66AK2Hxx
Multicore DSP+ARM KeyStone II SOC
Silicon Revisions 1.0, 1.1, 2.0, 3.0, 3.1

Silicon Errata

Literature Number: SPRZ402F
June 2013–Revised June 2018

Contents

1 Device and Development Support Tool Nomenclature 4
2 Package Symbolization and Revision Identification 5
3 ARM-Specific Information 7
4 Silicon Updates 7
Revision History 86

List of Figures

1 Lot Trace Code Example for 66AK2Hxx (AWW Package) 5
2 SRIO SerDes in Loopback Mode 20
3 Software Reset Using the RSTCTRL Register 30
4 Reset Source Code 31
5 DDR3 PHY 43
6 Satellite TeraNet Paths 46
7 Hang Scenario 47
8 Workaround 1 49
9 Workaround 2 50
10 RX Path Block Diagram 84

List of Tables

1 Lot Trace Codes 5
2 Silicon Revision Variables 6
3 Cortex-A15 Processor Version and REVIDR 7
4 Silicon Revision 1.0, 1.1, 2.0, 3.0, and 3.1 Updates 7
5 DDR3A 19
6 DDR3B 19
7 Impact on Various Lane Speeds 23
8 Master Behavior in Response to sstatus/rstatus Flagged Due to DDR3 ECC Errors 27
9 DDR3-1600 43
10 DDR3-1333 44
11 DDR3-1066 44
12 DDR3-800 45
13 SerDes peripherals impacted by RX Boost equalization problem 84

66AK2Hxx Multicore DSP+ARM KeyStone II SOC Silicon Revisions 1.0, 1.1, 2.0, 3.0, 3.1

This document describes the silicon updates to the functional specifications for the 66AK2Hxx family of fixed-/floating-point digital signal processors, including the 66AK2H06, 66AK2H12, and 66AK2H12. See the device-specific data manual for more information.

1 Device and Development Support Tool Nomenclature

To designate the stages in the product development cycle, TI assigns prefixes to the part numbers of all devices and support tools. Each family member has one of two prefixes: X or [blank]. These prefixes represent evolutionary stages of product development from engineering prototypes through fully qualified production devices/tools.

Device development evolutionary flow:

- **X:** Experimental device that is not necessarily representative of the final device's electrical specifications
- **[Blank]:** Fully qualified production device

Support tool development evolutionary flow:

- **X:** Development-support product that has not yet completed Texas Instruments internal qualification testing.
- **[Blank]:** Fully qualified development-support product

Experimental (X) and fully qualified [Blank] devices and development-support tools are shipped with the following disclaimer:

- ***Developmental product is intended for internal evaluation purposes***.

Fully qualified and production devices and development-support tools have been characterized fully, and the quality and reliability of the device have been demonstrated fully. TI's standard warranty applies.

Predictions show that experimental devices (X) have a greater failure rate than the standard production devices. Texas Instruments recommends that these devices not be used in any production system because their expected end-use failure rate still is undefined. Only qualified production devices are to be used.

TI device nomenclature also includes a suffix with the device family name. This suffix indicates the package type (for example, AAW), the temperature range (for example, blank is the default case temperature range), and the device speed range, in Megahertz (for example, blank is 1000 MHz [1 GHz]).

For device part numbers and further ordering information for 66AK2Hxx in the AWW package type, see the TI website www.ti.com or contact your TI sales representative.

2 Package Symbolization and Revision Identification

The device revision can be determined by the lot trace code marked on the top of the package. The location of the lot trace code for the AWW package is shown in Figure 1. The figure also shows an example of 66AK2Hxx package symbolization.

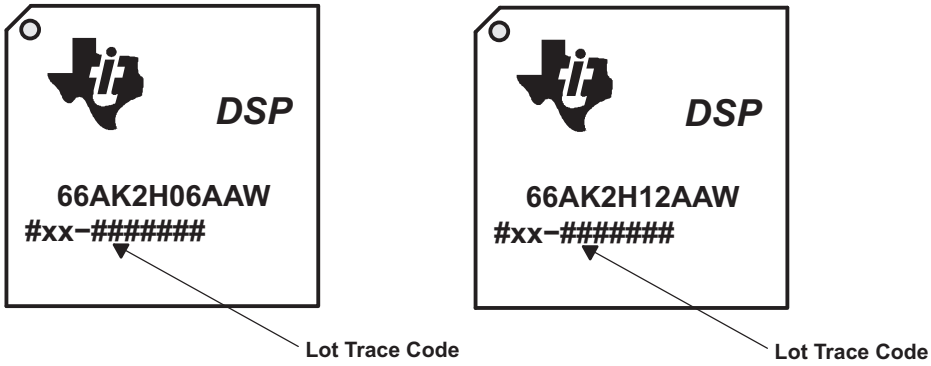


Figure 1. Lot Trace Code Example for 66AK2Hxx (AWW Package)

Silicon revision correlates to the lot trace code marked on the package. This code is of the format #xx-#######. Note that there may be an additional leading character (not shown in this example) and xx may actually be two or three characters. If xx is ***10***, then the silicon is revision 1.0. Table 1 lists the silicon revisions associated with each lot trace code for the 66AK2Hxx devices.

Table 1. Lot Trace Codes

Lot Trace Code (xx)	Silicon Revision	Comments
10	1.0	Initial silicon revision
11	1.1	Silicon revision 1.1
20	2.0	Silicon revision 2.0
30	3.0	Silicon revision 3.0
31	3.1	Silicon revision 3.1

The 66AK2Hxx device contains multiple read-only register fields that report revision values. The JTAG ID (JTAGID) and C66x CorePac Revision ID registers allow the customer to read the current device and CPU level revision of the 66AK2Hxx.

The JTAG ID register (JTAGID) is a read-only register that identifies to the customer the JTAG/Device ID.

The C66x CorePac Revision ID register is a read-only register that identifies to the customer the revision of the C66x CorePac. The value in the VERSION field of the C66x CorePac Revision ID Register changes based on the version of the C66x CorePac implemented on the device. More details on the C66x CorePac Revision ID register can be found in the part-specific data manual.

Table 2 shows the contents of the C66x CorePac REVID Register, and the JTAGID register for each silicon revision of the 66AK2Hxx device.

Table 2. Silicon Revision Variables

Silicon Revision	C66x CorePac REVID Register (address location: 0x0181_2000)	66AK2Hxx JTAGID Register (address location: 0x0262_0018)
1.0	0x0009_0000	0x0b98_102f
1.1	0x0009_0002	0x1b98_102f
2.0	0x0009_0003	0x2b98_102f
3.0	0x0009_0003	0x3b98_102f
3.1	0x0009_0003	0xbb98_102f

More details on the JTAG ID and CorePac Revision ID Registers can be found in the device-specific data manual.

3 ARM-Specific Information

This document does not list the errata for Cortex™-A15 MPCore. For the latest information regard ARM issues, please see the following ARM web pages:

For the latest information regarding ARM issues that may not be addressed in this errata document, see the following ARM Web pages:

- http://infocenter.arm.com/help/index.jsp
- http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.set.cortexa/index.html

Table 3 provides the ARM Cortex-A15 MPCore processor version and REVIDR used by the 66AK2Hxx.

Table 3. Cortex-A15 Processor Version and REVIDR

SoC	A15 Version	ARM REVIDR
66AK2Hxx PG1.1	r2p4	0x0002 (Bit 1 = 1)
66AK2Hxx PG2.0	r2p4	0x020A (Bits 1, 3, 9 = 1)
66AK2Hxx PG3.0	r2p4	0x020A (Bits 1, 3, 9 = 1)
66AK2Hxx PG3.1	r2p4	0x020A (Bits 1, 3, 9 = 1)

The ARM product revision r_mp_n indicates the major and minor revision status of the ARM core incorporated in this device. Additionally, for a specific product revision a few additional erratum may be fixed, which can be determined by reading the ARM REVIDR register where a set bit indicates that the erratum is fixed in this ARM revision. A combination of this information can be used when referring to the *ARM Processor Cortex™-A15 MPCore – Product Errata Notice* documentation to infer the erratum applicability to this device.

The definition of the Revision ID Register can be found at the following location:

- http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0438i/CIHEJBDJ.html

4 Silicon Updates

lists the silicon updates applicable to each silicon revision. For details on each advisory, click on the link below.

Table 4. Silicon Revision 1.0, 1.1, 2.0, 3.0, and 3.1 Updates

Category	Silicon Update Advisory	See	Applies To Silicon Revision				
			1.0	1.1	2.0	3.0	3.1
ROM	Boot ROM Missed EMIF PHY Configuration Registers	Advisory 1	X				
NOR XIP	Execution in Parallel (XIP) from NOR Flash on ARM Does Not Work	Advisory 2	X	X	X	X	X
Boot	ARM L2 Latency is Not Set Correctly	Advisory 3	X				
SRIO	SRIO Boot Mode Packet DMA Cleanup Missing	Advisory 4	X				
Sync-E	Sync-Ethernet Push Event IO Control Tie-Off Incorrect	Advisory 5	X				
ROM	EMIF Boot (NOR flash) Mode Not Working When ARM Master Boot	Advisory 8	X				
ARM	1.4-Ghz ARM is Not Supported	Advisory 9	X				
DDR3	DDR3A Lanes 5 and DDR3B Lane 8 Have PHY to PUB DFI Hold Time Failures	Advisory 10	X				
SRIO	Corruption of Control Characters In SRIO Line Loopback Mode	Advisory 13	X	X	X	X	X
Hyperlink	HyperLink Temporary Blocking	Advisory 14	X	X	X	X	X
Reset	$\overline{\text{RESETSTAT}}$ Signal Driven High	Advisory 15	X	X	X	X	X
SRIO	SRIO Control Symbols Are Sent More Often Than Required	Advisory 16	X	X	X	X	X

Table 4. Silicon Revision 1.0, 1.1, 2.0, 3.0, and 3.1 Updates (continued)

Category	Silicon Update Advisory	See	Applies To Silicon Revision				
			1.0	1.1	2.0	3.0	3.1
CCS	System Reset Operation Disconnects SoC from CCS	Advisory 17	X	X	X	X	X
ARM	ARM CorePac Power Down and Hibernation Modes Not Supported	Advisory 19	X	X			
DDR3	False DDR3 Write ECC Error Reported Under Certain Conditions	Advisory 20	X	X	X	X	X
DDR3	DDR3 Leveling Issue	Advisory 21	X	X			
PCIe	PCIE MSI/Legacy IRQ Does Not Work for Root Complex	Advisory 22	X	X	X	X	X
DDR3	SES Port Should Only be Used for DDR3A Accesses	Advisory 23	X	X			
PCIe	Descriptors Placed in PCIe Memory Space can Cause Problems	Advisory 28	X	X	X	X	X
10GbE	10GbE PCS Causes Data Corruption	Advisory 29	X	X	X	X	X
10GbE	10GbE PCB Channel Loss Limited to 20 dB	Advisory 30	X	X	X	X	X
HyperLink	HyperLink Channel Loss	Advisory 31	X	X	X	X	X
HyperLink	HyperLink Data Rate Limited to 40Gbaud Issue	Advisory 33	X	X	X	X	X
DDR3	DDR3 Limited to Highest Data Rates Due to Possibility of PLL Instability During Temperature Changes After Startup	Advisory 34	X	X	X		
DDR3	Restrictions on DDR3 PLL Configuration And DDR3nCLK to Eliminate DDR3 Errors Due to PLL Dynamic Phase Offset	Advisory 35	X	X	X		
TeraNet	Two Masters Accessing Two C66x CorePac L2 Memories Can Cause TeraNet Hang	Advisory 36	X	X	X		
USB	USB3.0 PHY Does Not Meet USB Specification: Rev 3.0 RX Jitter Tolerance Mask	Advisory 37	X	X	X	X	X
PCI	PCI-Express Hot Reset Not Handled During Boot	Advisory 38	X	X	X		
CPSW	CPSW Stall if Duplex Changes During Transmission	Advisory 39	X	X	X	X	X
ROM	NAND Boot Failure When Booting Single Block Backup Images	Advisory 40	X	X	X		
ROM	NAND Boot Failure With Bit Errors in ECC	Advisory 41	X	X	X		
ROM	ARM Ethernet Boot Reinitializes the Switch With Reset Isolation Enabled, Which May Cause Lockup	Advisory 42	X	X	X		
SerDes	SerDes RX Adapts to a BOOST Value of 0 and Cannot Move From a Value of 0	Advisory 44	X	X	X	X	X
PCIe	PCIe link power states other than L0 are not supported on some Keystone II SOCs	Advisory 45	X	X	X	X	X
QMSS	Queue Diversion Failure	Advisory 46	X	X	X		
ROM	SGMII, SRIO, Hyperlink, and PCIe boot may fail on some devices	Advisory 47	X	X	X	X	
SPI	SPI Boot Size Limitation, C66x Master Boot	Usage Note 1	X				
ARM	ARM Core Hangs if Timer is Accessed While ARM Core is in Wait-In-Reset State	Usage Note 2	X	X			
Debug	Debug System (DebugSS) Trace Buffer (TBR) EDMA via System Port is not Functional	Usage Note 3	X	X			
Boot	ARM Boot Can Fail When Interrupt Enabled	Usage Note 4	X	X			
Boot	Boot I^2C Frequency Incorrect	Usage Note 5	X	X	X	X	X

Table 4. Silicon Revision 1.0, 1.1, 2.0, 3.0, and 3.1 Updates (continued)

Category	Silicon Update Advisory	See	Applies To Silicon Revision				
			1.0	1.1	2.0	3.0	3.1
DDR3	Access to DDR3 Without Configuring PHY Properly Can Cause Hang	Usage Note 6	X	X	X	X	X
Power	C66x CorePac and ARM CorePac AVS Rails	Usage Note 7	X	X	X	X	X
Power	Core Wake Up on $\overline{\text{RESET}}$	Usage Note 9	X	X	X	X	X
I^2C	I^2C Bus Hang After Master Reset	Usage Note 10	X	X	X	X	X
PLL	Minimizing Main PLL Jitter	Usage Note 11	X	X	X	X	X
QMSS	Packet DMA Does Not Update RX PS Region Location Bit	Usage Note 13	X	X			
QMSS	Queue Proxy Access	Usage Note 14	X	X	X	X	X
Power	Initial Voltage Level Setting of CVDD Rail Power Supplies	Usage Note 17	X	X	X	X	X
Debug	DEBUGSS CTTBR ID Period Functionality Issue	Usage Note 18	X	X			
Boot	Boot Mode Change by Writing to DEVSTAT Register Does Not Survive Warm Reset	Usage Note 19	X	X	X	X	X
Hyperlink	HyperLink 0 and HyperLink1 PRIVIDs Not Generated by the HyperLink IP	Usage Note 20	X	X			
USB	USB EP15 Not Supported	Usage Note 21	X	X			
Boot	C66x Boot ROM Does Not Detect a Local Reset Properly	Usage Note 22	X	X	X	X	X
Boot	BOOTCOMPLETE Not Functional when ARM CorePac is a Boot Master	Usage Note 23	X	X	X	X	X
Power	Clock Alignment Issue When in Reset Isolation Mode	Usage Note 24	X	X	X	X	X
USB	USB Hangs When Doing a Master Access to Reserved Space	Usage Note 25	X	X	X	X	X
10GbE	Device Hang if 10GbE PCS Registers are Accessed After Performing a 10G Lane Reset	Usage Note 27	X	X	X		
SerDes	SerDes Fails to Adapt RX BOOST Equalization	Usage Note 28	X	X	X	X	X

Silicon Updates

Title **Page**

KeyStonelI.BTS_errata_advisory.1 — *Boot ROM Missed EMIF PHY Configuration Registers* 12
KeyStonelI.BTS_errata_advisory.2 — *Execution in Place (XIP) from NOR Flash on NOR XIP Does Not Work* 13
KeyStonelI.BTS_errata_advisory.3 — *ARM L2 Latency is Not Set Correctly* 14
KeyStonelI.BTS_errata_advisory.4 — *SRIO Boot Mode Packet DMA Cleanup Missing* 15
KeyStonelI.BTS_errata_advisory.5 — *Sync-Ethernet Push Event IO Control Tie-Off Incorrect* 16
KeyStonelI.BTS_errata_advisory.8 — *EMIF Boot (NOR flash) Mode Not Working When ARM Master Boot* 17
KeyStonelI.BTS_errata_advisory.9 — *1.4 GHz ARM is Not Supported* 18
KeyStonelI.BTS_errata_advisory.10 — *DDR3A Lanes 5 and DDR3B Lane 8 Have PHY to PUB DFI Hold Time Failures* 19
KeyStonelI.BTS_errata_advisory.13 — *Corruption of Control Characters In SRIO Line Loopback Mode Issue* ... 20
KeyStonelI.BTS_errata_advisory.14 — *HyperLink Temporary Blocking Issue* 21
KeyStonelI.BTS_errata_advisory.15 — *$\overline{\text{RESETSTAT}}$ Signal Driven High Issue* 22
KeyStonelI.BTS_errata_advisory.16 — *SRIO Control Symbols Are Sent More Often Than Required Issue* 23
KeyStonelI.BTS_errata_advisory.17 — *System Reset Operation Disconnects SoC from CCS Issue* 24
KeyStonelI.BTS_errata_advisory.19 — *ARM CorePac Powerdown and Hibernation Modes Not Supported* 25
KeyStonelI.BTS_errata_advisory.20 — *False DDR3 Write ECC Error Reported Under Certain Conditions* 26
KeyStonelI.BTS_errata_advisory.21 — *DDR3 Leveling issue* 29
KeyStonelI.BTS_errata_advisory.22 — *PCIE MSI/Legacy IRQ Does Not work for Root Complex* 33
KeyStonelI.BTS_errata_advisory.23 — *SES Port Should Only be Used for DDR3A Accesses* 34
KeyStonelI.BTS_errata_advisory.28 — *Descriptors Placed in PCIe Memory Space can Cause Problems* 35
KeyStonelI.BTS_errata_advisory.29 — *10GbE PCS Causes Data Corruption* 36
KeyStonelI.BTS_errata_advisory.30 — *10GbE PCB Channel Loss Limited to 20dB* 37
KeyStonelI.BTS_errata_advisory.31 — *HyperLink Channel Loss* 39
KeyStonelI.BTS_errata_advisory.33 — *HyperLink Data Rate Limited to 40 Gbaud Issue* 41
KeyStonelI.BTS_errata_advisory.34 — *DDR3 Limited to Highest Data Rates Due to Possibility of PLL Instability During Temperature Changes After Startup* 42
KeyStonelI.BTS_errata_advisory.35 — *Restrictions on DDR3 PLL Configuration and DDR3nCLK to Eliminate DDR3 Errors Due to PLL Dynamic Phase Offset* 43
KeyStonelI.BTS_errata_advisory.36 — *Two Masters Accessing Two C66x CorePac L2 Memories Can Cause TeraNet Hang* 46
KeyStonelI.BTS_errata_advisory.37 — *USB3.0 PHY Does Not Meet USB Specification: Rev 3.0 RX Jitter Tolerance Mask Across Non- Nominal Process* 51
KeyStonelI.BTS_errata_advisory.38 — *PCI-Express Hot Reset Not Handled During Boot* 52
KeyStonelI.BTS_errata_advisory.39 — *CPSW Stall if Duplex Changes During Transmission* 53
KeyStonelI.BTS_errata_advisory.40 — *ROM: NAND Boot Failure When Booting Single Block Backup Images* ... 55
KeyStonelI.BTS_errata_advisory.41 — *NAND Boot Failure With Bit Errors in ECC* 56
KeyStonelI.BTS_errata_advisory.42 — *ARM Ethernet Boot Reinitializes the Switch With Reset Isolation Enabled, Which May Cause Lockup* 57
KeyStonelI.BTS_errata_advisory.44 — *SerDes RX Adapts to a BOOST Value of 0 and Cannot Move From a Value of 0* 58
KeyStonelI.BTS_errata_advisory.45 — *PCIe link power states other than L0 are not supported on some Keystone II SOCs* 59
KeyStonelI.BTS_errata_advisory.46 — *Queue Diversion Failure* 60
KeyStonelI.BTS_errata_advisory.47 — *SGMII, SRIO, Hyperlink, and PCIe Boot Failure* 61
KeyStonelI.BTS_errata_usagenote.1 — *SPI Boot Size Limitation, C66x Master Boot* 62
KeyStonelI.BTS_errata_usagenote.2 — *ARM Core Hangs if Timer is Accessed While ARM Core is in Wait-In-Reset State* 63
KeyStonelI.BTS_errata_usagenote.3 — *Debug System (DebugSS) Trace Buffer EDMA via System Port Not Functional* 64
KeyStonelI.BTS_errata_usagenote.4 — *ARM Boot Can Fail When Interrupt Enabled* 65

Silicon Updates (continued)

KeyStonelI.BTS_errata_usagenote.5 — ***Boot I²C Frequency Incorrect*** 66
KeyStonelI.BTS_errata_usagenote.6 — ***Access to DDR3 Without Configuring PHY Properly Can Cause Hang*** .. 67
KeyStonelI.BTS_errata_usagenote.7 — ***C66x CorePac and ARM CorePac AVS Rails*** 68
KeyStonelI.BTS_errata_usagenote.9 — ***Core Wake Up on RESET Usage Note*** 69
KeyStonelI.BTS_errata_usagenote.10 — ***I²C Bus Hang After Master Reset Usage Note*** 70
KeyStonelI.BTS_errata_usagenote.11 — ***Minimizing Main PLL Jitter Usage Note*** 71
KeyStonelI.BTS_errata_usagenote.13 — ***Packet DMA Does Not Update RX PS Region Location Bit Usage Note*** 72
KeyStonelI.BTS_errata_usagenote.14 — ***Queue Proxy Access Usage Note*** 73
KeyStonelI.BTS_errata_usagenote.17 — ***Initial Voltage Level Setting of CVDD Rail Power Supplies Usage Note*** 74
KeyStonelI.BTS_errata_usagenote.18 — ***DEBUGSS CTTBR ID Period Functionality Issue***..... 75
KeyStonelI.BTS_errata_usagenote.19 — ***Boot Mode Change by Writing to DEVSTAT Register Does Not Survive Warm Reset*** 76
KeyStonelI.BTS_errata_usagenote.20 — ***HyperLink 0 and HyperLink1 PRIVIDs Not Generated by the HyperLink IP*** 77
KeyStonelI.BTS_errata_usagenote.21 — ***USB EP15 Not Supported*** 78
KeyStonelI.BTS_errata_usagenote.22 — ***C66x Boot ROM Does Not Detect a Local Reset Properly***..... 79
KeyStonelI.BTS_errata_usagenote.23 — ***BOOTCOMPLETE Not Functional when ARM CorePac is a Boot Master***..... 80
KeyStonelI.BTS_errata_usagenote.24 — ***Clock Alignment Issue When in Reset Isolation Mode***..... 81
KeyStonelI.BTS_errata_usagenote.25 — ***USB Hangs When Doing a Master Access to Reserved Space*** 82
KeyStonelI.BTS_errata_usagenote.27 — ***Device Hang if 10GbE PCS Registers are Accessed After Performing a 10G Lane Reset*** 83
KeyStonelI.BTS_errata_usagenote.28 — ***SerDes Fails to Adapt RX BOOST Equalization*** 84

KeyStoneII.BTS_errata_advisory.1

Boot ROM Missed EMIF PHY Configuration Registers

Revision(s) Affected 1.0

Details Boot ROM missed initializing some EMIF PHY Configuration registers.

On PG1.0 Boot ROM, the built-in DDR3 PHY configuration for the DWCPUB PHY does not include values needed for PHY Config. The following values are needed to support configuring the PHY registers:

- DDR3A_PLLCR offset 0x00000018
- DDR3A_DSGCR offset 0x00000040
- DDR3A_ZQ0CR1 offset 0x00000184
- DDR3A_ZQ1CR1 offset 0x00000194
- DDR3A_ZQ2CR1 offset 0x000001A4
- DDR3A_ZQ3CR1 offset 0x000001B4

Workaround If the non-default values are required in these registers, a two stage boot must be used.

KeyStoneII.BTS_errata_advisory.2

Execution in Place (XIP) from NOR Flash on NOR XIP Does Not Work

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details ARM cannot perform direct execution from a parallel NOR flash connected via ASYNC EMIF.

On these devices, ARM cannot perform direct execution from a parallel NOR flash connected via ASYNC EMIF. Direct execution from a parallel NOR flash does not work as ARM always generates 64-byte cacheline wrap mode accesses to EMIF. This is irrespective of marking this memory region as Device or Strongly Ordered as confirmed by ARM. ASYNC EMIF does not support 64-byte cacheline wrap accesses and therefore generates a bus error on receiving it. This causes an abort to happen in ARM. The ARM is, however, able to read data from a parallel NOR flash connected via ASYNC EMIF.

Workaround None. If code is stored on NOR flash, it must be copied from the NOR to another memory area and executed from there.

KeyStoneII.BTS_errata_advisory.3

ARM L2 Latency is Not Set Correctly

Revision(s) Affected 1.0

Details ARM L2 Latency is not set correctly in the Boot ROM

In PG1.0 devices, the boot ROM did not change the default value of L2 latency in the L2 Control register. The default value is 0 (which means two cycles). The required value is 3 (which means 4 cycles).

Workaround Either use one of the C66x boot modes or the I^2C or SPI boot mode in ARM master boot mode which does not setup the ARM PLL. Once the boot is done, the PLL can be changed and the latency setup can be programmed accordingly.

KeyStoneII.BTS_errata_advisory.4

SRIO Boot Mode Packet DMA Cleanup Missing

Revision(s) Affected 1.0

Details The issue is within the Boot ROM. In SRIO Boot Mode, the Packet DMA is not torn down when the SRIO boot is complete.

On PG1.0 devices, in SRIO boot mode, the boot ROM did not perform the Packet DMA cleanup (the Packet DMA is not torn down) when the boot is complete. All queues are emptied and the QM is torn down, but the Packet DMA is left up.

Workaround Before reconfiguration, the existing configuration should be torn down.

KeyStoneII.BTS_errata_advisory.5

Sync-Ethernet Push Event IO Control Tie-Off Incorrect

Revision(s) Affected 1.0

Details The tie-off for IO control for the two Sync-Ethernet Push Events is connected incorrectly in the design.

On PG1.0 devices, the tie-off for IO control for the two Sync-Ethernet Push Event (TSPUSHEVT0 (PPS Push Event from GPS for IEEE1588) and TSPUSHEVT1 (Push Event from BCN for IEEE1588)) IOs is connected incorrectly in the design. Consequence of this is that the TSPUSHEVT0 and TSPUSHEVT1 pads are being forced to outputs when not in reset, and thus **cannot** be selected as inputs. These pads are functionally used as inputs.

Workaround None.

KeyStoneII.BTS_errata_advisory.8

EMIF Boot (NOR flash) Mode Not Working When ARM Master Boot

Revision(s) Affected 1.0

Details EMIF (NOR flash) boot does not work when ARM is the boot master.

Because ARM cannot perform direct execution from a parallel NOR flash connected via ASYNC EMIF the boot mode EMIF (NOR flash) on this version of the device does not work when ARM is the boot master. The boot failed because the ROM boot loader simply branched to an address in the NOR to execute code stored on the NOR. This was how boot mode works in later versions of the device because the ROM boot loader instead copies to the code image from the NOR to internal memory and then executes the code copy.

Workaround Use C66x boot master if NOR flash boot is required.

KeyStoneII.BTS_errata_advisory.9

1.4 GHz ARM is Not Supported

Revision(s) Affected 1.0

Details 1.4 GHz ARM is not supported silicon revision listed in the table.

The ARM cannot fully function at 1.4 GHz with core power supplies of 900 mV nominal voltage.

Workaround Run the ARM at a 1.2 GHz.

KeyStoneII.BTS_errata_advisory.10

DDR3A Lanes 5 and DDR3B Lane 8 Have PHY to PUB DFI Hold Time Failures

Revision(s) Affected 1.0

Details

DDR3A Bit Errors in Data Byte Lane 5

It has been observed that operating DDR3A in 64-bit mode results in occasional read errors. The problem is observed across all frequencies of operation.

The read errors occur across byte lane 5 of the DDR3A interface due to hold time violations on the DFI interface between the PHY Utility Block and SDRAM PHY on this specific byte lane. Since the hold violations are only on byte lane 5, this issue impacts only 64-bit DDR3A operation and not 32-bit or 16-bit DDR3A operations.

Table 5. DDR3A

72 bit with ECC	Nonfunctional
64 bit without ECC	Nonfunctional
36 bit with ECC	Functional
32 bit without ECC	Functional
16 bit without ECC	Functional

Workaround The only stable work around available at this time is to operate the DDR3A interface in 32-bit or 16-bit mode. These configurations would ensure the faulty byte lane is never activated.

Details

DDR3B Bit Errors in ECC Byte Lane

It has been observed that error correction is currently nonfunctional on the DDR3B interface. Turning on ECC during DDR3B operation results in occasional read errors as a result of miscalculated ECC values.

Table 6. DDR3B

72 bit with ECC	Nonfunctional
64 bit without ECC	Functional
36 bit with ECC	Nonfunctional
32 bit without ECC	Functional
16 bit without ECC	Functional

Workaround The only stable work around available at this time is to operate DDR3B interface in non ECC mode. This configuration would ensure the faulty byte lane is never activated.

KeyStoneII.BTS_errata_advisory.13

Corruption of Control Characters In SRIO Line Loopback Mode Issue

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details The SRIO physical layer is configured in line-loopback mode on a per-port basis, by setting the LLB_EN bit in PLM_SP(n)_IMP_SPEC_CTL register. In line-loopback mode, the data from the SerDes receiver is looped back to the SerDes transmitter. Figure 2 shows the line loopback from SerDes RX to SerDes TX.

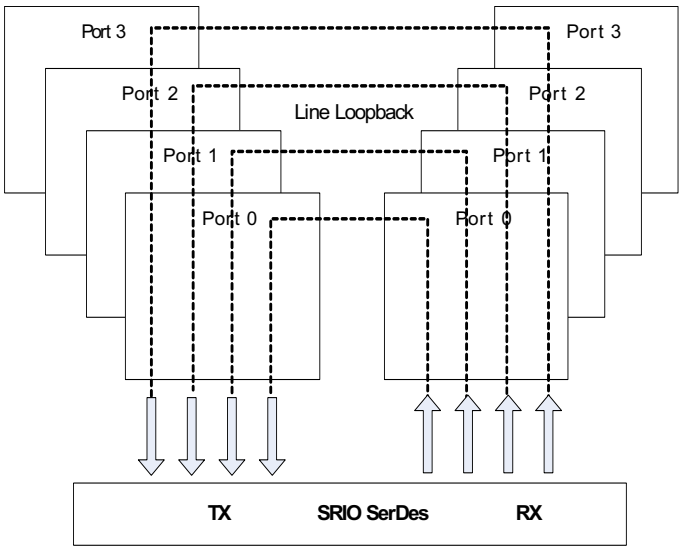


Figure 2. SRIO SerDes in Loopback Mode

The port does not provide any clock compensation when line loopback is enabled. The transmit clock must be externally synchronized with the receive clock. There is only a small FIFO that can compensate for PLL jitter or wander. Hence, correct operation of line loopback requires that the link partners use the same reference clock for the SRIO physical layer, in order to avoid overruns or underruns due to clock frequency mismatch between the link partners. As a result, line loopback mode is generally restricted to validation and qualification of board signal integrity in a lab environment.

When line loopback is enabled on one or more SRIO ports, any valid 10b code group that decodes to an illegal control character as defined by the RapidIO Specification (Revision 2.1) and whose most significant bit is 0, will be corrupted on transmission. This issue can be summarized as follows:

- 10b code group -> Legal control character -> No problem
- 10b code group -> Illegal control character and most significant bit is 0 -> Corruption

Workaround Instead of using PRBS sequences, users can qualify boards by using RapidIO-compliant data on the link and monitoring either per-lane error counters or port-level error counters. RapidIO-compliant data is less stressful than PRBS sequences, as the RapidIO-complaint 10b data has shorter 0s and 1s run lengths than PRBS sequences. Hence, RapidIO-compliant data represent a more accurate stimulus for this test. This should be acceptable for users whose RapidIO links are of short reach, which can be either 20 cm + 1 connector or 30 cm without a connector.

KeyStoneII.BTS_errata_advisory.14

HyperLink Temporary Blocking Issue

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details A HyperLink temporary blocking condition can exist when a local device is reading from a remote device so heavily that the responses from the remote device keep the response path continuously busy. Because responses have a higher priority than outgoing commands, the remote device is temporarily unable to send commands to the local device.

Workaround 1: Use a push messaging model instead of pull.

Workaround 2: If a pull model is needed, schedule breaks so the return path is not continuously busy.

KeyStoneII.BTS_errata_advisory.15

$\overline{\text{RESETSTAT}}$ Signal Driven High Issue

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details The $\overline{\text{RESETSTAT}}$ output signal should be driven low when a reset is applied and held low until the reset cycle is complete. If the device is using power sequencing where the 1.8 V (DVDD18) is present before the AVS core voltage (CVDD), the $\overline{\text{RESETSTAT}}$ signal may be driven high erroneously during the time between when DVDD18 is present and the CVDD is present.

Workaround One workaround is to use the CVDD before DVDD18 in the power sequencing. An alternative workaround is to ignore the $\overline{\text{RESETSTAT}}$ signal if the CVDD is not present during the power sequencing.

KeyStoneII.BTS_errata_advisory.16

SRIO Control Symbols Are Sent More Often Than Required Issue

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details Control symbols are SRIO physical layer message elements used to manage link maintenance, packet delimiting, packet acknowledgment, error reporting, and error recovery. Control symbols are used to manage the flow of transactions in the SRIO physical interconnect. The SRIO input status control symbols communicate the status of the physical link and packets in flight between the two SRIO link partners.

The bandwidth of the SRIO link is reduced because status control symbols are sent more often than required. Worst case impact is a 2.73 percent reduction in bandwidth for a 1× port operating at 1.25 Gbaud. This impact is reduced to 0.1 percent for a 4× port operating at 5 Gbaud. More details about this impact on various lane speed and port configurations can be found in Table 7.

Table 7. Impact on Various Lane Speeds

Lane Speed (Gbaud)	Percentage of Bandwidth Reduction		
	1x Port	2x Port	4x Port
1.25	2.73	1.37	0.68
2.5	1.17	0.59	0.29
3.125	0.86	0.43	0.21
5.0	0.39	0.20	0.10

Workaround None

KeyStoneII.BTS_errata_advisory.17

System Reset Operation Disconnects SoC from CCS Issue

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details CCS connection to targets will fail after system reset issued via CCS. CCS connection to targets will also fail after $\overline{\text{RESET}}$ reset of the device. A system reset, issued from CCS or by the $\overline{\text{RESET}}$ pin, can cause power reset to all C66x Corepacs and can cause the hardware states of debug logic (including hardware breakpoints) to get cleared. The result is that any existing CCS connection to those targets will get corrupted, terminating further access to the target.

Workaround 1: A new configuration option called Domain Power Loss Mode is added in the CCS target configuration for enabling the debug software to detect and handle the power loss event automatically.

To enable this option, in the CCS target configuration window, click on the sub-path of ICEPICK_D for each individual C66x Corepac. Then click on the property option **Domain Power Loss Mode** and select **Auto**.

The support for this new option will be released in the emupack update v5.0.586.0 or newer, patched to CCS5.1 GA.

Workaround 2: Before issuing a system reset, disconnect CCS from all DSP targets, issue the system reset, then reconnect CCS to the targets to continue debug operations.

KeyStoneII.BTS_errata_advisory.19

ARM CorePac Powerdown and Hibernation Modes Not Supported

Revision(s) Affected	1.0, 1.1
Details	ARM CorePac powerdown mode (SoC PSC initiated clock off or power off request) is not supported.
Workaround	None

KeyStoneII.BTS_errata_advisory.20

False DDR3 Write ECC Error Reported Under Certain Conditions

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Problem Summary: In the event that the read-modify-write (RMW) ECC feature for DDR3 is not supported or disabled, an L1D or L2 block writeback or writeback invalidate operation to ECC protected DDR3 space will flag a DDR3 write ECC error, even though neither the data nor the ECC values stored in the SDRAM will be corrupted.

Details The write ECC error interrupt can be enabled by setting the WR_ECC_ERR_SYS bit in the Interrupt Enable Set Register (IRQSTATUS_ SET_SYS) of the DDR3 controller.

Under normal conditions, a write access performed within the ECC protected address range to a 64-bit aligned address with a byte count that is 64-bit quanta is not expected to flag a write ECC error interrupt. The C66x cache controller always operates on whole cache lines, which are 128 bytes for the L2 cache and 64 bytes for L1D cache. However, a block writeback or writeback invalidate always generates a bounding single byte write (with its byte enables disabled) to the last address in that block. This single byte write violates both the alignment and quanta conditions causing the DDR3 controller to flag a write ECC error in the Interrupt Raw Status Register (IRQSTATUS_RAW_SYS) register. Since this bounding write is sent with its byte enables disabled, it does not actually reach the DDR memory and does not corrupt the stored data or ECC values. The write ECC error is thus a spurious error since no data or ECC value is actually corrupted. It should be noted that the DDR3 controller, in response to this sub-quanta write (the bounding single byte write), will report an error on an internal status line to the CPU that executed the writeback. This error status flags the MDMA error interrupt to the CPU and is interpreted as an MDMA data error (the STAT field in the CPU's MDMA Bus Error Register will be set to 0x4).

NOTE: 1: Since no bounding writes are generated with global writeback or global writeback invalidate operations, this issue is limited only to block writebacks to ECC protected region.

NOTE: 2: If the MDMA error flagged by a block coherence operation is followed by a true MDMA error flagged by a master executing a direct sub-quanta write, only the first MDMA error will be captured. Software must clear an MDMA error as soon as possible in order for future errors to be captured.

Workaround 1 The problem can be resolved by enabling the read-modify-write ECC feature for DDR3 (set RMW_EN=1 in the ECCCTL register of the DDR3 memory controller). The RMW feature is specifically designed to handle sub-quanta/non-aligned accesses to ECC protected space. With RMW enabled the bounding single byte write will not violate any quanta/alignment requirements and thus will not flag a write ECC error.

Silicon rev1.0 and 1.1 of this device do not support RMW. Silicon rev2.0 supports RMW.

Workarounds 2 and 3 should be explored if RMW is not available or not used.

Workaround 2 In order to differentiate a false write ECC error from a true error generated by alignment/quanta violations, the system should keep track of block writeback/writeback invalidate operations to the ECC protected memory space. If a write ECC error is confirmed for that operation, it can be safely ignored. There is only a single DDR3 error interrupt that will have to be processed by one of the C66x cores. Therefore, some special mechanism will be required for the system to keep track of which core performed the block writeback that caused the error. This mechanism may involve checking for the data error reported in the MDMA Bus Error Register.

NOTE: 3: A system must satisfy the alignment/quanta conditions so a true write ECC error is not expected and such errors should be isolated and removed as part of system software evaluation.

NOTE: 4: The system software must clear the write ECC error and MDMA error before they can be re-triggered by any successive error conditions. It should be noted that a race condition can exist if a subsequent ECC error (real or false) or MDMA error interrupt is triggered before the previous interrupt is cleared.

Workaround 3 A global coherence operation can be performed instead of a block operation. It should be noted that a global operation can possibly operate on more cache lines than the block operation, causing a larger than necessary cycle overhead and negatively impact memory system performance.

FAQ: Q: The C66x CorePac will receive an MDMA error in response to the DDR3 ECC error. Other masters may also see the DDR3 ECC error when transactions they have sent result in the error. How do these masters respond to a DDR3 ECC error?

A: The responses of the C66x CorePac, ARM CorePac, and other masters to the non-zero values returned on rstatus and sstatus due to DDR3 ECC errors are summarized in Table 8.

Table 8. Master Behavior in Response to sstatus/rstatus Flagged Due to DDR3 ECC Errors

Master	Bus Error Returned	Error Status Captured	Alarm Notification
C66x CorePac	sstatus, rstatus	Error status captured in the STAT field in the C66x CorePac's MDMA Bus Error Register will be set to 0x4.	The C66x CorePac's MDMA error interrupt is asserted.
ARM CorePac	sstatus, rstatus	Details on the exception are captured in the CP15 registers: Data Fault Status Register (DFSR), Instruction Fault Status Register (IFSR) or Auxiliary Data Fault Status Register (ADFSR). The address that generated the abort can be seen by reading Data Fault Address Register (DFAR) for synchronous aborts.	The ARM CorePac will trigger an external abort exception. They are disabled by default and should be enabled via the CPSR.A bit (Current Program Status Register).
PCIe	sstatus, rstatus	Interrupts are not generated and error status is not logged.	PCIe returns completion abort to requestor only for rstatus error. No completion abort is returned for a write error (sstatus).
EDMA TC	sstatus, rstatus	BUSERR and ERRDET registers capture the error information. The transfer of data from source to destination happens irrespective of error.	Error interrupt is generated if enabled within EDMA.
Multicore Navigator Infrastructure PktDMA	sstatus, rstatus	Error status is not logged.	Error interrupt is not reported
TSIP	sstatus, rstatus	Errors will be stored in the channels' interrupt queue along with the error codes.	TSIP asserts an error event (TSIPx_ERRINTn) when an error is queued for channel 'n'. CorePac[n] will receive TSIPx_ERRINTn.
SRIO	sstatus, rstatus	Error responses set a bit in the AMU_INT_ICSR register based on the CPRIVID of the transactions. The RIO_AMU_ERR_CAPT0, RIO_AMU_ERR_CAPT1 registers will contain the address of the non-posted transactions that failed along with the CPRIVID and CMSTID.	Each bit can be routed by software configuration through the Interrupt Condition Routing Register (ICRR) to a specific ARM or C66x CorePac for error handling. See the device data manual for interrupt mapping.

Table 8. Master Behavior in Response to sstatus/rstatus Flagged Due to DDR3 ECC Errors (continued)

Master	Bus Error Returned	Error Status Captured	Alarm Notification
HyperLink	sstatus, rstatus	When the serial link is active HyperLink will pass the rstatus. Rstatus is will be that of the remote slave read.	HyperLink Error interrupt (HyperLink_INT or VUSR_INT) are generated when error is received and are provided to CorePacs as secondary interrupts.
10GbE	sstatus, rstatus are not used	NA	NA

NOTE: Check your device data manual to see which masters are applicable.

KeyStoneII.BTS_errata_advisory.21

DDR3 Leveling issue

Revision(s) Affected 1.0, 1.1

Details

DDR Read/Write errors are reported after DDR initialization for <1% of power-on resets. When executing UBOOT from the MCSDK, this results in DDR POST (Power On Self Test) failure.

Sometimes DDR initialization fails with leveling errors (reported in DDR PHY status registers). Once the leveling errors are reported, DDR read/write operations fail. This issue happens only during initialization time and cannot occur during runtime. This issue occurs irrespective of the operating frequency of the device and is also independent of the operating frequency of the DDR. This issue is applicable for both DDR3A and DDR3B. ROM Boot modes supporting initialization of, and booting to DDR can fail due to this issue. Below is the list of boot modes for which ROM Bootloader performs DDR configuration (only if boot image is required to be stored in DDR):

- I_2C Master
- SPI
- EMIF NOR
- EMIF NAND

The root cause is identified to an issue that happens after the very first DDR3 PHY initialization, this first initialization happens directly from Power-on reset. The issue occurs because the clocking requirements of the DDR3 PHY are not met on the first initialization, this happens because the DDR3A and DDR3B SoC PLLs are not yet programmed to provide the requisite clock. There is a ratio requirement between the DDR3A/B PHY clocks and the core configuration clock, such that the core configuration clock must be less than or equal to the PHY clock. These clocks are asynchronous and the PHY support an asynchronous FIFO scheme with limited depth on the PHY clock domain. If the ratio requirement is not met then FIFO can overflow. This would mean that the configuration data for the PHY may be lost during the initialization sequence. Experiments have proven that if we apply Hard Reset once the clocking has been set up correctly, the issue is resolved.

Workarounds

DDR read/write failures can be avoided by checking the DDR PHY status register for any Leveling errors at the end of DDR PHY Initialization. For boot modes using DDR, the workarounds below can be used as a part of second-stage boot only. Once the leveling errors are detected, one of the following workarounds can be used.

Workaround 1

This workaround is to detect this error in the PHY Init function and apply Hard Reset to the device. Hard Reset of the device can be done by performing the following:

- *Software Reset using RSTCTRL register in PLL Controller*

Below is the logical explanation of the flow where the above Reset needs to be applied.

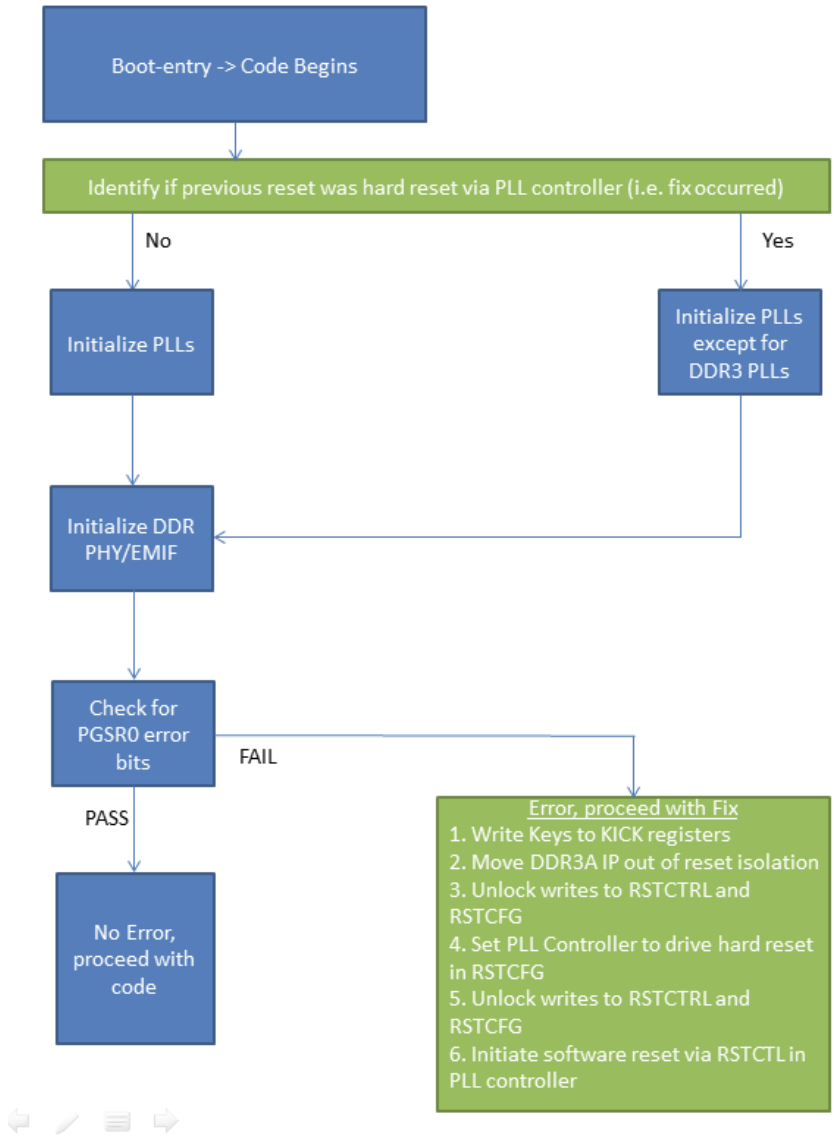


Figure 3. Software Reset Using the RSTCTRL Register

BLUE colored blocks show the normal execution while the GREEN colored blocks show the changes. The “Error, proceed with Fix” block is the block which explains the steps that need to be followed to apply the Software Reset using RSTCTRL register. The steps are mentioned for DDR3A. The same steps need to be used for DDR3B also.

Below is the source code for reference:

```
// Check PGSR0 Error bits of DDR3 PHY
    tmp = __read(DDR3A_DDRPHY_BASE + DDRPHY_PGSR0_OFFSET);
    if (((tmp & 0x00000E00) != 0x00000E00) || ((tmp & 0x0FE00000) != 0)) {

        // Write Keys to KICK registers to enable writes to registers in boot config space
        __write(KEYSTONE_KICK0_MAGIC, KEYSTONE_KICK0);
        __write(KEYSTONE_KICK1_MAGIC, KEYSTONE_KICK1);

        // Move DDR3A Peripheral out of reset isolation by setting MDCTL23[12] = 0
        tmp = __read(PSC_BASE + MDCTRL23_OFFSET);
        tmp &= ~(0x1000);
        __write(tmp, PSC_BASE + MDCTRL23_OFFSET);

        // Write 0x5A69 Key to RSTCTRL[15:0] to unlock writes to RSTCTRL and RSTCFG
        tmp = __read(PLL_CNTRL_BASE + MAIN_PLL_CTRL_RSTCTRL_OFFSET);
        tmp &= ~(0xFFFF);
        tmp |= 0x5A69;
        __write(tmp, PLL_CNTRL_BASE + MAIN_PLL_CTRL_RSTCTRL_OFFSET);

        // Set PLL Controller to drive hard reset on SW trigger by setting RSTCFG[13] = 0
        tmp = __read(PLL_CNTRL_BASE + MAIN_PLL_CTRL_RSTCFG_OFFSET);
        tmp &= ~(0x2000);
        tmp |= 0x0000;
        __write(tmp, PLL_CNTRL_BASE + MAIN_PLL_CTRL_RSTCFG_OFFSET);

        // Write 0x5A69 Key to RSTCTRL[15:0] to unlock writes to RSTCTRL and RSTCFG
        tmp = __ read(PLL_CNTRL_BASE + MAIN_PLL_CTRL_RSTCTRL_OFFSET);
        tmp &= ~(0xFFFF);
        tmp |= 0x5A69;
        __ write(tmp, PLL_CNTRL_BASE + MAIN_PLL_CTRL_RSTCTRL_OFFSET);

        // Write RSTCTRL[16] = 0 to initiate software reset via PLL controller
        tmp = __ read(PLL_CNTRL_BASE + MAIN_PLL_CTRL_RSTCTRL_OFFSET);
        tmp &= ~(0x10000);
        tmp = 0x00000;
        __ write(tmp, PLL_CNTRL_BASE + MAIN_PLL_CTRL_RSTCTRL_OFFSET);

        >> PLL Controller should now initiate Hard Reset
```

Figure 4. Reset Source Code

Another GREEN block shown in the above block diagram is normal sequence where U-Boot identifies whether the previous reset is Hard Reset or not and skips the PLL configuration as Hard Reset does not reset the PLLs. Execution can read the RSTYPE register bit field PLLCTRLRST to identify whether the previous reset is the Software Reset initiated by RSTCTRL register in PLL controller or not.

In ARM-side implementation, detection of the leveling error and applying Reset needs to be done in U-Boot. Board.c file can be modified to apply this workaround. Below are the functions that need to be modified:

- board_early_init (function) – Identify previous reset type and determine if PLLs should be initialized
- dram_init (function) - Detection of PGSR0 error and applying Software Reset steps as shown above

CCS Reset using Emulator

During development environment using CCS if the leveling error comes up on the DDR configuration then CCS Emulation Reset “System Reset” can be used to apply Hard Reset to SoC.

Once the above reset is applied and booting is performed again, the boot code needs to identify whether it is the Hard Reset OR it is ‘Power-on reset’ (i.e., $\overline{\text{POR}}$ or $\overline{\text{RESETFULL}}$). Once it is identified as a Hard Reset, the boot code needs to proceed with usual sequence without a DDR PLL initialization, and perform the DDR PHY configuration. During this second DDR PHY initialization after the Hard Reset the Leveling errors will not occur.

Workaround 2

If Workaround#1 has some other implications at the board level OR system level then this workaround can be used. This workaround is to detect this error in the PHY Init function and apply $\overline{\text{RESETFULL}}$ or $\overline{\text{POR}}$ to the device. Due to $\overline{\text{RESETFULL}}$ or $\overline{\text{POR}}$ this workaround has no surety of the reoccurrence of the same issue again, but it goes through the same probability of the occurrence (less than 1%) of the issue. Therefore, the likelihood of this issue re-occurring is less. At this point testing has not shown this issue occurring on two consecutive $\overline{\text{RESETFULL}}$ or $\overline{\text{POR}}$.

This method can also be used when utilizing the DDR initialization option within the ROM boot loader. If the boot is determined to fail by the external host, then a new $\overline{\text{RESETFULL}}$ or $\overline{\text{POR}}$ should be applied to the device and the boot sequence re-started.

KeyStoneII.BTS_errata_advisory.22

PCIE MSI/Legacy IRQ Does Not work for Root Complex

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details While testing the AER PCIE error handling and error recovery Linux driver software on KeyStone II EVM, it was found that only error interrupt #12 is raised. An unsupported request error is simulated by generating transaction to an EP with an address not in the BAR. The error is detected by Root complex.

As per PCIE spec 2.0, section 6.2.6, MSI/Legacy error interrupt to be raised as well platform specific interrupt. Currently, only platform specific interrupt (INT #12) is raised. AER driver depends on Legacy/MSI IRQ and can't function without this. This is the standard PCI driver in Linux to handle Error and do recovery.

Workaround None

KeyStoneII.BTS_errata_advisory.23

SES Port Should Only be Used for DDR3A Accesses

Revision(s) Affected 1.0, 1.1

Details

This errata is to notify about usage of SES only for DDR3A accesses.

MSMC provides two system access ports, SMS and SES. The SMS port provides direct access to MSMC SRAM. The SMS port resides at system addresses 0x0C000000 - 0x0CFFFFFF. The SES port provides access to both DDR3A and MSMC SRAM. The SES port resides at system address 0x80000000 - 0xFFFFFFFF. By default, MSMC directs access via SES to the DDR3A EMIF. Thus, after reset, system addresses 0x80000000 - 0xFFFFFFFF correspond to DDR3A. Using SES MPAX registers, programs can redirect some or all accesses in this range to MSMC SRAM.

The SES processes requests to MSMC SRAM and DDR3A internally on separate paths. However, responses for read and write commands to SES must merge into a common path before returning to the system. In particular, write commands must return a write status (sstatus) to indicate that the write completed, and MSMC must merge the write status streams from MSMC SRAM requests and DDR3A requests into a single stream returning to the system.

The SES prioritizes responses from MSMC RAM over responses from DDR3A. When SES is heavily loaded with a mixture of requests to MSMC SRAM and DDR3A, responses returned from MSMC SRAM will delay responses from DDR3A. An extended series of delays will eventually fill DDR3A's response pipeline.

When the DDR3A response pipeline fills, it incorrectly drops write responses (sstatus) for write requests from master ID 0x52 (sid = 0x52). MSMC uses this master ID internally for cache coherence operations. BCP_DIO1 also uses this master ID.

The dropped write responses lead to deadlock in one of two ways:

- Dropped write status for coherence writes causes the coherence machinery to stall indefinitely waiting for acknowledgement of its write.
- Dropped write status for BCP_DIO1 writes causes BCP to stall waiting for its outstanding writes to complete.

Workaround

An example scenario that could lead to a configuration triggering this errata is to use on-chip memory instead of off-chip to improve performance. In this scenario, both XMC and SES MPAX have aliases to MSMC SRAM in order to make the addressing transparent to application.

User must avoid aliased addresses on SES port by system masters

- Remove SES MPAX aliasing to MSMC.
- Consequently, any SW configuring any system master (eg., EDMAs) must translate aliased DDR3A addresses to global physical MSMC addresses (0x0c000000) such that they go through the SMS MPAX table instead of SES MPAX.

For example, application has 0x80000000 => 0x0c000000 in both SES and XMC. The SES alias must be removed. Thus, the software must translate 0x80000000 to 0x0c000000 then directly provide the 0x0c000000 address to the EDMAs. Due to the internal workings of the SoC, this causes the SoC to present the addresses to the SMS instead of SES which avoids the errata.

KeyStoneII.BTS_errata_advisory.28

Descriptors Placed in PCIe Memory Space can Cause Problems

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Problem Summary: Packet DMA can generate write transactions with partial byte enables when trying to access descriptors. This can cause problems if the descriptors are stored in PCIe memory space since PCIe cannot handle partial byte enables. Here, *partial byte enables* means that the transactions are accessing memory that is not a full 32-bit word.

Details Packet DMA can sometimes generate write transactions with partial byte enables asserted when trying to access descriptors. This can cause problems if the descriptors are placed in PCIe memory space since PCIe does not support transaction requests with partial byte enables. Such a transaction will corrupt the PCIe module's internal state machine and in turn corrupt the payload. This issue does not impact Packet DMA access to data buffers, only to descriptors since all byte enables are asserted in transactions involving data buffers. Thus, the data buffers may be stored anywhere including in PCIe memory space.

Workaround As long as host-mode descriptors are used and these descriptors are located in a memory space that can properly handle partial byte enables (such as L2 SRAM, DDR3 or MSMC), the issue will not affect Packet DMA accesses to PCIe memory space. As mentioned earlier, data buffers can be stored in PCIe memory space without any problems.

KeyStoneII.BTS_errata_advisory.29

10GbE PCS Causes Data Corruption

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Problem Summary: The 10GbE Physical Coding Sublayer (PCS) used in the 10GbE interface may corrupt its output data upon initialization.

Details Upon initial synchronization to an incoming data stream, the 10G PCS may not configure itself correctly. Due to this incorrect configuration, the incoming data from the receive port will be corrupted before it is sent to subsequent blocks in the data path. As this issue occurs during initialization, if for a synchronization event the PCS is initialized correctly then data corruption does not occur. If the issue occurs, the data corruption can be seen when the PCS reports PCS block errors or in packet loss at the user space software.

Workaround PG1.X-PG2.0-PG3.0:

The user may detect the PCS-R corrupt state when packet loss is observed. Due to the conditions explained in Usage Note #27, a read of the hiber or block error fields in the PCS-R RX Status register may cause a VBUS lock. Thus these fields may not be used to detect the corrupt state. When the corrupt state is detected, the user can assert and then deassert the corresponding Serdes signal detect. This signal detect reset forces the RX to re-adapt to the incoming data, and causes an interruption to the datastream to the PCS. This causes the PCS to resynchronize. This workaround is currently implemented in the latest 10GbE Linux drivers (since MCSDK 3.0.4).

PG3.0:

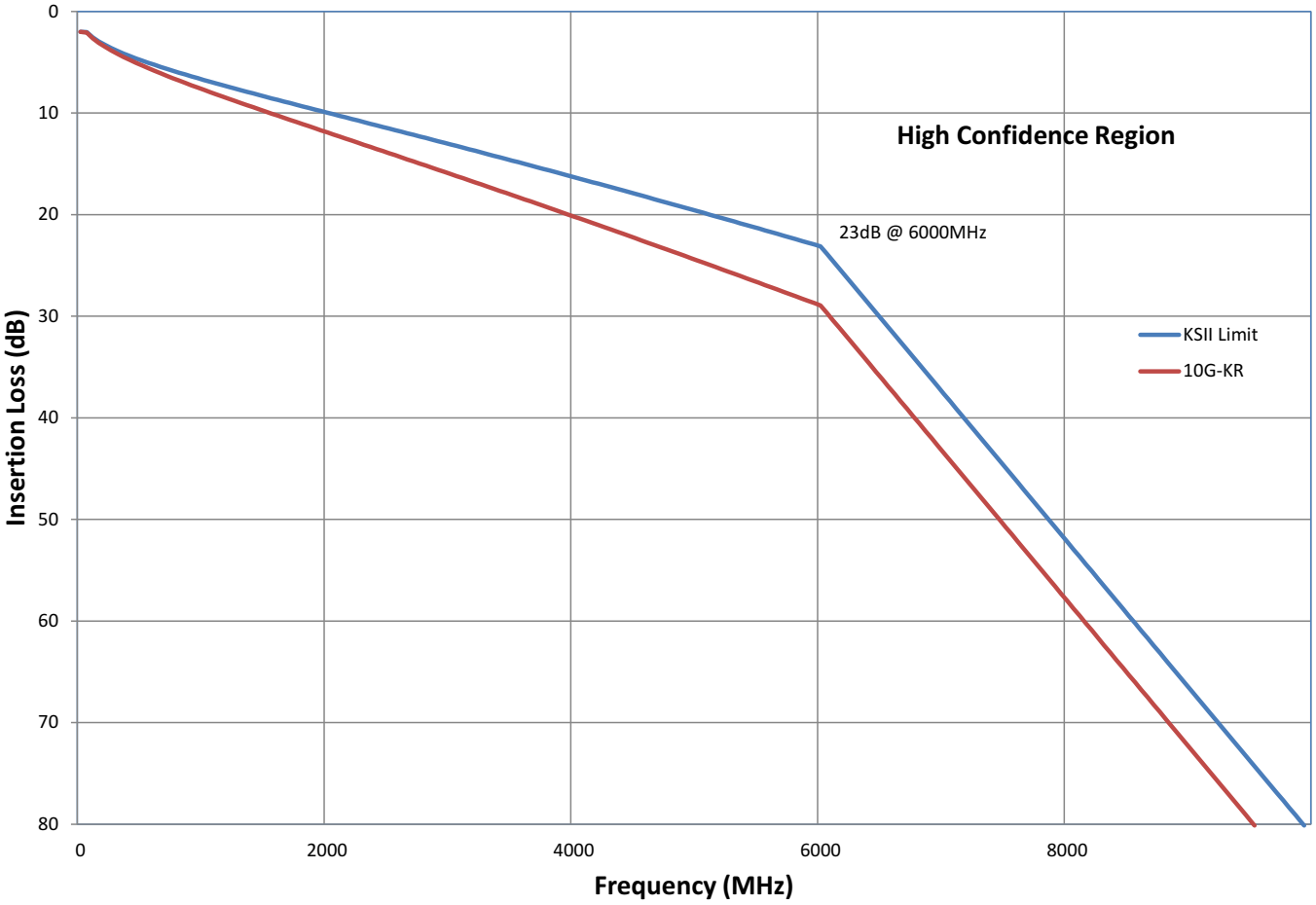
PG3.0 devices have fixed the issue described in Usage Note #27. Due to this design improvement, users may now detect the corrupt state by reading the PCS-R RX Status Register's hiber and block errors fields. This means that the PCS-R may be reinitialized and brought to a healthy state prior to packet loss.

KeyStoneII.BTS_errata_advisory.30

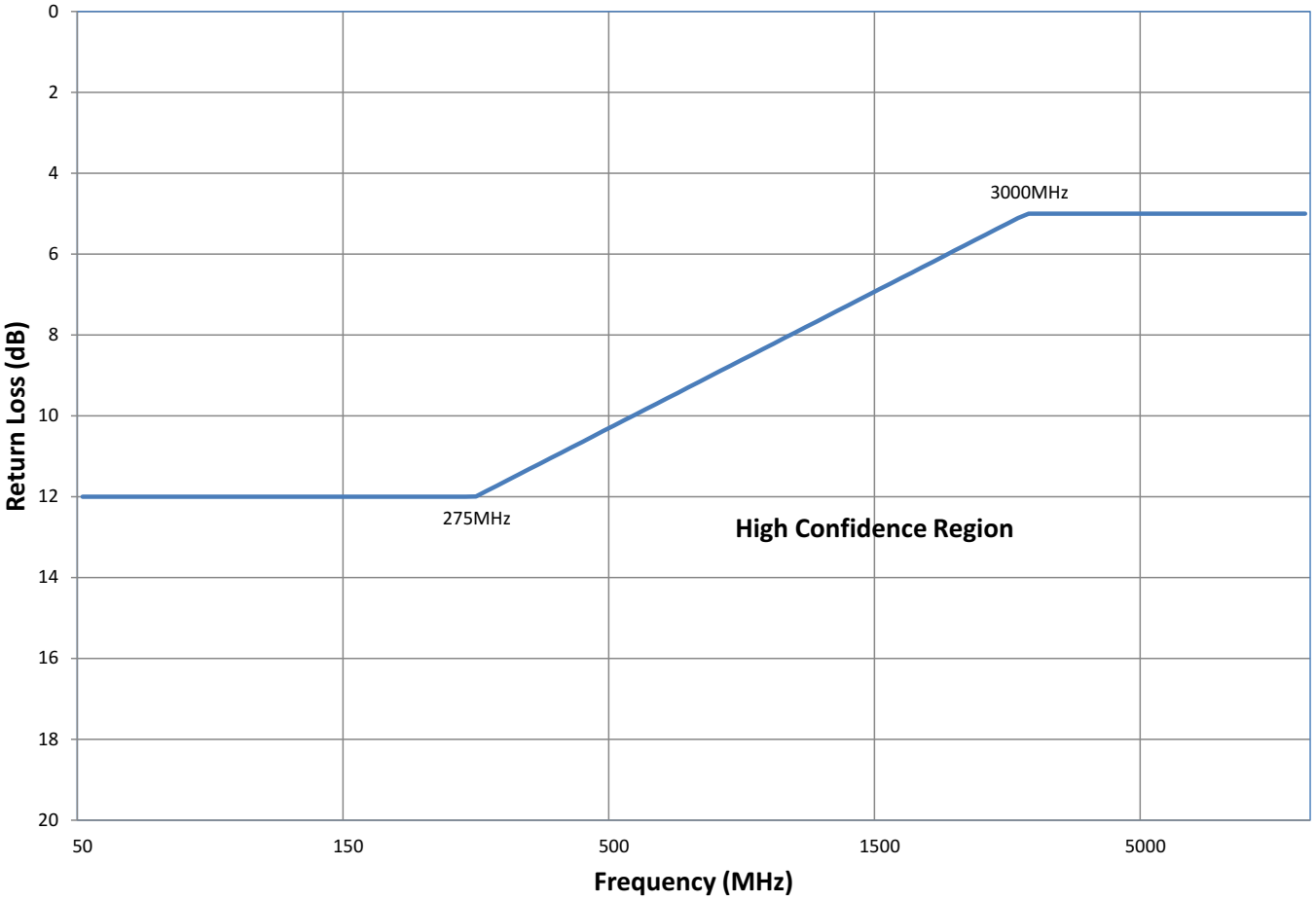
10GbE PCB Channel Loss Limited to 20dB

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details Device characterization has shown that bit error rate performance consistent with the 10GbE-KR standard as published in IEEE 802.3-2008 Annex 69B cannot be met across all device operating extremes. Production devices do operate at the required error rate across all rated operating conditions of voltage and temperature when the channel insertion loss is limited to 20dB at the Nyquist rate of 5.15625GHz. The figure below shows the Insertion Loss template from IEEE 802.3-2008 Annex 69B as well as the reduced Insertion Loss template.



Characterization has shown that channels must also be compliant with the Return Loss template contained in the IEEE 802.3-2008 Annex 69B specification without modification. This is shown below for convenience.



Workaround None.

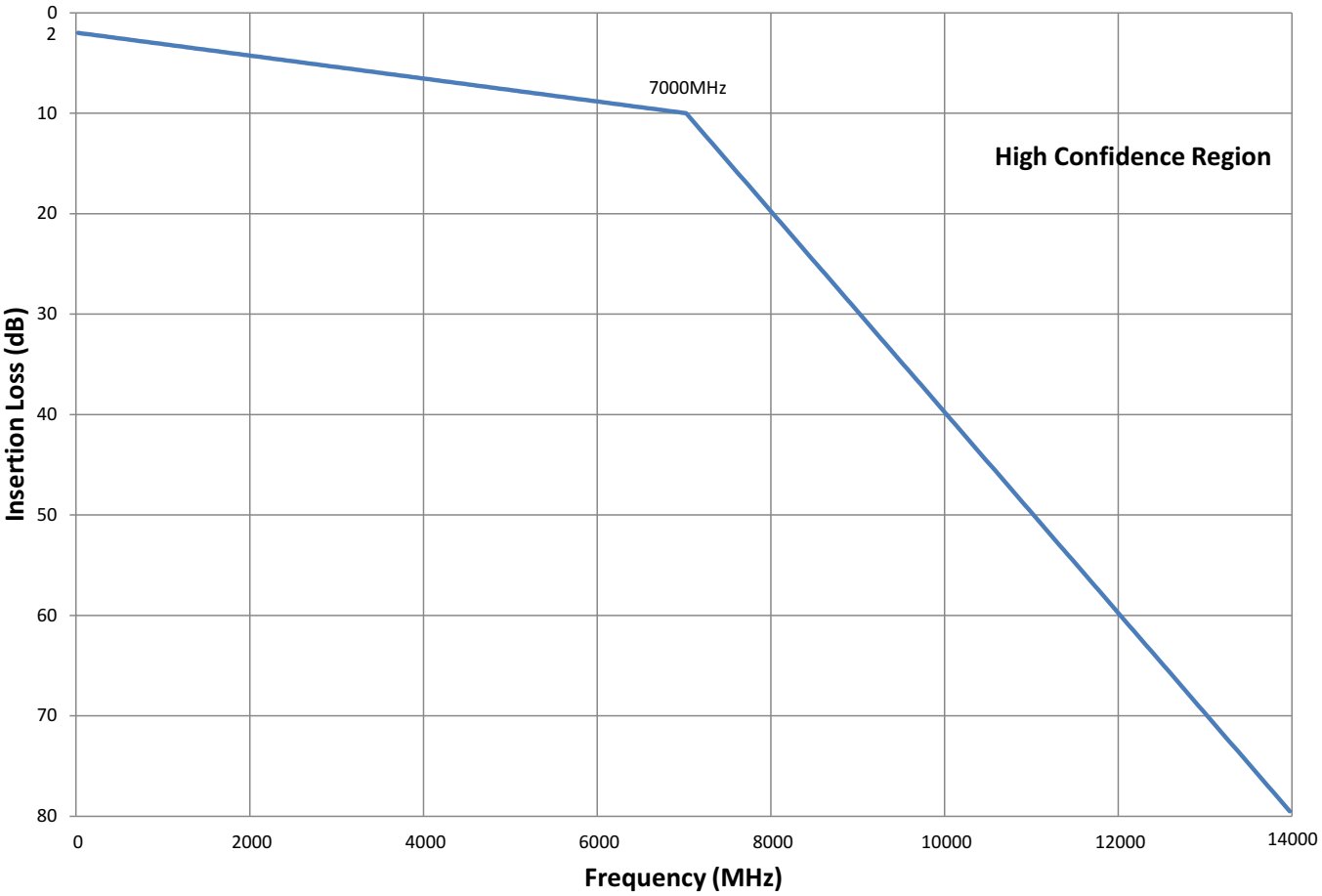
KeyStoneII.BTS_errata_advisory.31

HyperLink Channel Loss

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

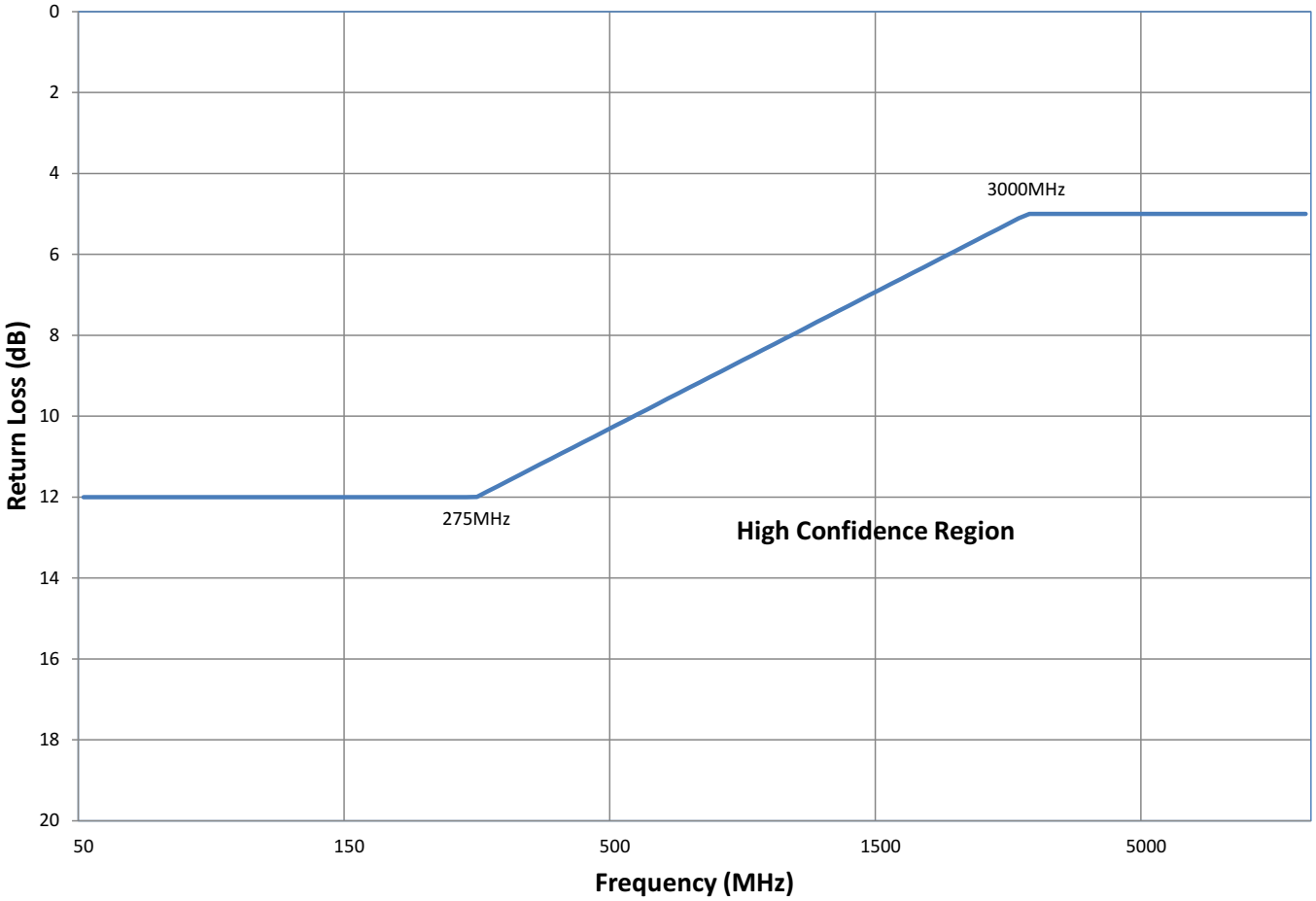
Details Device characterization has shown that expected bit error rate performance can only be met on layouts that meet constrained channel loss metrics. These can be examined by generating Insertion Loss and Return Loss graphs where loss is plotted versus frequency. Insertion Loss and Return Loss can be estimated during PCB layout using a 3D simulation tool. Insertion Loss and Return Loss should also be measured on early prototypes using a Vector Network Analyzer (VNA). Multiple PCB construction and layout enhancement techniques must be combined to achieve these channel performance requirements as stated in the KeyStone II SERDES User Guide.

An Insertion Loss measurement provides a composite view of the channel's loss versus frequency. Since HyperLink is a short-reach interface, channel loss is limited. Also, due to the high data rate, the Insertion Loss must be monotonic and without discontinuities that would degrade receiver operation. The template shown below provides Insertion Loss limits that will result in a robust channel. This template must be met for the entire channel from transmitter to receiver including the series capacitors.



Device characterization has also shown that channels should be compliant with the Return Loss template shown below. The Return Loss measurement indicates the amount of energy reflected back to the source which is not available to the receiver. Impedance variation in the channel will cause reflected energy that can cause this template to be violated. Careful attention to PCB construction and layout must be followed to meet this requirement. This template must be met for the entire channel from transmitter to receiver including the series capacitors. Note that this is the same template

defined in IEEE 802.3-2008 Annex 69B specification for 10GbE.



Workaround None.

KeyStoneII.BTS_errata_advisory.33

HyperLink Data Rate Limited to 40 Gbaud Issue

Revision(s) Affected: 1.0, 1.1, 2.0, 3.0, 3.1

Details: The HyperLink interface is currently limited to a maximum transfer rate of 10 Gbaud per lane (40 Gbaud for four lanes) due to a SerDes PLL limitation.

KeyStoneII.BTS_errata_advisory.34

DDR3 Limited to Highest Data Rates Due to Possibility of PLL Instability During Temperature Changes After Startup

Revision(s) Affected 1.0, 1.1, 2.0

Details: DDR3 PHY circuitry simulations have shown that there is a very small possibility of DDR PHY PLL instability when booting the devices at low temperatures and increasing the device temperature during run-time. This instability results in timing margin loss in the DDR3 interface when it occurs. Depending on system design and available DDR3 timing margin, this may result in DDR3 write data errors.

The possibility is highest when starting the device at very cold temperatures - approaching -40 degrees Celsius, and ramping to above 20 degrees Celsius. The probability is reduced when units are first initialized above 0 degrees Celsius and even less when initialized above 20 degrees Celsius. The possibility is highest if running the DDR3 PHY at 1066 MT/s or lower.

Workaround There is no hardware or software workaround at this time. In order to avoid the write data errors do not operate DDR3 PHY at 800 MT/s or 1066 MT/s in production systems. The recommendation is to only operate production systems at and above 1333 MT/s.

This does not preclude platform development activities that routinely use data rates as low as 800MT/s on customer prototypes when the DDR3 layout is suboptimal, or for pre-production development.

KeyStoneII.BTS_errata_advisory.35

Restrictions on DDR3 PLL Configuration and DDR3nCLK to Eliminate DDR3 Errors Due to PLL Dynamic Phase Offset

Revision(s) Affected 1.0, 1.1, 2.0

Details: DDR3 read operations may show errors due to increased levels of Dynamic Phase Offset (DPO) inside the DDR3 PHY. These errors are characterized by byte lane data corruption or intermittent bit errors. The byte lane data corruption can only be recovered using a device reset.

The DDR3 clocking architecture is illustrated in the figure below. The DDR3 reference clock input (DDR3nCLK) feeds the DDR3 SoC PLL, which outputs a clock to be used by the PHY Control Logic and provide an input to the PHY PLL.

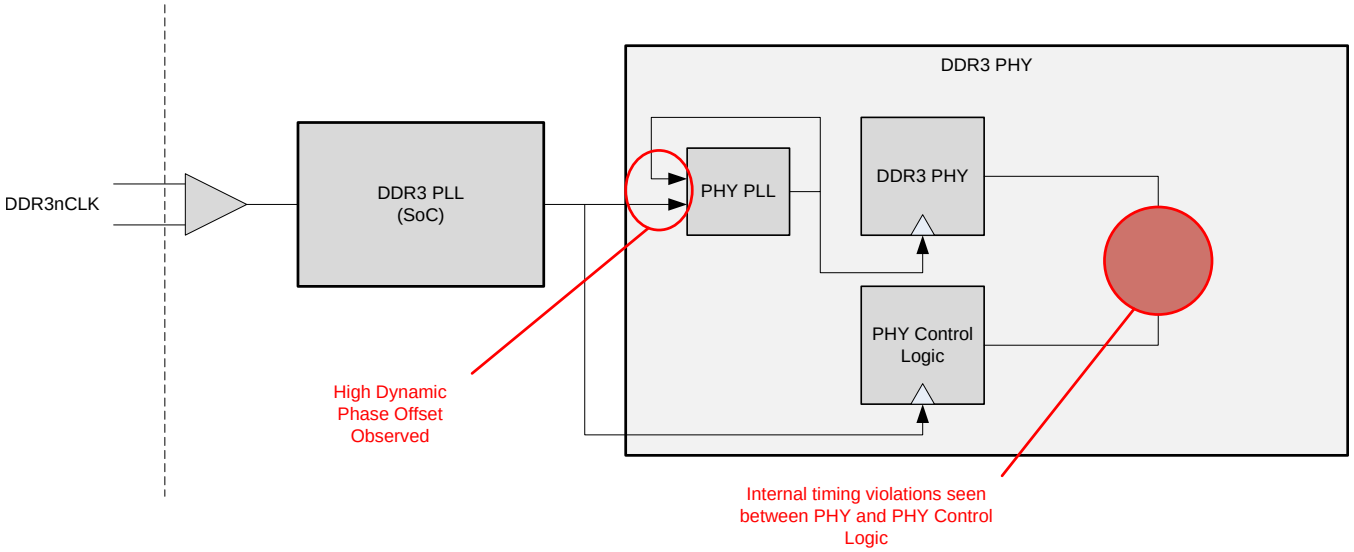


Figure 5. DDR3 PHY

The Dynamic Phase Offset (DPO) is a measure of the DDR3 PHY PLL's ability to track rapid phase changes in the output of the DDR3 SoC PLL. High DPO can create internal timing violations between the DDR3 PHY and DDR3 PHY Control Logic and result in data errors during reads. As indicated by the shaded red circle in the figure above, the timing violations occur in the data transfer between the PHY and PHY control logic. It was determined that the root cause of the elevated DPO were SoC PLL and PHY PLL configurations that were non-optimized for the timing requirements of the system.

Workaround The workaround for this behavior is a change in the PLL configurations used for the DDR3 SoC PLL and the DDR3 PHY PLL. New configurations have been generated for DDR3-1600, DDR3-1333, DDR3-1066, and DDR3-800 operation and are presented in the tables below. Per field alert, *"DDR3 Limited to Highest Data Rates Due to Possibility of PLL Instability During Temperature Changes after Startup"*, operation of DDR3-1066 and/or DDR3-800 is not recommended for affected Keystone2 devices in production systems

The DPO has only been characterized for these new configurations with a DDR3 reference clock frequency of 100 MHz.

Table 9. DDR3-1600

Configuration Name	Value	Register.BitField Name	Register Value
Reference Clock Input (DDR3nCLK)	100 MHz	N/A	N/A

Table 9. DDR3-1600 (continued)

Configuration Name	Value	Register.BitField Name	Register Value
PLL Reference Divider	1	DDR3nPLLCTRL0.PLLD	0
PLL Multiplier	40	DDR3nPLLCTRL0.PLLM	39
PLL Output Divider	10	DDR3nPLLCTRL0.PLLOD	9
PHY PLL Frequency Select (In DDR3 Initialization)	N/A	PLLCR.FRQSEL	0x3
PHY PLL Charge Pump Proportional Current Control (In DDR3 Initialization)	N/A	PLLCR.CPPC	0xF

Table 10. DDR3-1333

Configuration Name	Value	Register.BitField Name	Register Value
Reference Clock Input (DDR3nCLK)	100 MHz	N/A	N/A
PLL Reference Divider	1	DDR3nPLLCTRL0.PLLD	0
PLL Multiplier	40	DDR3nPLLCTRL0.PLLM	39
PLL Output Divider	12	DDR3nPLLCTRL0.PLLOD	11
PHY PLL Frequency Select (In DDR3 Initialization)	N/A	PLLCR.FRQSEL	0x3
PHY PLL Charge Pump Proportional Current Control (In DDR3 Initialization)	N/A	PLLCR.CPPC	0xF

Table 11. DDR3-1066 (1)

Configuration Name	Value	Register.BitField Name	Register Value
Reference Clock Input (DDR3nCLK)	100 MHz	N/A	N/A
PLL Reference Divider	3	DDR3nPLLCTRL0.PLLD	2
PLL Multiplier	112	DDR3nPLLCTRL0.PLLM	111
PLL Output Divider	14	DDR3nPLLCTRL0.PLLOD	13
PHY PLL Frequency Select (In DDR3 Initialization)	N/A	PLLCR.FRQSEL	0x3
PHY PLL Charge Pump Proportional Current Control (In DDR3 Initialization)	N/A	PLLCR.CPPC	0xF

(1) These speeds are not to be used in production systems with the affected K2 devices. This is in accordance with KeyStoneII.BTS_errata_advisory.34

Table 12. DDR3-800 (1)

Configuration Name	Value	Register.BitField Name	Register Value
Reference Clock Input (DDR3nCLK)	100 MHz	N/A	N/A
PLL Reference Divider	1	DDR3nPLLCTRL0.PLLD	0
PLL Multiplier	32	DDR3nPLLCTRL0.PLLM	31
PLL Output Divider	16	DDR3nPLLCTRL0.PLLOD	15
PHY PLL Frequency Select (In DDR3 Initialization)	N/A	PLLCR.FRQSEL	0x3
PHY PLL Charge Pump Proportional Current Control (In DDR3 Initialization)	N/A	PLLCR.CPPC	0xF

(1) These speeds are not to be used in production systems with the affected K2 devices. This is in accordance with KeyStoneII.BTS_errata_advisory.34

KeyStoneII.BTS_errata_advisory.36 *Two Masters Accessing Two C66x CorePac L2 Memories Can Cause TeraNet Hang*

Revision(s) Affected 1.0, 1.1. 2.0

Details When two masters send out write transactions to two C66x CorePac's L2 slave end points, a hang can occur when the C66x CorePacs are busy and stalling the incoming write transactions.

Each C66x CorePac has a TeraNet component in front of its slave (SDMA) port, which serves as a satellite TeraNet for C66x CorePac's SDMA port. Device masters communicate with the C66x CorePac's L2 memory through these satellite TeraNets. These satellite TeraNets have two input paths, path A and path B, and each path can be accessed by distinct set of masters. Thus, each write transaction appearing at the SDMA ports of C66x Corepac L2 memory can take one of the two possible paths, depending on the master that sent out that transaction.

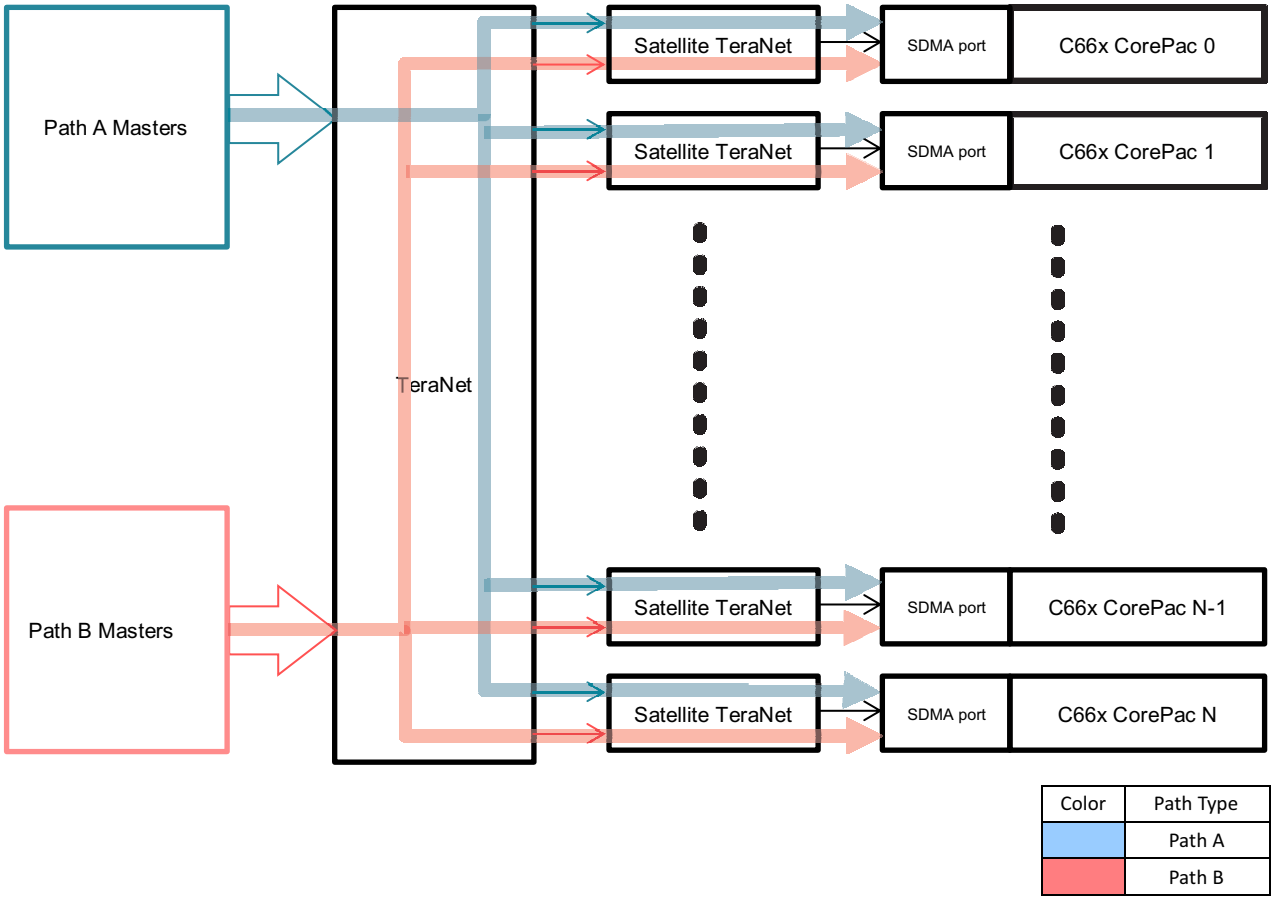


Figure 6. Satellite TeraNet Paths

Hang Scenario: When a master (MST A) that uses path A and another master (MST B) that uses path B send out two writes each (WA1, WA2 and WB1, WB2) to two C66x CorePac slave end points (SLV X, SLV Y), a hang can occur if the following conditions occur:

- Both masters send one write to each of the two slave end points in opposite order:
  - MST A sends two writes: first write WA1 to SLV X, and second write WA2 to SLV Y
  - MST B sends two writes: first write WB1 to SLV Y, and second write WB2 to SLV X
- Writes from two masters arrive on separate paths at slave endpoint:
  - MST A writes arrive on path A
  - MST B writes arrive on path B
- Both slaves are busy, so they stall the incoming write transactions.
- Both slaves select second writes when they are ready:
  - SLV X selects WB2
  - SLV Y selects WA2

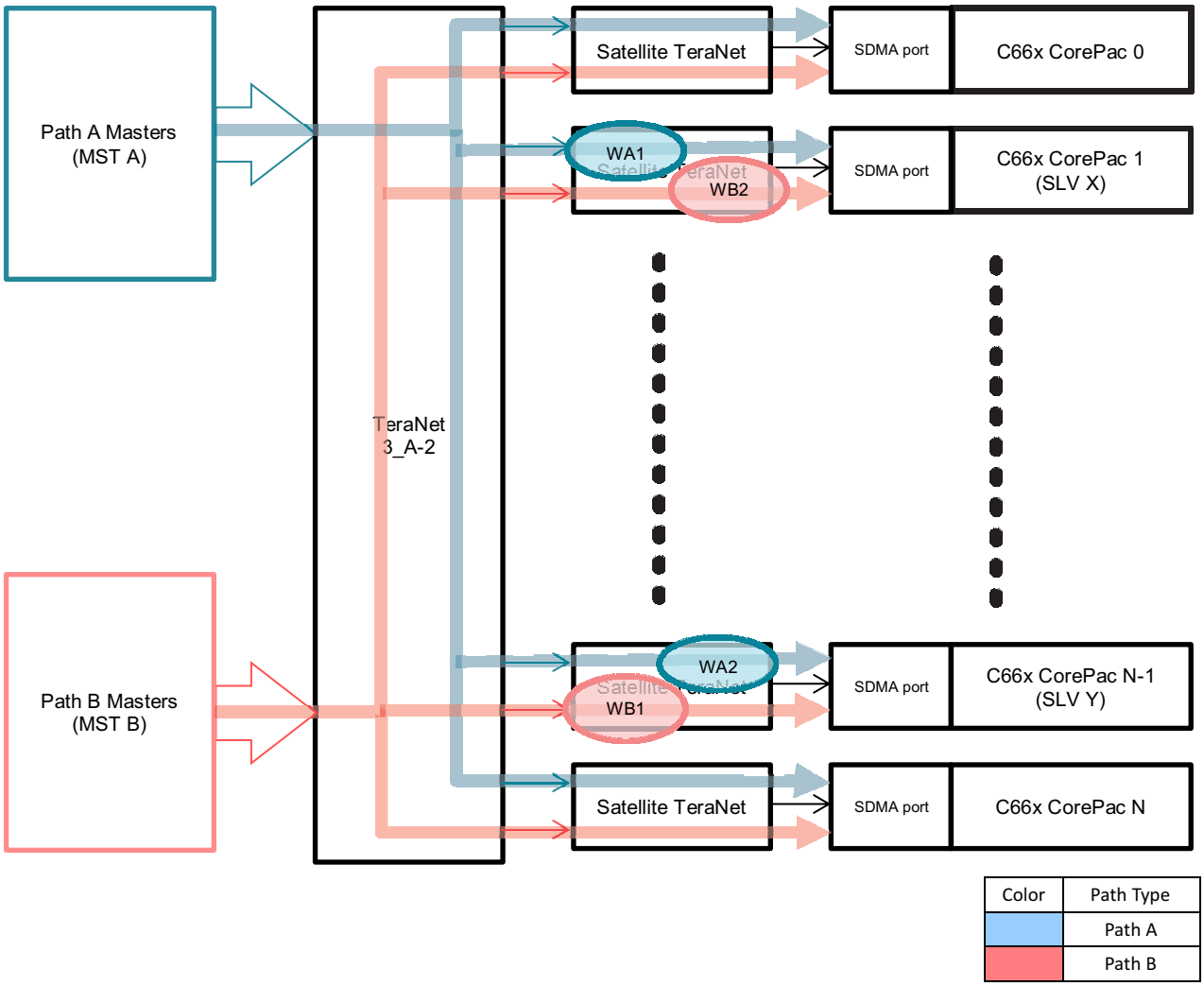


Figure 7. Hang Scenario

In the above scenario:

- The selection occurs such that if the satellite TeraNet in front on SLV X selects path B which has WB2, and satellite TeraNet in front on SLV Y selects path A which has WA2, result will be a hang.
- Based on selection, both the slaves receive a second write command transaction from each master, while the first write command transaction from each master will be later in the sequence for the slaves.
- As a result, slaves expect write data for second writes (WA2, WB2) and then write data for first writes (WA1, WB1).
- Masters do not send the write data for their second writes (WA2, WB2) until they send out write data for their first writes (WA1, WB1).
- Slave end points continue to wait for second-write data from masters before selecting first writes.
- Masters wait for first write to get selected by slave end points, to send out first-write data and then second-write data.
- The result is a deadlock, due to interdependency.

For the Keystone II devices mentioned, MST A and MST B can be any master from the “Path A Master” and “Path B Master” sets, respectively, as mentioned in ; SLV X, SLV Y can be any two C66x CorePac’s L2 SDMA port.

Workaround

If the system changes the traffic profile of the slaves that are affected, then some workarounds can avoid the hang condition.

Workaround #1:

- The deadlock requires writes arriving on both paths toward the slave, as eliminating writes on one of the paths will eliminate the deadlock.
- Only write to an affected slave by a master that is on the selected path, and stop any writes from any master that uses another path.
- The path to enable can be chosen per affected slave.
- Each affected slave end points should be assigned to either path A master or path B master set.
- All writes to that end point should always be from that pre-assigned set of masters.
- For example, for TCI6638K2K (see Figure 8) for affected slave end point SDMA ports of core 1,2, 6,7.
  - Assign path A masters to SDMA port of core 1, core2
  - Assign path B masters to SDMA port of core 6, core7
  - Allow writes only from path A masters to core 2, core 1 – SDMA port
  - Allow writes only from path B masters to core 6, core 7 – SDMA port
  - Path A masters can write to other affected slaves, which have been assigned path A masters
  - Path B masters can write to other affected slaves, which have been assigned path B masters

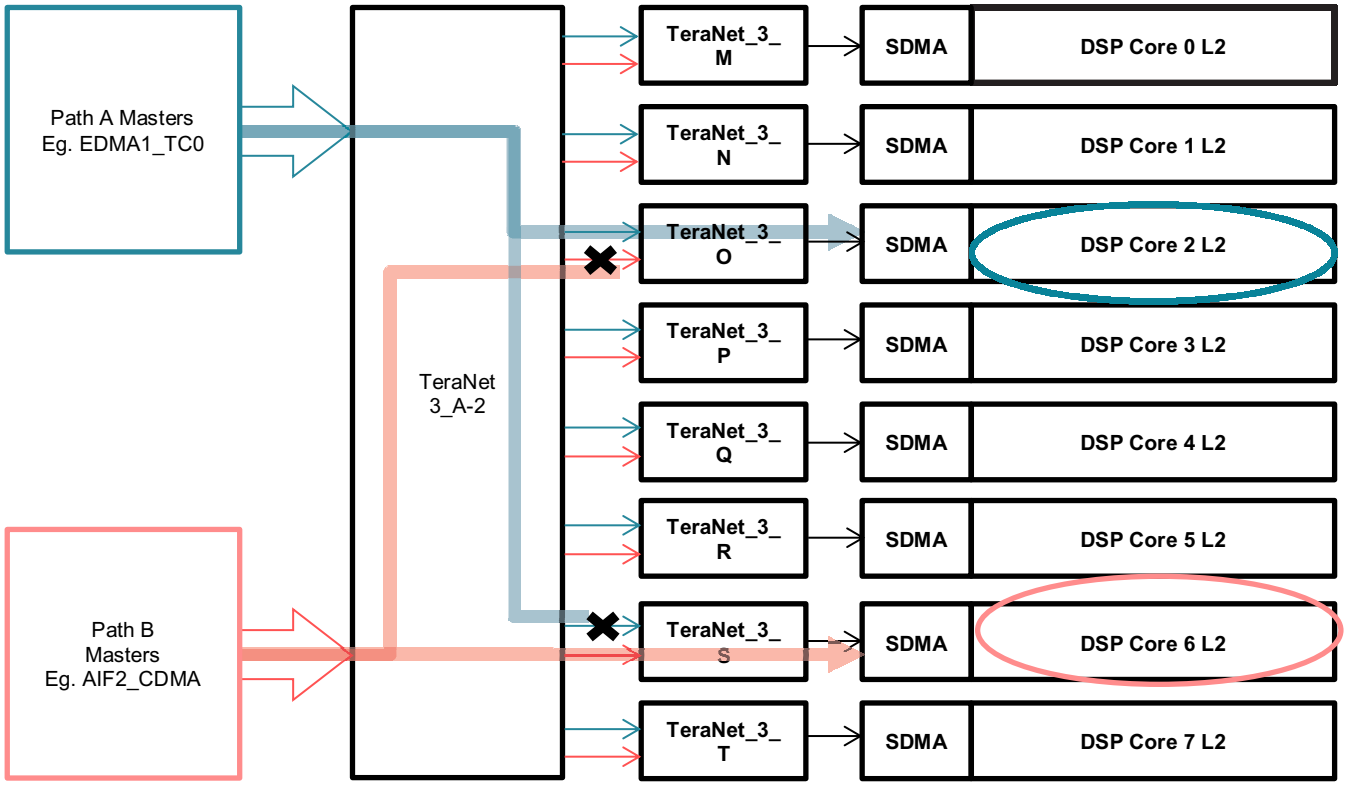


Figure 8. Workaround 1

Workaround #2:

- Prevent any master from writing to more than one of the affected slaves at the same time.
- Thus, if there is a multi-channel master, do not map channels to more than one affected slave.
- This eliminates the chance of an affected slave receiving the second write from a master that is waiting to write to another affected slave (as the writes are now all to the same affected slave).
- For example, see Figure 9 for the TCI6638K2K.
  - If master EDMA1_TC0 sends a write to the core 2 SDMA port, then it should not be allowed to send out any new writes to other affected masters until its existing write is complete.
  - Any other masters can write to other affected masters following the same rule. That is, they only write to one other affected slave at a time. For example, AIF2_CDMA master can write to the core 6 SDMA port, but not to an other affected master.
  - Other masters can write to affected slave that is currently being written, as long as that is the only write it sends out. For example, BCP_DIO could send one write to the core 2 SDMA port if it has no other write outstanding, and sends out no other write until this write completes.

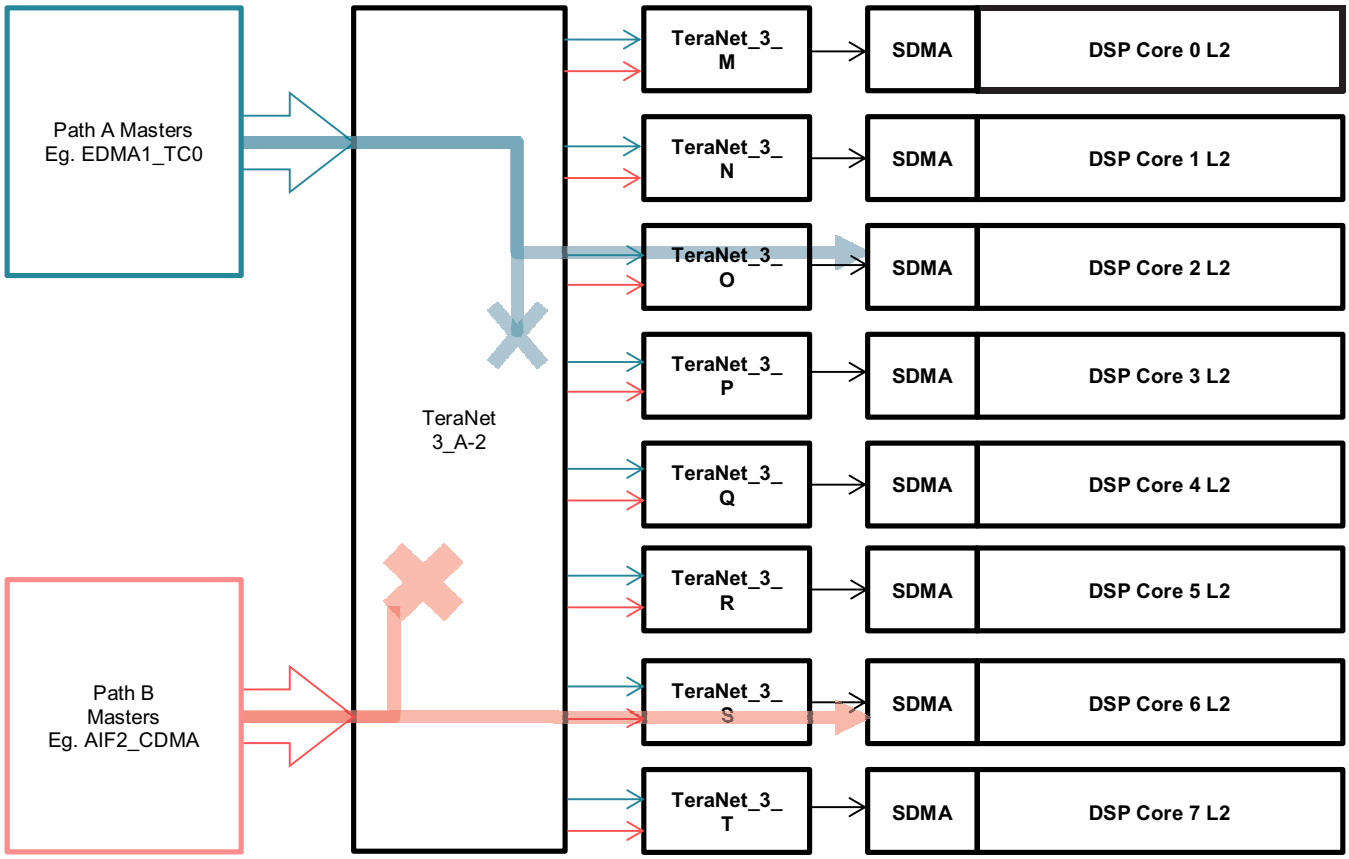


Figure 9. Workaround 2

KeyStoneII.BTS_errata_advisory.37 *USB3.0 PHY Does Not Meet USB Specification: Rev 3.0 RX Jitter Tolerance Mask Across Non- Nominal Process*

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details On silicon revision 2.0 and older, a small number of devices may experience bit errors or link failure in the presence of high frequency (>= 20MHz) phase jitter. This issue may persist on silicon versions 3.x, but is only seen when the case temperature is < –15C. Devices operating at case temperatures >= –15C are not affected.

Workaround None

KeyStoneII.BTS_errata_advisory.38 *PCI-Express Hot Reset Not Handled During Boot*

Revision(s) Affected 1.0, 1.1, 2.0

Details When receiving a PCI-Express (PCIe) Hot Reset request during the PCIe boot process, the PCI-Express Subsystem (PCIESS) disables the Link Training and Status State Machine (LTSSM) and suspends the PCIe bus in the Detect.Quiet state. With the LTSSM disabled, there will be no subsequent transition from Detect.Quiet to Detect.Active, and the PCIe boot process fails.

Workaround

1. Prevent the PCIe Root Complex (RC) from issuing PCIe Hot Reset during the PCIe boot process, or
2. Use PCIE as secondary boot media. Create a small secondary bootloader which boots from primary (non-pcie) boot media to implement a PCIe Hot Reset interrupt handler. When PCIe Hot Reset is detected, this interrupt handler should wait a minimum of 100 ms to ensure that any pending internal transactions are completed before issuing a local reset to the PCIESS. This local reset re-enables the LTSSM, allowing the PCIe to be re-initialized. This approach uses ROM to configure PCIe, but the interrupt handler and the idle loop required by the ROM are provided through the primary (non-pcie) boot.

KeyStoneII.BTS_errata_advisory.39 *CPSW Stall if Duplex Changes During Transmission*

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details

During CPSW packet transmission, due to a dynamic change in the internal state of the full duplex signal (triggered by a duplex change request), the following two errors may occur:

- A TX Underrun error that eventually causes the statistics block to hang. A follow up read of the statistics causes a bus hang.
- Transmission hang, preventing further transmission from the port. The full duplex signal can dynamically change from two sources:
  - When MAC_CTL.EXT_EN bit == 1. This allows HW control of the signal to change in response to the input full duplex signal.
  - When MAC_CTL.EXT_EN bit == 0. This allows SW control of the signal to change MAC_CTL.FULLDUPLEX.

Workaround

Master Mode

For master mode, software is expected to be in control of speed and duplex-switching operations. Frame transmission must be stopped before switching speed or duplex.

During changes to the link, the duplex is temporarily set to half.

1. Disable ALE forwarding on the port in ALE_PORTCTLn.PORT_STATE
2. Wait for MAC_CTL.IDLE to be set to indicate no pending packets
3. Set new speed and/or duplex setting
4. Initialize auto-negotiation
5. Enable ALE forwarding

Slave Mode

Frame transmission must be stopped before changing the full duplex signal.

While in slave mode, it cannot be predicted when a remote will switch speed and duplex, or restart auto-negotation. During changes to the link, the duplex is temporarily set to half.

1. Disable HW control of the duplex signal by setting MAC_CTL.EXT_EN = 0
2. Poll SGMII_STATUS.MR_AN_COMPLETE for auto-negotiation complete.
3. If the speed has changed from the last known speed: Update new speed in MAC_CTL.GIG
4. If the duplex has changed from our last known duplex:
   a. Disable ALE forwarding on the port in ALE_PORTCTLn.PORT_STATE
   b. Wait for MAC_CTL.IDLE to be set to indicate no pending packets
   c. Update new duplex setting in MAC_CTL.FULLDUPLEX
   d. Re-enable ALE forwarding
5. The speed and duplex can be attained from SGMII_MR_LP_ADV_ABILITY or MAC_STATUS.

Slave Mode Workaround Pseudo Code

```
if (SGMII_STATUS.MR_AN_COMPLETE) {
    new_speed = SGMII_MR_LP_ADV_ABILITY.SPEED; //OR MAC_STATUS.EXT_GIG
    new_duplex = SGMII_MR_LP_ADV_ABILITY.DUPLEX; //OR MAC_STATUS.EXT_FD
    if (new_speed != known_speed) {
        MAC_CTL.GIG = new_speed;
        known_speed = new_speed;
    }
    if (new_duplex != known_duplex) {
        ale_port_state = ALE_PORTCTLn.PORT_STATE;
        ALE_PORTCTLn.PORT_STATE = PORT_STATE_DISABLED;
        while (!MAC_STATUS.IDLE);
        MAC_CTL.FULLDUPLEX = new_duplex;
        ALE_PORTCTLn.PORT_STATE = ale_port_state;
        known_duplex = new_duplex;
    }
}
```

KeyStoneII.BTS_errata_advisory.40 *ROM: NAND Boot Failure When Booting Single Block Backup Images*

Revision(s) Affected 1.0, 1.1, 2.0

Details NAND boot fails under the following single-block-backup booting scenario when an unmarked bad block is found:

1. The entire image resides in a single block, and
2. Two images, a primary and backup, are placed in consecutive blocks, and
3. The ROM finds an uncorrectable error in the first block that is not pre-marked as bad, and
4. Proceeds to the second block

During NAND boot, the boot ROM goes block-by-block, reading and processing pages for a complete boot image. To avoid data corruption, the boot ROM skips bad blocks when they are pre-marked or detected, and moves to the next one. However, due to a boot ROM bug, the data of a bad block that is not pre-marked as bad is not discarded, so partially-read data will be a part of the complete boot image. The boot ROM skips without reading data from blocks marked bad

This affects scenarios where an image that fits within one block is written to multiple blocks as backup images. If the initial boot image experiences degradation but is never marked, the partial data will corrupt the backup image.

Workaround Mark bad and degraded blocks, and write image data to known-good blocks.

KeyStoneII.BTS_errata_advisory.41 *NAND Boot Failure With Bit Errors in ECC*

Revision(s) Affected 1.0, 1.1, 2.0

Details A bit error in ECC causes NAND boot to fail.

As NAND pages are read, the spare area is also read for ECC data to be used in error correction calculations. Both areas are susceptible to bit errors. During error correction, an offset is calculated to point to any bits that may be flipped. Due to a boot ROM bug, if the offset points to a bit in the ECC data, the entire page will be discarded.

Workaround None

KeyStoneII.BTS_errata_advisory.42 *ARM Ethernet Boot Reinitializes the Switch With Reset Isolation Enabled, Which May Cause Lockup*

Revision(s) Affected 1.0, 1.1, 2.0

Details ARM Ethernet boot will reinitialize the switch, regardless if reset isolation is enabled. If a packet is in transit through the switch, it may cause it to lock up.

Reset isolation maintains PLL settings to subsystems so they may remain operational through hard and soft-type resets. The boot ROM does not take into account if reset isolation is enabled on the switch, and reinitializes it for all reset types. This causes the switch to lose any configuration set before the reset occurred.

Workaround Disable reset isolation on the switch if an Ethernet boot is used.

KeyStoneII.BTS_errata_advisory.44 *SerDes RX Adapts to a BOOST Value of 0 and Cannot Move From a Value of 0*

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details When an incoming SerDes signal has too much high-frequency gain or has not experienced a significant amount of high-frequency loss on a channel, the BOOST block within the RX attempts to compensate by equalizing the signal during RX adaptation. Specifically, the high-frequency gain is compensated by reducing the adapted BOOST value. If the BOOST value adapts to the lower limit of the BOOST settings, it will adapt to a value of 0.

There is a deficiency in the SerDes PHY, where if the BOOST adapts to a value of 0, then any subsequent RX CDR resets (performed by toggling the RX signal detect) used to repeat RX adaptation will not be able to move the boost value away from a value of 0. The value of 0 remains, even if this value of 0 is not optimal for the incoming RX signal. In other words, if the BOOST reaches a value of 0 during RX adaptation, it remains at that value until the SerDes PHY is power-cycled.

Workaround Ideally, the high-frequency gain of the incoming SerDes signal must be low enough such that the BOOST does not adapt to a value of 0; it may be possible to reduce the high-frequency content of the incoming signal by lowering the slew-rate of the remote TX. Because the BOOST has compensated to its lowest limit, it is likely that the value of 0 is not optimal to properly equalize the incoming SerDes signal.

A workaround can be performed to manually bump the BOOST out of this 0 state and into a state in which the RX can re-adapt to a new BOOST value. The workaround sequence should be performed immediately after RX adaptation has occurred during initialization. The sequence is used in the CSL/MCSDK for all interfaces in diagnostic (diag) mode, and is used in XGE in functional mode. The sequence is captured in CSL/MCSDK v3.1.4.7 and later in the following function:

For PHY-A and PHY-B: *CSL_SerdesAttBoostPatch()*

KeyStoneII.BTS_errata_advisory.45 *PCIe link power states other than L0 are not supported on some Keystone II SOCs*

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details The PCIe controller is unable to detect when a receiver lane has entered the electrical idle condition. This makes it impossible for the controller to correctly manage power-down states for the lane. If a link power state change is initiated, either through Active State Power Management (ASPM) or software control, the controller cannot complete the steps necessary to enter and exit the new state. It will get stuck waiting for the electrical idle status to change.

Workaround The PCIe lane power state is L0 when lane initialization is complete. To avoid this problem, it must remain in L0 until the system is ready to power down. The system must not direct the PCIe lanes to enter the L1 or L2 power states. The link partners of affected receivers must not enter L0s, or transmit the Electrical Idle Ordered Set (EIOS) to prevent the affected receivers from entering L0s.

To ensure the device stays in L0, the ASPM Control field (ACTIVE_LINK_PM) in the Link Control Register (LINK_STAT_CTRL) should be set to 0 to disable ASPM. This value should be set on both the Keystone II device and the link partner device.

TI recommends changing the advertised ASPM capabilities through the AS_LINK_PM field in the Link Capabilities (LINK_CAP) configuration register. By default, this is set to advertise L0s support. Changing this field to a value of 0 indicates no support for ASPM.

For Linux users, the PCIe controller is initialized with Power Management disabled by default. Though Kconfig has the option to enable power management by configuring PCIe being in PCIEASPM_PERFORMANCE or PCIEASPM_POWERSAVE, users must not enable it.

KeyStoneII.BTS_errata_advisory.46 *Queue Diversion Failure*

Revision(s) Affected 1.0, 1.1, 2.0

Details

The problem affects the descriptor region ID stored in the linking ram when performing a queue diversion.

When performing a push, the QM translates the pushed address to an index based on the defined descriptor region definitions, and stores the region ID of the new descriptor in the linking ram index of current tail descriptor (else the queue ram if the queue is empty, or a push to head). On a pop, it uses the stored region ID to reconstruct the descriptor address.

When a queue diversion “to tail” is performed, the QM updates the region ID field in the linking ram entry of the destination queue’s tail descriptor to point to the linking ram entry of the first source descriptor. In cases where the destination queue is not empty, this bug causes this region ID field in the internal queue ram for the destination queue to be updated instead of the linking ram. If, however, the region ID for the descriptors in both source and destination queues are the same, the result is a correct queue diversion, though the bug is still present.

When a queue diversion “to head” is performed, the QM updates the destination queue ram to point to the first source descriptor; this is not affected by the bug.

Finally, the QMSS Accumulator firmware has a queue diversion feature that writes to the QMSS Queue Diversion register. Applications using this feature should check to see if the conditions for a problematic queue diversion exist. The other features of the Accumulator firmware are unaffected.

In summary, only queue diversions sharing all of the following attributes are affected:

- Queue diversion “to tail”
- Destination queue is not empty
- Region IDs for the descriptors in the source and destination queues are different

Workaround

1. Limit queue diversion to “divert to head” only. Select this workaround if the order of descriptors in the destination queue is not important.
2. Limit queue diversion to source and destination queues containing descriptors from the same descriptor region. Select this workaround if descriptor order is important, and it is not difficult to constrain diversions such that both queues contain descriptors from the same descriptor region.
3. Perform a double divert: first, divert a non-empty destination queue to the head of the source queue, then divert the source back to the destination queue. Select this workaround if descriptor order is important, workaround 2) is not feasible, and other processes are not pushing to the destination queue, which makes this workaround prone to race conditions (the result being out-of-order descriptors).
4. Test the destination queue for queue empty prior to diversion. Select this workaround if all others are not feasible, because a) it adds a small amount of overhead, and b) it is also subject to race conditions if other processes may push to the destination queue.

KeyStoneII.BTS_errata_advisory.47 *SGMII, SRIO, Hyperlink, and PCIe Boot Failure*

Revision(s) Affected 1.0, 1.1, 2.0, 3.0

Details SGMII, SRIO, Hyperlink, and PCIe boot may fail on certain devices because their SerDes links fail.

The Ethernet, SRIO, Hyperlink, and PCIe interfaces contain a Serializer/Deserializer (SerDes) unit. The SerDes requires a base configuration for operation, which includes a series of register writes to its memory mapped range.

The setting for the DFE comparator bandwidth, DFE_BW_SCALE, was selected to 1/2 speed to optimize lane power versus performance. However, this selection results in insufficient RX margin across process, voltage, and temperature variation, causing a link to fail on certain devices. The proper setting to cover devices across all PVT is full speed.

The following boot modes utilize the SerDes unit: SGMII, SRIO, Hyperlink, and PCIe. Because this setting is in ROM, it cannot be changed in the first stage, and boot ultimately fails on certain devices.

Workaround Avoid using the affected boot modes; use an alternate boot mode from onboard memory, such as SPI or NAND.

Use a first stage that configures the SerDes with a higher DFE_BW_SCALE value and the switch subsystem before re-entering the boot ROM for an Ethernet boot. Examples for two-stage booting can be found in the Keystone2 boot examples, and the reference code for setting up the switch can be found in the PDK.

KeyStoneII.BTS_errata_usagenote.1

SPI Boot Size Limitation, C66x Master Boot

Revision(s) Affected 1.0

Details The issue is within the Boot ROM. In C66x Master, SPI Boot Mode, the SPI Boot Size is limited by the Boot ROM to a single block.

In SPI Boot Mode, the SPI Boot Size is limited by the Boot ROM to a single block. The max block read size is 8K bytes for C66x master boot.

Workaround A workaround is available that contains the fix in that first block read. The workaround can simply be prepended to customer boot images when placed on the flash. To obtain the workaround contact the TI E2E Forum or your local TI representative.

KeyStoneII.BTS_errata_usagenote.2

ARM Core Hangs if Timer is Accessed While ARM Core is in Wait-In-Reset State

Revision(s) Affected 1.0, 1.1

Details While the ARM CorePac is held in Wait-In-Reset (WIR) state by the DEBUGSS, any access to the timer memory mapped registers by the Code Composer Studio will cause the ARM CorePac to hang.

Access to Timers associated with the ARM CorePac will hang when the ARM cores are held in reset by the DEBUGSS (a $\overline{\text{LRESET}}$ applied to the ARM CorePac). This scenario would happen typically in CCS code development environment when bringing up the CCS with all cores held in wait-in-reset. If there is a memory window that points to the Timer configuration space, the ARM CorePac will be hung.

NOTE: For a description of Wait-In-Reset, see the TI Embedded Processors Wiki.

Workaround To avoid this issue, use the GEL command, GEL_MapReset() that masks (unmap) the memory map configuration (including the Timer configuration space. Note that you cannot access those Timer registers from the CCS memory window.) To view other memory spaces, use the GEL_MapAddStr () to add them in.

KeyStoneII.BTS_errata_usagenote.3

Debug System (DebugSS) Trace Buffer EDMA via System Port Not Functional

Revision(s) Affected 1.0, 1.1

Details Debug System (DebugSS) Trace Buffer (TBR) EDMA via System Port is not functional.

The TBR in the DebugSS is used to capture the output of CP-Tracers and System Trace Module (STM). The trace data can be accessed by debugger or application from either the configuration port or the system port. The system port is ideal for another master such as EDMA to read the trace data to a larger memory region. This allows more trace data to be captured into on-chip memory beyond the limited size of trace buffer.

The DebugSS TBR clock is integrated incorrectly in the design. Consequence of this is preventing the usage of EDMA to drain the contents of TBR.

Workaround None for the EDMA use case from the system port. The TBR can still be accessed via the configuration port.

KeyStoneII.BTS_errata_usagenote.4

ARM Boot Can Fail When Interrupt Enabled

Revision(s) Affected 1.0, 1.1

Details ARM boot can fail when an interrupt is pending when interrupts are enabled.

From the ARM Boot ROM code, the “Enable interrupts” code below in the ROM is not implementing correctly.

```
*******************************************************************************
* Enable interrupts
*******************************************************************************

.def _chipEnableInts

_chipEnableInts:

cpsie i
bx  lr
```

If an interrupt is pending when the cpsie instruction executes the interrupt is immediately taken. The interrupt code executes normally, and even returns with the correct mode. However the next instruction, bx lr, does not execute. The code simply continues into the next instruction, which happens to be a data value. This results in an invalid instruction exception.

For devices including the C66x CorePac:

In a case of the C66x boot master and a $\overline{\text{RESET}}$ is applied, the $\overline{\text{RESET}}$ resets the ARM PLL causing the ARM boot code to execute very slowly. The system PLL was reset isolated and executed very quickly. The C66x boot code was loaded, and this code poked the IPC interrupt to wake up the ARM, but the slow ARM was still setting up translation tables. If there is an interrupt pending, the interrupt is taken when the interrupts were enabled.

This could also happen when the ARM is the boot master in the PCIe and Hyperlink boot modes. If the remote end generates an interrupt immediately after link detection it is possible that the ARM code has not yet reached the cpsie instruction. But in both of these modes, the ARM PLL will be enabled and the race is very short.

The consequence of this is that the ARM generates an invalid instruction exception.

Workaround Delay interrupts to the ARM for a few ms. This applies mostly to ARM core 0. Secondary ARM cores are usually woken up without an interrupt, however the same code can execute if the secondary ARM core wakes up and does not see a branch address, and in which case it enables interrupts and idles.

KeyStoneII.BTS_errata_usagenote.5

Boot I²C Frequency Incorrect

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details The issue is within the Boot ROM. Initial boot I²C frequency incorrect on $\overline{\text{RESET}}$ reset.

For I²C boot, the code to determine the device frequency assumes that the PLL is in bypass. This is true for power on reset, but for $\overline{\text{RESET}}$, with the PLL reset isolated this is incorrect. The code should check to see if the PLL is enabled and if it is, it should return the e-fuse device frequency.

On a normal $\overline{\text{POR}}$ or $\overline{\text{RESETFULL}}$, boot with I²C as the boot master, the boot code will correctly assume the system is running at 312 MHz (max possible) and scale the I²C clock to run at 20 KHz. So the actual frequency will scale based on the actual reference clock. After boot execute a $\overline{\text{RESET}}$, the initial frequency will be much higher, running faster by a multiple equal to the actual effective PLL multiplier value.

For example if the actual reference clock is 50 MHz and the device frequency is e-fused for 1400 MHz, the initial I²C will read using a data clock of 20 * 50 / 312 = 3.2 KHz. After a $\overline{\text{RESET}}$ reset, with the PLL reset isolated, the initial read will be at 20 * 1400 / 312 = 89 KHz. This will work with almost all I²C devices that are compliant to the 100 KHz bus standard specified in the original I²C specification. This represents the worst case for these devices.

Workaround None.

KeyStoneII.BTS_errata_usagenote.6

Access to DDR3 Without Configuring PHY Properly Can Cause Hang

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details If the DDR3 is not configured properly before the CorePac issues an access to DRR3, the device could lock up.

If the DDR3 PHY Utility Block (PUB), DDR3 PHY and the EMIF Controller are not configured or improperly configured, any access to the DRR3 memory space is issued by the CorePac, including opening a memory window view from the Code Composer Studio (CCS) that is pointed to the DDR3 memory space, the device could lock up.

Workaround It is recommended that before issuing an access to the DDR3, the device must properly initialize the DDR3 PUB, DDR3 PHY, and EMIF controller.

Refer to the KeyStone II DDR3 Programming Sequence documented in the KeyStone II DDR3 User Guide (SPRUGV8) or the KeyStone II DDR3 Initialization Sequence document (SPRABL2).

KeyStoneII.BTS_errata_usagenote.7

C66x CorePac and ARM CorePac AVS Rails

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details Separating the AVS rails for the C66x CorePac and the ARM CorePac exhibits a transient voltage on the ARM CorePac.

There are two different AVS rails specified on these devices, one for the C66x CorePacs, CVDD, and one for the ARM CorePac, CVDDT. On PG 1.0 silicon, an issue with transient voltage of the ARM CorePac is observed by TI test engineering if these two rails are supplied separately.

Workaround On PG 1.0 silicon, TI recommends that both AVS rails must be tied together.

On PG 1.1 silicon, these two rails are tied together internally.

KeyStoneII.BTS_errata_usagenote.9

Core Wake Up on RESET Usage Note

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details

Execution may start only on some CorePacs if CCS is connected to the device and reset is applied via the RESET pin on the device. In order to make sure that all the CorePacs wake up after reset via RESET pin, the device needs to be completely disconnected from the CCS before applying reset via RESET pin.

Some of the CorePacs do not wake up on RESET reset when the device is connected via CCS. When the device is connected via CCS the device stays in the emulation debug state. If the RESET reset is applied while the device is in the emulation debug state it causes some of the CorePacs to go into an unknown state and they don't start execution.

Resets using POR and RESETFULL do not exhibit this behavior.

This does not affect the normal usage of the device when CCS/emulator is not connected to the device since the device is not in emulation debug state when reset is applied using the RESET pin. This behavior can only happen in the lab environment where the CCS/emulator is connected to the device.

Workaround

Below is the sequence which must be followed to completely disconnect the device from CCS before applying RESET:

1. "Free Run" all the CorePacs
2. Disconnect all the CorePacs from CCS
3. Apply RESET

Steps 1 and 2 insure that all the debug states are cleared in the device. This will allow the CorePacs to wake up correctly on reset via RESET. Bypassing either step 1 or 2 will result in CorePacs that do not begin execution after reset.

KeyStoneII.BTS_errata_usagenote.10

I²C Bus Hang After Master Reset Usage Note

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details It is generally known that the I²C bus can hang if an I²C master is removed from the bus in the middle of a data read. This can occur because the I²C protocol does not mandate a minimum clock rate. Therefore, if a master is reset in the middle of a read while a slave is driving the data line low, the slave will continue driving the data line low while it waits for the next clock edge. This prevents bus masters from initiating transfers. If this condition is detected, the following three steps will clear the bus hang condition:

1. An I²C master must generate up to 9 clock cycles.
2. After each clock cycle, the data pin must be observed to determine whether it has gone high while the clock is high.
3. As soon as the data pin is observed high, the master can initiate a start condition.

KeyStoneII.BTS_errata_usagenote.11

Minimizing Main PLL Jitter Usage Note

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details Once the boot is complete, it is highly recommended that software reconfigure the Main PLL to the desired frequency, even if it is already achieved by the initial settings. To minimize the overall output jitter, the PLLs should be operated as close as possible to the maximum operating frequency. To maximize the VCO frequency within the PLL, the PLL should be clocked to 2× the intended frequency and the PLL Output Divider should be set to /2. The main PLL Output Divider should be set to divide-by-2 by the software by writing 0b0001 to bits [22:19] of the SECCTL register (address 0x02310108) in the PLL controller. A read-modify-write can be used to make sure other bits in the register are not affected. This register is documented in the part-specific data manual.

NOTE: It is only after programming the SECCTL register to enable the divide-by-2 that the following equation can used to program the PLL as specified in the data manual.

CLK = CLKIN × ((PLLM+1) ÷ ((OUTPUT_DIVIDE+1) × (PLLD+1)))

KeyStoneII.BTS_errata_usagenote.13

Packet DMA Does Not Update RX PS Region Location Bit Usage Note

Revision(s) Affected 1.0, 1.1 (Fixed in PG2.0)

Details The Packet DMA inside each of the Navigator-compliant modules fails to update the Protocol-Specific Region Location bit (bit 22 of Packet Descriptor Word 0) to a 1 when it is writing an RX host-mode packet to memory with protocol-specific (PS) words located in the start of the data buffer instead of the descriptor. This means that the software cannot use this bit to determine if any PS information present is located in the RX packet descriptor or at the beginning of the data buffer. The same problem will occur if the packet is sent directly to another Navigator-compliant module, as that module will not be able to determine the PS info location. This issue affects only host-type packets.

Workaround 1: Use monolithic-type packets only, thus eliminating the issue.

Workaround 2: Always place PS info in the descriptor instead of in the data buffer so that the PS location bit is always 0 and the issue does not apply.

Workaround 3: The software is responsible for configuring the Packet DMA RX flow tables, which include the PS location each flow will use. Thus, for packets sent to the DSP (not to another module directly), the software can be designed to keep track of the PS info location so it does not have to rely on the bit in the RX packet descriptor. This can be accomplished in many different ways. The following are a couple of examples:

- The software can always use the same PS location setting. This eliminates the need to find out the location from the RX descriptor.
- The software can place some form of identifier in one of the user-defined tag fields in the TX descriptor and configure the Packet DMA to pass that information through to the RX descriptor. The software can then use the identifier along with previously stored information to determine the PS info location.

KeyStoneII.BTS_errata_usagenote.14

Queue Proxy Access Usage Note

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details When there are multiple DSP cores potentially accessing the Queue Manager, the Queue N register A, B, C, and D should be accessed in the same burst. However, the C66x CorePac cannot generate bursts larger than 8 bytes. The Queue Proxy is designed to allow the C66x CorePac to push/pop descriptors using multiple transactions. However, when the C66x CorePac uses the Queue Proxy region for push and pop, the Queue Proxy may mix the transactions from non-CorePac system masters. This may lead to an error transaction, which causes a system deadlock.

Workaround The C66x CorePac should not use the Queue Proxy region to push/pop descriptors. The C66x CorePac should use VBUSM region (base address starts from 0x34000000) to push descriptors. When Queue N register C is needed for a push, the C66x CorePac should issue a DoubleWord write to generate an 8-byte burst write to Queue N register C and Queue N register D. When device is little endian mode, the Queue N register C and Queue N register D value need to be swapped. The C66x CorePac should use the VBUSP region (base address starts from 0x02A00000) to pop Queue N register D only. When packet size, byte count, and queue size information are needed for a certain queue, C66x CorePac should use the queue peek region to get the information.

KeyStoneII.BTS_errata_usagenote.17

Initial Voltage Level Setting of CVDD Rail Power Supplies Usage Note

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details Users are required to program their board CVDD supply initial value to 1.0 V on the device. The initial CVDD voltage at power-on will be 1.0 V nominal and it must transition to VID set value, immediately after being presented on the VCNTL pins. This is required to maintain full power functionality and reliability targets guaranteed by TI.

SmartReflex voltage scheme as defined by the device specific data manual and *Hardware Design Guide for KeyStone II Devices* is required.

KeyStoneII.BTS_errata_usagenote.18

DEBUGSS CTTBR ID Period Functionality Issue

Revision(s) Affected 1.0, 1.1

Details When more than one trace stream is captured in the CT-TBR module, the trace formatter feature must be enabled. Trace Formatter must be capable of embedding the Trace ID of the captured data along with the data in a formatted frame.

In order for trace decoder to decode such data when the trace capture buffer has wrapped around, the Trace ID of a source has to be regularly seen in the formatted frames. This is to allow the trace decoder to associate the trace data with a particular trace ID.

When the trace ID of a source does not change for a long time, the CT-TBR module supports a function whereby it repeats the Trace ID every nth formatted frame where n is a programmable value. This feature does not work meaning that a trace decoder may not be able to properly associate trace data with a trace source that produced it.

Workaround There is a workaround to enable internal trace source inside the CT-TBR known as the Sequence ID generator. This works because the trace ID seen by the trace formatter changes frequently allow such as trace decoder always can associate trace data with a particular source of the data.

KeyStoneII.BTS_errata_usagenote.19

Boot Mode Change by Writing to DEVSTAT Register Does Not Survive Warm Reset

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details Boot mode cannot be changed by writing to DEVSTAT register.

The current implementation of the bootmode logic does not allow software changes to the bootcfg register (DEVSTAT) to survive a warm reset. Instead during a warm reset, the current logic resets the bootcfg register to the value that was latched during the last power on reset. Therefore, any subsequent software configurations after the last power on reset to the DEVSTAT register will be overwritten by the assertion of warm reset.

Workaround None

KeyStoneII.BTS_errata_usagenote.20

HyperLink 0 and HyperLink1 PRIVIDs Not Generated by the HyperLink IP

Revision(s) Affected 1.0, 1.1

Details Hyperlink0 and Hyperlink1 PRIVIDs are not generated by HyperLink IP (they are supposed to be derived from the transaction address and internal mapping registers). They are instead generated by the programmable PRIVID set from SECMGR. This will impact the way the MPAX, MPUs, security firewalls are programmed for Hyperlink accesses.

In the secure device HyperLink 0 and HyperLink1 PRIVIDs are fixed and driven by SECMGR (0xE is default).

Workaround None

KeyStoneII.BTS_errata_usagenote.21

USB EP15 Not Supported

Revision(s) Affected	1.0, 1.1
Details	The USB EP15 in and out endpoint cannot be used in host or device mode.
Workaround	None

KeyStoneII.BTS_errata_usagenote.22

C66x Boot ROM Does Not Detect a Local Reset Properly

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details When the C66x CorePac is a boot master, the C66x boot ROM does not detect a local reset correctly and it will execute the boot ROM code as if it was a global reset.

Workaround None

KeyStoneII.BTS_errata_usagenote.23

BOOTCOMPLETE Not Functional when ARM CorePac is a Boot Master

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details When ARM CorePac is a boot master, the BOOTCOMPLETE pin will not go high as is expected when all the ARM CorePacs have exited the boot code.

Workaround None

KeyStoneII.BTS_errata_usagenote.24

Clock Alignment Issue When in Reset Isolation Mode

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details When the SOC is supporting SRIO reset-isolation or SRSS reset-isolation and there is a chip reset, the alignment between SRIO, SRSS and the rest of the chip clocks is not guaranteed.

Workaround This issue can be worked around by configuring both SYSCLK1 and SYSCLK2 coming out of the PLLCTRL module as reset isolated sources. If both those clocks are configured as reset isolated, then the device will be able to support reset isolation for both SRIO and SRSS, with the caveat however of additional power consumption during the time the chip is held in reset, because when the SYSCLK clocks are reset isolated they will not default to the bypass clock while reset is applied and instead will keep running at the frequency they were set prior to reset. The SYSCLK1 and SYSCLK2 become reset isolated by writing a '1' to the following PLLCTRL MMR bits: PLLCTRL.RSISO[1:0]. (Refer to the PLL Controller User Guide.)

KeyStoneII.BTS_errata_usagenote.25

USB Hangs When Doing a Master Access to Reserved Space

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

Details When accessing the reserved space using USB AXI on HOST side (by providing the reserved address value to Device Context Base Address Array Pointer) and Host does not provide any interrupt to SW, however it sets the bus_error status flag in USBSTS and GSTS. Hence the USB HOST cannot convey AXI Bus error directly to SW. SW has to make sure that if it does not get the proper response from the USB HOST, it should check the USBSTS/GSTS registers for possible error. On Device side, events generated by USB Device after data transfer, has information about AXI bus error.

Workaround Avoid accessing to reserved space.

KeyStoneII.BTS_errata_usagenote.27

Device Hang if 10GbE PCS Registers are Accessed After Performing a 10G Lane Reset

Revision(s) Affected 1.0, 1.1, 2.0

Problem Summary: The 10GbE Physical Coding Sublayer (PCS) used in the 10GbE interface may hang the device if access is attempted to its memory mapped registers (MMRs) after performing a SerDes lane reset.

Details The PCS registers may be inaccessible after performing a 10G lane reset. Lane resets are not needed after initialization in normal operational mode, but they are used if a lane rate re-negotiation is required. This inaccessibility is rooted in a component within the PCS called the ‘bridge’. This bridge connects the PCS memory-mapped registers (MMRs) to the VBUSP interface. The bridge reset is toggled whenever either of the lane_OK status from the 10G PHY-B SerDes is pulled low. This reset can cause the bridge interface to lock-up and render the PCS registers inaccessible if the bridge interface had been previously accessed. If a core attempts to access the PCS registers while the bridge interface is locked-up, that core will hang.

Workaround No software workaround has been identified for this issue. It is suggested that if a lane reset has been asserted/de-asserted after 10G PHY-B Serdes initialization, that all cores refrain from accessing the PCS register space.

KeyStoneII.BTS_errata_usagenote.28

SerDes Fails to Adapt RX BOOST Equalization

Revision(s) Affected 1.0, 1.1, 2.0, 3.0, 3.1

This problem impacts the following high speed interfaces on all current Keystone-II devices. Refer to the device data manual for applicability. It does not impact 10GbE operation.

Table 13. SerDes peripherals impacted by RX Boost equalization problem

Interface	Data Rate	Notes
AIF2	>4.9Gbps	8b/10b symbols are used during data transport. There is no link training
AIL	>4.9Gbps	8b/10b symbols are used during data transport. There is no link training
Hyperlink	>6.25Gbps	8b/10b symbols are used on Hyperlink only during link training
JESD	>4.9Gbps	8b/10b symbols are used during data transport. There is no link training
PCIe	5Gbps	8b/10b symbols are used during data transport. There is no link training
SRIO	5Gbps	8b/10b symbols are used during data transport. There is no link training

Problem Summary: The SerDes on some peripherals will not automatically adapt its RX equalization when provided with certain data encodings common to that peripheral's electrical standard.

Details The TI SerDes contains an adaptation algorithm that allows the RX equalization blocks to adapt their parameters to the SerDes data channel. The adaptation algorithm sets the optimal values for the ATT, BOOST, and DFE blocks in order to equalize the signal, maximize margin, and minimize channel distortion. For higher data rates (i.e. 5Gbps and greater), the high frequency gain provided by the BOOST block is generally considered necessary to compensate for the low-pass characteristics of data channels and widen the data eye.

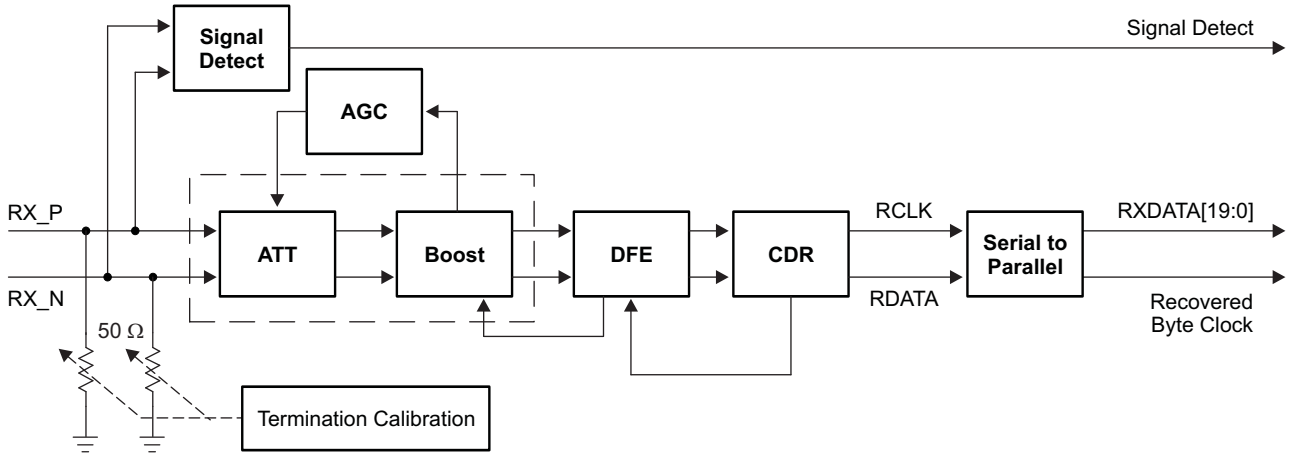


Figure 10. RX Path Block Diagram

The SerDes BOOST block was designed to require a specific set of patterns in order to best adapt its parameters to the channel. If these patterns are not found, the BOOST adaptation algorithm will not update the internal block settings and/or may incorrectly update these settings. The required data pattern is a series of six 0'-s followed by a 101 pattern. This pattern must occur numerous times on both odd and even bit alignments. This data pattern is not present in the 8b/10b coded symbols that are sent over many of the high speed peripheral links.

The data pattern is present in longer length PRBS patterns such as PRBS-23, and PRBS-31 used by the SerDes internal BIST hardware used during BER testing. The data pattern that is required is not present in the symbols used in initial Hyperlink training or in the other high speed SerDes interfaces during data transport.

If the BOOST has not been properly adapted for the received signal, then there is a chance that the SerDes will be unable to recover the RX data or the error rate may be higher than expected.

Workaround

The recommended workaround is to not use the RX equalization auto-adaptation mechanism with the impacted interfaces at higher data rates. As an alternative, a user can hardcode the optimal ATT and BOOST values for their channel.

These optimal ATT and BOOST values must be determined by the user through one of two options:

- Perform BER test with PRBS sequence and sweep across all values of ATT/BOOST while using a fixed set of TX parameters that have already been optimized. Use optimal ATT and BOOST value permutation that has the most margin. This is the value permutation that is both error-free and “farthest” from permutations with errors.

  • The SerDes DIAG tests can be found under *<TI_PDK_INSTALL_DIR>\packages\ti\diag\serdes_diag* in the latest CSL/PDK release and can be configured to perform this test.
- Perform a test where a PRBS data pattern is presented to the RX and the ATT/BOOST are allowed to auto-adapt. This method can also be used to identify the optimal ATT and BOOST value permutation that has the most margin.

The optimal ATT and BOOST values found by one of the two above methods can be hardcoded into the SerDes upon initialization.

Revision History

Changes from June 30, 2017 to June 30, 2018 (from E Revision (June 2017) to F Revision)	Page
• Replaced "66AK2H06/12/14" with "66AK2Hxx."	4

IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265