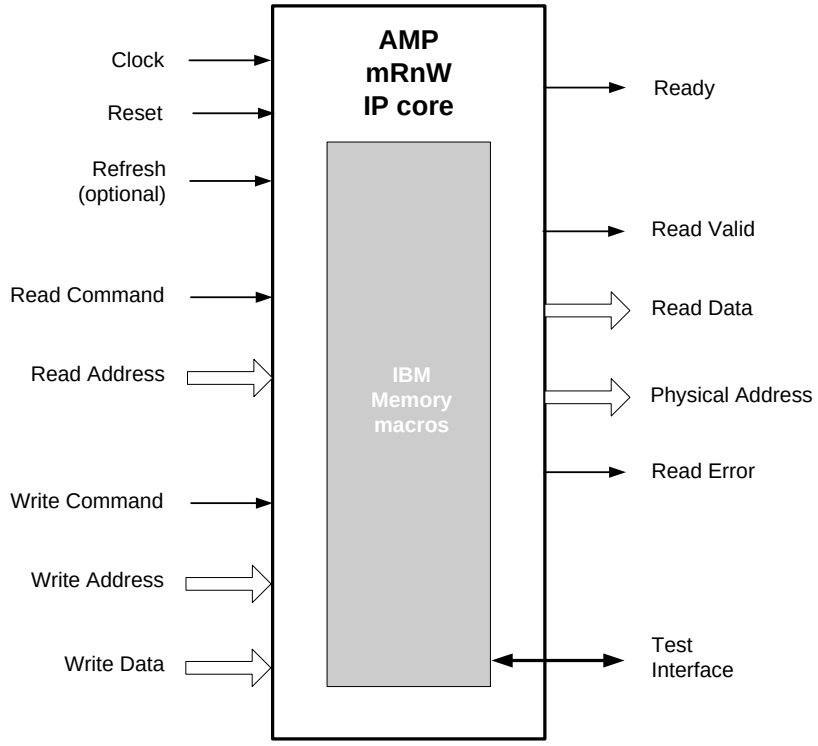
Multiport (mRnW) Algorithmic Memory Core

This datasheet gives an overview of a generic 'mRnW' memory core, a multi-port read-write core with 'm' read-ports and 'n' write-ports. The mRnW Algorithmic memory core wraps RTL-IP around standard IBM physical memories to support multi-port functionality. This datasheet covers cores with read and write port counts up to a maximum of four. Table 1 lists the various combinations of 'm' and 'n' currently offered. Higher port count cores are available upon customer request.

The core is offered as a compiler-based soft core in which memory depth and width are specified by the user. Specific information for a memory instance – Address/Data bus-widths, Clock frequency, Area, Power, Read Latency, Refresh scheme details (only for eDRAM based cores) – is made available to the user in a separate datasheet, auto-generated by the compiler.

Figure 1. Block Diagram



Functional Summary

- Parameterized number of address words and data widths.
- Master Clock synchronized, pipelined SRAM-like random-access interface.
- Unidirectional data-input bus and data-output bus.
- Pass-through test interface tounderlying memories, as specified in IBM memory databooks.

Table 1: AMP IP Core Types

Core Designation	Physical Read Ports	Physical Write Ports	Notes
4Ror1W[1]	4	1	Reads and write are mutually exclusive.
4R4W	4	4	Reads and writes can be simultaneous.
4R1W	4	1	(same as above)
3Ror1W[1]	3	1	Reads and write are mutually exclusive.
3R1W	3	1	Reads and writes can be simultaneous.
2Ror1W[1]	2	1	Reads and write are mutually exclusive.
2R4W	2	4	Reads and writes can be simultaneous.
2R3W	3	3	(same as above)
2R2W	2	2	(same as above)
2R1W	2	1	(same as above)
2RW	2*	2*	*Dual port read/write memory
1R4W	1	4	Reads and writes can be simultaneous
1R3W	1	2	(same as above)
1R2W	1	3	(same as above)
1R1W[1]	1	1	Read and write operations can occur simultaneously.
1R1RW	1	1*	Port 1 is read only; Port 2 is either read or write in a given cycle
1RW1W	1	1*	Port 1 is read or write in a given cycle; Port 2 is write only
1RW[1]	1*	1*	*Single port read/write memory

[1]These cores can be built with either SRAM or eDRAM as base memories. All other cores are built with SRAM only

Interface Description

Figure 2 shows the pin-interface diagram of a sample 4R4W core. For compactness, subscripts are appended to the port signal names to imply four independent ports. Subscripts {0,1,2,3}imply four independent read ports numbered port-0, port-1, port-2, port-3. Similarly subscripts {4,5,6,7} imply four independent write ports numbered port-4, port-5, port-6, port-7. This nomenclature is followed throughout this document. The interface pins are described in

Table *2*.

Figure 2.mRnW Signal Interface

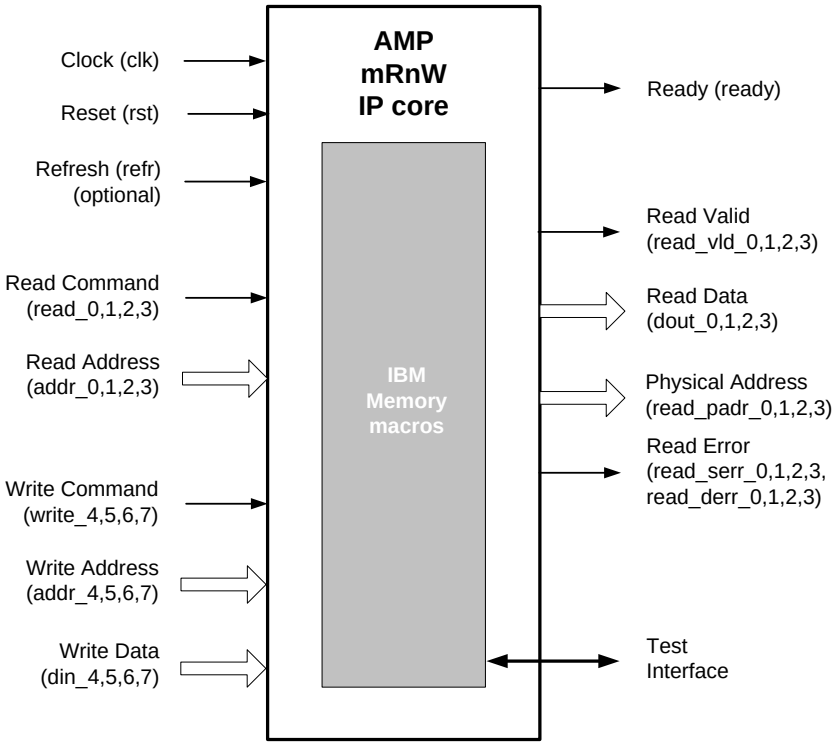


Table 2. Interface pins

Function	Signal	Count	Description	Input/Output
Clock and Initialization				
Clock	`clk`	1	Master clock input. All ports run synchronous to this clock.	Input
Reset	`rst`	1	Reset the mRnW core. Initialize the base memories according to the IBM specification before asserting Reset to the mRnW core.	Input
Ready	`ready`	1	The mRnW core is ready for operation. When Ready is asserted high, it stays high until a Reset or other interrupting event (like a two-bit ECC error) occurs. When Ready is negated low, a Reset is required to reassert Ready high.	Output
Refresh	`refr`	1	A refresh command is valid. (Only applicable for cores with eDRAM as base memories)	Input
Read Port Signals				
Read Address[2]	`addr_{0,1,2,3}`	1 to N	Read address bits (width *N* bits) to denote up to 2^N addresses. N is a parameter.	Input
Read Command	`read_{0,1,2,3)`	1	A read command is valid.	Input
Read Data	`dout_{0,1,2,3}`	1 to W	Data associated with the Read Address. The Read Data and the Read Data Valid signals appear after a fixed latency. W is a parameter.	Output
Read Data Valid	`read_vld_{0,1,2,3}`	1	Valid read data.	Output
Read Error	`read_serr_{0,1,2,3}`, `read_derr_{0,1,2,3}`	2	A read data error occurred. The read data is invalid.	Output
Physical Address	`read_paddr_{0,1,2,3}`	M	Physical address bits associated with read data. M is a parameter. There can be up to 2^M physical addresses.	Output
Write Port Signals				
Write Command	`write_{4,5,6,7}`	1	A write command is valid.	Input
Write Address[3]	`addr_{4,5,6,7}`	1 to N	Write address bits (width *N* bits). Parameter *N* is the same as that used for Read Address. N is a parameter.	Output
Write Data	`din_{4,5,6,7}`	1 to W	Data associated with the Write Address. W is a parameter.	Output
Test Interface				
Test Interface			Same as described in IBM databooks	

The different interfaces are described next.

[2] For 'n' RW cores (n=1,2,3,4), the port numbering is same for read and write ports. Further, the address busses are common. E.g. for a 2RW core, only *addr_0*, *addr_1* are offered. The other signals are *read_0,1, write_0,1, dout_0,1*, *paddr_0,1* and *din_0,1*.

[3] For cores with less than 4 read ports, the write port numbering begins where the read port numbering ends. E.g. for a core with two read ports and one write port, the numbering scheme would be read_0, read_1, write_2.

- **Clock and Initialization pins:** This set of signals is common to all ports. Clock is provided to the core via the clk input pin. The frequency is user determined. The clock pin is common to all the ports indicating full synchronous operation across the core (Multi-clock domain operation across ports is currently not supported).

  The core is initialized by asserting the reset pin (*rst*) for a minimum of ten clocks. The completion of reset process is indicated by assertion of the ready pin. The mRnW core is ready for normal operations on the next clock after this assertion.

- **Read Interface:** On each port, the interface accepts a Read command (read_{0,1,2,3}) and a corresponding Read address (*addr*_{0,1, 2,3}). The read data (dout_{0,1,2,3}) is available after a fixed latency along with the read valid signal (*read_vld*_{0,1,2,3}). The latency of a read access is fixed irrespective of operations on other ports. The *read_vld* signal is provided as a timing signal to enable the host chip to accept valid data. This way the host chip does not have to hardwire the latency value in its logic.

  Read Error: The assertion of read error signal indicates detection of a parity error in the internal data structures implemented in the core. Their occurrences are reported with the assertion of *read_serr (single-bit error),* or *read_derr (double-bit error)* outputs. The read physical address (described next) points to the memory location where the error occurred. The user is required to reset the memory core and initiate the reset process. If the error was 'soft' in nature then normal operations should ensue after reset. If errors repeat then user could choose to do BIST testing via pass-through test interface.

  Read error events happen only for certain memory cores, depending on the underlying algorithm. The core-specific datasheet states if the Read error outputs are used for that core. If not, these outputs are held inactive (low) throughout the course of operation.

  Read Physical Address: ECC implementation on the user data is the responsibility of the user. The user can choose to do this as part of data itself. Due to address mapping schemes deployed within the core, the physical location of data for a given user read address may change from access to access. The physical address of the read location (*read_padr*_{0,1,2,3}) is provided to help with address logging in case the user chooses to maintain statistics on ecc error events. The mapping between the physical address and physical memories (reference-designators) is detailed in the memory core-specific datasheet.

  Note: The read Physical address is mainly used during normal operation to help the user track any ecc error events seen in the data. But for those cores where a Read error event can happen, the physical address also points to the erring memory location during such an event.

- **Write Interface:** The interface accepts a write command (write_{4,5,6,7}), a write address (addr_{4,5,6,7}) and the corresponding write data (din_{4,5,6,7}).

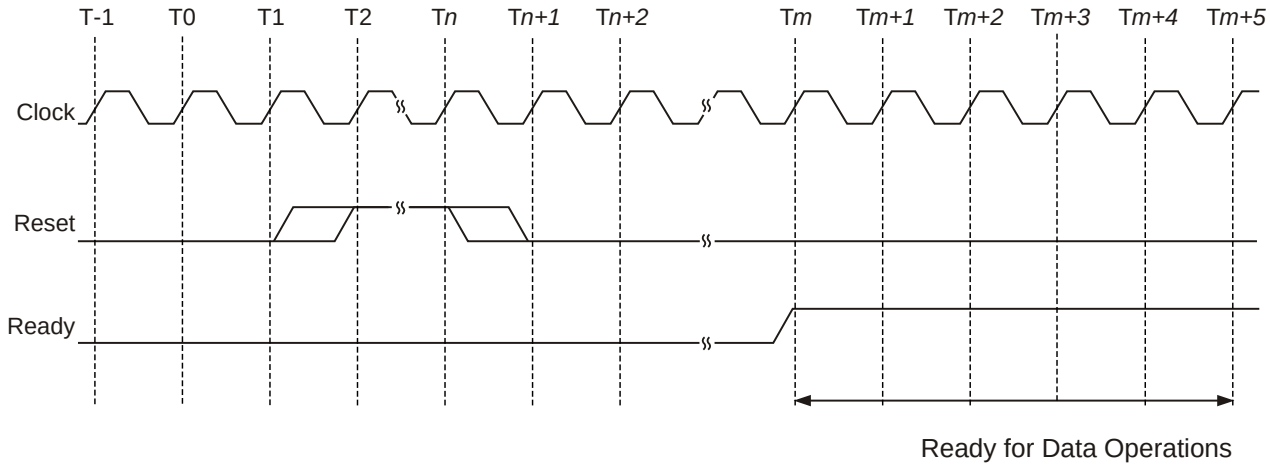
- **Test I/F:** This set of signals is common to all ports. AMP cores support all Test and Diagnostics features offered by IBM. All the associated pins are direct passed-thru to the base memories from the Test Interface.

Timing Diagrams

Figure 3 through Figure 6show timing diagrams for the mRnW core. For timings on the Test Interface, see the IBM memory databooks. The port signal names in the read and write timing diagrams represent signals for all the four read ports and write ports resp.

Initialization

Figure 3. Reset Timing



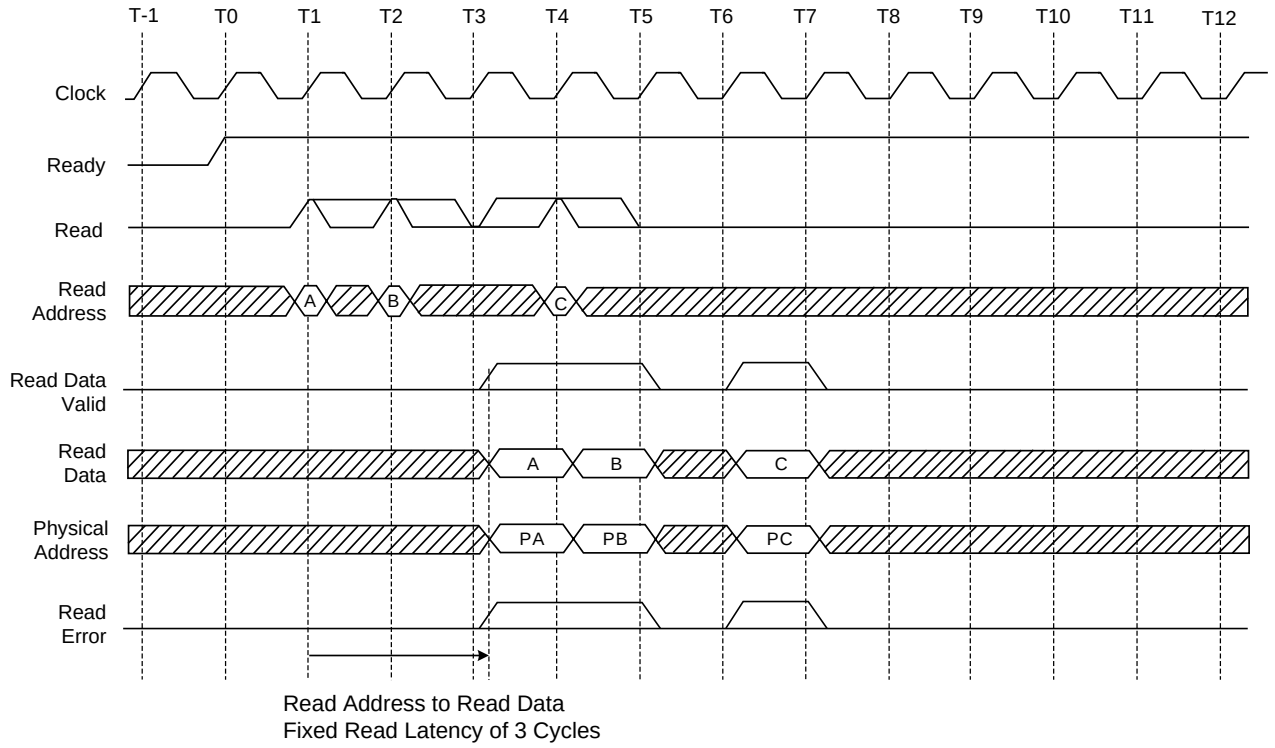
All inputs should stay at valid signal levels and not switch unnecessarily.
Addresses are random when shown in crosshatched pattern.

The reset sequence presented in Figure 3 brings the mRnW core out of reset. The reset pin is asserted for a minimum of 10 clock cycles.

The internal reset process begins after the de-assertion of *reset*. It completes a fixed time delay later. The completion is indicated by the assertion of *ready* pin. The reset to ready delay depends on the size of core memory and the algorithm implemented. Its value is specified in the memory core-specific datasheet. The host logic should wait until ready is asserted. The mRnW core is ready to accept data operations on the next clock after ready is asserted. The *ready* pin is kept asserted throughout the operation of the core. This is shown in the timing diagrams for read and write operations in the following sections.

Read Timing

Figure 4. Read Timing



All inputs should stay at valid signal levels and not switch unnecessarily.
Addresses are random when shown in crosshatched pattern.

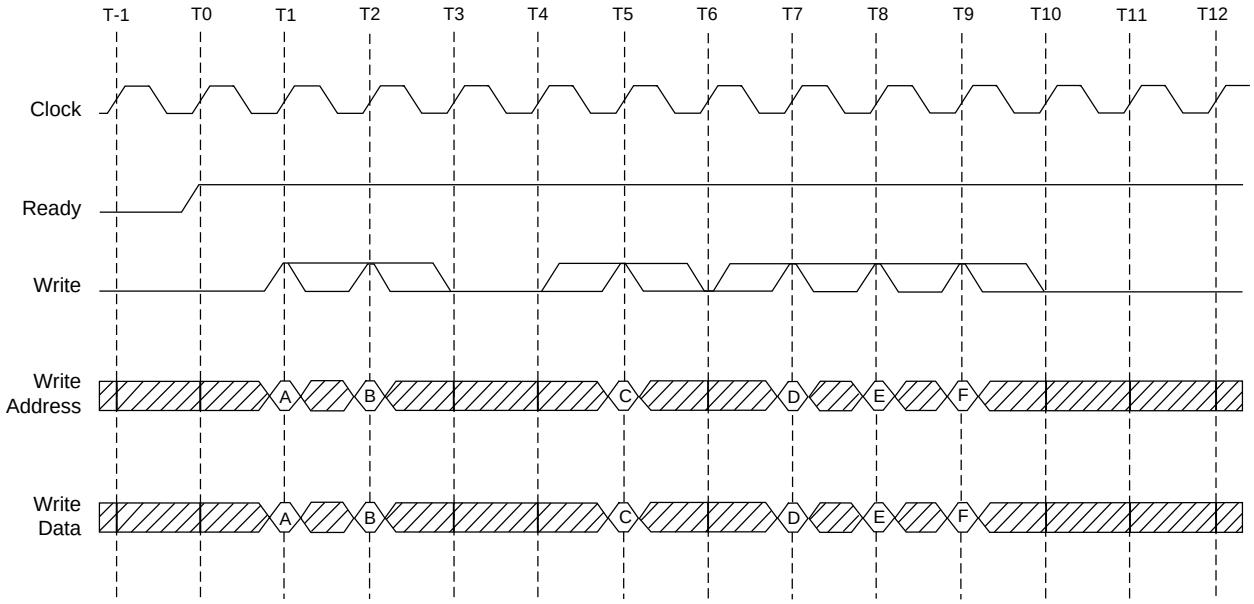
The timing diagram in Figure 4 shows three read requests - A, B and C respectively. The first read operation reads address A in clock cycle 1. Data for A is returned in clock cycle 3. Similar operations happen for addresses B and C in clock cycle 2 and 4 resp. Data for B is returned in clock cycle 4 while data for C is returned in clock cycle 6. The three read ports operate independently of each other. The signal names in the figure represent signals for ALL the read ports of the memory core.

A read latency of '3' clocks is assumed in the illustrations above. The latency of a given memory core is always fixed, irrespective of port operations on other ports. It depends on the inherent latencies of the base-memories and the algorithm type implemented. The validity of data for requests A, B, and C is indicated by the assertion of *read_vld* in clock cycles 3, 4 and 6.

The mRnW memory core does NOT provide error-protection for user data. The user is responsible for incorporating any error-detection/correction schemes as part of the data itself. The physical address (*read_paddr_{0,1,2,3}*) output is provided to help the user with any address/data logging for ECC statistics maintenance on user data. The mapping between physical address and the underlying physical memories is provided in the memory core-specific datasheet. The physical addresses (PA, PB, and PC) for each read request are returned alongside their data output – in cycles 3, 4 and 6 respectively.

Write Timing

Figure 5. Write Timing



All inputs should stay at valid signal levels and not switch unnecessarily.

The Write timing diagram in Figure 5shows six write requests for addresses – A, B, C, D, E, and F. The first write operation writes to address A in clock cycle 1. Data DA is written. The second write operation writes to address B in clock cycle 2. Data DB is written. Similarly, the write operations to addresses C, D, E, and F are done in clock cycles 5, 7, 8, 9 respectively, and the corresponding data DC, DD, DE, and DF are written.

Like the read ports, the signal names in the diagram imply simultaneous operation of all the four write ports, independent of each other.

Read and Write Interaction

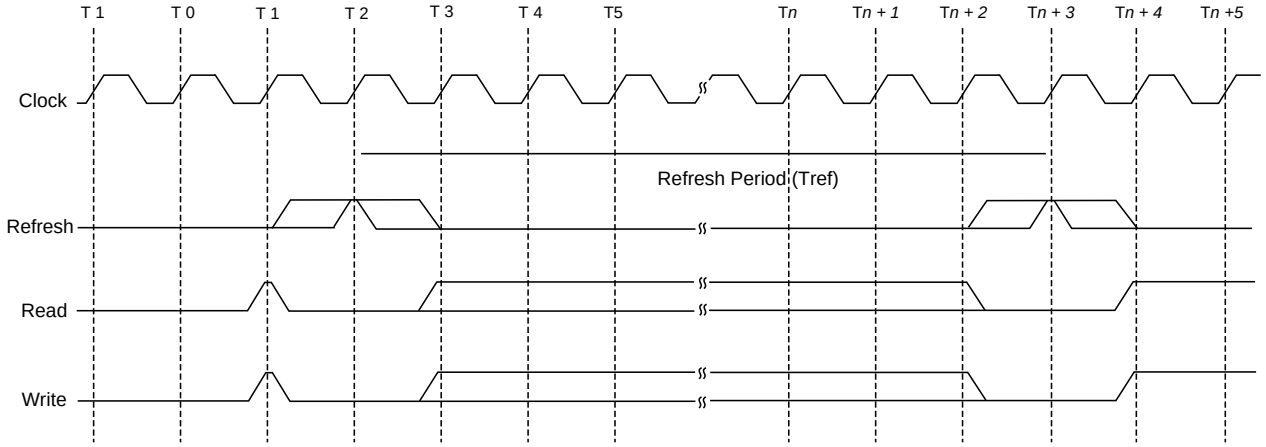
The read and write interfaces operate independently of each other unless the core is of type ‘mRornW’. For these cores the read operations are mutually exclusive with write operations.

When independent operation of read and write is allowed (core type mRnW) and if a read and a write transaction occur to the same address in the same clock cycle, the old value associated with the read address is returned on *dout* and the new value on the write data bus is posted to the address. In this sense, the read is considered to be executed before the write.

For memory cores with multiple write ports, when simultaneous writes to the same address from multiple ports occur then the data from the highest port number gets written to that location.

Refresh Timing

Figure 6. Refresh Timing



All inputs should stay at valid signal levels and not switch unnecessarily.

This section is optional. It applies to only those cores which are built with eDRAMs as base memories. Please refer to Table 1for the applicable core types.

Figure 6 shows the refresh operation for a One-in-N refresh scheme. Other refresh schemes for an eDRAM based core are possible. If implemented, they would be specific to a given core implementation and will be described in the core-specific datasheet.

A refresh command to the mRnW core initiates a refresh operation on the eDRAM instantiated inside. The refresh-controller implemented in the core manages the refresh bank address sequencing and issues it internally on the eDRAM's address bus. The user is NOT required to issue any addresses externally at the core interface.

The only requirement on the user's side is to provide one refresh command within the Tref window. A variation of this scheme is to provide a minimum number of refresh commands (Nminref) within the Tref window. Nminref depends on the size of the instantiated eDRAM core. The values of Nminref and Tref will be specified in the core-specific datasheet.

The user should NOT assert either the read or write commands on any port when *refr* is asserted. Doing so will result in erroneous read or write operation.