参考資料

Application Note

TDA4x の単一 eMMC を使用した 統合ブート モードによる冗長ブート

Johnny Kim *Jacinto 7*

概要

車載用先進運転支援システム (ADAS) では、特にブート プロセス時にシステムの信頼性が非常に重要になります。このアプリケーション ノートでは、TI の TDA4x Jacinto デバイスにおいて、単一の eMMC デバイスを使用して冗長ブート機能を実現する方法を紹介します。システムは、プライマリ ブート モードとして組込み型マルチメディア カード(eMMC) ブート モードを使用し、バックアップ ブート モードとしてマルチメディア カード/セキュア ディスク (MMC/SD) ブート モードを使用します。両方のパスは同じ eMMC ハードウェアを共有しますが、物理ブート モードの変更や追加のメモリ デバイスは必要ありません。

提案されている設計では、たとえアップデート中の予期せぬ電源遮断などによりプライマリのブート イメージが破損した場合でも、システムが正常にリカバリし、ブートできることを確実にします。これは、ブート ローダ イメージを慎重に配置し、異なるブート モードを活用し、同一デバイス内に別々のブート パスを用意することで実現されています。検証は、意図的にプライマリ イメージを破損させ、バックアップのブート パスへ自動的にフォールバックする様子を観察することで実施されます。このアプローチは、ADAS アプリケーション向けに、信頼性が高く、コスト効率に優れ、保守もしやすい冗長ブート設計を提供します。これにより、実際の自動車環境において、システムの堅牢性と可用性が確保されます。

目次

商標

1 はじめに

TI の TDA4x Jacinto デバイスは、オクタル シリアル ペリフェラル インターフェイス (OSPI) フラッシュメモリ、eMMC、ユニバーサル非同期送受信機 (UART) などのさまざまなメディアからのブートをサポートしています。電源投入後、マイコン ユニット (MCU) の ROM コードは、通常は物理的に設定されたブート モードの構成を読み取ることで、ブート シーケンスを開始します。この設定に基づいて、ROM コードはアクセスすべきブート メディアを判断し、必要なブート イメージの取得を行います。

デフォルトでは、ROM コードはまずプライマリのブート メディアからイメージの読み込みを試みます。プライマリのブート イメージが有効で、整合性および認証チェックに合格した場合、システムはこれらのイメージを使用して正常にブートします。ただし、プライマリのブート イメージが存在しない場合や破損している場合、ROM コードは自動的にバックアップのブート メディアによるブートを試みます。バックアップ メディアによるブート処理も、プライマリと同じ手順および検証ステップに従って実行されます。このデュアル パスのブート機構により、システムは 2 回のブート機会を持つことになり、特に ADAS アプリケーションで重要となる全体的な信頼性と耐障害性が大幅に向上します。ハイレベルのブート シーケンスはこのプロセスに従います。

1. 電源が供給され、MCU ROM のブート プロセスが開始されます。
2. MCU の ROM コードは、ブート モード ピンを読み取ってプライマリのブート ソースを識別します。
3. MCU の ROM コードは、ブート イメージの整合性を検証するように、デバイス管理セキュリティコントローラ (DMSC) に要求します。
4. 整合性チェックに合格した場合、システムはプライマリ イメージを使用してブートします。
5. 整合性チェックに失敗した場合、MCU の ROM コードは再度ブート モード ピンを読み取り、バックアップのブート ソースを識別します。
6. DMSC はバックアップ イメージの整合性チェックを実行します。
7. 正常に実行されると、システムはバックアップ メディアからブートします。

このプロセスにより、冗長なブート機能が提供され、イメージの破損やアップデート失敗時にも堅牢なリカバリが可能になります。

このアプリケーション ノートの以下のセクションでは、物理的なブート モードの切り替えや複数のメモリデバイスを使用することなく、単一の eMMC デバイスを用いて冗長ブート機構を構成する方法について説明します。

2 物理ブート モード DIP スイッチの設定

ブート手順を判定するために、ROM コードがチェックする以下の 2 組のピンがあります。BOOTMODE ピンと MCU_BOOTMODE ピン。これらのピンは、POST 構成や PLL 構成など、さまざまな設定を構成するために使用されます。

ただし、このアプリケーション ノートでは、対応する表に記載されている、以下のブート関連の構成項目に特に焦点を当てています。

- プライマリ ブート モード A
- プライマリ ブート モード B
- プライマリ ブート モードの構成
- バックアップ ブート モード
- バックアップ ブート モードの構成

表 2-1. MCU_BOOTMODE ピンのマッピング

9	8	7	6	5	4	3	2	1	0
構成後		予約済み	MCU のみ	プライマリ ブート モード A			PLL の構成		

表 2-2. BOOTMODE ピンのマッピング

7	6	5	4	3	2	1	0
バックアップ ブート モードの構成	プライマリ ブート モードの構成			バックアップ ブート モード			プライマリ ブート モード B

TDA4VM EVM では、これらのピンは以下のように DIP スイッチに物理的に割り当てられています。

- BOOTMODE[0:7] → BOOTMODE SW8[1:8]
- MCU_BOOTMODE[2:9] → BOOTMODE SW9[1:8]

注

TDA4 デバイスにおけるマッピングの構成は、共通プロセッサ ボード上で直接行われますが、カスタム ボードの設計内容によっては異なる場合があります。

eMMC メモリは、eMMC ブート モードとしても、MMC/SD カード ブート モードとしても使用できます。このアプリケーション ノートでは、単一の eMMC デバイスを使用し、eMMC ブート モードをプライマリ ブート モードとして、MMC/SD カード ブート モードをバックアップ ブート モードとして使用する方法を提案しています。

2.1 プライマリ ブート モード用の eMMC ブート モード スイッチ

表 2-3、表 2-4 および表 2-6 に、リセット後に最初に試行したブート モードとしてプライマリ ブート モードを構成するための詳細情報を示します。

eMMC ブートをプライマリ ブート モードとして構成するには、次のピン設定を使用します。

1. eMMC ブート モードを選択
   - プライマリ ブート モード B ピン：BOOTMODE[0] = 0b1
   - プライマリ ブート モード A ピン：MCU_BOOTMODE[3:5] = 0b100

表 2-3. MCU のみ = 0 のときのプライマリ ブート モードの選択

プライマリ ブート モード B ピン	プライマリ ブート モード A ピン			選択したブート モード
0	**MCU 5**	**MCU 4**	**MCU 3**	
1	0	0	1	eMMC

2. eMMC ブート設定の構成 (表 2-4 および表 2-5 も参照)
   - BOOTMODE[4:6] = 0b000 → 1.8V I/O、最大バス幅 (ポート 0 で 8 ビット)、ポート 0 を使用
3. MCU のみの構成
   - MCU_BOOTMODE[6] = 0b0

表 2-4. プライマリ ブート モードの構成

プライマリ ブート モード構成ピン			プライマリ ブート モード B ピン	プライマリ ブート モード A ピン			プライマリ ブート モード
6	**5**	**4**	**0**	**MCU 5**	**MCU 4**	**MCU 3**	
ポート	バス幅	電圧	1	0	0	1	eMMC

表 2-5. eMMC ブート構成フィールド

ブートモード ピン	フィールド	値	説明
6	ポート	0 1	ポート 0 ポート 1
5	バス幅	0 1	ポート別最大幅 (ポート 0 では 8 ビット) 1 ビットのみ
4	電圧	0 1	1.8V 3.3V

その結果、DIP スイッチの設定を次のように設定します。

- BOOTMODE SW8[1:8]: 1XXX_000X
- MCU_BOOTMODE SW9[1:8]: X100_0XXX

注

「X」は、この文脈において設定に影響しない「無視してよい値」を示します。

2.2 バックアップ ブート モード用の MMC/SD ブート モード スイッチ

表 2-6 および 表 2-7 はプライマリのブート モードが失敗した場合や復帰時に備えて、バックアップ ブート モードをどのように構成するかについての詳細情報を示します。

バックアップ ブート モードの場合は、eMMC ブートモードではなく MMC/SD カード ブート モードを構成します。これを実現するには、次のピン設定が必要です。

1. 表 2-6 で定義されているバックアップ ブート モード ピンを使用して、MMC/SD カード ブート モードを選択します。
   - BOOTMODE[1:3] = 0b101 → MMC/SD カード ブート モード

表 2-6. MCU のみ = 0 の場合のバックアップ モードの選択

バックアップ ブート モード ピン			選択したバックアップ ブート モード
3	2	1	
1	0	1	MMC/SD カード

2. MMC/SD ブート モード設定の構成し
   - BOOTMODE[4:5] = 0b00 → 4 ビット バス幅、ファイル システム (FS) モード
   - BOOTMODE[7] = 0b0 → ポート 0 を選択

表 2-7. MMC/SD ブート構成フィールド

ブートモード ピン	フィールド	値	説明
7	ポート	0 1	ポート 0 ポート 1
5	バス幅	0 1	4 ビット 1 ビット
4	FS/Raw	0 1	FS モード Raw モード

その結果、MMC/SD カードによるバックアップ ブート モードの DIP スイッチ設定は以下のとおりになります:

- BOOTMODE SW8[1:8]: X101_00X0
- MCU_BOOTMODE SW9[1:8]: XXXX_XXXX (バックアップ ブート設定には関係ありません)

MMC コントローラのポート 0 に接続された単一の eMMC デバイスで、プライマリおよびバックアップの両方のブート モードをサポートするには、以下の統一されたスイッチ設定を使用します。

- BOOTMODE SW8[1:8]: 1101_0000
- MCU_BOOTMODE SW9[1:8]: 0100_0000

3 操作環境

提案された冗長ブートの手法を検証するには、プライマリ ブート用とバックアップ ブート用の 2 セットのブート イメージが必要です。検証に使用される主なコンポーネントには、以下のバイナリが含まれます。

- セカンダリ ブートローダ (SBL)
- app (bootapp)
- TI の基本的なセキュリティ(TIFS)

単純化のために、QNX OS はハイレベル OS として選択されています。QNX コンソールを起動するには、次のバイナリが必要です。

- lateapp1、lateapp2
- atf_optee.appimage
- ifs_qnx.appimage

3.1 プライマリ ブート モード用に eMMC ブート イメージを準備

以下のビルド コマンドを使用して、プライマリ ブート モードとして機能する eMMC ブート用のブート イメージを生成できます。

```
cd ${PDK_PATH}/packages/ti/build
make -sj6 sbl_emmc_boot0_img BOARD=j721e_evm CORE=mcu1_0
make -sj6 boot_app_mmcsd_qnx HLOSBOOT=qnx BOARD=j721e_evm CORE=mcu1_0
```

ビルド プロセスが完了すると、次のバイナリが生成されます。

- tiboot3.bin (SBL)
- app (bootapp)

バイナリをビルドした後、あらかじめ定義されたオフセットに従って、それらを eMMC の boot1 パーティションにフラッシュします。

```
mmc dev 0 1
mmc partconf 0 1 1 1
mmc bootbus 0 2 0 0
fatload mmc 1 ${loadaddr} tiboot3.bin
mmc write ${loadaddr} 0x0 0x400
fatload mmc 1 ${loadaddr} tifs.bin
mmc write ${loadaddr} 0x400 0x1000
fatload mmc 1 ${loadaddr} app
mmc write ${loadaddr} 0x1400 0x2000
```

tiboot3.bin は 0x0 でフラッシュされ、tifs.bin は 0x400 でフラッシュされ、APP は 0x1400 でフラッシュされます。

3.2 バックアップ ブート モード用に MMC/SD ブート イメージを準備

バックアップ ブート モードとして MMC/SD ブートをサポートするには、別途ブート イメージ一式を準備する必要があります。適切なビルド コマンドを使用して、必要なバイナリを生成します。ビルド後、次のファイルが作成されます。

- tiboot3.bin (SBL)
- app (bootapp)

```
cd ${PDK_PATH}/packages/ti/build
make -sj6 sbl_emmc_uda_img BOARD=j721e_evm CORE=mcu1_0
make -sj6 boot_app_mmcsd_qnx HLOSBOOT=qnx BOARD=j721e_evm CORE=mcu1_0
```

これらのバイナリは、eMMC のユーザ データ領域(UDA) パーティションにコピーします。バックアップ ブート プロセスでは、この領域からバイナリを読み込みます。

```
mount /dev/mmcblk0p1 ./emmc_uda_partition
export DST_DIR=/home/root/emmc_uda_partition

cp tiboot3.bin ${DST_DIR}/
cp app ${DST_DIR}/
cp tifs.bin ${DST_DIR}/
sync
```

注

バックアップ ブート用に使用する tiboot3.bin を修正し、tifs.bin およびアプリケーションを、eMMC のブート パーティションではなく、eMMC UDA パーティションから正しく読み込めるようにますし。

3.3 eMMC メモリのレイアウト

表 3-1. プライマリ バックアップ ブートモードおよびバックアップ ブート モードの eMMC レイアウト

eMMC レイアウト	2 進	コメント
プライマリ ブート用のブート エリア パーティション (RAW モード)	tiboot3.bin	オフセット:0x0
	tifs.bin	オフセット:0x400
	アプリケーション	オフセット:0x1400
バックアップ ブート用のユーザ データ領域 (ファイル システム)	tiboot3.bin	
	tifs.bin	
	アプリケーション	
ユーザ データ領域 (ファイル システム)	lateapp1	プライマリ ブート モードおよびバックアップ ブート モードで共通
	lateapp2	
	atf_optee.appimage	
	ifs_qnx.appimage	

プライマリ アプリケーションおよびバックアップ アプリケーション (bootapp) は、指定されたバイナリを eMMC ユーザ データ領域 (UDA) パーティションから読み込む必要があります。

- lateapp1
- lateapp2
- atf_optee.appimage
- ifs_qnx.appimage

この構成により、どのブート パスが選択された場合でも、一貫した実行環境が提供されます。

3.4 冗長ブートの確認

プライマリ ブートおよびバックアップ ブート用の *tiboot3.bin* では、bootapp バイナリの配置場所に若干の違いがあります。冗長ブート メカニズムを確認するために、一意のログ メッセージが *tiboot3.bin* の各バージョンに挿入され、プライマリ ブート モードとバックアップ ブート モードを区別します。

システムは、以下のサンプル ログ出力に示されているように、eMMC のブート パーティションに設定されたプライマリ ブート モードから起動します。

```
SBL Revision: 01.00.10.01 (Jun 9 2024 - 13:16:51) from eMMC boot partition #1
TIFS ver: 9.2.4--v09.02.04 (Kool Koala)Starting Sciserver..... PASSED
 BOOT_APP (Jun 9 2024 - 13:16:54) from eMMC boot partition #1 in boot_app_main.c
MCU R5F App started at 7539 usecs
Loading BootImage
:
```

障害をシミュレートして冗長ブートをトリガするには、次のコマンドを使用して eMMC ブート パーティションの *tiboot3.bin* を破損します。

```
mmc dev 0 1
mmc partconf 0 1 1 1
mmc bootbus 0 2 0 0
mw ${loadaddr} 0x00 0x1000
mmc write ${loadaddr} 0x0 0x400
```

システムの再起動後、ROM コードは破損したプライマリ イメージを検出し、自動的にバックアップ ブート モードに切り替わります。この動作は、バックアップ *tiboot3.bin* に関連付けられた一意のログ出力によって確認できます。

```
SBL Revision: 01.00.10.01 (Jun 9 2024 - 13:46:16) from eMMC UDA partition
TIFS ver: 9.2.4--v09.02.04 (Kool Koala)Starting Sciserver..... PASSED
BOOT_APP (Jun 9 2024 - 13:46:18) from eMMC UDA partition in boot_app_main.c
MCU R5F App started at 7193 usecs
Loading BootImage
:
```

4 まとめ

このアプリケーション ノートでは、TI の TDA4x Jacinto デバイス上で、単一の eMMC デバイスを使用して冗長ブートを実装するための、堅牢かつ実用的なアプローチを示しています。このシステムは、eMMC ブートをプライマリ モード、MMC/SD ブートをバックアップ モードとして構成されており、ハードウェアの変更や複数のメモリデバイスを必要とせずに、ブート時の信頼性を確保しています。

このアプローチの主な利点は次のとおりです。

- プライマリおよびバックアップの両方のブート パスをサポートする単一の物理ブート モード構成
- プライマリ ブートとバックアップ ブートの両方に対応する共有 eMMC ハードウェア インターフェイス
- 破損やアップデート失敗時にバックアップ イメージへシームレスにフォールバック

イメージの準備、適切なブート ローダ構成、および意図的なイメージ破損による検証を通じて、この手法は、堅牢なブート動作が求められる自動車システムにおいて、信頼性の高い設計であることが実証されます。この設計は、実運用環境において採用することで、システムの可用性を向上させ、現場での保守作業を削減し、ADAS アプリケーションにおける安全性と堅牢性を全体的に高めることができます。

5 参考資料

1. テキサス・インスツルメンツ、『*TDA4VM 製品ページ*』
2. テキサス・インスツルメンツ、『*J721E DRA829/TDA4VM プロセッサ シリコン リビジョン 2.0、1.1 テクニカル リファレンス マニュアル*』
3. テキサス・インスツルメンツ、*J721EXCPXEVM: Jacinto™ 7 プロセッサ向け共通プロセッサ ボード*
4. テキサス・インスツルメンツ、*PROCESSOR-SDK-RTOS-J721E* および *PROCESSOR-SDK-QNX-J721E*

重要なお知らせと免責事項

テキサス・インスツルメンツは、技術データと信頼性データ (データシートを含みます)、設計リソース (リファレンス デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、テキサス・インスツルメンツ製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した テキサス・インスツルメンツ製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとします。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている テキサス・インスツルメンツ製品を使用するアプリケーションの開発の目的でのみ、テキサス・インスツルメンツはその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。テキサス・インスツルメンツや第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、テキサス・インスツルメンツおよびその代理人を完全に補償するものとし、テキサス・インスツルメンツは一切の責任を拒否します。

テキサス・インスツルメンツの製品は、テキサス・インスツルメンツの販売条件、または ti.com やかかる テキサス・インスツルメンツ製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。テキサス・インスツルメンツがこれらのリソースを提供することは、適用されるテキサス・インスツルメンツの保証または他の保証の放棄の拡大や変更を意味するものではありません。

お客様がいかなる追加条項または代替条項を提案した場合でも、テキサス・インスツルメンツはそれらに異議を唱え、拒否します。

郵送先住所:Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

重要なお知らせと免責事項

テキサス・インスツルメンツは、技術データと信頼性データ (データシートを含みます)、設計リソース (リファレンス デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、 テキサス・インスツルメンツ製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した テキサス・インスツルメンツ製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとします。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている テキサス・インスツルメンツ製品を使用するアプリケーションの開発の目的でのみ、 テキサス・インスツルメンツはその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。 テキサス・インスツルメンツや第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、 テキサス・インスツルメンツおよびその代理人を完全に補償するものとし、 テキサス・インスツルメンツは一切の責任を拒否します。

テキサス・インスツルメンツの製品は、 テキサス・インスツルメンツの販売条件、または ti.com やかかる テキサス・インスツルメンツ製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。 テキサス・インスツルメンツがこれらのリソースを提供することは、適用される テキサス・インスツルメンツの保証または他の保証の放棄の拡大や変更を意味するものではありません。

お客様がいかなる追加条項または代替条項を提案した場合でも、 テキサス・インスツルメンツはそれらに異議を唱え、拒否します。

郵送先住所：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265