第 21 章　プログラマーとユーティリティ

本章ではデバイスの書き込みツールであるプログラマー（Programmer）と、付随するユーティリティ・ツールについて記述します。本節は Diamond 組み込みの場合についてのオペレーションを前提に記述しています。スタンドアローン版ではオペレーションや表示が全く同じではない場合がありますのでご留意ください。

21.1 基本的な書き込み手順とオペレーション

FPGA デバイスのプログラミング、またはコンフィグレーション（以下『書き込み』という表記と併用）の ” 初回の ” 基本手順は次の通りです。ここでデバイスへの書き込み用ファイル（.bit / .jed、ビットストリーム、パターン、イメージなどとも表現される）は生成済みのものとします。

1. ダウンロードケーブルをターゲットボードに接続
2. プログラマーの起動とケーブルの指定・検出
3. デバイススキャン（JTAG チェイン内のデバイス検出）
4. デバイスへの書き込みファイルとアクセスモード、オペレーションを指定
5. 書き込み（設定したオペレーション）の実行（選択オペレーションによってはベリファイまで）
6. チェインファイルの保存

一度作業を終了する前に、後述のチェインファイル ”<file name>.xcf” を保存しておくと、二回目以降同じ（JTAG）チェイン構成の場合は、ステップ 2 の後に xcf を呼び出すことでステップ 3 と 4 は省略できます。次節以降、それぞれのステップについて詳細を記述します。

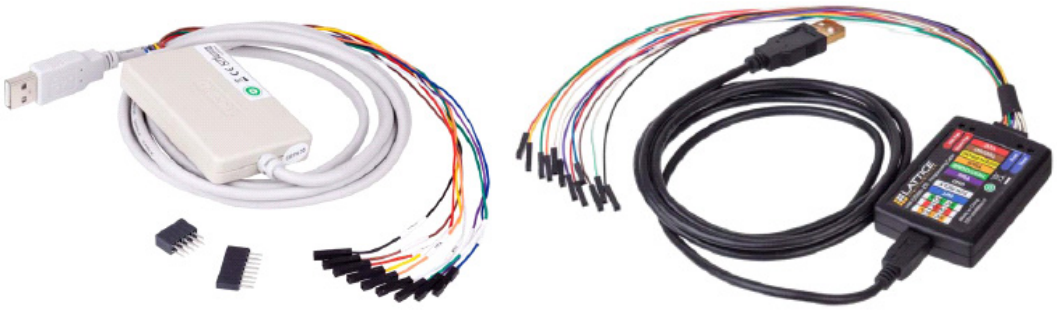
なお、ここで ” プログラミング ” とはオンチップ・フラッシュメモリや SPI フラッシュメモリに対する書き込み、” コンフィグレーション ” はデバイス内部コンフィグレーション SRAM への書き込み、という意図であり、書き込み対象によって用語の使い分けをしています。これはラティスのデータシートやテクニカルノート全般で共通です。

21.1.1 ダウンロードケーブルの接続

書き込みの準備として、まずダウンロードケーブルをターゲットボードに接続します。最新のダウンロードケーブルは USB 対応 HW-USBN-2B です（種類や詳細についてはユーザガイド『UG48 Programming Cables』をご参照ください）。図 21-1 に、最新 USB ダウンロードケーブル（右）と、最も出荷数の多いその前バージョンの USB ケーブル（左）の外観図を示します。

図 21-1. USB ダウンロードケーブル外観

HW-USBN-2A　　HW-USBN-2B（最新）



ケーブルドライバが正しくインストールされていない場合や使用ケーブルタイプの設定が異なると、使用する PC 環境で最初に実行する場合にエラーやウォーニングが表示されます。通常は Diamond のインストール時にドライバのインストールを促すプロンプトが表示されますので、インストール作業が正常終了すれば、用意が整っているはずです。問題がある場合は、21.xx 節をご参照ください。

MachXO シリーズの機能として、プログラマーを用いて PC の USB ポートからデバイスのハードマクロ EFB が備える SPI や I2C ポートを介してのプログラミング・アクセスが可能です。HW-USBM-2A では SPI ポートからのアクセスのみが可能です。HW-USBM-2B は Diamond 3.1 以降でサポートされており、I2C ポートからのアクセス機能もサポートするようになっています。

21.1.2 プログラマーの起動

プログラマーの起動には幾つかの方法があります。

① Windows スタートメニューから起動

[スタート] → [すべてのプログラム] → [Lattice Diamond 3.x] → [Accessories] → [Diamond Programmer] と選択して起動

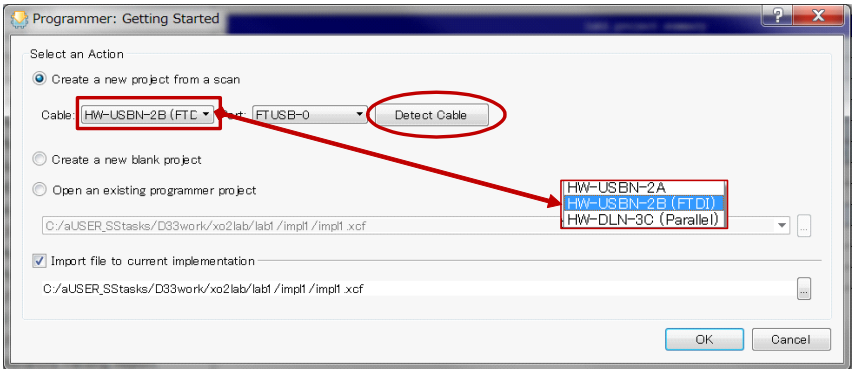
② Diamond を立ち上げてから Programmer を指定して起動

プログラマのアイコン  をクリックするか、”Tools” メニューバーから ”Programmer” を選択して起動

③ Diamond の File List 内の ”Programming Files” セクションに（アクティブな）既存のチェインファイル（.xcf）がある場合、これをダブルクリックして起動

プロジェクトで初めて起動した時は図 21-2 のような画面が立ち上がります。

図 21-2. プロジェクトで初めてプログラマーを起動した後の表示



21.1.3 ケーブルの指定・検出

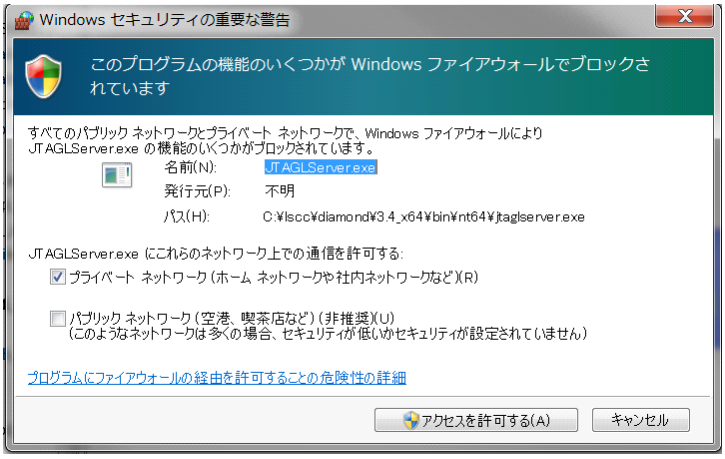
図 21-2 において、初回は『Create a new project from scan』ボタンが選択されています。ケーブルがボードに接続されていないとコンソール（プロジェクトナビゲータ下部の『Output』窓枠）にエラーメッセージが出力されます。

”Cable” 部で選択可能なケーブルタイプは三つあります（図 21-2 内の右）が、プルダウンで直接指定することができます。或いは『Detect Cable』ボタンをクリックすることで、ケーブルタイプを自動判別する（スキャンする）ことも可能です。一旦プログラマー・プロジェクトを作成後に変更することも勿論可能です。

『Create a new blank project』を選択した場合で、作業しているインプリメンテーション下に既存の xcf ファイルが存在する場合、デフォルトで最下部の『Import file to current implementation』にチェックが入っています。インポートしない場合は、チェックをはずします。

初めて実行した際に、図 21-3 のような Windows の警告メッセージが出る場合がありますが、右下『アクセスを許可する』をクリックして先に進みます。

図 21-3. 初回アクセス時の Windows 警告メッセージ



検出が完了するとその旨を通知する図 21-4 のようなメッセージが出ますので、”OK” をクリックして抜けます。

図 21-4. 自動検出アクションでの通知画面

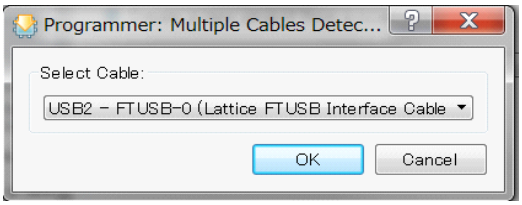
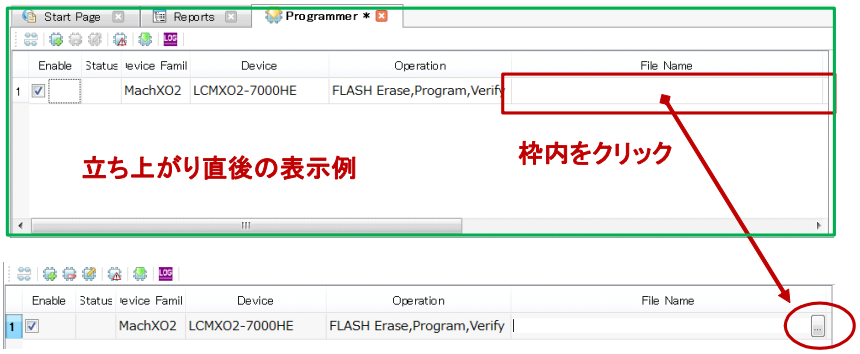


図 21-5 は『Create a new blank project』を選択した場合の起動後の表示例です。『File Name』欄に作業中のインプリメンテーションで生成した書き込みファイル（.bit, .jed）が自動的に選択・表示されますが、本図のように『File Name』セルがブランクの場合は、まず枠内をクリックします。これによって、枠の右端にブラウズ操作ができるようにボタンが表れますので（図 21-5 内の下部）、これをクリックしてターゲットファイルを選択します。

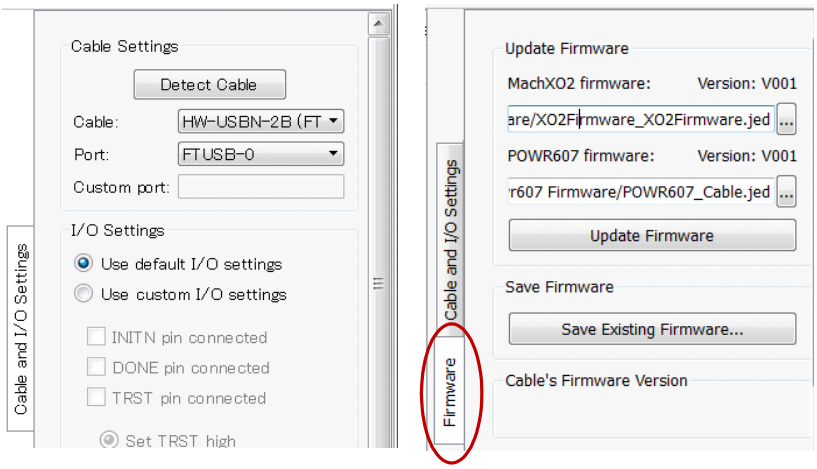
図 21-5. ブランクプロジェクトとして立ち上がり直後の表示画面



なお、プロジェクト起動後でもケーブル設定を変更可能です。プログラマー起動後のウィンドウで右側にのような表示セクションがあります（図 21-6 左）。図 21-2 と同様の操作ができるようになっています。次項で説明するスキャン動作が不成功の場合、本セクションで正しい設定に変更することが可能です。

また、図 21-6（右）に示すように Diamond 3.2 以降は『Firmware』タブが追加になっています。ケーブルが ”HW-USBN-2B” の場合で、ファームウェアが最新でない場合に限り、本タブが表示されますので、容易に更新作業が可能です。

図 21-6. ケーブル設定変更セクション（左）とファームウェア更新タブ（右）



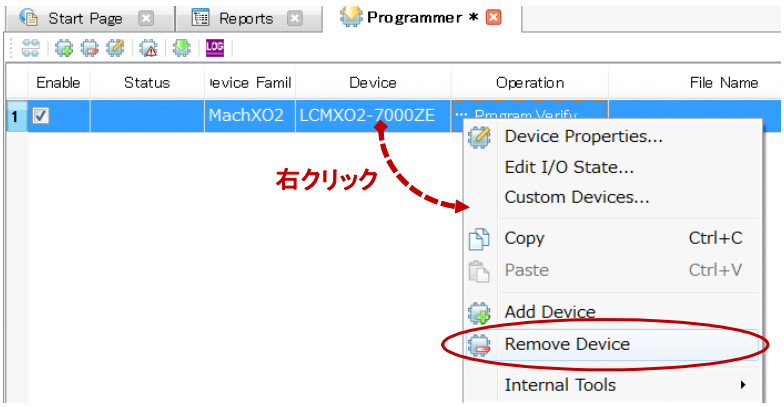
21.1.4 スキャン（デバイスの検出）

新規デザインの書き込みや、ターゲットボードが新規または変更後の（後述するチェインファイル xcf を作成済みでない）場合などの操作です。

前項 21.1.3 で記述した『Create a new project from scan』で立ち上げた場合、図 21-5 の『Device Family』および『Device』欄は同時に実行されるスキャン結果を反映して表示されますので、スキャン操作は不要です。

なお、デバイスの識別はデバイスから『Device ID』を読み出すことで行います。同一ファミリの派生品間で ID 番号を共有するケースでは、厳密に正確な Device 表記にならない事があります（例えば MachXO2-7000HE に対して MachXO2-7000ZE など）。こうした場合、注意を促す意味でデバイスのセルが黄色で表示されます。セルをクリックすると正しいデバイスの候補がプルダウン表示されますので、選択します。

図 21-7. デバイスの削除



JTAG チェインが複数デバイスよりなる場合や、例えば『Create a new blank project』を選択・起動後の初期表示デバイスと異なる作業をする場合（当該インプリメンテーションと直接関連のないボードやデバイスのプログラミングの場合）など、意図する行のどこかで右クリックして当該行を削除し（図 21-7）、その後スキャンすることも可能です。

図 21-8. スキャンアイコン

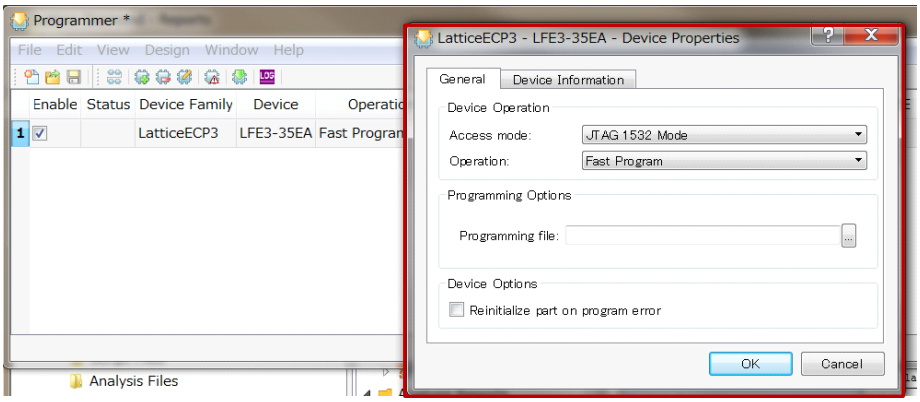
スキャンは、プログラマー内上部のアイコンをクリックする（図 21-8）か、プログラマーをデタッチして、メニューの [Design] → [Scan] を選択します。スキャンが正常に行われると（チェインに接続されている）

デバイス名が表示されます。他社デバイスは全て JTAG-NOP と表示されます。

21.1.5 書き込みファイルとアクセスモードの設定

まず**アクセスモードとオペレーションを指定**します。当該デバイス行の『Operation』セルか、行のどこかをダブルクリックすると、図 21-9 のような設定ウィンドウが立ち上がります（LatticeECP3 の例）。デバイスファミリによって『Access mode』（”アクセスモード”）と『Operation』（”オペレーション”）の（初期）表示項目は異なります。

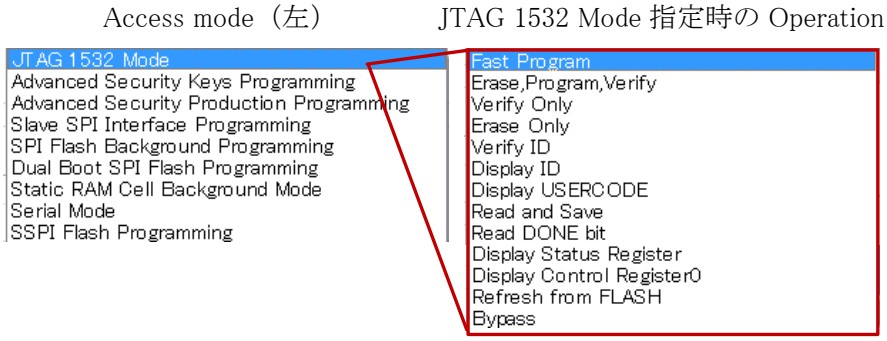
図 21-9. LatticeECP3 の場合の初期画面



LatticeECP3 ファミリの場合の Access mode 選択肢と”JTAG 1532 Mode”指定時の Operation 選択肢を、図 21-10 に例として示します。”JTAG 1532 Mode”以外のモードを選択した場合、それぞれのアクセスモードで個別のオプションがありますが、個々のオペレーションの意味はほぼ自明ですのでここでの説明は割愛します（本章末尾を参照）。

なお、Diamond 3.2 以降では図 21-9 の画面で『Device Information』タブがあります。デバイスの各属性情報が確認できます。

図 21-10. LatticeECP3 の場合の各オプション（一部）



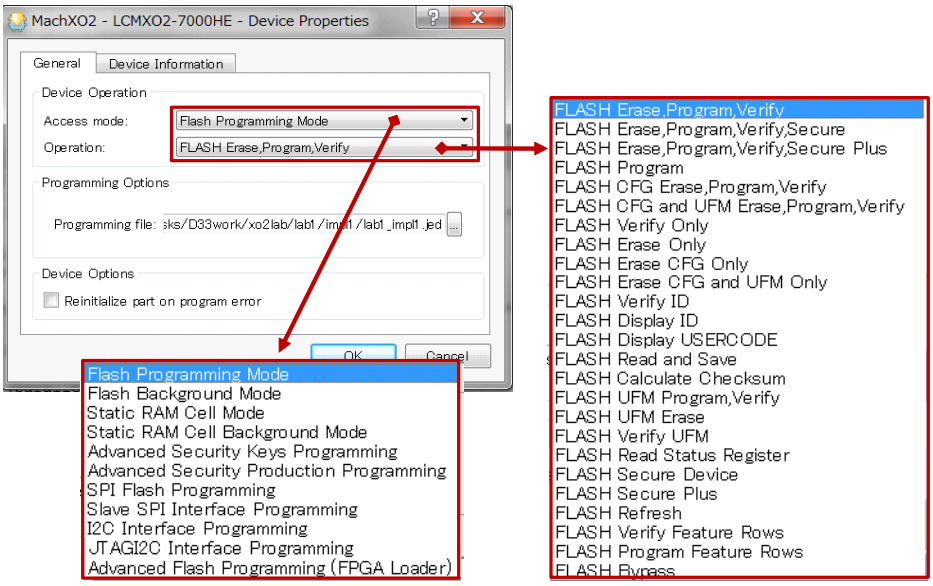
同様に MachXO2 ファミリの場合の例を図 21-11 に示します。

次に**書き込みファイルの指定**を行います。起動・スキャン後に『File Name』セルがブランクの場合、図 21-5 のように Data File セルをクリック後ブラウズボタンを押して所望のファイルを指定するのが一般的です。LatticeECP シリーズの場合は、拡張子が bit（または rbt）のファイルを、MachXO シリーズの場合は、拡張子が jed（または isc）のファイルを指定します。

或いは、図 21-9 の設定ウィンドウを抜ける時に、中央部の”Programming File”欄にブラウズして指定することも可能です。この場合、指定ファイルがターゲットデバイス用の書き込みファイルでない場合は、エラーメッセージが直ちに表示されます。プログラマーの Data File セルで指定した場合は、オペレーション（書き込み）実施時になってようやく誤りがあれば通知されます。

図 21-11. MachXO2 ファミリの場合の各オプション（一部）

Access mode（左下）　　Flash Programming Mode 指定時の Operation（右）



書き込みファイルを指定すると、『File Date/Time』『Checksum』『USERCODE』の各セルがファイルから読み取られた情報で埋められます。次項で説明する xcf ファイルで既存プロジェクトを起動した場合などで、かつ同名称の書き込みファイルを繰り返し用いる場合には、特にタイムスタンプ情報は最新かどうかを確認する手助けとなります。

リードバックしてファイルに保存するオペレーションでは、”Programming file” の欄は保存するファイル名にしておきます。

なお、コンフィグレーション用書き込みファイルの標準的なファイルタイプは以下の通りです。

.jed　MachXO シリーズ・オンチップフラッシュなど

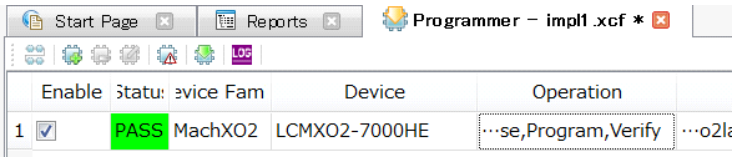
.bit　外付け SPI フラッシュ（単独）、LatticeECP シリーズオンチップ SRAM、MachXO3L NVCM、iCE40 シリーズ NVCM など

.mcs 外付け SPI フラッシュ（デュアルブート）

21.1.6 書き込み（オペレーション）の実行

各種設定が完了しましたのでメニューアイコン列から  をクリックして、書き込み（設定したオペレーション）を実行します。プログラマーをデタッチし、メニューで [Design] → [Program] と選択することでも可能です。

図 21-12. オペレーション正常終了の例



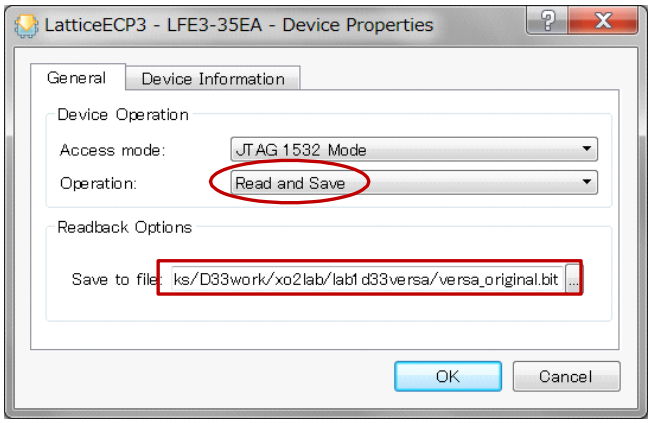
プログラマーを実行後、『Status』セルに緑色で PASS（もしくは黄色で DONE）と表示されれば、プログラミング（オペレーション）が正常に終了したことを示します。FAIL（赤）が表示された場合には、何らかの理由により、オペレーションが失敗しています。オペレーション失敗の原因が不明の場合は、ログファイルにヒントがある場合がありますのでチェックします。ログファイルのクリアはデタッチ状態で [Design] → [Clear Log File] と指定して行います。

21.1.7 Read & Save（オペレーション）の実行

生成した書き込みファイルをプログラミングする前に、直前の状態で FPGA に書き込まれているデータをリードバックしてファイルに保存したい場合があります。この場合のオペレーションが ”Read and Save” です。図 21-13 に LatticeECP シリーズの場合の例を示します。

リードバック後に保存するファイル名を入力し、アイコン をクリックして実行します。プログラマーのステータスが ”DONE” となれば完了です。

図 21-13. Read and Save オペレーション



21.1.8 チェインファイルの保存

作業したプログラマー・プロジェクトは JTAG チェイン情報を含む一連の設定であり、”チェインファイル” (拡張子 xcf) として保存できます。保存アイコン をクリックするか、デタッチ状態でメニューから [File] → [Save <Implementation 名 >.xcf] または [Save <Implementation 名 >.xcf as ...] を選択します。保存したチェインファイルは、Diamond の File List ウィンドウ内 ”Programming Files” セクションに自動的にアクティブなチェインファイル（太字）として取り込まれます。次回以降ダブルクリックして同じ設定で起動可能です。

21.2 外付け SPI フラッシュメモリのプログラミング

ラティスの FPGA ではコンフィグレーション用やデュアルブート機能対応のために外部 SPI フラッシュメモリを接続できます。本節ではこれらに関するオペレーションについて記述します。

21.2.1 コンフィグレーション用外部 SPI フラッシュ（LatticeECP シリーズ）

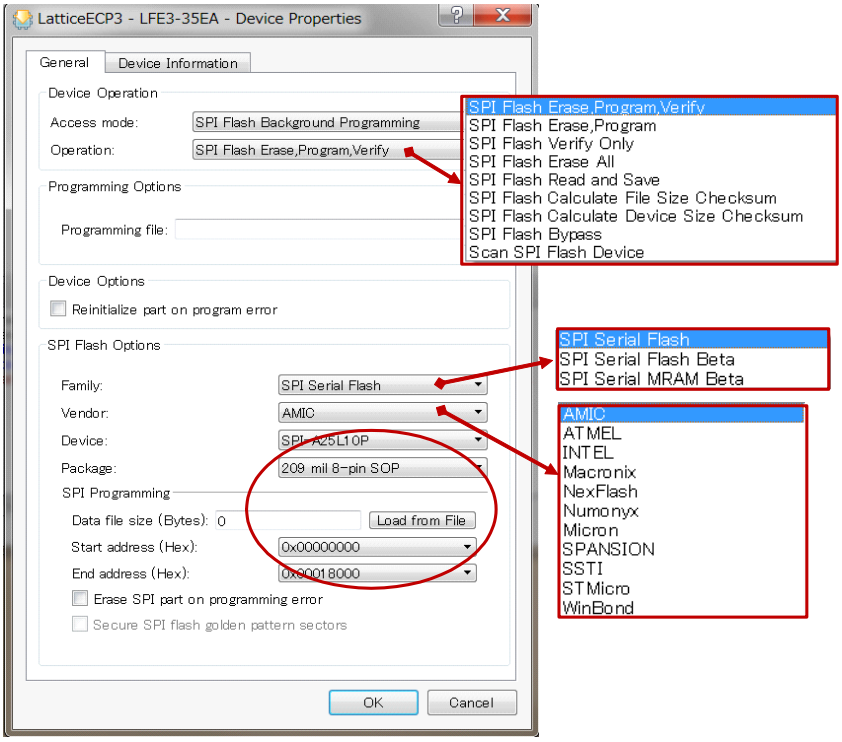
オンチップにコンフィグレーション SRAM のみを集積する LatticeECP シリーズ（LatticeECP2M / ECP3 / ECP5）では、外付け SPI フラッシュメモリからコンフィグレーション・データをロードして起動するモードが最も良く利用されます。

このような場合、オンボードの SPI フラッシュメモリを FPGA 経由で JTAG ポートからプログラミング（書き換え）することができます。図 21-14 は LatticeECP3 の場合です。プログラマーで『Device Operation』を設定する際に、[Access mode] 行で ”SPI Flash Background Programming” を選択し、[Operation] 行では図中に示す選択可能なオペレーションからどれかを指定します。例えば書き換える場合は ”SPI Flash Erase, Program, Verify” の一連の動作を指定します。

次にオンボードで接続されている SPI フラッシュメモリを『SPI Flash Options』部で指定します。[Family] 行で、”SPI Serial Flash” がラティスで機能検証済みのメモリ一覧から選択する場合、”SPI Serial Flash Beta” はツールに登録されているもののラティスにて実デバイスでの機能検証が完了していないメモリから選択する

場合です。後者は機能検証が完了次第 Diamond の後のバージョンで ”Beta” から移動する可能性があります。プログラマの機能動作としては同一です。

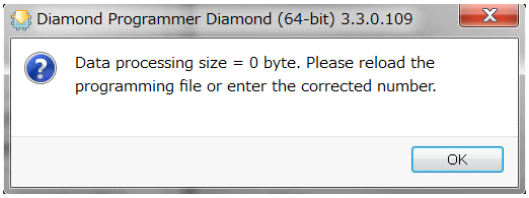
図 21-14. オペレーションと SPI フラッシュメモリの選択



次いで [Vendor] を選択後、適切な [Device] と [Package] を指定します。[Data file size (Bytes)] 行は上部にある [Programming file] を指定後、（ファイルを指定しても自動で読み出されない場合）『Load from File』ボタンをクリックすることで入力します。このセルがブランクのまま『OK』ボタンを押すと、図 21-15 のようなメッセージが表示されて先に進むことができませんので、ご留意ください。

また bit ファイルが指定した SPI フラッシュメモリに収まらないとか、SPI フラッシュメモリの型番が不一致の場合も、それぞれエラーメッセージが出力されますので、入力して解消するようにします。

図 21-15. ”Data file size” セルが値 0 の場合のメッセージ



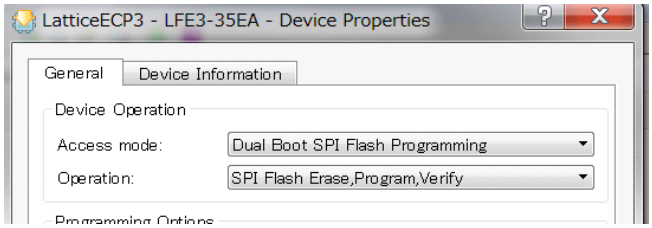
なお、次節で記述するデュアルブートモードで、一方のビットストリームのみを書き換える場合、”Start Address” に当該先頭アドレスを設定します。この情報が正しくないと正常に起動しないなどの障害となりますので、ご留意ください。

21.2.2 デュアルブート用外部 SPI フラッシュ（LatticeECP シリーズ）

LatticeECP シリーズでは外付け SPI フラッシュメモリに二本のコンフィグレーション・データ（ゴールデン・ビットストリームとプライマリ・ビットストリーム）を保存する、いわゆるデュアルブート機能をサポートします。また、デバイスファミリによって、両ビットストリームのストアするセクタ（開始アドレス）やジャンプコマンド（0x00400000）のストアされるセクタが異なります。詳細はテクニカルノート TN1216 などをご参照ください。

プログラマーのオペレーションは図 21-16 のように ”Dual Boot SPI Flash Programming” を選択します。

図 21-16. デュアルブート・オペレーションの選択



ここで、書き込みファイル（”PROM Hex” ファイルとも呼ばれる）を事前に作成しておく必要があります。これはユーティリティであるデプロイメント（Deployment）ツールを使用します。21.4 節をご参照ください。

なお、Diamond 3.1 以降では、Deployment Tool でマルチブート『Multiple Boot』もサポートされています（LatticeECP5）。

21.2.3 デュアルブート用外部 SPI フラッシュ（MachXO シリーズ）

MachXO シリーズもデュアルブート・モードをサポートしますが、SRAM ベースの LatticeECP シリーズのように外付けフラッシュメモリに 2 本のビットストリームを保持する機構とは異なります。オンチップ不揮発メモリと外付け SPI フラッシュメモリにそれぞれ 1 本づつのビットストリームを保存しておき、マスター SPI モードと SDM モードの組み合わせで実現しています。オンチップメモリには MachXO2 の場合がプライマリ、MachXO3L の場合はゴールデンが保存されます。起動は必ずプライマリからで、これらの順序をユーザが変える手段はありません。

MachXO シリーズの場合、『Operation』の [Access mode] 行（図 21-11 の左下を参照）で選択するのが ”SPI Flash Programming” であること、および外部 SPI フラッシュメモリにおくイメージの開始アドレスが ”0x010000” でなければならないことを除いて、SPI フラッシュメモリの指定方法などは 21.2.1 項の LatticeECP シリーズと同様です。

なお、Diamond のスプレッドシートビューなどで指定するグローバル制約（Global Preference）のコンフィグレーション指定は、MachXO3L の場合（MASTER_SPI_PORT = ENABLE）、

デュアルブート～ CONFIGURATION = CFG、（TN1279、Dual Boot 項）

マスター SPI モード（単一イメージのみを外部 SPI に置く）～ CONFIGURATION = EFB_USER

MachXO2 の場合、

デュアルブート～ CONFIGURATION = CFG、CFG_EBRUFM、CFGUFM のどれか（TN1204、Dual Boot 項）

マスター SPI モード（単一イメージのみを外部 SPI に置く）～ CONFIGURATION = EFB_USER

なお、MachXO シリーズの場合、SPI マクロを集積する EFB（組み込みファンクションブロック）は SPI マスターになります。クロック**信号 spi_clk** は出力（**MCLK**）になり、外部プルアップ抵抗 1kΩ を負荷することが推奨されています。この際に抵抗をプルダウンにすると、プログラマーからのプログラミングが失敗しますので、ご留意ください。**外部抵抗は必ずプルアップ**です。それぞれテクニカルノートの ”Hardware Checklist” にも明記されていますので、ご参照ください。

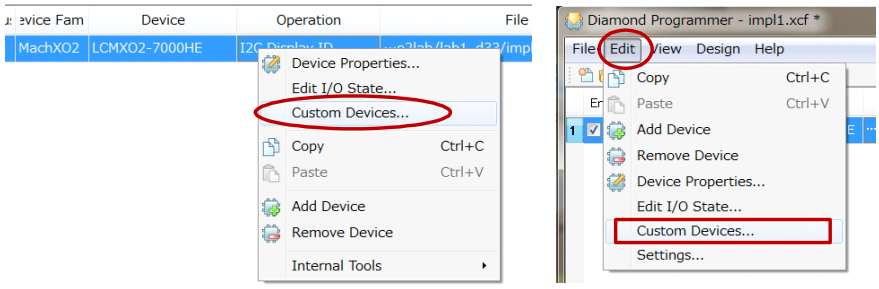
21.2.4 ”カスタム ”SPI フラッシュメモリの登録・使用

プログラマーの『SPI Flash Options』部で指定する SPI フラッシュメモリ候補に、意図するデバイスがない場合に対応するための機能です。最初にカスタム品として手元で登録した後に使用できるようになります。

意図するデバイス行を選択後右クリックするか（図 21-17、左）、デタッチしたプログラマーで［Edit］メニューから（図 21-17、右）、いずれも ”Custom Devices ...” を選択します。

図 21-17. SPI フラッシュメモリ登録ウィンドウの呼び出し

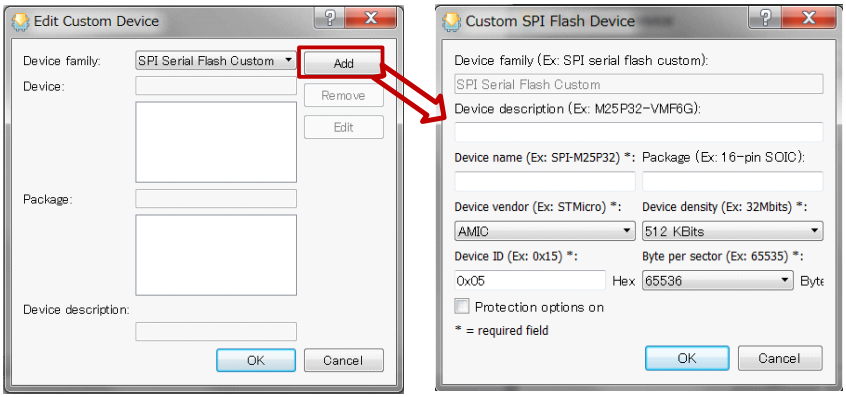
右クリックから　　メニューから



表示されるウィンドウで最初に Add ボタンをクリックし（図 21-18、左）、次いで適宜名称の登録と詳細なパラメータを選択・入力します（後に呼び出す場合に判別がつくような登録名称とするようにします）。

図 21-18. SPI フラッシュメモリ登録ウィンドウ

初期画面　　詳細登録画面



21.2.5 SPI メモリ内にユーザデータを共存

外付け SPI フラッシュメモリ内に、コンフィグレーションデータとユーザデータを共存することが可能です。この場合、特段に留意するべき事項はなく、以下の基本的な点を押さえるのみです。

- 使用するデバイス（ファミリと規模）によってビットストリーム・ファイルや、デュアルブート時のジャンプコマンドの開始アドレスは一義的に決まるセクターは避ける
- バイナリかテキスト（Hex）ファイルかで LSB – MSB 順が逆になる

本機能に関してラティスが提供するツールはありませんので、いずれにしろ SPI アクセスするためにファイル処理手段は、論理回路にしろ組み込みプログラムにしろ、ユーザが作成することになります。

21.3 拡張・特定機能

21.3.1 各種 ID 値の設定（User Code / TraceID / Custom ID Code）

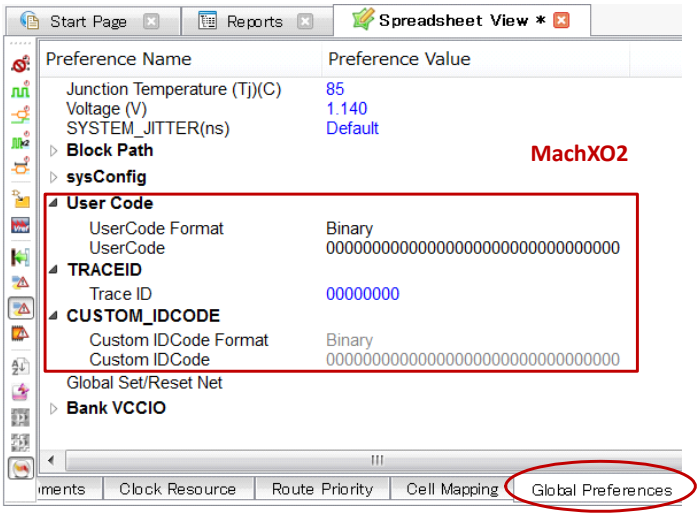
ラティス FPGA では、デバイス内にユーザー固有のコードを格納することが可能です。は MachXO2 の例で、スプレッドシートビューの『Global Preference』タブを示します。User Code（USERCODE）、TraceID、そして Custom ID Code がデザインのコンパイル時に設定可能です。これら ID 機能はデバイスファミリによって異なり、LatticeECP3 では User Code のみが対応しています。

User Code は 32 ビットで、例えば実装するデザインのバージョン番号など、履歴管理や任意のコードを設定します。MachXO シリーズの TraceID は 64 ビットで、そのうち 56 ビットはラティス出荷時に製造上の固

有の値が入ります（個体レベルで識別できるものです）。残りの 8 ビットをユーザが任意に用いることが可能です。

そして同様に MachXO シリーズの Custom ID Code は図 21-19 の Global Preference 内の ”sysConfig” 項に含まれる『MY_ASSP』オプションが ON の時に有効になります。21.1.4 項で言及した ”Device ID” は標準品では規定の値ですが、MY_ASSP=ON 時はこの Custom ID Code がこれに置き換わります。

図 21-19. スプレッドシートビューによるの各 ID 値設定（MachXO2）



上記スプレッドシートビューによる設定以外に制約ファイル <file name>.lpf での指定も可能です。

USERCODE ASCII ″abcd″ ;
TRACEID ″01010101″ ;
CUSTOM_IDCODE HEX ″13579bdf″ ;

このような Diamond の標準フローの一部として設定した ID 情報は、Diamond が生成する書き込みファイルに反映されます。これに対して生成済みファイルやプログラム済みデバイスの ID 情報を書き換える場合は、ユーティリティツールを用いることができます。USERCODE は『USERCODE エディタ』での編集を（21.4.3 項参照）、TraceID および Custom ID Code の編集は『FEATURE ROW エディタ』（21.4.2 項参照）がサポートしています。

21.3.2 I2C・SPI ポートからのプログラミング（MachXO シリーズ）

MachXO2/3L ファミリでは、ハードマクロ EFB の備える I2C（プライマリ）ポートや SPI ポートから、PC の USB 2.0 ポートを介してプログラマーを用いたプログラミングが可能です。本項ではケーブルとして **HW-USBN-2B を用いる前提**で記述します。

21.3.2.1 ケーブルの接続

本ケーブルをフライワイヤでターゲットボードに接続しますが、I2C / SPI に関わる信号線のみの割り当てを以下に示します（電源・グランドも省略）。詳細は『UG48 Programming Cables』をご参照ください。

表 21-1. プログラミングケーブルのピンと定義

信号	ケーブルでの名称	線材の色	PC からの方向	XO2/XO3L ピン	信号の説明
Test Data Output	SDO/TDO	茶	Input	SPISO	SPI、データ入力
Test Data Input	SDI/TDI	橙	Output	SISPI	SPI、データ出力
Enable	ispEN/Enable/PROG/SN	黄	Output	SN	SPI、チップセレクト
Test Clock	SCLK/TCK	白	Output	MCLK	SPI、クロック
I2C Clock	SCL（HW-USBN-2B のみ）	白・黄ストライプ	Output	SCL	I2C、クロック
I2C Data	SDA（HW-USBN-2B のみ）	白・緑ストライプ	Output	SDA	I2C、データ

21.3.2.2 グローバル制約の設定

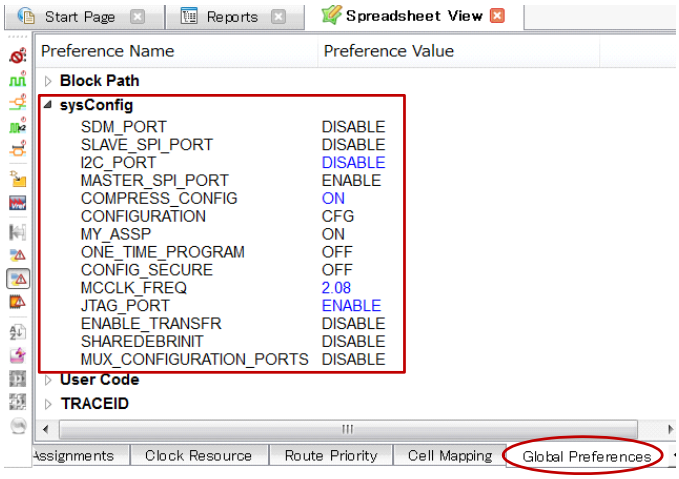
これら機能を有効にするためには、コンフィグレーション関連グローバル制約の 1 つであるオプションをそれぞれイネーブルしたビットストリームを生成・プログラミングしておく必要があります。

I2C アクセス　I2C_PORT = ENABLE

SPI アクセス　SLAVE_SPI_PORT = ENABLE

スプレッドシートビューで設定すると制約ファイル <file>.lpf に書き出されますので、手編集も可能です。MachXO3L に ”sysCONFIG” セクション下のアイテムを示します。

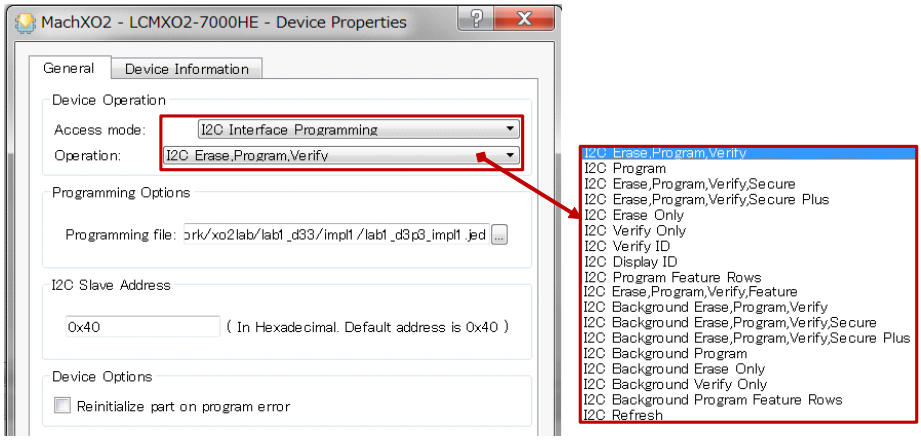
図 21-20. スプレッドシートビューの sysCONIFIG 部制約アイテム（MachXO3L）



21.3.2.3 プログラマーのオペレーション

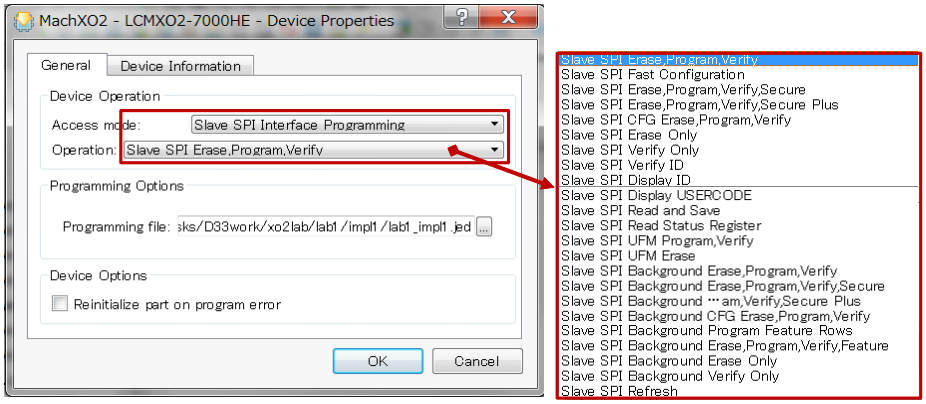
これまでと同様にプログラマーを立ち上げ、アクセスモードとオペレーションを設定します。I2C ポート経由の場合を図 21-21 に、SPI ポートの場合を図 21-22 に示します。

図 21-21. I2C ポートのアクセス・オペレーション



指定するアクセスモードは ”I2C Interface Programming” です。オペレーションは適宜意図するアクションを選択します。

図 21-22. SPI ポートのアクセス・オペレーション



指定するアクセスモードは”Slave SPI Interface Programming”です。オペレーションは適宜意図するアクションを選択します。

21.3.3 暗号化キーと暗号化済み書き込みファイルのプログラミング

本機能は LatticeECP シリーズのみの対応で、MachXO シリーズはサポートしていません。

まず、書き込みファイル（ビットストリーム）の暗号化を実行するには通常の Diamond パッケージに加えて暗号化機能用のパッケージ（Encryption Package）が必要です。ユーザがラティスのウェブサイトからまずリクエストに対する許可の申請をします。その後ダウンロード先 URL を含むメールが送付されてくるので、ダウンロード、インストールを済ませます。こうした手順は米国の法律に基づいた対処であり、必ず必要です。日本国内のユーザであれば、問題なく許可されるはずです。

暗号化パッケージをインストールしていないと。本節で紹介しているようなスクリーンダンプ例が得られませんのでご留意ください。本項では、パッケージのインストールが完了しているものとして、暗号化キーと暗号化されている書き込み（ビットストリーム）ファイルのプログラミングについて記述します。ビットストリーム・ファイルの暗号化手順についてはデプロイメント・ツールで記述します（21.4.1.3 参照）。書き込みデータの暗号化処理自体は、Diamond 本体のフローやパラメータ設定ではサポートされていません。

ちなみに、コンフィグレーションメモリが揮発性のこれらファミリにおいて、唯一暗号キーを保持するオンチップメモリのみ不揮発性であり、かつ OTP（一度のみプログラム可能）です。すなわちキーを一度プログラムした後は変更（書き換え）ができませんので、ご注意ください。

図 21-23. 暗号化対応パッケージのダウンロード先

Diamond 3.3 32-bit Encryption Pack for Linux	3.3	10/6/2014		714 KB
Diamond 3.3 32-bit Encryption Pack for Windows	3.3	10/6/2014		2.9 MB
Diamond 3.3 32-bit for Linux	3.3	10/6/2014	RPM	1.1 GB
Diamond 3.3 32-bit for Windows	3.3	10/6/2014	ZIP	1.5 GB
Diamond 3.3 64-bit Encryption Pack for Linux	3.3	10/6/2014		720 KB
Diamond 3.3 64-bit Encryption Pack for Windows	3.3	10/6/2014		2.9 MB
Diamond 3.3 64-bit for Linux	3.3	10/6/2014	RPM	1.2 GB
Diamond 3.3 64-bit for Windows	3.3	10/6/2014	ZIP	1.5 GB
Diamond Programmer 1.3 Encryption Installer for Linux	1.3	7/18/2011		27 KB
Diamond Programmer 1.3 Encryption Installer for Windows	1.3	7/18/2011		1.5 MB
Diamond Programmer 1.3 Standalone for Linux	1.3	7/18/2011	RPM	100.4 MB
Diamond Programmer 1.3 Standalone for Linux md5 checksum	1.3	7/18/2011	MD5	0.1 KB
Diamond Programmer 1.3 Standalone for Windows	1.3	7/18/2011	ZIP	34.3 MB

暗号化したビットストリームをプログラミングをするためには、以下の二つのオプションがあります。

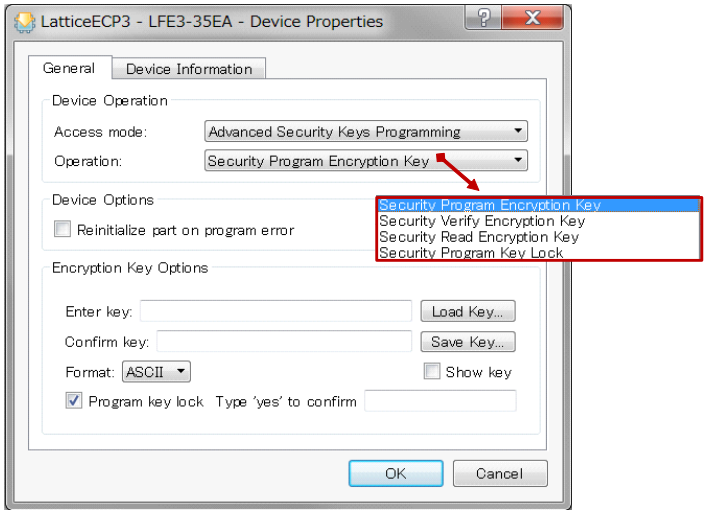
1. 暗号キーのみをプログラミングする
2. 暗号キーと、そのキーで暗号化済みの書き込みファイルを一度にプログラミングする

それぞれで指定するオプションが異なります。以下それらの手順を記述します。

21.3.3.1 暗号キーのみのプログラミング

暗号キーのみをプログラミングする場合は、プログラマーの設定で、図 21-24 に示すようにアクセスモードは [Advanced Security Keys Programming]、オペレーションは [Security Program Encryption Key] を選択します。

図 21-24. アクセスモードとオペレーションの指定



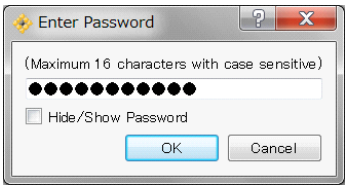
書き込み（ビットストリーム）ファイルを暗号化する際に、指定した暗号化キーをファイルに保存できます（<file name>.bek）。同ファイルが存在する場合には、『Load Key...』ボタンで読み込み指定が可能です（図 21-25 のように暗号キーファイルのパスワード入力を求められます）。キーファイルが存在しない場合、[Enter key] と [Confirm key] 欄に暗号キーを入力後、『Save Key...』ボタンをクリックすることで保存できます。

オプションと暗号キーを入力完了後、本ウィンドウを抜けて、プログラマーでオペレーションを実行します。プログラマーの Status が "PASS" と表示されれば成功です。

なお、"Program key lock" をチェック（イネーブル）した場合、暗号キーファイルをリードバックする手段は存在しません。したがって、キーが正しいかどうかは、暗号化ビットストリームをプログラムすることに

よってのみ確認できます。別の暗号キーにするか（デバイスにプログラムしていない場合）、同じ操作を実行してキーファイルを上書きする（キーではなく、キーファイル）の操作は勿論可能です。

図 21-25. 『Load Key...』をクリック後のパスワード入力画面



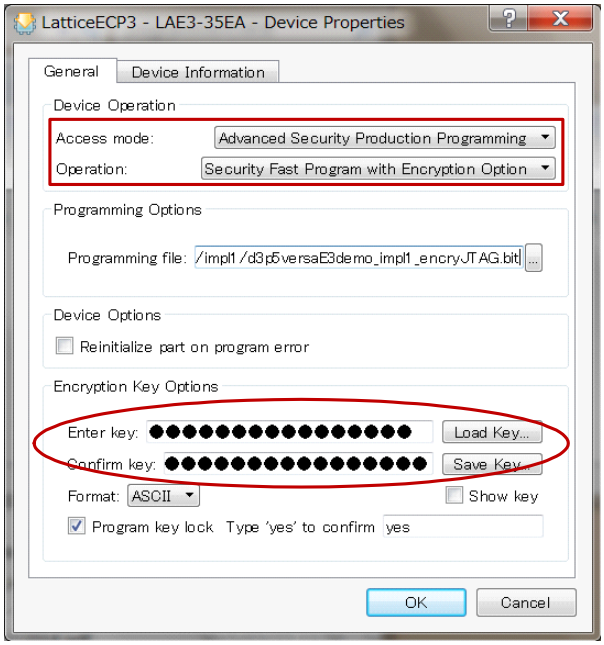
21.3.3.2 暗号化済み書き込みファイルのプログラミング / ダウンロード

暗号化済みビットストリーム（書き込みファイル）をダウンロードする場合、選択肢は二つあります。

1. まだ暗号キーをプログラムしていない場合
2. 前項で示す暗号キーをプログラム済みの場合

前者の場合、暗号キーと暗号化済みビットストリームを一度のオペレーションでプログラミング（ダウンロード）します。選択するオプションは図 21-26 のようにアクセスモードが [Advanced Security Production Programming]、オペレーションが [Security Fast Program with Encryption Option] です。

図 21-26. 暗号キーとビットストリームのプログラミング



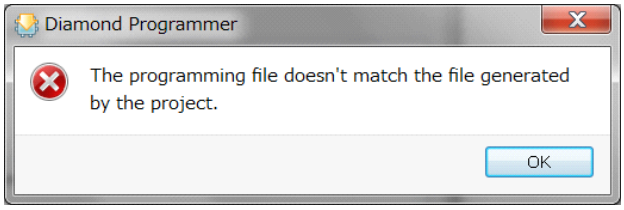
同図下部にあるように、『Load Key ...』ボタンで書き込みファイルの暗号化時に保存したキーファイルがあれば読み込む、などをして、暗号キーを同時に指定します。[Programming file:] で指定する書き込みファイルは、暗号化時にコンフィグレーション・モードを JTAG として選択しておく必要があります（21.4.1.3 項）。暗号化していない場合（図 21-27）や、モードが JTAG でない場合などはそれぞれエラーメッセージが表示されて、本ウィンドウ入力が完了できませんので、ご留意ください。

最下段の [Program key lock] は前項に記述したとおりです。

なお ECP5 では、このオペレーション完了後に別途書き込みファイル（ビットストリーム）をダウンロードする場合に、暗号化済みビットストリームのみを受け付けるオプションが追加されています。

全て設定後、プログラマーに戻りオペレーションを実行します。

図 21-27. キー未書き込みで暗号化済みファイルを指定



次に第二の場合で、暗号キーをプログラムした後に、別オペレーションで暗号化済み書き込みファイルをダウンロードするケースです。

コンフィグレーション SRAM に直接ダウンロードする際のアクセスモードは [ISC1532 Mode]、オペレーションは [Fast Program] を選択します。外付け SPI フラッシュメモリにプログラムする際にはアクセスモードが [SPI Flash Background Programming]、オペレーションは [SPI Flash Erase, Program Verify] などを指定します。

ここで、暗号化の際に指定するコンフィグレーション・モードもチェックされませんので、任意で構いません。また、LatticeECP3 ファミリの場合、コンフィグ SRAM にしろ SPI フラッシュメモリにしろ、暗号化済みでも暗号化していないファイルでもどちらでも問題なく終了できます。ECP5 で既に言及したオプションがある点とは異なります。

21.4 各種ユーティリティツール

21.4.1 デプロイメントツール

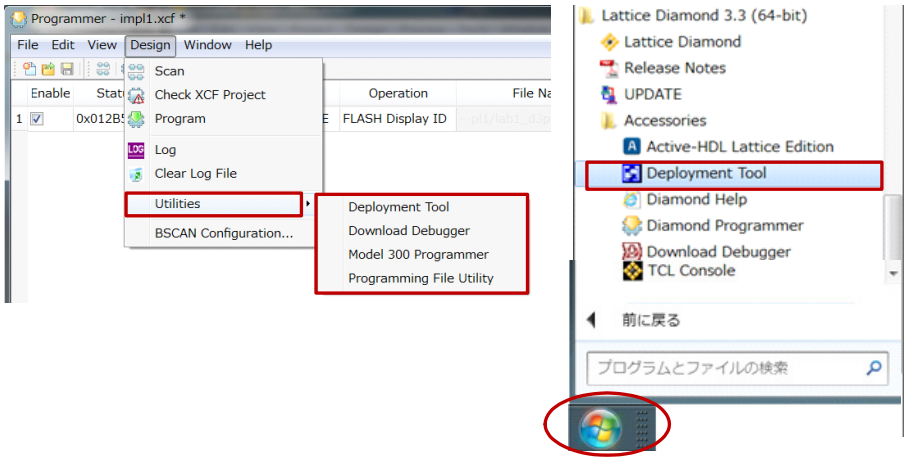
書き込みファイル（.jed、.bit）を組み込みコントローラが扱うファイルフォーマットに変換する際や、デュアルブート機能対応 PROM ファイルを生成する場合などのファイル変換ユーティリティがデプロイメント（Deployment）ツールです。

21.4.1.1 デプロイメントツールの起動

デプロイメントツールの起動は、デタッチしたプログラマーから [Design]-->[Utilities]-->[Deployment Tool] と選択するか（図 21-28、左）、Windows の Start メニューから Lattice Diamond 3.x を選択し、その下 ”Accessories” を展開すると表示される [Deployment Tool] を選択します（図 21-6、右）。

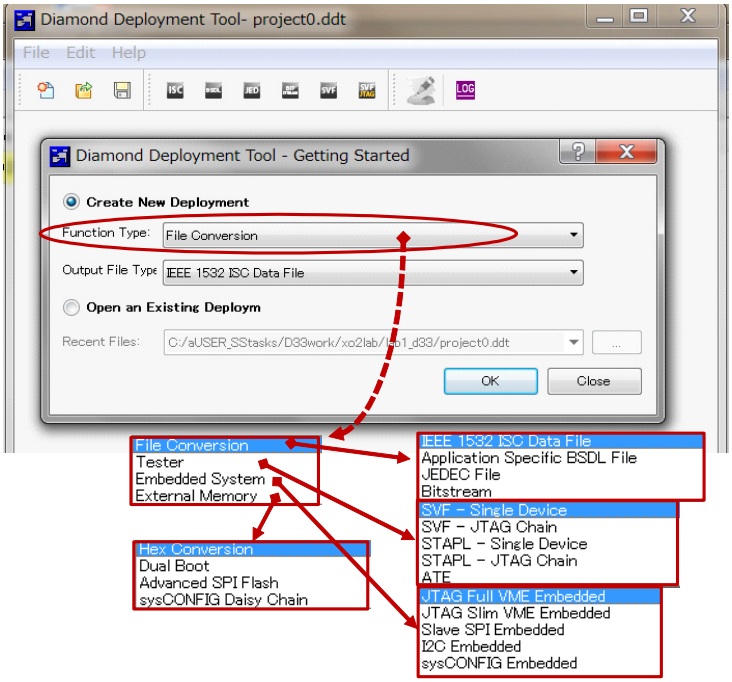
図 21-28. デプロイメントツールの起動方法

プログラマーのメニューから　　Windows Start メニューから



起動後の初期画面は図 21-29 のようになります。[Function Type] として選択できる四つのタイプと、それぞれの生成ファイル形式を合わせて示しています。

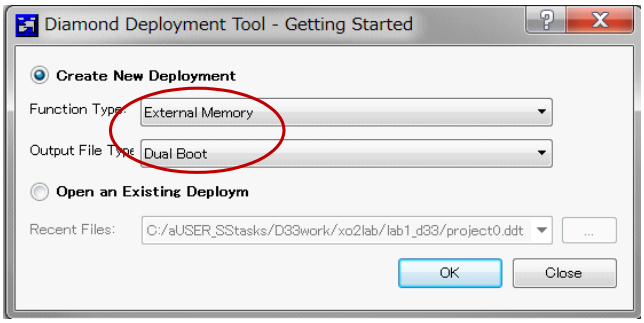
図 21-29. デプロイメントツール起動後とファンクションタイプ



21.4.1.2 デュアルブート用 PROM ファイル生成

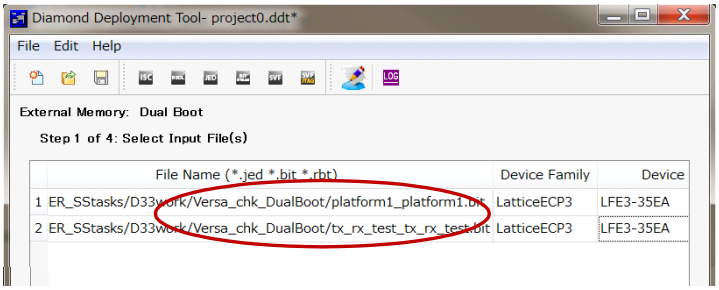
LatticeECP シリーズをターゲットとする、デュアルブート用の二本のビットストリーム（ゴールデンとプライマリ）は生成済みのものとします。統合されるファイルは Hex フォーマットで、通常拡張子は mcs です（後述）。

図 21-30. デュアルブート用 PROM ファイル生成



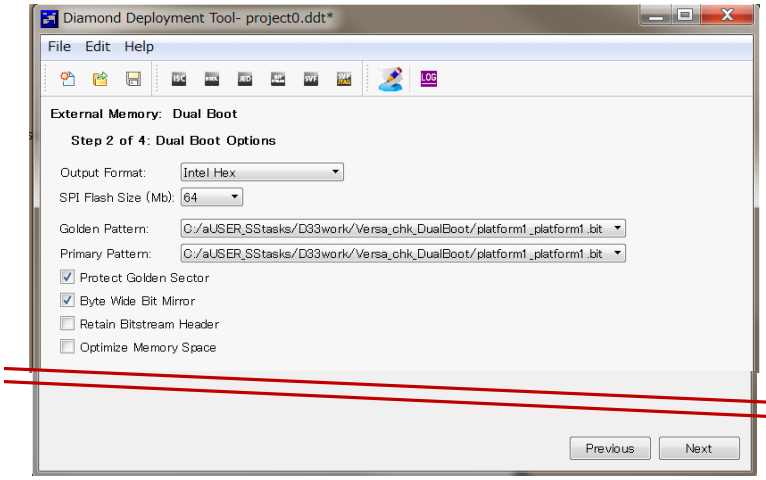
まず最初のウィンドウで、[Function Type] として ”External Memory” を、[Output File Type] として ”Dual Boot” を選択します（図 21-30）。

図 21-31. ビットストリーム・ファイル 2 本の指定例



OK をクリックすると、使用するファイル 2 つの指定ウィンドウが表れますので、意図するファイルをブラウズして入力します。上がゴールデン、下がプライマリです。Next ボタンをクリックして進みます。

図 21-32. 生成ファイルのオプション指定例

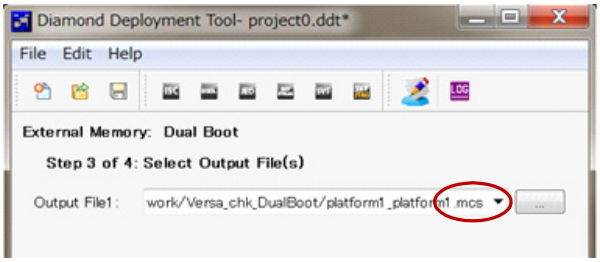


次のウィンドウは生成するファイルのオプションです。[Output Format] は Intel Hex、Motorola Hex、Extended Tektronix Hex から選択します。通常は Intel Hex です（拡張子 mcs）。フラッシュメモリのサイズを選択し、ゴールデンとプライマリが正しいことを確認します。その下に 4 つのオプションがあります。

- Protect Golden Sector　　ゴールデンに保護を掛けます（万が一のため、オンにしておきます）
- Byte Wide Bit Mirror　　ビット反転。Hex ファイル出力では基本的に反転ですのでチェックします
- Retain Bitstream Header　ヘッダ情報を保持する場合にチェックします。通常は残します（オフ）
- Optimize Memory Space　SPI フラッシュメモリのサイズに余裕がない場合です。オンにしなければいけない状況は推奨しません

通常は図 21-32 の例のようにゴールデンの保護とビット反転（ミラーリング）をオンにします。Next ボタンを押して先に行きます。生成ファイルとフォルダ位置の確認・指定ウィンドウが表示されますので（図 21-33）、問題なければ右下の Next ボタンをクリックします（図 21-33 では割愛）。

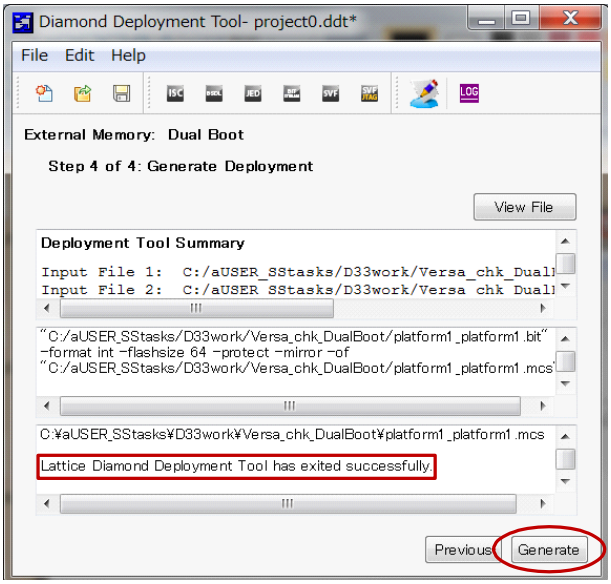
図 21-33. ファイル名とフォルダ指定



因みに、コンフィグレーション用ビットストリームのファイル内のバイトデータ表記は、バイナリファイルの場合が LSB から MSB の順で書かれ、Hex ファイルの場合は MSB から LSB の順で書かれた形式でなければなりません。例えば、ビットストリームのプリアンブル（先頭を示すキーワード）は元々バイナリファイルでの 16 進表記 0xBDB3 ですが、Hex ファイルでの表記は 0xBDCD となります（0xBD の反転が 0xBD、0x03 は 0xCD）。

次にファイル生成します。右下の Generate ボタンをクリックするか、上部のアイコン をクリックします。ウィンドウ下部のコンソールにログ表示されますので、”... successfully” となれば正常終了です。右上の ”View File” ボタンをクリックすると、テキストビューアーが立ち上がり、生成したファイルが閲覧できます。

図 21-34. ファイル生成とチェック

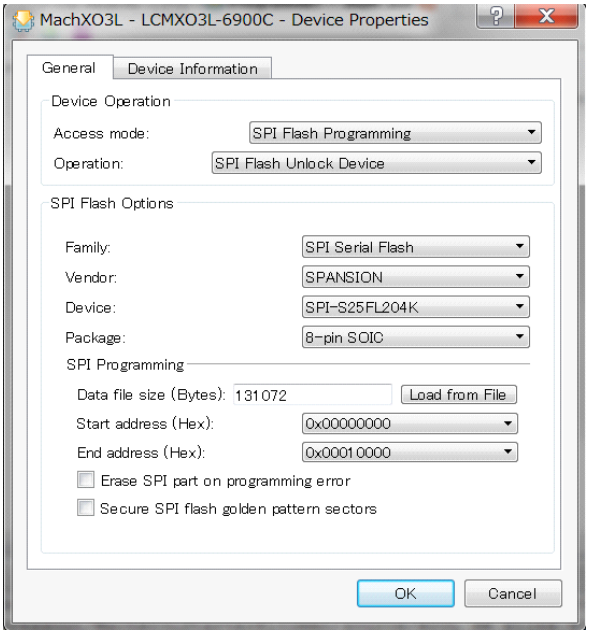


また アイコンをクリックするなどしてログファイルをチェック（保存）することを推奨します。内容としてジャンプコマンド、プライマリとゴールデン両ビットストリームの開始・終了アドレス（セクター）情報が書き出されています。特に開始セクター（アドレス）は一方のビットストリームのみを更新して書き換える場合に必要となります。

デバイスファミリ毎に両ビットストリームとジャンプコマンドのセクタは決められています。デプロイメントツールは自動的に対処してくれますが、ユーザが何らかの別の手段で生成する場合は注意が必要です。詳細はテクニカルノート TN1216 Table.5 をご参照ください。

終了するには、ウィンドウ右上隅の X をクリックしますが、プロジェクトファイル ddt として保存するかどうかの確認を促されます。保存しておけば、後で繰り返し作業する場合などに有用です。

図 21-35. SPI フラッシュメモリの保護解除（MachXO3L）



ここで、MachXO3L で保護をかけたプライマリ・イメージを更新して書き換える際には、始めに保護の解除が必要です。[Operation] で [SPI Flash Unlock Device] を選択します。最下段の ”Secure SPI flash golden pattern sectors” はチェックしないようにします（MachXO2 は未サポートです）。

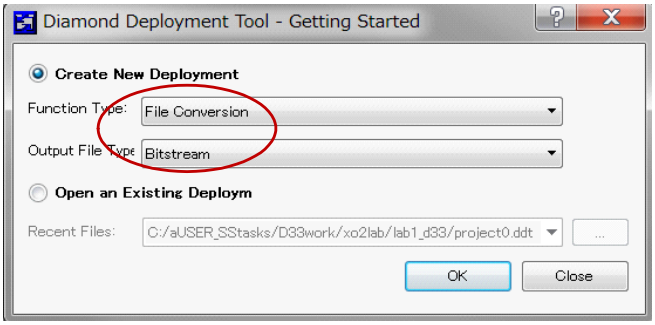
21.4.1.3 書き込みファイルの暗号化

書き込みファイル（ビットストリーム）の暗号化を実行するには通常の Diamond パッケージに加えて暗号化機能用のパッケージ（Encryption Package）が必要です。ユーザがラティスのウェブサイトからまずリクエストに対する許可の申請をします。その後ダウンロード先 URL を含むメールが送付されてくるので、ダウンロード、インストールを済ませます。こうした手順は米国の法律に基づいた対処であり、やむを得ません。日本国内のユーザであれば、問題なく許可されるはずです。

既に言及したとおり、暗号化パッケージをインストールしていないと。本節で示すようなスクリーンダンプ例が得られませんのでご留意ください。

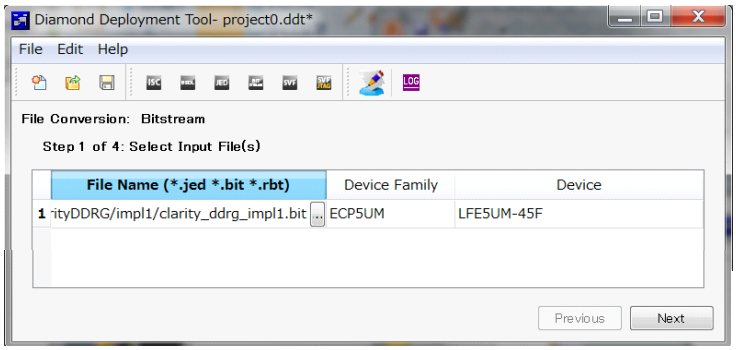
作業手順は次の通りです。まずデプロイメントツールを起動し、[Function Type] は ”File Conversion” に、[Output File Type] は ”Bitstream” を選択します（図 21-36）。既に作成済みデプロイメントツール・プロジェクトがあれば、下部の『Open an Existing Deploym』をチェックして ddt ファイルをブラウズすることで指定します。OK ボタンをクリックします。

図 21-36. 暗号化作業の開始



次のステップは暗号化するファイルの指定です（図 21-37）。ブラウズして Next ボタンで進みます。

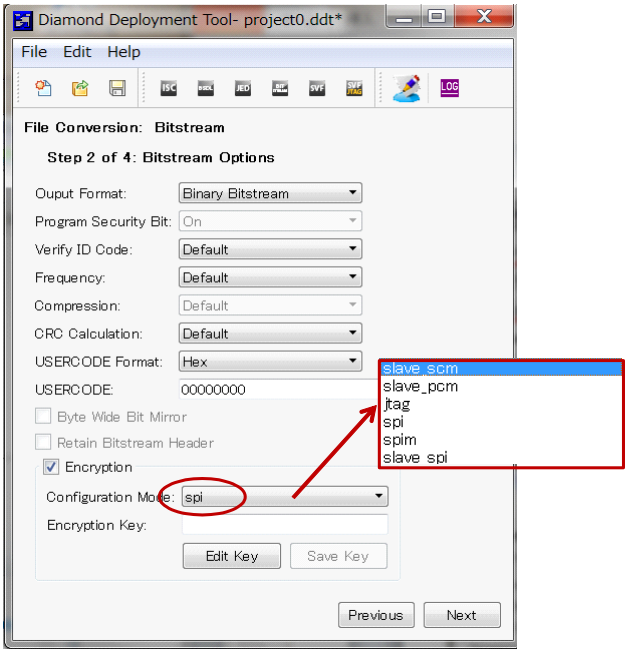
図 21-37. 暗号化対象ファイルの指定



次はオプションの詳細設定です（図 21-38）。通常は特に変更しません。USERCODE を編集することも可能です。

最下部の『Encryption』ボタンをチェックして有効にし、[Configuration Mode] の選択をハードウェアや Diamond のオプション指定と合致させます。非暗号化ビットストリームはこのコンフィグレーション・モードには非依存ですが、暗号化ビットストリームでは合致することが要件です。プログラマーを用いて JTAG からコンフィグレーションする場合は ”jtag” です。

図 21-38. 暗号化詳細設定



次に暗号化キーを入力します。『Edit Key』ボタンをクリックして表示される図 21-39 のウィンドウで指定します。入力後 ”OK” をクリックして抜けると、図 21-38 で『Save Key』ボタンがアクティブになりますので、（必要に応じて）保存します。

図 21-39. 暗号化キーの編集・入力

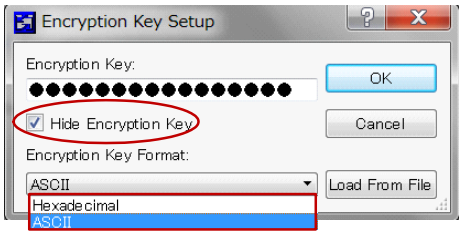
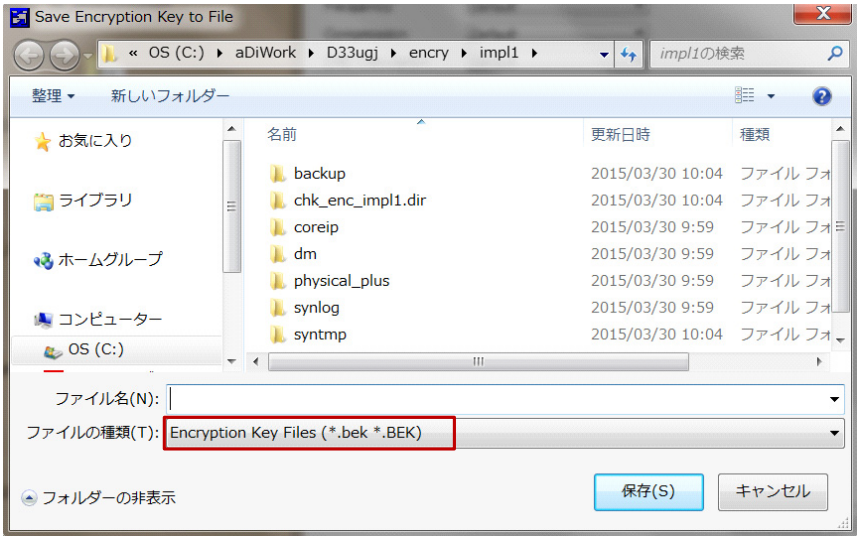
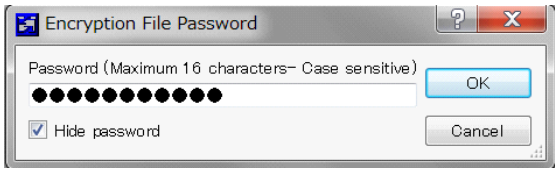


図 21-40. 暗号化キーの保存



ファイル名を入力して『保存』をクリックすると、パスワードの入力を求められます。知る必要のない人に見られないようにするための措置です。問題なければ ”Save key successfully” というメッセージの小ウィンドウがポップアップします。

図 21-41. 暗号化キーファイルのパスワード



次にキーが入力された状態で図 21-38 に戻りますので、『Next』で次ステップに進み、出力ファイル名の指定・確認を行います（図 21-42）。元のビットストリーム・ファイルと異なる名称にする場合にファイル名とフォルダを明記します。『Next』で次に進みます。

図 21-42. 出力ファイル名の指定・確認

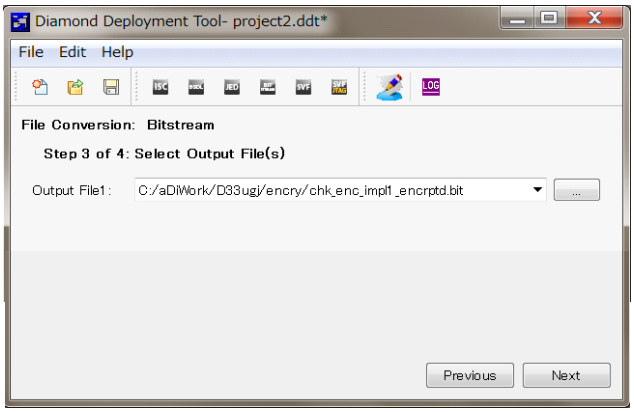
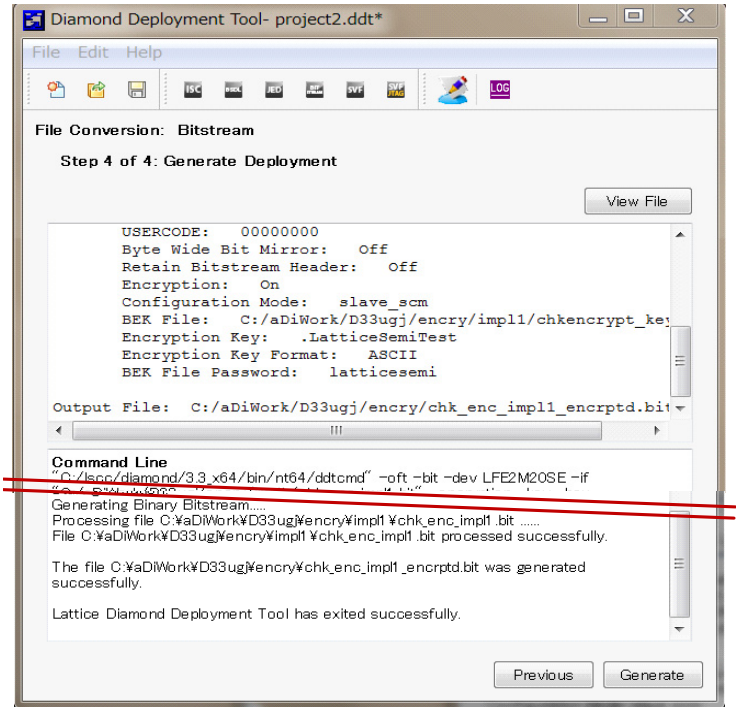


図 21-43. 暗号化ビットストリームの生成



最後が暗号化ビットストリームの生成です（図 21-43）。右下の『Generate』ボタンか上部の アイコンをクリックします。下段の窓枠で図のように "... excited successfully" と表示されれば正常終了です。

アイコン をクリックしてログの確認や、『View File』をクリックしてテキストエディタで生成物の内容をチェックできます。

最後にデプロイメント・プロジェクト ddt を保存後、ウィンドウ外枠の右上すみの X 印をクリックして終了します。

21.4.2 フィーチャ行エディタ（MachXO2/XO3L）

ハードマクロ EFB がサポートする I2C や SPI インターフェイスを介して、外部コントローラから最初に（デバイスがブランク時や消去状態で）プログラミングする際には、フィーチャ行（Feature Row）は通常最低一度はアクセス（ライト）することになります。しかし、フィーチャ行を変更するのは開発時、及び量産（出荷）時のみとすることを強く推奨しています。こ れは、特にフィーチャ行は使用可能にするコンフィグレーショ ン・ポートの制御を行うためです。ノイズが多いなど望ましくない特定の環境下では、意図せぬアクセスのために誤った値を書き換えてしまい、従ってアクティブなコンフィグレーション・ポートが使用できなくなり、結果的にその後のアップデートを行えなくなるケースが生起する可能性が否定できません。

何らかのやむを得ない事由（含デバッグ等）でフィーチャ行に書かれているビットの設定値を確認したり、編集することが避けられない場合はフィーチャ行エディタが活用できます。起動するには、デタッチしたプログラマーで **Design-->Utilities-->Programming File Utility** を順に選択してユーティリティツールを立ち下げ（図 21-44）、その後 **Tools --> Feature Row Editor** を指定して行います。また Programming File Utility については、図 21-28 に示すと同様の手順で Windows のスタートメニューから立ち上げることもできます。

図 21-44. フィーチャ行エディタの起動

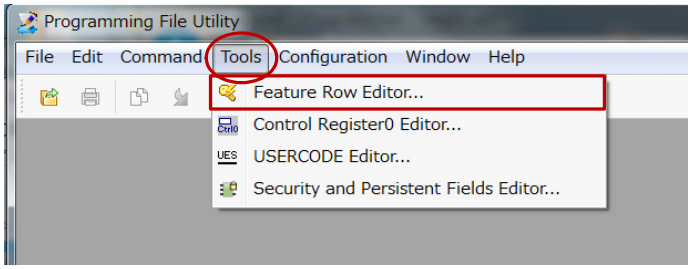
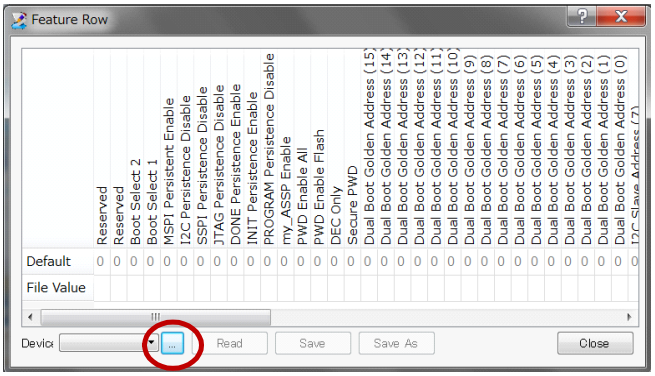


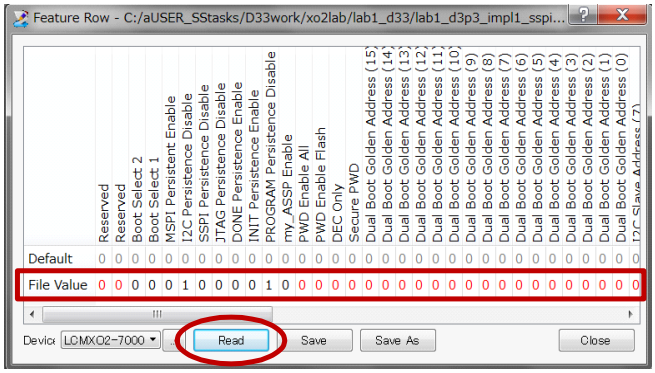
図 21-45 左のような起動直後、赤丸で示すブラウズボタンで jed ファイルを指定し、デバイスタイプ等を読み取ってアクセスの準備をします（MachXO2 の jed ファイルを指定した例）。次に "Read" ボタンをクリックしてファイルからフィーチャ行情報を抽出して、同右のように表示します。

図 21-45. 起動後の表示例

起動直後



JEDEC ファイルを読み込んだ後



黒色で表記されているビットのいづれかをクリックするたびに、当該ビット表示がトグルします。赤色表記のビットは編集できません。編集後、”Save” や ”Save As” で Jedec ファイルとして（別名で）保存します。その後この新たに保存したファイルでプログラムします。

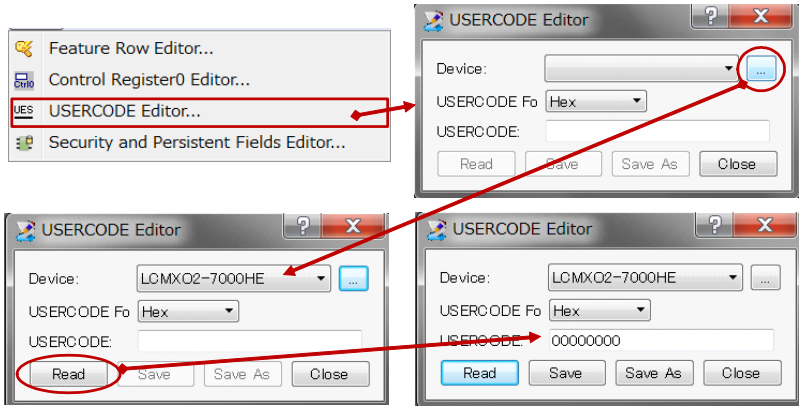
なお、フィーチャ行エディタでは 21.3.1 項で言及した TraceID および Custom ID Code の編集もサポートします。図 21-45 では示していませんが、本ウィンドウの右側にフィールドがあります。

21.4.3 USERCODE エディタ

特にユーザーコードを編集するツールが USERCODE エディタです。起動するには、図 21-44 のウィンドウで [Tools]-->[USERCODE Editor...] と選択します。

操作手順は前項のフィーチャ行エディタと同様です。表示されるウィンドウ（右上）で [Device] 行右端のブラウズボタンで該当ビットストリーム・ファイルを指定します。デバイスタイプが読み込まれ、表示に反映されます（左下）。”Read” ボタンをクリックして値を抽出して表示します（右下）。これで所望の値に編集して保存し直します。デフォルトでは 16 進表記になっています。

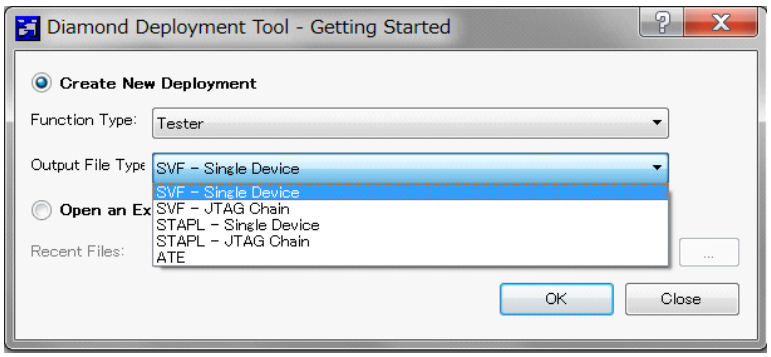
図 21-46. USERCODE エディタの起動から編集まで



21.4.4 ダウンロード・デバッガ

意図せぬ理由で、ISP/JTAG プログラミングが失敗する場合、いったん書き込みファイル（.bit、.jed）を SVF ファイルに変換し、これを用いたデバッグを実行することで原因を究明できる場合があります。ダウンロード・デバッガ（Download Debugger）は、ダウンロードケーブルで接続された実デバイスを動作させながら、SVF ファイルをデバッグするためのツールです。組み込みコントローラ用 VME ファイルのデバッグでも使用できますが、本節では SVF ファイルを例にとり記述します。

図 21-47. デプロイメントツールのファンクションタイプ選択

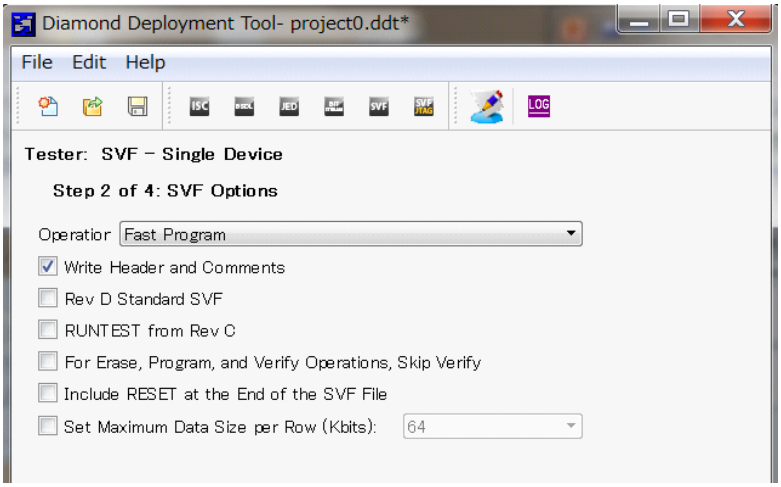


SVF ファイルが未生成の場合、まずデプロイメントツールでフォーマット変換します。起動方法は 21.4.1 節をご参照ください。ファンクションタイプは ”Tester”、出力ファイルタイプは SVF を選択します。JTAG

チェイン内にターゲットの単一デバイスのみか、ターゲットを含む複数デバイスかによりオプション指定等が異なります。

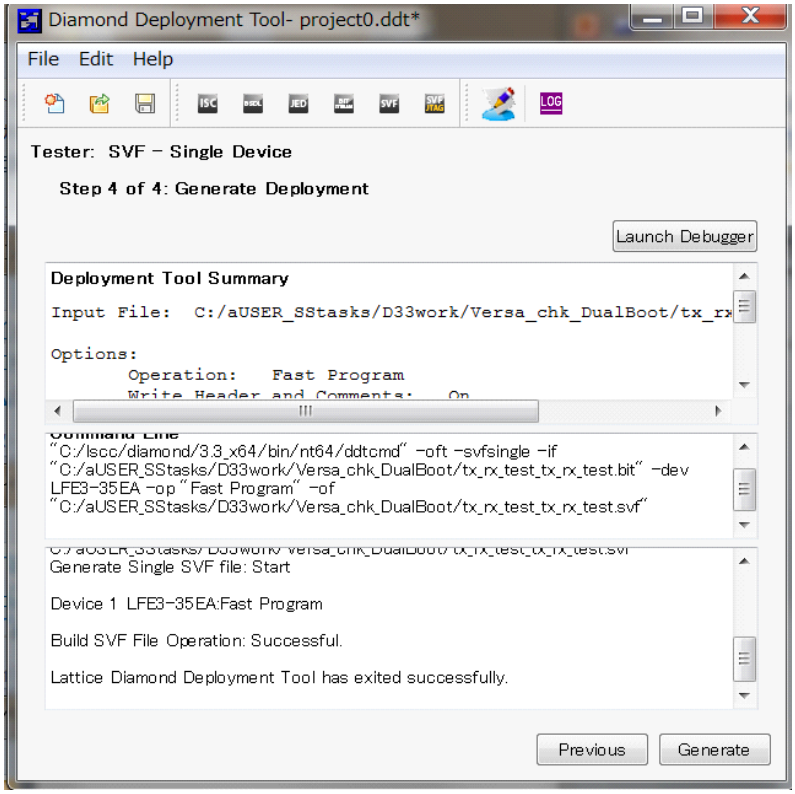
ステップ 1 でソースファイルを指定後、ステップ 2 で図 21-48 のようにオペレーション（Operator）と各オプションを指定します。選択できるオペレーションはターゲットデバイスに依存します。

図 21-48. SVF 変換の各種オプション指定



その後ステップ 3 で出力ファイル名とフォルダを指定し、ステップ 4 で変換を実行（Generate）します（図 21-49）。『... successfully』と表示されれば終了です。作業を繰り返す際に備えてデプロイメント・プロジェクト（.ddt）として保存しておきます。次回からはファイルのオープンで一連の作業が迅速に行えます。

図 21-49. SVF ファイルの生成（変換）



以上で準備が整いましたのでダウンロード・デバッガを起動します。図 21-49 中の右上にある『Launch Debugger』をクリックすることで連続して作業ができます。この場合初期画面は図 21-51 のようになり、生成した SVF をロードした状態で立ち上がります。

或いは一度デプロイメント・ツールを終了し、図 21-50 のように、プログラマーから起動することも可能です。この場合、何もファイルをロードしていないブランク状態でデバッガが立ち上がりますので、File --> Open と辿りデバッグのターゲットとする SVF ファイル（別作業の場合は STP、VME ファイルなどが選択可）を指定します。これにより、図 21-51 のような初期状態になります。

図 21-50. プログラマーからダウンロード・デバッガの起動

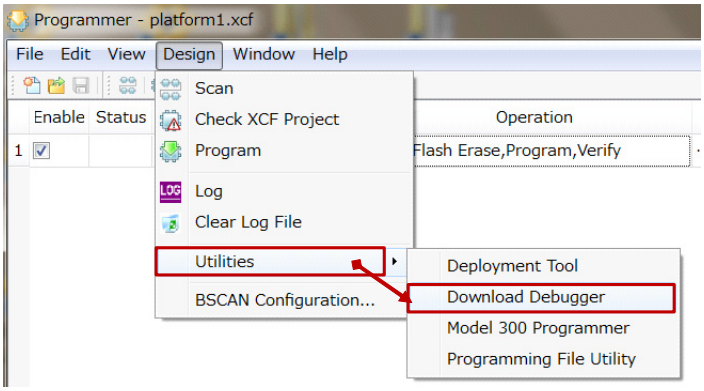
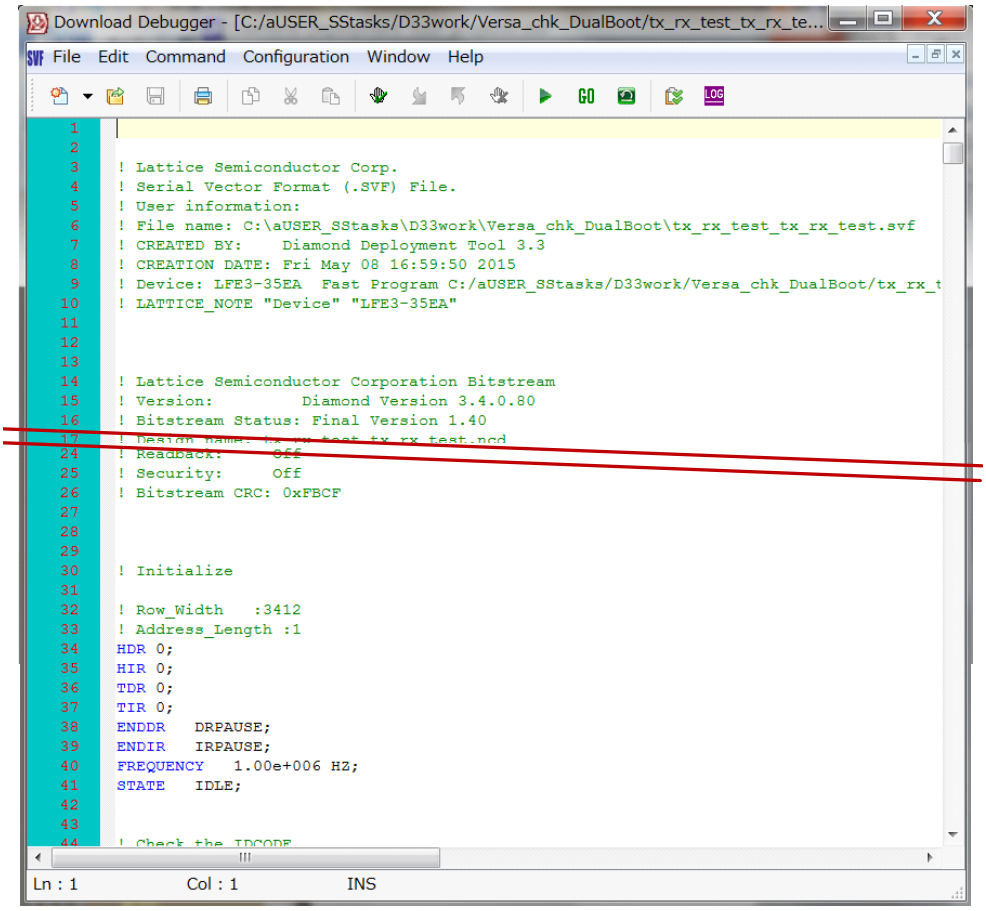


図 21-51. SVF ファイル読み込み後のデバッガ初期画面

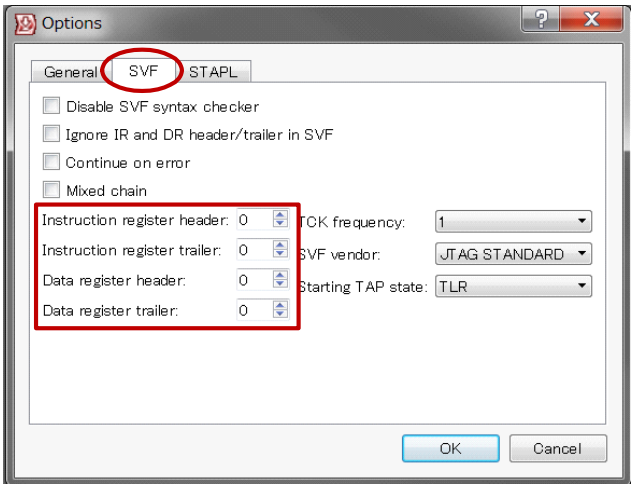


デバッガの操作は、組み込みソフトウェアをデバッグする際のプリミティブな手法と類似しています。上部にあるアイコン列を活用してステップ実行、ブレークポイントの設定・解除、ブレークポイントまでの実行、などを行います。アイコンとオペレーションは次の通りです。

アイコン	オペレーション	アイコン	オペレーション
	Toggle Breakpoint		Next Breakpoint
	Previous Breakpoint		Clear All Breakpoint
	Step	GO	GO
	Reset		

なお、チェイン内に複数のデバイスがある場合、オプション設定が必要です。 アイコンをクリックするか、ダウンロード・デバッガのメニューで Debugger -->Options... と辿り、SVF タブを設定することでオプション設定画面を表示させます。

図 21-52. SVF オプション設定



JTAG チェイン内にあるターゲット・デバイスをデバッグする場合、前（header：ヘッダ）と後ろ（trailer：トレイラ）にあるデバイスの命令レジスタ（Instruction Register）長 [単位：bit] を指定します（図 21-52、赤枠)。ない場合は 0 です。図 21-53 にヘッダ長・トレイラ長の設定例を示します。

図 21-53. JTAG チェインの構成とレジスタ長の例

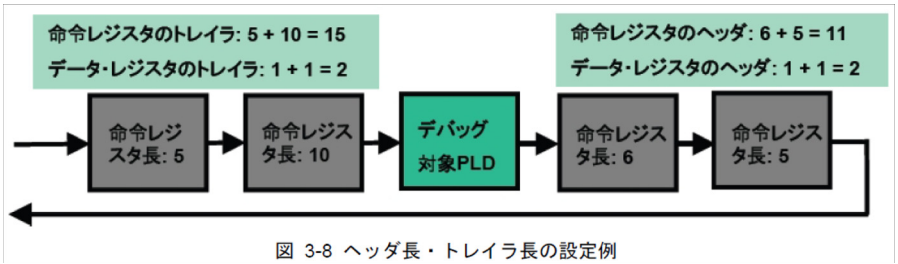


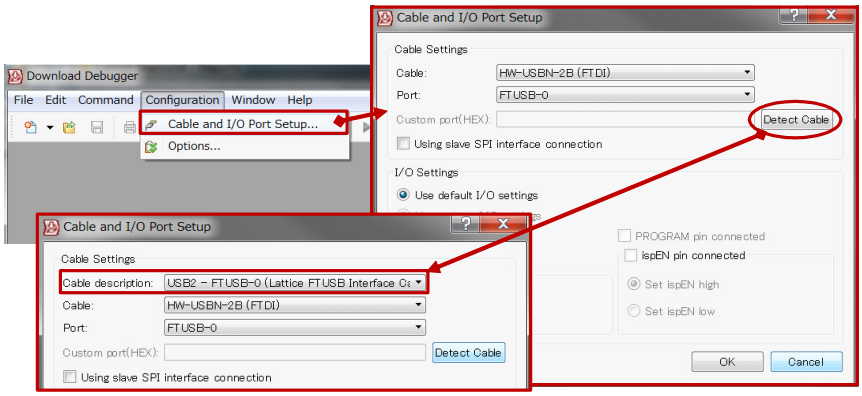
図 3-8 ヘッダ長・トレイラ長の設定例

21.5 その他の機能

21.5.1 ダウンロードケーブルと I/O ポートの設定

デバイススキャンが成功しない場合、ダウンロードケーブルの設定が正しくない可能性があります。念のため Configuration --> Cable and I/O Port Setup... と選択後、『Cable Detect』ボタンをクリックして自動検出して、意図する設定済みの状態と相違ないことを確認します。

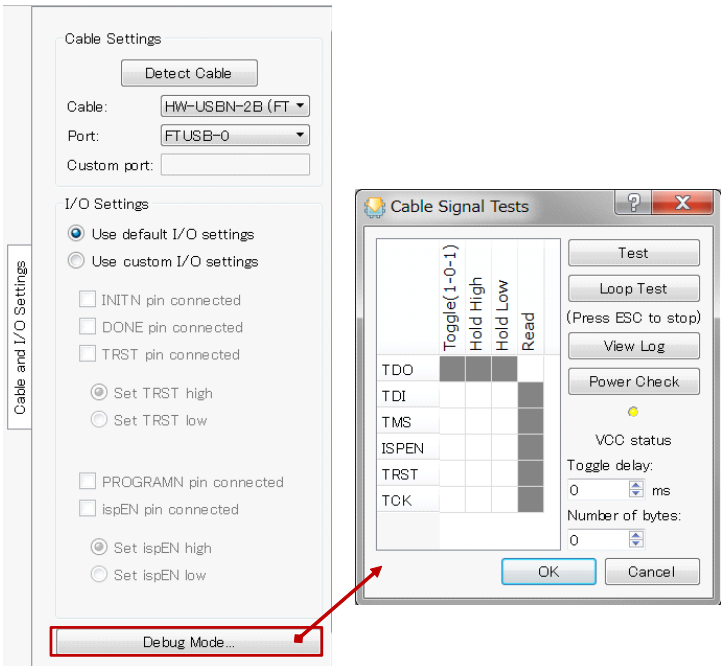
図 21-54. ケーブル設定の再確認



21.5.2 JTAG 信号線接続の確認

ダウンロードケーブルの設定が正しくてもデバイススキャンが失敗する場合、フライワイヤ形式のケーブルや PCB 配線が各信号線の期待する配線と合致していないことが考えられます。この場合 GUI を用いて接続テストをすることができます。プログラマーのウィンドウ右側に図 21-55 の左部のようなセクションがあります。『Debug Mode ...』ボタンをクリックすると右部のようなウィンドウが表示されますので、『Test』ボタンなどで接続状態が確認できます。アクションと対象信号は左側の表内の白色セルをクリックして指定します。

図 21-55. 信号接続テスト GUI



例えば TDO の Read セルをクリックするとチェックマークが表示されますので、その後『Test』をクリックします。リード結果がログファイルに書き出されますので、実際の論理レベルと合致するかが確認できます。

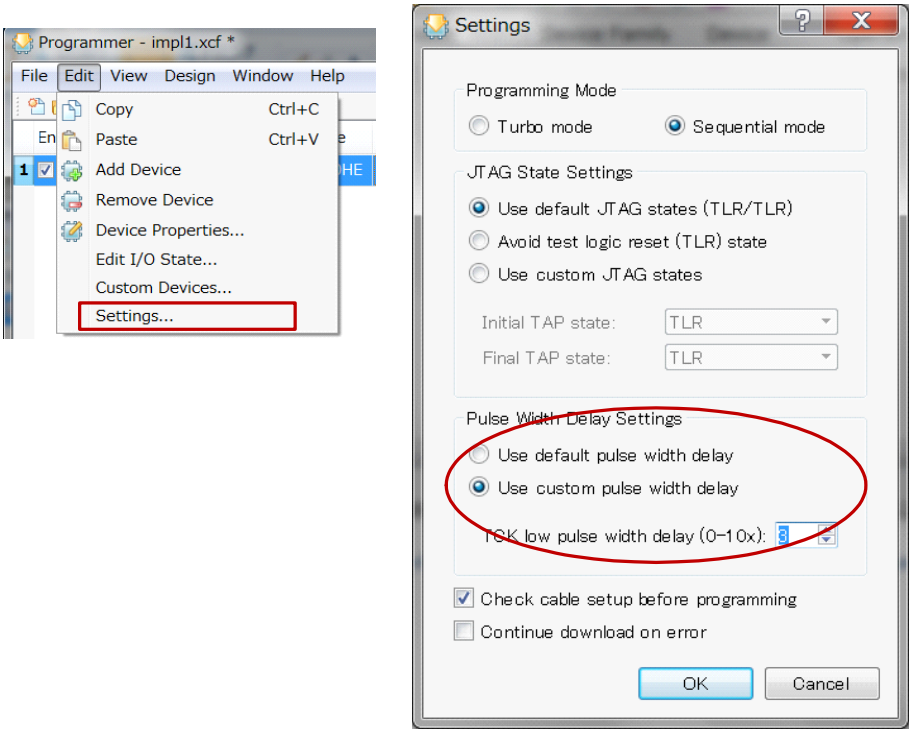
TDO 以外は全て出力信号ですので、トグル（Toggle）・High 固定（Hold High）・Low 固定（Hold Low）のいずれかアクションを意図する信号に対するセルをチェックします。『Test』ボタンをクリックすると、指定したアクション（レベル）に出力がドライブされますので、接続の正しさを確認できます。

21.5.3 JTAG クロック周波数の変更

レガシーデバイスとか、何らかの事由で JTAG からのオペレーションが失敗する際に、クロック TCK の周波数を遅くすることで問題が解消する場合があります。本項はその方法を記述します。

プログラマーをデタッチした後、図 21-56 のように Edit --> Settings... と辿ると右のような画面が表示されます。『Pulse Width Delay Settings』部の［Use custom pulse width delay］ボタンをイネーブルし、[TCK low pulse width delay (0-10x):] 行の値を変更します。デフォルトは 1 で、大きくするほど等価的に周波数が遅くなります。

図 21-56. TCK 周波数の変更



目安の周波数は以下の通りです。

HW-USBN-2A	Freq. = 6/ (1+ TCK Delay) [MHz]	‘0’--> 6MHz, ‘1’ --> 3 MHz, etc.
HW-USBN-2B	Freq. = 4/ (1+ TCK Delay) [MHz]	‘0’--> 4MHz, ‘1’ --> 2MHz, etc.

21.6 （参考）プログラミング時間

参考として各ファミリのオペレーション所用時間を示します。コンフィグレーション SRAM への書き込み時間はファミリに拘わらず数秒程度で終了します。MachXO シリーズの内蔵不揮発メモリへのオペレーション時間を以下に転記します。詳細はそれぞれのテクニカルノートをご参照ください。

図 21-57. MachXO3L のプログラミング時間（TN1294、Table 98）

NVCM/Flash Performance

Table 98. NVCM/Flash Performance in MachXO3L/LF Device[1]

		MachXO3L/LF -640	MachXO3L/LF -1300	MachXO3L/LF -1300 256 Ball Package	MachXO3L/LF -2100	MachXO3L/LF -2100 324 Ball Package	MachXO3L/LF -4300	MachXO3L/LF -4300 400 Ball Package	MachXO3L/LF -6900
CFG Erase (tEraseCFG)	Min.	800	800	1100	1100	1800	1800	2800	2800
	Max.	1400	1400	1900	1900	3100	3100	4800	4800
CFG Program (tProgramCFG)	All	500	500	740	740	1400	1400	2200	2200
	1 page	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
NVCM1/UFM Erase (tEraseNVCM1/UFM)	Min.	400	400	500	500	600	600	900	900
	Max.	700	700	900	900	1000	1000	1600	1600
NVCM1/UFM Program (tProgramNVCM1/UFM)	All	110	110	140	140	180	180	480	480
	1 page	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2

1. All times are averages, in (ms). SRAM erase times are < 0.1 ms.

注；”CFG” はコンフィグレーション用不揮発メモリを指す

図 21-58. MachXO2 のプログラミング時間（TN1264、Table 97）

Flash Memory Erase and Program Performance

Table 97. Flash Memory (UFM/Configuration) Performance in MachXO2 Devices[1]

		MachXO2 -256	MachXO2 -640	MachXO2 -640U	MachXO2 -1200	MachXO2 -1200U	MachXO2 -2000	MachXO2 -2000U	MachXO2 -4000	MachXO2 -7000
CFG Erase (tEraseCFG)	Min.	400	600	800	800	1100	1100	1800	1800	2800
	Max.	700	1100	1400	1400	1900	1900	3100	3100	4800
CFG Program (tProgramCFG)	All	130	270	500	500	740	740	1400	1400	2200
	1 page	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2
UFM Erase (tEraseUFM)	Min.	—	300	400	400	500	500	600	600	900
	Max.	—	600	700	700	900	900	1000	1000	1600
UFM Program (tProgramUFM)	All	—	40	110	110	140	140	180	180	480
	1 page	—	0.2	0.2	0.2	0.2	0.2	0.2	0.2	0.2

1. All times are averages, in (ms). SRAM erase times are < 0.1ms.

なお、通常は上に示す様な数値に近くなるはずですが、使用する PC によっては、USB ドライバとクロック生成の関連で上に示す値より大幅に大きくなるケースも報告されていますので、ご了解ください。

外付け SPI フラッシュメモリへのプログラミング時間は種々のパラメータが関与するので一義的に示すことはできませんが、上記値より極端に大きくなることはないようです。

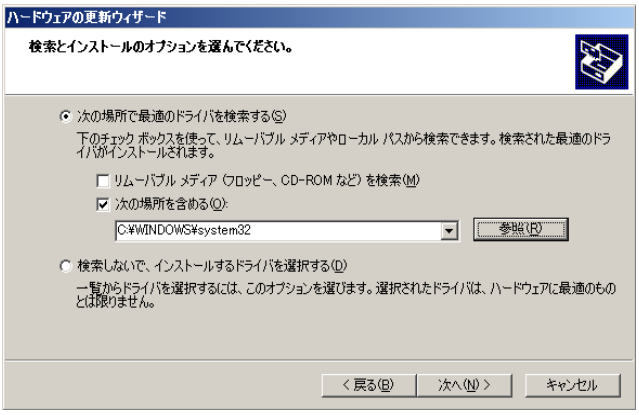
21.7 USB ドライバのインストール

USB ドライバのインストールが事前に完了していない場合、ターゲットボードを初めて USB ポートに接続した場合に、インストールを促す Windows のメッセージが表示されます。

図 21-59. USB ドライバのインストール要求初期画面

『一覧または指定の場所からインストールする』を選択し次に進みます。(Windows は Administrator 権限でログインしている必要があります)。

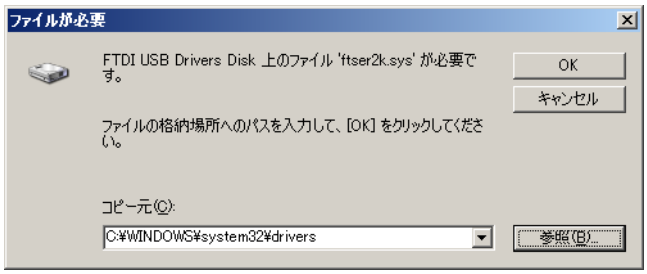
図 21-60. USB ドライバのフォルダ指定



『次の場所を含める』を選択して、”参照”ボタンで”C:¥WINDOWS¥system32”をブラウズして指定します。”次へ”をクリックします。

FTDI USB スレーブデバイスがボード上にある場合、続いて図 21-61 に示すような画面が表示されます。

図 21-61. FTDI USB ドライバのコピー元指定



”参照”ボタンで”C:¥WINDOWS¥system32¥drivers”をブラウズして指定し、”OK”をクリックします。これ以外に ” ファイルが必要 ” と促される場合がありますが、同様に ”C:¥WINDOWS¥system32” か ”C:¥WINDOWS¥system32¥drivers”を指定して手順を完了します。

システム・プロパティ（コントロールパネル）のデバイスマネージャで、ボードを接続した状態で正常に認識されていることを確認します。

図 21-62. デバイスマネージャ

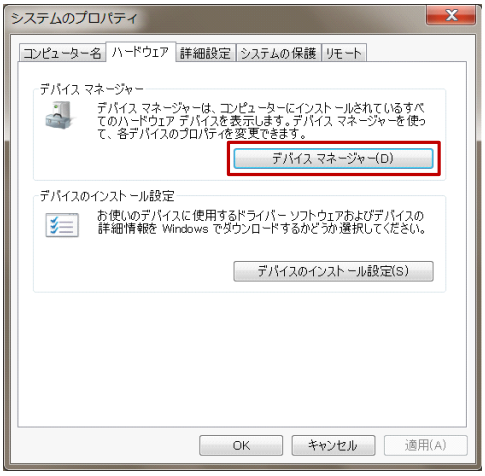
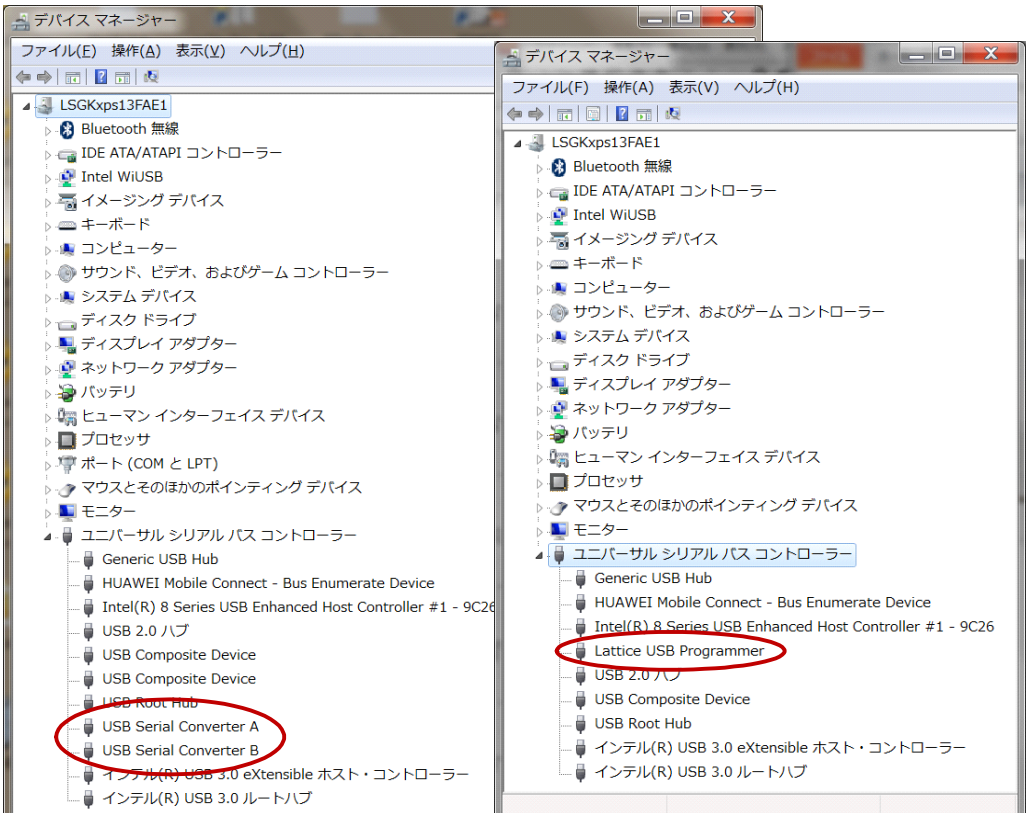


図 21-63. デバイスマネージャの USB ドライバ表示例（左：FTDI あり、右：なし）



21.8 改訂履歴

Ver.	Date	page	内容
3.3 v1.0	June 2015	--	初版（Diamond JUG Ver.3.3.1）

--- *** ---