PCI DMA Core

FPGA Test Plan

Example Only

Company:	
Contact:	

Author:	
Date:	
Revision:	1.0

Revision History

Revision	Date	Author	Description
1.0		PLP	First Draft

Applicable Documents

Table of Contents

1 Introduction

This document describes the architecture of Testbench and provides a detailed test plan.
All DMA tests include these conditions:

- Minimum data length transfer
- Maximum data length transfer
- PCI bus DWD/DWOD termination at any phase
- Interrupt functions and their mask logics
- PCI master abort
- PCI target abort
- DMA abort or DMA stop
- Tx DATA FIFO full and Rx DATA FIFO empty
- Tx and Rx DMAs work at same time

The tests are organized in 7 parts:

- PCI initialization: test PCI configuration registers, and initialize them for following tests.
  This section tests will cover design specifiction section 5.1.1
  see section 3.1 for more detail

- Register test: test FPGA registers. It is to verify register reset default value and general writing/reading operations.
  This section tests will cover design specification section 5.1.2, 5..1.3 and 5.1.4
  see section 3.2 for more detail

- Linear Tx DMA test: test FPGA working in Linear Tx DMA mode.
  This section tests will cover design specification section 5.2.1 and 5.2.2.1
  See section 3.3 for more detail

- Linear Rx DMA test: test FPGA working in Linear Rx DMA mode.
  This section tests will cover design specification section 5.2.1 and 5.2.3.1
  See section 3.4 for more detail

2 General Description

Testbench consists of

1. FPGA (PCI DMA Core) :
   Unit Under Test

2. PCI host
   Simulate PCI host to initiate register operations and interrupt control
   PCI memory1 (32 bit): BAR0 of host target
   PCI memory2 (64 bit): BAR1 of host target
   FIFO control : in Tx DMA tests, it pops data out of Tx DATA FIFO and verify them
   In Rx DMA tests, it pushes data into Rx DATA FIFO

3. Test Control :
   Set up test commands

4. PCI arbiter :
   Arbitrate the requests for PCI bus

Figure 1 shows a block diagram of the testbench.

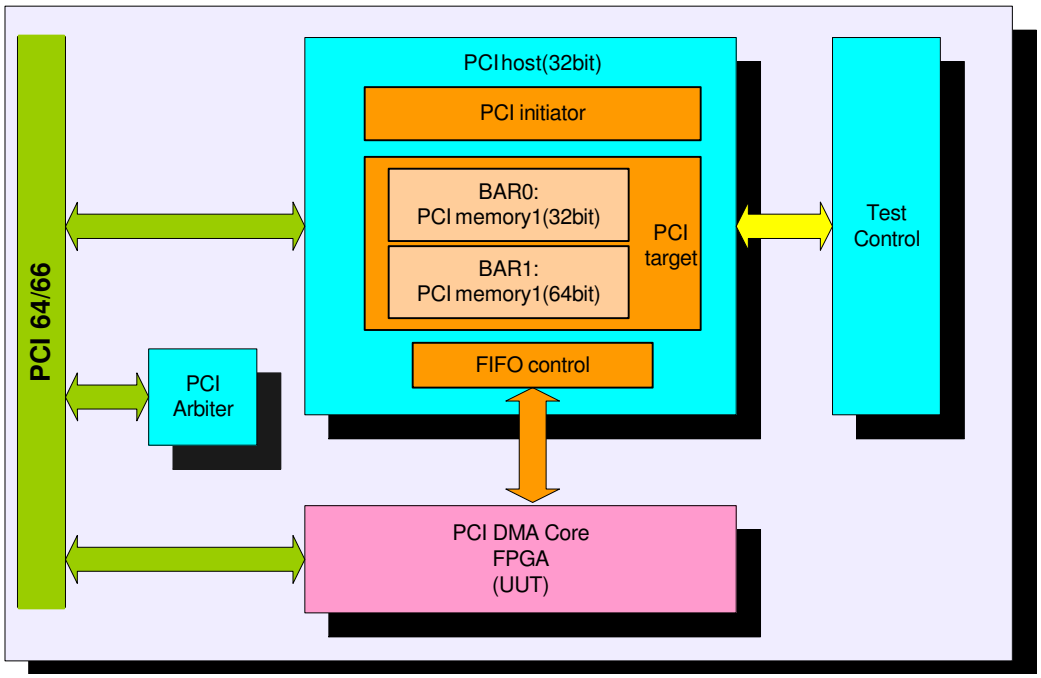

Figure 1 Testbench Architecture

3 Test Plan

3.1 PCI initialization

This section will test:

- PCI core configuration registers logics
  - BAR registers (BAR0 enabled, all other BARs disabled)
  - Status register
  - Command register
- Config PCI core for following testing
  - Set up BAR0, command, Latency Timer registers

Test steps:

a) Read Status Register
b) Set-up Command Register in configuration register space
   - Enable memory access
   - Enable bus mastering
c) Confirm Command Register correctly programmed
   - Read Back Command Register
   - verify register contents. If error, stop
d) Clear status register in configuration register space
e) For all base address registers, BARs
   a. Determine BAR size
   b. Verify if correct size. If error, stop
   c. Program base address
f) Set-up Latency Timer in configuration register space
g) Read back and verify register contents. If error, stop

3.2 Registers test

This section will test:

- registers startup default values
- registers reading/writing logics
- read only and reserved fields or registers

Test steps:

Step b) to e) for testing register 0x00, 0x04, 0x08, 0x24, 0x28, 0x84 only

a) read registers, verify startup default values
b) write different data to different registers
c) read back registers and verify.
d) write inverted value of data to registers
e) read them back and verify.

step f) and g) for testing registers 0x0c, 0x2c

f) set ‘stop’ and ‘DMA64’, read back and verify

g) set 'start' and 'abort', read back and verify
step h) for testing read only and reserved registers
h) write a data to all read only or reserved registers, then read back and verify
i) initialize all registers to 0 for future tests

3.3 Linear Tx DMA tests

3.3.1 Normal condition, minimum data length, interrupt enabled (32bit)

This section will test:

- Linear Tx DMA engine in 32bit mode
- DMA and PCI initiator termination logics in single dword data transfer
- Tx DMA interrupt logics (without interrupt mask)
- Tx DATA FIFO flags and data logics
- Tx Linear DMA start in critical condition

Test steps:

a) Set TX_DMA_ADDR to point to PCI host memory1 (BAR0)
b) Set TX_DMA_LEN = 1
c) Enable Tx_DMA interrupt, mask other interrupts
d) Set TX_DMA_CTRL to start DMA (DMA64 = 0)
e) Wait for interrupt
f) Check Tx DMA registers and interrupt registers
g) Clear interrupt, double check interrupt registers
h) Pop data out from Tx FIFO, verify data, test done

3.3.2 Normal condition, minimum data length, interrupt masked (64bit)

This section will test:

- Linear Tx DMA engine in 64bit mode
- DMA and PCI initiator termination logics in single qword data transfer
- Tx DMA interrupt logics (with interrupt mask)
- Tx DATA FIFO flags and data logics
- Tx Linear DMA start in critical condition

Test steps:

a) Set TX_DMA_ADDR to point to PCI host memory2 (BAR1)
b) Set TX_DMA_LEN = 1
c) Mask all interrupts
d) Set TX_DMA_CTRL to start DMA (DMA64 = 1)
e) Wait until bit Tx_DMA_INT of Current Interrupt Sources is set
f) Check PCI interrupt pin and Interrupt Cause register, check Tx DMA registers
g) Clear all interrupts, double check interrupt registers
h) Pop data out from Tx FIFO, verify data, test done

3.3.3 Normal condition, maximum data length (32 bit)

This section will test:

- Initiator and Linear Tx DMA logics under condition of Tx FIFO full
- Initiator pause and re-request logics (caused by Tx FIFO full)
- Tx FIFO flags and full/empty protection logics
- Tx DMA data length in non-qword-aligned
- Use 2K+3bytes (instead of actual maximum bytes) to test FIFO full

Test steps:

a) Set TX_DMA_ADDR to point to PCI host memory1 (BAR0)
b) Set TX_DMA_LEN = 803h
c) Enable Tx_DMA interrupt, mask other interrupts
d) Set TX_DMA_CTRL to start DMA (DMA64 = 0)
e) Wait until Tx FIFO is almost full, begin pop data from TxFIFO
f) Wait for Tx interrupt
g) Check Tx DMA registers and inerrupt registers
h) Clear interrupt, double check interrupt registers
i) Pop all data out from Tx FIFO, verify data, test done

3.3.4 PCI DWD/DWOD conditions, maximum length (64bit)

This section will test:

- Initiator re-request logics (caused by host target DWD/DWOD)
- Re-request for last qword (split it into two dword transfer)
- PCI DWD/DWOD/RETRY logics in PCI core and initiator
- Use 2K+3bytes (instead of actual maximum bytes) to test FIFO full

Test steps:

a) Set TX_DMA_ADDR to point to PCI host memory2 (BAR1)
b) Set TX_DMA_LEN = 803h
c) Enable Tx Interrupts, mask other interrupts
d) Set PCI host target to terminate on DWD/DWOD/RETRY conditions
e) Start Tx DMA (DMA64 = 1)
f) Wait until Tx FIFO is almost full, begin pop data from TxFIFO
g) Wait for Tx interrupt
h) Check Tx DMA registers and interrupt registers
i) Clear interrupt, double check interrupt registers
j) Pop all data out from Tx FIFO, verify data, test done

3.3.5 Abort DMA test

This section will test:

- Tx Linear DMA abort logic: reset Tx Linear DMA engine and Tx FIFO
- PCI initiator and PCI core on Tx DMA abort: should be safely reset to idle state

- Tx Linear DMA status and FIFO flags

Test steps:
a) set TX_DMA_ADDR to point to PCI host memory2 (BAR1)
b) set Tx_DMA_LEN = 803h
c) Start Tx DMA (DMA64 =1)
d) After data transfer start, write Tx abort
e) Check DMA status register, clear abort, test done

3.3.6 Stop DMA test

This section will test:
- Tx Linear DMA stop logic: see Tx Linear DMA flow chart in design spec 5.2.2.1
- Tx Linear DMA status : check Tx DMA pending status
- Check Tx FIFO flags : no reset on Tx FIFO
- Check Tx DMA interrupt: no interrupt if stop before data transfer done

Test steps:
a) set TX_DMA_ADDR to point to PCI host memory1 (BAR0)
b) set TX_DMA_LEN = 200h
c) start Tx DMA(DMA64=1)
d) write Tx stop, check DMA status registers (DMA pending)
e) wait until Tx DMA active bit in Tx_DMA_STAT register goes low
f) check Tx DMA status register and interrupt registers
g) clear stop, pop data out from Tx FIFO, verify data, test done

3.3.7 PCI Target Abort test

This section will test:
- PCI core target abort logic
- Linear Tx DMA and PCI core error handling on target abort
- Target abort interrupt logic including interrupt mask and clearing logics
- Linear Tx DMA, TxFIFO states and flags: reset Tx FIFO

Test steps:
a) set TX_DMA_ADDR register to PCI host memory2 (BAR1)
b) set Tx_DMA_LEN = 803h
c) Enable Tabort Interrupt, mask other registers
d) Set PCI memory2 to act Target Abort response to any access
e) Start Tx DMA (DMA64 = 1)
f) Wait for Interrupt
g) check Tx DMA registers and interrupt registers
h) Clear interrupts, and double Tx DMA registers and check interrupt registers
i) Mask all interrupts
j) Start Tx DMA (DMA64 = 1)

k) Wait until TABORT_INT bit of Current Interrupt is set
l) check PCI interrupt pins and all interrupt registers, check Tx DMA registers
m) clear interrupt, double check interrupt registers, test done

3.3.8 PCI Master Abort test

This section will test:

- PCI core Master abort logic
- Linear Tx DMA and initiator error handling on master abort
- Master abort interrupt logic including interrupt mask and clearing logics
- Linear Tx DMA, TxFIFO states and flags: reset Tx FIFO

Test steps:

a) set an invalid address to TX_DMA_ADDR register
b) set Tx_DMA_LEN = 1
c) Enable Mabort Interrupt, mask other interrupts
d) Start Tx DMA (DMA64=1)
e) Wait for Interrupt
f) check Tx DMA registers and interrupt registers
g) Clear interrupt, double check Tx DMA registers and interrupt registers
h) Maks all interrupts
i) Start Tx DMA (DMA64=0)
j) Wait until MABORT_INT bit in Current Interrupt is set
k) Check PCI interrupt pins and all interrupt registers, check Tx DMA registers
l) clear interrupts, double check interrupt registers, test done

3.4 Linear Rx DMA tests

3.4.1 Normal condition, minimum data length, interrupt enabled (32bit)

This section will test:

- Linear Rx DMA engine in 32bit mode
- DMA and PCI initiator termination logics in single dword data transfer
- Rx DMA interrupt logics (without interrupt mask)
- Rx DATA FIFO flags and data logics
- Rx Linear DMA start in critical condition (less than Rx FIFO half full)
- Data ready after DMA start (when DMA starts, FIFO is empty)

Test steps:

a) set RX_DMA_ADDR to point to Rx DMA buffer in PCI memory1
b) set RX_DMA_LEN = 1
c) Enable Rx_DMA interrupt, mask other interrupts
d) set RX_DMA_CTRL to start DMA (DMA64 = 0)
e) push one data into Rx FIFO

f) wait for interrupt
g) check Rx DMA registers and interrupt registers
h) clear interrupts, double check interrupt registers
i) verify received data, test done

3.4.2 Normal condition, minimum data length,interrupt masked (64bit)

This section will test:

- Linear Rx DMA engine in 64bit mode
- DMA and PCI initiator termination logics in single qword data transfer
- Rx DMA interrupt logics (with interrupt mask)
- Rx DATA FIFO flags and data logics
- Rx Linear DMA start in critical condition (less than Rx FIFO half full)
- Data ready before DMA start (when DMA starts, FIFO is not empty)

Test steps:

a) set RX_DMA_ADDR to point to Rx DMA buffer in PCI memory2
b) set RX_DMA_LEN = 1
c) mask all interrupts
d) push one data into Rx FIFO
e) set RX_DMA_CTRL to start DMA (DMA64 = 1)
f) wait until bit Rx_DMA_INT of Current Interrupt Sources is set
g) check PCI interrupt pin and Interrupt Cause register, check Rx DMA registers
h) clear interrupts, double check interrupt registers
i) verify received data, test done

3.4.3 Normal condition, maximum data length (32 bit)

This section will test:

- Initiator and Linear Rx DMA logics under condition of Rx FIFO empty
- Initiator re-request logics (caused by Rx FIFO empty)
- Rx FIFO flags and full/empty protection logics
- Rx DMA data length in non-qword-aligned
- Data ready after DMA start (when DMA starts, FIFO is empty)
- Use 2K+5bytes (instead of actual maximum bytes) to test FIFO full

Test steps:

a) set RX_DMA_ADDR to point to Rx DMA buffer in PCI memory1
b) set RX_DMA_LEN = 805h
c) Enable Rx_DMA interrupt, mask other interrupts
d) set RX_DMA_CTRL to start DMA (DMA64 = 0)
e) push half number of data into Rx FIFO
f) wait until Rx FIFO is empty, push remainning data into Rx FIFO
g) wait for Rx interrupt

h) check Rx DMA registers and interrupt registers
i) clear interrupt, double check interrupt registers
j) verify received data, test done

3.4.4 PCI DWD/DWOD conditions, maximum data length (64bit)

This section will test:

- Initiator re-request logics (caused by host target DWD/DWOD)
- Re-request for last qword (split it into two dword transfer)
- PCI DWD/DWOD/RETRY logics in PCI core and initiator
- Data ready before DMA start (when DMA starts, FIFO is not empty)
- Use 2K+5bytes (instead of actual maximum bytes) to test FIFO full

Test steps:

a) set RX_DMA_ADDR to point to Rx DMA buffer in PCI memory2
b) set RX_DMA_LEN = 805h
c) Enable Rx Interrupts, mask other interrupts
d) set PCI host target to terminate on DWD/DWOD/RETRY conditions
e) Start Rx DMA (DMA64 = 1)
f) Push data into Rx FIFO if it is not full
g) wait for Rx interrupt
h) check Rx DMA registers and interrupt registers
i) clear interrupts, double check interrupt registers
j) verify received data, test done

3.4.5 Abort DMA test

This section will test:

- Rx Linear DMA abort logic: reset Rx Linear DMA engine and Rx FIFO
- PCI initiator and PCI core on Rx DMA abort: should be safely reset to idle state
- Rx Linear DMA status and FIFO flags

Test steps:

a) set RX_DMA_ADDR to point to Rx DMA buffer in PCI memory2
b) set Rx_DMA_LEN = 805h
c) push data into Rx FIFO till it is full
d) Start Rx DMA (DMA64 =1)
e) Write Rx abort
f) Check DMA status register, clear abort, test done

3.4.6 Stop DMA test

This section will test:

- Rx Linear DMA stop logic: see Rx Linear DMA flow chart in design spec 5.2.3.1

- Rx Linear DMA status : check Rx DMA pending status
- Check Rx FIFO flags : no reset on Rx FIFO
- Check Rx DMA interrupt: no interrupt if stop before data transfer done

Test steps:

a) set RX_DMA_ADDR to point to Rx DMA buffer in PCI memory1
b) set RX_DMA_LEN = 200h
c) push 64 qwords (=200h bytes) data in Rx FIFO
d) start Rx DMA(DMA64=1)
e) write Rx stop, check DMA status register (DMA pending)
f) wait until Rx DMA active bit in Rx_DMA_STAT register goes low
g) check Rx DMA registers and interrupt registers
h) clear stop, verify received data, test done

3.4.7 PCI Target Abort test

This section will test:

- PCI core target abort logic
- Linear Rx DMA and PCI core error handling on target abort
- Target abort interrupt logic including interrupt mask and clearing logics
- Linear Rx DMA, RxFIFO states and flags: reset Rx FIFO

Test steps:

a) set RX_DMA_ADDR register to Rx DMA buffer in PCI memory2
b) set Rx_DMA_LEN = 400h
c) Enable Tabort Interrupt, mask other registers
d) Set PCI memory2 to act Target Abort response to any access
e) Push data into Rx FIFO, start Rx DMA (DMA64 = 1)
f) Wait for Interrupt
g) check Rx DMA registers and interrupt registers
h) Clear interrupts, double check Rx DMA registers and interrupt registers
i) Mask all interrupts
j) Push data into Rx FIFO, start Rx DMA (DMA64 = 1)
k) Wait until TABORT_INT bit of Current Interrupt is set
l) check PCI interrupt pins and all interrupt registers, check Rx DMA registers
m) clear interrupts, couble check interrupt registers, test done

3.4.8 PCI Master Abort test

This section will test:

- PCI core Master abort logic
- Linear Rx DMA and initiator error handling on master abort
- Master abort interrupt logic including interrupt mask and clearing logics
- Linear Rx DMA, RxFIFO states and flags: reset Rx FIFO

Test steps:

a) set an invalid address to RX_DMA_ADDR register
b) set Rx_DMA_LEN = 1
c) Enable Mabort Interrupt, mask other interrupts
d) Push data into Rx FIFO, start Rx DMA (DMA64=1)
e) Wait for Interrupt
f) Check Rx DMA registers and interrupt registers
g) Clear interrupt, double check Rx DMA registers and interrupt registers
h) Maks all interrupts
i) Push data into Rx FIFO, start Rx DMA (DMA64=0)
j) Wait until MABORT_INT bit in Current Interrupt is set
k) Check PCI interrupt pins and all interrupt registers, check Rx DMA registers
l) clear interrupt, double check interrupt registers, test done

3.4.9 Both Tx and Rx DMA test

This section will test:

- DMA arbiter logic

Test steps:

a) Set Tx and Rx DMA LEN = 20h
b) Push 128 qwords into Rx FIFO
c) Set both DMA_1st_ADDR registers, start DMAs 16 times
d) After all DMAs finished, check both DMA status
e) Verify received data in Rx DMA buffer
f) Pop data out of Tx FIFO and verify them
g) Clear interrupts, test done