ANALOG DEVICES
Media Platforms and Services Group

Blackfin Anomaly list

for Revision(s) ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

Created Tue Nov 2 10:28:07 2004 (IM)

These anomalies represent the currently known differences between revisions of Blackfin devices and the functionality specified in the Blackfin data sheets and User's Manual. A revision number with the form "-x.x" is branded on all parts.

Bits 31:28 of the BF535's CHIP ID Register (CHIPID) differentiates between pre-and-post 1.0 revisions. For revisions prior to 1.0, bits 31:28 of the CHIPID register will read a value of 0. For revisions 1.0 and later, bits 31:28 of the CHIPID register will read a value of 1.

Revision	CHIPID[31:28]
0.2	0000
1.0, 1.1, 1.2, 1.3	0001

Index	ID	Anomaly Summary	ADSP-BF535-0.2	ADSP-BF535-1.0	ADSP-BF535-1.1	ADSP-BF535-1.2	ADSP-BF535-1.3
1	05000005	The Scratchpad SRAM can only be accessed using DAG0	x	x	x	x	x
2	05000006	The ILAT register cannot be cleared without servicing an interrupt	x	x	x	x	x
3	05000009	The AZ bit of the ASTAT register is undefined when using RND	x	x	x	x	x

Index	ID	Anomaly Summary	ADSP-BF535-0.2	ADSP-BF535-1.0	ADSP-BF535-1.1	ADSP-BF535-1.2	ADSP-BF535-1.3
4	05000010	SIGNBITS operation returns incorrect value	x	x	x	x	x
5	05000013	SPORT: Improper behavior when RFS/TFS is active low in the late framing mode	x	.	.	.	.
6	05000015	Core current levels are higher than necessary at 300MHz	x	.	.	.	.
7	05000016	PCI: Extra BAR in Config Space	x	.	.	.	.
8	05000018	System Interrupt Controller Interrupt Mask Register polarity is different than the polarity used in IMASK	x	.	.	.	.
9	05000019	Undefined ASTAT bits	x	x	x	x	x
10	05000023	Core Timer counts during emulation mode in revision 0.2 silicon	x	.	.	.	.
11	05000024	Multi-issue instructions that store to L1 data memory and load from scratchpad do not work properly	x	x	x	x	x

Index	ID	Anomaly Summary	ADSP-BF535-0.2	ADSP-BF535-1.0	ADSP-BF535-1.1	ADSP-BF535-1.2	ADSP-BF535-1.3
12	05000025	Tristate pins are outputs during power up	x	.	.	.	.
13	05000029	IrDA® functionality does not function correctly	x	.	.	.	.
14	05000035	UART THRE bit latency	x	x	x	x	x
15	05000036	Rev 0.2 and 1.0 silicon are not PC133 compliant	x	x	.	.	.
16	05000037	Writing and reading ILOC bits	x	x	x	x	x
17	05000038	Boundary Scan does not work in Rev 0.2 silicon	x	.	.	.	.
18	05000039	Icache Flush doesn't invalidate the Fill Buffer	x	x	x	x	x
19	05000040	PLL transitions incorrect coming out of reset	x	x	x	.	.
20	05000041	Icache bus error does not Flush the Fill Buffer	x	x	x	x	x
21	05000042	Core hangs on a specific condition	x	.	.	.	.
22	05000043	Sport Transmit Underflow (TUVF) special clearing instructions	x	x	x	x	x
23	05000044	Incorrect ASTAT flag	x	x	x	x	x

Index	ID	Anomaly Summary	ADSP-BF535-0.2	ADSP-BF535-1.0	ADSP-BF535-1.1	ADSP-BF535-1.2	ADSP-BF535-1.3
		operations					
24	05000046	Shifter does not consider all 32 bits when shift value is a D register	x	x	x	x	x
25	05000047	PCI register changes	x	.	.	.	.
26	05000048	Speculative fetch of illegal or undefined memory location causes HW error	x	x	x	x	x
27	05000051	Accessing MMRs with Data Cache disabled and CPLBs enabled	x	x	x	x	x
28	05000052	USB: On a HOST OUT (PC->ADSP-BF535) disarm is not functioning correctly	x	x	x	.	.
29	05000053	USB: EPLEN is not updating correctly at the end of a short packet	x	x	x	.	.
30	05000054	Problems with data cache writeback mode	x	x	x	x	x
31	05000055	Setting breakpoints in an emulator session with cache enabled	x	x	x	x	x
32	05000056	USB: ARM bit not clearing	x	x	x	.	.

Index	ID	Anomaly Summary	ADSP-BF535-0.2	ADSP-BF535-1.0	ADSP-BF535-1.1	ADSP-BF535-1.2	ADSP-BF535-1.3
		on short packets					
33	05000057	RTC integrity when core powered off	x	x	x	x	x
34	05000060	Serial clock is not gated by SPORT enable bit	x	x	x	x	x
35	05000061	1.38V minimum VDDcore required	x	x	x	.	.
36	05000062	SPORT Multichannel Mode starting and stopping	x	x	x	x	x
37	05000071	CLI instruction can allow an asynchronous interrupt occur as it is committing	x	x	x	x	x
38	05000164	Store to Load Forwarding in Write Through Mode	x	x	x	x	x
39	05000165	Data cache dirty bit set when a load-miss-fill is in progress	x	x	x	x	x
40	05000170	PLL clkout may be incorrect in multiplier mode coming out of reset	.	.	.	x	x
41	05000178	Rev 1.2 and 1.3 silicon do not meet TDCAD requirement of the PC133 spec	.	.	.	x	x
42	05000195	Latency in	x	x	x	x	x

Index	ID	Anomaly Summary	ADSP-BF535-0.2	ADSP-BF535-1.0	ADSP-BF535-1.1	ADSP-BF535-1.2	ADSP-BF535-1.3
		Writes to RND_MOD bit					
43	05000210	IDCODE register not JTAG IEEE 1149.1 compliant	x	x	x	x	x
Key: x = anomaly exists in revision . = Not applicable							

ANOMALIES

1. 05000005 - The Scratchpad SRAM can only be accessed using DAG0:

Description:
The ADSP-BF535 HW Reference indicates that the L1 Scratchpad memory can be accessed from either DAG0 or DAG1. On the ADSP-BF535, an attempted access by DAG1 to the Scratchpad SRAM will result in an exception.

Workaround:
The workaround is to avoid an access to this memory space by DAG1.
To avoid doing this, do not place any data into the L1 Scratchpad memory if it will be accessed by compiled code. In addition, anything placed in L1 Scratchpad memory must be declared as "volatile." For multi-issue assembly instructions of the form:

```
DSP32 || Load || Store

or

DSP32 || Load || Load
```

care should be taken to avoid the case where the Load/Store instructions both access the Scratchpad memory in a single instruction or any instruction where DAG1 makes an access to the scratchpad memory.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

2. 05000006 - The ILAT register cannot be cleared without servicing an interrupt:

Description:
The ADSP-BF535 HW Reference manual indicates that the ILAT register is Read/Write. In the current revisions of silicon, ILAT is read only.

Workaround:
The only way to clear ILAT is to actually service the interrupt and execute an RTI.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

3. 05000009 - The AZ bit of the ASTAT register is undefined when using RND:

Description:
The AZ bit of the ASTAT register is undefined after using the RND option in an instruction, e.g. R1.L=R0 (RND);

Workaround:
If the value of AZ is important to user code, the AZ bit must be checked prior to executing an instruction with the RND option.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

4. 05000010 - SIGNBITS operation returns incorrect value:

Description:
The SIGNBITS instruction returns the an incorrect value, e.g.

```
R0.L= SIGNBITS A0;
returns -8 thru 30 if A0=0x8000000000 thru A0=0x0000000001
returns 39 if A0=-1
returns 39 if A0=0
```

It should return 31 when A0=-1 and A0=0

Workaround:
The SIGNBITS instruction will produce incorrect results for the case described above. Care should be taken when using this instruction.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

5. 05000013 - SPORT: Improper behavior when RFS/TFS is active low in the late framing mode:

Description:
For SPORT0 and SPORT1:
1. With internal RFS/TFS, the RFS/TFS won't frame the data when RFS/TFS is active low, in late framing mode. An active high RFS/TFS will frame the data.

2. With internal RFS/TFS, the RFSDIV/TFSDIV must be set to a factor of 4 smaller than what would be expected to achieve the desired RFS/TFS frequency.

Workaround:
Do not use late framing mode with active low.

Applies to revision(s):
ADSP-BF535-0.2

6. 05000015 - Core current levels are higher than necessary at 300MHz:

Description:
The core current level in the rev 0.2 silicon is higher than necessary. Several modifications will be implemented in the production silicon which will reduce the core current significantly.

Workaround:
The production silicon draws significantly less current. The rev 0.2 silicon can be run at lower frequencies and lower voltage levels to reduce power levels until the production silicon is available.

Applies to revision(s):

ADSP-BF535-0.2

7. 05000016 - PCI: Extra BAR in Config Space:

Description:
The Configuration Space registers in the PCI core have an extra Base Address Register. This causes the System BIOS (in a PCI system) to see the ADSP-BF535 device as having one extra Memory or IO window (type depends on BIOS design). The only side effect should be that an invalid Memory window of size 16 bytes or an invalid IO window of size 4 bytes will be allocated in the system for the ADSP-BF535 device. Again, because of the nature of the register, the exact behavior is not covered by the PCI spec and will be BIOS specific. In any case this extra window should not cause a problem if the ADSP-BF535 device specific driver is written to ignore the extra window.

Workaround:
The ADSP-BF535 device specific driver should be written to ignore the extra window on revision 0.2 silicon.

Applies to revision(s):
ADSP-BF535-0.2

8. 05000018 - System Interrupt Controller Interrupt Mask Register polarity is different than the polarity used in IMASK:

Description:
The System Interrupt Controller Interrupt Mask Register (SIC_IMASK), as described in the ADSP-BF535 Hardware Reference manual, has the following polarity:

A bit value of 1 enables the corresponding interrupt.
A bit value of 0 disables the corresponding interrupt.

On the ADSP-BF535 rev 0.2 silicon, the polarity of the SIC_IMASK register is inverted in that a value of 1 actually disables the interrupt and a value of 0 enables the interrupt. In addition, the ADSP-BF535 Hardware Reference Manual states the reset value of SIC_IMASK is equal to 0x0000 (all interrupts disabled). Rev 0.2 silicon initializes the SIC_IMASK register to "all interrupts disabled" but due to the polarity inversion, the reset value of SIC_IMASK is 0x001FFFFF. The production version of the part will correct the polarity of the SIC_IMASK bits and reset value of the SIC_IMASK register to match the values as defined in the ADSP-BF535 Hardware Reference Manual.

Workaround:
When using rev 0.2 ADSP-BF535 silicon, a value of 0 must be used to enable the peripheral interrupts via the SIC_IMASK register. When the production part is released, the ADSP-BF535 will operate as described in the HW Reference Manual.

The following code sequence can be used to write code that will work properly on any silicon revision. It checks the upper byte of the CHIP_ID register. Revision 0.2 silicon will have a value of 0 in the upper byte. All future revisions will have a non-zero value in the upper byte.

```
//Check revision of silicon
        P0.L =  0x48C0;    //Memory mapped address of CHIPID register
        P0.H =  0xFFC0;
        R0 = [P0];
        R1.H = 0xf000;
        R1.L = 0x0000;
        R1 = R1 & R0;
        CC = AZ;        // If the CHIPID is non-zero, a "1" in
//SIC_IMASK will enable system interrupts
        if !CC jump new;
old:

        P3.L = SIC_IMASK & 0xffff;
        P3.H = SIC_IMASK >> 16;
        R6 = [p3];
        BITCLR(R6,17);         //Unmask the PF Interrupt A in System
Interrupt Mask Register
        [ P3 ] = R6;
        jump set_pointer;

new:
         P3.L = SIC_IMASK & 0xffff;
        P3.H = SIC_IMASK >> 16;
        R6 = [P3];
         BITSET(R6,17);   // any rev of 1.0 or later will have system
interrupts
//enabled by writing a 1 to the appropriate bit
        [ P3 ] = R6;
start:
```

Applies to revision(s):
ADSP-BF535-0.2

9. 05000019 - Undefined ASTAT bits:

Description:
Chapter 2 of the HW Reference and the Instruction Set Reference both refer to several bits in the ASTAT register which are not available in the ADSP-BF535 silicon. These bits are listed as follows:

```
      Bit 12: AC0
      Bit 13: AC1
      Bit 16: AV0
      Bit 17: AV0S
      Bit 18: AV1
      Bit 19: AV1S
      Bit 24: V
      Bit 25: VS
```

These bits are undefined and should not be used in the ADSP-BF535. This added functionality will be available in future Blackfin parts. In future Blackfin parts, these bits will perform as described in the ADSP-BF535 BLACKFIN DSP HARDWARE REFERENCE, REV 3.0, JULY 2004 and ISR documentation.

Workaround:
Use the remaining ASTAT bits described in the ADSP-BF535 BLACKFIN DSP HARDWARE REFERENCE, REV 3.0, JULY 2004.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

10. 05000023 - Core Timer counts during emulation mode in revision 0.2 silicon:

Description:

The core timer continues to count when the processor is in emulation mode. As described in the Hardware Reference Manual, the core timer should stop counting whenever the ADSP-BF535 is servicing an emulation event.

Workaround:
There is currently no way for the core timer to stop automatically when an emulation event occurs. As a result, the core timer will continue to count at the CCLK rate while the processor is servicing the emulation reset. The watchdog timer on the ADSP-BF535 does stop during emulation mode.

Applies to revision(s):
ADSP-BF535-0.2

11. 05000024 - Multi-issue instructions that store to L1 data memory and load from scratchpad do not work properly:

Description:
The following conditions will cause incorrect behavior in a multi-issue instruction:

A store to L1 Data memory A or B in parallel with a load from the scratchpad memory

Workaround:
Multi-issue instructions which perform a store to either L1 Data bank A or B in parallel with a load from the scratchpad memory should be avoided.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

12. 05000025 - Tristate pins are outputs during power up:

Description:
The bi-directional pins of the ADSP-BF535 have their outputs enabled during the time that VDD_IO (or VDD_PCI) is on and VDD_INT is off. The pins will drive high or low arbitrarily. Therefore connecting the bi-directional pins to each other or to ground will cause excessive current during power up.

Workaround:
The workaround is to avoid connecting bi-directional signals to each other or directly to outputs or to power or ground. The current drawn during this condition should be small during most reset scenarios.

Applies to revision(s):
ADSP-BF535-0.2

13. 05000029 - IrDA® functionality does not function correctly:

Description:
The IrDA® functionality on UART0 of the ADSP-BF535 does not function properly.

Workaround:
Use silicon revision 1.0 or later

Applies to revision(s):
ADSP-BF535-0.2

14. 05000035 - UART THRE bit latency:

Description:
There is a latency issue with the Transmit Holding Register of the UART which can cause a transmit data overrun situation to occur. In polling mode, the THR Empty (THRE) Bit within

the Line Status Register is cleared too late. This can force a subsequent polling operation to fail.

Workaround:
The Transmit Holding Register Empty (THRE) bit can be polled in software to acknowledge that the previous character loaded into the UART has been sent; at this point, the Transmit Holding Register can be loaded with the new character.

The following code is required to allow error-free transmissions via the UART by polling the THRE bit of the Line Status Register:

```
/*
   p0 = 0xffc0.1800  in case of UART 0
   p0 = 0xffc0.1c00  in case of UART 1

   #define THRE 0x05
   #define LSR 0x0A
   #define THR 0x00
*/

putc:
               [--sp] = r1;
putc_wait:
               r1 = w[p0+LSR];
               CC = BITTST(r1,THRE);
               if !CC jump putc_wait;

               w[p0+THR] = r0;
               ssync;

putc_workaround:
               r1 = w[p0+LSR];
               CC = BITTST(r1,THRE);
               if CC jump putc_workaround;

               r1 = [sp++];
               rts;
```

Please note that the putc_workaround loop may fail, if interrupted and delayed by any interrupt service routine.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

15. 05000036 - Rev 0.2 and 1.0 silicon are not PC133 compliant:

Description:
Rev 0.2 and 1.0 silicon do not meet the PC133 specification. The External Bus Interface Unit will not function reliably above 100Mhz. Also, for revision 0.2 silicon, the TSDAT requirement (Data setup prior to rising edge of CLKOUT) of the PC133 specification is not met.

The parameters out that are out of specification are listed below:

Name	Spec (max)	r0.2 Margin
TSDAT	2.1ns	-750ps

For revision 1.0 silicon, tDO (data output delay) is out of specification as follows:

Name	Spec (max)	r1.0 Margin
TDO	6.0ns	-150ps

Workaround:
The ADSP-BF535 rev 0.2 and 1.0 silicon meet PC100 from both a functional and timing standpoint.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0

16. 05000037 - Writing and reading ILOC bits:

Description:
When using the IMEM_CONTROL register to initialize the ILOC bits (to lock the ways) with bit positions 3 through 6, the data corresponding to these four bits are read back from bit positions 4 to 7.

Workaround:
The Ways can still be locked using bits 3-6. If the bits need to be read back, the value written to bit 3 will be read back from bit 4. The value written to bit 4 will be read back from bit 5. The value written to bit 5 will be read back from bit 6. The value written to bit 6 will be read back from bit 7.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

17. 05000038 - Boundary Scan does not work in Rev 0.2 silicon:

Description:
The Boundary scan functionality does not function correctly on Rev 0.2 silicon.

Workaround:
Use revision 1.0 silicon or later if Boundary Scan is required

Applies to revision(s):
ADSP-BF535-0.2

18. 05000039 - Icache Flush doesn't invalidate the Fill Buffer:

Description:
When a new cache line is fetched, the line will stay in the fill-buffer until a new miss occurs. Thus, if the most recently-fetched line is flushed, and no new misses occur before the instruction cache attempts to fetch the same line again, the fetch will result in a buffer hit.

Workaround:
The workaround for this problem is to force an instruction cache miss before the same line is fetched again.

The problem occurs only if you try to flush a line that could be in the fill buffer. If the user does not flush the next 4 lines when the flush instruction is being executed, there is no possibility of the Flushed line being in the fill buffer.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

19. 05000040 - PLL transitions incorrect coming out of reset:

Description:
A timing problem in the PLL digital logic may cause an undesired state transition when coming out of reset. There are several conditions that must be met for this undesired transition to happen. First, the reset must be initiated by asserting the reset pin. Software resets will not cause undesired state transitions. Second, a specific timing relationship between the deassertion edge on the reset pin and the ADSP-BF535 internal SCLK must

exist. Under these conditions, the ADSP-BF535 may either a) transition to the bypass state when coming out of reset, b) come up with the incorrect CCLK value, or c) come up with the correct CCLK value but with an incorrect SCLK value. When CCLK is set to run at 300MHz, SCLK may come up incorrectly at 150MHz (SSEL = CCLK/2). If b) or c) happens, the part will not boot. The workaround must be followed.

Workaround:
A hardware and software workaround exists. The part must be placed in bypass by pulling the external bypass pin high. To ensure that the part does come up in bypass, the following circuit must be used for reset and CLKIN. This circuit should do the following: Keep RESET_B to the DSP low, as an initial condition. RESET_B to the DSP must rise smoothly, shortly after the falling edge of the DSP CLKIN. RESET_B must pass through a negative edge flip-flop clocked by CLKIN. Two flip-flops are used to reduce the chance of meta-stable outputs and insure a clean rise of the new RESET_B signal. The gate is used to insure the initial condition of RESET_B prior to the start of CLKIN. The speed of this circuit is frequency and duty cycle dependent. At the maximum CLKIN frequency of 33 MHz and 60% duty cycle, the total delay between CLKIN and the new reset signal must be less than 14 ns. Longer CLKIN cycle times provide for slower circuits that have longer propagation delay. The software below will then transition the ADSP-BF535 to the desired frequency of operation.

```
// Begin Bypass workaround.
// PLL is in Bypass mode when external bypass pin pulled high
// Setup Watchdog Timer to generate wakeup event to bring DSP out of
idle
// Set watchdog count value to allow enough time for transition and
PLL lock
                  r7.l = 0xfff1;
               r7.h = 0x0000;
               p0.l = WDOGCNT & 0xffff;
               p0.h = WDOGCNT >> 16;
               [p0] = r7;
               ssync;


// Program the PLLCTL register to multiply core and system clock
frequency by
// setting MSEL and SSEL bits.  In this example, MSEL=15x and
SSEL=CCLK/2.5.
// This can be changed to another core and system frequency of
choice.
                  r7.l = 0x1F00;
               r7.h = 0x0001;
               p0.l = PLLCTL & 0xffff;
               p0.h = PLLCTL >> 16;
               [p0] = r7;
               ssync;

//After desired MSEL and SSEL frequency ratios are programmed, turn
bypass mode
// off using the following code sequence.
               p0.l = PLLCTL & 0xffff;
               p0.h = PLLCTL >> 16;
               r7 = [p0];              // Read the value in PLLCTL
               bitclr(r7,8);           // Set bypass mode off
               [p0] = r7;              // set the PLLCTL to: PLL not
bypassed
```

```
                ssync;

// The following code sequence reloads the Watchdog counter.
                r7.l = 0x0000;
                r7.h = 0x0000;
                p0.l = WDOGSTAT & 0xffff;
                p0.h = WDOGSTAT >> 16;
                [p0] = r7;
                ssync;

// This enables the watchdog GP interrupt.
                r7.l = 0x0004;
                p0.l = WDOGCTL & 0xffff;
                p0.h = WDOGCTL >> 16;
                W[p0] = r7;
                ssync;

// Enter IDLE mode to allow PLL to changed to programmed frequency.
// The programmed watchdog wakeup event will bring the DSP out of
IDLE.
                 cli r7;
               idle;
               ssync;
                 sti r7;

// Clear WD flag and disable WD events after DSP comes of IDLE in new
frequency.
                r7.l = 0x8006;
                p0.l = WDOGCTL & 0xffff;
                p0.h = WDOGCTL >> 16;
                W[p0] = r7;
                ssync;

//  End bypass workaround
```

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1

20. 05000041 - Icache bus error does not Flush the Fill Buffer:

Description:
If a bus error occurs during a line fetch, the line is not cached (as expected). It is however kept in the buffer until a new miss occurs. Thus, if no new misses occur before the instruction cache attempts to fetch the same line again, the fetch will result in a buffer hit.

Workaround:
The only known work around for this problem is to force an instruction cache miss to occur before the line obtained during the bus-error is re-fetched.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

21. 05000042 - Core hangs on a specific condition:

Description:
The core hangs when an interrupt occurs while the first store instruction is in the store buffer and the conditional jump instruction and the second store instruction are still in the pipeline.

The fourth push instruction in the Interrupt Service Routine causes the core to hang.

Workaround:
Avoid this combination of instructions after a jump instruction.

Applies to revision(s):
ADSP-BF535-0.2

22. 05000043 - Sport Transmit Underflow (TUVF) special clearing instructions:

Description:
The ADSP-BF535 BLACKFIN DSP HARDWARE REFERENCE, PRELIMINARY EDITION, NOVEMBER 2001 is not clear on the process steps necessary to clear a SPORT transmit underflow error.

TUVF is an underflow bit within the SPORT_STAT register, and it is sticky. This same value is sent to the DMA Controller, and it is used to set the ERR IRQ status bit. The TUVF bit is cleared on a hard reset. From that moment on, if it is set, it stays set until the SPORT is re-enabled. That is, the SPORT must be disabled and then re-enabled again. Once the SPORT is enabled, it takes 4 transmit clocks to clear the TUVF. Once TUVF is clear, the DMA's ERR IRQ bit can then be cleared. (with a write of a "1" to the bit).

Workaround:
The following sequence will clear a SPORT transmit underflow condition:
1) write TSPEN = 0 and wait > 1 TSCLK 2) write TSPEN = 1 and wait > 5 TSCLK + 2 sclks 3) write TSPEN = 0 and wait > 1 TSCLK

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

23. 05000044 - Incorrect ASTAT flag operations:

Description:
AN Flag for Max/Min is dependent on order of operands.
If

```
 R0=+1; R1=-1;

          R2 = MIN(R1,R0); sets AN
          R2 = MIN(R0,R1); does not set AN
```

AN and AZ flags do not work properly for instructions of the form dreg=(dreg+dreg)<<1,2;

The ABS instruction sometimes sets the AN flag even if neither the input or output are negative.

Problem with SIGNBITS instruction:

R0.L = SIGNBITS A0; returns -8 through 30 if A0 =0x80000000000 and A0 = 0x0000000001 39 if A0 = -1 (this is the bug) 39 if A0=0 (should match the above "-1" case)

Workaround:
None

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

24. 05000046 - Shifter does not consider all 32 bits when shift value is a D register:

Description:
The arithmetic shift instruction masks and ignores bits that are greater than 6 bits. Future

versions of Blackfin DSPs will support larger shift values as described in the Arithmetic Shift section of the Blackfin DSP Instruction Set Reference.

Workaround:
None

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

25. 05000047 - PCI register changes:

Description:
The PCI Configuration Device ID register(PCI_CFG_DIC) has a reset value of 0x00001535 in silicon revisions 1.0 and later. The PCI Configuration Vendor ID Register(PCI_CFG_VIC)has a reset value of 0x000011D4 in silicon revisions 1.0 and later. The values of these registers are 0x0 for revision 0.2 silicon.

The PCI Device Memory Bar Mask Register(PCI_DMBARM) and the PCI Device IO Bar Mask Register(PCI_DIBARM) are not reset during a PCI reset.

Workaround:
Provided as info only

Applies to revision(s):
ADSP-BF535-0.2

26. 05000048 - Speculative fetch of illegal or undefined memory location causes HW error:

Description:
A hardware error is generated when an illegal or undefined address access is attempted, even if the offending instruction is not actually executed. Given a loop written in C of the form:

```
        while (p && p->some_value)
                p = p->next;
```

the code generated is:

```
loop_check:
check p == 0
if true, skip out of loop
load [p+offset-to-some_value]
check that value == 0
if true, skip out of loop
loop_body:
load [p+offset-to-next]
goto loop_check
```

which is correct. Although there is a test to check whether p is zero, p->some_value is still pre-loaded speculatively, and that causes a hardware error if p does not point to a valid location. Other illegal accesses, such as an illegal MMR access, will result in a similar behavior.

Workaround:
In the example provided, the NULL pointer corresponds to the first SDRAM address, so if SDRAM bank 0 is initialized, a hardware error will not be generated. In general assembly programs, a CSYNC instruction can be used to prevent the offending instruction from entering the pipeline.

If compiled code is used and SDRAM is not initialized, hardware errors should not be enabled.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

27. 05000051 - Accessing MMRs with Data Cache disabled and CPLBs enabled:

Description:
When the program is configured with DCPLBs enabled but with cache disabled and in Write Back mode, an exception (0x23 - data access CPLB Protection Violation) is generated on all MMR accesses.

```
1) This problem only occurs when cplbs are enabled
and cache is disabled.

2) DCPLBs MUST be turned off in emulation mode - so
we end up getting an unrecoverable event with any
emulation breakpoints.
```

Workaround:
Use Write Through mode in the DCPLB setup for all MMR space and set as non-cacheable.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

28. 05000052 - USB: On a HOST OUT (PC->ADSP-BF535) disarm is not functioning correctly:

Description:
On a HOST OUT (PC->ADSP-BF535), disarming an endpoint should not allow any more transfers to occur on the endpoint. However, a transfer does occur when a endpoint is disarmed.

Example:
A Bulk Out transfer should NAK when a endpoint is disarmed. But it does not and the USB device accepts the packet.

Workaround:
There are two ways that the ARM bit in the USBD_EPCFGx should be cleared

```
* When the transfer is complete
* When the Byte count goes to zero.
```

The application software would have to be modified to take the interrupts (USBD_BCSTAT and USBD_TC interrupts) and toggle the direction bit (USBD_DIR) in the USBD_EPCFGx register. This will NAK the transfers.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1

29. 05000053 - USB: EPLEN is not updating correctly at the end of a short packet:

Description:
On a HOST OUT (PC -> Device), the EPLEN (endpoint length) register decrements by the packet size rather than the actual number of bytes transferred. This is not true if the EPLEN was programmed to be less than packet size.

Example:
Starting EPLEN = 320 Transferring 259 bytes Ending EPLEN = 0 (320-320 since 320 is a multiple of 64) Expected EPLEN = 61

Workaround:
A header could be added indicating the number of bytes that are being transferred whenever doing a HOST OUT transaction. This will be fixed in the next silicon revision.

Applies to revision(s):

ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1

30. 05000054 - Problems with data cache writeback mode:

Description:
When a data write is made to L2 memory with data cache on and configured as writeback, a different value is sometimes read back.

Workaround:
Avoid using writeback mode in the referenced silicon revisions prior to rev 1.2. For silicon revisions of 1.2 or later, an SSYNC instruction must be the first instruction of any asynchronous interrupt service routine whenever data cache is configured as "write back".

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

31. 05000055 - Setting breakpoints in an emulator session with cache enabled:

Description:
When debugging the ADSP-BF535 with cache enabled, selecting "run" from a breakpoint in VisualDSP++ does not advance the program counter. The breakpoint can be stepped over using the single step key (f11). At this point, run can be selected and the program counter will advance.

Workaround:
In addition to the workaround described in the problem description, two CPLB's can be reserved for emulator use: 1 DCPLB for the MMR area, and 1 ICPLB for the reset vector based on the boot mode. Please add these two memory areas and CPLB initialization values to your cache init routine. Attached is a simple example using cache which shows where these two entries can be used.

```
// Addresses of the memory areas we configure.
cplb_addrs:
        .byte4 =
                0xFFC00000, // MMRs FOR EMULATOR
                0xEF000000,    // RESET VECTOR FOR EMULATOR   <-- this
address depends on the boot mode selected


// Bits to set in the CPLBs
cplb_set:
.byte4 =
                (PAGE_SIZE_4MB|CPLB_DIRTY|CPLB_SUPV_WR|CPLB_VALID),
        // MMRs FOR EMULATOR
                (PAGE_SIZE_4MB|CPLB_DIRTY|CPLB_SUPV_WR|CPLB_VALID),
        // // RESET VECTOR FOR EMULATOR
```

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

32. 05000056 - USB: ARM bit not clearing on short packets:

Description:
On a HOST OUT (PC -> Blackfin), the endpoint does not get disarmed at the end of a transfer if the programmed endpoint length is greater than the number of bytes received by the device. However the transfer complete interrupt is set indicating end of transfer.

As long as the endpoint is disarmed, no transfer takes place on that endpoint. The Host will receive NAKs indicating the device is not ready, and will retransmit if required. By not disarming, the endpoint is ready for next transfer. The firmware will then have problems distinguishing between subsequent packets.

```
Example:
    Starting
    EPLEN = 320
    EPCFG = armed for bulk out with packet size of 64
    Host send 259 bytes

    Ending
    EPCFG = armed for bulk out with packetsize of 64
    EPINTR = TC is set, BC is not set, PC is set
```

Workaround:
Take the interrupt and clear the ARM bit in the endpoint configuration register.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1

33. 05000057 - RTC integrity when core powered off:

Description:
When power to the core is removed, the RTC can lose all settings even if the RTC battery is still supplying power. If power to the core is removed, the RTC must be re-programmed with the correct values when power to the core is restored.

Workaround:
None

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

34. 05000060 - Serial clock is not gated by SPORT enable bit:

Description:
If one of the serial clocks, TCLK or RCLK, is configured to be generated internally, the clock is generated regardless whether the SPORT module is enabled or not. The serial clock divider, which is updated only when the SPORT is enabled, can produce unwanted clock pulses with the former SPORT frequency settings generated until the enable bit is set. If the SPORT is enabled for the first time after reset, the observed frequency is SCLK/2 since the clock divide registers reset all to zero. Even if the ICLK bit is set at the same like the SPORT enable bit, a set of unwanted clock pulses is generated due to the SPORT enable latency.

Note also that these leading clock pulses should not cause any issues if the SPORT is disabled and re-enabled again with the SCKDIV settings.

Workaround:
To disable the internally generated clock, first clear the enable bit. In a subsequent, separate write operation, also clear the ICLK bit.

Connect devices that tolerate the incorrect leading clock pulses. Note that most of devices ignore these pulses anyways, since no valid frame sync is generated during the critical period. If the connected device cannot tolerate incorrect clock pulse, the clock signal still can be gated and synchronized by off-chip logic controlled by a general-purpose flag pin during initial SPORT setup and run-time reconfigurations.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

35. 05000061 - 1.38V minimum VDDcore required:

Description:
For silicon revisions prior to 1.2, the part will not operate properly if VDDcore is below 1.38V.

Workaround:
For silicon revisions prior to 1.2, have VDDcore above 1.38V. Silicon revisions after 1.2 can operate properly with VDDcore below 1.38V as specified in the datasheet.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1

36. 05000062 - SPORT Multichannel Mode starting and stopping:

Description:
If using the SPORTs in multichannel mode, the data will be corrupted if the SPORTs are first enabled, then disabled, and re-enabled again.

Workaround:
The following sequence provides correct data transmission if enabling, disabling, and re-enabling the SPORTs are necessary:

```
Disable SPORT -> Disable MCM operation -> Set FLSH bit in DMA config
register -> wait 4 SPORT clock cycles -> CLR FLSH bit in DMA config
register -> Stop DMA
```

This sequence must be completed before the next frame sync comes in.

The SPORTs must be stopped on completion of a DMA transfer. If they are stopped midway through a multichannel mode DMA transfer, then on restart you will get unpredictable results.

Both RX and TX are stopped separately. TX is stopped in the TX interrupt routine when it completes transmitting. RX is stopped in the RX interrupt routine when it completes receiving. The Disabling of MCM operation MUST be done in the RX interrupt routine, this step should not be done in the TX Interrupt routine.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

37. 05000071 - CLI instruction can allow an asynchronous interrupt occur as it is committing:

Description:
The CLI instruction is intended to block interrupt processing for critical sections of code. The CLI instruction commits at the end of the pipe, and starts blocking interrupts until an STI instruction re-enables interrupt processing.

If a higher priority interrupt occurs just before the CLI commits, the interrupt may be serviced immediately after the CLI has committed. This means that rather than a few critical cycles before executing the STI, the entire higher priority interrupt service routine will be executed.

If a third, highest priority interrupt signals during this period, it will not be serviced until the STI is finally executed. This additional latency is unintended.

Workaround:
The STI instruction should block interrupts just before its commit point, so that it operates atomically in the instruction stream. This is similar in behavior to the POP RETI instruction. A POP RETI can be used as a software workaround. Replace each CLI instruction with:

```
   RETI=[sp++]; // blocks new interrupts, sets IPEND<4>
   CLI R0;      // modifies IMASK
   [--sp]=RETI; // restores stack to what it was, unsets IPEND<4>.
```

Note that we don't have to actually be pointing to a valid RETI on the stack for this to work; as long as there is a stack, this will block interrupts for the CLI instruction.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

38. 05000164 - Store to Load Forwarding in Write Through Mode:

Description:
When the L1 Data memory is configured as a Cache in Write Through mode, under certain conditions when a store is followed by a load to the same address, the load can return incorrect data.

The conditions are detailed below:

```
   - DCACHE is configured in WT mode
   - A Load starts a line fill to a certain address
```

Between the above Load and the below Store, one of the following events needs to occur:

```
   * Mispredicted Branch

   * Interrupt

   * Exception

   * RTI/RTE/RTN/RTX instruction

   - A store to an address in the same line is issued to the Store
     buffer.
   - The current store misses in the DCache as a line fill to the
     line is in progress.
   - The store buffer writes the store to the Write Buffer
   - The line fill completes and the data is written into the Cache
     (Remember the line fill has the data without the latest store)
   - A load is issued to the same address as in the Write Buffer
   - The load "hits" in the Data Cache and provides incorrect data.
```

Workaround:
Place an SSYNC instruction to separate the Store from the first line fill. This will ensure the store gets out of the Store buffer after the line fill is completed.

OR Operate the Data Cache in WB mode. This issue does not exist in WB mode

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

39. 05000165 - Data cache dirty bit set when a load-miss-fill is in progress:

Description:
When a Fill is in progress due to a Load Miss and one of the following events occur :

```
        - Mispredicted Branch
        - RTI, RTE, RTX, RTE, RTN instructions
        - Interrupts and Exceptions
```

```
      - Single Step Mode
```

on completion of the Fill when the line is written into the Data cache, the Dirty Bit for the line is incorrectly Set. The dirty bit is set in both Write Back (WB) and Write Through (WT) modes.

Examples of Code that will expose the errata:

```
=========================================
1.     r0 = 1;
       cc = r0;
       if cc jump 0x4;
       r0 = [p0] = r0;      // Load that misses and is Aborted
                            //  due to BRT

2.     r0 = 1;
       cc = r0;
       [p0] = r0;           // Load that misses and
       if cc jump 0x4;      // Generates a Kill EX3 after
                            // the load completes
                            // but before the Fill finishes

3.     [p0] = r0;           // Load that misses
       RTI;                 // Generates a Kill EX3
                            //after the load completes
                            // but before the Fill finishes.
```

1. Since the Dirty bit is set, when that line is either replaced or "Flushed", it will cause a copy back to the lower levels of memory, thereby, impacting the overall performance.

2. Since the Dirty bit is set for Loads, when the line is either replaced or Flushed, copy back will be issued. This will also occur for Read-Only sections of memory, therefore could have a impact to the functionality of the system.

3. Since the Dirty bit is set for the WT mode as well, copy back that are issued may cause coherency issues with lower level memory unless managed by SW.

Workaround:
When programming in C, make sure any buffer that is malloc'ed is "flushed" before re-using this memory space.

Also ensure the following conditions are met:

```
   - Mispredicted Branch
      - Place an SSYNC at the First Instruction of the Branch Target

   - Interrupt & Exceptions
      - Place an SSYNC at the First instruction of the ISR

   - RTI/RTE/RTN/RTX instruction
      - Place a CSYNC after every RTI/RTN/RTE/RTX instruction
```

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

40. 05000170 - PLL clkout may be incorrect in multiplier mode coming out of reset:

Description:
When powering up the Processor in multiplier mode (BYPASS pin low), the system clock may have an incorrect duty-cycle or show an unexpected DC offset or both. This can adversely affect the Processor's operation.

Workaround:
The recommended way of powering up the Processor is by using the PLL's BYPASS mode (BYPASS pin high) and setting the core and system clocks in software as shown in the code example below. The MSEL and DF pins are irrelevant in this mode (the corresponding bits are set to zero in the PLL_CTL register) and booting will execute at CLKIN/2 speed. If this is unacceptable, the Processor must be monitored in some way. For instance, the user code could set a flag pin or communicate with a host. Absence of such a signal would trigger a reset of the device.

```
// Begin Bypass workaround.
// PLL is in Bypass mode when external bypass pin pulled high
// Setup Watchdog Timer to generate wakeup event to bring DSP out of
idle
// Set watchdog count value to allow enough time for transition and
PLL lock
// The following sets the watchdog to timeout after approx. 26600
SCLK cycles.
// This is a conservative number that will work for all combinations
of
// frequency settings.
// It can be adapted to the particular frequencies in your
application in the
// following fashion:
// A minimum time of 2000 CLKIN cycles is recommended as per the
datasheet.
// The watchdog is clocked by SCLK. SCLK is (CLKIN/2)/SSEL at power-
up and will
// be CCLK/SSEL after the transition. If we assume that the end value
is
// applied for the entire duration of the transition (worst case), it
follows
// that the correct value for the watchdog counter is equal to:
// counter_value = 2000 * CCLK_ratio / SSEL
// where SSEL = 2, 2.5, 3, 4 and
// CCLK_ratio = ratio between CCLK and CLKIN, defined by MSEL and DF
settings.

r7.l = 0x6800; // can be changed according to formula above
r7.h = 0x0000;
p0.l = WDOGCNT & 0xffff;
p0.h = WDOGCNT >> 16;
[p0] = r7;
ssync;


// Program the PLLCTL register to multiply core and system clock
frequency by
// setting MSEL and SSEL bits.  In this example, MSEL=15x and
SSEL=CCLK/2.5.
// This can be changed to another core and system frequency of
choice.
                 r7.l = 0x1F00;
r7.h = 0x0001;
p0.l = PLLCTL & 0xffff;
p0.h = PLLCTL >> 16;
[p0] = r7;
```

```
ssync;

//After desired MSEL and SSEL frequency ratios are programmed, turn
bypass mode
// off using the following code sequence.
p0.l = PLLCTL & 0xffff;
p0.h = PLLCTL >> 16;
r7 = [p0];              // Read the value in PLLCTL
bitclr(r7,8);           // Set bypass mode off
[p0] = r7;              // set the PLLCTL to: PLL not bypassed
ssync;

// The following code sequence reloads the Watchdog counter.
r7.l = 0x0000;
r7.h = 0x0000;
p0.l = WDOGSTAT & 0xffff;
p0.h = WDOGSTAT >> 16;
[p0] = r7;
ssync;

// This enables the watchdog GP interrupt.
r7.l = 0x0004;
p0.l = WDOGCTL & 0xffff;
p0.h = WDOGCTL >> 16;
W[p0] = r7;
ssync;

// Enter IDLE mode to allow PLL to changed to programmed frequency.
// The programmed watchdog wakeup event will bring the DSP out of
IDLE.
                cli r7;
idle;
ssync;
                sti r7;

// Clear WD flag and disable WD events after DSP comes of IDLE in new
frequency.
r7.l = 0x8006;
p0.l = WDOGCTL & 0xffff;
p0.h = WDOGCTL >> 16;
W[p0] = r7;
ssync;

//  End bypass workaround
```

When the part is booted in PLL bypass mode, it is important to run the workaround code before connecting with an ICE. This is due to the fact that CCLK must be at least twice the JTAG frequency to make a successful connection.

Applies to revision(s):
ADSP-BF535-1.2, ADSP-BF535-1.3

41. 05000178 - Rev 1.2 and 1.3 silicon do not meet TDCAD requirement of the PC133 spec:

Description:
Rev 1.2 and 1.3 silicon does not meet TDCAD requirement of the PC133 specification.

For 25 degrees C and above,

```
Name     Spec     Measured Value
TDCAD    0.6 ns   0.7 ns
```

Workaround:
From a functional and timing standpoint, the SCLK speed can be lowered in order to meet the TDCAD requirement.

Applies to revision(s):
ADSP-BF535-1.2, ADSP-BF535-1.3

42. 05000195 - Latency in Writes to RND_MOD bit:

Description:
For the case when a "move" is made to ASTAT followed by a multiply/MAC instruction, the RND_MOD bit is used directly from the ASTAT register, and the value does not change until the new ASTAT value is committed.

The following example demonstrates this issue. The result of the R0.L and R1.L multiply is incorrect.

```
   R2 = ASTAT;   //Read ASTAT register.
   BITSET(R2, ASTAT_RND_MOD_P);    //Set the RND_MOD bit.
   ASTAT = R2;   //Write result back to ASTAT register.
   R0.L = R0.L*R1.L;   //Perform multiply.
   R0 = R0.L (X);      //Sign extend R0
```

Workaround:
Inserting two nops after the write to the ASTAT register takes care of the latency. In the example code above that demonstrates this problem, the following example code resolves the issue.

```
   R2 = ASTAT;   //Read ASTAT register.
   BITSET(R2, ASTAT_RND_MOD_P);    //Set the RND_MOD bit.
   ASTAT = R2;   //Write result back to ASTAT register.
   nop; nop;     //Insert two nops to resolve latency issue.
   R0.L = R0.L*R1.L;   //Perform multiply.
   R0 = R0.L (X);      //Sign extend R0
```

In this example, the result of the R0.L and R1.L multiply is correct.
This workaround has also been implemented in the mult_r() library function, and will be available in the August 2004 update of the VisualDSP++ 3.5 release.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3

43. 05000210 - IDCODE register not JTAG IEEE 1149.1 compliant:

Description:
The ADSP-BF535 does not meet the JTAG IEEE 1149.1 spec for the IDCODE register. The JTAG IDCODE register on the ADSP-BF535 shifts out the MSB first. However, the JTAG IEEE spec states that the LSB must be scanned first.

Workaround:
JTAG software should be changed such that it looks for a bit-reversed IDCODE output.

Applies to revision(s):
ADSP-BF535-0.2, ADSP-BF535-1.0, ADSP-BF535-1.1, ADSP-BF535-1.2, ADSP-BF535-1.3