Summary

This white paper shortly describes proprietary Xylon Memory Bus (XMB) interface. This interface can be used in customized and highly-optimized System on Chip (SoC) Xilinx® designs. Practical implementation of the XMB is based on Xylon's logiMEM SDR/DDR SDRAM Memory Controller IP core, which enables an easy connection of various peripheral IP cores to SDRAM memory chips. The logiMEM enables multiple peripherals to share the common memory pool (same SDRAM chips) and its available memory bandwidth.

The logiMEM memory controller supports several system memory interfaces, which can be used at the same time: CoreConnect OPB Slave interface, CoreConnect PLB interface, Xilinx CacheLink (XCL) and Xylon Memory Bus (XMB) interfaces. At the same time, majority of logicBRICKS™ IP cores supports several memory interfaces, which can be configured at the synthesis time by provided HDL generics.

The XMB memory interface is very simple and uses minimal FPGA resources. At the same time enables work with higher memory clocks that enable better utilization of available memory bandwidth (greater throughput). Standard bus architectures, i.e. CoreConnect™ PLB bus, must provide higher level of configurability, support very versatile processor and peripheral cores, etc. Such complexity unavoidably requests more silicon resources, and in FPGAs, decreases clock speeds due to prolonged routing connections.

Xylon Memory Bus

Xylon Memory Interface (XMB) is a simple, small, fast and efficient proprietary interface solution for connecting various logicBRICKS™ IPs to external SDR or DDR SDRAM memory through the logiMEM SDR/DDR SDRAM Memory Controller IP core. XMB interface supports 32/64-bit single and burst read/write accesses.

Signal	Direction*1	Description
req	Input	Memory request
wr	Input	Memory write
ack	Output	Memory acknowledge
addr	Input	Memory address for read/write requests
data	Input	Write data bus
data_be	Input	Write enable selecting data bytes during write cycles
wrack	Output	Memory acknowledge for write cycle
burst_len	Input	Burst length bus
data_out	Output	Read data bus
data_valid	Output	Data valid signal during data read

**1 Direction refers to the logiMEM's inputs and outputs.*

Table 1: XNB Bus Signals

A single instance of the logiMEM supports up to 10 XMB bus ports (system ports). Table 1 lists XMB signals available at every bus port.

Read Cycles

The Read command instructs controller to perform a single or burst read cycle starting from the SDRAM memory address specified by *addr*. The value of burst length bus (*burst_len*) defines the length of the burst read cycle. The actual number of words to be read equals to this value incremented by 1.

Address should be given in bytes and word aligned. For more information about this alignment please see chapter Address Alignment. The word's width depends of used logiMEM port's configuration and equals to either 32-bit or 64-bit XMB data width.

The Read command is issued by inactive (low) Write Enable signal (*wr*) and active read request (*req*) signal to a controller. Data address (*addr*) and data burst length (*burst_len*) should be inserted and valid too.

When the logiMEM memory controller grants an access to a memory port, it sets the acknowledge signal (*ack*) active for one clock period. The request (*req*) signal issued by memory client must be immediately cleared off. The controller issues adequate commands to SDRAM and reads data. Read data are routed to the *data_out* bus and the data valid signal (*data_valid*) stays asserted during a whole burst read memory cycle. The Read data burst appears (*data_out*) a few clock periods after the requested acknowledgment.

Read Operation Quotations:

- Asserted request (req) along with the valid address (addr) and data burst length (burst_len). Write enable signal (wr) inactive (low).
- The request (req) stays asserted until controller acknowledges signal (ack) that clears it off
- The controller presents the first Read burst data (data_out) and the data valid signal (data_valid) a few clock cycles following an assertion of ack signal.

Figure 1 presents Read Cycle timing diagram.

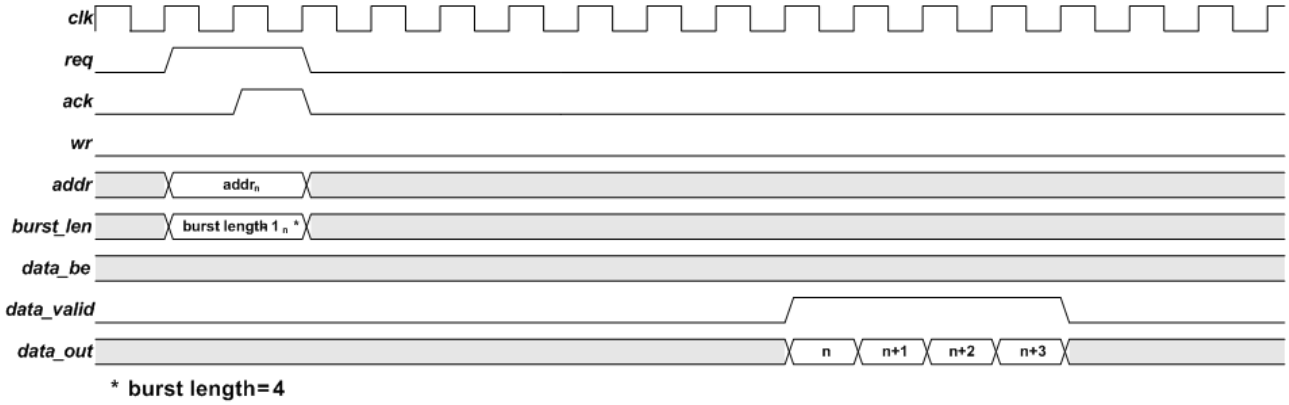

Figure 1: Read Cycle timing diagram

Back to Back Read Cycle

The following timing diagrams (Figure 2, Figure 3 and Figure 4) show two consecutive read cycles. Consecutive access timing depends of configured logiMEM's bank policy rule: "Multi-Bank open" policy or "Always close" bank policy and state of memory itself (row already open or not).

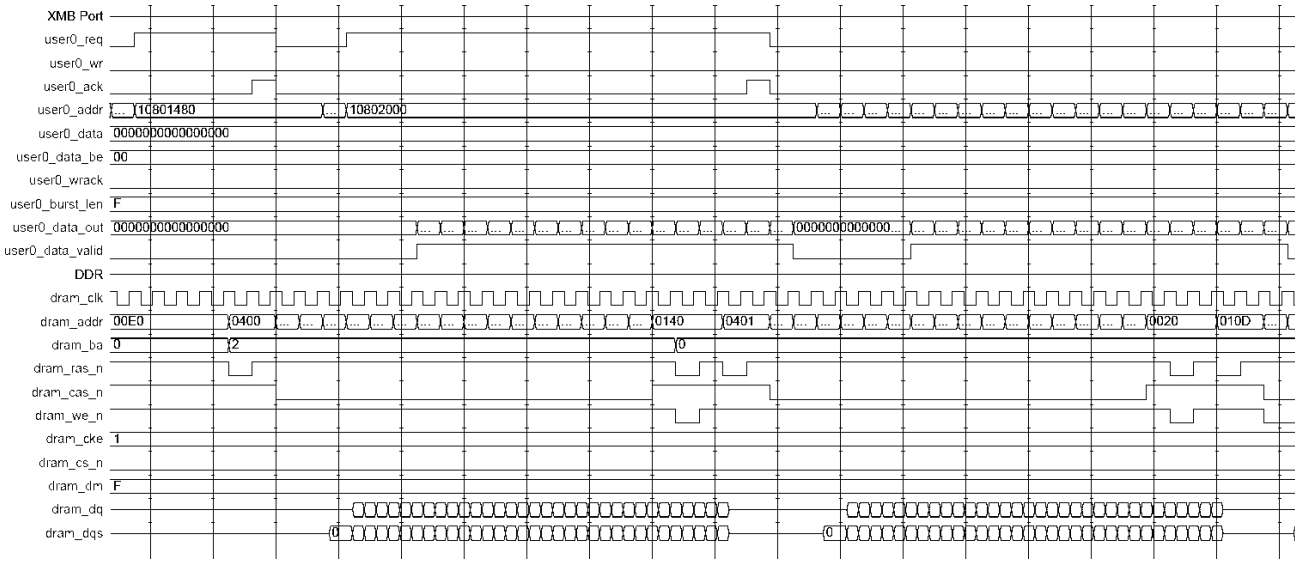

Figure 2: Back to Back Read Cycle (“Always close” bank policy)

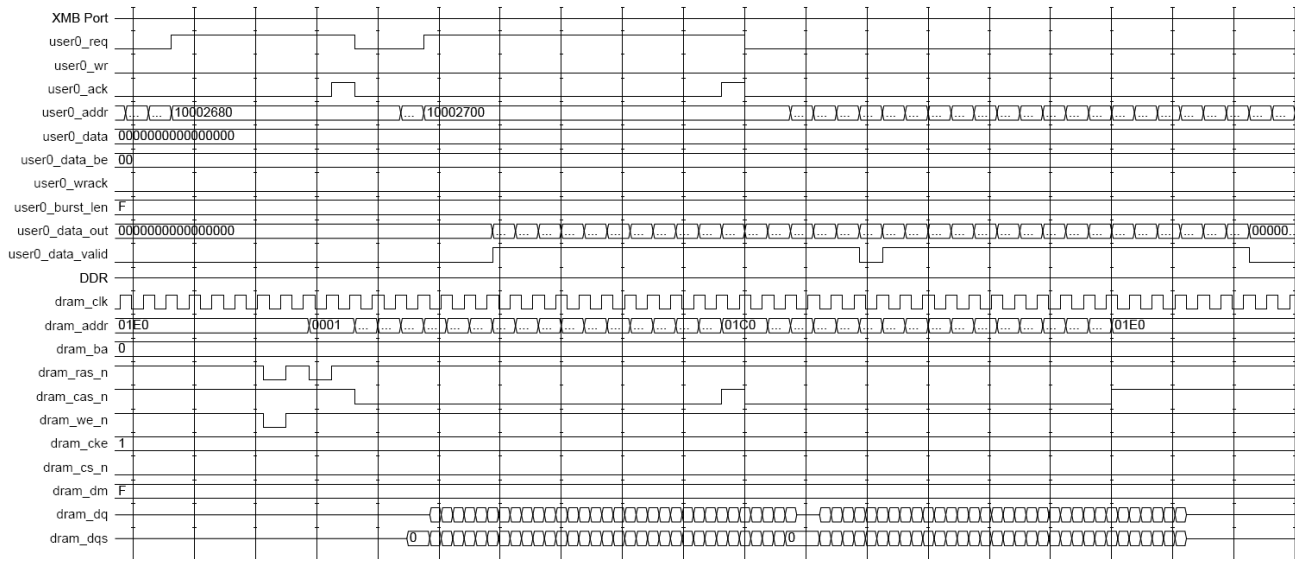

Figure 3: Back to Back Read Cycle (same row/same bank – “Multi-Bank open” policy)

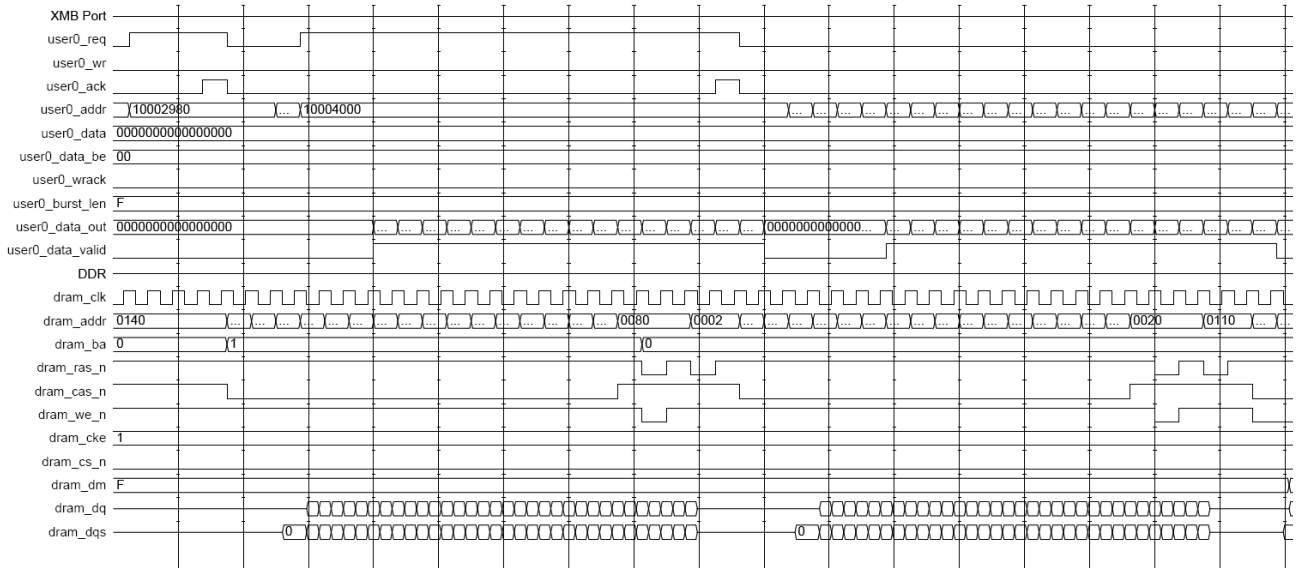

Figure 4: Back to Back Read Cycle (different row/same bank – "Multi-Bank open" policy)

Write cycle

The Write command instructs controller to perform a single or burst Write cycle to the SDRAM starting at the memory address specified by *addr*. Write cycles are similar to Read cycles. The value of burst length bus (*burst_len*) defines the length of burst Write cycle. The actual number of words to be written equals to this value incremented by 1.

An address should be given in bytes and word aligned. For more information about this alignment please see chapter Address Alignment. During request (*req*) to the controller, write enable signal (wr) should be active (high), indicating the Write command. The address (*addr*) and data burst length (*burst_len*) should be valid.

When the logiMEM memory controller grants access to a memory port, it sets acknowledge signal (*ack*) active for one clock period. The request (*req*) must be cleared off immediately. When write data acknowledge signal (*wrack*) become active, valid data and byte enables should be set to respective buses (*data* and *data_be* bus). The remainder of the burst data follows at every consecutive clock. Write data acknowledge signal (*wrack*) remains active until the end of data transfer. The *wrack* may become active together with activation of acknowledge signal (*ack*) at the earliest. The controller issues an adequate commands and route data to the SDRAM memory.

Only word's selected bytes are written into the SDRAM. Byte enable signals select bytes for writing. Each *data_be*[x] signal enables write of one byte of data word (*data_be*[0] enables write of *data*[7:0], *data_be*[1] enables write of *data*[15:8], …).

Write Operation Quotations:

- An asserted request (*req*) along with the valid address (*addr*) and data burst length (*burst_len*), write enable (*wr*) signal active (high)
- The request (*req*) stays asserted until the controller acknowledge signal (*ack*) clears it off
- Wait for activation of signal wrack and set valid data and byte enables on their buses (*data* and *data_be* bus).

Write cycles' timing diagrams are shown by Figure 5.

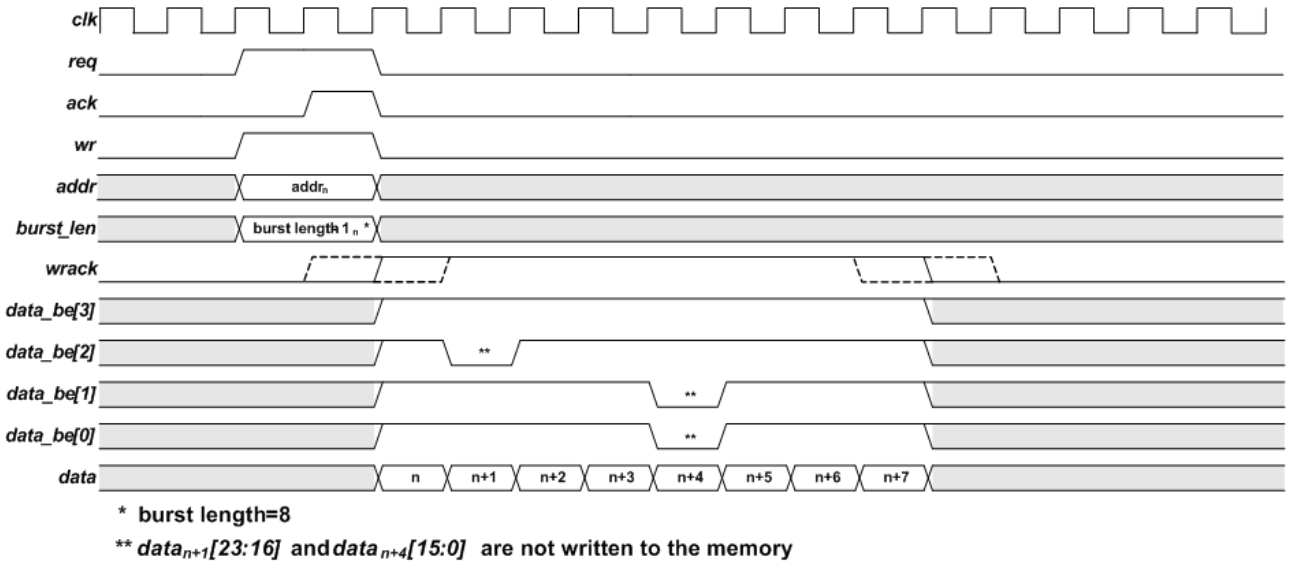

Figure 5: Write Cycle timing diagram

Address Alignment

Xylon Memory Bus uses byte addressing. System address must be aligned to a word boundary and the address word's size must equal to the system data bus width. Unused address bits are ignored (please see Figure 6, Figure 7 and Figure 8).

For 16-bit wide DDR SDRAM device the system address will be converted as it is shown in Figure 6.

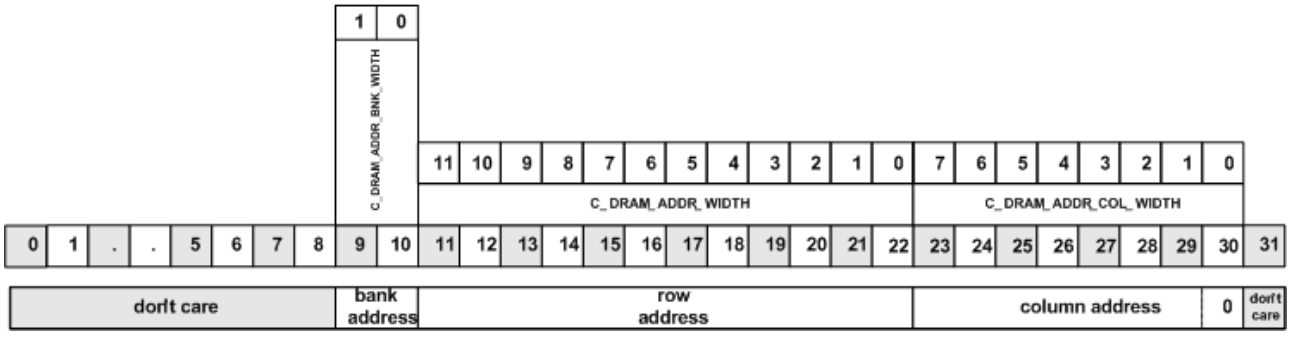

Figure 6: System interface address conversion for a 16-bit memory data width

For 32-bit wide SDR/DDR SDRAM device the system address will be converted as it is shown in Figure 7.

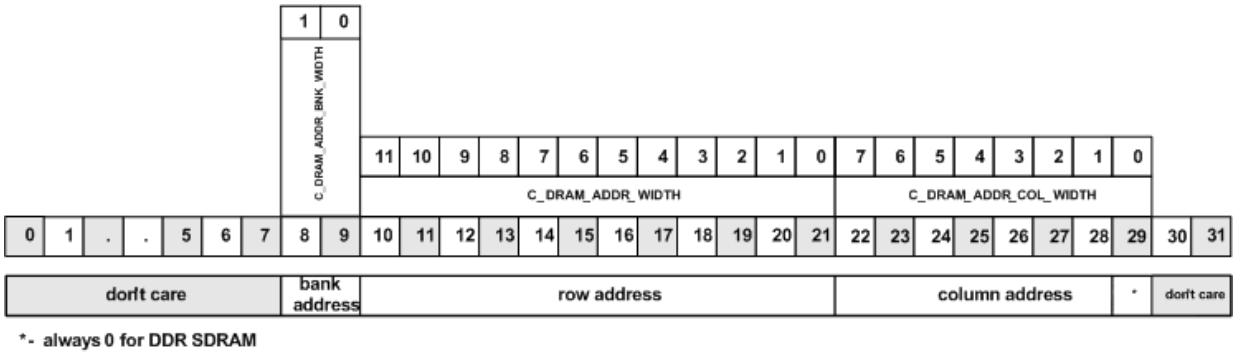

Figure 7: System interface address conversion for a 32-bit memory data width

For 64-bit wide SDR SDRAM device the system address will be converted as it is shown in Figure 8.

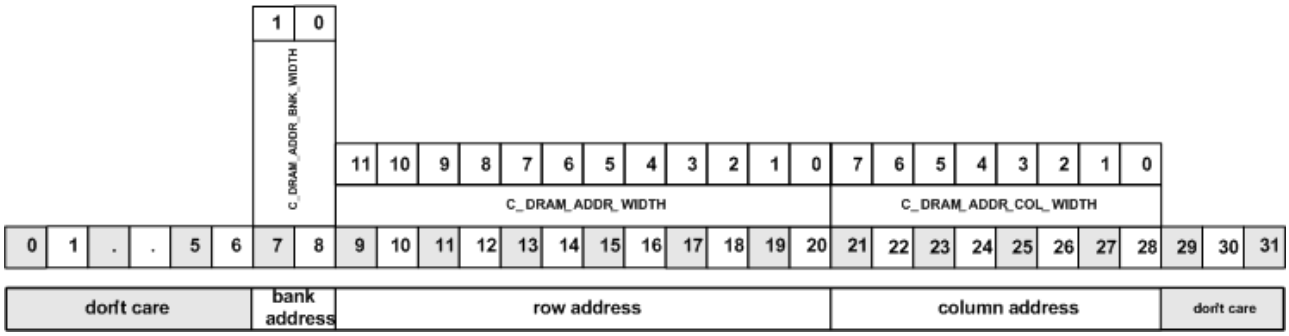

Figure 8: System interface address conversion for a 64-bit memory data width

Revision History

Version	Date	Note
1.00	18.06.2004.	Initial release.
2.00	03.05.2010.	Updated, new document template

Xylon d.o.o.
Fallerovo setaliste 22,
10000 Zagreb, Croatia
www.logicbricks.com