第 15 章　プログラミング・ユーティリティ

15.1 デプロイメント・ツール（Deployment Tool）

デプロイメント・ツールはファイル変換ユーティリティです。デュアルブート / マルチブート機能対応で外付け SPI フラッシュメモリーに書く mcs ファイル（”PROM ファイル ”）を生成する場合や、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場合、或いは ECP5 ファミリーや Crosslink ファミリーが対応する Dual-/Quad-SPI フラッシュメモリー・インターフェイスを用いたり、またチェインを構成する場合など、種々のケースで用います。

15.1.1 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] と選択します（図 15-1、左）。または Windows の Start メニューから Lattice Diamond 3.12 を展開すると表示される [Deployment Tool] を選択しても起動できます（図 15-1、右）。

図 15-1. デプロイメント・ツールの起動

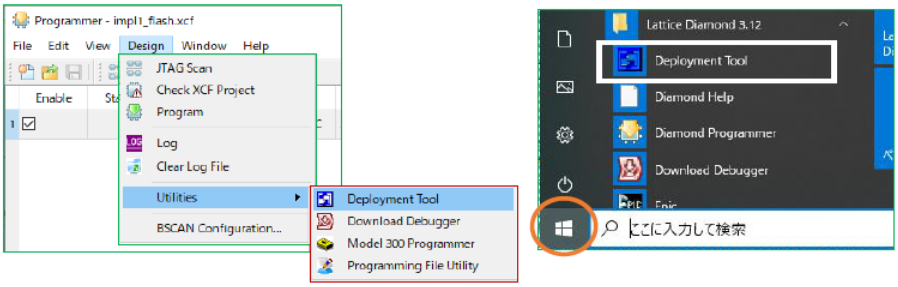
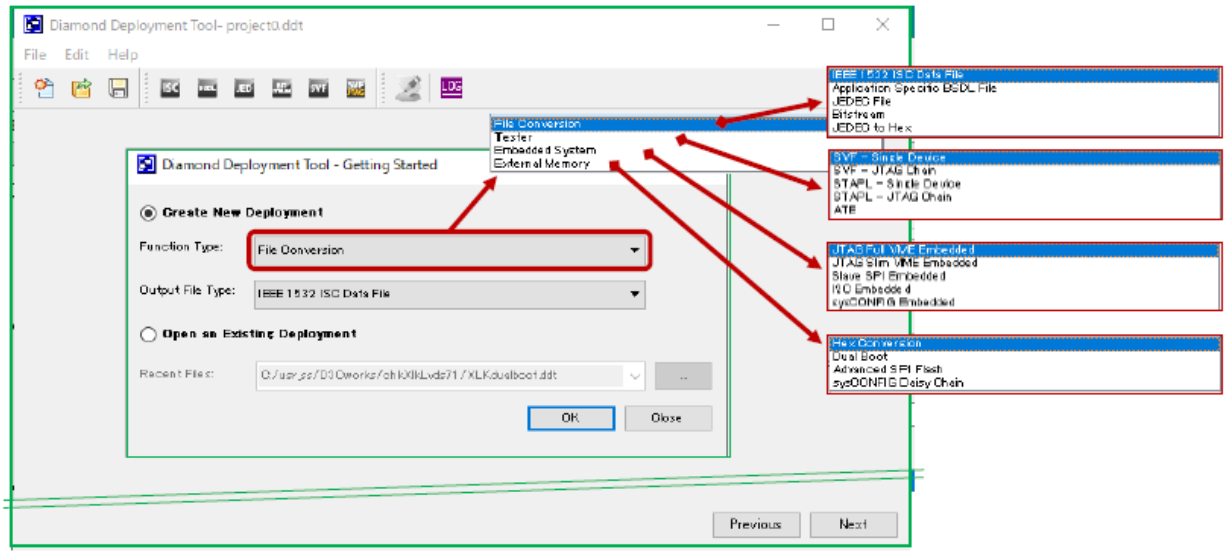


図 15-2. デプロイメントツール起動後とファンクションタイプ

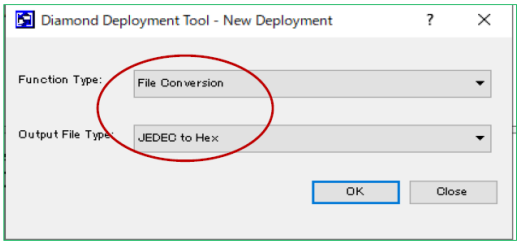


起動後の初期画面は図 15-2 のようになります。「Function Type」として四つが選択でき、それぞれに対する生成ファイル形式があります。殆どの作業でステップ１（対象ファイル入力)、ステップ３（出力ファイル名とフォルダーの指定)、およびステップ４（ファイル生成・実行）は共通の操作になります。ステップの進行はそれぞれウィンドウ下部の『Next』ボタンで行います。戻る場合は『Previous』ボタンをクリックします。開始画面には戻れませんが、ステップ１までは戻ることができます。

15.1.2 jedec から hex への変換

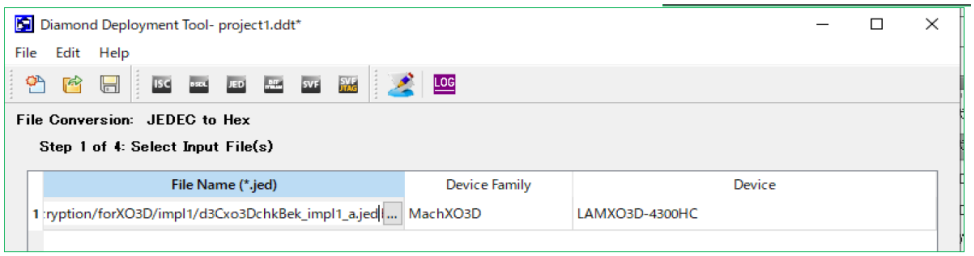
しばしば jedec ファイルを 16 進表記にして確認・作業する必要がある場合があります。デプロイメント・ツールではこの変換が容易にできます。起動後、最初のウィンドウで、「Function Type」として ”File Conversion” を、「Output File Type」として ”JEDEC to Hex” を選択して『OK』をクリックします

図 15-3. jedec から hex 変換の開始



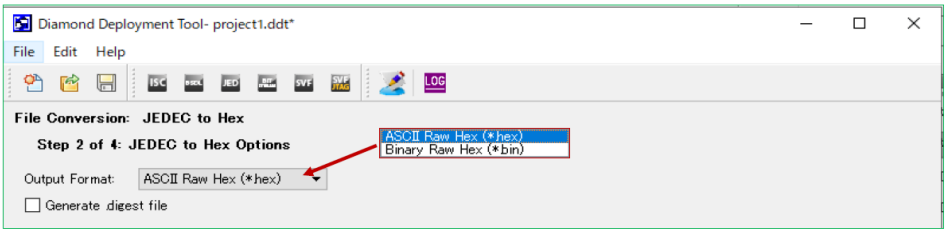
次にステップ１で対象のファイルをブラウズして指定します（図 15-4)。

図 15-4. 対象ファイルの指定



ステップ２は出力フォーマットの指定です（図 15-5)。通常は ”ASCII Row Hex” です。

図 15-5. 出力フォーマットの確認

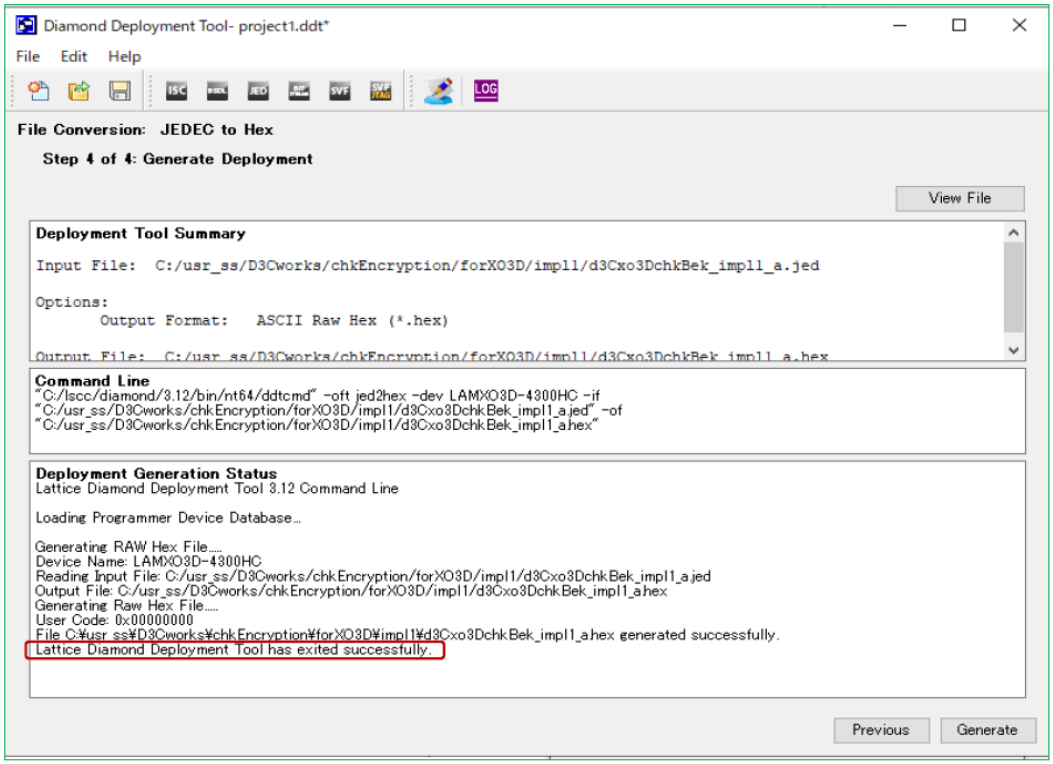


ステップ３は出力ファイル名とフォルダーの指定です（図 15-6)。

図 15-6. 出力ファイル名とフォルダーの指定

ステップ４がファイル生成・実行です（図 15-7）。上部のアイコン をクリックするか、右下の『Generate』ボタンをクリックします。ステータス窓で ”... successfully.” となっていることを確認して終了します。

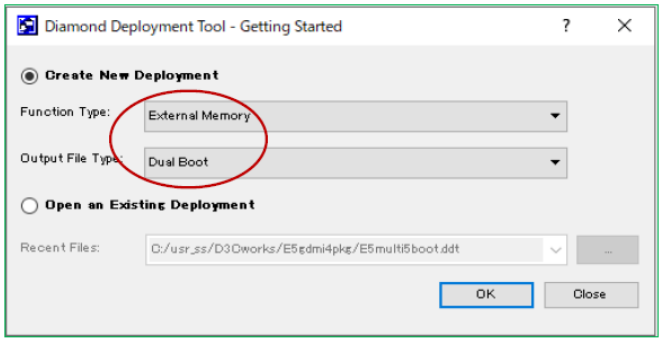
図 15-7. 変換実行と結果確認



15.1.3 デュアルブート用 mcs ファイル生成：ECP5、Crosslink

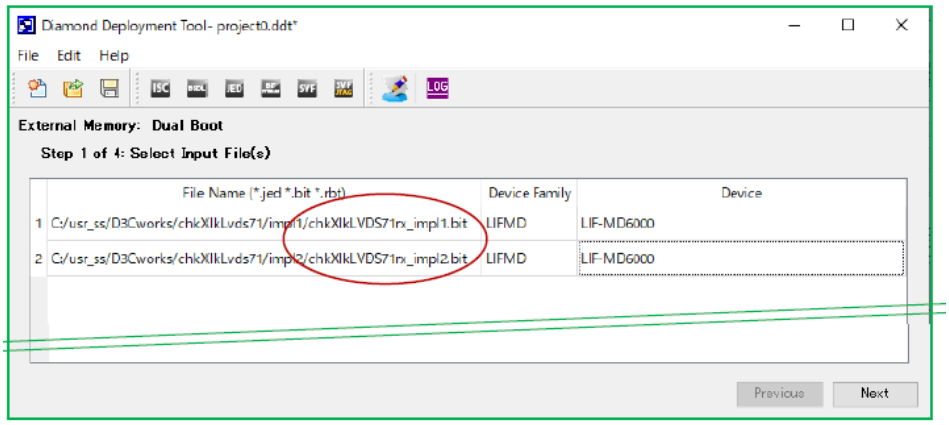
ECP5 ファミリーと Crosslink が対応するデュアルブート機能用に、外付け SPI フラッシュメモリーにストアする二本のビットストリーム・ファイルを結合して *.mcs ファイルを生成します。デュアルブート機能は、CRC エラーなど何らかの理由で ” プライマリ ” ビットストリームでの起動に失敗すると、自律的に ” ゴールデン ” ビットストリームで再起動を試みるものです。

図 15-8. デュアルブート用ファイル生成の開始



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” を選択して『OK』をクリックします（図 15-8）。

図 15-9. ビットストリーム・ファイル 2 本の指定例



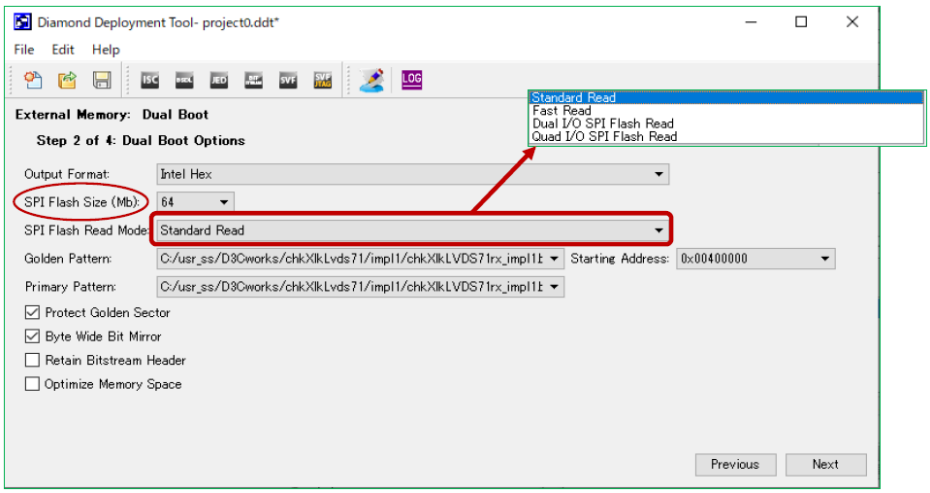
次に使用するファイルの指定ウィンドウが表れます（図 15-9）。生成済みの .bit ファイルをブラウズして入力します。上がゴールデン、下がプライマリです。『Next』ボタンをクリックして進みます。

次のウィンドウは生成するファイルのオプション設定です（図 15-10）。「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）。フラッシュメモリーのサイズを選択し、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 フラッシュメモリー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

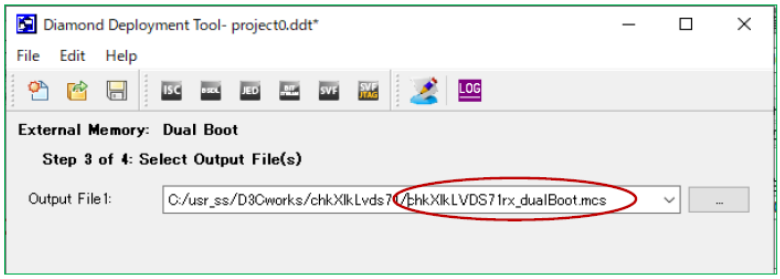
通常は図 15-10 の例のようにゴールデンの保護とビット反転（ミラーリング）をオンにし、『Next』ボタンを押して次に進みます。「SPI Flash Read Mode」はデフォルトが ”Standard Read” です。第 15.1.5 項に記述する、リードデータ線を複数とする ”Dual/Quad SPI リード ” の場合は ”Dual (Quad) I/O SPI Flash Read” を選択します。

図 15-10. 生成ファイルのオプション指定例



生成ファイル（拡張子 .mcs）とフォルダー位置の確認・指定ウィンドウが表示されますので入力し（図 15-11）、右下の『Next』ボタンをクリックします（図 15-11 では割愛）。

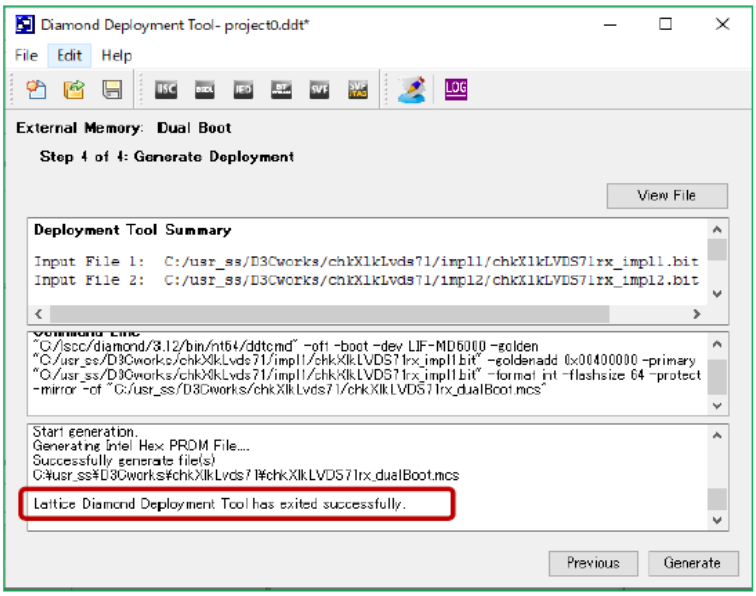
図 15-11. ファイル名とフォルダー指定例



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 "... successfully" とログ表示されれば正常終了です（図 15-12)。

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。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。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・ファミリー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、ユーザーが何らかの別の手段で生成する場合は注意が必要です。

図 15-12. ファイル生成とチェック



デプロイメント・ツールの終了は、ウィンドウ右上隅の "X" 印をクリックしますが、プロジェクトファイル（拡張子 .ddt）を保存するかどうかの確認を促されます。保存しておけば、後で修正などの繰り返し作業する場合などに有用です。

15.1.4 マルチブート用 mcs ファイル生成：ECP5

ECP5 ファミリーは、最大四本までの代替えビットストリーム・ファイルを外付け SPI フラッシュメモリーに格納し、それらを自律的に切り替えるマルチブート機能があります。そのための mcs ファイルを生成します。Dual Boot のように出力ファイルタイプとしては定義されていませんので、図 15-13 のように最初のウィンドウでは「Function Type」を "External Memory"、「Output File Type」を "Advanced SPI Flash" にします。

図 15-13. マルチブート用ファイル作成の開始

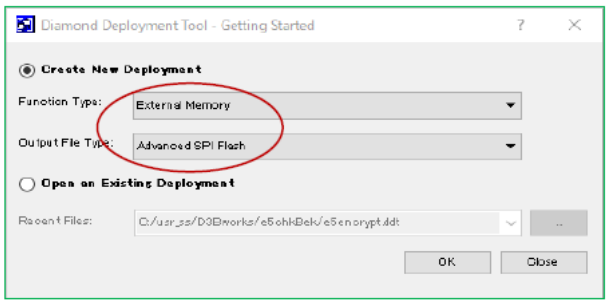
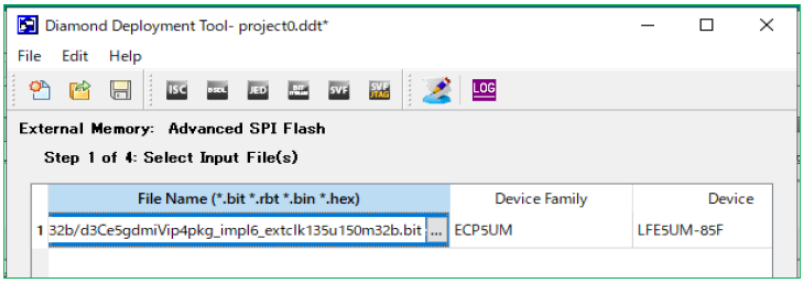
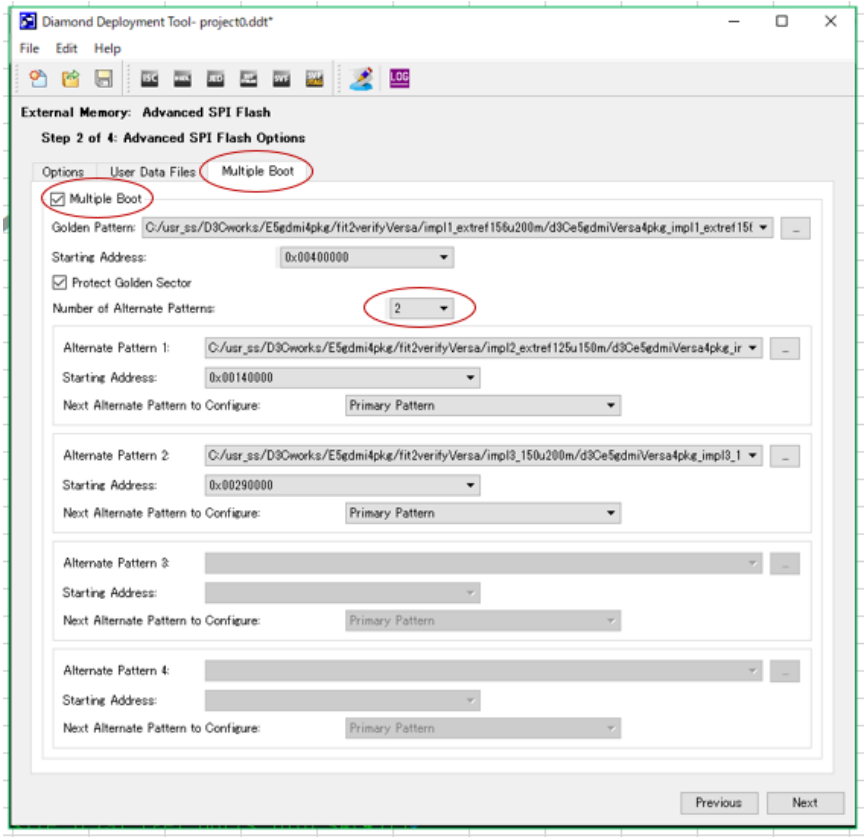


図 15-14. マルチブート用ファイル作成：ステップ 1



次のステップ 1（図 15-14）では ”ゴールデン ”ビットストリーム・ファイルを指定します。ステップ 2 で変更も可能です。指定後、ウィンドウ下部の『Next』ボタンで次に進みます。

図 15-15. マルチブート用ファイル作成：ステップ 2



ステップ２ではマルチブート・モードの指定、ビットストリーム・ファイルの指定を行います（図 15-15）。タブ［Multiple Boot］を選択して最上部の「Multiple Boot」チェックボックスをチェックしてイネーブルします。「Number of Alternate Patterns」のプルダウン（2 ～ 4）からゴールデン以外のパターン数を指定し、ぞれぞれのセルに対してファイルをブラウズして指定します。ゴールデンにはステップ１で与えたファイルが入っていますので、このステップで変更することも可能です。

指定パターン数に対して SPI フラッシュメモリーのサイズが不足する場合は、ウォーニング・メッセージが表示されますので、[Options] タブを選択して「SPI Flash Size」部を適宜変更します。「Starting Address」は特段の理由がない限り変更しないことを推奨します。

この後のステップはデュアルブートと同様です（図 15-11、図 15-12）。

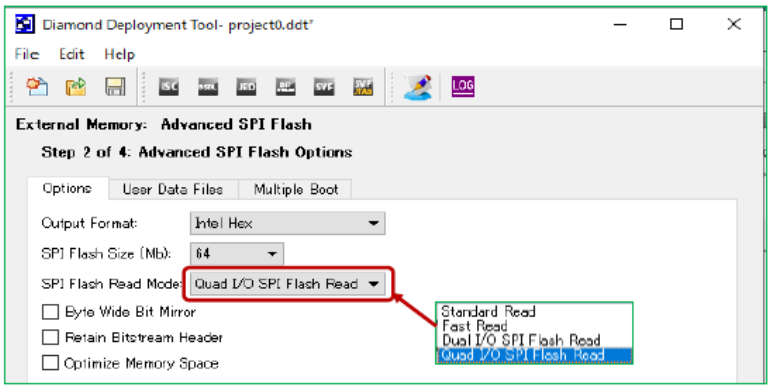
15.1.5 Dual/Quad SPI リード用 mcs ファイル生成：ECP5

ECP5 ファミリーは、MASTER SPI インターフェイスで外付け SPI フラッシュメモリーからコンフィグレーションする際に、データ線を二本（Dual）または四本（Quad）にすることができます（Dual/Quad SPI リード）。この場合にも SPI フラッシュメモリーに格納するビットストリーム・ファイルは単独ですが、やはりデプロイメント・ツールで mcs に変換します。

作業開始のウィンドウではマルチブートと同様で、「Function Type」を ”External Memory”、「Output File Type」を ”Advanced SPI Flash” にします（図 15-16）。次のステップ１でのファイル指定も同じです。

異なるのはステップ２で、のように「SPI Flash Read Mode」を ”Dual (Quad) I/O SPI Flash Read” にします。

図 15-16. Dual/Quad SPI リード用ファイル作成：ステップ２



ステップ３と４はデュアルブートと同様です。

15.1.6 MSPI/SSPI デイジーチェイン対応 SPI 用 mcs ファイルの生成：ECP5

ECP5 には種々コンフィグレーション・モードがありますが、図 15-17 のように MSPI モードの ECP5 を先頭にして SSPI モードの ECP5 を複数個、同一チェインにする構成（デイジーチェイン）があり得ます（各信号線にはプルアップ / ダウン抵抗が必要ですが割愛しています。DONE/INITn 信号線は Wired-OR 接続しています）。この場合に SPI フラッシュメモリーには各デバイスのビットストリーム・ファイルを結合した mcs ファイルを格納し、これはデプロイメント・ツールで作成します。

なお、本構成での各ビットストリーム・ファイルの作成は、Diamond において次のような設定が必要です。

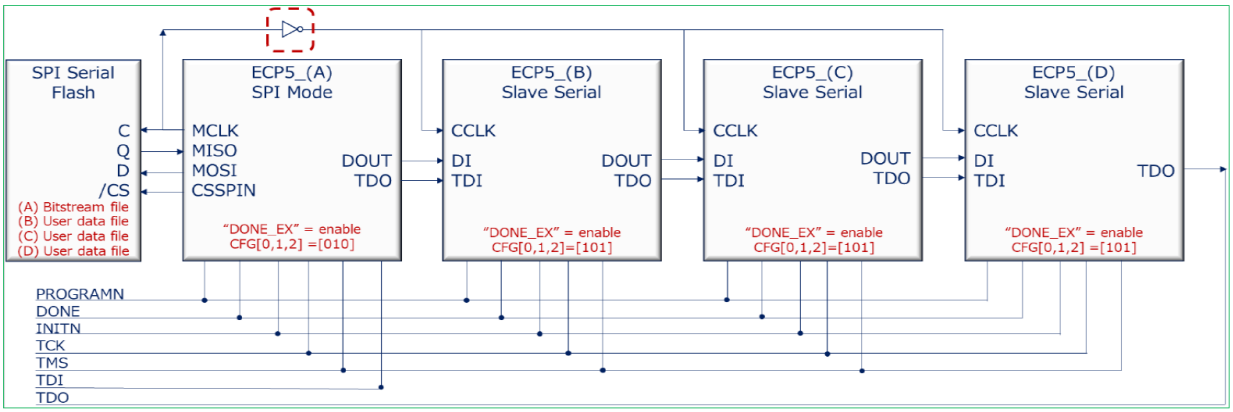
- SPI フラッシュメモリを接続する先頭の ECP5 はスプレッドシート・ビュー［Global Preferenecs］タブの sysCONFIG セクションで、MASPTER_SPI_PORT=ENABLE、DONE_EX=ON にします
- 二個目以降の ECP5 はスプレッドシート・ビュー［Global Preferenecs］タブの sysCONFIG セクションで、SLAVE_SPI_PORT=ENABLE、DONE_EX=ON にします
- アクティブなストラテジー設定を立ち上げ、”Bitstream” プロセスの ”Chain Mode” オプションを全

てのデバイスで [Bypass] にします

以上の設定でそれぞれ ”Export Files” プロセスまで実行してビットストリーム・ファイルを生成します。その後、これらファイルに対して第 15.4 節に記述する操作で MCLKB ビットを編集します。これら手順で準備した対象ファイルを用いて次のステップ以降を行います。

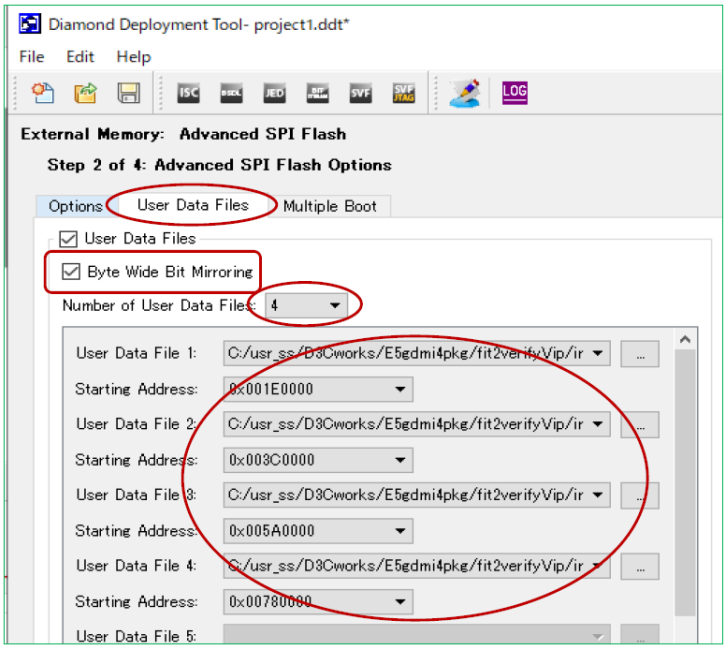
デプロイメント・ツールでの開始はマルチブートと同じように（図 15-13）、「Function Type」を ”External Memory”、「Output File Type」を ”Advanced SPI Flash” にします。ステップ 1 のファイル指定も図 15-14 と同様ですが、これには先頭の MSPI モードの ECP5 用ビットストリーム・ファイルを与えます。

図 15-17. ECP5 の MSPI/SSPI デイジーチェイン構成例



ステップ 2 が各ファイルの指定など詳細設定です（図 15-18）。[User Data Files] タブで「User Data Files」ボックスと「Byte Wide Bit Mirroring」ボックスを共にチェックします。「Number of User Data Files」プルダウンを操作し、SSPI モードのデバイス数に一致する値にします。値に相当する各フィールドがアクティブになりますので、それぞれブラウズ・ボタンをクリックして適切なビットストリーム・ファイルを指定します。

図 15-18. MSPI/SSPI チェイン用 mcs 生成のステップ 2 の例



「Starting Address」各セルは自動的に値が入りますので、ユーザーが編集する必要はありません。

[Options] タブ（図 15-18 では図示していません）の「SPI Flash Size」部で SPI フラッシュメモリーのサイ

ズを指定しますが、全ファイルのサイズに対して SPI フラッシュメモリーが小さい場合は、ウォーニング・メッセージが表示されますので、適宜変更します。

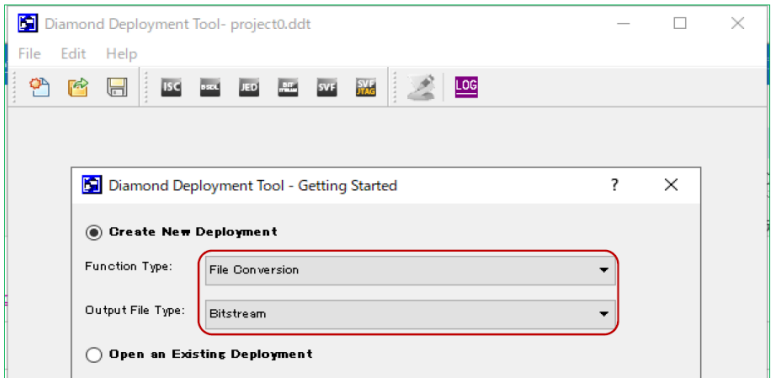
また、[Options] タブ内にも「Byte Wide Bit Mirror」オプションがありますが、こちらはチェックしないでおきます。次のステップ３の出力ファイル名の設定とステップ４ ”Generate” は他と同様です。

15.1.7 ビットストリーム・ファイルの暗号化

ビットストリーム・ファイル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（第 14.6.1 項に関連記述がありますので、ご参照ください）。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。

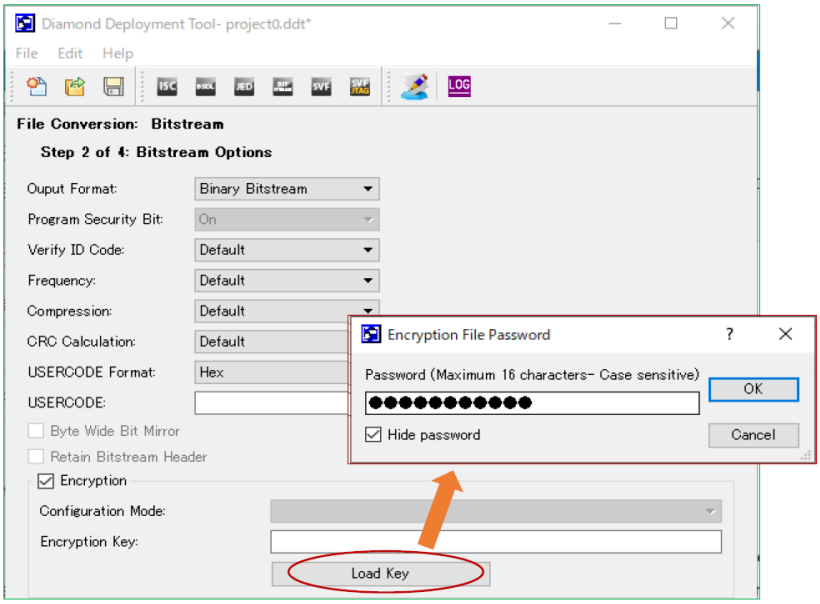
作業手順は次の通りです。まずデプロイメントツールを起動し、「Function Type」は ”File Conversion” に、「Output File Type」は ”Bitstream” を選択します（図 15-19）。既に作成済みデプロイメントツール・プロジェクトがあれば、下部の『Open an Existing Deploym』をチェックして ddt ファイルをブラウズすることで指定します。『OK』ボタンをクリックします。

図 15-19. 暗号化ステップの開始



次のステップは暗号化対象ファイルの指定ですが、これは他と同様の方法でブラウズして『Next』ボタンで進みます。次のステップ２はオプションの詳細設定です（図 15-20）。

図 15-20. 暗号化のステップ２

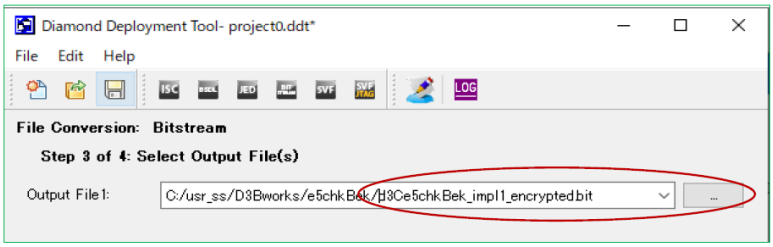


最下部の「Encryption」ボックスをチェックして有効にして暗号化キーを入力します。『Load Key』ボタンをクリックして表示されるファイル・ブラウザのウィンドウで暗号化キーファイル（.bek）を指定すると、図 15-20 内の小ウィンドウが表示されますので、暗号化キーファイルのパスワードを入力します。パスワードが一致すれば、『OK』をクリックして抜けた後に「Encryption Key」セルが ”●” で満たされます。『Next』ボタンで次に進みます。

パスワードが一致しないとこのステップを完了できませんので、ご注意下さい。また「Encryption Key」セルを直接編集することやパスワードの変更などはできません。パスワードの設定と暗号化キーファイルの生成は Diamond の ”Security Tool” で行います。第 14.6.3 項をご参照下さい。

ステップ 3 は出力ファイル名と場所の指定です（図 15-21）。暗号化したことが容易に判別できる名称にするようにします。『Next』で次に進みます。

図 15-21. 暗号化ステップ 3



ステップの最後が暗号化ビットストリームの生成です。右下の『Generate』ボタンか上部の アイコンをクリックします。他と同様に ”... excited successfully” と表示されれば正常終了です。

15.1.8 SPI フラッシュメモリー内にユーザーデータを共存

外付け SPI フラッシュメモリー内に、コンフィグレーションデータとユーザー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ありませんが、以下の基本的な点を押さえておきます。

- 使用するデバイス（ファミリーと規模）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
- デバイスによっては SPI アクセスの信号線はコンフィグレーション用ポートが共用できないため、注意深くピン配置を行う

SPI フラッシュメモリーのユーザーデータにアクセスするための手段は、ユーザーが用意します。共存させるためのツールとしてデプロイメント・ツールが使用できます。MSPI/SSPI チェインの場合と同様の手順ですが、図 15-18 においてビットストリーム・ファイルではなく、意図するユーザーデータを指定します。また、開始アドレスの変更はユーザー依存です。

15.2 フィーチャー行エディター（MachXO2/XO3L/XO3LF）

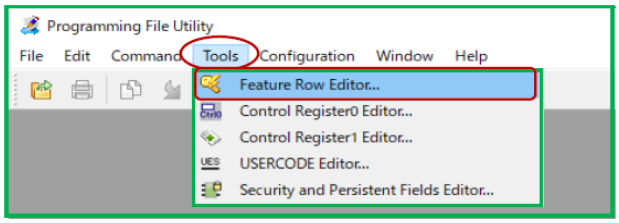
注：フィーチャー行のみの意図的な変更は開発時（及び量産出荷時）に限ることを強く推奨しています。

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、フィーチャー行（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。こ れは、特にフィーチャー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。

何らかのやむを得ない事由（含デバッグ等）でフィーチャー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ー行エディターが活用できます。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 15-22）、その後 **Tools → Feature Row Editor** を指定して行います。また Programming File Utility

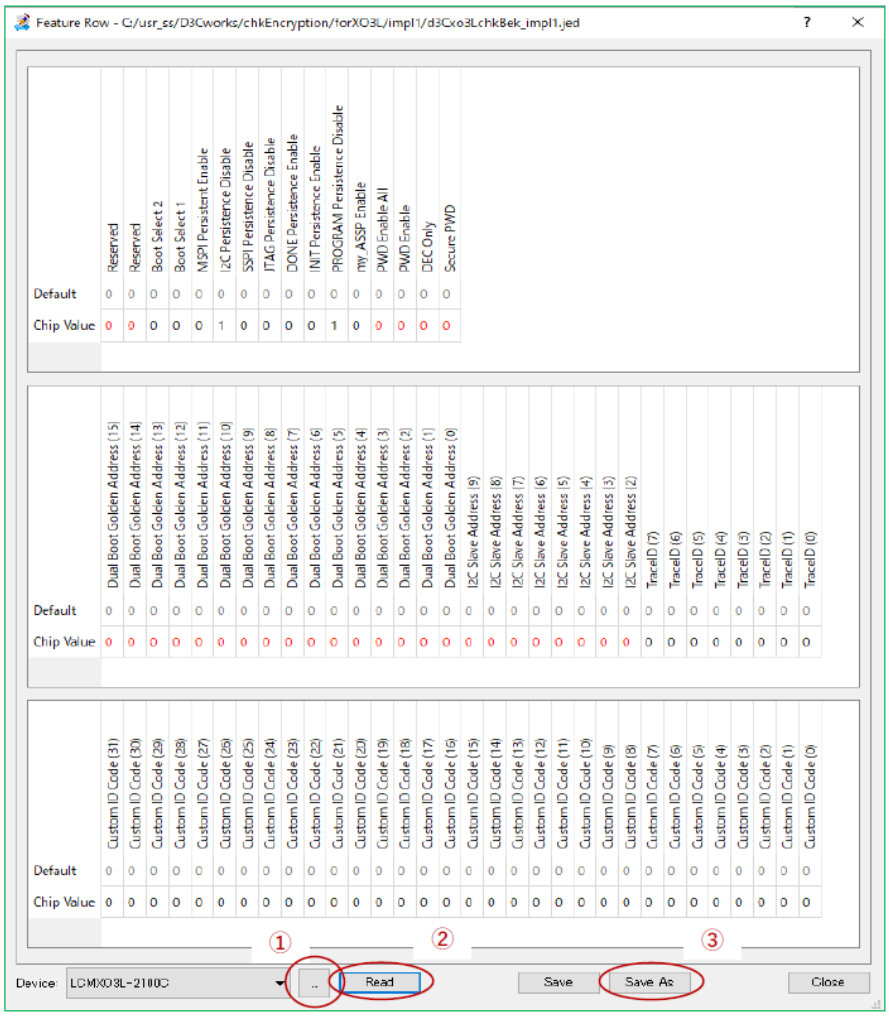
については、図 15-1 に示すと同様の手順で Windows のスタート メニューから立ち上げることもできます。

図 15-22. フィーチャー行エディターの起動



起動後、まず①ブラウズボタンで jed ファイルをブラウズ後に指定します。次に②『Read』ボタンをクリックすることでファイルからフィーチャー行情報を抽出すると、図 15-23 のように表示されます。

図 15-23. フィーチャー行エディター、jedec リード後の表示例（MachXO3L）



MachXO3D の場合は〈xxx_a.jed〉〈xxx_b.jed〉ファイルではなく、〈xxx.fea〉ファイルを対象に指定します。表示されるウィンドウは図 15-23 とはやや異なります。

黒色で表記されているビットのいず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、③『Save As』で Jedec ファイルとして別名で保存します。

なお、フィーチャー行エディターでは、第 16.3.2 項で記述する TraceID および Custom ID Code の編集ができます。図 15-23 の中に含まれていることが理解できます。

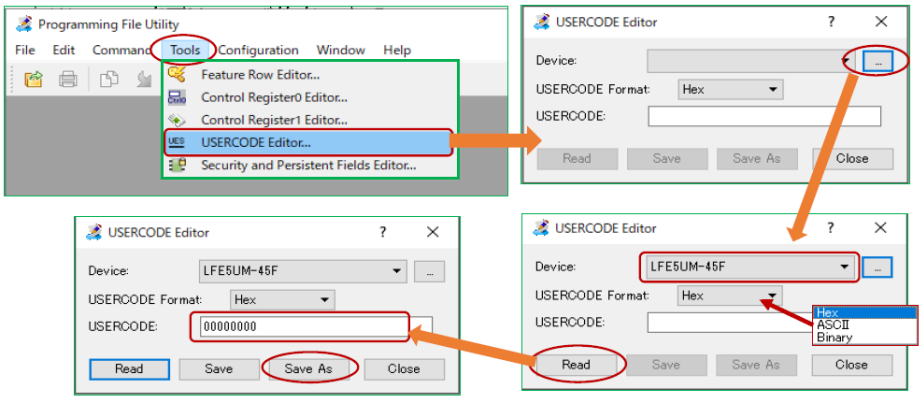
意図せず不明瞭な理解に基づくアクセスを行うと、不適切な値に書き換えてしまい(アクティブなコンフィグレーション・ポートが使用できなくなり)、結果的にその後のアップデートを行えなくなる状態になる可能性が否定できません。注意の上に注意が必要です。

15.3 USERCODE エディター

特にユーザーコードに限定して編集するツールが USERCODE エディターです。起動は図 15-24 のように[Tools] → [USERCODE Editor...] を選択します。

表示されるウィンドウの右上「Device」行右端のブラウズボタンで、意図するビットストリーム・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（図 15-24 右下)。『Read』ボタンをクリックして値を抽出して表示します（図 15-24 左下)。これで所望の値に編集し、『Save As』で別名保存します。デフォルトでは 16 進表記になっていますが、選択が可能です。

図 15-24. USERCODE エディターの起動から編集までの例

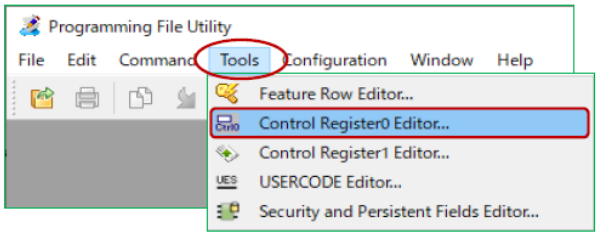


15.4 コントロール・レジスタ 0 エディター

第 15.1.6 項で記述した ECP5 MSPI/SSPI デイジーチェイン・コンフィグレーション用の mcs ファイル生成に関連して、チェインに接続される全ての ECP5 は、このコントロール・レジスタ 0 エディターを用いてビットストリーム・ファイルを編集する必要があります。

それ以外でユーザーが用いる必要があるケースは特に認識されていません。

図 15-25. コントロール・レジスタ 0 エディターの起動

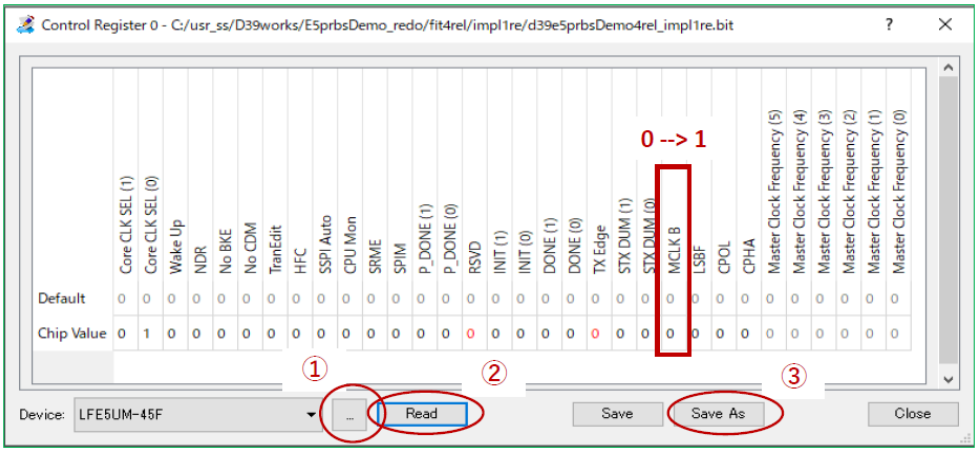


起動は Tools → Control Register0 Editor... と選択します（図 15-25)。

起動後、まず①ブラウズボタンで bit ファイルをブラウズ後に指定します。次に②『Read』ボタンをクリックすることで図 15-26 のようにファイルから情報が抽出され表示されます。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、③『Save As』で bit ファイルを別名で保存します。

ECP5 MSPI/SSPI チェインの構成時は、”MCLKB” ビットを ’1’ に変更する必要があります。編集後のビットファイルが、mcs ファイルとして結合する対象です。『Save』または『Save As』で保存して使用します。

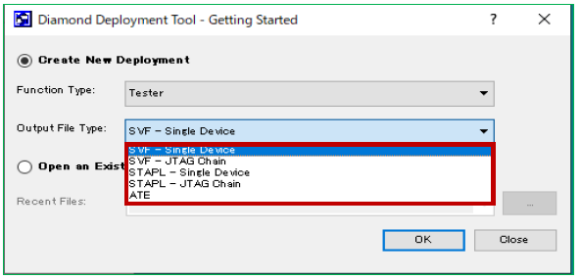
図 15-26. コントロール・レジスタ 0 エディターの作業例（ECP5）



15.5 ダウンロード・デバッガー

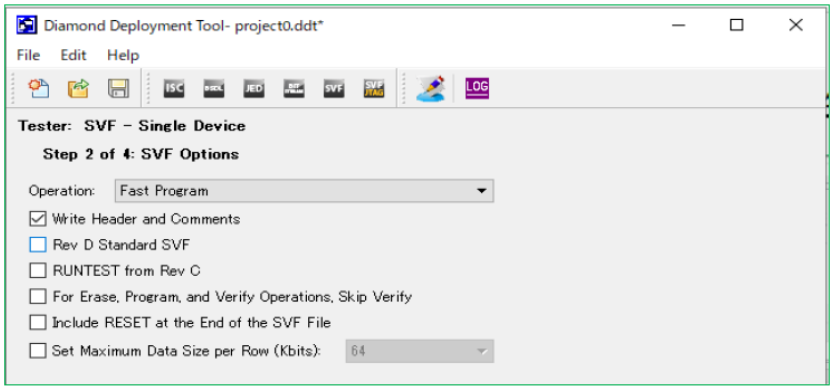
何らかの理由で JTAG プログラミングが失敗する場合、いったん書き込みファイル（.bit、.jed）を SVF ファイルに変換し、これを用いてデバッグを実行することによって原因を究明できる場合があります。ダウンロード・デバッガー（Download Debugger）は、ダウンロードケーブルで接続された実デバイスを動作させながら、SVF ファイルを実行するためのツールです。”svf (serial vector format)” は元来 PCB テスターで用いられる書式です。

図 15-27. デプロイメントツールのファンクションタイプ選択



SVF ファイルを未生成の場合、まずデプロイメントツールでフォーマット変換します。「Function Type」は ”Tester”、「Output File Type」は ”SVF” を選択します（図 15-27）。JTAG チェイン内にターゲットの単一デバイスのみでは ”SVF – Single Device” を、ターゲットを含む複数デバイスの JTAG チェインの場合は ”SVF – JTAG Chain” です。

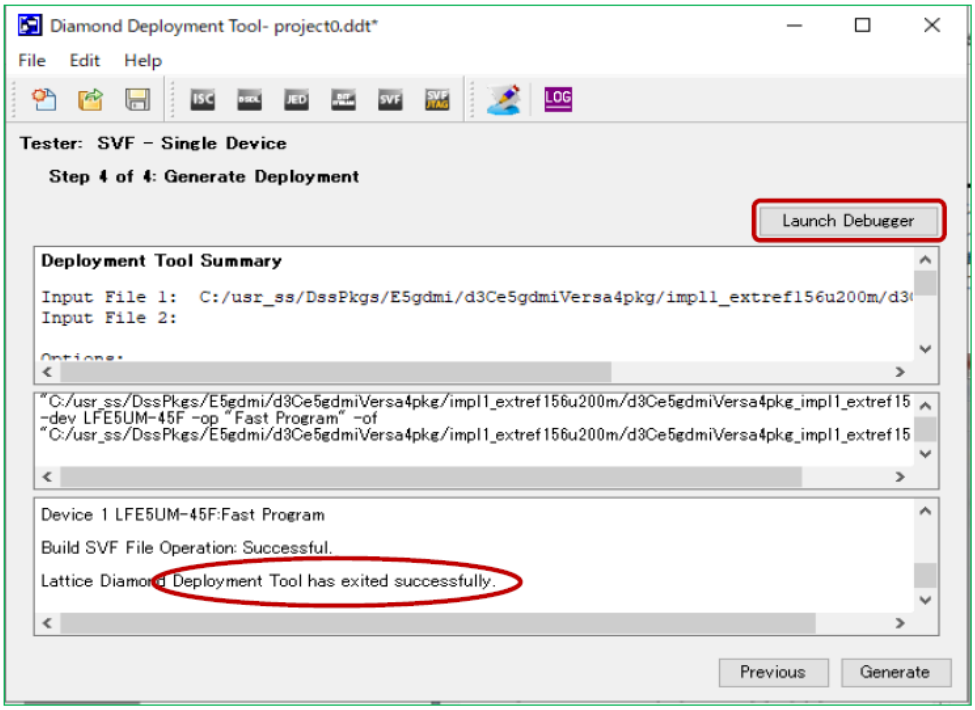
図 15-28. SVF 変換の各種オプション指定



ステップ 1 でソースファイルを指定後、ステップ 2 で図 15-28 のようにオペレーション（Operation）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。

その後ステップ 3 で出力ファイル名とフォルダーを指定し、ステップ 4 で変換を実行（Generate）します（図 15-29）。”... 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。

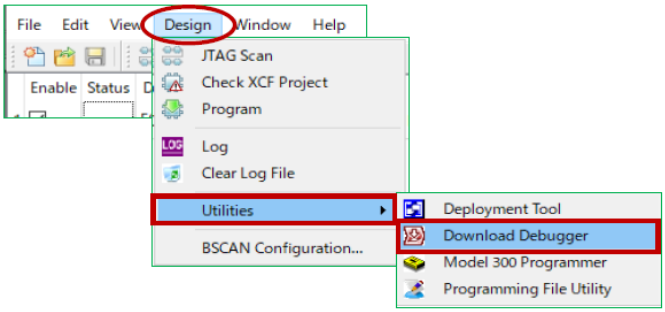
図 15-29. SVF ファイルの生成・変換例



以上で準備が整いましたのでダウンロード・デバッガーを起動します。図 15-29 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。表示は生成した SVF をロードした状態で立ち上がり、ブレークポイントを設定したりできるようになります（図 15-31）。

または一度デプロイメント・ツールを終了し、図 15-30 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ーが立ち上がりますので、File → Open とたど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 15-31 のような初期状態になります。

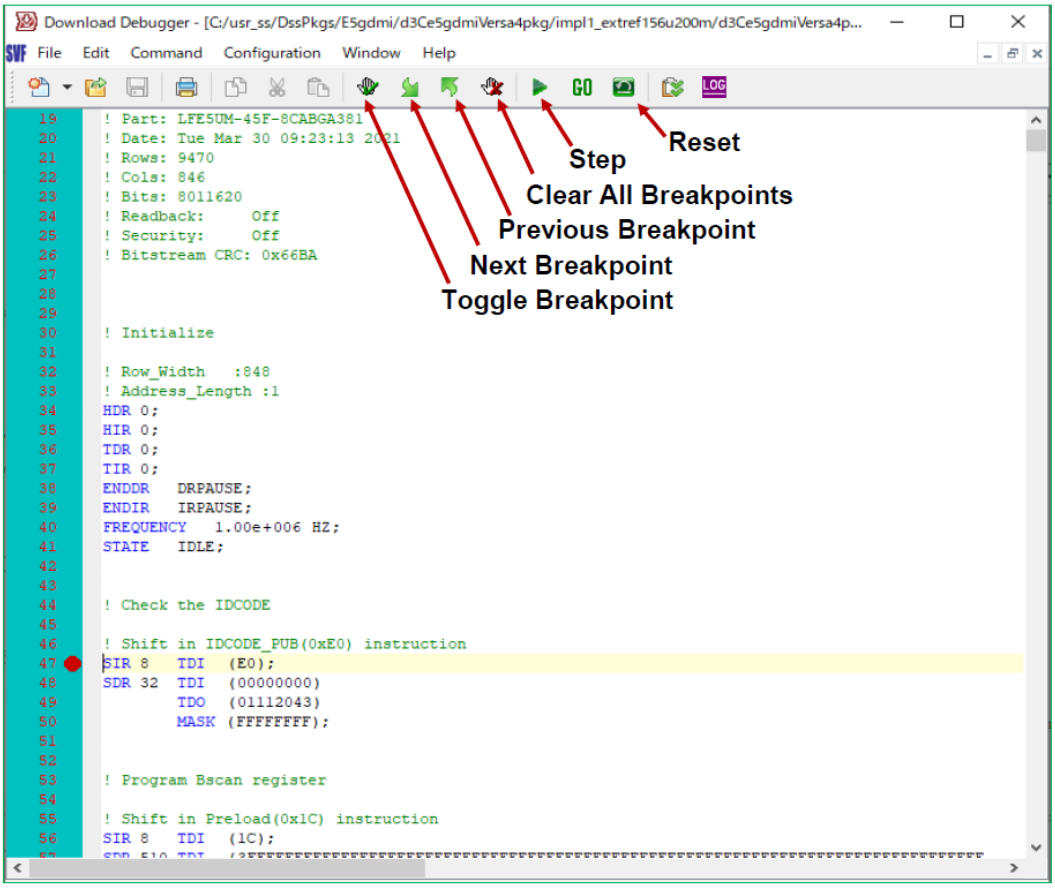
図 15-30. プログラマーからダウンロード・デバッガーを起動



デバッガーの操作は、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

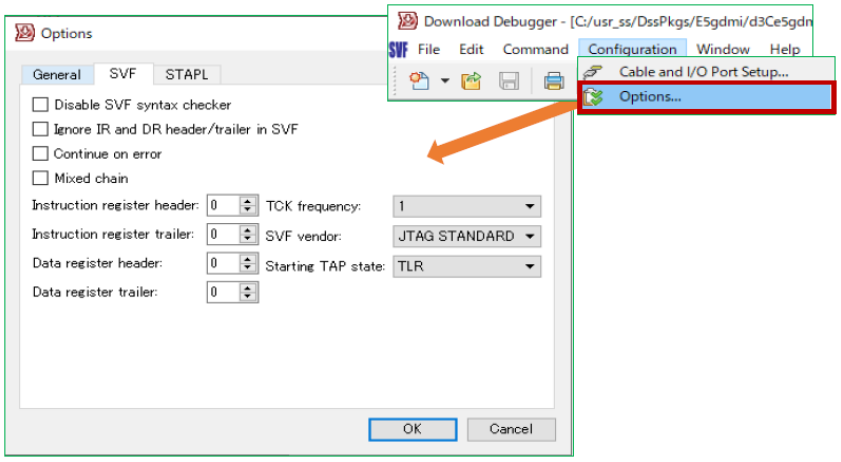
実行、などを行います。アイコンとオペレーションは図中に示す通りです。

図 15-31. SVF デバッガー表示例



なお、チェイン内に複数のデバイスがある場合は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・デバッガーのメニューで Debugger → Options... とたどり、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 15-32. 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：ヘッダー）と後ろ（trailer：トレイラー）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 15-32、赤枠）。ない場合は 0 です。図 15-33 にヘッダー長、トレイラー長の設定例を示します。

図 15-33. JTAG チェインの構成とレジスタ長の例

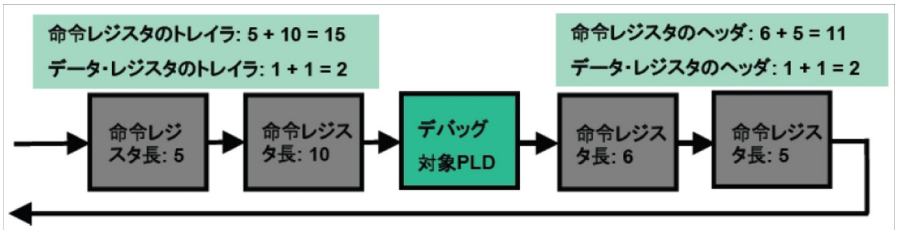


図 3-8 ヘッダ長・トレイラ長の設定例

--- *** ---