# AM5K2E04/02
# Multicore ARM KeyStone II SOC
# Silicon Revision 1.0

# Silicon Errata

Literature Number: SPRZ416B
May 2014–Revised August 2015

# Contents

# List of Figures



# List of Tables

# AM5K2E04/02 Multicore ARM KeyStone II SOC Silicon Revision 1.0

This document describes the silicon updates to the functional specifications for the AM5K2E04/02 fixed-/floating-point digital signal processor. 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 AM5K2Ex in the ABD 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 ABD package is shown in Figure 1. The figure also shows an example of AM5K2Ex package symbolization.

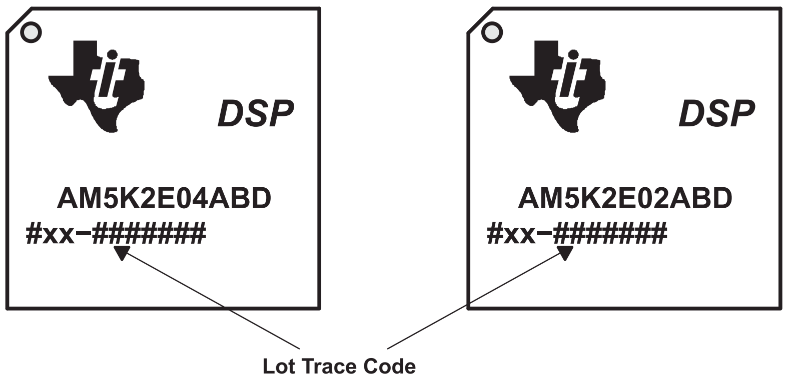



**Figure 1. Lot Trace Code Example for AM5K2E04/02 (ABD Package)**

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 AM5K2Ex devices.

**Table 1. Lot Trace Codes**

| Lot Trace Code (xx) | Silicon Revision | Comments |
|---|---|---|
| 10 | 1.0 | Initial silicon revision |

The AM5K2Ex 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 AM5K2Ex.

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 AM5K2Ex device.

**Table 2. Silicon Revision Variables+**

| Silicon Revision | AM5K2Ex JTAGID Register (address location: 0x0262_0018) |
|---|---|
| 1.0 | 0x0b98_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 AM5K2Ex.

**Table 3. Cortex-A15 Processor Version and REVIDR**

| SoC | A15 Version | ARM REVIDR |
|---|---|---|
| AM5K2E04/02 | r2p4 | 0x020A<br>• REVIDR[1] = 1'b<br>• REVIDR[3] = 1'b<br>• REVIDR[9] = 1'b |

The ARM product revision $r_m p_n$ 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

Table 4 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 Updates**

| Category | Silicon Update Advisory | See [1] | Applies To Silicon Revision 1.0 |
|---|---|---|---|
| NOR XIP | Execution in Place (XIP) from NOR Flash on ARM Does Not Work | Advisory 2 | X |
| RESET | ~~RESETSTAT~~ Signal Driven High | Advisory 15 | X |
| DDR3 | False DDR3 Write ECC Error Reported Under Certain Conditions | Advisory 20 | X |
| PCIe | PCIE MSI/Legacy IRQ Does Not Work for Root Complex | Advisory 22 | X |
| Boot | Boot ROM NAND Cannot Cross Bad Blocks | Advisory 24 | X |
| Boot | ROM Ethernet Boot Failure | Advisory 25 | X |
| PCIe | Descriptors Placed in PCIe Memory Space can Cause Problems | Advisory 28 | X |
| 10GbE | 10GbE PCS Causes Data Corruption | Advisory 29 | X |
| 10GbE | 10GbE PCB Channel Loss Limited to 20dB | Advisory 30 | X |
| HyperLink | HyperLink Channel Loss | Advisory 31 | X |
| HyperLink | HyperLink Data Rate Limited to 40Gbaud Issue | Advisory 33 | X |
| DDR3 | DDR3 Limited to Highest Data Rates Due to Possibility of PLL Instability During Temperature Changes After Startup | Advisory 34 | X |
| DDR3 | Restrictions on DDR3 PLL Configuration And DDR3nCLK to Eliminate DDR3 Errors Due to PLL Dynamic Phase Offset | Advisory 35 | X |
| Boot | ARM Boot Can Fail When Interrupt Enabled | Usage Note 4 | X |

(1) Not all KeyStone II errata apply to all KeyStone II parts. Therefore, numbering gaps in the errata list are normal.

**Table 4. Silicon Revision 1.0 Updates (continued)**

| Category | Silicon Update Advisory | See (1) | Applies To Silicon Revision 1.0 |
|---|---|---|---|
| Boot | Boot $I^2C$ Frequency Incorrect | Usage Note 5 | X |
| DDR3 | Access to DDR3 Without Configuring PHY Properly Can Cause Hang | Usage Note 6 | X |
| Power | Core Wake Up on $\overline{\text{RESET}}$ | Usage Note 9 | X |
| $I^2C$ | $I^2C$ Bus Hang After Master Reset | Usage Note 10 | X |
| PLL | Minimizing Main PLL Jitter | Usage Note 11 | X |
| Power | Initial Voltage Level Setting of CVDD Rail Power Supplies | Usage Note 17 | X |
| HyperLink | HyperLink 0 and HyperLink1 PRIVIDs Not Generated by the HyperLink IP | Usage Note 20 | X |
| USB | USB Hangs When Doing a Master Access to Reserved Space | Usage Note 25 | X |
| 10GbE | Device Hang if 10GE PCS Registers are Accessed After Performing a 10G Lane Reset | Usage Note 27 | X |
| SerDes | SerDes Fails to Adapt RX BOOST Equalization | Usage Note 28 | X |

## Silicon Updates

### KeyStoneII.BTS_errata_advisory.2

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

**Revision(s) Affected** 1.0

**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.15

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

**Revision(s) Affected** 1.0

**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.20

***False DDR3 Write ECC Error Reported Under Certain Conditions***

**Revision(s) Affected** 1.0

**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.

This device 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 5.

**Table 5. 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 5. 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.22

***PCIE MSI/Legacy IRQ Does Not work for Root Complex***

**Revision(s) Affected**

**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.24

***Boot ROM NAND Cannot Cross Bad Blocks***

**Revision(s) Affected** 1.0

**Details**

NAND <ARM> boot fails if boot image crosses a bad block boundary.

During a NAND primary boot, the boot ROM will read and process an image from the NAND device specified by the bootstrap pins. When the boot ROM encounters a pre-marked bad block or detects one through error correction, data processing will reset and invalidate previously read good data. This is opposite U-Boot NAND reads which skip bad blocks and continue reading data.

In other words, if a primary boot image (such as U-Boot) spans multiple blocks with a bad block separating valid data, the boot ROM will not read the complete boot image. Instead it will read only the data following a bad block. Boot will ultimately fail without a complete boot image.

**Table 6. Proper way to write boot image starting with Data 0**

| Block # : | Block 0 | Block 1(Bad) | Block 2 | Block 3 | Block 4 |
|---|---|---|---|---|---|
| Data # : | XXXXX | XXXXX | Data 0 | Data 1 | Data 2 |
| Result : | Skipped | Skipped | ^ Data 0 will be the start of the image. | | |

**Table 7. Do not write boot image this way**

| Block # : | Block 0 | Block 1(Bad) | Block 2 | Block 3 | Block 4 |
|---|---|---|---|---|---|
| Data # : | Data 0 | XXXXX | Data 1 | Data 2 | Data 3 |
| Result : | Thrown Away | Skipped | ^ Data 1 will now be the start of the image. | | |

**Workaround**

If the boot image is large enough to require multiple blocks, locate a sufficient amount of consecutive good blocks to store boot image. Tools such as U-Boot have built-in commands to list bad NAND block as well as the ability to write to a given offset.

This issue does not affect a 2nd stage boot to Linux from U-Boot; so the kernel and filesystem may be written to skip bad blocks as per default behavior. Default U-Boot NAND reads will skip bad blocks and continue reading data.

If NAND boot is used as primary boot to boot U-Boot (or boot image) then, ‘NAND writer utility’ which writes U-Boot (or boot image) to NAND needs to understand this errata and make sure that U-Boot (or boot image) is burned in consecutive blocks with no bad blocks in the middle. In some cases, the ‘NAND writer utility’ could be U-Boot itself then customer U-Boot needs to modify to incorporate this errata workaround.

### KeyStoneII.BTS_errata_advisory.25

***ROM Ethernet Boot Failure***

**Revision(s) Affected** 1.0

**Details**

As per Ethernet boot sequence, the booting device sends BOOTP packets out. In response, the boot master sends the BOOTP response packet. In the failing scenario, the BOOTP packets will be seen exiting the device, but the BOOTP response appears to be ignored by booting device.

The root cause is identified as an uninitialized value in the Ethernet driver in the BootROM code. Packets that arrive to the device are held up in the Ethernet and not passed to the BootROM code for processing. This happens all times when Ethernet boot is selected as primary boot.

This issue is applicable to Ethernet boot for both the C66x CorePac as the boot master and the ARM CorePac as the boot master, please refer to the device specific datasheets for the boot master and boot modes supported.

**Workaround**

The only workaround is to use a different boot mode (other than Ethernet Boot) as the Primary Boot and then perform the Ethernet Boot. The Primary Boot can be used to initialize the above mentioned uninitialized value and then re-enter the Ethernet Boot as Secondary Boot mode.

Step 1. Enable NETCP.

Step 2. Initialize the registers:

```
#define PA_BASE 0x24000000
ROM Errata workaround Function:

memset ((unsigned int *)(PA_BASE + 0x408400), 0, 32 * sizeof(unsigned int));
memset ((unsigned int *)(PA_BASE + 0x418400), 0, 32 * sizeof(unsigned int));

*((unsigned int *)(PA_BASE + 0x409814)) = 0x10;
*((unsigned int *)(PA_BASE + 0x409820)) = 0x4000;
*((unsigned int *)(PA_BASE + 0x409824)) = 0xfffc0000;
*((unsigned int *)(PA_BASE + 0x409810)) = 0x80000600;
```

Step 3. Write the Ethernet boot mode configuration to the DEVSTAT register.

Step 4. Branch to re-enter the ROM boot function.

## KeyStoneII.BTS_errata_advisory.28

***Descriptors Placed in PCIe Memory Space can Cause Problems***

**Revision(s) Affected** 1.0

**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

**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.

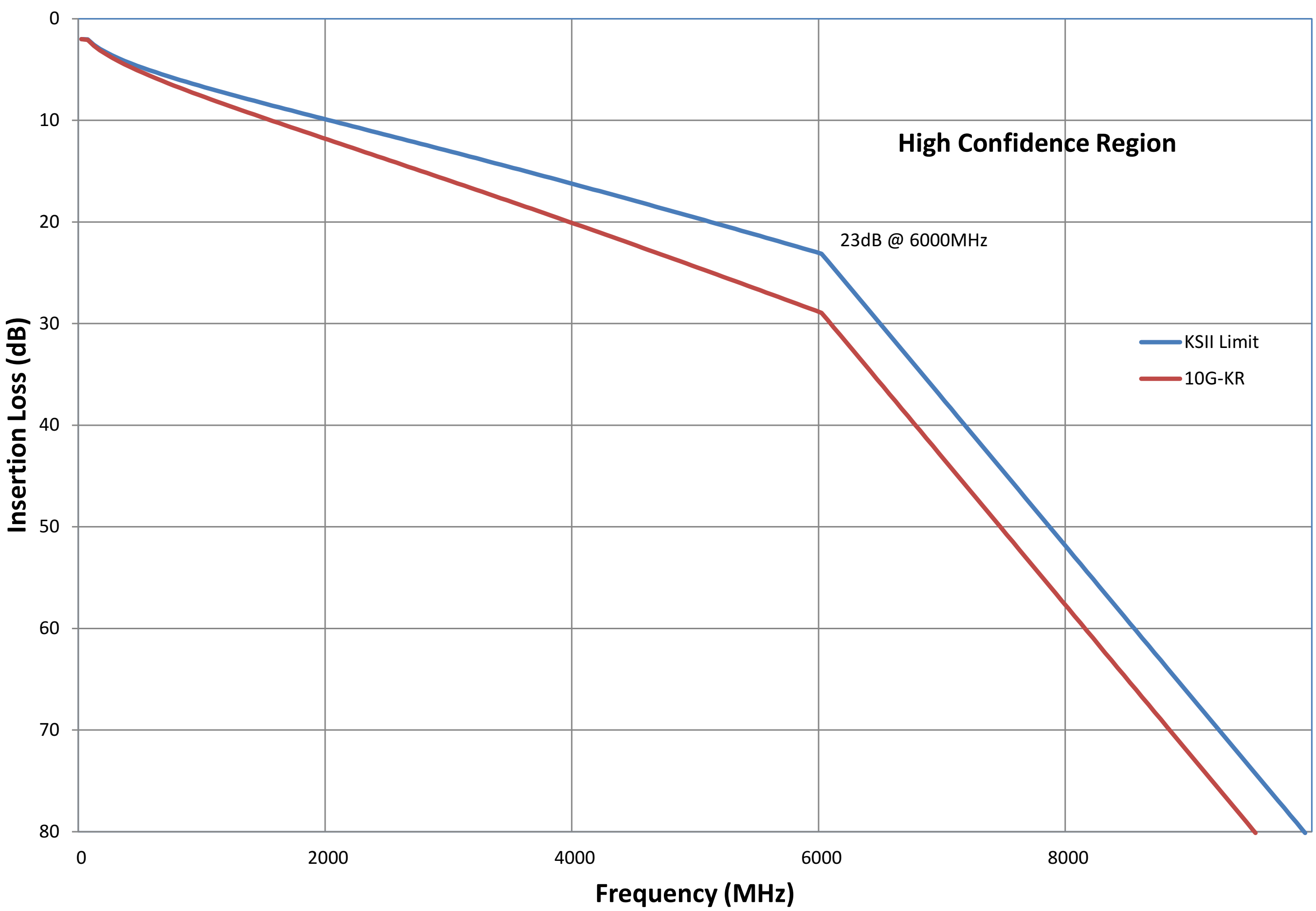
**Workaround** As a workaround, a user can assert and then deassert the corresponding Serdes signal detect. This signal detect reset will force the RX to re-adapt to the incoming data, and will cause an interruption to the datastream to the PCS. This causes the PCS to re-synchronize. This workaround is currently implemented in the latest 10GbE Linux drivers (since MCSDK 3.0.4).

## KeyStoneII.BTS_errata_advisory.30

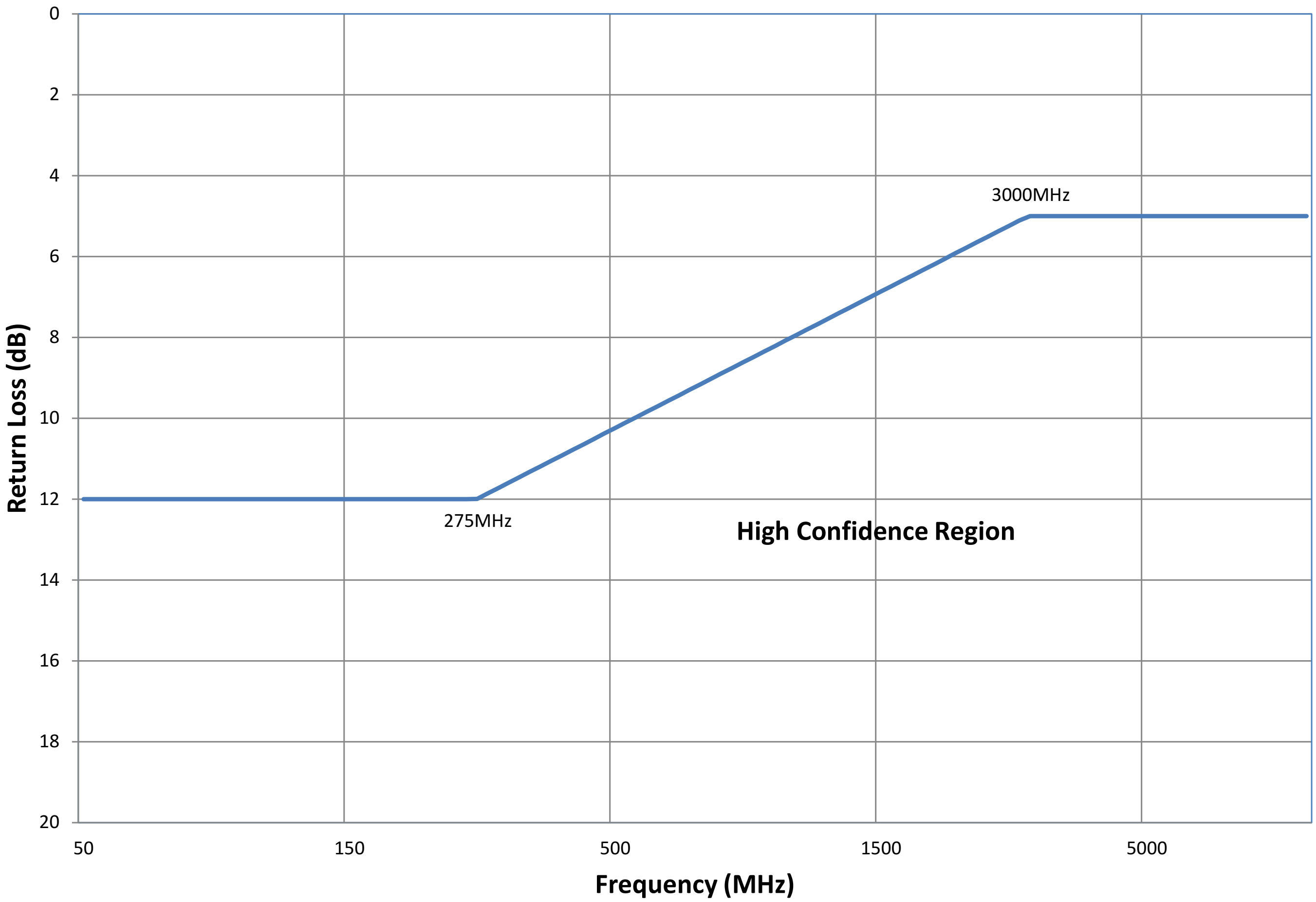
***10GbE PCB Channel Loss Limited to 20dB***

**Revision(s) Affected** 1.0

**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.

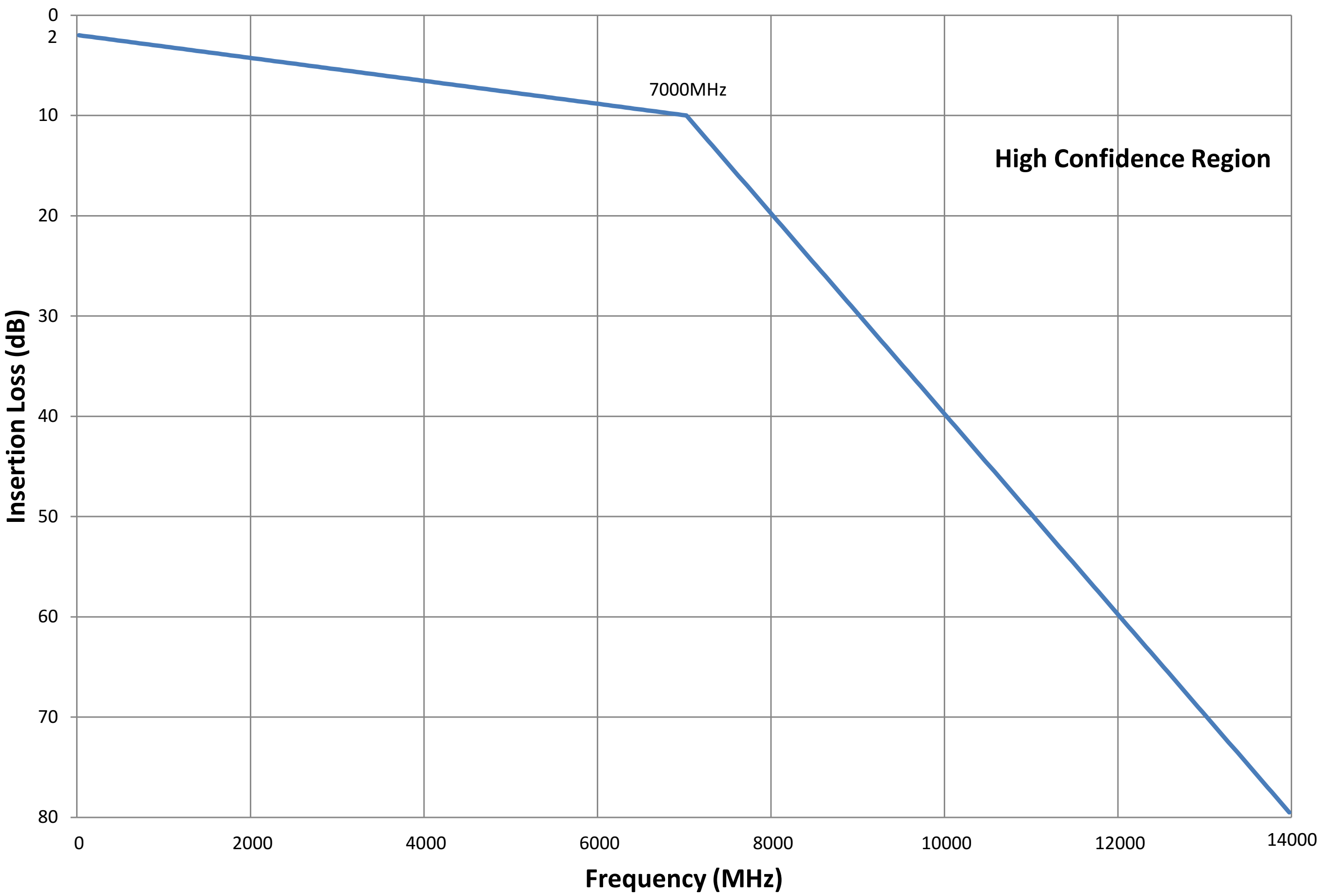
## KeyStoneII.BTS_errata_advisory.31

***HyperLink Channel Loss***

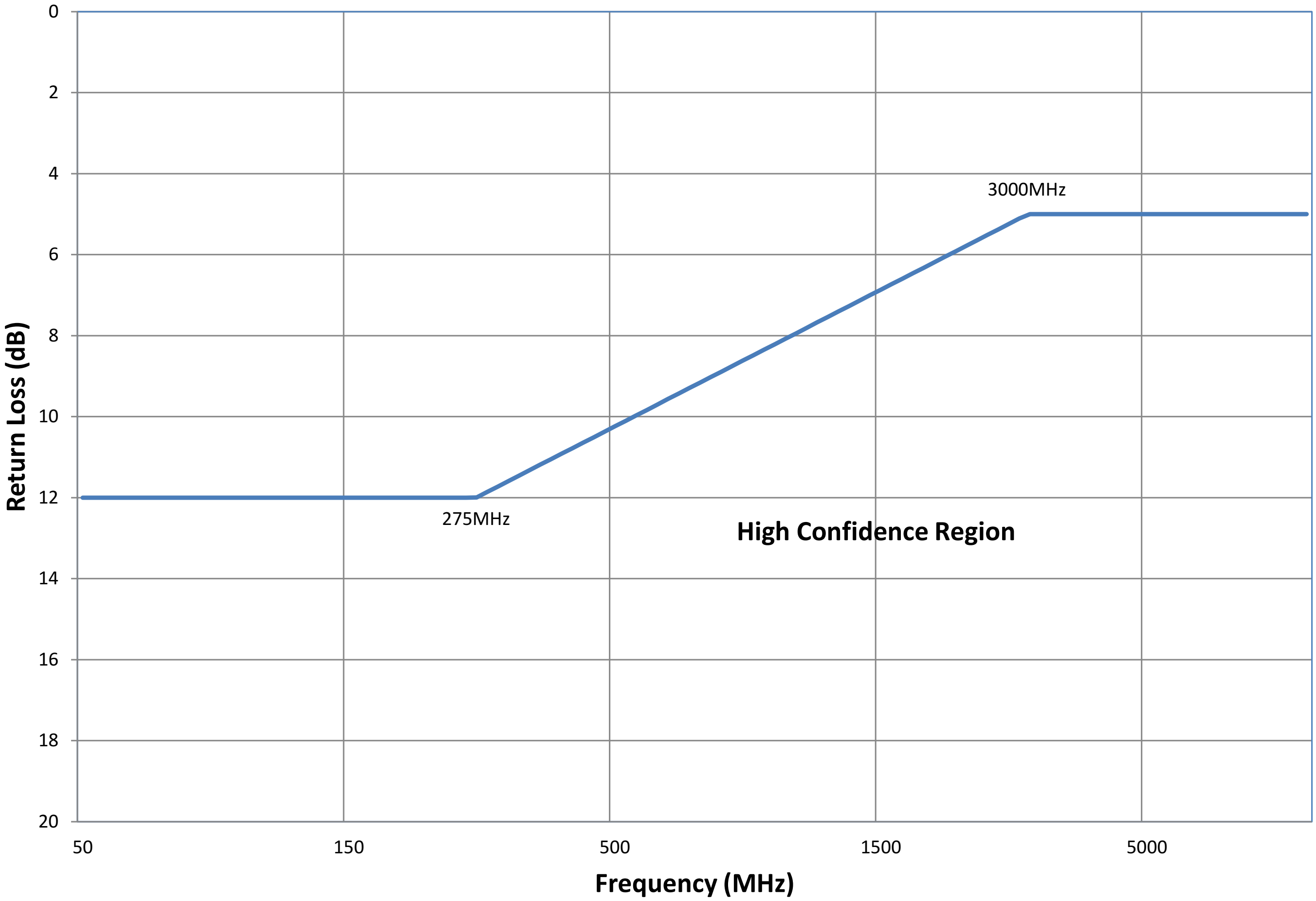
**Revision(s) Affected** 1.0

**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

**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

**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

**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.

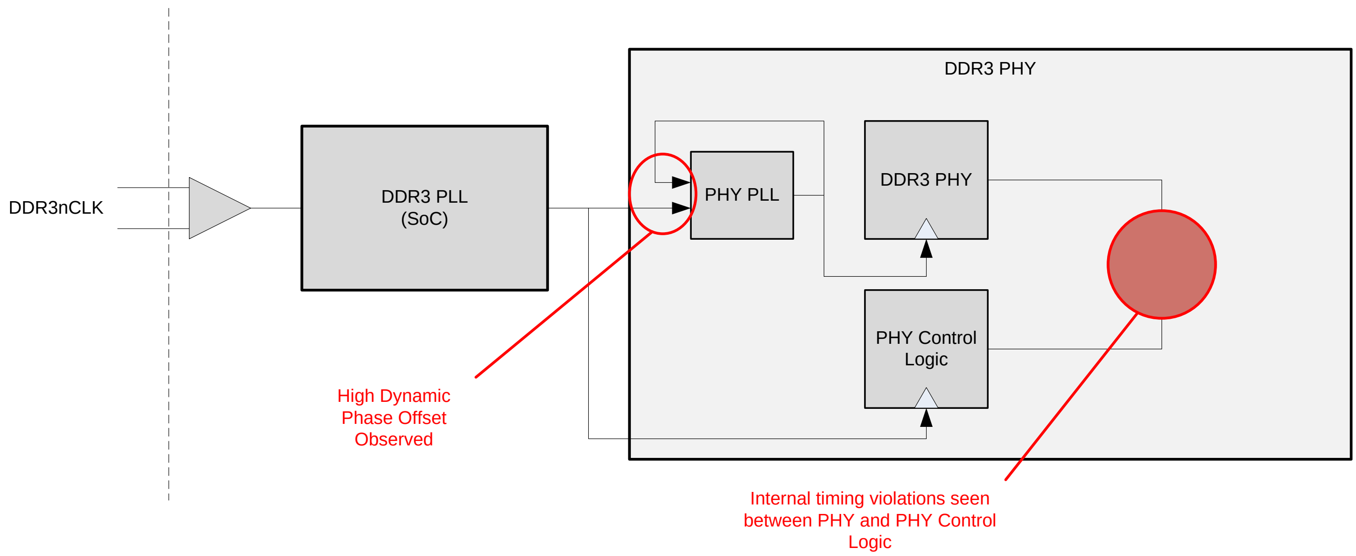



**Figure 2. DDR3 PHY**

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 8. DDR3-1600**

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

**Table 8. 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 9. 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 10. DDR3-1066 [(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 | 40 | DDR3nPLLCTRL0.PLLM | 39 |
| PLL Output Divider | 15 | DDR3nPLLCTRL0.PLLOD | 14 |
| 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 11. 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 | 40 | DDR3nPLLCTRL0.PLLM | 39 |
| PLL Output Divider | 20 | DDR3nPLLCTRL0.PLLOD | 19 |
| 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_usagenote.4

***ARM Boot Can Fail When Interrupt Enabled***

**Revision(s) Affected** 1.0

**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 $I^2C$ Frequency Incorrect***

**Revision(s) Affected** 1.0

**Details** The issue is within the Boot ROM. Initial boot $I^2C$ frequency incorrect on $\overline{\text{RESET}}$ reset.

For $I^2C$ 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 $I^2C$ as the boot master, the boot code will correctly assume the system is running at 312 MHz (max possible) and scale the $I^2C$ 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 $I^2C$ 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 $I^2C$ devices that are compliant to the 100 KHz bus standard specified in the original $I^2C$ 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

**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.9

***Core Wake Up on RESET Usage Note***

**Revision(s) Affected** 1.0

**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

### *$I^2C$ Bus Hang After Master Reset Usage Note*

**Revision(s) Affected** 1.0

**Details** It is generally known that the $I^2C$ bus can hang if an $I^2C$ master is removed from the bus in the middle of a data read. This can occur because the $I^2C$ 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 $I^2C$ 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

**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.17

***Initial Voltage Level Setting of CVDD Rail Power Supplies Usage Note***

**Revision(s) Affected** 1.0

**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.20

***HyperLink 0 and HyperLink1 PRIVIDs Not Generated by the HyperLink IP***

**Revision(s) Affected** 1.0

**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.25

***USB Hangs When Doing a Master Access to Reserved Space***

**Revision(s) Affected** 1.0

**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

**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

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 12. 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.

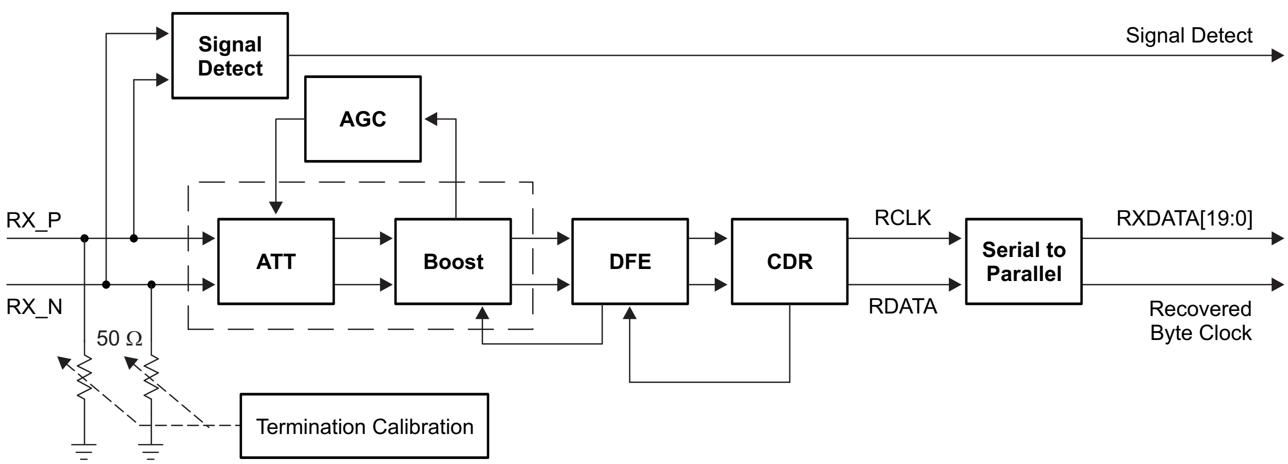



**Figure 3. RX Path Block Diagram**

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 5, 2015 to September 1, 2015 (from A Revision (June 2015) to B Revision)** **Page**

| **Products** | | **Applications** | |
|---|---|---|---|
| Audio | www.ti.com/audio | Automotive and Transportation | www.ti.com/automotive |
| Amplifiers | amplifier.ti.com | Communications and Telecom | www.ti.com/communications |
| Data Converters | dataconverter.ti.com | Computers and Peripherals | www.ti.com/computers |
| DLP® Products | www.dlp.com | Consumer Electronics | www.ti.com/consumer-apps |
| DSP | dsp.ti.com | Energy and Lighting | www.ti.com/energy |
| Clocks and Timers | www.ti.com/clocks | Industrial | www.ti.com/industrial |
| Interface | interface.ti.com | Medical | www.ti.com/medical |
| Logic | logic.ti.com | Security | www.ti.com/security |
| Power Mgmt | power.ti.com | Space, Avionics and Defense | www.ti.com/space-avionics-defense |
| Microcontrollers | microcontroller.ti.com | Video and Imaging | www.ti.com/video |
| RFID | www.ti-rfid.com | | |
| OMAP Applications Processors | www.ti.com/omap | **TI E2E Community** | e2e.ti.com |
| Wireless Connectivity | www.ti.com/wirelessconnectivity | | |


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