SRL FIFO Core Specification

Author: Andrew Mulcock
Andrewm@opencores.orgs

Rev. 0.1
January 22, 2008

Revision History

Rev.	Date	Author	Description
0.1		Andrew Mulcock	First Draft

Contents

1 Introduction

The SRL FIFO is a group of First In First Out (FIFO's) registers, based upon the shift register principle. These are aimed in particular at the Xilinx range of FPGA's that have very efficient shift register implementations, known as SRL.

Shift register based FIFOs have been around for years, The definitive article on shift register based FIFOs is a Xilinx document. Not to surprising when according to wiki, it's Peter Alfke who is now at Xilinx who invented the first electronic (ASIC) FIFO.

Initially there are two versions of the FIFO available,

- Srl_fifo_16
- Srl_fifo_32

These are to cover FPGA's that have an inherent SRL of 16 long or 32 long.

Features:

- **Standard VHDL, no instantiated blocks**
- **Small size, 8 bit wide , 32 deep FIFO uses 19 LUTs in a Spartan 3A**
- **Operates from a wide range of input clock frequencies**
- **Static synchronous design**
- **Fully synthesizable**

2
IO ports & Parameters

2.1 Core Parameters

Parameter	Default	Description
Width	8	How many bits wide is the FIFO data path

2.1.1 Width
The width of the FIFOs data in and data out ports is set by the GENERIC Width.

2.2 Ports

Port	Width	Direction	Description
Reset	1	Input	Synchronous active high reset
Clk	1	Input	Clock
Data_in	Width	Input	Data in to FIFO
Data_out	Width	Output	Data out of FIFO
Write	1	Input	Active High Write. High with clock to write new data in
Read	1	Input	Active High Write. High with clock to write new data in
Data_present	1	Output	Synchronous to clock, High when FIFO is not empty
Half_full	1	Output	Synchronous to clock, High when FIFO is ½ full
Full	1	Output	Synchronous to clock, High when FIFO is full

3 Operation

3.1 Reset

The RESET is required to be active for at least two clock cycles to initialize the FIFO.

3.2 Writing

Data can be written to the FIFO provided the FIFO is not full. This is indicated by the FULL output being a '1'.Writing is achieved by the WRITE being high and the DATA_IN inputs being stable on the rising edge of the clock.

Writing to the FIFO if the FIFO is full is none destructive on the FIFO contents or flags.

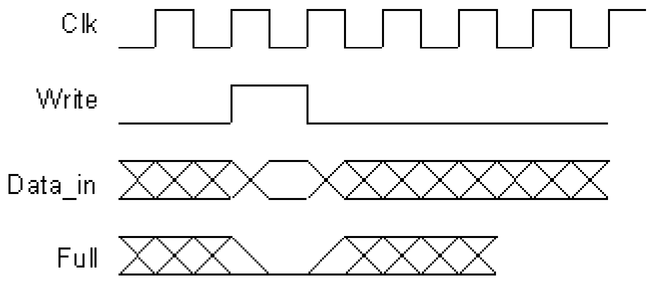

Figure 1 Write cycle

3.3 Reading

Data can be read from the FIFO provided the FIFO has data in it. This is indicated by the DATA_PRESENT output being a '1'. Reading is achieved by the READ input being high on the rising edge of the clock.

Reading from an empty FIFO is none destructive on the flags or data in the FIFO.

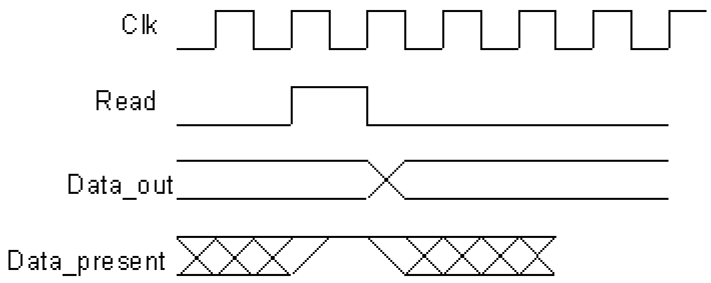

Figure 2 Read cycle

3.4 Full Flag

The full flag goes high on the rising edged of the clock when data is being written into the last position of the FIFO and no read is happening.

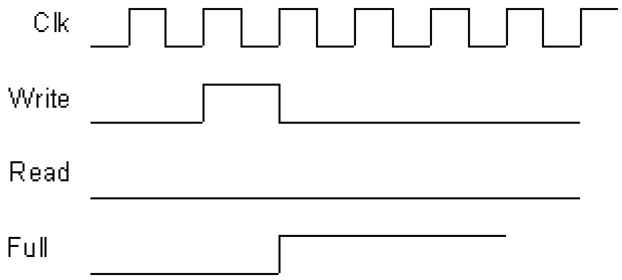

Figure 3 Full Flag, going full

The full flag goes low on the rising edge of the clock when data is being read from the FIFO and no write is happening.

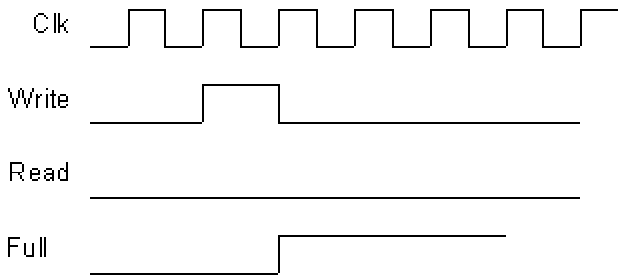

Figure 4 Full Flag, going none full

3.5 Data_Present Flag

The Data_present flag goes high on the rising edged of the clock when the first data is being written into the FIFO.

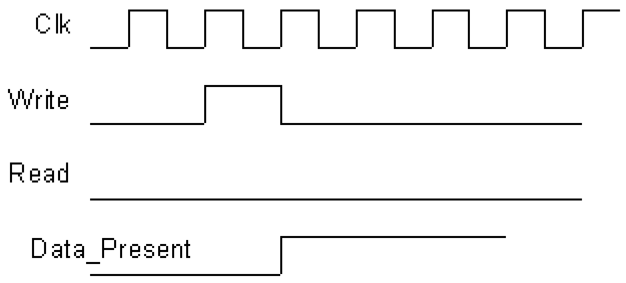

Figure 5 Data_present Flag, first data written in

The Data_present flag goes low on the rising edged of the clock when the last data is being read from the FIFO provided no write is happening.

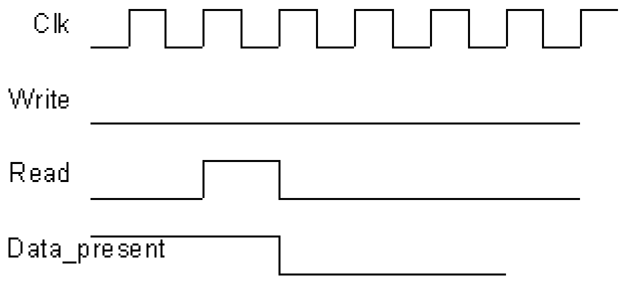

Figure 6 Data_present Flag, last data read from FIFO

5 Architecture

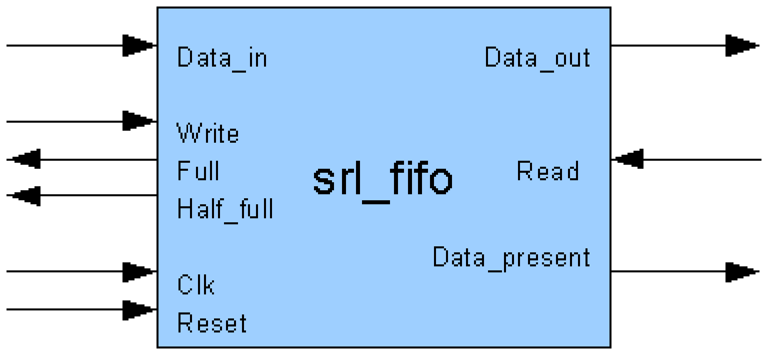

Figure 7 Top level Architecture