1

File description

1.1 "fifo_control.v"

Module FIFO_CONTROL includes control logic for single FIFO. It consists of read and write address generation and full, almost full, empty and almost empty status generation. It also generates read and write allow signals, which are used for enabling/disabling memory used for FIFO. Control logic can be used for independent read and write clocks.

1.2 "wbw_wbr_fifos.v"

WBW_WBR_FIFOS module is actual interface for WISHBONE write and read FIFOs. It instantiates necessary control logic (FIFO_CONTROL) and RAM modules, which can be synchronous or asynchronous, depending on parameter definition. It provides control, data and status interface signals for both WISHBONE write and WISHBONE read FIFOs.

1.3 "pciw_pcir_fifos.v"

PCIW_PCIR_FIFOS module is actual interface for PCI write and read FIFOs. It instantiates necessary control logic (FIFO_CONTROL) and RAM modules, which can be synchronous or asynchronous, depending on parameter definition. It provides control, data and status interface signals for both PCI write and PCI read FIFOs. It's almost identical to WBW_WBR_FIFOs – it's done separately because of more readable connection to other modules.

1.4 “dp_async_ram.v”

Module DP_ASYNC_RAM is just a behavioral asynchronous dual port RAM for simulation purposes.

1.5 “dp_sram.v”

Module DP_SRAM is behavioral synchronous dual port RAM for simulation purposes.

1.6 “wb_tb.v”

Module WB_TB is a little test bench (more demonstration than test bench) for WBW_WBR_FIFOS module.

1.7 “pci_tb.v”

Module PCI_TB is a little test bench (more demonstration than test bench) for PCIW_PCIR_FIFOS module.

2

Operation

2.1 Parameters

FIFO parameters are defined in global "constants.v" file in their own section. Available parameters, their description and FIFO behavior impact:

Parameter	Description
WBW_DEPTH	Number of locations in WBW FIFO. Note that one location is always unused. Minimum number is 8.
WBW_ADDR_LENGTH	Address length for WBW_FIFO RAM. It must be log2(WBW_DEPTH).
WBR_DEPTH	Number of locations in WBR FIFO. Note that one location is always unused. Minimum number is 8.
WBR_ADDR_LENGTH	Address length for WBR_FIFO RAM. It must be log2(WBR_DEPTH).
PCIW_DEPTH	Number of locations in PCIW FIFO. Note that one location is always unused. Minimum number is 8.
PCIW_ADDR_LENGTH	Address length for PCIW_FIFO RAM. It must be log2(PCIW_DEPTH).
PCIR_DEPTH	Number of locations in PCIR_FIFO. Note that one location is always unused. Minimum number is 8.
PCIR_ADDR_LENGTH	Address length for PCIR_FIFO RAM. It must be log2(PCIR_DEPTH).

FPGA BIG	If FPGA is defined, RAMB4_S16_S16 modules are instantiated for FIFOs storage space. If BIG is commented out, than 3 Block Select RAM+ modules are shared between two FIFOs (max depth is 128). FPGA must provide at least 6 of them. This slows down FIFO operation a bit, since it requires a turnaround cycle between writes and reads. If BIG is defined, 3 Block Select RAMs are used for each FIFO. That means a FPGA with at least 12 of them. (max depth 256)
SYNCHRONOUS	Only has meaning when FPGA is not defined (commented out). It invokes elaboration of FIFO control logic that supports synchronous RAM storage for FIFOs. If FPGA and SYNCHRONOUS are undefined, asynchronous RAM storage and control logic is assumed.

2.2 Module descriptions

2.2.1 WBW_WBR_FIFO

Module incorporates both WISHBONE slave unit's FIFOs: WISHBONE read and WISHBONE write FIFO. They are joined in a single module because of same storage space sharing in certain implementations.

Module connections:

- Common signals

Signal name	Size	Direction	Description
wb_clock_in	1	I	WISHBONE bus clock input. It's WBR_FIFO's read clock and WBW_FIFO's write clock
pci_clock_in	1	I	PCI bus clock input. It's WBR_FIFO's write clock and WBW_FIFO's read clock
reset_in	1	I	System reset input

- WBW_FIFO signals

Signal name	Size	Direction	Description
wbw_wenable_in	1	I	WBW_FIFO's write enable input. When FIFO is not full and write enable is high, data provided on data inputs will be written to FIFO on rising WISHBONE clock edge.
wbw_addr_data_in	32	I	WBW_FIFO's address / data input. Address entry is always first entry of transaction with one or more data entries following.
wbw_cbe_in	4	I	WBW_FIFO's bus command / byte enable input. First entry of transaction written with address is PCI bus command, byte enables are written with data entries (they are negated SEL_I signals from WISHBONE since PCI BE signals are active low)
wbw_control_in	4	I	WBW_FIFO's control bus input – encoded control values are written to FIFO to communicate to PCI side. Common controls are defining address entry, intermediate and last data entry of transaction, so opposite side knows when to terminate the cycle.
wbw_renable_in	1	I	WBW_FIFO's read enable input. When FIFO is not empty and read enable is high, data provided on data outputs will change to next written location on rising PCI clock edge. Reading side must only assert read enable when it intends to read currently provided data.
wbw_addr_data_out	32	O	WBW_FIFO's address / data output. Address entry is always first entry of transaction with one or more data entries following.
wbw_cbe_out	4	O	WBW_FIFO's bus command / byte enable output. First entry of transaction read concurrently with address is PCI bus command, byte enables are read concurrently with data entries (they are negated SEL_I signals from WISHBONE since PCI BE signals are active low)

wbw_control_out	4	O	WBW_FIFO's reading side monitors current control bus output for various encoded values, so it knows when to start or end current cycle in progress.
wbw_flush_in	1	I	WBW_FIFO's flush input. It doesn't actually delete or initialize storage space – it just resets address counters so FIFO is marked empty. Any data in the FIFO is therefore invalid. Flush signal should remain asserted for at least one PCI or WB clock cycle.
wbw_almost_full_out	1	O	WBW_FIFO's almost full status output. Indicates that only one location in the FIFO is empty. When WB side writes to the FIFO and samples this signal asserted on rising WB clock edge, it should stop writing immediately, since last location was just written.
wbw_full_out	1	O	WBW_FIFO's full status output – indicates that no free locations are currently available in the FIFO.
wbw_almost_empty_out	1	O	WBW_FIFO's almost empty status output. Indicates that only one location in the FIFO is still valid. When PCI side reads from the FIFO and samples this signal asserted on rising PCI clock edge, it should stop reading immediately, since last location was just read and FIFO is empty.
wbw_empty_out	1	O	WBW_FIFO's empty status output – indicates that no locations containing valid data are currently available in the FIFO.
wbw_transaction_ready_out	1	O	Indicates that at least one unfinished transaction is present in the WBW_FIFO. Transaction ready generation is done by monitoring control bus input and output. When control bus input indicates last data phase, that means one complete transaction was transferred to the FIFO. When control bus output indicates last data phase it means that one complete transaction was pulled out of FIFO.

- WBR_FIFO signals

Signal name	Size	Direction	Description
wbr_wenable_in	1	I	WBR_FIFO's write enable input. When FIFO is not full and write enable is high, data provided on data inputs will be written to FIFO on rising PCI clock edge.
wbr_data_in	32	I	WBR_FIFO's data input.
wbr_be_in	4	I	WBR_FIFO's byte enable input. Byte enables are written with data entries (they are negated BE# signals from PCI bus since WB SEL_I signals are active high)
wbr_control_in	4	I	WBR_FIFO's control bus input – encoded control values are written to FIFO to communicate to WB side. Common controls are defining intermediate and last data entry of transaction, so opposite side knows when to terminate the cycle.
wbr_renable_in	1	I	WBR_FIFO's read enable input. When FIFO is not empty and read enable is high, data provided on data outputs will change to next written location on rising WB clock edge. Reading side must only assert read enable when it intends to read currently provided data.
wbr_data_out	32	O	WBR_FIFO's data output.
wbr_be_out	4	O	WBR_FIFO's byte enable output. Byte enables are read concurrently with data entries (they are negated BE# signals from PCI since WB SEL_I signals are active high)
wbr_control_out	4	O	WBR_FIFO's reading side monitors current control bus output for various encoded values, so it knows when to start or end current cycle in progress.
wbr_flush_in	1	I	WBR_FIFO's flush input. It doesn't actually delete or initialize storage space – it just resets address counters so FIFO is marked empty. Any data in the FIFO is therefore invalid. Flush signal should remain asserted for at least one PCI or WB clock cycle.

wbr_almost_full_out	1	O	WBR_FIFO's almost full status output. Indicates that only one location in the FIFO is empty. When PCI side writes to the FIFO and samples this signal asserted on rising PCI clock edge, it should stop writing immediately, since last location was just written.
wbr_full_out	1	O	WBR_FIFO's full status output – indicates that no free locations are currently available in the FIFO.
wbr_almost_empty_out	1	O	WBR_FIFO's almost empty status output. Indicates that only one location in the FIFO is still valid. When WB side reads from the FIFO and samples this signal asserted on rising WB clock edge, it should stop reading immediately, since last location was just read and FIFO is empty.
wbr_empty_out	1	O	WBR_FIFO's empty status output – indicates that no locations containing valid data are currently available in the FIFO.
wbr_transaction_ready_out	1	O	Indicates that at least one unfinished transaction is present in the WBR_FIFO. Transaction ready generation is done by monitoring control bus input and output. When control bus input indicates last data phase, that means one complete transaction was transferred to the FIFO. When control bus output indicates last data phase it means that one complete transaction was pulled out of FIFO.

2.2.2 PCIW_PCIR_FIFO

Module incorporates both PCI target unit's FIFOs: PCI read and PCI write FIFO. They are joined in a single module because of same storage space sharing in certain implementations.

Module connections:

- Common signals

Signal name	Size	Direction	Description
wb_clock_in	1	I	WISHBONE bus clock input. It's PCIR_FIFO's write clock and PCIW_FIFO's read clock
pci_clock_in	1	I	PCI bus clock input. It's PCIR_FIFO's read clock and PCIW_FIFO's write clock
reset_in	1	I	System reset input

- PCIW_FIFO signals

Signal name	Size	Direction	Description
pciw_wenable_in	1	I	PCIW_FIFO's write enable input. When FIFO is not full and write enable is high, data provided on data inputs will be written to FIFO on rising PCI clock edge.
pciw_addr_data_in	32	I	PCIW_FIFO's address / data input. Address entry is always first entry of transaction with one or more data entries following.
pciw_cbe_in	4	I	PCIW_FIFO's bus command / byte enable input. First entry of transaction written with address is PCI bus command, byte enables are written with data entries (they are negated BE# signals from PCI since WISHBONE SEL_O signals are active high)
pciw_control_in	4	I	PCIW_FIFO's control bus input – encoded control values are written to FIFO to communicate to WB side. Common controls are defining address entry, intermediate and last data entry of transaction, so opposite side knows when to terminate the cycle.

pciw_renable_in	1	I	PCIW_FIFO's read enable input. When FIFO is not empty and read enable is high, data provided on data outputs will change to next written location on rising WB clock edge. Reading side must only assert read enable when it intends to read currently provided data.
pciw_addr_data_out	32	O	PCIW_FIFO's address / data output. Address entry is always first entry of transaction with one or more data entries following.
pciw_cbe_out	4	O	PCIW_FIFO's bus command / byte enable output. First entry of transaction read concurrently with address is PCI bus command, byte enables are read concurrently with data entries (they are negated BE# signals from PCI since WB SEL_O signals are active high)
pciw_control_out	4	O	PCIW_FIFO's reading side monitors current control bus output for various encoded values, so it knows when to start or end current cycle in progress.
pciw_flush_in	1	I	PCIW_FIFO's flush input. It doesn't actually delete or initialize storage space – it just resets address counters so FIFO is marked empty. Any data in the FIFO is therefore invalid. Flush signal should remain asserted for at least one PCI or WB clock cycle.
pciw_almost_full_out	1	O	PCIW_FIFO's almost full status output. Indicates that only one location in the FIFO is empty. When PCI side writes to the FIFO and samples this signal asserted on rising PCI clock edge, it should stop writing immediately, since last location was just written.
pciw_full_out	1	O	PCIW_FIFO's full status output – indicates that no free locations are currently available in the FIFO.
pciw_almost_empty_out	1	O	PCIW_FIFO's almost empty status output. Indicates that only one location in the FIFO is still valid. When WB side reads from the FIFO and samples this signal asserted on rising WB clock edge, it should stop reading immediately, since last location was just read and FIFO is empty.

pciw_empty_out	1	O	PCIW_FIFO's empty status output – indicates that no locations containing valid data are currently available in the FIFO.
pciw_transaction_ready_out	1	O	Indicates that at least one unfinished transaction is present in the PCIW_FIFO. Transaction ready generation is done by monitoring control bus input and output. When control bus input indicates last data phase, that means one complete transaction was transferred to the FIFO. When control bus output indicates last data phase it means that one complete transaction was pulled out of FIFO.

- PCIR_FIFO signals

Signal name	Size	Direction	Description
pcir_wenable_in	1	I	PCIR_FIFO's write enable input. When FIFO is not full and write enable is high, data provided on data inputs will be written to FIFO on rising WB clock edge.
pcir_data_in	32	I	PCIR_FIFO's data input.
pcir_be_in	4	I	PCIR_FIFO's byte enable input. Byte enables are written with data entries (they are negated SEL_O signals from WB bus since PCI BE# signals are active high)
pcir_control_in	4	I	PCIR_FIFO's control bus input – encoded control values are written to FIFO to communicate to PCI side. Common controls are defining intermediate and last data entry of transaction, so opposite side knows when to terminate the cycle.

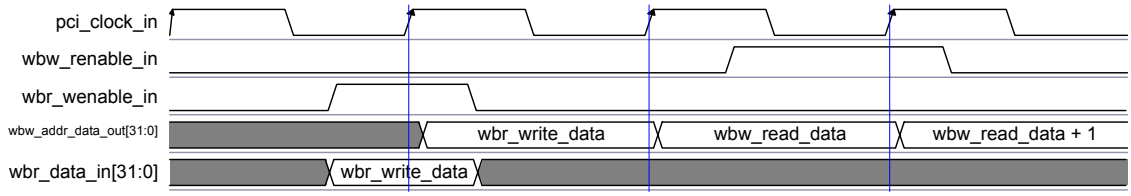
pcir_renable_in	1	I	PCIR_FIFO's read enable input. When FIFO is not empty and read enable is high, data provided on data outputs will change to next written location on rising PCI clock edge. Reading side must only assert read enable when it intends to read currently provided data.
pcir_data_out	32	O	PCIR_FIFO's data output.
pcir_be_out	4	O	PCIR_FIFO's byte enable output. Byte enables are read concurrently with data entries (they are negated SEL_O signals from WB since PCI BE# signals are active low)
pcir_control_out	4	O	PCIR_FIFO's reading side monitors current control bus output for various encoded values, so it knows when to start or end current cycle in progress.
pcir_flush_in	1	I	PCIR_FIFO's flush input. It doesn't actually delete or initialize storage space – it just resets address counters so FIFO is marked empty. Any data in the FIFO is therefore invalid. Flush signal should remain asserted for at least one PCI or WB clock cycle.
pcir_almost_full_out	1	O	PCIR_FIFO's almost full status output. Indicates that only one location in the FIFO is empty. When WB side writes to the FIFO and samples this signal asserted on rising WB clock edge, it should stop writing immediately, since last location was just written.
pcir_full_out	1	O	PCIR_FIFO's full status output – indicates that no free locations are currently available in the FIFO.
pcir_almost_empty_out	1	O	PCIR_FIFO's almost empty status output. Indicates that only one location in the FIFO is still valid. When PCI side reads from the FIFO and samples this signal asserted on rising PCI clock edge, it should stop reading immediately, since last location was just read and FIFO is empty.
pcir_empty_out	1	O	PCIR_FIFO's empty status output – indicates that no locations containing valid data are currently available in the FIFO.

pcir_transaction_ready_out	1	O	Indicates that at least one unfinished transaction is present in the PCIR_FIFO. Transaction ready generation is done by monitoring control bus input and output. When control bus input indicates last data phase, that means one complete transaction was transferred to the FIFO. When control bus output indicates last data phase it means that one complete transaction was pulled out of FIFO.

2.3 Waveforms

2.3.1 FPGA w/o BIG

As stated in parameters section, in small FPGAs Block Ram sharing is performed, which requires turnaround cycle between writes and reads. Turnaround is not necessary between reads and writes. For example – when PCI master interface writes data to WBR_FIFO on PCI side, it must wait for at least one clock cycle before it can perform a read from WBW_FIFO. Since PCI bus protocol also defines turnaround cycles and bus idle states between transactions, this shouldn't have any impact at all to overall bridge performance. In any case, FPGA implementation uses synchronous storage space, so control logic for synchronous RAMs is implemented.



First clock edge: Data from a read transaction is written to WBR_FIFO. Data is written on the rising edge of PCI clock when wbr_wenable_in is high. Data is also mirrored to wbw_addr_data_out output of WBW_FIFO. Second rising edge is turnaround cycle, when mirrored write data is replaced by actual data coming out of WBW_FIFO. On the third clock edge this data is read from FIFO and next data is provided immediately after rising PCI clock edge when wbw_renable_in is high.

2.3.2 FPGA with BIG defined

This implementation doesn't share Blocks of RAM between FIFOs, so turnaround is not necessary. Everything else is the same as in small FPGAs (w/o BIG). Implementation of control logic is for synchronous storage space.

2.3.3 SYNCHRONOUS

If some other storage space is provided and is synchronous, than this storage should be instantiated and properly connected to control logic in files wbw_wbr_fifos.v and pciw_pcir_fifos.v. Parameter SYNCHRONOUS should be defined, otherwise FIFOs may not work correctly. If storage is not synchronous, than this parameter should be commented out for reduction of additional logic and higher speeds.

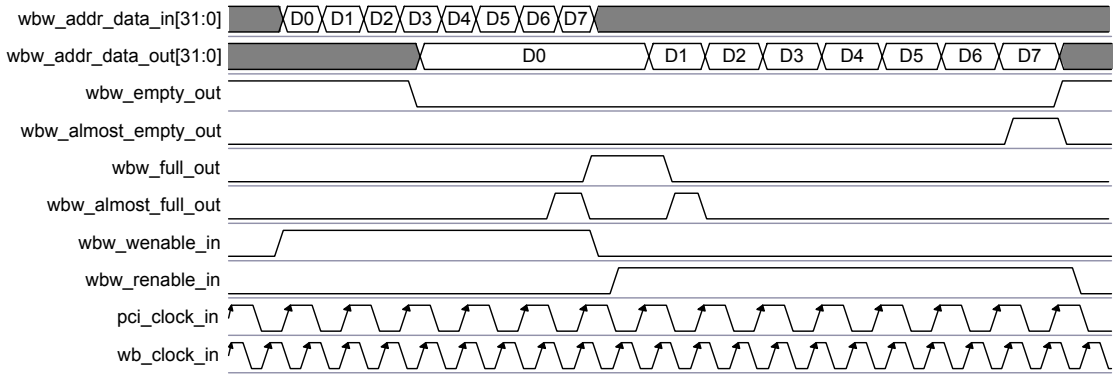


Diagram shows basic SYNCHRONOUS storage WBW_FIFO operation. WBW_FIFO has WISHBONE clock for write port and PCI clock for read port. There is one special case to note in the diagram – empty/not empty status transition. Empty status generation is registered and synchronized to read (PCI) clock. Empty status is stretched for one PCI clock cycle to assure that valid data is present on the output bus. Data can change only on rising clock edges of PCI clock. There is a reason for stretching empty status: When FIFO is empty, read and write addresses are equal. On the first PCI clock edge that control logic senses difference in read/write address, there is no assurance that there has been enough time for data to be properly written to the FIFO's location pointed by read pointer. That's why empty status is stretched and on the next read clock edge data is 100% valid. Operation is analogous for other FIFOs also.

Almost empty status generation is synchronous to read clock. When it's asserted it will remain asserted for at least one read clock cycle.

Almost full and full statuses generation is synchronous to write clock. Both statuses remain asserted for at least one clock cycle.

There is not much difference in timing between synchronous and asynchronous storage, except for stretched empty status, additional logic and registers needed in synchronous implementation for providing data on outputs from FIFO on any rising clock edge w/o wait cycles.