ANALOG DEVICES

Blackfin® 組込みプロセッサ

シリコン・アノーマリ・リスト

ADSP-BF561

ADSP-BF561 シリコン・アノーマリについて

これらのアノーマリには、Blackfin ADSP-BF561 製品においてすでに知られているレビジョン間の相違、および ADSP-BF561 データシートとハードウェア・リファレンス・ブックで規定される機能に対する相違を示します。

シリコン・レビジョン

シリコン・レビジョン番号は"-x.x"の形式で、すべてのデバイスに表示してあります。レビジョンを識別するために、DSPID コア MMR レジスタのインプリメンテーション・フィールド・ビット<15:8>を下図のように使うことができます。

Silicon REVISION	DSPID<15:8>
0.5	0x0005
0.3	0x0003

アノーマリ・リストのレビジョン履歴

次のレビジョン履歴には、アノーマリ・リストのレビジョンとアノーマリ・リストの各レビジョンでの主要な変更を記載します。

Date	Anomaly List Revision	Data Sheet Revision	Additions and Changes
11/7/2008	Q	D	Added Anomalies: 05000412, 05000416, 05000425, 05000426, 05000428, 05000443 Revised Anomalies: 05000283, 05000315
02/08/2008	P	C	Added anomalies: 05000363, 05000403
11/15/2007	O	B	Removed Silicon Revision 0.2 Added anomalies: 05000277, 05000339, 05000343, 05000357, 05000362, 05000371 Revised anomalies: 05000323
03/28/2007	N	A	Added anomalies: 05000315, 05000320, 05000326, 05000331, 05000332, 05000333
12/07/2006	M	A	Added anomalies: 05000301, 05000302, 05000305, 05000307, 05000310, 05000312, 05000313, 05000323
08/10/2006	L	A	Added anomalies: 05000254, 05000287, 05000288
05/25/2006	K	A	Added anomalies: 05000274, 05000275, 05000276, 05000277, 05000278, 05000281, 05000283
03/10/2006	J	A	Added anomalies: 05000265, 05000267, 05000269, 05000270, 05000272

NR002863Q

アナログ・デバイセズ株式会社

本　　社／〒105-6891　東京都港区海岸 1-16-1　ニューピア竹芝サウスタワービル
電話 03（5402）8200

大阪営業所／〒532-0003　大阪府大阪市淀川区宮原 3-5-36　新大阪トラストタワー
電話 06（6350）6868

シリコン・アノーマリの一覧

次の表に、ADSP-BF561 アノーマリの一覧と各アノーマリの適用されるシリコン・レビジョンを示します。

番号	ID	説明	0.3	0.5
1	05000074	スロット 1 で dsp32shiftimm を使用しスロット 2 で P レジスタ・ストアを行うマルチ発行命令はサポートされていません	x	x
2	05000099	UART ライン・ステータス・レジスタ (UART_LSR) ビットが同時に更新されない	x	.
3	05000120	TESTSET 命令の使用が 32 ビット整列のメモリ・ロケーションに限定される	x	x
4	05000122	16 ビット・システム MMR レジスタをアクセスするとき Rx.H を使用できない	x	x
5	05000127	ある条件下で、SIGNBITS 命令が機能しません	x	x
6	05000149	IMDMA S1/D1 チャンネルがストールすることがある	x	x
7	05000156	0 フレーム同期による PPI GP 受信 (入力) モードのとき、タイマを PWM 出力モードで使用できない	x	.
8	05000166	8〜16 の PPI データ長で上位ビットがゼロにならない	x	x
9	05000167	外部フレーム同期がアクティブの間に SPORT をターンオンさせるとデータが破壊する	x	x
10	05000168	自動リフレッシュ時に SDRAM に対してパワーアップ・シーケンスを発行すると不定な動作が発生する	x	.
11	05000169	データ CPLB ページ・ミスにより、ライトスルー・データ・キャッシュ書き込みが失われる	x	.
12	05000171	ブート ROM が SICA_IWRx ウェイクアップ・レジスタを変更する	x	.
13	05000174	キャッシュ・フィル・バッファ・データが失われる	x	.
14	05000175	シーケンサ・ストールとメモリ・ストールの同時発生	x	.
15	05000176	-1 ×-1 の乗算の後に続きアキュムレータの飽和を行うとオーバーフロー・ビットがアサートされる	x	.
16	05000179	汎用 TX または RX モードで、PPI_COUNT を 0 に設定できない:	x	.
17	05000180	0 フレーム同期を持つ PPI モードで PPI_DELAY が機能しない	x	x
18	05000181	PPI をディスエーブルすると、PPI コンフィギュレーション・レジスタがリセットされる	x	.
19	05000182	内部メモリ DMA がフル・スピードで動作しない	x	x
20	05000184	外部フレーム同期を使う PPI TX モードに対するタイマ・ピンの制約	x	.
21	05000185	2 個の外部フレーム同期を使う TX モードで FS2 の前に FS1 がアサートされると PPI 送信が早期に行われる	x	.
22	05000186	PPI パッキング・イネーブル時およびデータ長 >8 ビットの場合、上位 PPI ピンが駆動される	x	.
23	05000187	停止後 IMDMA データが壊れる	x	x
24	05000188	メモリ内でのディスクリプタとバッファの配置に関する IMDMA の制約	x	.
25	05000189	投機的フェッチ・キャンセル時の偽保護例外	x	.
26	05000190	コア電圧 < 1V で PPI が機能しない	x	x
27	05000193	極性設定を変更した際のエッジ検出入力での偽 I/O ピン割り込みの発生	x	.
28	05000194	特定のモードで SPORT を再起動するとデータ破壊が発生する	x	.
29	05000198	先行するメモリ読み出しがストールした際に MMR アクセスが失敗する	x	.
30	05000199	キャリ・フィックス時にカレント DMA アドレスが不正な値を表示する	x	.
31	05000200	非アクティブ・チャンネル中のある条件で SPORT TFS と DT の駆動が正しくない	x	.
32	05000202	特定のデュアル DAG 状況で無限ストールの可能性がある	x	.
33	05000204	キャッシュモードがライトスルーの"キャッシュ・ラインをリード時にのみ割り当て"のとき読み出しデータが不正	x	.
34	05000205	特定のシーケンスで DMA エラーまたは DMA 停止が発生する	x	.
35	05000207	"停電" 状態からの回復	x	.

番号	ID	説明	0.3	0.5
36	05000208	PLL_STAT レジスタの VSTAT ステータス・ビットが機能しない	x	x
37	05000209	演算ユニット内のスピード・パスが特定の命令に影響を与える	x	.
38	05000215	UART TX 割り込みが誤ってマスクされる	x	.
39	05000219	ブート時の NMI イベントで予期しない状態が発生する	x	.
40	05000220	キャッシュされた外部メモリとキャッシュされない内蔵 L2 メモリでデータが壊れる	x	.
41	05000225	UART スタート・ビットが不正なパルス幅になる	x	.
42	05000227	スクラッチパッド・メモリ・バンクを読み出すと、不正なデータが返される	x	.
43	05000230	UART レシーバがある条件でボーレート差に対して弱くなる	x	.
44	05000231	UART STB ビットがレシーバ設定値に悪影響を与える	x	.
45	05000232	マルチチャンネル・モードで SPORT データ送信ラインが誤って駆動される	x	.
46	05000242	PLL_CTL レジスタの DF ビットがハードウェア・リセットに応答しない	x	.
47	05000244	I キャッシュがオンの場合、コントロールの変更付近で CSYNC/SSYNC/IDLE により不具合が発生する	x	.
48	05000245	条件付き分岐のシャドウ内のアクセスで偽ハードウェア・エラーが発生する	x	x
49	05000248	TESTSET 動作により他のコア上でストールが発生する	x	.
50	05000250	マルチチャンネル(TDM)モードのある条件でデータ・ワードのビット・シフトが正しくない	x	.
51	05000251	予約済み領域内の MMR アクセスに対して例外が発生されない	x	.
52	05000253	タイマの最大外部クロック速度	x	.
53	05000254	外部クロックを使ったシングルショット PWM 出力モードのパルス幅が正しくない	.	x
54	05000257	短いハードウェア・ループ時の割り込み/例外により不正な命令がフェッチされることがある	x	.
55	05000258	ICPLB データ・レジスタのビット 9 とビット 12 が異なるとき、命令キャッシュが壊れる	x	.
56	05000260	ICPLB_ステータス MMR レジスタが壊れることがある	x	.
57	05000261	DCPLB FAULT ADDR MMR レジスタが壊れる	x	.
58	05000262	データ・キャッシュへのストアが失われることがある	x	.
59	05000263	ICPLB 例外を受理するとハードウェア・ループが壊れる	x	.
60	05000264	ハードウェア・ループ内の最後から 2 番目の命令で CSYNC/SSYNC/IDLE により無限ストールが発生する	x	.
61	05000265	外部 SPORT TX と RX クロックの低速な入力エッジ・レートでノイズに対して弱くなる	x	x
62	05000266	IMDMA を使う前に IMDMA ディスティネーション IRQ ステータスを読み出す必要がある	.	x
63	05000267	ある条件下で IMDMA がデータを壊す	x	x
64	05000269	活発な I/O 動作により、内部電圧レギュレータ出力電圧 (Vddint)が上昇する	x	.
65	05000270	活発な I/O 動作により、内部電圧レギュレータ(Vddint)の出力電圧が低下する	x	.
66	05000272	あるデータ・キャッシュ・ライトスルー・モードが Vddint <= 0.9V で失敗する	x	x
67	05000274	外部同期メモリに対するデータ・キャッシュ・ライトバックが失われる	x	x
68	05000275	PPI タイミングとサンプリング情報の更新	x	x
69	05000276	非ゼロ PPI_DELAY を持つ外部フレーム同期 PPI モードに対するタイミング条件の変更	x	x
70	05000277	エッジ検出の 1SCLK サイクル後の I/O データ・レジスタへの書き込みが割り込みをクリアする	x	.
71	05000278	DMA 実行中にペリフェラルをディスエーブルすると DMA システムが不安定になる	x	.
72	05000281	ISR コンテキストが回復されないとき偽ハードウェア・エラー例外が発生する	x	.
73	05000283	特定のステージでシステム MMR 書き込みが停止されると、直ちにこれがストールする	x	x
74	05000287	ある条件下で読み出しを行うと不正なデータが返される	x	.
75	05000288	FIFO が満杯になると SPORT が不正なデータを受信する	x	.

番号	ID	説明	0.3	0.5
76	05000301	メモリーメモリ間 DMA のソース/ディステネーション・ディスクリプタは同じメモリ空間にある必要がある。	x	x
77	05000302	DMA MMR レジスタに対する書き込みの後の SSYNC が正しく処理されない	x	x
78	05000305	PLL_CTL レジスタの SPORT_HYS ビットが機能しない	x	.
79	05000307	ハイバネート中に SCKELOW ビットが状態を維持しない	x	.
80	05000310	予約済みメモリの境界でのフェッチにより、偽ハードウェア・エラーが発生する	x	x
81	05000312	SSYNC、CSYNC、または LT、LB、LC レジスタへのロードが割り込みを受けるとエラーが発生する	x	x
82	05000313	シングル・フレーム同期モードの最初の転送で PPI がレベル検出になる	x	x
83	05000315	次のシステム MMR アクセスでシステム MMR 書き込みの停止が異常完了する	x	x
84	05000320	SPI マスタ・ブートの後、PF2 出力がアサートされたままになる	.	x
85	05000323	特定の条件下で GPIO フラグ・ピンが誤動作する	x	x
86	05000326	ワード長 >16 ビットのとき SPORT セカンダリ受信チャンネルが機能しない	.	x
87	05000331	24 ビット SPI ブート・モードが機能しない	x	.
88	05000332	スレーブ SPI ブート・モードが機能しない	x	.
89	05000333	エッジ検出後の 1 SCLK サイクルでのフラグ・データ・レジスタ書き込みにより、割り込みステータスがクリアされる	x	.
90	05000339	PLL_CTL レジスタの ALT タイミング・ビットが機能しない	x	.
91	05000343	メモリ DMA FIFO により、外部メモリに対する書き込みのスループット低下が発生する	x	.
92	05000357	チャンネル 0 をディスエーブルすると、シリアル・ポート(SPORT)マルチチャンネル送信に不具合が発生する	x	x
93	05000362	列アドレス幅の競合により SDRAM エラーが発生する	x	x
94	05000363	UART ブレーク信号の問題	x	.
95	05000366	ITU-R 656 モードで PPI アンダーフロー・エラーが検出されなくなる	x	x
96	05000371	サブルーチンの継続時間が 5 サイクルを下回るとき、RETS レジスタが壊れる可能性がある	x	x
97	05000403	レベル検出の外部 GPIO ウェイクアップにより無限ストールが発生する	x	x
98	05000412	ライトバック・データ・キャッシュをイネーブルしたとき、TESTSET 命令によりデータが壊れる	x	x
99	05000416	投機的フェッチにより、不要な外部 FIFO 動作が発生する	x	x
100	05000425	特定の設定でマルチチャンネル SPORT チャンネルがずれる	x	x
101	05000426	間接ポインタ命令の投機的フェッチにより、偽ハードウェア・エラーが発生する	x	x
102	05000428	コア B からの投機的 L2 メモリ読み出しの後に L2/L3 メモリ書き込みが喪失/破壊される	.	x
103	05000443	ハードウェア・ループの終わりにある IFLUSH 命令により無限ストールが発生する	x	x

キー: x = アノーマリが存在するレビジョン

. =適用なし

シリコン・アノーマリの詳細リスト

次のリストに、ADSP-BF561 のすでに知られているすべてのシリコン・アノーマリを示します。説明、対策、適用シリコン・レビジョンを含みます。

1. 05000074—スロット 1 に dsp32shiftimm を使用しスロット 2 で P レジスタ・ストアを行うマルチ発行命令はサポートされていません:

説明:

スロット 1 に dsp32shiftimm を使用しスロット 2 で P レジスタ・ストアを行うマルチ発行命令はサポートされていません。これは例外発生の原因になります。次のタイプの命令はサポートされていません。これは、スロット 1 に dsp32shiftimm を伴う状態で、P3 レジスタがスロット 2 でストアされるためです。

```
R0 = R0 << 0x1 || [ P0 ] = P3 || NOP; //Not Supported - Exception
```

サポートされている命令の例:

```
R0 = R0 << 0x1 || [ P0 ] = R1 || NOP;
R0 = R0 << 0x1 || R1 = [ P0 ] || NOP;
R0 = R0 << 0x1 || P3 = [ P0 ] || NOP;
```

対策:

アセンブリ・プログラムで、マルチ発行命令を 2 つの命令に分離してください。VisualDSP++ランタイム・ライブラリはサポートしてない命令を使用しません。また、このアノーマリの影響を受けるデバイスとシリコン・レビジョンをターゲットとするとき、VisualDSP++ Blackfin コンパイラはサポートしてない命令を生成しません。

適用レビジョン:

0.3、0.5

2. 05000099—UART ライン・ステータス・レジスタ (UART_LSR) ビットが同時に更新されない:

説明:

ライン・ステータス・レジスタ (UART_LSR) ビットが同時に更新されません。UART_LSR レジスタを直接ポーリングすると、見落とすライン・ステータス状態があります。

対策:

ポーリング・モードを使用する場合、UART_LSR と UART_RBR の読み出し、および UART_THR の書き込みを安全に行うタイミングを調べるために SIC_ISR レジスタをポーリングする必要があります。受信データがレディになるタイミングと受信エラーの有無を調べるためのコーディング例を次に示します。ポーリング・モードに対して、UART_TX、UART_RX、UART エラーの各割り込みは SIC_IMASK の中でマスクされています(割り込みの発生なし)。

```
#define IRQ_UART_RX 0x4000
#define IRQ_UART_ERROR 0x40

  p0.l = lo(UART_GCTL);
  p0.h = hi(UART_GCTL);

  p2.l = lo(SIC_ISR);
  p2.h = hi(SIC_ISR);

  r1 = PEN | WLS(8) (z);
  w[p0+UART_LCR-UART_GCTL] = r1;

  r1 = ERBFI | ELSI (z);
  w[p0+UART_IER-UART_GCTL] = r1;

receive_polling:

  r2 = w[p2] (z);
  CC = bittst (r2, bitpos (IRQ_UART_RX));
  if !CC jump receive_polling;

data_ready:

  csync;
  r1 = w[p0+UART_LSR-UART_GCTL] (z);
  r0 = w[p0+UART_RBR-UART_GCTL] (z);

  CC = bittst (r2, bitpos (IRQ_UART_ERROR));
  if CC jump error_handler;

  [i0++] = r0;
  jump receive_polling;
```

適用レビジョン:

0.3

3. 05000120—TESTSET 命令の使用が 32 ビット整列のメモリ・ロケーションに限定される:

説明:

16 ビット整列のアドレスでは、TESTSET 命令が正しくない結果を出します。

対策:

TESTSET は、32 ビット整列のメモリ・ロケーションでのみ使用してください。32 ビット境界に変数を配置する必要がある場合は、 ALIGN(4) ディレクティブを使用してください。

VisualDSP++ ランタイム・ライブラリは、このアノーマリに対する対策を含んでいます。この対策は該当するシリコン・レビジョンに対してイネーブルされるか、またはスイッチ '-workaround testset-align' を使って手動でイネーブルすることができます。この対策は、インクルード ccblkfn.h 内で定義された testset_t タイプが 2 バイト・タイプではなく 4 バイト・タイプとして定義されていることを確認することから構成されています。

適用レビジョン:

0.3、0.5

4. 05000122—16 ビット・システム MMR レジスタをアクセスするとき Rx.H を使用できない:

説明:

16 ビット・システム MMR レジスタをアクセスする際、データ・レジスタの上位半分を使用できません。上位半分のレジスタを使用すると、不正なデータがシステム MMR レジスタに書き込まれますが、例外は発生しません。例えば(P0 が 16 ビット・システム MMR を指す場合)、次のアクセスは失敗します:

```
w[P0] = R5.H;
```

対策:

16 ビット・システム MMR レジスタをアクセスする際、他の形式の 16 ビット転送を使用してください。例えば(P0 が 16 ビット・システム MMR を指す場合):

```
w[p0] = r5.l;
r4.l = w[p0];
r3 = w[p0](z);
w[p0] = r3;
```

VisualDSP++ Blackfin コンパイラは通常、コードの生成時に問題の命令を発行しません。定数アドレスを用いて MMR ロードを行うケース(例えば*MMR_Reg = value)では、レジスタ・ハーフをスワップするパック命令を挿入します。しかし、コンパイル時に未知のポインタ(例えば関数に対するパラメータ)を MMR に対するポインタとして識別することはできません。VisualDSP++ランタイム・ライブラリもこのアノーマリを回避します。

適用レビジョン:

0.3、0.5

5. 05000127—ある条件下で、SIGNBITS 命令が機能しません:

説明:

SIGNBITS 命令が 32 ビット値で動作すると、次のシーケンスのように、SIGNBITS の前にある命令が SIGNBITS のオペランドを生成する場合、正しく動作しないことがあります。

```
r0 = ashift r2 by r3.l;
r1.l = signbits r0;
```

対策:

この問題の回避には次の 2 つの対策があります:

1) SIGNBITS 命令に nop を配置します:

```
r0 = ashift r2 by r3.l;
nop;
r1.l = signbits r0;
```

2) SIGNBITS のオペランド・レジスタが前の命令に依存していないことを確認します:

```
r0 = ashift r2 by r3.l;
// **** another useful instruction that is not updating r0 ****;
r1.l = signbits r0;
```

VisualDSP++Blackfin コンパイラには、このハードウェア・アノーマリに対する対策が含まれています。コンパイラは、該当するシリコン・レビジョンとデバイス番号に対してこの対策を自動的にイネーブルします。あるいは、コンパイラ・フラグ'-workaround signbits'を指定することにより、対策を手動でイネーブルすることができます。

対策をイネーブルすると、__WORKAROUND_SIGNBITS が、コンパイル、アセンブル、リンク・ステージで定義されます。

適用レビジョン:

0.3、0.5

6. 05000149—IMDMA S1/D1 チャンネルがストールすることがある:

説明:

内部メモリ DMA (IMDMA) エンジンは、S0/D0 (高優先順位) および S1/D1 (低優先順位)と呼ばれる 2 対のメモリ DMA チャンネルを持っています。低優先順位のディステネーション・チャンネル D1 は、D0 によりオフにされることがあります。D0 がデータの到着を待っているが、例えば S0 がエラー状態にあるか またはイネーブルされていないために、受信できないときに問題が発生します。これらのケースでは、D0 はデータを得ることはなく、D1 も無限にストールします。

対策:

S0/D0 が常にエラーなしで完了することを確認してください (例えば、ステータスを頻繁にチェックします)。エラーが発生する場合は、S0 と D0 をディスエーブルしてください。

適用レビジョン:

0.3、0.5

7. 05000156— 0 フレーム同期による PPI GP 受信 (入力) モードのとき、タイマを PWM 出力モードで使用できない:

説明:

PPI0 が 0 フレーム同期による汎用受信 (入力) モードのとき、タイマ 8 を PWM 出力モードで使用できません。PPI1 が 0 フレーム同期による汎用受信 (入力) モードのとき、タイマ 10 を PWM 出力モードで使用できません。

対策:

PWM 出力信号には、他のタイマ (0〜7、9、11)または未使用の PPI インターフェースに接続されているタイマを使用することができます。

適用レビジョン:

0.3

8. 05000166—8〜16 の PPI データ長で上位ビットがゼロにならない:

説明:

PPI データ長が 8〜16 の場合、メモリに受信された PPI データの上位ビット(PPI データの一部ではない)はゼロである必要があります。例えば、10 ビットの PPI データ長を使用する場合、メモリ内の上位 6 ビットはゼロである必要があります。しかし、PPI は上位 6 PPI データ・ピン(PFx ピンとして共用)にあるデータは何でもキャプチャします。

対策:

ソフトウェアでの対策としては、受信データを処理する際に上位 6 ビットをマスクして無視します。

適用レビジョン:

0.3、0.5

9. 05000167—外部フレーム同期がアクティブの間に SPORT をターンオンさせるとデータが破壊する:

説明:

SPORT は、外部フレーム同期信号をレベル検出します。SPORT を外部フレーム同期に設定し、かつ SPORT を最初にイネーブルしたときフレーム同期がアクティブの場合、イネーブルされると直ちに SPORT はデータの受信を開始します。これがフレームの中程で発生して、不正なデータが受信されてしまいます。

このアノーマリはステレオ・シリアル・モード (I2S およびその派生)にも適用されます。ただし、LRFS ビットまたは RRFST ビット(いずれか一方のみ)がセットされている場合は除きます。

対策:

SPORT が完全にイネーブルされるまで、外部フレーム同期を遅延させてください。SPORT がイネーブルされるまで遅延させることができない外部フレーム同期を持つシリアル・デバイスを使用している場合は、プログラマブル・フラグ・ピンをフレーム同期に接続することができます。SPORT フレーム同期を連続的にサンプルするように PFx ピンを設定して、RFS / TFS 信号が非アクティブ状態になったときに SPORT をイネーブルします。

ステレオ・シリアル・モードの場合、可能なとき LRFS ビットまたは RRFST ビット (いずれか一方のみ)をセットしてください。

適用レビジョン:

0.3、0.5

10. 05000168—自動リフレッシュ時に SDRAM に対してパワーアップ・シーケンスを発行すると不定な動作が発生する

説明:

最初に SDRAM がパワーアップし自動リフレッシュ・モードになると、後続のパワーアップに無関係に自動リフレッシュが実行されます。このために、パワーアップ(SDRAM モード・レジスタの再設定)と同時に SDRAM デバイスに対して自動リフレッシュ・コマンドが送信されることがあります。この時点から後に不定な動作が発生することがあります。

注: 大部分のアプリケーションでは、SDRAM のパワーアップ・シーケンスと MODE レジスタの書き込みは一度で済みます。パワーアップ・シーケンスが完了した後、SDRAM モード・レジスタの変更が必要ないかぎり、PSSE ビット (SDRAM パワーアップ・シーケンス・イネーブル)を再度設定しないでください。

対策:

SDRAM モード・レジスタを再設定する必要がある場合は (後続パワーアップ)、最初に SDRAM をセルフ・リフレッシュ・モードに設定する必要があります。

一般的な手順

1) SSYNC 命令を発行して、すべての待ち状態のメモリ動作を完了させます。

2) EBIU_SDGCTL 内の SRFS (SDRAM セルフ・リフレッシュ・イネーブル)をセットして、SDRAM をセルフ・リフレッシュ・モードにします。

3) SSYNC 命令を発行して、セルフ・リフレッシュ・ビットへの書き込みを完了させます。

4) EBIU_SDRRC と EBIU_SDBCTL を必要に応じて再設定します。

5) EBIU_SDGCTL 内で、PSSE (SDRAM パワーアップ・シーケンス・イネーブル) をセットし、SRFS (SDRAM セルフ・リフレッシュ・ディスエーブル)をクリアします。

6) SSYNC 命令を発行します。

適用レビジョン:

0.3

11. 05000169—データ CPLB ページ・ミスにより、ライトスルー・データ・キャッシュ書き込みが失われる:

説明:

データ・キャッシュがイネーブルされ、かつライトスルー (ライトバックではなく) モードでキャッシュ・ラインへの書き込み時に CPLB ミスが発生すると、ラインは誤って"dirty"とマークされます。ライトスルー・モードではデータが直ちにメモリへ書き込まれるため、これが発生することはありません。

この誤ったマークにより、次の場合にミス書き込みが発生します。

1) 誤ってマークされたラインは、ビクティム処理され、さらに、

2) ビクティム処理されたラインが L2 メモリへ送られる前に同じラインに対する書き込みが発生し、さらに、

3) 1)でビクティム処理されるラインの前に 2)での書き込みが L2 へ送られます。 これは、転送のスケジューリング優先順位のために発生します。

対策:

1)現在キャッシュ中のライトスルー・キャッシュ可能な L2 ロケーションに対する書き込み時に、データ CPLB ページ・ミスを回避するか、
または、

2)ライトスルーとして設定されたデータ CPLB ページのスワップ・アウト前に、CPLB ディスクリプタを失っている、ライトスルー・キャッシュされたすべてのラインに対して FLUSHINV (最終 SSYNC を使用)を実行します。

この問題は、キャッシュ・システムのライトバック・モードでは発生しません。

適用レビジョン:

0.3

12. 05000171—ブート ROM が SICA_IWRx ウェイクアップ・レジスタを変更する

説明:

パワーアップ/リセットでブート ROM が実行され、SICA_IWR レジスタ値がデフォルトのリセット値 0xFFFF から変更されても、これらが回復されません。

対策:

ユーザ・コードは、これらのレジスタのデフォルト値に依存しないでください。必要とする値を明確に設定してください。

適用レビジョン:

0.3

13. 05000174—キャッシュ・フィル・バッファ・データが失われる

説明:

データ・キャッシュをイネーブルした状態で、フィル・バッファで書き込み用にスケジュールされたデータが、イベント・シーケンスが次の通りに発生した場合(極めて希ですが)に書き込まれません。

1) デュアル DAG 命令 (同じ命令内で 2 回のメモリ転送)は、キャッシュ・ミスなしで、メモリ内の同じサブバンクをアクセスする必要があります (衝突)。

2) 直後の命令は、キャッシュ・ラインにヒットする書き込みである必要があります。ただし、このキャッシュ・ラインはフィルされるラインであり、かつこのフィル動作は同じサイクル内で完了しつつある動作である必要があります。

3) 新しいフィル動作が起動され、かつ同じフィル・バッファを使います。

対策:

このイベント・シーケンスそのものを回避してください。

対策には次が含まれますが、これらに限られるものではありません。

1) データを別々のサブバンクに配置して衝突を避けるか、
または、

2) nop またはその他の非書き込み命令を上記問題の 2 つの命令の間に挿入するか、
または、

3) 上記 2 つ目の命令として説明されたシングル DAG 書き込みを、同じアドレスからのダミー読み出しを含むデュアル DAG 命令へ変換してください (この場合はもちろん、次の命令が同じシーケンスを発生しないことを確認してください)。

適用レビジョン:

0.3

14. 05000175—シーケンサ・ストールとメモリ・ストールの同時発生

説明:

シーケンサ・ストール (レジスタの依存性により発生する演算ストールや特定の MMR レジスタ・アクセスによるストールなど)がメモリ・ストール(メモリの同じサブバンクに対するデュアル DAG の 衝突、キャッシュ・ミスなど)と同時に発生すると、後続トランザクションが希に喪失してしまう可能性があります。各ストールが同時に発生する必要があります。これは、ストールの開始サイクルと継続時間が一致することを意味します。

対策:

この問題に対する最も容易な対策は、演算ストールが生じないようにプログラムし、かつ MMR アクセスの前後を CSYNC 命令で取り囲むことにより、プロセッサ側のストールを制限することです。

内蔵のパフォーマンス・モニタを使って次の方法によりストールを解析することができます。

次のコードを使うと、オーバーラップしているコア・ストール数とメモリ・フリーズ・サイクル数をカウントすることができます。これは、必ずしも継続時間が異なるために問題が発生することを示すものではありません。例えば、演算ストールは一般に 1 サイクルまたは 2 サイクル続き、一方キャッシュ・ミスにより発生するメモリ・フリーズは多サイクル継続しますが、これにより問題は発生しませんが、カウンタはインクリメントします。

```
// Set up performance counters to monitor overlapping stalls
r0.h=0x0091;
r0.l=0xd21b;
p0.h=PFCTL;
p0.l=PFCTL;
[p0]=r0;            // wake up first
[p0]=r0;            // then enable

p0.h=PFCNTR0;
p0.l=PFCNTR0;
r0=0; [p0++]=r0;    // clear both counters
[p0]=r0;
```

この後、2 番目のパフォーマンス・カウンタ (PFCNTR1)がオーバーラップ・ストールのサイクル数をカウントします。

既存コード内でストールを見つけ出す 1 つの方法は、2 番目のカウンタを 0xffffffff に初期化することです。

```
r0.h=0xffff;
r0.l=0xffff;
p0.h=PFCNTR1;
p0.l=PFCNTR1;
[p0]=r0;
```

カウンタがオーバーフローすると、ハードウェア・エラー状態が発生して、これが割り込み 5 になります。割り込み 5 ルーチン内には、rti 命令だけを構成しブレーク・ポイントを配置します。 割り込みを受けたルーチンに戻ると、ストールは上記リターン・ポイントの数ライン上のコードにあるはずです。nop 命令を挿入して、演算ストールを解消させ、CSYNC を挿入して MMR ストールを回避させます。この手順をパフォーマンス・カウンタがゼロを維持するまで繰り返します。

適用レビジョン:

0.3

15. 05000176—-1 ×-1 の乗算の後に続きアキュムレータの飽和を行うとオーバーフロー・ビットがアサートされる:

説明:

1.15 フォーマットの -1×-1 の乗算の後に 32 ビット境界のアキュムレータ飽和命令が続くと、飽和命令によりオーバーフロー・ビットが誤ってセットされることがありますが、正しいアキュムレータ演算結果が得られます。

例えば、次の命令シーケンスでこの動作が発生します。

```
r2 =0;
a0 =0;              // clear accumulator
r0= 0x8000 (z);     // -1 in 1.15 format
r1=r0;

a0 -= r0.l * r1.l; // multiply -1 by -1 and subtract from accumulator
astat = r2;        // clear astat
a0 = a0 ( s );     // Will incorrectly set the overflow bit

a0 = 0;            // second example
a1 = 0;

a0 -= r0.l * r1.l ( w32 );     // same multiplication as above
astat = r2;                    // clear astat
a0 -= a1 ( w32 );              // subtract 0 from accumulator
                               // (will incorrectly overflow)
```

対策:

このような命令シーケンスの後のオーバーフロー・ビットを無視するか、または-1×-1 の乗算の後ろにダミーのアキュムレート命令を追加します。例えば、R3 の値がゼロで、初期条件は例の場合と同じとすると、

```
a0 -= r0.l * r1.l ;            // multiply -1 by -1 and subtract from accumulator
a0 += r3.l * r3.l;             // dummy add zero to previous result
astat = r2;                    // clear astat
a0 = a0 ( s );                 // Will now correctly clear the overflow bit
```

VisualDSP++ コンパイラはアキュムレータ飽和命令を生成しません。また、VisualDSP++ ランタイム・ライブラリはアキュムレータ飽和命令を使用しません。

適用レビジョン:

0.3

16. 05000179—汎用 TX または RX モードで、PPI_COUNT を 0 に設定できない:

説明:

汎用モードでは、PPI は少なくとも 2 ワードのブロックを受信または送信する必要があります。シングル・ワード転送 (PPI_COUNT 値 = 0) は機能しません。

対策:

なし

適用レビジョン:

0.3

17. 05000180— 0 フレーム同期を持つ PPI モードで PPI_DELAY が機能しない:

説明:

セルフ・トリガーの PPI の連続サンプリング動作では、PPI_DELAY レジスタで指定される遅延カウントが無視されます。このモードがイネーブルされると直ちに、データが転送されます。

対策:

遅延が必要な場合は、ソフトウェアで受信データを無視するか、あるいは少なくとも 1 フレーム同期を使うモードを使用してください。

適用レビジョン:

0.3、0.5

18. 05000181—PPI をディスエーブルすると、PPI コンフィギュレーション・レジスタがリセットされる

説明:

PPI をディスエーブルすると、すべての PPI MMR がリセット値に戻ります。

対策:

PPI をディスエーブルした後に再イネーブルするごとに、すべての関係する MMR を再設定してください。

適用レビジョン:

0.3

19. 05000182—内部メモリ DMA がフル・スピードで動作しない

説明:

内部メモリ DMA コントローラが、500MHz のコア・クロック周波数までしか動作しません。この規定値を超えるクロック周波数では予期しない動作が発生します。

対策:

IMDMA を使用する場合は、コア・クロック周波数を最大 500MHz に設定してください。

適用レビジョン:

0.3、0.5

20. 05000184—外部フレーム同期を使う PPI TX モードに対するタイマ・ピンの制約

説明:

外部フレーム同期を使う送信モードで、フレーム同期に接続されるタイマ・ピン (PPI0 の場合タイマ 8 および/または 9 、PPI1 の場合タイマ 10 および/または 11)は、入力に設定する必要があります。各タイマはイネーブルする必要があり、汎用としては使用できません。

対策:

なし

適用レビジョン:

0.3

21. 05000185—2 個の外部フレーム同期を使う TX モードで FS2 の前に FS1 がアサートされると PPI 送信が早期に開始される

説明:
FS1 が FS2 の前にアサートされると、データ送信の前に PPI は FS2 のアサーションを待ちません。

対策:
可能な場合は、1 個の外部フレーム同期モードまたは 2 個の内部発生フレーム同期を使ってください。

適用レビジョン:
0.3

22. 05000186—PPI パッキング・イネーブル時およびデータ長 >8 ビットの場合、上位 PPI ピンが駆動される

説明:
PPI データのパッキングをイネーブルし、かつ PPI ポートのデータ長 (DLEN)を 10 ビット以上の値に設定すると、8 を超える上位ビット (およびピン) が駆動されます (0 または 1 に駆動)。

対策:
パッキングは 8 ビット・データでのみ意味があるため、8 ビットの正しいデータ長を設定することにより、これを容易に回避することができます。

適用レビジョン:
0.3

23. 05000187—停止後 IMDMA データが壊れる

説明:
IMDMA 転送時、ソフトウェアが対応するソース・チャンネルの前にディステネーション・チャンネルを停止させると、後続の転送で不正なデータまたは前の転送からのデータを受け取ります。
停止の順序を変えた場合、またはエラーにより DMA チャンネルが停止する場合には、この問題は発生しません。

対策:
IMDMA 転送をディスエーブルする場合、ソース・チャンネルを先にディスエーブルしてください。

適用レビジョン:
0.3、0.5

24. 05000188—メモリ内でのディスクリプタとバッファの配置に関する IMDMA の制約

説明:
ディスクリプタ・モードでは、IMDMA が予期しない動作を行うケースが幾つかあります。
1) 高速 (L1) メモリ内のデータ・バッファ、低速 (外部) メモリ内のディスクリプタ
2) L1 メモリ内のデータ・バッファとディスクリプタ、ただしバンクは異なります。
3)異なる L1 バンク内のデータに対して動作する 2 個の IMDMA チャンネル

対策:
上記ケースは一般に次の配置に注意することにより回避できます。
1) ディスクリプタを L1 (データ・バッファと同じバンク) または L2 メモリに配置します。
2) データ・バッファとディスクリプタを L1 の同じバンクに配置します。
3) すべてのチャンネルのデータ・バッファとディスクリプタを L1 の同じバンクに配置します。

適用レビジョン:
0.3

25. 05000189—投機的フェッチ・キャンセル時の偽保護例外:

説明:

投機的フェッチのキャンセルにより偽コードまたは偽データが発生した場合、これらの保護例外が発生します。例えば、有効ページの最後のワードにあるジャンプ命令または rts 命令が別の有効ページへ分岐し、かつ無効または存在しないメモリ・ロケーションから投機的命令フェッチが行われている場合に、例外が発生します。同様に、ポスト・インクリメントの間接データ・メモリ・アクセスで、保護された、または存在しないメモリ・ロケーション(例えば、R0 = [P0++]、ここで P0 はページの最終ワードを指します)に対して投機的アクセスが行われた場合も、不正な例外が発生します。

対策:

1)ページ境界に分岐命令またはデータを配置しないでください。境界の前では、予約済みメモリ空間を使い、少なくとも 76 バイトを空けてください。これにより偽例外の発生を防止することができます。

2)例外の有効/無効をアクションの前に判断する例外処理を設けてください。これは、CODE_FAULT_ADDR (または DATA_FAULT_ADDR)レジスタ値が有効ページ内のアドレスであることを確認することにより行うことができます。この場合、アクションは不要です。

デフォルトの VisualDSP++ LDF には、このハードウェア・アノーマリに対する対策が含まれています。該当するシリコン・レビジョンでは対策が自動的にイネーブルされます。あるいは、リンク時にマクロ __WORKAROUND_AVOID_LDF_BLOCK_BOUNDARIES を定義して、対策を手動でイネーブルすることができます。イネーブルされると、LDF は有効なメモリ・ブロックの境界に 76 バイトを予約します。

適用レビジョン:

0.3

26. 05000190—コア電圧 < 1V で PPI が機能しない

説明:

コア電圧を 1 V より低く設定すると、PPI が機能しなくなります。

対策:

コア電圧レベルを 1 V 以上に設定してください。

適用レビジョン:

0.3、0.5

27. 05000193—極性設定を変更した際のエッジ検出入力での偽 I/O ピン割り込みの発生:

説明:

次のシナリオを考えてみます:

1) ピンをエッジ検出入力に設定します。
2) 立ち上がりエッジで割り込みが発生します。
3) 入力レベルは一定で 0 です。
4) 極性設定を変更して、立ち下がりエッジで割り込みが発生するようにします。

この場合、入力にエッジが実際に存在しなくとも誤割り込みが発生します。これは、後続のすべての書き込みで極性レジスタのこのビットに 1 を書き込むときにも発生します。極性レジスタが 0 にリセットされると、期待通りに割り込みは発生しません。

外部ピン・レベルが 1 である逆の場合には、極性ビットが(0 から 1 へではなく)1 から 0 へ変わったとき(さらに極性レジスタのこのビットへ後続書き込みで 1 を書き込むとき)、誤割り込みが発生します。

複数の I/O ピンがエッジ検出割り込みに設定されている場合は、極性レジスタに対する任意の変更により、これらすべての I/O ピンが上記の影響を受けます。この場合の対策は、これらすべてのピンに適用する必要があります。

同様の考慮は、入力イネーブル・レジスタにも適用されます。エッジ検出割り込みのイネーブル中にこの設定を変更したときにも、不要な割り込みが発生する原因になります。

対策:

極性 (および/または入力イネーブル) レジスタを変更する前に、割り込みをディスエーブルし (PFA ビットまたは PFB IMASK ビットをクリア)、レジスタ設定を変更し、通常通りに割り込み要求をクリアし (データを書き込むかレジスタをクリア)、その後割り込みを再度イネーブルしてください。

適用レビジョン:

0.3

28. 05000194—特定のモードで SPORT を再起動するとデータ破壊が発生する:

説明:

立ち下がりエッジを選択して内部 SPORT クロックまたは外部 SPORT クロックを使用する場合 (SPORTx_TCR1:TCKE、SPORTx_RCR1:RCKE)、SPORT をディスエーブルした後に再イネーブルすると、最初に送信または受信されたワードが壊れる可能性があります。

対策:

内部 SPORT クロックの場合: SPORT をディスエーブルした後に SPORT コンフィギュレーション 1 レジスタ (SPORTx_TCR1:ITCLK、SPORTx_RCR1:IRCLK)のビット 1 に 0 を書き込みます。これにより、SPORT クロックが外部に切り替えられて、SPORT を完全にリセットできるようになります。

立ち下がりエッジを選択した外部 SPORT クロックの場合: SPORT をディスエーブルした後に、SPORT コンフィギュレーション 1 レジスタ (SPORTx_TCR1:TCKE、SPORTx_RCR1:RCKE)のビット 14 に 0 を書き込みます。これにより、SPORT が完全にリセットできるようになります。

適用レビジョン:

0.3

29. 05000198—先行するメモリ読み出しがストールした際に MMR アクセスが失敗する:

説明:

メモリ読み出しのストール後に MMR 読み出しが続くと、MMR 読み出しが失敗します (不正なデータが読み出されます)。同様に、メモリ読み出しのストール後に MMR 書き込みが続くと、書き込みが実行されません (失われます)。メモリ読み出しを含む命令は、

```
reg = [ Preg ], etc.
reg = [ Ireg ], etc.
stack pop, stack pop multiple
UNLINK
TESTSET
PREFETCH
```

および並行発行された上記すべて。

メモリ書き込みを含む命令は、

```
[ Preg ] = reg, etc.
[ Ireg ] = reg, etc.
stack push, stack push multiple
LINK
```

および並行発行された上記すべて。

対策:

MMR アクセスの前に NOP (または任意の非メモリ・アクセス)を配置すると、この問題が防止されます。

VisualDSP++Blackfin コンパイラには、このハードウェア・アノーマリに対する対策が含まれています。コンパイラは、該当するシリコン・レビジョンとデバイス番号に対してこの対策を自動的にイネーブルします。あるいは、コンパイラ・フラグ'-workaround sdram-mmr-read' を指定することにより、対策を手動でイネーブルすることができます。

イネーブルされると、コンパイラは、ロードと MMR ロードとの間に NOP 命令を挿入します。コンパイラは、定数アドレスで MMR ロードを行う場合(例えば*MMR_ADDR = value)は NOP 命令を挿入しますが、コンパイル時に未知のポインタ (例えば関数へのパラメータ)を MMR へのポインタとして識別できません。

対策をイネーブルすると、__WORKAROUND_SDRAM_MMR_READ が、コンパイル、アセンブル、リンク・ステージで定義されます。

適用レビジョン:

0.3

30. 05000199—キャリ・フィックス時にカレント DMA アドレスが不正な値を表示する:

説明:

キャリ・フィックス・サイクルでアクティブ・チャンネルの DMA カレント・アドレス・レジスタ (DMAx_CURR_ADDR) が読み出されると、レジスタの上位半分が 1 だけずれます。LSB は新しい値で更新されますが、MSB は前の値を維持します。レジスタを 2 回目に読み出すと、正しい値が返されます。

アドレスが 64k 境界を超えるように DMA アドレスが変更されると、キャリ・フィックス・サイクルが発生します。DMA アドレスが 64k アドレス境界を跨ぐことがない場合、読み出しは正しく行われます。

対策:

1) 64K アドレス境界を超える DMA アドレスは回避してください。
または
2) DMA カレント・アドレス・レジスタを 2 回読み出して、読み出した値を確認してください。

適用レビジョン:

0.3

31. 05000200—非アクティブ・チャンネル中のある条件で SPORT TFS と DT の駆動が正しくない:

説明:

マルチチャンネル・モードでは、SPORT の MRCS レジスタを使って、アクティブにするチャンネル(送信または受信するチャンネル) と無視するチャンネルを選択します。各無視するチャンネルでは、DT 出力がスリー・ステートになります。"マルチチャンネル DMA パッキング" がディスエーブルされたとき、問題が発生します。このモードでは、非アクティブ・チャンネルの場合、データ・ワードの上位ビット (MSB)が DT ラインに出力され、これに対応して TFS が 1 ビットの継続時間ハイ・レベルに駆動されて有効データを表示します。

対策:

可能な対策は、 "マルチチャンネル DMA パッキング"をイネーブルすることです。このモードでは、アクティブ・チャンネルだけでメモリとの間で DMA が実行され、非アクティブ・チャンネルは実質的に無効にされますが、このモードでは、非パックド DMA モードのようにアクティブ/非アクティブ・チャンネルをダイナミックに変更することはできません。したがって、これはすべてのアプリケーションで可能ではありません。

適用レビジョン:

0.3

32. 05000202—特定のデュアル DAG 状況で無限ストールの可能性がある:

説明:

この問題が発生するためには、プロセッサが非キャッシャブルまたはライトスルー・キャッシャブル L2 または書き込まれたばかりの外部メモリ・アドレスに対するデータ・メモリ読み出しを実行する必要があります。"最近の書き込み" は再度読み出されたとき、現に書き込みバッファに保持されている必要があります。 この読み出しは、この"最近の書き込み" が書き込みバッファからディステネーション・メモリ・ロケーションへ排出された同じクロック・サイクルで実行される必要があります。

"最近の書き込み"の読み出しの直前に、デュアル DAG アクセスを実行する必要があります。デュアル DAG アクセスは互いに衝突する必要がありますが、"最近の書き込み" アドレスと何らかの関係があると考えられます。

"最近の書き込み"の読み出しの直後に、読み出し、書き込み、またはプリフェッチを行わない必要があります (その他の場合には、このアノーマリは発生しません)。

注: デュアル DAG の衝突は、両 DAG が L1 メモリまたは L1 キャッシュ内の同じサブバンクをアクセスするときに発生します。

対策:

1) デュアル DAG アクセスの衝突の直後に、最近書き込まれた L2 アドレスを読み出さないでください。
または

2) 予防的なデュアル DAG/L2 読み出し命令を SSYNC の前に配置して、前の書き込みが書き込みバッファ内に存在しなくなるようにしてください。
または

3) すべての L2 をライトバック・キャッシャブルに設定してください (書き込みバッファの使用を回避)。

VisualDSP++Blackfin コンパイラには、このハードウェア・アノーマリに対する対策が含まれています。コンパイラは、該当するシリコン・レビジョンとデバイス番号に対してこの対策を自動的にイネーブルします。あるいは、コンパイラ・フラグ '-workaround infinite-stall-202'を指定することにより、対策を手動でイネーブルすることができます。イネーブルされると、コンパイラは、PREFETCH[SP]命令を挿入してアノーマリ状態を回避します。

対策をイネーブルすると、_WORKAROUND_INFINITE_STALL_202 が、コンパイル、アセンブル、リンク・ステージで定義されます。

適用レビジョン:

0.3

33. 05000204—キャッシュ・モードがライトスルーの"キャッシュ・ラインをリード時のみ割り当て"のとき読み出しデータが不正:

説明:

ライトスルー・キャッシュがイネーブルされ、かつ DCPLB_DATAx:CPLB_L1_AOW = 0 (キャッシュ・ラインをリード時のみのみ割り当て)のとき、次のシナリオで不正なデータが読み出されることがあります。

- no-allocate-on-write モードでライトスルー・キャッシャブルであるアドレスに対して書き込むと、キャッシュ・ミスが発生する。
- 書き込みがまだストア・バッファ内である間に (書き込みバッファまたはディステネーション・メモリ・ロケーションになる前)に上記アドレスが読み出された。
- その後、ストア・バッファから取り出した後に再度上記アドレスを読み出した。キャッシュから返されたデータは正しくないが、ディステネーション・メモリ・ロケーションの値は正しい。

対策:

データ・キャッシュをライトスルーに設定するとき、DCPLB_DATAx:CPLB_L1_AOW = 1 (キャッシュ・ラインを読み出しと書き込み時に割り当て) を設定して、このアノーマリの発生を防止してください。

適用レビジョン:

0.3

34. 05000205—特定のシーケンスで DMA エラーまたは DMA 停止が発生する

説明:

1 つの L1 メモリ・バンク (バンク A またはバンク B)からの 3 回以上の連続 DMA 読み出しが長時間ストールすると (コアまたはキャッシュ動作に起因)、問題が発生します。読み出しを要求している DMA コントローラまたは発生する順序に無関係です。この状況では、後続の L1 DMA 読み出しが別のバンクから開始されると、直前の 2 回の読み出しのデータが衝突して、データが壊れる可能性があります。

対策:

同じ L1 メモリ・バンク内にすべての DMA データを配置します (または可能な場合 DMA データを両コア間に分割)。

適用レビジョン:

0.3

35. 05000207—"停電" 状態からの回復:

説明:

"停電"が発生すると、ハードウェア・リセット・ピンを使って内部電圧レギュレータをリセットすることができません。"停電"とは、VDDext がデータシートで規定された範囲を下回るが、正しい値に戻る前に 0 V まで低下しない状態と定義されます。

対策:

"停電"から回復するためには、プロセッサが完全にパワーダウンして、電源が戻る必要があります。

適用レビジョン:

0.3

36. 05000208—PLL_STAT レジスタの VSTAT ステータス・ビットが機能しない:

説明:

PLL_STAT レジスタの VSTAT ステータス・ビットが機能しません。内部電圧レギュレータが安定したか否かを調べるとき、この値に依存することは推奨されません。

対策:

内部電圧レギュレータを介して電圧を変えるときは、電圧が変化するために少なくとも 40 μ sec 待ってください。40 μ sec 後に、VSTAT ビットの状態に関係なく新しい値が設定されます。

適用レビジョン:

0.3、0.5

37. 05000209—演算ユニット内のスピード・パスが特定の命令に影響を与える:

説明:

次の命令では、前の命令がその命令のオペランドを生成するとき正しくない動作が発生することがあります。影響を受ける命令は:

```
EXTRACT (x)
DEPOSIT (x)
SIGNBITS
EXPADJ
```

Signbits 命令の例を示します:

```
r0 = ashift r2 by r3.l;
r1.l = signbits r0;
```

対策:

この問題の回避には次の 2 つの対策があります:

1) signbits 命令の前に nop を配置します:

```
r0 = ashift r2 by r3.l;
nop;
r1.l = signbits r0;
```

2) signbits のオペランド・レジスタが前の命令に依存していないことを確認します:

```
r0 = ashift r2 by r3.l;
// ** another useful instruction that is not updating r0 **;
r1.l = signbits r0;
```

VisualDSP++Blackfin コンパイラには、このハードウェア・アノーマリに対する対策が含まれています。コンパイラは、該当するシリコン・レビジョンとデバイス番号に対してこの対策を自動的にイネーブルします。あるいは、コンパイラ・フラグ'-workaround dreg-comp-latency' を指定することにより、対策を手動でイネーブルすることができます。VisualDSP++ ランタイム・ライブラリも、必要に応じてこのアノーマリ状態を回避します。イネーブルされると、コンパイラは 2 つの命令の間に NOP 命令を挿入します。この最初の命令では値を DREG に割り当て、2 つ目の命令では DREG を SIGNBITS、EXTRACT、DEPOSIT または EXPADJ の各命令に対するパラメータとして使います。

この対策がイネーブルされると、コンパイラもマクロ _WORKAROUND_DREG_COMP_LATENCY と __WORKAROUNDS_ENABLED をソース、アセンブリ、リンク・ビルド・ステージで定義します。

適用レビジョン:

0.3

38. 05000215—UART TX 割り込みが誤ってマスクされる:

説明:

UART TX 割り込みで、IIR レジスタを読み出し、かつ THR レジスタを書き込まないと (TX 割り込みをクリアするため)、UART TX 割り込みがマスクされます。ストリングの終わりに到達した場合に、ISR 内でこれが発生することがあります。この場合、ETBEI ビットをイネーブル/ディスエーブルしても、割り込みイネーブルの状態は影響を受けませんが、これを行う必要があります。

対策:

UART TX 割り込み内で ETBEI ビットをクリアしてストリングを終わらせて、RTI を実行してください。割り込みを再イネーブルするときは、ETBEI ビットをセットするだけで済みます。これにより、THR レジスタがエンプティのとき、プロセッサは UART TX 割り込みサービス・ルーチンへ直接行きます。

適用レビジョン:

0.3

39. 05000219—ブート時の NMI イベントで予期しない状態が発生する:

説明:

ブート時 NMI ピンがアサートされると、ブート ROM 内にハンドラがないためブート・プロセスが失敗します。動作は予測できません。

対策:

ブート・シーケンス時に NMI ピンをアサートしないでください。

適用レビジョン:

0.3

40. 05000220—キャッシュされた外部メモリとキャッシュされない内蔵 L2 メモリでデータが壊れる

説明:

この問題は、外部メモリがライトバック・キャッシュ・モードにあり、内蔵 L2 メモリがキャッシュされない場合に発生します。このシナリオで、この問題が発生するためには次のイベント・シーケンスが発生する必要があります。

1) キャッシュが外部メモリへライトバックされますが、書き込みが遅延させられます (SDRAM 行の変更または DMA または他のコアからのアクセスのような外部メモリ・インターフェース上でのその他の動作のために)。

2) 内蔵 L2 メモリ(キャッシュなし)に対する後続の書き込みアクセスが試みられた。

この場合、L2 書き込みと外部メモリのデータが壊されます。

逆に、L2 メモリがライトバック・キャッシュ・モードにあり、外部メモリがキャッシュされない場合にも、頻度は小さいですがこの問題が発生します。L2 メモリに対するアクセスが高速の場合、ライトバックをストールさせるのはさらに困難ですが、この状況が存在する場合、対策が必要です。

対策:

この問題は、上記条件のいずれかが起こらない場合には回避できます。

可能な対策は、

1) 内蔵 L2 メモリもキャッシュ可能に設定する。
または

2) ライトスルー・キャッシュ・モードを使う。
または

3) 限定された量のデータ・ロケーションに対する必要な書き込みが"頻繁でない"場合は、 "ssync;" 命令を書き込みの前に挿入します。これにより、すべての待ち状態のキャッシュ書き込みが確実に完了します。

適用レビジョン:

0.3

41. 05000225—UART スタート・ビットが不正なパルス幅になる:

説明:

UART インターフェースから送信されたワードのスタート・ビットの幅が不正になります。

1 より大きいクロック分周比に対して (UART_DLL レジスタと UART_DLH レジスタにより指定)、パルス幅は公称ビット時間の 14/16 または 15/16 の値と見なすことができます。データ、パリティ、ストップの各ビットは正しい幅になります。データは正しく受信されます。UART のタイミングについては、アノーマリ 05000230 と 05000231 を参照してください。

対策:

なし

適用レビジョン:

0.3

42. 05000227—スクラッチパッド・メモリ・バンクを読み出すと、不正なデータが返される:

説明:

スクラッチパッド・メモリを読み出すと、ある条件下で不正なデータが返されます。スクラッチパッド・メモリの読み出しの直後にさらに読み出し(非スクラッチパッド・ロケーションを含む任意ロケーション)が続く場合で、かつアクセスされるアドレスが同じ下位アドレス・ビットを持たない場合に、この問題が発生します。これは、転送の 1 つがバイト・アクセスである必要があることを意味しています。他のアクセスは、最初のアクセスとは異なるバイト境界でのバイト、16 ビット、または 32 ビット・アクセスである必要があります。さらに、スクラッチパッド・メモリ読み出しの直前の命令は、2 個の DAG バンクの衝突、非 L1 メモリ・データのフェッチ、またはキャッシュ・ライン・フィルに起因するメモリ・ストールを発生する必要があります。

命令がスクラッチパッド読み出しの直後に読み出しを行わない場合は、問題は発生しません。また、連続する非バイト読み出しも正常に機能します。

対策:

アセンブリ・プログラマにとって最もシンプルな対策は、各スクラッチパッド読み出しの後ろに非読み出し命令を配置することです。C プログラマにとって 1 つのソリューションは、スクラッチパッドにデータをマップしないことです。

VisualDSP++Blackfin コンパイラには、このハードウェア・アノーマリに対する対策が含まれています。コンパイラは、該当するシリコン・レビジョンとデバイス番号に対してこの対策を自動的にイネーブルします。あるいは、コンパイラ・フラグ' -workaround scratchpad-read' を指定することにより、対策を手動でイネーブルすることができます。対策をイネーブルして、(2)と(3)の少なくとも一方がバイト・ロードである 3 個のロード命令シーケンスが発生すると (またはマルチイッシュ命令の一部として発生すると)、nop が(1)と(2)の間または(2)と(3)の間に挿入されます:

```
load instruction (1);
load instruction (2);
load instruction (3);
```

マクロ_WORKAROUND_SCRATCHPAD_READ が、コンパイル、アセンブル、リンク・ステージで定義されます。

VisualDSP++ランタイム・ライブラリも、該当するシリコン・レビジョンとデバイス番号に対してこのアノーマリ状態を回避します。

適用レビジョン:

0.3

43. 05000230—UART レシーバがある条件でボーレート差に対して弱くなる:

説明:

同期通信で、トランスミッタとレシーバのビット・クロックが公称値から一定パーセント値だけ異なることがあります。正確な差は、送信ワード構成と信号品質のようなその他の外部ファクタに依存します。

Blackfin UART レシーバでは、ビット・クロックが送信側クロックより低速または高速である場合、偏差が異なります。Blackfin UART レシーバは、送信側が少し低いビット・レートで動作する場合、その差に良く耐えますが、送信側が Blackfin レシーバより少し高いビット・レートで動作する場合は、連続転送(ワード間にギャップがない場合)で通信に失敗することがあります。

この理由は、標準構成の場合、レシーバは他のビットと同様に、ストップ・ビットを 16 回サンプルした後に、新しいスタート・ビット条件を検出するためです。ストップ・ビットが検出されたか否かの判定は 9 番目のサンプルの後に行われます。ストップ・ビットが検出されると、レシーバは直ちに次のワードを読み出すことができるべきですが、Blackfin レシーバでは残りの 7 サンプルも使用してしまいます。

これによる影響は、サンプリング・エラーが上記のようなデータ転送で蓄積することです。

シングル転送またはワード間にギャップがある転送ではサンプリング・エラーが蓄積されないため、このアノーマリの影響ありません。

対策:

送信側は、低い(または等しい) ビット・レートで動作する必要があります。

これを保証できない(または不可能な)場合は、送信側を 1 ストップ・ビットより長く送信するように設定し、ワード間に必要なギャップを挿入してください。

双方向通信チャンネルでトランスミッタとレシーバが Blackfin デバイスである場合は、上記対策でこの問題を解決できません。これについては、アノーマリ 05000225 と 05000231 を参照してください。

適用レビジョン:

0.3

44. 05000231—UART STB ビットがレシーバ設定値に悪影響を与える:

説明:

STB ビットは、トランスミッタで発生するストップ・ビット数を制御します。

ただし、この設定値はレシーバでサンプルするストップ・ビット数にも悪影響を与えます。正しい動作は、レシーバが常に 1 ストップ・ビットをサンプル/テストすることですが、レシーバは、STB ビットで設定されるストップ・ビット数をサンプル/テストすることになります。この不正な動作は、フレーム・エラーの検出にも影響を与えます。

このアノーマリは、2 個アノーマリ 05000230 に対する対策を、2 個の Blackfin デバイスで構成される双方向リンクのケースに適用できなくすることに注意してください。

対策:

なし

適用レビジョン:

0.3

45. 05000232—マルチチャンネル・モードで SPORT データ送信ラインが誤って駆動される

説明:

マルチチャンネル・モードでは、SPORT の MRCS レジスタを使って、アクティブにするチャンネル(送信または受信するチャンネル) と無視するチャンネルを選択します。無視される各チャンネルは、DT 出力がスリー・ステートになります。このアノーマリは、そのケースに当てはまりません。DT 出力が代わりに意図的に駆動されます。

対策:

これは、複数 SPORT のトランスミッタが同じラインを共用するアプリケーションにのみ影響します。この場合、外付けロジックが必要です。

適用レビジョン:

0.3

46. 05000242—PLL_CTL レジスタの DF ビットがハードウェア・リセットに応答しない:

説明:

ハードウェア・リセットの前に DF ビットをセットすると、ハードウェア・リセットの後に PLL は CLKIN の 2 分周を続けますが、PLL_CTL レジスタの DF ビット自体はクリアされます。

対策:

リセット後に CLKIN の 2 分周が不要な場合には、DF をクリアして PLL を再設定してください。

適用レビジョン:

0.3

47. 05000244—I キャッシュがオンの場合、コントロールの変更付近で CSYNC/SSYNC/IDLE により不具合が発生する:

説明:

命令キャッシュをイネーブルすると、コントロールの変更付近(非同期の例外/割り込みも含む)で CSYNC/SSYNC/IDLE により予期しない結果が発生することがあります。

この問題を発生する最も一般的なシーケンスの例は、BRCC (NP)とそれに続く次の 3 つの命令内の何処かに CSYNC/SSYNC/IDLE がある場合です:

```
BRCC X [predicted not taken]
nop
nop
CSYNC/SSYNC/IDLE // this instruction is bad in any of the 3 instructions following BRCC X
```

この問題に遭遇するもう 1 つのシーケンスは、CSYNC/SSYNC/IDLE を指す BRCC (BP)の後ろに、投機的にフェッチされた CSYNC/SYNC/IDLE が 2 サイクル以内に BRCC に"追い付く"ことを可能にする命令のストールが続く場合です:

```
BRCC X (bp)
Y: ... ...
X: CSYNC/SSYNC/IDLE
```

このシーケンスでは、不具合の再現が極めて困難です。BRCC の前のストールと非常に特殊なキャッシュ動作との正確な組み合わせが必要です。

対策:

命令キャッシュをオフにすることが、このアノーマリを回避する 1 つの方法です。

アセンブリ・コードでは上記シナリオを回避する必要があります。VisualDSP++ Blackfin アセンブラは、CSYNC/SSYNC/IDLE 命令がこのアノーマリ状態が発生しそうな場合に警告 ea5507 を発生します。

非同期イベントに無関係なすべてのケースに対しては、VisualDSP++ Blackfin コンパイラが該当するシリコン・レビジョンとデバイス番号に対して自動的にイネーブルされる対策を含んでいます。この対策は、-workaround speculative-syncs コンパイラ・フラグを使って手動でイネーブルすることもできます。イネーブルすると、_WORKAROUND_SPECULATIVE_SYNCS マクロが、コンパイル、アセンブル、リンクのビルド・ステージで定義されます。この対策では NOP を挿入して、次の形式のアノーマリに対してアノーマリ状態を回避します:

```
IF CC JUMP ...;                IF CC JUMP X (BP);          LSETUP(LT, LB)
CSYNC/SSYNC/IDLE               ...                               LT:   CSYNC/SSYNC/IDLE
CSYNC/SSYNC/IDLE               ...                                     CSYNC/SSYNC/IDLE
CSYNC/SSYNC/IDLE               X: CSYNC/SSYNC/IDLE                     IF CC JUMP X:
                                                                  LB: NOP;
                                                           X: ...
```

VisualDSP++ランタイム・ライブラリも、該当するシリコン・レビジョンとデバイス番号に対してこのアノーマリ状態を回避します。

非同期割り込みイベントの場合、割り込みをディスエーブルし、SSYNC/CSYNC/IDLE の前に 2 個の NOP を詰めることにより SSYNC/CSYNC/IDLE 命令を保護することができます:

```
CLI R0;
NOP; NOP;          // 2 Padding Instructions
CSYNC/SSYNC/IDLE
STI R0;
```

例外に対しては、メモリのキャッシャブル領域に対する任意のアクセスの後に 3 個の NOP を詰める必要があります。

最後に、この対策はスーパーバイザ・モード命令を使って割り込みをディスエーブルおよびイネーブルするため、ユーザ・モードには適用されません。ユーザ領域では、CSYNC 命令または SSYNC 命令を使用しないでください。

適用レビジョン:

0.3

48. 05000245—条件付き分岐のシャドウ内のアクセスで偽ハードウェア・エラーが発生する:

説明:

条件付きジャンプが採用したパスと反対側のコントロール・フロー上で、あるロードが予約済みまたは不正メモリにアクセスすると、偽ハードウェア・エラーが発生します。

次のシーケンスに、これが発生する状況を示します:

シーケンス#1:

"非採用と予測される"分岐に対しては、パイプラインは非採用と予測された分岐命令にシーケンシャルに続く命令をロードします。パイプラインのデザインにより、これらの命令は、分岐の誤予測によりアボートされる前に、投機的に実行することができます。このアノーマリは、分岐に続く 3 個のいずれかの命令スロットにハードウェア・エラーを発生させるロードが含まれている場合に発生します:

```
BRCC X [predicted not taken]
r0 = [p0];          // If any of these three loads accesses non-existent
r1 = [p1];          // memory, such as external SDRAM when the SDRAM
r2 = [p2];          // controller is off, then a hardware error will result.
```

シーケンス#2:

"採用が予測される"分岐の場合は、分岐のディステネーションにある 1 命令スロットが、ハードウェア・エラーを発生させるアクセスを含むことができません:

```
BRCC X (bp)
Y: ... ...
X: r0 = [p0];       // If this instruction accesses non-existent memory, // such as external SDRAM
when the SDRAM controller // is off, then a hardware error will result.
```

対策:

アセンブリでプログラムする場合は、上記条件を回避する必要があります。

VisualDSP++Blackfin コンパイラには、このハードウェア・アノーマリに対する対策が含まれています。コンパイラは、該当するシリコン・レビジョンとデバイス番号に対してこの対策を自動的にイネーブルします。あるいは、コンパイラ・フラグ'-workaround speculative-loads'を指定することにより、対策を手動でイネーブルすることができます。

対策がイネーブルされると、コンパイラは、NOP 命令を挿入してアノーマリ状態を回避します。

対策をイネーブルすると、_WORKAROUND_SPECULATIVE_LOADS が、コンパイル、アセンブル、リンク・ビルド・フェーズで定義されます。

この対策の重複適用を回避する種々のチェックがコンパイラ内にあります。例えば、対策を適用する前に、ロードについて次を確認します:

・SP または FP を使用してないこと(この場合スタックからであるため不正ではありません)
・揮発性評価タイプのアドレスでないこと(この場合既知の有効アドレスから)
・コンパイラに未知のアドレスからであること
・分岐ターゲットで重複されていないこと(この場合、投機的実行可能)
・ジャンプの前に実行されないこと(この場合、投機的実行可能)

VisualDSP++ランタイム・ライブラリも、該当するシリコン・レビジョンとデバイス番号に対してこのアノーマリ状態を回避します。

適用レビジョン:

0.3、0.5

49. 05000248—TESTSET 動作により他のコア上でストールが発生する:

説明:

一方のコアが内部 L2 メモリに対して TESTSET 動作を行うと、他方のコアが同時に L2 メモリをアクセスする場合、他方のコアをロックアウトします。

対策:

内部 L2 メモリに対するすべての TESTSET 動作の直後に、L2 メモリに対する(ダミー) 書き込みが続く必要があります。

VisualDSP++C/C++コンパイラには、このアノーマリに対する対策が含まれています。testset 命令の直後の L2 定義の変数に対する書き込みをコンパイラが自動的に発行することにより回避が行われます。これは、 __builtin_testset() 呼び出し用に生成されるコードの一部として実行されます。コンパイラは、該当するシリコン・レビジョンに対してこの対策を自動的にイネーブルします。あるいは、コンパイラ・フラグ"-workaround l2-testset-stall"を指定することにより、対策を手動でイネーブルすることができます。

適用レビジョン:

0.3

50. 05000250—マルチチャンネル(TDM)モードのある条件でデータ・ワードのビット・シフトが正しくない:

説明:

マルチチャンネル・モードでは、フレーム同期の幅が実際のデータ・フレーム幅より 1 ビットだけ大きい場合(すなわち"非アクティブ・ビット"が 1 つある場合)、送信フレームの先頭ワードが 1 ビット左にシフトされるため、LSB が 2 番目のワードの MSB になってしまいます。他の全ワードは正常に送信されます。

フレーム同期幅が実際のフレーム幅と一致する場合、または 1 ビット大きい場合はすべてのデータが正常に送信されます。

例えば、16 ビット・データが 8 ワードある場合、データ・フレームでは 128 ビットになります。

```
If RFSDIV = 127 --> all data words are CORRECT
If RFSDIV = 128 --> first word is INCORRECT
If RFSDIV = 129 --> all data words are CORRECT
If RFSDIV = 130 --> all data words are CORRECT
```

対策:

RFSDIV レジスタ値にデータ・ビット数+/- 1 を設定して上記ケースを回避してください。

適用レビジョン:

0.3

51. 05000251— 予約済み領域内の MMR アクセスに対して例外が発生されない

説明:

アドレス 0xFFC01900～0xFFC01AFF の領域は、システム MMR メモリ空間内で予約されていますが、この領域に対するアクセスによってハードウェア・エラー割り込みが発生されません。

対策:

なし

適用レビジョン:

0.3

52. 05000253—タイマの最大外部クロック速度:

説明:

汎用タイマは、TMRx ピンに PWM 出力波形を発生します。このタイミングは、システム・クロック(SCLK)の周期または外部から入力されたクロック(TMRCLK または TACLK)の周期で決定されます。正常動作のためには、SCLK が使用するソース(TMRCLK または TACLK)より高速である必要があります。

データシートとハードウェア・リファレンス・マニュアルの仕様では、TMRCLK と TACLK の速度は最大 1/2 SCLK まで可能です。

ただし、最大レートはこの規定値より低くなります。最小 SCLK/TMRCLK または SCLK/TACLK 比は 2.5～2.7 です。正確な値はまだキャラクタライゼーションされていません。

対策:

最小 SCLK/TMRCLK または SCLK/TACLK 比を 3 にして安全に使用してください。

適用レビジョン:

0.3

53. 05000254—外部クロックを使ったシングルショット PWM 出力モードのパルス幅が正しくない:

説明:

タイマが PWM_OUT モードで、かつシステム・クロックではなく外部クロック(PPI_CLK またはフラグ・ピンに入力される信号)で駆動され、かつシングル・パルス・モード(PERIOD_CNT = 0)である場合、発生されるパルス幅が+1 または-1 カウントだけずれることがあります。他の全モードはこのアノーマリの影響を受けません。

対策:

推奨対策は、シングルパルス・モードではなく連続モードを使用することです。タイマをイネーブルした後、直ちにディスエーブルすることができます。タイマはシングル・パルスを発生し、周期の終わりまでカウントした後に実質的に自分自身をディスエーブルします。発生される波形は所望の長さになります。

PULSEWIDTH が所望の幅である場合、次のシーケンスによりシングル・パルスが発生されます:

```
TIMERx_CONFIG = PWM_OUT|CLK_SEL|PERIOD_CNT|IRQ_ENA;    // Optional: PULSE_HI|TIN_SEL|EMU_RUN
TIMERx_PERIOD = PULSEWIDTH + 2;                        // Slightly bigger than the width
TIMERx_WIDTH = PULSEWIDTH;
TIMER_ENABLE = TIMENx;
TIMER_DISABLE = TIMDISx;
<wait for interrupt (at end of period)>
```

適用レビジョン:

0.5

54. 05000257—短いハードウェア・ループ時の割り込み/例外により不正な命令がフェッチされることがある:

説明:

4 命令長より短いハードウェア・ループが、割り込みまたは例外に起因してループの終わりに中断されると、予期しない動作が発生することがあります。この状況では、命令フェッチのレイテンシを小さくするために使用されているプロセッサのループ・バッファが正しく動作しなくなるため、ループの終わりで正しくない命令がフェッチされてしまうことがあります。

対策:

このアノーマリには幾つかの対策があります。1 つ目は、すべての割り込み/例外ハンドラ内でループ・カウンタ・レジスタ(LC0 と LC1)に書き込みを行うことにより、ループ・バッファをクリアすることです:

```
R0=LC0;
LC0=R0;
R0=LC1;
LC1=R0;
```

2 つ目は、コンテキスト・スイッチ・コード内にループ・カウンタを使用することです:

```
[--SP] = LC0;
[--SP] = LC1;

<interrupt code>

LC1 = [SP++];
LC0 = [SP++];
```

最後の対策は、ループ長を 4 命令以上にするため、ループに NOP を追加することです。

また、イベント・ハンドラでハードウェア・ループを使用している場合、LCx レジスタに書き込みが行われるごとに、対応するループ・バッファがクリアされるため、上記ステップは不要です。

VisualDSP++Blackfin コンパイラには、このアノーマリに対する対策が含まれています。コンパイラは、該当するシリコン・レビジョンとデバイス番号に対してこの対策を自動的にイネーブルします。あるいは、コンパイラ・フラグ'-workaround short-loop-exceptions-257'を指定することにより、対策を手動でイネーブルすることができます。対策をイネーブルすると、コンパイラは LC0 レジスタと LC1 レジスタの退避と回復を割り込みハンドラと例外ハンドラに追加します(既に存在しない場合)。これらのレジスタが処理ルーチン内で使用されている場合には、通常コンパイラはこれらを退避させるだけです。

対策をイネーブルすると、_WORKAROUND_SHORT_LOOP_EXCEPTIONS が、コンパイル、アセンブル、リンク・ビルド・フェーズで定義されます。

適用レビジョン:

0.3

55. 05000258—ICPLB データ・レジスタのビット 9 とビット 12 が異なるとき、命令キャッシュが壊れる:

説明:

ICPLB データ MMR のビット 9 とビット 12 が異なるとき、キャッシュが正しく更新されません。例えば、特定のキャッシュ・ラインについて、そのキャッシュ・ラインの値がキャッシュ内に存在しない間、キャッシュ・タグが有効になることがあります。

対策:

各 ICPLB エントリ内で、ビット 12 の状態をビット 9 に設定してください。

VisualDSP++Blackfin ランタイム・ライブラリには、このアノーマリに対する対策が含まれています。

cplb_mgr and _cplb_init ルーチン(これはランタイム・ライブラリのデフォルト・キャッシュ・サポートに含まれています)がビット 12 (CPLB_L1_CHBL)と同じ状態をビット 9 (予約済み)に設定します。

適用レビジョン:

0.3

56. 05000260—ICPLB_ステータス MMR レジスタが壊れることがある:

説明:

例外を発生させた CPLB を調べるとき、ICPLB ステータス・レジスタに頼ることができません。このレジスタは次の場合に壊れます:

1)ページの最後の 64 ビット内へジャンプする命令があり(ICPLB により指定)、さらに、

2)これらの最後の 64 ビット内に存在する命令が命令例外を発生し、さらに、

3)投機的命令フェッチが、次のページまでインクリメントして、別の命令例外の原因に遭遇する。

これらすべての条件が満たされると、ICPLB_STATUS は最初の例外原因ではなく投機的命令フェッチの方を反映するようになります。

対策:

命令保護違反と ICPLB の複数ヒットをこのレジスタを使わずに処理してください。

ICPLB_FAULT_ADDR レジスタを使って、例外を発生したアドレスを調べてください:

1) CPLB ミスの場合、例外により問題のアドレスをカバーする CPLB が単純にスワップ・インされます。

2) CPLB の複数ヒットの場合、ICPLB_FAULT_ADDR レジスタを使って、例外を発生したアドレスを探し、次にすべての CPLB エントリについて問題のアドレスをカバーする CPLB を探してください。

3)保護違反例外の場合、処理はユーザ固有になります。

適用レビジョン:

0.3

57. 05000261—DCPLB FAULT ADDR MMR レジスタが壊れる:

説明:

DCPLB_FAULT_ADDR MMR レジスタが壊れることがあります。これが発生した場合、アボートされたデータ・メモリ・アクセスは、保護例外とストール (2 個の DAG の衝突、キャッシャブル・メモリへのアドレシングやミス、または L2 からの単なるフェッチなどにより発生)を発生します。

対策:

1)データ例外ハンドラへの最初のエントリ時には、そのハンドラから（サービスの実行なしに）直ちにリターンし、同じデータ CPLB 例外ハンドラ内の 2 つ目のパスで DCPLB_FAULT_ADDR を信頼します。例外の原因がサービスされていなければ、この例外が再発生して 2 つ目のパスを発生させます。例外ハンドラは、戻った直後に誤って生成されることがないので、2 つ目のパスでは、DCPLB_FAULT_ADDR レジスタが正しくなります。(高い優先順位の例外ではなく）同じ例外の 2 つ目のパスで正しく応答できるように、最初のパスで RETX レジスタのコピーを取得して、2 つ目のパスと比較する必要があります。

または

2)例外の誤処理により発生する偽例外に耐性を持たせます。保護違反、CPLB ミス、CPLB 複数ヒットの 3 種類のデータ・メモリ例外の場合、推奨ソフトウェア対策は次の通りです:

a)データ保護例外の場合、ハンドラ内で DCPLB_FAULT_ADDR レジスタの代わりに DCPLB_STATUS レジスタを使用します。これにより、フル・アドレスの例外の代わりに保護違反のページが提供されます。理想的ではないですが、これが十分な情報を提供するものと思われます。

b)データ CPLB ミス例外の場合、DCPLB_FAULT_ADDR レジスタを使いますが、このレジスタに報告されるアドレスが現在の真の例外ではなく前にキャンセルされた投機的例外であることの警告を受けるようにします。したがって、このアドレスは:

i) ロード済みディスクリプタを持っているページを指す
または

ii) 実際にフェッチされることのないアドレスを指す

i)の場合、CPLB ミス・ハンドラが冗長な CPLB エントリを発生することがありますが(さらにページ・チェックが実行されない限り)、この希に発生させる冗長なページ・ディスクリプタを削除する複数 CPLB ヒット・ハンドラが存在する場合、これにより耐性を持つようになります。

ii)の場合、DCPLB_FAULT_ADDR レジスタは例外ハンドラから戻った直後に不正になることがないため、誤った DCPLB_FAULT_ADDR レジスタ値の繰り返しに起因する例外ハンドラの無限ループを発生させることなく、CPLB ミス・ハンドラから無意味なアドレスを無視することができます。

c)データ CPLB 複数ヒット例外の場合、上記問題に対応するようなハンドラを設けます。複数 CPLB 例外が発生しないことに頼らないでください。

VisualDSP++Blackfin ランタイム・ライブラリには、このアノーマリに対する対策が含まれています。この対策では、DCPLB ミス例外が特定の PC から発生した最初のとき、これを無視します。障害アドレスの修正は 2 回目に保証されます。

適用レビジョン:

0.3

58. 05000262—データ・キャッシュへのストアが失われることがある:

説明:

連続する2つのデュアルDAG動作の最初に対象となったサブバンクに対して確定した待機中の書き込みが次の場合に失われます:

1)データ・キャッシュがイネーブルされ、さらに、

2)最初のデュアル DAG アクセスで、DAG0 がキャッシュ・ミスになり、DAG1 が読み出しで、かつ両アクセスが同じ非 L1 サブバンクに対するエイリアスで、さらに、

3) 2 つ目のデュアル DAG が次の命令であり、かつ DAG1 が L1 SRAM のアクセス(読み出しまたは書き込み)であり、さらに、

4)最初のデュアル DAG アクセス後 3 クロック・サイクル以内にフローの予期しない変更があり、ユーザがフロー変化を制御していない場合

対策:

1) データ・キャッシュを使わないか、または、

2)最初のデュアル DAG アクセスが次の場合に、連続するデュアル DAG メモリ・アクセスを回避してください:

a) 両 DAG が L2 を対象にし、さらに、

b) 両 DAG が同じサブバンクをエイリアスし、さらに、

c) DAG1 による読み出しを含み、その直後に、DAG1 が L1 アクセスを行う 2 つ目のデュアル DAG アクセスが続く場合

VisualDSP++Blackfin コンパイラには、このアノーマリに対する対策が含まれています。コンパイラは、該当するシリコン・レビジョンとデバイス番号に対してこの対策を自動的にイネーブルします。あるいは、コンパイラ・フラグ '-workaround stores-to-data-cache-262'を指定することにより、対策を手動でイネーブルすることができます。

対策がイネーブルされると、コンパイラが、アノーマリを発生するデュアル DAG 命令を発行しないようにします。コンパイラは、対策が不要なケースを調べるために使用可能なすべてのバンク情報を使用しようとします。

対策をイネーブルすると、_WORKAROUND_LOST_STORES_TO_DATA_CACHE_262 が、コンパイル、アセンブル、リンク・ビルド・フェーズで定義されます。

VisualDSP++ランタイム・ライブラリは、該当する場合、このアノーマリを回避するようにビルドおよび修正されています。

適用レビジョン:

0.3

59. 05000263—ICPLB 例外を受理するとハードウェア・ループが壊れる:

説明:

ハードウェア・ループ・ロジックにエラーがあり、このためプロセッサのループ実行中に命令 ICPLB 例外が発生すると、不正な命令が実行されることがあります。

対策:

次のいずれかを行います。

1) ハードウェア・ループの使用を回避する。または、

2) ハードウェア・ループが L1 メモリ内にのみ存在することを確認する。または、

3) L1 メモリの外側に存在するハードウェア・ループの実行中に ICPLB 例外が発生しないことを確認する。

ハードウェア・ループが L1 メモリ内にある場合、このループは、例えば、CPLB ページ境界を超えて有効な CPLB 定義を持たないページへ行くなどにより ICPLB 例外を発生しません。また、ターゲット・アドレスで ICPLB 例外が発生したとき、ループ内から非 L1 メモリへ分岐させないでください。ループ中に割り込みが発生して、割り込みサービス・ルーチン(ISR)が非 L1 メモリ内にある場合は、ISR から ICPLB 例外を発生させないでください。

適用レビジョン:

0.3

60. 05000264—ハードウェア・ループ内の最後から 2 番目の命令で CSYNC/SSYNC/IDLE により無限ストールが発生する:

説明:

SSYNC、CSYNC、または IDLE がハードウェア・ループの最後から 2 番目の命令として配置されている場合、同期を実行しようとすると、プロセッサが無限ストールに入る可能性があります。

対策:

ハードウェア・ループの最後から 2 番目の命令として、SSYNC、CSYNC、または IDLE 命令を使わないでください。

割り込みまたは例外によりプロセッサがストールから抜け出すため、DMA または割り込みの実行中はこの問題は表面化しません。

VisualDSP++Blackfin コンパイラには、このアノーマリに対する対策が含まれています。コンパイラは、該当するシリコン・レビジョンとデバイス番号に対してこの対策を自動的にイネーブルします。あるいは、コンパイラ・フラグ'-workaround pre-loop-end-sync-stall-264'を指定することにより、対策を手動でイネーブルすることができます。

対策をイネーブルすると、コンパイラはハードウェア・ループの最後から 2 番目の命令が、このアノーマリを発生する可能性を持つ CSYNC、SSYNC または IDLE 命令にならないようにします。

対策をイネーブルすると、_WORKAROUND_PRE_LOOP_END_SYNC_STALL_264 が、コンパイル、アセンブル、リンク・ビルド・フェーズで定義されます。

適用レビジョン:

0.3

61. 05000265—外部 SPORT TX と RX クロックの低速な入力エッジ・レートでノイズに対して弱くなる:

説明:

ノイズの多いボード環境と外部 SPORT の受信クロック(RSCLK)と送信クロック(TSCLK)の低速入力エッジ・レートとの組み合わせにより、多様な問題が観測されます。RSCLK/TSCLK で予期しない高周波変化が発生すると、SPORT がノイズから生ずる余分なグリッチ・クロック・パルスを認識することがあります。

RSCLK/TSCLK での高周波変化は、多くの場合外部シリアル・クロックの立ち上がりまたは立ち下がりエッジのノイズから発生します。このノイズと低速で変化するシリアル・クロック信号とが組み合わさると、クロック入力の高い感度のため、幅の狭い余分なビット・クロックが発生することがあります。低速なスルーレート入力により、スイッチング・ポイント付近のクロック入力のノイズで、クロック入力がスイッチング・ポイントを通過し、さらに再通過するようになります。この発振により、シリアル・ポートの内部ロジックでグリッチ・クロック・パルスが発生します。

このグリッチ・クロック・パルスに起因して観測される問題は:

- ステレオ・シリアル・モードの場合、これは左/右データの入れ換えとなるフレーム同期外れとして現れます。
- マルチチャンネル・モードの場合、これは不正確なまたはスキップされたフレームを表す MFD カウントとして現れます。
- ノーマル(アーリー)フレーム同期モードの場合、受信データ・ワードが 1 ビット右にシフトされます。MSB が誤って符号拡張モードでキャプチャされます。
- すべてのモードで、フレーム同期の開始と、受信または送信最終ビットとの間のすべての入力クロックにノイズがある場合、受信または送信データ・ワードが部分的に右シフトされて現れます。

ステレオ・シリアル・モードでは(SPORTx_RCR2 のビット 9 がセット)、RSCLK/TSCLK に予期しない高周波変化があると、SPORT がワード・クロックの立ち上がりまたは立ち下がりエッジを誤ることがあります。このために、ステレオ・シリアル・データの左または右ワードが失われます。これは、ステレオ・オーディオ信号を聞いているときに左/右チャンネルの入れ換えとして観測されます。SPORT の内部ロジックでノイズから余分なビット・クロック・パルスが発生すると、FS エッジ検出ロジックが狭い幅のパルスを発生し、同時に、SPORT が次の'通常の'ビット・クロック周期内で外部 FS 信号の検出ができなくなります。エッジ検出ロジックから出力される幅の狭い FS パルスは、SPORT のシーケンシャルなロジックから無視されます。FS ロジックのエッジ検出部分は既に'トリガー済み'であるため、次の'通常の' RSCLK は RFS の変化をこれ以上検出しません。I2S/EIAJ モードでは、このために、2 つの左/右チャンネルとして 1 ステレオ・サンプルが検出/転送され、すべての後続チャンネルはメモリ内でワードが入れ代わります。

マルチチャンネル・モードでは、マルチチャンネル・フレーム遅延(MFD)ロジックが余分な同期パルスを受信して、カウントを早めるかダブル・カウントしてしまいます(カウントが既に開始している場合)。MFD がゼロのときは、カウントが 1 サイクル早く開始されているため 15 へロールオーバーします。

アーリー・フレーム同期モードでは、FS がアクティブになった同じクロック・サイクルの駆動エッジでノイズが発生すると、FS ロジックは余分なパルスを受信して 1 サイクル早くワード長のカウントを開始します。最初のビットが 2 回サンプルされ、最後のビットがスキップされます。

すべてのモードでは、FS がアクティブになった後の任意のサイクルでノイズが発生すると、ビット・カウント・ロジックが余分なパルスを受信して、カウントが早く進み過ぎます。これが動作ユニットで発生すると、1 サイクル進んでワード長のカウントが終わります。ノイズが発生したビットは 2 回サンプルされて、最後のビットはスキップされます。

対策:

1)ビット・クロックのスルーレートを大きくするか、またはシリアル・ビット・クロックの立ち上がりと立ち下がり時間を短くして、ノイズに対する感度を小さくして、変化の近傍のノイズにより、狭い幅のノイズからビット・クロック・パルスが発生しないようにします。狭い高周波数パルスは、SPORT またはフレーム同期の検出に影響を与えません。EMI 条件を満たし適切な場合にはエッジをできるだけ急峻にします。

2)可能な場合、内部発生のビット・クロックとフレーム同期を使います。

3)正しい PCB デザイン方法に従います。TSCLK ラインから RSCLK をシールドして、シリアル・クロックの混入を小さくします。

4)ボード上の RSCLK、TSCLK、フレーム同期の各パターンを分離して、FS がスイッチするとき駆動エッジで発生する混入を小さくします。

ステレオ・シリアル・モードで観測される問題に対する特別な対策は、フレーム同期信号を遅延させて、ノイズから混入するビット・クロック・パルスにより、フレーム同期処理を開始させないようにすることです。これは、ビット・クロック・パターンの直列抵抗よりフレーム同期パターンの直列抵抗が大きい場合に実現できます。ビット・クロックが VDD の 10%閾値(立ち下がりエッジ・ビット・クロック)または 90%閾値(立ち上がりエッジ・ビット・クロック)を通過するまでフレーム同期変化が 50%ポイントを通過しないようにします。

ノイズ耐性を向上させるため、PLL_CTL レジスタのビット 15 をセットし、次に該当する PLL プログラム・シーケンスを実行して、入力ピンでオプションのヒステリシスをイネーブルすることができます。

適用レビジョン:

0.3、0.5

62. 05000266—IMDMA を使う前に IMDMA ディステネーション IRQ ステータスを読み出す必要がある

説明:

IMDMA コントロール・レジスタをアクセスする前に IMDMA ディステネーション IRQ ステータス・レジスタからの読み出しがない場合、IMDMA 転送が正しく動作しません。

対策:

IMDMA コントロール・レジスタに対するすべての書き込みの前に、IMDMA ディステネーション・チャンネル 0 の IRQ ステータス・レジスタを読み出してください。 これは、各書き込みの前ではなく、リセットの後に 1 回だけ必要です。

適用レビジョン:

0.5

63. 05000267—ある条件下で IMDMA がデータを壊す

説明:

L1 メモリまたは内蔵 L2 メモリから内蔵 L2 メモリへデータを転送する際に、L2 メモリ内に"バンク競合"が存在する場合、またはコア・アクセス (命令またはデータのフェッチ)が IMDMA アクセスと同時に行われた場合に、データが破壊されることがあります。"バンク競合"は、IMDMA とシステム DMA が同じサイクル内で同じ L2 サブバンクをアクセスようとした場合に発生します。L2 メモリは 16 個の 8KB バンクで構成されています。

内蔵 L2 メモリから L1 メモリへのデータ転送で、バンク競合が回避されたときでもデータが壊れることがあります。

このアノーマリは、 L1 から L1 メモリへの転送には適用されません。

対策:

L1 メモリまたは内蔵 L2 メモリから内蔵 L2 メモリへの転送でこの不具合を回避するためには、次の条件を満たす必要があります。

1) IMDMA をイネーブルする際、先にソース・チャンネルをイネーブルし、次に SSYNC 命令を発行します。次に、ディステネーション・チャンネルをイネーブルし、次にもう一度 SSYNC 命令を発行します。

2) システム DMA チャンネルが IMDMA チャンネルと同じ L2 サブバンクをアクセスしないようにします。

3) IMDMA が L2 メモリから L1 メモリへデータを転送中に、内蔵 L2 メモリ空間に対するコア・アクセス (命令フェッチまたはデータ・フェッチ)が発生しないようにします。

内蔵 L2 メモリから L1 メモリへの転送は、システム MEMDMA (DMA コントローラ 1 または 2)を使って行う必要があります。

適用レビジョン:

0.3、0.5

64. 05000269—活発な I/O 動作により、内部電圧レギュレータ出力電圧 (Vddint)が上昇する:

説明:

内部電圧レギュレータは、特に VDDext が高い時に、活発な I/O 動作により混入する電源とグラウンドのノイズ過渡電圧に敏感です。このために、VDDint が設定された値より高くなることがあります。場合によっては、値がデータシートの上限を超えることがあります。VDDint は、I/O 動作が減少または停止すると設定された値に戻ります。

BGA パッケージを採用するデバイスは、LQFP パッケージを採用するデバイスより敏感です。

現在まで、規定値の上限を超える電圧は、疑似的に I/O を活発に動作させたテストを実行中にのみ観測されています(例えば、アドレス・ラインとデータ・ラインのすべてのビットが各クロックでトグル)。スペック外の動作は、アプリケーション・コードを実行するユーザのアプリケーションでは観測されていません。

対策:

この問題は、外部電圧レギュレータを使用した場合には発生しません。この問題がアプリケーションで発生するか否かを調べるときは、次の条件/セットアップで VDDint 波形を観測してください:

- VDDext 電源の偏差に基づいて最大 VDDext を入力します。
- 定常状態(非スタートアップ)でアプリケーションを実行します。
- オシロスコープを最小グラウンドと信号ループで VDDint に接続します。
- 通常の過渡電圧を回避するため、設定する値より 10%大きい値と絶対最大電圧値 (最大 Vddint 仕様についてはデータシートを参照)との間の VDDint 値でトリガーするようにオシロスコープを設定します。

すべてのデバイスが、この問題に同じように敏感ではないことに注意してください。最小 10 個のデバイスで上記監視を繰り返してください。

問題が発生する場合には、I/O が活発に動作している間 VDDint 値が大きくなります。VDDint の最大値が最大 VDDint 以下に維持される場合には、信頼性の問題はありませんが、消費電力は大きくなります。

問題は、VDDext、I/O の動作度、VDDint の設定値の関数であるため、次の技術を使ってこの問題を軽減/改善することができます:

- 問題は、VDDext 値が 3.3V より高いときに発生しやすいため、偏差+/- 2%以下の 3.3V (または以下) のレギュレータを使うことが最適です。
- VDDext に十分なバイパスを設けます。
- 可能な場合 I/O 動作を減らします (例えば、動作 SCLK 周波数レートを下げます)。
- アプリケーション・ノート EE-228 の条件に従います。さらに、ユーザ・アプリケーションの定格条件を満たす最小ゲート電荷定格を持つ PMOS FET を使用します。

適用レビジョン:

0.3

65. 05000270—活発な I/O 動作により、内部電圧レギュレータ(Vddint)の出力電圧が低下する:

説明:

活発な I/O 動作により、VDDint が低下することがあります。ループの設定ポイントの発生に使用されるリファレンス電圧が、電源ノイズにより低下します。電圧は、アプリケーションの動作周波数を満たすために必要な最小値より低いレベルに低下することがあります。VDDint は、I/O 動作が停止すると設定された値に戻ります。

対策:

この問題は、外部電圧レギュレータを使用した場合には発生しません。この問題がアプリケーションで発生するか否かを調べるときは、次の条件/セットアップで VDDint 波形を観測してください:

・VDDext 電源の偏差に基づいて最大 VDDext を入力します。
・定常状態(非スタートアップ)でアプリケーションを実行します。
・オシロスコープを最小グラウンドと信号ループで VDDint に接続します。
・設定する値より 5%低い VDDint 値でトリガーするようにオシロスコープを設定します。次の項目により、この問題を軽減できることがあります:
・可能な場合、SCLK 周波数を下げて I/O 動作を減らします。
・観測される低下値に近い値(50mV の整数倍)だけ電圧レギュレータの設定値を大きくします。
・VDDext に十分なバイパスを設けます。

適用レビジョン:

0.3

66. 05000272—あるデータ・キャッシュ・ライトスルー・モードが Vddint <= 0.9V で失敗する:

説明:

ライトスルー・モードでデータ・キャッシュをイネーブルし、DCPLB の AOW ビットをセットしないで、Vddint を 0.9V 以下にすると、データが破壊されることがあります。

対策:

Vddint <= 0.9V のとき、ライトバック・モードでデータ・キャッシュを動作させるか、またはライトスルー・モードの動作中に、DCPLB の AOW ビットをセットしてください。Vddint が 0.9V より高いときは、このアノーマリは発生しません。

適用レビジョン:

0.3、0.5

67. 05000274—外部同期メモリに対するデータ・キャッシュ・ライトバックが失われる

説明:

コア・クロックがシステム・クロックより少なくとも 2 倍高速でない場合、キャッシュ・ビクティム処理中に外部同期メモリに対する書き込みが失われてしまうことがあります。キャッシュ・ビクティムは実質的に 256 ビット幅の書き込みであるため、外部同期メモリに対する 32 ビット幅を超える書き込みアクセスを実行する際は、その指示を出すたけであることに注意してください。

対策:

コア・クロック(CCLK)がシステム・クロック(SCLK)より少なくとも 2 倍高速であることを確認するか、あるいはデータ・キャッシュを使う場合ライトスルー・ポリシーを使います。これはこのモードでキャッシュ・ビクティム処理が発生しないためです。

適用レビジョン:

0.3、0.5

68. 05000275—PPI タイミングとサンプリング情報の更新

説明:

ADSP-BF561: Blackfin® 組込み対称型マルチプロセッサ・データシート (Rev. 0、2/2005)の PPI タイミング情報と対応する図は誤っています。

対策:

正しい情報については最新のデータシートをご覧ください。

適用レビジョン:

0.3、0.5

69. 05000276—非ゼロ PPI_DELAY を持つ外部フレーム同期 PPI モードに対するタイミング条件の変更:

説明:

プロセッサのデータシートの PPI タイミング図は、PPI_DELAY レジスタがゼロに設定された PPI モードにのみ適用されます。

対策:

PPI_DELAY レジスタの非ゼロ値に対して、次の情報が適用されます:

データシートでは、POLC = 0 のとき、フレーム同期は PPI クロックの立ち下がりエッジでサンプルされ、対応するセットアップ時間は、このエッジに対して表示されています。PPI_DELAY レジスタが非ゼロ値のとき、フレーム同期セットアップ時間が PPI クロック周期の 1/2 だけ大きくなります。データのサンプルを示す既存の図のポイントで、遅延のカウントが開始されます。

データシートでは、POLC = 1 のとき、フレーム同期は PPI クロックの立ち上がりエッジでサンプルされ、対応するセットアップ時間は、このエッジに対して表示されています。PPI_DELAY レジスタが非ゼロ値のとき、フレーム同期セットアップ時間が PPI クロック周期の 1/2 だけ大きくなります。データのサンプルを示す既存の図のポイントで、遅延のカウントが開始されます。

適用レビジョン:

0.3、0.5

70. 05000277—エッジ検出の 1SCLK サイクル後の I/O データ・レジスタへの書き込みが割り込みをクリアする:

説明:

I/O データ・レジスタへの書き込み(データ、レジスタのクリア、セット、トグル)がエッジ検出割り込みでエッジが検出された 1 システム・クロック・サイクルに発生した場合、セットされてから 1 システム・クロック・サイクル後にこのビットがクリアされます。

ビットが割り込みを発生するように設定されている場合は、割り込みが発生しますが、割り込みを発生させたビットが表示されません。コア・クロックが動作していないとき、または SIC_IMASK ビットが割り込みをイネーブルするように設定されていないときは、割り込みが失われます。

対策:

1 つのエッジ検出ソースのみが 1 つの割り込みに割り当てられた場合、それが割り込み原因と見なされるので、SIC_ISR レジスタと I/O レジスタの読み出し命令は不要です。イネーブルされると、すべての割り込みが正しく実行されることに注意してください。

エッジ検出割り込みの代わりにレベル検出割り込みを使ってください。割り込みサービス・ルーチンの再起動を防止して、次のエッジを検出するために、受信エッジの間で極性をトグルさせてください。2 つのエッジの間の遅延が割り込みサービス・ルーチンのサービスに十分であるか、または要求ラインに使用できる場合、これが適用できます。極性のトグルは、両エッジを探す場合に使用することができますが、1 個だけのエッジの場合は、他方の割り込みを無視する必要があります。

適用レビジョン:

0.3

71. 05000278—DMA 実行中にペリフェラルをディスエーブルすると DMA システムが不安定になる:

説明:

DMA の実行中で、かつ関係する DMA チャンネルがディスエーブルされる前に、ペリフェラル(PPI、SPORT、SPI など)がディスエーブルされると、DMA システムが壊れます。複数の DMA チャンネルが並行動作するアプリケーションでは、このアノーマリはデータが喪失するか、またはごちゃ混ぜのデータが転送されることを示しています。アノーマリは 1 個の DMA チャンネルを使うアプリケーションにも影響しますが、ユーザ・コードによりペリフェラルがシャットダウンされる場合その影響は見えません。

対策:

DMA チャンネルが動作中の場合、ペリフェラルの関係する DMA チャンネルをディスエーブルした後にペリフェラル自体をディスエーブルしてください。

DMA チャンネルが停止している場合は、ペリフェラルをディスエーブルした後に、関係する DMA チャンネルをディスエーブルする必要があります。チャンネルがディスエーブルされると、DMA ユニットはペリフェラル割り込みを無視して、それを直接割り込みコントローラへ渡すため、偽割り込みが発生します。

適用レビジョン:

0.3

72. 05000281— ISR コンテキストが回復されないとき偽ハードウェア・エラー例外が発生する:

説明:

ケースによっては、コンテキストを回復しない既存の割り込みサービス・ルーチン(ISR)が必要になります。次のシーケンスを考えてみます:

```
ISR_Exit:
  raise 14;          // instruction A
  rti;               // instruction B
```

このシーケンスは現在の割り込みレベルから戻ると、レベル 14 の割り込みサービス・ルーチンを直ちに実行します。理想的に、後者はユーザ・レベルに戻る前にコンテキストを回復するため、最初の ISR で時間を節約します。

問題の説明のため、最初の割り込みは次の命令で:

```
Rx = [Py];           // instruction C
```

あるいは同様の命令で発生するものとします。

プロセッサは ISR へジャンプします(RETI には命令 C のアドレスが含まれます)。プロセッサが上記命令 B に到達したとき ISR が Py を変更すると、命令 C を投機的にフェッチし、これが無効アドレスを指すようになります。命令 A のため、命令 B は実行されませんが、ハードウェア・エラー状態がラッチされます。ハードウェア例外が、次のシステム MMR 読み出しで発生します。

対策:

多くの場合、割り込みから戻る前にコンテキストが回復されるため、これは問題になりません。

上記のようなケースでは、投機的フェッチがハードウェア・エラーを発生しないロケーションで RETI レジスタをロードすることで十分です(上記"`raise; rti;`"シーケンスの前)。

適用レビジョン:

0.3

73. 05000283—特定のステージでシステム MMR 書き込みが停止されると、直ちにこれがストールする:

説明:

次のシーケンスを考えてみます:
1)システム MMR 書き込みがストールさせられます。
2)システム MMR 書き込みがストールさせられている間(書き込みが停止)、割り込み/例外が発生します。
3)割り込み/例外サービス・ルーチンが SSYNC 命令を実行します。

このアノーマリが発生するためには、プログラム・フローの変化により、実行パイプラインの特定の 1 ステージで書き込みが停止させられる必要があります。この場合、このアノーマリのために、MMR ロジックが停止させられたシステム MMR アクセスがまだ有効であると見なすようになります。このため、SSYNC がプロセッサを無限に、または高い優先順位からの割り込みまたはイベントにより、割り込まれるまでストールさせられます。

同様に、システム MMR 書き込みが条件付き分岐のような命令自体により停止させられる場合、ストア・バッファがフルで、低速な外部メモリへ出力しているときに無限ストールが発生します。

```
cc = r0 == r0;      // always true
if cc jump skip;
W[p0] = r1.l;       // System MMR access is fetched and killed
skip: ...
```

注:ユーザがデバッグ・ツールを使ってプロセッサをハンドラ内で停止させようとすると、無限ストールによりエミュレーション・イベントもロックアウトされます。

対策:

対策は、アプリケーションに副作用を与えない別のシステム MMR アクセス停止により、MMR ロジックをリセットすることです。例えば、CHIPID レジスタからの読み出しを使います。各割り込み/例外ハンドラの開始に実行される次のコードは、このアノーマリの対策になります:

```
cc = r0 == r0;      // always true
p0.h = 0xffc0;      // System MMR space CHIPID
p0.l = 0x0014;
if cc jump skip;    // always skip MMR access, but MMR access is fetched and killed
r0 = [p0];          // bogus System MMR read to work around the anomaly
skip: ...           // continue with handler code
```

条件付き分岐により MMR 書き込みが停止されるケースでは、2 個の NOP または他の非 MMR 命令を条件付き分岐の直後のロケーションへ挿入することで十分です。

注:デバッグ・セッションでのロックアップを防止するためには、ハンドラ・コード内でプロセッサを停止させることが必要な場合、必ず上記コードの後ろにブレーク・ポイントを設定してください。

適用レビジョン:

0.3、0.5

74. 05000287—ある条件下で読み出しを行うと不正なデータが返される

説明:

特定の条件下で、読み出しを行うと不正なデータが返されます。

この問題は、ポート B を介する 2 つの書き込みが、ポート A を介して行われる読み出しを挟んで、行われたときに発生します。問題が発生するためには、次の条件を満たす必要があります。

1) データ・キャッシュがイネーブルされる

2) 非キャッシュ可能読み出しが、2 つのキャッシュ可能書き込みの間で行われる

次のコードにこの問題の発生を示します。

```
p4.h = 0xfeb0;      //P4 and P5 point to 2 cacheable locations
p4.l = 0x2000;
p5.h = 0xfeb0;
p5.l = 0x2002;

w[p5] = r0;         // uses byte enable for the upper half of the 32-bit word
w[p4] = r0;         // uses byte enable for the lower half of the 32-bit word nop;
nop;
i1.l = 0x0102;      // i1 points to non-cacheable location
i1.h = 0x2400;
mnop || i0 += 2 || r2.l = w[i1]; // problem occurs HERE
```

この場合は、読み出しによりレジスタの下位半分からのデータが返されますが、代わりに、上位半分が返されます。

この問題は、リンク命令 (複数回の 32 ビット書き込みを実行) または 2 つの 32 ビット書き込みの後にも発生します。

注: Blackfin プロセッサでは、SSYNC または CSYNC が書き込みの後でかつ読み出しの前に使用されない限り、読み出しの方が書き込みより優先されます。

対策:

この問題には次の 3 つの対策があります:

1) データ・キャッシュをディスエーブルします。
または

2) DAG0 と DAG1 のポート・プリファレンスがポート B に設定されるように、DMEM_CONTROL レジスタを変更します。
または、

3) 書き込みの後に CSYNC を追加します。

適用レビジョン:

0.3

75. 05000288—FIFO が満杯になると SPORT が不正なデータを受信する:

説明:

SPORT が次のように設定されると、不正なデータを受信します:

1)セカンダリ受信データをイネーブル(RXSE=1)、すなわちワード長> 16 ビット。
さらに、

2) RX FIFO に 8 ワードのデータが詰まっている。
さらに、

3)さらに 1 ワードが SPORT に入力される。

この場合、データを保持する余裕が残っているため、オーバーフローがアサートされません。FIFO からデータが取り出されることなく次のデータが受信されると、オーバーフローがアサートされます。

このアノーマリにより、セカンダリ・データ(RxSEC=1)の代わりにプライマリ・データが受信されるかまたはワードガスワップ（SLEN>0xF）します。後続のワードは正常に受信されます。

対策:

問題の説明で示した状態を回避してください。

FIFO オーバーフローの近くで動作させないようにしてください。

適用レビジョン:

0.3

76. 05000301—メモリーメモリ間 DMA のソース/ディステネーション・ディスクリプタは同じメモリ空間にある必要がある:

説明:

MemDMA のソースとディステネーション・ディスクリプタが異なるメモリ空間(1 つは内部メモリ、他方は外部メモリ)にあり、かつトラフィック制御がターンオンされている場合、フェッチ中のディスクリプタ・ワードのソース・ディスクリプタ・カウントが、カレント・ディステネーション・ディスクリプタ・カウント(元のソース・ディスクリプタ・カウントに一致しないことがあります)の値により破壊されることがあります。このために、ソースがフェッチするディスクリプタ・エレメント数が意図した値に一致しなくなります。

これにより発生する 1 つの結果として、ディスクリプタの幾つかのエレメントがロードされなくなります。別の可能な結果としては、余分なディスクリプタ・エレメントのフェッチが行われることがあります。余分なフェッチ数が多いとき、先頭の数レジスタ(例えば、次のディスクリプタ・ポインタ)で正常データが不正データにより上書きされる場合、ディスクリプタ・エレメント・ポインタもオーバーフローして、レジスタの設定された開始点に戻ります。この最後のケースでは、DMA が無効なポインタをフェッチしたとき、次のディスクリプタをフェッチするまで DMA がエラーしたように見えません。

対策:

ソースとディステネーションのディスクリプタを同じメモリ空間に配置してください。両方とも、外部メモリまたは内部メモリに配置する必要があります。

適用レビジョン:

0.3、0.5

77. 05000302—DMA MMR レジスタに対する書き込みの後の SSYNC が正しく処理されない:

説明:

DMA チャンネルがメモリからディスクリプタをフェッチする許可を得ているとき、SSYNC 命令の有無に関係なく、同じ DMA コントローラに対応するシステム MMR に対する書き込みがディスクリプタのフェッチ完了まで遅延させられます。

この動作による 1 つの悪影響は、ISR コードが正しいシーケンスを実行して割り込み要求をクリアする DMA 割り込みの場合です:

```
p0.h = hi(DMA3_IRQ_STATUS);
p0.l = lo(DMA3_IRQ_STATUS);
r0.l = 0x0001;
w[p0] = r0.l;                   // Write-1-to-Clear Interrupt Request
ssync;                          // Allow write to complete rti;
```

同じ DMA コントローラの別の DMA チャンネルが書き込みのときに、ディスクリプタをフェッチ中である場合、この書き込みが遅延されるため、この遅延が後続の SSYNC 命令の継続時間を超えると、ISR コードが RTI 命令を実行して、再度 ISR が起動されます。これは、DMAx_IRQ_STATUS ビットがまだクリアされていないために発生します。この動作は、ディスクリプタ・フェッチでビジー中の DMA コントローラに関係するすべてのシステム MMR に対する書き込みに対しても当てはまります。

対策:

MMR レジスタからのダミー読み出しが SSYNC の前に挿入されると、これにより、前の書き込みが完了した後に読み出しを実行できることが保証されます。例えば、上の例を使用して、IRQ ステータス・レジスタを書き込んだ後にこのレジスタをリードバックします:

```
p0.h = hi(DMA3_IRQ_STATUS);
p0.l = lo(DMA3_IRQ_STATUS);
r0.l = 0x0001; w[p0] = r0.l;    // Write-1-to-Clear Interrupt Request
r0.l = w[p0];                   // Insert dummy read before ssync
ssync;                          // Allow write to complete
rti;
```

適用レビジョン:

0.3、0.5

78. 05000305—PLL_CTL レジスタの SPORT_HYS ビットが機能しない:

説明:

PLL_CTL レジスタの SPORT ヒステリシス・ビット(SPORT_HYS、ビット 15)が機能しません。このビットを読み出すと常に 0 が返され、1 を書き込むと無視されます。

対策:

なし。

適用レビジョン:

0.3

79. 05000307—ハイバネート中に SCKELOW ビットが状態を維持しない:

説明:

VR_CTL の SCKELOW ビット(ビット 15)がハイバネート・リセット・シーケンス中にステータスを維持しないため、ブート・シーケンスでこれを読み出して、プロセッサがコールド・スタートしたのか、またはハイバネート状態から抜け出たのか調べることができません。

対策:

ハイバネート機能を使用している場合は、ハイバネートに入る前にアプリケーション・コードにより VR_CTL を外部メモリの特定ロケーションへコピーすることができます。初期化ブロックでこれを調べると、コンテキストの待避が事前に行われた否かを知ることができます。例えば、ソース・コード内に 32 ビットのデータ・エレメントを生成して、LDF を使って、特定のロケーションへ待避させます。LDF 内に:

```
RESOLVE(32BitDataLabel, 32BIT_ADDR_IN_EXTERNAL_SDRAM_DATA_SEGMENT);
```

次に、ソース・コード内でハイバネートに入る準備をしているとき、この疑似コードを次のように使うことができます:

```
#define VR_CTL_HIBERNATE_VALUE 0x000080DC                  // Your value for VR_CTL goes here

PTR_TO_32BitDataLabel = VR_CTL_HIBERNATE_VALUE;            // 32-Bit Access
VR_CTL = VR_CTL_HIBERNATE_VALUE;                           // 16-Bit Access

CLI/IDLE PLL Programming Sequence;                         // Latch write to VR_CTL
```

次に、初期化ブロックで、この疑似コードを使ってこの 32 ビット値を調べて、このロケーションが前に書き込まれたか否かを知ることができます:

```
Read PTR_TO_32BitDataLabel;
Compare to VR_CTL_HIBERNATE_VALUE;

if TRUE

  Execute post-hibernate boot sequence;

else

  Perform full boot;
```

適用レビジョン:

0.3

80. 05000310—予約済みメモリの境界でのフェッチにより、偽ハードウェア・エラーが発生する:

説明:

予約済みメモリ、または有効な CPLB でカバーされている L1 命令キャッシュ・メモリ(命令キャッシュがイネーブルされている場合)の境界でのフェッチにより、偽ハードウェア・エラーが発生します(外部メモリ・アドレシング・エラー)。

対策:

1)ページ境界に分岐命令またはデータを配置しないでください。境界の前では、予約済みメモリ空間を使い少なくとも 76 バイトを空けてください。これにより偽例外の発生を防止することができます。

2)例外が有効か否かをアクションの前に判断する例外処理を設けてください。これは、CODE_FAULT_ADDR (または DATA_FAULT_ADDR)レジスタ値が有効ページ内のアドレスであることを確認することにより行うことができます。この場合、アクションは不要です。

このアノーマリは、L1_code_cache (命令キャッシュがイネーブルされている場合)の境界でも発生することに注意してください。

適用レビジョン:

0.3、0.5

81. 05000312—SSYNC、CSYNC、または LT、LB、LC レジスタへのロードが割り込みを受けるとエラーが発生する:

説明:

命令キャッシュがイネーブルされた場合、次の命令に割り込まれると、無効なコードが実行されます:

```
CSYNC
SSYNC
LCx =
LTx = (only when LCx is non-zero)
LBx = (only when LCx is non-zero)
```

この問題が発生すると、不正命令例外などの様々な不具合が発生します。その他にも、例外、ハードウェア・エラー、または有効な命令ではあるが予期したものと異なる命令として表示されます。

対策:

すべての **SSYNC**、**CSYNC**、"**LCx =**"、"**LTx =**"、 "**LBx =**" 命令の前に **cli** を追加して割り込みをディスエーブルし、さらにこれらの各命令の後ろに **sti** を配置して割り込みを再イネーブルします。既に割り込みが禁止されているコード内でこれらの命令が実行されると、この問題は発生しません。

割り込みのネスティングをイネーブルする割り込みサービス・ルーチンでは、必ず LCx、LTx、LBx の各レジスタをプッシュした後に RETI をプッシュして、割り込みのネスティンクをイネーブルしてください。ISR コンテキストの回復時に逆を行うと、RETI をポップした後にループ・レジスタがロードされることが保証されるため、割り込みのネストがディスエーブルされて、このアノーマリ状況からロードが保護されます。例えば、

```
INT_HANDLER:
[--sp] = astat;
[--sp] = lc0;       // push loop registers before pushing RETI
[--sp] = lt0;
[--sp] = lb0;
[--sp] = lc1;
[--sp] = lt1;
[--sp] = lb1;
[--sp] = reti;      // push RETI to enable nested interrupts
[--sp] = ...
       // body of interrupt handler
... = [sp++];
reti = [sp++];      // pop RETI to disable interrupts
lb1 = [sp++];       // it is now safe to load the loop registers
lt1 = [sp++];
lc1 = [sp++];
lb0 = [sp++];
lt0 = [sp++];
lc0 = [sp++];
astat = [sp++];
```

最後に、この対策はスーパーバイザ・モード命令を使って割り込みをディスエーブルおよびイネーブルするため、ユーザ・モードには適用されません。ユーザ領域では、CSYNC 命令または SSYNC 命令を使用しないでください。また、ループ・レジスタを直接ロードしないでください。その代わり、**LSETUP** 命令を使って構成できるハードウェア・ループを使ってください。これにより、ループ範囲が 2046 バイトに制限されます。

適用レビジョン:

0.3、0.5

82. 05000313—シングル・フレーム同期モードの最初の転送で PPI がレベル検出になる:

説明:

PPI をシングル外部フレーム同期でトリガーするように設定すると、最初の転送を除くすべての転送で、フレーム同期のエッジが必要とされます。最初の転送でのみ、フレーム同期入力がレベル検出されます。フレーム同期がアクティブ状態にあると、このために PPI が転送を開始します。このため、PPI が早めに開始されてしまいます。

このアノーマリは、PPI が 2 または 3 フレーム同期を使用する場合には適用されません。

対策:

PPI でシングル外部フレーム同期を使用するとき、PPI がイネーブルされたときにフレーム同期が非アクティブ状態となるようにしてください。

適用レビジョン:

0.3、0.5

83. 05000315—次のシステム MMR アクセスでシステム MMR 書き込みの停止が異常完了する:

説明:

次のシーケンスを考えてみます:
1)システム MMR 書き込みがストールさせられます。
2)システム MMR 書き込みがストールさせられている間(書き込みが停止)、割り込み/例外が発生します。
3)割り込み/例外サービス・ルーチンが任意の MMR をアクセス(読み出しまたは書き込み)します。

このアノーマリが発生するためには、プログラム・フローの変化により、実行パイプラインの特定の 1 ステージで書き込みが停止させられる必要があります。この場合、このアノーマリのために、MMR ロジックが停止させられたシステム MMR アクセスがまだ有効であると見なすようになります。ハンドラ内でのシステム MMR に対する次のアクセス(読み出し/書き込み)により、前にストールした書き込みが異常完了させられます。

同様に、システム MMR 書き込みが条件付き分岐のような命令自体により停止させられる場合、ストア・バッファがフルで、低速な外部メモリへ出力しているときに異常書き込みが発生します。

```
cc = r0 == r0;      // always true
if cc jump skip;
W[p0] = r1.l;       // System MMR access is fetched and killed
skip: ...
```

注:デバッグ・ツールを使った次のシステム MMR アクセスの前に、ハンドラ内でプロセッサが停止した場合、プロセッサは書き込みの完了を待つため無限にストールして、エミュレート・イベントがロックアウトされます。

対策:

対策は、分岐のシャドウ内の別のシステム MMR アクセス停止により、MMR ロジックをリセットすることです。例えば、システム MMR CHIPID レジスタからの読み出しをセットアップして、その後それを停止させると、システムに他の副作用を与えないアクセス停止が生成されます。したがって、各ハンドラの開始に実行される次のコードは、このアノーマリの対策になります:

```
cc = r0 == r0;      // always true
p0.h = 0xffc0;      // System MMR space CHIPID
p0.l = 0x0014;
if cc jump skip;    // always skip System MMR access, but it is fetched and killed
r0 = [p0];          // bogus System MMR read to work around the anomaly
skip: ...           // continue with handler code
```

条件付き分岐によりシステム MMR 書き込みが停止されるケースでは、2 個の NOP または他の非 MMR 命令を条件付き分岐の直後のロケーションへ挿入することで十分です。

注:デバッグ・セッションでのロックアップを防止するためには、ハンドラ・コード内でプロセッサを停止させることが必要な場合、必ず上記コードの後ろに所望のブレーク・ポイントを設定してください。

適用レビジョン:

0.3、0.5

84. 05000320— SPI マスタ・ブートの後、PF2 出力がアサートされたままになる

説明:

マスタ SPI ブート・モードの場合、ブート ROM は FIO0_DIR レジスタに書き込みを行って PF2 を出力としてイネーブルし、 FIO0_FLAG_S レジスタと FIO0_FLAG_C レジスタを使って PF2 に出力される SPI チップ・セレクトを制御して、SPI スレーブを制御します。ブートが完了すると、PF2 が出力としてイネーブルされ、ハイ・レベルに駆動されます。

対策:

このブート・モードを使う場合は、必要に応じて FIO0 レジスタを正しくリセットしてください。ブート完了後に PF2 を別の用途に使う場合は、FIO0_DIR と FIO0_FLAG_D を共に 0x0000 にリセットする必要があります。

適用レビジョン:

0.5

85. 05000323—特定の条件下で GPIO フラグ・ピンが誤動作する

説明:

フラグ IO システム MMR (FIOx_ プレフィックスを持つ任意のレジスタ)に対するアクセスの後ろに、別のシステム MMR アクセス(フラグ IO ブロック外)が続くと、フラグの入力ドライバが短時間アクティブになります。このため、フラグ・ラッチに保持されている出力値が誤ってクリアされて、出力ピンの状態に不要な変化が発生します。

MMR アクセスのある組み合わせでのみこの不具合が発生し、フラグ・レジスタ・アクセスのタイプ(書き込み、読み出し、読み出しアボート)により不具合が変わります。幾つかの不具合は非常に希に発生するため、システム評価時に検出されない可能性があります。このため、アドレス・ビット 4、5、または 6 が GPIO レジスタと続いてアクセスされる MMR との間で異なっている任意の MMR の組み合わせでこの不具合が発生するものと見なされています。さらに、この問題が発生するためには、連続して 2 回の MMR アクセスが起こる必要はありません。アクセスは、無制限のサイクル数/命令数で分離することができます。

複数の FIOx_ レジスタに対するアクセスは互いに妨げにならないため、入力として設定されたフラグ・ピンは影響を受けません。

対策:

フラグ・レジスタに対するアクセスの各シーケンスは、安全なレジスタの読み出しで終了する必要があります。"安全なレジスタ" とは、直前にアクセスされた FIOx レジスタと同じアドレス・ビット 4、5、6 を持つ任意の非 FIOx システム MMR と定義されます。対策では、この規則が条件付きジャンプや割り込みのような不連続なプログラム・フローにより乱されないようにする必要があります。歓迎される副作用として、FIOx MMR 読み出しアボートが完全に回避されます。例えば、次のシーケンスは安全です:

```
P5.H = hi(FIO0_FLAG_D); P5.L = lo(FIO0_FLAG_D);
P4.H = hi(SICA_SYSCR); P4.L = lo(SICA_SYSCR);
cli R7;             /* avoid interrupts */
nop; nop; nop;      /* three cycles after CLI before 1st FIO read access */
/* any FIOx_ sequence */
R6 = w[P5](z);
R5 = w[P5+FIO0_MASKA_D-FIO0_FLAG_D](z); /* FIO0_MASKA_D read */
R6 = R5 & R6;
w[P5+FIO0_FLAG_C-FIO0_FLAG_D] = R6; /* last FIOx_ access to FIO0_FLAG_C (0xFFC00704) */
R5 = w[P4](z);                      /* dummy read from SICA_SYSCR (0xFFC00104) */
sti R7;             /* restore interrupts */
```

SICA_SYSCR と FIO0_FLAG_C のアドレス・ビット 4、5、6 が b#000 であることに注意してください。このため、SICA_SYSCR 読み出しが、FIO0_FLAG_C のアクセスにより発生するこのクリティカルな状況を安全に解決します。各 FIOx レジスタの安全なレジスタのリストを次に示します。

If the last FIOx access was to...	Then a "safe register" is...
FIOx_FLAG_D/C/S/T	SICA_SYSCR
FIOx_MASKA_D/C/S/T	UART_SCR
FIOx_MASKB_D/C/S/T	UART_GCTL
FIOx_DIR/POLAR/EDGE/BOTH	SPORT0_STAT
FIOx_INEN	DMA1_1_CONFIG

インクルード・ファイル sys/05000323.h は VisualDSP++ 4.5 Update 6 と VisualDSP++ 5.0 Update 2 に添付されています。このファイルには、C/C++ による MMR の読み出しと書き込み用のマクロのグループが含まれており、これらを使うと読み出しまたは書き込みを安全に行うことができます。この対策を必要とするデバイスとシリコン・レビジョンのプロジェクトをビルドする際、コンパイル、アセンブル、リンクの各ステージでマクロ __WORKAROUND_FLAGS_MMR_ANOM_323 が定義されます。対策を手動でイネーブルするときは、-D__WORKAROUND_FLAGS_MMR_ANOM_323 スイッチを使ってください。詳細については、sys/05000323.h ファイルのコメントを参照してください。

適用レビジョン:

0.3、0.5

86. 05000326—ワード長 >16 ビットのとき SPORT セカンダリ受信チャンネルが機能しない

説明:

SPORT をワード長>16 のセカンダリ・モードに設定すると(SPORTx_RCR2 レジスタで、RXSE ビットをセットし、SLEN フィールド> 0x0F を設定)、SPORT レシーバがスタックして、受信 FIFO の DRxSEC ピンで受信された先頭ワードの上位 16 ビットをラッチします。FIFO 内の残りのスペースには、2 番目のプライマリ・ワードおよびセカンダリ・ワードのシリアル・データではなく、このデータが書き込まれまれ、オーバーフロー・エラーが発生します。

対策:

SPORT の受信セカンダリ側をイネーブルする際は、シリアル・ワード長>16 ビットを使用しないでください。

適用レビジョン:

0.5

87. 05000331—24 ビット SPI ブート・モードが機能しない

説明:

24 ビット SPI ブート・モードが機能しません。

対策:

16 ビット SPI ブート・モードは正しく動作します。したがって 16 ビット・アドレス指定可能な SPI デバイスを代わりに使用することができます。

適用レビジョン:

0.3

88. 05000332—スレーブ SPI ブート・モードが機能しない

説明:

スレーブ SPI ブート・モードが機能しません。

対策:

BMODE[1:0] コンフィギュレーション b#10 を使用しないでください。代わりに別のブート・モードを使ってください。

適用レビジョン:

0.3

89. 05000333—エッジ検出後の 1 SCLK サイクルでのフラグ・データ・レジスタ書き込みにより、割り込みステータスがクリアされる

説明:

フラグ・データ・レジスタへの書き込みが、エッジ検出割り込みでエッジが検出された 1 システム・クロック・サイクル後に発生した場合、セットされてから 1 システム・クロック・サイクル後にこのフラグ・ビットがクリアされます。

ビットが割り込みを発生するように設定されている場合は、割り込みが発生しますが、割り込みを発生させたビットが表示されません。コア・クロックが動作していないとき、または SIC_IMASKx ビットが割り込みをイネーブルするように設定されていないときは、割り込みが失われます。

フラグ・データ・レジスタに対する書き込みには、データ、クリア、セット、トグルが含まれます。

対策:

可能な場合、同じポートで異なる GPIO 機能の混在を回避してください。入力エッジ検出ピンは 1 ポート専用にすることができます。異なるポートを使って出力ピンをトグルすることにより、エッジ検出 SIC_ISRx レジスタまたはフラグ・レジスタに格納されている検出したエッジが上書き/クリアされるのを防止することができます。複数のエッジ検出入力ピンを 1 本の割り込みに割り当てる必要がある場合に、この対策を使うことができます。

1 つのエッジ検出ソースのみが 1 つの割り込みに割り当てられた場合、それが割り込み原因と見なされるので、SIC_ISRx レジスタと FIOx_FLAG レジスタの読み出し命令は不要です。イネーブルされると、すべての割り込みが正しく入力されることに注意してください。

別のオプションは、代わりにレベル検出割り込みを使うことです。割り込みサービス・ルーチンの再起動を防止して、次のエッジを検出するために、受信エッジの間で極性をトグルさせてください。2 つのエッジの間の遅延が割り込みサービス・ルーチンのサービスに十分であるか、または要求ラインに使用できる場合、この対策を使うことができます。極性のトグルは、両エッジを探す場合に使用することができますが、1 個だけのエッジの場合は、他方の割り込みを無視する必要があります。

適用レビジョン:

0.3

90. 05000339—PLL_CTL レジスタの ALT タイミング・ビットが機能しない

説明:

PLL_CTL レジスタの ALT_TIMING ビット (ビット 4)が機能しません。このビットは、PPI データがサンプルされるエッジ(PPI フレーム同期が基準)をソフトウェアから設定できるようにします。このビットを読み出すと常に 0 が返され、1 を書き込むと無視されます。

対策:

なし。

適用レビジョン:

0.3

91. 05000343—メモリ DMA FIFO により、外部メモリに対する書き込みのスループット低下が発生する

説明:

前のアノーマリに対処するために行ったデザイン変更のために、外部メモリに対するメモリ DMA 書き込み動作では、深さ 4 の FIFO ではなく、深さ 3 の FIFO を使用するようになりました。このため、外部メモリが内部メモリ・バッファから出力されるデータに追いつくことができる場合、メモリ DMA スループットは最大 25%低下します。出力側メモリ DMA FIFO の深さは、レビジョン 0.5 のシリコンでは 4 に戻されました。

この問題はメモリ DMA だけの問題であるため、ペリフェラル DMA チャンネルには適用されないことに注意してください。

対策:

なし。

適用レビジョン:

0.3

92. 05000357—チャンネル 0 をディスエーブルすると、シリアル・ポート(SPORT)マルチチャンネル送信に不具合が発生する:

説明:

チャンネル 0 をディスエーブルしてマルチチャンネル・モードを設定すると、次のすべての条件を満たすとき、DMA 送信データが正しくない SPORT チャンネルへ送信されます:

1) 外部受信フレーム同期 (SPORTx_RCR1 で IRFS = 0)

2) ウインドウ・オフセット = 0 (SPORTx_MCMC1 で WOFF = 0)

3) マルチチャンネル・フレーム遅延 = 0 (SPORTx_MCMC2 で MFD = 0)

4) DMA 送信パッキングをディスエーブル (SPORTx_MCMC2 で MCDTXPE = 0)

この特別な設定を使用したとき、チャンネル 0 がディスエーブルされていても、非パック・モードでチャンネル 0 プレースホルダ内にある内容は何であっても最初に送信されるため、マルチチャンネル送信データが壊されます。このため、出力ウインドウで 1 ワードのデータ・シフトが発生し、これがシリアル・ストリーム内の後続の各ウインドウで繰り返されます。例えば、チャンネル 0 がディスエーブルされ、チャンネル 1〜7 が送信用にイネーブルされて、非パック送信バッファが{0、1、2、3、4、5、6、7}であり、かつウインドウ・サイズが 8 チャンネルの場合、一連の出力ウインドウ内の予測されるデータ・シーケンスは次のようになります:

1234567--1234567--1234567--1234567

このアノーマリにより、出力は次のように見えます:

0123456--7012345--6701234--5670123

対策:

これに対しては幾つかの対策があります:

1) マルチチャンネル・モードをディスエーブルします。

2) 内部受信フレーム同期を使います。

3) マルチチャンネル・フレーム遅延 >0 を使います。

4) ウインドウ・オフセット >0 を使います。

5) DMA 送信パッキングをイネーブルします。

6) チャンネル 0 をディスエーブルしないようにします。

適用レビジョン:

0.3、0.5

93. 05000362—列アドレス幅の競合により SDRAM エラーが発生する

説明:

バンク 2 と 3 に対する列アドレス幅の設定が、SDRAM コントローラ内で誤配線されています。バンク 3 の列アドレス幅 (EB3CAW) の設定とバンク 2 の設定が一致しない場合、バンク 2 をアクセスするとエラーが発生します。

対策:

バンク 2 を使う場合、バンク 2 とバンク 3 が EBIU_SDBCTL レジスタで同じ列アドレス幅に設定されていることを確認してください。これはバンク 3 のイネーブルの有無に関係ありません。

適用レビジョン:

0.3、0.5

94. 05000363—UART ブレーク信号の問題:

説明:

ブレーク信号が受信されると、UART コントローラはエラー割り込みを 1 回発生する必要がありますが、ブレーク信号がアクティブとなる各ビット時間に対してエラー割り込みを発生して、コントローラが多くのエラー割り込みを発生してしまいます。 例えば、57600 のボー・レートで約 250ms 間ブレーク信号がラインをロー・レベルに維持すると、ストリーム内にスタート・ビットまたはストップ・ビットがないことと無関係に、約 1400 回のエラー割り込みが発生されます。 内部的には、ブレーク信号の間にサンプルされるデータはすべて 0 であるため、UART_RBR レジスタ内でデータは 0 として受信されます。

別の問題は、ブレーク信号自体のタイミングです。次の有効なキャラクタが送信されるタイミングに応じて、ブレーク信号の結果により次の有効なキャラクタが 2 つの無効なキャラクタに分割されます。これは受信される先頭ビットが前の不正データの後ろに追加され、有効なキャラクタの残りの部分が次のキャラクタとしてサンプルされるためです。ブレークの後にデータが連続ストリームとして続く場合、この動作は UART を介して受信される後続キャラクタのストリームで伝搬することがあります。

対策:

1 つのブレーク信号により発生する多くの割り込みに対して、ソフトウェア内で多くの割り込みを 1 つにまとめてサービスする以外に対策はありません。発生される各エラー割り込みは個別にサービスする必要があります。例えば、ソフトウェアではフラグを使ってこれを制御することができます。UART エラー割り込みハンドラがフラグをセットし、後続の割り込み要求をこのフラグがクリアされるまでスキップすると、複数のブレークを 1 つとして処理することができます。UART RX 割り込み内で同じフラグをクリアすることにより、ブレークが完了して、UART から新しい有効なデータが受信されたことを表示することができます。

ブレークの後に分割されるデータは、回復できません。ホストがブレークの後に次のデータの送信を 1 キャラクタ分のビット時間だけ遅延させると、Blackfin UART はブレーク信号から回復して、有効なデータの受信を再開できるようになります。例えば、1 スタート・ビット、1 パリティ・ビット、2 ストップ・ビットを持つ 8 ビット・データの場合、ホストはブレーク信号の後少なくとも 12 UART ビット時間待って、次の有効データを発行する必要があります。

適用レビジョン:

0.3

95. 05000366—ITU-R 656 モードで PPI アンダーフロー・エラーが検出されなくなる:

説明:

PPI ポートが ITU-R 656 出力モードに設定されると、PPI FIFO アンダーランが発生したとき、FIFO アンダーラン・ビット(PPI_STATUS の UNDR)セットされなくなります。帯域幅が十分でない場合、または調停による遅延のために PPI DMA がバス・アクセスの取得に失敗した場合に、アンダーランが発生します。

対策:

なし。

適用レビジョン:

0.3、0.5

96. 05000371—サブルーチンの継続時間が 5 サイクルを下回るとき、RETS レジスタが壊れる可能性がある:

説明:

RTS 命令がサブルーチンの開始から 4 実行サイクル以内に配置されると、正常にリターンしなくなることがあります。例えば、

```
CALL STUB_CODE;
...
...
...
STUB_CODE:
  RTS;
```

これが発生した場合、RETS 内のビットに不具合が発生すると、これによりプロセッサが誤ったアドレスに分岐して、無効なコードが実行されることがあります。

対策:

サブルーチン内で RTS の前に少なくとも 4 実行サイクルある場合、CALL 命令と RTS 命令がこの問題に遭遇するような方法で並ぶことはありません。NOP は 1 サイクル命令であるため、次に示す対策はすべての不具合ケースに対して有効です:

```
CALL STUB_CODE;

...
...
...
STUB_CODE:
  NOP; // These 4 NOPs can be any combination of instructions
  NOP; // that results in at least 4 core clock cycles.
  NOP;
  NOP;
  RTS;
```

分岐予測は、このシナリオの要因にはなりません。4 サイクル以内で RTS 命令に到達するサブルーチン内部の条件付きジャンプは、この不具合が発生する原因にはなりません。非同期イベント(割り込み、例外、NMI)も、この不具合を発生しません。

VisualDSP++ 4.5 更新 6 と VisualDSP++ 5.0 アップデート 2 から、ツールにはこのアノーマリの対策が含まれています。C/C++コンパイラの対策では、RTS の前に NOP 命令または無条件 JUMP 命令を挿入して stub 関数コードの生成を回避しています。2 個を超える NOP が必要とされる場合には、コード・サイズを最適化(-Os)するときに、JUMP 対策の派生が使用されます。このアノーマリを発生させるコードに対しては、検出して警告(ea5516)を発するようにアセンブラが変更されています。ランタイム・ライブラリと VDK サポート・ライブラリも、このアノーマリを変更するように修正されています。

VisualDSP++では、影響を受けるプロセッサに対してビルドするとき、これらの対策が自動的にイネーブルされます。コンパイラの対策は、-workaround avoid-quick-rts-371 スイッチを使って手動でイネーブルすることができます。アセンブラの警告は、-anomaly-detect 05000371 スイッチを使って制御されます。この対策をイネーブルすると、_WORKAROUND_AVOID_QUICK_RTS_371 マクロが、コンパイル、アセンブル、リンクのステージで定義されます。

適用レビジョン:

0.3、0.5

97. 05000403—レベル検出の外部 GPIO ウェイクアップにより無限ストールが発生する:

説明:

レベル検出の GPIO イベントを使ってプロセッサを低消費電力のスリープ動作モードからウェイクアップさせる場合、ウェイクアップ・パルス幅が狭すぎるとき、プロセッサが無限にストールすることがあります。これが発生した場合、レベルが GPIO ピンで検出されたことにより PLL がスリープ・モードから遷移し始めますが、コアがアイドル状態から抜け出して実行を再開する十分な時間を確保する前にそのトリガー・レベルがなくなると、スリープ・モードに戻ってしまいます。

このため、プロセッサは正常にウェイクアップしません。この時点ではハードウェア・リセットでのみこのストール状態から抜け出すことができます。

対策:

このアノーマリを回避する方法は 2 つあります:

1)ウェイクアップ・イベントの発生に使用するピンにエッジ検出機能を使います。

2)ウェイクアップ信号のエッジのノイズを除去して、少なくとも 3 システム・クロック(SCLK)サイクル間トリガー・レベルを維持します。

適用レビジョン:

0.3、0.5

98. 05000412—ライトバック・データ・キャッシュをイネーブルしたとき、TESTSET 命令によりデータが壊れる

説明:

TESTSET 命令を使って L2 メモリ上で動作し、かつライトバック・モードを使ってキャッシュされた外部メモリ内にデータが存在する場合、外部メモリおよび/または L2 メモリ内にあるデータが壊されることがあります。この問題は両コアで発生します。

この問題が発生するためには、片方のコアが TESTSET とキャッシュ・アクセスの両方を開始する必要があります。一方のコアが TESTSET を発行し、他方のコアがキャッシュ・ライン要求を発行する場合には、この問題は発生しません。同様に、TESTSET とキャッシュ・アクセスが同じメモリ空間 (L2/L2 または外部/外部)を対象とする場合にも、この問題は発生しません。

問題が発生した場合には、L2 および/または外部メモリ内のデータが壊されます。

対策:

ライトバック・キャッシュを使用しないか、または SSYNC 命令を使って TESTSET 命令を実行してください。SSYNC 命令を使って TESTSET 命令を日こうするときは、次を行ってください。

```
    CLI R0;
    R1 = [P0]; /* If the address that P0 points to is not covered by an installed CPLB,
perform a dummy read to make sure CPLB is installed */
    NOP;
    NOP;
    SSYNC;
    TESTSET (P0);
    STI R0;
```

TESTSET 命令を使って、ライトバック・データと同じ空間にある非キャッシュ可能ロケーションで動作する場合にも、この問題が回避されます。例えば、TESTSET 命令が非キャッシュ可能外部メモリで動作し、ライトバック・キャッシュ可能ロケーションも外部メモリ内にある場合です。

適用レビジョン:

0.3、0.5

99. 05000416—投機的フェッチにより、不要な外部 FIFO 動作が発生する:

説明:

外部 FIFO デバイスが同期メモリ・バンクに接続されると、プロセッサが投機的にメモリ・アクセスを行うことができるので、不正な動作が発生します。これは、FIFO がデータを Blackfin に提供するため、フェッチが投機的に行われるごとに、あるいは投機的アクセスがキャンセルされたとき、データが失われるためです。"投機的"フェッチとは、パイプライン内で起動され、完了前に停止される読み出しです。これは、フローの変更(割り込みまたは例外など)の際に、または分岐のシャドウをアクセスする際に発生します。この動作は、Blackfin プログラマズ・リファレンスで説明されています。

発生する別のケースとしては、フロー変更が例外から発生するようなハードウェア・ループの一部としてアクセスが実行されるケースがあります。例外はディスエーブルできないため、割り込みをディスエーブルした場合でも、例外から投機的フェッチが発生する例を次に示します:

```
CLI R3;                                  /* Disable Interrupts */
LSETUP( loop_s, loop_e) LC0 = P2;
        loop_s: R0 = W[P0];              /* Read from a FIFO Device */
        loop_e: W[P1++] = R0;            /* Write that Generates a Data CPLB Page Miss */
STI R3;                                  /* Enable Interrupts */
RTS;
```

この例では、ハードウェア・ループ内での読み出しが、割り込みをディスエーブルして FIFO に対して行われています。ループ内での書き込みからデータ CPLB 例外が発生すると、ループ内での読み出しが投機的に行われます。

対策:

まず、コア読み出しによりアクセスを行う場合、コア読み出しを行う前に割り込みをターンオフさせます。次に、読み出し命令が見えないように、割り込みがターンオフされる前にパイプラインの読み出しフェーズを保護する必要があります。:

```
CLI R0;
NOP; NOP; NOP; /* Can Be Any 3 Instructions */
R1 = [P0];
STI R0;
```

例外から同じ不要な動作が発生しないように保護するため、読み出しをフローの変更から離す必要があります:

```
CLI R3;                          /* Disable Interrupts */
LSETUP( loop_s, loop_e) LC0 = P2;
        loop_s: NOP;             /* 2 NOPs to Pad Read */
                NOP;
                R0 = W[P0];
        loop_e: W[P1++] = R0;
STI R3;                          /* Enable Interrupts */
RTS;
```

ループを構成して、最後に NOP のパディングを配置することができます:

```
LSETUP( .Lword_loop_s, .Lword_loop_e) LC0 = P2;
        .Lword_loop_s: R0 = W[P0];
                W[P1++] = R0;
                NOP;             /* 2 NOPs to Pad Read */
        .Lword_loop_e: NOP;
```

これらの両シーケンスは、フロー変更から読み出しが投機的に実行されることを防止します。挿入された 2 個の NOP は、投機的アクセスを防止するため、パイプライン内に十分な分離を提供します。これらの NOP は、任意の 2 個の命令にすることができます。

DMA 転送を使って行う読み出しは、投機的アクセスから保護する必要はありません。

適用レビジョン:

0.3、0.5

100. 05000425—特定の設定でマルチチャンネル SPORT チャンネルがずれる:

説明:

マルチチャンネル・モードでシリアル・ポートを使う場合、SPORT に非常に特別な設定が行われると、送信チャンネルと受信チャンネルがずれます:

1)ウインドウ・オフセット(WOFF) = 0。

2)フレーム遅延(MFD) > 0。

3)ウインドウ・サイズが 8 の奇数倍(すなわち WSIZE が偶数)。

4) RFS パルス間の間隔がウインドウ継続時間に一致する。

この設定を使用すると、最初のウインドウが完了したとき、マルチチャンネル・モード・チャンネル・イネーブル・レジスタが誤ラッチされるために、不正チャンネル割り当てに従って TDV 信号が駆動され、不正なチャンネルで受信データがサンプルされます。したがって、最初のウインドウは正常に送受信されますが、2 番目以後のすべてのウインドウがずれてしまい、送受信されたデータが壊れます。

このエラーは、外部クロック、内部クロック、RFS に対して発生します。

対策:

幾つかの対策が可能です:

1) 0 以外のウインドウ・オフセットを使います。

2) 0 のフレーム遅延を使います。

3) 8 の偶数倍のウインドウ・サイズを使います。

4)内部 RFS の場合、SPORTx_RFSDIV が少なくともウインドウ・サイズ(イネーブルされるチャンネル数* SLEN)に一致することを確認します。

適用レビジョン:

0.3、0.5

101. 05000426—間接ポインタ命令の投機的フェッチにより、偽ハードウェア・エラーが発生する:

説明:

間接ジャンプまたは条件付きジヤンプが採用したパスと反対側のコントロール・フロー上で予約済みメモリまたは不正メモリを指すポインタを使うコールがあると、偽ハードウェア・エラーが発生します。これは、一般に無効な関数ポインタ(例えば-1 を設定)を使う場合に発生します。例えば、

```
    CC = P2 == -0x1;
    IF CC JUMP skip;
    CALL (P2);
  skip:
    RETS;
```

IF CC JUMP 命令をコミットできる前に、パイプラインがアドレス-1 (0xffffffff)に対する投機的命令フェッチを発行するため、偽ハードウェア・エラーが発生します。これは、オフェンデイング命令が実際に実行されないための偽ハードウェア・エラーです。これは、条件付き分岐(不採用が予測される)の 2 つの命令内で次のようにポインタが使用されると発生します:

```
    BRCC X [predicted not taken]
    Y: JUMP (P-reg);    // If either of these two p-regs describe non-existent
      CALL (P-reg);     // memory, such as external SDRAM when the SDRAM
    X: RETS;            // controller is off, then a hardware error will result.
```

対策:

If 命令キャッシュがオンであるか、または ICPLB がイネーブルされる場合、このアノーマリは適用されません。

命令キャッシュがオフであり、ICPLB がディスエーブルされる場合、間接ポインタ命令は分岐命令から 2 命令離れる必要があります。これは NOP を使って実現することができます:

```
    BRCC X [predicted not taken]
    Y: NOP;             // These two NOPs will properly pad the indirect pointer
      NOP;              // used in the next line.
      JUMP (P-reg);
      CALL (P-reg);
    X: RETS;
```

適用レビジョン:

0.3、0.5

102. 05000428—コア B からの投機的 L2 メモリ読み出しの後に L2/L3 メモリ書き込みが喪失/破壊される

説明:

この問題は、割り込みまたは例外により投機的読み出しアクセスが発生する場合には発生せず、コア B からアクセスを実行した場合にのみ発生します。

内部 L2 メモリまたは外部 L3 メモリに対する書き込みの後に内部 L2 メモリからの投機的読み出しが続く場合、この書き込みが喪失/破壊されることがあります。L2/L3 メモリに対する書き込みが喪失/破壊されるためには、投機的に行われたロードと同じ L2 アドレスに対する書き込みである必要はありません。このアノーマリが発生するためには、分岐のシャドウ内での読み出しから投機的読み出しが発生する必要があります。アクセスは連続するアクセスである必要がないため、投機的読み出しと書き込みの間に複数の命令が存在するときにもこの問題は発生します。

"分岐が発生しない" 場合:

```
BRCC X [predicted not taken]
R0 = [P0];          // If any of these three loads accesses L2 memory from Core
R1 = [P1];          // B, speculative execution in the pipeline causes the
R2 = [P2];          // anomaly trigger condition.
X: ...              // Any number of instructions...
[P3] = R0;          // This write can be corrupted or lost, if write is to L2/L3
```

"分岐が発生する" 場合:

```
BRCC X (bp) [predicted taken]
...                 // Predicted Jumped Instructions
X: NOP;             // Added NOP at JUMP target takes the read out of the shadow
R0 = [P0];          // of the branch, thereby avoiding the anomaly trigger condition.
...                 // Any number of instructions...
[P1] = R0;          // This L2/L3 write is now safe.
```

対策:

分岐のシャドウ内で L2 読み出しを行わないでください。可能な場合、読み出しをジャンプの前に移動してください。これが不可能な場合は、読み出しを分岐のシャドウを超えるように遅延させてください。

"分岐が発生しない" 場合には、分岐命令と読み出し命令との間に任意の 3 個の命令を追加して、読み出しが投機的に実行されないようにしてください。

```
BRCC X [predicted not taken]
NOP; NOP; NOP;      // Pad the read with 3 instructions
R0 = [P0];          // Even if these reads access L2 memory from Core B, they
R1 = [P1];          // will not be speculatively executed, therefore avoiding
R2 = [P2];          // the anomaly trigger condition.
X: ...              // Any number of instructions...
[P3] = R0;          // This L2/L3 write is now safe.
```

"分岐が発生する" 場合には、投機的読み出しが分岐のターゲットになることはできません:

```
BRCC X (bp) [predicted taken]
...                 // Predicted Jumped Instructions
X: NOP;             // Added NOP at JUMP target takes the read out of the shadow
R0 = [P0];          // of the branch, thereby avoiding the anomaly trigger condition.
...                 // Any number of instructions...
[P1] = R0;          // This L2/L3 write is now safe.
```

適用レビジョン:

0.5

103. 05000443—ハードウェア・ループの終わりにある IFLUSH 命令により無限ストールが発生する

説明:

IFLUSH 命令がループの終わりに配置されると、プロセッサが無限にストールします。例えば、次の 2 つのコード例ではループから抜け出すことはできません。

```
P1 = 2;
LSETUP (1f, 2f) LC1 = P1;
1: NOP;
2: IFLUSH[P0++];
```

```
LSETUP (1f, 2f) LC1 = P1;
1: NOP; NOP; NOP; NOP;          // Any number of instructions...
2: IFLUSH[P0++];
```

対策:

ハードウェア・ループの終わりに IFLUSH 命令を配置しないでください。ループの終わりの IFLUSH を任意の命令で置き換えると、次のように問題が回避されます:

```
LSETUP (1f, 2f) LC1 = P1;
1: IFLUSH[P0++];
2: NOP;                         // Pad the loop end
```

適用レビジョン:

0.3、0.5