Host Interface for Streaming Applications in a Xilinx Virtex-II Pro FPGA

Vinay Vaddepalli

vinay@ittc.ku.edu
Overview

- Introduction
- Issues
- Design & Architecture
- Resources used
- Verification
- Summary
- Questions
Introduction

• Evolution of Main memory
  • Capacity
  • Latency
  • Bandwidth
• Mitigating handshaking overhead
• FERP (Full Empty Register Pipe) is a very small component
• Standard target for RCADE
  • Developed at Clemson University
Issues

• On-Chip Peripheral Bus vs Processor Local Bus
  • Bus width
  • Read / Write lines
  • Speed

• Global State Machine vs Local State Machine
  • Rugged
  • Component updates
Design & Architecture

- Slave on the Processor Local Bus
- Connects to the PLB using the Xilinx PLB IPIF (IP Interface)
Design & Architecture

• Support for variable no. of data streams
  - NUM_DATA_STREAMS VHDL generic
  - Max of eight streams

• Eight data streams
  - Eight input slave registers
  - Eight output slave registers

• One slave register acts as a status register
Design & Architecture
Design & Architecture

• FIFO Stream
  • Instantiated in the *user_logic.vhd* file using a for-generate construct
Design & Architecture

• FIFO Stream
  • Component view
Design & Architecture

• FIFO Stream

  • Input FIFO
    - Synchronous 511x32 FIFO
    - 511x36 FIFO is described in Xilinx App 258
    - Uses 512x36 Block RAM
      - Synchronous dual-ported
    - full/empty status lines
    - fifo_count line
    - Burst capable
Design & Architecture

- FIFO Stream
  - Input FIFO
Design & Architecture

• FIFO Stream

  • Input FIFO

    – Write State Machine
      - PLB pulls `slv_write_enable` high
      - This signal triggers the State Machine
      - Slave registers contents are presented to the FIFO
      - The State Machine can accept a data element every clock cycle
Design & Architecture

• FIFO Stream
  • Input FERP
    - Full Empty Register Pipe
    - Developed at Clemson University
    - Local state machine
    - Two element FIFO
    - Simultaneous read and write
    - Width is parameterized
Design & Architecture

- FIFO Stream
  - Input FERP
Design & Architecture

- FIFO Stream
  - Input FERP
    - Write State Machine
    - Check if FIFO is not empty
    - Check if FERP is not full
    - Read data from FIFO
    - Write to the FERP
Design & Architecture

• FIFO Stream
  • Adder
    - Place holder
    - Interfaces to the FERPs
    - Adds '5' to the input data element
    - Logic that implements the arithmetic
    - State machine to read from a FERP
    - State machine to write to a FERP
Design & Architecture

- FIFO Stream
  - Adder
Design & Architecture

• FIFO Stream
  • Output FERP
    - Similar to the input FERP
    - Interfaces the Adder to the FIFO
    - Adder checks if the FERP is not full
    - Adder writes data to the FERP
Design & Architecture

• FIFO Stream
  • Output FIFO
    - Similar to the input FIFO
    - Write State Machine
      - Check FERP for data
      - Check FIFO for space
      - Reads data from the FERP
      - Writes to the FIFO
Design & Architecture

• FIFO Stream

• Output FIFO
  - Read State Machine
    - Puts data in the output slave register
    - PLB read request to the slave is the trigger
    - Burst capable
Design & Architecture

• FIFO Stream
  • Output FIFO
    – Read Acknowledge
      – PLB after issuing the read request, waits for a read acknowledge
      – FIFO has a read latency of '1' clock cycle
      – PLB read request is delayed by '1' clock cycle to generate the acknowledge
Design & Architecture

• Host Interface

• Status Register
  - Slave device
  - User must monitor the FIFOs
  - Input FIFO `fifocount` less than 31
  - Input FIFO `fifocount` greater than 448
  - Output FIFO `fifocount` greater than 448
Resources Used

- Host interface
  - Optimization Goal: Speed
  - NUM_DATA_STREAMS: 8
  - Number of Slices: 2225 out of 13696
  - Number of BRAMs: 16 out of 136
Verification

- Host interface
  - Synthesized a design with eight streams
  - Wrote a series of values to each of the streams
  - Read the status register to see if it reflected the streams' fullness
  - Read the individual streams' output registers to see if the arithmetic had taken place
Summary

• Studied the functionality of FERPs
• Designed a stream by putting together FERPs and FIFOs with the relevant state machine
• Built a parameterized host interface
• Verified functionality with a simple adder slotted into the host interface
Questions