFM ウィザード・ページ 4 内のオプションと設定を示します。

  Next をクリックします。ALTUFM ウィザード・ページ 3 が表示されます。

表 4. MegaWizard Plug-In Manager [page 4 of 7] のオプション （1 / 2）

オプション	設定	その他のオプション
What is the write configuration for the I2C protocol?	Single byte write	—
	Page write	8 bytes、16 bytes、32 bytes から選択

表 4. MegaWizard Plug-In Manager [page 4 of 7] のオプション （2 / 2）		
オプション	設定	その他のオプション
Write protect	Use the 'wp' (write protect) input	Write protect applies to the full memory
		Write protect applies only to the upper half of the memory
What erase method should be used in I2C protocol?	Device Slave Address Full Erase (3 LSBs are 111)	—
	Sector Erase Triggered by Byte Address (1)	Sector 0: Trigger erase when writing to binary address (MSB is always '0')
		Sector 1: Trigger erase when writing to binary address (MSB is always '1')
	Sector Erase Triggered by 'a2' bit	—
	No Erase	—

表 4 の注 :

(1) このオプションは、**What is the write configuration for the I2C protocol?** で **Single byte write** を選択した場合にのみ使用できます。

- **None**、**Parallel**、または **Serial Peripheral Interface**（**SPI**）を選択すると、ALTUFM ウィザード・ページ 4 が表示されます。表 5 に、ALTUFM ウィザード・ページ 3 のオプションと設定を示します。

表 5. MegaWizard Plug-In Manager [page 4 of 6] のオプション		
オプション	設定	
Do you want to specify the initial content of the memory?	No, leave it blank	
	Yes, use this file for the memory content	File name: を入力するか確認します。
What is the oscillator frequency for the User Flash Memory? (for simulation only)	3.33 MHz または 5.56 MHz を選択します。	
What is the erase time for the User Flash Memory? (for simulation only)	値を入力します。	
What is the program time for the User Flash Memory? (for simulation only)	値を入力します。	

6. **Next** をクリックします。ALTUFM ウィザード・ページ 5 が表示されます。

7. ALTUFM ウィザード・ページ 5 で、このオプションを選択すると、そのネットリストのファイルも使用可能です。グレーのチェックマークは自動的に生成されるファイルを示し、赤色のチェックマークはオプションのファイルを示します。**Next** をクリックします。

8. ALTUFM ウィザード・ページ 6 には、生成されるファイル・タイプのリストが表示されます。自動生成されたバリエーション・ファイルには、ページ 2a で指定した言語のラッパー・コードが含まれます。ページ 7 では、生成するファイルのその他のタイプを指定することができます。AHDL インクルード・ファイル (< ファンクション名 >**.inc**)、VHDL コンポーネント宣言ファイル（< ファンクション名 >**.cmp**）、Quartus II シンボル・ファイル（< ファンクション名 >**.bsf**）、インスタン化テンプレート・ファイル（< ファンクション名 >**.v**）、Verilog HDL ブラック・ボックス・ファイル（< ファンクション名 >**_bb.v**）から選択します。**Finish** をクリックします。

実装

本デザイン例は、EPM240 をはじめとする MAX II CPLD を使用して実装できます。このデザイン例の UFM では、I²C インタフェースを持つように設定されています。MAX II UFM へのアクセスは、PC を利用した I²C バス環境シミュレータを使用してデモストレーションしています。実装では、このアプリケーション・ノートに添付されているデザイン例ソース・コードの使用と MAX II の GPIO に対する UFM の I²C インタフェース・ラインの割り当てを採用しています。次に、I²C シミュレータを使用して UFM をアクセスし、読み出しまたは書き込みを行います。このシミュレータは、PC パラレル・ポートとそれに接続するインタフェース・ハードウェアにより、I²C 準拠の 2 線式バスを実現しています。I²C 環境の設定について詳しくは、Maxim/Dallas Semiconductor のアプリケーション・ノート AN3230 を参照してください。

www.maxim-ic.com/appnotes.cfm/an_pk/3230

同様の無償ソフトウェアは、以下からダウンロードすることができます。

http://files.dalsemi.com/system_extension/AppNotes/AN3315/ParDS2W.exe

PC のパラレル・ポートを介したインタフェース・ハードウェアと、このユーティリティ・プログラムにより、MAX II CPLD の UFM との I²C 通信が可能となります。すなわち、I²C マスタである PC ユーティリティ・プログラムから、I²C スレーブである UFM の読み書きが可能となります。

表 6 に、このデザイン例の MDN-B2 デモ・ボード上での実装を示します。

表 6. EPM240G のピン・アサインメント

信号	ピン
SCLK	ピン 39
SDA	ピン 40
a1	ピン 37
a2	ピン 38

未使用ピンは、Quartus II ソフトウェアの **Device and Pin Options** ダイアログ・ボックスで、**As input tri-stated** に割り当ててあります。Quartus II ソフトウェアの Assignment Editor で、SCLK ピンと SDA ピンに対し、**Auto Open-Drain Pins** をイネーブルにしてください。これらの設定の後にコンパイルを行います。

MDN-B2 デモ・ボードで本デザインを実行させるには、以下の手順を実行してください。

- デモ・ボードの電源をオンにします（スライド・スイッチ SW1 を使用）。
- デモ・ボード上の JTAG ヘッダ JP5 とプログラミング・ケーブル（ByteBlaster II または USB-Blaster）を使用してデザインを MAX II CPLD へダウンロードします。
- プログラミング・プロセスの起動前と起動中、デモ・ボードの SW4 を押し続けます。完了したら、電源をオフにして JTAG コネクタを取り外します。
- PC 上で、パラレル・ポートで駆動される I²C 環境を次のように設定します。
  - I²C で規定されたプロトコルを使用してスレーブと通信するために、Maxim パラレル・ポート・ユーティリティのようなソフトウェア・ユーティリティをダウンロードします。パラレル・ポート・ソフトウェアをインストールします。この例では、プログラム **ParDS2W.exe** を使用しています。
    http://files.dalsemi.com/system_extension/AppNotes/AN3315/ParDS2W.exe
  - このパラレル・ポート・ユーティリティのために、Windows XP または Windows 2000 内でパラレル・ポートをアクセスできるようにするパラレル・ポート・ドライバをインストールする必要があります。Direct-IO（www.direct-io.com）からドライバが提供されており、以下からダウンロードすることができます。
    www.direct-io.com/Direct-IO/directio.exe
  - インストレーションの後、Direct-IO プログラムを設定する必要があります。Windows のコントロール・パネル内で、Direct IO アイコンをクリックします。パラレル・ポートの開始アドレスと終了アドレスを入力します。ほとんどの場合 378h ～ 37Fh ですが、以下の設定を調べて、

PC のパラレル・ポート・アドレスを確認してください。Control Panel/System/Hardware/Device Manager/Ports/EC P Printer port (LPT)/Resources

- パラレル・ポートが ECP 以外のタイプに設定されている場合は、PC 起動時に BIOS 設定を変更して ECP へ変更してください。
- Direct IO コントロール・パネルの **Security** タブを選択し、**ParDS2W.exe** プログラムのディレクトリ・パスを調べます。**Open** をクリックして、**Add** をクリックします。このユーティリティのパスが、**Allowed Processes** フィールドに表示されます。**OK** をクリックして、Control Panel ウィンドウを閉じます。
- MDN-B2 デモ・ボードに添付されている I²C ドングルをパラレル・ポートに接続します。必要に応じて拡張コードを使用して、パラレル・ポート接続をデモ・ボードの近くまで延ばします。
- I²C パラレル・ポート・ドングル上にある 4 ピン・ソケットを、ソケットの赤マークと JP3 ヘッダの `pin#1` を合わせて、デモ・ボードの I²C ヘッダ（JP3）に接続します。
- SW3（MDN-B2 デモ・ボードの 8 ウェイ DIP スイッチ）のスイッチ 1 とスイッチ 2 を ON にします。
- ParDS2W プログラムを開いて、PC の該当するパラレル・ポート・アドレスを選択し（Direct IO 設定時の表示のように）、**Address2-Wire Device Address** に `B0H` を設定します。
- 最後に、**Test Circuit** タブで I²C の設定をテストして、Status ウィンドウに Test PASS メッセージが表示されるのを確認してください。これが表示されると、I²C 環境の設定が完了します。

- パラレル・ポート・ユーティリティ・プログラムの **2-Wire Utility** セクションで **One byte Write/Read** セクションを使用し、アドレスとデータ（各々 1 バイト長、16 進の 2 桁）を指定して、任意のメモリ・アドレス・ロケーションへシングル・バイト **WRITE** 命令を実行します。
- 同様に、同じアドレス・ロケーションへ **READ** 命令を実行して、そのアドレス・ロケーションの値を調べます。書き込んだ値と同じ値になっているはずです。書き込みを行わないかぎり、他のすべてのアドレス・ロケーションの値は FFh となっているはずです。
- アドレス・ロケーションへ **WRITE** を複数回行う場合は、その前に消去を行う必要があります。
- スレーブ・アドレスとして BEh を選択し、FFh の **WRITE** 動作を行うことにより、すべての値を消去することができます。この動作により、UFM の値は FF に戻されます。
- デモ・ボードにある SW3（8 ウェイ DIP スイッチ）のスイッチ 1 とスイッチ 2 を使用して、I²C スレーブ・アドレス（a2、a1）の 5 番目と 6 番目のビットを設定することができます。フラッシュ・メモリ・メガファンクションのインスタンス化時に、両ビットが 0 に設定され、I²C スレーブ・アドレスの最初の 4 ビットの MSB が設定されます（このケースでは 1011 すなわち Bh）。

ソース・コード

このアプリケーション・ノートのデザイン例は、Verilog HDL を使用して作成しており、MDN-B2 デモ・ボードを使用したデモとなっています。ソース・コード、テストベンチ、および完成したQuartus IIプロジェクトは、以下から入手可能です。

www.altera.co.jp/literature/an/an489_design_example.zip

まとめ

MAX II CPLD は独自のユーザ・フラッシュ・メモリを内蔵しているため、不揮発性メモリのサポートを必要とする広範囲なロジック・ソリューションを実装するための最適な選択肢になっています。さらに、MAX II デバイスは、低消費電力、使いやすいパワーオン機能、多電圧機能、および内蔵オシレータを備えているため、非常に汎用性の高いプログラマブル・ロジック・デバイスとなっています。

関連情報

以下に、関連資料を示します。

- MAX II CPLD ホームページ :
  www.altera.co.jp/products/devices/cpld/max2/mx2-index.jsp
- MAX II デバイスの資料ページ :
  www.altera.co.jp/literature/lit-max2.jsp
- MAX II パワーダウン・デザイン :
  www.altera.co.jp/support/examples/max/exm-power-down.html
- MAX II デバイス・アプリケーション・ノート :
  「AN 428: MAX II CPLD のデザイン・ガイドライン」
  「AN 422: MAX II CPLD を使用したポータブル・システムにおける消費電力の管理」

改訂履歴

表 7 に、このアプリケーション・ノートの改訂履歴を示します。

表 7. 改訂履歴

日付 & バージョン	変更内容	概要
2007 年 12 月 v1.0	初版	—

101 Innovation Drive
San Jose, CA 95134
www.altera.com
Literature Services:
literature@altera.com