IDT® Tsi578
Design Notes

April 11, 2014

GENERAL DISCLAIMER

Integrated Device Technology, Inc. ("IDT") reserves the right to make changes to its products or specifications at any time, without notice, in order to improve design or performance. IDT does not assume responsibility for use of any circuitry described herein other than the circuitry embodied in an IDT product. Disclosure of the information herein does not convey a license or any other right, by implication or otherwise, in any patent, trademark, or other intellectual property right of IDT. IDT products may contain errata which can affect product performance to a minor or immaterial degree. Current characterized errata will be made available upon request. Items identified herein as "reserved" or "undefined" are reserved for future definition. IDT does not assume responsibility for conflicts or incompatibilities arising from the future definition of such items. IDT products have not been designed, tested, or manufactured for use in, and thus are not warranted for, applications where the failure, malfunction, or any inaccuracy in the application carries a risk of death, serious bodily injury, or damage to tangible property. Code examples provided herein by IDT are for illustrative purposes only and should not be relied upon for developing applications. Any use of such code examples shall be at the user's sole risk.

The IDT logo is registered to Integrated Device Technology, Inc. IDT and CPS are trademarks of Integrated Device Technology, Inc.

About this document

This document describes design notes for the Tsi578. Design notes are unique functional characteristics of the device that may or may not be described in the *Tsi578 User Manual* and should be reviewed when designing with the Tsi578. This document should also be used in with the *Tsi578 Device Errata* document.

Revision History

April 11, 2014, Formal Status

Added "Packet discard on link partner failure"

July 2009, Formal Status

This is the current version of the document. It has been updated to reflect IDT formatting. There have been no technical changes.

May 2009, Formal Status

The following design notes were added to this document:

- "Multi-master clock generation error" on page 6
- "Link start-up error indications" on page 6
- "Assertion of PORT_ERR on an even numbered port" on page 7

Design Notes

1. Port power-down and default configuration

When a port is powered down, the port loses the port write destinationID that is stored for that particular port. Multicast settings, port write settings and other non-port specific registers return to their default power up settings after a port reset. After port reset, there is no way to determine that the configuration for a particular port is correct.

For example, if a port is shut down and then restored, the port write destination ID is reset to zero. The port write destination ID must be reset for the whole device after a port has been shut down and restored. Similarly, multicast settings for the entire device must be re-written when a port has been shut down and restored.

For more information on port power down registers that return to their default settings, refer to the *Tsi578 User Manual*.

2. JTAG device identification numbers

The JTAG device identification numbers have been modified to identify the versions of the Tsi578. The Z1 identification number is 0x00573167, the Z2 is 0x10573167, and production number is 0x20573167.

3. Four 1x links to 4x port training

Connecting four 1x links to a 4x port is not supported and may result in false lane alignment.The Tsi578 correctly detects alignment characters during the 4x training process. However, connecting a four lane 1x port to the Tsi578 while the Tsi578 is expecting to train with a 4x link partner causes the Tsi578 to remain in a PORT_UNINIT state and to never complete the training process in 1x mode.

This situation occurs because the Tsi578 is expecting align characters to be inserted on the four lanes by the link partner in order to complete the receive FIFO skew alignment process. Since the link partner is sending 1x links, it never inserts the align characters required to assist in the completion of the 4x link training process and since all four lanes are active, it never downgrades to 1x mode.

4. Default port transmit electrical characteristics may not be optimal

The Tsi578's default transmit amplitude and pre-emphasis settings may not be optimal in all applications, depending on the trace losses on the printed wiring board. This may result in a degradation in the Bit Error Rate (BER).

The BER can be improved by overwriting the power-on default values of the SRIO MAC x SerDes Configuration Global register to increase the output amplitude and apply more pre-emphasis to the output signal.

5. Masterless I^2C bus busy

This design note applies only to designs that require the Tsi578 to load registers from an I^2C EEPROM.

Because EEPROM devices do not have reset pins, if the Tsi578 is reset the EEPROM is unaffected and can continue to drive data. If the EEPROM continues to drive the I^2C data signal to 0, the Tsi578 is not able to load register values after reset is removed. Unexpected operation after a reset can result if register values cannot be loaded.

The *I^2C Specification (*for multiple master support) specifies that the I^2C bus is considered busy when the I^2C data signal is 0.

Hardware Work Around

To avoid this condition, design the reset of the Tsi578, and all other I^2C masters, so that the I^2C bus is always idle before asserting reset for the Tsi578 or any I^2C bus master.

Software Work Around

To implement software work around, complete the following steps:

1. Determine that the I^{2}C bus is busy while there is no master. To do so, read the registers in Table 1; the register values must match those values described in the table.

Table 1: Register Values to Diagnose Masterless I^{2}C Bus Busy

Register Name	Register Offset	Register Value Descriptions
I^{2}C Interrupt Status Register	1D11C	BL_OK and BL_FAIL bits are both 0
I^{2}C Event and Event Snapshot Registers	1D300	0x00001F00 ANDed with the register value = 0
Internal I^{2}C Status Register 1	1D3D0	0x0000000F ANDed with the register value = 0x0000000B
Internal I^{2}C Status Register 2	1D3D4	Register Value = 0x0000021
Internal I^{2}C Status Register 2, read 200 microseconds later	1D3D4	Register Value = 0x0000020
Internal I^{2}C Status Register 3	1D3D8	0x00000E00 ANDed with the register value = 0x00000600

2. Issue a reset on the I^{2}C bus. Driving nine I^{2}C clock cycles completes an interrupted transfer. An I^{2}C clock cycle occurs whenever the I^{2}C clock is driven low for at least five microseconds, and then is released to be high.

Multiple I^{2}C EEPROM devices document driving nine I^{2}C clock cycles for reset.

The register accesses listed in Table 2 drives I^{2}C clock cycles to complete the interrupted I^{2}C bus transfer. The sequence of writes in the table must be repeated nine times.

Table 2: Creating an I^{2}C Bus Reset

Register Name	Register Offset	Register Value
Internal I^{2}C Control Register	1D3C0	Write 0x00000008, wait five microseconds.
Internal I^{2}C Control Register	1D3C0	Write 0x00000000, wait five microseconds.

3. Trigger a reset of the Tsi578 to perform the register value loading. There are a number of different methods to reset the Tsi578 documented in the *Tsi578 User Manual*. It is also possible for the host processor to reset the Tsi578 by system specific means.

To implement the software work around, the Tsi578 must be configured to allow host processor access after reset using the power-up configuration pins.

For more information, refer to the *Tsi578 Hardware Manual*.

Testing

The software solution can be tested using the IDT JTAG Register Access Software (available at www.idt.com). This software includes scripts which recreate the Masterless I^2C Bus Busy condition.

6. DONE bit incorrectly cleared

Writing 0 to the SEND bit in the SPx_SEND_MCS register clears the DONE bit. The correct behavior is that the DONE bit should retain its current value of 1.

To avoid incorrectly clearing the DONE bit, do not write to this register with the SEND bit set to 0.

7. Multi-master clock generation error

When the Tsi578's I^2C Interface is in a multi-master system, the I^2C Interface does not generate a correct clock low period. The error can occur when all the following conditions are met:

- Both the external master and the I^2C Interface are generating the clock
- The external master pulls the clock low two reference clock cycles before the I^2C clock high timer expires

These conditions are possible only when an external master illegally attempts to use the bus when the Tsi578 is the bus master. In an expected configuration, two masters are unlikely to be operating at the same time. As well, experiencing this issue requires precise timing between the two masters. Because of these factors, this situation is not likely to occur in a system.

This issue does not occur when theTsi578's I^2C Interface is the only master.

8. Link start-up error indications

The Tsi578 can indicate errors on the port, in addition to a PORT_OK status, when the Tsi578 is in a 1x mode configuration and is connected to a 4x mode link partner. Other errors may be indicated in addition to, but not limited to, PORT_ERR and IMP_SPEC_ERR. This is an expected and normal behavior.

During the link initialization process, the 4x mode link partner expects a signal on all four of its receivers. The Tsi578, however, only generates a signal on one lane. After the discovery timer times out without a signal on all four lanes, the port enters the silent state and returns to begin the 1x mode link initialization process. The link initialization then completes successfully in a 1x mode configuration with both link partners indicating a PORT_OK status.

At the start of the link initialization, the Tsi578 registered the fact that there was a signal present during the first attempt at achieving a link, then the signal disappeared, then returned. The disappearance of the signal during the silent period imposed by the 4x mode link partner causes the assertion of the PORT_ERR. The assertion of IMP_SPEC_ERR is a consequence of the assertion of PORT_ERR.

The assertion of PORT_ERR and IMP_SPEC ERR do not inhibit the exchange of packets once PORT_OK is achieved. Writing a 1 to the PORT_ERR bit in the RIO Port x Error and Status CSR, and writing a 0 to the IMP_SPEC_ERR bit in the RIO Port x Error Detect CSR, clears the error indications.

9. Assertion of PORT_ERR on an even numbered port

When a Tsi578 port is configured to operate as two 1x mode ports, an error incurred by the odd numbered port that causes a PORT_ERR indication on the odd numbered port also causes the assertion of the PORT_ERR bit in the even numbered port. This is also the case when the even numbered status is indicated as PORT_UNINIT.

The causes of PORT_ERR on all four lanes of the MAC of the port are OR'd together to assert the PORT_ERR bit in the Rio Port x Error Detect CSR of the even numbered port. The assertion of the signals that make up the even numbered port's PORT_ERR bit are qualified with the achievement of 10b synchronization on the lanes and not with the achievement of a PORT_OK status condition on the port.

The assertion of PORT_ERR on the even numbered port due to an error on the odd numbered port does not inhibit synchronization, link initialization, and subsequent packet traffic flow on the even numbered port. The PORT_ERR bit can be cleared by writing a 0 to the bit in the Rio Port x Error Detect CSR.

The cause of the PORT_ERR bit assertion on the odd numbered port must be dealt with in the normal procedure of Serial RapidIO error management to determine, and then rectify, the cause for the assertion.

If the following occurs (on either the both the even and odd ports) the odd port's PORT_ERR is asserted:

- The PORT_ERR_EN bit in the SP_CTL_INDEP register is set
- And the PW_DIS bit in the SPx_MODE CSR are cleared

When the odd port's PORT_ERR is asserted, both ports perform the following:

- PORT_WRITE packets are generated
- The PORT_W_PEND bits in the corresponding SPx_ERR_STATUS CSR are set

These actions result in the receipt by the host of two PORT_WRITE packets, a legitimate one from the odd port and a spurious one from the even port.

10. Packet discard on link partner failure

The Tsi578 was designed to allow systems to continue operating when a link partner has been reset or otherwise failed. The device must perform two actions to allow a system to continue to operate:

- Notify the system host that a link partner has failed
- Discard packets destined for the failed link partner

Port-writes, triggered when the ERR_RATE_CNT bit in the RapidIO Port x Error Rate CSR exceeds the ERR_RFT bit threshold in the RapidIO Port x Error Rate Threshold CSR, should be used to notify the system host that a failure has occurred.

Two discard mechanisms should be used:

- Set the DROP_EN and STOP_FAIL_EN bits in the RapidIO Port x Control CSR to discard packets when ERR_RATE_CNT in the RapidIO Port x Error Rate CSR exceeds the ERR_RFT threshold in the RapidIO Port x Error Rate Threshold CSR, and the port is not in output error-stopped state. This is known as the “standard” discard mechanism.
- Enable the Dead Link Timer with the minimum time value in order to discard packets until the link has reinitialized.

Usually, systems follow a “fail stop” philosophy. Once a fault is detected on a link, all traffic destined for the link must be discarded until software recovers the link. Packet discard due to the Dead Link Timer will cease once the link has reinitialized. If the link partner has failed only temporarily, or the link has reinitialized due to a high temporary bit error rate, the standard discard mechanism will operate after the link has reinitialized. However, if the link partner was reset, the port will detect an ackID mismatch, enter output error-stopped state, and will be unable to discard packets.

Systems that must support link partner reset must set the PORT_LOCKOUT bit in the RapidIO Serial Port x Control CSR before the link reinitializes in order to ensure that packet discard continues. The implication is that software must be capable of receiving a port-write, processing the port write, and setting the PORT_LOCKOUT bit in less than 80 microseconds.

Note that the board reset controller could support holding devices in reset for a period long enough to ensure that software can set the PORT_LOCKOUT bit.