

Key Design Features

- Synthesizable, technology independent VHDL Core
- Dual-clock architecture
- Configurable data width
- Gray-coded read/write pointers
- Full/Empty flags
- Simple valid/ready pipeline interface protocol
- Output register option for improved timing

Applications

- Re-synchronizing between clock domains
- Registering the datapath in and off chip
- General purpose buffering and rate adaptation

Pin-out Description

Pin name	I/O	Description	Active state
clk_a	in	Input clock domain a	rising edge
clk_b	in	Input clock domain b	rising edge
reset	in	Asynchronous reset	low
fifo_full	out	FIFO full flag (clock domain a)	high
fifo_empty	out	FIFO empty flag (clock domain b)	high
a_data [dw-1:0]	in	FIFO input data	data
a_val	in	Indicates valid data at the FIFO input	high
a_rdy	out	Indicates that the FIFO is ready to accept data	high
b_data [dw-1:0]	out	FIFO output data	data
b_val	out	Indicates that the FIFO contains valid data	high
b_rdy	in	Indicates that the output is ready to receive data	high

Generic Parameters

Generic name	Description	Type	Valid range
dw	FIFO data width	integer	≥ 1
regout	Enable output data register - improves output timing when enabled	boolean	TRUE/FALSE

Block Diagram

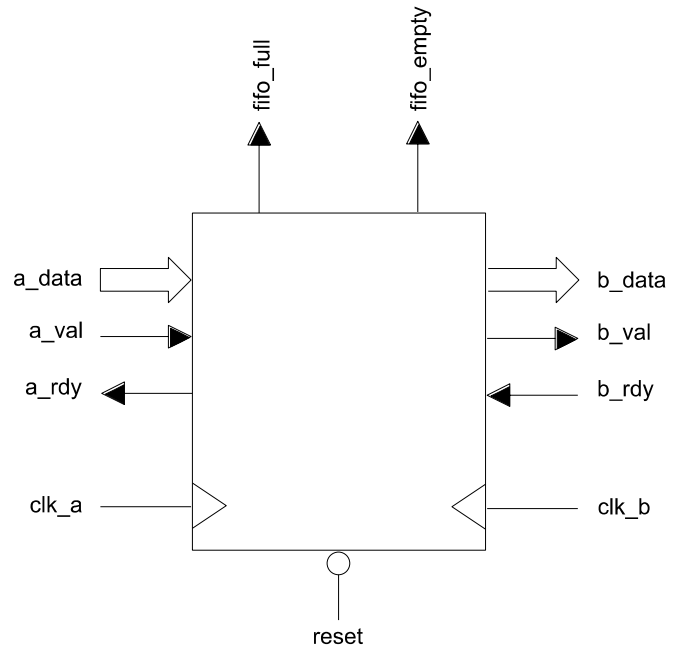


Figure 1: Asynchronous FIFO

General Description

FIFO_ASYNC is an Asynchronous FIFO with a fixed depth of 16 and a configurable data width. The FIFO uses register-based storage.

The FIFO has two independent clocks 'a' and 'b' in order to allow the re-synchronization of the datapath across different clock domains. Internally, the FIFO uses gray-encoded read and write pointers. This ensures that the pointers are resynchronized as single-bit transitions only.

Data flows in and out of the FIFO in accordance with the valid/ready protocol. Data is written to the FIFO on the rising clock-edge of *clk_a* when *a_val* is high and *a_rdy* is high. Data is read from the FIFO on the rising clock-edge of *clk_b* when *b_val* is high and *b_rdy* is high.

In addition to these handshake signals, the flags *fifo_full* and *fifo_empty* are also provided for a more traditional approach to FIFO interfacing. The signal *fifo_full* is synchronized to clock domain 'a' and *fifo_empty* to clock domain 'b'. The signals *a_val* and *b_rdy* can respectively be considered as the traditional write and read strobes into the FIFO.

It is important to note that when the FIFO is configured with the output register set to true, then the signal *fifo_empty* is delayed by 1 clock cycle due to the 1 cycle latency of the output register. For this reason it is recommended that only the valid/ready handshake signals are used for flow-control when the output register is enabled.

Functional Timing Diagrams

The following timing diagrams relate to a 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.

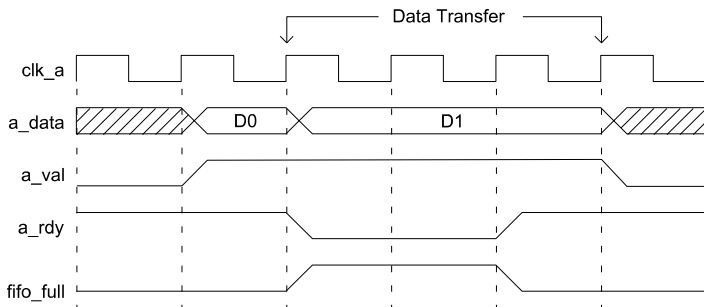


Figure 2: FIFO write operation

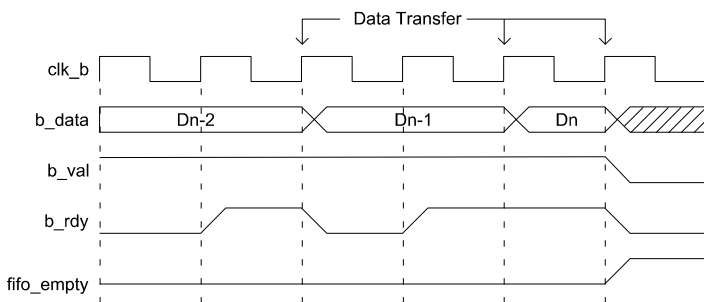


Figure 3: FIFO read operation

For more examples of the valid/ready protocol and its implementation see ZIPcores application note: app_note_zc001.pdf.

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 <i>regout</i> is set to true
fifo_async.vhd	Top-level block
fifo_async_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_async.vhd
3. fifo_async_bench.vhd

The VHDL testbench instantiates the FIFO component and the user may modify the generic parameters as required. In addition, the user may specify the frequency of *clk_a* and *clk_b* in order to observe the data flow under different asynchronous clocking conditions.

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 test bench also generates randomized valid/ready handshake signals at the input and output of the FIFO in order to verify that the flow-control is working correctly.

The simulation generates two text files: *fifo_async_in.txt* and *fifo_async_out.txt*. These files respectively contain the input and output data captured at the FIFO interfaces during the test. If the files match then the test has been successful.

Synthesis

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

- fifo_async.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 5 and the Altera Stratix III series of FPGA devices. The lowest and highest speed grade devices have been chosen in both cases for comparison.

Trial synthesis results are shown with the generic parameters set to: *dw* = 32, *regout* = true.

Resource usage is specified after Place and Route.

VIRTEX 5

Resource type	Quantity used
Slice register	71
Slice LUT	56
Block RAM	0
DSP48	0
Clock frequency (worst case)	285 MHz
Clock frequency (best case)	379 MHz

STRATIX III

Resource type	Quantity used
Register	585
ALUT	210
Block Memory bit	0
DSP block 18	0
Clock frequency (worst case)	290 MHz
Clock frequency (best case)	383 MHz

Revision History

Revision	Change description	Date
1.0	Initial revision	21/04/2008