



XAPP1232 (v1.0) 2016 年 3 月 3 日

Vivado Design Suite を使用した USER_ACCESS によるビットストリーム識別

著者 : Kyle Wilkinson

概要

このアプリケーション ノートでは、FPGA ユーザー デザインにアクセス可能なビット ストリーム識別のソリューションについて説明します。Vivado® Design Suite の write_bitstream の使用手順および Vivado ツールを使用した識別情報へのアクセス手順について説明します。

はじめに

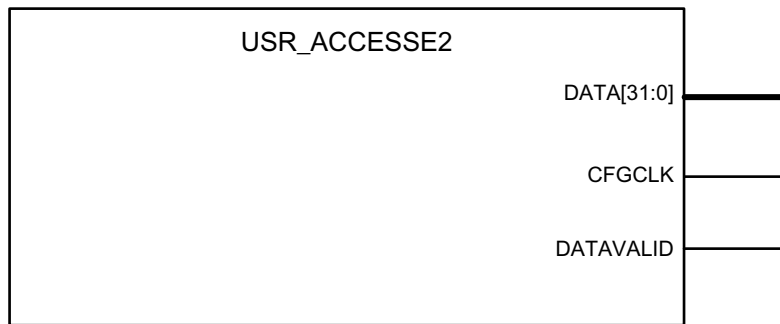
ビットストリームの追跡は、多くのアプリケーションで有用です。たとえば、特定デザインのバージョンを識別したり、デザインを最適化するために複数回実行したインプリメンテーションを追跡する場合などのより複雑な状況ではシリアルナンバーを付けるために使用します。デザインにはあらゆる種類のスタティック コードを組み込むことができるため、バージョンデータのサイズやフォーマットには無限の可能性があります。ただし、デザインの一部あるいは全体を再コンパイルする必要があり、その作業は退屈で制限があります。すべての 7 シリーズ FPGA および UltraScale™ FPGA に含まれる USER_ACCESS レジスタは、ファブリック アクセス可能な 32 ビットのレジスタで、ビットストリーム生成プロセス中にバージョン情報を組み込むことができるため、設計およびインプリメンテーション時間に与える影響を最小限にして、柔軟性を提供する最適なソリューションです。

USER_ACCESS プリミティブ

FPGA コンフィギュレーション ロジックには、USER_ACCESS という読み出しおよび書き込み可能な 32 ビット レジスタが含まれています。このレジスタには、ユーザー デザインから直接アクセスできます。7 シリーズ デバイスと UltraScale デバイスの両方で、USER_ACCESS レジスタのプリミティブ名は USER_ACCESS2 です (図 1 参照)。このアプリケーション ノートでは、これらのプリミティブすべてを USER_ACCESS と簡略して示しています。このコンポーネントを使用することによって、FPGA ビットストリームで格納された 32 ビット値へ FPGA ロジックから直接アクセスできます。

USER_ACCESS は、コンフィギュレーション インターフェイス、JTAG、または内部コンフィギュレーション アクセス ポート (ICAP) を介して変更できますが、このアプリケーション ノートの目的上、ここでは静的な値として考えます。コンフィギュレーション インターフェイスから USER_ACCESS レジスタが更新されると、DATAVALID 出力ポートが切り替わります。CFGCLK は、コンフィギュレーションクロックを反映します。ここでは動的書き込み動作については説明していません。このレジスタへの動的書き込み動作の詳細は、各 FPGA ファミリのコンフィギュレーション ユーザー ガイドで、「AXSS レジスタ」を検索して参照してください。

- 『UltraScale アーキテクチャ コンフィギュレーション ユーザー ガイド』(UG570) [参照 1]
- 『7 シリーズ FPGA コンフィギュレーション ユーザー ガイド』(UG470) [参照 2]



X1232_01_030515

図 1 : USR_ACCESS コンポーネントの概略図

タイムスタンプ

USR_ACCESS レジスタは、32 ビットのユーザー指定値を用いてコンフィギュレーションするか、あるいはタイムスタンプ機能を使用してビットストリーム生成コマンド (`write_bitstream`) で自動的に読み込むことができます。リビジョン、デザイン追跡、シリアルナンバー タイプのアプリケーションには、ユーザー指定値が使用できます。インプリメンテーションが複数回実行され、それによってデザイン最適化の値が変更されていても、ソース デザイン自体は変更されていない場合には、タイムスタンプ機能が有益です。タイムスタンプ値はビットストリーム ファイルのタイムスタンプと比較されて、デバイスのデザインを数あるソースからその 1 つと関連付けます。ソース コードを変更してタイムスタンプ機能をインプリメントするのは簡単ではありません。USR_ACCESS を使用することで、より正確なタイムスタンプ値を得ることができます。

USR_ACCESS の Write_Bitstream プロパティ

コマンドを利用する場合は、[Generate Bitstream] プロセス (`write_bitstream`) で、以下の Tcl コマンドを使用します。

```
set_property BITSTREAM.CONFIG.USR_ACCESS NONE |0x<8-digit hex> |TIMESTAMP
[current_design]
```

このオプションに値が入力されない場合、または「NONE」と入力された場合は、このレジスタに対して実行される動作はありません (デフォルトはすべて 0)。

NONE - DEFAULT

8 文字の 16 進数値が検出された場合、この値が USR_ACCESS レジスタへ格納されます。

0XXXXXXXX

値としてキーワード、TIMESTAMP と入力された場合は、

TIMESTAMP

`write_bitstream` が、次のフォーマットで、その時点のタイムスタンプを 32 ビットの USR_ACCESS レジスタへ挿入します。

```
dddddd MMMM_yyyyyy_hhhhh_mmmmmm_ssssss
(ビット 31) ..... (ビット 0)
```

説明：

dddddd = 1 ~ 31 で「日」を示す 5 ビット

MMMM = 1 ~ 12 で「月」を示す 4 ビット

yyyyyy = 0 ~ 63 で「西暦」を示す 6 ビット (2000 年 ~ 2063 年)

hhhhh = 0 ~ 24 で「時」を示す 5 ビット


mmmmm = 0 ~ 60 で「分」を示す 6 ビット

sssss = 0 ~ 60 で「秒」を示す 6 ビット

TIMESTAMP 値を使用する場合、分と秒の値はファイルのタイムスタンプと一致しない場合があります。タイムスタンプ値は、ビットストリーム生成プロセスが開始されるときに取得されますが、オペレーティングシステムファイルのタイムスタンプはファイル生成プロセスの最後で取得されます。したがって、マシンのスピードや `write_bitstream` に要求される動作の複雑さなどによって、これらの値は厳密には一致しない場合があります。同様に、「時」または「日」の終りに近い段階でファイル生成が行われる場合も一致しない可能性があります。

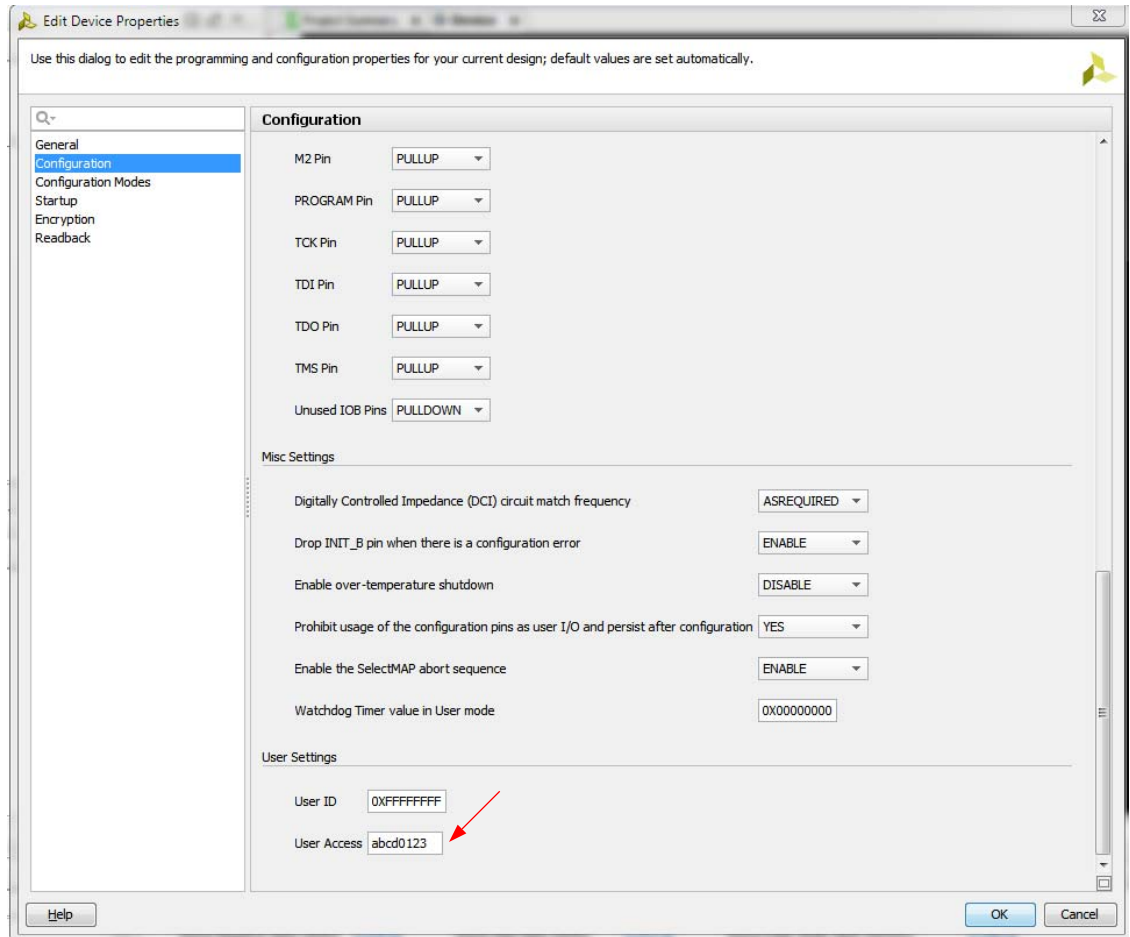
Vivado ツールのフロー

Vivado ツールを使用した場合、USR_ACCESS 機能のインプリメンテーションにはいくつかの方法があります。USR_ACCESS レジスタの設定に推奨される方法は、I/O の配置 (LOC) や IOSTANDARD 制約と同様に、USR_ACCESS レジスタをザイリンクス デザイン制約 (XDC) ファイルに追加してから合成とインプリメンテーションを実行することです。代わりに、[Edit Device Properties] プロセスの GUI を使用して、このオプションを入力することもできます。Vivado 統合設計環境 (IDE) または GUI を使用してオプションにアクセスするには、次の手順に従います。

1. [Synthesized Design] または [Implemented Design] を開きます。
2. [Tools] メニューで [Edit Device Properties] をクリックします。
3. [Configuration] をクリックします。
4.  図 2 に示すように、8 桁の 16 進数値を [User Access] に入力します。
5. タイムスタンプ機能を使用する場合は、「abcd0123」を「TIMESTAMP」に変更します。

注記：このオプションに何も入力しない場合はデフォルトの NONE となり、レジスタ値がすべて 0 になります。

6. [OK] をクリックします。
7. [Generate Bitstream] をクリックします。
8. [Save] をクリックします。XDC が更新されます。[Synthesis and Implementation] を再実行します。



X1232_02_030515

図 2 : Edit Device Properties

write_bitstream プロパティが正しく書き込まれたことを確認するには、XDC ファイルを開き、USR_ACCESS プロパティを検索します。次に例を示します。

```
set_property BITSTREAM.CONFIG.USR_ACCESS abcd0123 [current_design]
#####
# End
#####
```

Vivado ツールで USR_ACCESS レジスタを読み出す機能

Vivado ハードウェア マネージャー機能を使用して、USR_ACCESS レジスタを読み出すことができます。USR_ACCESS レジスタは、選択された hw_device レジスタに含まれています。有効な hw_target に接続すると、USR_ACCESS レジスタを確認する機能が使用可能になります。これによって、レジスタに意図した値が実際に書き込まれたことを確認したり、デバイスが適切な BIT ファイルを使用してコンフィギュレーションされたかどうかを確認できます。USR_ACCESS レジスタが正しく設定されたことを GUI を使用して確認するには、[図 3](#) を参照してください。

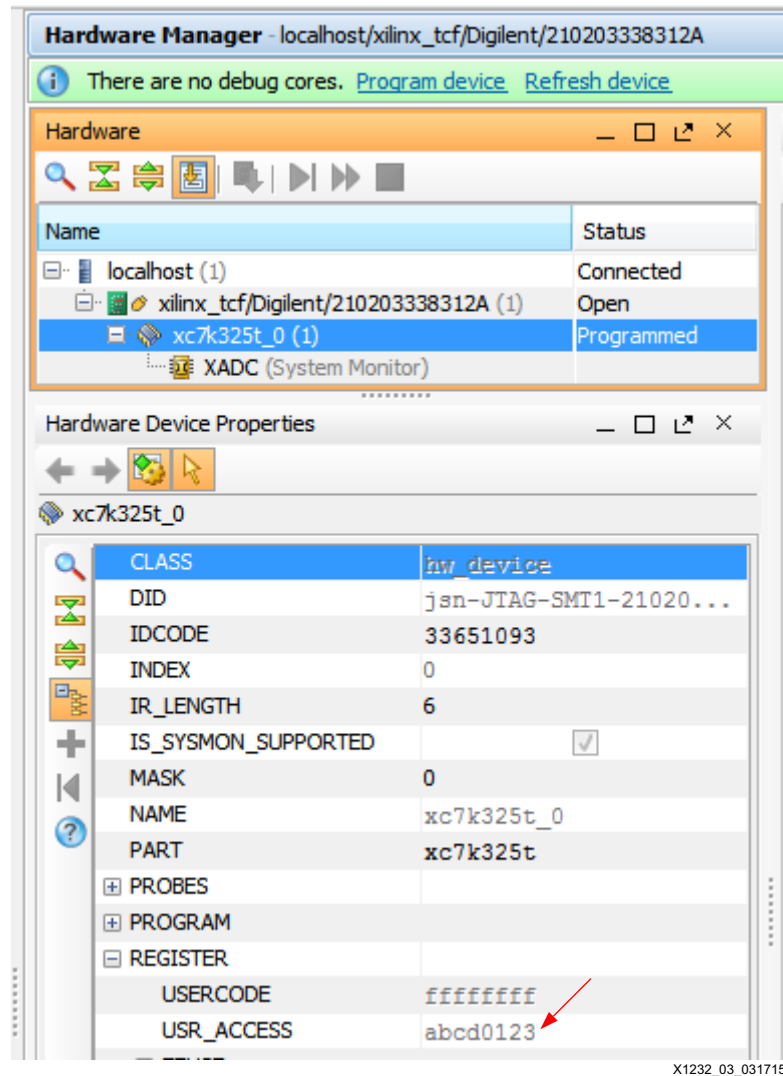
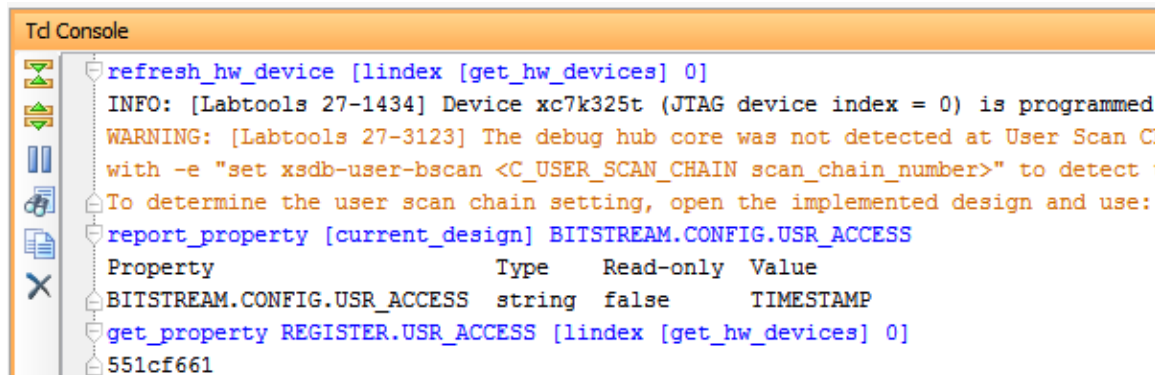


図 3 : Hw_target デバイス プロパティ

Tclを使用してUSR_ACCESSレジスタが正しく設定されたことを確認するには、デバイスのREGISTER.USR_ACCESSのget_propertyを使用します。

```
get_property REGISTER.USR_ACCESS [lindex [get_hw_devices] 0]
```

図 4 を参照してください。



```
Tcl Console
refresh_hw_device [lindex [get_hw_devices] 0]
INFO: [Labtools 27-1434] Device xc7k325t (JTAG device index = 0) is programmed
WARNING: [Labtools 27-3123] The debug hub core was not detected at User Scan Cl
with -e "set xsdb-user-bscan <C_USER_SCAN_CHAIN scan_chain_number>" to detect t
To determine the user scan chain setting, open the implemented design and use:
report_property [current_design] BITSTREAM.CONFIG.USR_ACCESS
Property                Type    Read-only  Value
BITSTREAM.CONFIG.USR_ACCESS string  false      TIMESTAMP
get_property REGISTER.USR_ACCESS [lindex [get_hw_devices] 0]
551cf661
```

X1232_04_031715

図 4 : Tcl get_property

まとめ

USR_ACCESS は、デザインに変更を加えた場合のビットストリームの追跡や、ソースコードは変更せずに特定のインプリメンテーションを実行したり、再インプリメントを実行した場合のビットストリーム追跡に使用する機能です。

付録 A : インスタンスレーション テンプレート

ここでは、7シリーズデバイスのインスタンスレーションテンプレートを示します。ポート名は、各 UltraScale デバイスで同一です。

- Artix®-7 : USR_ACESSE2
- Kintex®-7 : USR_ACESSE2
- Virtex®-7 : USR_ACESSE2

7シリーズデバイスの VHDL インスタンスレーションテンプレートは次のとおりです。

```
Library UNISIM;
use UNISIM.vcomponents.all;

USR_ACCESS_7series_inst : USR_ACESSE2

port map (
  CFGCLK => CFGCLK, -- Not utilized in the static use case in this application note
  DATA => DATA, -- 32-bit output Configuration Data output
  DATAVALID => DATAVALID -- Not utilized in the static use case in this application note
);
```

注記 : インスタンスレーションは、Vivado 言語テンプレートでも使用できます。

7 シリーズ デバイスの Verilog インスタンス化テンプレートは次のとおりです。

```
USR_ACCESSE2 USR_ACCESS_7series_inst (
CFGCLK(CFGCLK), // Not utilized in the static use case in this application note
DATA(DATA), // 32-bit output Configuration Data output
DATAVALID(DATAVALID) // Not utilized in the static use case in this application note
);
```

付録 B: ビットストリーム構成

ビットストリーム内の USR_ACCESS レジスタ値を見つけるには、次のコマンドを探します。

```
Type 1, Write command, address 01101, 1 word

00110000000000011010000000000001 - 0x3001A001
```

コマンドの後にある 32 ビット値が USR_ACCESS レジスタの値です。コンフィギュレーションポートを介して、これらの値へ読み出し/書き込みを実行する際の構文については、各デバイスのコンフィギュレーション ユーザー ガイド [参照 1] または [参照 2] にある「コンフィギュレーションの詳細」を参照してください。

タイムスタンプ値を含む Kintex-7 デバイスのビットストリーム (ロービット ファイル [.rbt] フォーマット) の一部を示します。

```
001100000000000100010000000000001
000000000000000000000000000000000
001100000000000110100000000000001 ← タイプ 1、書き込み、アドレス 01101、1 ワード
01010101000111001111011001100001 ← USR_ACCESS 値
001100000000000100110000000000001
000000000000000000000000000000000
```

上記の例には、次のタイムスタンプ値が含まれています。REGISTER.USR_ACCESS:0x551cf661

32 ビット USR_ACCESS 値を分解すると、2014 年 10 月 10 日午後 3 時 25 分 33 秒のタイムスタンプが得られます。

```
01010 1010 001110 01111 011001 100001
```

説明:

「日」を示す 5 ビット : 01010 = 10 番目の日
「月」を示す 4 ビット : 1010 = 10 番目の月
「西暦」を示す 6 ビット : 001110 = 14 番目の年 (2014 年)
「時」を示す 5 ビット : 01111 = 15
「分」を示す 6 ビット : 011001 = 25
「秒」を示す 6 ビット : 100001 = 33

参考資料

- 『UltraScale アーキテクチャ コンフィギュレーション ユーザー ガイド』(UG570: [英語版](#)、[日本語版](#))
- 『7 シリーズ FPGA コンフィギュレーション ユーザー ガイド』(UG470: [英語版](#)、[日本語版](#))

改訂履歴

次の表に、この文書の改訂履歴を示します。

日付	バージョン	内容
2016年3月3日	1.0	初版

法的通知

本通知に基づいて貴殿または貴社(本通知の被通知者が個人の場合には「貴殿」、法人その他の団体の場合には「貴社」。以下同じ)に開示される情報(以下「本情報」といいます)は、ザイリンクスの製品を選択および使用することのためにのみ提供されます。適用される法律が許容する最大限の範囲で、(1)本情報は「現状有姿」、およびすべて受領者の責任で (with all faults) という状態で提供され、ザイリンクスは、本通知をもって、明示、黙示、法定を問わず(商品性、非侵害、特定目的適合性の保証を含みますがこれらに限られません)、すべての保証および条件を負わない(否認する)ものとします。また、(2)ザイリンクスは、本情報(貴殿または貴社による本情報の使用を含む)に関係し、起因し、関連する、いかなる種類・性質の損失または損害についても、責任を負わない(契約上、不法行為上(過失の場合を含む)、その他のいかなる責任の法理によるかを問わない)ものとし、当該損失または損害には、直接、間接、特別、付随的、結果的な損失または損害(第三者が起こした行為の結果被った、データ、利益、業務上の信用の損失、その他あらゆる種類の損失や損害を含みます)が含まれるものとし、それは、たとえ当該損害や損失が合理的に予見可能であったり、ザイリンクスがそれらの可能性について助言を受けていた場合であったとしても同様です。ザイリンクスは、本情報に含まれるいかなる誤りも訂正する義務を負わず、本情報または製品仕様のアップデートを貴殿または貴社に知らせる義務も負いません。事前の書面による同意のない限り、貴殿または貴社は本情報を再生産、変更、頒布、または公に展示してはなりません。一定の製品は、ザイリンクスの限定的保証の諸条件に従うこととなるので、<http://japan.xilinx.com/legal.htm#tos> で見られるザイリンクスの販売条件を参照して下さい。IP コアは、ザイリンクスが貴殿または貴社に付与したライセンスに含まれる保証と補助的条件に従うこととなります。ザイリンクスの製品は、フェイルセーフとして、または、フェイルセーフの動作を要求するアプリケーションに使用するために、設計されたり意図されたりしていません。そのような重大なアプリケーションにザイリンクスの製品を使用する場合のリスクと責任は、貴殿または貴社が単独で負うものです。<http://japan.xilinx.com/legal.htm#tos> で見られるザイリンクスの販売条件を参照してください。

© Copyright 2016 Xilinx, Inc. Xilinx, Xilinx のロゴ、Artix、ISE、Kintex、Spartan、Virtex、Zynq、およびこの文書に含まれるその他の指定されたブランドは、米国およびその他の各国のザイリンクス社の商標です。すべてのその他の商標は、それぞれの所有者に帰属します。

この資料に関するフィードバックおよびリンクなどの問題につきましては、jpn_trans_feedback@xilinx.com まで、または各ページの右下にある [フィードバック送信] ボタンをクリックすると表示されるフォームからお知らせください。いただきましたご意見を参考に早急に対応させていただきます。なお、このメールアドレスへのお問い合わせは受け付けておりません。あらかじめご了承ください。