High Performance SHARC-FX DSP Core With Arm-Based Connectivity

Silicon Anomaly List

ADSP-21834/21835/21836/21837/ADSP-SC834/SC835

ABOUT ADSP-21834/21835/21836/21837/ADSP-SC834/SC835 SILICON ANOMALIES

These anomalies represent the currently known differences between revisions of the SHARC®-FX ADSP-21834/21835/21836/21837/ADSP-SC834/SC835 product(s) and the functionality specified in the ADSP-21834/21835/21836/21837/ADSP-SC834/SC835 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. The REVID bits <31:28> of the TAPC0_IDCODE register can be used to differentiate the revisions as shown below.

Silicon REVISION	TAPC0_IDCODE.REVID
0.0	b#0000

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	Data Sheet Revision	Additions and Changes
04/03/2024	B	PrC	Initial Version

One Analog Way, Wilmington, MA 01887 U.S.A.

SUMMARY OF SILICON ANOMALIES

The following table provides a summary of ADSP-21834/21835/21836/21837/ADSP-SC834/SC835 anomalies and the applicable silicon revision(s) for each anomaly.

No.	ID	Description	Rev 0.0
1	25000002	Transactions on SPU and SMPU MMR Regions May Cause Errors	x
2	25000004	GP Timer Generates First Interrupt/Trigger One Edge Late in EXTCLK Mode	x
3	25000005	Writes to the SPI_SLVSEL Register Do Not Take Effect	x
4	25000006	Unreliable SPDIF Receiver Clock Output Pulse Width at Sample Rates Above 96KHz	x
5	25000007	ASRCxMUTE interrupt does not get latched when ASRC mute out signal is de-asserted for ASRC8 to ASRC15 instances.	x
6	25000008	While using alternate pin-mux option provided in Boot ROM for SPI0 and SPI2 Target boot mode, Target select signal does not work as expected.	x
7	25000010	asm("Break1 15") instruction is not working as expected.	x
8	25000011	Possibilities of Spurious Parity Error detection from Cortex-M33 RAM & ROM	x
9	25000013	Memory Error Exception could be incorrectly reported for correctable errors on L1 instruction RAM on the SHARC-FX core	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-21834/21835/21836/21837/ADSP-SC834/SC835 including a description, workaround, and identification of applicable silicon revisions.

1. 25000002 - Transactions on SPU and SMPU MMR Regions May Cause Errors:

DESCRIPTION:
Non-secure reads or writes to the upper half of each SPU and SMPU instance's MMR space will be erroneously blocked and cause a bus error when configured as a non-secure completer.

For each instance of the SPU and SMPU, the affected MMR address range can be calculated as follows:
- Lower bound = Instance Address Offset + **`0x800`**
- Upper bound = Instance Address Offset + **`0xFFF`**

WORKAROUND:
Do not access the documented system MMR ranges from a non-secure completer.

APPLIES TO REVISION(S):
0.0

2. 25000004 - GP Timer Generates First Interrupt/Trigger One Edge Late in EXTCLK Mode:

DESCRIPTION:
When any GP Timer is configured in External Clock mode, the first interrupt/trigger should occur when the corresponding **`TIMER_DATA_ILAT`** bit sets after the **`TIMER_TMRn_CNT`** register reaches the value programmed in the **`TIMER_TMRn_PER`** register. Instead, the interrupt/trigger and the setting of the **`TIMER_DATA_ILAT`** bit occur one signal edge later. At this point, the **`TIMER_TMRn_CNT`** register will have rolled over to 1. Subsequent interrupts/triggers occur after the correct number of edges.

For example, if **`TIMER_TMRn_PER`**=7, the first interrupt/trigger will occur after the timer pin samples eight edges. From that point forward, interrupts/triggers will correctly occur every seven signal edges.

WORKAROUND:
For interrupts/triggers to occur every **n** edges detected on the timer pin, the **`TIMER_TMRn_PER`** register must be configured to **n-1** for the initial event and then reprogrammed to **n** for subsequent events, as shown in the following pseudocode:

```
TIMER_TMRn_PER = n-1;    // Configure PERIOD register with n-1
TIMER_RUN_SET = 1;       // Enable the timer
TIMER_TMRn_PER = n;      // Configure PERIOD register with n
```

The second write to the **`TIMER_TMRn_PER`** register does not take effect until the 2nd period; therefore, this sequence can be performed when the timer is first enabled.

APPLIES TO REVISION(S):
0.0

3. 25000005 - Writes to the SPI_SLVSEL Register Do Not Take Effect:

DESCRIPTION:
A single write to the **`SPI_SLVSEL`** register should change the state of the register and cause the modified software-controlled SPI target selects to assert or de-assert. Instead, a single write to **`SPI_SLVSEL`** has no effect.

WORKAROUND:
Any write to **`SPI_SLVSEL`** should be done twice (back-to-back) with the same value in order for the change to take effect.

APPLIES TO REVISION(S):
0.0

4. 25000006 - Unreliable SPDIF Receiver Clock Output Pulse Width at Sample Rates Above 96KHz:

DESCRIPTION:
When the sampling rate of the SPDIF receiver input stream (FS_Rate) is above 96 KHz, the positive pulse width of the SPDIF receiver TDM output clock (SPDIF_RX_TDMCLK_O) can be as low as the period of the SPDIF receiver module clock, and the negative pulse width can be as high as one SPDIF receiver module clock period less than the ideal clock period. As a result, audio peripherals such as the ASRC, SPORT, and DAI pins may not function properly when SPDIF_RX_TDMCLK_O is used as the clock source.

WORKAROUND:
Do not use the SPDIF_RX_TDMCLK_O output clock as the source for external peripherals when the FS rate is above 96 KHz.

The Precision Clock Generator (PCG) can be used to divide the clock down such that audio peripherals like the ASRC, SPORT, and DAI pins may function internally; however, the clock and frame sync outputs from the PCG will still exhibit the duty cycle problem and must not be used to interface with external components.

APPLIES TO REVISION(S):
0.0

5. 25000007 - ASRCxMUTE interrupt does not get latched when ASRC mute out signal is de-asserted for ASRC8 to ASRC15 instances.:

DESCRIPTION:
The ASRC mute out signal is asserted (= High) when the ASRC starts up. When the ASRC locks to a new sampling rate, the mute out signal is de-asserted (= Low). The DAI interrupt registers(DAI_IMSK_FE, DAI_IMSK_RE) are programmed to generate an interrupt to the core on rising edge or falling edge of ASRC mute out signal. The DAI has its own interrupt controller which indicates to the core of any audio peripheral event. The mute out signals for ASRC8 to ASRC15 instances have been mapped over a level-sensitive line to the DAI interrupt controller. As a result, the DAI interrupt controller is able to detect only the rising edge of ASRC mute out signal, but misses the falling edge of ASRC mute out signal.
Since the DAI interrupt controller does not detect the falling edge of ASRC mute out signal, this problem is observed for interrupt routed either directly by core or via SEC for DAI interrupts.

WORKAROUND:
Poll for ASRC_RAT.MUTEOUTx bit instead of relying on ASRCxMUTE interrupt in an application. Once the ASRC_RAT.MUTEOUTx bit is cleared, the output can be read as the ASRC is now locked to new sampling rate.

APPLIES TO REVISION(S):
0.0

6. 25000008 - While using alternate pin-mux option provided in Boot ROM for SPI0 and SPI2 Target boot mode, Target select signal does not work as expected.:

DESCRIPTION:
When alternate pin-mux bit (ROM_BCMD_SPI_ALTMUX) is set in dbootcommand for SPI0 and SPI2 target boot mode, the target select signal (SSb) on (PE_15) for SPI0 and (PE_02) for SPI2 should be enabled. This is not enabled by boot ROM. The issue is applicable for both power on reset and bootROM API calls.

WORKAROUND:
1. For Power On Reset boot Scenario:
   a. Instead using the target select signal (SSb) on PE_15 for SPI0 and PE_02 for SPI2, user can use PA_09 for SPI0 and PA_05 for SPI2 target select signals, respectively.
   b. If Hyper RAM or Hyper Flash is connected to xSPI (PORT A), then SPI1 can only be used for SPI target boot mode to avoid pin conflicts on PORT A.

2. For Boot ROM API Scenario:
   a. User should configure PADS_CFG1 register to enable alternate target select signals on pins PE_15 and PE_02 before calling ROM API. Following register should be configured for enabling the signals.
      i. For SPI0:
      ```
      *pREG_PADS0_PCFG1 |=(1<<26)
      ```
      ii. For SPI2:
      ```
      *pREG_PADS0_PCFG1 |= (1<<27)
      ```

APPLIES TO REVISION(S):
0.0

7. 25000010 - asm("Break1 15") instruction is not working as expected.:

DESCRIPTION:
Execution of asm("Break1 15") instruction raises an exception. The exception handler in CCES works around the issue by executing a "Break.n 1" instruction to halt the application.

WORKAROUND:
Execution of asm("Break1 15") instruction raises an exception. The exception handler in CCES works around the issue by executing a "Break.n 1" instruction to halt the application.

APPLIES TO REVISION(S):
0.0

8. 25000011 - Possibilities of Spurious Parity Error detection from Cortex-M33 RAM & ROM:

DESCRIPTION:
Parity error detection for Cortex-M33's ROM and RAM(L1 I-RAM and D-RAM) gets registered in MEPU. When Cortex-M33-CLK and SYSCLK are not equal, accesses to Cortex-M33 RAM and ROM memory can cause spurious parity error detection to be registered in MEPU.

WORKAROUND:
1. Parity error detection works fine when Cortex-M33-CLK and SYSCLK are in 1:1 sync mode, i.e when CCLK = 800 MHz/600 MHz, Cortex-M33-CLK can be configured to 400/300 MHz(sourced from SYSCLK)

2. When CCLK = 1 GHz, parity error interrupt generation must be disabled in MEPU. With CCLK = 1 GHz and parity error detection disabled, Cortex-M33-CLK can be configured to 333.3 MHz(sourced from SCLK1_0).

APPLIES TO REVISION(S):
0.0

9. 25000013 - Memory Error Exception could be incorrectly reported for correctable errors on L1 instruction RAM on the SHARC-FX core:

DESCRIPTION:
If an l32r , l32i or l32i.n instruction to load from IRAM is followed by an unconditional immediate branch, call, or return instruction, the branch might affect memory error correction to IRAM. If one of these loads from IRAM detects a single-bit error, the memory error detection hardware must write corrected data back to IRAM in a late stage of the pipeline. An unconditional branch that is handled in the R2-stage of the pipeline could disable such a write back to IRAM in a later stage. Then when a replay occurs to re-read the IRAM data, this could result in a hard error being signaled. The problem could occur if an l32i.n or l32i load from IRAM is followed 1 or 2 cycles later in the pipeline by an early branch instruction (CALL0, CALL8, J, RFDO). The issue could also occur if an l32r load from IRAM is followed 1, 2 or 3 cycles later by an early branch instruction.

WORKAROUND:
If a memory error exception is reported, the exception handler could determine whether the exception was caused by a correctable single-bit error in IRAM that was signaled due to this anomaly, or due to an uncorrectable memory error.

1. The exception handler could load from the memory error address in IRAM, using an instruction sequence that is unaffected by this anomaly. If the memory error exception was triggered by this anomaly, then such a sequence to load from IRAM will correct the single-bit error.
2. However, if reading from the memory error address in IRAM causes another memory error, then it was caused by an uncorrectable hard error.

APPLIES TO REVISION(S):
0.0