ボードサポートパッケージの構築

Bradford Harrison
Senior Technical Writer

マイクロテックリサーチでは、XRAYデバッガモニタとSpectra BSPという2つのタイプのボード・サポート・パッケージ(BSP)を提供しています。XRAYデバッガモニタ(XDM)は、オペレーティングシステムを必要としないシングルスレッド・アプリケーションのデバッグを行います。Spectra BSPは、Xトレースモニタ とXトレース・プロトコルを採用し、シングルスレッド・アプリケーション(非OSモードデバッギング)あるいはVTRX/OS上で動作するマルチスレッド・アプリケーション(OSモードデバッギング)のデバッグをサポートしています。ここでは、より複雑なSpectra BSPを中心に、両方のタイプのBSPについて述べていきます。

BSP開発ツール

組込みシステムの開発にはBSPが必要です。なぜなら、商品化されているマイクロプロッセサ搭載のターゲットボードは、そのボードを起動、初期化し、ホストシステムとのシリアル(RS232C)接続を行うだけのベーシックなデバッグモニタが搭載されているに過ぎないからです。従って、開発エンジニアは組込みソフトウェア開発のためのボードサポートサービスを自らが提供しなければなりません。
BSPは、あらかじめ組み込まれたものを購入するか、あるいは開発エンジニアやサードパーティのコンサルタントによって作られます。プリビルドBSPは、すべてのターゲットボードに対して用意されているわけではなく、特にカスタムハードウェアについてはそれが顕著です。従って、組込みシステムソフトウェアのベンダは、カスタムハードウェアに対するBSPを開発するためのツールを提供しています。
BSP開発ツールは、ボードやプロセッサの構成を幅広くサポートし、使い易いものでなければなりません。そして、開発エンジニアにはそのBSPを自由にカスタマイズできる能力が要求されます。これらのBSPは、メーカから販売された組込みシステムのソフトウェア開発環境全体をサポートするサービスを提供します。

XRAYデバッガモニタ(XDM)

XDMを作成するために、マイクロテックリサーチではモニタ・コンフィギュレーション・ツール(MCT)を提供しています。このメニュー方式のホスト・ユーティリティは高機能のデバッグモニタを作成します。作成されたモニタは、ターゲットボードに組み込まれた既存モニタによってダウンロードされます。開発エンジニアはXDMのスタートアドレスからgoステートメントを実行し、XDMがボードをコントロールしてシングルタスク・アプリケーションを開発するためのすべてのXRAYのコマンドおよびサービスをサポートします。
MCTは様々なターゲットボードを直接サポートします。MCTによって生成されたBSPは直ちにダウンロードされ、ターゲット上で実行されます。ターゲットボードがMCTによって直接サポートされていない場合、開発エンジニアは「最も類似した」(少なくともそのボードと同一ファミリのプロセッサ用に構成された)BSPを生成し、コンパイルしてターゲットにダウンロードする前にソースコードを変更します。このコードの変更方法の詳細を示すドキュメントはすべてマイクロテックリサーチから提供されますが、開発エンジニアはプロセッサと周辺デバイスに関する情報を自分で調べ利用しなければなりません。
XDM実行部は、ブートおよび初期化コードとリンクされ、完全なBSPパッケージとなります。このBSPは、メーカーのデバッグモニタを組み込む必要がないように、PROMに書き込まれます。ブートおよび初期化コードは、ボードメーカによって供給されるドキュメンテーションあるいはソフトウェアを基に書かれます。ブートおよび初期化コードはプログラマがXRAYの逆アセンブル機能を使って、供給されたデバッグモニタから得ることがしばしばあります。次いでそれをアセンブルし、XDMとリンクします。

Spectra BSP

Spectra BSPはXDMよりも多くのサービスを提供します。これはシングルスレッドのアプリケーションあるいはOS上で動作するマルチスレッドのアプリケーションをサポートし、デバッグします。このOSにはVRTX/OSが多く用いられます(これは他のものでもよいのですが)。
Spectra BSPの開発では、マイクロテックリサーチのBSPBuilder開発ツールおよび手法、インクルードフォーム(テンプレート)、手順、メイクファイル、開発テスト、ヘッダファイルそしてライブラリを使用します。
フォーム、手順およびテストは、比較的一般的なものです。その他のターゲット仕様は、BSPBuilderによって供給されるライブラリと、BSPBuilderが各種のプロセッサやI/Oデバイスをサポートするために供給する組込み(バイナリ)ドライバで構成されます。開発者が、サポートされているプロセッサおよびI/Oデバイスを含むボードにBSPを組み込む場合、プリビルドドライバを使用するとBSPの開発時間を大幅に短縮することができます。
BSPBuilderフォームは、開発エンジニアがプリビルドドライバを使用しない場合、デバイスドライバを作成するために使用されます。各種のプロセッサおよびI/Oデバイスに対するシリアル、イーサネット、

タイマおよび共有メモリドライバは、このフォームを使って生成され、バイナリドライバと結合されて1つのBSPが形成されます。

BSPBuilderの開発プロセス

BSPBuilderのBSP開発は、図1に示すように繰り返されます。開発エンジニアは通常シリアルドライバから始めますが、ボード上にシリアルポートが1つしかない場合は、イーサネットドライバを最初に開発することもしばしばあります。シリアルポートが2つある場合は、一方のポートに対するシリアルドライバを開発中に、もう一方のポートをボードのデバッグモニタ用に接続したままにしておくことができます。シリアルポートが1つしかない場合には、イーサネット・デバイスドライバの開発中、シリアルポートをデバッグモニタ用にそのまま使用することができます。

どちらの場合も、目的は、動作可能なSpectra BSPを組込み、動作可能なSpectraブリッジを確立するために動作可能なドライバを開発することにあります。Spectraブリッジとは、ターゲット上で動作するXtraceデーモンとホスト上で動作するターゲットマネジャーの接続のことを言います。Spectraブリッジが使用可能であれば、シリアル、イーサネットあるいは共有メモリデバイスをXtraceとアプリケーション間で共有でき、高レベルのデバッグが可能になります。

チップ対応したデバイスドライバがボード用に存在する場合は、バイナリドライバが使用されます。そうでない場合は、ドライバを開発するためにフォームが使用されます。どちらの場合も、ドライバは次に、BSPBuilderが供給するメイクファイルを使って適当なBSPBuilderテストコード、ファイルおよびライブラリとリンクします。これらのファイルとライブラリには次のものが含まれます。

- logio.lib － Spectraのロジカル I/O (*logio*) レイヤが実装されたモジュールを含みます。
- board.c － ボードの初期化コードを含みます。**board.c**は、ボードの設定で変更できないような部分を扱うコードを含みます。これらの関数はボードのデバイスがインストールされていてもいなくても実行する必要があります。
- devcnfg.c － ボード上のデバイスにプログラムするのに必要な情報を含むコンフィギュレーションデータ構造を定義します。たとえば、あるデータ構造は与えられたシリアル デバイスのデバイスのタイプ (パケットまたはtty)、クロック速度、ボーレート、パリティ、ストップビット、レシーブベクタ、トランスミットベクタ、エラーベクタを指定します。またこの構造は、ポートアドレスと割り込みを可能にしたり禁止する関数を指定します。それぞれのデバイス用に、**devcnfg.c** はデバイスディスクプリタと*logio*メソッドを宣言し、また、ボード上のすべてのデバイスを*logio*デバイスとしてインストールするための関数を定義します。開発エンジニアは個々のボード用に、このファイルを修正することが必要です。
- boot.lib － ターゲットボードを初期化し、電源投入またはリセット後に既知の状態にするコードを含みます。このライブラリは、*logio* を初期化し、*logio* デバイスとしてデバイスを作成して初期化し、コンソールをセットアップし、メモリマップをROMからRAMへコピーし、メモリマップ内のオーバーラップをチェックして、ブートアイテムリストを作成します。
- cpu.lib － プロセッサ固有の機能をサポートします。これは、キャッシュおよびMMU関数、レジスタセット(浮動小数点を含む)および割り込み制御関数を定義します。
- packt.lib － Xtrace プロトコルによって使用されるシリアルライン上のパケット化された通信をサポートするコードを含みます。このライブラリは、パケット内のデータをカプセル化し、チェックサムを計算して比較し、データをパケットから取り出す関数を含みます。

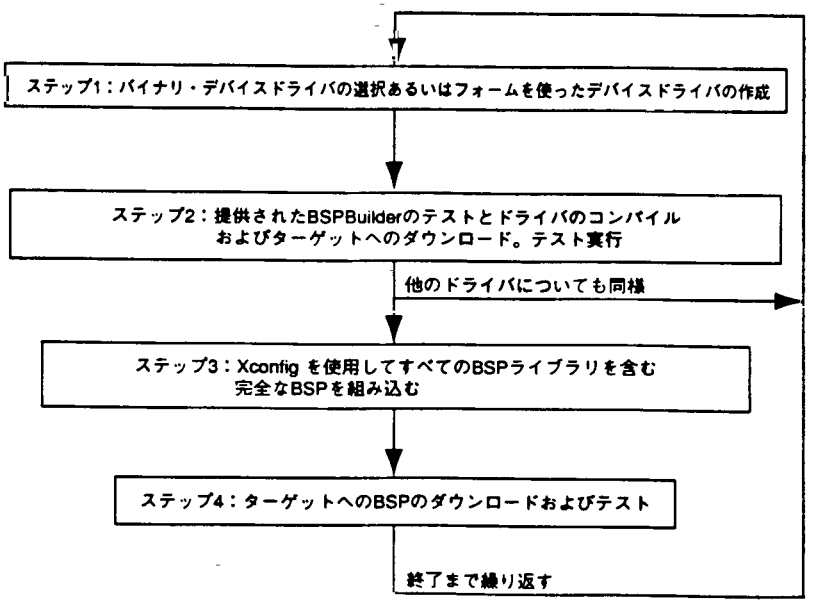

図1. Spectra BSP開発プロセス

実行可能ファイルは、既存のターゲットボード上のデバッグモニタを使ってターゲットにダウンロードされます。デバイスドライバは、ターゲットのテストコードとホスト上のツールを組み合わせてテストされます。

最初のイーサネットあるいはシリアルドライバに問題がある(ターゲットからの応答が全くない)場合、BSPBuilderのメモリコンソール・デバッギング機能を使うことができます。これはメモリに診断するものを書き込むために、ドライバコードに**printf** 命令を挿入できるようにするものです。ドライバが誤動作した場合、その誤動作の原因を突き止めるために、デバッグモニタを使ってメモリを調べることができます。

ドライバのチェックがすむと、次のドライバに進むか、あるいはそのドライバと、完全なBSPを組み込むために必要な残りのBSPBuilderのファイルとライブラリを含むBSPを作成します。これらのファイルとライブラリは、次のようなものです。

- crt0.s － ターゲットボードを立ち上げるのに必要なすべてのコードを含みます。立ち上げやリセット時に使用するスタックを定義し、プログラムカウンタとスタックポインタを初期化し、すべての割り込みを禁止します。また、キャッシュメモリ(存在しアクセス可能ならば) を消去し無効にした後、RAMをクリアします。ボードがデバッグPROMと一緒に提供される場合、**crt0.s** 内のコードはデバッグPROM内に存在するコードとほとんど同じになります。
- bootcnfg.c － テンプレート**bootcnfg.tpl** からXconfig によって生成され、関数、定数データ、テーブル、ブートスタック、Xtrace の作業空間およびその他の環境設定値を持つブート関係の事柄を宣言します。
- devices.c － テンプレート**devices.tpl** からXconfig によって生成され、ボードのデバイスのテーブルを宣言します。それぞれのデバイスに対して、テーブルはデバイス名(デバイスを識別するための文字列)とそれをアクセスするための*logio*メソッドとを一致させます。*logio* 方式は2つの別のデータ構造を示すポインタを持つデータ構造です。これらのうち最初のものは、あるデバイスファミリのメンバに有効なオペレーションを定義します。2番目のものは、指定されたデバイスに関するすべての情報を含みます。デバイスは、最初のデータ構造によって特徴づけられたデバイスのクラス(タイマ、シリアル、イーサネットまたは共有メモリ)に属します。
- vtdm.lib － Xtrace をカプセル化します。このライブラリはターゲット上のXtrace デーモン(デバッグサーバとしても知られている) として実行する機能を持っています。デバッグサーバは、Spectraブリッジデバイスを通して(ホスト上で動作する)ターゲットマネージャと通信します。これら3つの要素であるデバッグサーバ、Spectraブリッジ、ターゲットマネージャは、Spectraデバッグ接続を構成します。
- router.lib － Spectraブリッジを介してメッセージを転送する機能があります。このライブラリは、シングルボードとマルチボードのどちらの構成でも使用でき、アプリケーションに対してターゲット上で動作するデバッグサーバ(Xtrace)とSpectraブリッジを共有できるようにします。

特定のボードに対してカスタマイズされる主なファイルは、ボードの立ち上げ、初期化およびデバイス構成にかかわる**crt0.s**、**board.c**、および**devcnfg.c**です。

完全なBSPはダウンロードされテストされます。このBSPは有効なシリアルあるいはイーサネットドライバを1つだけ持ち、有効なSpectraブリッジを確立するために使われます。ブリッジを通してターゲットマネージャとコネクションサーバを接続することができ、BSPと共に働くSpectraホストツールセットを使用することができます。

この時点で多くの開発エンジニアは、既存のボード・デバッグモニタを動作させなくてもすむように、BSPを含むPROMを作成しようとします。

開発プロセスは、すべてのデバイスドライバが動作し、BSPがすべてテストされるまで繰り返されます。最後のテストとして、OSが動作してアプリケーションを開発するのにBSPが使われる前に、BSPをチェックします。

BSPBuilderテスト

BSPBuilderでは、パケットの送受信テストと同様に、キャラクタの送受信テストを行うコードを含むいくつかのレベルのテストが採用されています。ホスト側の**rstest** ユーティリティは、Spectraブリッジが必要であるコネクションサーバを使わずに、シリアルリンクを介してシリアルドライバとパケットの送受信を行うために提供されます。ほとんどのUNIXシステムに含まれているる**ethertst** ユーティリティは、イーサネット・ドライバをテストします。

ドライバが正しく動作し、BSPがXconfigユーティリティによって組み込まれていれば、BSPはターゲットにダウンロードされ、標準Spectraホストツールを使用してBSPがテストされます。たとえば、Spectraデバッガ用のXSHあるいはXRAYを使用してレジスタをチェックしたり、BSPのサービスを使ったメモリロケーションを調べることができます。ターゲットマネージャは、ホスト上で動作させる必要があり、また、シリアル接続を通すならば、コネクションサーバも動作させなければなりません。

BSPBuilderはこの他に、タイマと共有メモリドライバに関するテストとシリアルコンソール・テストを含みます。

logio

SpectraBSPの開発の難所は、*logio* です。*logio*は関数ライブラリ **logio.lib** として提供されているので、デバイス指定コールではなく、ロジカル関数コールによって動作するデバイス・クラスとしてデバイスを扱います。*logio* デバイスが作成されると、論理的に他の同様のデバイスと同じように扱うことができます。
logio は、BSPに対して３つの重要なサービスを提供します。

1．BSP割り込み処理(あるいは割り込みに対する問い合わせ機能)
2．BSPの初期化
3．アプリケーション、ISRまたはその他のソフトウェアからのBSPサービスへのアクセス

Xtrace 、すなわちSpectraデバッグモニタは、すべての割り込みについて最初に反応します。この機能はXtrace とアプリケーションの間でブリッジを共有するために必要となります。*logio* はXtrace が*logio* 割り込み処理をコールするときだけ、割り込みと見なされます。*logio* のデフォルトの割り込み処理は、ベクタ毎に発生したそれぞれのロジカルイベント（割り込みソース）に対する処理をコールします。これらのロジカルイベント処理によって、何が割り込みを発生させたか、またそれは指定されたイベントかどうかがチェックされます。イベントに応じて発生した割り込みが処理されている場合、その関数は、該当するISRをコールし、**TRUE**を返します。その他のイベントが割り込みの原因になっている場合、その関数は、**FALSE**を返します。
logio の初期化ルーチンは、ロジカルデバイスを作成あるいは削除し、デバイスIDを返し、デバイスを初期化し、ドライバによって送受信されるデータを蓄えるための固定サイズのバッファのプールを初期化します。
logio はアプリケーションやISRがデバイスを操作するのに使用される以下の７つの標準関数を提供します。

- initialize
- read
- write
- getmsg
- putmsg
- control
- poll

これらの標準関数は、４つの「インターフェース」ファイル、すなわち、それぞれシリアル、イーサネット、タイマおよび共有メモリドライバ用に、**serial_2.o**、**ether_1.o**、**timer_1.o** 、**shmem_1.o** の各々の中に収められています。ソースコードも参照用としてのみ供給されます。ドライバが作成されるときに、特定のバイナリが特定のドライバフォームで開発されコンパイルされたソースコードとリンクされます。(提供されたバイナリドライバを使用している場合は、既にコンパイル済みのドライバで供給されているので、このコードを使う必要はありません。)
ドライバフォーム、すなわち、それぞれシリアル、イーサネット、タイマおよび共有メモリドライバ用の**sform**、**eform**、**tform** そして**vmeform** は、開発エンジニアによって記述されるデバイス特有の関数用の「ボイラープレート」コードを提供します。それぞれのドライバ開発用に多くの関数があり、それらはファンクションオペレーション(FOPS)テーブルを使用する７つの高レベル関数によってアクセスされます。FOPSテーブルは、適切なフォームから開発され、適切なインターフェースファイルでコンパイルされたデバイス特有の関数へのポインタから成っています。低レベルのデバイス特有の関数と高レベルインターフェース関数との分類については、**externio**を参照してください。
開発エンジニアは、そのフォームで開発中に関数へのアクセス方法を正確に知るための、インターフェース・ファイル用に提供されたソースコードをしばしば必要としますが、そのフォームそのものにそれぞれのデバイスに対する特定のコードを提供するのに役立つコメントが含まれています。コンパイル済みのデバイスドライバが、そのデバイスに対して動作可能であれば、理論上そのプロセスは簡単なものになり、それを上述したライブラリとファイルにリンクするだけですみます。

結論

BSPが開発され、テストされ、アプリケーション開発用にプラットフォームとして使用されるようになると、さらに変更が必要となる場合もしばしばあります。すなわち、Xtrace がBSPから除かれたり、*Configuration Tool User's Guide and Reference* で述べられる手順に従って、BSPが必要とするメモリ容量が減らされる場合があるのです。
BSPBuilderは、現在、プロセッサ・ライブラリとバイナリ・デバイスドライバを持つMotorola 68KとPowerPC プロセッサファミリをサポートしています。しかしながら、その方式は一般的で、他のプロセッサやデバイスに対するBSPを開発するために、その書式や*logio* コールを使用することができます。

マイクロテックリサーチのコンサルティングサービスでは、カスタムのターゲットボードに対するBSPの開発を支援するために開発エンジニアからのお問い合わせに応じています。またBSPBuilderのトレーニングクラスも設けています。