Notes

This document is intended to reflect some of the design considerations that need to be applied when designing a system based on the RC32334. This document is intended to record subtle behaviors of the RC32334 that should be considered early in the design process to avoid lengthy debug time.

Document Revision History

April 20, 2000: First version of the document.

July 19, 2000: Added three sections: Byte and Halfword PCI Read Accesses, PCI Hold Time, and High Latency Devices on PCI Bus.

November 17, 2003: Added UART Mode.

June 1, 2004: Added Arbiter Idle Grant Mode.

Board Reset from EJTAG Probe

The EJTAG specification requires the RST* pin on the EJTAG header to reset the board. Many ICE probes use the RST* pin (which is open drain output) to sense whether power is applied to the probe. When the EJTAG probe initiates a board reset by bringing RST* low, the signal is driven low solidly.

However, at the end of the reset procedure the EJTAG probe cannot drive the RST* signal back high because this pin is an open drain output. The EJTAG probe must rely on a pullup resistor supplied by the board.

Even with a resistor as small as 1000 ohms, it takes a full 100ns to pull this signal high even when it is only driving one load. If that single load is an EPLD, the signal tends to oscillate a bit as it crosses the Vih threshold for the EPLD. This causes the EPLD to register the deassertion and reassertion of the RST* signal multiple times. Because all of the IDT parts require continuous reset pulses in the order of 100ms or greater, this oscillation causes the part to fail reset in an unpredictable way.

Listed below are two methods to avoid this problem.

1. The reset signal from the ICE probe can be routed through the power management IC. Generally, these ICs contain an input which initiates a reset of the circuit. The pulse width of this reset depends on the resistor/capacitor combination attached to the IC.

2. If the signal goes into an EPLD, the assertion of RST* can be used to initiate a cold reset of the processor and board circuit. But when RST* deasserts, a counter with a count length of greater than two microseconds should be used before deasserting the cold reset to the processor and board.

The S334 board uses the second method.

Operation of UART in Polled Transmit Mode

The RC32334 provides 2 UARTs which are designed to be compatible with both the 16450 and the 16550. The 16550 is identical to the 16450 except that the 16550 provides a 16 byte FIFO on both receive and transmit sides.

UARTs for the 16550 are enabled by setting bit 0 of the Buffer Control Register, BCR[0].

There are two modes in which users can program the UARTs:

- *Interrupt driven mode, when it is supported by the UART controller and the board design*
- *Polled mode.*

The RC32334 UARTs behave in compliance with the 16550 specifications in the interrupt driven mode.

FIFOs were introduced in the 16550 to enable the 16 bytes of FIFOs to be filled (transmit) or emptied (receive) in a single execution of the interrupt handler, thereby reducing the load on the CPU. However, these FIFOs can be used in polled mode, although the benefit is less compared to the interrupt driven mode.

Typically, in the polled transmit mode, the Line Status Register (LSR) checks to see if it is appropriate for the software to write the next byte to the UART or the FIFO. The 16550 specifications state that LSR[5] can be guaranteed to be 1 when the Transmit Holding Register (THR) is empty, and LSR[5] can be guaranteed to be a 0 when THR is not empty. A non-empty THR implies at least one byte in the FIFO buffer. Therefore, writing a single byte to the transmit FIFO ought to result in LSR[5] returning a 0. This does not happen on the RC32334.

IDT recommends two possible procedures for ensuring the current operation of the RC32334 when it is used in polled transmit mode:

- Test LSR[6] bit instead of LSR[5] bit. LSR[6] bit, when set to 1 by the controller, indicates to the user that the transmit buffer as well as the THR is empty. This will allow the user to transmit one byte at a time in the polled mode.
- Set the DMA mode in the Buffer Control Register, BCR[3], to 1. This will activate the TXRDY interrupt signal when the transmit FIFO buffer is completely empty and will deactivate the TXRDY signal when the buffer is completely full. The state of the TXRDY signal can be probed by software through the Expansion Interrupt Controller. Register Group 5 deals with UART channel 0 and Group 6 deals with UART channel 1. TXRDY Interrupt State can be read through the “Interrupt Pending Register” corresponding to the UART channel under consideration. Once a completely empty buffer condition is sensed by polling the Interrupt Pending Register, up to 16 bytes can be written to the transmit FIFO in a single attempt without worrying about the level of fullness of the buffer, etc.

Byte and Halfword PCI Read Accesses

When operating as the PCI master, the RC32334 treats all PCI read accesses to the PCI memory space as 32-bit accesses, and it will drive all four byte enable signals active, regardless of whether the read was for an 8-bit, 16-bit, or 32-bit access. The RC32334 will then pick the 8-bit, 16-bit, or 32-bit value it wants out of the 32-bit value that is returned. However, when accessing the PCI I/O space, the byte enables will be driven on an individual byte basis during reads.

Therefore, if the RC32334 is to be connected to a target device that requires individual byte enables to be driven during 8-bit or 16-bit reads, that device must be mapped into the PCI I/O space. This usually applies to target devices that do not support speculative reads from memory. Speculative reads means that performing a read does not alter the data stored in the location.

PCI Hold Time

The following pins in revisions ZA and ZB of the RC32334 require a hold time of 1 ns instead of 0 ns as specified in PCI Specifications 2.1

pci_ad[31:0]	pci_cbe_n[3:0]	pci_par	pci_frame_n	pci_trdy_n	pci_irdy_n
pci_stop_n	pci_idsel_n	pci_perr_n	pci_serr_n	pci_rst_n	pci_devsel_n
pci_req_n[2]	pci_req_n[1]	pci_req_n[0]	pci_gnt_n[0]	pci_inta_n	pci_lock_n

The PCI Specification requires a minimum tval of 2 ns, providing the necessary 1 ns hold time.

High Latency Devices on PCI Bus

In order to accommodate devices with a high latency on the PCI bus, three timers internal to the RC32334 must be managed correctly.

These timers are:

- The CPU BusTimeout register (1800_0740 - 1800_07048) (refer to Chapter 16 in the RC32334 User Reference Manual).
- The IP BusTimeout register (1800_0750 - 1800_0758) (also Chapter 16 of the manual).
- The PCI TRDY register (40h in the PCI configuration register block) (refer to Chapter 12 of the manual).

The CPU BusTimout, the IP BusTimeout, and the PCI TRDY registers must all be programmed with values large enough to keep them from timing-out while waiting for the slow PCI device to respond. Otherwise, a timeout interrupt will be generated and the system will jump into the interrupt handler.

The CPU BusTimeout register and the IP BusTimeout register decrement at the system clock frequency, while the TRDY register decrements at the PCI clock frequency.

Clock Implementation for RC32364 and RC3233x

It is recommended to slightly delay the RC32364 clock with respect to the RC32134 clock by a small value [<0.5 nscc]. This is required to give extra hold time for the control signals and data from the RC32364 to the RC32134. This can be accomplished by using a larger clock time to the RC32364 compared to the RC32134.

UART Mode

Do not change between UART 16450 and 16550 modes (bit [0] of the FIFO control register) and then flush the FIFOs while there are characters still in the FIFO TX buffer waiting to be sent. This may result in a UART transmit status malfunction. If this happens, the TE and THR bits in the UARTxLS register may be permanently cleared until the interface is reset. Characters can otherwise still be transmitted and the other status signals will continue to function.

Arbiter Idle Grant Mode

The Arbiter Idle Grant Mode feature in the PCI Subsystem is designed to optimize PCI bus utilization. This feature is enabled by setting the Arbiter Idle Grant bit in the PCI Arbitration register. When the Arbiter Idle Grant Mode is enabled, the internal arbiter withdraws a Grant during idle cycles and grants the bus to another Master requesting the bus. When it is not enabled, the internal arbiter will either have to wait for the current master to withdraw its request or wait until a PCI Arbiter timeout occurs before it grants the bus to another Master. A PCI Arbiter timeout will occur when 16 PCI clock cycles have elapsed and the current Master (which was granted the bus) has not asserted a PCI FRAME signal.

However, prior to leveraging the Idle Grant Mode for efficient bus grants, it is important to consider the following design aspects. The PCI subsystem uses address stepping for configuration Write/ Read cycles. During address stepping, the address is placed on the PCI bus prior to the PCI FRAME being asserted to allow the address to propagate to all the peripherals. Therefore, during address stepping, PCI cycles are considered to be idle by the internal arbiter. For a heavily loaded bus, there is a chance that the internal arbiter may grant the bus to another Master during address stepping, thereby disallowing Host configuration Read/ Write commands to complete. If this event happens frequently, it will lead to the queuing of configuration commands which will eventually result in an IP Bus timeout.

When an IP Bus timeout occurs, the command FIFO is flushed and all the pending commands are discarded without completing. This behavior is undesirable and should be avoided. However, the likelihood of such behavior is very rare because configuration Read/ Write operations are performed primarily during initialization. Even if a system needs to issue configuration Read/ Write commands after initialization, such cycles are not frequent and the probability of having another Master requesting the bus more than once during the address stepping operation is low.

Nonetheless, the following steps should be taken as a preventive measure:

1. Increase the IP Bus Timeout value to allow more time for the pending configuration commands to finish. The IP Bus Timeout value can be increased by changing the value in the IP Bus Timeout Compare register.

2. Disable the Idle Grant Mode. This step is recommended if the PCI bus is heavily loaded (qualified by the likelihood of more than one Master requesting the bus for most of the time) and the system needs to perform configuration Read/ Writes after initialization.