参考資料

Application Note

TDA4x と *AM6x* の熱管理

TEXAS INSTRUMENTS

概要

Jacinto™ 7 TDA4 シリーズと Sitara™ AM6x プロセッサは、Keystone アーキテクチャ基づいて TI が導入した最新世代のプロセッサです。これらのプロセッサはヘテロジニアス・マルチコア・アーキテクチャを基盤としており、異なるアプリケーションコアを統合し、対応するコアに異なるタスクを割り当てて並列処理することで、TI プロセッサの利点を最大限に引き出します。プロセッサに集積される演算能力が高まるほど発熱量も増加し、熱設計点を超えて動作を継続すると不可逆的な損傷を引き起こす可能性があります。したがって、プロセッサの寿命に影響を与えずに演算能力を最大限に活用するためには、熱管理が極めて重要となります。すべての TI プロセッサは、オンチップ温度センサと熱管理専用の VTM (電圧熱管理)モジュールを統合しています。オンチップ温度センサの基本動作原理は、半導体材料の物理的特性を活用することにあります。温度が変化すると、半導体材料の抵抗値または電位が変化し、出力電気信号の変化を引き起こします。この変化はマイクロコントローラまたは他の回路によって読み取られ処理され、温度測定を可能にします。

VTM モジュールは電圧管理機能も担っています。本アプリケーションノートでは、TI プロセッサに統合された VTM モジュールの機能について、その動作原理、使用方法、実装されているソフトウェアおよびハードウェア保護方式を含めて紹介します。ソフトウェアコードや具体的なアプリケーション例を議論する際には、AM62A プロセッサをケーススタディとして使用します。対応するコード変更やコマンドは、すべての TDA4x および AM6x シリーズプロセッサに適用可能です。ただし、SDK バージョンの更新に伴い、対応するコーディングパスの修正が必要となる場合があります。

目次

商標

English Document: SDAA146

1 VTM モジュール

VTM (電圧熱管理) モジュールは、内蔵温度センサとユーザーがプログラムした温度イベントに関連する制御、ステータス、割り込み、イベント生成機能を提供します。チップ熱管理においては、主に温度測定、警告出力、および割り込み機能に重点を置いています。図 1-1 はプロセッサ内部の VTM のブロック図を示します。VTM はプロセッサのリセットコントローラと直接接続されているため、プロセッサをリセットするコマンドを直接送信できます。さらに、VTM にはデバイステスト中に設定されるデバイス固有の動作電圧(AVSVNOM)を保存するための一連のメモリマップトレジスタが含まれています。これらの値は、システムの AVS ソフトウェアがデバイスの動作電圧を調整し、最適な動作条件(消費電力と性能のバランス)を達成するために使用できます。

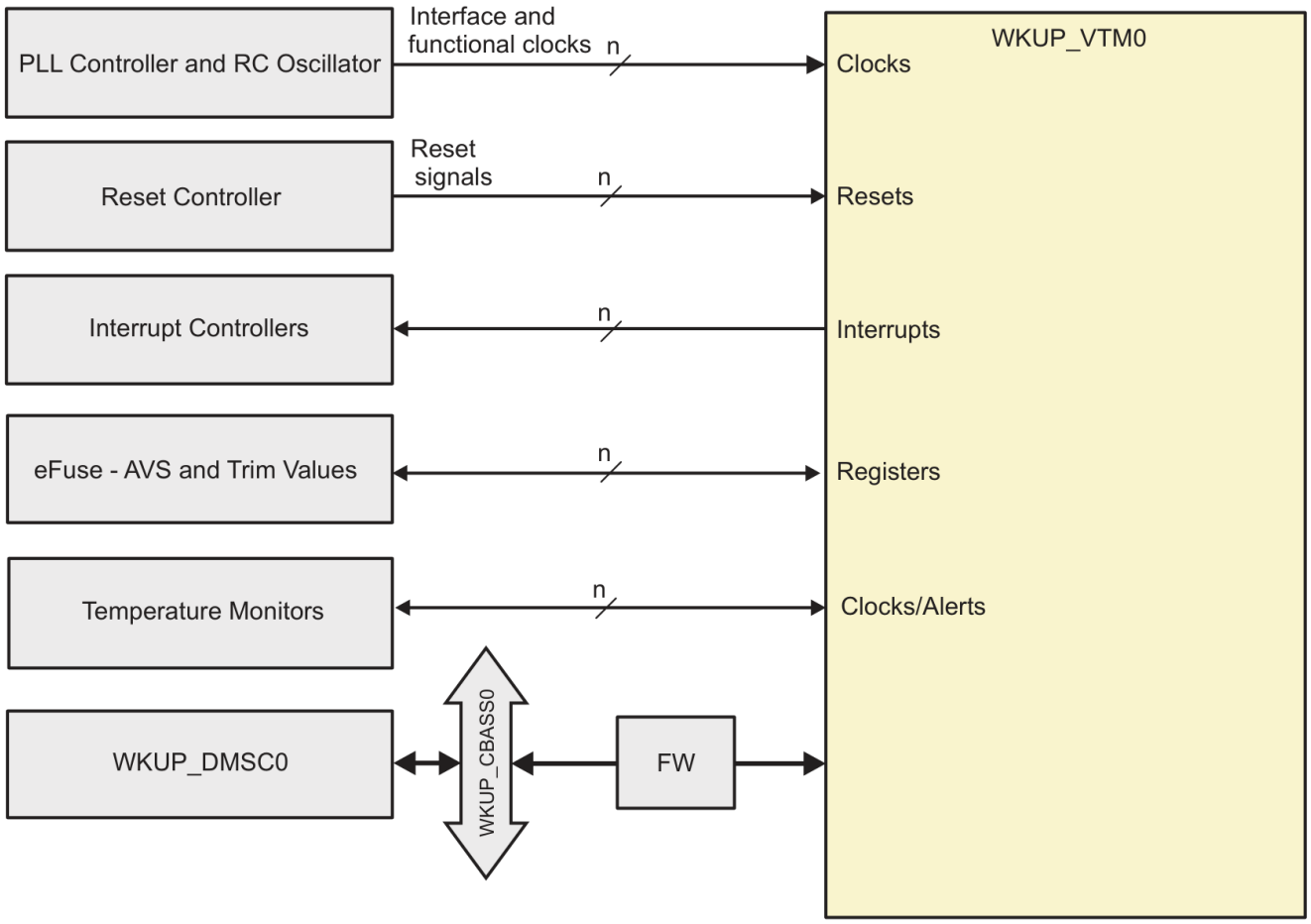

図 1-1. VTM のシステム図

TI には多数の SOC シリーズがあり、それらの VTM システム接続は非常に類似しています。TI には多数の SOC シリーズがあり、それらの VTM システム接続は非常に類似しています。VTM は一般的に、ウェイクアップドメイン(WKUP)に配置されており、SOC が起動した際に最初に動作を開始します。温度管理は電源投入直後に介入し、SOC 全体の正常な動作と長い耐用年数を保証します。

1.1 VTM モジュールの説明

TI SOC における VTM の代表的なレイアウトを図 1-2 に示します。これは、オンチップ温度センサなどの温度モニターが、発熱領域の近くに配置されていることを示しています。VTM モジュールは、内部接続を使用してチップ内の温度モニタを制御します。1 つの VTM で、レジスタを使用して最大 8 つの温度センサを制御できます。温度センサは単独で定期的に更新しないため、VTM は定期的に温度センサを有効化し、報告される温度データを継続的に更新させます。温度センサから返される温度値は VTM レジスタによって捕捉され、VTM 内の対応するレジスタに保存されます。センサーが有効化されていない場合、VTM は省電力化とセンサー使用頻度低減のためリセット状態を維持し、これによりセンサー寿命を最大化します。

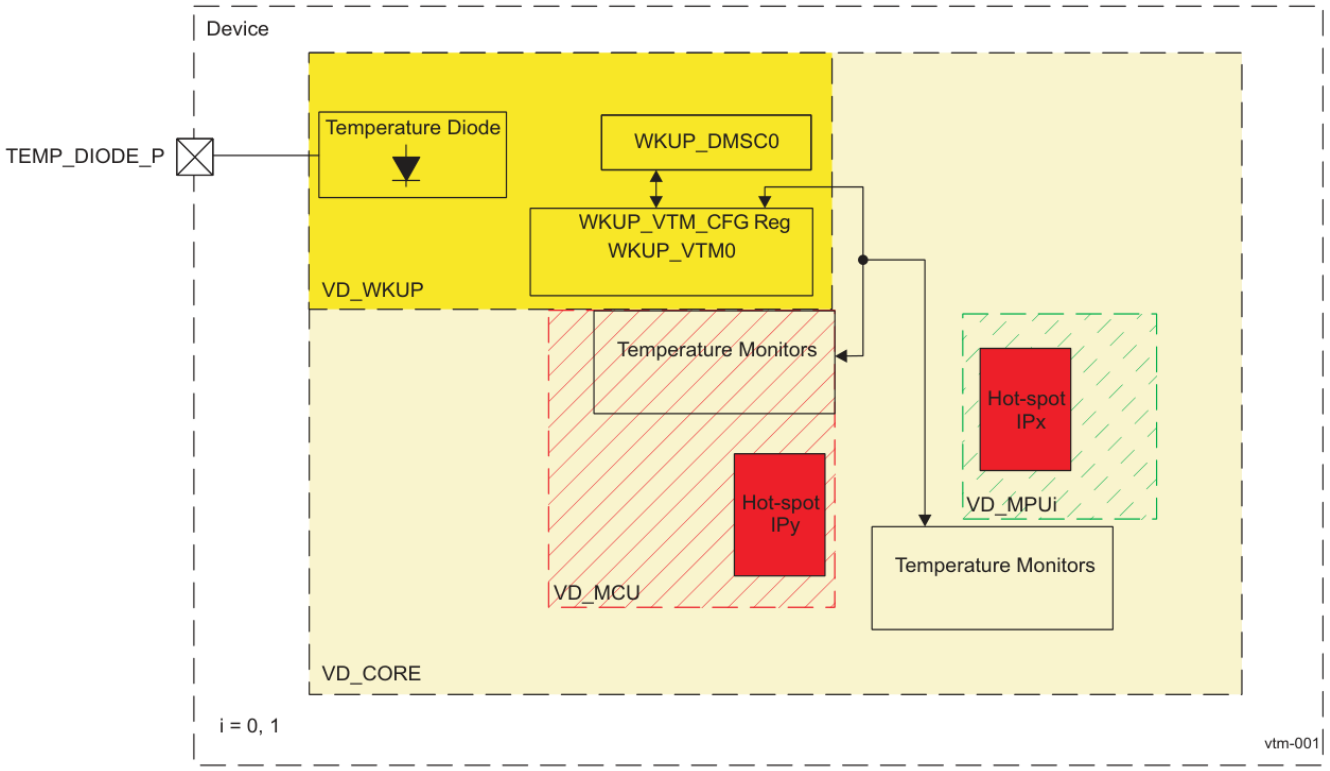

図 1-2. VTM レイアウト

SOC が異なると、通常は異なる数の温度センサが必要になります。原理は、センサを熱源の近くに配置し、すべての加熱源をカバーすることです。表 1-1 は、すべての TI プロセッサにおいて、位置情報と関連付けられた温度センサを収集します。お客様は適用される対応する SOC に応じて結果をルックアップできます。この表は、ホットスポットの物理的近接度に基づいて分類されます。センサは主に 2 つの熱 ソース間や異なる電源ドメイン境界に配置されるため、明確な定義は困難です。本表はセンサ配置の概略的な参考としてのみご利用ください。詳細については、TRM (テクニカル リファレンス マニュアル)を参照してください。

表 1-1. オンチップ温度センサ

オンチップセンサ	合計数	A72/A53	DDR コントローラ	C7x	R5F	GPU	CODEC	DPHY
TDA4VH	7	✓	✓	✓	✓	✓	✓	✓
TDA4VE/VL/AL	7	✓	✓	✓	✓	✓	✓	✓
TDA4VM	5	✓	✓	✓	✓	✓	-	-
TDA4VEN	3	✓	✓	-	-	✓	-	-
DRA821	3	✓	✓	-	✓	-	-	-
AM62A	3	✓	✓	✓	-	-	-	-
AM62P	3	✓	✓	-	-	✓	-	-
AM62x	2	✓	✓	-	-	-	-	-
AM64/AM24	2	✓	✓	-	-	-	-	-
AM62L	1	✓	-	-	-	-	-	-

コアは SOC 内で最も高温になる箇所であるため、すべての SOC では Arm コアの周囲にセンサが配置されています。DDR は SOC 全体のデータスループットを担い、高速で動作するため、過熱のリスクが非常に高くなります。したがって、一般的には、監視のために DDR コントローラにセンサを配置する必要もあります。

1.2 VTM の動作原理および使用方法

前のセクションでは、各 VTM が最大 8 つの温度センサを制御でき、VTM レジスタが温度センサからデータをキャプチャしてレジスタに保存できることについて説明しました。このレジスタの名前は、WKUP_VTM_TMPSENS_STAT_j[9-0] DATA_OUT です。温度値は、対応するレジスタに 10 ビットのバイナリ数として保存されます。表 1-2 に、いくつかの典型的な実際の温度値と、それに対応する 10 ビット温度値を示します。

表 1-2. オンチップ温度センサレジスタの値と実際の値変換

実際の温度	-40	-25	0	25	50	75	100	105	125	150
10 ビット Dec	28	77	164	260	366	485	620	648	773	949
10 ビット Hex	01C	04D	0A4	104	16E	1E5	26C	288	305	3B5

前述の表参照方法は、読み取った値と実際の温度値との大まかな対応関係を示すことができます。この方法は、レジスタに保存されている温度データと実際の温度値との変換にも適用できます。この表参照方法では、概算の温度範囲しか提供されませんが、正確な温度値を得るには、次の式を使用して計算する必要があります。

$$y = 6.0373 \times 10^{-8} \times x^3 - 1.7058 \times 10^{-4} \times x^2 + 0.32512x - 49.002 - 9.2627 \times 10^{-12} \times x^4 \quad (1)$$

上記の式を Python スクリプトでプロットすると、次のような図が生成されます。これは、ほぼ単調増加関数です。このスクリプトは、SOC から読み取ったレジスタ値を入力し、実行後に、対応する正確な温度値を自動的に計算され、注釈が付けられます。スクリプトはこちらからダウンロードできます。

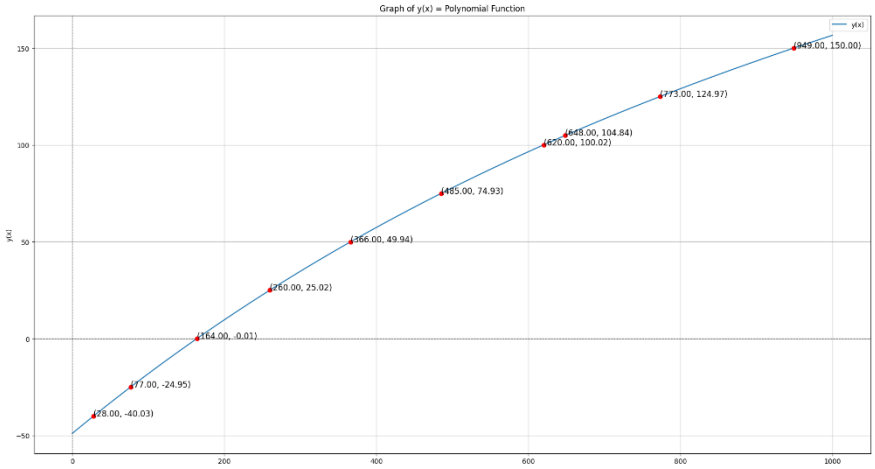

図 1-3. レジスタ値と実際の値の計算式図

これにより、対応する実際の温度を正確に計算できるため、システム温度保護ロジックの変更がより容易になります。例えば、WKUP_VTM_MISC_CTRL2[25-16] MAXT_OUTRG_ALERT_THR0 と WKUP_VTM_MISC_CTRL2[9-0] MAXT_OUTRG_ALERT_THR を変更することで、SOC の最大温度保護のスレッショルドを変更できます。

テキサス インスツルメンツが提供するデフォルトの Linux ソフトウェアでは、オンチップ温度センサの値を読み取るための対応するインターフェイスとコマンドが用意されています。使用されるコマンドは類似しています。例えば、AM62A 評価基板では、以下のコマンドを使用して 3 つの温度センサーの値を読み取ることができます。すべての温度センサの値はミリ摂氏単位で読み取られ、結果は次のように表示されます。センサ 1、センサ 2、センサ 3 の温度はそれぞれ 43.2℃、38.7℃、38.9℃です。センサの位置の違いにより、測定温度は変動する可能性があり、平均値から ±5℃ の偏差が生じます。この範囲は、対応する SOC の最新データシートで定義されています。

```
root@am62axx-evm:/opt/edgeai-gst-apps# cat /sys/class/thermal/thermal_zone*/temp
43209
38749
38983
```

2 TI プロセッサのハードウェア温度保護

これまでのセクションでは主に、TI プロセッサの温度管理を担うハードウェア温度管理モジュールについて紹介しました。VTM モジュールはオンチップ温度センサから値を読み取ることができます。温度を取得後、VTM は測定された温度に基づいて SOC ハードウェア全体の熱保護を実行します。

2.1 VTM の過熱保護スレッショルド

VTM の各温度監視レジスタグループは、対応する温度モニターで温度サンプリングを実行し、最大 3 つのアラート信号をトリガするように設定できます。最初のアラート信号は、スレッショルドポイント THPT1 によって生成される 10 ビットのインクリメンタルポイント GT_TH1_ALERT (過温度アラートコンパレータ 1 結果)です。この値は、VTM_TMPSENSx TH[25-16] TH1_VAL で設定できます。2 番目のアラート信号は、スレッショルドポイント THPT2 によって生成される 10 ビットのインクリメンタルポイント GT_TH2_ALERT (過温度アラートコンパレータ 2 結果)です。この値は、VTM_TMPSENSx_TH2[9-0] TH2_VAL で設定できます。3 番目のアラート信号は、スレッショルドポイント THPT0 によって生成される 10 ビットのインクリメンタルポイント LT_TH0_ALERT (低温アラートコンパレータ結果)です。この値は、VTM_TMPSENSx_TH[9-0] TH0_VAL で設定できます。

通常、設定では THPT2 > THPT1 > THPT0 の順序に従う必要があります。動作原理は以下の説明文で詳述します。

TH1 は、VTM_VTM_TMPSENS_TH_j[25-16] TH1_VAL で定義された スレッショルドを超える温度を示す早期アラートとして設定されます。TH2 は、VTM_VTM_TMPSENS_TH2_j[9-0] TH2_VAL で定義されたスレッショルドを超える温度を示す警告とし設定てされます。この概念では、TH1 はプロセッサの温度上昇を示し、TH2 は直ちに対応を要する状態を示します。TH0 は、VTM_VTM_TMPSENS_TH_j[9-0] TH0_VAL で定義されたスレッショルドより低い温度がセンサによって検出されたときにトリガするよう構成されます。この割り込みは、温度が元の TH1 レベルを下回ったことを示し、温度緩和対策の緩和や緊急状態からの解除が可能となります。なお、LT_TH0_INT が有効化されている場合、TH1 および TH2 割り込みが有効化またはトリガされているか否かにかかわらず、読み取り温度が TH0 を下回ると LT_TH0_INT がトリガされることに留意してください。これら 3 つの割り込みはソフトウェアで検出可能であり、SOC 熱管理方式の設計に適用できます。お客様はニーズに応じて設計可能であり、これは一般的なロジックの参考例です。各センサは、過熱または低温のスレッショルドを個別に設定できます。いずれかのセンサが警告をトリガする限り、対応する割り込みが生成されます。下の図に示すように、すべてのセンサは TH0、TH1、TH2 の低温および過熱割り込みに対応する 3 つの割り込みを生成できます。お客様ソフトウェアはその後、割り込みを処理できます。なお、割り込みはセンサが連続モード時にのみ有効化されます。単一サンプリングでは割り込みはトリガされません。

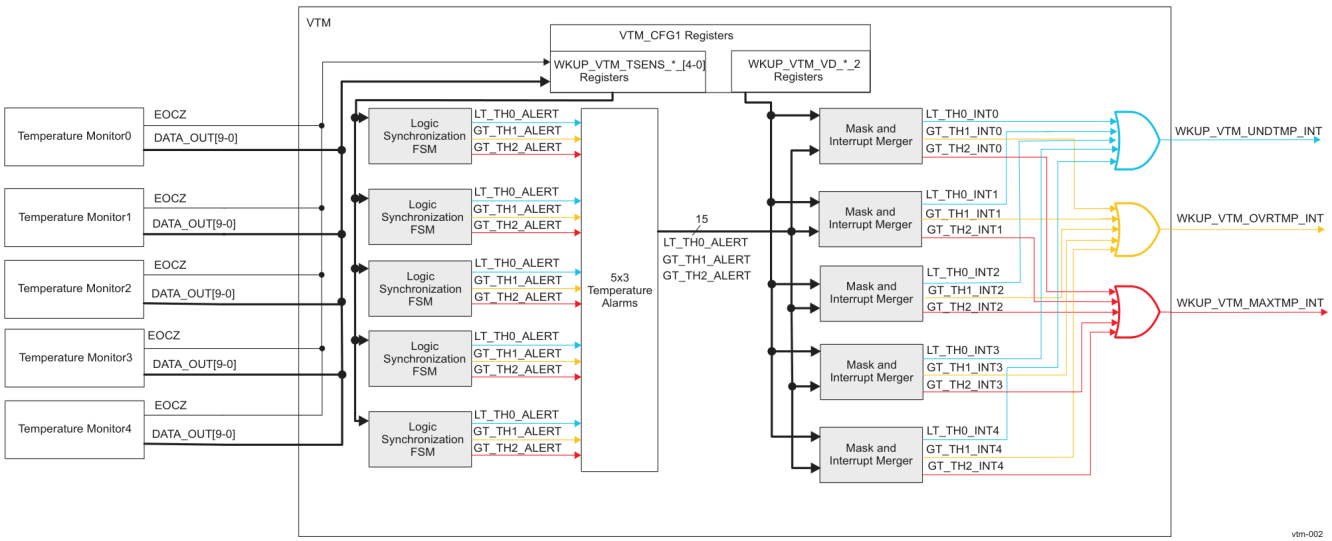

図 2-1. VTM 温度保護割り込みメカニズム図

最終的なスレッショルドは MAXT_OUTRG_ALERT_THR です。このスレッショルドを超えると、デバイスは強制的にリセットされます (また PLL は直接無効化されます)。デバイスが十分に冷却され(MAXT_OUTRG_ALERT_THR0 未満)、内部リセットが解除されると、デバイスは再起動します。この温度保護は、ハードウェアベースの保護メカニズムです。トリガされると、ハードウェアがリセットを実行します。お客様は、対応する スレッショルドと、トリガが有効になっているかどうかを変更できます。ただし、SOC が直ちにリセット状態に入るため、ソフトウェア操作が一切不可能となることから、この割り込みをソフトウェア設計に利用することはできません。また、この保護機能は、センサの動作モードによって制限されません。

2.2 最高ハードウェア温度保護

SOC が最大動作温度を超過せずに動作し、正常な耐用年数を保証するため、TI プロセッサにはハードウェアベースの最大温度保護メカニズムが組み込まれています。このメカニズムはソフトウェアの介入を必要とせず、TI SOC ではデフォルトで有効になっています。このセクションでは、この保護の内部メカニズムについて紹介します。TI SOC が温度が最 スレッショルドを超えた場合に VTM アラートをトリガするには、以下の条件が満たされる必要があります。

1. VTM の最高温度保護を効果的に出力するためには、少なくとも 1 つの温度センサ(具体的には WKUP_VTM_TMPSENS_CTRL_j[11] MAXT_OUTRG_EN = 1 を設定)を有効化するよう VTM を設定する必要があります。
2. さらに、ビット WKUP_VTM_MISC_CTRL[0] ANYMAXT_OUTRG_ALERT_EN を「1」に設定する必要があります。このビットは、VTM の最高温度保護の警告信号出力のスイッチを制御します。
3. 入力および出力経路を確保するコンテキストでは、例えば、温度センサを正しく設定および有効化し、最高温度保護出力を有効にした後、チップ温度が WKUP_VTM_MISC_CTRL2[9-0] MAXT_OUTRG_ALERT_THR でプログラムされた値を超えた場合、センサがプログラムされた最高温度を検出すると、VTM 出力は「1」に駆動されます。
4. SOC レベルでは、VTM は THERM_MAXTEMP_OUTRANGE_ALERT を出力し、PLL コントローラをサーマルリセット状態に移行します。同時に、PLL コントローラはリセット状態に入り、すべての PLL は同時にバイパスモードに移行します。
5. ポイント 4 に基づき、デバイスは消費電力を大幅に低減し、数秒以内にデバイスの温度を低下させます。
6. VTM は、温度を継続的に監視します。VTM が WKUP_VTM_MISC_CTRL2[25-16] MAXT_OUTRG_ALERT_THR0 でプログラムされた値に対応するコードを検出すると、VTM は THERM_MAXTEMP_OUTRANGE_ALERT を 0 に駆動します。
7. ポイント 6 が完了すると、PLL コントローラはリセット状態を終了し、ブートシーケンスを再開します。同時に、PLL はバイパスモードを終了し、チップ全体を目標周波数で開始して、通常動作を開始します。

最高温度ハードウェア保護のフローチャートを、図 2-2 に示します

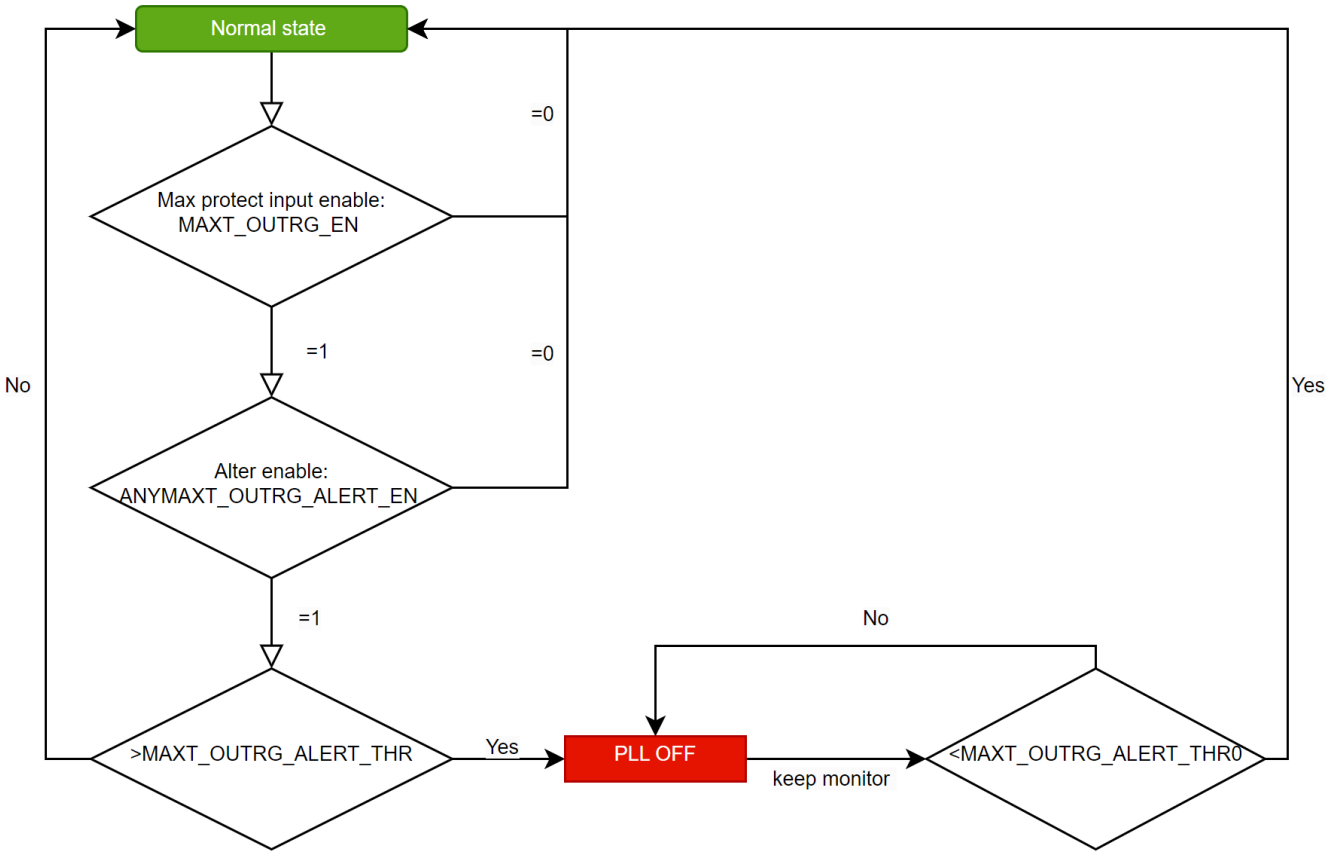

図 2-2. VTM 最高温度ハードウェア保護図

TI SOC のすべてのシリーズで、最高温度保護がデフォルトで有効になっています。SOC を最高温度を超えて動作させると、SOC の耐用年数が永久的に損なわれ、この影響は不可逆的です。各 SOC の通常動作温度範囲については、対応するデバイスのデータシートを参照してください。最高温度保護が作動すると、SOC のクロック PLL はバイパスモードに移行し、SOC 全体のクロック信号が事実上無効化されます。この時点で、VTM はセンサからの信号を使用します。

次の例では、AM62A をケーススタディとして使用しています。TRM に詳細が記載されている特定のレジスタアドレスに対応するレジスタを読み取ることで、SOC 最高温度保護値を取得できます。具体的には、WKUP_VTM_MISC_CTRL2[25-16] MAXT_OUTRG_ALERT_THR0 のエンコード値は 0x288 であり、Python スクリプトを使用すると、この値は実際の温度 105°C に対応します。同様に、WKUP_VTM_MISC_CTRL2[9-0] MAXT_OUTRG_ALERT_THR のエンコード値は 0x2F8 であり、これは実際の温度 123°C に対応します。したがって、AM62A のデフォルト最大ハードウェア温度保護値は 123°C です。システムは温度が 105°C を下回った後にのみ再起動します。

```
root@am62axx-evm:/opt/edgeai-gst-apps# devmem2 0x00b01010
/dev/mem opened.
Memory mapped at address 0xffff8488e000.
Read at address  0x00B01010 (0xffff8488e010): 0x028802F8
```

3 ソフトウェア温度保護戦略

VTM モジュールは、主に SOC 全体の温度を管理するハードウェアモジュールです。VTM モジュールには、最高温度に対するハードウェア保護機能、柔軟な温度保護スレッショルド、そしてユーザー設定可能な設計のための過熱保護割り込みが搭載されていますが、ハードウェアベースの温度保護には限界があります。最高温度に達した際に SOC クロックを停止する以外に、ハードウェアモジュールは周波数低下などの他の熱保護対策を直接制御することはできません。さらに、テキサス インスツルメンツのプロセッサは、マルチコア・ヘテロジニアス・アーキテクチャであり、プロセッサモデルごとに温度センサの数や推奨動作温度が異なるため、VTM の温度保護スレッショルドはデフォルトで 0 に設定されています。つまり、最高温度保護のみがデフォルトで有効になっています。その他の温度保護の割り込みには、システム要件に基づいてユーザーを構成する必要があります。したがって、VTM モジュールには現在、SOC の停止以外の温度保護戦略はありません。補足として、ソフトウェアレベルの温度保護戦略を追加しました。ソフトウェアは VTM モジュールを通じて温度値を取得し、対応する割り込みまたはフラグを生成して、システムが周波数を下げる、高負荷の動作を停止する、システム負荷を下げるなどの対策を講じられるようにします。これは、システムの機能安全にとって不可欠です。

3.1 オプションのソフトウェア温度保護対策

TI のプロセッサは、対応するデバイス用の消費電力推定スプレッドシートを提供しています。このシートは測定データとシミュレーションデータに基づいて消費電力を推定します。SOC の動作消費電力は、電気的パラメータ、プロセスの変動、環境条件、プロセッサ動作時の使用事例などのさまざまな要因に依存しますが、SOC の消費電力の大まかな基準として役立ちます。したがって、このツールを一般的な参考として、周波数スケーリングや負荷低減などのソフトウェア対策の評価に活用できます。SOC の温度と消費電力は積極的に相関しており、消費電力が増加すると温度上昇が速くなります。したがって、ソフトウェア熱管理の主な目標は、SOC の消費電力を削減することです。

表 3-1 は AM62A プロセッサを例として使用し、対応するツールはこちらからダウンロードできます。この方法は、他のテキサスインスツルメンツプロセッサにも適用できます。表 3-1 の結果は、接合部温度 125°C かつすべてのコアが最大周波数で動作する条件下で統計的に分析されています。

表 3-1. AM62A の負荷変動時の消費電力

ケース	電力/mW	A53	DDR	C7x	R5F	DM_R5	Δ 電力/mW
通常のステータス	5136	80%	80%	80%	80%	80%	0
A53 負荷低下	4936	20%	80%	80%	80%	80%	200
DDR 負荷低下	4689	80%	20%	80%	80%	80%	447
C7x 負荷低下	4181	80%	80%	20%	80%	80%	955
R5F 負荷低下	5113	80%	80%	80%	20%	80%	23
DM 負荷低下	5109	80%	80%	80%	80%	20%	27
全負荷低下	3127	20%	20%	20%	20%	20%	2009

表 3-2 の結果は、125°C の接合部温度、および 80% の負荷で動作するすべてのコアの下で行われたものです。

表 3-2. AM62A 動作周波数変化時の消費電力

ケース	電力/mW	A53	DDR	C7x	R5F	DM_R5	Δ 電力/mW
通常のステータス	5136	1400MHz	3200MHz	1000MHz	800MHz	800MHz	0
A53 周波数低下	4837	700MHz	3200MHz	1000MHz	800MHz	800MHz	299
DDR 周波数低下	5091	1400MHz	1600MHz	1000MHz	800MHz	800MHz	45
C7x 周波数低下	4473	1400MHz	3200MHz	500MHz	800MHz	800MHz	663
R5F 周波数低下	5109	1400MHz	3200MHz	1000MHz	400MHz	800MHz	27
DM 周波数低下	5084	1400MHz	3200MHz	1000MHz	800MHz	400MHz	52
すべての周波数低下	4220	700MHz	1600MHz	500MHz	400MHz	400MHz	916

計算結果から、AM62A における C7x コアが消費電力に最も大きな影響を与えています。したがって、システムの温度制御方式を設計する際には、目標温度に到達した後、C7x コアの周波数と負荷を優先的に低減させる必要があります。

3.2 Linux 温度保護ロジック

SDK 11.0 (またはそれ以前) から、Linux SDK は Linux VTM ドライバの組み込みを開始しました。Linux カーネル VTM と SOC のハードウェア VTM は、2 つの異なる概念です。カーネル VTM フレームワークは、デバイスツリー構成(例:カーネルサーマルバインディングドキュメントで定義される k3-am62-thermal.dtsi)を用いてセンサ温度を監視し、CPU 周波数の低下、シャットダウン、Linux の再起動などの対応措置を実行します。Linux で使用するセンサ温度は、AM62x SOC ハードウェア VTM モジュールから取得されます。

カーネルデバイスツリー k3-am62a-thermal.dtsi の定義と設定は、単なる一例です。お客様は、Linux のシャットダウンや再起動など、プロジェクトの要件に応じてこれを変更できます。カーネルデバイスツリー k3-am62a-thermal.dtsi のデフォルト設定では、温度が 105°C に達するとカーネルがシャットダウンシーケンスをトリガーします。Linux SDK 11.0 では、Linux VTM ドライバもハードウェア VTM 向けに同様の割り込みスレッショルドを定義しており、最大 3 つのプログラム可能な温度スレッショルドを許可します。このうち 2 つは スレッショルド超過時用、1 つは スレッショルド未満時用であり、VTM がカーネルに動作を促すアラートを発信できるようにします。例えば、最初のスレッショルドを超過した場合、カーネルは CPU の電圧とクロック速度を低下させるよう通知され、SoC 全体の温度を安定化させることができます。SoC 温度が上昇し続ける場合、2 番目のスレッショルドを用いてより積極的な対策を実施できます。例えば、2 番目のスレッショルドを超えた時点で、電源オフコマンドを発行してデバイスを完全にシャットダウンすることが可能です。スレッショルド温度は、デバイスツリーで定義された値を使用してカーネル内で設定できます。これにより、カーネルが SoC をシャットダウンする*臨界*温度を設定できます。SoC が過熱すると、カーネルに*パッシブ*アラートを送信し、cpufreq ドライバを冷却デバイスとして登録することで MPU 周波数を低減できます。ソフトウェアの熱保護 スレッショルドはコード変更により調整可能であり、例えばシャットダウン温度を 105°C から 125°C に調整することで、ソフトウェア保護を産業用グレードから車載用グレードの温度仕様に切り替えることができます。以下のコードは、95°C を臨界温度とし、2°C のヒステリシス制御を設定しています。95°C 到達時に冷却措置を開始するパッシブ状態に移行し、105°C 到達時には即時シャットダウン(2°C のヒステリシス制御)を行います。

```
/* From arch/arm64/boot/dts/ti/k3-am62a-thermal.dtsi */
thermal_zones: thermal-zones {
     main0_thermal: main0-thermal {
          polling-delay-passive = <250>;     /* milliSeconds */
          polling-delay = <500>;          /* milliSeconds */
          thermal-sensors = <&wkup_vtm0 0>;
          trips {
               main0_alert: main0-alert {
                    temperature = <95000>;
                    hysteresis = <2000>;
                    type = "passive";
               };
               main0_crit: main0-crit {
                    temperature = <105000>;     /* milliCelsius */
                    hysteresis = <2000>;     /* milliCelsius */
                    type = "critical";
               };
          };
          cooling-maps {
               map0 {
                    trip = <&main0_alert>;
                    cooling-device =
                         <&cpu0 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>,
                         <&cpu1 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>,
                         <&cpu2 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>,
                         <&cpu3 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>;
               };
          };
     };
```

コアがパッシブ状態になると、コアは優先度の低いプロセスを強制終了することで、対応するコアの負荷を軽減できます。重要なタスクでは、動作周波数を下げることで消費電力を低減できます。動的周波数選択(DFS)は、CPU 周波数を動的に調整してパフォーマンスを最適化する効果的な手法です。詳細については、チュートリアルを参照してください。

DFS 方式では現在、A コアの周波数を容易に調整することが可能です。次のセクションでは、他のコアの周波数を変更する方法について説明します。まず、TRM (クロックセクション) から、コアに関連付けられたクロックソースと PLL と分周器を特定します。PLL に基づいて分周器を変更することで、最終的な出力クロックが速度グレードを満たすようにします。

以下に、AM62A の C7x コアクロックを変更する方法の例を示します。PLL は MAIN_PLL7 HSDIV0 として識別され、対応するデバイスの TRM に記載されています。変更内容は以下のとおりです。元の 1GHz メインクロック周波数が分割され、500MHz とします。他のコアについても同様の方法で周波数調整が可能です。変更後は Linux の k3conf コマンドで対応するクロック周波数を確認できます。お客様は、k3conf set clock $CLOCKID $FREQ コマンドを使用してクロックを変更することも可能です。デバイスによってはデバイス ID の変更が必要な場合があります。例えば、AM62A の C7x は ID 208 と 211 に対応します。

```
k3conf dump clock 208
k3conf dump clock 211
output:
|-----------------------------------------------------------------------------------------------|
| Device ID | Clock ID | Clock Name                          | Status          | Clock Frequency |
|-----------------------------------------------------------------------------------------------|
|   208     |     0    | DEV_C7X256V0_C7XV_CORE_0_C7XV_CLK | CLK_STATE_READY | 500000000   |
|-----------------------------------------------------------------------------------------------|
|   211     |     0    | DEV_C7X256V0_CLK_C7XV_CLK                       | CLK_STATE_READY |
500000000       |
|   211     |     7    | DEV_C7X256V0_CLK_PLL_CTRL_CLK                   | CLK_STATE_READY |
500000000       |
diff --git a/source/drivers/device_manager/rm_pm_hal/rm_pm_hal_src/pm/soc/am62ax/clocks.c
b/source/drivers/device_manager/rm_pm_hal/rm_pm_hal_src/pm/soc/am62ax/clocks.c
index 2122081..7438db5 100644
--- a/source/drivers/device_manager/rm_pm_hal/rm_pm_hal_src/pm/soc/am62ax/clocks.c
+++ b/source/drivers/device_manager/rm_pm_hal/rm_pm_hal_src/pm/soc/am62ax/clocks.c
@@ -2440,7 +2440,7 @@ static const struct clk_data_div_reg clk_data_hsdiv0_16fft_main_7_hsdiv0 = {
 static const struct clk_data_div_reg clk_data_hsdiv0_16fft_main_7_hsdiv0 = {
       .data_div          = {
            .n          = 128,
-           .default_div     = 2,
+           .default_div     = 4,
       },
       .reg               = 0x00680000UL + (0x1000UL * 7UL) + 0x80UL + (0x4UL * 0UL),
       .bit               = 0,
```

上記のコード変更後は、SDK 内の lib ファイルをクリーンビルドする必要があります。また、変更後のコードは DM (デバイス管理)コア上で動作するため、DM コアに対応するファームウェアのクリーンビルドも実施してください。

3.3 Linux で未使用のコアを無効化する

TI のプロセッサは、TDA4VH など、8 つの A72 コア、8 つの R5F コア、4 つの DSP C7x コアを含む、異なるコアで構成されるヘテロジニアスアーキテクチャを採用しています。一部のお客様は、8 つの A72 コアと R5F コア用に TDA4VH を選択していますが、4 つの DSP C7x コアは使用しません。TI は、特定のプロセッサ用のオープンソース SDK をスーパーセット構成で提供し、フルスペックのパフォーマンスを提供し、対応するソフトウェア変更を実行して未使用のコアを削除し、不要な電力消費を削減します。以下の例は、SDK 10.0 の TDA4VH に含まれる 4 つの未使用 DSP C7x コアを削除し、実際の動作温度を下げるソフトウェア変更を示しています。

```
diff --git a/arch/arm64/boot/dts/ti/k3-j784s4-evm.dts b/arch/arm64/boot/dts/ti/k3-j784s4-evm.dts
index de256005f..dff4c4408 100644
--- a/arch/arm64/boot/dts/ti/k3-j784s4-evm.dts
+++ b/arch/arm64/boot/dts/ti/k3-j784s4-evm.dts
@@ -1310,28 +1310,28 @@
 };
 &c71_0 {
-       status = "okay";
+       status = "disabled";
        mboxes = <&mailbox0_cluster4 &mbox_c71_0>;
        memory-region = <&c71_0_dma_memory_region>,
                        <&c71_0_memory_region>;
 };
&c71_1 {
-       status = "okay";
+       status = "disabled";
        mboxes = <&mailbox0_cluster4 &mbox_c71_1>;
        memory-region = <&c71_1_dma_memory_region>,
                        <&c71_1_memory_region>;
 };
 &c71_2 {
-       status = "okay";
+       status = "disabled";
        mboxes = <&mailbox0_cluster5 &mbox_c71_2>;
        memory-region = <&c71_2_dma_memory_region>,
                        <&c71_2_memory_region>;
 };
 &c71_3 {
-       status = "okay";
+       status = "disabled";
        mboxes = <&mailbox0_cluster5 &mbox_c71_3>;
        memory-region = <&c71_3_dma_memory_region>,
                        <&c71_3_memory_region>;
 };
```

4 まとめ

TDA4x および AM6x は、TI プロセッサの最新世代として、強力な VTM 温度管理モジュールを搭載しています。このモジュールにより、TI のヘテロジニアス・マルチコア・プロセッサは効率的かつ高性能となり、安全な温度範囲内で最大能力を発揮して動作することが可能となります。プロセッサの処理能力が高ければ高いほど、発熱量も大きくなります。したがって、基板のハードウェア設計段階全体において、お客様は部品の熱設計を考慮する必要があります。実際のシステム動作時には、システムは常に部品の指定動作温度範囲内で動作する必要があります。本文で言及する温度保護対策は異常状況に対する補助的措置であり、通常の熱管理ソリューションとは見なすべきではありません。VTM はオンチップ温度の測定のみをサポートします。システム基板上の DDR や eMMC など他の箇所の温度管理については、お客様自身が温度センサやその他のセンサを設置する必要があります。本アプリケーションノートでは、VTM モジュールの動作メカニズムと、TI SOC で現在有効化されている対応するハードウェアおよびソフトウェアの温度保護戦略について紹介します。

5 参考資料

1. テキサス インスツルメンツ、『AM62Ax Sitara™ プロセッサ』、データシート。
2. テキサス インスツルメンツ、『TDA4VM プロセッサ』、データシート。
3. テキサス インスツルメンツ、*AM62Ax Sitara プロセッサ テクニカル リファレンス マニュアル*、テクニカル リファレンス マニュアル。
4. *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*、テクニカルリファレンスマニュアル
5. テキサス インスツルメンツ、*(+) [FAQ] TDA4VM/TDA4VL/TDA4AL/TDA4VH/DRA821 :Jacinto SDK をデバイスバリエーションに対応させるにはどうすればよいですか? YXZ - プロセッサフォーラム - プロセッサ - TI E2E サポートフォーラム*、FAQ (よくある質問)。
6. テキサス インスツルメンツ、*(+) [FAQ] AM625 / AM623 / AM620-Q1 / AM62Ax / AM62D-Q1 / AM62Px / AM62L / AM64x / AM243x (ALV, ALX) カスタム ボード ハードウェア設計 – 電圧および熱マネージャ (VTM) - プロセッサ フォーラム - プロセッサ - TI E2E サポート フォーラム*、FAQ。
7. テキサス インスツルメンツ、*(+) [FAQ] TDA4VM :Linux を使用して J7 ファミリの SoC 上でオンダイ温度を読み取る方法*、FAQ (よくある質問)。
8. テキサス インスツルメンツ、*(+) AM625 :Linux サーマルシャットダウン*、フォーラム。
9. テキサス インスツルメンツ、*サーマル精度による CPU、GPU、SoC の性能向上方法*』、FAQ(よくある質問)。
10. テキサス インスツルメンツ、*(+) [FAQ] SK-AM62 :AM62A と AM62X の電力と温度を測定するにはどうすればよいですか?* 、FAQ (よくある質問)。

6 改訂履歴

Changes from Revision * (October 2025) to Revision A (May 2026)	Page
• 5℃ を 2℃ に変更	4
• 55℃ を 95℃ に変更	9
• 125℃ を 105℃ に変更	9
• 5℃ を 2℃ に変更	9
• ドキュメント全体にわたって表、図、相互参照の採番方法を更新	12

重要なお知らせと免責事項

TI 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、TI 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 TI 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 TI 製品を使用するアプリケーションの開発の目的でのみ、TI 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。TI 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、TI およびその代理人を完全に補償するものとし、TI は一切の責任を拒否します。

TI の製品は、 TI の販売条件 、 TI の総合的な品質ガイドライン 、 ti.com または TI 製品などに関連して提供される他の適用条件に従い提供されます。TI がこれらのリソースを提供することは、適用される TI 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。 TI がカスタム、またはカスタマー仕様として明示的に指定していない限り、TI の製品は標準的なカタログに掲載される汎用機器です。

お客様がいかなる追加条項または代替条項を提案する場合も、TI はそれらに異議を唱え、拒否します。

最終更新日：2025 年 10 月