Application Note

AM64x USB 3.1 Gen 1 Host TX Electrical Compliance: Spot Check Approach

Luis Parga *Sitara MPU*

ABSTRACT

USB 3.1 Gen 1 transmitter (TX) compliance testing can be a challenging task when having to use complex equipment like a Jitter Bit Error Ratio Tester (J-BERT) to enter the different test modes needed for TX compliance. Fortunately, entering the aforementioned test modes can also be done without a J-BERT by manually changing the necessary compliance registers. Making use of the AM64x USB3.0 subsystem configured in host mode, this document describes the steps to manually enable the different compliance test patterns needed for USB SuperSpeed 5Gbps TX compliance testing when in host mode without the need for a J-BERT. This approach does not intend to replace the official test procedure where J-BERT is used. The intent of this method is to aim the engineer to do a spot check of the IP.

Due to the high data rate of 5 Gigabits per second of the SuperSpeed USB interface and the possibility of interface users having poorly shielded or unshielded external cables, it is mandatory to use Spread Spectrum Clocking (SSC) to distribute the radio frequency (RF) energy over a wider frequency range in order to minimize Electromagnetic Interference (EMI). This document also focuses on showing how to enable SSC for the Phase-Locked Loop (PLL) of the Serializer/Deserializer (SERDES) block for the USB3.0 controller.

Table of Contents

List of Figures

Trademarks

1 Introduction

The main purpose of this document is to show how to manually get the necessary patterns to be transmitted without having to use external test equipment to inject Ping.LFPS to the AM64x USB3.0 receiver (RX) lines to interchange the needed compliance patterns. The method described in this document specifically focuses on USB 3.0 TX compliance making use of the AM64B Starter Kit Evaluation Module (SK EVM) board using the Linux SDK (Version: 12.00.00.07.04) in SD BOOT mode.

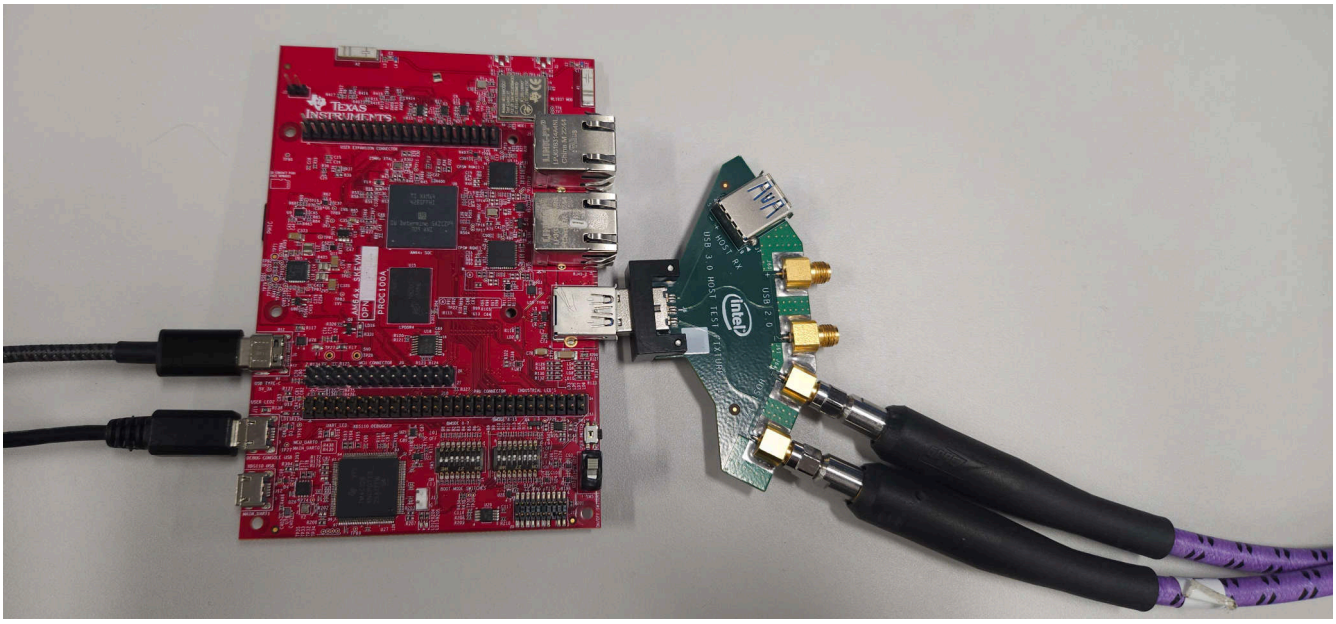

Figure 1-1. AM64B SKEVM with USB SS Test Fixture and SMA Phase-Matched Cable Pair to Oscilloscope

The below equipment is used for the approach discribed in this document:

- AM64B SKEVM with soldered AM64x SoC
- 5V, 3A AC/DC Power Supply (via USB-PD Type-C input) for AM64B SKEVM
- USB Type-C cable
- 2x 50Ω-Termination/ Phase-Matched/ Low-Loss SMA cables with at least 16GHz of bandwidth
- USB 3.0 USB-IF Electrical Test Fixture: USB Host Test Fixture 1 (or USB 3.2 Gen1 Host Test Measurement Fixture)
- High Performance Oscilloscope with at least 16GHz of bandwidth
- USB2.0 Standard-A to Micro-B male-to-male cable

2 SERDES PLL SSC - Linux SDK Configuration

2.1 DTS Configuration for SSC

The AM64x USB3.0 physical interface natively supports SSC, but Linux SDK Version: 12.00.00.07.04 and predecessors do not have SSC enabled by default. This miss is likely due to the need to bypass SERDES PLL SSC for early silicon bring-up activities. In the processor Linux SDK, the integration of the AM64x USB3.0 subsystem is handled via the Device Tree Source (DTS) configuration. To enable SSC for the USB3.0 SERDES PLL on the SK-AM64B Starter Kit board, the k3-am642-sk.dts at the below location needs to be modified.

```
<SDK_INSTALL_DIR>/board-support/ti-linux-kernel-<version>/arch/arm64/boot/dts/ti/k3-am642-sk.dts
```

As per the USB 3.0 specifcation, USB 3.1 Gen 1 SSC requirements specify a modulation rate between 30 and 33 kHz, a triangular-shaped modulation, and a downspread modulation depth between 0.4% and 0.5% as shown in the figure below.

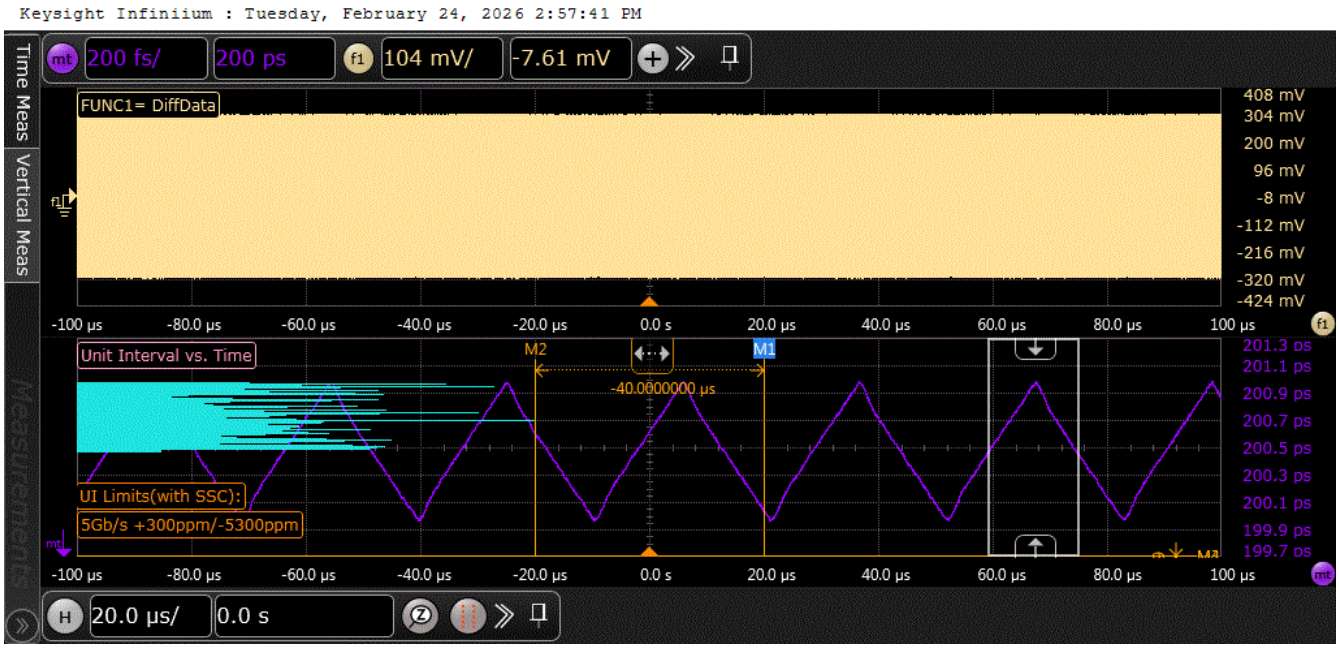

Figure 2-1. 5Gbps Carrier Frequency with SSC Modulation

To implement these changes, the below code needs to be added to the SERDES wizard wrapper (WIZ) node &serdes_wiz0 in the DTS.

```
ti,core-clk-sel = <1>;   /* Select internal ref clock */
ti,ssc-enable;     /* Enable SSC */
ti,ssc-type = <1>;  /* 1 for Downspread */
ti,ssc-frequency-hz = <33000>; /* 33kHz fmod */
ti,ssc-depth-per-mil = <5>; /* 0.5% depth */
```

The below also needs to be added to the serdes0_usb_link: phy@0 node.

```
cdns,ssc-mode = <2>;   /* 2 for SSC enabled (internal) */
```

The below figure shows the updated configuration that enables SSC using triangular-shape SSC modulation profile with down spread modulation output.

```
&serdes_wiz0 {
        bootph-all;

        ti,core-clk-sel = <1>;  /* Select internal reference clock */
        ti,ssc-enable;  /* Enable SSC */
        ti,ssc-type = <1>;          /* 1 for Downspread */
        ti,ssc-frequency-hz = <33000>;
        ti,ssc-depth-per-mil = <5>; /* 0.5% depth */
};

&serdes0 {
        bootph-all;
        serdes0_usb_link: phy@0 {
                bootph-all;
                reg = <0>;
                cdns,num-lanes = <1>;
                #phy-cells = <0>;
                cdns,phy-type = <PHY_TYPE_USB3>;
                resets = <&serdes_wiz0 1>;
                cdns,ssc-mode = <2>; /* value of 2 for SSC enabled (internal) */
        };
};
```

Figure 2-2. DTS Changes to Enable SERDES PLL SSC for USBSS

2.2 DTS Compiling

In order for the updated device tree configuration to be compiled correctly, the kernel needs to be configured with the correct architecture and cross-compiler by running the below commands from the Linux kernel's built-in build system.

```
$ cd <SDK_INSTALL_DIR>/board-support/ti-linux-kernel-<version>/
$ export CROSS_COMPILE=<SDK_INSTALL_DIR>/board-support/ti-linux-kernel-<version>/linux-devkit/
sysroots/x86_64-arago-linux/usr/bin/aarch64-oe-linux/aarch64-oe-linux-
$ export ARCH=arm64
$ make dtbs
```

2.3 Device Tree Blob (DTB) Copy to SKEVM Boot-Up SD Card

After compiling the .dts changes and with the bootup SD card of the AM64B SKEVM mounted to the Linux machine (i.e. Ubuntu) used for SDK installer changes, the k3-am642-sk.dtb file needs to be copied to the /root/boot/dtb/ti/ of the SD card with the following steps.

```
$ cd <SDK_INSTALL_DIR>/board-support/ti-linux-kernel-<version>/arch/arm64/boot/dts/ti/
$ sudo cp k3-am642-sk.dtb /media/<your_username>/root/boot/dtb/ti/k3-am642-sk.dtb
$ sudo eject /media/<your_username>/root/
```

3 Low-Frequency Periodic Signaling (LFPS)

For LFPS tests (shown below), the USB link is initiated as soon as the USB3.0 receiver (RX) detects a 50Ω termination after connecting to the oscilloscope via the USB-IF host test fixture 1. This causes the Port Link State (PLS) of the Port Status and Control (PORTSC) register (0x0F410490) to transition from Rx.Detect (0x0A0002A0) to Polling state (0x0E0002E0). The TX then starts transmitting LFPS bursts continuously. The LFPS pattern is specifically used for the below tests.

1. LFPS Peak-Peak Differential Output Voltage
2. LFPS Period
3. LFPS Burst Width
4. LFPS Repeat Time Interval
5. LFPS Rise Time
6. LFPS Fall Time
7. LFPS Duty Cycle
8. LFPS AC Common Mode Voltage

The very low-frequency bursts of about 16.5MHz with bursts happening approximately every 11µ seconds look like the waveforms in the figure below.

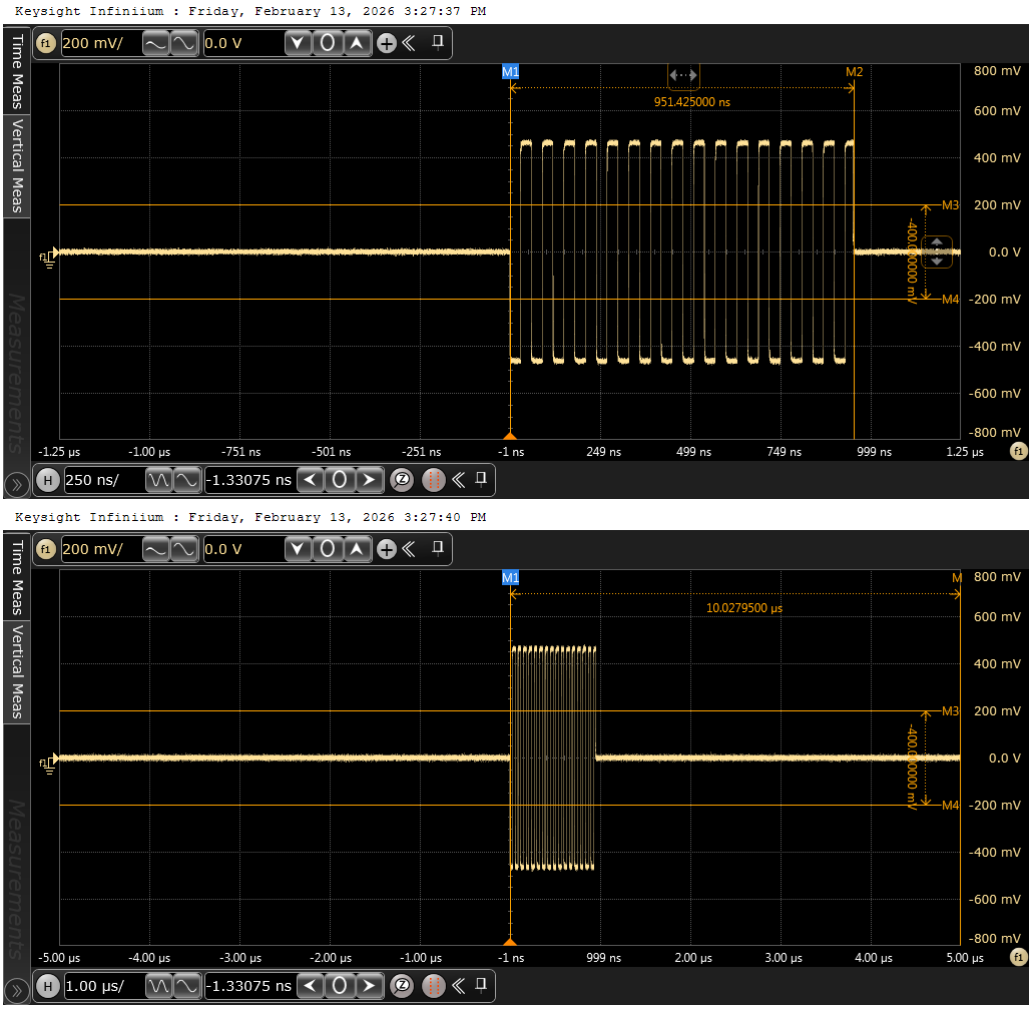

Figure 3-1. LFPS Oscilloscope Waveform

4 CP0 - CP1 Transmission Pattern Setup

The USB3.0 xHCI controller needs to be in compliance mode for all subsequent non-LFPS tests. The Port Link State Write Strobe (LWS) of the PORTSC register needs to first be set to ‘1’ to be able to write to the Port Link State (PLS) bit field of the PORTSC register. The link state can be forced to transition from Rx.Detect into compliance mode after writing to the PLS bit field and when the AM64x USB3.0 controller receiver detects a 50Ω termination. Forcing the PLS into compliance mode causes the Link Training and Status State Machine (LTSSM) to automatically jump to Polling.Compliance. The below command, which needs to be entered before connecting the USB test fixture, sets the LWS to '1' and the PLS to 10 to enable a link transtition to the Compliance state.

```
~# devmem2 0x0F410490 w 0x0A010340
```

The compliance pattern 0 (CP0), which is an 8b/10b scrambled pseudo-random data pattern (PRBS-16), is the default compliance pattern transmitted by the PHY when the USB3.0 xHCI controller transitions to compliance mode. The CP0 pattern looks like the below waveform and is used for many of the eye diagram and jitter tests.

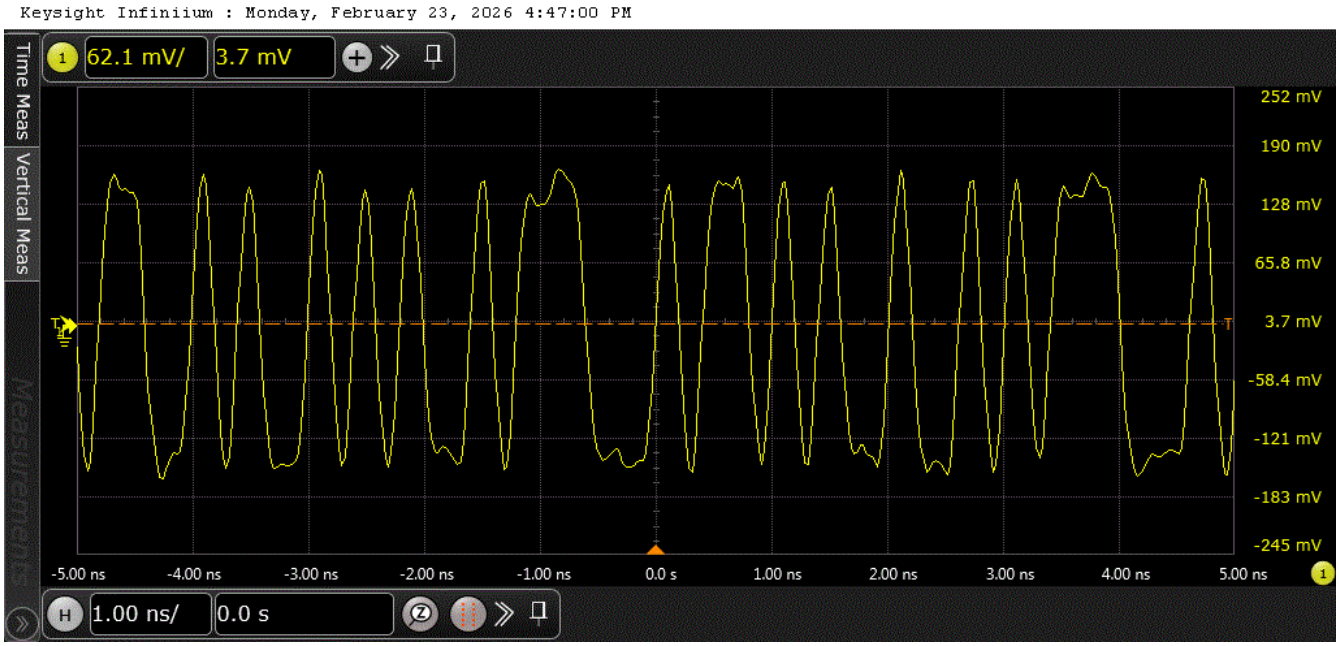

Figure 4-1. CP0 Compliance Pattern - Oscilloscope Waveform

The CP0 pattern is specifically used for the below tests with and without Continuous-Time Linear Equalization (CTLE).

1. 5G Short Channel Maximum Deterministic Jitter
2. 5G Short Channel Template Test
3. 5G Short Channel Differential Output Voltage

The CP1 compliance pattern is a continuous 2.5GHz Nyquist clock signal for the 5Gbps data rate of the USB3.1 Gen 1 controller. This pattern is specifically important to perform the SSC and random jitter PHY compliance tests listed below.

1. 5G TSSC - Minimum Frequency Deviation
2. 5G TSSC - Maximum Frequency Deviation
3. 5G SSC Modulation Rate
4. 5G SSC Slew Rate
5. 5G Short Channel Random Jitter (RJ)

CP0 is what the USB3.0 xHCI host controller transmits by default after entering compliance mode (Polling.Compliance). To have the TX pipeline transmit CP1 without the need of an external instrument injecting a Ping.LFPS packet for a CP0 to CP1 transition, the SuperSpeed Port Link Control register of the xHCI host

controller needs to be modified by changing bits 24:21 from 4'b0000 to 4'b0001. The 2 below register writes need to be done after having booted up Linux on the AM64B SKEVM and before connecting the USB SS test fixture. The 2nd command forces CP1 to be used when entering compliance mode.

```
~# devmem2 0x0F410490 w 0x0A010340
~# devmem2 0x0F41812C w 0x18230C10
```

The CP1 2.5GHz clock signal pattern looks like the below figure.

Figure 4-2. CP1 Compliance Pattern - Oscilloscope Waveform

5 Summary

This paper's main goal is to show the main steps to manually configure the AM64x USB3.0 susbsystem compliance patterns needed for host mode TX compliance. These manual changes allow for spot checking of the USB SS IP and prevent having to use a cumbersome piece of external equipment like a J-BERT or a signal generator. If the Linux SDK version being used for compliance testing checks does not already have SSC enabled by default for the USB3.0 SERDES PLL, this document also shows how to properly reconfigure and build the device tree to incorporate SSC. Note that this approach does not aim to replace the official TX Compliance testing approach where a J-BERT or arbitrary waveform generator is used. This approach is simply intended to be used as a high level check.

6 References

- Texas Instruments, *AM64x/AM243x Processors Silicon Revision 2.0 Texas Instruments Families of Products*, technical reference manual.
- USB, *USB3.1 Specification*, webpage.
- USB xHCI Specification (Section 6): *Xtensible Host Controller Interface for Universal Serial Bus (xHCI)*, specifications.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you fully indemnify TI and its representatives against any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale, TI’s General Quality Guidelines, or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products. Unless TI explicitly designates a product as custom or customer-specified, TI products are standard, catalog, general purpose devices.

TI objects to and rejects any additional or different terms you may propose.

Last updated 10/2025