参 考 資 料

TMS320DM355 DVEVM
（TMDSEVM355 Evaluation Module Kit）

入門マニュアル

Literature Number: JAJU096
2008.9

評価用ボード、キットに関する重要なお知らせ

テキサス・インスツルメンツ(以下、TIと言います)は、同梱された製品を以下の条件で提供いたします。

本評価用ボードないしキットは、**技術開発、デモンストレーション、若しくは評価目的にのみ**使用されると想定されています。従って、TI は、本品が一般的消費者のための完成品であるとは見做していません。本品を取り扱う方は、電子工学に関する実務経験を有し、且つ良識ある技術的実務基準に従って取り扱う方でなければなりません。それゆえ、本品は、半導体集積回路製品や回路基盤を組み込んだ最終製品において通常要求されるような製造物安全や環境保全のための手段を含む設計上、販売上、若しくは製造上の保護的措置に関しては、未完成品であります。

本評価用ボードないしキットには、電磁気適合性に関するEUの指令、制限下にある物質(RoHS)、リサイクル(WEEE)、FCC、CE、UL等に関する基準は適用されませんので、従って、これらの指令若しくはその他の指令の技術的要求事項には適合しない可能性があります。

もし、本評価用ボードないしキットがユーザーズ・ガイドに示された仕様に合致しない場合は、本品の送付から30日以内に返品して頂ければ本品に対して支払われた代金の全額を返金いたします。本保証は、TI が本品のお客様に対して提供する唯一の保証であり、商品性があることの保証、特定目的に合致することの保証を含めた明示的保証、黙示的保証、法定の保証その他ありとあらゆる保証を排除して適用される保証であります。

本品の使用者は、本品を適正且つ安全に使用することについての全責任を負うものとします。さらに、もし万一使用者による本品の取扱いにより TI が何らかの請求、訴え等を提起された場合は、TI に補償を提供するものとします。本品は、開放的構造になっているため、使用者は、その責任をもって、静電気放電（ESD）に関する適切な予防対策で必要なもの全てを取らなくてはならないものとします。

上記に規定された補償を除き、いずれの当事者も、間接的、特別的、偶発的、派生的損害については責任を負わないものとします。

TIは、現在、多数の顧客と本品に関して取引を行っているため、TIとお客様(貴社、貴殿)との本品に関する取引は、排他的なものではないものとします。

TI は、**本品を利用するお客様の製品に関する支援、お客様の製品設計、ソフトウェアが動くかどうか、特許侵害、もしくはここに記載されている役務の提供については一切責任を負わないものとします。**

本品を取り扱う前に、必ず、ユーザーズ・ガイドをご覧下さい。とりわけ、ユーザーズ・ガイドの中の「警告と禁止事項(Warning and Restrictions)」に関するお知らせをご覧下さい。そのお知らせには、温度と電圧に関する安全についての重要な情報が含まれています。TIの環境ないし安全に関するプログラムについての追加的情報を得るためには、TI のアプリケーション・エンジニアに連絡して頂くか、若しくはTI のウェッブ・サイトwww.tij.co.jp/jcorp/docs/esh/をご覧下さい。

TIは、本品の提供によって、本品若しくは役務が使用され得る若しくは使用されているところの機械、方法、組み合わせをカバーする若しくはそれらに関する特許、その他の知的財産権を許諾するものではないものとします。

FCCに関する警告

本評価ボードないしキットは、**技術開発、デモンストレーション、若しくは評価目的にのみ**使用されると想定されており、従って TI は、本品が一般的消費者のための完成品であるとは見做していません。本品は、高周波（RF）エネルギーを発生、使用、かつ放射し得ることがあり、且つ電波干渉に対抗するための適切な保護を提供する目的で設定された FCC 規則第 15 章に従ったコンピューティング・デバイスの制限に適合するか否かの試験は行われておりません。本装置を、無線通信に対する干渉が起り得る他の環境下で操作する場合は、使用者は、自らの費用により、当該干渉を是正するために必要とされる何らかの手段を取らなくてはならないものとします。

最初にお読みください

本書について

DVEVM（デジタルビデオ評価モジュール）は、DaVinci アーキテクチャをご覧いただき、マルチメディア・エンジンとしての DaVinci の能力と性能をユーザに評価していただくための評価プラットフォームです。

本書では、ボードに関する概要と付属ソフトウェアについて説明します。DVEVM について「理解を深める」最初の資料として、読者の方々がご使用になることを想定しています。他の資料は、さらに理解を深めるためにご利用ください。このような資料の完全なリストについては、この製品に同梱している DVEVM の資料目録を参照してください。

表記規則

本書では、次の表記規則を使用します。

- プログラム・リスト、プログラム例、および対話表示は、タイプライタの活字に似た特殊な活字（mono-spaced font）で示してあります。例は、強調のため、ボールド（**bold**）で示してあります。対話表示についても、ユーザが入力するコマンドとシステムが表示する項目（プロンプト、コマンド出力、エラー・メッセージなど）と区別するために、ボールド（**bold**）で示しています。
- 大括弧（[および]）は、オプションのパラメータを識別します。パラメータを使用する場合、指定内容はこの括弧内に入力します。大括弧が**ボールド**でない限り、括弧そのものは入力する必要がありません。

商標

Texas Instruments ロゴおよび Texas Instruments は、Texas Instruments の登録商標です。TI、DaVinci、DaVinci ロゴ、XDS、Code Composer、Code Composer Studio、Probe Point、Code Explorer、DSP/BIOS、RTDX、Online DSP Lab、DaVinci、TMS320、TMS320C54x、TMS320C55x、TMS320C62x、TMS320C64x、TMS320C67x、TMS320C5000、および TMS320C6000 は、Texas Instruments の商標です。

MS-DOS、Windows、および Windows NT は、Microsoft Corporation の商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

Linux は Linus Torvalds の登録商標です。

Solaris、SunOS、および Java は、Sun Microsystems, Inc. の商標または登録商標です。

その他のすべてのブランド名、製品名、およびサービス名は、それぞれの会社または団体の商標あるいは登録商標です。

注意

製品に同梱されているペリフェラルの型番や仕様は変更される場合があります。

目次

第 1 章

DVEVM の概要

本章では、DVEVM（デジタルビデオ評価モジュール）の概要を説明します。

項目 **ページ**

1.1 キットの内容

DVEVM キットには、次のハードウェアが付属しています。これらのコンポーネントの接続方法については、2.1 節「ハードウェアのセットアップ方法」で説明します。

- **EVM ボード**。このボードにはデジタル・メディア向けシステムオンチップ DaVinci TMS320DM355 が搭載されています。
- **ケーブル**。開発を行うために使用する、シリアル&イーサネット・ケーブルがキットに付属しています。
- **赤外線リモート・コントロール**（Phillips）。このユニバーサル・リモート・コントロールは、アプリケーションのデモを行うためのユーザ・インターフェイスを提供するために組み込まれています。

DVEVM キットには、次のソフトウェア CD-ROM も付属しています。ソフトウェア・コンポーネントの使用方法については、第 4 章を参照してください。

- DaVinci デジタルソフトウェア開発キット。
- MontaVista Linux Pro v4.0.1 ターゲットの TI DaVinci デモ・バージョン。
- MontaVista Linux Pro v4.0.1 ツールの TI DaVinci デモ・バージョン。
- A/V メディアクリップ
- Spectrum Digital EVM Tools

1.2 ボードの構成

EVM は、マルチメディア・アプリケーションで利用可能なペリフェラルが組み込まれた状態になっています。次のブロック図に、主なハードウェア・コンポーネントを示します。

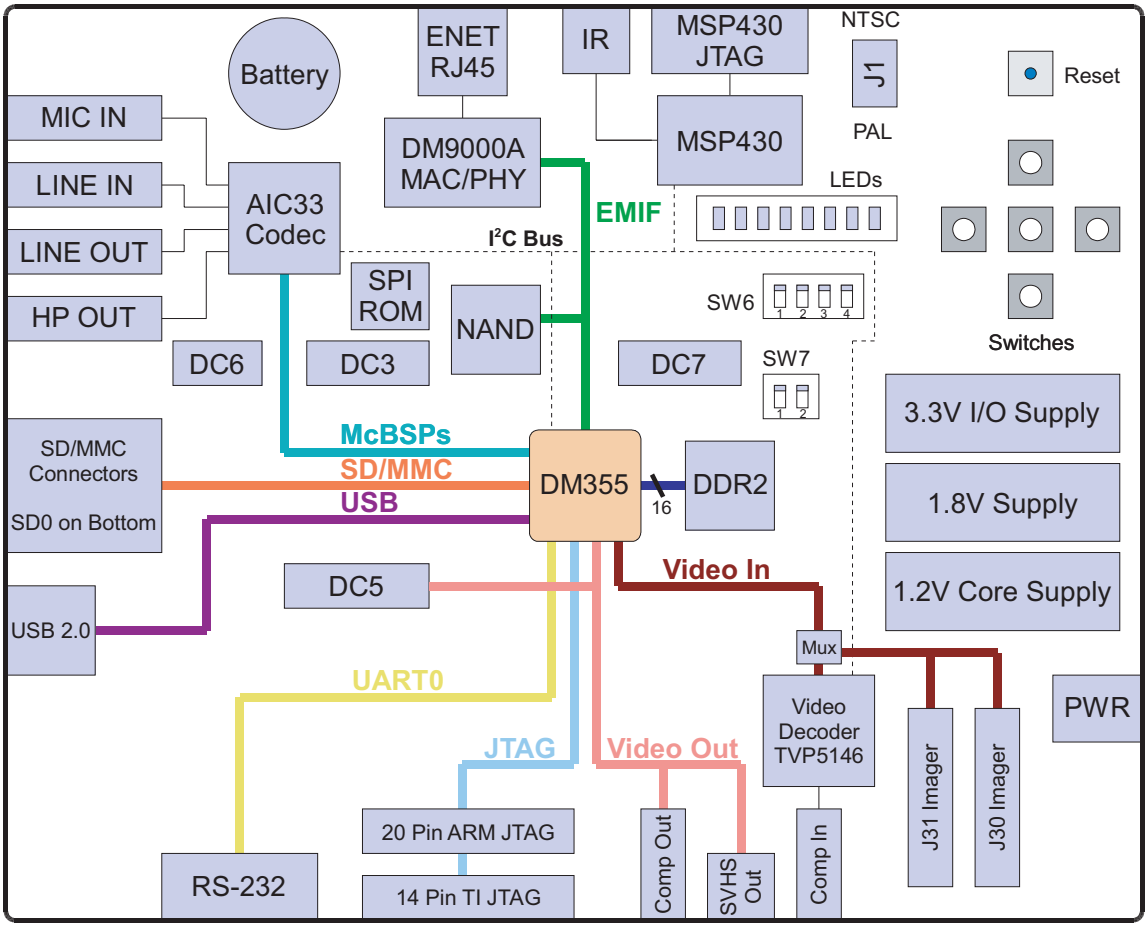


Diagram provided courtesy of Spectrum Digital Inc.

図 1-1. DVEVM ハードウェア・ブロック図

DVEVM ハードウェアの詳細は、DaVinci EVM の Web サイト（http://c6000.spectrumdigital.com/evmdm355/）をご覧ください。

DaVinci EVM には、ボードに電力が供給されない時に MSP430 のリアルタイム・クロックにバックアップ電力を供給するための、バッテリ・ホルダーがあります。バッテリはこのキットには付属していません。推奨バッテリのパーツ番号については、Spectrum Digital 社の DaVinci EVM 技術資料を参照してください。

1.3 次に行うこと

DVEVM を評価し、DM355 用のアプリケーションの開発を始めるためには、まず本書をお読みください。本書では、ハードウェアを接続することからはじめて、ソフトウェアをテストし、アプリケーションを開発するところまでを順に説明します。

DaVinci テクノロジーおよび DM355 アーキテクチャの詳細を理解している場合には、次の内容をご覧ください。

- Spectrum Digital 社の Web サイト：
  http://c6000.spectrumdigital.com/evmdm355/
- TI DaVinci ソフトウェア・アップデート：
  http://www.ti.com/dvevmupdates
- DaVinci プロセッサ情報を共有する TI Linux コミュニティ：
  http://linux.davincidsp.com
- 『Codec Engine Application Developer's Guide』（資料番号 SPRUE67）
- TI DaVinci テクノロジー・デベロッパーズ・ウィキ：
  http://wiki.davincidsp.com
- DVEVM に付属する CD-ROM に含まれる PDF 形式の他の資料

第 2 章

EVM ハードウェアのセットアップ

本章では、EVM ハードウェアのセットアップ方法について説明します。

項目 **ページ**

2.1 ハードウェアのセットアップ方法

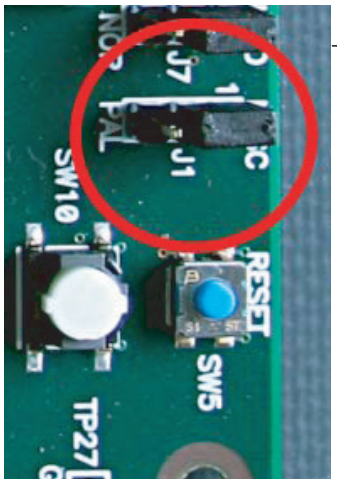
DVEVM に付属しているハードウェアをセットアップするには、これから説明する手順に従ってください。特定のペリフェラルにアクセスする必要がなければ、説明を読まずに次のステップへ進むことができます。たとえば、シリアルケーブルを使う必要がない場合には、次のステップへ進んでください。

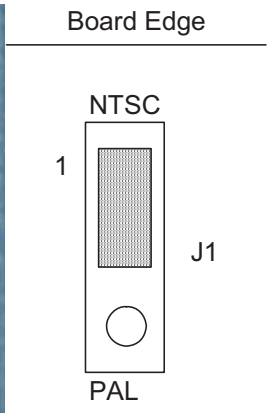
1) EVM ボードは静電放電（ESD）の影響を受けやすいので、ボードが損傷しないように、静電気防止用ストラップまたは他のデバイスを使用してください。

   機器の電源を入れる前には必ず、通信ケーブルを接続してください。

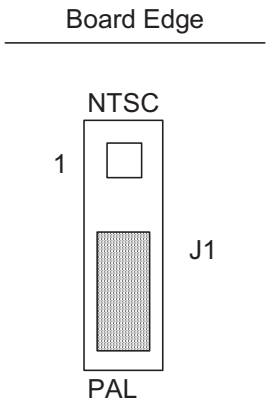
2) J1 ジャンパが、図のように正しいビデオ・フォーマット（NTSC または PAL）にセットされていることを確認してください。

注： U-boot では、Boot up 時にこのジャンパの設定を確認し、*videostd* 環境変数を決めています。この J1 ジャンパを切り替えることで、ビデオ出力設定を簡単に NTSC および PAL のどちらかへ変更することができます。



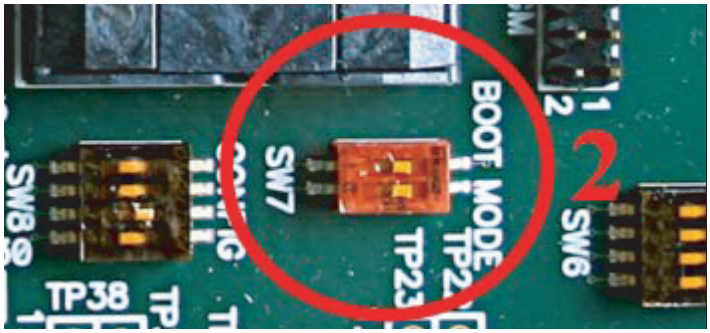


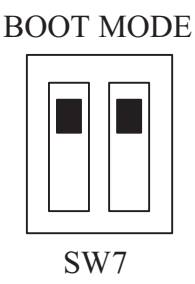
NTSC Format



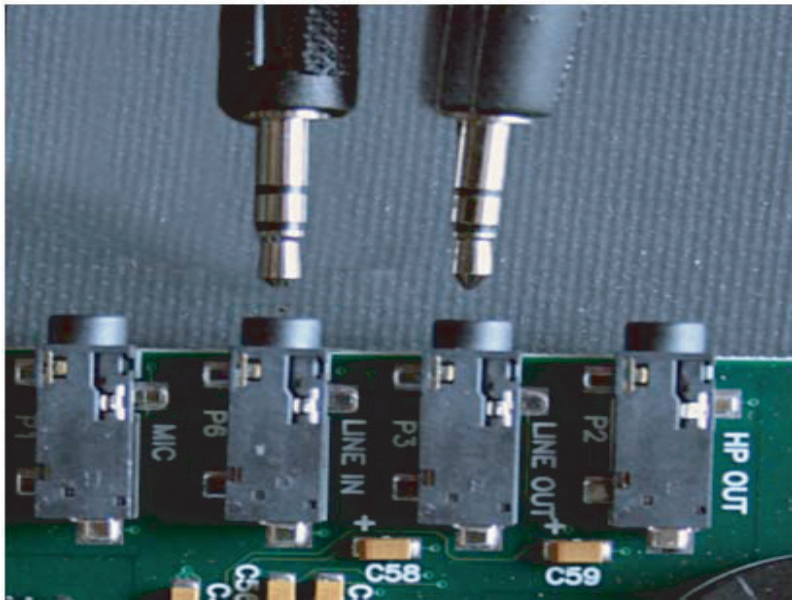
PAL Format

3) NAND フラッシュからブートを行わせるモードにするため、EVM 上の SW7 スイッチが図のようにセットされていることを確認してください（黒い正方形はスイッチの位置を示しています）。

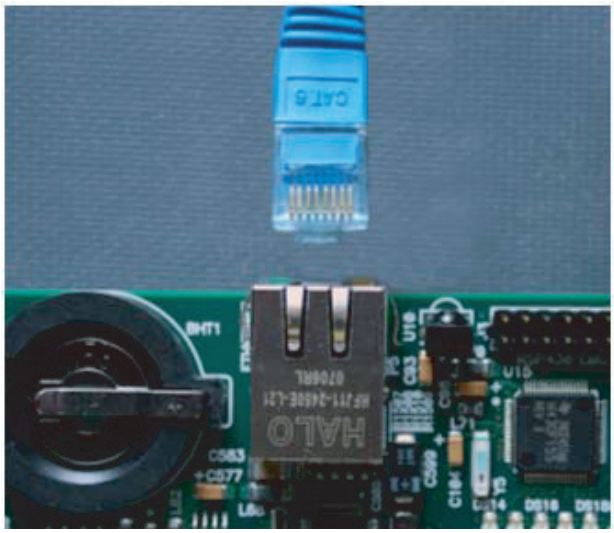




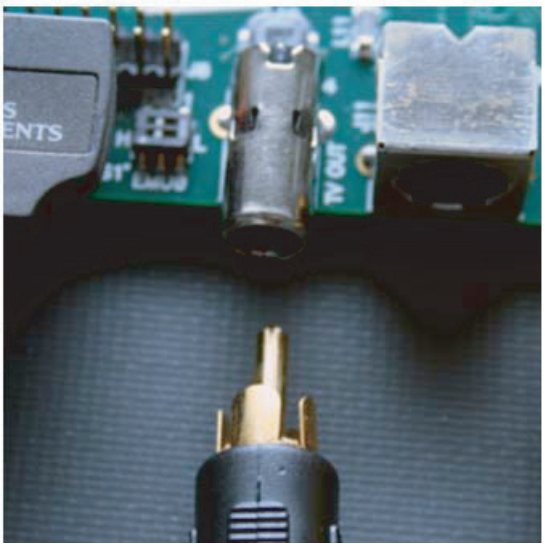
4) LINE OUT（P3）にオーディオ・スピーカーを、LINE IN（P6）にオーディオ・ソースを接続します。



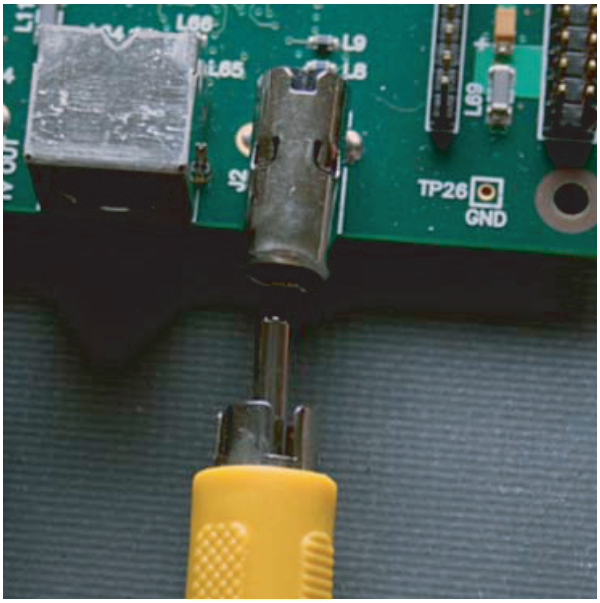
5) イーサネット接続を使用する場合には、イーサネット・ケーブルを EVM ボードのイーサネット・ポートおよびイーサネット・ネットワーク・ポートに接続します。ネットワーク接続を有効にするには、U-Boot の *bootarg* に“ ip=dhcp ”を含んでいなければならないことに注意してください。



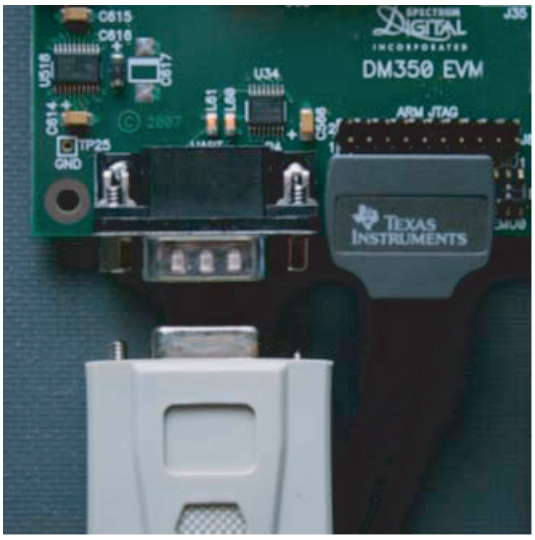
6) 映像出力を表示するには、ビデオ ディスプレイをコンポジット・ビデオ・コネクタ（J4）に接続します（キットにはビデオ ディスプレイは含まれていません）。



7) ビデオソース（たとえば、カメラや DVD プレーヤー）などの映像入力を、コンポジット・ビデオ・コネクタ（J2）に接続します。

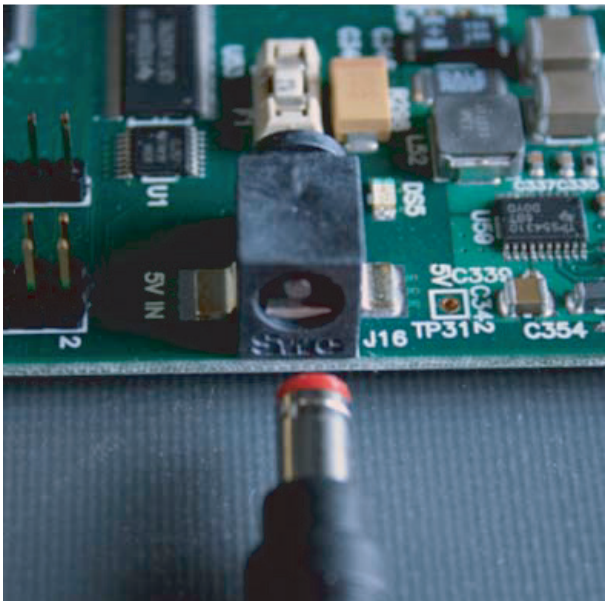


8) コンソール・ウィンドウ用に UART ポートを使用する場合には、RS-232 ヌルモデム・ケーブルを EVM の UART ポート（P4）およびホストのワークステーションの COM ポートに接続します。
コンソール・ウィンドウの使用方法の詳細については、2.2 節 「コンソール・ウィンドウに接続する方法」を参照してください。



9) ビデオの入出力のために使用する装置の電源を入れます。

10) EVM ボードの電源に、電源ケーブルを差し込みます。ESD 保護のため、ボードに電源コードを挿入してから、もう片方のプラグを電源投入してください。



11) ビデオ ディスプレイには、デモ・ソフトウェアの初期画面が表示されているはずです。赤外線リモート・コントロールを使い、ソフトウェアを実行します（第 3 章を参照）。

2.2 コンソール・ウィンドウに接続する方法

EVM のブート・メッセージの表示と割り込みを可能にするコンソール・ウィンドウをオープンすることができます。その手順は次のとおりです。

1) EVM のシリアルポートと PC のシリアル（例えば COM1）ポートをシリアルケーブルで接続します。

2) ワークステーション上でターミナル・アプリケーションを起動し、（例えば、Linux なら Minicom、Windows なら HyperTerminal）次のような構成でシリアルポートに接続します。
   - 転送速度：115200 bps
   - データ：8 bit
   - パリティ：なし
   - ストップ・ビット：1
   - フロー・コントロール：なし

3) EVM の電源を入れると、ブートシーケンス・メッセージが表示されます。ブートシーケンスを中断するには、何かキーを押して、U-Boot コマンドシェルでコマンドを入力します（「Hit any key to stop auto boot」のメッセージが表示されているときのみ有効）。本書では、U-Boot コマンドシェルで入力するコマンドは、`EVM #` プロンプトで示します。

第 3 章

デモンストレーション・ソフトウェアの実行方法

本章では、DVEVM に付属しているデモンストレーション・ソフトウェアの実行方法について説明します。

項目 **ページ**

3.1 デフォルトのブート構成

DVEVM を箱から取り出してセットアップし、ボードの電源を入れると、フラッシュから EVM ブートが起こり、しばらくしてデモが自動的に開始します。スタンドアロンでデモを実行するために、NFS マウントや TFTP サーバは必要ありません。

注： デフォルトでは、U-Boot の bootargs の定義で“ IP=off ”をセットしており、これはイーサネットコネクションを無効にします。

すぐに使用できるブートのパラメータ構成は A.4.1 項 を参照してください。他のブート方法については、以下に記載されています。

- NAND フラッシュのファイル・システムを使用した、TFTP 経由のブート（A.4.2 項）
- NFS ファイル・システムを使用した、フラッシュからのブート（A.4.3 項）
- NFS ファイル・システムを使用した、TFTP 経由のブート（A.4.4 項）
- PAL ビデオ・モードまたは NTSC ビデオ・モードのどちらかを選んでブートする方法（2.1 節）

標準ブートを中止するには、コンソール・ウィンドウで何かキーを押します（2.2 節を参照）。またブート構成を変更する場合には、A.4 節 「その他のブート方法」も参照してください。

3.2 スタンドアロン・デモの開始方法

EVM ハードウェアを接続すると、事前にインストールされているデモ・プログラムが自動的に実行されて ビデオ ディスプレイに表示されます。デモではオーディオ、ビデオ、音声のエンコードとデコードが行われます。デモを行う方法は、2 種類あります。

- **スタンドアロン。**これは電源投入時のデフォルトのモードです。デフォルトのブート構成では、ワークステーションに接続しなくても、デモは自動的に実行します。

  スタンドアロン・デモは DVSDK にセットアップされます。/example/dvevmdemo が /etc/rc.d/init.d へコピーされ、/etc/rc.d/rc3id/S88demo とシンボリカルにリンクされます。ボードが boot up して、runlevel 3 に入ると、このファイルはデモウェブサーバとデモインターフェイスを開始するために実行されます。
- **コマンドライン。**EVM をワークステーションに接続し、必要なソフトウェアをインストールする（4.3.1 項 「ターゲット Linux ソフトウェアのインストール方法」を参照）と、ボードの Linux コマンドラインからデモを実行することができます。コマンドラインからデモを実行する方法については、DVSDK のリリースノートよりリンクされているデモの資料を参照してください。

注： コマンドラインからデモを実行する場合には、スタンドアロン・モードのデモによって使用される *interface* プロセスが実行されていないことを確認してください。さもなくば、デバイスドライバがオープンできず、エラーメッセージが表示されるでしょう。

EVM ボードがブートすると、ビデオ ディスプレイにはリモート・コントロールの画像が表示されます。赤外線リモート・コントロールを使うと、デモを制御できます。

もし、表示されるリモート・コントロールの画像がこの画像と違うものであれば、そのリモート・コントロールの中で同じラベルが書かれているボタンを見つけてください。

スタンドアロン・モードでは以下の手順でデモを実行します。

1) 電池が赤外線リモート・コントロールに確実に装着されていることを確認します。
2) 初期画面に赤外線リモート・コントロールの図が表示されます。これはスタンドアロン・デモを実行するために使います。少し時間をかけて、さまざまなボタンの機能を確認してください。
3) これは汎用のリモート・コントロールなので、DVEVM デモを実行するのに必要なコードが使えるように設定する必要があるかもしれません。設定を行うには、リモート・コントロールの赤い光が点灯するまで、「Code Search」ボタンを押し続けてください。次に「DVD」ボタンを押して、コードとして「0020」と入力します。
4) TV または他のモードで、リモート・コントロールを誤って押してしまった場合には、「DVD」ボタンを押してリモート・コントロールを正しいモードに戻します。
5) リモート・コントロールが DVD+0020 コードを受け付けない場合には、完全にリセットするために電池をはずし、Power キーを最低でも 1 分間押してから電池を再び装着してください。その後、手順 3 にあるようにリモート・コントロールをプログラムします。

3.3 スタンドアロン・デモの実行方法

1) リモート・コントロールの図からメイン・メニュー画面に移動するために、リモート・コントロールの「Play」または「OK」を押します。次のような画面が表示されます。

Encode + Decode デモでは、ビデオの録画と再生を行うことができます。Encode デモでは、選択したフォーマットでオーディオ / 音声の録音およびビデオの録画を行います。Decode デモでは、選択したオーディオ / 音声ファイルおよびビデオファイルを再生します。

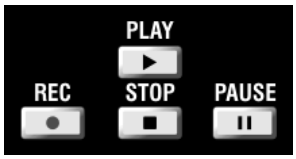
2) 上下矢印キーを使って、選択するデモを変更します。その後、「OK」または「Play」を押すと、選択したデモに切り替わります（この時点で「Power」を押すと、デモを完全に終了できます）。

3) デモの中では、設定画面から作業を開始します。デモを実行するために使用できる操作ボタンは画面の下部で確認し、現在の設定は画面の右上で確認します。

4) 上下矢印キーを使って、設定を変更したいメニューに移動します。

5) 左右の矢印キーを使って、変更したいメニュー設定が表示されるまでオプションを繰り返し移動します。

6) 「Play」を押して、Encode デモまたは Decode デモを開始します。「Rec」（録画）を 2 回押すと、Encode デモを開始します。「Stop」を押すと、メイン・メニューに戻ります。



7) デモの実行中には、設定、プロセッサの負荷、レートに関するデータが表示されます。静的な設定は右側に表示されます。レポートされる動的なデータは左側に表示されます。

8) この情報はビデオ画面に重なって表示されます。そのため、表示されるビデオは実際のビデオより暗くなります。ビデオがもっとはっきりと見えるようにするために情報表示を非表示にするには、赤外線リモート・コントロールの「Info/Select」ボタンを押します。デモの実行中に、リモート・コントロールの左右の矢印キーを使うと、OSD（オーバーレイ）の透明性を変更できます。

9) デモを終了するか、または一時停止する場合は、「Stop」または「Pause」を押します。設定画面から「Stop」を押すと、メイン・メニューに戻ります。

デモはコーデック・エンジンを使って、アプリケーションがアルゴリズムを実行できるようにしています。

3.3.1 Encode + Decode デモについて

Encode + Decode デモを実行すると、ビデオの録画と再生を行うことができます。ソースから入力されたデータは、エンコードされ、次にデコードされて、ビデオ ディスプレイに送られます。エンコードしたデータは、ファイルにはなりません。

Encode + Decode はビデオ処理だけを行っています。オーディオや音声のエンコードやデコードは行っていません。サポートされているビデオのアルゴリズムは、MPEG4（ファイル拡張子 .mpeg4）です。

表 3-1. Encode + Decode デモを行うための赤外線リモート・コントロールのボタン

赤外線リモート・コントロールのボタン	モード	動作
上下	--	-- 何もしない --
左右	セットアップ	解像度（ZOOM、CIF、D1）を変更します。
Play または OK	セットアップ	デモを開始します。
Record	--	-- 何もしない --
Info/Select	セットアップ	デモ用のブロック図の表示 / 非表示を切り換えます。
Info/Select	実行	情報表示の表示 / 非表示を切り換えます。
左右	実行	情報の透過レベルを変更します。
Pause	実行	デモを一時停止します（Play を押すと再開します）。
Stop	セットアップ / 実行	前の画面に戻ります。

ビデオ信号は、コーデック・エンジンによってビデオ・エンコーダおよびデコーダに渡されます。

コマンドラインからこのデモを実行させるには、3.4 節 「コマンドラインからのデモの実行方法」を参照してください。

3.3.2 Encode デモについて

Encode + Decode デモと同様に、Encode デモもビデオをエンコードします。さらに、オーディオまたは音声もエンコードします。オーディオまたは音声のソースはマイクロフォンです。

エンコードされたデータは、EVM の NAND フラッシュのファイルシステムに書き込まれます。書き込まれるファイル名は demo.mpeg4、demompeg4.g711 です。これらのファイルが存在する場合、必要に応じて上書きされます。

設定情報および負荷や各種レートに関して収集された動的なデータを表示する目的以外には出力はデコードされずに、ビデオ ディスプレイまたはスピーカーに送られません。

オーディオ・エンコーダではなく、音声エンコーダだけを使うことができることに注目してください。サポートされているビデオ・アルゴリズムは、MPEG4（ファイル拡張子 .mpeg4）です。サポートされている音声アルゴリズムは、G.711（ファイル拡張子 .g711）です。

表 3-2. Encode デモを行うための赤外線リモート・コントロールのボタン

赤外線リモート・コントロールのボタン	モード	動作
上下	セットアップ	オプション選択を変更します。
左右	セットアップ	選択したオプションの設定を変更します。
Play	セットアップ	Decode デモのセットアップに切り換えます。
Record （2 回）または OK	セットアップ	Encode デモを開始します。
Info/Select	セットアップ	デモ用のブロック図の表示 / 非表示を切り換えます。
Info/Select	実行	情報表示の表示 / 非表示を切り換えます。
左右	実行	情報の透過レベルを変更します（情報が前面に表示されているため、Encode デモが表示されません）。
Pause	実行	デモを一時停止します（Play を押すと再開します）。
Stop	セットアップ / 実行	前の画面に戻ります。

アプリケーションは Linux を使用した ARM 上で動作しています。ビデオおよびオーディオ信号は、コーデック・エンジンによってエンコーダに渡されます。

コマンドラインからこのデモを実行させるには、3.4 節 「コマンドラインからのデモの実行方法」を参照してください。

3.3.3 Decode デモについて

Decode デモでは、選択したオーディオ / 音声ファイルおよびビデオファイルを再生します。デコードするビデオ・ファイルとオーディオ / 音声ファイルを選択します。左右の矢印ボタンを使って、EVM 上の NAND 型フラッシュ・メモリに格納されている、デモに使うファイルと Encode デモで作成したファイルを選択します。デコードされた信号は、ビデオディスプレイおよびスピーカーに送られます。

サポートされているビデオ・アルゴリズムは、MPEG4（ファイル拡張子 .mpeg4）、また、サポートされている音声アルゴリズムは、G.711（ファイル拡張子 .g711）です。

表 3-3. Decode デモを行うための赤外線リモート・コントロールのボタン

赤外線リモート・コントロールのボタン	モード	動作
上下	--	-- 何もしない --
左右	セットアップ	別のファイルの組み合わせを選択します。
Play または OK	セットアップ	Decode デモを開始します。
Record	--	-- 何もしない --
Info/Select	セットアップ	デモ用のブロック図の表示 / 非表示を切り換えます。
Info/Select	実行	情報表示を切り換えます。
左右	実行	情報の透過レベルを変更します。
Pause	実行	デモを一時停止します（Play を押すと再開します）。
Stop	セットアップ / 実行	前の画面に戻ります。

アプリケーションは Linux を使用した ARM 上で動作しています。ビデオおよびオーディオ信号は、コーデック・エンジンによってデコーダに渡されます。

コマンドラインからこのデモを実行させるには、3.4 節 「コマンドラインからのデモの実行方法」を参照してください。

3.4 コマンドラインからのデモの実行方法

EVM ボードのシリアルポートに接続している Terminal Window 上で、Linux シェルからデモアプリケーションを実行することができます。これは 3.2 節 「スタンドアロン・デモの開始方法」で述べたものと同じ内容のデモです。

コマンドラインからデモアプリケーションを実行する前に、CMEM とアクセラレータカーネルモジュールをロードされている必要があります。“ lsmod ”コマンドを使用して、これらがロードされているかを確認してください。

されていない場合には、次の命令を使用して、これらのモジュールをロードします。

```
Target $ cd /opt/dvsdk/dm355
Target $ ./loadmodules.sh
```

次の命令を使うと、デモのコマンドラインオプションが参照できます。

```
Target $ ./encodedecode -h
Target $ ./encode -h
Target $ ./decode -h
```

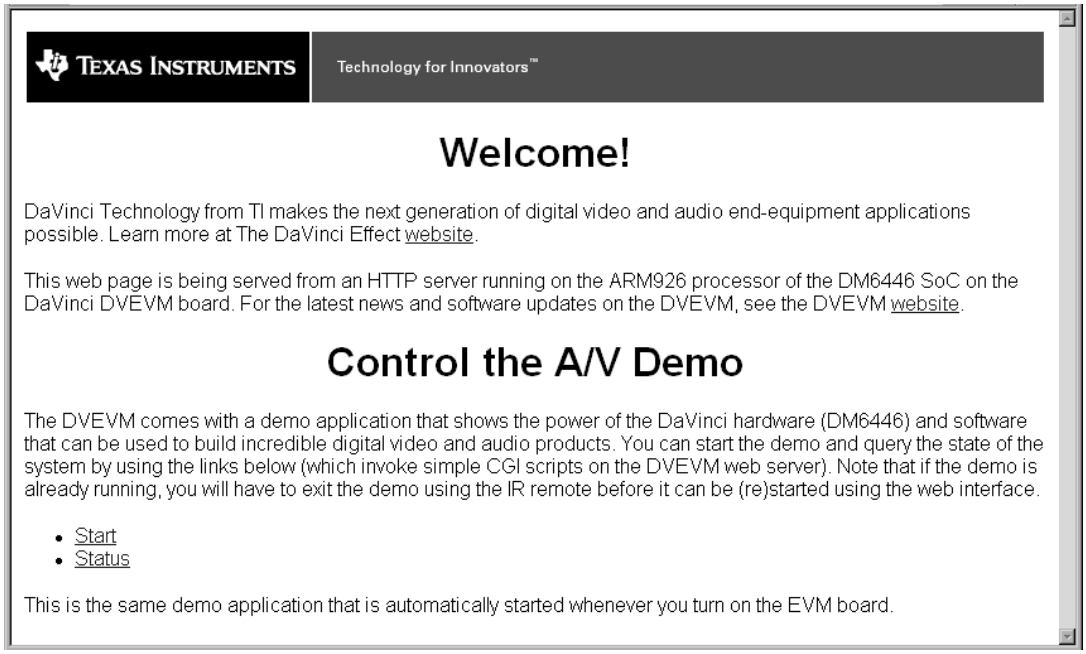
これら、コマンドラインオプションのリストは、ホスト上の DVDSK パッケージのそれぞれのデモディレクトリ上で見つけることもできます。（ファイル名：encode.txt, decode.txt, encodedecode.txt）

3.5 ネットワーク・デモの実行方法

標準 TCP/IP ネットワーク・サポートの例として、DVEVM には、小規模の HTTP Web サーバの例が含まれています。この Web サーバは、Linux のスタートアップ・シーケンスの一部として起動されます。標準 TCP/IP ポート 80 で Web ブラウザからのリクエストに応えるように設定されています。

EVM ボードのブート後に、PC を EVM ボードの接続されている同一ネットワークに接続します。「http://ip-address-of-evm」という形式の URL を Web ブラウザ（Internet Explorer、Firefox、Opera）に入力します。A/V デモのメイン・メニューの右下隅に、ボードの IP アドレスが表示されます。

DaVinci テクノロジーおよび DVEVM ソフトウェアに関する情報を含む Web ページが表示されます。



この Web ページを使用して、ボードと対話形式で A/V デモを実行します（3.3 節「スタンドアロン・デモの実行方法」を参照）。EVM の 2 つのわかりやすい CGI スクリプトを使うと、（デモがまだ実行されていないという前提で）デモを開始

して、ボード上でどのようなプロセスが実行されているか確認することができます。デモが Web ページから起動されることを確認する場合は、最初にデモを終了させてください（メイン・メニューから Power ボタンを使用します）。

Web サーバ・ソフトウェアは、THTTPD（http://www.acme.com/software/thttpd/）というオープンソース・パッケージです。THTTPD は、サイズが小さく、高速で、移植性が高くなるように設計されています。このソース・コードは、DVEVM ソフトウェアに付属しています。Web サイトから最新版を直接ダウンロードすることができます。Web サーバと CGI スクリプトは、ターゲットの /opt/dvsdk/dm355/web ディレクトリにインストールされています。

第 4 章

DVEVM ソフトウェアのセットアップ

本章では、DVEVM に付属しているソフトウェアの使用方法について説明します。

項目 **ページ**

4.1 ソフトウェアの概要

アプリケーションの開発を始めるには、DVEVM 開発環境をインストールする必要があります。ここでは、DVEVM ソフトウェアを開発用のホストにロードする際に必要な手順の概要を説明します。まず始めに、付属の CD またはその中のファイルが必要になります。

DaVinci ソフトウェア・アプローチは、相互運用性があり、最適化されたすぐに使えるビデオ・コーデックとオーディオ・コーデックを提供します。これは内蔵アクセラレータを利用します。これらのコーデックは、構成可能なフレームワークに組み込まれていて、ソフトウェアが迅速に開発できるように一般的なオペレーティング・システム（Linux など）用の公開 API として提供されます。

DVEVM には以下のソフトウェアが付属しています。

- **スタンドアロン・デモンストレーション・ソフトウェア**
  このソフトウェアは、EVM 上の NAND 型フラッシュ・メモリ上にあります。このデモではオーディオ、ビデオ、音声のエンコードとデコードを行います。別のデモでは、ボードのネットワーク機能を示します。3.2 節 「スタンドアロン・デモの開始方法」、3.5 節 「ネットワーク・デモの実行方法」を参照してください。
- **CD 1 of 4: MontaVista Linux Pro v4.0.1 システム・ツール**
  DVEVM キットに付属のバージョンは、デモ版で、次のファイルが含まれます。
  - mvl_4_0_1_demo_sys_setuplinux.bin
    このインストール・ファイルには、MontaVista ツールの開発ツール・チェーンが含まれます。
- **CD 2 of 4: MontaVista Linux Pro v4.0.1 ターゲット・ファイル・システム**
  DVEVM キットに付属のバージョンは、デモ版で、次のファイルが含まれます。
  - mvl_4_0_1_demo_target_setuplinux.bin
    このインストール・ファイルには、MontaVista のターゲット・ファイル・システムが含まれます。
- **CD 3 of 4: TI DVSDK ソフトウェア**
  この CD には、デモ・アプリケーション、コーデック・エンジン・ソフトウェア、コーデック・サーバ例、および DVEVM のドキュメント類が収められ、次のファイルが含まれます。
  - spruf73#.pdf（本書の英語版資料、# はバージョンを示しています）
  - dvsdk_setuplinux_#_#_#_#.bin
  - mvl_4_0_1_demo_lsp_setuplinux_#_#_#_#.bin
  - xdc_setuplinux_#_#_#_#.bin
  - dm355_flash_image_#_#_#_#.tar（NAND フラッシュのリカバリ用のファイルが含まれます。）
- **CD 4 of 4: A/V データ**
  A/V サンプル・データが data.tar.gz 内に含まれています。
- **EVMDM355 BSL and Target Content CD**
  Spectrum Digital 社から提供されるボード・サポート・ソフトウェアです。

MontaVista Software Inc. の同意の下、Texas Instruments は Linux Professional Edition v4.0.1 組み込み向けオペレーティング・システムのデモ版と開発ツールを提供しています。DVEVM には、このデモ版が含まれています。このデモ版は、MontaVista 社が提供する完全版の Professional Edition のサブセットです。DevRocket™ などのツールや Professional Edition の資料は含まれていませんが、その他の部分ではお客様が DaVinci プラットフォームを評価するには十分な機能があり、利用価値があります。また、このリリースには MontaVista 社のユーザ・ライセンスは含まれていないため、MontaVista Software Inc. からの直接的なお客様サポート、保証、または損害賠償は提供されません。

MontaVista Linux の製品リリースが含まれている DVSPB（Digital Video Software Production Bundle）を注文することができます。DVSPB には、完全な MontaVista ライセンスと DevRocket IDE が含まれます。

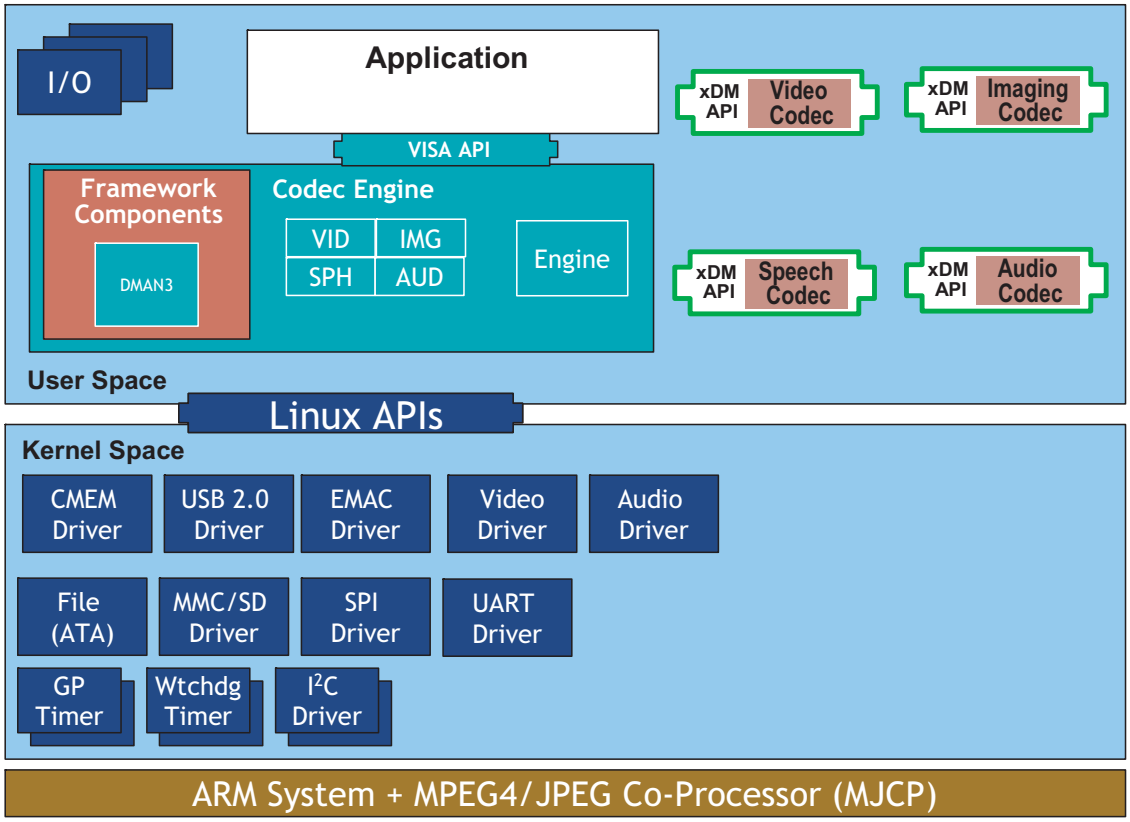
4.1.1 本書のコマンド・プロンプト

本書では、コマンドは入力される環境を示すプロンプトの後に示されます。

- **host $**
  ホスト Linux ワークステーションのシェル・ウィンドウにコマンドを入力することを表します。
- **EVM #**
  EVM ボードのシリアル・ポートに接続されたコンソール・ウィンドウ内の U-Boot シェルにコマンドを入力することを表します。(2.2 節)
- **target $**
  EVM ボードのシリアル・ポートに接続された端末ウィンドウ内の Linux シェルにコマンドを入力することを表します。

4.1.2 ソフトウェア・コンポーネント

次の図は、DVEVM キット上でのアプリケーション開発に使用するソフトウェア・コンポーネントを示します。



上記の図では、すべて ARM 上で実行されています。アプリケーションは、入出力（I/O）とアプリケーション処理を扱います。ビデオ、画像、音声、オーディオ信号を処理するために、コーデック・エンジンから提供される VISA API を使用します。コーデック・エンジンは、xDM ベースのコーデックを使用します。(**later**Codec Engine can also process signals on the ARM side if you configure the engine to do so.) 詳細は、『Codec Engine Application Developer's Guide』（資料番号 SPRUE67）を参照してください。

また、ARM 上の Linux アプリケーションでは、ドライバやタイマー等多くの API が使用可能です。

4.2 インストールの準備

ホスト・システム上で、3 枚の DVEVM デモンストレーション CD をマウントし、次の .bin ファイルを 1.2 GB 以上の空き容量がある一時的な場所にコピーします。インストール・ファイルはソフトウェアをインストールした後に削除できるため、/tmp などのディレクトリを推奨します。

- ❑ mvl_4_0_1_demo_sys_setuplinux.bin
- ❑ mvl_4_0_1_demo_target_setuplinux.bin
- ❑ mvl_4_0_1_demo_lsp_setuplinux_#_#_#_#.bin
- ❑ dvsdk_setuplinux_#_#_#_#.bin
- ❑ xdc_setuplinux_#_#_#_#.bin

これらのインストーラーは、1.3 節でご紹介した、「TI DaVinci ソフトウエア・アップデートサイト」にて最新版を入手できます。

X ウィンドウが使用可能であることを確認し、DISPLAY 環境変数を次の値に指定します。たとえば：

csh:

```
host $ setenv DISPLAY cnabc0314159d1:0
```

ksh または bash:

```
host $ export DISPLAY=cnabc0314159d1:0
```

4.3 ソフトウェアのインストール方法

DVEVM で使用されるソフトウェアを以下の手順でインストールします。

- ❑ 4.3.1 項「ターゲット Linux ソフトウェアのインストール方法」
- ❑ 4.3.2 項「DVSDK ソフトウェアのインストール方法」
- ❑ 4.3.3 項「A/V デモ・ファイルのインストール方法」
- ❑ 4.3.4 項「ターゲットからのアクセス用に共有ファイル・システムをエクスポートする方法」
- ❑ 4.3.5 項「共有ファイル・システムのテスト」

4.3.1 ターゲット Linux ソフトウェアのインストール方法

ここでは、ターゲット・ボードで使用する Linux をインストールする方法について説明します。これは MontaVista Linux Pro v4.0.1 のデモ版です。

ターゲットとユーザのホストの Linux ワークステーションでは、別々のバージョンの Linux が使用されることに注意してください。次の Linux ホスト・オペレーティング・システムを DVEVM と一緒に使用することができます。

- ❑ Red Hat Enterprise Linux v3（サーバ・エディション）
- ❑ Red Hat Enterprise Linux v4（サーバ・エディション）

Linux ソフトウェアをインストールする手順は以下のとおりです。

1) **root** でホスト Linux ワークステーションにログインしてください。これにより、MontaVista Linux をインストールするグラフィカル・インストーラを正しく使用することができます。

2) Linux ツール、Linux カーネル、およびファイル・システムをインストールするためにコピーした一時保存のディレクトリから、次の bin ファイルをそれぞれ実行してください（ここで **#_#_#_#** は現行のバージョン番号です）。もし bin ファイルの実行ができなかったら、ファイルの実行権限を確認してください（コマンド：`chmod +x *.bin`）。

```
host $ ./mvl_4_0_1_demo_sys_setuplinux.bin
host $ ./mvl_4_0_1_demo_target_setuplinux.bin
host $ ./mvl_4_0_1_demo_lsp_setuplinux_#_#_#_#.bin
```

デフォルトのインストール・ディレクトリから変更し、/opt/mv_pro_4.0.1 ディレクトリにインストールすることをお勧めします。

3) これらのファイルを実行し終えたら、次のファイルがインストールしたディレクトリ /opt/mv_pro_4.0.1 等にあることを確認してください。
   - ■ mvltools4.0.1-no-target.tar.gz
   - ■ mvl4.0.1-target_path.tar.gz
   - ■ DaVinciLSP-#_#_#_#.tar.gz

4) 上記 tar ファイルを解凍するディレクトリに移動します。たとえば：

```
host $ cd /opt/mv_pro_4.0.1
```

5) tar ファイルを（root で）次のコマンドを使って解凍します。

```
host $ tar zxf mvltools4.0.1-no-target.tar.gz
host $ tar zxf mvl4.0.1-target_path.tar.gz
host $ tar zxf DaVinciLSP-#_#_#_#.tar.gz
```

これで MontaVista のディレクトリ構造が /opt/mv_pro_4.0.1/montavista/ ディレクトリ以下に作成されます。

注： DVSDK に付属して出荷される LSP はマルチプラットフォーム用の LSP であり、個々のプラットフォーム向けに設定されていません。出荷時に付属する LSP のデモやサンプルのビルドを確認することはできません。したがって、はじめに LSP をユーザエリアにコピーし、DM355 向けに設定 / ビルドする必要があります。詳細は、4.5 節を参照してください。

4.3.2 DVSDK ソフトウェアのインストール方法

DVSDK ソフトウェアには、コーデック・エンジン・コンポーネント、サンプル・データ・ファイル、xDAIS と xDM のヘッダ・ファイル、および Linux 用連続メモリ・アロケータ（CMEM）が含まれています。

Linux インストーラを使用して DVSDK ソフトウェアを次の手順でインストールします。

1) **ユーザ・アカウント**を使用してログインします。

2) DVSDK CD から前もってコピーした、DVSDK のインストーラを起動します。たとえば次のとおりです。

```
host $ cd /tmp
host $ ./dvsdk_setuplinux_#_#_#_#.bin
```

DVSDK は /home/<*useracct*>/dvsdk_#_# にインストールされます。

3) 前の手順で CD からコピーした XDC のインストーラを実行します。たとえば：

```
host $ cd /tmp
host $ ./xdc_setuplinux_#_#_#_#.bin
```

インストール先を選択する画面が表示されたら、デフォルトのディレクトリを使用せず、手順 2) で作成したディレクトリを使用してください。たとえば、/home/<*useracct*>/dvsdk_#_# です。

4) これで一時的な場所に置いた .bin ファイルを削除することができます。

注： ディレクトリ上で「 `rm -rf`」コマンドを使用すると、これらのコンポーネントのいずれかをアンインストールすることができます。インストーラによって作成された uninstall ファイルは無視してください。

4.3.3 A/V デモ・ファイルのインストール方法

4 枚目の CD には、デモで使用する A/V ファイルが収められています。前の節の手順を実行したら、以下の手順を実行して A/V ファイルをインストールします。

1) 上記でセットアップした DVSDK ディレクトリ上のデモディレクトリに移動します。たとえば次のとおりです。

```
host $ cd /home/<useracct>/dvsdk_#_#/demos
```

2) A/V データ CD をマウントし、ファイルをユーザの DVSDK ディレクトリにコピーします。たとえば次のとおりです。

```
host $ cp /mnt/cdrom/data.tar.gz .
```

3) A/V データ・ファイルを解凍します。たとえば次のとおりです。

```
host $ tar zxf data.tar.gz
```

4.3.4 ターゲットからのアクセス用に共有ファイル・システムをエクスポートする方法

ボードの NAND フラッシュにはファイル・システムが組み込まれていますが、開発ではターゲット・ボードの NFS をホスト Linux ワークステーションのファイル・システムにマウントする方が便利です。アプリケーションをテストしたら、それをスタンドアロン・デモンストレーション用にボードのフラッシュに保管できます。

ボードでターゲット・ファイル・システムをマウントするには、事前にホスト Linux ワークステーションでそのターゲット・ファイル・システムをエクスポートしておく必要があります。このファイル・システムは、NFS（Network File System）サーバを使用します。エクスポートされたファイル・システムには、ターゲット・ファイル・システムとユーザの実行可能ファイルが含まれます。

次の手順で NFS サーバからファイル・システムをエクスポートしてください。この手順は一度だけ実行する必要があります。

1) **ユーザ**・アカウントでホスト Linux ワークステーションにログインします。以降の手順では、ユーザのホーム・ディレクトリを「~」と示します。

2) 次のコマンドを実行して、MontaVista ファイル・システムの場所を用意します。例えば：

```
host $ cd /home/<useracct>
host $ mkdir -p workdir/filesys
host $ cd workdir/filesys
```

3) ホスト Linux ワークステーションでユーザから「**root**」に切り替えます。

```
host $ su root
```

4) 次のコマンドを実行して、*<useracct>* で共有領域に書き込むための権限をもつターゲット・ファイル・システムのコピーを作成します。*<useracct>* をユーザ名に置き換えてください。/opt/mv_pro_4.0.1 以外の場所にインストールした場合は、cp コマンドでユーザの場所を使用します。

```
host $ cp -a /opt/mv_pro_4.0.1/montavista/pro/devkit/arm/v5t_le/target/* .
host $ chown -R <useracct> opt
```

5) ホスト Linux ワークステーションで /etc/exports ファイルを編集します。filesys 領域のエクスポート用に次の行を追加します。*<useracct>* をユーザ名に置き換えてください。ルート・ディレクトリからの絶対パスを使用してください。~ はファイル・システムによってはエクスポートに対して機能しない場合があります。

```
/home/<useracct>/workdir/filesys *(rw,no_root_squash,no_all_squash,sync)
```

> **注：** 上記のコマンド中、* と (の間にはスペースを入れないように注意してください。

6) root として次のコマンドを使用して、NFS サーバに構成の変更を認識させ、次に NFS を再起動させます。

```
host $ /usr/sbin/exportfs -av
host $ /sbin/service nfs restart
```

> **注：** 「**exportfs -rav**」コマンドを使用すれば、全てのディレクトリを再エクスポートできます。/etc/init.d/nfsを使用すれば、NFSステータスが実行されていることを確かめることができます。

7) サーバーのファイヤーウォールがオフになっていることを確認します。

```
host $ /etc/init.d/iptables status
```

もし、ファイヤーウォールが実行されている場合には、次のコマンドで、ディスエーブルします。

```
host $ /etc/init.d/iptables stop
```

4.3.5 共有ファイル・システムのテスト

NFS セットアップをテストするには、次の手順を踏んでください。

1) ホスト Linux ワークステーションの IP アドレスを次のようにして取得します。eth0 イーサネット・ポートに関連付けられている IP アドレスを検索します。

```
host $ /sbin/ifconfig
```

2) 端末エミュレーション・ウィンドウを開き、2.2 節に従って、RS-232 経由で EVM ボードに接続します。Windows ワークステーションの場合は、HyperTerminal を使用できます。Linux ワークステーションの場合は、Minicom を使用できます。

3) EVM ボードの電源を入れ、コンソール・ウィンドウでキーを押して自動ブートシーケンスを終了させます（2.2 節）。

4) コンソール・ウィンドウで、次の環境変数をセットします。

```
EVM # setenv nfshost <ip address of your nfs host>
EVM # setenv rootpath <directory to mount>
EVM # setenv bootargs console=ttyS0,115200n8 noinitrd rw
 ip=dhcp root=/dev/nfs
 nfsroot=$(nfshost):$(rootpath),nolock mem=116M
 video=davincifb:vid0=720x576x16,2500K:vid1=720x576x16,
```

```
2500K:osd0=720x576x16,2025K
davinci_enc_mngr.ch0_output=COMPOSITE
davinci_enc_mngr.ch0_mode=$(videostd)
```

「**setenv bootargs**」命令は 1 行で入力されていることに注意してください。また、入力時にテンキーパッドを使用すると余分な文字が入ってしまうことがあるため、テンキーパッドからの数字入力を避ける必要があります。

<directory to mount> は、4.3.4 項の 5) で指定したディレクトリと同じものでなければなりません。たとえば：

/home/*<useracct>*/workdir/filesys

ヒント： printenv コマンドを使用することで、現在の環境設定を表示できます。また、次回設定を行う時のために、これらの setenv コマンドを .txt ファイルにまとめて保存しておくと便利です。

---

5) この環境を保存して、EVM ボードの電源を入れるたびにこれらのコマンドを再入力する必要がないようにします。

```
EVM # saveenv
```

6) NFS を使用してボードをブートします。

```
EVM # boot
```

7) パスワードを要求されない " root " としてログインすることができます。

TFTP または NFS、およびボード上の NAND フラッシュによるブート方法については、A.4 節「その他のブート方法」を参照してください。

4.3.6 評価版 / 製品版コーデックの使用上の注意

DM355 DVSDK には、次のコーデックが含まれます。

- ❑ シーケンシャル JPEG デコーダ
- ❑ シーケンシャル JPEG エンコーダ
- ❑ MPEG4 シンプル・プロファイル・デコーダ
- ❑ MPEG4 シンプル・プロファイル・エンコーダ

これらのコーデックは、" デモ用 " として提供されます。これらのコーデックを製品開発環境で用いるには、DVEVM アップデートサイト（http://www.ti.com/dvevmupdates）にて適切なライセンス・アグリーメントの元で最新バージョンをダウンロードしてください。

4.4 ビルド / 開発環境のセットアップ方法

次の手順で開発とビルドの環境をセットアップしてください。

1) NFS ホスト・システムの**ユーザ**・アカウントにログインします（root ではありません）。

2) MontaVista ツール・チェーンのホスト・ツールとクロス・コンパイラ（arm_v5t_le-gcc）を検出できるように PATH をセットします。たとえば、MontaVista LSP のデフォルトのインストールでは、次のような定義をユーザのシェル・リソース・ファイル（~/.bashrc など）に追加します。

```
PATH="/opt/mv_pro_4.0.1/montavista/pro/devkit/arm/v5t_le/bin:
     /opt/mv_pro_4.0.1/montavista/pro/bin:
      /opt/mv_pro_4.0.1/montavista/common/bin:$PATH"
```

/opt/mv_pro_4.0.1 以外の場所にインストールした場合は、その位置を PATH で指定します。

3) .bashrc ファイルを変更した後には、次のコマンドを必ず実行してください。

```
host $ source .bashrc
```

4.4.1 簡単なプログラムを作成して EVM 上で実行する方法

4.3.4 項「ターゲットからのアクセス用に共有ファイル・システムをエクスポートする方法」および 4.4 節「ビルド / 開発環境のセットアップ方法」の手順を実行したことを確認してください。

NFS ホスト・システム上で、次の手順をユーザとして実行します（root ではありません）。

1) `host $ mkdir /home/<useracct>/workdir/filesys/opt/hello`

2) `host $ cd /home/<useracct>/workdir/filesys/opt/hello`

3) 次の内容の hello.c というファイルを作成します。

```
#include <stdio.h>

int main() {
   printf("Buongiorno DaVinci!\n");
   return 0;
}
```

4) `host $ arm_v5t_le-gcc hello.c -o hello`

ターゲット・ボード上で、次の手順を実行します。ターゲットのコンソール・ウィンドウ（2.2 節）または telnet セッションのいずれかを使用できます。

1) `target $ cd /opt/hello`

2) ./hello を実行します。出力は次のようになります。

```
Buongiorno DaVinci!
```

4.5 新しい Linux カーネルのビルド方法

ターゲットの Linux カーネルのソースを変更する場合は、そのカーネルを再ビルドする必要があります。その後、DVEVM ボードのフラッシュにインストール済みのカーネルを置き換えるか、またはネットワーク接続上でカーネルをブートするように U-Boot ユーティリティで TFTP を使用するかしてこのカーネルをブートします。

新しいカーネルをビルドする場合は、事前に 4.4 節 「ビルド / 開発環境のセットアップ方法」および 4.4.1 項 「簡単なプログラムを作成して EVM 上で実行する方法」を完了していることを確認してください。

次の手順で Linux カーネルを再ビルドします。

1) ユーザ・アカウントにログインします（root ではありません）。

2) Rules.make ファイルを使い、**PLATFORM** 変数をセットします。この方法については、4.6 節に記述されています。

3) 次のようなコマンドを使用して、ホーム・ディレクトリに MontaVista Linux Support Package（LSP）のローカル作業コピーを作成します。このコピーには、組み込み Linux 2.6.10 カーネルと DaVinci ドライバが含まれます。/opt/mv_pro_4.0.1 以外の場所にインストールした場合は、cp コマンドでユーザの場所を使用します。

```
host $ cd /home/<useracct>
host $ mkdir -p workdir/lsp
host $ cd workdir/lsp
host $ cp -R /opt/mv_pro_4.0.1/montavista/pro/devkit/lsp/ti-davinci .
```

4) 次のコマンドを使用して、DaVinci のデフォルトを使用するカーネルを構成します。CROSS_COMPILE はコンパイル時に使用される実行ファイルに対してプレフィックスを指定することに注意してください。

```
host $ cd ti-davinci
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- davinci_dm355_evm_defconfig
```

5) カーネル・オプションを変更するには、「make menuconfig」か「make xconfig」コマンドを使用する必要があります。

MontaVista のデフォルトのカーネル・オプションをイネーブルするために、次のコマンドを使用します。

```
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- checksetconfig
```

6) 次のコマンドを使用して、カーネルをコンパイルします。

```
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- uImage
```

7) カーネルがロード可能なモジュールで構成されるのであれば、これらのモジュールをリビルドしインストールするという以下の命令を使用します。

```
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le- modules
host $ make ARCH=arm CROSS_COMPILE=arm_v5t_le-
   INSTALL_MOD_PATH=/home/<useracct>/workdir/filesys modules_install
```

8) 次のコマンドを使用して、U-Boot が TFTP で EVM にダウンロードできる場所に uImage をコピーします。これらのコマンドは、ユーザがデフォルトの TFTP ブート領域 /tftpboot を使用することを想定しています。他の TFTP ルートの場所を使用する場合は、/tftpboot をユーザ指定の TFTP ルートの場所に変更してください（これらのコマンドは root として実行するか、または `chown uImage` としてファイルのオーナーシップを変更してください）。

```
host $ cp /home/<useracct>/workdir/lsp/ti-davinci/arch/arm/boot/uImage /tftpboot
host $ chmod a+r /tftpboot/uImage
```

TFTP サーバーのセットアップについては、A.3 節をご覧ください。

Linux のビルド構成オプションの詳細については、標準 Linux カーネルの参考資料またはオンライン・ソースを参照してください。

4.6 ターゲットに合わせて DVEVM ソフトウェアを再ビルドする方法

デモ・ファイルを /opt/dvevm ディレクトリに置くには、DVEVM ソフトウェアを再ビルドする必要があります。この場合は、次の手順を実行します。

1) ディレクトリを dvsdk_#_# に変更します。

2) dvsdk_#_#/Rules.make ファイルを編集します。

   ■ 次のように、PLATFORM をセットします。

   PLATFORM=dm355

   ■ 次のように、DVEVM のインストール・ディレクトリのトップレベルを DVSDK_INSTALL_DIR にセットします。

   DVSDK_INSTALL_DIR=/home/*<useracct>*/dvsdk_#_#

   ■ 次のように、EXEC_DIR が、NFS エクスポート済みの opt ディレクトリを指定していることを確認します。

   EXEC_DIR=/home/*<useracct>*/workdir/filesys/opt/dvsdk/dm355

   ■ 次のように、MVTOOL_DIR が、MontaVista Linux ツールディレクトリを指定していることを確認します。

   MVTOOL_DIR=/opt/mv_pro_4.0.1/montavista/pro/devkit/arm/v5t_le

   ■ LINUXKERNEL_INSTALL_DIR が次のように定義されていることを確認します。

   LINUXKERNEL_INSTALL_DIR=/home/*<useracct>*/workdir/lsp/ti-davinci/

3) Rules.make が入っている同じディレクトリ内で、次のコマンドを使用して DVSDK デモ・アプリケーションをビルドし、生成されたバイナリを EXEC_DIR で指定したターゲット・ファイル・システムに置きます。

```
host $ make clean
host $ make
host $ make install
```

4.7 新しい Linux カーネルのブート方法

新しいカーネルをビルドしたら、それを使用して DaVinci ボードをブートするために TFTP 経由でボードに転送する必要があります。ここでは、4.5 節 「新しい Linux カーネルのビルド方法」の手順を完了していること、およびブート・ファイル uImage が /tftpboot（または他のサイト固有の TFTP でアクセス可能な場所）にコピーされていることを想定しています。

1) EVM ボードの電源を入れ、コンソール・ウィンドウでキーを押して自動ブートシーケンスを終了します（2.2 節）。

2) 次の環境変数をセットします（この場合、ユーザがデフォルトの何も変更されていない U-Boot 環境から起動することを想定しています。
   U-Boot のデフォルト環境については、3.1 節 「デフォルトのブート構成」を参照してください）。

```
EVM # setenv bootcmd 'dhcp;bootm'
EVM # setenv serverip <tftp server ip address>
EVM # setenv bootfile uImage
EVM # setenv bootargs mem=116M console=ttyS0,115200n8
  root=/dev/mtdblock3 rw rootfstype=yaffs2 ip=dhcp
  video=davincifb:vid0=720x576x16,2500K:vid1=720x576x16,
  2500K:osd0=720x576x16,2025K
  davinci_enc_mngr.ch0_output=COMPOSITE
  davinci_enc_mngr.ch0_mode=$(videostd)
EVM # saveenv
```

「**setenv bootargs**」命令は 1 行で入力されていることに注意してください。

3) ボードをブートします。

```
EVM # boot
```

この構成により、新しい Linux カーネルが TFTP を介して NAND フラッシュを使用したファイル・システムでブートします。NFS ファイル・システムを使用したブートについては A.4.4 項をご覧ください。

ホストワークステーションが TFTP サーバを実行していることを確かめる方法についてや、もし実行されていない場合、何をするべきなのかについては A.3 節に記載されています。

ブートについての更なる詳細については A.4 節をご覧ください。

4.8 Digital Video Test Bench (DVTB) を使う方法

Digital Video Test Bench (DVTB) は、DVDSK を使用したプラットフォームにおいて、統一したデータフローを実行できるように開発された Linux ユーティリティです。DVTB は、ビデオ、画像、オーディオ、スピーチのストリームをエンコード／デコードするために、コーデック・エンジン VISA API と Linux ドライバペリフェラル API を使用します。

DVTB を使用するれば、データフローを始める前に、コーデックやペリフェラルのコンフィグレーションを行うことができます。これにより、さまざまなケースでシステムを評価することが可能になります。

DVSDK のインストールにより、DVTB は下記のディレクトリに置かれます。
/home/<*useracct*>/dvsdk_#_#/dvtb_#_#_#
#_#_# は DVTB のバージョンです。（例えば、1_23_000）

ターゲット・ファイル・システムに DVTB をインストールするには、DVSDK がインストールされたホストマシン上で、次のステップを実行します。

1) 4.6 節でご説明した通り、Rules.make ファイル上で PLATFORM が正しく定義されていることを確認してください。

2) 次のコマンドを実行します。

```
host $ cd /home/<useracct>/dvsdk_#_#/dvtb_#_#_#
host $ make clean CONFIGPKG=dm355
host $ make CONFIGPKG=dm355
```

3) デバイスのターゲット・ファイル・システム上で、「dvtb-d」と「dvtb-r」のバイナリを /opt/dvsdk/dm355 へコピーし、そこで、実行します。DSP で実行させるためには同じディレクトリにしてください。

DVTB についての詳細は、以下のドキュメントをご参照ください。

- **リリース・ノート**

  /home/<useracct>/dvsdk_#_#/dvtb_#_#_#/docs/dvtb_release_notes.pdf

- **ユーザーズ・ガイド**

  /home/<useracct>/dvsdk_#_#/dvtb_#_#_#/docs/dvtb_user_guide.pdf

付録 A

追加手順

この付録では、ユーザのセットアップ内容や特定の目的に応じて使用できるオプションの手順について説明します。

項目 **ページ**

A.1 ビデオ信号の変更方法

U-Boot では、boot-up 時に J1 ジャンパーセッティングを読んで、videostd 環境変数を決めます。U-Boot で Bootcmd が videostd 変数を使用してビデオ出力をセットする方法では（Bootcmd のサンプルは A.4 節「その他のブート方法」をご参照ください）NTSC と PAL の切り替えは J1 ジャンパーを変えるだけで簡単に行えます。

J1 ジャンパーのセッティングに基づいた Bootargs を自動的にアップデートするためには、以下のオプションを使用してください。

```
EVM # setenv bootargs 'mem=116M console=ttyS0,115200n8
   root=/dev/mtdblock3 rw rootfstype=yaffs2 ip=dhcp
   video=davincifb:vid0=720x576x16,2500K:vid1=720x576x16,
   2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE'
EVM # setenv bootcmd 'setenv setboot setenv bootargs
   \$(bootargs) davinci_enc_mngr.ch0_mode=\$(videostd);run
   setboot;nboot 0x80700000 0 0x400000;bootm'
```

もし、bootcmd で videostd 変数を使用したくない時は、bootcmd セッティング中で以下のオプションを使用してください。下記の記載で、NTSC と PAL の違いは太字で表しています。

NTSC

```
video=davincifb:vid0=720x576x16,2500K:vid1=720x576x16,
2500K:osd0=720x576x16,2025K
davinci_enc_mngr.ch0_output=COMPOSITE
davinci_enc_mngr.ch0_mode=ntsc
```

PAL

```
video=davincifb:vid0=720x576x16,2500K:vid1=720x576x16,
2500K:osd0=720x576x16,2025K
davinci_enc_mngr.ch0_output=COMPOSITE
davinci_enc_mngr.ch0_mode=pal
```

A.2 デモ・アプリケーションを Third Party Menu に追加する方法

以降の手順を実行すると、ユーザが作成したデモを Third Party Menu に追加することができます。ユーザ・インターフェイスで一度に表示できるデモは 4 つのみです。5 つ以上のデモを追加すると、アルファベット順の最初の 4 つが表示されます。

1) デモ用に次のファイルを作成します。

- **logo.jpg**
  これはデモの説明の横に表示されるサードパーティ会社のロゴです。この画像は 50x50 のサイズの JPEG フォーマットです。
- **readme.txt**
  これはテキスト・ファイルです。ファイルの最初の 40 文字は、デモの簡単な説明です。デモ・インターフェイスには最大 40 文字が表示されますが、改行文字を検出すると停止します。たとえば、このファイルに「Video Phone demo」や「Network Audio demo」などと記述できます。
- **app.sh**
  これはユーザのデモを起動する実行可能ファイルです。このファイルはデモの実行可能ファイル自体か、またはこの実行可能ファイルを実行するシェル・スクリプトです（シェル・スクリプトの場合は、その実行可能ビットがすべてに対してセットされていることを確認してください）。スクリプトは次のようなものです。

  ```
  #!/bin/sh
  exec ./mydemoname
  ```

- **その他のファイル**
  app.sh がシェル・スクリプトの場合、ユーザのデモ実行ファイルには別の名前が付きます。また、実行ファイルで使用されるデータ・ファイルやその他のファイルを用意しなければならない場合もあります。

注： デモ・アプリケーションは、相対パスを使用して実行時に必要なファイルにアクセスする必要があります。これはデモが実行される場所とは別の場所にアーカイブが展開されるためです。

---

2) 上記のリスト内のファイルをすべてアーカイブする gzip を使って圧縮された tar ファイル（末尾が .tar.gz）を作成します。たとえば、logo.jpg、readme.txt、および app.sh のファイルがある場合は、次のコマンドを使用できます。

```
tar cvzf ti_videophone.tar.gz logo.jpg readme.txt app.sh
```

この tar ファイルには、慣例として < 会社名 >_< デモ名 >.tar.gz（ファイル名にはスペースを使用しない）と名前を付けます。たとえば、Texas Instruments が作成するテレビ電話用デモの名前は、ti_videophone.tar.gz のようになります。この名前は、すべてのデモが同一のディレクトリにインストールされるため固有でなければなりません。

必要な 3 つのファイルは、アーカイブの一番上のレベルのディレクトリ内に置く必要があります。その他のファイルはサブディレクトリに置くことができますが、デモがそのファイルへのアクセスに相対参照を使用する場合のみです。たとえば、アーカイブでは次のディレクトリ構造を使用できます。

```
|-- app.sh
|-- data
|   |-- datafile1
|   `-- datafile2
|-- logo.jpg
`-- readme.txt
```

作成したファイルのフォーマットを確認するには、Linux で次のコマンドを実行します。その結果として、「gzip compressed data」と示されます。

```
file <filename>.tar.gz
```

3) ユーザのアーカイブをターゲットのインストール・ディレクトリの「thirdpartydemos」サブディレクトリに置きます。これは DVEVM ソフトウェアがターゲット・ファイル・システムにインストールされた場所です。ターゲットのインストール・ディレクトリのデフォルトは /opt/dvevm です。したがって、デモ・アーカイブのデフォルトの場所は /opt/dvevm/thirdpartydemos です。この場所にはアーカイブの内容を展開しないでください。デモが実行されるたびに舞台裏で展開が実行されます。

A.3 TFTP サーバーの設定方法

TFTP サーバーがセットアップされているかどうかをチェックするには、以下のコマンドを使用します。

```
host $ rpm -q tftp-server
```

もし、セットアップされていない場合には次の手順で設定してください。

1) MontaVista Linux Demo Edition （4.3.1 項を参照してください）をまだインストールしていない場合には、インターネット上の多くの場所から Linux ホストのために TFTP サーバーをダウンロードすることが可能です。「tftp-server」をサーチしてください。

2) TFTP のインストール時に、次のコマンドを使用します。-#.#-# はファイル名のバージョン部分です。

   ```
   host $ rpm -ivh tftp-server-#.#-#.rpm
   ```

   次のように表示されます。

   ```
    warning: tftp-server-#.#-#.rpm:
    V3 DSA signature: NOKEY, key ID 4f2a6fd2
   Preparing... ########################################### [100%]
   1:tftp-server ########################################### [100%]
   ```

3) 次のコマンドで、TFTP がインストールされているかどうか確認してください。

   ```
   host $ /sbin/chkconfig --list | grep tftp
   ```

TFTP サーバーをオンにするには次のコマンドを使用します。

```
/sbin/chkconfig tftp on
```

TFTP ファイルがあるデフォルトのルートロケーションは /tftpboot です。

A.4 その他のブート方法

EVM のデフォルト構成では、ボードの NAND フラッシュのファイル・システムを使用してフラッシュからブートします。ボードをブートする他の方法は、次のとおりです。

- ❑ NAND フラッシュのファイル・システムによる TFTP ブート（A.4.2 項）
- ❑ NFS ファイル・システムによるフラッシュ・ブート（A.4.3 項）
- ❑ NFS ファイル・システムによる TFTP ブート（A.4.4 項）

4.3.6 項では、PAL ビデオ・モードと NTSC ビデオ・モードのブート方法について説明しています。

以降の節では、各ブート方法を有効にするための環境変数の設定方法を示します。

これらのモードのいずれかでブートする手順は、次のとおりです。

1) EVM ボードの電源を入れ、コンソール・ウィンドウでキーを押して自動ブートシーケンスを終了します（2.2 節）。

   以降の節に記述されている、使用したいブート・モードの環境変数をセットします。（「**setenv bootargs**」コマンドは 1 行で記述することに注意してください）

2) これらの設定をデフォルトとして今後も使用したい場合は、その環境を保存します。

```
EVM # saveenv
```

3) ユーザが行った設定を使用してボードをブートします。

```
EVM # boot
```

A.4.1 ボードの NAND フラッシュのファイル・システムを使用したフラッシュからのブート

これがすぐに使用できるデフォルトのブート構成です。

このモードでブートするには、自動ブートシーケンス中断後に次のパラメータをセットします。

```
EVM # setenv bootcmd 'nboot 0x80700000 0 0x400000;bootm'
EVM # setenv bootargs console=ttyS0,115200n8 ip=dhcp
   root=/dev/mtdblock3 rw rootfstype=yaffs2 mem=116M
   video=davincifb:vid0=720x576x16,2500K:vid1=720x576x16,
   2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE
   davinci_enc_mngr.ch0_mode=$(videostd)
EVM # boot
```

ブート時、次の行でブート・モードを確認します。

```
## Booting image at 80700000 ...
```

A.4.2 ボードの NAND フラッシュのファイル・システムを使用した TFTP 経由のブート

このモードでブートするには、自動ブートシーケンス中断後に次のパラメータをセットします。

```
EVM # setenv bootcmd 'dhcp;bootm'
EVM # setenv bootargs console=ttyS0,115200n8 ip=dhcp
   root=/dev/mtdblock3 rw rootfstype=yaffs2 mem=116M
   video=davincifb:vid0=720x576x16,2500K:vid1=720x576x16,
   2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE
   davinci_enc_mngr.ch0_mode=$(videostd)
EVM # setenv serverip <tftp server ip address>
EVM # setenv bootfile <kernel image>
EVM # boot
```

ブート時、次の行でブート・モードを確認します。

```
TFTP from server 192.168.160.71; our IP address is 192.168.161.186
Filename 'library/davinci/0.4.2/uImage'.
...
## Booting image at 80700000 ...
```

A.4.3 NFS ファイル・システムを使用したフラッシュからのブート

このモードでブートするには、自動ブートシーケンス中断後に次のパラメータをセットします。

```
EVM # setenv bootcmd 'nboot 0x80700000 0 0x400000;bootm'
EVM # setenv nfshost <ip addr of nfs host>
EVM # setenv rootpath <directory to mount*>
EVM # setenv bootargs console=ttyS0,115200n8 noinitrd rw
   ip=dhcp root=/dev/nfs nfsroot=$(nfshost):$(rootpath),
   nolock mem=116M video=davincifb:vid0=720x576x16,
   2500K:vid1=720x576x16,2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE
   davinci_enc_mngr.ch0_mode=$(videostd)
EVM # boot
```

例えば、*<directory to mount>* は次のようになります。

/home/*<useracct>*/workdir/filesys

ブート時、次の行でブート・モードを確認します。

```
## Booting image at 80700000 ...
...
Starting kernel ...
...
VFS: Mounted root (nfs filesystem).
```

A.4.4 NFS ファイル・システムを使用して TFTP 経由でブートする方法

このモードでブートするには、自動ブートシーケンス中断後に次のパラメータをセットします。

```
EVM # setenv bootcmd 'dhcp;bootm'
EVM # setenv serverip <ip addr of tftp server>
EVM # setenv bootfile <name of kernel image>
EVM # setenv rootpath <root directory to mount>
EVM # setenv nfshost <ip addr of nfs host>
EVM # setenv bootargs console=ttyS0,115200n8 noinitrd rw
   ip=dhcp root=/dev/nfs nfsroot=$(nfshost):$(rootpath),nolock
   mem=116M video=davincifb:vid0=720x576x16,
   2500K:vid1=720x576x16,2500K:osd0=720x576x16,2025K
   davinci_enc_mngr.ch0_output=COMPOSITE
   davinci_enc_mngr.ch0_mode=$(videostd)
EVM # boot
```

<root directory to mount> はワークステーション上でセットしたファイル・システムと一致していなければなりません。たとえば次のようになります。

/home/<*useracct*>/workdir/filesys

ブート時、次の行でブート・モードを確認します。

```
TFTP from server 192.168.160.71; our IP address is 192.168.161.186
Filename 'library/davinci/0.4.2/uImage-dm355'.
...
Starting kernel ...
...
VFS: Mounted root (nfs filesystem).
```

A.5 ブートローダのアップデートおよびリストア

DM355EVM ボード上には 2GB の NAND 型フラッシュ・メモリが搭載されています。EVM ボードをリセットすると、ROM ブートローダ（RBL）が実行され、ボードの初期化や、NAND フラッシュ・メモリから内部メモリへ UBL（User Bootloader）と呼ばれる小さなプログラムのロードが行われます。UBL は NAND フラッシュ・メモリから U-Boot ブートローダ・プログラムを順番にロードします。U-Boot ブートローダ・プログラムは Linux カーネルのロードとスタートのために必要なプログラムです。

したがって、NAND フラッシュ・メモリには、これら 2 種類のブートローダ・イメージが保存されていることになります。UBL と U-Boot です。このセクションでは、これらブートローダ・イメージが破損してしまったり、アップデートする必要がある場合に、UBL と U-Boot をフラッシュへ書き込む方法を説明します。

U-Boot イメージが破損していなければ、U-Boot 自体を使ってアップデートすることができます。もしフラッシュ上に実行可能な U-Boot イメージが存在しない場合には、Code Composer Studio（CCS）とエミュレータを使用して、これらのイメージを元に戻す必要があります。以下、この両方の手順について説明します。

DVSDK をインストールすると、UBL、U-Boot、またこれらの NAND フラッシュへのプログラミングについて、通常次のディレクトリで情報を得ることができます。

`/home/<useracct>/dvsdk_#_#/PSP_#_#_#_#/board_utilities`

A.5.1 U-Boot を使用した U-Boot のアップデート

元の状態の U-Boot イメージが存在している場合には、それ自身を使用して、U-Boot イメージのアップデートを行うことができます。次のステップに従ってください。

> **注：** この作業には NAND フラッシュの一部の領域の消去が含まれています。誤って必要以上の領域やフラッシュ全体を消去してしまった場合には A.5.2 項、A.6 節を参照して NAND フラッシュの復旧を行ってください。

1) 次のどちらか 1 つの方法で、EVM ボートに IP アドレスを割り当てます。

   ■ 自動で行われるブートシーケンスを中止した後、ワークステーションが、スタンドアロンネットワーク、またはクロスケーブルを使用したネットワーク接続の場合､次のコマンドを使用して､静的 IP アドレスを割り当てます。

   ```
   EVM # setenv ipaddr <static IP address>
   ```

   ■ 動的な IP アドレスを割り当てるには、次のコマンドを使用します。

   ```
   EVM # dhcp
   EVM # setenv ipaddr <IP address returned by dhcp>
   ```

2) TFTP サーバに IP アドレスをセットします。

   ```
   EVM # setenv serverip <TFTP server IP address>
   ```

3) これらのセッティングをフラッシュ・メモリにセーブします。

   ```
   EVM # saveenv
   ```

4) U-Boot をロードします。ロードを行うためには、U-Boot イメージは TFTP ディレクトリ（通常は /tftp）へコピーされる必要があります。また、このとき、tftp コ マンドではサーバにあるイメージ名とディスティネーションアドレスが特定される必要があります。この場合、ディスティネーションアドレスは 0x80700000 から始まる DDR のメモリスペース中から任意に選ばれます。

   ```
   EVM # tftp 0x80700000 <u-boot file name>
   ```

NAND Device	Total Size	Sector Size	U-Boot Load Address
MT29F16G08FAA (SLC) (default NAND chip on DM355 EVM)	2 GiB	128 KiB	0x140000
MT29F4G08AAA (SLC)	512 MiB	128 KiB	0x140000
MT29F16G08QAA (MLC)	2 GiB	256 KiB	0x280000

5) <U-Boot Load Address> より前の領域で、転送するバイト値よりも大きなサイズ分の消去を行います。

```
EVM # nand erase <U-Boot Load Address> 0x20000
```

6) フラッシュ（0x80700000）にある新しい U-Boot を NAND デバイスの U-Boot Load Address へ書き込みます。

```
EVM # nand write 0x80700000 <U-Boot Load Address>  0x20000
```

7) ボードの電源を入れなおすか、リブートのためのリセットを実行します。シリアル・ターミナル・コンソール・アウトプット上でビルドが実行された日時を確認することで、U-Boot が正しく実行されていることを確認してください。

A.5.2 エミュレータと CCS を使用した UBL と U-Boot ブートローダのアップデート

フラッシュ上に U-Boot（または UBL）イメージが存在しない場合には、CCS とエミュレータを使用してこれらのイメージをリストアすることができます。次のステップに従ってください。

1) NAND プログラマ・ユーティリティを見つけてください。NAND プログラマ・バイナリは以下のディレクトリにあります。
/home/<useracct>/dvsdk_#_#/PSP_#_#_#_#/bin/dm355
また、NAND プログラマ・ユーティリティのソースは DVSDK がインストールされた以下のディレクトリから引き出すことができます。
/home/<useracct>/dvsdk_#_#/PSP_#_#_#_#/board_utilities

Ver3.3 以上の CCS と XDS560 か XDS510 エミュレータをインストールした PC ワークステーション上でこれらのユーティリティを使用します。

2) DM355 GEL ファイルを用いて CCS Setup を使用し、DM355 EVM ボードと CCS をコネクトするためのコンフィグレーションを行います。.ccs と .gel ファイルは、PSP パッケージに含まれていません。下記よりこれらをダウンロードすることが可能です。
http://c6000.spectrumdigital.com/evmdm355

3) EVM ボード上の JTAG コネクタと接続し、EVM ボードに電源を供給します。

4) CCS を立ち上げ、デバイスとのコネクトを行います（Alt+C）。

5) Load Program で NAND_programmer.out をロードし、RUN します（F5）。

6) ダイアログ・ボックスに UBL のパスとファイル名（ublDM355-nand.bin）を入力します。

7) ダイアログ・ボックスに U-Boot のパスとファイル名（u-boot-1.2.0-dm355_evm.bin）を入力します。

8) プログラムのアップデートが終了すると CCS には書き込み終了のメッセージ（「NAND Flash programming is completed」）が表示されます。このような状態になったら CCS を終了して電源を入れなおしてください。起動後 U-boot プロンプトが表示されたら、いずれかのキーを押して自動ブートシーケンスを停止させて A.4 に記載のあるいずれかのブート設定を行ってください。

A.6 NAND フラッシュのリストア

この節では、EVM ボード上の DVSDK NAND フラッシュ・メモリの内容をリストアする 2 種類の方法を説明します。これらの内容には、Linux カーネルとファイル・システム、デモ・アプリケーション・ソフトウエアを含んでいます。

DVSDK NAND イメージは DVSDK の CD#3 に含まれます。（または、エクストラネット http://www.ti.com/dvevmupdates）
ファイル名：dm355_flash_image_#_#_#_#.tar（#_#_#_# はバージョンです）

A.6.1 NFS を使用した NAND フラッシュのリストア

Linux カーネル（uImage）は TFTP 経由で NAND フラッシュにロードすることができます。カーネルをロードするために、サーバの tftp ディレクトリ上（通常は /tftp）にあるカーネルのファイル名とデスティネーション・アドレスを指定して下さい。

次の手順でコマンドを実行し、カーネル・イメージのダウンロードと NAND への書き込みを行います。

1) 次のどちらか１つの方法で、EVM ボートに IP アドレスを割り当てます。
   - ワークステーションが、スタンドアロンネットワーク、またはクロスケーブルを使用したネットワーク接続の場合、次のコマンドを使用して、静的 IP アドレスを割り当てます。

     ```
     EVM # setenv ipaddr <static IP address>
     EVM # setenv serverip <tftp server IP address>
     EVM # tftp 0x80700000 uImage
     ```

   - 動的な IP アドレスを割り当てるには、次のコマンドを使用します。

     ```
     EVM # setenv bootfile uImage
     EVM # setenv serverip <tftp server IP address>
     EVM # dhcp
     ```

2) 次のコマンドを実行すると、カーネル・イメージのダウンロードと NAND フラッシュへの書き込みが行われます。

   ```
   EVM # nand erase 0x400000 0x200000
   EVM # nand write 0x80700000 0x400000 0x200000
   ```

一旦ダウンロードされたカーネル・バイナリを NAND フラッシュへ対応させるには、YAFFS2 イメージ（dm355_flash_image_#_#_#_#.tar）を NFS 経由でマウントします。YAFFS2 イメージは NFS サーバのルート・ディレクトリ上に存在していなければなりません。次のステップに従ってください。

1) DVSDK CD から、NFS マウントしたルート・ディレクトリ（例えば /home/<*useracct*>/workdir/filesys）へ dm355_flash_image_#_#_#_#.tar ファイルをコピーします。

2) カーネルへのブートと NFS へのマウントには、環境変数「bootcmd」をセットします（または EVM IP アドレスのために「dhcp」コマンドを使ってください）。

   ```
   EVM # setenv bootcmd 'nboot 0x80700000 0 0x400000; bootm'
   EVM # setenv bootargs console=ttyS0,115200n8 noinitrd
     ip=dhcp root=/dev/nfs rw nfsroot=<nfs_host_ip>:<nfs_root_path> mem=116M
     video=davincifb:vid0=720x576x16,2500K:vid1=720x576x16,
     2500K:osd0=720x576x16,2025K davinci_enc_mngr.ch0_output=COMPOSITE
     davinci_enc_mngr.ch0_mode=$(videostd)
   ```

   注： NFS はテンポラリのファイル・システムなので、これらの変数はセーブする必要はありません。

3) Linux カーネルをブートするため、boot コマンドを実行します。

4) EVM にルートでログインし、以下の U-Boot コマンドを実行して、YAFFS2 イメージを NAND フラッシュへマウントします。

   ```
   EVM # mkdir /mnt/nand
   EVM # flash_eraseall /dev/mtd3
   ```

```
EVM # mount -t yaffs2 /dev/mtdblock3 /mnt/nand/
EVM # cd /mnt/nand
EVM # tar xf /dm355_flash_image_#_#_#_#.tar
EVM # cd
EVM # umount /mnt/nand
EVM # reboot
```

5) EVM が再起動されたら、Esc キーを押して U-Boot プロンプトに戻ってください。ただちに、A.4.1 項で記述されている U-Boot 環境変数をリストアすることができます。

A.6.2 RAM ディスクと 2GB の SD カードを使用した NAND フラッシュのリストア方法

この手順は、A.3 節の TFTP セットアップが完了した後に行ってください。Ramdisk イメージ（Ramdisk.gz）は TFTP によって DDR メモリにロードすることができます。YAFFS2 イメージ（dm355_flash_image_#_#_#_#.tar）は、カードリーダ等を使用して SD カードにあらかじめ置いてください。作業を始める前に、A.5.1 項の作業 1)、2) を参照し EVM ボードに IP アドレスを割り当ててくださいい。手順は以下のようになります。

1) /home/<useracct>/dvsdk_#_#/PSP_#_#/bin ディレクトリから、RAM ディスクイメージを見つけ出し、ホストの /tftp ディレクトリへコピーします。

2) ブートシーケンスを中止した後で、次のように、RAM に RAM ディスクイメージをダウンロードします。

```
EVM # tftp 0x82000000 ramdisk.gz
```

3) カーネルにブートするためと RAM ディスクにマウントするために、以下の環境設定を行ってください。

```
EVM # setenv bootcmd 'nboot 0x80700000 0 0x400000; bootm'
EVM # setenv bootargs mem=116M console=ttyS0,115200n8
  root=/dev/ram0 rw initrd=0x82000000,4M ip=off
  video=davincifb:vid0=720x576x16,2500K:vid1=720x576x16,
  2500K:osd0=720x576x16,2025K
  davinci_enc_mngr.ch0_output=COMPOSITE
  davinci_enc_mngr.ch0_mode=$(videostd)
```

注： RAM ディスクはテンポラリのファイルシステムなので、これらの変数は NAND フラッシュへセーブする必要はありません。

---

4) 2GB の SD カードを DM355EVM の MMC/SD スロットに入れます。

注： カーネルブート後にカードが挿入されると、カーネルがロックアップします。

---

5) 次のコマンドで、カーネルへのブートを実行します。

```
EVM # boot
```

6) root ユーザー名を使い、EVM へログインします。（「root」はパスワードが必要ありません）

7) 次のコマンドで、MMC/SD へのマウント、NAND への YAFFS2 イメージの転送を実行します。

```
EVM # mkdir /mnt/mmc
EVM # mkdir /mnt/nand
EVM # mount -t vfat /dev/mmcblk0 /mnt/mmc/
(または EVM # mount -t vfat /dev/mmcblk0p1 /mnt/mmc/

EVM # flash_eraseall /dev/mtd3
EVM # mount -t yaffs2 /dev/mtdblock3 /mnt/nand/
EVM # cd /mnt/nand
EVM # tar xf /mnt/mmc/dm355_flash_image_#_#_#_#.tar
EVM # cd
EVM # umount /mnt/nand
EVM # reboot
```

最初の “mount“ コマンドは、VFAT 形式のファイルシステムのマウントを想定しています。tar コマンドはおよそ 1 分ほど掛かります。

vfat ファイルシステムを使用した場合、dm355_flash_image_#_#_#_#.tar ファイルは DOS 8.3 形式のファイルフォーマットでセーブされた dm355_fl.tar として確認できます。

索引

R

S

T

U

V

Y

あ

い

き

く

け

こ

さ

し

す

せ

そ

Made in Japan
’08.09

日本テキサス・インスツルメンツ株式会社

JAJU096
（SPRUF73a）

ご注意

日本テキサス・インスツルメンツ株式会社（以下TIJといいます）及びTexas Instruments Incorporated（TIJの親会社、以下TIJないしTexas Instruments Incorporatedを総称してTIといいます）は、その製品及びサービスを任意に修正し、改善、改良、その他の変更をし、もしくは製品の製造中止またはサービスの提供を中止する権利を留保します。従いまして、お客様は、発注される前に、関連する最新の情報を取得して頂き、その情報が現在有効かつ完全なものであるかどうかご確認下さい。全ての製品は、お客様とTIJとの間に取引契約が締結されている場合は、当該契約条件に基づき、また当該取引契約が締結されていない場合は、ご注文の受諾の際に提示されるTIJの標準販売契約約款に従って販売されます。

TIは、そのハードウェア製品が、TIの標準保証条件に従い販売時の仕様に対応した性能を有していること、またはお客様とTIJとの間で合意された保証条件に従い合意された仕様に対応した性能を有していることを保証します。検査およびその他の品質管理技法は、TIが当該保証を支援するのに必要とみなす範囲で行なわれております。各デバイスの全てのパラメーターに関する固有の検査は、政府がそれ等の実行を義務づけている場合を除き、必ずしも行なわれておりません。

TIは、製品のアプリケーションに関する支援もしくはお客様の製品の設計について責任を負うことはありません。TI製部品を使用しているお客様の製品及びそのアプリケーションについての責任はお客様にあります。TI製部品を使用したお客様の製品及びアプリケーションについて想定されうる危険を最小のものとするため、適切な設計上および操作上の安全対策は、必ずお客様にてお取り下さい。

TIは、TIの製品もしくはサービスが使用されている組み合せ、機械装置、もしくは方法に関連しているTIの特許権、著作権、回路配置利用権、その他のTIの知的財産権に基づいて何らかのライセンスを許諾するということは明示的にも黙示的にも保証も表明もしておりません。TIが第三者の製品もしくはサービスについて情報を提供することは、TIが当該製品もしくはサービスを使用することについてライセンスを与えるとか、保証もしくは是認するということを意味しません。そのような情報を使用するには第三者の特許その他の知的財産権に基づき当該第三者からライセンスを得なければならない場合もあり、またTIの特許その他の知的財産権に基づきTI からライセンスを得て頂かなければならない場合もあります。

TIのデータ・ブックもしくはデータ・シートの中にある情報を複製することは、その情報に一切の変更を加えること無く、かつその情報と結び付られた全ての保証、条件、制限及び通知と共に複製がなされる限りにおいて許されるものとします。当該情報に変更を加えて複製することは不公正で誤認を生じさせる行為です。TIは、そのような変更された情報や複製については何の義務も責任も負いません。

TIの製品もしくはサービスについてTIにより示された数値、特性、条件その他のパラメーターと異なる、あるいは、それを超えてなされた説明で当該TI製品もしくはサービスを再販売することは、当該TI製品もしくはサービスに対する全ての明示的保証、及び何らかの黙示的保証を無効にし、かつ不公正で誤認を生じさせる行為です。TIは、そのような説明については何の義務も責任もありません。

TIは、TIの製品が、安全でないことが致命的となる用途ないしアプリケーション（例えば、生命維持装置のように、TI製品に不良があった場合に、その不良により相当な確率で死傷等の重篤な事故が発生するようなもの）に使用されることを認めておりません。但し、お客様とTIの双方の権限有る役員が書面でそのような使用について明確に合意した場合は除きます。たとえTIがアプリケーションに関連した情報やサポートを提供したとしても、お客様は、そのようなアプリケーションの安全面及び規制面から見た諸問題を解決するために必要とされる専門的知識及び技術を持ち、かつ、お客様の製品について、またTI製品をそのような安全でないことが致命的となる用途に使用することについて、お客様が全ての法的責任、規制を遵守する責任、及び安全に関する要求事項を満足させる責任を負っていることを認め、かつそのことに同意します。さらに、もし万一、TIの製品がそのような安全でないことが致命的となる用途に使用されたことによって損害が発生し、TIないしその代表者がその損害を賠償した場合は、お客様がTIないしその代表者にその全額の補償をするものとします。

TI製品は、軍事的用途もしくは宇宙航空アプリケーションないし軍事的環境、航空宇宙環境にて使用されるようには設計もされていませんし、使用されることを意図されておりません。但し、当該TI製品が、軍需対応グレード品、若しくは「強化プラスティック」製品としてTIが特別に指定した製品である場合は除きます。TIが軍需対応グレード品として指定した製品のみが軍需品の仕様書に合致いたします。お客様は、TIが軍需対応グレード品として指定していない製品を、軍事的用途もしくは軍事的環境下で使用することは、もっぱらお客様の危険負担においてなされるということ、及び、お客様がもっぱら責任をもって、そのような使用に関して必要とされる全ての法的要求事項及び規制上の要求事項を満足させなければならないことを認め、かつ同意します。

TI製品は、自動車用アプリケーションないし自動車の環境において使用されるようには設計されていませんし、また使用されることを意図されておりません。但し、TIがISO/TS 16949の要求事項を満たしていると特別に指定したTI製品は除きます。お客様は、お客様が当該TI指定品以外のTI製品を自動車用アプリケーションに使用しても、TIは当該要求事項を満たしていなかったことについて、いかなる責任も負わないことを認め、かつ同意します。

弊社半導体製品の取り扱い・保管について

半導体製品は、取り扱い、保管・輸送環境、基板実装条件によっては、お客様での実装前後に破壊/劣化、または故障を起こすことがあります。

弊社半導体製品のお取り扱い、ご使用にあたっては下記の点を遵守して下さい。

1. 静電気
   - 素手で半導体製品単体を触らないこと。どうしても触る必要がある場合は、リストストラップ等で人体からアースをとり、導電性手袋等をして取り扱うこと。
   - 弊社出荷梱包単位（外装から取り出された内装及び個装）又は製品単品で取り扱いを行う場合は、接地された導電性のテーブル上で（導電性マットにアースをとったもの等）、アースをした作業者が行うこと。また、コンテナ等も、導電性のものを使うこと。
   - マウンタやはんだ付け設備等、半導体の実装に関わる全ての装置類は、静電気の帯電を防止する措置を施すこと。
   - 前記のリストストラップ・導電性手袋・テーブル表面及び実装装置類の接地等の静電気帯電防止措置は、常に管理されその機能が確認されていること。
2. 温・湿度環境
   - 温度：0～40℃、相対湿度：40～85%で保管・輸送及び取り扱いを行うこと。（但し、結露しないこと。）
   - 直射日光があたる状態で保管・輸送しないこと。
3. 防湿梱包
   - 防湿梱包品は、開封後は個別推奨保管環境及び期間に従い基板実装すること。
4. 機械的衝撃
   - 梱包品（外装、内装、個装）及び製品単品を落下させたり、衝撃を与えないこと。
5. 熱衝撃
   - はんだ付け時は、最低限260℃以上の高温状態に、10秒以上さらさないこと。（個別推奨条件がある時はそれに従うこと。）
6. 汚染
   - はんだ付け性を損なう、又はアルミ配線腐食の原因となるような汚染物質（硫黄、塩素等ハロゲン）のある環境で保管・輸送しないこと。
   - はんだ付け後は十分にフラックスの洗浄を行うこと。（不純物含有率が一定以下に保証された無洗浄タイプのフラックスは除く。）

以上