High Performance SHARC-FX DSP Core With ARM-Based Connectivity/Security

Silicon Anomaly List

ADSP-21844/ 21846/ ADSP-SC844/SC846

ABOUT ADSP-21844/21846/ADSP-SC844/SC846 SILICON ANOMALIES

These anomalies represent the currently known differences between revisions of the SHARC®-FX ADSP-21844/21846/ADSP-SC844/SC846 product(s) and the functionality specified in the ADSP-21844/21846/ADSP-SC844/SC846 data sheet(s) and the Hardware Reference Manual(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 REVISIONS

Silicon REVISION	TAPC0_IDCODE.REVID
0.1	b#0001

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
06/23/2026	A	PrD	Initial Version

NR004995A

SUMMARY OF SILICON ANOMALIES

The following table provides a summary of ADSP-21844/21846/ADSP-SC844/SC846 anomalies and the applicable silicon revision(s) for each anomaly.

No.	ID	Description	Rev 0.1
1	DSPSI-23	Transactions on SPU and SMPU MMR Regions May Cause Errors	x
2	DSPSI-25	GP Timer Generates First Interrupt/Trigger One Edge Late in EXTCLK Mode	x
3	DSPSI-26	Writes to the SPI_SLVSEL Register Do Not Take Effect	x
4	DSPSI-27	Unreliable SPDIF Receiver Clock Output Pulse Width at Sample Rates Above 96KHz	x
5	DSPSI-33	Correctable SHARC-FX L1 IRAM Errors May Generate Memory Error Exception	x
6	DSPSI-36	Boot ROM May Encounter Error When CRC Protection Is Enabled with Page Mode Enabled	x
7	DSPSI-41	Universal Serial Bus (USB) Controller and Associated USB/GPIO Pins Are Not Functional	x
8	DSPSI-42	EMSI Boot from eMMC User Area May Fail Due to Missing CMD7 Response During Cleanup	x

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

DETAILED LIST OF SILICON ANOMALIES

The following list details all known anomalies for the ADSP-21844/21846/ADSP-SC844/SC846 including a description, workaround, and identification of applicable silicon revisions.

1. DSPSI-23 - Transactions on SPU and SMPU MMR Regions May Cause Errors:

DESCRIPTION:

When configured as a non-secure completer, non-secure reads or writes to the MMR space of the upper half of each SPU and SMPU instance are erroneously blocked and cause a bus error.

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):

0.1

2. DSPSI-25 - GP Timer Generates First Interrupt/Trigger One Edge Late in EXTCLK Mode:

DESCRIPTION:

When any GP Timer is configured in External Clock (EXTCLK) 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 events 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 occurs 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):

0.1

3. DSPSI-26 - 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):

0.1

4. DSPSI-27 - 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):

0.1

5. DSPSI-33 - Correctable SHARC-FX L1 IRAM Errors May Generate Memory Error Exception:

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 determines 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 uses an instruction sequence that is unaffected by this anomaly to load from the memory error address in IRAM. If the memory error exception was triggered by this anomaly, then such a sequence to load from IRAM will correct the single-bit error.
1. 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):

0.1

6. DSPSI-36 - Boot ROM May Encounter Error When CRC Protection Is Enabled with Page Mode Enabled:

DESCRIPTION:

If CRC protection of the payload is enabled with page mode enabled, the Boot ROM may encounter an error and jump to the error handler, resulting in a boot failure.

WORKAROUND:

If CRC protection of the payload is needed, the page mode must be disabled. If page mode is enabled by default, it can be disabled by programming **`dBootCmd`** in the OTP or through a second-stage boot.

APPLIES TO REVISION(S):

0.1

7. DSPSI-41 - Universal Serial Bus (USB) Controller and Associated USB/GPIO Pins Are Not Functional:

DESCRIPTION:

The Universal Serial Bus (USB) controller is not supported, and the pins corresponding to USB signals cannot be used as GPIO pins.

WORKAROUND:

Do not use the USB controller, and do not enable GPIO functions on the PH10:15 or PI00:05 pins.

APPLIES TO REVISION(S):

0.1

8. DSPSI-42 - EMSI Boot from eMMC User Area May Fail Due to Missing CMD7 Response During Cleanup:

DESCRIPTION:

EMSI boot from eMMC devices may fail when booting from the user area partition because the eMMC device does not respond to CMD7. After the boot kernel finishes loading the boot stream, it calls a cleanup function that sends CMD7 to select the device. The EMSI controller then waits for a response from the eMMC device. However, most eMMC devices do not respond to CMD7 when they are being deselected. As a result, the EMSI controller never receives a response, and the Boot ROM hangs in the cleanup function instead of jumping to the user application.

This issue affects both Power-on reset boot and Boot ROM API-based eMMC boot flows. Booting from the boot area is not affected because CMD7 is not sent during cleanup when booting from the boot area.

WORKAROUND:

To avoid this issue, the EMSI cleanup function should be bypassed so that the eMMC deselect command (CMD7) is not issued at the end of the boot process. Skipping CMD7 prevents the EMSI controller from waiting indefinitely for a response and allows the EMSI boot flow to complete successfully and transfer execution to the loaded application.

Because the cleanup function is skipped, the user application is responsible for restoring the EMSI controller state. Specifically, the application must restore the EMSI registers to their reset default values and deselect and reinitialize the eMMC device before using it, as follows:

1. For power-up boot of non-secure applications, Initcode shall be used to set the **`BITM_ROM_BFLAG_NORESTORE`** flag in **`pBootConfig->dFlags`**. This will ensure the cleanup routine is not executed when EMSI boot completes.

```
if (pBootStruct != NULL)
{  pBootStruct->dFlags |= BITM_ROM_BFLAG_NORESTORE; }
```

2. For power-up boot of secure applications, enable secure attribute **`0x80000001`** in the secure boot stream and set bit 0 in its value. This prevents the cleanup routine from being executed when the secure EMSI boot flow completes.

3. For ROM API boot, set the **BITM_ROM_BFLAG_NORESTORE** flag in **pBootConfig->dFlags** and pass the flags in the boot API call as shown below:

```
adi_rom_Boot( Address, BITM_ROM_BFLAG_NORESTORE, 0, 0, emsi_BootCommand,  0 );
```

APPLIES TO REVISION(S):

0.1