Memory Usage Guide for MachXO Devices

October 2010 Technical Note TN1092

Introduction

This technical note discusses memory usage for the Lattice MachXO device family. It is intended to be used by design engineers as a guide in integrating the EBR and PFU based memories for these device families in ispLEVER® and Lattice Diamond™ design software.

The architecture of these devices provides resources for memory intensive applications. The sysMEM™ Embedded Block RAM (EBR) complements its distributed PFU-based memory. Single-Port RAM, Dual-Port RAM, Pseudo Dual-Port RAM, FIFO and ROM memories can be constructed using the EBR. LUTs and PFU can implement Distributed Single-Port RAM, Dual-Port RAM and ROM.

The capabilities of the EBR Block RAM and PFU RAM are referred to as primitives and are described later in this document. Designers can utilize the memory primitives in two ways:

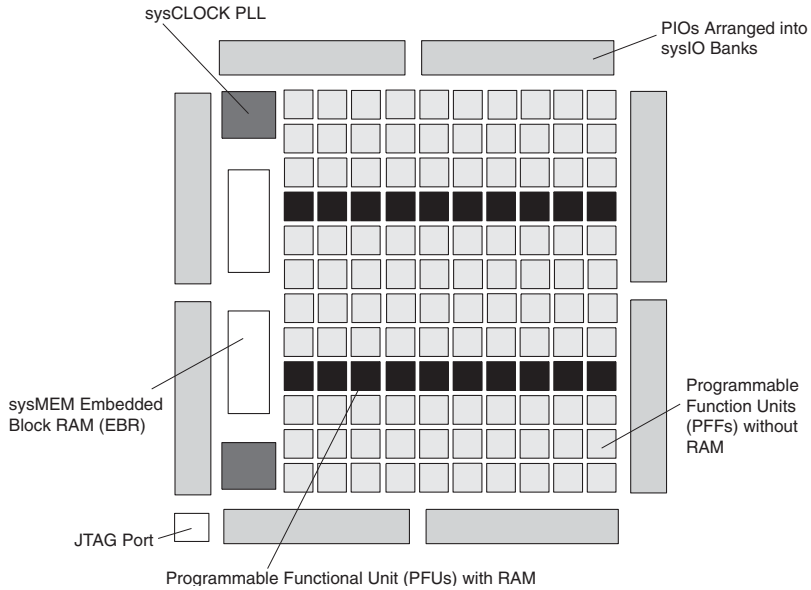
- Via **IPexpress™** – The IPexpress GUI allows users to specify the memory type and size that is required. IPexpress takes this specification and constructs a netlist to implement the desired memory by using one or more of the memory primitives.
- Via the **PMI (Parameterizable Module Inferencing)** – PMI allows experienced users to skip the graphical interface and utilize the configurable memory modules on the fly from the ispLEVER Project Navigator or Diamond. The parameters and the control signals needed either in Verilog or VHDL can be set. The top-level design will have the parameters defined and signals declared so the interface can automatically generate the black box during synthesis.

The remainder of this document discusses these approaches, utilizing IPexpress, PMI inference, memory modules and memory primitives.

MachXO Device Memories

Only the MachXO1200 and MachXO2280 devices contain the sysMEM EBR blocks along with an array of logic blocks called PFUs (or PFFs) surrounded by Programmable I/O Cells (PICs). This is shown in Figure 9-1.

Figure 9-1. Logical View of MachXO1200 and MachXO2280 Devices



The PFU contains the building blocks for logic and Distributed RAM and ROM. The PFF provides the logic building blocks without the distributed RAM This document describes the memory usage and implementation for both Embedded Memory Blocks (EBRs) and Distributed RAM of the PFU. Refer to the MachXO Family Data Sheet for details on the hardware implementation of the EBR and Distributed RAM.

The logic blocks are arranged in a two-dimensional grid with rows and columns as shown in the figures below. The physical location of the EBR and Distributed RAM follows the row and column designation. Since the Distributed RAM is part of the PFU resource, it follows the PFU/PFF row and column designation. The EBR occupies two columns per block to account for the wider port interface.

Utilizing IPexpress

Designers can utilize IPexpress to easily specify a variety of memories in their designs. These modules will be constructed using one or more memory primitives along with general purpose routing and LUTs as required. The available primitives are:

- Single Port RAM (RAM_DQ) – EBR-based
- Dual PORT RAM (RAM_DP_TRUE) – EBR-based
- Pseudo Dual Port RAM (RAM_DP) – EBR-based
- Read Only Memory (ROM) – EBR-Based
- First In First Out Memory (Dual Clock) (FIFO_DC) – EBR-based
- Distributed Single Port RAM (Distributed_SPRAM) – PFU-based
- Distributed Dual Port RAM (Distributed_DPRAM) – PFU-based
- Distributed ROM (Distributed_ROM) – PFU/PFF-based
- RAM Based Shift Register (RAM_Based_Shift_Register) – PFU-based
- Distributed Shift Register (RAM_Based_Shift_Register) - PFU based (see IPexpress Help for details)

IPexpress Flow

For generating any of these memories, create (or open) a project for the MachXO devices.

From Diamond or the ispLEVER Project Navigator, select **Tools > IPexpress**. Alternatively, users can also click on the button in the toolbar shown below when the MachXO devices are targeted in the project.

This opens the IPexpress window as shown in Figure 9-2.

Figure 9-2. IPexpress Main Window, ispLEVER

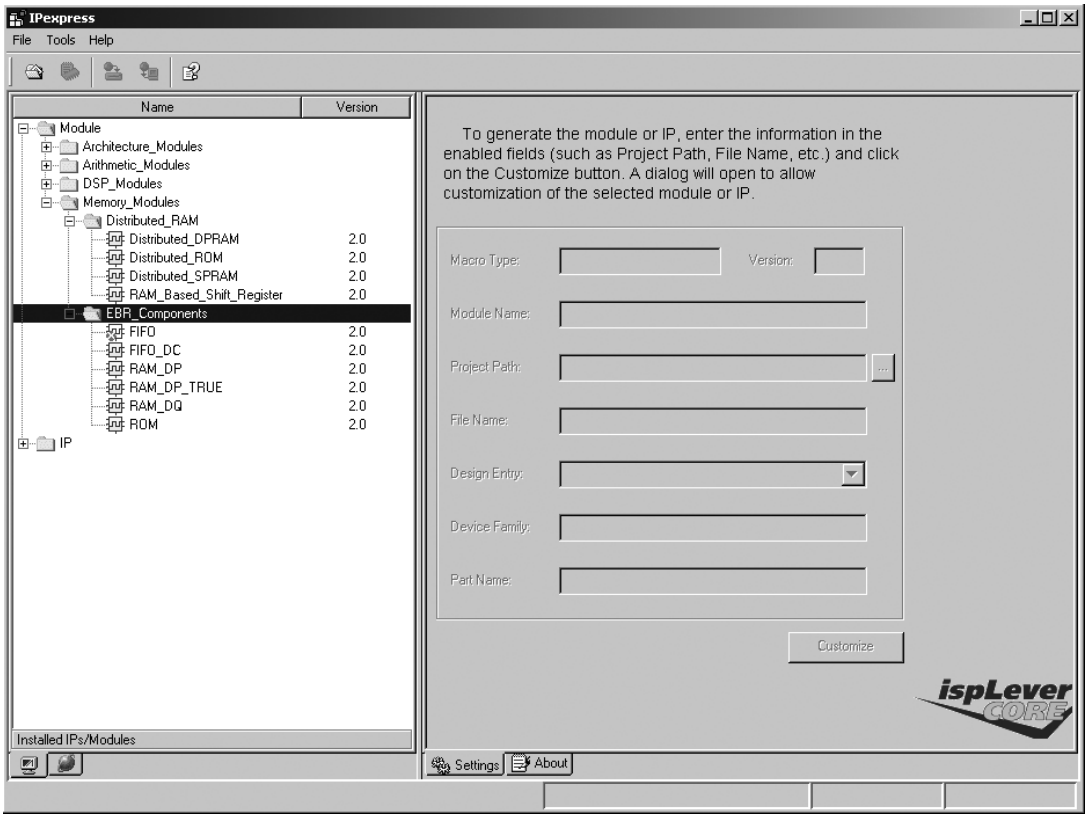
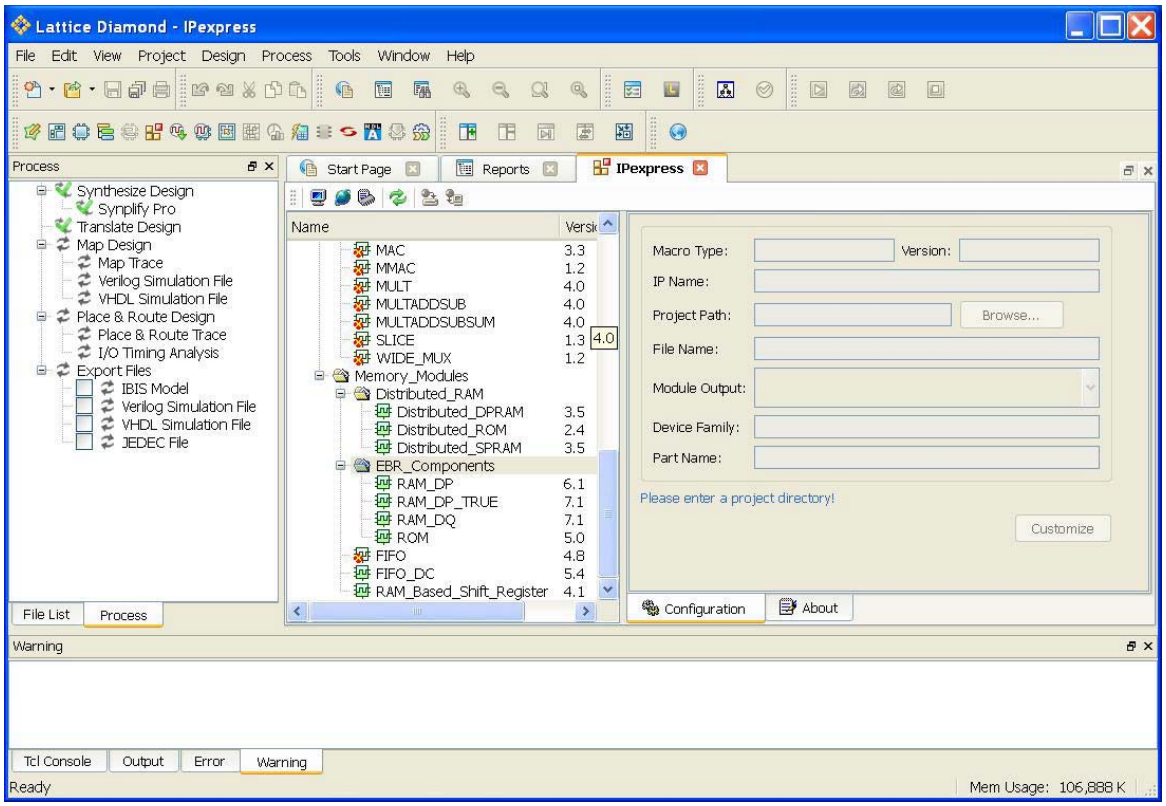


Figure 9-3. IPexpress Main Window, Diamond



The left pane of this window includes the Module Tree. The EBR-based Memory Modules are under the **EBR_Components** and the PFU-based Distributed Memory Modules are under **Storage_Components** as shown in Figure 9-2 and Figure 9-3.

As an example, let us consider generating an EBR-based Pseudo Dual Port RAM of size 512x16. Select RAM_DP under the EBR_Components. The right pane changes as shown in Figure 9-4 and Figure 9-5.

Figure 9-4. Generating Pseudo Dual Port RAM (RAM_DP) Using IPexpress in ispLEVER

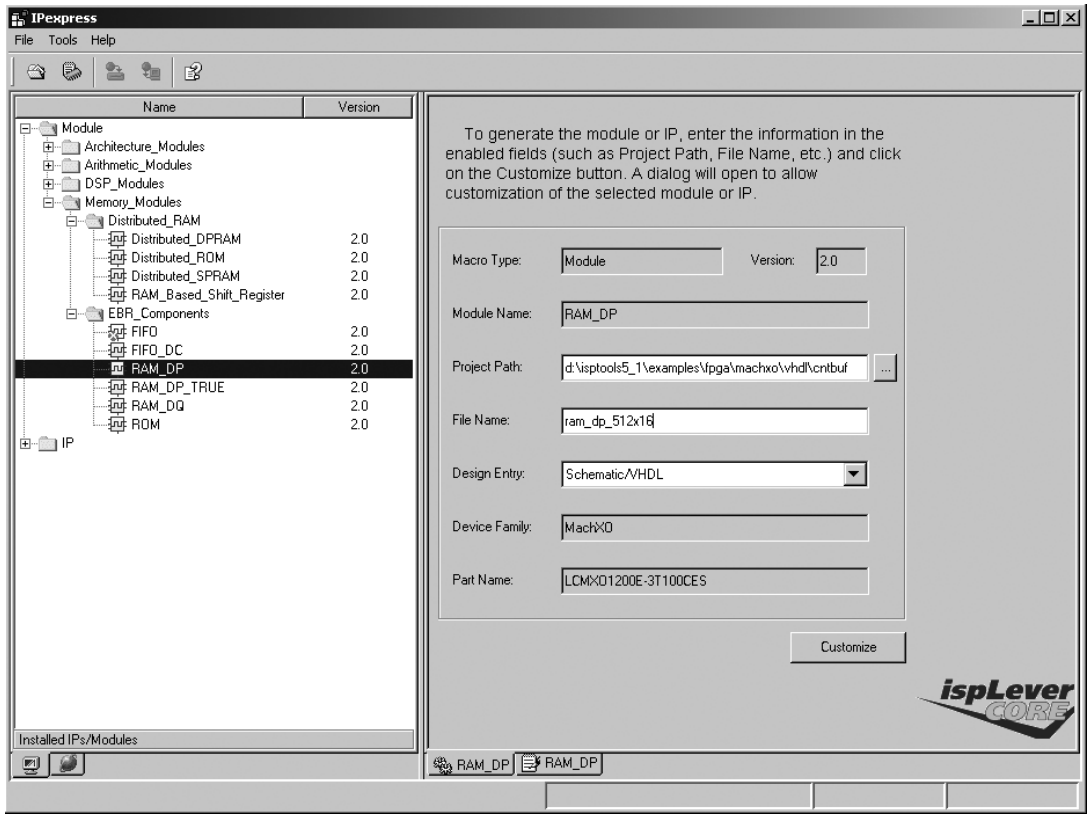
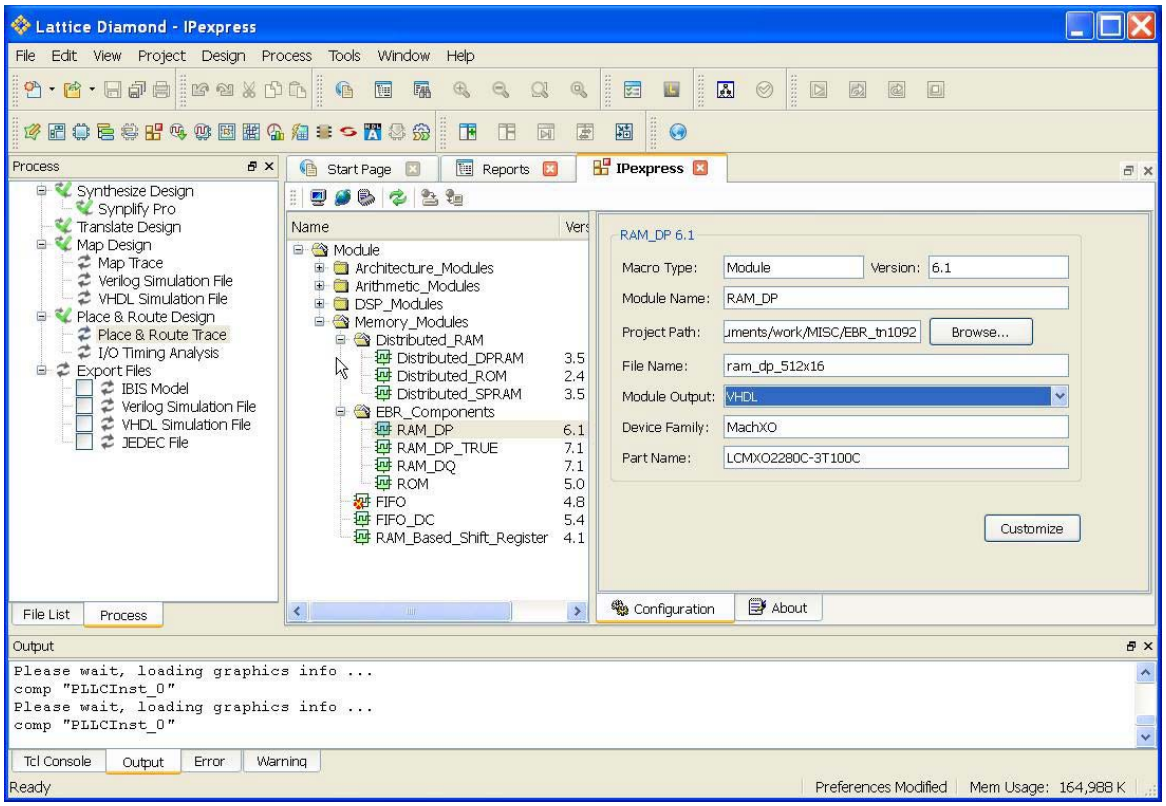


Figure 9-5. Generating Pseudo Dual Port RAM (RAM_DP) Using IPexpress in Diamond



In this right pane, options like the **Device Family**, **Macro Type**, **Category**, and **Module_Name** are device and selected module dependent. These cannot be changed in IPexpress.

Users can change the directory where the generated module files will be placed by clicking the **Browse** button in the **Project Path**.

The **Module Name** text box allows users to specify an entity name for the module they are about to generate. Users must provide this entity name.

Design entry, Verilog or VHDL, by default, is the same as the project type. If the project is a VHDL project, the selected design entry option will be “Schematic/ VHDL”, and “Schematic/ Verilog-HDL” if the project type is Verilog-HDL.

The **Device** pull-down menu allows users to select different devices within the same family, MachXO in this example.

Then click the **Customize** button.

This opens another window where users can customize the RAM (Figure 9-6 and Figure 9-7).

Figure 9-6. Generating Pseudo Dual Port RAM (RAM_DP) Module Customization in ispLEVER

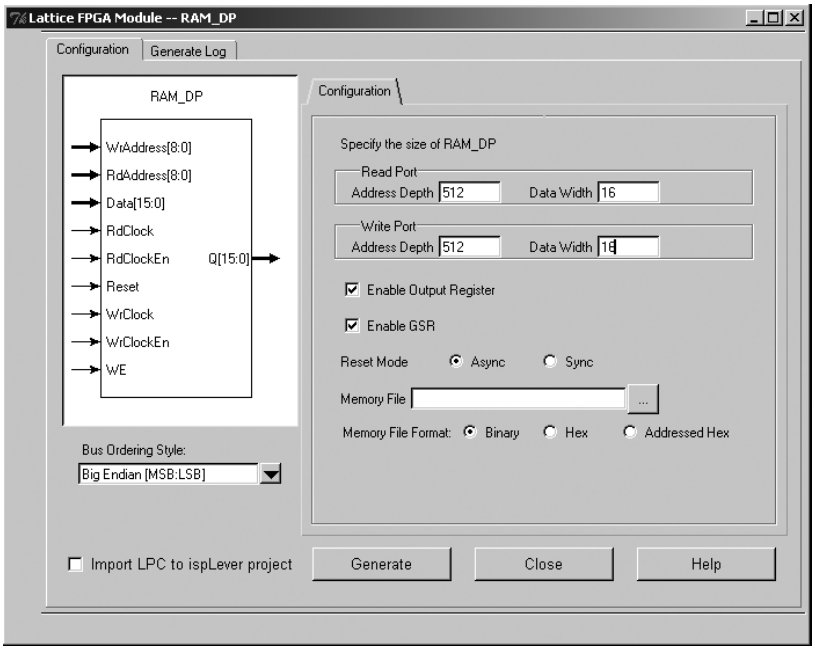
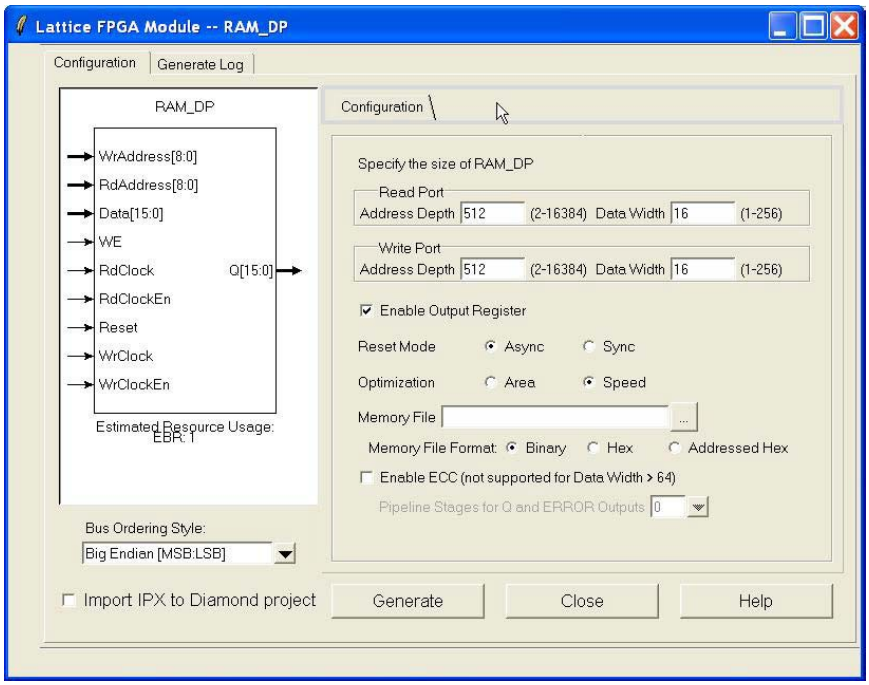


Figure 9-7. Generating Pseudo Dual Port RAM (RAM_DP) Module Customization in Diamond



The left side of this window shows the block diagram of the module. The right side includes the Configuration tab where users can choose options to customize the RAM_DP such as (e.g. specify the address port sizes and data widths).

Users can specify the address depth and data width for the **Read Port** and the **Write Port** in the text boxes provided. In this example we are generating a Pseudo Dual Port RAM of size 512 x 16. Users can also create the RAMs of different port widths in case of Pseudo Dual Port and True Dual Port RAMs.

The Input Data and the Address Control is always registered, as the hardware only supports the clocked write operation for the EBR based RAMs. The check box **Enable Output Registers**, inserts the output registers in the Read Data Port, as the output registers are optional for the EBR-based RAMs.

Users have the option to set the **Reset Mode** as Asynchronous Reset or Synchronous Reset. **Enable GSR** can be checked to be enable the Global Set Reset.

Users can also pre-initialize their memory with the contents specified in the **Memory File**. It is optional to provide this file in the RAM; however for ROM, the Memory File is required. These files can be of Binary, Hex or Addresses Hex format. The details of these formats are discussed in the Initialization File section of this document.

At this point, users can click the **Generate** button to generate the module they have customized. A VHDL or Verilog netlist is then generated and placed in the specified location. Users can incorporate this netlist in their designs.

Another important button is the **Load Parameters** button. IPexpress stores the parameters the user has specified in an <module_name>.lpc file. This file is generated along with the module. Users can click on the Load Parameter button to load the parameters of a previously generated module to re-visit or make changes to them.

Once the module is generated, user can either instantiate the *.lpc or the Verilog-HDL/ VHDL file in top-level module of their design.

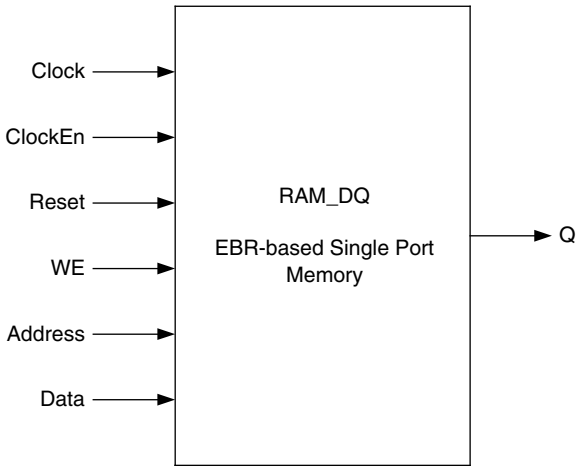
The various memory modules, both EBR and Distributed, are discussed in detail in this document.

Memory Modules

Single Port RAM (RAM_DQ) – EBR Based: The EBR blocks in the MachXO devices can be configured as Single Port RAM or RAM_DQ. IPexpress allows users to generate the Verilog-HDL or VHDL along EDIF netlist for the memory size, as per design requirements.

IPexpress generates the memory module as shown in Figure 9-8.

Figure 9-8. Single Port Memory Module Generated by IPexpress



Since the device has a number of EBR blocks, the generated module makes use of these EBR blocks, or primitives, and cascades them to create the memory sizes specified by the user in the IPexpress GUI. For memory sizes smaller than an EBR block, the module will be created in one EBR block. For memory sizes larger than one EBR block, multiple EBR blocks can be cascaded in depth or width (as required to create these sizes).

In Single Port RAM mode, the input data and address for the ports are registered at the input of the memory array. The output data of the memory is optionally registered at the output.

The various ports and their definitions for the Single Port Memory are listed in Table 9-1. The table lists the corresponding ports for the module generated by IPexpress and for the EBR RAM_DQ primitive.

Table 9-1. EBR-based Single Port Memory Port Definitions

Port Name in the Generated Module	Port Name in the EBR Block Primitive	Description	Active State
Clock	CLK	Clock	Rising Clock Edge
ClockEn	CE	Clock Enable	Active High
Address	AD[x:0]	Address Bus	—
Data	DI[y:0]	Data In	—
Q	DO[y:0]	Data Out	—
WE	WE	Write Enable	Active High
Reset	RST	Reset	Active High
—	CS[2:0]	Chip Select	—

Reset (or RST) resets only the input and output registers of the RAM. It does not reset the contents of the memory.

Chip Select (CS) is a useful port in the EBR primitive when multiple cascaded EBR blocks are required by the memory. The CS signal forms the MSB for the address when multiple EBR blocks are cascaded. CS is a 3-bit bus, so it can cascade eight memories easily. If the memory size specified by the user requires more than eight EBR blocks, the ispLEVER or Diamond software automatically generates the additional address decoding logic which is implemented in the PFU (external to the EBR blocks).

Each EBR block consists of 9,216 bits of RAM. The values for x (address) and y (data) for each EBR block for the devices are listed in Table 9-2.

Table 9-2. Single Port Memory Sizes for 9K Memories in MachXO

Single Port Memory Size	Input Data	Output Data	Address [MSB:LSB]
8K x 1	DI	DO	AD[12:0]
4K x 2	DI[1:0]	DO[1:0]	AD[11:0]
2K x 4	DI[3:0]	DO[3:0]	AD[10:0]
1K x 9	DI[8:0]	DO[8:0]	AD[9:0]
512 x 18	DI[17:0]	DO[17:0]	AD[8:0]
256 x 36	DI[35:0]	DO[35:0]	AD[7:0]

Table 9-3 shows the various attributes available for the Single Port Memory (RAM_DQ). Some of these attributes are user selectable through the IPexpress GUI. For detailed attribute definitions, refer to Appendix A.

Table 9-3. Single Port RAM Attributes for MachXO

Attribute	Description	Values	Default Value	User Selectable through IPexpress
DATA_WIDTH	Data Word Width	1, 2, 4, 9, 18, 36	18	Yes
REGMODE	Register Mode (Pipelining)	NOREG, OUTREG	NOREG	Yes
RESETMODE	Selects Register Type	ASYNC, SYNC	ASYNC	Yes
CSDECODE	Chip Select Decode	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111 0b000	0b000	No
WRITEMODE	Read/Write Mode	NORMAL, WRITETHROUGH, READBEFOREWRITE	NORMAL	Yes
ENABLE GSR	Enable or Disable Global Set Reset	Enable (1), Disable (0)	1	Yes
INITVAL	Initialization Value	0x000000000000000000000000000000000000 0000000000000000000000000000000000000 0000000....0xFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF FFFFFFFFFFFFFFF FFFFFFF (80-bit hex string)	0x0000000 000000000 000000000 000000000 000000000 000000000 000000000 000000000 000 0000000	—

The Single Port RAM (RAM_DQ) can be configured as NORMAL, READ BEFORE WRITE or WRITE THROUGH modes. Each of these modes affects what data comes out of the port Q of the memory during the write operation followed by the read operation at the same memory location.

Additionally, users can select to enable the output registers for RAM_DQ. Figures 9-9 to 9-14 show the internal timing waveforms for the Single Port RAM (RAM_DQ) with these options.

Figure 9-9. Single Port RAM Timing Waveform - NORMAL Mode, without Output Registers

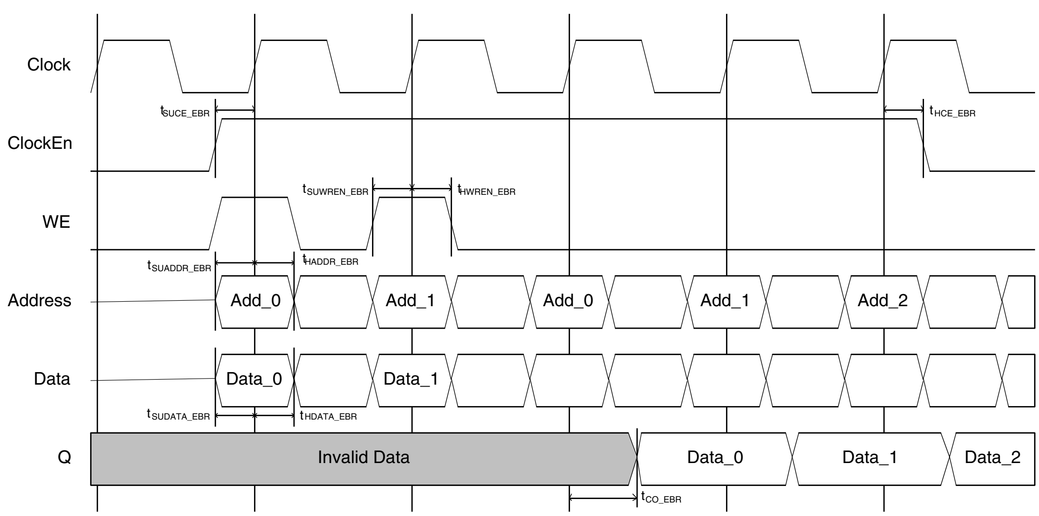


Figure 9-10. Single Port RAM Timing Waveform - NORMAL Mode, with Output Registers

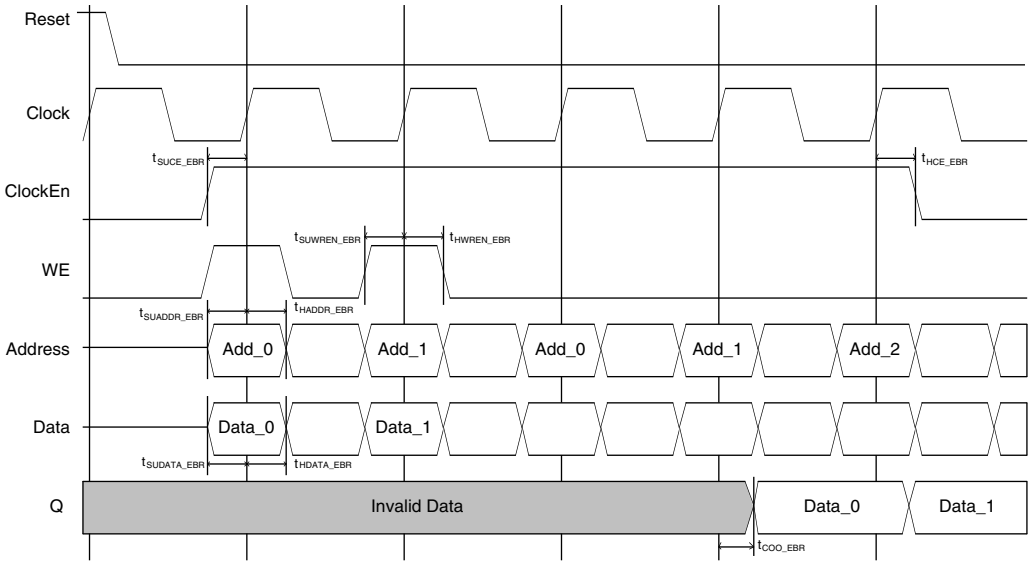


Figure 9-11. Single Port RAM Timing Waveform - READ BEFORE WRITE Mode, without Output Registers

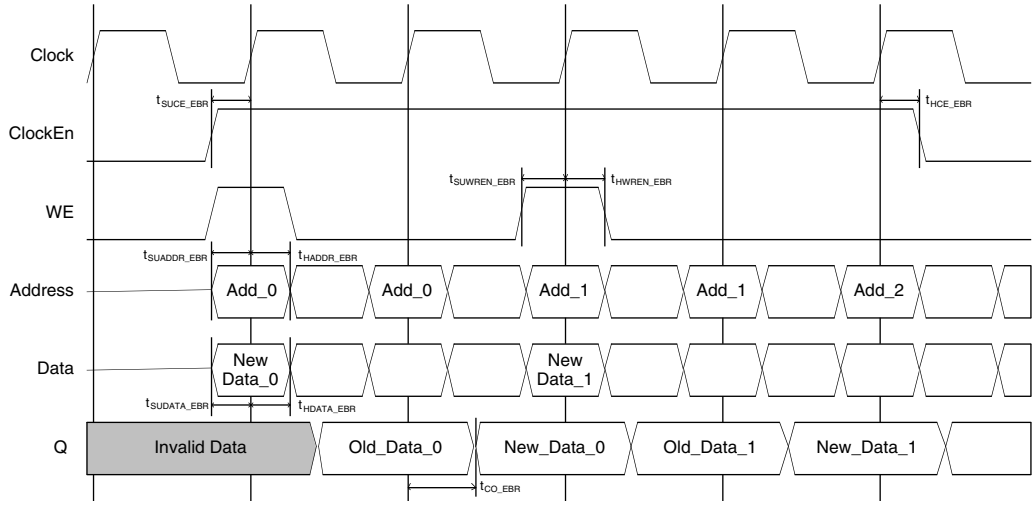


Figure 9-12. Single Port RAM Timing Waveform - READ BEFORE WRITE Mode, with Output Registers

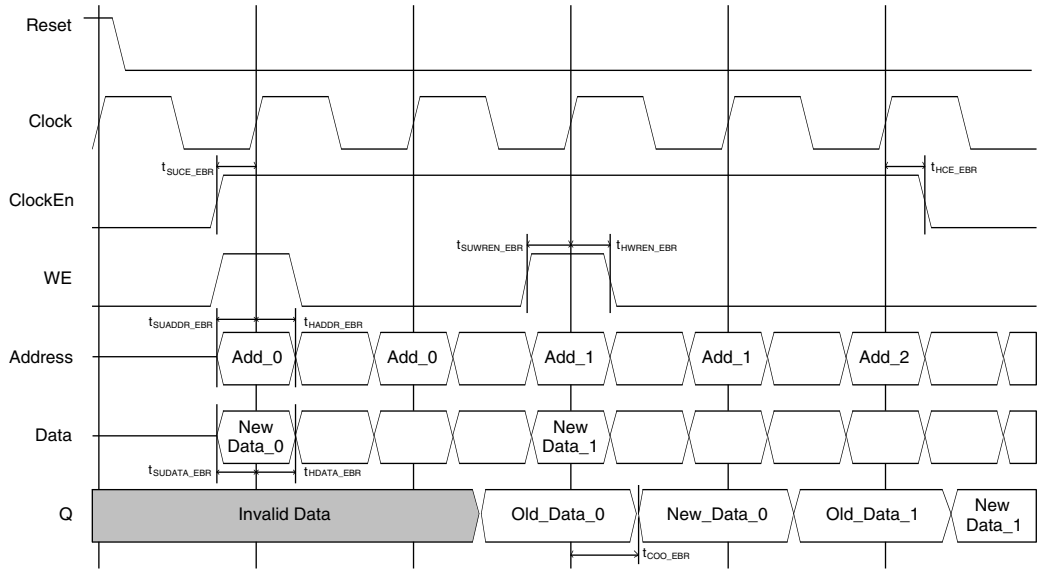


Figure 9-13. Single Port RAM Timing Waveform - WRITE THROUGH Mode, without Output Registers

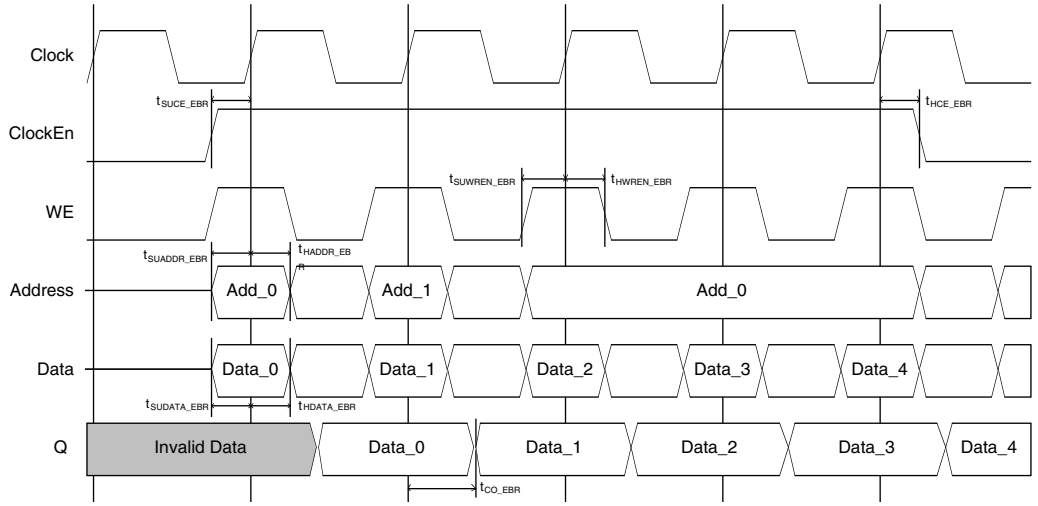
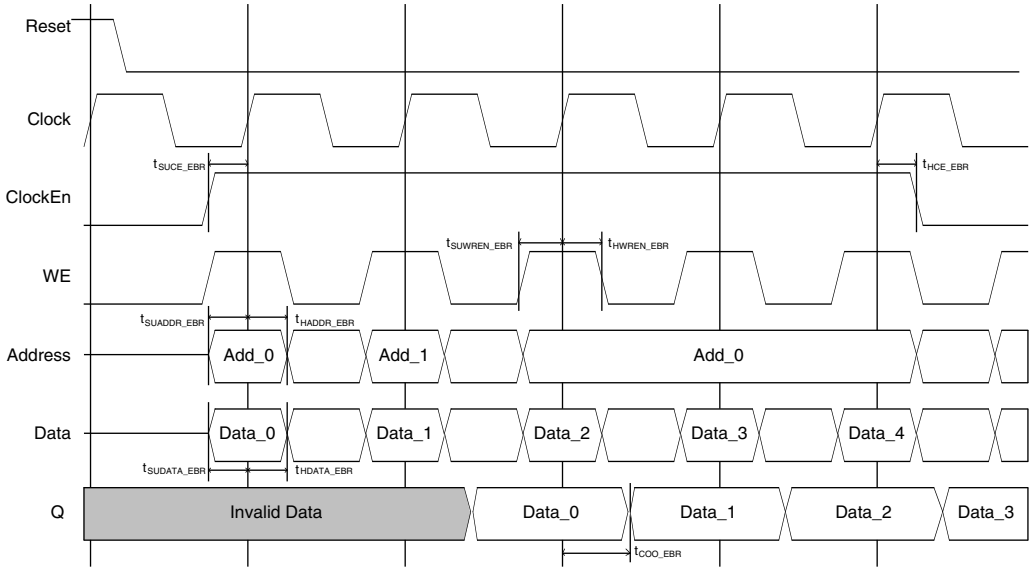


Figure 9-14. Single Port RAM Timing Waveform - WRITE THROUGH Mode, with Output Registers

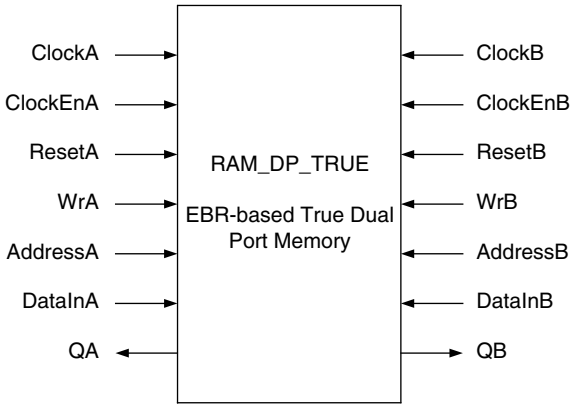


True Dual Port RAM (RAM_DP_TRUE) – EBR Based

The EBR blocks in MachXO devices can be configured as True-Dual Port RAM or RAM_DP_TRUE. IPexpress allows users to generate the Verilog-HDL, VHDL or EDIF netlists for various memory sizes depending on design requirements.

IPexpress generates the memory module as shown in Figure 9-15.

Figure 9-15. True Dual Port Memory Module generated by IPexpress



The generated module makes use of these EBR blocks or primitives. For memory sizes smaller than an EBR block, the module will be created in one EBR block. When the specified memory is larger than one EBR block, multiple EBR blocks can be cascaded in depth or width (as required to create these sizes).

In True Dual Port RAM mode, the input data and address for the ports are registered at the input of the memory array. The output data of the memory is optionally registered at the output.

The various ports and their definitions for Single Port Memory are in Table 9-4. The table lists the corresponding ports for the module generated by IPexpress and for the EBR RAM_DP_TRUE primitive.

Table 9-4. EBR based True Dual Port Memory Port Definitions

Port Name in Generated Module	Port Name in the EBR Block Primitive	Description	Active State
ClockA, ClockB	CLKA, CLKB	Clock for PortA and PortB	Rising Clock Edge
ClockEnA, ClockEnB	CEA, CEB	Clock Enables for Port CLKA and CLKB	Active High
AddressA, AddressB	ADA[x1:0], ADB[x2:0]	Address Bus Port A and Port B	—
DataA, DataB	DIA[y1:0], DIB[y2:0]	Input Data Port A and Port B	—
QA, QB	DOA[y1:0], DOB[y2:0]	Output Data Port A and Port B	—
WrA, WrB	WEA, WEB	Write enable Port A and Port B	Active High
ResetA, ResetB	RSTA, RSTB	Reset for PortA and PortB	Active High
	CSA[2:0], CSB[2:0]	Chip Selects for each port	—

Reset (or RST) resets only the input and output registers of the RAM. It does not reset the contents of the memory.

Chip Select (CS) is a useful port in the EBR primitive when multiple cascaded EBR blocks are required by the memory. The CS signal forms the MSB for the address when multiple EBR blocks are cascaded. Since CS is a 3-bit bus, it can cascade eight memories easily. However, if the memory size specified by the user requires more than eight EBR blocks, the ispLEVER or Diamond software automatically generates the additional address decoding logic, which is implemented in the PFU external to the EBR blocks.

Each EBR block consists of 9,216 bits of RAM. The values for x's (for address) and y's (data) for each EBR block for the devices are listed in Table 9-5.

Table 9-5. MachXO Dual Port Memory Sizes for 9K Memory

Dual Port Memory Size	Input Data Port A	Input Data Port B	Output Data Port A	Output Data Port B	Address Port A [MSB:LSB]	Address Port B [MSB:LSB]
8K x 1	DIA	DIB	DOA	DOB	ADA[12:0]	ADB[12:0]
4K x 2	DIA[1:0]	DIB[1:0]	DOA[1:0]	DOB[1:0]	ADA[11:0]	ADB[11:0]
2K x 4	DIA[3:0]	DIB[3:0]	DOA[3:0]	DOB[3:0]	ADA[10:0]	ADB[10:0]
1K x 9	DIA[8:0]	DIB[8:0]	DOA[8:0]	DOB[8:0]	ADA[9:0]	ADB[9:0]
512 x 18	DIA[17:0]	DIB[17:0]	DOA[17:0]	DOB[17:0]	ADA[8:0]	ADB[8:0]

Table 9-6 shows the various attributes available for the True Dual Port Memory (RAM_DP_TRUE). Some of these attributes are user-selectable through the IPexpress GUI. For detailed attribute definitions, refer to the Appendix A.

Table 9-6. MachXO Dual Port RAM Attributes

Attribute	Description	Values	Default Value	User Selectable through IPexpress
DATA_WIDTH_A	Data Word Width Port A	1, 2, 4, 9, 18	18	YES
DATA_WIDTH_B	Data Word Width Port B	1, 2, 4, 9, 18	18	YES
REGMODE_A	Register Mode (Pipelining) for Port A	NOREG, OUTREG	NOREG	YES
REGMODE_B	Register Mode (Pipelining) for Port B	NOREG, OUTREG	NOREG	YES
RESETMODE	Selects the Reset type	ASYNC, SYNC	ASYNC	YES
CSDECODE_A	Chip Select Decode for Port A	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	000	NO
CSDECODE_B	Chip Select Decode for Port B	000, 001, 010, 011, 100, 101, 110, 111	000	NO
WRITEMODE_A	Read / Write Mode for Port A	NORMAL, WRITETHROUGH, READBEFOREWRITE	NORMAL	YES
WRITEMODE_B	Read / Write Mode for Port B	NORMAL, WRITETHROUGH, READBEFOREWRITE	NORMAL	YES
ENABLE GSR	Enables Global Set Reset	ENABLE, DISABLE	ENABLED	YES

The True Dual Port RAM (RAM_DP_TRUE) can be configured as NORMAL, READ BEFORE WRITE or WRITE THROUGH modes. Each of these modes affects what data comes out of port Q of the memory during the write operation followed by the read operation at the same memory location. Detailed discussions of the WRITE modes and the constraints of the True Dual Port can be found in Appendix A.

Additionally users can select to enable the output registers for RAM_DP_TRUE. Figures 9-16 to 9-21 show the internal timing waveforms for the True Dual Port RAM (RAM_DP_TRUE) with these options.

Figure 9-16. True Dual Port RAM Timing Waveform - NORMAL Mode, without Output Registers

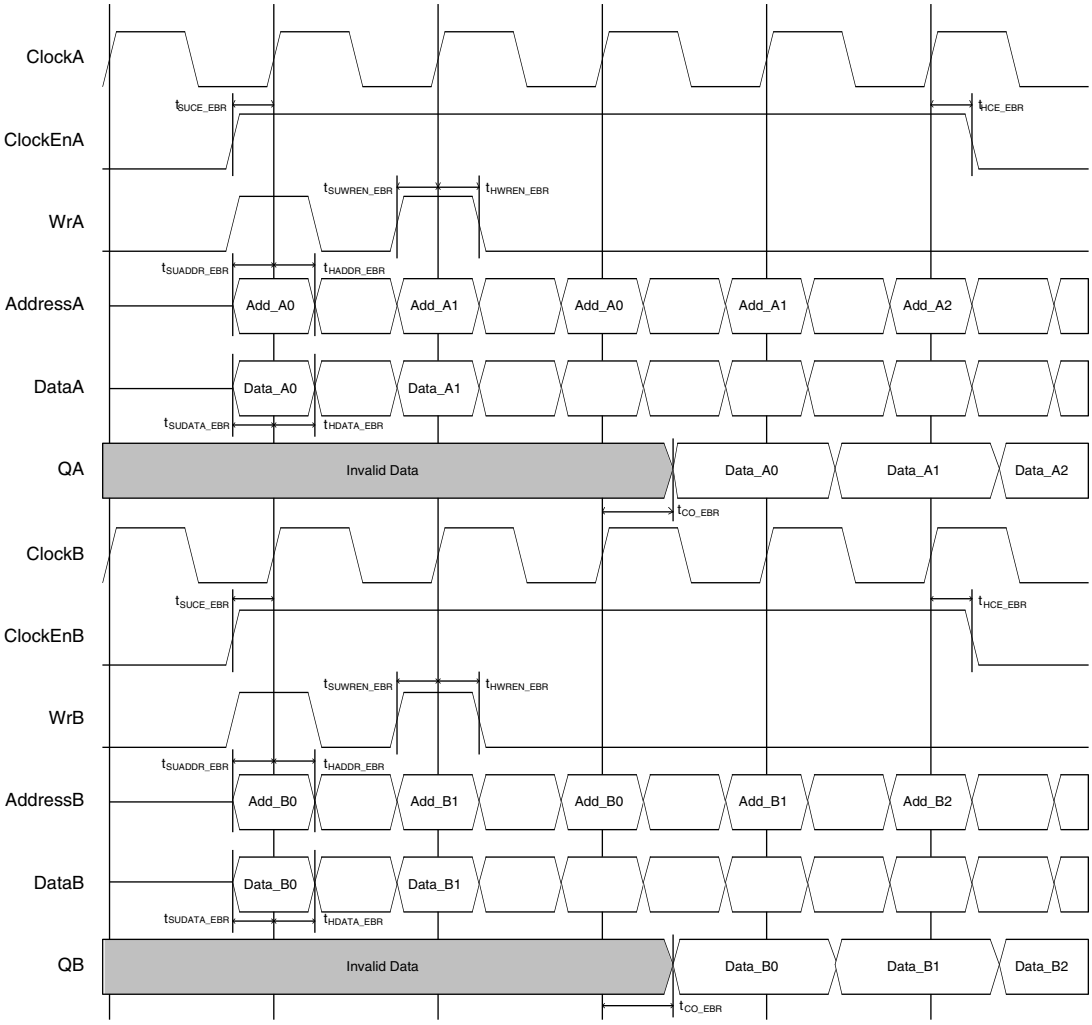


Figure 9-17. True Dual Port RAM Timing Waveform - NORMAL Mode with Output Register

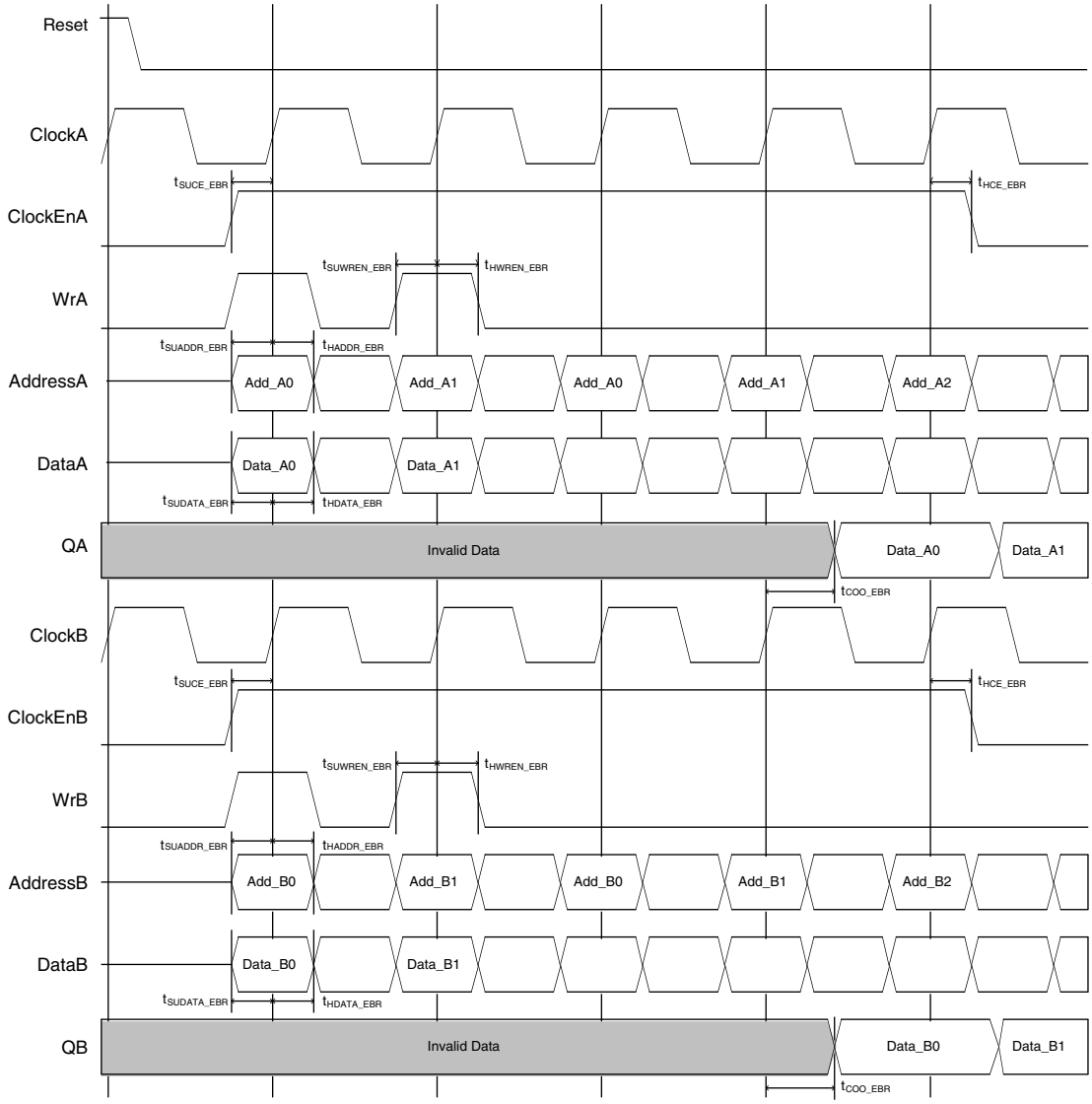


Figure 9-18. True Dual Port RAM Timing Waveform - READ BEFORE WRITE Mode, without Output Registers

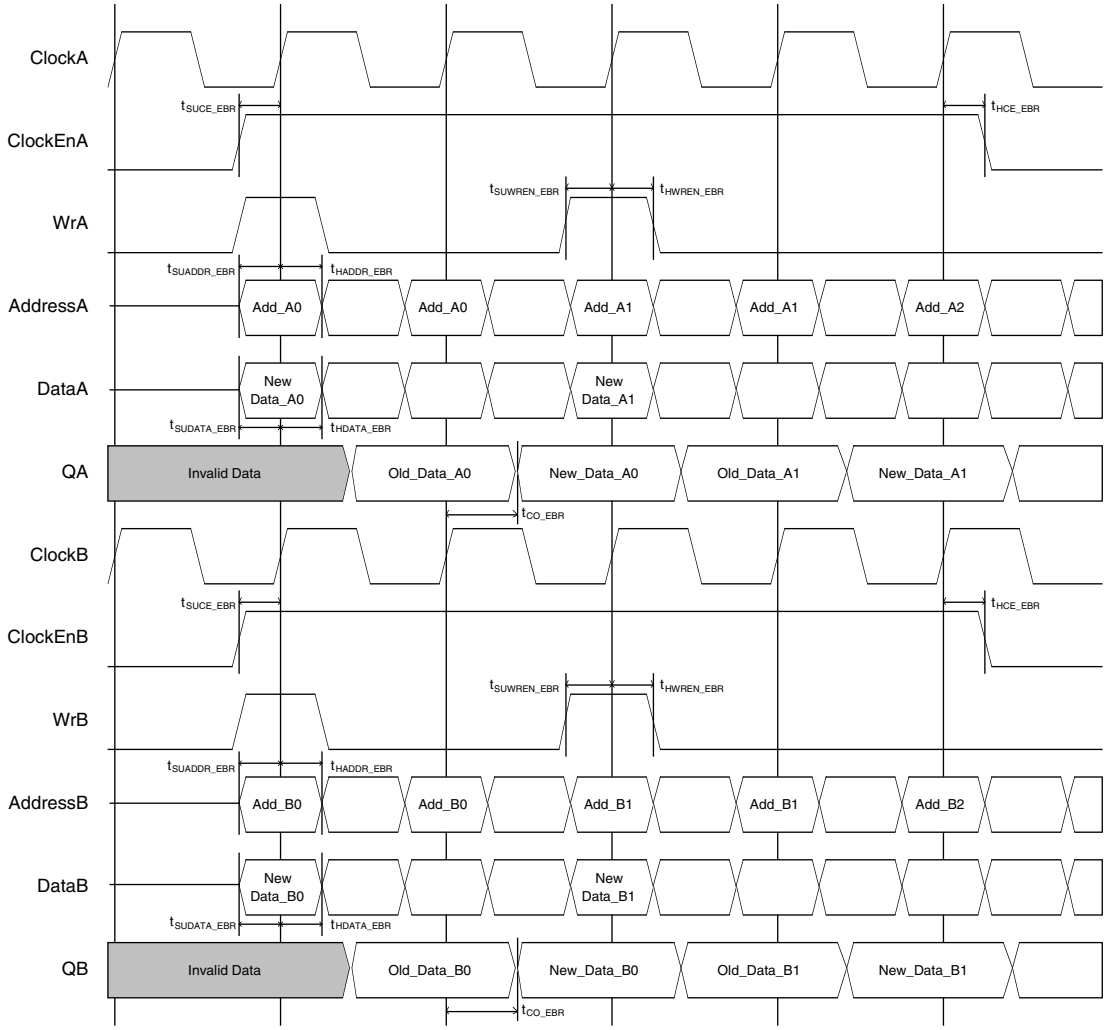


Figure 9-19. True Dual Port RAM Timing Waveform - READ BEFORE WRITE Mode, with Output Registers

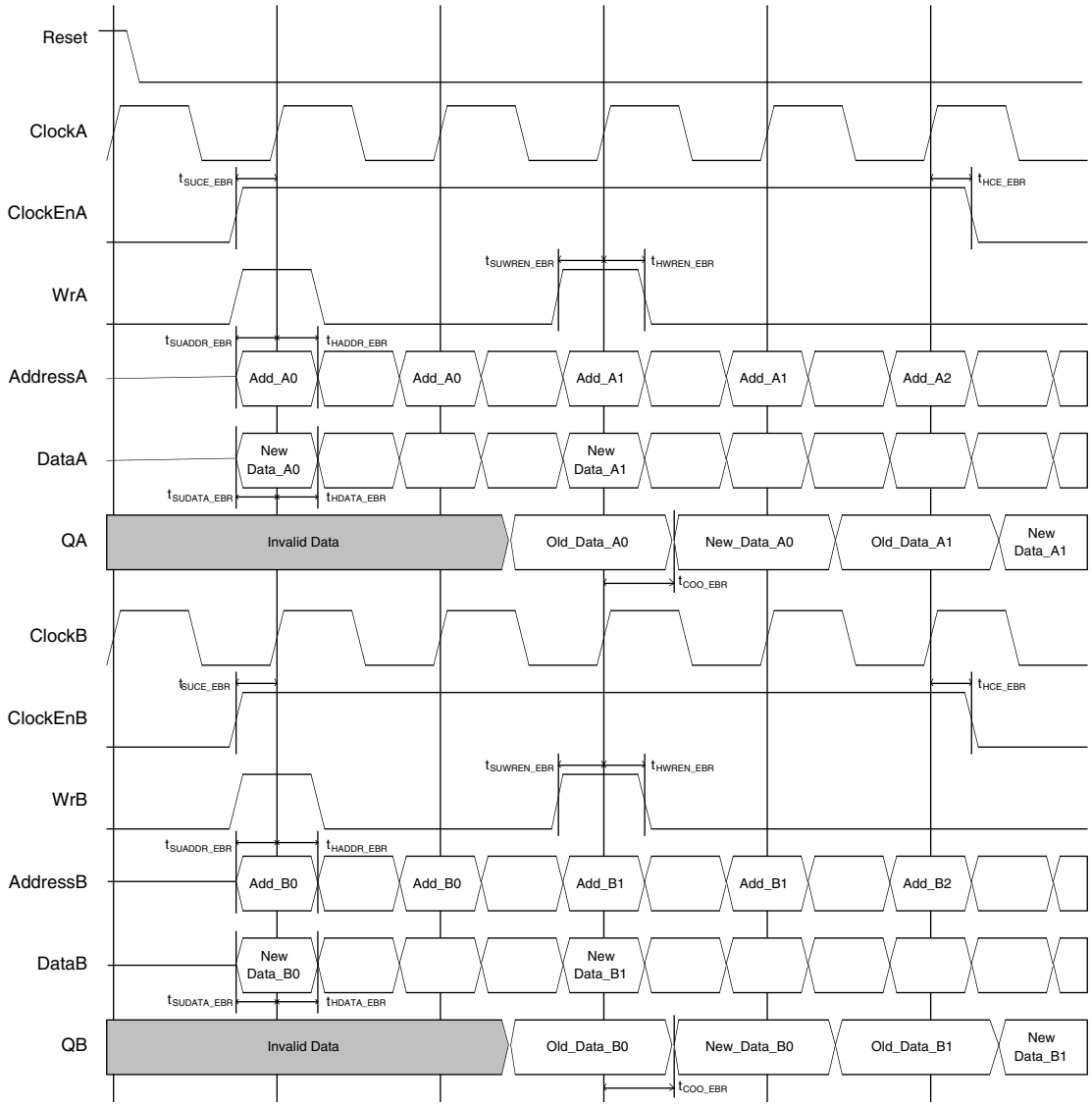


Figure 9-20. True Dual Port RAM Timing Waveform - WRITE THROUGH Mode, without Output Registers

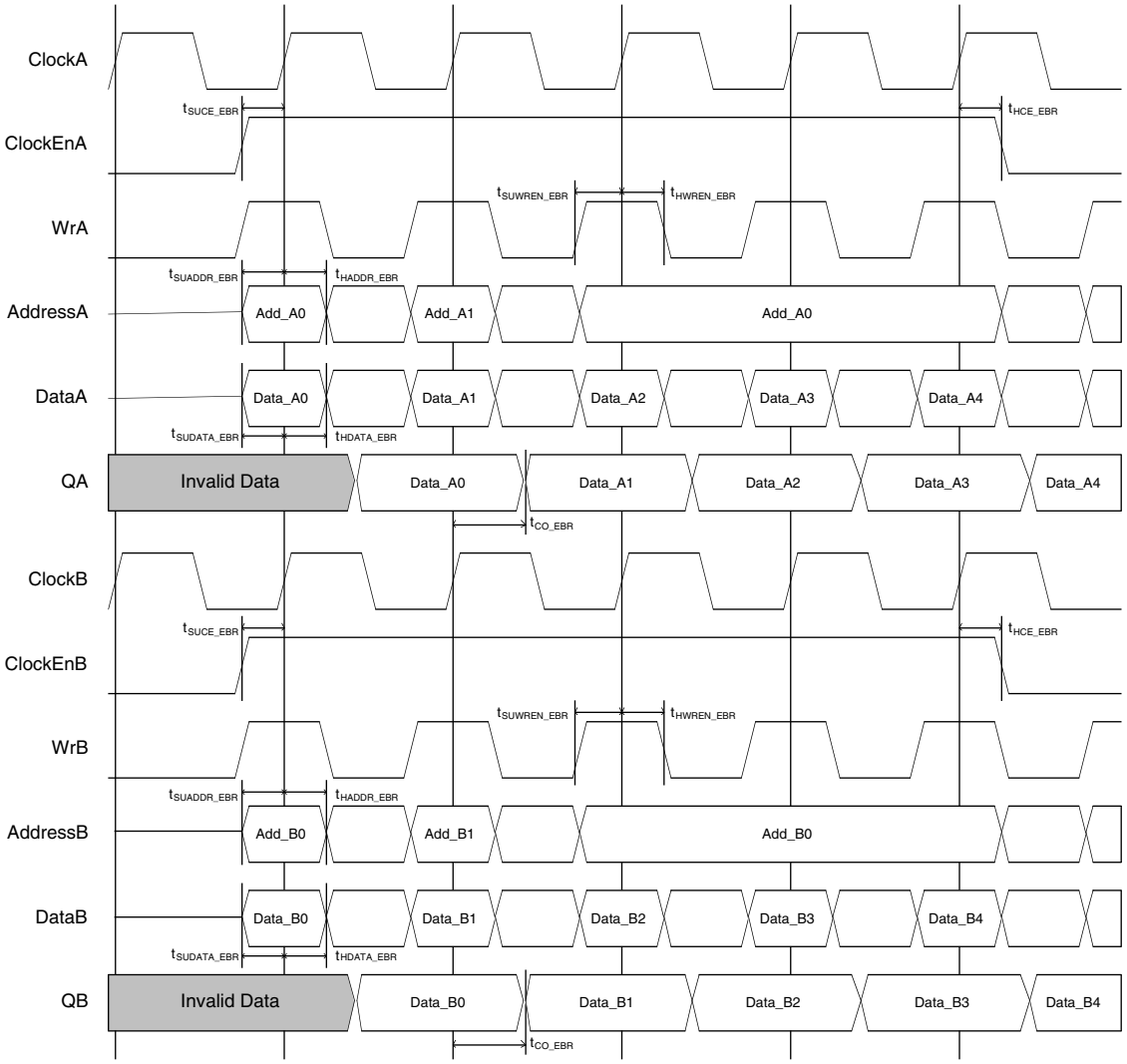
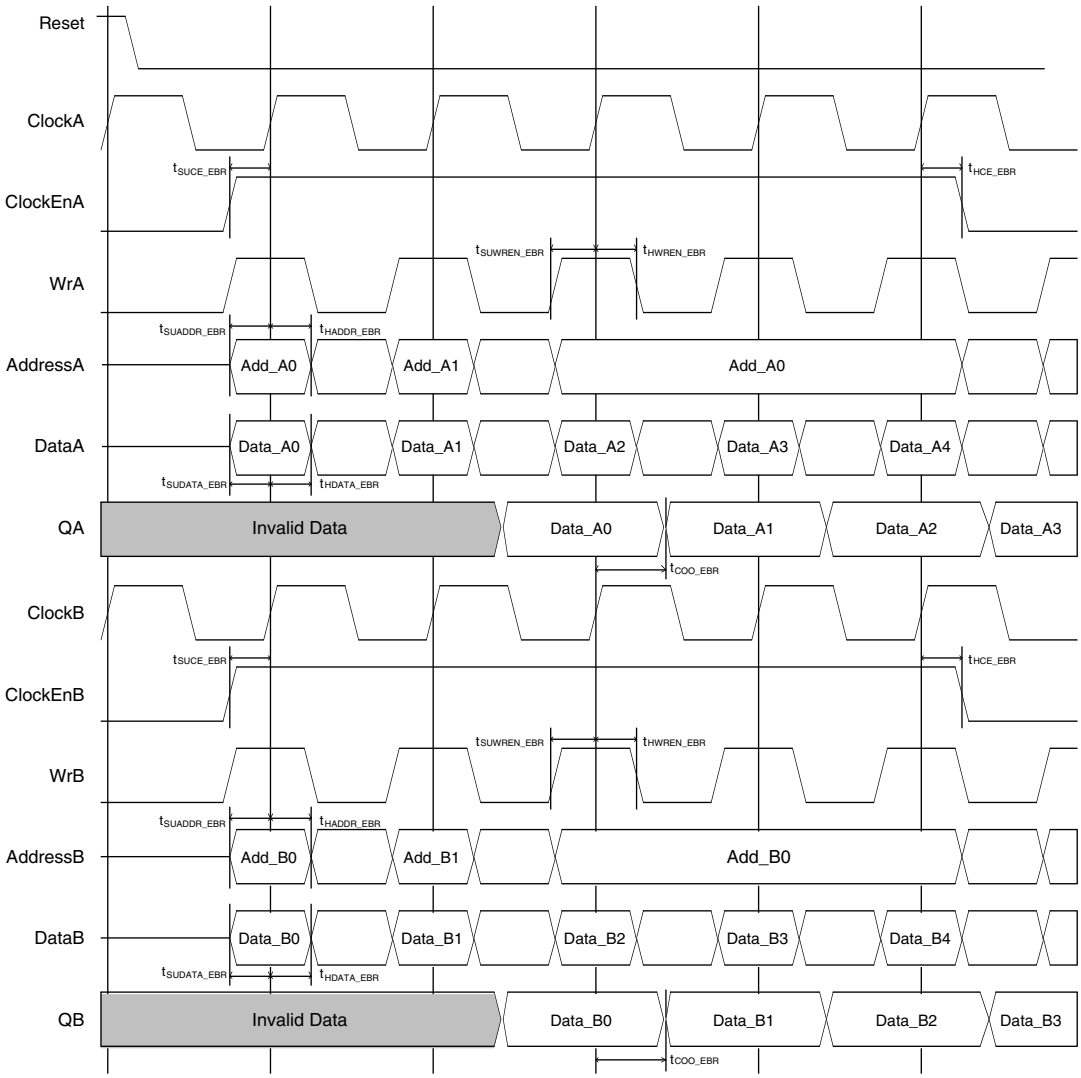


Figure 9-21. True Dual Port RAM Timing Waveform - WRITE THROUGH Mode, with Output Registers

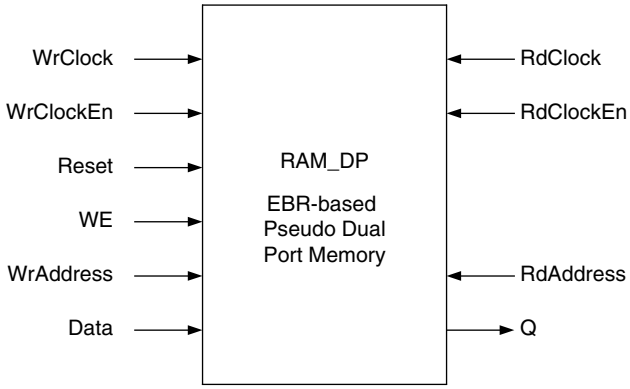


Pseudo Dual Port RAM (RAM_DP)- EBR Based

The EBR blocks in the MachXO devices can be configured as Pseudo-Dual Port RAM or RAM_DP. IPexpress allows users to generate the Verilog-HDL or VHDL along EDIF netlist for the memory size as per design requirements.

IPexpress generates the memory module as shown in Figure 9-22.

Figure 9-22. Pseudo Dual Port Memory Module Generated by IPexpress



The generated module makes use of these EBR blocks or primitives. For the memory sizes smaller than an EBR block, the module will be created in one EBR block. If the specified memory is larger than one EBR block, multiple EBR blocks can be cascaded in depth or width (as required to create these sizes).

In Pseudo Dual Port RAM mode, the input data and address for the ports are registered at the input of the memory array. The output data of the memory is optionally registered at the output.

The various ports and their definitions for the Single Port Memory are listed in Table 9-7. The table lists the corresponding ports for the module generated by IPexpress and for the EBR RAM_DP primitive.

Table 9-7. EBR-Based Pseudo-Dual Port Memory Port Definitions

Port Name in Generated Module	Port Name in the EBR Block Primitive	Description	Active State
RdAddress	ADR[x1:0]	Read Address	—
WrAddress	ADW[x2:0]	Write Address	—
RdClock	CLKR	Read Clock	Rising Clock Edge
WrClock	CLKW	Write Clock	Rising Clock Edge
RdClockEn	CER	Read Clock Enable	Active High
WrClockEn	CEW	Write Clock Enable	Active High
Q	DO[y1:0]	Read Data	—
Data	DI[y2:0]	Write Data	—
WE	WE	Write Enable	Active High
Reset	RST	Reset	Active High
—	CS[2:0]	Chip Select	—

Reset (RST) resets only the input and output registers of the RAM. It does not reset the contents of the memory.

Chip Select (CS) is a useful port when multiple cascaded EBR blocks are required by the memory. The CS signal forms the MSB for the address when multiple EBR blocks are cascaded. Since CS is a 3-bit bus, it can cascade eight memories easily. However, if the memory size specified by the user requires more than eight EBR blocks, the

Diamond or ispLEVER software automatically generates the additional address decoding logic, which is implemented in the PFU external to the EBR blocks.

Each EBR block consists of 9,216 bits of RAM. The values for x's (for address) and y's (data) for each EBR block for the devices are as per Table 9-8.

Table 9-8. MachXO Pseudo-Dual Port Memory Sizes for 9K Memory

Pseudo-Dual Port Memory Size	Input Data Port A	Input Data Port B	Output Data Port A	Output Data Port B	Read Address Port A [MSB:LSB]	Write Address Port B [MSB:LSB]
8K x 1	DIA	DIB	DOA	DOB	RAD[12:0]	WAD[12:0]
4K x 2	DIA[1:0]	DIB[1:0]	DOA[1:0]	DOB[1:0]	RAD[11:0]	WAD[11:0]
2K x 4	DIA[3:0]	DIB[3:0]	DOA[3:0]	DOB[3:0]	RAD[10:0]	WAD[10:0]
1K x 9	DIA[8:0]	DIB[8:0]	DOA[8:0]	DOB[8:0]	RAD[9:0]	WAD[9:0]
512 x 18	DIA[17:0]	DIB[17:0]	DOA[17:0]	DOB[17:0]	RAD[8:0]	WAD[8:0]
256 x 36	DIA[35:0]	DIB[35:0]	DOA[35:0]	DOB[35:0]	RAD[7:0]	WAD[7:0]

Table 9-9 shows the various attributes available for the Pseudo-Dual Port Memory (RAM_DP). Some of these attributes are user-selectable through the IPexpress GUI. For detailed attribute definitions, refer to Appendix A.

Table 9-9. MachXO Pseudo-Dual Port RAM Attributes

Attribute	Description	Values	Default Value	User Selectable through IPexpress
DATA_WIDTH_W	Write Data Word Width	1, 2, 4, 9, 18, 36	18	YES
DATA_WIDTH_R	Read Data Word Width	1, 2, 4, 9, 18, 36	18	YES
REGMODE	Register Mode (Pipelining)	NOREG, OUTREG	NOREG	YES
RESETMODE	Selects the Reset type	ASYNC, SYNC	ASYNC	YES
CSDECODE_W	Chip Select Decode for Write	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	0b000	NO
CSDECODE_R	Chip Select Decode for Read	0b000, 0b001, 0b010, 0b011, 0b100, 0b101, 0b110, 0b111	0b000	NO
ENABLE GSR	Enables Global Set Reset	ENABLE, DISABLE	ENABLED	YES

Users have the option of enabling the output registers for Pseudo-Dual Port RAM (RAM_DP). Figures 9-23 and 9-24 show the internal timing waveforms for Pseudo-Dual Port RAM (RAM_DP) with these options.

Figure 9-23. PSEUDO DUAL PORT RAM Timing Diagram – without Output Registers

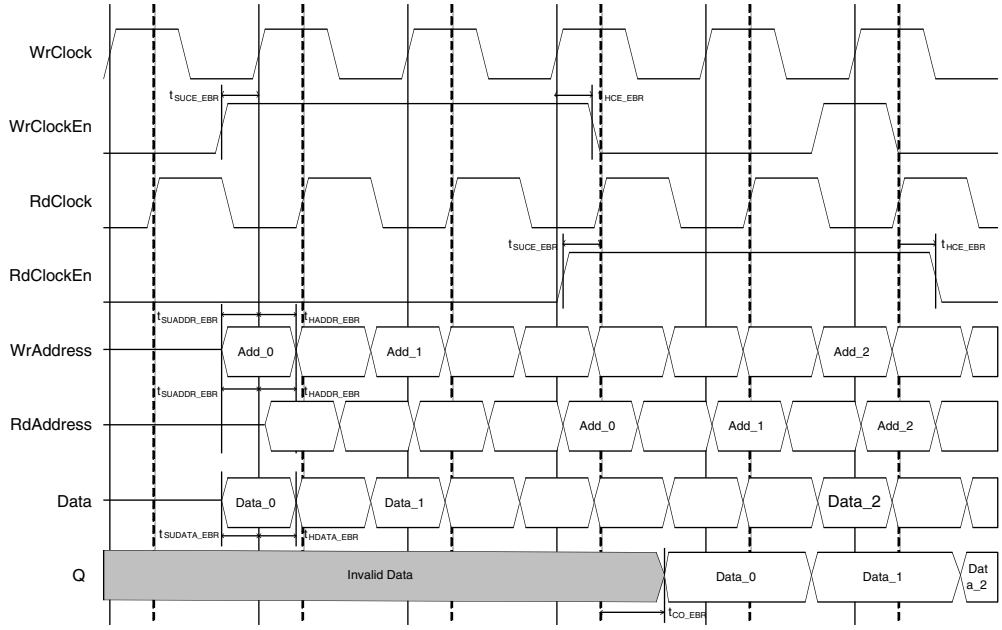
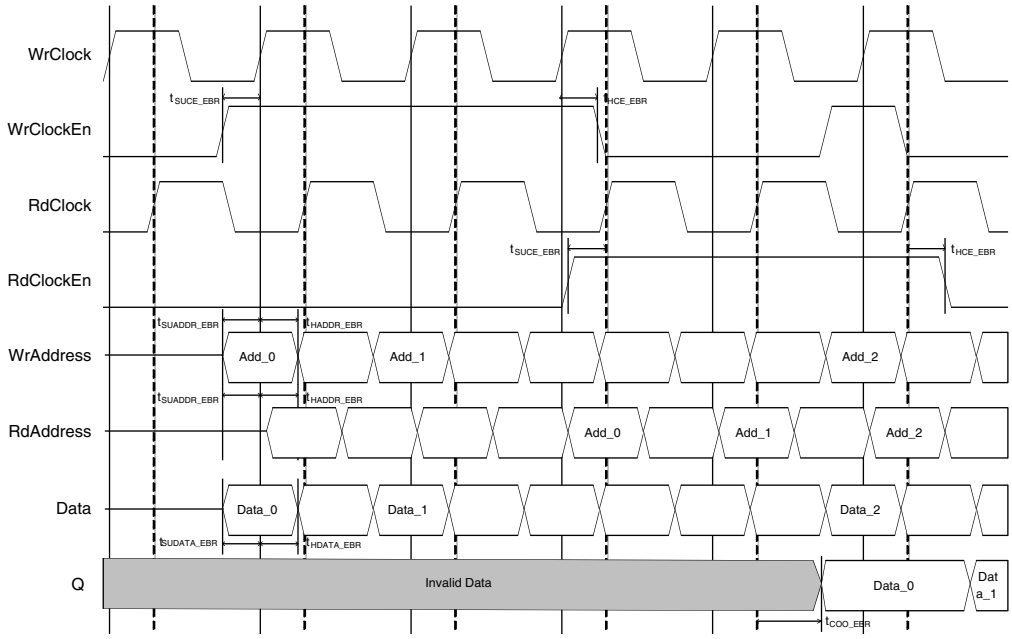


Figure 9-24. PSEUDO DUAL PORT RAM Timing Diagram – with Output Registers

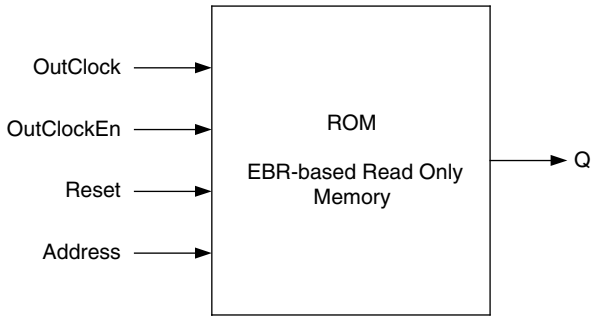


Read Only Memory (ROM) – EBR Based

The EBR blocks in the MachXO devices can be configured as Read Only Memory or ROM. IPexpress allows users to generate the Verilog-HDL or VHDL along EDIF netlist for the memory size, as per design requirement. Users are required to provide the ROM memory content in the form of an initialization file.

IPexpress generates the memory module as shown in Figure 9-25.

Figure 9-25. ROM – Read Only Memory Module Generated by IPexpress



The generated module makes use of these EBR blocks or primitives. For the memory sizes smaller than an EBR block, the module will be created in one EBR block. If the specified memory is larger than one EBR block, multiple EBR blocks can be cascaded, in depth or width (as required to create these sizes).

In ROM mode, the address for the port is registered at the input of the memory array. The output data of the memory is optionally registered at the output.

The various ports and their definitions for the ROM are listed in Table 9-10. The table lists the corresponding ports for the module generated by IPexpress and for the ROM primitive.

Table 9-10. EBR-based ROM Port Definitions

Port Name in Generated Module	Port Name in the EBR Block Primitive	Description	Active State
Address	AD[x:0]	Read Address	—
OutClock	CLK	Clock	Rising Clock Edge
OutClockEn	CE	Clock Enable	Active High
Reset	RST	Reset	Active High
—	CS[2:0]	Chip Select	—

Reset (RST) resets only the input and output registers of the RAM. It does not reset the contents of the memory.

Chip Select (CS) is a useful port when multiple cascaded EBR blocks are required by the memory. The CS signal forms the MSB for the address when multiple EBR blocks are cascaded. Since CS is a 3-bit bus, it can cascade eight memories easily. However, if the memory size specified by the user requires more than eight EBR blocks, the Diamond or ispLEVER software automatically generates the additional address decoding logic, which is implemented in the PFU external to the EBR blocks.

While generating the ROM using IPexpress, the user must provide the initialization file to pre-initialize the contents of the ROM. These files are the *.mem files and they can be of Binary, Hex or the Addressed Hex formats. The initialization files are discussed in detail in the Initializing Memory section of this document.

Users have the option of enabling the output registers for Read Only Memory (ROM). Figures 9-26 and 9-27 show the internal timing waveforms for the Read Only Memory (ROM) with these options.

Figure 9-26. ROM Timing Waveform – without Output Registers

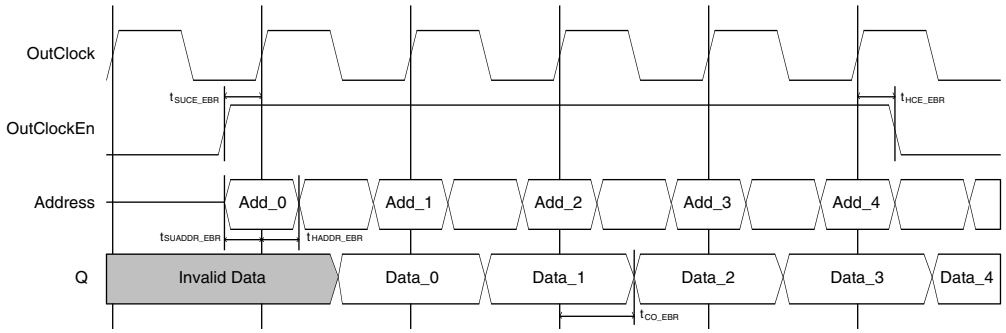
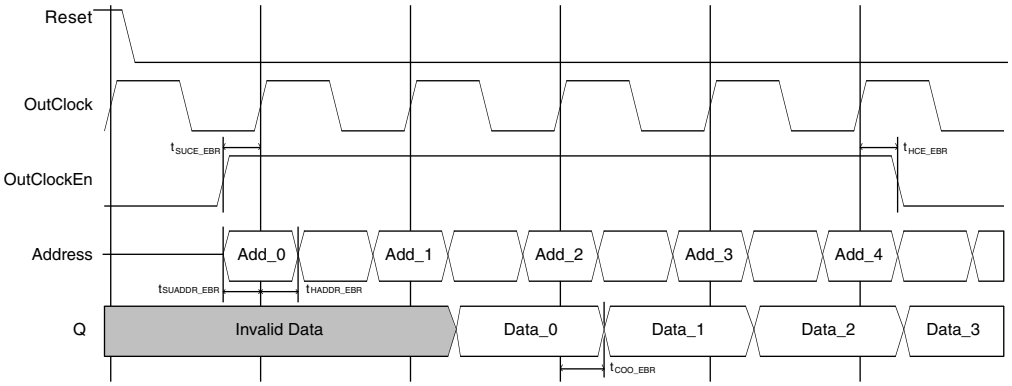


Figure 9-27. ROM Timing Waveform – with Output Registers

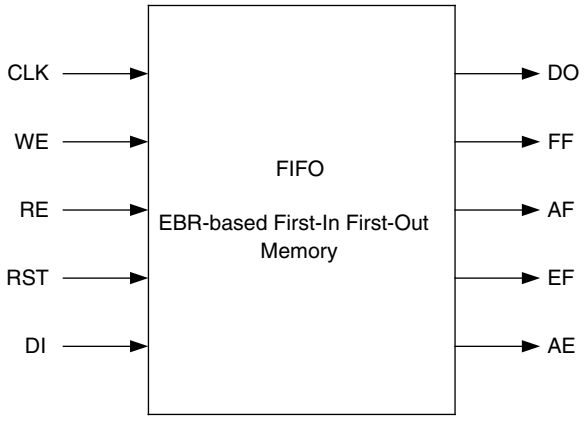


First In First Out (FIFO, FIFO_DC) - EBR Based

The EBR blocks in MachXO devices can be configured as Dual Clock First In First Out Memories, FIFO_DC. IPexpress allows users to generate the Verilog-HDL or VHDL along EDIF netlist for the memory size, according to design requirements.

IPexpress generates the FIFO_DC memory module as shown in Figure 9-28.

Figure 9-28. FIFO Module Generated by IPexpress



The generated module makes use of these EBR blocks or primitives. For the memory sizes smaller than an EBR block, the module will be created in one EBR block. If the specified memory is larger than one EBR block, multiple EBR blocks can be cascaded, in depth or width (as required to create these sizes).

A clock is always required, as only synchronous write is supported. The various ports and their definitions for the FIFO_DC are listed in Table 9-11.

Table 9-11. EBR-based FIFO_DC Memory Port Definitions

Port Name in Generated Module	Port Name in Primitive	Description	Active State
CLKR		Read Port Clock	Rising Clock Edge
CLKW		Write Port Clock	Rising Clock Edge
WE		Write Enable	Active High
RE		Read Enable	Active High
RST		Reset	Active High
DI		Data Input	—
DO		Data Output	—
FF		Full Flag	Active High
AF		Almost Full Flag	Active High
EF		Empty Flag	Active High
AE		Almost Empty	Active High

Reset (RST) resets only the input and output registers of the RAM. It does not reset the contents of the memory.

The various supported sizes for the FIFO_DC for MachXO are listed in Table 9-12.

Table 9-12. MachXO FIFO_DC Data Widths Sizes

FIFO Size	Input Data	Output Data
8K x 1	DI	DO
4K x 2	DI[1:0]	DO[1:0]
2K x 4	DI[3:0]	DO[3:0]
1K x 9	DI[8:0]	DO[8:0]
512 x 18	DI[17:0]	DO[17:0]
256 x 36	DI[35:0]	DO[35:0]

Table 9-13 shows the various attributes available for the FIFO_DC. Some of these attributes are user-selectable through the IPexpress GUI. For detailed attribute definitions, refer to Appendix A.

Table 9-13. FIFO and FIFO_DC Attributes for MachXO

Attribute	Description	Values	Default Value
DATA_WIDTH_W	Data Width Write Mode	1, 2, 4, 9, 18, 36	18
DATA_WIDTH_R	Data Width Read Mode	1, 2, 4, 9, 18, 36	18
REGMODE	Register Mode	NOREG, OUTREG	NOREG
RESETMODE	Select Reset Type	ASYNC, SYNC	ASYNC
CSDECODE_W	Chip Select Decode for Write Mode	0b00, 0b01, 0b10, 0b11	0
CSDECODE_R	Chip Select Decode for Read Mode	0b00, 0b01, 0b10, 0b11	0
GSR	Enable Global Set Reset	ENABLED, DISABLED	ENABLED
AEPOINTER	Almost Empty Pointer	Ob00000000000000, ..., 0b11111111111111	—
AFPOINTER	Almost Full Pointer	Ob00000000000000, ..., 0b11111111111111	—
FULLPOINTER	Full Pointer	Ob00000000000000, ..., 0b11111111111111	—
FULLPOINTER1	Full Pointer1	Ob00000000000000, ..., 0b11111111111111	—
AFPOINTER1	Almost Full Pointer1	Ob00000000000000, ..., 0b11111111111111	—
AEPOINTER1	Almost Empty Pointer1	Ob00000000000000, ..., 0b11111111111111	—

FIFO_DC Flags

The FIFO_DC have four flags available: Empty, Almost Empty, Almost Full and Full. Almost Empty and Almost Full flags have a programmable range.

The program ranges for the four FIFO_DC flags are specified in Table 9-14.

Table 9-14. FIFO_DC Flag Settings

FIFO Attribute Name	Description	Programming Range	Program Bits
FF	Full flag setting	2^N - 1	14
AFF	Almost full setting	1 to (FF-1)	14
AEF	Almost empty setting	1 to (FF-1)	14
EF	Empty setting	0	5

The only restriction on the flag setting is that the values must be in a specific order (Empty=0, Almost Empty next, followed by Almost Full and Full, respectively). The value of Empty is not equal to the value of Almost Empty (or Full is equal to Almost Full). In this case, a warning is generated and the value of Empty (or Full) is used in place of Almost Empty (or Almost Full). When coming out of reset, the active high flags Empty and Almost Empty are set to high, since they are true.

The user should specify the absolute value of the address at which the Almost Empty and Almost Full flags will go true. For example, if the Almost Full flag is required to go true at the address location 500 for a FIFO of depth 512, the user should specify a value of 500 in IPexpress.

The Empty and Almost Empty flags are always registered to the read clock and the Full and Almost Full flags are always registered to the write clock.

At reset both the write and read counters are pointing to address zero. After reset is de-asserted data can be written into the FIFO_DC to the address pointed to by the write counter at the positive edge of the write clock when the write enable is asserted.

Similarly, data can be read from the FIFO_DC from the address pointed to by the read counter at the positive edge of the read clock when read enable is asserted.

Read Pointer Reset (RPReset) is used to indicate a retransmit, and is more commonly used in “packetized” communications. In this application, the user must keep careful track of when a packet is written into or read from the FIFO_DC.

The data output of the FIFO_DC can be registered or non-registered through a selection in IPexpress. The output registers are enabled by read enable. A reset will clear the contents of the FIFO_DC by resetting the read and write pointers and will put the flags in the initial reset state.

FIFO_DC Operation

If the output registers are not enabled it will take two clock cycles to read the first word out. The register for the flag logic causes this extra clock latency. In the architecture of the emulated FIFO_DC, the internal read enables for reading the data out is controlled not only by the read enable provided by the user but also the empty flag. When the data is written into the FIFO, an internal empty flag is registered using write clock that is enabled by write enable (WrEn). Another clock latency is added due to the clock domain transfer from write clock to read clock using another register which is clocked by read clock that is enabled by read enable.

Internally, the output of this register is inverted and then ANDed with the user-provided read enable that becomes the internal read enable to the RAM_DP which is at the core of the FIFO_DC.

Thus, the first read data takes two clock cycles to propagate through. During the first data out, read enable goes high for one clock cycle, empty flag is de-asserted and is not propagated through the second register enabled by the read enable. The first clock cycle brings the Empty Low and the second clock cycle brings the internal read enable high (RdEn and !EF) and then the data is read out by the second clock cycle. Similarly, the first write data after the full flag has a similar latency.

If the user has enabled the output registers, the output registers will cause an extra clock delay during the first data out as they are clocked by the read clock and enabled by the read enable.

1. First RdEn and Clock Cycle to propagate the EF internally.
2. Second RdEn and Clock Cycle to generate internal Read Enable into the DPRAM.
3. Third RdEn and Clock Cycle to get the data out of the output registers.

Figure 9-29. FIFO_DC without Output Registers (Non-Pipelined)

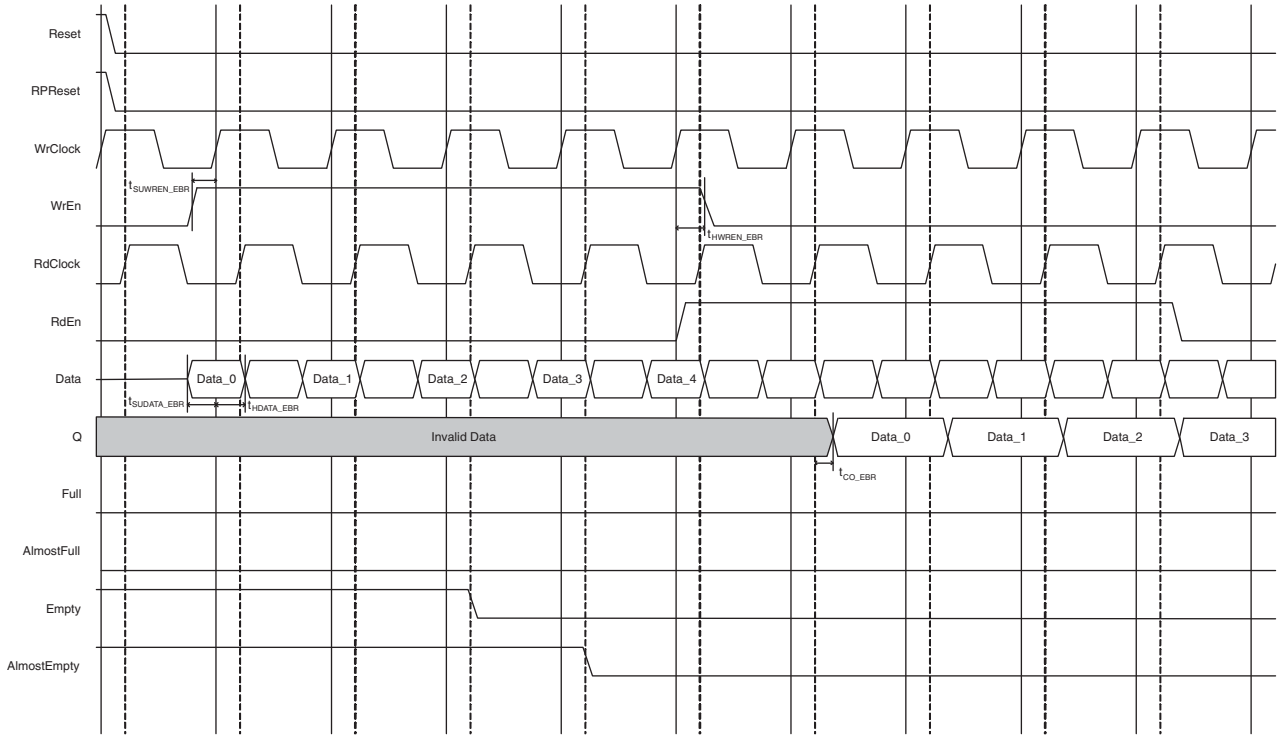
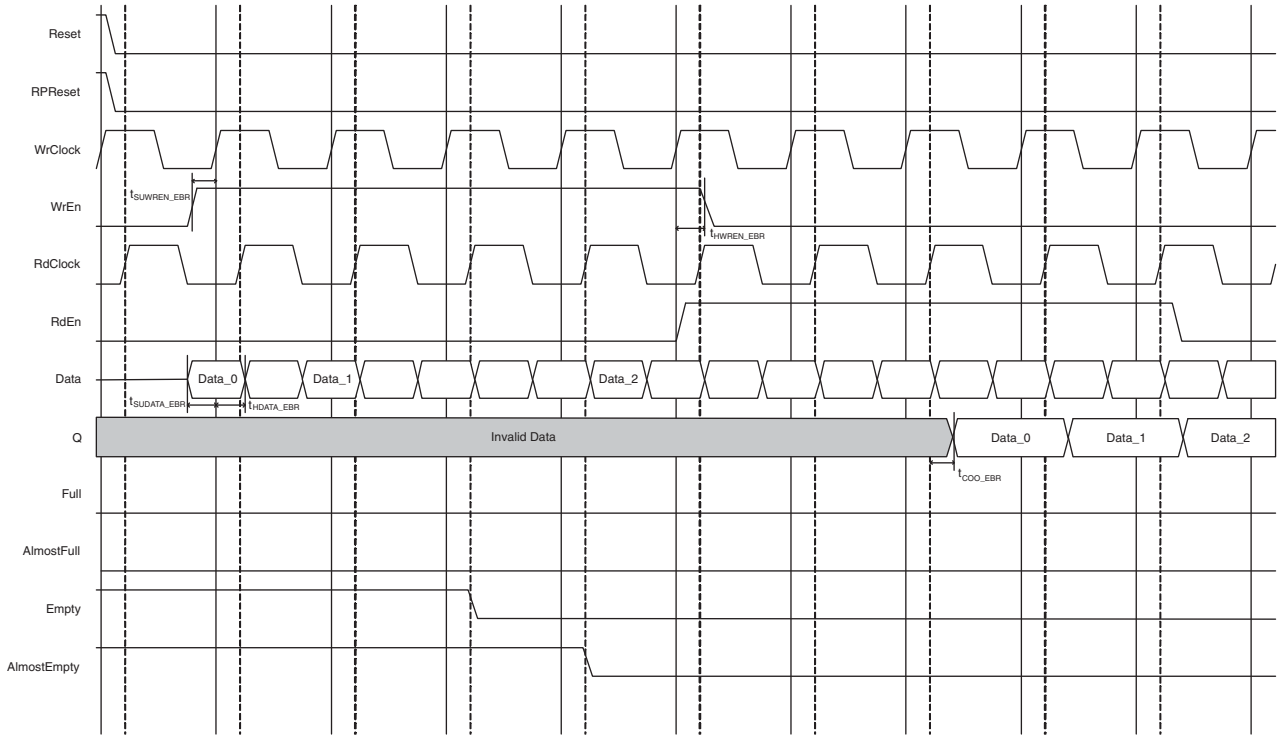


Figure 9-30. FIFO_DC with Output Registers (Pipelined)

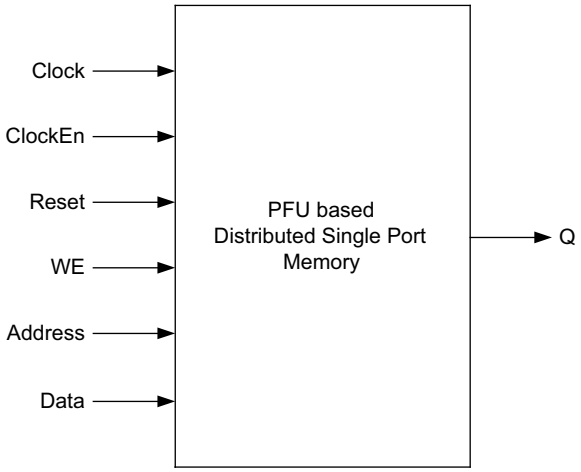


Distributed Single Port RAM (Distributed_SPRAM) – PFU Based

PFU based Distributed Single Port RAM is created using the 4-input LUT (Look-Up Table) available in the PFU. These LUTs can be cascaded to create larger distributed memory sizes.

Figure 9-31 shows the Distributed Single Port RAM module as generated by IPexpress.

Figure 9-31. Distributed Single Port RAM Module Generated by IPexpress



The generated module makes use of the 4-input LUT available in the PFU. Additional logic such as clock and reset is generated by utilizing the resources available in the PFU.

Ports such as Read Clock (RdClock) and Read Clock Enable (RdClockEn) are not available in the hardware primitive. These are generated by the IPexpress when the user wants the to enable the output registers in the IPexpress configuration.

The various ports and their definitions are listed in Table 9-15. The table lists the corresponding ports for the module generated by IPexpress and for the primitive.

Table 9-15. PFU-based Distributed Single Port RAM Port Definitions

Port Name in Generated Module	Port Name in the PFU Primitive	Description	Active State
Clock	CK	Clock	Rising Clock Edge
ClockEn	—	Clock Enable	Active High
Reset	—	Reset	Active High
WE	WRE	Write Enable	Active High
Address	AD[3:0]	Address	—
Data	DI[1:0]	Data In	—
Q	DO[1:0]	Data Out	—

Ports such as Clock Enable (ClockEn) are not available in the hardware primitive. These are generated by IPexpress when the user wishes the to enable the output registers in the IPexpress configuration.

Users have the option of enabling the output registers for Distributed Single Port RAM (Distributed_SPRAM). Figures 9-32 and 9-33 show the internal timing waveforms for the Distributed Single Port RAM (Distributed_SPRAM) with these options.

Figure 9-32. PFU-Based Distributed Single Port RAM Timing Waveform – Without Output Registers

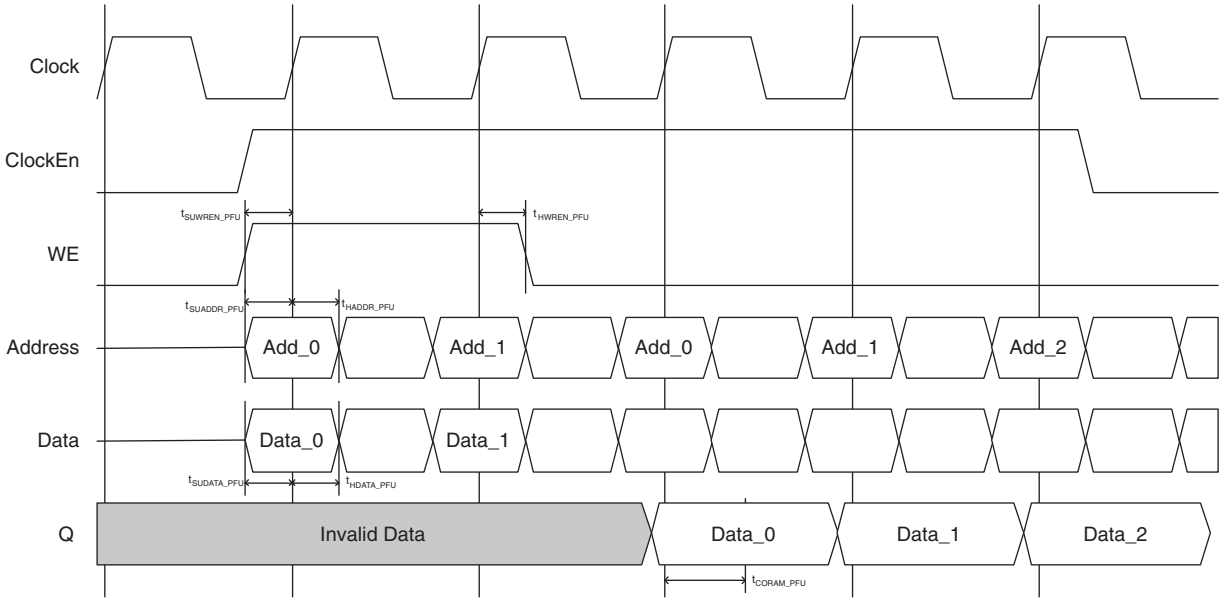
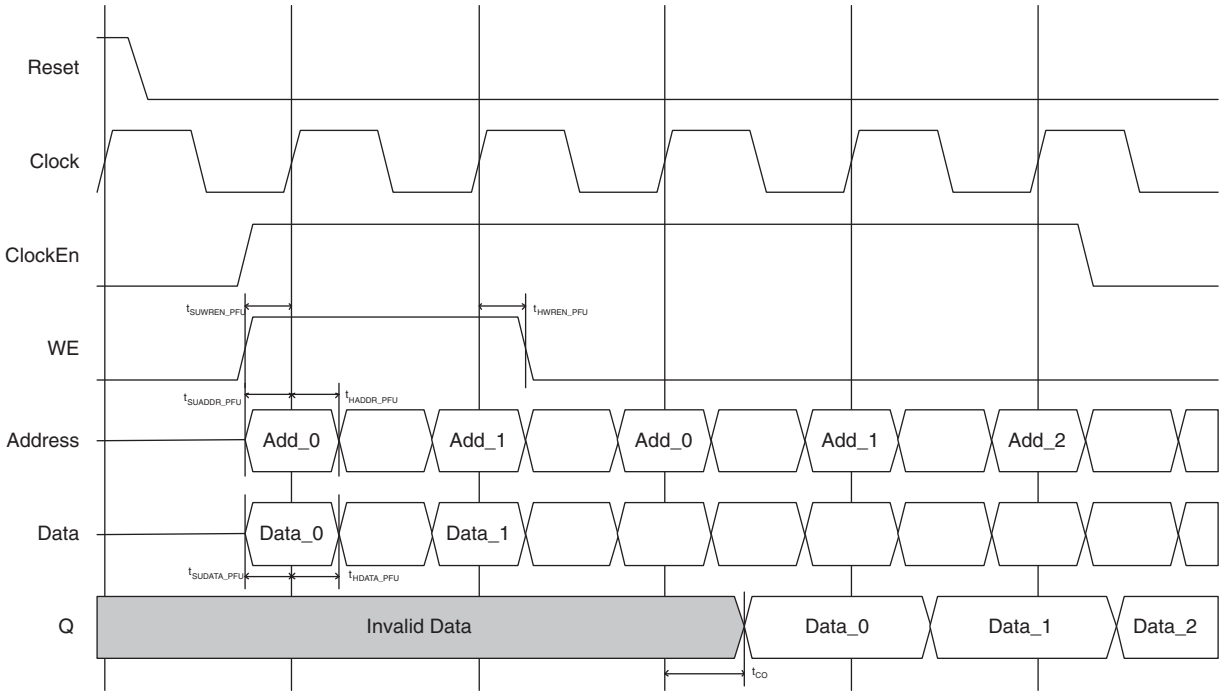


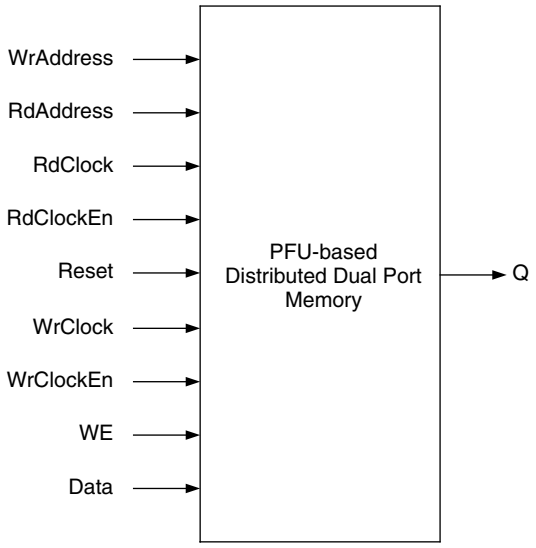
Figure 9-33. PFU-Based Distributed Single Port RAM Timing Waveform - With Output Registers



Distributed Dual Port RAM (Distributed_DPRAM) – PFU Based

PFU-based Distributed Dual Port RAM is also created using the 4-input LUT (Look-Up Table) available in the PFU. These LUTs can be cascaded to create larger distributed memory sizes.

Figure 9-34. Distributed Dual Port RAM Module Generated by IPexpress



The generated module makes use of the 4-input LUT available in the PFU. Additional such as Clock and Reset is generated by utilizing the resources available in the PFU.

The various ports and their definitions are listed in Table 9-16. The table lists the corresponding ports for the module generated by IPexpress and for the primitive.

Table 9-16. PFU based Distributed Dual-Port RAM Port Definitions

Port Name in Generated Module	Port Name in the EBR Block Primitive	Description	Active State
WrAddress	WAD[3:0]	Write Address	—
RdAddress	RAD[3:0]	Read Address	—
RdClock	—	Read Clock	Rising Clock Edge
RdClockEn	—	Read Clock Enable	Active High
WrClock	WCK	Write Clock	Rising Clock Edge
WrClockEn	—	Write Clock Enable	Active High
WE	WRE	Write Enable	Active High
Data	DI[1:0]	Data Input	—
Q	RDO[1:0]	Data Out	—

Ports such as Read Clock (RdClock) and Read Clock Enable (RdClockEn) are not available in the hardware primitive. These are generated by IPexpress when the user wants the to enable the output registers in the IPexpress configuration.

Users have the option of enabling the output registers for Distributed Dual Port RAM (Distributed_DPRAM). Figures 9-35 and 9-36 show the internal timing waveforms for the Distributed Dual Port RAM (Distributed_DPRAM) with these options.

Figure 9-35. PFU-Based Distributed Dual Port RAM Timing Waveform – without Output Registers (Non-Pipelined)

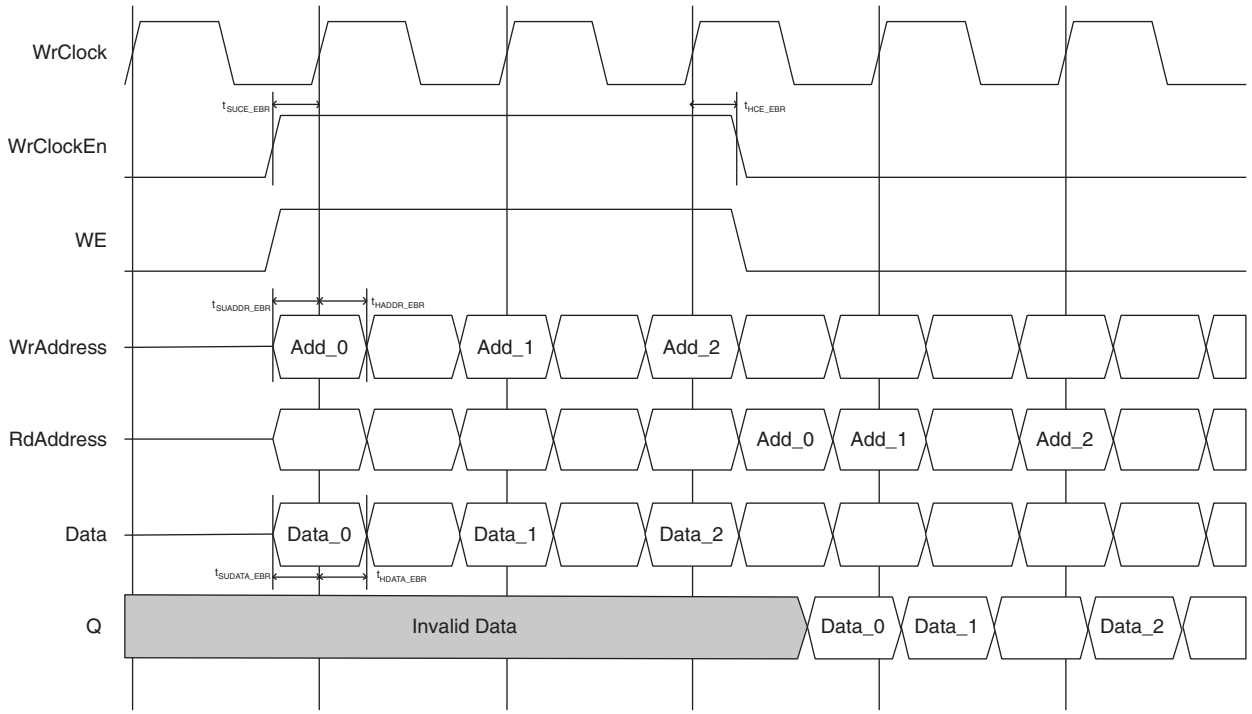
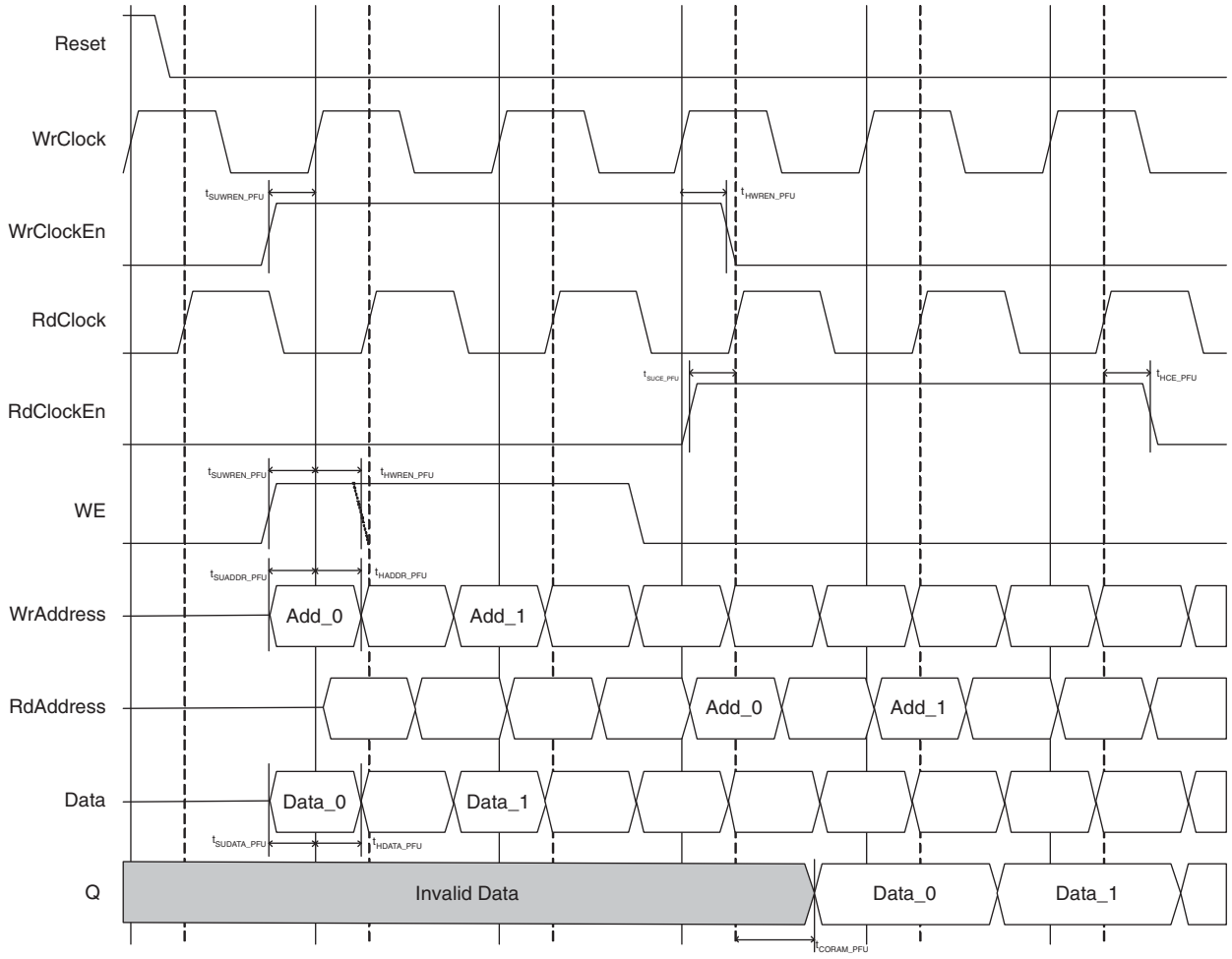


Figure 9-36. PFU-Based Distributed Dual Port RAM Timing Waveform – with Output Registers (Pipelined)

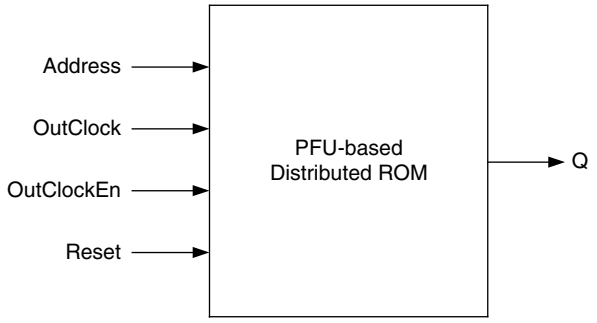


Distributed ROM (Distributed_ROM) – PFU Based

PFU-based Distributed ROM is also created using the 4-input LUT (Look-Up Table) available in the PFU. These LUTs can be cascaded to create larger distributed memory sizes.

Figure 9-37 shows the Distributed ROM module as generated by IPexpress.

Figure 9-37. Distributed ROM Generated by IPexpress



The generated module makes use of the 4-input LUT available in the PFU. Additional logic like clock and reset is generated by utilizing the resources available in the PFU.

Ports such as Out Clock (OutClock) and Out Clock Enable (OutClockEn) are not available in the hardware primitive. These are generated by IPexpress when the user wants the to enable the output registers in the IPexpress configuration.

The various ports and their definitions are listed in Table 9-17. The table lists the corresponding ports for the module generated by IPexpress and for the primitive.

Table 9-17. PFU-based Distributed ROM Port Definitions

Port Name in Generated Module	Port Name in the PFU Block Primitive	Description	Active State
Address	AD[3:0]	Address	—
OutClock	—	Out Clock	Rising Clock Edge
OutClockEn	—	Out Clock Enable	Active High
Reset	—	Reset	Active High
Q	DO	Data Out	—

Users have the option of enabling the output registers for Distributed ROM (Distributed_ROM). Figures 9-38 and 9-39 show the internal timing waveforms for the Distributed ROM with these options.

Figure 9-38. PFU-Based ROM Timing Waveform – Without Output Registers

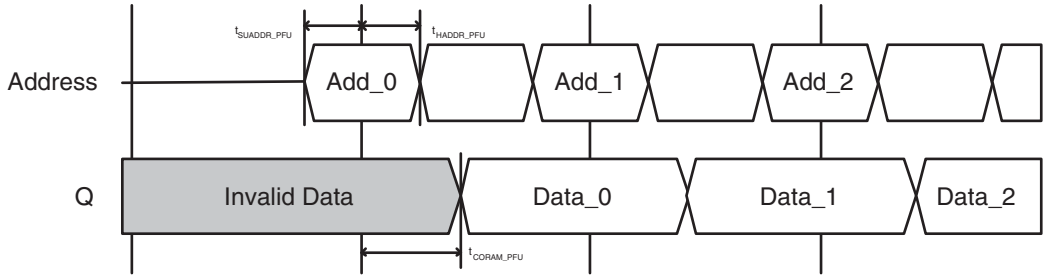
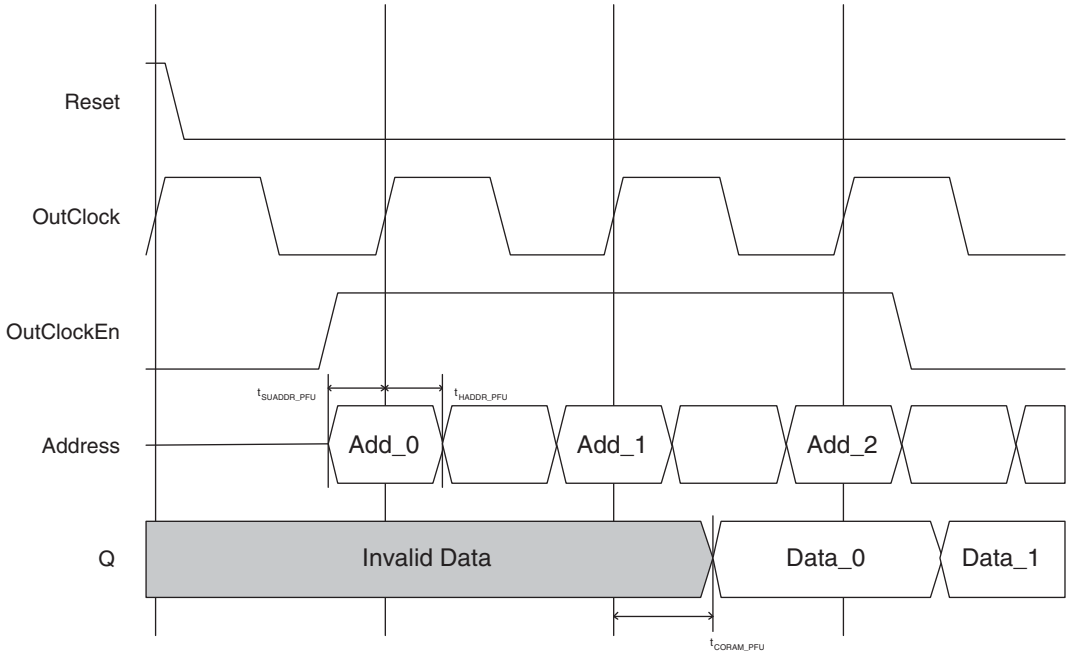


Figure 9-39. PFU-Based ROM Timing Waveform – with Output Registers



Initializing Memory

In the EBR based ROM or RAM memory modes and the PFU based ROM memory mode, it is possible to specify the power-on state of each bit in the memory array. Each bit in the memory array can have one of two values: 0 or 1.

Initialization File Formats

The initialization file is an ASCII file, which can be created or edited using any ASCII editor. IPexpress supports three types of memory file formats:

1. Binary file
2. Hex file
3. Addressed Hex

The file name for the initialization file is *.mem (<file_name>.mem). Each row depicts the value to be stored in a particular memory location and the number of characters (or the number of columns) represents the number of bits for each address (or the width of the memory module).

The initialization file is primarily used for configuring the ROMs. The EBR in RAM mode can also use the initialization file to preload the memory contents.

Binary File

The binary file is a text file of 0's and 1's. The rows indicate the number of words and the columns indicate the width of the memory.

```
Memory Size 20x32
00100000010000000010000001000000
00000001000000010000000100000001
00000010000000100000001000000010
00000011000000110000001100000011
00000100000001000000010000000100
00000101000001010000010100000101
```

```
00000110000001100000011000000110
00000111000001110000011100000111
00001000010010000000100001001000
00001001010010010000100101001001
00001010010010100000101001001010
00001011010010110000101101001011
00001100000011000000110000001100
00001101001011010000110100101101
00001110001111100000111000111110
00001111001111110000111100111111
00010000000100000001000000010000
00010001000100010001000100010001
00010010000100100001001000010010
00010011000100110001001100010011
```

Hex File
The hex file is essentially a text file of hex characters arranged in a similar row-column arrangement. The number of rows in the file is same as the number of address locations, with each row indicating the content of the memory location.

```
Memory Size 8x16
A001
0B03
1004
CE06
0007
040A
0017
02A4
```

Addressed Hex
Addressed hex consists of lines of address and data. Each line starts with an address, followed by a colon, and any number of data. The format of the memfile is address: data data data data ... where address and data are hexadecimal numbers.

- A0: 03 F3 3E 4F
- B2: 3B 9F

The first line puts 03 at address A0, F3 at address A1, 3E at address A2,and 4F at address A3. The second line puts 3B at address B2 and 9F at address B3.

There is no limitation on the values of address and data. The value range is automatically checked based on the values of addr_width and data_width. If there is an error in an address or data value, an error message is printed. Users need not specify data at all address locations. If data is not specified at a certain address, the data at that location is initialized to 0. IPexpress makes memory initialization possible through both the synthesis and simulation flows.

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
—	—	Previous Lattice releases.
April 2006	01.2	Updated the Initializing Memory section.
October 2006	01.3	Added dual port memory access notes in Appendix A.
September 2007	01.4	Updated FIFO_DC without Output Registers (Non-pipelined) and FIFO_DC with Output Registers (Pipelined) waveforms.
October 2010	01.5	Updated for Lattice Diamond design software support.

Appendix A. Attribute Definitions

DATA_WIDTH
Data width is associated with the RAM and FIFO elements. The DATA_WIDTH attribute defines the number of bits in each word. It takes the values defined in the RAM size tables in each memory module.

REGMODE
REGMODE, or the Register mode attribute, is used to enable pipelining in the memory. This attribute is associated with the RAM and FIFO elements. The REGMODE attribute takes the NOREG or OUTREG mode parameter that disables and enables the output pipeline registers.

RESETMODE
The RESETMODE attribute allows users to select the mode of reset in the memory. This attribute is associated with the block RAM elements. RESETMODE takes two parameters: SYNC and ASYNC. SYNC means that the memory reset is synchronized with the clock. ASYNC means that the memory reset is asynchronous to clock.

CSDECODE
CSDECODE, or the Chip Select Decode attributes, are associated to block RAM elements. Chip Select (CS) is a useful port when multiple cascaded EBR blocks are required by the memory. The CS signal forms the MSB for the address when multiple EBR blocks are cascaded. CS is a 3-bit bus, so it can cascade eight memories easily. CSDECODE takes the following parameters: “000”, “001”, “010”, “011”, “100”, “101”, “110”, and “111”. CSDECODE values determine the decoding value of CS[2:0]. CSDECODE_W is chip select decode for write and CSDECODE_R is chip select decode for read for Pseudo Dual Port RAM. CSDECODE_A and CSDECODE_B are used for true dual port RAM elements and refer to the A and B ports.

WRITEMODE
The WRITEMODE attribute is associated with the block RAM elements. It takes the NORMAL, WRITETHROUGH, and READBEFOREWRITE mode parameters.

In NORMAL mode, the output data does not change or get updated during the write operation. This mode is supported for all data widths.

In WRITETHROUGH mode, the output data is updated with the input data during the write cycle. This mode is supported for all data widths.

In READBEFOREWRITE mode, the output data port is updated with the existing data stored in the write address, during a write cycle. This mode is supported for x9, x18 and x36 data widths.

WRITEMODE_A and WRITEMODE_B are used for dual port RAM elements and refer to the A and B ports in case of a True Dual Port RAM.

For all modes of the True Dual Port module, simultaneous read access from one port and write access from the other port to the same memory address is not recommended. The read data may be unknown in this situation. Also, simultaneous write access to the same address from both ports is not recommended. When this occurs, the data stored in the address becomes undetermined when one port tries to write a 'H' and the other tries to write a 'L'.

It is recommended that users implement control logic to identify this situation if it occurs and then either:

1. Implement status signals to flag the read data as possibly invalid, or
2. Implement control logic to prevent the simultaneous access from both ports.

GSR
GSR, the Global Set/ Reset attribute, is used to enable or disable the global set/reset for the RAM element.