Logos2 Family FPGA Dedicated RAM Module (DRM) User Guide

(UG040002, V1.2)

(14.07.2024)

Shenzhen Pango Microsystems Co., Ltd.

Revisions History

Document Revisions

Version	Date of Release	Revisions
V1.2	14.07.2024	Initial release.

About this Manual

Terms and Abbreviations

Terms and Abbreviations	Meaning
DRM	Dedicated RAM Module
SP	Single Port
SDP	Simple Dual Port
DP	True Dual Port
FIFO	First In First Out
GTP	Generic Technology Primitive
NW	Normal-Write
RBW	Read-before-Write
TW	Transparent-Write
OR	Output Register
IR	Input Register
ECC	Error Correcting Code

Table of Contents

Tables

Figures

Chapter 1 General Introduction

The Logos2 family FPGA's DRM can be configured with 2 storage units of 18K bits or 1 storage unit of 36K bits. Each DRM supports DP (True Dual Port) RAM mode, SP (Single Port) RAM mode, SDP (Simple Dual Port) RAM mode, ROM mode, and synchronous/asynchronous FIFO (First In First Out) mode. DRM resources also support Input Register (IR) and Output Register (OR), allowing superior performance when DRMs are cascaded. The total number of DRMs depends on the type of Logos2 family device.

IPs embedded in DP RAM, SP RAM, SDP RAM, ROM, and synchronous/asynchronous FIFO modules can be conveniently generated through the IP Compiler tool embedded in the Pango Design Suite software by Shenzhen Pango Microsystems Co., Ltd.

1.1 Features

Table 1-1 shows the features of the Logos2 family FPGA's DRM.

Table 1-1 List of Features

Function	Description
Memory Capacity	One 36K unit or two 18K units, supporting a maximum width of 72 bits
DP RAM Mode	Up to 36-bit data width
SDP RAM Mode	Up to 72-bit data width
SP RAM Mode	Up to 72-bit data width
ROM Mode	Up to 72-bit data width
FIFO Mode	Asynchronous and synchronous modes
Write Mode	DP and SP support Normal-Write, Transparent-Write, and Read-before-Write modes
Byte Write Enable	Supported
Optional Output Registers	Supported
Hard Cascade	Two adjacent 36K modules cascaded (with the same data width on both ports) support 64Kx1
ECC	Single-bit error correction and dual-bit error detection are available only in 36K (SDP/FIFO) 512x72 mode
Synchronous/Asynchronous Reset of Output Registers	Supported

Note: Under SP mode, 32 bits or above write mode is prohibited from being set to TW or RBW modes, and should be set to the default NW mode. Refer to 5. 5 Read and Write Operations.

1.2 Resource Scale

Table 1-2 Logos2 Resource Scale

Device	Resource Quantity/36K Bits
PG2L25H	55
PG2L50H	85
PG2L100H	155
PG2L200H	415

Note: For the resource scale of the Logos2 family FPGA's DRM, refer to "***DS04001_Logos2 Family FPGA Device Datasheet***".

1.3 Supported Modes

Each DRM supports DP (True Dual Port) RAM mode, SP (Single Port) RAM mode, SDP (Simple Dual Port) RAM mode, ROM mode, and synchronous/asynchronous FIFO (First In First Out) mode.

The port of DRM supports two types of data widths: one is 2^N (including 1/2/4/8/16/32/64 bits). The other is 9×2^N (including 9/18/36/72 bit). DP RAM and SDP RAM modes also support mixed bit widths, meaning the two ports can be configured with different bit widths. For example, an SDP RAM can be configured to 16Kx1 at the write port and 512x32 at the read port, thereby saving the serial-to-parallel conversion logic from 1 bit to 32 bits.

Both DP RAM and SDP RAM support 36K and 18K modes. In DP RAM mode, DRM has a maximum data width of 36 bits, and both Port A and Port B in the DRM can independently perform read and write operations, supporting different clocks. In SDP RAM mode, the DRM data width increases to 72 bits, with one of Port A and Port B for data writing and the other for data reading, both read and write ports also support different clocks.

For the allowed bit width combinations for DP RAM and SDP RAM in 36K and 18K DRM modes, refer to Table 3-2 and Table 3-3, Table 4-2 and Table 4-3, respectively.

In the 18K mode for both SP RAM and ROM modes, DRM contains two ports, SP RAM mode allows independent read and write operations on these two ports, while in ROM mode the port is read-only; when sharing two pairs of ports, DRM contains only one port. For the allowed bit width combinations for SP RAM and ROM in 36K and 18K DRM modes, refer to Table 5-2 and Table 5-3, Table 6-2 and Table 6-3, respectively.

In FIFO mode, one port is for data writing and the other for data reading, with read and write ports can use different clocks. For the 36K FIFO mode configuration and the 18K FIFO mode configuration, refer to Table 7-3 and Table 7-4.

In the SDP/FIFO 512x72 memory mode, a DRM supports 72-bit data ECC correction, including 64 valid data bits and 8 bits for ECC checksum bit storage. In ECC mode, a DRM supports single-bit error correction and dual-bit error detection functions, while outputting ECC_SBITERR (single-bit error correction flag), ECC_DBITERR (dual-bit error detection flag), ECC parity, and read address in ECC mode (ECC parity and read address are only supported in SDP mode).

Multiple DRMs can be combined into larger DP RAM, SDP RAM, SP RAM, ROM, or FIFO through cascade. For this, GTP_DRM provides an additional 3-bit address extension (CS[2:0]), commonly used for deeply-extended applications.

For Address and Data Port Mapping, refer to Section 9.1 in the Appendix.

Chapter 2 GTP Description

GTP (Generic Technology Primitive) can be directly instantiated in the design. For details on the GTPs supported by the Logos2 family FPGA and their usage instructions, refer to "***UG040007_Logos2 Family Products GTP User Guide***". For the specific availability, ports, and parameters of each GTP, refer to GTP user guide.

The related GTPs of the Logos2 family FPGA's DRM forms the basis for each DRM mode. Users can configure DRM GTPs to implement different modes and functions of DRMs. Four related GTPs are supported by the Logos2 family FPGA's DRM, including GTP_DRM36K_E1, GTP_DRM18K_E1, GTP_FIFO36K_E1, and GTP_FIFO18K_E1.

2.1 GTP_DRM36K_E1

Instantiate GTP_DRM36K_E1 to implement DP, SDP, SP, and ROM modes. For the GTP structure block diagram, refer to Figure 2-1. For port descriptions and parameter descriptions, refer to Table 2-1 and Table 2-2. For GTP_DRM36K_E1 Instantiation Template, refer to the Appendix.

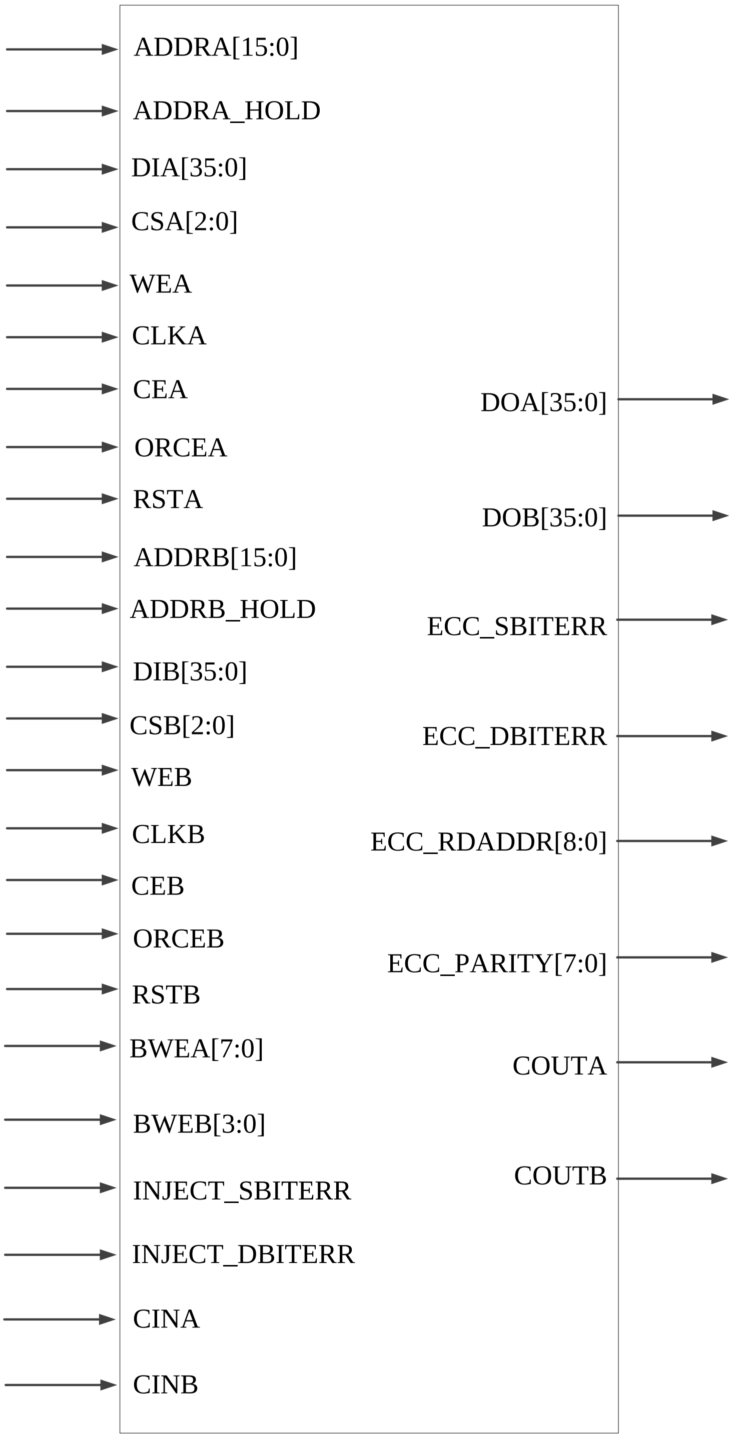

Figure 2-1 GTP_DRM36K_E1 Structure Diagram

Table 2-1 GTP_DRM36K_E1 Port Description

Port	**Input/Output**	**Description**
ADDRA[15:0]	Input	Port A address input bus
ADDRA_HOLD	Input	Port A address input hold
DIA[35:0]	Input	Port A data input bus
CSA[2:0]	Input	Port A address extension
WEA	Input	Port A write enable
BWEA[7:0]	Input	Port A byte write enable
CLKA	Input	Port A clock
CEA	Input	Port A input register clock enable
ORCEA	Input	Port A output register clock enable
RSTA	Input	Port A data register reset
DOA[35:0]	Output	Port A data output bus
ADDRB[15:0]	Input	Port B address input bus
ADDRB_HOLD	Input	Port B address input hold
DIB[35:0]	Input	Port B data input bus
CSB[2:0]	Input	Port B address extension
WEB	Input	Port B write enable
BWEB[3:0]	Input	Port B byte write enable
CLKB	Input	Port B clock
CEB	Input	Port B input register clock enable
ORCEB	Input	Port B output register clock enable
RSTB	Input	Port B data register reset
CINA	Input	Used for deep cascading in 64Kx1 mode, with the input cascaded to the data output of Port A of the adjacent DRM
CINB	Input	Used for deep cascading in 64Kx1 mode, with the input cascaded to the data output of Port B of the adjacent DRM
INJECT_SBITERR	Input	Single-bit error injection in ECC mode
INJECT_DBITERR	Input	Dual-bit error injection in ECC mode
DOB[35:0]	Output	Port B data output bus
COUTA	Output	Used for deep cascading in 64Kx1 mode, with the output cascaded to the data output of Port A of the adjacent DRM
COUTB	Output	Used for deep cascading in 64Kx1 mode, with the output cascaded to the data output of Port B of the adjacent DRM
ECC_SBITERR	Output	Single-bit error flag in ECC mode
ECC_DBITERR	Output	Dual-bit error flag in ECC mode
ECC_PARITY[7:0]	Output	ECC encode checksum bit output
ECC_RDADDR[8:0]	Output	ECC decode read address output

Note: For detailed DRM port usage instructions, refer to 9. 3 DRM Port Signal Description.

Table 2-2 GTP_DRM36K_E1 Parameter Descriptions

Parameter	Description	Setting Value
CSA_MASK[2:0]	Port A address extension control signal	3’b000 ~ 3’b111
CSB_MASK[2:0]	Port B address extension control signal	3’b000 ~ 3’b111
DATA_WIDTH_A	Data width of Port A	1, 2, 4, 8, 16, 32, 9, 18, 36, 64, 72
DATA_WIDTH_B	Data width of Port B	1, 2, 4, 8, 16, 32, 9, 18, 36, 64, 72
WRITE_MODE_A	Port A write mode	"NORMAL_WRITE" "TRANSPARENT_WRITE" "READ_BEFORE_WRITE"
WRITE_MODE_B	Port B write mode	"NORMAL_WRITE" "TRANSPARENT_WRITE" "READ_BEFORE_WRITE"
DOA_REG	Port A output register	0 = Do not enable output register 1 = Enable output register
DOB_REG	Port B output register	0 = Do not enable output register 1 = Enable output register
RST_TYPE	Reset mode selection	"SYNC": Synchronous reset "ASYNC": Asynchronous reset
RAM_MODE	RAM mode selection	"TRUE_DUAL_PORT": DP RAM "SIMPLE_DUAL_PORT": SDP RAM "SINGLE_PORT": SP RAM "ROM": ROM
GRS_EN	Global reset enable signal (Internal Chip)	"FALSE": Global Reset not enabled; "TRUE": Global Reset enabled.
DOA_REG_CLKINV	Port A output register clock inversion	0 = Clock not inverted 1 = Clock inverted
DOB_REG_CLKINV	Port B output register clock inversion	0 = Clock not inverted 1 = Clock inverted
RSTA_VAL	Port A output set/reset value	36'h0_0000_0000~36'hF_FFFF_FFFF
RSTB_VAL	Port B output set/reset value	36'h0_0000_0000~36'hF_FFFF_FFFF
RAM_CASCADE	64Kx1 hard cascade mode	"NONE": No hard cascade "UPPER": Serves as a hard cascade data output module "LOWER": Serves as a hard cascade additional module
ECC_WRITE_EN	ECC write mode enable	"FALSE": Not enabled "TRUE": Enabled
ECC_READ_EN	ECC read mode enable	"FALSE": Not enabled "TRUE": Enabled
INIT_00 INIT_01 INIT_02 … INIT_7F	For RAM initialization configuration parameters, refer to 9.2 Initialization Configuration Parameter Mapping	288'b0 ~2^288-1
INIT_FILE	Initialization file path;	"NONE": No initialization file is specified, and the initialization data will be the value set by the parameter INIT_XX; "XXX": XXX represents the specific initialization file path, which overrides the parameter INIT_XX as the initial value for DRM

Parameter	**Description**	**Setting Value**
BLOCK_X	Data cascade coordinates when DRM36K is cascaded, used to map the same initialization file to different cascaded DRMs	Depends on the cascaded DRM coordinates
BLOCK_Y	Address cascade coordinates when DRM36K is cascaded, used to map the same initialization file to different cascaded DRMs	Depends on the cascaded DRM coordinates
RAM_DATA_WIDTH	Maximum data width after DRM cascading, used to map the same initialization file to different cascaded DRMs	Depends on the number of cascaded DRMs
RAM_ADDR_WIDTH	Maximum address width after DRM cascading, used to map the same initialization file to different cascaded DRMs	Depends on the number of cascaded DRMs
INIT_FORMAT	Initialization file format	"BIN": Binary "HEX": Hexadecimal

Note: SDP mode does not support write mode settings. When configuring DRM to SDP mode using GTP, it is disallowed to manually modify the write mode parameter WRITE_MODE_A/B.

2.2 GTP_DRM18K_E1

Instantiate GTP_DRM18K_E1 to implement DP, SDP, SP, and ROM modes. For the GTP structure block diagram, refer to Figure 2-2. For port descriptions and parameter descriptions, refer to Table 2-3 and Table 2-4. For GTP_DRM18K_E1 Instantiation Template, refer to the Appendix.

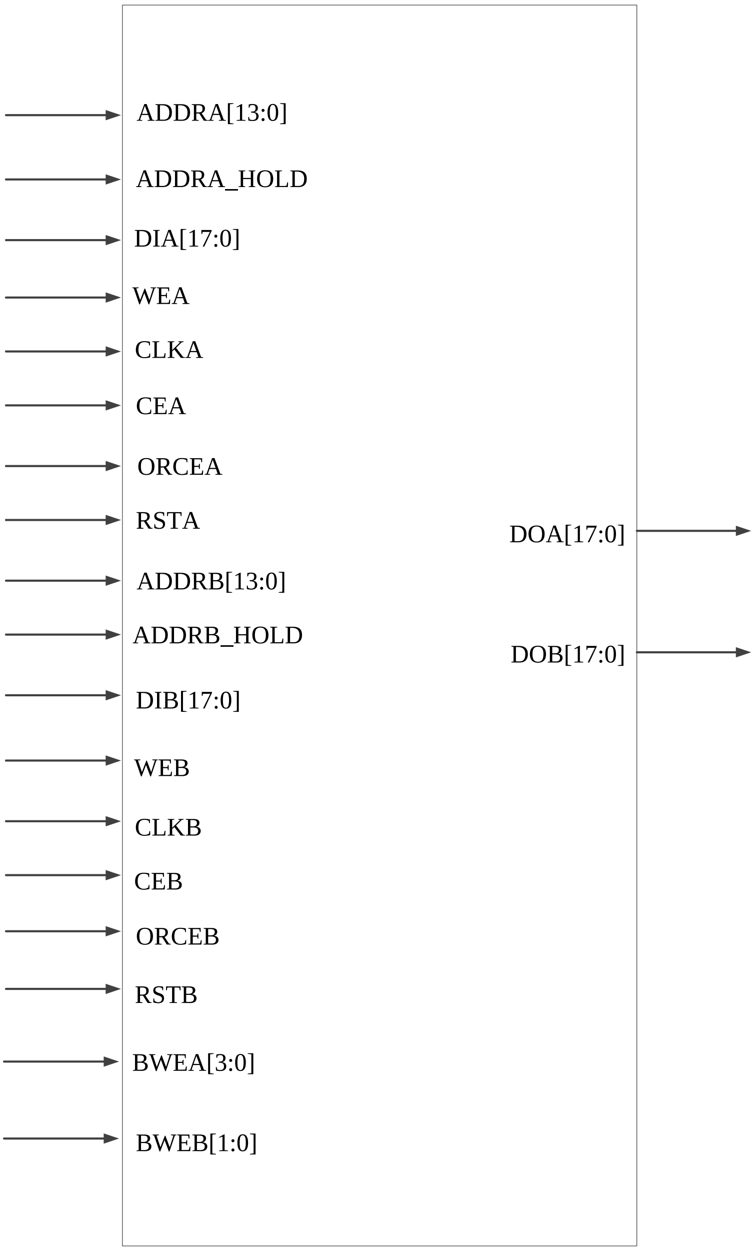

Figure 2-2 GTP_DRM18K_E1 Structure Diagram

Table 2-3 GTP_DRM18K_E1 Port Descriptions

Port	Input/Output	Description
ADDRA[13:0]	Input	Port A address input bus
ADDRA_HOLD	Input	Port A address input hold
DIA[17:0]	Input	Port A data input bus
WEA	Input	Port A write enable
BWEA[3:0]	Input	Port A byte write enable
CLKA	Input	Port A clock
CEA	Input	Port A input register clock enable
ORCEA	Input	Port A output register clock enable
RSTA	Input	Port A data register reset
DOA[17:0]	Output	Port A data output bus
ADDRB[13:0]	Input	Port B address input bus
ADDRB_HOLD	Input	Port B address input hold
DIB[17:0]	Input	Port B data input bus
WEB	Input	Port B write enable
BWEB[1:0]	Input	Port B byte write enable
CLKB	Input	Port B clock
CEB	Input	Port B input register clock enable
ORCEB	Input	Port B output register clock enable
RSTB	Input	Port B data register reset
DOB[17:0]	Output	Port B data output bus

Note: For detailed DRM port usage instructions, refer to 9. 3 DRM Port Signal Description.

Table 2-4 GTP_DRM18K_E1 Parameter Descriptions

Parameter	Description	Setting Value
DATA_WIDTH_A	Data width of Port A	1, 2, 4, 8, 16, 32, 9, 18, 36
DATA_WIDTH_B	Data width of Port B	1, 2, 4, 8, 16, 32, 9, 18, 36
WRITE_MODE_A	Port A write mode	"NORMAL_WRITE" "TRANSPARENT_WRITE" "READ_BEFORE_WRITE"
WRITE_MODE_B	Port B write mode	"NORMAL_WRITE" "TRANSPARENT_WRITE" "READ_BEFORE_WRITE"
DOA_REG	Port A output register	0 = Do not enable output register 1 = Enable output register
DOB_REG	Port B output register	0 = Do not enable output register 1 = Enable output register
RST_TYPE	Reset mode selection	"SYNC": Synchronous reset "ASYNC": Asynchronous reset
RAM_MODE	RAM mode selection	"TRUE_DUAL_PORT": DP RAM "SIMPLE_DUAL_PORT": SDP RAM "SINGLE_PORT": SP RAM "ROM": ROM
GRS_EN	Global reset enable signal (Internal Chip)	"FALSE": Global Reset not enabled; "TRUE": Global Reset enabled.

Parameter	Description	Setting Value
DOA_REG_CLKINV	Port A output register clock inversion	0 = Clock not inverted 1 = Clock inverted
DOB_REG_CLKINV	Port B output register clock inversion	0 = Clock not inverted 1 = Clock inverted
RSTA_VAL	Port A output set/reset value	18'h0_0000~18'h3_FFFF
RSTB_VAL	Port B output set/reset value	18’h0_0000~18’h3_FFFF
INIT_00 INIT_01 INIT_02 … INIT_3F	For RAM initialization configuration parameters, refer to 9.2 Initialization Configuration Parameter Mapping	288'b0 ~2^288-1
INIT_FILE	Initialization file path	"NONE": No initialization file is specified, and the initialization data will be the value set by the parameter INIT_XX; "XXX": XXX represents the specific initialization file path, which overrides the parameter INIT_XX as the initial value for DRM
BLOCK_X	Data cascade coordinates when DRM18K is cascaded, used to map the same initialization file to different cascaded DRMs	Depends on the cascaded DRM coordinates
BLOCK_Y	Data cascade coordinates when DRM18K is cascaded, used to map the same initialization file to different cascaded DRMs	Depends on the cascaded DRM coordinates
RAM_DATA_WIDTH	Maximum data width after DRM cascading, used to map the same initialization file to different cascaded DRMs	Depends on the number of cascaded DRMs
RAM_ADDR_WIDTH	Maximum address width after DRM cascading, used to map the same initialization file to different cascaded DRMs	Depends on the number of cascaded DRMs
INIT_FORMAT	Initialization file format	"BIN": Binary "HEX": Hexadecimal

Note: SDP mode does not support write mode settings. When configuring DRM to SDP mode using GTP, it is disallowed to manually modify the write mode parameter WRITE_MODE_A/B.

2.3 GTP_FIFO36K_E1

Instantiate GTP_FIFO36K_E1 to implement FIFO mode. For the GTP structure block diagram, refer to Figure 2-3. For port descriptions and parameter descriptions, refer to Table 2-5 and Table 2-6. For GTP_FIFO36K_E1 Instantiation Template, refer to the Appendix.

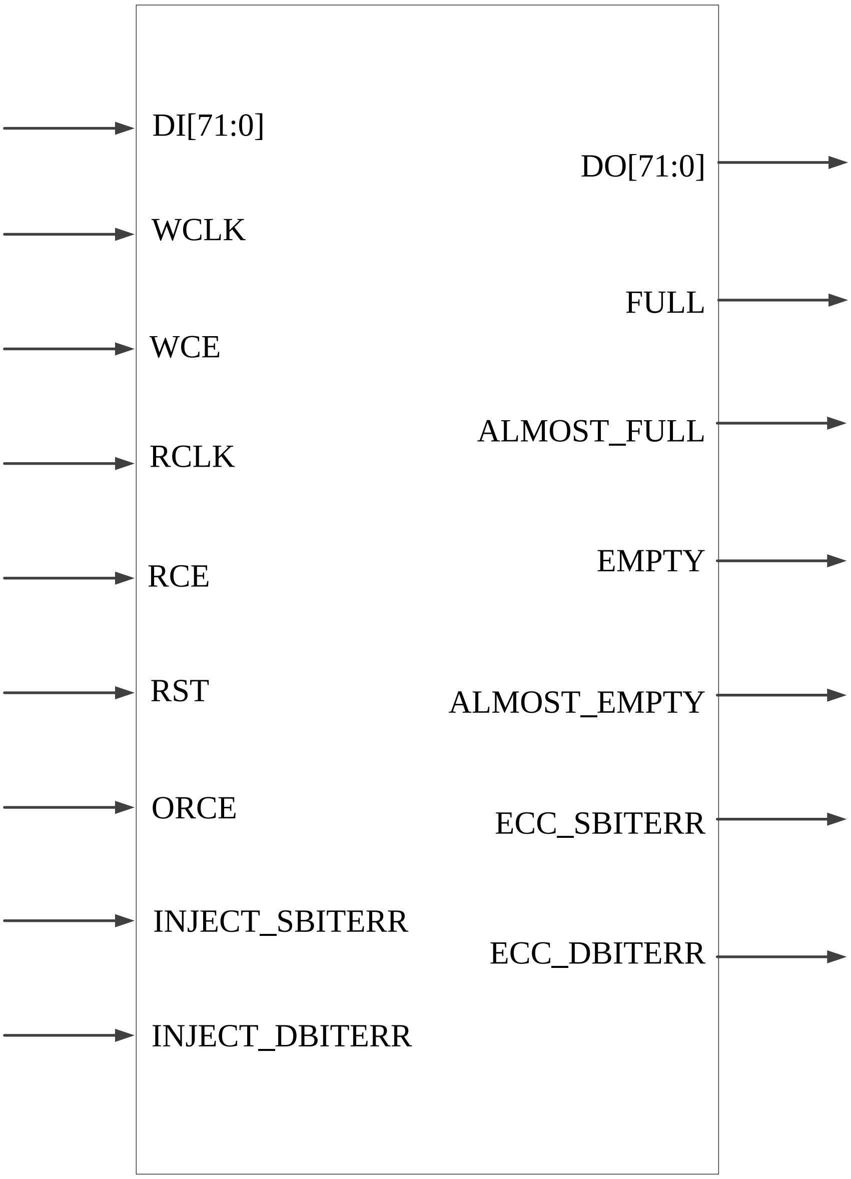

Figure 2-3 GTP_FIFO36K_E1 Structure Diagram

Table 2-5 GTP_FIFO36K_E1 Port Descriptions

Port	**Input/Output**	**Description**
DI[71:0]	Input	Data input
WCLK	Input	Write clock signal
RCLK	Input	Read clock signal
WCE	Input	Write enable signal
RCE	Input	Read enable signal
RST	Input	Reset signal
ORCE	Input	Output register clock enable signal
INJECT_SBITERR	Input	Single-bit error injection in ECC mode
INJECT_DBITERR	Input	Dual-bit error injection in ECC mode
DO[71:0]	Output	Data output
EMPTY	Output	Read port empty flag
FULL	Output	Write port full flag
ALMOST_EMPTY	Output	Read port almost empty flag
ALMOST_FULL	Output	Write port almost full flag
ECC_SBITERR	Output	Single-bit error flag in ECC mode
ECC_DBITERR	Output	Dual-bit error flag in ECC mode

Table 2-6 GTP_FIFO36K_E1 Parameter Descriptions

Parameter	**Description**	**Setting Value**
GRS_EN	Global reset enable signal (Internal Chip)	"TRUE": Global Reset enabled; "FALSE": Global Reset not enabled.
DATA_WIDTH	FIFO data width	1, 2, 4, 8, 9, 16, 18, 32, 36, 64, 72
SYNC_FIFO	Asynchronous/Synchronous FIFO Selection	"TRUE": Use synchronous FIFO; "FALSE": Use asynchronous FIFO.
ALMOST_FULL_OFFSET	When the FIFO is almost full and the difference between the write and read pointers is greater than or equal to ALMOST_FULL_ OFFSET, the almost_full flag is set to 1.	DATA_WIDTH= 1bit: sync FIFO:1–32767 async FIFO:1–32764 2bits: sync FIFO:1–16383 async FIFO:1–16380 4bits: sync FIFO:1–8191 async FIFO:1–8188 8/9bits: sync FIFO:1–4095 async FIFO: 1–4092 16/18bits: sync FIFO:1–2047 async FIFO:1–2044 32/36bits: sync FIFO: 1–1023 async FIFO: 1–1020 64/72bits: sync FIFO: 1–511 async FIFO: 1–508

Parameter	**Description**	**Setting Value**
ALMOST_EMPTY_OFFSET	When the FIFO is almost empty and the difference between the read and write pointers is less than or equal to ALMOST_EMPTY_OFFSET, the almost_empty flag is set to 1.	DATA_WIDTH= 1bit: sync FIFO:1–32767 async FIFO:4–32767 2bits: sync FIFO:1–16383 async FIFO: 4–16383 4bits: sync FIFO:1–8191 async FIFO: 4–8191 8/9bits: sync FIFO:1–4095 async FIFO: 4–4095 16/18bits: sync FIFO:1–2047 async FIFO: 4–2047 32/36bits: sync FIFO: 1–1023 async FIFO: 4–1023 64/72bits: sync FIFO: 1–511 async FIFO: 4–511
ECC_WRITE_EN	Write port ECC mode enable	"TRUE": Enabled; "FALSE": Not enabled.
ECC_READ_EN	Read port ECC mode enable	"TRUE": Enabled; "FALSE": Not enabled.
RST_VAL	Output set/reset value	72'h00_0000_0000_0000_0000~ 72'hFF_FFFF_FFFF_FFFF_FFFF
USE_EMPTY	Enable read empty flag	1: Read empty flag enabled; 0: Read empty flag not enabled.
USE_FULL	Enable write full flag	1: Write full flag enabled; 0: Write full flag not enabled.
DO_REG	Output register enable	1: Enabled; 0: Not enabled

2.4 GTP_FIFO18K_E1

Instantiate GTP_FIFO18K_E1 to implement FIFO mode. For the GTP structure block diagram, refer to Figure 2-4. For port descriptions and parameter descriptions, refer to Table 2-7 and Table 2-8. For GTP_FIFO18K_E1 Instantiation Template, refer to the Appendix.

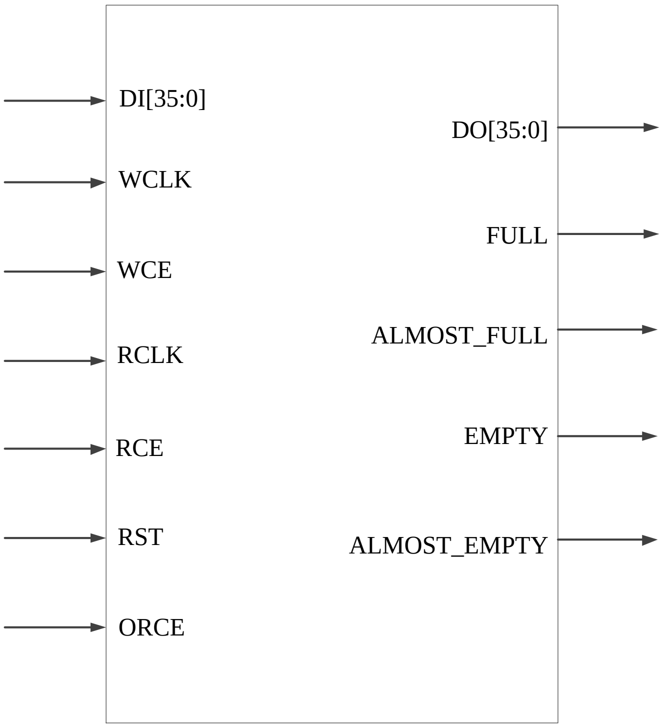

Figure 2-4 GTP_FIFO18K_E1 Structure Diagram

Table 2-7 GTP_FIFO18K_E1 Port Descriptions

Port	**Input/Output**	**Description**
DI[35:0]	Input	Data write in
WCLK	Input	Write clock signal
RCLK	Input	Read clock signal
WCE	Input	Write enable signal
RCE	Input	Read enable signal
RST	Input	Reset signal
ORCE	Input	Output register clock enable signal
DO[35:0]	Output	Data read out
EMPTY	Output	Read port empty flag
FULL	Output	Write port full flag
ALMOST_EMPTY	Output	Read port almost empty flag
ALMOST_FULL	Output	Write port almost full flag

Table 2-8 GTP_FIFO18K_E1 Parameter Descriptions

Parameter	Description	Setting Value
GRS_EN	Global reset enable signal (Internal Chip)	"TRUE": Global Reset enabled; "FALSE": Global Reset not enabled.
DATA_WIDTH	FIFO data width	1,2,4,8,9,16,18,32,36
SYNC_FIFO	Asynchronous/Synchronous FIFO Selection	"TRUE": Use synchronous FIFO; "FALSE": Use asynchronous FIFO.
ALMOST_FULL_OFFSET	When the FIFO is almost full and the difference between the write and read pointers is less than or equal to ALMOST_FULL_OFFSET, the almost_full flag is set to 1.	DATA_WIDTH= 1bit: sync FIFO:1–16383 async FIFO:1–16380 2bits: sync FIFO:1–8191 async FIFO:1–8188 4bits: sync FIFO:1–4095 async FIFO: 1–4092 8/9bits: sync FIFO:1–2047 async FIFO:1–2044 16/18bits: sync FIFO: 1–1023 async FIFO: 1–1020 32/36bits: sync FIFO: 1–511 async FIFO: 1–508
ALMOST_EMPTY_OFFSET	When the FIFO is almost empty and the difference between the read and write pointers is less than or equal to ALMOST_EMPTY_OFFSET, the almost_empty flag is set to 1.	DATA_WIDTH= 1bit: sync FIFO:1–16383 async FIFO: 4–16383 2bits: sync FIFO:1–8191 async FIFO: 4–8191 4bits: sync FIFO:1–4095 async FIFO: 4–4095 8/9bits: sync FIFO:1–2047 async FIFO: 4–2047 16/18bits: sync FIFO: 1–1023 async FIFO: 4–1023 32/36bits: sync FIFO: 1–511 async FIFO: 4–511
RST_VAL	Output set/reset value	36’h0 – 36’hF_FFFF_FFFF
USE_EMPTY	Enable read empty flag	1: Read empty flag enabled; 0: Read empty flag not enabled.
USE_FULL	Enable write full flag	1: Write full flag enabled; 0: Write full flag not enabled.
DO_REG	Output register enable	1: Enabled; 0: Not enabled

Chapter 3 DP Mode

3.1 Introduction

The port mode of RAM is determined by the parameter RAM_MODE. When the value of RAM_MODE is "TRUE_DUAL_PORT", RAM enters DP mode. This document will detail the 36K DP RAM, which has essentially the same structure and functions as the 18K, with the only difference being the bit width.

DP RAM supports:

- Dual-port read operation
- Dual-port write operation
- Port A/B to read and Port B/A to write
- Independent bit width settings for two ports

3.2 Ports Descriptions

In DP RAM mode, the DP RAM has Ports A and B, which are completely independent except for sharing DRM content. Their structures are completely symmetrical. Figure 3-1 shows the structure of the 36K DP RAM.

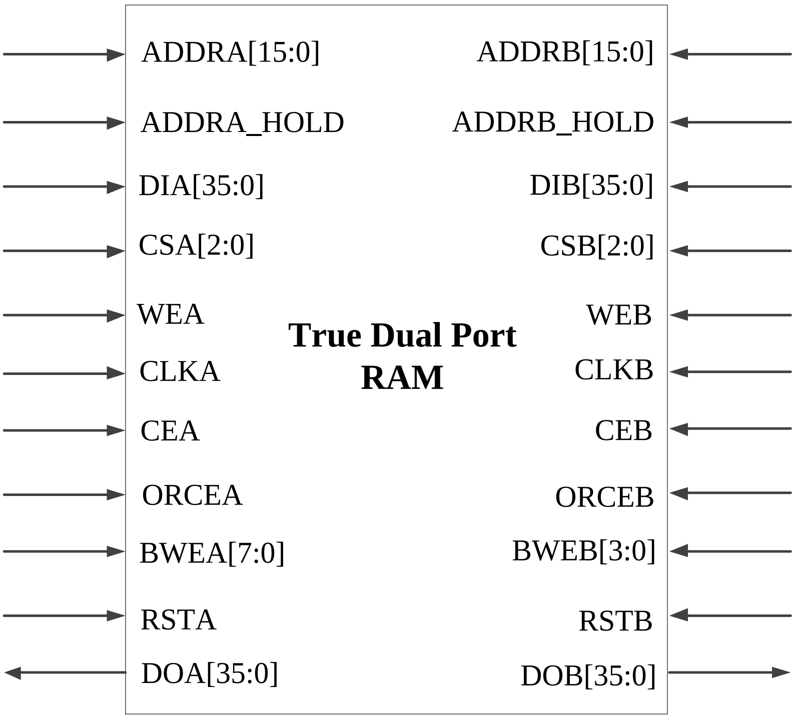

Figure 3-1 DP RAM Data Ports

Table 3-1 lists the port names and descriptions for DP RAM mode.

Table 3-1 Port Names and Descriptions of DP RAM Mode

Port	Direction	Description	Port	Direction	Description
ADDRA	Input	Port A address input	ADDRB	Input	Port B address input
ADDRA_HOLD	Input	Port A address hold signal	ADDRB_HOLD	Input	Port B address hold signal
DIA	Input	Port A data input bus	DIB	Input	Port B data input bus
CSA	Input	Port A address extension	CSB	Input	Port B address extension
WEA	Input	Port A write enable	WEB	Input	Port B write enable
CLKA	Input	Port A clock	CLKB	Input	Port B clock
CEA	Input	Port A input register clock enable	CEB	Input	Port B input register clock enable
ORCEA	Input	Port A output register clock enable	ORCEB	Input	Port B output register clock enable
RSTA	Input	Port A output register reset	RSTB	Input	Port B output register reset
BWEA	Input	Port A byte enable signal	BWEB	Input	Port B byte enable signal
DOA	Output	Port A data output bus	DOB	Output	Port B data output bus

Note: The 18K DRM has no CSA/CSB port and does not support address extension. To extend address depth through DRM cascading, it is recommended to use 36K DRM.

3.3 Bit Width Combinations

The port bit width of RAM is determined by the parameter DATA_WIDTH_A/DATA_WIDTH_B in the GTP. For example, when the value of DATA_WIDTH_A is 4, the data width of Port A is set to 4 bits. DP mode supports setting different data widths for Ports A and B.

Table 3-2 shows the allowed bit width combinations for True Dual Port RAM mode in 36K DRM mode.

Table 3-2 Allowed Bit Width Combinations for True Dual Port RAM Mode in 36K DRM Mode

		Port B								
		32Kx1	**16Kx2**	**8Kx4**	**4Kx8**	**2Kx16**	**1Kx32**	**4Kx9**	**2Kx18**	**1Kx36**
Port A	**32Kx1**	√	√	√	√	√	√			
	16Kx2	√	√	√	√	√	√			
	8Kx4	√	√	√	√	√	√			
	4Kx8	√	√	√	√	√	√			
	2Kx16	√	√	√	√	√	√			
	1Kx32	√	√	√	√	√	√			
	4Kx9							√	√	√
	2Kx18							√	√	√
	1Kx36							√	√	√

Note: √ indicates the supported bit width combinations.

Table 3-3 shows the allowed bit width combinations for True Dual Port RAM mode in 18K DRM mode.

Table 3-3 Allowed Bit Width Combinations for True Dual Port RAM Mode in 18K DRM Mode

		Port B 0/1						
		16Kx1	**8Kx2**	**4Kx4**	**2Kx8**	**1Kx16**	**2Kx9**	**1Kx18**
Port A 0/1	**16Kx1**	√	√	√	√	√		
	8Kx2	√	√	√	√	√		
	4Kx4	√	√	√	√	√		
	2Kx8	√	√	√	√	√		
	1Kx16	√	√	√	√	√		
	2Kx9						√	√
	1Kx18						√	√

Note: √ indicates the supported bit width combinations.

3.4 Timing Parameter

Table 3-1 lists the typical timing parameters and descriptions for 36K DRM. The timing parameters and diagrams provided hereinafter are all based on 36K DRM. The timing parameters and diagrams for 18K DRM are similar to those of 36KCM. For detailed timing parameters, refer to "***DS04001_Logos2 Family FPGA Device Datasheet***".

Table 3-4 Typical Timing Parameters for DRM

Parameter	**Control Signals**	**Description**
$T_{su\_36K\_ce}$	CEA/B	Clock enable signal setup time
$T_{hd\_36K\_ce}$		Clock enable signal hold time
$T_{su\_36K\_we}$	WEA/B	Write enable signal setup time
$T_{hd\_36K\_we}$		Write enable signal hold time
$T_{su\_36K\_be}$	BEA/B	Byte write enable signal setup time
$T_{hd\_36K\_be}$		Byte write enable signal hold time
$T_{su\_36K\_ad}$	ADA/B	Address input signal setup time
$T_{hd\_36K\_ad}$		Address input signal hold time
$T_{su\_36K\_d}$	DA/B	Data input signal setup time
$T_{hd\_36K\_d}$		Data input signal hold time
$T_{su\_36K\_oe}$	OCEA/B	Output register enable signal setup time
$T_{hd\_36K\_oe}$		Output register enable signal hold time
$T_{su\_36K\_rst}$	RSTA/B	Output register/latch synchronous reset signal setup time
$T_{hd\_36K\_rst}$		Output register/latch synchronous reset signal hold time
T_{co_36K}	CLK to Q	Data output delay with respect to clock edge (without output register)
$T_{co\_36K\_reg}$		Data output delay with respect to clock edge (with output register)

Figure 3-2 is the DRM timing diagram. The write timing in the diagram is based on the TW write mode of Port A, which is the same as that of the write mode of Port B. Refer to 3.5 Read and Write Operations for the write timings of other write modes. The read timings are similar in different write modes. The reset timing uses asynchronous reset as an example. When synchronous reset is used, the reset signal needs to meet both the setup and hold time requirements. Mem in the diagram refers to the old data stored at the corresponding address.

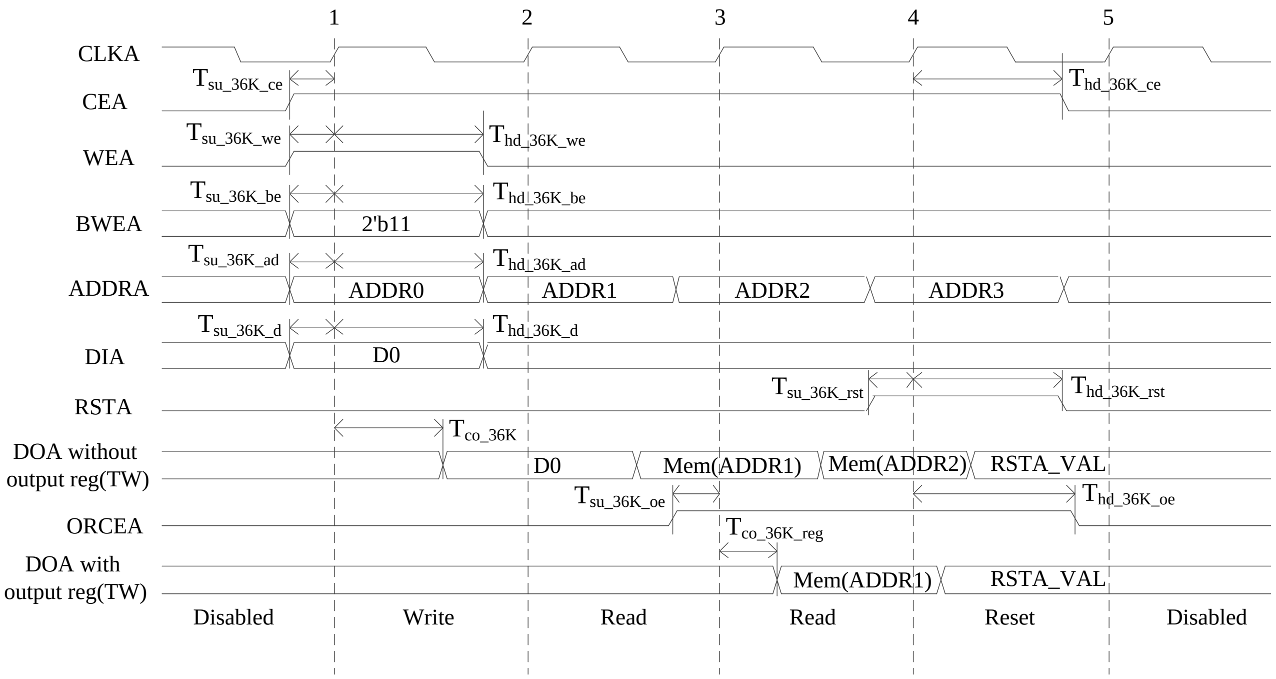

Figure 3-2 DRM Timing Diagram (TW Write Mode of Port A in DP Mode)

Before the 1st clock rising edge, CEA, WEA and BWEA are pulled high and meet their respective setup time requirements. CSA, ADDRA, and DIA signals reach a stable state and meet the setup time requirements; after correctly sampling the control signals, data, and address signals, DRM writes the input data D0 into address ADDR0. D0 appears at output port DOA in the same cycle after an output delay of T_{co_36K};

Before the 2nd clock rising edge, WEA and BWEA are pulled down after meeting the hold time requirements, while CEA remains high and ADDRA reaches a stable state and meets the setup time requirement; after correctly sampling the control signals, data, and address signals, DRM reads the data of address ADDR1, and the data appears at output port DOA in the same cycle after an output delay of T_{co_36K};

Before the 3rd clock rising edge, ADDRA reaches a stable state and meets the setup time requirements; after correctly sampling the control signal, data, and address signal, DRM reads the data of address ADDR2, and the data appears at output port DOA in the same cycle after an output delay of T_{co_36K};

Before the 4th clock rising edge, RSTA reaches a stable state and meets the $T_{su\_36K\_rst}$ time requirements, and output port DOA outputs the default reset value RSTA_VAL;

Before the 5th clock rising edge, CEA is pulled down after meeting the hold time requirements, and DRM is disabled, with the output port unchanged;

If the output register is enabled, the DRM output data will be delayed for an additional cycle;

however, the reset operation will be completed in the current cycle without delay.

3.5 Read and Write Operations

Depending on the data output from the same port during data writing, the port write operations of DRM support three modes: Normal Write mode (NW), Transparent Write mode (TW), and Read before Write mode (RBW), with the default write mode set to NW. The write mode of the RAM is determined by the parameter WRITE_MODE_A/WRITE_MODE_B in the GTP. When the value of WRITE_MODE_A is "NORMAL_WRITE", the write mode for Port A of the DRM is set to NW. The read timings are similar in different write modes.

In DP mode, there are two relatively independent ports. Performing a read and a write operation on the same address through both ports simultaneously will cause a conflict. DRM prohibits both ports from writing data to the same address simultaneously and from performing a read and a write on the same address at the same time; this must be avoided in actual applications through user logic.

3.5.1 Normal Write Mode

Figure 3-3 is the read-write timing diagram of Normal Write mode. When a user writes data via a port of the DRM, the output data of that port is not updated at this time. Mem in the diagram refers to the old data stored at the corresponding address.

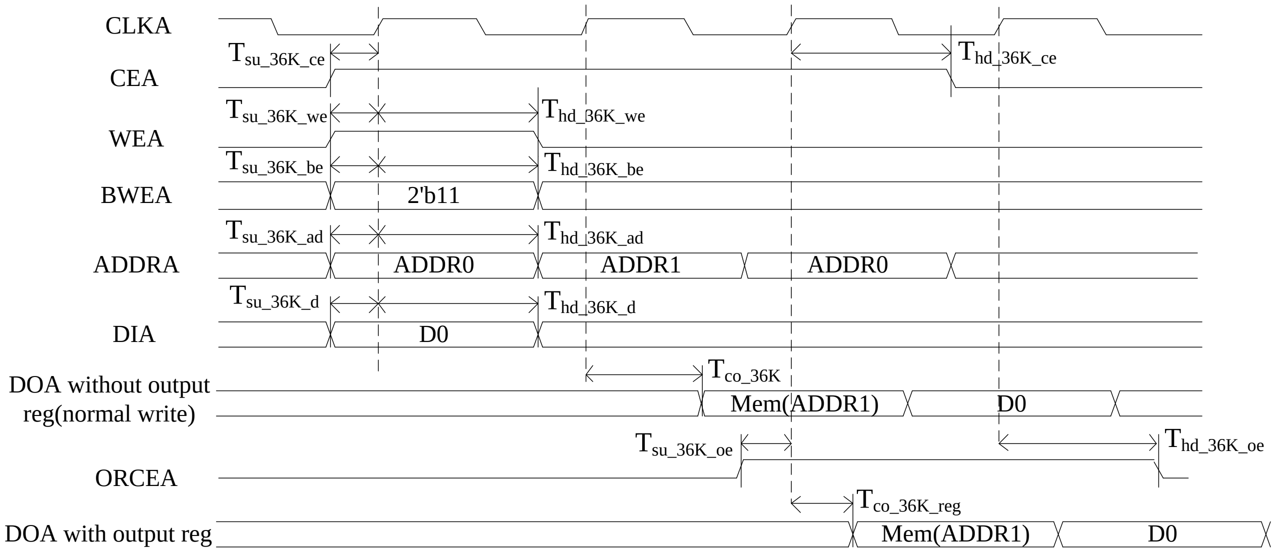

Figure 3-3 Read-Write Timing Diagram of Normal Write Mode

3.5.2 Transparent Write Mode

Figure 3-4 is the read-write timing diagram of Transparent Write mode. When a user writes data via a port of the DRM, the written data is directly output to the output port at the same time as it is written to the RAM, with an output delay of T_{co_36K} from the clock rising edge to data read. Mem in the diagram refers to the old data stored at the corresponding address.

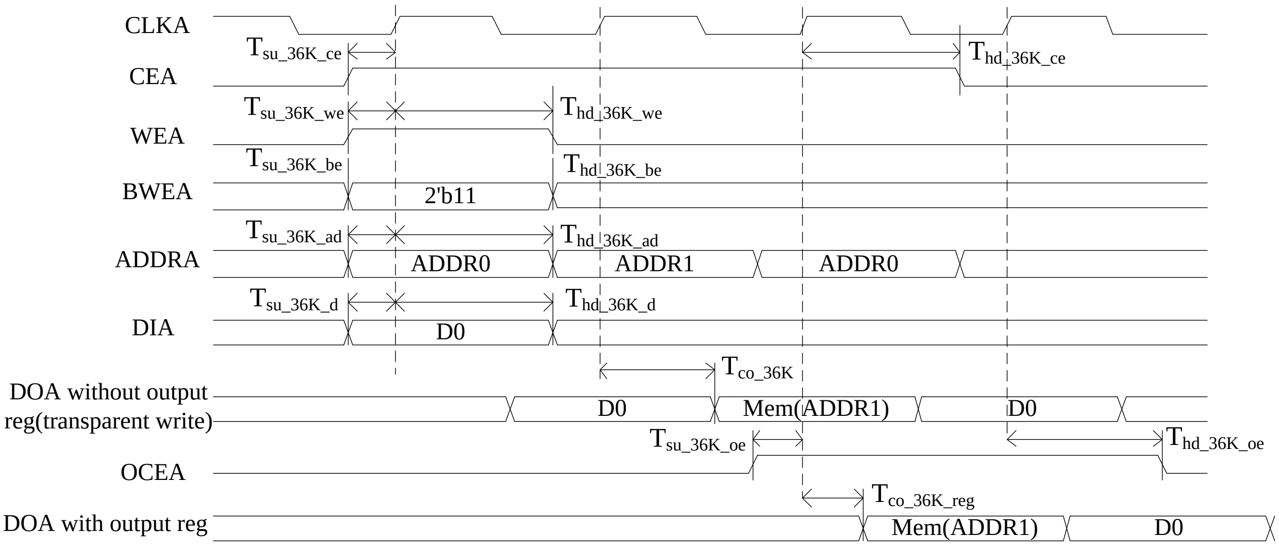

Figure 3-4 Read-Write Timing Diagram of Transparent Write Mode

3.5.3 Read Before Write Mode

Figure 3-5 is the read-write timing diagram of Read before Write mode. When a user writes data via a port of the DRM, the original data at that address is first read and output to the output port, with an output delay of T_{co_36K} from the clock rising edge to data read. Mem in the diagram refers to the old data stored at the corresponding address.

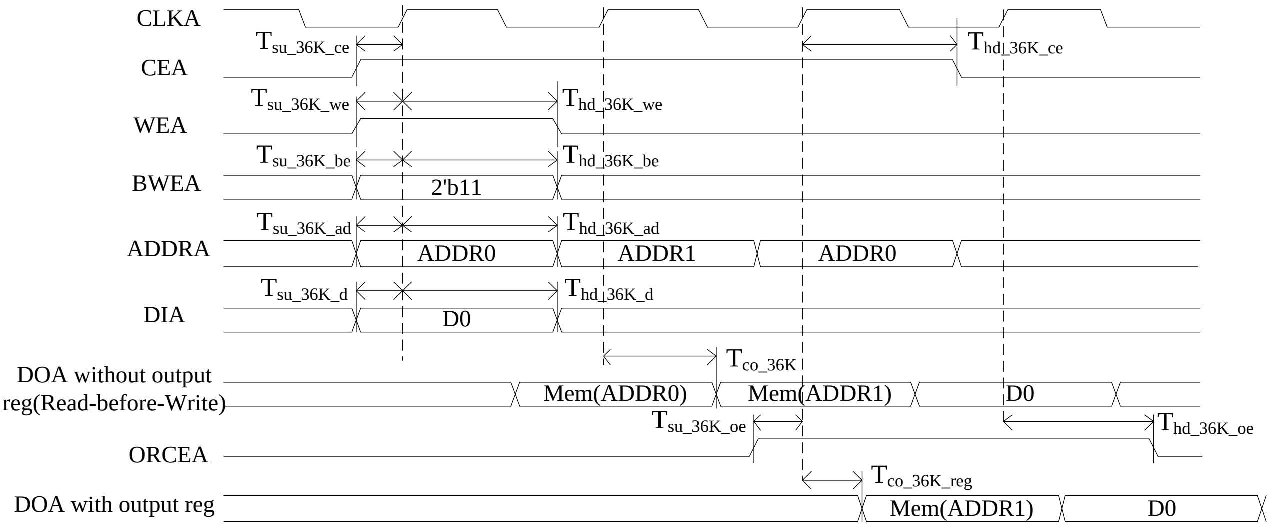

Figure 3-5 Read-Write Timing Diagram of Read before Write Mode

3.6 Byte Enable

DRM supports the Byte-Write mode for write operations, which writes to selected data bytes through the BWEA/BWEB signals (active-high) while masking the writing to other bytes at the same address index. This mode is mainly used to operate on narrower data buses when the data bus width is limited. For example, operations can be performed only on 9-bit data on an 18-bit data bus. In DP mode, when the port width is 2^N bits, byte enable for 16/32-bit write operations is supported, with each byte containing 8 bits. When the port width is 9×2^N bits, byte enable for 18/36-bit write operations is supported, with each byte containing 9 bits. For instance, when the data width of Port A is 32 bits, BWEA contains four bits, with BWEA[3] controlling data [31:24] and BWEA[2] controlling data [23:16]; and so on. When BWEA[3:0]=4'b0001, only data[7:0] is written to the current address during a valid write operation. During design, the following two aspects need to be considered:

- Port A and Port B have independent byte enable controls.
  - Port A: BWEA[7:0], where BWEA[7:4] is only valid in SDP mode and SP mode, and inactive in DP mode;
  - Port B: BWEB[3:0].
- The byte enable signal is always active in x18(16)/x36(32) bit data width mode; when not in use, this signal should be connected to a high level.

In DP mode, the byte enable write operation mode only supports the data port width for writing of x18(16)/x36(32). When the data port width for writing is x9(x8), the byte enable function is prohibited, as it duplicates the write enable function, and the byte enable signal in the GTP becomes inactive.

The byte enable write operation mode can coexist with NW, TW, or RBW write operation modes., however, it should be noted that in TW Mode, while the 16/18/32/36 data bit width byte enable for writing, if there is any 1 bit change in the byte enable signal compared to the previous clock cycle, the transparent read data will be invalid at this time. Figure 3-6 shows the read/write timing diagram of DRM in byte enable mode under TW write mode, where Mem refers to the old data stored at the corresponding address.

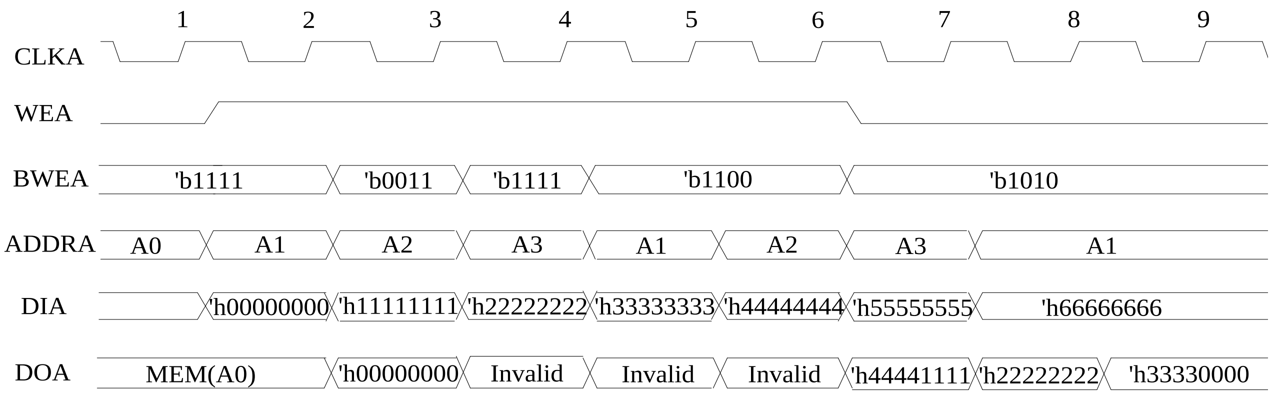

Figure 3-6 Byte Enable Read/Write Timing Diagram (TW Mode)

On the rising edge 2 of the clock, the write enable WEA is valid, but the byte enable signal BWEA has not changed compared to the previous clock cycle, so the read and write behavior is normal; On the rising edge 3,4, and 5 of the clock, the write enable WEA is valid and the byte enable signal BWEA has changed compared to the previous clock cycle. Therefore, the data is written to the DRM normally, but the output data from DOA port is Invalid; On the rising edge 6 of the clock, the write enable WEA is valid, but the byte enable signal BWEA has not changed compared to the previous clock cycle, so the read and write behavior is normal; On the rising edge 7 of the clock, the byte enable signal BWEA has changed compared to the previous clock cycle but the write enable WEA is 0. At this time, no byte enable write operation is performed, so the data stored in address

A3 is read out normally.

3.7 Internal Registers

The DRM internal registers include Input Register (IR), Output Register (OR), and Internal Latch.

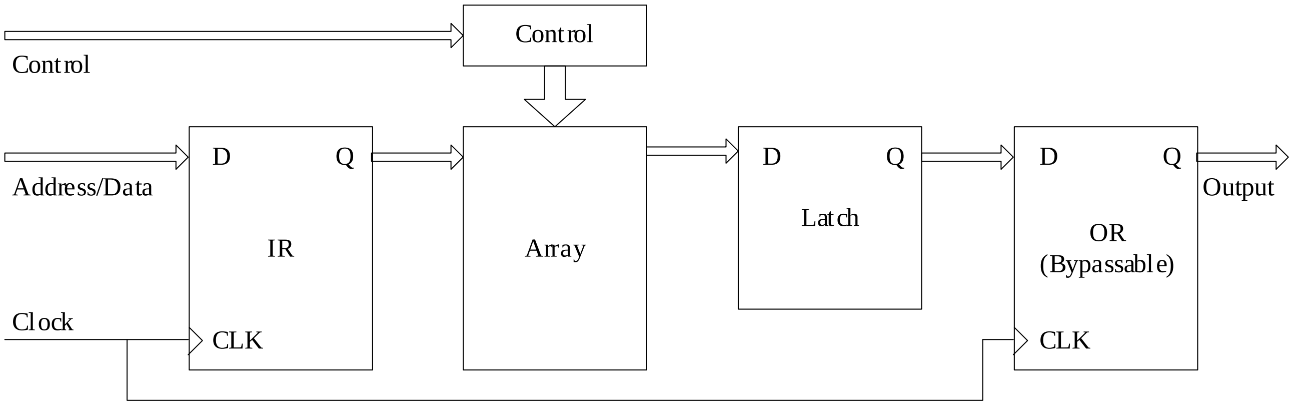

Figure 3-7 Logic Diagram of DRM Registers

Write operations can be synchronized through Input Registers (IRs) for port addresses, data, and some control signals. The IRs cannot be bypassed or configured. An address hold selection path is built into the address input port, with its logic implementation as shown in Figure 3-8:

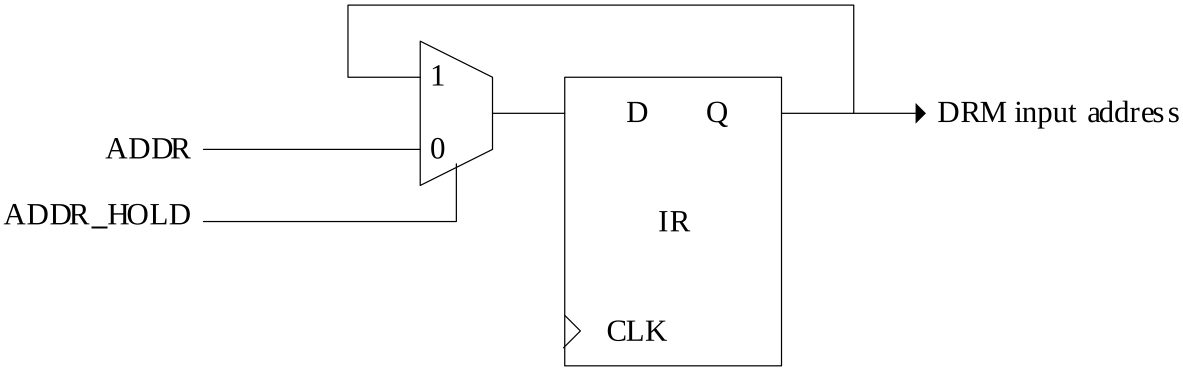

Figure 3-8 Logic Diagram of Address IR

Address hold signal ADDRA_HOLD controls the address hold selection path. If ADDRA_HOLD remains at a high level, the output address data from the input register remains unchanged, with the timing diagram as shown in Figure 3-9:

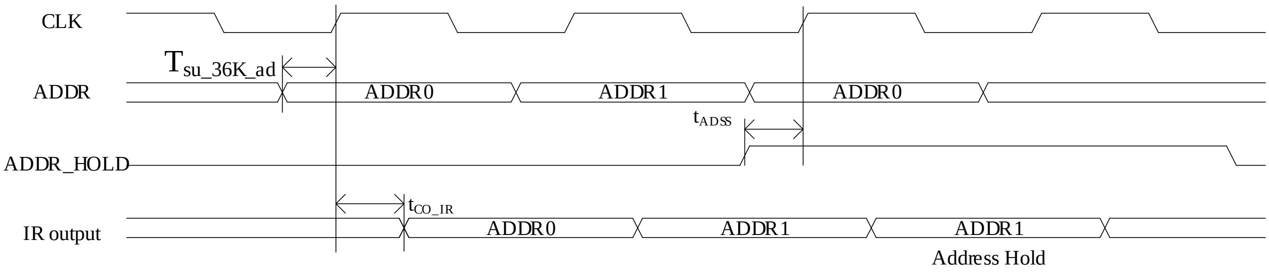

Figure 3-9 Timing Diagram of Address IR

For data output ports, DRM provides an optional Output Register (OR) for improved timing performance. Whether the output register is valid is determined by the GTP parameter DOA_REG/DOB_REG. For example, when DOA_REG is 1, Port A's output register is valid; when DOA_REG is 0, Port A's output register is bypassed. During read operations, when the output register is bypassed, DOA/DOB output is the latch output, occurring on the rising edge of the same read clock cycle. Port B is similar.

As shown in Figure 3-10, when Port A's output register is valid, it can be controlled by an independent clock enable signal ORCEA. When ORCEA is at a high level, the output data from the output register changes with the input on every rising edge of the clock cycle; when ORCEA is at a low level, the output data from the output register remains unchanged; when the output register is bypassed, ORCEA is inactive. When ORCEA is a constant 1, the use of the output register increases the delay of Port A read operation from one clock cycle to two clock cycles (Case 1); in pipeline design applications with flow control, the user can also flexibly control ORCEA with logic (Case 2). Mem in the diagram refers to the old data stored at the corresponding address.

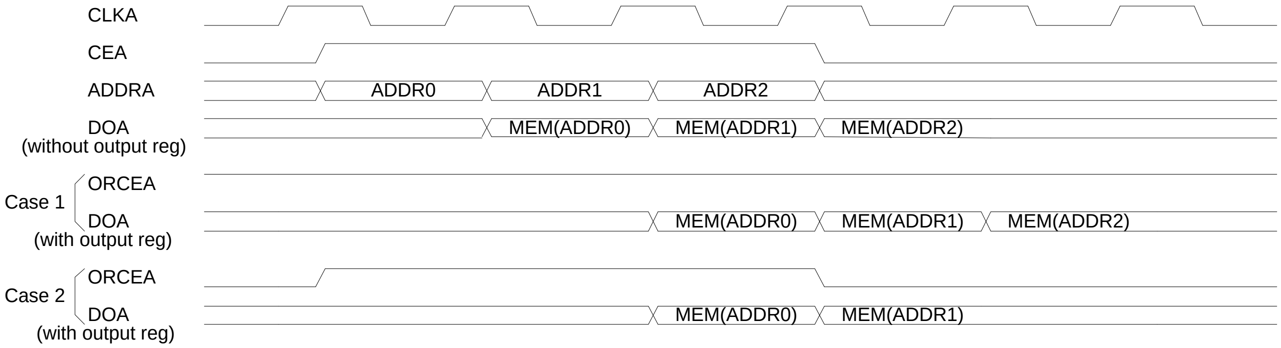

Figure 3-10 Read Timing with Output Register

3.8 Hard Cascade Function

In 36K mode, two adjacent DRMs can be cascaded to a 64Kx1 memory without using external logic resources. The A/B ports of the adjacent DRMs should be configured as 32Kx1 mode simultaneously. The top-level cascade direction is from bottom to top, as shown in Figure 3-11:

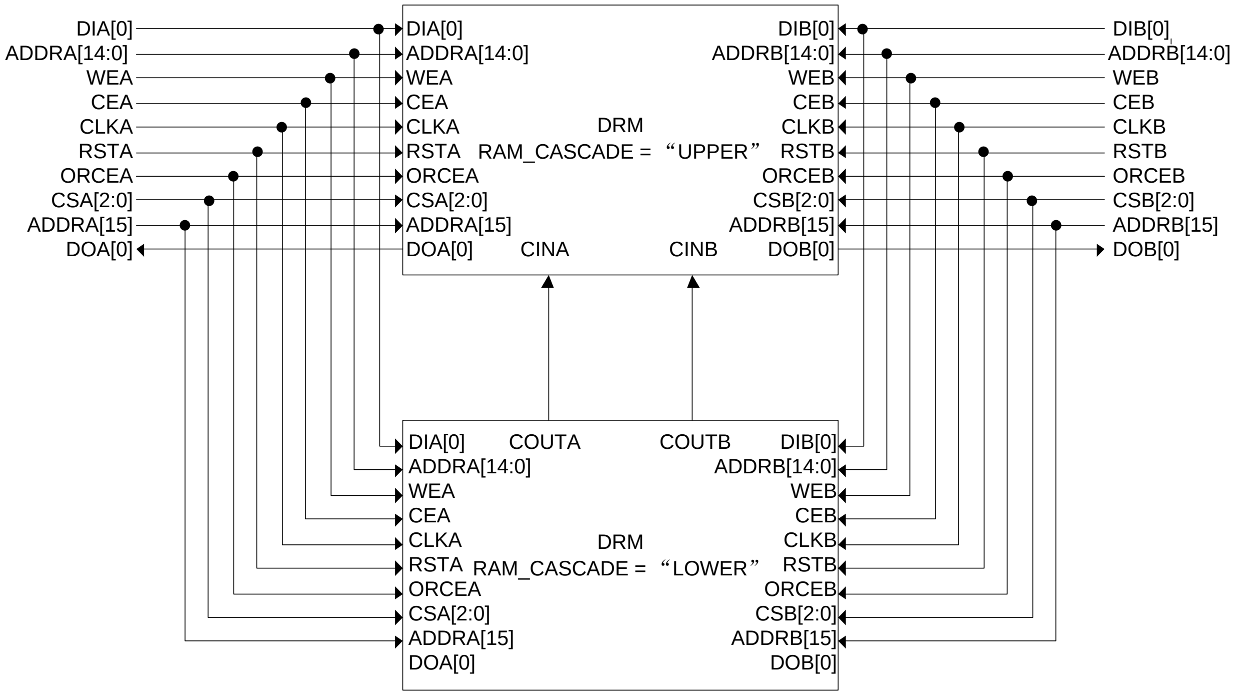

Figure 3-11 Hard Cascade Diagram

In hard cascade mode, the input control, data, and address of the two cascaded DRMs should be connected to the same source. Simultaneously, the mode configuration (parameters other than RAM_CASCADE in GTP) should be the same. When parameter RAM_CASCADE = "UPPER", DRM cascades with the adjacent DRM below, with the parameter of DRM below configured as RAM_CASCADE = "LOWER". The cascaded DRM below stores the corresponding upper 32K address, and the DRM above stores the corresponding lower 32K address. In non-hard cascade mode, the address hard cascade signal, i.e., the highest bit ADDRA[15]/ADDRB[15] of address inputs of Port A/B, needs to be connected to a high-level input.

The hard cascade function extends the address depth of DRM to 64K, i.e., the address width is extended from 15 bits to 16 bits, but the data width is fixed at 1. For a larger data width, cascade multiple basic units of hard cascade—64Kx1 memories (i.e., concatenating data ports, with other parameters and ports unchanged) to form a 64KxN memory; for a larger address width, perform address cascade on the basis of hard cascade. Refer to 3.9.3 Hard Cascade Configuration for Multiple 36K DRMs.

Compared with ordinary address cascade, the hard cascade function does not require external logic resources, and performs data carry through the cascaded data input and output ends of two adjacent DRMs (CINA/CINB and COUTA/COUTB), offering better performance.

3.9 Application Example

3.9.1 Single 36K DRM Configuration

This section illustrates the GTP configuration steps for a single 36K DRM. Users can also directly generate the DRM IP using the IP Compiler tool embedded in the Pango Design Suite software. Refer to "DRM/FIFO IP User Guide UG041002" attached in the IP Compiler tool.

The example is for DP mode with mixed bit widths, with Port A configured as 4Kx8, Port B as 1Kx32 in byte enable write mode. Two ports have different clocks, enabling output registers, in TW write mode. The configuration steps for an 18K DRM are similar.

Configure a single 36K DRM as follows:

1. Configure the DRM parameters as per Table 3-5:

Table 3-5 Parameter Configuration of DP Mode for a Single 36K DRM

Parameter Name	**Configuration Value**	**Description**
DATA_WIDTH_A	8	Configure Port A to 4Kx8 mode
DATA_WIDTH_B	32	Configure Port B to 1Kx32 mode
WRITE_MODE_A	"TRANSPARENT_WRITE"	Configure the write mode of Port A to TW
WRITE_MODE_B		Configure the write mode of Port B to TW
DOA_REG	1	Enable Port A output register
DOB_REG		Enable Port B output register
RAM_MODE	"TRUE_DUAL_PORT"	Configure the DRM to DP mode

2. Connect the DRM ports as per Table 3-6:

Table 3-6 Port Connections of DP Mode for a Single 36K DRM

Port	**Interfacing Signals**	**Description**
ADDRA[15:0]	{1'b1,addra[11:0],3'b0}	Connect the input address signal addra[11:0] to ADDRA[14:3], ADDRA[15] to a high level, and ADDRA[2:0] to a low level. Refer to 9.1 Address and Data Port Mapping for detailed address connection instructions.
ADDRB[15:0]	{1'b1,addrb[9:0],5'b0}	Connect the input address signal addrb[9:0] to ADDRB[14:5], ADDRB[15] to a high level, and ADDRB[4:0] to a low level.
ADDRA_HOLD	1'b0	Do not use Port A/B address hold function, connected to a low level
ADDRB_HOLD		
DIA[35:0]	dia[7:0]	Connect the input data signal dia[7:0] to DIA[7:0], with the unused high-bit DIA ports floating. Refer to 9.1 Address and Data Port Mapping for detailed data port connection instructions.
DIB[35:0]	{1'b0, dib[31:24], 1'b0, dib[23:16], 1'b0, dib[15:8], 1'b0, dib[7:0]}	Connect the input data signal dib[31:0] to the GTP port {DIB[34:27], DIB[25:18], DIB[16:9], DIB[7:0]}, with DIB[8], DIB[17], DIB[26] and DIB[35] as additional byte information bits. Refer to 9.4 Additional Information Bits for Bytes. Leave the signal unused or connect it to a low level when the data width is 2^N.
CSA[2:0]	3'b0	Address extension function is not used, connected to a low level
CSB[2:0]		

Port	Interfacing Signals	Description
BWEA[7:0]	8'hff	Byte enable function is not used on Port A. Connect BWEA to a high level.
BWEB[3:0]	bweb	Byte enable function is not used on Port B. Connect the byte enable signal bweb to BWEB.
DOA[35:0]	doa[7:0]	Connect the output data signal doa[7:0] to the GTP port DOA[7:0]
DOB[35:0]	{1'bz, dob[31:24], 1'bz, dob[23:16], 1'bz, dob[15:8], 1'bz, dob[7:0]}	Connect the output data signal dob[31:0] to the GTP port {DOB[34:27], DOB[25:18], DOB[16:9], DOB[7:0]}, with DOB[8], DOB[17], DOB[26] and DOB[35] left unused.

3. Other signals for A/B ports: Connect the Port A/B's clock, clock enable, write enable, output register clock enable, and data register reset to the corresponding GTP ports;
4. Other unused parameters: Do not set other unused parameters, use default values;
5. Other unused ports: Leave other unused ports floating (ports that must be connected even if not used have been described above).

The configured GTP is as follows:

```
GTP_DRM36K_E1 #(
.DATA_WIDTH_A    (8),
.DATA_WIDTH_B    (32),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.DOA_REG         (1),
.DOB_REG         (1),
.RAM_MODE        ("TRUE_DUAL_PORT")
) GTP_DRM36K_E1_inst (
.DOA             (doa[7:0]    ), // OUTPUT[35:0]
.DOB             (dob_gtp[35:0]), // OUTPUT[35:0]
.ADDRA           ({1'b1,addra[11:0],3'b0}), // INPUT[15:0]
.ADDRB           ({1'b1,addrb[9:0],5'b0}), // INPUT[15:0]
.BWEA            (8'hff       ), // INPUT[7:0]
.BWEB            (bweb[3:0] ), // INPUT[3:0]
.CSA             (3'b0        ), // INPUT[2:0]
.CSB             (3'b0        ), // INPUT[2:0]
.DIA             (dia[7:0]    ), // INPUT[35:0]
.DIB             ({1'b0,dib[31:24],1'b0,dib[23:16],1'b0,dib[15:8],1'b0,dib[7:0]}), // INPUT[35:0]
.ADDRA_HOLD      (1'b0        ), // INPUT
.ADDRB_HOLD      (1'b0        ), // INPUT
.CEA                (cea          ), // INPUT
.CEB                (ceb          ), // INPUT
.CLKA               (clka         ), // INPUT
.CLKB               (clkb         ), // INPUT
.ORCEA              (orcea        ), // INPUT
.ORCEB              (orceb        ), // INPUT
.RSTA               (rsta         ), // INPUT
.RSTB               (rstb         ), // INPUT
.WEA                (wea          ), // INPUT
.WEB                (web          )   // INPUT
);
assign dob[31:0] = {dob_gtp[34:27],dob_gtp[25:18],dob_gtp[16:9],dob_gtp[7:0]};
```

3.9.2 Configuration for Multiple 36K DRMs Cascaded

The cascading mentioned in this document specifically refers to address depth cascading. For data cascading, each DRM has the same configuration, so users only need to concatenate the data ports, which will not be separately described in this document.

Address depths are cascaded through 36K DRMs without the hard cascade function or external logic to process the address. Up to 3 bits of DRM address can be cascaded (CSA/CSB bit width is 3), with a total address width of 18 bits; the maximum data width for data cascading depends on the device's DRM resources.

This section illustrates the GTP cascading configuration steps for multiple 36K DRMs. Users can also directly generate the IP of the cascaded DRM using the IP Compiler tool embedded in the Pango Design Suite software. Refer to "DRM/FIFO IP User Guide UG041002" attached in the IP Compiler tool.

Four DP 36K DRMs are cascaded in the example, each with Ports A/B configured as 32Kx1, with a total data width of 2 bits and a total address width of 16 bits. Since the 18K DRM lacks address extension related signals and parameters, it is not recommended to use 18K DRM for cascading.

Name the 4 cascaded DRMs DRM_1_1, DRM_1_2, DRM_2_1, and DRM_2_2, respectively. Taking DRM_1_2 as an example, the first digit 1 represents the address cascading coordinate, with 2 levels of address cascading, extending the address from 15 bits of a single DRM to 16 bits; the second digit 2 represents the data cascading coordinate, with 2 levels of data cascading, extending the data from 1 bit of a single DRM to 2 bits. To cascade multiple 36K DRMs, configure them as follows:

1. Single DRM configuration: Configure the four 36K DRMs as 32Kx1 DP 36K DRMs, following similar steps in 3.9.1 Single 36K DRM Configuration;
2. Configure the DRM address extension control parameters as per Table 3-7:

Table 3-7 Cascading Parameter Configuration of DP Mode for a Single 36K DRM

Parameter Name	DRM Name	Configuration Value
CSA_MARK	DRM_1_1	3'b000
	DRM_1_2	3'b000
	DRM_2_1	3'b001
	DRM_2_2	3'b001
CSB_MARK	DRM_1_1	3'b000
	DRM_1_2	3'b000
	DRM_2_1	3'b001
	DRM_2_2	3'b001

3. Connect the cascade-related ports of the DRM as per Table 3-8:

Table 3-8 Cascading Port Connections of DP Mode for a Single 36K DRM

Port	DRM Name	Interfacing Signals
CSA	DRM_1_1	{2'b0,addra[15]}
	DRM_1_2	
	DRM_2_1	
	DRM_2_2	
CSB	DRM_1_1	{2'b0,addrb[15]}
	DRM_1_2	
	DRM_2_1	
	DRM_2_2	
ADDRA[15:0]	DRM_1_1	{1'b1,addra[14:0]}
	DRM_1_2	
	DRM_2_1	
	DRM_2_2	
ADDRB[15:0]	DRM_1_1	{1'b1,addrb[14:0]}
	DRM_1_2	
	DRM_2_1	
	DRM_2_2	
DIA[35:0]	DRM_1_1	dia[0]
	DRM_1_2	dia[1]
	DRM_2_1	dia[0]
	DRM_2_2	dia[1]

Port	DRM Name	Interfacing Signals
DIB[35:0]	DRM_1_1	dib[0]
	DRM_1_2	dib[1]
	DRM_2_1	dib[0]
	DRM_2_2	dib[1]
DOA[35:0]	DRM_1_1	doa_0[0]
	DRM_1_2	doa_0[1]
	DRM_2_1	doa_1[0]
	DRM_2_2	doa_1[1]
DOB[35:0]	DRM_1_1	dob_0[0]
	DRM_1_2	dob_0[1]
	DRM_2_1	dob_1[0]
	DRM_2_2	dob_1[1]

4. Output data selection: When addra[15] is 0 after a delay of one beat through a register, the data output from Port A is doa_0[1:0], and when it is 1, the data output from Port A is doa_1[1:0]. Port B operates similarly.

The configured GTP is as follows:

```
GTP_DRM36K_E1 #(
.CSA_MASK           (3'b000),
.CSB_MASK           (3'b000),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.RAM_MODE           ("TRUE_DUAL_PORT")
) DRM_1_1 (
.DOA                    ({doa_0[0]}), // OUTPUT[35:0]
.DOB                    ({dob_0[0]}), // OUTPUT[35:0]
.ADDRA                  ({1'b1,addra[14:0]}), // INPUT[15:0]
.ADDRB                  ({1'b1,addrb[14:0]}), // INPUT[15:0]
.BWEA                   (8'hff        ), // INPUT[7:0]
.BWEB                   (4'hf          ), // INPUT[3:0]
.CSA                   ({2'b0,addra[15]}), // INPUT[2:0]
.CSB                   ({2'b0,addrb[15]}), // INPUT[2:0]
.DIA                  ({dia[0]}    ), // INPUT[35:0]
```

```
.DIB                ({dib[0]}   ), // INPUT[35:0]
.ADDRA_HOLD         (addra_hold), // INPUT
.ADDRB_HOLD         (addrb_hold), // INPUT
.CEA                (cea        ), // INPUT
.CEB                (ceb        ), // INPUT
.CLKA               (clka       ), // INPUT
.CLKB               (clkb       ), // INPUT
.ORCEA              (1'b0       ), // INPUT
.ORCEB              (1'b0       ), // INPUT
.RSTA               (rsta       ), // INPUT
.RSTB               (rstb       ), // INPUT
.WEA                (wea        ), // INPUT
.WEB                (web        )   // INPUT
);
GTP_DRM36K_E1 #(
.CSA_MASK           (3'b000),
.CSB_MASK           (3'b000),
.DATA_WIDTH_A   (1),
.DATA_WIDTH_B   (1),
.WRITE_MODE_A   ("TRANSPARENT_WRITE"),
.WRITE_MODE_B   ("TRANSPARENT_WRITE"),
.RAM_MODE           ("TRUE_DUAL_PORT")
) DRM_1_2 (
.DOA                ({doa_0[1]}), // OUTPUT[35:0]
.DOB                ({dob_0[1]}), // OUTPUT[35:0]
.ADDRA              ({1'b1,addra[14:0]}), // INPUT[15:0]
.ADDRB              ({1'b1,addrb[14:0]}), // INPUT[15:0]
.BWEA               (8'hff      ), // INPUT[7:0]
.BWEB               (4'hf       ), // INPUT[3:0]
.CSA                ({2'b0,addra[15]}), // INPUT[2:0]
.CSB                ({2'b0,addrb[15]}), // INPUT[2:0]
.DIA                ({dia[1]}   ), // INPUT[35:0]
.DIB                ({dib[1]}   ), // INPUT[35:0]
.ADDRA_HOLD         (addra_hold), // INPUT
```

```
.ADDRB_HOLD       (addrb_hold), // INPUT
.CEA              (cea         ), // INPUT
.CEB              (ceb         ), // INPUT
.CLKA              (clka        ), // INPUT
.CLKB              (clkb        ), // INPUT
.ORCEA              (1'b0        ), // INPUT
.ORCEB              (1'b0        ), // INPUT
.RSTA             (rsta        ), // INPUT
.RSTB             (rstb        ), // INPUT
.WEA               (wea          ), // INPUT
.WEB               (web          )   // INPUT
);
GTP_DRM36K_E1 #(
.CSA_MASK          (3'b001),
.CSB_MASK          (3'b001),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.RAM_MODE          ("TRUE_DUAL_PORT")
) DRM_2_1 (
.DOA               ({doa_1[0]}), // OUTPUT[35:0]
.DOB               ({dob_1[0]}), // OUTPUT[35:0]
.ADDRA               ({1'b1,addra[14:0]}), // INPUT[15:0]
.ADDRB               ({1'b1,addrb[14:0]}), // INPUT[15:0]
.BWEA               (8'hff        ), // INPUT[7:0]
.BWEB               (4'hf          ), // INPUT[3:0]
.CSA              ({2'b0,addra[15]}), // INPUT[2:0]
.CSB              ({2'b0,addrb[15]}), // INPUT[2:0]
.DIA             ({dia[0]}    ), // INPUT[35:0]
.DIB             ({dib[0]}    ), // INPUT[35:0]
.ADDRA_HOLD       (addra_hold), // INPUT
.ADDRB_HOLD       (addrb_hold), // INPUT
.CEA              (cea         ), // INPUT
```

```
.CEB                (ceb          ), // INPUT
.CLKA                (clka         ), // INPUT
.CLKB                (clkb         ), // INPUT
.ORCEA               (1'b0         ), // INPUT
.ORCEB               (1'b0         ), // INPUT
.RSTA               (rsta         ), // INPUT
.RSTB               (rstb         ), // INPUT
.WEA                 (wea          ), // INPUT
.WEB                 (web          )    // INPUT
);
GTP_DRM36K_E1 #(
.CSA_MASK          (3'b001),
.CSB_MASK          (3'b001),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.RAM_MODE          ("TRUE_DUAL_PORT")
) DRM_2_2 (
.DOA                 ({doa_1[1]}), // OUTPUT[35:0]
.DOB                 ({dob_1[1]}), // OUTPUT[35:0]
.ADDRA                ({1'b1,addra[14:0]}), // INPUT[15:0]
.ADDRB                ({1'b1,addrb[14:0]}), // INPUT[15:0]
.BWEA                (8'hff        ), // INPUT[7:0]
.BWEB                (4'hf         ), // INPUT[3:0]
.CSA                ({2'b0,addra[15]}), // INPUT[2:0]
.CSB                ({2'b0,addrb[15]}), // INPUT[2:0]
.DIA                ({dia[1]}    ), // INPUT[35:0]
.DIB                ({dib[1]}    ), // INPUT[35:0]
.ADDRA_HOLD        (addra_hold), // INPUT
.ADDRB_HOLD        (addrb_hold), // INPUT
.CEA                (cea          ), // INPUT
.CEB                (ceb          ), // INPUT
.CLKA                (clka         ), // INPUT
```

```
.CLKB             (clkb         ), // INPUT
.ORCEA            (1'b0         ), // INPUT
.ORCEB            (1'b0         ), // INPUT
.RSTA             (rsta         ), // INPUT
.RSTB             (rstb         ), // INPUT
.WEA              (wea          ), // INPUT
.WEB              (web          )  // INPUT
);
//Output data selection
always @(posedge clka or posedge rsta)
   begin
      if (rsta)
         sel_a <= 1'b0;
      else if (~addra_hold & cea)
         sel_a <= addra[15];
      end
always @(posedge clkb or posedge rstb)
   begin
      if (rstb)
         sel_b <= 1'b0;
      else if (~addrb_hold & ceb)
         sel_b <= addrb[15];
      end
assign doa[1:0] = (sel_a)?doa_1[1:0]:doa_0[1:0];
assign dob[1:0] = (sel_b)?dob_1[1:0]:dob_0[1:0];
```

3.9.3 Hard Cascade Configuration for Multiple 36K DRMs

To hard cascade multiple 36K DRMs, hard cascade every two 36K DRMs to a 64Kx1 memory, with a total address width of 16 bits, then cascade address depth of 3 bits (CSA/CSB bit width is 3) to the maximum address width of 19 bits, without adding external logic for address processing.

This section illustrates the GTP hard cascade configuration steps for multiple 36K DRMs. Users can also directly generate the IP of the hard cascaded DRM by selecting the resource type as DRM64K using the IP Compiler tool embedded in the Pango Design Suite software. Refer to "DRM/FIFO IP User Guide UG041002" attached in the IP Compiler tool.

For hard cascade, configure the 36K DRMs to a fixed 32Kx1 mode. Hard cascade two 36K DRMs to a basic unit of the hard cascade—a 64Kx1 memory. Eight DP 36K DRMs are cascaded in the example, each with Ports A/B configured as 32Kx1, with a total data width of 2 bits and a total address width of 17 bits. 18K DRMs do not support hard cascade.

Name the 8 DRMs DRM_1_1_upper, DRM_1_1_lower, DRM_1_2_upper, DRM_1_2_lower, DRM_2_1_upper, DRM_2_1_lower, DRM_2_2_upper, and DRM_2_2_lower. Taking DRM_1_2_upper as an example, the first digit 1 represents the address cascading coordinate, with 2 levels of address cascading, extending the address from 16 bits of a single basic unit for hard cascade to 17 bits; the second digit 2 represents the data cascading coordinate, with 2 levels of data cascading, extending the data from 1 bit of a single DRM to 2 bits; the word "upper" represents the upper 36K DRM in the hard cascade.

To hard cascade multiple 36K DRMs, configure them as follows:

1. Single DRM configuration: Configure the eight 36K DRMs as 32Kx1 DP 36K DRMs, following similar steps in 3.9.1 Single 36K DRM Configuration;
2. Configure the DRM address extension control parameters as per Table 3-9:

Table 3-9 Hard Cascading Parameter Configuration of DP Mode for a Single 36K DRM

Parameter Name	**DRM Name**	**Configuration Value**
CSA_MARK	DRM_1_1_upper	3'b000
	DRM_1_1_lower	3'b000
	DRM_1_2_upper	3'b000
	DRM_1_2_lower	3'b000
	DRM_2_1_upper	3'b001
	DRM_2_1_lower	3'b001
	DRM_2_2_upper	3'b001
	DRM_2_2_lower	3'b001
CSB_MARK	DRM_1_1_upper	3'b000
	DRM_1_1_lower	3'b000
	DRM_1_2_upper	3'b000
	DRM_1_2_lower	3'b000
	DRM_2_1_upper	3'b001
	DRM_2_1_lower	3'b001
	DRM_2_2_upper	3'b001
	DRM_2_2_lower	3'b001

3. Connect the cascade-related ports of the DRM as per Table 3-10:

Table 3-10 Hard Cascading Port Connections of DP Mode for a Single 36K DRM

Port	**DRM Name**	**Interfacing Signals**

Port	DRM Name	Interfacing Signals
CSA	DRM_1_1_upper	{2'b0,addra[16]}
	DRM_1_1_lower	
	DRM_1_2_upper	
	DRM_1_2_lower	
	DRM_2_1_upper	
	DRM_2_1_lower	
	DRM_2_2_upper	
	DRM_2_2_lower	
CSB	DRM_1_1_upper	{2'b0,addrb[16]}
	DRM_1_1_lower	
	DRM_1_2_upper	
	DRM_1_2_lower	
	DRM_2_1_upper	
	DRM_2_1_lower	
	DRM_2_2_upper	
	DRM_2_2_lower	
ADDRA[15:0]	DRM_1_1_upper	{addra[15:0]}
	DRM_1_1_lower	
	DRM_1_2_upper	
	DRM_1_2_lower	
	DRM_2_1_upper	
	DRM_2_1_lower	
	DRM_2_2_upper	
	DRM_2_2_lower	
ADDRB[15:0]	DRM_1_1_upper	{addrb[15:0]}
	DRM_1_1_lower	
	DRM_1_2_upper	
	DRM_1_2_lower	
	DRM_2_1_upper	
	DRM_2_1_lower	
	DRM_2_2_upper	
	DRM_2_2_lower	
DIA[35:0]	DRM_1_1_upper	dia[0]
	DRM_1_1_lower	dia[0]
	DRM_1_2_upper	dia[1]
	DRM_1_2_lower	dia[1]
	DRM_2_1_upper	dia[0]
	DRM_2_1_lower	dia[0]
	DRM_2_2_upper	dia[1]

Port	DRM Name	Interfacing Signals
	DRM_2_2_lower	dia[1]
DIB[35:0]	DRM_1_1_upper	dib[0]
	DRM_1_1_lower	dib[0]
	DRM_1_2_upper	dib[1]
	DRM_1_2_lower	dib[1]
	DRM_2_1_upper	dib[0]
	DRM_2_1_lower	dib[0]
	DRM_2_2_upper	dib[1]
	DRM_2_2_lower	dib[1]
DOA[35:0]	DRM_1_1_upper	doa_0[0]
	DRM_1_2_upper	doa_0[1]
	DRM_2_1_upper	doa_1[0]
	DRM_2_2_upper	doa_1[1]
DOB[35:0]	DRM_1_1_upper	dob_0[0]
	DRM_1_2_upper	dob_0[1]
	DRM_2_1_upper	dob_1[0]
	DRM_2_2_upper	dob_1[1]

4. Output data selection: When addra[16] is 0 after a delay of one beat through register, the data output from Port A is doa_0[1:0], and when it is 1, the data output from Port A is doa_0[1:0]. Port B operates similarly;
5. Configure the hard cascade parameters and connect hard cascade ports for the two DRMs corresponding to each basic unit of the hard cascade as upper and lower as per 3.8 Hard Cascade Function.

The configured GTP is as follows:

```
GTP_DRM36K_E1 #(
.CSA_MASK          (3'b000),
.CSB_MASK          (3'b000),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.RAM_MODE          ("TRUE_DUAL_PORT"),
.RAM_CASCADE      ("LOWER")
) DRM_0_0_lower (
.DOA                  (                 ), // OUTPUT[35:0]
```

```
.DOB              (              ), // OUTPUT[35:0]
.ADDRA             ({addra[15:0]}), // INPUT[15:0]
.ADDRB             ({addrb[15:0]}), // INPUT[15:0]
.BWEA              (8'hff      ), // INPUT[7:0]
.BWEB              (4'hf       ), // INPUT[3:0]
.CSA              ({2'b0,addra[16]}), // INPUT[2:0]
.CSB              ({2'b0,addrb[16]}), // INPUT[2:0]
.DIA              ({dia[0]}    ), // INPUT[35:0]
.DIB              ({dib[0]}    ), // INPUT[35:0]
.ADDRA_HOLD       (1'b0        ), // INPUT
.ADDRB_HOLD       (1'b0        ), // INPUT
.CEA              (1'b1        ), // INPUT
.CEB              (1'b1        ), // INPUT
.CLKA             (clka        ), // INPUT
.CLKB             (clkb        ), // INPUT
.ORCEA            (1'b0        ), // INPUT
.ORCEB            (1'b0        ), // INPUT
.RSTA             (rsta        ), // INPUT
.RSTB             (rstb        ), // INPUT
.WEA              (wea         ), // INPUT
.WEB              (web         ), // INPUT
.CINA             (            ), // INPUT
.CINB             (            ), // INPUT
.COUTA            (couta[0]    ), // OUTPUT
.COUTB            (coutb[0]    )    // OUTPUT
);
GTP_DRM36K_E1 #(
.CSA_MASK          (3'b000),
.CSB_MASK          (3'b000),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.RAM_MODE          ("TRUE_DUAL_PORT"),
.RAM_CASCADE     ("UPPER")
) DRM_0_0_upper (
.DOA                 ({doa_0[0]}), // OUTPUT[35:0]
.DOB                 ({dob_0[0]}), // OUTPUT[35:0]
.ADDRA                ({addra[15:0]}), // INPUT[15:0]
.ADDRB                ({addrb[15:0]}), // INPUT[15:0]
.BWEA                 (8'hff        ), // INPUT[7:0]
.BWEB                 (4'hf         ), // INPUT[3:0]
.CSA                 ({2'b0,addra[16]}), // INPUT[2:0]
.CSB                 ({2'b0,addrb[16]}), // INPUT[2:0]
.DIA                ({dia[0]}    ), // INPUT[35:0]
.DIB                ({dib[0]}    ), // INPUT[35:0]
.ADDRA_HOLD       (1'b0          ), // INPUT
.ADDRB_HOLD       (1'b0          ), // INPUT
.CEA                 (1'b1          ), // INPUT
.CEB                 (1'b1          ), // INPUT
.CLKA                (clka          ), // INPUT
.CLKB                (clkb          ), // INPUT
.ORCEA               (1'b0          ), // INPUT
.ORCEB               (1'b0          ), // INPUT
.RSTA                (rsta          ), // INPUT
.RSTB                (rstb          ), // INPUT
.WEA                 (wea            ), // INPUT
.WEB                 (web            ), // INPUT
.CINA                (couta[0]   ), // INPUT
.CINB                (coutb[0]   ), // INPUT
.COUTA               (               ), // OUTPUT
.COUTB               (               )    // OUTPUT
);
GTP_DRM36K_E1 #(
.CSA_MASK          (3'b000),
.CSB_MASK          (3'b000),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A   ("TRANSPARENT_WRITE"),
.WRITE_MODE_B   ("TRANSPARENT_WRITE"),
.RAM_MODE           ("TRUE_DUAL_PORT"),
.RAM_CASCADE      ("LOWER")
) DRM_0_1_lower (
.DOA                   (                    ), // OUTPUT[35:0]
.DOB                   (                    ), // OUTPUT[35:0]
.ADDRA                 ({addra[15:0]}), // INPUT[15:0]
.ADDRB                 ({addrb[15:0]}), // INPUT[15:0]
.BWEA                  (8'hff          ), // INPUT[7:0]
.BWEB                  (4'hf            ), // INPUT[3:0]
.CSA                   ({2'b0,addra[16]}), // INPUT[2:0]
.CSB                   ({2'b0,addrb[16]}), // INPUT[2:0]
.DIA                   ({dia[1]}    ), // INPUT[35:0]
.DIB                   ({dib[1]}    ), // INPUT[35:0]
.ADDRA_HOLD        (1'b0           ), // INPUT
.ADDRB_HOLD        (1'b0           ), // INPUT
.CEA                   (1'b1           ), // INPUT
.CEB                   (1'b1           ), // INPUT
.CLKA                  (clka           ), // INPUT
.CLKB                  (clkb           ), // INPUT
.ORCEA                 (1'b0           ), // INPUT
.ORCEB                 (1'b0           ), // INPUT
.RSTA                  (rsta           ), // INPUT
.RSTB                  (rstb           ), // INPUT
.WEA                   (wea             ), // INPUT
.WEB                   (web             ), // INPUT
.CINA                  (                  ), // INPUT
.CINB                  (                  ), // INPUT
.COUTA                 (couta[1]   ), // OUTPUT
.COUTB                 (coutb[1]   )    // OUTPUT
);
GTP_DRM36K_E1 #(
.CSA_MASK           (3'b000),
.CSB_MASK          (3'b000),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.RAM_MODE          ("TRUE_DUAL_PORT"),
.RAM_CASCADE      ("UPPER")
) DRM_0_1_upper (
.DOA                    ({doa_0[1]}), // OUTPUT[35:0]
.DOB                    ({dob_0[1]}), // OUTPUT[35:0]
.ADDRA                  ({addra[15:0]}), // INPUT[15:0]
.ADDRB                  ({addrb[15:0]}), // INPUT[15:0]
.BWEA                   (8'hff          ), // INPUT[7:0]
.BWEB                   (4'hf            ), // INPUT[3:0]
.CSA                    ({2'b0,addra[16]}), // INPUT[2:0]
.CSB                    ({2'b0,addrb[16]}), // INPUT[2:0]
.DIA                   ({dia[1]}    ), // INPUT[35:0]
.DIB                   ({dib[1]}    ), // INPUT[35:0]
.ADDRA_HOLD       (1'b0           ), // INPUT
.ADDRB_HOLD       (1'b0           ), // INPUT
.CEA                    (1'b1           ), // INPUT
.CEB                    (1'b1           ), // INPUT
.CLKA                   (clka           ), // INPUT
.CLKB                   (clkb           ), // INPUT
.ORCEA                  (1'b0           ), // INPUT
.ORCEB                  (1'b0           ), // INPUT
.RSTA                   (rsta           ), // INPUT
.RSTB                   (rstb           ), // INPUT
.WEA                    (wea             ), // INPUT
.WEB                    (web             ), // INPUT
.CINA                   (couta[1]    ), // INPUT
.CINB                   (coutb[1]    ), // INPUT
.COUTA                  (                    ), // OUTPUT
.COUTB                  (                    )    // OUTPUT
```

```
);
GTP_DRM36K_E1 #(
.CSA_MASK          (3'b001),
.CSB_MASK          (3'b001),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.RAM_MODE          ("TRUE_DUAL_PORT"),
.RAM_CASCADE      ("LOWER")
) DRM_1_0_lower (
.DOA                  (                ), // OUTPUT[35:0]
.DOB                  (                ), // OUTPUT[35:0]
.ADDRA               ({addra[15:0]}), // INPUT[15:0]
.ADDRB               ({addrb[15:0]}), // INPUT[15:0]
.BWEA                 (8'hff        ), // INPUT[7:0]
.BWEB                 (4'hf          ), // INPUT[3:0]
.CSA                  ({2'b0,addra[16]}), // INPUT[2:0]
.CSB                  ({2'b0,addrb[16]}), // INPUT[2:0]
.DIA                  ({dia[0]}    ), // INPUT[35:0]
.DIB                  ({dib[0]}    ), // INPUT[35:0]
.ADDRA_HOLD       (1'b0          ), // INPUT
.ADDRB_HOLD       (1'b0          ), // INPUT
.CEA                  (1'b1          ), // INPUT
.CEB                  (1'b1          ), // INPUT
.CLKA                 (clka          ), // INPUT
.CLKB                 (clkb          ), // INPUT
.ORCEA               (1'b0          ), // INPUT
.ORCEB               (1'b0          ), // INPUT
.RSTA                 (rsta          ), // INPUT
.RSTB                 (rstb          ), // INPUT
.WEA                  (wea            ), // INPUT
.WEB                  (web            ), // INPUT
.CINA                 (                ), // INPUT
```

```
.CINB              (               ), // INPUT
.COUTA              (couta[2]    ), // OUTPUT
.COUTB              (coutb[2]    )    // OUTPUT
);
GTP_DRM36K_E1 #(
.CSA_MASK           (3'b001),
.CSB_MASK           (3'b001),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.RAM_MODE           ("TRUE_DUAL_PORT"),
.RAM_CASCADE      ("UPPER")
) DRM_1_0_upper (
.DOA                  ({doa_1[0]}), // OUTPUT[35:0]
.DOB                  ({dob_1[0]}), // OUTPUT[35:0]
.ADDRA                ({addra[15:0]}), // INPUT[15:0]
.ADDRB                ({addrb[15:0]}), // INPUT[15:0]
.BWEA                 (8'hff         ), // INPUT[7:0]
.BWEB                 (4'hf           ), // INPUT[3:0]
.CSA                  ({2'b0,addra[16]}), // INPUT[2:0]
.CSB                  ({2'b0,addrb[16]}), // INPUT[2:0]
.DIA                  ({dia[0]}    ), // INPUT[35:0]
.DIB                  ({dib[0]}    ), // INPUT[35:0]
.ADDRA_HOLD        (1'b0           ), // INPUT
.ADDRB_HOLD        (1'b0           ), // INPUT
.CEA                  (1'b1           ), // INPUT
.CEB                  (1'b1           ), // INPUT
.CLKA                 (clka           ), // INPUT
.CLKB                 (clkb           ), // INPUT
.ORCEA                (1'b0           ), // INPUT
.ORCEB                (1'b0           ), // INPUT
.RSTA                 (rsta           ), // INPUT
.RSTB                 (rstb           ), // INPUT
```

```
.WEA              (wea          ), // INPUT
.WEB              (web          ), // INPUT
.CINA             (couta[2]   ), // INPUT
.CINB             (coutb[2]   ), // INPUT
.COUTA            (             ), // OUTPUT
.COUTB            (             )   // OUTPUT
);
GTP_DRM36K_E1 #(
.CSA_MASK          (3'b001),
.CSB_MASK          (3'b001),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.RAM_MODE          ("TRUE_DUAL_PORT"),
.RAM_CASCADE      ("LOWER")
) DRM_1_1_lower (
.DOA              (             ), // OUTPUT[35:0]
.DOB              (             ), // OUTPUT[35:0]
.ADDRA            ({addra[15:0]}), // INPUT[15:0]
.ADDRB            ({addrb[15:0]}), // INPUT[15:0]
.BWEA             (8'hff       ), // INPUT[7:0]
.BWEB             (4'hf         ), // INPUT[3:0]
.CSA              ({2'b0,addra[16]}), // INPUT[2:0]
.CSB              ({2'b0,addrb[16]}), // INPUT[2:0]
.DIA              ({dia[1]}), // INPUT[35:0]
.DIB              ({dib[1]}), // INPUT[35:0]
.ADDRA_HOLD       (1'b0          ), // INPUT
.ADDRB_HOLD       (1'b0          ), // INPUT
.CEA              (1'b1          ), // INPUT
.CEB              (1'b1          ), // INPUT
.CLKA             (clka          ), // INPUT
.CLKB             (clkb          ), // INPUT
.ORCEA            (1'b0          ), // INPUT
```

```
.ORCEB              (1'b0          ), // INPUT
.RSTA              (rsta          ), // INPUT
.RSTB              (rstb          ), // INPUT
.WEA               (wea           ), // INPUT
.WEB               (web           ), // INPUT
.CINA              (              ), // INPUT
.CINB              (              ), // INPUT
.COUTA              (couta[3]   ), // OUTPUT
.COUTB              (coutb[3]   )    // OUTPUT
);
GTP_DRM36K_E1 #(
.CSA_MASK          (3'b001),
.CSB_MASK          (3'b001),
.DATA_WIDTH_A    (1),
.DATA_WIDTH_B    (1),
.WRITE_MODE_A    ("TRANSPARENT_WRITE"),
.WRITE_MODE_B    ("TRANSPARENT_WRITE"),
.RAM_MODE          ("TRUE_DUAL_PORT"),
.RAM_CASCADE      ("UPPER")
) DRM_1_1_upper (
.DOA                ({doa_1[1]}), // OUTPUT[35:0]
.DOB                ({dob_1[1]}), // OUTPUT[35:0]
.ADDRA               ({addra[15:0]}), // INPUT[15:0]
.ADDRB               ({addrb[15:0]}), // INPUT[15:0]
.BWEA               (8'hff         ), // INPUT[7:0]
.BWEB               (4'hf          ), // INPUT[3:0]
.CSA                ({2'b0,addra[16]}), // INPUT[2:0]
.CSB                ({2'b0,addrb[16]}), // INPUT[2:0]
.DIA                ({dia[1]}), // INPUT[35:0]
.DIB                ({dib[1]}), // INPUT[35:0]
.ADDRA_HOLD       (1'b0          ), // INPUT
.ADDRB_HOLD       (1'b0          ), // INPUT
.CEA               (1'b1          ), // INPUT
.CEB               (1'b1          ), // INPUT
```

```
.CLKA               (clka         ), // INPUT
.CLKB               (clkb         ), // INPUT
.ORCEA              (1'b0         ), // INPUT
.ORCEB              (1'b0         ), // INPUT
.RSTA               (rsta         ), // INPUT
.RSTB               (rstb         ), // INPUT
.WEA                (wea          ), // INPUT
.WEB                (web          ), // INPUT
.CINA               (couta[3]     ), // INPUT
.CINB               (coutb[3]     ), // INPUT
.COUTA              (             ), // OUTPUT
.COUTB              (             )   // OUTPUT
);
//Output data selection
always @(posedge clka or posedge rsta)
   begin
      if (rsta)
         sel_a <= 1'b0;
      else
         sel_a <= addra[16];
      end
always @(posedge clkb or posedge rstb)
   begin
      if (rstb)
         sel_b <= 1'b0;
      else
         sel_b <= addrb[16];
      end
assign doa[1:0] = (sel_a)?doa_1[1:0]:doa_0[1:0];
assign dob[1:0] = (sel_b)?dob_1[1:0]:dob_0[1:0];
```

Chapter 4 SDP Mode

4.1 Introduction

The port mode of the RAM is determined by the parameter RAM_MODE; when the value of RAM_MODE is "SIMPLE_DUAL_PORT", the RAM enters SDP mode. This document will detail the 36K SDP RAM, which has essentially the same structure and functions as the 18K, with the only difference being the bit width.

SDP RAM mode supports:

- Dual-port read and write, with one port as the read port while the other as the write port.
- Neither port can perform read and write operations simultaneously.
- Independent bit width settings for two ports
- Under 18K mode with x32/x36 bit width and 36K mode with x64/x72 bit width, Port A is fixed as the write control port (active when WEA is high), and Port B is fixed as the read control port (active when WEB is low).

4.2 Ports Descriptions

Figure 4-1 shows the structure of the 36K SDP RAM.

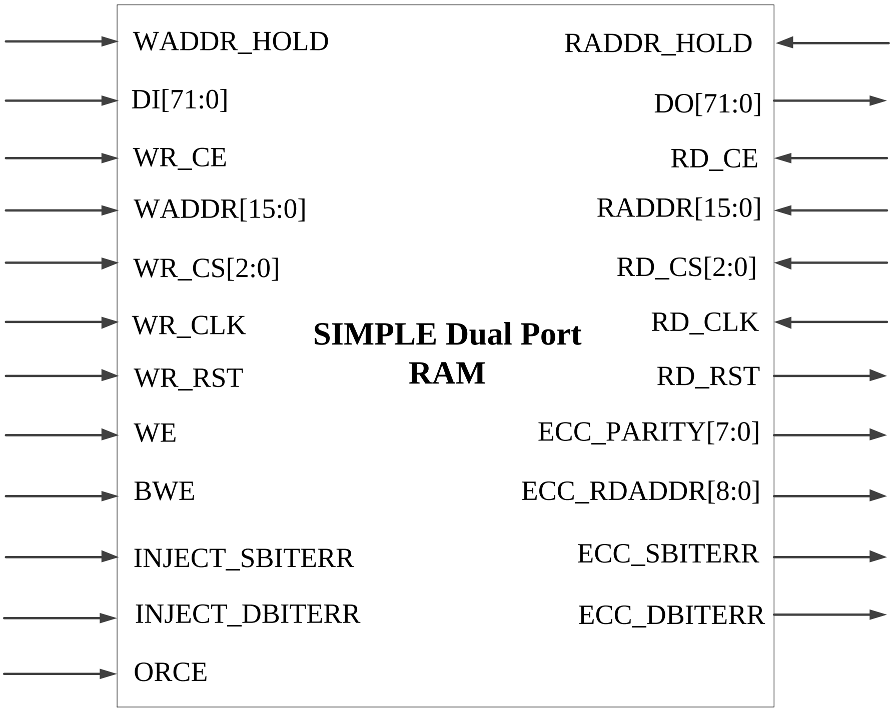

Figure 4-1 SDP RAM Data Ports

Each 18K and 36K DRM can also be configured as SDP (Simple Dual Port) RAM. In this mode, the DRM port data width increases to 72 bits. The SDP RAM mode includes two pairs of ports (each pair has two different ports, A and B), with one of the ports for data writing and the other for data reading, and both reading and writing ports support different clocks. Similar to DP RAM, performing a read and a write operation on the same address through both ports simultaneously will cause a conflict.

Table 4-1 lists the port names and descriptions for SDP RAM mode.

Table 4-1 Port Names and Descriptions of SDP RAM Mode

Port	Direction	Description	Port	Direction	Description
WADDR_HOLD	Input	Write port address hold signal	RADDR_HOLD	Input	Read port address hold signal
DI	Input	Write port data input	DO	Output	Read port data output
WR_CE	Input	Input register clock enable	RD_CE	Input	Input register clock enable
WADDR	Input	Write port address input	RADDR	Input	Read port address input
WR_CS	Input	Write port address extension	RD_CS	Input	Read port address extension
WR_CLK	Input	Write port clock	RD_CLK	Input	Read port clock
WR_RST	Input	Write port output register reset	RD_RSTB	Input	Read port output register reset
WE	Input	Write enable	ECC_PARITY[7:0]	Output	ECC mode checksum bit output
BWE	Input	Byte enable signal	ECC_RDADDR[8:0]	Output	ECC mode read address output
INJECT_SBITERR	Input	ECC mode single-bit error injection	ECC_SBITERR	Output	ECC mode single-bit error flag
INJECT_DBITERR	Input	ECC mode dual-bit error injection	ECC_DBITERR	Output	ECC mode dual-bit error flag
ORCE	Input	Output register clock enable			

Note:

1. Under SDP mode with x64/x72 bit width, DI is formed by concatenating DIA/DIB from Ports A/B, and DO likewise. Refer to 9. 1 Address and Data Port Mapping.
2. The 18K DRM has no CSA/CSB port and does not support address extension. To extend address depth through DRM cascading, it is recommended to use 36K DRM.

4.3 Bit Width Combinations

The port bit width of RAM is determined by the parameter DATA_WIDTH_A/DATA_WIDTH_B in the GTP. For example, when the value of DATA_WIDTH_A is 4, the data width of Port A is set to 4 bits. SDP mode supports setting different data widths for Ports A and B.

Table 4-2 shows the allowed bit width combinations for Simple Dual Port RAM mode in 36K DRM mode.

Table 4-2 Allowed Bit Width Combinations for Simple Dual Port RAM Mode in 36K DRM Mode

		Write Ports										
		32Kx1	**16Kx2**	**8Kx4**	**4Kx8**	**2Kx16**	**1Kx32**	**512x64**	**4Kx9**	**2Kx18**	**1Kx36**	**512x72**
Read Ports	**32Kx1**	√	√	√	√	√	√	√				
	16Kx2	√	√	√	√	√	√	√				
	8Kx4	√	√	√	√	√	√	√				
	4Kx8	√	√	√	√	√	√	√				
	2Kx16	√	√	√	√	√	√	√				
	1Kx32	√	√	√	√	√	√	√				
	512x64	√	√	√	√	√	√	√				
	4Kx9								√	√	√	√
	2Kx18								√	√	√	√
	1Kx36								√	√	√	√
	512x72								√	√	√	√

Note: √ indicates the supported bit width combinations.

Table 4-3 shows the allowed bit width combinations for Simple Dual Port RAM mode in 18K DRM mode.

Table 4-3 Allowed Bit Width Combinations for Simple Dual Port RAM Mode in 18K DRM Mode

		Write Ports								
		16Kx1	**8Kx2**	**4Kx4**	**2Kx8**	**1Kx16**	**512x32**	**2Kx9**	**1Kx18**	**512x36**
Read Port 0/1	**16Kx1**	√	√	√	√	√	√			
	8Kx2	√	√	√	√	√	√			
	4Kx4	√	√	√	√	√	√			
	2Kx8	√	√	√	√	√	√			
	1Kx16	√	√	√	√	√	√			
	512x32	√	√	√	√	√	√			
	2Kx9							√	√	√
	1Kx18							√	√	√
	512x36							√	√	√

Note: √ indicates the supported bit width combinations.

4.4 Timing Parameter

The timing parameters of DRM in SDP mode are the same as those in DP mode. Refer to 3.4 Timing Parameter.

4.5 Read and Write Operations

SDP mode does not support write mode settings. It is forbidden to set it to TW or RBW mode during use, and it must be set to the default NW mode. That is, the values of WRITE_MODE_A/WRITE_
MODE_B in GTP must be the default "NORMAL_WRITE", otherwise, the DRM will enter an abnormal write operation state.

In SDP mode, there are two relatively independent ports. Performing a read and a write operation on the same address through both ports simultaneously will cause a conflict. DRM prohibits both ports from writing data to the same address simultaneously and from performing a read and a write on the same address at the same time; this must be avoided in actual applications through user logic. As shown in Figure 4-2 Read-Write Timing Diagram for SDP Mode, the user writes data from Port A of the DRM and reads data from Port B of the DRM. Mem in the diagram refers to the old data stored at the corresponding address.

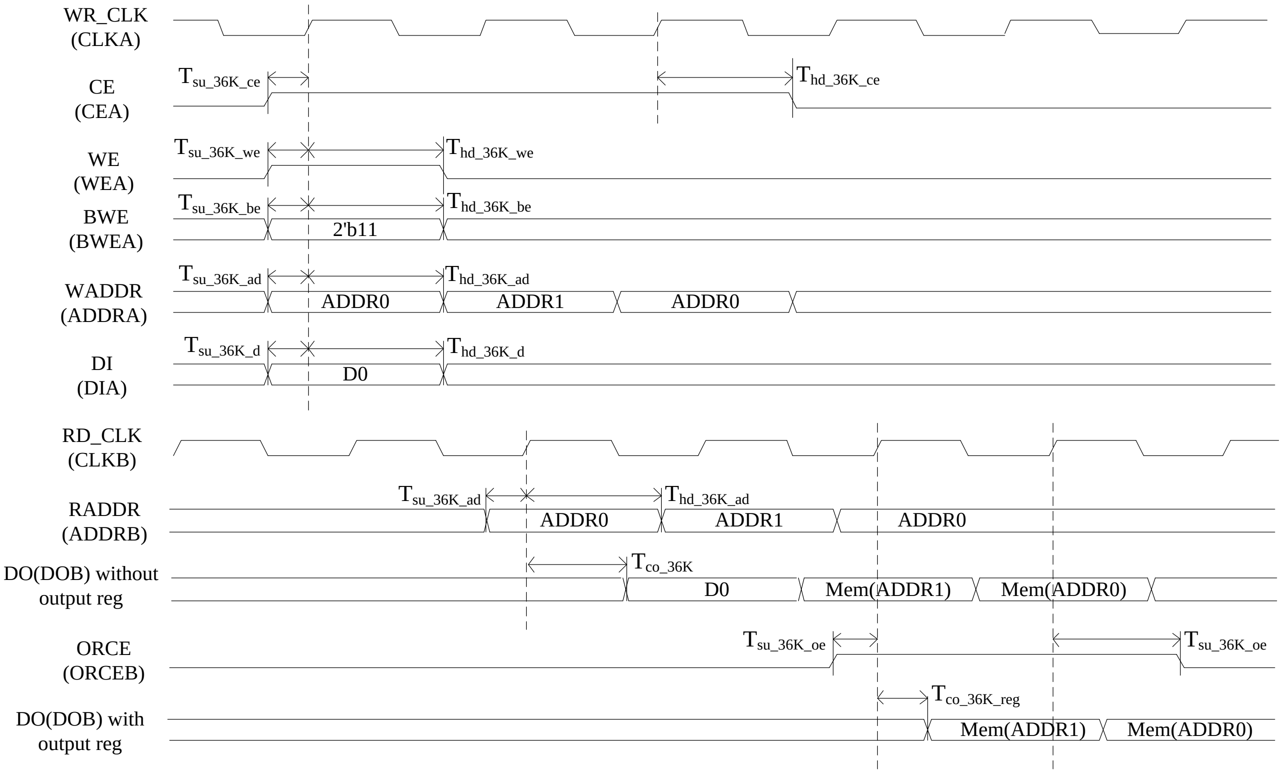

Figure 4-2 Read-Write Timing Diagram for SDP Mode

4.6 Byte Enable

The byte enable function of SDP mode is similar to that of DP mode. Refer to 3.6 Byte Enable. The main differences in byte enable between SDP mode and DP mode are as follows:

- In SDP mode, one port is fixed as a write port, and the other port is fixed as a read port, connecting the byte enable signal to a low level;
- The byte enable signal in SDP mode also supports a data width of x72(64) bits, where Port A is fixed as a write port, and the byte enable signal WEA[7:4] is active; for other bit widths, when Port A is a write port, WEA[7:4] is connected to a high level, and when Port A is a read port, WEA[7:4] is connected to a low level.

Figure 4-3 shows the read/write timing diagram for the DRM in byte enable mode under SDP mode, where Mem refers to the old data stored at the corresponding address.

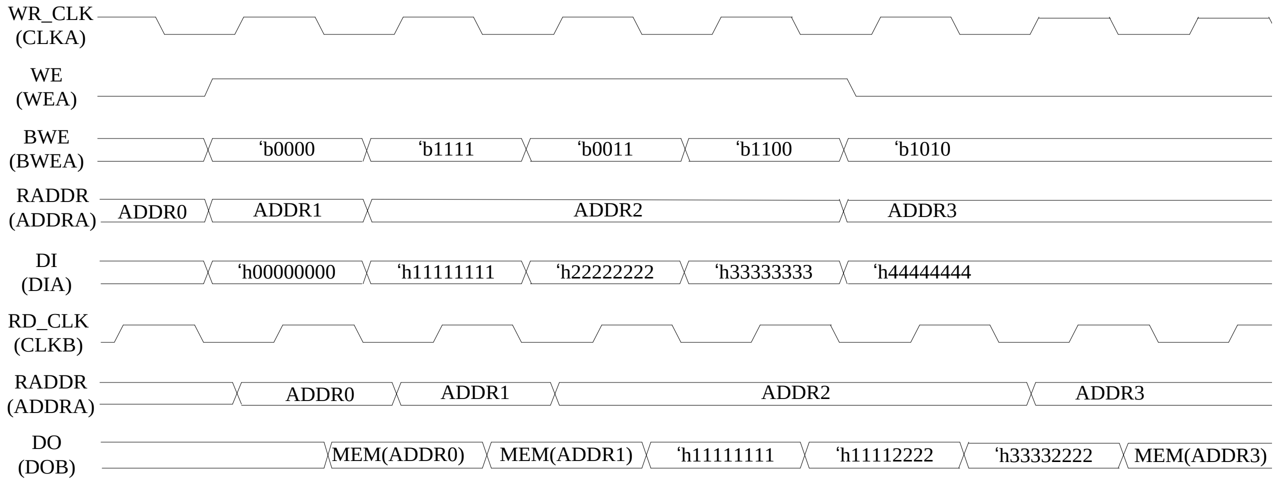

Figure 4-3 Byte Enable Read/Write Timing Diagram (SDP Mode)

4.7 Internal Registers

The internal registers of SDP mode are the same as those of DP mode. Refer to 3.7 Internal Registers.

4.8 Hard Cascade Function

The hard cascade function of SDP mode is the same as that of DP mode. Refer to 3.8 Hard Cascade Function.

4.9 Application Example

4.9.1 Single 36K DRM Configuration

This section illustrates the GTP configuration steps for a single 36K DRM. Users can also directly generate the DRM IP using the IP Compiler tool embedded in the Pango Design Suite software. Refer to "DRM/FIFO IP User Guide UG041002" attached in the IP Compiler tool.

The example is for SDP mode with mixed bit widths, with read Port A configured as 4Kx9 and write Port B configured as 512x72 (by concatenating DOA/DOB of Port A/B). Both ports have different clocks, enabling output registers. The configuration steps for an 18K DRM are similar.

Configure a single 36K DRM as follows:

1. Configure the DRM parameters as per Table 4-4:

Table 4-4 Parameter Configuration of SDP Mode for a Single 36K DRM

Parameter Name	Configuration Value	Description
DATA_WIDTH_A	9	Configure Port A to 4Kx9 mode
DATA_WIDTH_B	72	Configure Port B to 512x72 mode
WRITE_MODE_A	"NORMAL_WRITE"	Configure the write mode of Port A to NW
WRITE_MODE_B		Configure the write mode of Port B to NW
DOA_REG	1	Enable Port A output register
DOB_REG		Enable Port B output register
RAM_MODE	"SIMPLE_DUAL_PORT"	Configure the DRM to SDP mode

2. Connect the DRM ports as per Table 4-5:

Table 4-5 Port Connections of SDP Mode for a Single 36K DRM

Port	Interfacing Signals	Description
ADDRA[15:0]	{1'b1,waddr[11:0],3'b0}	Port A serves as the write port. Connect the write address signal waddr[11:0] to ADDRA[14:3], ADDRA[15] to a high level, and ADDRA[2:0] to a low level. Refer to 9.1 Address and Data Port Mapping for detailed address connection instructions.
ADDRB[15:0]	{1'b1,raddr[8:0],6'b0}	Port B serves as the read port. Connect the read address signal raddr[8:0] to ADDRB[14:6], ADDRB[15] to a high level, and ADDRB[5:0] to a low level.
ADDRA_HOLD	1'b0	Do not use Port A/B address hold function, connected to a low level
ADDRB_HOLD		
DIA[35:0]	di[8:0]	Connect the write data signal di[8:0] to DIA[8:0], leaving the unused high-bit DIA ports floating. Refer to 9.1 Address and Data Port Mapping for detailed data port connection instructions.
DIB[35:0]	Float or connect to a low level	Port B is a read port without data concatenation for writing, and the GTP port DIB is either unused or connected to a low level.

Port	Interfacing Signals	Description
CSA[2:0]	3'b0	Address extension function is not used, connected to a low level
CSB[2:0]		
BWEA[7:0]	8'hff	Port A is a write port, without using byte enable function. Connect BWEA to a high level.
BWEB[3:0]	4'b0	Port B is a read port, with write operation prohibited. Connect BWEB to a low level.
DOA[35:0]	do[35:0]	Concatenate Ports A/B to achieve a 72-bit data width, with DOA outputting the lower 36 bits of read data.
DOB[35:0]	do[71:36]	Concatenate Ports A/B to achieve a 72-bit data width, with DOB outputting the upper 36 bits of read data.
WEA	we	Connect the write enable signal we to the WEA of write port A
WEB	1'b0	Connect the write enable WEB of read port B to a low level

3. Other signals for read/write ports: Connect the clock, clock enable, output register clock enable, and data register reset of the read/write ports to the corresponding Ports A/B of GTP respectively;
4. Other unused parameters: Do not set other unused parameters, use default values;
5. Other unused ports: Leave other unused ports floating (ports that must be connected even if not used have been described above).

The configured GTP is as follows:

```
GTP_DRM36K_E1 #(
.DATA_WIDTH_A    (9),
.DATA_WIDTH_B    (72),
.WRITE_MODE_A    ("NORMAL_WRITE"),
.WRITE_MODE_B    ("NORMAL_WRITE"),
.DOA_REG            (1),
.DOB_REG            (1),
.RAM_MODE           ("SIMPLE_DUAL_PORT")
) GTP_DRM36K_E1_inst (
.DOA                    (do[35:0]    ), // OUTPUT[35:0]
.DOB                    (do[71:36] ), // OUTPUT[35:0]
.ADDRA                  ({1'b1,waddr[11:0],3'b0}), // INPUT[15:0]
.ADDRB                  ({1'b1,raddr[8:0],6'b0}), // INPUT[15:0]
.BWEA                   (8'hff          ), // INPUT[7:0]
.BWEB                   (4'b0            ), // INPUT[3:0]
.CSA                   (3'b0            ), // INPUT[2:0]
.CSB                   (3'b0            ), // INPUT[2:0]
.DIA          (di[8:0]    ), // INPUT[35:0]
.DIB          (           ), // INPUT[35:0]
.ADDRA_HOLD   (1'b0       ), // INPUT
.ADDRB_HOLD   (1'b0       ), // INPUT
.CEA          (wr_ce      ), // INPUT
.CEB          (rd_ce      ), // INPUT
.CLKA         (wr_clk     ), // INPUT
.CLKB         (rd_clk     ), // INPUT
.ORCEA        (orce       ), // INPUT
.ORCEB        (orce       ), // INPUT
.RSTA         (wr_rst     ), // INPUT
.RSTB         (rd_rst     ), // INPUT
.WEA          (we         ), // INPUT
.WEB          (1'b0       )  // INPUT
);
```

4.9.2 Configuration for Multiple 36K DRMs Cascaded

Cascade of multiple 36K DRMs in SDP mode is the same as that in DP mode. Refer to 3.9.2 Configuration for Multiple 36K DRMs Cascaded.

4.9.3 Hard Cascade Configuration for Multiple 36K DRMs

Hard cascade of multiple 36K DRMs in SDP mode is the same as that in DP mode. Refer to 3.9.3 Hard Cascade Configuration for Multiple 36K DRMs.

Chapter 5 SP Mode

5.1 Introduction

The port mode of the RAM is determined by the parameter RAM_MODE; when the value of RAM_MODE is "SINGLE_PORT", the RAM enters SP mode. This document will detail the 36K SP RAM, which has essentially the same structure and functions as the 18K, with the only difference being the bit width.

SP RAM mode supports:

- Single port read operation
- Single port write operation
- Under 18K mode with x32/x36 bit width and 36K mode with x64/x72 bit width, Port A is fixed as the write control port (active when WEA is high), and Port B is fixed as the read control port (active when WEB is low).

5.2 Ports Descriptions

Each 18K and 36K DRM can also be configured as SP (Single Port) RAM. In this mode, the data width of the DRM port increases to 72 bits. In SP RAM mode, when two ports are shared, the DRM includes only one port, and read and write operations are allowed on this port. Figure 5-1 shows the structure of the 36K SP RAM.

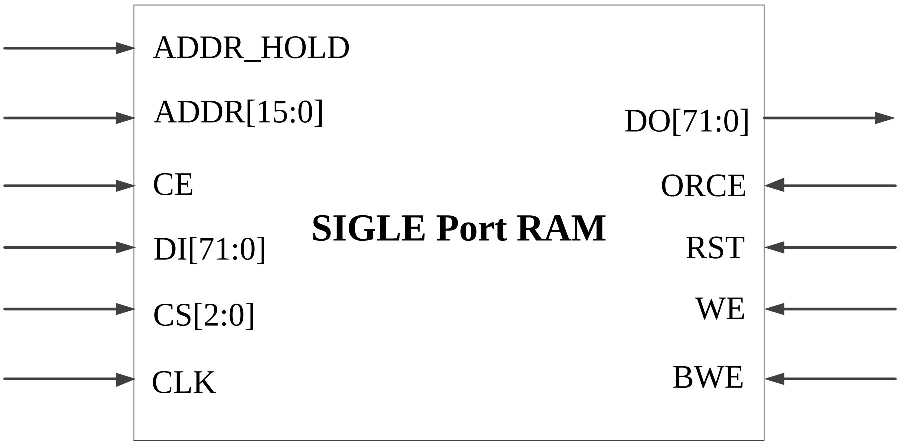

Figure 5-1 SP RAM Data Ports

Table 5-1 lists the port names and descriptions for SP RAM mode.

Table 5-1 Port Names and Descriptions of SP RAM Mode

Port	Direction	Description	Port	Direction	Description

Port	Direction	Description	Port	Direction	Description
ADDR_HOLD	Input	Address hold signal	DO	Output	Read data output
ADDR	Input	Address Inputs	ORCE	Input	Output register clock enable
CE	Input	Input register clock enable	RST	Input	Output register reset
DI	Input	Write data input	WE	Input	Write enable
CS	Input	Address extension	BWE	Input	Byte enable signal
CLK	Input	Port clock			

Note:

1. Under SP mode with x64/x72 bit width, DI is formed by concatenating DIA/DIB from Ports A/B, and DO likewise. For more details, refer to 9. 1 Address and Data Port Mapping.
2. The 18K DRM has no CSA/CSB port and does not support address extension. To extend address depth through DRM cascading, it is recommended to use 36K DRM.

5.3 Bit Width Combinations

The port bit width of RAM is determined by the parameter DATA_WIDTH_A/DATA_WIDTH_B in the GTP. For example, when the value of DATA_WIDTH_A is 4, the data width of Port A is set to 4 bits. Ports A and B in SP mode must be set to the same data width.

Table 5-2 shows the allowed bit widths for Single Port RAM mode in 36K DRM mode.

Table 5-2 Allowed Bit Widths for Single Port RAM Mode in 36K DRM Mode

Mode	32Kx1	16Kx2	8Kx4	4Kx8	2Kx16	1Kx32	512x64	4Kx9	2Kx18	1Kx36	512x72
SP RAM	√	√	√	√	√	√	√	√	√	√	√

Note:

1. √ indicates supported data width combinations;
2. In the Single Port RAM mode of 36K DRM mode, direct setting of TW and RBW write modes is prohibited for 1Kx32 and 1Kx36 modes, and an additional configuration is required; setting of TW and RBW write modes is prohibited for 512x64 and 512x72 modes. Refer to 5. 5 Read and Write Operations.

Table 5-3 shows the allowed bit widths for Single Port RAM mode in 18K DRM mode.

Table 5-3 Allowed Bit Widths for Single Port RAM Mode in 18K DRM Mode

Mode	16Kx1	8Kx2	4Kx4	2Kx8	1Kx16	512x32	2Kx9	1Kx18	512x36
SP RAM	√	√	√	√	√	√	√	√	√

Note:

1. √ indicates supported data width combinations.
2. In the Single Port RAM mode of 18K DRM mode, direct setting of TW, RBW write modes is prohibited for 512x32 and 512x36 modes, and an additional configuration is required. Refer to 5. 5 Read and Write Operations.

5.4 Timing Parameter

The timing parameters of DRM in SP mode are the same as those in DP mode. Refer to 3.4 Timing Parameter.

5.5 Read and Write Operations

The read and write operations of each write mode in SP mode are similar to the single port in DP mode. Refer to 3.5 Read and Write Operations.

In SP mode, DRM supports setting the write mode to NW (Normal Write) for all bit widths. For the data port widths of 18 bits and below in SP mode, the write mode can also be directly set to TW (Transparent Write) or RBW (Read before Write). However, for the data port widths of 32 bits and more in SP mode, it is prohibited to set the write mode directly to TW and RBW. It must be set to the default NW mode, with details as shown in Table 5-4:

Table 5-4 Availability for Write Mode Settings in SP Mode

Data Port Width	Supported or Not	Implementation of Write Modes
1 (18K/36K)	Supported	In GTP, directly set the mode parameter to SP mode, and for 18K and 36K with widths of 18 bit and below, directly set the write modes to NW, TW, and RBW.
2 (18K/36K)	Supported	
4 (18K/36K)	Supported	
8 (18K/36K)	Supported	
9 (18K/36K)	Supported	
16 (18K/36K)	Supported	
18 (18K/36K)	Supported	
32 (18K/36K)	Not supported	In GTP, write modes cannot be set directly for 32/36 bit widths in SP mode, and the default NW must be used; TW and RBW require additional configurations. Refer to the following section for details.
36 (18K/36K)	Not supported	
64 (36K)	Not supported	In GTP, write modes cannot be set for 64/72 bit widths in SP mode, and the default NW must be used. To set the write mode in this scenario, instantiate two 32/36-bit 36K SP RAMs to implement a width of 64/72 bits, but in some cases, this may cause an increase in the required DRM resources.
72 (36K)	Not supported	

TW and RBW modes cannot be implemented through direct configuration in DRM for 32/36 bits in SP mode; additional configurations are required, with the specific implementation as follows:

➢ 32/36 bits in 18K SP RAM mode:

- Configure both Port A and Port B to 16/18 bits in DP mode (32/36 bits in SP mode);
- Configure Port A and Port B to the same write mode, and other configuration bits to the same;
- Connect the CE, CS, WE, ORCE, RST, CLK, and other control signals of Port A and Port B in parallel;

- Splicing DIA and DIB as data inputs; splicing DOA and DOB as data outputs;
- Connect both Port A and Port B address input ports to the address inputs; connect ADDRA[4] and ADDRB[4] to 0 or 1 (opposite), as shown in Figure 5-2:

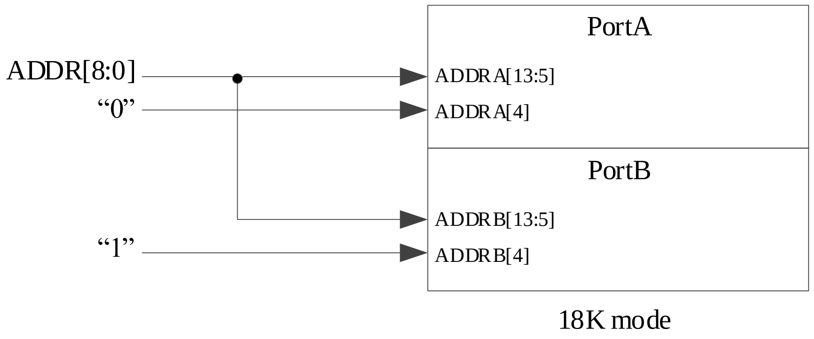

Figure 5-2 Address Line Configuration for TW and RBW Modes in 32/36 Bits 18K SP Mode

- For the connection of BWEA and BWEB signals, refer to the connections of two ports in the 16/18 bits DP mode respectively.

➢ 32/36 bits in 36K SP RAM mode:

- Configure both Port A and Port B to 32/36 bits in DP mode (32/36 bits in SP mode);
- Use one port for read-write operations, set the write enable of the other port to 0, and the read data port is unused, as shown in Figure 5-3:

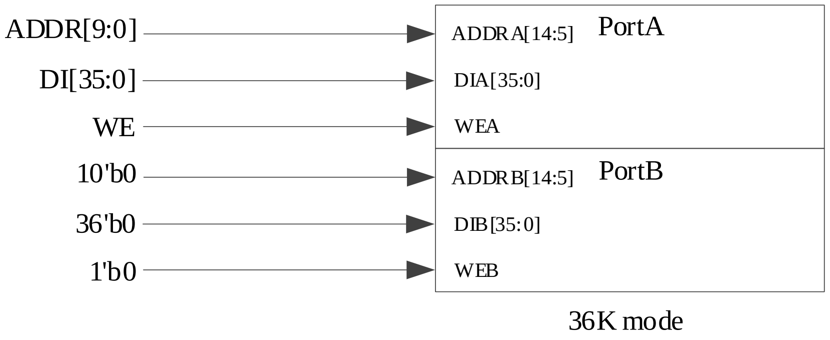

Figure 5-3 Port Signal Configuration for TW and RBW Modes in 36 Bits 36K SP Mode

- For the connection of the byte enable signal, refer to the connections in the 32/36 bits DP mode of read/write ports.

- 64/72 bits in 36K SP RAM mode:

  It is prohibited to configure the write mode as TW or RBW for 64/72 bits in 36K mode. If TW or RBW mode is required in this scenario, instantiate two 32/36 bits 36K SP RAMs (configured as described above) to concatenate data ports to achieve 64/72 bits, as shown in Figure 5-4:

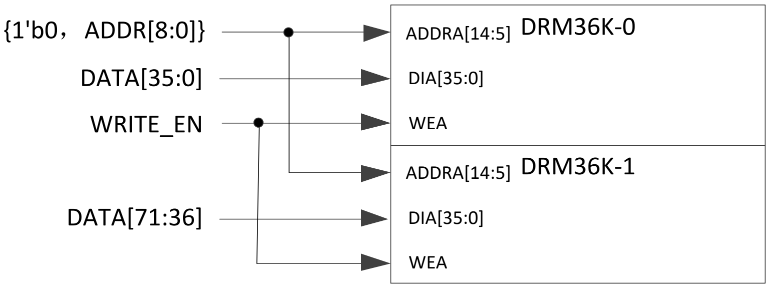

Figure 5-4 Port Signal Configuration for TW and RBW Modes in 72 Bits 36K SP Mode

5.6 Byte Enable

The byte enable function of SP mode is similar to that of DP mode. Refer to 3.6 Byte Enable.

5.7 Internal Registers

The internal registers of SP mode are the same as those of DP mode. Refer to 3.7 Internal Registers.

5.8 Hard Cascade Function

The hard cascade function of SP mode is the same as that of DP mode. Refer to 3.8 Hard Cascade Function.

5.9 Application Example

5.9.1 Single 36K DRM Configuration

This section illustrates the GTP configuration steps for a single 36K DRM. Users can also directly generate the DRM IP using the IP Compiler tool embedded in the Pango Design Suite software. Refer to "DRM/FIFO IP User Guide UG041002" attached in the IP Compiler tool.

The example is for SP mode, with read/write ports configured as 512x64, output register enabled, in NW write mode. The configuration steps for an 18K DRM are similar.

Configure a single 36K DRM as follows:

1. Configure the DRM parameters as per Table 5-5:

Table 5-5 Parameter Configuration of SP Mode for a Single 36K DRM

Parameter Name	**Configuration Value**	**Description**
DATA_WIDTH_A	64	Configure Port A to 512x64 mode
DATA_WIDTH_B		Configure Port B to 512x64 mode
WRITE_MODE_A	"NORMAL_WRITE"	Configure the write mode of Port A to NW
WRITE_MODE_B		Configure the write mode of Port B to NW
DOA_REG	1	Enable Port A output register
DOB_REG		Enable Port B output register
RAM_MODE	"SINGLE_PORT"	Configure DRM to SP mode

2. Connect the DRM ports as per Table 5-6:

Table 5-6 Port Connections of SP Mode for a Single 36K DRM

Port	**Interfacing Signals**	**Description**
ADDRA[15:0]	{1'b1,addr[8:0],6'b0}	Connect Port A/B addresses to the same address signal. Connect the input address signal addr[8:0] to ADDRA[14:6], ADDRA[15] to a high level, and ADDRA[5:0] to a low level. The same with ADDRB. For detailed address connection instructions, refer to 9.1. Address and Data Port Mapping.
ADDRB[15:0]		
ADDRA_HOLD	1'b0	If the A/B port address hold function is not used, connect to a low level. If the address hold function is used, Ports A/B must be connected to the same signal.
ADDRB_HOLD		
DIA[35:0]	{1'b0, di[31:24], 1'b0, di[23:16], 1'b0, di[15:8], 1'b0, di[7:0]}	Concatenate Ports A/B to achieve a 64-bit write data port, with DIA inputting the lower 32-bit write data di[31:0]. For detailed data port connection instructions, refer to 9.1. Address and Data Port Mapping.
DIB[35:0]	{1'b0, di[63:56], 1'b0, di[55:48], 1'b0, di[47:40], 1'b0, di[39:32]}	Concatenate Ports A/B to achieve a 72-bit write data port, with DIB inputting the upper 32-bit write data di[63:32]. DIB[8], DIB[17], DIB[26] and DIB[35] serve as byte additional information bits. Refer to 9.4. Additional Information Bits for Bytes.
CSA[2:0]	3'b0	Do not use the address extension function, connecting to a low level. If the address extension function is used, Ports A/B must be connected to the same signal.
CSB[2:0]		
BWEA[7:0]	8'hff	Byte enable function is not used. Connect BWEA to a high level.
BWEB[3:0]	4'h0	BWEB is not used as a byte enable signal. Connect it to a low level.
DOA[35:0]	{1'bz, do[31:24], 1'bz, do[23:16], 1'bz, do[15:8], 1'bz, do[7:0]}	Concatenate Ports A/B to achieve a 64-bit read data port, with DOA outputting the lower 32-bit read data do[31:0]. Leave DOA[8], DOA[17], DOA[26] and DOA[35] unused.
DOB[35:0]	{1'bz, do[63:56], 1'bz, do[55:48], 1'bz, do[47:40], 1'bz, do[39:32]}	Concatenate Ports A/B to achieve a 64-bit read data port, with DOB outputting the upper 32-bit read data do[63:32]. Leave DOB[8], DOB[17], DOB[26] and DOB[35] unused.

3. Other signals for read/write ports: Connect the clock, clock enable, output register clock enable, data register reset and write enable of the read/write ports to the corresponding Ports A/B of GTP respectively; Ports A/B must be connected to the same signal.
4. Other unused parameters: Do not set other unused parameters, use default values;
5. Other unused ports: Leave other unused ports floating (ports that must be connected even if not used have been described above).

The configured GTP is as follows:

```
GTP_DRM36K_E1 #(
.DATA_WIDTH_A   (64),
.DATA_WIDTH_B   (64),
.WRITE_MODE_A   ("NORMAL_WRITE"),
.WRITE_MODE_B   ("NORMAL_WRITE"),
.DOA_REG            (1),
.DOB_REG            (1),
.RAM_MODE           ("SINGLE_PORT")
) GTP_DRM36K_E1_inst (
.DOA                    (doa[35:0] ), // OUTPUT[35:0]
.DOB                    (dob[35:0] ), // OUTPUT[35:0]
.ADDRA                   ({1'b1,addr[8:0],6'b0}), // INPUT[15:0]
.ADDRB                   ({1'b1,addr[8:0],6'b0}), // INPUT[15:0]
.BWEA                   (8'hff         ), // INPUT[7:0]
.BWEB                   (4'h0          ), // INPUT[3:0]
.CSA                   (3'b0          ), // INPUT[2:0]
.CSB                   (3'b0          ), // INPUT[2:0]
.DIA                   ({1'b0,di[31:24],1'b0,di[23:16], 1'b0, di[15:8],1'b0,di[7:0]}), // INPUT[35:0]
.DIB                   ({1'b0,di[63:56],1'b0,di[55:48],1'b0,di[47:40],1'b0,di[39:32]}), // INPUT[35:0]
.ADDRA_HOLD       (1'b0          ), // INPUT
.ADDRB_HOLD       (1'b0          ), // INPUT
.CEA                   (ce             ), // INPUT
.CEB                   (ce             ), // INPUT
.CLKA                   (clk            ), // INPUT
.CLKB                   (clk            ), // INPUT
.ORCEA                  (orce          ), // INPUT
.ORCEB                  (orce          ), // INPUT
.RSTA                 (rst            ), // INPUT
.RSTB                 (rst            ), // INPUT
.WEA                  (we             ), // INPUT
.WEB                  (we             )    // INPUT
);
assign do[63:0] = {dob[34:27],dob[25:18],dob[16:9],dob[7:0],doa[34:27],doa[25:18],doa[16:9],doa[7:0]};
```

5.9.2 Configuration for Multiple 36K DRMs Cascaded

Cascade of multiple 36K DRMs in SP mode is the same as that in DP mode. Refer to 3.9.2 Configuration for Multiple 36K DRMs Cascaded.

5.9.3 Hard Cascade Configuration for Multiple 36K DRMs

Hard cascade of multiple 36K DRMs in SP mode is the same as that in DP mode. Refer to 3.9.3 Hard Cascade Configuration for Multiple 36K DRMs.

Chapter 6 ROM Mode

6.1 Introduction

The port mode of the RAM is determined by the parameter RAM_MODE. When the value of RAM_MODE is "ROM", the RAM enters ROM mode. This document will detail the 36K ROM, which has essentially the same structure and functions as the 18K, with the only difference being the bit width.

ROM mode only supports data read and does not support data write.

6.2 Ports Descriptions

The DRM can be configured as ROM, with ROM contents initialized through the configuration interface. In ROM mode, the ports for both 18K and 36K modes are read-only. Figure 6-1 illustrates the structure of ROM mode.

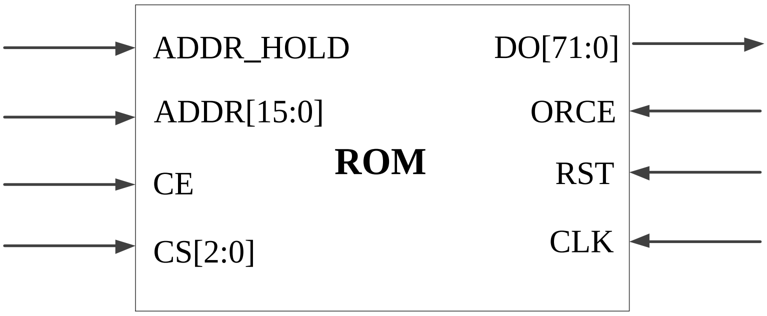

Figure 6-1 ROM Data Ports

Table 6-1 lists the port names and descriptions for ROM mode.

Table 6-1 Port Names and Descriptions of ROM Mode

Port	Direction	Description	Port	Direction	Description
ADDR_HOLD	Input	Address hold signal	DO	Output	Read data output
ADDR	Input	Address Inputs	ORCE	Input	Output register clock enable
CE	Input	Input register clock enable	RST	Input	Output register reset
CS	Input	Address extension	CLK	Input	Port clock

Note:

1. Under ROM mode with x64/x72 bit width, DI is formed by concatenating DIA/DIB from Ports A/B, and DO likewise. Refer to 9. 1 Address and Data Port Mapping.
2. The 18K DRM has no CSA/CSB port and does not support address extension. To extend address depth through DRM cascading, it is recommended to use 36K DRM.

6.3 Bit Width Combinations

The port bit width of RAM is determined by the parameter DATA_WIDTH_A/DATA_WIDTH_B in the GTP. For example, when the value of DATA_WIDTH_A is 4, the data width of Port A is set to 4 bits. Ports A and B in ROM mode must be set to the same data width.

Table 6-2 shows the allowed bit widths for ROM mode in 36K DRM mode.

Table 6-2 Allowed Bit Widths for ROM Mode in 36K DRM Mode

Mode	**32Kx1**	**16Kx2**	**8Kx4**	**4Kx8**	**2Kx16**	**1Kx32**	**512x64**	**4Kx9**	**2Kx18**	**1Kx36**	**512x72**
ROM	√	√	√	√	√	√	√	√	√	√	√

Note: √ indicates the supported bit width combinations;

Table 6-3 shows the allowed bit widths for ROM mode in 18K DRM mode.

Table 6-3 Allowed Bit Widths for ROM Mode in 18K DRM Mode

Mode	**16Kx1**	**8Kx2**	**4Kx4**	**2Kx8**	**1Kx16**	**512x32**	**2Kx9**	**1Kx18**	**512x36**
ROM	√	√	√	√	√	√	√	√	√

Note: √ indicates the supported bit width combinations.

6.4 Timing Parameter

In ROM mode, DRM does not support write operations; the timing parameters and the timing of read operations are the same as those in DP mode. Refer to 3.4 Timing Parameter.

6.5 Internal Registers

The internal registers of ROM mode are the same as those of DP mode. Refer to 3.7 Internal Registers.

6.6 Hard Cascade Function

The hard cascade function of ROM mode is the same as that of DP mode. Refer to 3.8 Hard Cascade Function.

6.7 Application Example

6.7.1 Single 36K DRM Configuration

This section illustrates the GTP configuration steps for a single 36K DRM. Users can also directly generate the DRM IP using the IP Compiler tool embedded in the Pango Design Suite software. Refer to "DRM/FIFO IP User Guide UG041002" attached in the IP Compiler tool.

The example is for ROM mode, with the read/write port configured as 2Kx18 and output register enabled. The configuration steps for an 18K DRM are similar.

Configure a single 36K DRM as follows:

1. Configure the DRM parameters as per Table 6-4:

Table 6-4 Parameter Configuration of ROM Mode for a Single 36K DRM

Parameter Name	Configuration Value	Description
DATA_WIDTH_A	18	Configure Port A to 2Kx18 mode
DATA_WIDTH_B		Configure Port B to 2Kx18 mode
DOA_REG	1	Enable Port A output register
DOB_REG		Enable Port B output register
RAM_MODE	"ROM"	Configure DRM to ROM mode
INIT_00	{18'h0000f, ……… 18'h00003, 18'h00002, 18'h00001, 18'h00000}	Set the initial value parameters of the ROM to the corresponding address values, that is, the 1-18 bits of INIT_00 corresponding to address 0 are set to 0; the 19-36 bits corresponding to address 1 are set to 1, and so on. Refer to the specific initialization parameter mapping rules in 9.2 Initialization Configuration Parameter Mapping;
…...	……	
INIT_7F	{18'h007ff, ……… 18'h007f3, 18'h007f2, 18'h007f1, 18'h007f0}	
RSTA_VAL	{18'h01234}	Set the reset value of the output data to 18'h01234.
RSTB_VAL		

2. Connect the DRM ports as per Table 6-5:

Table 6-5 Port Connections of ROM Mode for a Single 36K DRM

Port	Interfacing Signals	Description
ADDRA[15:0]	{1'b1,addr[10:0],4'b0}	Connect Port A/B addresses to the same address signal. Connect the input address signal addr[10:0] to ADDRA[14:4], ADDRA[15] to a high level, and ADDRA[3:0] to a low level. The same with ADDRB. For detailed address connection instructions, refer to 9.1 Address and Data Port Mapping.
ADDRB[15:0]		
ADDRA_HOLD	1'b0	If the A/B port address hold function is not used, connect to a low level. If the address hold function is used, Ports A/B must be connected to the same signal.
ADDRB_HOLD		
DIA[35:0]	Unused	ROM does not support write operations, with ports DIA/DIB floating
DIB[35:0]		
CSA[2:0]	3'b0	Do not use the address extension function, connecting to a low level. If the address control function is used, Ports A/B must be connected to the same signal.
CSB[2:0]		
BWEA[7:0]	8'h0	ROM does not support write operations. Connect

Port	Interfacing Signals	Description
BWEB[3:0]	4'h0	BWEA/BWEB to a low level.
DOA[35:0]	do[17:0]	Connect the output data signal do[17:0] to DOA[17:0]
DOB[35:0]	Unused	No data concatenation performed, with DOB floating
WEA	1'b0	ROM does not support write operations, connecting WEA/WEB to a low level
WEB		

3. Other signals for read/write ports: Connect the clock, clock enable, output register clock enable, and data register reset of the read/write ports to the corresponding Ports A/B of GTP respectively; Ports A/B must be connected to the same signal.
4. Other unused parameters: Do not set other unused parameters, use default values;
5. Other unused ports: Leave other unused ports floating (ports that must be connected even if not used have been described above).

The configured GTP is as follows:

```
GTP_DRM36K_E1 #(
.DATA_WIDTH_A    (18),
.DATA_WIDTH_B    (18),
.DOA_REG             (1),
.DOB_REG             (1),
.RAM_MODE            ("ROM"),
.RSTA_VAL          ({18'h01234}),
.RSTB_VAL          ({18'h01234}),
.INIT_00             ({18'h0000f, 18'h0000e, 18'h0000d, 18'h0000c, 18'h0000b, 18'h0000a, 18'h00009,
18'h00008,  18'h00007,  18'h00006,  18'h00005,  18'h00004,  18'h00003,  18'h00002,  18'h00001,
18'h00000}),
…
…
.INIT_7F             ({18'h007ff,  18'h007fe,  18'h007fd,  18'h007fc,  18'h007fb,  18'h007fa,  18'h007f9,
18'h007f8,   18'h007f7,   18'h007f6,   18'h007f5,   18'h007f4,   18'h007f3,   18'h007f2,   18'h007f1,
18'h007f0})
) GTP_DRM36K_E1_inst (
.DOA                    (do[17:0]     ), // OUTPUT[35:0]
.DOB                    (                     ), // OUTPUT[35:0]
.ADDRA                  ({1'b1,addr[10:0],4'b0}), // INPUT[15:0]
.ADDRB                  ({1'b1,addr[10:0],4'b0}), // INPUT[15:0]
.BWEA           (8'b0       ), // INPUT[7:0]
.BWEB           (4'b0       ), // INPUT[3:0]
.CSA            (3'b0       ), // INPUT[2:0]
.CSB            (3'b0       ), // INPUT[2:0]
.DIA            (           ), // INPUT[35:0]
.DIB            (           ), // INPUT[35:0]
.ADDRA_HOLD     (1'b0       ), // INPUT
.ADDRB_HOLD     (1'b0       ), // INPUT
.CEA            (ce         ), // INPUT
.CEB            (ce         ), // INPUT
.CLKA           (clk        ), // INPUT
.CLKB           (clk        ), // INPUT
.ORCEA          (orce       ), // INPUT
.ORCEB          (orce       ), // INPUT
.RSTA           (rst        ), // INPUT
.RSTB           (rst        ), // INPUT
.WEA            (1'b0       ), // INPUT
.WEB            (1'b0       )  // INPUT
);
```

6.7.2 Configuration for Multiple 36K DRMs Cascaded

Cascade of multiple 36K DRMs in ROM mode is the same as that in DP mode. Refer to 3.9.2 Configuration for Multiple 36K DRMs Cascaded.

6.7.3 Hard Cascade Configuration for Multiple 36K DRMs

Hard cascade of multiple 36K DRMs in ROM mode is the same as that in DP mode. Refer to 3.9.3 Hard Cascade Configuration for Multiple 36K DRMs.

Chapter 7 FIFO Mode

7.1 Introduction

DRM_FIFO consists of DRM and DRM_FIFO_CTRL modules, and the DTM_FIFO_CTRL module includes: read/write pointer generation and status flag generation logic. Figure 7-1 shows the top-level structural block diagram of FIFO:

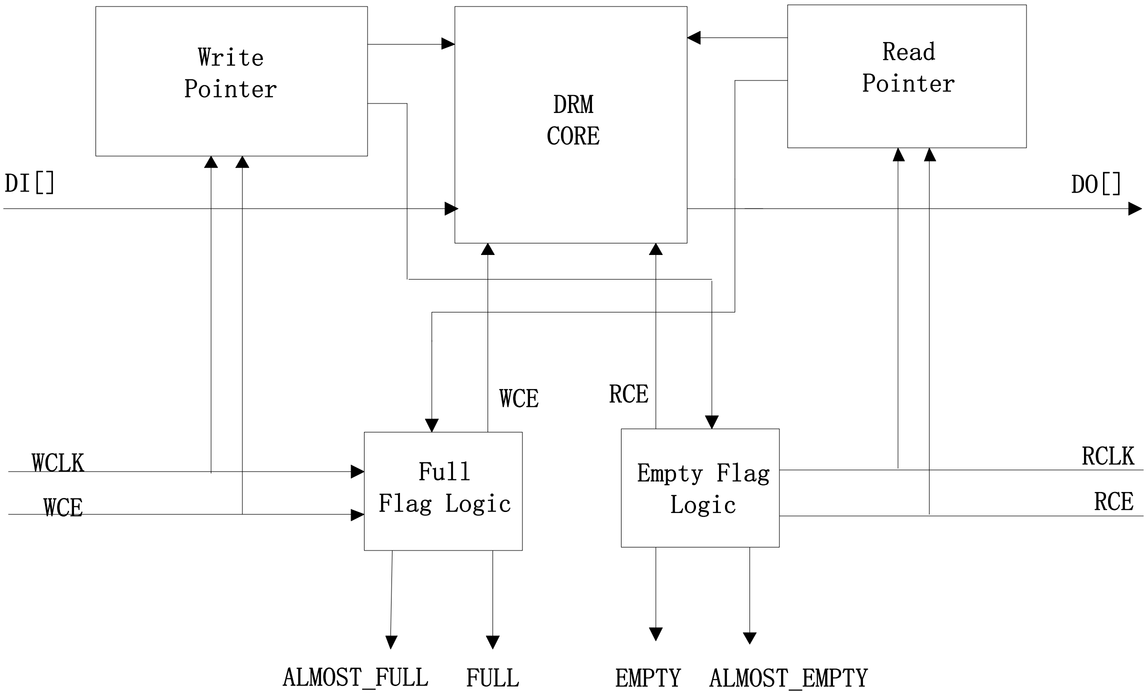

Figure 7-1 Top-Level Structural Block Diagram of FIFO

Users can instantiate GTP_DRM to configure it to SDP mode and then implement the DRM_FIFO_CTRL module through user logic, or directly instantiate GTP_FIFO to implement DRM_FIFO. This document will detail the 36K GTP_FIFO, which has essentially the same structure and functions as the 18K, with the only difference being the bit width.

GTP_FIFO is divided into synchronous and asynchronous FIFO, determined by the parameter SYNC_FIFO. When the value of SYNC_FIFO is "TRUE", synchronous FIFO mode is enabled; when the value of SYNC_FIFO is "FALSE", asynchronous FIFO mode is enabled.

Figure 7-2 shows the asynchronous FIFO mode. In this mode, when write enable is active and FIFO is not full, data is written to FIFO on the rising edge of the write clock; when read enable is active and FIFO is not empty, data is read from FIFO on the rising edge of the read clock. When the FIFO is full, a full signal is generated and synchronized to the write clock domain. When the FIFO is empty, an empty signal is generated and synchronized to the read clock domain.

Figure 7-3 shows the synchronous FIFO mode. In this mode, read and write share the same clock. When write enable is active and FIFO is not full, data is written to FIFO on the rising edge of the clock; when read enable is active and FIFO is not empty, data is read from FIFO on the rising edge of the clock. A full signal is generated when the FIFO is written full, and an empty signal is generated when the FIFO is read empty.

7.2 Ports Descriptions

Figure 7-2 shows the structural diagram in asynchronous FIFO mode:

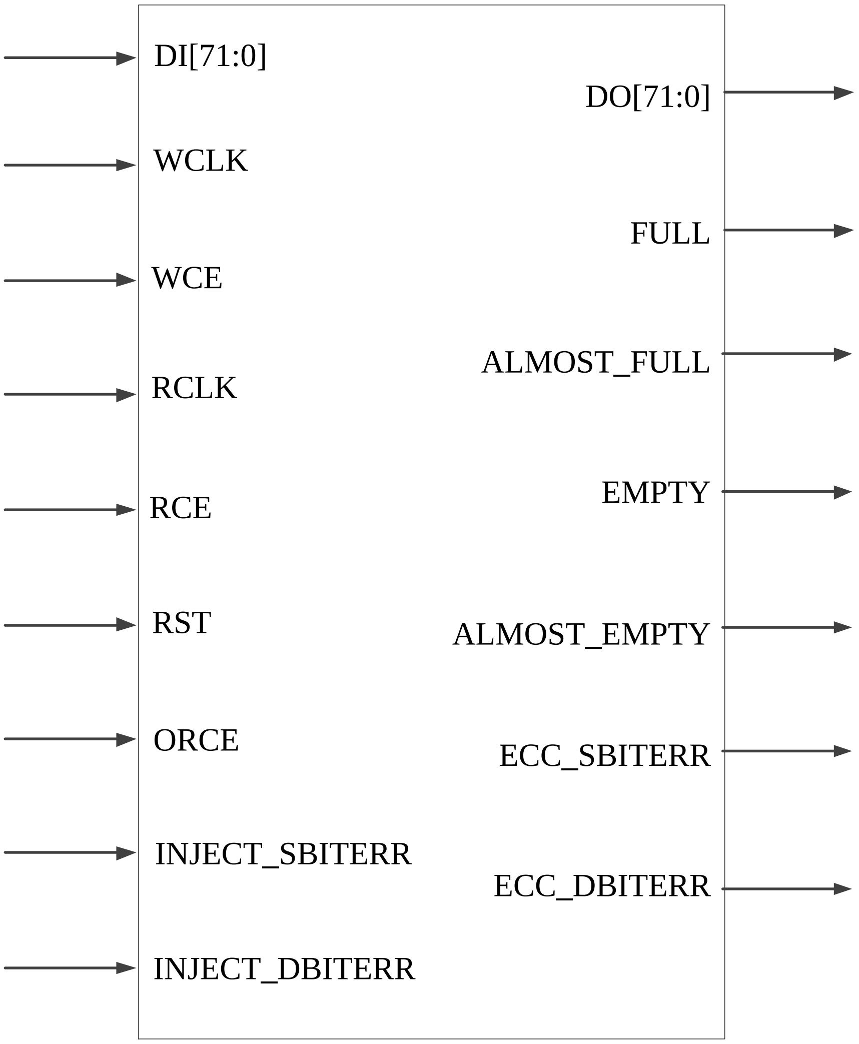

Figure 7-2 Asynchronous FIFO Mode

Table 7-1 lists the port names and descriptions for asynchronous FIFO mode.

Table 7-1 Port Names and Descriptions of Asynchronous FIFO Mode

Port	Direction	Description	Port	Direction	Description
DI	Input	Write data input	DO	Output	Read data output
WCLK	Input	Write port clock	RCLK	Input	Read port clock
WCE	Input	Write port clock enable	RCE	Input	Read port clock enable
RST	Input	Output register reset	ORCE	Input	Output register clock enable
FULL	Output	Full flag signal	EMPTY	Output	Empty flag signal
ALMOST_FULL	Output	Almost full flag signal	ALMOST_EMPTY	Output	Almost empty flag signal
INJECT_SBITERR	Input	ECC mode single-bit error injection	ECC_SBITERR	Output	ECC mode single-bit error flag
INJECT_DBITERR	Input	ECC mode dual-bit error injection	ECC_DBITERR	Output	ECC mode dual-bit error flag

Figure 7-3 shows the structural diagram in synchronous FIFO mode:

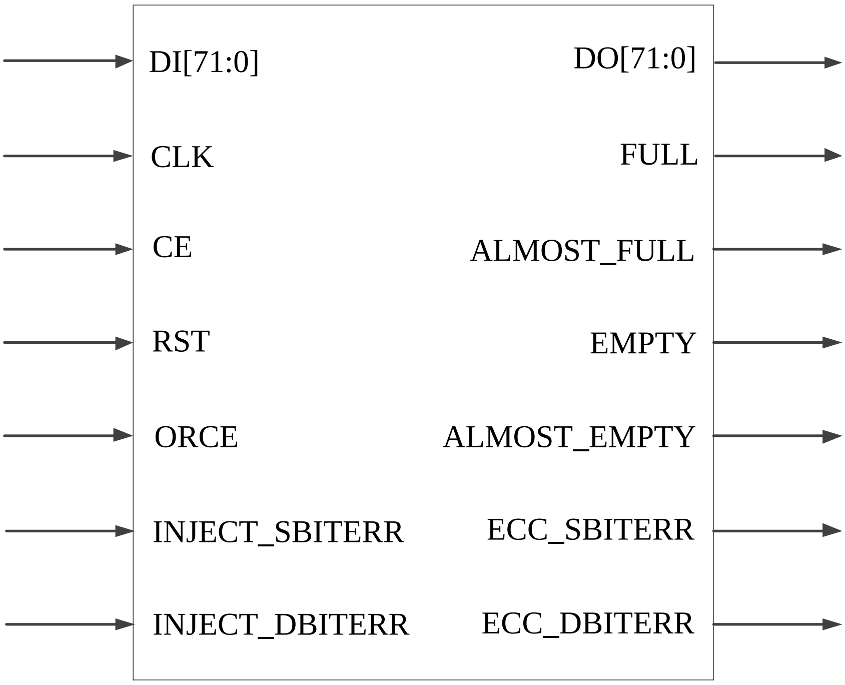

Figure 7-3 Synchronous FIFO Mode

Table 7-2 lists the port names and descriptions for synchronous FIFO mode.

Table 7-2 Port Names and Descriptions of Synchronous FIFO Mode

Port	Direction	Description	Port	Direction	Description
DI	Input	Write data input	DO	Output	Read data output

CLK	Input	Port clock	CE	Input	Port clock enable
RST	Input	Output register reset	ORCE	Input	Output register clock enable
FULL	Output	Full flag signal	EMPTY	Output	Empty flag signal
ALMOST_FULL	Output	Almost full flag signal	ALMOST_EMPTY	Output	Almost empty flag signal
INJECT_SBITERR	Input	ECC mode single-bit error injection	ECC_SBITERR	Output	ECC mode single-bit error flag
INJECT_DBITERR	Input	ECC mode dual-bit error injection	ECC_DBITERR	Output	ECC mode dual-bit error flag

7.3 Bit Width Combinations

The port width of GTP_FIFO is determined by the parameter DATA_WIDTH in the GTP; for example, when the value of DATA_WIDTH is 4, the port data width is set to 4 bits. GTP_FIFO does not support mixed bit widths.

Table 7-3 shows the allowed bit width combinations for 36K GTP_FIFO mode.

Table 7-3 Allowed Bit Widths for 36K GTP_FIFO Mode

Mode	**32Kx1**	**16Kx2**	**8Kx4**	**4Kx8**	**2Kx16**	**1Kx32**	**512x64**	**4Kx9**	**2Kx18**	**1Kx36**	**512x72**
FIFO	√	√	√	√	√	√	√	√	√	√	√

Note: √ indicates the supported bit width combinations;

Table 7-4 shows the allowed bit width combinations for 18K GTP_FIFO mode.

Table 7-4 Allowed Bit Widths for 18K GTP_FIFO Mode

Mode	**16Kx1**	**8Kx2**	**4Kx4**	**2Kx8**	**1Kx16**	**512x32**	**2Kx9**	**1Kx18**	**512x36**
FIFO	√	√	√	√	√	√	√	√	√

Note: √ indicates the supported bit width combinations.

7.4 Read and Write Operations

The read and write operation instructions in this section only apply to GTP_FIFO, specifically GTP_FIFO36K_E1 and GTP_FIFO18K_E1.

7.4.1 FIFO Timing Parameters

Table 7-5 lists the typical timing parameters and descriptions for 36K GTP_FIFO. The timing parameters and diagrams provided hereinafter are all based on 36K GTP_FIFO. The timing parameters and diagrams for 18K GTP_FIFO are similar to those of 36K GTP_FIFO. For detailed timing parameters, refer to "***DS04001_Logos2 Family FPGA Device Datasheet***".

Table 7-5 Typical Timing Parameters for GTP_FIFO

Parameter	**Control Signals**	**Description**
Tsu_fifo_wctl	WCE	Write enable signal setup time
Thd_fifo_wctl		Write enable signal hold time
Tsu_fifo_rctl	RCE	Read enable signal setup time
Thd_fifo_rctl		Read enable signal hold time
Tsu_36K_d	DI	Input data setup time
Thd_36K_d		Input data hold time
Tco_36K	RCLK to Q	Data output delay with respect to clock edge (without output register)
Tco_36K_reg		Data output delay with respect to clock edge (with output register)
Tco_flag_full	WCLK to ALMOSTFULL/FULL	The delay of each flag signal with respect to the clock edge
Tco_flag_empty	RCLK to ALMOSTEMPTY/EMPTY	

7.4.2 Write Timing

The EMPTY signal indicates the FIFO is empty. When WCE is active and data is successfully written, in synchronous FIFO mode, the EMPTY signal is set to "0" after 1 CLK clock cycle; in asynchronous FIFO mode, the EMPTY signal is set to "0" at the 3rd RCLK clock rising edge (rising edge ② in Figure 7-4) after valid data is successfully written (rising edge ① in Figure 7-4). When WCE remains active, the ALMOST_EMPTY signal is cleared (rising edge ④ in Figure 7-4) with a delay based on the configuration of ALMOST_FULL_OFFSET. If the rising edge of WCLK is close to that of RCLK, the EMPTY and ALMOST_EMPTY signals may be cleared with an additional delay of one RCLK clock cycle.

Figure 7-4 shows the write timing to empty asynchronous FIFO:

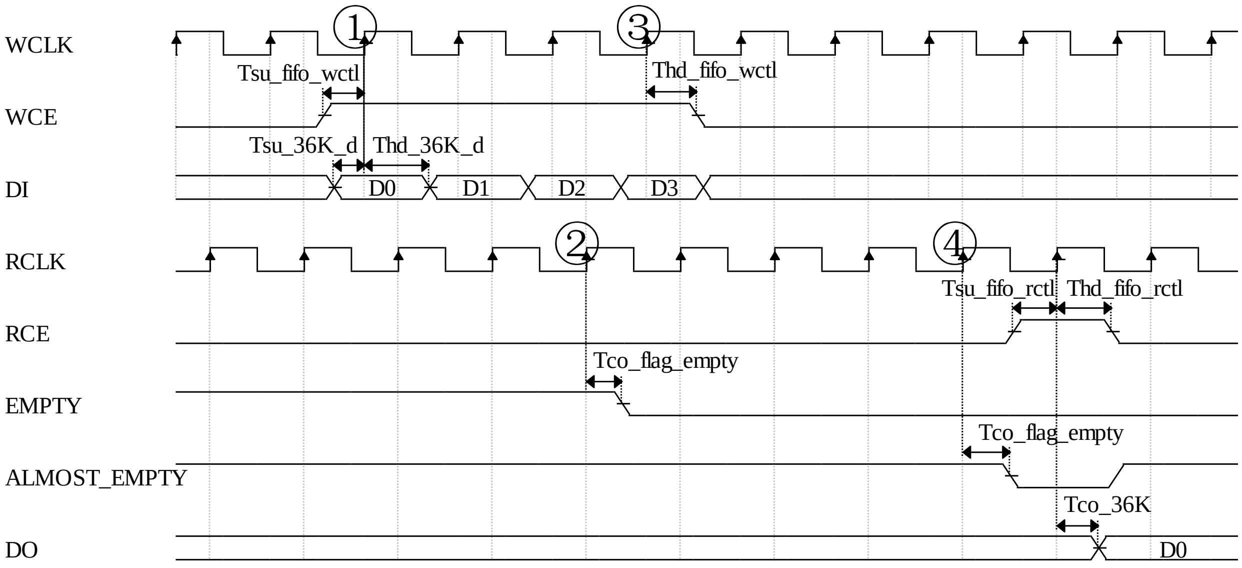

Figure 7-4 Write Timing to Empty Asynchronous FIFO

When the FIFO is almost full and the difference between the write and read pointers equals to the set value, the ALMOST_FULL flag is set to 1 at the WCLK rising edge (rising edge ② in Figure 7-5), according to the configuration of ALMOST_FULL_OFFSET. When the FIFO is full (rising edge ③ in Figure 7-5), the write pointer will not increase, halting any further write operations, and the FULL signal is immediately set to 1.

Figure 7-5 shows the write timing to almost full asynchronous FIFO:

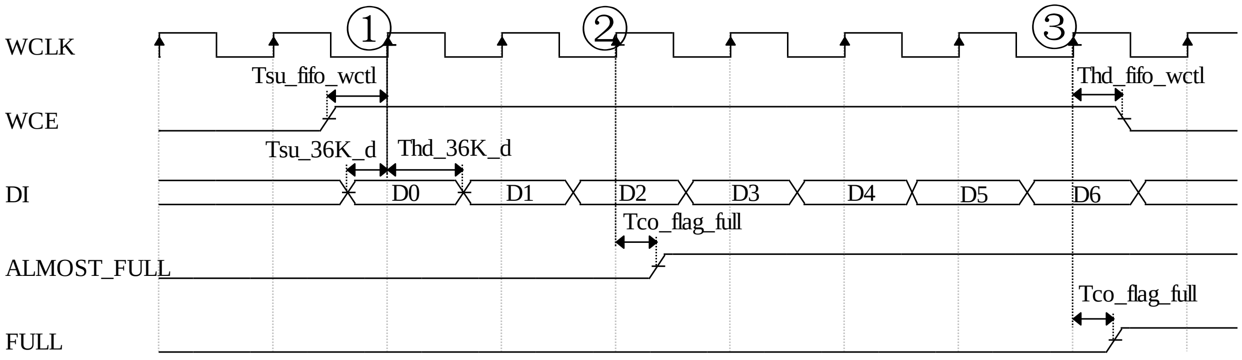

Figure 7-5 Write Timing to Almost Full Asynchronous FIFO

7.4.3 Read Timing

The FULL signal indicates that the FIFO is full. After valid data is read out with RCE, in synchronous FIFO mode, the FULL signal is set to "0" after 1 CLK clock cycle; in asynchronous FIFO mode, the FULL signal is set to "0" at the 3rd WCLK clock rising edge (rising edge ② in Figure 7-6) after valid data is read out (rising edge ① in Figure 7-6). When RCE remains active, the ALMOST_FULL signal is cleared with a delay (rising edge ③ in Figure 7-6) based on the configuration of ALMOST_FULL_OFFSET. If the rising edge of WCLK is close to that of RCLK, the FULL and ALMOST_FULL signals may be cleared with an additional delay of one WCLK clock cycle.

Figure 7-6 shows the read timing from full asynchronous FIFO:

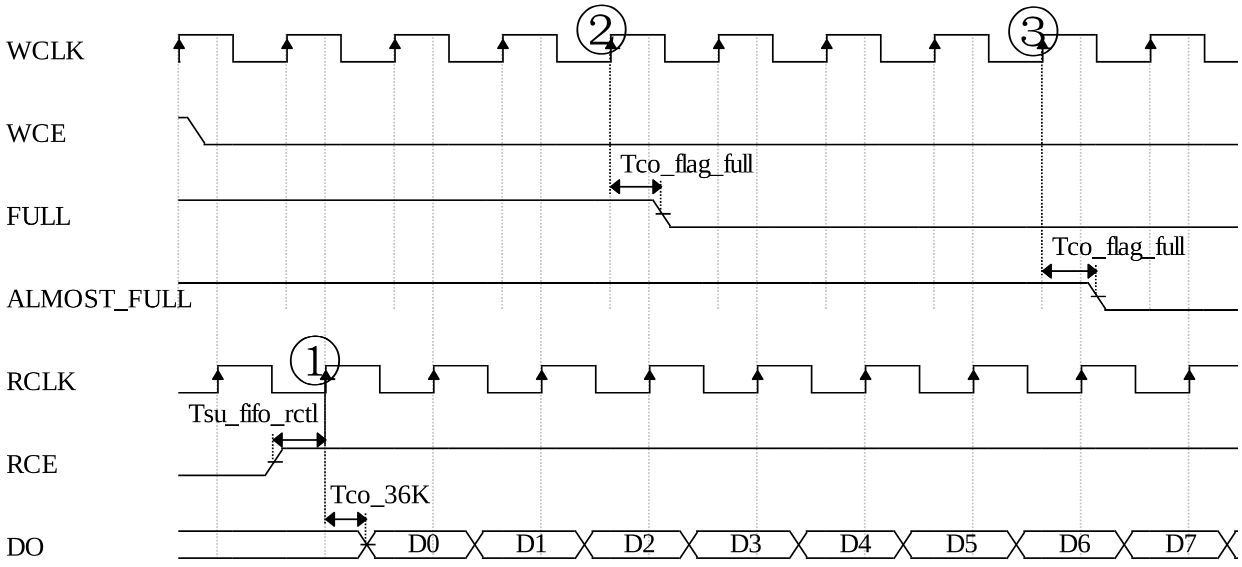

Figure 7-6 Read Timing from Full Asynchronous FIFO

When the FIFO is almost empty and the difference between the write and read pointers equals to the set value, ALMOST_EMPTY is set to 1 in advance (rising edge ② in Figure 7-7), according to the configuration of ALMOST_FULL_OFFSET. When the FIFO is empty (rising edge ③ in Figure 7-7), the read pointer will not increase, and the EMPTY signal is immediately set to 1.

Figure 7-7 shows the read timing from almost empty asynchronous FIFO:

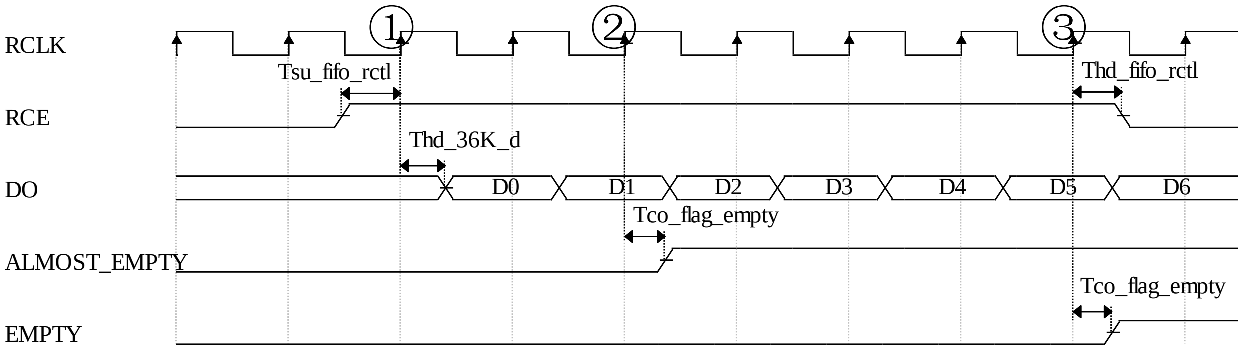

Figure 7-7 Read Timing from Almost Empty Asynchronous FIFO

7.4.4 Flag Signal Reset Timing

The reset signal RST is an asynchronous reset signal used to reset all flag signals. When it is set to 1, the flag signals EMPTY and ALMOST_EMPTY are set to 1, and FULL and ALMOST_FULL are set to 0. After setting RST to 1, maintain the reset for at least 5 read and write clock cycles to ensure that all internal states and flag signals of the FIFO are reset to the correct values. Figure 7-8 shows the reset timing diagrams for all flag signals:

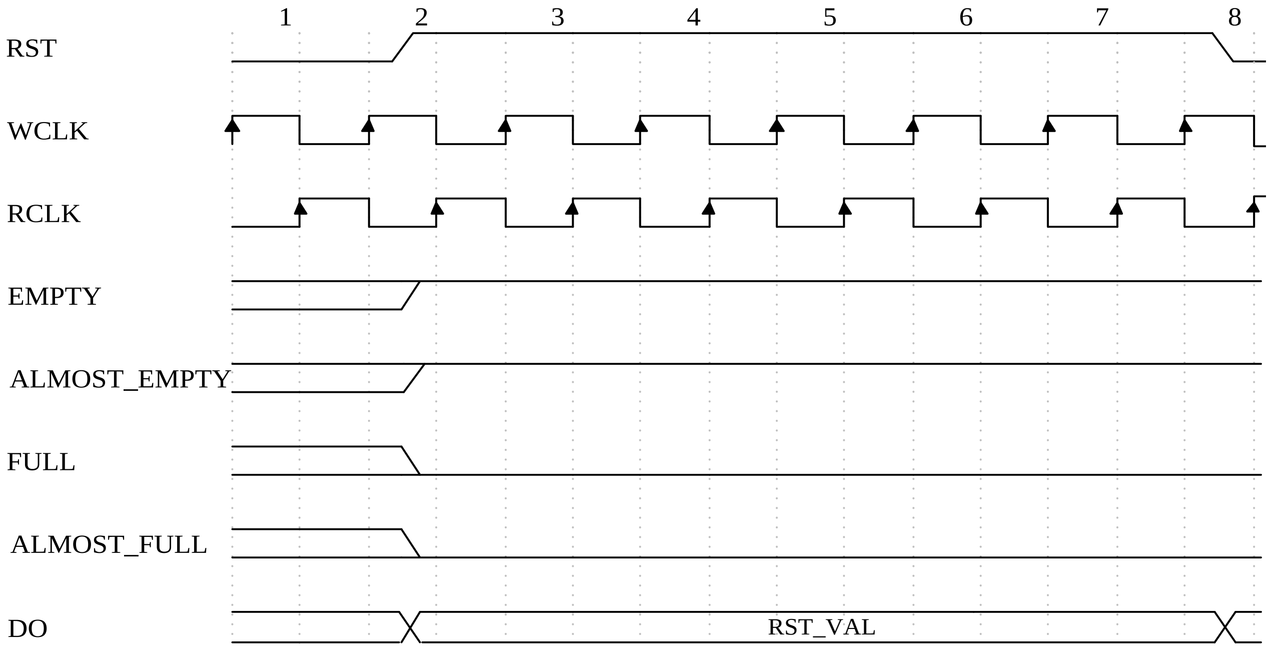

Figure 7-8 FIFO Flag Signal Reset Timing Diagram

7.5 Internal Registers

The internal registers of FIFO mode are the same as those of DP mode. Refer to 3.7 Internal Registers.

7.6 Application Example

This section illustrates the configuration steps for a single 36K GTP_FIFO. Users can also directly generate the GTP_DRM base FIFO IP using the IP Compiler tool embedded in the Pango Design Suite software. Refer to "DRM/FIFO IP User Guide UG041002" attached in the IP Compiler tool.

Asynchronous FIFO mode is taken as an example, with the port data width configured as 1Kx36 and output registers enabled. The configuration steps for a 18K FIFO are similar.

Configure a single 36K FIFO as follows:

1. Configure the FIFO parameters as per Table 7-6:

Table 7-6 Parameter Configuration of a Single 36K FIFO

Parameter Name	**Configuration Value**	**Description**
DATA_WIDTH	36	Configure the port to 1Kx36 mode
DO_REG	1	Enable port output register
SYNC_FIFO	"FALSE"	Asynchronous FIFO mode
USE_EMPTY	1	Empty flag enable
USE_FULL	1	Full flag enable
ALMOST_EMPTY_OFFSET	'd4	Almost empty signal is set to 1 when the number of data items stored in the FIFO is less than or equal to 4
ALMOST_FULL_OFFSET	'd1020	Almost full signal is set to 1 when the number of data items stored in the FIFO is larger than or equal to 1020

2. Connect the FIFO port as per Table 7-7:

Table 7-7 Port Connections of a Single 36K FIFO

Port	**Interfacing Signals**	**Description**
DI[71:0]	di[35:0]	Connect the input data signal di[35:0] to DI[35:0], leaving high-bit DI ports unused or connected to a low level.
DO[71:0]	do[35:0]	Connect the input data signal do[35:0] to DO[35:0], leaving the high-bit DO ports unused.

3. Other signals for read/write ports: Connect the clock, read/write clock enable, output register clock enable, data register reset, and flag signal of the read/write ports to the corresponding GTP read/write ports respectively;
4. Other unused parameters: Do not set other unused parameters, use default values;
5. Other unused ports: Leave other unused ports floating (ports that must be connected even if not used have been described above).

The configured GTP is as follows:

```
GTP_FIFO36K_E1 #(
.DATA_WIDTH                    (36),
.DO_REG                       (1),
.ALMOST_FULL_OFFSET     ('d1020),
.ALMOST_EMPTY_OFFSET    ('d4),
.USE_EMPTY                    (1),
.USE_FULL                     (1),
.SYNC_FIFO                    ("FALSE")
) GTP_FIFO36K_E1_inst (
.DO                          (do[35:0]      ), // OUTPUT[71:0]
.DI                          (di[35:0]      ), // INPUT[71:0]
.ALMOST_EMPTY                (almost_empty), // OUTPUT
.ALMOST_FULL                 (almost_full ), // OUTPUT
.EMPTY                       (empty         ), // OUTPUT
.FULL                        (full          ), // OUTPUT
.ORCE                        (orce          ), // INPUT
.RCE                         (rce           ), // INPUT
.RCLK                        (rclk          ), // INPUT
.RST                         (rst           ), // INPUT
.WCE                         (wce           ), // INPUT
.WCLK                        (wclk          )    // INPUT
);
```

Chapter 8 ECC Mode

8.1 Introduction

Only when a DRM is in single 36K mode and configured as SDP/FIFO in 512x72 memory mode, does it support 72-bit data ECC with single-bit error correction and dual-bit error detection, including 64 valid data bits and 8 bits for ECC checksum bit storage. At the same time, it outputs ECC_SBITERR (single-bit error correction flag), ECC_DBITERR (dual-bit error detection flag), ECC encode and read address in ECC mode (ECC encode checksum bits and read address are only supported in SDP mode).

To configure ECC mode, set the read/write data width of GTP_DRM36K_E1 (configured as SDP mode) or GTP_FIFO36K_E1 to 72, and then set the parameters ECC_WRITE_EN and ECC_READ_EN to "TRUE".

- When ECC encoding and ECC decoding are enabled:
    - When there is a single-bit error in the 64-bit data readout, the error can be corrected to read out the correct data and set the ECC_SBITERR status flag.
    - When there is a dual-bit error in the 64-bit data readout, the error cannot be corrected to read out data, with the ECC_DBITERR status flag set.
- The ECC encoding module allows users to insert single-bit or dual-bit errors.
    - Insert single-bit errors into WD[30].
    - Insert dual-bit errors into WD[62] and WD[30].

8.2 Ports Descriptions

Figure 8-1 shows the structural diagram of ECC mode for SDP RAM:

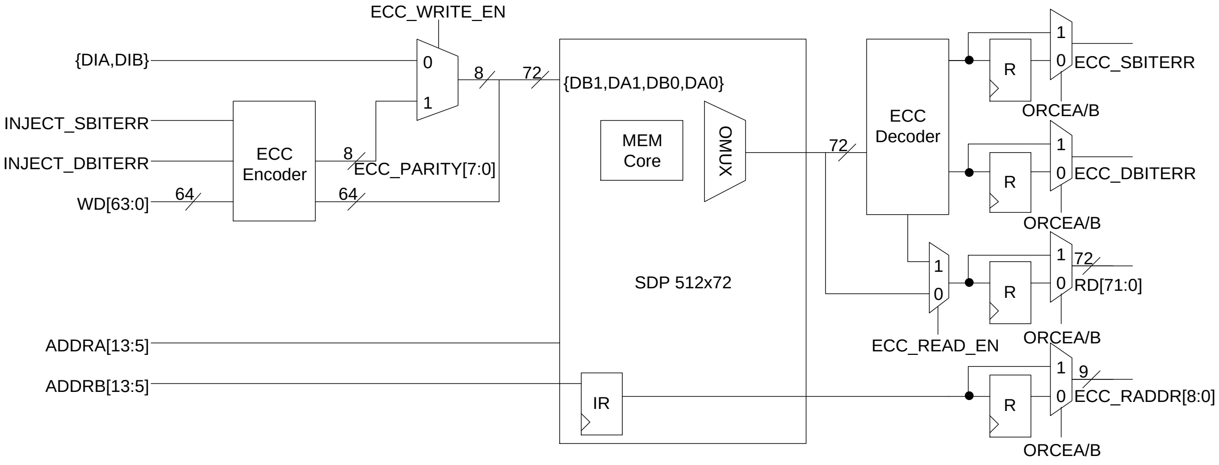

Figure 8-1 Structural Diagram of ECC Mode for SDP RAM

WD[63:0]={DIB[34:27, 25:18, 16:9, 7:0], DIA[34:27, 25:18, 16:9, 7:0]} is the valid 64-bit write data for ECC mode; RD[71:0]={DOB, DOA} is the 72-bit readout data for ECC mode. Among them, ECC_PARITY={DOB[35, 26, 17, 8], DOA[35, 26, 17, 8]} is the 8-bit ECC checksum code output, and dout[63:0]={DOB[34:27, 25:18, 16:9, 7:0], DOA[34:27, 25:18, 16:9, 7:0]} is the valid 64-bit readout data in this mode.

8.3 Read and Write Operations

Figure 8-2 and Figure 8-3 show the read/write timing diagrams of ECC mode for SDP RAM. When A’ = A, the injected single-bit error is corrected, and the single-bit error flag signal is set to 1; when C’’[63, 61:31, 29:0] = C[63, 61:31, 29:0], C’’[62, 30] ≠ C[62, 30], the injected dual-bit error is not corrected but is detected, and the dual-bit error flag signal is set to 1. The read/write timing of ECC mode for FIFO is similar.

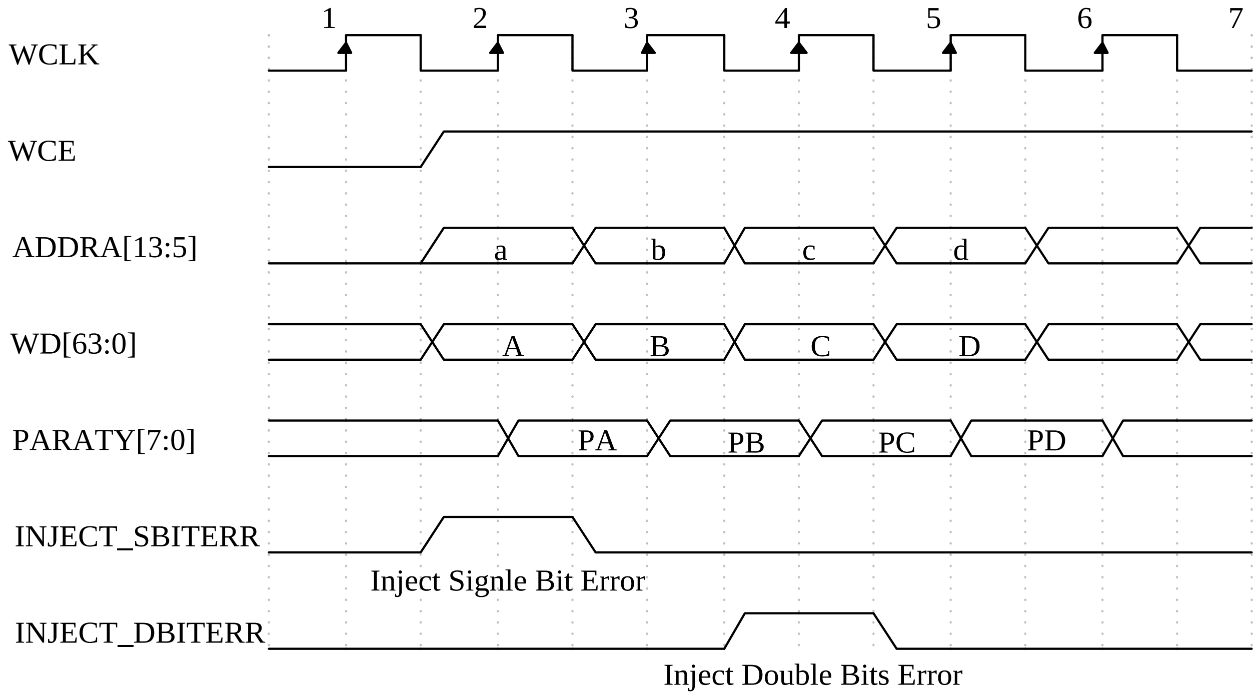

Figure 8-2 Write Timing Diagram of ECC Mode for SDP RAM

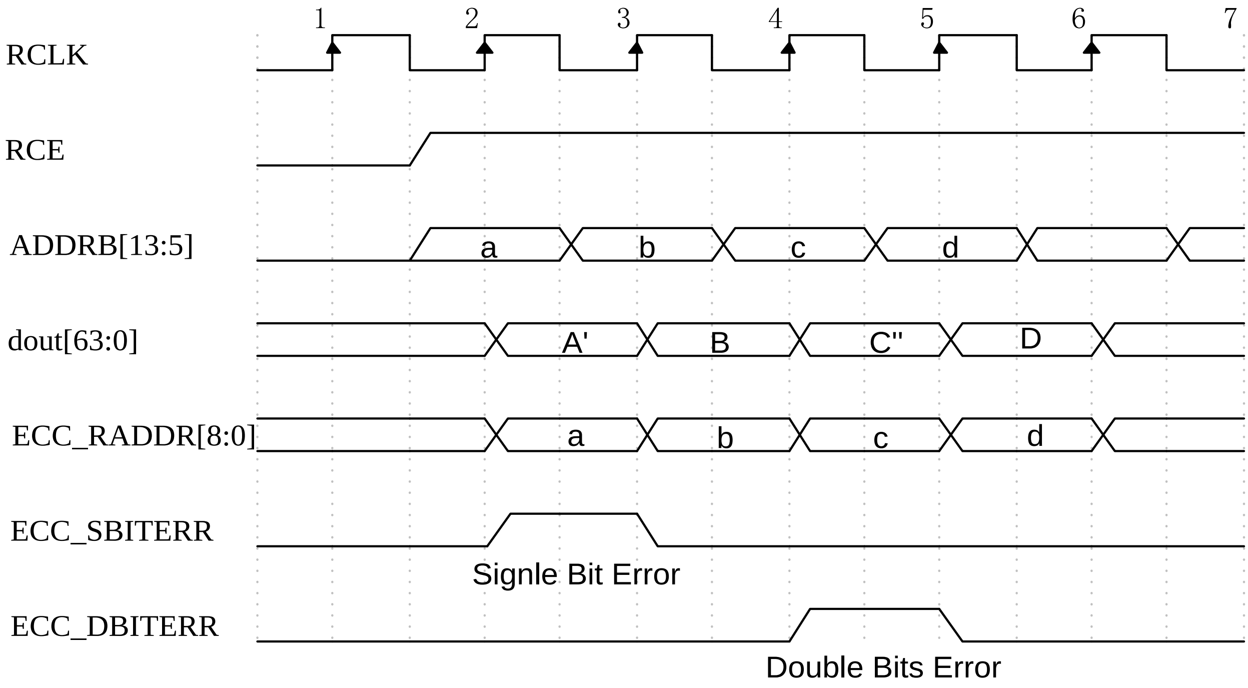

Figure 8-3 Read Timing Diagram of ECC Mode for SDP RAM

8.4 Application Example

8.4.1 Configuration of ECC Mode for SDP RAM

This section provides an example of the ECC mode configuration steps for a single 36K DRM. In the example, the port width is 512x72 (by concatenating the Port A/B DOA/DOB), with different clocks for the two ports and output register enabled.

Configure a single 36K DRM as follows:

1. Configure a single DRM as SDP RAM mode with a port width of 512x72. Refer to 4.9.1 Single 36K DRM Configuration;
2. Configure the ECC parameters of DRM as per Table 8-1:

Table 8-1 Parameter Configuration of ECC Mode for SDP RAM

Parameter Name	Configuration Value	Description
ECC_WRITE_EN	"TRUE"	Enable ECC read/write function
ECC_READ_EN		

3. Connect the DRM ports as shown in Table 8-2:

Table 8-2 Port Connections of ECC Mode for SDP RAM

Port	Interfacing Signals	Description
DIA[35:0]	{1'b0, di[31:24], 1'b0, di[23:16], 1'b0, di[15:8], 1'b0, di[7:0]}	Concatenate Ports A/B to achieve a 72-bit write data port, with DIA inputting the lower 32-bit ECC mode valid write data di[31:0]. DIA[8], DIA[17], DIA[26] and DIA[35] (for byte additional information bits, refer to 9.4 Additional Information Bits for Bytes) serve as ECC checksum bit storage, left unused or connected to a low level.
DIB[35:0]	{1'b0, di[63:56], 1'b0, di[55:48], 1'b0, di[47:40], 1'b0, di[39:32]}	Concatenate Ports A/B to achieve a 72-bit write data port, with DIB outputting the upper 32-bit ECC mode valid write data di[63:32]. DIB[8], DIB[17], DIB[26] and DIB[35] serve as ECC checksum bit storage, left unused or connected to a low level.
DOA[35:0]	{1'bz, do[31:24], 1'bz, do[23:16], 1'bz, do[15:8], 1'bz, do[7:0]}	Concatenate Ports A/B to achieve a 72-bit read data port, with DOA outputting the lower 32-bit ECC mode valid read data do[31:0]. DOA[8], DOA[17], DOA[26] and DOA[35] serve as ECC checksum bit storage and are left unused.
DOB[35:0]	{1'bz, do[63:56], 1'bz, do[55:48], 1'bz, do[47:40], 1'bz, do[39:32]}	Concatenate Ports A/B to achieve a 72-bit read data port, with DOB outputting the upper 32-bit ECC mode valid read data do[63:32]. DOB[8], DOB[17], DOB[26] and DOB[35] serve as ECC checksum bit storage, and are left unused.

4. Other signals for read/write ports: Connect the clock, clock enable, output register clock enable, data register reset and ECC signals of the read/write ports to the corresponding Ports A/B of GTP respectively;
5. Other unused parameters: Do not set other unused parameters, use default values;
6. Other unused ports: Leave other unused ports floating (ports that must be connected even if not used have been described above).

The configured GTP is as follows:

```
GTP_DRM36K_E1 #(
.DATA_WIDTH_A      (72),
.DATA_WIDTH_B      (72),
.WRITE_MODE_A      ("NORMAL_WRITE"),
.WRITE_MODE_B      ("NORMAL_WRITE"),
.DOA_REG           (1),
.DOB_REG           (1),
.RAM_MODE          ("SIMPLE_DUAL_PORT"),
.ECC_READ_EN       ("TRUE"),
.ECC_WRITE_EN      ("TRUE")
) GTP_DRM36K_E1_inst (
.DOA               (doa              ), // OUTPUT[35:0]
.DOB               (dob              ), // OUTPUT[35:0]
.ADDRA             ({1'b1,addr[8:0],6'b0}), // INPUT[15:0]
.ADDRB             ({1'b1,addr[8:0],6'b0}), // INPUT[15:0]
.BWEA              (8'hff            ), // INPUT[7:0]
.BWEB              (4'b0             ), // INPUT[3:0]
.CSA               (3'b0             ), // INPUT[2:0]
.CSB               (3'b0             ), // INPUT[2:0]
.DIA               ({1'b0,di[31:24],1'b0,di[23:16],1'b0,di[15:8],1'b0,di[7:0]}), // INPUT[35:0]
.DIB               ({1'b0,di[63:56],1'b0,di[55:48],1'b0,di[47:40],1'b0,di[39:32]}), // INPUT[35:0]
.ADDRA_HOLD        (1'b0             ), // INPUT
.ADDRB_HOLD        (1'b0             ), // INPUT
.CEA               (ce               ), // INPUT
.CEB               (ce               ), // INPUT
.CLKA              (wr_clk           ), // INPUT
.CLKB              (rd_clk           ), // INPUT
.ORCEA             (orce             ), // INPUT
.ORCEB             (orce             ), // INPUT
.RSTA              (wr_rst           ), // INPUT
.RSTB              (rd_rst           ), // INPUT
.WEA               (we               ), // INPUT
.WEB               (1'b0             ), // INPUT
.ECC_DBITERR      (ecc_dbiterr    ), // OUTPUT
.ECC_SBITERR      (ecc_sbiterr    ), // OUTPUT
.INJECT_DBITERR   (inject_dbiterr), // INPUT
.INJECT_SBITERR   (inject_sbiterr)   // INPUT
);
assign  do[63:0]  =  {dob[34:27],dob[25:18],dob[16:9],dob[7:0],doa[34:27],doa[25:18],doa[16:9],
doa[7:0]};
```

8.4.2 Configuration of ECC Mode for FIFO

This section provides an example of the ECC mode configuration steps for a single 36K FIFO. In the example, the port width is 512x72, with different clocks for the two ports and output register enabled.

Configure a single 36K FIFO as follows:

1. Configure a single FIFO. Refer to 7.6 Application Example;
2. Configure the ECC parameters of FIFO as per Table 8-3:

Table 8-3 Parameter Configuration of ECC Mode for FIFO

Parameter Name	Configuration Value	Description
ECC_WRITE_EN	"TRUE"	Enable ECC read/write function
ECC_READ_EN		

3. Connect the FIFO ports as shown in Table 8-4:

Table 8-4 Port Connections of ECC Mode for FIFO

Port	Interfacing Signals	Description
DI[72:0]	{1'b0, di[63:56], 1'b0, di[55:48], 1'b0, di[47:40], 1'b0, di[39:32], 1'b0, di[31:24], 1'b0, di[23:16], 1'b0, di[15:8], 1'b0, di[7:0]}	DI inputs 64-bit ECC mode valid write data di[63:0]. DI[8], DI[17], DI[26], DI[35], DI[44], DI[53], DI[62] and DI[71] (for byte additional information bits, refer to 9.4 Additional Information Bits for Bytes) serve as ECC checksum bit storage, floating or connected to a low level.
DOB[35:0]	{1'bz, do[63:56], 1'bz, do[55:48], 1'bz, do[47:40], 1'bz, do[39:32], 1'bz, do[31:24], 1'bz, do[23:16], 1'bz, do[15:8], 1'bz, do[7:0]}	DO outputs 64-bit ECC mode valid read data do[63:0]. DO[8], DO[17], DO[26], DO[35], DO[44], DO[53], DO[62] and DO[71] serve as ECC checksum bit storage, and are left unused.

4. Other signals for read/write ports: Connect the clock, clock enable, output register clock enable, data register reset and ECC signals of the read/write ports to the corresponding GTP read/write ports respectively;
5. Other unused parameters: Do not set other unused parameters, use default values;

6. Other unused ports: Leave other unused ports floating (ports that must be connected even if not used have been described above).

The configured GTP is as follows:

```
GTP_FIFO36K_E1 #(
.DATA_WIDTH         (72),
.DO_REG             (1),
.ECC_READ_EN        ("TRUE"),
.ECC_WRITE_EN       ("TRUE"),
.USE_EMPTY          (1),
.USE_FULL           (1),
.SYNC_FIFO          ("FALSE")
) GTP_FIFO36K_E1_inst (
.DO                 (do_gtp          ), // OUTPUT[71:0]
.DI                 ({1'b0, di[63:56], 1'b0, di[55:48], 1'b0, di[47:40], 1'b0, di[39:32], 1'b0, di[31:24], 1'b0, di[23:16], 1'b0, di[15:8], 1'b0, di[7:0]}), // INPUT[71:0]
.EMPTY              (empty           ), // OUTPUT
.FULL               (full            ), // OUTPUT
.ORCE               (orce            ), // INPUT
.RCE                (rce             ), // INPUT
.RCLK               (rclk            ), // INPUT
.RST                (rst             ), // INPUT
.WCE                (wce             ), // INPUT
.WCLK               (wclk            ), // INPUT
.ECC_DBITERR        (ecc_dbiterr     ), // OUTPUT
.ECC_SBITERR        (ecc_sbiterr     ), // OUTPUT
.INJECT_DBITERR     (inject_dbiterr), // INPUT
.INJECT_SBITERR     (inject_sbiterr)   // INPUT
);
assign do[63:0] = {do_gtp[70:63], do_gtp[61:54], do_gtp[52:45], do_gtp[43:36], do_gtp[34:27], do_gtp[25:18], do_gtp[16:9], do_gtp[7:0]};
```

Chapter 9 Appendix A

9.1 Address and Data Port Mapping

Table 9-1 and Table 9-2 show the address and data port mapping for 36K and 18K DRM. The Logos2 Family DRM can be configured with a single 36K-bit memory or two 18K-bit memories.

Table 9-1 Address and Data Port Mapping for 36K DRM Mode

DRM Port Mode	**Address Port Mapping**		**Data Port Mapping**			
	Port A Address	**Port B Address**	**Port A data input bus**	**Port B data input bus**	**Port A data output bus**	**Port B data output bus**
32K*1	ADDRA[14:0]	ADDRB[14:0]	DIA[0]	DIB[0]	DOA[X], where X is any bit between 16~9, 7~0	DOB[X], where X is any bit between 16~9, 7~0
16K*2	ADDRA[14:1]	ADDRB[14:1]	DIA[1:0]	DIB[1:0]	DOA[X+1:X],X=0,2,4,9,11,12,15	DOB[X+1:X],X=0,2,4,9,11,12,15
8K*4	ADDRA[14:2]	ADDRB[14:2]	DIA[3:0]	DIB[3:0]	DOA[X+3:X],X=0,4,9,12	DOB[X+3:X],X=0,4,9,12
4K*8	ADDRA[14:3]	ADDRB[14:3]	DIA[7:0]	DIB[7:0]	DOA[7:0] or DOA[16:9]	DOB[7:0] or DOB[16:9]
4K*9	ADDRA[14:3]	ADDRB[14:3]	DIA[8:0]	DIB[8:0]	DOA[8:0] or DOA[17:9]	DOB[8:0] or DOB[17:9]
2K*16	ADDRA[14:4]	ADDRB[14:4]	DIA[16:9,7:0]	DIB[16:9,7:0]	DOA[16:9,7:0]	DOB[16:9,7:0]
2K*18	ADDRA[14:4]	ADDRB[14:4]	DIA[17:0]	DIB[17:0]	DOA[17:0]	DOB[17:0]
1K*32 (DP)	ADDRA[14:5]	ADDRB[14:5]	DIA[34:27,25:18,16:9,7:0]	DIB[34:27,25:18,16:9,7:0]	DOA[34:27,25:18,16:9,7:0]	DOB[34:27,25:18,16:9,7:0]
1K*36 (DP)	ADDRA[14:5]	ADDRB[14:5]	DIA[35:0]	DIB[35:0]	DOA[35:0]	DOB[35:0]
1K*32 (SP/SDP)	ADDRA[14:5]	ADDRB[14:5]	DIA[34:27,25:18,16:9,7:0]	N/A	N/A	DOB[34:27,25:18,16:9,7:0]
1K*36 (SP/SDP)	ADDRA[14:5]	ADDRB[14:5]	DIA[35:0]	N/A	N/A	DOB[35:0]
512*64	ADDRA[14:6]	ADDRB[14:6]	{DIB[34:27,25:18,16:9,7:0],DIA[34:27,25:18,16:9,7:0]}	N/A	N/A	{DOB[34:27,25:18,16:9,7:0],DOA[34:27,25:18,16:9,7:0]}
512*72	ADDRA[14:6]	ADDRB[14:6]	{DIB,DIA}	N/A	N/A	{DOB,DOA}

Table 9-2 Address and Data Port Mapping for 18K DRM Mode

DRM Port Mode	**Address Port Mapping**		**Data Port Mapping**			
	Port A Address	**Port B Address**	**Port A data input bus**	**Port B data input bus**	**Port A data output bus**	**Port B data output bus**
16K*1	ADDRA[13:0]	ADDRB[13:0]	DIA[0]	DIB[0]	DOA[X], where X is any bit between 16~9, 7~0	DOB[X], where X is any bit between 16~9, 7~0
8K*2	ADDRA[13:1]	ADDRB[13:1]	DIA[1:0]	DIB[1:0]	DOA[X+1:X], X=0,2,4,9,11,12,15	DOB[X+1:X], X=0,2,4,9,11,12, 15
4K*4	ADDRA[13:2]	ADDRB[13:2]	DIA[3:0]	DIB[3:0]	DOA[X+3:X], X=0,4,9,12	DOB[X+3:X], X=0,4,9,12
2K*8	ADDRA[13:3]	ADDRB[13:3]	DIA[7:0]	DIB[7:0]	DOA[7:0] or DOA[16:9]	DOB[7:0] or DOB[16:9]
2K*9	ADDRA[13:3]	ADDRB[13:3]	DIA[8:0]	DIB[8:0]	DOA[8:0] or DOA[17:9]	DOB[8:0] or DOB[17:9]
1K*16	ADDRA[13:4]	ADDRB[13:4]	DIA[16:9,7:0]	DIB[16:9,7:0]	DOA[16:9,7:0]	DOB[16:9,7:0]
1K*18	ADDRA[13:4]	ADDRB[13:4]	DIA[17:0]	DIB[17:0]	DOA[17:0]	DOB[17:0]
512*32	ADDRA[13:5]	ADDRB[13:5]	DIB[16:9,7:0], DIA[16:9,7:0]}	N/A	N/A	DOB[16:9,7:0], DOB[16:9,7:0]}
512*36	ADDRA[13:5]	ADDRB[13:5]	{DIB[17:0], DIA[17:0]}	N/A	N/A	{ DOB[17:0], DOA[17:0]}

Table 9-3 and Table 9-4 show the data address mappings for data widths of x1, x2, x4, x8, x16, x32 and x9, x18, x36, respectively.

Table 9-3 Data Address Mapping for Different Data Widths (x1, x2, x4, x8, x16, x32)

<table>
<tr><th>Data Width</th><th colspan="32">Lowest-bit port address (compared with the least-bit port address in the maximum data width mode)</th></tr>
<tr><td>32</td><td colspan="32">0</td></tr>
<tr><td>16</td><td colspan="16">1</td><td colspan="16">0</td></tr>
<tr><td>8</td><td colspan="8">3</td><td colspan="8">2</td><td colspan="8">1</td><td colspan="8">0</td></tr>
<tr><td>4</td><td colspan="4">7</td><td colspan="4">6</td><td colspan="4">5</td><td colspan="4">4</td><td colspan="4">3</td><td colspan="4">2</td><td colspan="4">1</td><td colspan="4">0</td></tr>
<tr><td>2</td><td colspan="2">15</td><td colspan="2">14</td><td colspan="2">13</td><td colspan="2">12</td><td colspan="2">11</td><td colspan="2">10</td><td colspan="2">9</td><td colspan="2">8</td><td colspan="2">7</td><td colspan="2">6</td><td colspan="2">5</td><td colspan="2">4</td><td colspan="2">3</td><td colspan="2">2</td><td colspan="2">1</td><td colspan="2">0</td></tr>
<tr><td>1</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
</table>

Table 9-4 Data Address Mapping for Different Data Widths (x9, x18, x36)

<table>
<tr><th>Data Width</th><th colspan="36">Lowest-bit port address (compared with the least-bit port address in the maximum data width mode)</th></tr>
<tr><td>36</td><td colspan="36">0</td></tr>
<tr><td>18</td><td colspan="18">1</td><td colspan="18">0</td></tr>
<tr><td>9</td><td colspan="9">3</td><td colspan="9">2</td><td colspan="9">1</td><td colspan="9">0</td></tr>
<tr><td>Index</td><td>35</td><td>34</td><td>33</td><td>32</td><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
</table>

9.2 Initialization Configuration Parameter Mapping

INIT_XX is the initialization configuration parameter for DRM to initialize DRM and determine the initial value of the memory. By default, the initial values of DRM are all set to 0. GTP_DRM18K_E1 includes 64 initialization configuration parameters from INIT_00 to INIT_3F, and GTP_DRM36K_E1 includes 128 initialization configuration parameters from INIT_00 to INIT_7F. Each configuration parameter is 288-bit, configuring the 288 bits of memory at the corresponding DRM address.

The following formulas and the table show the position of the memory bit mapped by each INIT_XX (YY in the formula is the decimal number converted from the hexadecimal number XX):

$$\text{Highest bit} = [(YY+1)*288)]-1$$

$$\text{Lowest bit} = (YY)*288$$

Table 9-5 INIT_XX Parameter Mapping

Parameter	**Highest Bit**	**Lowest Bit**
INIT_00	287	0
INIT_01	575	288
INIT_02	863	576
……	……	……
INIT_20	9503	9216
……	……	……
INIT_3F	18431	18144
……	……	……
INIT_7F	36575	36288

When the data width is 2^N, namely 1/2/4/8/16/32/64 bits, in the above table, every 9 bits of INIT_XX data only maps the lower 8 bits to memory; when the data width is 9×2^N, namely 9/18/36/72 bits, all INIT_XX data is mapped to memory. For example, when a DRM is configured as 8x4K, INIT_00[7:0] is the initial value of the data is read out when the address addr corresponding to 0, and INIT_00[16:9] is the initial value of the data is read out when the address addr corresponding to 1; when a DRM is configured as 9x4K, INIT_00[8:0] is the initial value of the data is read out when the address addr corresponding to INIT_00[8:0] is 0, and INIT_00[17:9] is the initial value of the data is read out when the address addr corresponding to 1.

For the application example of the initialization configuration parameter INIT_XX, refer to 6.7.1 Single 36K DRM Configuration.

9.3 DRM Port Signal Description

1. Address bus (ADDRA[15:0], ADDRB[15:0]): Ports A/B read/write address. Among them, ADDRA[15] and ADDRB[15] are hard cascade address extension signals, and should be connected to a high level when in non-hard cascade mode; the number of valid bits of the remaining address buses depends on the read and write data width. For example, if DRM36K is configured to a read/write data width of 32 bits, the address width is 10 bits, with valid address inputs as ADDRA[14:5] and ADDRB[14:5]. The remaining ADDRA[4:0] and ADDRB[4:0] should be connected to a low level. DRM18K does not support hard cascade and has no hard cascade address extension signal. Ports A/B read/write addresses of DRM18K are ADDRA[13:0] and ADDRB[13:0]. For example, if DRM18K is configured with a 32-bit read/write data width, the address width is 9 bits. The valid address inputs are ADDRA[13:4] and ADDRB[13:4], and the remaining ADDRA[3:0] and ADDRB[3:0] should be connected to a low level. For detailed address mapping, refer to 9.1 Address and Data Port Mapping.
2. Address hold signals (ADDRA_HOLD, ADDRB_HOLD): When this port is set to a high level, the corresponding port address input remains the previous value, and does not vary with the input address; when this port is set to a low level, the corresponding port address input does vary with input address. When the address hold is not in use, this port should be connected to a low level.
3. Data buses (DIA[35:0], DIB[35:0], DOA[35:0], DOB[35:0]): Read and write data ports for Ports A and B. For detailed data port mapping, refer to Appendix 9.1 Address and Data Port Mapping.
4. Address extension signals (CSA[2:0], CSB[2:0]): Multiple DRMs can be cascaded into a larger DP RAM, SDP RAM, SP RAM, ROM, or FIFO. For this, Ports A and B of the DRM respectively provide an additional 3-bit address extension, commonly used for deeply-extended applications. Refer to 3.9.3 Hard Cascade Configuration for Multiple 36K DRMs.
5. Write enable signals (WEA, WEB): Write operation control signals for DRM. For its detailed write operation timing, refer to 3.5 Read and Write Operations.
6. Byte enable signals (BWEA[7:0], BWEB[3:0]): Byte-enable write operation control signals of DRM. Each bit controls whether the data of one byte is written to the corresponding address when write enable is pulled high. Specially, in SDP mode, byte-enable write operation is controlled only by BWEA[7:0]. When byte-enable mode is not used, the byte enable signals should be connected to a high level.
7. Clock-related signals (CLKA, CLKB, CEA, CEB): Clock and enable signals for Ports A and B of the DRM. Under SDP mode, CLKA and CEA are write clocks and their enable signals, CLKB and CEB are read clocks and their enable signals.

8. Output register related signals (ORCEA, ORCEB, RSTA, RSTB): Enable and reset signals for the output registers of Ports A and B.
9. Cascade signals (CINA, CINB, COUTA, COUTB): Used for deep cascading in 64Kx1 mode, with the input/output cascaded to the data output of Port A/B of the adjacent DRM.
10. ECC mode related signals (INJECT_SBITERR, INJECT_DBITERR, ECC_SBITERR, ECC_DBITERR, ECC_PARITY[7:0], ECC_RDADDR[8:0]): These signals are respectively single-bit/dual-bit error injection signal, single-bit/dual-bit error flag signal, ECC encode checksum bits output, and ECC decode read address output.

9.4 Additional Information Bits for Bytes

In 18K mode with data widths of x9 and x18, and 36K mode with data widths of x9, x18, x36 and x72, there is one additional information storage bit per byte, as shown in Table 9-6:

Table 9-6 List of Additional Information Bits for Bytes

Port	**Mode**	**Data Input**		**Data Input**	
		Byte	**Additional Information Bit**	**Byte**	**Additional Information Bit**
A	18K, 36K	DIA[7:0]	DIA[8]	dia[7:0]	dia[8]
	18K, 36K	DIA[16:9]	DIA[17]	dia[16:9]	dia[17]
	36K	DIA[25:18]	DIA[26]	dia[25:18]	dia[26]
	36K	DIA[34:27]	DIA[35]	dia[34:27]	dia[35]
B	18K, 36K	DIB[7:0]	DIB[8]	dib[7:0]	dib[8]
	18K, 36K	DIB[16:9]	DIB[17]	dib[16:9]	dib[17]
	36K	DIB[25:18]	DIB[26]	dib[25:18]	dib[26]
	36K	DIB[34:27]	DIB[35]	dib[34:27]	dib[35]

The above additional information bits must be left unused or connected to a low level in x1, x2, x4, x8, x16, x32, and x64 modes.

9.5 GTP_DRM36K_E1 Instantiation Template

```
GTP_DRM36K_E1 #(
.GRS_EN("TRUE"),
.CSA_MASK(3'b000),
.CSB_MASK(3'b000),
.DATA_WIDTH_A(18),
.DATA_WIDTH_B(18),
.WRITE_MODE_A("NORMAL_WRITE"),
.WRITE_MODE_B("NORMAL_WRITE"),
.DOA_REG(0),
.DOB_REG(0),
.DOA_REG_CLKINV(0),
.DOB_REG_CLKINV(0),
.RSTA_VAL(36'b0),
.RSTB_VAL(36'b0),
.RST_TYPE("SYNC"),
.RAM_MODE("TRUE_DUAL_PORT"),
.RAM_CASCADE("NONE"),
.ECC_READ_EN("FALSE"),
.ECC_WRITE_EN("FALSE"),
.INIT_00(288'b0),
.INIT_01(288'b0),
.INIT_02(288'b0),
.INIT_03(288'b0),
.INIT_04(288'b0),
.INIT_05(288'b0),
.INIT_06(288'b0),
.INIT_07(288'b0),
.INIT_08(288'b0),
.INIT_09(288'b0),
.INIT_0A(288'b0),
.INIT_0B(288'b0),
.INIT_0C(288'b0),
.INIT_0D(288'b0),
.INIT_0E(288'b0),
.INIT_0F(288'b0),
.INIT_10(288'b0),
.INIT_11(288'b0),
.INIT_12(288'b0),
.INIT_13(288'b0),
.INIT_14(288'b0),
.INIT_15(288'b0),
.INIT_16(288'b0),
.INIT_17(288'b0),
.INIT_18(288'b0),
.INIT_19(288'b0),
.INIT_1A(288'b0),
.INIT_1B(288'b0),
.INIT_1C(288'b0),
.INIT_1D(288'b0),
.INIT_1E(288'b0),
.INIT_1F(288'b0),
.INIT_20(288'b0),
.INIT_21(288'b0),
.INIT_22(288'b0),
.INIT_23(288'b0),
.INIT_24(288'b0),
.INIT_25(288'b0),
.INIT_26(288'b0),
.INIT_27(288'b0),
.INIT_28(288'b0),
.INIT_29(288'b0),
.INIT_2A(288'b0),
.INIT_2B(288'b0),
.INIT_2C(288'b0),
.INIT_2D(288'b0),
.INIT_2E(288'b0),
.INIT_2F(288'b0),
.INIT_30(288'b0),
.INIT_31(288'b0),
.INIT_32(288'b0),
.INIT_33(288'b0),
.INIT_34(288'b0),
.INIT_35(288'b0),
.INIT_36(288'b0),
.INIT_37(288'b0),
.INIT_38(288'b0),
.INIT_39(288'b0),
.INIT_3A(288'b0),
.INIT_3B(288'b0),
.INIT_3C(288'b0),
.INIT_3D(288'b0),
.INIT_3E(288'b0),
.INIT_3F(288'b0),
.INIT_40(288'b0),
.INIT_41(288'b0),
.INIT_42(288'b0),
.INIT_43(288'b0),
.INIT_44(288'b0),
.INIT_45(288'b0),
.INIT_46(288'b0),
.INIT_47(288'b0),
.INIT_48(288'b0),
.INIT_49(288'b0),
.INIT_4A(288'b0),
.INIT_4B(288'b0),
.INIT_4C(288'b0),
.INIT_4D(288'b0),
.INIT_4E(288'b0),
.INIT_4F(288'b0),
.INIT_50(288'b0),
.INIT_51(288'b0),
.INIT_52(288'b0),
.INIT_53(288'b0),
.INIT_54(288'b0),
.INIT_55(288'b0),
.INIT_56(288'b0),
.INIT_57(288'b0),
.INIT_58(288'b0),
.INIT_59(288'b0),
.INIT_5A(288'b0),
.INIT_5B(288'b0),
.INIT_5C(288'b0),
.INIT_5D(288'b0),
.INIT_5E(288'b0),
.INIT_5F(288'b0),
.INIT_60(288'b0),
.INIT_61(288'b0),
.INIT_62(288'b0),
.INIT_63(288'b0),
.INIT_64(288'b0),
.INIT_65(288'b0),
.INIT_66(288'b0),
.INIT_67(288'b0),
.INIT_68(288'b0),
.INIT_69(288'b0),
.INIT_6A(288'b0),
.INIT_6B(288'b0),
.INIT_6C(288'b0),
.INIT_6D(288'b0),
.INIT_6E(288'b0),
.INIT_6F(288'b0),
.INIT_70(288'b0),
.INIT_71(288'b0),
.INIT_72(288'b0),
.INIT_73(288'b0),
.INIT_74(288'b0),
.INIT_75(288'b0),
.INIT_76(288'b0),
.INIT_77(288'b0),
.INIT_78(288'b0),
.INIT_79(288'b0),
.INIT_7A(288'b0),
.INIT_7B(288'b0),
.INIT_7C(288'b0),
.INIT_7D(288'b0),
.INIT_7E(288'b0),
.INIT_7F(288'b0),
.INIT_FILE("NONE"),
.BLOCK_X(0),
.BLOCK_Y(0),
.RAM_DATA_WIDTH(9),
.RAM_ADDR_WIDTH(12),
.INIT_FORMAT("BIN")
) GTP_DRM36K_E1_inst (
.DOA               (doa               ),
.DOB               (dob               ),
.ECC_PARITY        (ecc_parity        ),
.ECC_RDADDR        (ecc_rdaddr        ),
.ADDRA             (addra             ),
.ADDRB             (addrb             ),
.BWEA              (bwea              ),
.BWEB              (bweb              ),
.CSA               (csa               ),
.CSB               (csb               ),
.DIA               (dia               ),
.DIB               (dib               ),
.COUTA             (couta             ),
.COUTB             (coutb             ),
.ECC_DBITERR       (ecc_dbiterr       ),
.ECC_SBITERR       (ecc_sbiterr       ),
.ADDRA_HOLD        (addra_hold        ),
.ADDRB_HOLD        (addrb_hold        ),
.CEA               (cea               ),
.CEB               (ceb               ),
.CINA              (cina              ),
.CINB              (cinb              ),
.CLKA                 (clka                ),
.CLKB                 (clkb                ),
.INJECT_DBITERR (inject_dbiterr),
.INJECT_SBITERR (inject_sbiterr),
.ORCEA                (orcea               ),
.ORCEB                (orceb               ),
.RSTA                 (rsta                ),
.RSTB                 (rstb                ),
.WEA                  (wea                 ),
.WEB                  (web                 )
);
```

9.6 GTP_DRM18K_E1 Instantiation Template

```
GTP_DRM18K_E1 #(
.GRS_EN("TRUE"),
.DATA_WIDTH_A(18),
.DATA_WIDTH_B(18),
.DOA_REG(0),
.DOB_REG(0),
.DOA_REG_CLKINV(0),
.DOB_REG_CLKINV(0),
.RSTA_VAL(18'b0),
.RSTB_VAL(18'b0),
.RST_TYPE("SYNC"),
.RAM_MODE("TRUE_DUAL_PORT"),
.WRITE_MODE_A("NORMAL_WRITE"),
.WRITE_MODE_B("NORMAL_WRITE"),
.INIT_00(288'b0),
.INIT_01(288'b0),
.INIT_02(288'b0),
.INIT_03(288'b0),
.INIT_04(288'b0),
.INIT_05(288'b0),
.INIT_06(288'b0),
.INIT_07(288'b0),
.INIT_08(288'b0),
.INIT_09(288'b0),
.INIT_0A(288'b0),
.INIT_0B(288'b0),
.INIT_0C(288'b0),
.INIT_0D(288'b0),
.INIT_0E(288'b0),
.INIT_0F(288'b0),
.INIT_10(288'b0),
.INIT_11(288'b0),
.INIT_12(288'b0),
.INIT_13(288'b0),
.INIT_14(288'b0),
.INIT_15(288'b0),
.INIT_16(288'b0),
.INIT_17(288'b0),
.INIT_18(288'b0),
.INIT_19(288'b0),
.INIT_1A(288'b0),
.INIT_1B(288'b0),
.INIT_1C(288'b0),
.INIT_1D(288'b0),
.INIT_1E(288'b0),
.INIT_1F(288'b0),
.INIT_20(288'b0),
.INIT_21(288'b0),
.INIT_22(288'b0),
.INIT_23(288'b0),
.INIT_24(288'b0),
.INIT_25(288'b0),
.INIT_26(288'b0),
.INIT_27(288'b0),
.INIT_28(288'b0),
.INIT_29(288'b0),
.INIT_2A(288'b0),
.INIT_2B(288'b0),
.INIT_2C(288'b0),
.INIT_2D(288'b0),
.INIT_2E(288'b0),
.INIT_2F(288'b0),
.INIT_30(288'b0),
.INIT_31(288'b0),
.INIT_32(288'b0),
.INIT_33(288'b0),
.INIT_34(288'b0),
.INIT_35(288'b0),
.INIT_36(288'b0),
.INIT_37(288'b0),
.INIT_38(288'b0),
.INIT_39(288'b0),
.INIT_3A(288'b0),
.INIT_3B(288'b0),
.INIT_3C(288'b0),
.INIT_3D(288'b0),
.INIT_3E(288'b0),
.INIT_3F(288'b0),
.INIT_FILE("NONE"),
.BLOCK_X(0),
.BLOCK_Y(0),
.RAM_DATA_WIDTH(9),
.RAM_ADDR_WIDTH(11),
.INIT_FORMAT("BIN")
) GTP_DRM18K_E1_inst (
.DOA        (doa        ),
.DOB        (dob        ),
.ADDRA      (addra      ),
.ADDRB      (addrb      ),
.BWEA          (bwea        ),
.BWEB          (bweb        ),
.DIA           (dia         ),
.DIB           (dib         ),
.ADDRA_HOLD (addra_hold),
.ADDRB_HOLD (addrb_hold),
.CEA           (cea         ),
.CEB           (ceb         ),
.CLKA          (clka        ),
.CLKB          (clkb        ),
.ORCEA         (orcea       ),
.ORCEB         (orceb       ),
.RSTA          (rsta        ),
.RSTB          (rstb        ),
.WEA           (wea         ),
.WEB           (web         )
);
```

9.7 GTP_FIFO36K_E1 Instantiation Template

```
GTP_FIFO36K_E1 #(
.GRS_EN("TRUE"),
.DATA_WIDTH(18),
.DO_REG(0),
.ECC_READ_EN("FALSE"),
.ECC_WRITE_EN("FALSE"),
.ALMOST_FULL_OFFSET('b0),
.ALMOST_EMPTY_OFFSET('b0),
.RST_VAL('b0),
.USE_EMPTY(0),
.USE_FULL(0),
.SYNC_FIFO("FALSE")
) GTP_FIFO36K_E1_inst (
.DO                    (do                  ),
.DI                    (di                  ),
.ALMOST_EMPTY     (almost_empty    ),
.ALMOST_FULL      (almost_full     ),
.ECC_DBITERR      (ecc_dbiterr     ),
.ECC_SBITERR      (ecc_sbiterr     ),
.EMPTY            (empty           ),
.FULL             (full            ),
.INJECT_DBITERR (inject_dbiterr),
.INJECT_SBITERR (inject_sbiterr),
.ORCE             (orce            ),
.RCE              (rce             ),
.RCLK             (rclk            ),
.RST              (rst             ),
.WCE              (wce             ),
.WCLK             (wclk            )
);
```

9.8 GTP_FIFO18K_E1 Instantiation Template

```
GTP_FIFO18K_E1 #(
.GRS_EN("TRUE"),
.DATA_WIDTH(18),
.DO_REG(0),
.ALMOST_FULL_OFFSET('b0),
.ALMOST_EMPTY_OFFSET('b0),
.RST_VAL('b0),
.USE_EMPTY(0),
.USE_FULL(0),
.SYNC_FIFO("FALSE")
) GTP_FIFO18K_E1_inst (
.DO               (do            ),
.DI               (di            ),
.ALMOST_EMPTY     (almost_empty),
.ALMOST_FULL      (almost_full ),
.EMPTY            (empty         ),
.FULL             (full          ),
.ORCE                  (orce          ),
.RCE                   (rce           ),
.RCLK                  (rclk          ),
.RST                   (rst           ),
.WCE                   (wce           ),
.WCLK                  (wclk          )
);
```

Disclaimer

Copyright Notice

Disclaimer

1. This document only provides information in stages and may be updated at any time based on the actual situation of the products without further notice. The Company assumes no legal responsibility for any direct or indirect losses caused by improper use of this document.
2. This document is provided "as is" without any warranties, including but not limited to warranties of merchantability, fitness for a particular purpose, non-infringement, or any other warranties mentioned in proposals, specifications, or samples. This document does not grant any explicit or implied intellectual property usage license, whether by estoppel or otherwise.
3. The Company reserves the right to modify any documents related to its family products at any time without prior notice.