Key Design Features

- Synthesizable, technology independent VHDL IP Core
- Fully synchronous design
- Configurable depth and data width
- Register or RAM-based storage[1]
- Full / Empty flags and FIFO fullness counter
- Uses a simple valid-ready streaming protocol
- Compatible with other streaming protocols such as: AMBA® AXI4-stream, Altera® Avalon-ST and Xilinx® local-link
- 1 cycle latency[2]
- Very high-speed operation achieving 400 MHz+ on basic FPGA devices

Applications

- General purpose buffering
- Adapting to different data rates
- Interfacing between other pipeline elements
- Registering the data-path in a pipeline to improve timing

Pin-out Description

Pin name	I/O	Description	Active state
clk	in	Synchronous clock	rising edge
reset	in	Asynchronous reset	low
fifo_full	out	FIFO full flag	high
fifo_empty	out	FIFO empty flag	high
fifo_fullness [log2d:0]	out	Counter indicating number of entries in the FIFO that are currently used	counter
datain [dw-1: 0]	in	FIFO input data	data
datain_val	in	Indicates valid data at the FIFO input	high
datain_rdy	out	Indicates that the FIFO is ready to accept data	high
dataout [dw-1:0]	out	FIFO output data	data
dataout_val	out	Indicates that the FIFO contains valid data	high
dataout_rdy	in	Indicates that the output is ready to receive data	high

Block Diagram

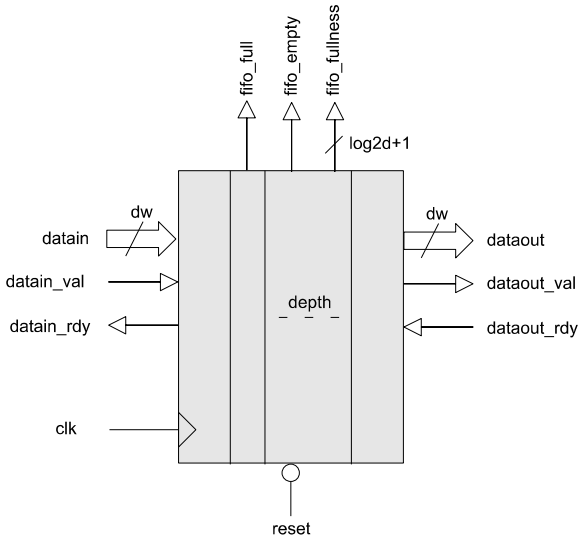

Figure 1: Synchronous FIFO

Generic Parameters

Generic name	Description	Type	Valid range
dw	FIFO data width	integer	≥ 1
depth	FIFO depth	integer	≥ 2
log2d	Log2 FIFO depth	integer	log2 (depth)
regout	Enable output data register - improves output timing when enabled	boolean	TRUE/FALSE

General Description

FIFO_SYNC is a general purpose synchronous FIFO with configurable depth and data width.

Data flows in and out of the FIFO in accordance with the valid-ready pipeline protocol[3]. Data is written to the FIFO on a rising clock-edge when *datain_val* is high and *datain_rdy* is high. Data is read from the FIFO on a rising clock-edge when *dataout_val* is high and *dataout_rdy* is high.

In addition to these handshake signals, the flags *fifo_full* and *fifo_empty* are also provided for compatibility with standard FIFO interfaces. The signals *datain_val* and *datout_rdy* can respectively be considered as the traditional write and read strobes into the FIFO. The counter value *fifo_fullness* indicates how many entries are currently used in the FIFO.

1 Type of inferred storage depends on synthesis tool configuration
2 If output register is enabled then latency is 2 cycles

3 For more examples of the valid-ready protocol and it's implementation see Zipcores application nore: app_note_zc001.pdf

It is important to note that when the FIFO is configured with the output register set to true, then the signals *fifo_empty* and *fifo_fullness* are delayed by 1 clock cycle due to the 1 cycle latency of the output register.

If the FIFO is empty, then there is a 1 clock cycle delay (or latency) between a write at the input and the read data being available at the output. If the output register is enabled, then the latency is 2 clock cycles.

Functional Timing Diagrams

The following timing diagrams relate to a 16-deep FIFO design with the output register option set to false.

Figure 2 shows a FIFO write operation when the FIFO state changes from almost full to full. Figure 3 demonstrates a FIFO read operation when the FIFO state changes from almost empty to empty. Note that data is only transferred at the FIFO interfaces on a rising clock edge when valid and ready are both active high.

For more examples of the valid-ready protocol and it's implementation see Zipcores application note: app_note_zc001.pdf.

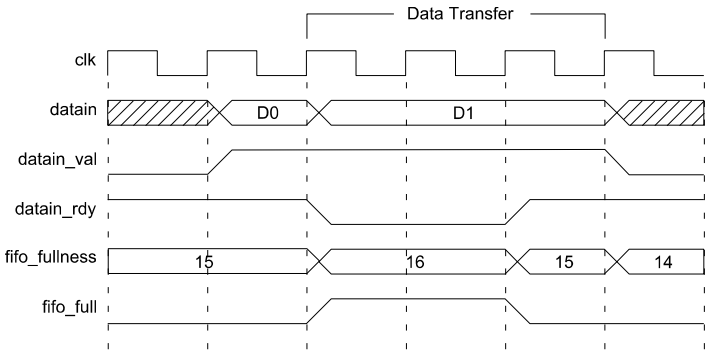

Figure 2: FIFO write operation

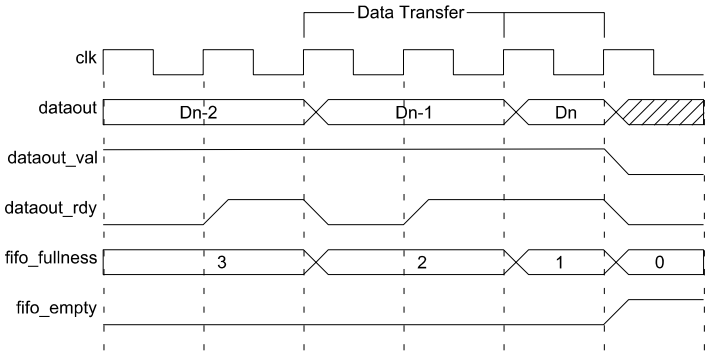

Figure 3: FIFO read operation

Source File Description

All source files are provided as text files coded in VHDL. The following table gives a brief description of each file.

Source file	*Description*
pipeline_reg.vhd	FIFO output register for the case when the generic *regout* is set to true
fifo_sync.vhd	Top-level block
fifo_sync_bench.vhd	Top-level test bench

Functional Testing

An example VHDL testbench is provided for use in a suitable VHDL simulator. The compilation order of the source code is as follows:

1. pipeline_reg.vhd
2. fifo_sync.vhd
3. fifo_sync_bench.vhd

The VHDL testbench instantiates the FIFO component and the user may modify the generic parameters as required.

The simulation must be run for at least 3 ms during which time a randomized input data sequence will be written to the FIFO.

In addition to random input data, the testbench also generates randomized valid-ready handshake signals at the input and output of the FIFO in order to verify that the the flow-control is working correctly.

The simulation generates two text files called: *fifo_sync_in.txt* and *fifo_sync_out.txt*. These files respectively contain the input and output data captured at the FIFO interfaces during the test. Matching input and output files indicate that the test has run successfully.

Synthesis

The files required for synthesis and the design hierarchy is shown below:

- fifo_sync.vhd
  - pipeline_reg.vhd

The VHDL core is designed to be technology independent. However, as a benchmark, synthesis results have been provided for the Xilinx® Virtex 6 and Spartan-6 FPGA devices. Synthesis results for other FPGAs and technologies can be provided on request.

Note that as the FIFO is highly generic in nature, the synthesis results will vary significantly with choice of parameters. It is also important to note that the type of inferred storage used by the FIFO is largely dependent on the attributes set up in the synthesis tool.

If the output register is enabled, then internally, a FIFO memory read has a 1 clock-cycle latency. This generally permits the tool to infer a dual-port RAM. If the output register is disabled, then distributed RAM (registers) are normally inferred.

Trial synthesis results are shown with the generic parameters set to: dw = 16, depth = 32, log2d = 5, regout = true.

Resource usage is specified after Place and Route.

VIRTEX 6

Resource type	***Quantity used***
Slice register	33
Slice LUT	39
Block RAM	0
DSP48	0
Occupied slices	14
Clock frequency (approx)	420 MHz

SPARTAN 6

Resource type	***Quantity used***
Slice register	33
Slice LUT	56
Block RAM	0
DSP48	0
Occupied slices	19
Clock frequency (approx)	370 MHz

Revision History

Revision	***Change description***	***Date***
1.0	Initial revision	16/04/2008
1.1	Includes more efficient pipeline register component	07/02/2014