TigerSHARC Embedded Processor

Silicon Anomaly List — ADSP-TS201S

ABOUT ADSP-TS201S SILICON ANOMALIES

These anomalies represent the currently known differences between revisions of the TigerSHARC® ADSP-TS201S product(s) and the functionality specified in the ADSP-TS201S data sheet(s) and the Hardware Reference book(s).

SILICON REVISIONS

A silicon revision number with the form "-x.x" is branded on all parts (see the data sheet for information on reading part branding). The silicon revision can also be electronically read by reading the IDCODE register either via JTAG or DSP code.

The following DSP code can be used to read the register:

```
<UREG> = IDCODE;;
```

Silicon REVISION	IDCODE[31:28]
2.0	0100
1.2	0011

ANOMALY LIST REVISION HISTORY

The following revision history lists the anomaly list revisions and major changes for each anomaly list revision.

Date	Anomaly List Revision	Applicable Data Sheet Revision	Additions and Changes
07/16/2014	S	C	Modified anomalies: 03000377
07/10/2014	R	C	Added anomalies: 03000377
01/08/2008	Q	C	Added anomalies: 03000376
09/25/2007	P	C	Added anomalies: 03000375
08/08/2007	O	C	Modified anomalies: 03000373
05/23/2007	N	C	Added anomalies: 03000372, 03000373
09/22/2006	M	B	Added anomalies: 03000371
08/01/2006	L	B	Added anomalies: 03000360, 03000364, 03000367, 03000368 Modified anomalies: 03000357
01/21/2006	K	A	Added anomalies: 03000358, 03000359
10/27/2005	J	A	Added anomalies: 03000357
10/03/2005	I	A	Added anomalies: 03000348, 03000349, 03000350, 03000351, 03000352, 03000353

One Technology Way, P.O.Box 9106, Norwood, MA 02062-9106 U.S.A.
Tel: 781.329.4700 **www.analog.com**
Fax: 781.461.3113 **©2014 Analog Devices, Inc. All rights reserved.**

SUMMARY OF SILICON ANOMALIES

The following table provides a summary of ADSP-TS201S anomalies and the applicable silicon revision(s) for each anomaly.

No.	ID	Description	1.2	2.0
1	03000168	First instruction of an ISR has to be quad-aligned	x	x
2	03000192	Protection does not prevent writes	x	x
3	03000206	RTI or RETI instruction followed by CJMP relative	x	x
4	03000208	Link read status after read Rx register	x	x
5	03000211	BTB on jump from internal to external address	x	x
6	03000214	Trace buffer samples the wrong PC at WP1 exception	x	x
7	03000220	Read of link status after error in a write to a link control	x	x
8	03000236	External Port DMA chaining may fail when paused	x	x
9	03000263	FLYBY On 64 Bit Bus Supports Long Transactions Only	x	x
10	03000296	Predicted jump from external to internal updates the BTB	x	.
11	03000298	Missed stall on ACS/MAX/TMAX after conditional ACS/MAX/TMAX	x	.
12	03000300	Link Transmits Port Enable Timing Restrictions	x	.
13	03000302	Reading from SDRAM when SDREN bit in SDRCON is clear causes deadlock	x	x
14	03000304	Interrupt routine for the same interrupt is serviced every time we reach idle with HW interrupt enable bit clear	x	x
15	03000306	PC of interrupt during idle is corrupted due to stall	x	x
16	03000315	Aborted conditional memory access may cause an exception	x	.
17	03000316	Do not use IALU or Sequencer instructions in the first quad of an ISR	x	x
18	03000319	Missing exception on illegal read from external memory to sequencer/debug registers	x	.
19	03000323	Condition flag not updated after sum/abs instruction is followed by an aborted ALU instruction	x	.
20	03000326	External DAB Results in Erroneous Results in Certain Cases	x	.
21	03000328	Conditional E2 instructions that are aborted still generate exceptions	x	.
22	03000329	Some exceptions are missed when they follow a not-acknowledged transaction	x	.
23	03000330	Link Port Transmit Rise / Fall Times Violate Spec	x	x
24	03000331	Weakest Drive Strength Setting is Not Functional	x	x
25	03000332	EXCAUSE is not updated when watchpoint exception occurs and emulation is enabled	x	.
26	03000333	Single-step software exception is not generated when return from the exception routine is a BTB hit	x	x
27	03000339	Locking cache may cause memory access failures	x	.
28	03000341	Link Port LVDS Electrical Characteristics	x	.
29	03000342	Link Port LVDS AC Characteristics	x	.
30	03000347	Do not update global static flags in the same line with write SQ / BD registers	x	x
31	03000348	Extra writes to AutoDMA may cause cluster bus deadlock	x	x
32	03000349	Emulation mode allows multiple writes to SYSCON and SDRCON	x	x
33	03000350	FLOAT BY instruction with truncate (T) option generates wrong result	x	x
34	03000351	Subtract with divide by two instruction with signed round to nearest even option generates wrong result	x	x
35	03000352	Link Transmits Port Enable Timing Restrictions	.	x
36	03000353	Conditional E2 instructions that are aborted still update EXCAUSE	.	x
37	03000357	Link Port LVDS Electrical Characteristics	.	x
38	03000358	Initial load MR register instruction can fail	.	x
39	03000359	Compute block source register busses can fail to initialize properly	.	x
40	03000360	Predicated RETIB load (store) instructions fail	x	x
41	03000364	AC Signal Specifications - Input Hold	x	.

No.	ID	Description	1.2	2.0
42	03000367	Link Port Transmit 4-bit mode enable failure	.	x
43	03000368	Link port receive enable after disable failure	x	x
44	03000371	SCLK_Vref Specification	x	x
45	03000372	TIMER enable/disable restriction	x	x
46	03000373	PLL Failure to Lock at Power-Up	x	x
47	03000375	DMA for link port to/from external memory may be blocked	x	x
48	03000376	Link port receive to link port transmit DMA fails	x	.
49	03000377	Internal DRAM Corruption Under Specific Conditions	x	x

Key: x = anomaly exists in revision
. = Not applicable

DETAILED LIST OF SILICON ANOMALIES

The following list details all known silicon anomalies for the ADSP-TS201S including a description, workaround, and identification of applicable silicon revisions.

1. 03000168 - First instruction of an ISR has to be quad-aligned:

DESCRIPTION:
If the first instruction of an interrupt service routine is not quad-aligned, the sequencer may fail to execute properly.

WORKAROUND:
Quad-align all ISRs with `.align_code 4;` directive. Note that if the ISR is in a separate module, the linker will quad-align it automatically.

APPLIES TO REVISION(S):
1.2, 2.0

2. 03000192 - Protection does not prevent writes:

DESCRIPTION:
When writing to the protected registers (i.e. all registers except the compute block and the IALU) in USER mode, the transaction is executed and the data is written to the target register. The exception, if enabled, is still generated.

WORKAROUND:
None.

APPLIES TO REVISION(S):
1.2, 2.0

3. 03000206 - RTI or RETI instruction followed by CJMP relative:

DESCRIPTION:
Relative CJMP and CJMP_CALL branch instructions can branch to an unpredictable target.

WORKAROUND:
All CJMP and CJMP_CALL branch instructions must be used with option (ABS).

APPLIES TO REVISION(S):
1.2, 2.0

4. 03000208 - Link read status after read Rx register:

DESCRIPTION:
If data is read from link receive buffer, and immediately on the next cycle the status register is read, it may still contain the old (before the buffer read) value.

WORKAROUND:
The LRSTATx register should always be read twice sequentially as follows:

```
XR0 = LRSTATx;;
XR1 = LRSTATx;;
```

The second read will ensure the contents of the status register have been updated. If a destructive read is desired, the sequence should be as follows:

```
XR0 = LRSTATx;;
XR1 = LRSTATx;;
XR2 = LRSTATCx;;
```

APPLIES TO REVISION(S):
1.2, 2.0

5. 03000211 - BTB on jump from internal to external address:

DESCRIPTION:
In case of jump from internal address to external address with prediction "taken", the entry is written into the BTB, but with valid bit cleared. This causes a useless replacement in BTB since the new entry has LRU value cleared (i.e. invalid).

WORKAROUND:
This is a minor performance issue. When it is known that branching is to external memory, use of (NP) option will prevent a useless BTB entry.

APPLIES TO REVISION(S):
1.2, 2.0

6. 03000214 - Trace buffer samples the wrong PC at WP1 exception:

DESCRIPTION:
In case of WP1 exception, the TRCB samples the next instruction address after the instruction that caused the exception.

Example:
The WP1 generates exception on transaction on JALU access (line 0x00080025). The TRCB will sample the next address (0x00080026). After the exception (RTI) the TRCB will sample this address again.

```
TRCB0 0x00080026
TRCB1 0x00010000 - Exception routine address
TRCB2 0x00010010 - RTI
TRCB3 0x00080026
```

WORKAROUND:
None

APPLIES TO REVISION(S):
1.2, 2.0

7. 03000220 - Read of link status after error in a write to a link control:

DESCRIPTION:
If data written to one of the link control registers (LTCTL or LRCTL) causes an error condition, and on the next cycle of the SOC bus the corresponding status register (LTSTAT or LRSTAT) is read, the read access might not show that error.

WORKAROUND:
Execute a dummy read on the SOC bus first, for example:

```
LTCTL = <Ureg>;;
<Ureg> = LTSTATx;;   /* this is a dummy read to be ignored */
<Ureg> = LTSTATx;;   /* this is the real read to be used */
```

APPLIES TO REVISION(S):
1.2, 2.0

8. 03000236 - External Port DMA chaining may fail when paused:

DESCRIPTION:
If an external port DMA is paused during chaining when one of the two TCB's has chained while the other has not, and then TCBs are modified, un-pausing the DMA will result in a chaining failure and the DMA channel will be stuck.

WORKAROUND:
If such a DMA process is required (as it may be in cases of chain insertions), after pausing the type field of the two DP registers must be examined. If both are legal or both are inactive (000), continue with the modification of the TCB's as required. Otherwise (which is the case when one has chained and the other has not), un-pause the channel, wait, pause and check again.

APPLIES TO REVISION(S):
1.2, 2.0

9. 03000263 - FLYBY On 64 Bit Bus Supports Long Transactions Only:

DESCRIPTION:
FLYBY transactions must be long transactions when in the 64 bit external bus mode.

WORKAROUND:
When the external memory is set to 64 bit bus, a FLYBY DMA TCB must be set to long word transactions.

APPLIES TO REVISION(S):
1.2, 2.0

10. 03000296 - Predicted jump from external to internal updates the BTB:

DESCRIPTION:
Predicted branch from external memory space to internal memory space when the BTB is enabled may fail.

WORKAROUND:
All branches in code in external memory should be (NP).

APPLIES TO REVISION(S):
1.2

11. 03000298 - Missed stall on ACS/MAX/TMAX after conditional ACS/MAX/TMAX:

DESCRIPTION:
If a conditional ACS, MAX or TMAX (T/MAX) or TR load is, within the next two cycles, followed by an ACS or T/MAX instruction line, the last instruction may give wrong results. For example,

```
a. (1) if <cond>; do, sTR1 = T/MAX();;
   (2) inst ;;
   (3) sTR2 = ACS( TR1, ...) ;;

b. (1) if <cond>; do, TR1 = T/MAX();;
   (2) TR2 = ACS (TR1, ...);;
```

WORKAROUND:
Either of the following:
1. Use T/MAX instruction unconditionally.
2. After a conditional T/MAX instruction, add two lines of `NOP;;`.

APPLIES TO REVISION(S):
1.2

12. 03000300 - Link Transmits Port Enable Timing Restrictions:

DESCRIPTION:
TVACANT of the LTSTATx register does not correctly indicate the DLL lock state. A software delay is required between enabling the link transmit port and initializing the corresponding DMA or polling routine responsible for driving the transmit data. The minimum required delay is (75000 x TL) where TL is the period of the corresponding LxCLKO.

Also, when disabling and reenabling a link transmit port, a delay of at least 100 ns is required after the disable and before the reenable.

WORKAROUND:
1. Implement the required delay in software, from enabling of a link transmit port to initialization of the corresponding TCB or polling routine.
2. Implement a 100ns delay in software, from disabling of a link transmit port to reenabling of the same port.

APPLIES TO REVISION(S):
1.2

13. 03000302 - Reading from SDRAM when SDREN bit in SDRCON is clear causes deadlock:

DESCRIPTION:
Reading SDRAM when SDREN bit in SDRCON register is clear will cause deadlock.

WORKAROUND:
Ensure that SDRAM is enabled in SDRCON before accessing it.

APPLIES TO REVISION(S):
1.2, 2.0

14. 03000304 - Interrupt routine for the same interrupt is serviced every time we reach idle with HW interrupt enable bit clear:

DESCRIPTION:
If the processor is at idle, global interrupts enable bit SQCTL[2] is cleared and an interrupt arrives, sequencer operation may become unpredictable.

WORKAROUND:
Always enable global interrupts SQCTL[2] before idle - this is normal operation even without the anomaly.

APPLIES TO REVISION(S):
1.2, 2.0

15. 03000306 - PC of interrupt during idle is corrupted due to stall:

DESCRIPTION:
If an instruction that generates a transaction to the OFIFO occurs within 3 or less core clock cycles prior to an **`IDLE`** instruction, the code execution following an interrupt which breaks the processor out of IDLE may become unpredictable.

WORKAROUND:
Add 3 **`NOP;;`** lines before the Idle:

```
nop;;
nop;;
nop;;
Idle;;
```

and ensure that there are no branches to these **`NOP;;`** and **`IDLE;;`** instruction lines.
Also, to take care of possible returns from interrupts, ensure that either:
1. Last 3 instruction lines of an ISR contain no OFIFO accesses (i.e. no accesses to SOC or external memory).
OR
2. All **`RTI`** and **`RETI`** instructions have option **`(NP)`**. Thus, in case an interrupt occurs during these NOPs, the return from that interrupt will have a pipeline flush that will insert the required number of STALL cycles before the IDLE.

APPLIES TO REVISION(S):
1.2, 2.0

16. 03000315 - Aborted conditional memory access may cause an exception:

DESCRIPTION:
The following instruction

```
if <cond>; do UREG = Q[J/Kx + ...];;
or
if <cond>; do UREG = Q[J/Kx += ...];;
or
if <cond>; do UREG = L[J/Kx + ...];;
or
if <cond>; do UREG = L[J/Kx += ...];;
```

may cause a misaligned or illegal access software exception if the condition is false. The exception can happen even if this aborted access would have been aligned if the condition passed.

WORKAROUND:
1. Disable software exceptions.
2. Recognize a misaligned access or illegal address exception in the software exception handler and exit the handler.

APPLIES TO REVISION(S):
1.2

17. 03000316 - Do not use IALU or Sequencer instructions in the first quad of an ISR:

DESCRIPTION:
An IALU or a sequencer instruction in the first quad word of an interrupt service routine may cause unpredictable results.

WORKAROUND:
Do not use IALU or Sequencer instructions in the first quad of an ISR.

APPLIES TO REVISION(S):
1.2, 2.0

18. 03000319 - Missing exception on illegal read from external memory to sequencer/debug registers:

DESCRIPTION:
A read from external memory to sequencer (CJMP, RETI, RETIB, RETS, DBGE, LC0, LC1, SQCTL) or debug (WPxCTL, WPxL, WPxH, CCNTx, PRFM, EMUCTL, EMUDAT, EMUIR) registers is illegal and should generate an exception. No exception is generated. In case of the sequencer register, EXCAUSE field of SQSTAT is updated with the correct value of 0x7, but in case of the debug register, there is no indication of the illegal access at all.

WORKAROUND:
When loading sequencer or debug registers from external memory, load the external memory data into a UREG first, then move the data from UREG to the actual destination register.

APPLIES TO REVISION(S):
1.2

19. 03000323 - Condition flag not updated after sum/abs instruction is followed by an aborted ALU instruction:

DESCRIPTION:
If an instruction SUM or ABS is followed by an ALU instruction which is aborted (due to the failed condition or change in flow), the first instruction (SUM or ABS) will not update the flags.
Example:

```
xr26=sum  Br21(U);;
if xmle; do, xFr21:20=extd  r25;;
```

If the condition xmle fails, the first instruction will not update the flags. Note that the second instruction may be aborted due to an interrupt and, thus, does not have to be conditional to cause the problem.

WORKAROUND:
Separate the second instruction from SUM/ABS instruction by one instruction. Make sure that there is no predicted jump in parallel with SUM/ABS instruction that, with a BTB hit, branches to a conditional compute instruction.

APPLIES TO REVISION(S):
1.2

20. 03000326 - External DAB Results in Erroneous Results in Certain Cases:

DESCRIPTION:
Read accesses of external memory that use DABs or SDABs may return wrong data.

WORKAROUND:
Do not use external DAB and SDAB read accesses.

APPLIES TO REVISION(S):
1.2

21. 03000328 - Conditional E2 instructions that are aborted still generate exceptions:

DESCRIPTION:
Some illegal instructions that are conditional on CONDE2 (compute block condition) may generate an exception, even if the condition was evaluated as false and the instruction was aborted.

The problem occurs for the following illegal instructions:
1. Two external loads to the same quad register in the same instruction line.
2. Misaligned access by the IALU.
3. Illegal address access by the ILAU.

WORKAROUND:
Ensure that all conditional instructions are legal, even if the condition is expected to evalute as false.

APPLIES TO REVISION(S):
1.2

22. 03000329 - Some exceptions are missed when they follow a not-acknowledged transaction:

DESCRIPTION:
If an instruction that generates an exception follows a memory access with a not-acknowledge, the exception may be missed.

The exception types affected by this problem are:
1. Misaligned access (EXCAUSE=0100).
2. Illegal address (EXCAUSE=0111).
3. Invalid instruction line (EXCAUSE=0011).

In the 1st and 2nd cases, the EXCAUSE field in SQSTAT is updated, indicating that there should have been an exception. In the 3rd case, EXCAUSE is not updated.

WORKAROUND:
Note that none of these exceptions should occur in normally operating systems, they are used for debug only. Exception of case #3 should not occur ever, since the assembler does not generate invalid instructions. If a system is suspected of having illegal accesses, as defined in cases 1 and 2 and does need to be debugged for those accesses, user can setup a timer interrupt that periodically checks EXCAUSE for values of 0100 and 0111.

APPLIES TO REVISION(S):
1.2

23. 03000330 - Link Port Transmit Rise / Fall Times Violate Spec:

DESCRIPTION:
The link port transmit rise and fall times do not meet the data sheet specification and are anomalized as follows:

Parameter	Max
tREO	350ps
tFEO	350ps

WORKAROUND:
None

APPLIES TO REVISION(S):
1.2, 2.0

24. 03000331 - Weakest Drive Strength Setting is Not Functional:

DESCRIPTION:
The weakest drive strength (DS[2:0]=000 if CIMP[1:0]=00 and DS[2:0]=111 if CIMP[1:0]=10) is not functional. Output drivers do not meet specified Voh and Vol levels in this mode.

WORKAROUND:
Do not use DS[2:0]=000 with CIMP[1:0]=00 and do not use DS[2:0]=111 with CIMP[1:0]=10.

APPLIES TO REVISION(S):
1.2, 2.0

25. 03000332 - EXCAUSE is not updated when watchpoint exception occurs and emulation is enabled:

DESCRIPTION:
If there is a watchpoint exception and the emulation is enabled, EXCAUSE field in SQSTAT is not updated, although the flow does branch to the exception vector. This does not affect watchpoints in emulation.

WORKAROUND:
Use watchpoints in software with caution. They can be used to branch execution to the exception vector, but EXCAUSE will not correctly identify the source of the exception.

APPLIES TO REVISION(S):
1.2

26. 03000333 - Single-step software exception is not generated when return from the exception routine is a BTB hit:

DESCRIPTION:
If a return (RTI) from a software exception is a BTB hit, single-step software exceptions will not be generated for a few cycles following the return.

WORKAROUND:
At the end of the exception routine use `RTI (NP)` instead of `RTI`.

APPLIES TO REVISION(S):
1.2, 2.0

27. 03000339 - Locking cache may cause memory access failures:

DESCRIPTION:
Locking any part of memory cache may cause memory access failures.

WORKAROUND:
Do not lock any part of cache.

APPLIES TO REVISION(S):
1.2

28. 03000341 - Link Port LVDS Electrical Characteristics:

DESCRIPTION:

Link ports do not meet electrical characteristics specs. Link Port Electrical Characteristics are anomalized as follows:

Transmit:

Parameter	Min	Max
Voh		1.85V
Vol	0.92V	
Vod	0.34V	0.65V
Vocm	1.20V	1.50V

Receive:

Parameter	Min	Max
Vid	0.22V	0.85V
Vicm	0.60V	1.57V

The anomalized specs are achieved by increasing the link port drive strength from nominal, as described in the workaround.

WORKAROUND:

To enable link port functionality, all link ports must use increased transmit drive strength. This is done by setting the test mode strap TM2 to low and adding the following lines to code before transmitting any data on link ports:

```
j0 = j31+0x9F01;;
TEST_MODES = j0;;
```

APPLIES TO REVISION(S):

1.2

29. 03000342 - Link Port LVDS AC Characteristics:

DESCRIPTION:

The maximum frequency and several timing requirements and characteristics of the link ports do not meet data sheet specifications and are restricted as follows:

Transmit:

Parameter	Description	Conditions	Min	Max	Unit
tLCLKOP	LxCLKOUT Period	tCCLK = 2.00ns	3.0	12.5	ns
tLCLKOP	LxCLKOUT Period	tCCLK = 1.67ns	2.5	12.5	ns
tLDOS	LxDATO Setup	tLCLKOP >= 8.0ns	1.05		ns
tLDOS	LxDATO Setup	8 > tLCLKOP >= 4.0ns	0.525		ns
tLDOS	LxDATO Setup	tLCLKOP < 4.0ns	0.325		ns
tLDOH	LxDATO Hold	tLCLKOP >= 8.0ns	1.05		ns
tLDOH	LxDATO Hold	8 > tLCLKOP >= 4.0ns	0.525		ns
tLDOH	LxDATO Hold	tLCLKOP < 4.0ns	0.325		ns
tLCLKOH	LxCLKOUT High	LCR=1.5	0.25*tLCLKOP	0.70*tLCLKOP	ns
tLCLKOL	LxCLKOUT Low	LCR=1.5	0.30*tLCLKOP	0.75*tLCLKOP	ns

Receive:

Parameter	Description	Conditions	Min	Max	Unit
tLCLKIP	LxCLKIN Period	tCCLK = 2.00ns	3.0	20.0	ns
tLCLKIP	LxCLKIN Period	tCCLK = 1.67ns	2.5	20.0	ns
tLDIS	LxDATI Setup		0.300		ns
tLDIH	LxDATI Hold		0.300		ns

WORKAROUND:

None

APPLIES TO REVISION(S):

1.2

30. 03000347 - Do not update global static flags in the same line with write SQ / BD registers:

DESCRIPTION:
Instructions that update static flags issued in parallel with instructions that update sequencer registers or debug registers may fail to update the static flags properly. Example:

```
ISF0 = LC1E; SQCTLST=0x200;;
```

The instructions that update static flags are of the form:

```
{X|Y|XY}SF1|SF0 = Compute_Cond.;
{X|Y|XY}SF1|SF0 += AND|OR|XOR Compute_Cond.;
ISF1|ISF0 = IALU_Cond.|Compute_Cond.|Seq_Cond.;
ISF1|ISF0 += AND|OR|XOR IALU_Cond.|Compute_Cond.|Seq_Cond.;
```

The sequencer and debug registers affected are:

```
SQCTL, SQCTLST, SQCTLCL, WP0CTL, WP1CTL, WP2CTL, WP0L, WP0H
WP1L, WP1H, WP2L, WP2H, PRFM, PRFCNT
```

WORKAROUND:
Do not issue instructions that update static flags in parallel with instructions that update sequencer or debug registers.

APPLIES TO REVISION(S):
1.2, 2.0

31. 03000348 - Extra writes to AutoDMA may cause cluster bus deadlock:

DESCRIPTION:
A cluster bus deadlock may occur if more transactions are written to the AutoDMA channel than are configured for in the channel TCB. If extra data is written after the channel has completed the specified number of data transfers, the deadlock may occur.

Note: This problem also applies to DMA chaining but only for the last DMA sequence in the chain.

WORKAROUND:
Do not to write more transactions to the AutoDMA channel than configured for in the TCB.

APPLIES TO REVISION(S):
1.2, 2.0

32. 03000349 - Emulation mode allows multiple writes to SYSCON and SDRCON:

DESCRIPTION:
Writing multiple times to SYSCON and SDRCON in supervisor mode is controlled by the SYS_REG_WE strap. When this strap is set to 0, SYSCON and SDRCON can only be written once in supervisor mode. However, emulation mode allows SYSCON and SDRCON to be written multiple times regardless of the strap setting. SYSCON and SDRCON reside outside of the extended core in the SOC domain. If an emulation event (halt, breakpoint, etc.) occurs after a write transaction to these registers has executed but before the transaction gets to the destination register, the write may not be prevented.

WORKAROUND:
Do not write multiple times to SYSCON or SDRCON with the SYS_REG_WE strap set to 0.

APPLIES TO REVISION(S):
1.2, 2.0

33. 03000350 - FLOAT BY instruction with truncate (T) option generates wrong result:

DESCRIPTION:
The `FLOAT BY` instruction with truncation option i.e. `FRs = FLOAT Rm BY Rn (T)`, does not produce correct results under all conditions.

WORKAROUND:
Do not use the `FLOAT BY` instruction with the `(T)` option.

APPLIES TO REVISION(S):
1.2, 2.0

34. 03000351 - Subtract with divide by two instruction with signed round to nearest even option generates wrong result:

DESCRIPTION:
The subtract with divide by two instruction with signed round to nearest even option i.e. `Rs = (Rm - Rn)/2 ( )`, does not produce correct results under all conditions.

WORKAROUND:
Do not use the subtract with divide by two instruction with the signed round to nearest even ( ) option.

APPLIES TO REVISION(S):
1.2, 2.0

35. 03000352 - Link Transmits Port Enable Timing Restrictions:

DESCRIPTION:
When disabling and re-enabling a link transmit port, a delay of at least 100 ns is required after the disable and before the re-enable. Also, after enabling the transmitter, the transmit DLL requires the following number of SOCCLK cycles to stabilize and be ready to transmit (as indicatd by TVACANT of the LTSTATx register):

```
SPD 1:1   = 0x07FFF  SOC cycles
SPD 1:1.5 = 0x0CFFF  SOC cycles
SPD 1:2   = 0x11FFF  SOC cycles
SPD 1:4   = 0x24FFF  SOC cycles
```

This delay is automatically observed by the DMA controller for DMA driven transactions.

WORKAROUND:
Implement a 100ns delay in software, from disabling of a link transmit port to reenabling of the same port.

APPLIES TO REVISION(S):
2.0

36. 03000353 - Conditional E2 instructions that are aborted still update EXCAUSE:

DESCRIPTION:
Some illegal instructions that are conditional on CONDE2 (compute block condition) may update EXCAUSE, even if the condition was evaluated as false and the instruction was aborted.

The problem occurs for the following illegal instructions:
1. Two external loads to the same quad register in the same instruction line.
2. Misaligned access by the IALU.
3. Illegal address access by the ILAU.

WORKAROUND:
There are two options:
1. Do not rely on the EXCAUSE value unless actively servicing an actual software exception.
OR
2. Ensure that all conditional instructions are legal, even if the condition is expected to evaluate as false.

APPLIES TO REVISION(S):
2.0

37. 03000357 - Link Port LVDS Electrical Characteristics:

DESCRIPTION:
Link ports do not meet the data sheet electrical characteristics specifications. The data sheet specifications can only be achieved by increasing the link port drive strength from nominal, as described in the workaround.

WORKAROUND:
To enable link port functionality, all link ports must use increased transmit drive strength. This is done by setting the test mode strap TM2 to low and adding the following lines to code before transmitting any data on link ports:

```
j0 = j31+0x9F01;;
TEST_MODES = j0;;
```

APPLIES TO REVISION(S):
2.0

38. 03000358 - Initial load MR register instruction can fail:

DESCRIPTION:
The very first load MR register instruction can fail to load the correct value.
Affected load MR register instructions are of the form:

```
{X|Y|XY}{S|L} MRa = Rmd {({SE|ZE}{NF})}
{X|Y|XY} MR4 = Rm {(NF)}

where MRa is either MR1:0 or MR3:2
```

For example:

```
MR3:2 = XR5:4;
```

WORKAROUND:
All load MR register instructions will execute successfully if a single transfer MR register instruction precedes the very first load MR register instruction.

Transfer MR register instructions are of the form:

```
{X|Y|XY}{S} Rsd = MRa {({U}{S}{NF})}
{X|Y|XY} Rsq = MR3:0 {({U}{S}{NF})}
{X|Y|XY} Rs = MR4

where MRa is either MR1:0 or MR3:2
```

For example:

```
YR7:6 = MR1:0;
```

This workaround need only be execute once after reset.

APPLIES TO REVISION(S):
2.0

39. 03000359 - Compute block source register busses can fail to initialize properly:

DESCRIPTION:
The compute block source register busses may not initialize properly during reset. If the busses do not initialize properly an erroneous value can be transferred from the compute block register file to the compute units (ALU, CLU, MULTIPLIER, or SHIFTER), and subsequently produce an erroneous result in a compute block instruction.

WORKAROUND:
The compute block source register busses can be deliberately initialized with the following code sequence:

```
MR1:0 += R1:0*R3:2;;
R4 = R0 + R1;
LBUFTX0 = XR3:0;;
LBUFTX0 = YR3:0;;
```

This code sequence should be executed prior to execution of any meaningful compute block instruction. The code sequence need only be execute once after reset.

APPLIES TO REVISION(S):
2.0

40. 03000360 - Predicated RETIB load (store) instructions fail:

DESCRIPTION:
A predicated RETIB load (store) instruction can partially execute when the predicated condition is evaluated as FALSE. A predicated RETIB load (store) instruction is of the form:

```
IF <cond>; DO, <RETIB load (store)>;;
IF <cond>, JUMP|CALL|CJMP|CJMP_CALL; ELSE, <RETI load (store)>;;
```

WORKAROUND:
A predicated RETIB load (store) instruction can be executed with the following code sequence:

```
IF <cond>, JUMP _past;;
<RETIB load (store)>;;
<predicated instruction>;;
_past:
<non-predicated instruction>;;
```

The RETIB load (store) instruction and predicated instruction(s) will be executed if the branch condition is FALSE. If the branch condition is TRUE, the RETIB load (store) instruction and predicated instruction(s) will be skipped. The non-predicated instruction(s) will always be executed.

APPLIES TO REVISION(S):
1.2, 2.0

41. 03000364 - AC Signal Specifications - Input Hold:

DESCRIPTION:
The input hold time requirement does not meet the data sheet specification for all signals. The signals affected and the associated actual input hold time requirements are as follows:

Name	Description	Input Hold (min)	Unit
ADDR31-0	External Address Bus	0.75	ns
DATA63-0	External Data Bus	0.75	ns
ACK	Acknowledge for Data High to Low	0.75	ns
ACK	Acknowledge for Data Low to High	0.75	ns
$\overline{\text{HBR}}$	Host Bus Request	0.75	ns
$\overline{\text{BOFF}}$	Back Off Request	0.75	ns
$\overline{\text{BRST}}$	Burst Pin	0.75	ns
$\overline{\text{BR7-0}}$	Multiprocessing Bus Request Pins	0.75	ns

WORKAROUND:
None

APPLIES TO REVISION(S):
1.2

42. 03000367 - Link Port Transmit 4-bit mode enable failure:

DESCRIPTION:
The link port transmitter generates erroneous transitions on LxCLKOUT whenever the TDSIZE bit in the LTCTLx registers is transitioned from 0 to 1. These erroneous transitions may result in communication failures between the transmitter and the receiver.

WORKAROUND:
Booting:
Link Port booting from a TS20x master must be performed in 1-bit mode only.

Options after successfully booting (by any method) include the following:

1) Use 1-bit mode always for TS20x link port transmit.
2) Disable and re-enable the receiver after the TS20x link port transmitter has been enabled in 4-bit mode and before the transmitter transfers valid data.

For option 2, in order to ensure synchronization, the transmitter must indicate to the receiver that the transmit link port is enabled. The receiver must then indicate to the transmitter that the receive link port has been disabled and re-enabled. This can be done through flags or a cluster bus connection if possible.

If no additional connection exists other than a link port connection, the following sequence may be used to ensure synchronization between transmitter and receiver:

Transmitter:
1) Send 4 QW to receiver (fills the receiver's fifo and leaves at least one QW in the transmit buffer.
2) Poll LTSTATn for "empty" status.
3) Execute delay for sufficient time to allow receiver to disable and re-enable. This step is relatively in sync with step 3 of the receiver below.
4) Return to normal flow.

Receiver:
1) Enable core ISR for LINKn.
2) Read the 4 QW in the core ISR (using polling of LRSTATn).
3) Disable and re-enable REN bit in LRCTLn. This step is relatively in sync with step 3 of the transmitter above.
4) Disable core ISR for LINKn.
5) Return to normal flow.

APPLIES TO REVISION(S):
2.0

43. 03000368 - Link port receive enable after disable failure:

DESCRIPTION:
Enabling the link port receiver too soon after disabling it may result in the receiver's FIFO failing to be reset. This will in turn result in subsequent data reception failures.

WORKAROUND:
Insert a dummy read from the SOC domain between the disable and re-enable of the link port receiver such as in the following:

```
LRCTL0 = xR0;;     // Disable LP0 RX
xR0 = LRCTL0;;     // Dummy SOC read transaction
LRCTL0 = xR1;;     // Re-enable LP0 RX
```

APPLIES TO REVISION(S):
1.2, 2.0

44. 03000371 - SCLK_Vref Specification:

DESCRIPTION:
The data sheet specification for SCLK_Vref does not apply to the following operating conditions:

Parameter	Description	Value	Unit
Vdd	Vdd for internal logic	<=1.10	V
Vclock_drive	Voltage range for SCLK input	>Vdd_io	V
SCLKRAT2:0	Core to System Clock Ratio	000 (4x) or 001 (5x)	-

WORKAROUND:
SCLK_Vref must be set to (Vclock_drive x 0.50)+/-5%

APPLIES TO REVISION(S):
1.2, 2.0

45. 03000372 - TIMER enable/disable restriction:

DESCRIPTION:
Disabling the timer at the exact moment that it expires can block the reload of the timer period. A subsequent re-enable will result in the TIMER decrementing from the largest possible value (0xFFFFFFFF-FFFFFFFF).

WORKAROUND:
If disable and re-enable of the TIMER is required, the TIMER value should be manually reloaded before re-enabling. If preserving the present count is required, the following sequence should be used:

1) Read the TIMERHx register.
2) Disable the timer.
3) Read the TIMERLx.
4) If TIMERHx=0x00000000 (from step 1) AND TIMERLx=0xFFFFFFFF (from step 3) then reload the TMRIN registers manually.
5) Re-enable the timer.

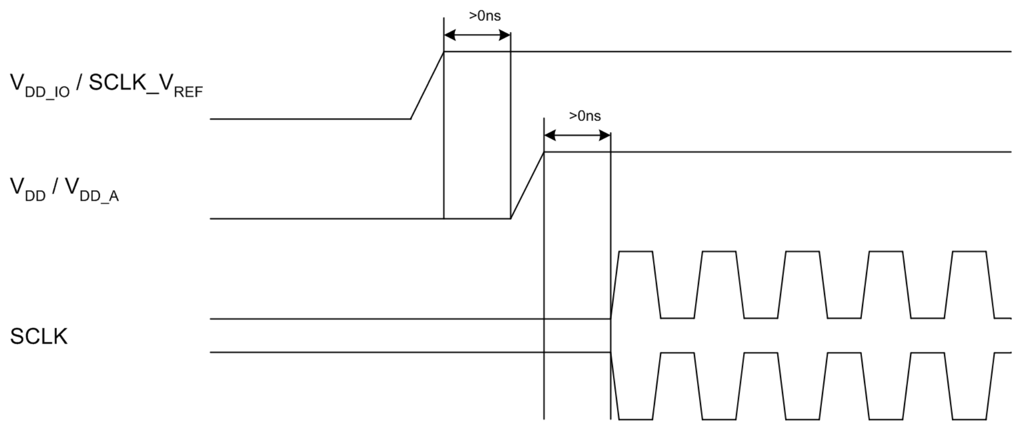
Step 1 is to ensure that the timer did not just correctly decrement from 0x00000001 00000000.

APPLIES TO REVISION(S):
1.2, 2.0

46. 03000373 - PLL Failure to Lock at Power-Up:

DESCRIPTION:
The PLL may fail to lock correctly to the SCLK input during power-up.

WORKAROUND:
Vdd/Vdd_a must be held low until after Vdd_io and SCLK_Vref are stable and within specification. Additionally, SCLK must be held high or low until after Vdd/Vdd_a is stable and within specification. The following figure illustrates the workaround:



APPLIES TO REVISION(S):
1.2, 2.0

47. 03000375 - DMA for link port to/from external memory may be blocked:

DESCRIPTION:
DMA between link ports and external memory may be blocked by an external port DMA. If the DMA sequences are concurrent, transactions for the link port DMA to/from external memory may not occur at the expected rate and may even be blocked until the completion of the external port DMA (internal to external or external to internal memory).

This problem occurs only for the following specific SCLKRAT settings:

SCLKRAT = 4, 6, 8, 10, or 12

WORKAROUND:
Divide large external port DMAs into multiple, smaller count DMAs to allow link port to/from external memory DMAs transactions to be granted more frequently.

APPLIES TO REVISION(S):
1.2, 2.0

48. 03000376 - Link port receive to link port transmit DMA fails:

DESCRIPTION:
Link port receive to link port transmit DMA does not work properly. The counter of the link port receiver DMA channel updates erroneously when the link transmit buffer is full.

WORKAROUND:
Do not use link port receive to link port transmit DMA. Use DMA from link port receiver to memory followed by DMA from memory to link port transmitter to achieve the same data flow.

APPLIES TO REVISION(S):
1.2

49. 03000377 - Internal DRAM Corruption Under Specific Conditions:

DESCRIPTION:

If a read transaction of an internal memory location immediately happens to follow the last cycle of a refresh command to that same memory page, data corruption of the memory location may occur. This error manifests itself in the form of bit flips of memory locations, with zeros returned in the place of expected logical ones. This error is frequency dependent and only applies to 600 MHz parts.

WORKAROUND:

The intent of this workaround is to ensure that page refreshes of internal memory are allowed to happen unhindered by any other system activity. In order to do this, it is necessary to enable a special test mode by setting the test mode strap TM2 to low and adding the following lines to code:

```
j0 = j31 + 0xFF01;;
TEST_MODES = j0;;
```

The user needs to incorporate the workaround code at the beginning of their application, and allocate the High-Priority Timer1 (the highest user-defined interrupt) for the workaround. The following steps need to be followed to implement the workaround in software. The accompanying example code assumes that the main application is written in C, while the HP Timer1 ISR is written in assembly language.

1) In the initial startup code, program the High-Priority Timer1 period to a value equal to refresh rate, enable interrupts, and set up the interrupt service routine (ISR) at a pre-determined memory location within internal memory.

```
// stop Timer 1 in case it is already enabled for any reason
iTemp = __builtin_sysreg_read(__INTCTL);
iTemp &= ~INTCTL_TMR1RN;
__builtin_sysreg_write(__INTCTL, iTemp);


//now cleanly re-configure HPTimer1 and ISR
// clear existing IRQs:
__builtin_sysreg_write(__ILATCLH, INT_TIMER1H);
// write pointer of ASM-routine to IRQ vector table:
__builtin_sysreg_write(__IVTIMER1HP, (unsigned int)(void*)_HPTimer1Isr);

// Init Timer 1
#define TIMER1_PERIOD (((4499+1)/2)-2)
// Note required refresh rate for 600 MHz operation is 4499
__builtin_sysreg_write(__TMRIN1L, TIMER1_PERIOD);
__builtin_sysreg_write(__TMRIN1H, 0);
// Un-mask interrupt
iTemp = __builtin_sysreg_read(__IMASKH);
iTemp |= INT_TIMER1H;
__builtin_sysreg_write(__IMASKH, iTemp);
// Enable Global Interupt (GIE):
iTemp = __builtin_sysreg_read(__SQCTL);
__builtin_sysreg_write(__SQCTL, (iTemp | SQCTL_GIE));
```

2) The code within the High-Priority Timer1 ISR will keep cache enabled, turn on the refresh rate and set the refresh counter value to a small number, and remain within the ISR until the refresh counter expires. This will ensure that a refresh signal will be propagated to internal memory. At the end of the ISR, refresh is turned off while cache is kept enabled. Ensure that the ISR is placed at a specific location in internal memory (in case of this example, it is placed at location 0x40000).

```
#include <defts201.h>
#include "signal.h"
#include "cache_macros.h"

#define _JPUSH 8
#define DIS_REFRESH (0x60000000)
```

```
.section seg_Timer1Isr; // defined to be at 0x40000 in LDF file

.global __HPTimer1Isr;

.align_code 4;
__HPTimer1Isr:
nop; nop; nop; nop;; // Workaround for anomaly 03-00-0316 (ADSP-TS20x)
// save everything we use!
// J-register
q[j27 += -_JPUSH]    = j31:28;;  // J31=old JSTAT or Zero for indexed address,
// J30=scratch, J29=scratch, J28=unused

// Now enable refresh rate and set to a small value (4 in this case)
J29 = j31 + CACMD_EN + CACMD_REFRESH | 4;;
CACMDALL= J29;;

// Wait until refresh counter expires, with enough NOPs.
// The extra NOPs ensure that refresh goes through, since the refresh counter
// runs at core clock frequency, while memory refresh happens at half that frequency.
nop;; nop;; nop;; nop;; nop;; nop;; nop;; nop;;

//keep cache enabled, but disable refresh until next Timer 1 HP interrupt
J29 = j31 + DIS_REFRESH + CACMD_EN ;;
CACMDALL = J29;;
nop;;nop;;nop;;nop;;nop;;

// restore J-register
j31:28  = q[j27 + _JPUSH];;

rti (abs) (np);         // return from interrupt, restoring
nop;
[j27 += _JPUSH] = j1;;  // Important: Post-increment for stack pointer;

.__HPTimer1Isr.end:
```

Optionally: While the example workaround code shown here refreshes one page per occurrence of the HP Timer1 ISR, a customer can choose to modify the workaround further by reducing the timer frequency by a factor of N (i.e., increasing the timer period by a factor of N), and add more NOPs inside the HP Timer1 ISR so that N pages are now refreshed per occurrence of the HP Timer1 ISR. The trade-off is that the system will now be interrupted less often by this timer ISR, but the increased length spent within the ISR will lock out everything else in the system for the duration of the ISR.

3) Lock the ISR code for High-Priority Timer1 in cache. The following function is called in the main code and illustrates how to do this, assuming that the starting memory location of the Timer1 HP ISR is at address 0x40000.

```
void LockTimer1IsrInCache(void)
{
  // Check location of ISR here! Has to be 0x40000
  asm("#include <defts201.h>");
  asm("J0= 0x00000000;;" // Address Offset from the start of the block
      "CCAIR2= J0;;"     // (Offset = 0, so we start at the beginning of block 2 = 0x40000)
      // Length = Number of cache needed ways - 1
      // Each cache has 128 sets
      // Each set has 4 ways
      // Each way has 256 bits = 8*32bit words
      // There is a lock bit for every way
      "J2 = CACHE_INIT_LOCK | (0x06 << CACMD_LEN_P) | CACMD_NOSTALL;;"
      "CACMD2 = j2;;"
      "nop;;nop;;nop;;nop;;nop;;"
      "_chk_castat:"
      "xr0 = CASTAT2;;"
      "xbitest r0 by CASTAT_COM_ACTIVE_P;;"
      ".align_code 4;"
      "if nxseq, jump _chk_castat (NP);nop;nop;nop;;"
      "nop;;nop;;nop;;nop;;");
}
```

Optionally; If the option described in step 2 above is applied to increase the number of pages to be refreshed in a single interrupt, the user must insure that the number of cache ways locked provides for sufficient space in locked cache to hold the increased size of the ISR. In the example above, 0x06 ways are locked. These 0x06 ways are sufficient to hold the ISR provided in the example above.

4) Turn off refresh but keep cache enabled. Enable High-Priority Timer1.

```
asm("j0 = j31+0x0001;;"
    "TEST_MODES = j0;;"
    "nop;;nop;;nop;;nop;;nop;;nop;;"
    "nop;;nop;;nop;;nop;;nop;;nop;;"
    "j0 = j31 + DIS_REFRESH + CACMD_EN;;"
    "CACMDALL = j0;;"
    "nop;;nop;;nop;;nop;;nop;;nop;;"
    "nop;;nop;;nop;;nop;;nop;;nop;;");

// Finally, "enable" HP Timer1
asm(
    "xr2 = intctl;;"                        // read INTCTL
    "xr3 = BSET r2 by 0X5(nF);;"            // set  INTCTL_TMR1RN bit
    "intctl = xr3;;");
```

5) The above sequence can now be followed by the rest of the application code.

APPLIES TO REVISION(S):

1.2, 2.0