Arora V BSRAM & SSRAM

User Guide

UG300-1.3.2E, 02/02/2024

Revision History

Date	Version	Description
04/20/2023	1.0E	Initial version published.
05/25/2023	1.1E	● Read-before-Write in dual port mode and single port mode is not supported. ● A note added in the functional description of dual port mode, semi dual port mode, and Semi Dual Port Mode with ECC Function respectively.
06/30/2023	1.2E	138K devices do not support Read-before-Write in dual port mode and single port mode.
09/08/2023	1.3E	Figure 5-1 RAM16S1 Timing Diagram and Figure 5-5 RAM16SDP1 Timing Diagram updated.
12/12/2023	1.3.1E	The Read-before-Write mode description added: Arora V 138K devices do not support Read-before-Write mode.
02/02/2024	1.3.2E	● Note added to Table 5-1 SSRAM Modes, adding information on devices that do not support some SSRAM primitives. ● Description of Read-before-Write mode added: Arora V 25K devices do not support Read-before-Write in dual port mode.

Contents

Contents i
List of Figures iii
List of Tables v
1 About This Guide 1
1.1 Purpose 1
1.2 Related Documents 1
1.3 Terminology and Abbreviations 1
1.4 Support and Feedback 2
2 Overview 3
2.1 BSRAM Features 3
2.2 BSRAM Configuration Mode 4
3 BSRAM Primitive 6
3.1 Dual Port Mode 6
3.2 Single Port Mode 18
3.3 Semi Dual Port Mode 24
3.4 Semi Dual Port Mode with ECC Function 30
3.5 Read Only Mode 37
4 BSRAM Output Reset 42
5 SSRAM Primitive 45
5.1 RAM16S1 45
5.2 RAM16S2 48
5.3 RAM16S4 50
5.4 RAM16SDP1 52
5.5 RAM16SDP2 55
5.6 RAM16SDP4 57
5.7 ROM16 59
6 IP Generation 62
6.1 BSRAM Dual Port Mode 62
6.2 BSRAM Semi Dual Port Mode with ECC Function 65
6.3 SSRAM Single Port Mode 67

7 Initialization File 69
7.1 Bin File 69
7.2 Hex File 69
7.3 Address-Hex File 70

List of Figures

Figure 3-1 Timing Diagram of DPB/DPX9B Normal Mode (Bypass Mode)...................................... 7
Figure 3-2 Timing Diagram of DPB/DPX9B Normal Mode (Pipeline Mode)..................................... 8
Figure 3-3 Timing Diagram of DPB/DPX9B Write-Through Mode (Bypass Mode) 9
Figure 3-4 Timing Diagram of DPB/DPX9B Write-Through Mode (Pipeline Mode) 10
Figure 3-5 Timing Diagram of DPB/DPX9B Read-before-Write Mode (Bypass Mode)..................... 11
Figure 3-6 Timing Diagram of DPB/DPX9B Read-before-Write Mode (Pipeline Mode).................... 12
Figure 3-7 DPB/DPX9B Port Diagram 13
Figrue 3-8 SP/SPX9 Port Diagram 19
Figure 3-9 Timing Diagram of Semi Dual Port BSRAM Normal Mode (Bypass Mode)..................... 24
Figure 3-10 Timing Diagram of Semi Dual Port BSRAM Normal Mode (Pipeline Mode).................. 25
Figrue 3-11 SDPB/SDPX9B Port Diagram 26
Figure 3-12 SDP36KE Port Diagram 31
Figure 3-13 ROM Timing Diagram (Bypass Mode)... 37
Figure 3-14 ROM Timing Diagram (Pipeline Mode).. 38
Figrue 3-15 pROM/ pROMX9 Port Diagram 38
Figure 4-1 Output Reset Structure Diagram 42
Figure 4-2 Synchronous Reset Timing Diagram (Pipeline Mode) 43
Figure 4-3 Synchronous Reset Timing Diagram (Bypass Mode) 43
Figure 4-4 Asynchronous Reset Timing Diagram (Pipeline Mode).. 43
Figure 4-5 Asynchronous Reset Timing Diagram (Bypass Mode).. 44
Figure 5-1 RAM16S1 Timing Diagram 46
Figure 5-2 RAM16S1 Port Diagram 46
Figure 5-3 RAM16S2 Port Diagram 48
Figure 5-4 RAM16S4 Port Diagram 50
Figure 5-5 RAM16SDP1 Timing Diagram 53
Figure 5-6 RAM16SDP1 Port Diagram 53
Figure 5-7 RAM16SDP2 Port Diagram 55
Figure 5-8 RAM16SDP4 Port Diagram 57
Figure 5-9 ROM16 Timing Diagram 59
Figure 5-10 ROM16 Port Diagram 60
Figure 6-1 IP Customization of DPB 63
Figure 6-2 IP Customization of SDP36KE 65

Figure 6-3 IP Customization of RAM16S 67

List of Tables

Table 1-1 Terminology and Abbreviations 1
Table 2-1 BSRAM Configuration Mode 4
Table 2-2 BSRAM Data Width and Address Width 4
Table 2-3 Data Width Configuration in Dual Port Mode 5
Table 2-4 Data Width Configuration in Semi Dual Port Mode 5
Table 2-5 Data Width Configuration in Semi Dual Port Mode with ECC Function 5
Table 3-1 DPB/DPX9B Data Width and Address Width Configuration Relationship 12
Table 3-2 DPB/DPX9B Port Description 13
Table 3-3 DPB/DPX9B Parameter Description 14
Table 3-4 SP/SPX9 Data Width and Address Width Configuration Relationship 19
Table 3-5 SP/ SPX9 Port Description 19
Table 3-6 SP/SPX9 Parameter Description 20
Table 3-7 SDPB/SDPX9B Data Width and Address Width Configuration 25
Table 3-8 SDPB/SDPX9B Port Description 26
Table 3-9 SDPB/ SDPX9B Parameter Description 27
Table 3-10 SDP36KE Device Supported 30
Table 3-11 SDP36KE Data Width and Address Width Configuration Relationship 31
Table 3-12 SDP36KE Port Description 31
Table 3-13 SDP36KE Parameter Description 32
Table 3-14 pROM/ pROMX9 Data Width and Address Width Configuration Relationship 38
Table 3-15 pROM/ pROMX9 Port Description 39
Table 3-16 pROM/pROMX9 Parameter Description 39
Table 5-1 SSRAM Modes 45
Table 5-2 RAM16S1 Port Description 46
Table 5-3 RAM16S1 Parameter Description 46
Table 5-4 RAM16S2 Port Description 48
Table 5-5 RAM16S2 Parameter Description 49
Table 5-6 RAM16S4 Port Description 51
Table 5-7 RAM16S4 Parameter Description 51
Table 5-8 RAM16SDP1 Port Description 53
Table 5-9 RAM16SDP1 Parameter Description 54
Table 5-10 RAM16SDP2 Port Description 55

Table 5-11 RAM16SDP2 Parameter Description 56
Table 5-12 RAM16SDP4 Port Description 58
Table 5-13 RAM16SDP4 Parameter Description 58
Table 5-14 ROM16 Port Description 60
Table 5-15 ROM16 Parameter Description 60

1 About This Guide

1.1 Purpose

Arora Ⅴ BSRAM & SSRAM User Guide mainly describes the features, operating modes, primitive introduction, and IP generation of GOWINSEMI Arora Ⅴ BSRAM and SSRAM, aiming to provide the application instructions for you.

1.2 Related Documents

The latest user guides are available on the GOWINSEMI Website. You can find the related documents at www.gowinsemi.com:

- DS981, GW5AT series of FPGA Products Data Sheet
- DS1103, GW5A series of FPGA Products Data Sheet
- DS1104, GW5AST series of FPGA Products Data Sheet
- SUG100, GOWIN FPGA Designer User Guide

1.3 Terminology and Abbreviations

The terminology and abbreviations used in this manual are as shown in Table 1-1.

Table 1-1 Terminology and Abbreviations

Terminology and Abbreviations	Meaning
BSRAM	Block Static Random Access Memory
CFU	Configurable Function Unit
DP	Dual Port 16K Block SRAM
ECC	Error Checking and Correction
ROM	Read Only Memory
SDP	Semi Dual Port 16K Block SRAM
SDP36KE	Semi Dual Port 36K Block SRAM with ECC function
SP	Single Port 16K Block SRAM
SSRAM	Shadow Static Random Access Memory

1.4 Support and Feedback

Gowin Semiconductor provides customers with comprehensive technical support. If you have any questions, comments, or suggestions, please feel free to contact us directly by the following ways.

Website: www.gowinsemi.com

E-mail: support@gowinsemi.com

2 Overview

GOWINSEMI Arora V FPGA products provide abundant memory resources, including Block Static Random Access Memory (BSRAM) and Shadow Static Random Access Memory (SSRAM).

Each BSRAM can be configured up to 36 Kbits, with configurable data bit width and address depth. Each BSRAM has two ports: Port A and Port B. Each port has independent clock, address, data and control signals for independent read and write operations. And the two ports share the same memory space.

The Configurable Function Unit (CFU) is the basic unit that forms the core of GOWINSEMI Arora V FPGA products. It can be configured as SSRAM, including 16 x 4 bits static random memory (SRAM) or read-only memory (ROM16), depending on the applications.

2.1 BSRAM Features

- The maximum capacity of a BSRAM is 18 Kbits
- The clock frequency is up to 380 MHz (230 MHz in Read-before-Write[1] mode)
- Supports Single Port mode (SP)
- Supports Dual Port mode (DP)
- Supports Semi Dual Port mode (SDP)
- Supports Semi Dual Port mode with ECC function (SDP36KE)
- Supports Read Only mode (ROM)
- Supports up to 72-bit data width
- Dual Port and Semi Dual Port support independent clocks and independent data width
- Read mode supports Register Output and Bypass Output
- Write mode supports Normal mode, Read-before-Write[1] mode, and write-through mode

Note!

- [1] Arora V 138K devices do not support Read-before-Write mode.

- [1] Arora V 25K devices do not support Read-before-Write in dual port mode.

2.2 BSRAM Configuration Mode

In addition to SDP36KE, each BSRAM can be configured to 16 Kbits and 18 Kbits. SDP36KE supports 36 Kbits. Address depth and data width and of the five modes can be configured as shown in Table 2-1.

Table 2-1 BSRAM Configuration Mode

Capacity	Single Port Mode	Dual Port Mode	Semi Dual Port Mode	Semi Dual Port Mode with ECC function	Read Only Mode
16 Kbits	16K x 1	16K x 1	16K x 1	-	16K x 1
	8K x 2	8K x 2	8K x 2	-	8K x 2
	4K x 4	4K x 4	4K x 4	-	4K x 4
	2K x 8	2K x 8	2K x 8	-	2K x 8
	1K x 16	1K x 16	1K x 16	-	1K x 16
	512 x 32	-	512 x 32	-	512 x 32
18 Kbits	2K x 9	2K x 9	2K x 9	-	2K x 9
	1K x 18	1K x 18	1K x 18	-	1K x 18
	512 x 36	-	512 x 36	-	512 x 36
36 Kbits	-	-	-	512 x 72	-

In addition to SDP36KE, the data width of the address line for each BSRAM is 14bit, that is AD[13:0], and the maximum address depth is 16,384. The data width of the address line for SDP36KE is 9 bit, that is AD[8:0], and the maximum address depth is 512. Different data widths use different address line, as shown in Table 2-2.

Table 2-2 BSRAM Data Width and Address Width

Capacity	Configuration Mode	Data Width	Address Depth	Address Width
16 Kbits	16K x 1	[0:0]	16,384	[13:0]
	8K x 2	[1:0]	8,192	[13:1]
	4K x 4	[3:0]	4,096	[13:2]
	2K x 8	[7:0]	2,048	[13:3]
	1K x 16	[15:0]	1,024	[13:4]
	512 x 32	[31:0]	512	[13:5]
18 Kbits	2K x 9	[8:0]	2,048	[13:3]
	1K x 18	[17:0]	1,024	[13:4]
	512 x 36	[35:0]	512	[13:5]
36 Kbits	512 x 72	[71:0]	512	[8:0]

Dual Port mode, Semi Dual Port mode, and Semi Dual Port mode with ECC function support independent read/write clocks and independent read/write data width. In Dual Port mode, the data widths supported by

Port A and Port B are as shown in Table 2-3. In Semi Dual Port mode, the data widths supported by Port A and Port B are as shown in Table 2-4. In Semi Dual Port mode with ECC function, the data widths supported by Port A and Port B are as shown in Table 2-5.

Table 2-3 Data Width Configuration in Dual Port Mode

Capacity	Port B	Port A						
		16K x 1	8K x 2	4K x 4	2K x 8	1K x 16	2K x 9	1K x 18
16 Kbits	16K x 1	Yes	Yes	Yes	Yes	Yes	N/A	N/A
	8K x 2	Yes	Yes	Yes	Yes	Yes	N/A	N/A
	4K x 4	Yes	Yes	Yes	Yes	Yes	N/A	N/A
	2K x 8	Yes	Yes	Yes	Yes	Yes	N/A	N/A
	1K x 16	Yes	Yes	Yes	Yes	Yes	N/A	N/A
18 Kbits	2K x 9	N/A	N/A	N/A	N/A	N/A	Yes	Yes
	1K x 18	N/A	N/A	N/A	N/A	N/A	Yes	Yes

Table 2-4 Data Width Configuration in Semi Dual Port Mode

Capacity	Port B	Port A										
		16K x 1	8K x 2	4K x 4	2K x 8	1K x 16	512 x32	2K x 9	1K x18	512 x 36	1K x 36	512 x 72
16 Kbits	16K x 1	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A	N/A	N/A
	8K x 2	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A	N/A	N/A
	4K x 4	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A	N/A	N/A
	2K x 8	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A	N/A	N/A
	1K x 16	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A	N/A	N/A
	512 x 32	Yes	Yes	Yes	Yes	Yes	Yes	N/A	N/A	N/A	N/A	N/A
18 Kbits	2K x 9	N/A	N/A	N/A	N/A	N/A	N/A	Yes	Yes	Yes	N/A	N/A
	1K x 18	N/A	N/A	N/A	N/A	N/A	N/A	Yes	Yes	Yes	N/A	N/A

Table 2-5 Data Width Configuration in Semi Dual Port Mode with ECC Function

Capacity	Port B	Port A
		512 x 72
36 Kbits	512 x 72	Yes

3 BSRAM Primitive

Block SRAM is a block static random access memory. The software model based on the BSRAM characteristics includes Single Port mode (SP/SPX9), Dual Port mode (DPB/DPX9B), Semi Dual Port mode (SDPB/SDPX9B), Semi Dual Port mode with ECC function (SDP36KE), and Read Only mode (pROM/pROMX9).

3.1 Dual Port Mode

Primitive Introduction

DPB/DPX9B (True Dual Port 16K Block SRAM/True Dual Port 18K Block SRAM), 16K/18K dual port BSRAM.

Functional Description

DPB/DPX9B works in Dual Port mode with 16 Kbits/18 Kbits memory capacity. Port A and Port B can read/write independently[1]. DPB/DPX9B supports two read modes (bypass mode and pipeline mode) and three write modes (normal mode, write-through mode and Read-before-Write mode[2]).

Note!

- [1] Performing read and write operations to the same address at the same time is not recommended.
- [2] 138K devices do not support Read-before-Write mode.
- [2] Arora V 25K devices do not support Read-before-Write in dual port mode.

- Read mode:
  READ_MODE0 and READ_MODE1 enable or disable the output pipeline register at A port and B port. When the output pipeline register is used, the read operation needs extra clock cycle.
- Write mode:
  WRITE_MODE0 and WRITE_MODE1 configure write mode at Port A and Port B including normal mode, write-through mode and Read-before-Write mode. The corresponding internal timing diagrams of different modes are shown from Figure 3-1 to Figure 3-6.

Figure 3-1 Timing Diagram of DPB/DPX9B Normal Mode (Bypass Mode)

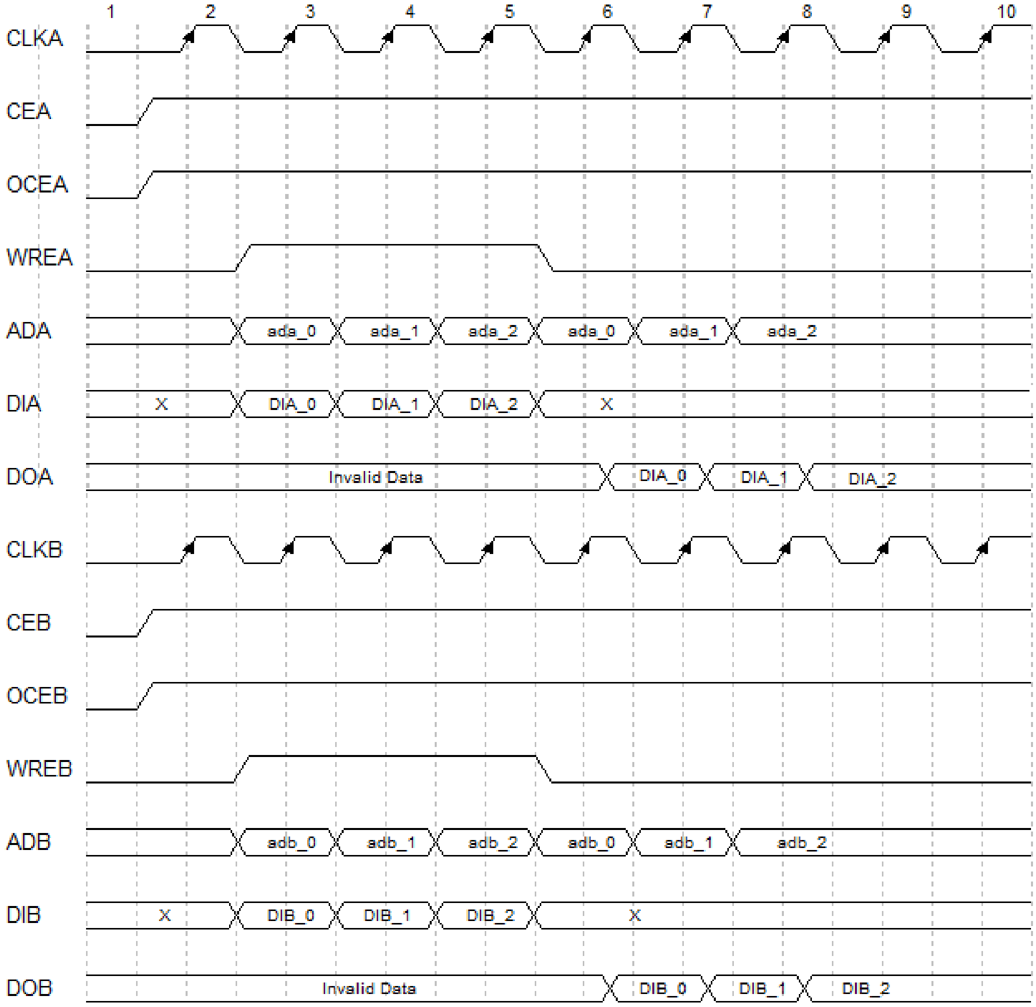


Figure 3-2 Timing Diagram of DPB/DPX9B Normal Mode (Pipeline Mode)

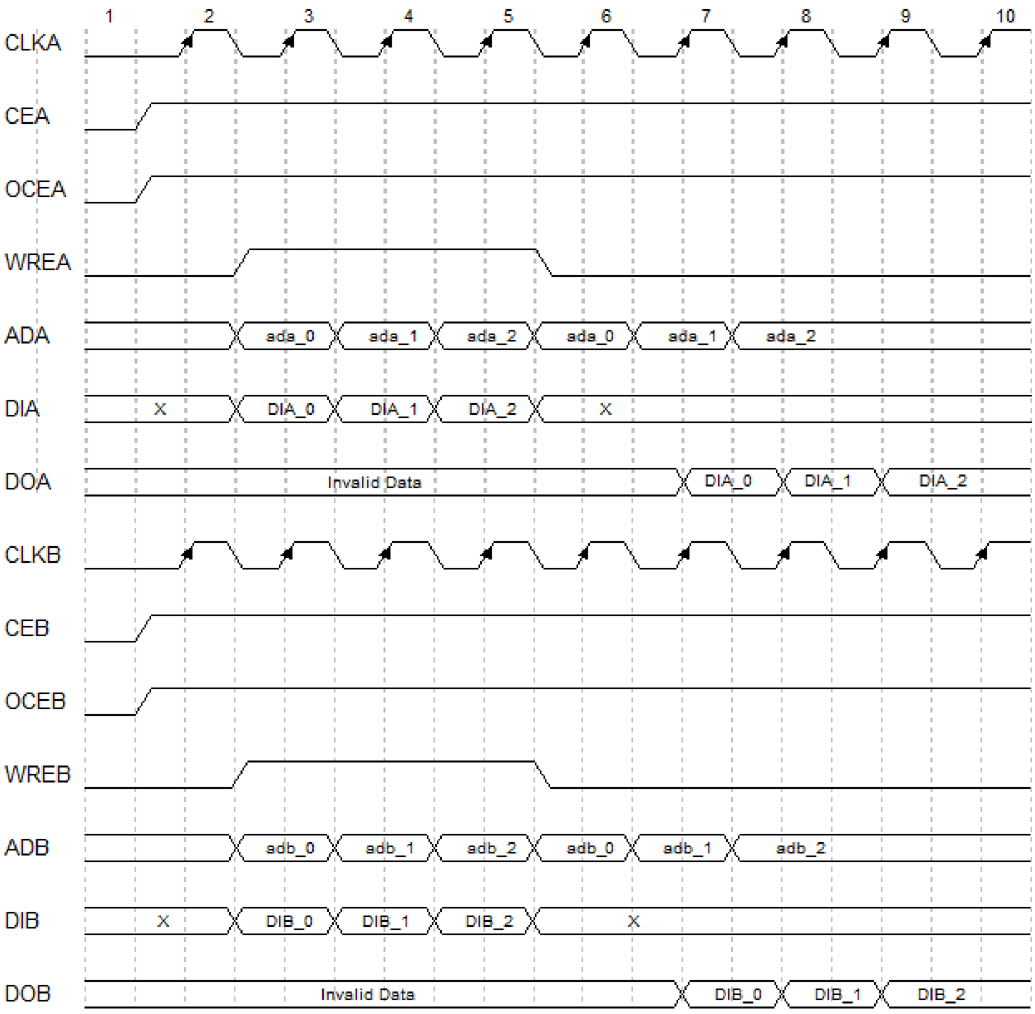


Figure 3-3 Timing Diagram of DPB/DPX9B Write-Through Mode (Bypass Mode)

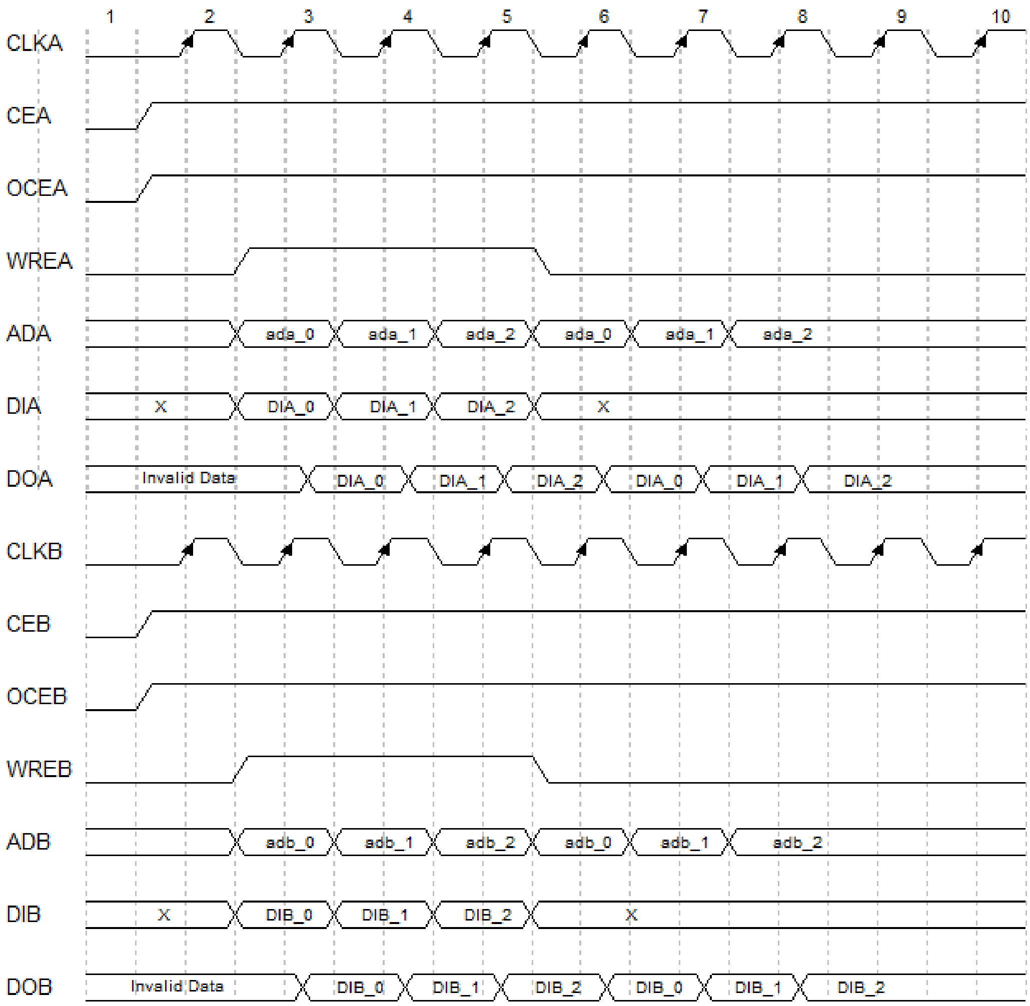


Figure 3-4 Timing Diagram of DPB/DPX9B Write-Through Mode (Pipeline Mode)

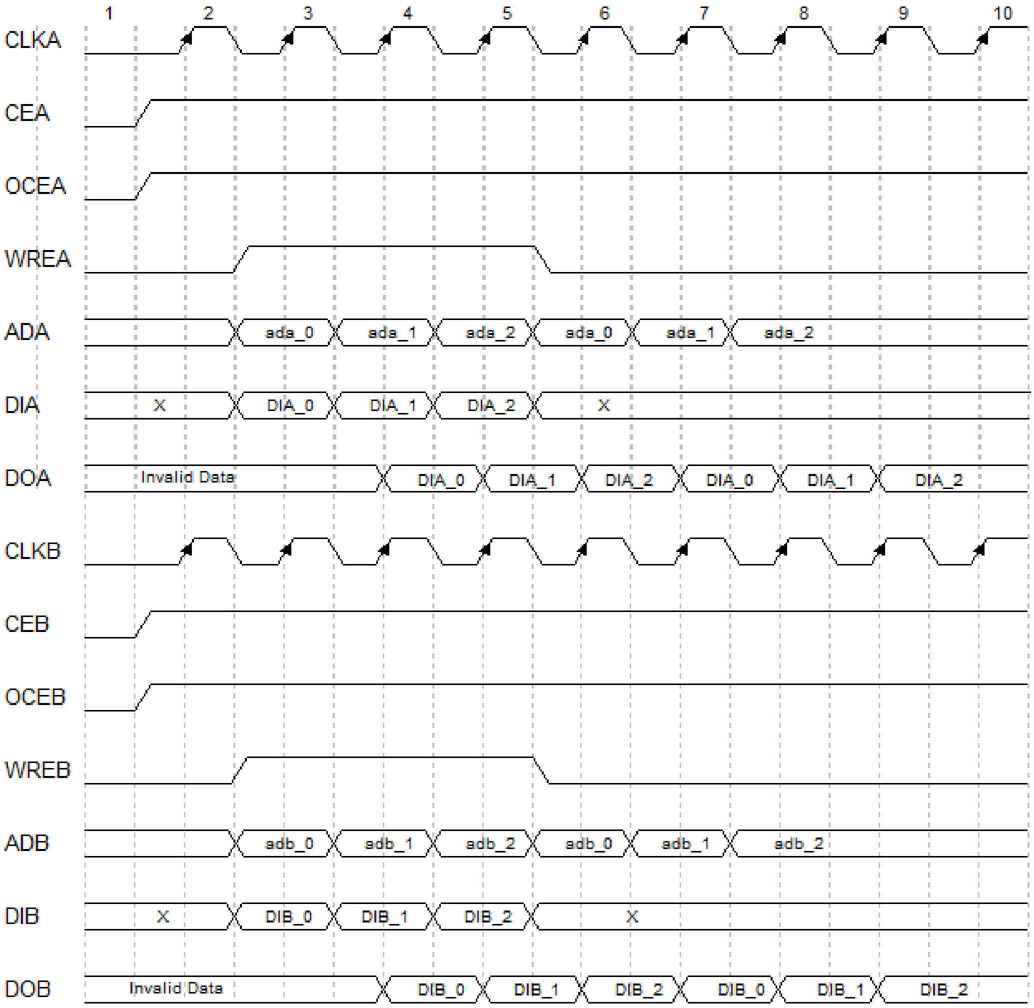


Figure 3-5 Timing Diagram of DPB/DPX9B Read-before-Write Mode (Bypass Mode)

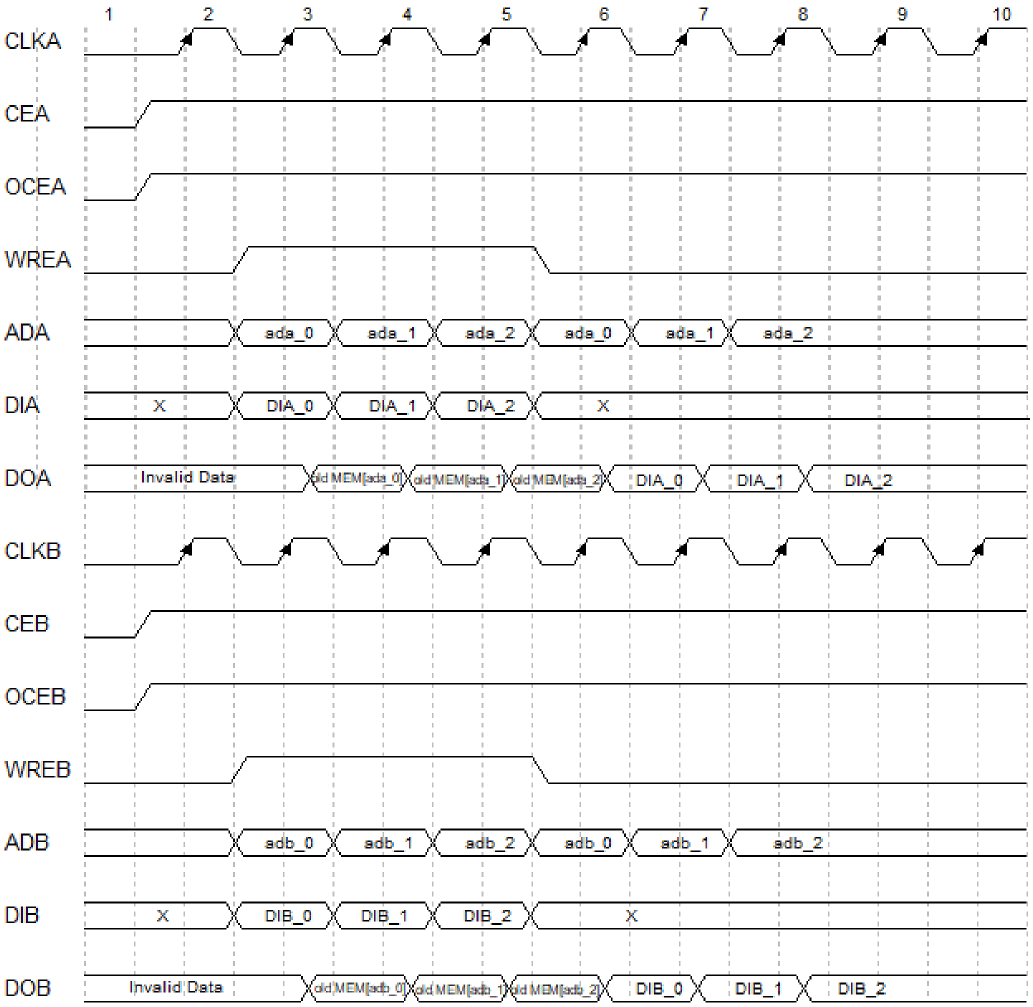
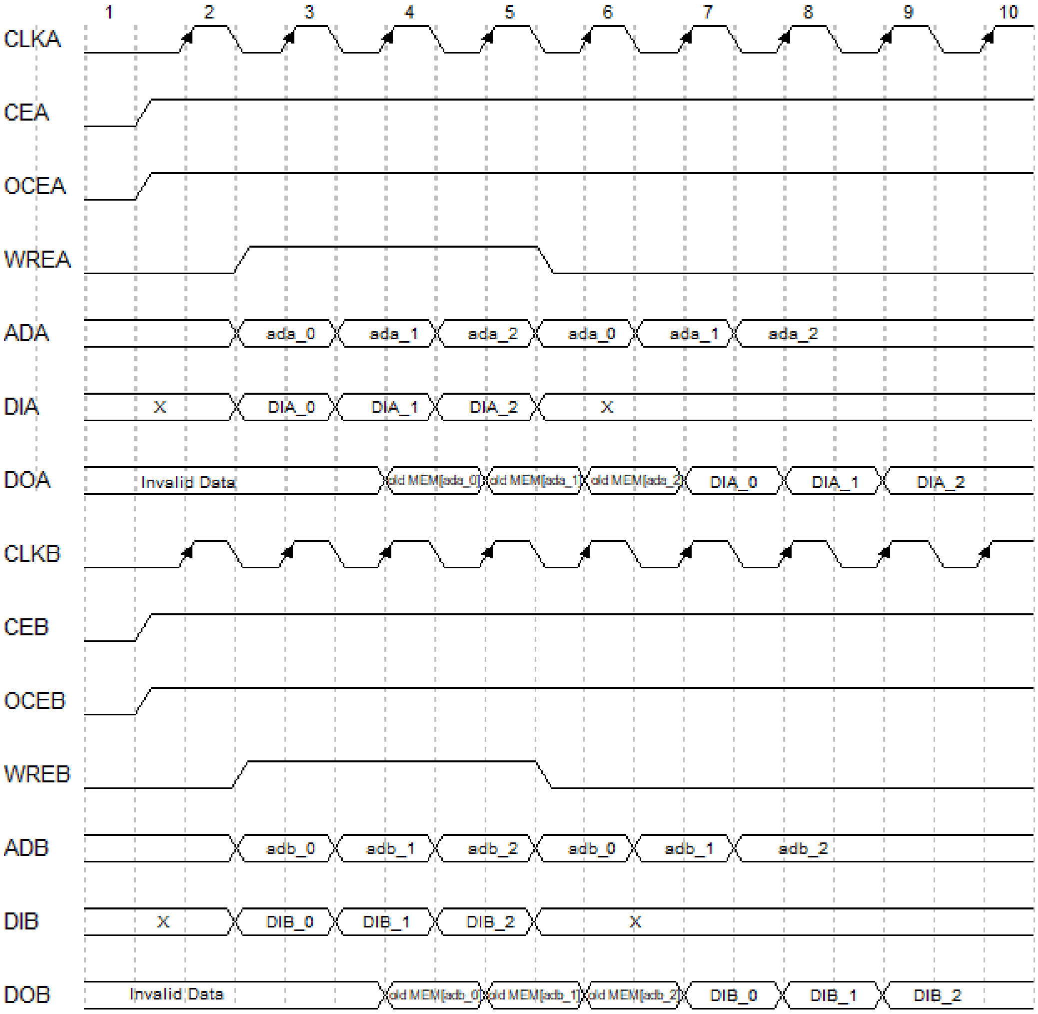


Figure 3-6 Timing Diagram of DPB/DPX9B Read-before-Write Mode (Pipeline Mode)



- Reset mode:
  Support synchronous reset, asynchronous reset, and global reset.

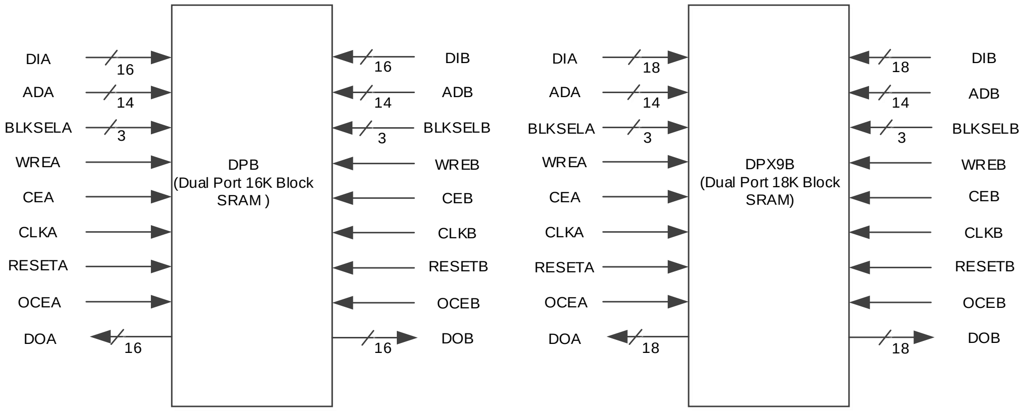
Configuration Relationship

Table 3-1 DPB/DPX9B Data Width and Address Width Configuration Relationship

Dual Port Mode	BSRAM Capacity	Data Width	Address Width
DPB	16 Kbits	1	14
		2	13
		4	12
		8	11
		16	10
DPX9B	18 Kbits	9	11
		18	10

Port Diagram

Figure 3-7 DPB/DPX9B Port Diagram



Port Description

Table 3-2 DPB/DPX9B Port Description

Port Name	I/O	Description
DOA[15:0]/DOA[17:0]	Output	Port A data output signal
DOB[15:0]/DOB[17:0]	Output	Port B data output signal
DIA[15:0]/DIA[17:0]	Input	Port A data input signal
DIB[15:0]/DIB[17:0]	Input	Port B data input signal
ADA[13:0]	Input	Port A address input signal
ADB[13:0]	Input	Port B address input signal
WREA	Input	Port A write enable input signal ● 1: write ● 0: read
WREB	Input	Port B write enable input signal ● 1: write ● 0: read
CEA	Input	Port A clock enable signal, active-high.
CEB	Input	Port B clock enable signal, active-high.
CLKA	Input	Port A clock input signal
CLKB	Input	Port B clock input signal
RESETA	Input	Port A reset input signal, synchronous reset and asynchronous reset supported, active-high. RESETA reset register, rather than the value in reset register
RESETB	Input	Port B reset input signal, synchronous reset and asynchronous reset supported, active-high. RESETB: reset register, rather than the value in reset register
OCEA	Input	Port A output clock enable signal used in Pipeline mode, invalid in Bypass mode.

Port Name	I/O	Description
OCEB	Input	Port B output clock enable signal used in Pipeline mode, invalid in Bypass mode.
BLKSELA[2:0]	Input	BSRAM Port A block selection signal for multiple BSRAM memory units cascading to realize capacity expansion
BLKSELB[2:0]	Input	BSRAM Port B block selection signal for multiple BSRAM memory units cascading to realize capacity expansion

Parameter Description

Table 3-3 DPB/DPX9B Parameter Description

Name	Type	Value	Default	Description
READ_MODE0	Integer	1'b0, 1'b1	1'b0	Port A read mode configuration ● 1'b0: bypass mode ● 1'b1: pipeline mode
READ_MODE1	Integer	1'b0, 1'b1	1'b0	Port B read mode configuration ● 1'b0: bypass mode ● 1'b1: pipeline mode
WRITE_MODE0	Integer	2'b00, 2'b01, 2'b10	2'b00	Port A write mode configuration ● 2'b00: Normal mode ● 2'b01: Write-through mode ● 2'b10: Read-before-Write mode
WRITE_MODE1	Integer	2'b00, 2'b01, 2'b10	2'b00	Port B write mode configuration ● 2'b00: Normal mode ● 2'b01: Write-through mode ● 2'b10: Read-before-Write mode
BIT_WIDTH_0	Integer	DPB: 1,2, 4, 8, 16 DPX9B: 9,18	DPB:16 DPX9B:18	Port A data width configuration
BIT_WIDTH_1	Integer	DPB: 1, 2, 4, 8, 16 DPX9B: 9, 18	DPB: 16 DPB: 18	Port B data width configuration
BLK_SEL_0	Integer	3'b000~3'b111	3'b000	When BSRAM Port A block selection parameter is equal to Port BLKSELA parameter, the BSRAM is selected. The software will handle expansion automatically when IP Core Generator is used to expand storage capacity.

Name	Type	Value	Default	Description
BLK_SEL_1	Integer	3'b000~3'b111	3'b000	When BSRAM Port B block selection parameter is equal to Port BLKSELB parameter, the BSRAM is selected. The software will handle expansion automatically when IP Core Generator is used to expand storage capacity.
RESET_MODE	String	"SYNC", "ASYNC"	"SYNC"	Reset mode configuratiom ● SYNC: synchronized reset ● ASYNC: asynchronous reset
INIT_RAM_00~ INIT_RAM_3F	Integer	DPB: 256'h0…0~256'h1…1 DPX9B: 288'h0…0~288'h1…1	DPB: 256'h0…0 DPX9B: 288'h0…0	Used to set up the initialization data of BSRAM memory unit

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation.

The primitive instantiation is described with the example of DPB.

Verilog Instantiation:

```
DPB bram_dpb_0 (
    .DOA({doa[15:8],doa[7:0]}),
    .DOB({doa[15:8],dob[7:0]}),
    .CLKA(clka),
    .OCEA(ocea),
    .CEA(cea),
    .RESETA(reseta),
    .WREA(wrea),
    .CLKB(clkb),
    .OCEB(oceb),
    .CEB(ceb),
    .RESETB(resetb),
    .WREB(wreb),
    .BLKSELA({3'b000}),
    .BLKSELB({3'b000}),
    .ADA({ada[10:0],3'b000}),
    .DIA({{8{1'b0}},dia[7:0]})
```

```
          .ADB({adb[10:0],3'b000}),
          .DIB({{8{1'b0}},dib[7:0]})
      );
      defparam bram_dpb_0.READ_MODE0=1'b0;
      defparam bram_dpb_0.READ_MODE1=1'b0;
      defparam bram_dpb_0.WRITE_MODE0=2'b00;
      defparam bram_dpb_0.WRITE_MODE1=2'b00;
      defparam bram_dpb_0.BIT_WIDTH_0 = 8;
      defparam bram_dpb_0.BIT_WIDTH_1 = 8;
      defparam bram_dpb_0.BLK_SEL_0 = 3'b000;
      defparam bram_dpb_0.BLK_SEL_1=3'b000;
      defparam bram_dpb_0.RESET_MODE = "SYNC";
      defparam bram_dpb_0.INIT_RAM_00 =
256'h00A000000000000B00A000000000000B00A000000000000B00A00
0000000000B;
      defparam
bram_dpb_0.INIT_RAM_3E=256'h00A000000000000B00A000000000000
B00A000000000000B00A000000000000B;
      defparam
bram_dpb_0.INIT_RAM_3F=256'h00A000000000000B00A000000000000
B00A000000000000B00A000000000000B;
```

Vhdl Instantiation:

```
    COMPONENT  DPB
          GENERIC  (
                BIT_WIDTH_0:integer:=16;
                BIT_WIDTH_1:integer:=16;
                READ_MODE0:bit:='0';
                READ_MODE1:bit:='0';
                WRITE_MODE0:bit_vector:="00";
                WRITE_MODE1:bit_vector:="00";
                BLK_SEL_0:bit_vector:="000";
                BLK_SEL_1:bit_vector:="000";
                RESET_MODE  :  string:="SYNC";
                INIT_RAM_00:bit_vector:=X"000000000000000
0000000000000000000000000000000000000000000000000";
                INIT_RAM_01:bit_vector:=X"000000000000000
0000000000000000000000000000000000000000000000000";
```

```
                        INIT_RAM_3F:bit_vector:=X"000000000000000
000000000000000000000000000000000000000000000000"
                );
                PORT (
                        DOA,DOB:OUT std_logic_vector(15 downto 0):
=conv_std_logic_vector(0,16);
                        CLKA,CLKB,CEA,CEB,OCEA,OCEB,RESETA,
RESETB,WREA,WREB:IN std_logic;
                        ADA,ADB:IN std_logic_vector(13 downto 0);
                        BLKSELA:IN std_logic_vector(2 downto 0);
                        BLKSELB:IN std_logic_vector(2 downto 0);
                        DIA,DIB:IN std_logic_vector(15 downto 0)
                );
        END COMPONENT;
        uut:DPB
            GENERIC MAP(
                        BIT_WIDTH_0=>16,
                        BIT_WIDTH_1=>16,
                        READ_MODE0=>'0',
                        READ_MODE1=>'0',
                        WRITE_MODE0=>"00",
                        WRITE_MODE1=>"00",
                        BLK_SEL_0=>"000",
                        BLK_SEL_1=>"000",
                        RESET_MODE=>"SYNC",
        INIT_RAM_00=>X"0000000000000000000000000000000000000000
000000000000000000000000",
        INIT_RAM_01=>X"0000000000000000000000000000000000000000
000000000000000000000000",
        INIT_RAM_3F=>X"0000000000000000000000000000000000000000
000000000000000000000000"
            )
            PORT MAP (
                DOA=>doa,
                DOB=>dob,
                CLKA=>clka,
                CLKB=>clkb,
```

```
        CEA=>ceb,
        CEB=>ceb,
        OCEA=>ocea,
        OCEB=>oceb,
        RESETA=>reseta,
        RESETB=>resetb,
        WREA=>wrea,
        WREB=>wreb,
        ADA=>ada,
        ADB=>adb,
        BLKSELA=>blksela,
        BLKSELB=>blkselb,
        DIA=>dia,
        DIB=>dib
    );
```

3.2 Single Port Mode

Primitive Introduction

SP/SPX9 (Single Port 16K BSRAM/Single Port 18K BSRAM), 16K/18K single Port BSRAM.

Functional Description

SP/SPX9 works in single port mode with 16 Kbits/18 Kbits memory capacity. Port A and Port B can read/write independently. SP/SPX9 supports two read modes (Bypass mode and Pipeline mode) and three write modes (normal mode, write-through mode and Read-before-Write mode[1]).

Note!

138K devices do not support Read-before-Write mode.

- Read mode:
  READ_MODE enable or disable the output pipeline register. When the output pipeline register is used, the read operation needs extra clock cycle.
- Write mode:
  Normal mode, write-through mode and Read-before-Write mode are configured WRITE_MODE. The corresponding internal timing diagrams for different read/write modes of the single port BSRAM can be referred to the Port A and Port B timing diagrams of dual port BSRAM, as shown from Figure 3-1 to Figure 3-6.
- Reset mode:
  Support synchronous reset, asynchronous reset, and global reset.

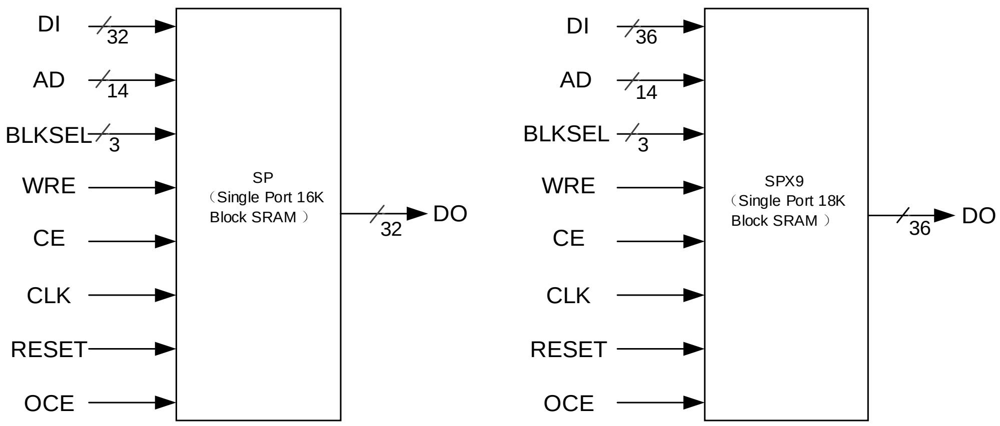
Configuration Relationship

Table 3-4 SP/SPX9 Data Width and Address Width Configuration Relationship

Single Port Mode	BSRAM Capacity	Data Width	Address Width
SP	16 Kbits	1	14
		2	13
		4	12
		8	11
		16	10
		32	9
SPX9	18 Kbits	9	11
		18	10
		36	9

Port Diagram

Figrue 3-8 SP/SPX9 Port Diagram



Port Description

Table 3-5 SP/ SPX9 Port Description

Port Name	I/O	Description
DO[31:0]/DO[35:0]	Output	Data output signal
DI[31:0]/DI[35:0]	Input	Data input signal
AD[13:0]	Input	Address input signal
WRE	Input	Write enable input signal ● 1: write ● 0: read
CE	Input	Clock enable input signal, active-high.
CLK	Input	Clock input signal

Port Name	I/O	Description
RESET	Input	Reset input signal, synchronous reset and asynchronous reset supported, active-high. RESET: reset register, rather than the value in reset register.
OCE	Input	Output clock enable signal used in Pipeline mode, invalid in Bypass mode
BLKSEL[2:0]	Input	BSRAM block selection signal for multiple BSRAM memory units cascading to realize capacity expansion.

Parameter Description

Table 3-6 SP/SPX9 Parameter Description

Name	Type	Value	Default	Description
READ_MODE	Integer	1'b0, 1'b1	1'b0	Read mode configuration ● 1'b0: bypass mode ● 1'b1: pipeline mode
WRITE_MODE	Integer	2'b00, 2'b01, 2'b10	2'b00	Write mode configuration ● 2'b00: Normal mode ● 2'b01: Write-through mode ● 2'b10: Read-before-Write mode
BIT_WIDTH	Integer	SP: 1, 2, 4, 8, 16, 32 SPX9: 9, 18, 36	SP:32 SPX9:36	Data width configuration
BLK_SEL	Integer	3'b000~3'b111	3'b000	BSRAM block selection parameter is equal to BLKSEL, and the BSRAM is selected. The software will handle expansion automatically when IP Core Generator is used to expand storage capacity.
RESET_MODE	String	"SYNC", "ASYNC"	"SYNC"	Reset mode configuration ● SYNC: synchronized reset ● ASYNC: asynchronous reset
INIT_RAM_00~ INIT_RAM_3F	Integer	SP: 256'h0…0~256'h1…1 SPX9: 288'h0…0~288'h1…1	SP: 256'h0…0 SPX9: 288'h0…0	Used to set up the initialization data of BSRAM memory unit

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP

Core Generator. For the details, you can refer to Chapter 6 IP Generation. The primitive instantiation is described with the example of SP.

Verilog Instantiation:

```
SP bram_sp_0 (
    .DO({dout[31:8], dout[7:0]}),
    .CLK(clk),
    .OCE(oce),
    .CE(ce),
    .RESET(reset),
    .WRE(wre),
    .BLKSEL({3'b000}),
    .AD({ad[10:0], 3'b000}),
    .DI({{24{1'b0}}, din[7:0]})
);
defparam bram_sp_0.READ_MODE=1'b0;
defparam bram_sp_0.WRITE_MODE=2'b00;
defparam bram_sp_0.BIT_WIDTH = 8;
defparam bram_sp_0.BLK_SEL = 3'b000;
defparam bram_sp_0.RESET_MODE = "SYNC";
defparam bram_sp_0.INIT_RAM_00 =
256'h00A000000000000B00A000000000000B00A000000000000B00
A000000000000B;
defparam
bram_sp_0.INIT_RAM_01=256'h00A000000000000B00A000000000
000B00A000000000000B00A000000000000B;
defparam
bram_sp_0.INIT_RAM_3F=256'h00A000000000000B00A000000000
000B00A000000000000B00A000000000000B;
```

Vhdl Instantiation:

```
COMPONENT SP
    GENERIC(
        BIT_WIDTH:integer:=32;
        READ_MODE:bit:='0';
        WRITE_MODE:bit_vector:="01";
        BLK_SEL : bit_vector:="000";
        RESET_MODE : string:="SYNC";
        INIT_RAM_00:bit_vector:=X"00A000000000000B
```

```
00A000000000000B00A000000000000B00A000000000000B";
                    INIT_RAM_01:bit_vector:=X"00A000000000000B
00A000000000000B00A000000000000B00A000000000000B";
                    INIT_RAM_3F:bit_vector:=X"00A000000000000B
00A000000000000B00A000000000000B00A000000000000B"
            );
            PORT (
                    DO:OUT std_logic_vector(31 downto 0):=conv_
std_logic_vector(0,32);
                    CLK,CE,OCE,RESET,WRE:IN std_logic;
                    AD:IN std_logic_vector(13 downto 0);
                    BLKSEL:IN std_logic_vector(2 downto 0);
                    DI:IN std_logic_vector(31 downto 0)
            );
     END COMPONENT;
     uut:SP
          GENERIC MAP(
                      BIT_WIDTH=>32,
                      READ_MODE=>'0',
                      WRITE_MODE=>"01",
                      BLK_SEL=>"000",
                      RESET_MODE=>"SYNC",
                      INIT_RAM_00=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B",
                      INIT_RAM_01=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B",
                      INIT_RAM_02=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B",
                      INIT_RAM_3F=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B"
          )
          PORT MAP (
              DO=>dout,
              CLK=>clk,
              OCE=>oce,
              CE=>ce,
              RESET=>reset,
```

```
        WRE=>wre,
        BLKSEL=>blksel,
        AD=>ad,
        DI=>din
    );
```

3.3 Semi Dual Port Mode

Primitive Introduction

SDPB/SDPX9B(Semi Dual Port 16K Block SRAM/Semi Dual Port 18K Block SRAM), 16K/18K Semi Dual Port mode BSRAM.

Functional Description

SDPB/SDPX9b works in Semi Dual Port mode with 16 Kbits/18 Kbits memory capacity. Port A writes and Port B reads[1]. SDPB/SDPX9B supports two read modes (Bypass mode and Pipeline mode) and one write mode (Normal mode).

Note!

[1] Performing read and write operations to the same address at the same time is not recommended.

- Read mode:
  READ_MODE enable or disable the output pipeline register. When the output pipeline register is used, the read operation needs extra clock cycle.
- Write mode:
  SDPB/SDPX9B Port A is for write operation, and Port B is for read operation, supporting Normal mode. The corresponding internal timing diagrams of different read modes for Semi Dual Port BSRAM are shown in Figure 3-9 and Figure 3-10.

Figure 3-9 Timing Diagram of Semi Dual Port BSRAM Normal Mode (Bypass Mode)

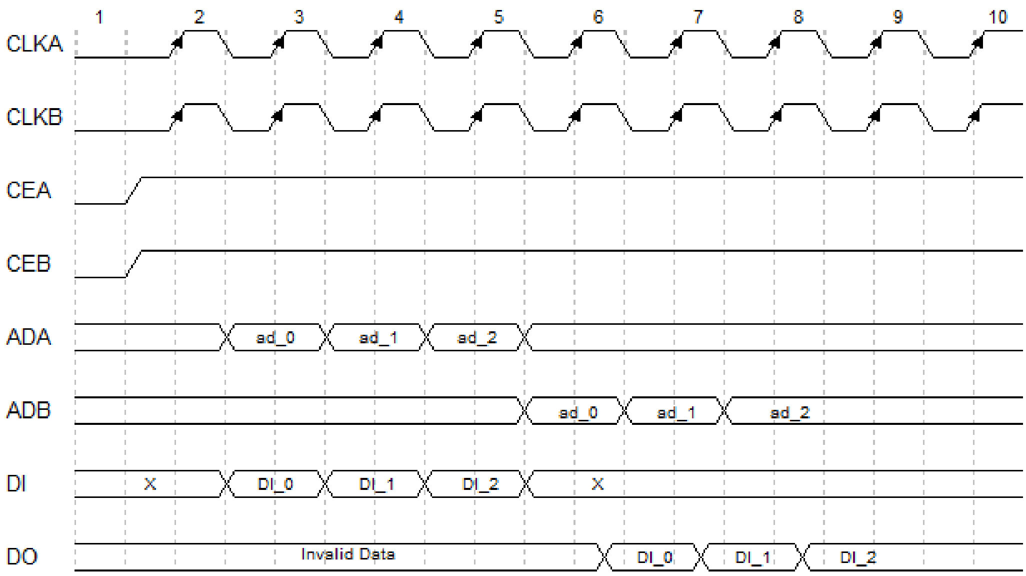
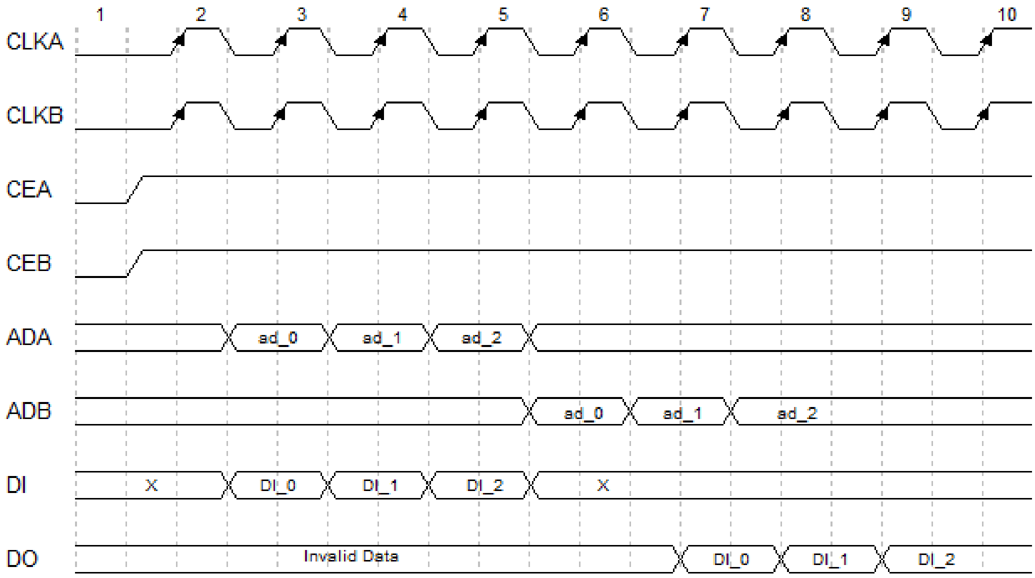


Figure 3-10 Timing Diagram of Semi Dual Port BSRAM Normal Mode (Pipeline Mode)



- Reset mode:
  Support synchronous reset, asynchronous reset, and global reset.
- Byte_enable:
  byte_enable is controlled by 8-bit byte enable port.
- Cascade:
  Support cascade

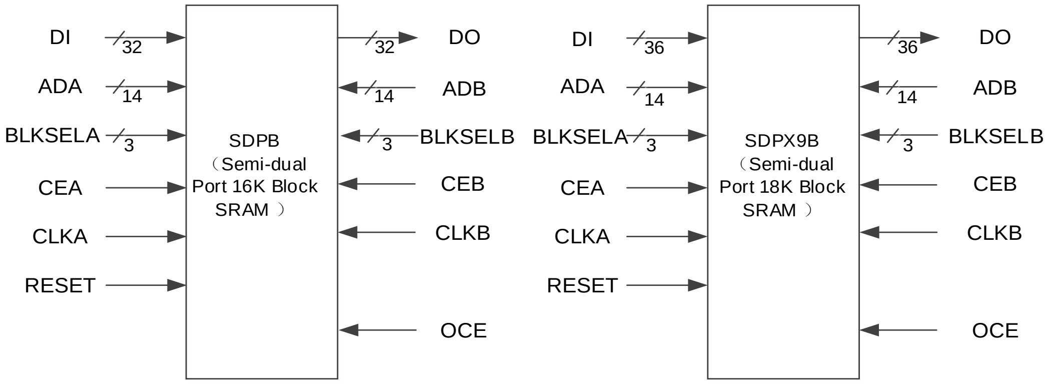
Configuration Relationship

Table 3-7 SDPB/SDPX9B Data Width and Address Width Configuration Relationship

Semi Dual Port Mode	BSRAM Capacity	Data Width	Address Width
SDPB	16Kbits	1	14
		2	13
		4	12
		8	11
		16	10
		32	9
SDPX9B	18Kbits	9	11
		18	10
		36	9

Port Diagram

Figrue 3-11 SDPB/SDPX9B Port Diagram



Port Description

Table 3-8 SDPB/SDPX9B Port Description

Port Name	I/O	Description
DO[31:0]/DO[35:0]	Output	Data output signal
DI[31:0]/DI[35:0]	Input	Data input signal
ADA[13:0]	Input	Port A address input signal
ADB[13:0]	Input	Port B address input signal
CEA	Input	Port A clock enable signal, active-high.
CEB	Input	Port B clock enable signal, active-high.
CLKA	Input	Port A clock input signal
CLKB	Input	Port B clock input signal
RESET	Input	Reset input signal, synchronous reset and asynchronous reset supported, active-high. RESET: reset register, rather than the value in reset register.
OCE	Input	Output clock enable signal used in Pipeline mode, invalid in Bypass mode
BLKSELA[2:0]	Input	BSRAM Port A block selection signal for multiple BSRAM memory units cascading to realize capacity expansion
BLKSELB[2:0]	Input	BSRAM Port B block selection signal for multiple BSRAM memory units cascading to realize capacity expansion.

Parameter Description

Table 3-9 SDPB/ SDPX9B Parameter Description

Name	Type	Value	Default	Description
READ_MODE	Integer	1'b0, 1'b1	1'b0	Read mode configuration ● 1'b0: bypass mode ● 1'b1: pipeline mode
BIT_WIDTH_0	Integer	SDPB: 1, 2, 4, 8, 16, 32 SDPX9B: 9, 18, 36	SDPB: 32 SDPX9B: 36	Port A data width configuration
BIT_WIDTH_1	Integer	SDPB: 1, 2, 4, 8, 16, 32 SDPX9B: 9, 18, 36	SDPB: 32 SDPX9B: 36	Port B data width configuration
BLK_SEL_0	Integer	3'b000~3'b111	3'b000	When BSRAM Port A block selection parameter is equal to Port BLKSELA parameter, the BSRAM is selected. The software will handle expansion automatically when IP Core Generator is used to expand storage capacity.
BLK_SEL_1	Integer	3'b000~3'b111	3'b000	When BSRAM Port B block selection parameter is equal to Port BLKSELB parameter, the BSRAM is selected. The software will handle expansion automatically when IP Core Generator is used to expand storage capacity.
RESET_MODE	String	"SYNC", "ASYNC"	"SYNC"	Reset mode configuration ● SYNC: synchronized reset ● ASYNC: asynchronous reset
INIT_RAM_00~ INIT_RAM_3F	Integer	SDPB:256'h0…0~256'h1…1 SDPX9B:288'h0…0~288'h1…1	SDPB:256'h0…0 SDPX9B: 288'h0…0	Used to set up the initialization data of BSRAM memory unit

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation. The primitive instantiation is described with the example of SDPB.

Verilog Instantiation:

```
SDPB bram_sdpb_0 (
        .DO({dout[31:16], dout[15:0]}),
        .CLKA(clka),
        .CEA(cea),
        .CLKB(clkb),
        .CEB(ceb),
        .RESET(reset),
        .OCE(oce),
        .BLKSELA({3'b000}),
        .BLKSELB({3'b000}),
        .ADA({ada[9:0], 2'b00, byte_en[1:0]}),
        .DI({{16{1'b0}},din[15:0]}),
        .ADB({adb[9:0],4'b0000})
);
defparam bram_sdpb_0.READ_MODE=1'b1;
defparam bram_sdpb_0.BIT_WIDTH_0=16;
defparam bram_sdpb_0.BIT_WIDTH_1=16;
defparam bram_sdpb_0.BLK_SEL_0 = 3'b000;
defparam bram_sdpb_0.BLK_SEL_1=3'b000;
defparam bram_sdpb_0.RESET_MODE = "SYNC";
defparam bram_sdpb_0.INIT_RAM_00 =
256'h00A000000000000B00A000000000000B00A000000000000B00
A000000000000B;
defparam
bram_sdpb_0.INIT_RAM_3F=256'h00A000000000000B00A0000000
00000B00A000000000000B00A000000000000B;
```

Vhdl Instantiation:

```
 COMPONENT  SDPB
            GENERIC(
                    BIT_WIDTH_0:integer:=16;
                    BIT_WIDTH_1:integer:=16;
                    READ_MODE:bit:='0';
                    BLK_SEL_0:bit_vector:="000";
                    BLK_SEL_1:bit_vector:="000";
                    RESET_MODE  :  string:="SYNC";
```

```
INIT_RAM_00:bit_vector:=X"00A000000000000B00A000000000000B00A000000000000B00A000000000000B";
INIT_RAM_01:bit_vector:=X"00A000000000000B00A000000000000B00A000000000000B00A000000000000B";
INIT_RAM_3F:bit_vector:=X"00A000000000000B00A000000000000B00A000000000000B00A000000000000B"
        );
        PORT (
                DO:OUT std_logic_vector(31 downto 0):=conv_std_logic_vector(0,32);
                CLKA,CLKB,CEA,CEB:IN std_logic;
                OCE,RESET:IN std_logic;
                ADA,ADB:IN std_logic_vector(13 downto 0);
                BLKSELA:IN std_logic_vector(2 downto 0);
                BLKSELB:IN std_logic_vector(2 downto 0);
                DI:IN std_logic_vector(31 downto 0)
        );
END COMPONENT;
uut:SDPB
    GENERIC MAP(
                BIT_WIDTH_0=>16,
                BIT_WIDTH_1=>16,
                READ_MODE=>'0',
                BLK_SEL_0=>"000",
                BLK_SEL_1=>"000",
                RESET_MODE=>"SYNC",
                INIT_RAM_00=>X"00A000000000000B00A000000000000B00A000000000000B00A000000000000B",
                INIT_RAM_01=>X"00A000000000000B00A000000000000B00A000000000000B00A000000000000B",
                INIT_RAM_3F=>X"00A000000000000B00A000000000000B00A000000000000B00A000000000000B"
                )
        PORT MAP (
            DO=>dout,
            CLKA=>clka,
            CEA=>cea,
```

```
        CLKB=>clkb,
        CEB=>ceb,
        RESET=>reset,
        OCE=>oce,
        BLKSELA=>blksela,
        BLKSELB=>blkselb,
        ADA=>ada,
        DI=>din,
        ADB=>adb
    );
```

3.4 Semi Dual Port Mode with ECC Function

Primitive Introduction

SDP36KE (Semi Dual Port 36K Block SRAM with ECC function), 36K Semi Dual Port BSRAM with ECC Function.

Device supported

Table 3-10 SDP36KE Device Supported

Family	Series	Device
Arora	GW5AT	GW5AT-138, GW5AT-138 B Veriosn
	GW5AST	GW5AST-138 B Veriosn

Functional Description

SDP36KE works in Semi Dual Port mode with 36 Kbits memory capacity. Port A writes and Port B reads[1]. SDP36KE supports two read modes (Bypass mode and Pipeline mode) and one write mode (Normal mode).

Note!

[1] Performing read and write operations to the same address at the same time is not recommended.

- Read mode:
  READ_MODE enable or disable the output pipeline register. When the output pipeline register is used, the read operation needs extra clock cycle.
- Write mode:
  SDP36KE Port A is for write operation, and Port B is for read operation, supporting Normal mode.
- Reset mode:
  Supports synchronous reset, asynchronous reset, and global reset.
- Parity Check:
  The data port is a combination of 64 + 8 = 72 bits, i.e.64-bit

input/output data (DI/DO) + 8-bit input/output data (DIP/DOP); the 8-bit DIP/DOP is the parity input/output.

- ECC:
  Support ECC with data width 72bits. ECC supports Standard mode, Encoder-only mode, and Decoder-only mode.
  - Standard ECC: Enable encoder and decoder at the same time, and ECC function can be implemented by using encoder and decoder.
  - Encoder-only ECC: Enable encoder and disable decoder, and the read value is output immediately without decoding.
  - Decoder-only ECC: Disable encoder and enable decoder.
- Cascade:
  Support cascade

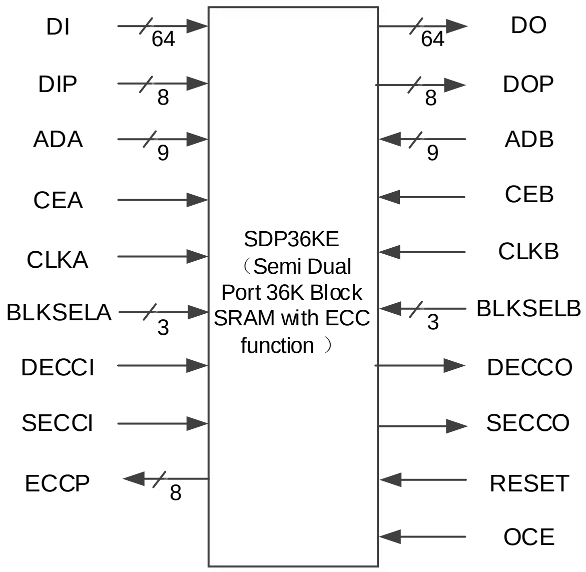
Configuration Relationship

Table 3-11 SDP36KE Data Width and Address Width Configuration Relationship

Semi Dual Port Mode	BSRAM Capacity	Data Width	Address Width
SDP36KE	36 Kbits	72	9

Port Diagram

Figure 3-12 SDP36KE Port Diagram



Port Description

Table 3-12 SDP36KE Port Description

Port Name	I/O	Description
DO[63:0]	Output	Data output signal
DI[63:0]	Input	Data input signal

Port Name	I/O	Description
DIP[7:0]	Input	DIP can be as the parity input, and it can be used as data in non-ECC mode, but not in ECC mode.
DOP[7:0]	Output	DOP can be as the parity output, and it can be used as data in non-ECC mode, but not in ECC mode.
ECCP[7:0]	Output	ECC encoder check bit
ADA[8:0]	Input	Port A address input signal
ADB[8:0]	Input	Port B address input signal
CEA	Input	Port A clock enable signal, active-high.
CEB	Input	Port B clock enable signal, active-high.
CLKA	Input	Port A clock input signal
CLKB	Input	Port B clock input signal
RESET	Input	Output reset signal
OCE	Input	Output clock enable signal used in Pipeline mode, invalid in Bypass mode
BLKSELA[2:0]	Input	BSRAM Port A block selection signal for multiple BSRAM memory units cascading to realize capacity expansion
BLKSELB[2:0]	Input	BSRAM Port B block selection signal for multiple BSRAM memory units cascading to realize capacity expansion.
DECCI	Input	Input 2-bit error signal
SECCI	Input	Input 1-bit error signal
DECCO	Output	2-bit error is detected
SECCO	Output	1-bit error is detected

Parameter Description

Table 3-13 SDP36KE Parameter Description

Name	Type	Value	Default	Description
READ_MODE	Integer	1'b0, 1'b1	1'b0	Read mode configuration ● 1'b0: bypass mode ● 1'b1: pipeline mode
BLK_SEL_A	Integer	3'b000~3'b111	3'b000	When BSRAM Port A block selection parameter is equal to Port BLKSELA parameter, the BSRAM is selected. The software will handle expansion automatically when IP Core Generator is used to expand storage capacity.

Name	Type	Value	Default	Description
BLK_SEL_B	Integer	3'b000~3'b111	3'b000	When BSRAM Port B block selection parameter is equal to Port BLKSELB parameter, the BSRAM is selected. The software will handle expansion automatically when IP Core Generator is used to expand storage capacity.
RESET_MODE	String	"SYNC", "ASYNC"	"SYNC"	Reset mode configuration ● SYNC: synchronized reset ● ASYNC: asynchronous reset
ECC_WRITE_EN	String	"TRUE", "FALSE"	"FALSE"	ECC Encoder Configuration ● TRUE: enable ECC Encoder ● FALSE: disable ECC Encoder
ECC_READ_EN	String	"TRUE", "FALSE"	"FALSE"	ECC Decoder Configuration ● TRUE: enable ECC Decoder ● FALSE: disable ECC Decoder
INIT_RAM_00~ INIT_RAM_7F	Integer	256'h0…0~256'h1…1	256'h0…0	Used to specify the initial value of 32-Kbit memory, output by DO.
INITP_RAM_00~ INITP_RAM_0F	Integer	256'h0…0~256'h1…1	256'h0…0	Used to specify the initial value of 4-Kbit memory, output by DOP.
INIT_FILE	String	"NONE", "*.ini"	"NONE"	Specify initialization file NONE: if there is no initialization file, specify the memory value by INIT_RAM_00~ INIT_RAM_7F and INITP_RAM_00~ INITP_RAM_0F.

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation.

Verilog Instantiation:

```
SDP36KE bram_sdp36ke_0 (
        .DI({{28{1'b0}},din[35:0]}),
        .DO({dout[63:36],dout[35:0]}),
        .DIP({8{1'b0}}),
        .DOP({dout[71:64]}),
        .ECCP(eccp),
        .ADA(ada),
        .ADB(adb),
        .CLKA(clka),
        .CLKB(clkb),
        .CEA(cea),
        .CEB(ceb),
        .OCE(oce),
        .RESET(reset),
        .BLKSELA({3'b000}),
        .BLKSELB({3'b000}),
        .DECCI(decci),
        .SECCI(secci),
        .DECCO(decco),
        .SECCO(secco)
);
defparam bram_sdp36ke_0.ECC_WRITE_EN = "FALSE";
defparam bram_sdp36ke_0. ECC_READ_EN = "FALSE";
defparam bram_sdp36ke_0.READ_MODE = 1'b0;
defparam bram_sdp36ke_0.BLK_SEL_A = 3'b000;
defparam bram_sdp36ke_0.BLK_SEL_B = 3'b000;
defparam bram_sdp36ke_0.RESET_MODE = "SYNC";
defparam bram_sdp36ke_0.INIT_FILE = "NONE";
defparam bram_sdp36ke_0.INIT_RAM_00 =
256'h00A000000000000B00A000000000000B00A000000000000B00
A000000000000B;
defparam bram_sdp36ke_0.INIT_RAM_7F =
256'h00A000000000000B00A000000000000B00A000000000000B00
A000000000000B;
defparam bram_sdp36ke_0.INITP_RAM_00 =
256'h00A000000000000B00A000000000000B00A000000000000B00
```

```
A000000000000B;
defparam bram_sdp36ke_0.INITP_RAM_0F =
256'h00A000000000000B00A000000000000B00A000000000000B00
A000000000000B;
```

Vhdl Instantiation：

```
COMPONENT SDP36KE
      GENERIC(
            ECC_WRITE_EN:string:="FALSE";
            ECC_READ_EN:string:="FALSE";
            READ_MODE:bit:='0';
            BLK_SEL_A:bit_vector:="000";
            BLK_SEL_B:bit_vector:="000";
            RESET_MODE:string:="SYNC";
            INIT_FILE:string:="NONE";
            INIT_RAM_00:bit_vector:=X"00A000000000000
B00A000000000000B00A000000000000B00A000000000000B";
            INIT_RAM_7F:bit_vector:=X"00A000000000000
B00A000000000000B00A000000000000B00A000000000000B";
            INITP_RAM_00:bit_vector:=X"00A00000000000
0B00A000000000000B00A000000000000B00A000000000000B";
            INITP_RAM_0F:bit_vector:=X"00A00000000000
0B00A000000000000B00A000000000000B00A000000000000B"
      );
      PORT(
            DO:OUT std_logic_vector(63 downto 0):=conv_
std_logic_vector(0,64);
            DOP:OUT std_logic_vector(7 downto 0):=conv_
std_logic_vector(0,8);
            ECCP:OUT std_logic_vector(7 downto 0):=conv
_std_logic_vector(0,8);
            DECCO,SECCO:OUT std_logic:=conv_std_logic;
            DECCI,SECCI:IN std_logic;
            ADA,ADB:IN std_logic_vector(9 downto 0);
            CLKA,CLKB,CEA,CEB:IN std_logic;
            OCE,RESET:IN std_logic;
            BLKSELA:IN std_logic_vector(2 downto 0);
            BLKSELB:IN std_logic_vector(2 downto 0);
            DIP:IN std_logic_vector(7 downto 0);
```

```
                        DI:IN  std_logic_vector(63  downto  0)
                    );
            END  COMPONENT;
            uut:SDP36KE
                GENERIC  MAP(
                        ECC_WRITE_EN=>"FALSE";
                        ECC_READ_EN=>"FALSE";
                        READ_MODE=>'0';
                        BLK_SEL_A=>"000";
                        BLK_SEL_B=>"000";
                        RESET_MODE=>"SYNC";
                        INIT_FILE=>"NONE";
                        INIT_RAM_00=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B",
                        INIT_RAM_7F=>X"00A000000000000B00A00
0000000000B00A000000000000B00A000000000000B",
                        INITP_RAM_00=>X"00A000000000000B00A0
00000000000B00A000000000000B00A000000000000B",
                        INITP_RAM_0F=>X"00A000000000000B00A0
00000000000B00A000000000000B00A000000000000B"
                        )
                PORT MAP(
                    DI=>din,
                    DO=>dout,
                    DIP=>dip,
                    DOP=>dop,
                    ECCP=>eccp,
                    ADA=>ada,
                    ADB=>adb,
                    CLKA=>clka,
                    CLKB=>clkb,
                    CEA=>cea,
                    CEB=>ceb,
                    OCE=>oce,
                    RESET=>reset,
                    BLKSELA=>blksela,
```

```
        BLKSELB=>blkselb,
        DECCI=>decci,
        SECCI=>secci,
        DECCO=>decco,
        SECCO=>secco
    );
```

3.5 Read Only Mode

Primitive Introduction

pROM/pROMX9(16K/18K Block ROM) is 16K/18K block read only memory.

Functional Description

pROM/pROMX9 works in Read Only mode with 16 Kbits/18 Kbits memory capacity. pROM/pROMX9 supports 2 modes (Bypass mode and Pipeline mode).

READ_MODE enable or disable the output pipeline register. When the output pipeline register is used, the read operation needs extra delay cycle.

The corresponding internal timing diagrams for different read modes of ROM can be referred to the Port B timing diagrams of Semi Dual Port BSRAM, as shown in Figure 3-13 and Figure 3-14.

Figure 3-13 ROM Timing Diagram (Bypass Mode)

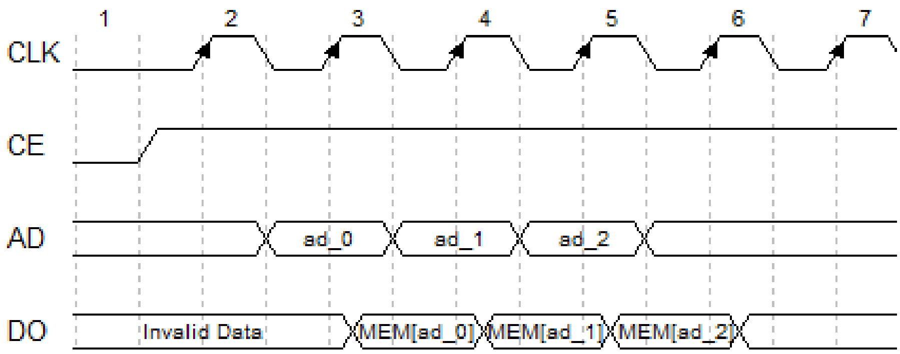
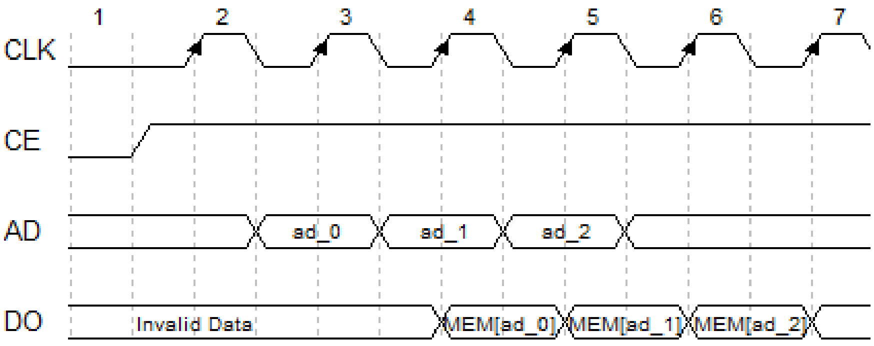


Figure 3-14 ROM Timing Diagram (Pipeline Mode)



- Reset mode:
  Supports synchronous reset, asynchronous reset, and global reset.

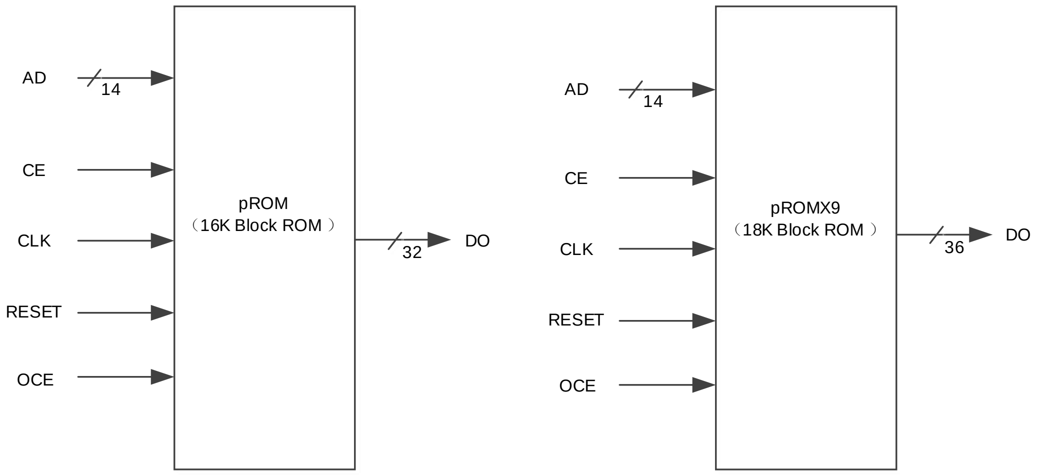
Configuration Relationship

Table 3-14 pROM/ pROMX9 Data Width and Address Width Configuration Relationship

Read Only Mode	BSRAM Capacity	Data Width	Address Width
pROM	16 Kbits	1	14
		2	13
		4	12
		8	11
		16	10
		32	9
pROMX9	18 Kbits	9	11
		18	10
		36	9

Port Diagram

Figrue 3-15 pROM/ pROMX9 Port Diagram



Port Description

Table 3-15 pROM/ pROMX9 Port Description

Port Name	I/O	Description
DO[31:0]/DO[35:0]	Output	Data output signal
AD[13:0]	Input	Address input signal
CE	Input	Clock enable input signal, active-high.
CLK	Input	Clock input signal
RESET	Input	Reset input signal, synchronous reset and asynchronous reset supported, active-high. RESET: reset register, rather than the value in reset register.
OCE	Input	Output clock enable signal used in Pipeline mode, invalid in Bypass mode.

Parameter Description

Table 3-16 pROM/pROMX9 Parameter Description

Name	Type	Value	Default	Description
READ_MODE	Integer	1'b0, 1'b1	1'b0	Read mode configuration ● 1'b0: bypass mode ● 1'b1: pipeline mode
BIT_WIDTH	Integer	pROM: 1, 2, 4, 8, 16, 32 pROMX9: 9, 18, 36	pROM: 32 pROMX9: 36	Data width configuration
RESET_MODE	String	"SYNC", "ASYNC"	"SYNC"	Reset mode configuration ● SYNC: synchronized reset ● ASYNC: asynchronous reset
INIT_RAM_00~ INIT_RAM_3F	Integer	pROM:256'h0…0~256'h1…1 pROMX9:288'h0…0~288'h1…1	pROM:256'h0…0 pROMX9: 288'h0…0	Used to set up the initialization data of BSRAM memory unit

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation. The primitive instantiation is described with the example of pROM.

Verilog Instantiation:

```
pROM bram_prom_0 (
    .DO({dout[31:8], dout[7:0]}),
    .CLK(clk),
```

```
        .OCE(oce),
        .CE(ce),
        .RESET(reset),
        .AD({ad[10:0],3'b000})
    );
    defparam bram_prom_0.READ_MODE=1'b0;
    defparam bram_prom_0.BIT_WIDTH = 8;
    defparam bram_prom_0.RESET_MODE="SYNC";
    defparam bram_prom_0.INIT_RAM_00=256'h9C23645D0F78986FFC3E36E141541B95C19F2F7164085E631A819860D8FF0000;
    defparam bram_prom_0.INIT_RAM_01 = 256'h000000000000000000000000000000000000000000000000000000FFFFFFBDCF;
```

Vhdl Instantiation:

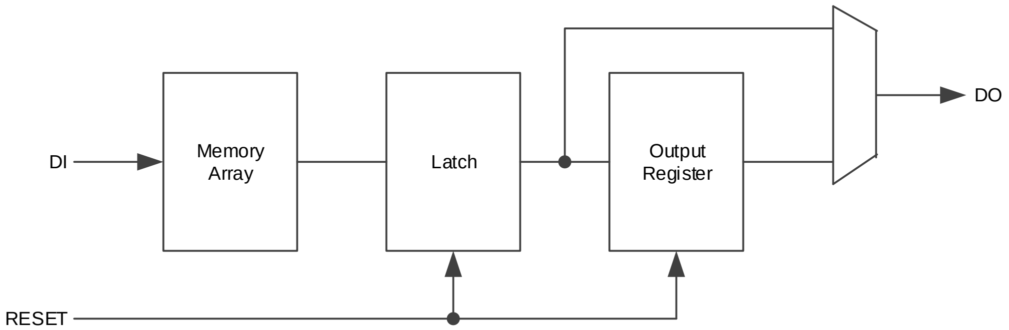
```
    COMPONENT pROM
        GENERIC(
                BIT_WIDTH:integer:=1;
                READ_MODE:bit:='0';
                RESET_MODE:string:="SYNC";
                INIT_RAM_00:bit_vector:=X"9C23645D0F78986FFC3E36E141541B95C19F2F7164085E631A819860D8FF0000";
                INIT_RAM_01:bit_vector:=X"000000000000000000000000000000000000000000000000000000FFFFFFBDCF"
        );
        PORT(
                DO:OUT std_logic_vector(31 downto 0):=conv_std_logic_vector(0,32);
                CLK,CE,OCE,RESET:IN std_logic;
                AD:IN std_logic_vector(13 downto 0)
        );
    END COMPONENT;
    uut:pROM
        GENERIC MAP(
                BIT_WIDTH=>1,
                READ_MODE=>'0',
                RESET_MODE=>"SYNC",
```

```
INIT_RAM_00=>X"9C23645D0F78986FFC3E36
E141541B95C19F2F7164085E631A819860D8FF0000",
INIT_RAM_01=>X"00000000000000000000000
00000000000000000000000000000FFFFFFBDCF  "
)
PORT MAP (
    DO=>do,
    AD=>ad,
    CLK=>clk,
    CE=>ce,
    OCE=>oce,
    RESET=>reset
);
```

4 BSRAM Output Reset

The output module supports reset and outputs reset data 0. The structure diagram is as shown in Figure 4-1.

Figure 4-1 Output Reset Structure Diagram



The output port outputs 0 when the RESET signal is active high.

RESET supports synchronous reset and asynchronous reset. When you call the primitives library directly, set the parameter "RESET_MODE". Select the reset mode through software windows when you use IP Core Generator. For the details, you can refer to Chapter 6 IP Generation.

RESET is valid for both latch and output register, so when RESET is enabled, port outputs 0 in both pipeline mode and bypass mode.

Figure 4-2, Figure 4-3, Figure 4-4, and Figure 4-5 show the reset timing in different modes. DO_RAM indicates to the data in the memory array; DO indicates to the output port data.

Register output modes are as follows:

- When synchronous reset is enabled, DO reset to 0 at CLK rising edge.
- When asynchronous reset is enabled, DO reset to 0 accordingly, no need to wait for the CLK rising edge.
- Reset invalid, and when OCE is valid, DO outputs DO_RAM.
- Reset invalid, and when OCE is invalid, DO retains the previous output data.

Bypass modes are as follows:

- When synchronous reset is enabled, DO reset to 0 at CLK rising edge.
- When asynchronous reset is enabled, DO is reset to 0 accordingly, no need to wait for the CLK rising edge.
- Reset invalid, DO outputs DO_RAM regardless of whether OCE is valid or not.

Figure 4-2 Synchronous Reset Timing Diagram (Pipeline Mode)

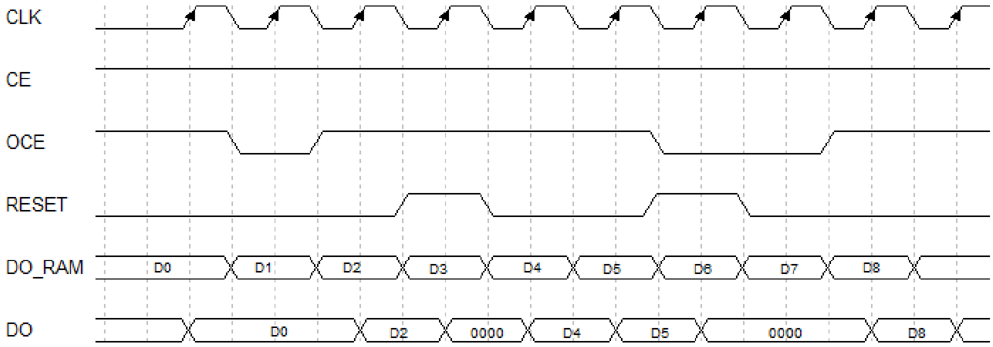


Figure 4-3 Synchronous Reset Timing Diagram (Bypass Mode)

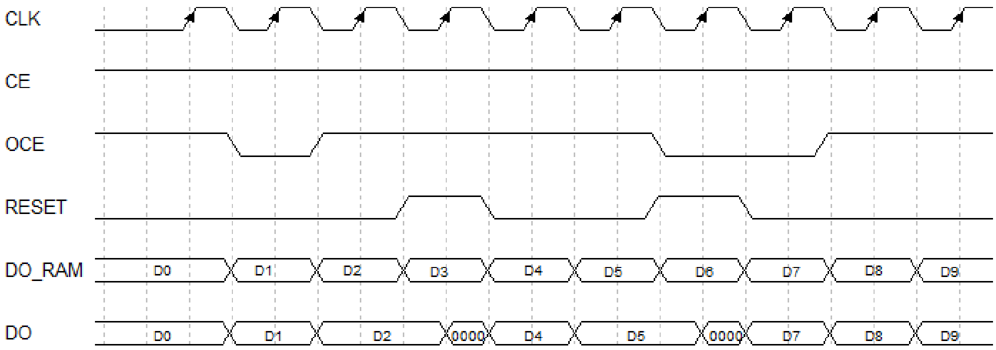


Figure 4-4 Asynchronous Reset Timing Diagram (Pipeline Mode)

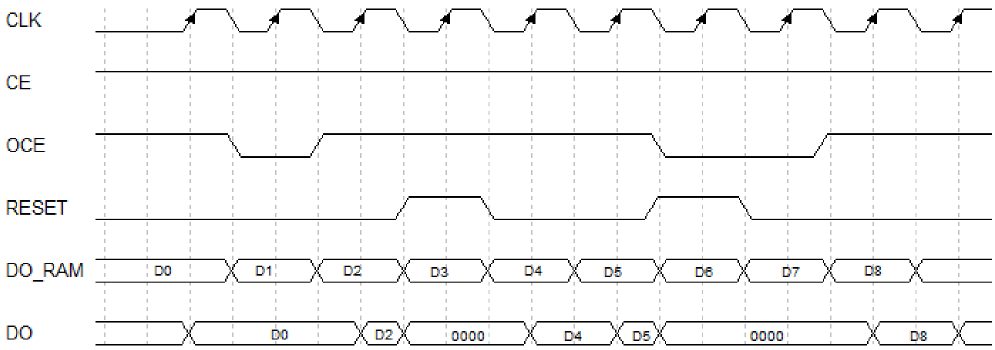
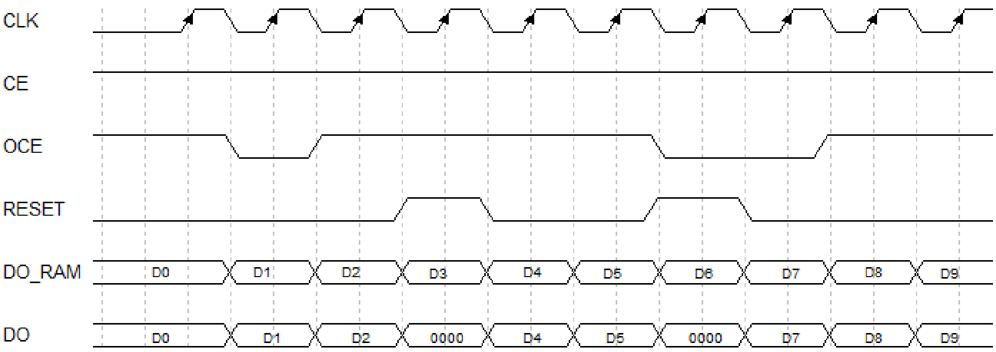


Figure 4-5 Asynchronous Reset Timing Diagram (Bypass Mode)



5 SSRAM Primitive

The Shadow SRAM can be configured as Single Port mode, Semi Dual Port mode and Read Only mode, as shown in Table 5-1.

Table 5-1 SSRAM Modes

Primitive	Description
RAM16S1	Single port SSRAM with 16 address depth and 1-bit data width
RAM16S2	Single port SSRAM with 16 address depth and 2-bit data width
RAM16S4	Single port SSRAM with 16 address depth and 4-bit data width
RAM16SDP1	Semi dual port SSRAM with 16 address depth and 1-bit data width
RAM16SDP2	Semi dual port SSRAM with 16 address depth and 2-bit data width
RAM16SDP4	Semi dual port SSRAM with 16 address depth and 4-bit data width
ROM16	ROM with 16 address depth and 1-bit data width

Note!

The GW5AST-138B, GW5A-138B, GW5AT-138B, GW5AS-138B, GW5AT-75B, GW5A-25A, GW5AS-25A, and GW5AR-25A devices do not support the RAM16S1, RAM16S2, RAM16S4, RAM16SDP1, RAM16SDP2, and RAM16SDP4 primitives.

5.1 RAM16S1

Primitive Introduction

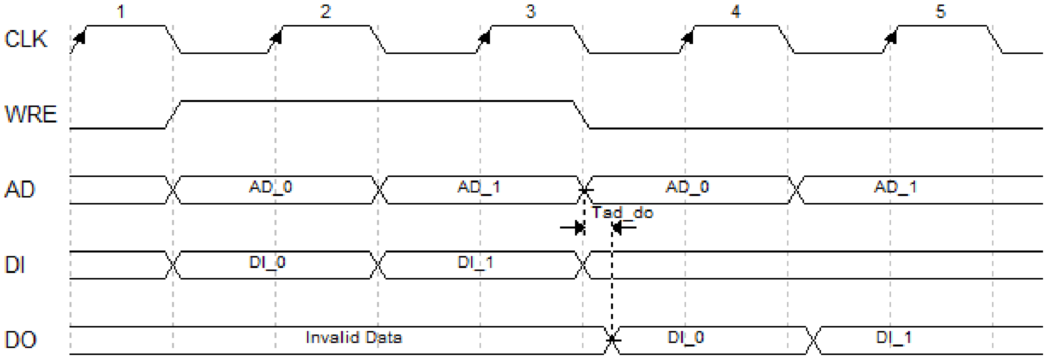
RAM16S1 (16-Deep by 1-Wide Single-port SSRAM) is a single port SSRAM with 16 address depth and 1-bit data width.

Functional Description

RAM16S1 is a single port SSRAM with 1-bit data width. The read and write addresses are the same. And the write operation is performed when WRE is high, at which time the data is loaded into the corresponding address of the memory at the rising edge of CLK. Read operation reads the data specified by the address corresponding to the data stored in RAM. That is, SSRAM is implemented by the LUT configuration of CFU, with synchronous write and asynchronous read. However, if the application requires synchronous read, you can use the registers associated with each LUT to implement the synchronous read function.

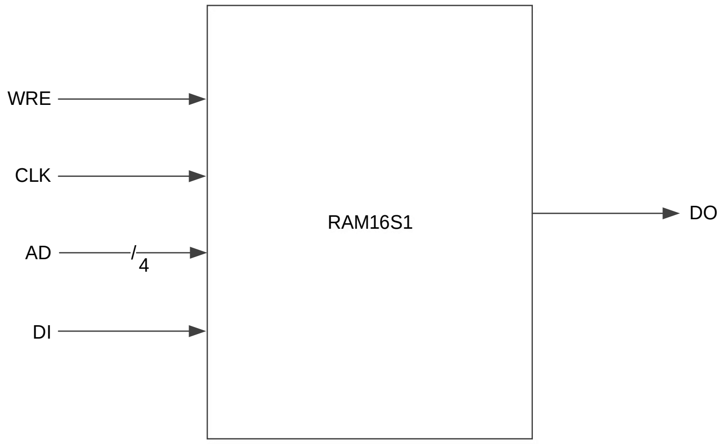
The timing diagram in normal mode is as shown in Figure 5-1.

Figure 5-1 RAM16S1 Timing Diagram



Port Diagram

Figure 5-2 RAM16S1 Port Diagram



Port Description

Table 5-2 RAM16S1 Port Description

Port	I/O	Description
DI	Input	Data input signal
CLK	Input	Clock input signal
WRE	Input	Write enable input signal
AD[3:0]	Input	Address input signal
DO	Output	Data output signal

Parameter Description

Table 5-3 RAM16S1 Parameter Description

Parameter	Range	Default Value	Description
INIT_0	16'h0000~16'hffff	16'h0000	RAM16S1 initialization value

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation.

Verilog Instantiation:

```
RAM16S1 instName(
        .DI(DI),
        .WRE(WRE),
        .CLK(CLK),
        .AD(AD[3:0]),
        .DO(DOUT)
);
defparam instName.INIT_0=16'h1100;
```

Vhdl Instantiation:

```
COMPONENT RAM16S1
        GENERIC (INIT:bit_vector:=X"0000");
        PORT(
                DO:OUT std_logic;
                DI:IN std_logic;
                CLK:IN std_logic;
                WRE:IN std_logic;
                AD:IN std_logic_vector(3 downto 0)
        );
END COMPONENT;
uut:RAM16S1
        GENERIC MAP(INIT=>X"0000")
        PORT MAP (
            DO=>DOUT,
            DI=>DI,
            CLK=>CLK,
            WRE=>WRE,
            AD=>AD
        );
```

5.2 RAM16S2

Primitive Introduction

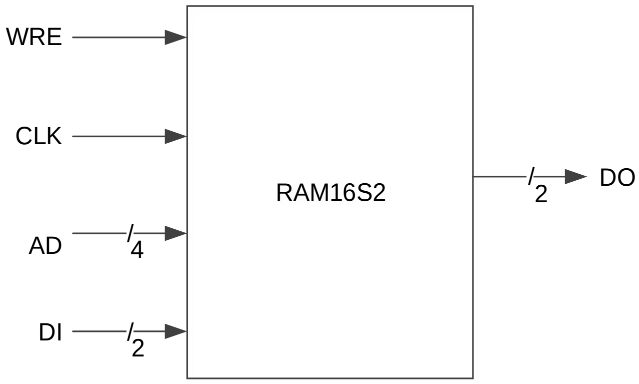
RAM16S2 (16-Deep by 2-Wide Single-port SSRAM) is a single port SSRAM with 16 address depth and 2-bit data width.

Functional Description

RAM16S2 is a single port SSRAM with 2-bit data width. The read and write addresses are the same. And the write operation is performed when WRE is high, at which time the data is loaded into the corresponding address of the memory at the rising edge of CLK. Read operation reads the data specified by the address corresponding to the data stored in RAM. That is, SSRAM is implemented by the LUT configuration of CFU, with synchronous write and asynchronous read. However, if the application requires synchronous read, you can use the registers associated with each LUT to implement the synchronous read function. The timing diagram is as shown in Figure 5-1.

Port Diagram

Figure 5-3 RAM16S2 Port Diagram



Port Description

Table 5-4 RAM16S2 Port Description

Port	I/O	Description
DI[1:0]	Input	Data input signal
CLK	Input	Clock input signal
WRE	Input	Write enable input signal
AD[3:0]	Input	Address input signal
DO[1:0]	Output	Data output signal

Parameter Description

Table 5-5 RAM16S2 Parameter Description

Parameter	Range	Default Value	Description
INIT_0~INIT_1	16'h0000~16'hffff	16'h0000	RAM16S2 initialization value

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation.

Verilog Instantiation:

```
RAM16S2 instName(
        .DI(DI[1:0]),
        .WRE(WRE),
        .CLK(CLK),
        .AD(AD[3:0]),
        .DO(DOUT[1:0])
);
defparam instName.INIT_0=16'h0790;
defparam instName.INIT_1=16'h0f00;
```

Vhdl Instantiation:

```
COMPONENT RAM16S2
        GENERIC (INIT_0:bit_vector:=X"0000";
                 INIT_1:bit_vector:=X"0000"
        );
        PORT(
                DO:OUT std_logic_vector(1 downto 0);
                DI:IN std_logic_vector(1 downto 0);
                CLK:IN std_logic;
                WRE:IN std_logic;
                AD:IN std_logic_vector(3 downto 0)
        );
END COMPONENT;
uut:RAM16S2
        GENERIC MAP(INIT_0=>X"0000",
                    INIT_1=>X"0000"
        )
        PORT MAP (
```

```
            DO=>DOUT,
            DI=>DI,
            CLK=>CLK,
            WRE=>WRE,
            AD=>AD
        );
```

5.3 RAM16S4

Primitive Introduction

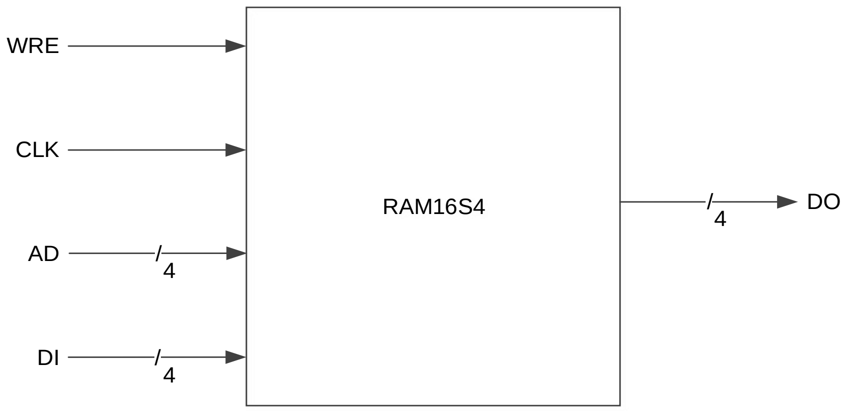
RAM16S4 (16-Deep by 4-Wide Single-port SSRAM) is a single port SSRAM with 16 address depth and 4-bit data width.

Functional Description

RAM16S4 is a single port SSRAM with 4-bit data width. The read and write addresses are the same. And the write operation is performed when WRE is high, at which time the data is loaded into the corresponding address of the memory at the rising edge of CLK. Read operation reads the data specified by the address corresponding to the data stored in RAM. That is, SSRAM is implemented by the LUT configuration of CFU, with synchronous write and asynchronous read. However, if the application requires synchronous read, you can use the registers associated with each LUT to implement the synchronous read function. The timing diagram is as shown in Figure 5-1.

Port Diagram

Figure 5-4 RAM16S4 Port Diagram



Port Description

Table 5-6 RAM16S4 Port Description

Port	I/O	Description
DI[3:0]	Input	Data input signal
CLK	Input	Clock input signal
WRE	Input	Write enable input signal
AD[3:0]	Input	Address input signal
DO[3:0]	Output	Data output signal

Parameter Description

Table 5-7 RAM16S4 Parameter Description

Parameter	Range	Default Value	Description
INIT_0~INIT_3	16'h0000~16'hffff	16'h0000	RAM16S4 initialization value

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation.

Verilog Instantiation:

```
RAM16S4 instName(
    .DI(DI[3:0]),
    .WRE(WRE),
    .CLK(CLK),
    .AD(AD[3:0]),
    .DO(DOUT[3:0])
);
defparam instName.INIT_0=16'h0450;
defparam instName.INIT_1=16'h1ac3;
defparam instName.INIT_2=16'h1240;
defparam instName.INIT_3=16'h045c;
```

Vhdl Instantiation:

```
COMPONENT RAM16S4
        GENERIC (INIT_0:bit_vector:=X"0000";
                 INIT_1:bit_vector:=X"0000";
                 INIT_2:bit_vector:=X"0000";
                 INIT_3:bit_vector:=X"0000"
```

```
        );
         PORT(
                DO:OUT std_logic_vector(3 downto 0);
                DI:IN std_logic_vector(3 downto 0);
                CLK:IN std_logic;
                WRE:IN std_logic;
                AD:IN std_logic_vector(3 downto 0)
        );
END COMPONENT;
uut:RAM16S4
        GENERIC MAP(INIT_0=>X"0000",
                    INIT_1=>X"0000",
                    INIT_2=>X"0000",
                    INIT_3=>X"0000"
        )
        PORT MAP (
            DO=>DOUT,
            DI=>DI,
            CLK=>CLK,
            WRE=>WRE,
            AD=>AD
        );
```

5.4 RAM16SDP1

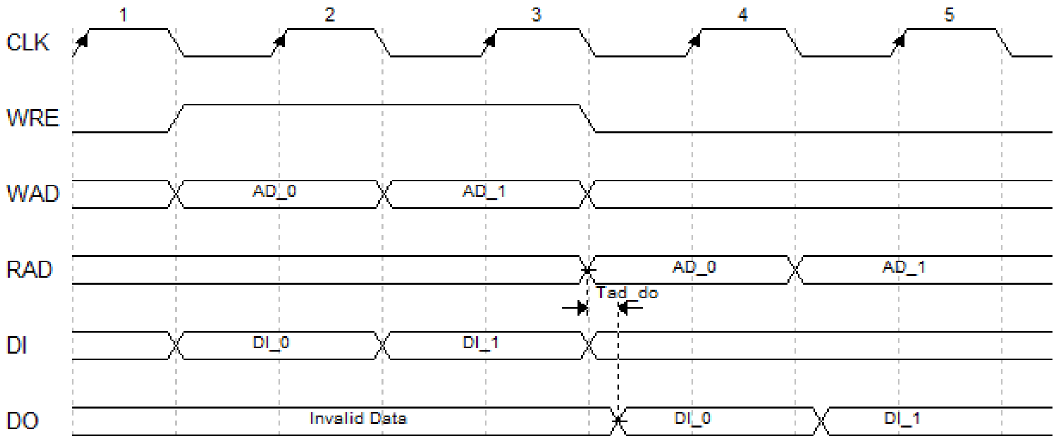
Primitive Introduction

RAM16SDP1 (16-Deep by 1-Wide Semi Dual-port SSRAM) is a semi dual port SSRAM with 16 address depth and 1-bit data width.

Functional Description

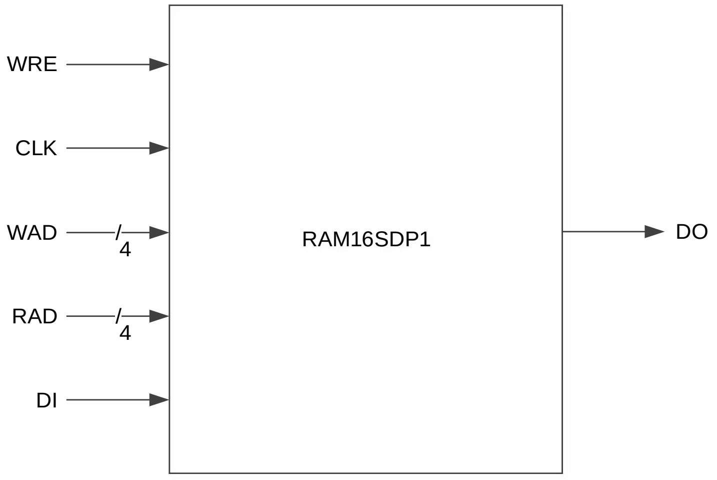
RAM16SDP1 is a semi dual port SSRAM with 1-bit data width. It has two asynchronous addresses, write address WAD and read address RAD. And the write operation is performed when WRE is high, at which time the data is loaded into the corresponding write address of the memory at the rising edge of CLK. The read operation is based on the read address to determine the data in the corresponding location of the output RAM. The timing diagram in normal mode is as shown in Figure 5-5.

Figure 5-5 RAM16SDP1 Timing Diagram



Port Diagram

Figure 5-6 RAM16SDP1 Port Diagram



Port Description

Table 5-8 RAM16SDP1 Port Description

Port	I/O	Description
DI	Input	Data input signal
CLK	Input	Clock input signal
WRE	Input	Write enable input signal
WAD[3:0]	Input	Write address signal
RAD[3:0]	Input	Read address signal
DO	Output	Data output signal

Parameter Description

Table 5-9 RAM16SDP1 Parameter Description

Parameter	Range	Default Value	Description
INIT_0	16'h0000~16'hffff	16'h0000	RAM16SDP1 initialization value

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation.

Verilog Instantiation:

```
RAM16SDP1 instName(
    .DI(DI),
    .WRE(WRE),
    .CLK(CLK),
    .WAD(WAD[3:0]),
    .RAD(RAD[3:0]),
    .DO(DOUT)
);
defparam instName.INIT_0=16'h0100;
```

Vhdl Instantiation:

```
COMPONENT RAM16SDP1
        GENERIC (INIT_0:bit_vector:=X"0000");
        PORT(
                DO:OUT std_logic;
                DI:IN std_logic;
                CLK:IN std_logic;
                WRE:IN std_logic;
                WAD:IN std_logic_vector(3 downto 0);
                RAD:IN std_logic_vector(3 downto 0)
        );
END COMPONENT;
uut:RAM16SDP1
        GENERIC MAP(INIT_0=>X"0000")
        PORT MAP (
            DO=>DOUT,
            DI=>DI,
```

```
        CLK=>CLK,
        WRE=>WRE,
        WAD=>WAD,
        RAD=>RAD
    );
```

5.5 RAM16SDP2

Primitive Introduction

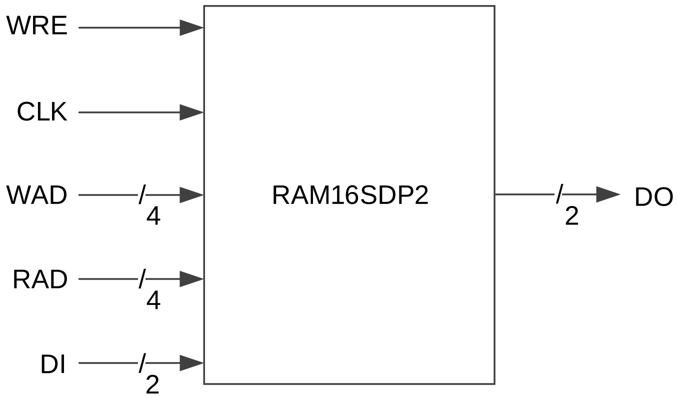
RAM16SDP2 (16-Deep by 2-Wide Semi Dual-port SSRAM) is a semi dual port SSRAM with 16 address depth and 2-bit data width.

Functional Description

RAM16SDP2 is a semi dual port SSRAM with 1-bit data width. It has two asynchronous addresses, write address WAD and read address RAD. And the write operation is performed when WRE is high, at which time the data is loaded into the corresponding write address of the memory at the rising edge of CLK. The read operation is based on the read address to determine the data in the corresponding location of the output RAM. The timing diagram is as shown in Figure 5-5.

Port Diagram

Figure 5-7 RAM16SDP2 Port Diagram



Port Description

Table 5-10 RAM16SDP2 Port Description

Port	I/O	Description
DI[1:0]	Input	Data input signal
CLK	Input	Clock input signal
WRE	Input	Write enable input signal
WAD[3:0]	Input	Write address signal
RAD[3:0]	Input	Read address signal
DO[1:0]	Output	Data output signal

Parameter Description

Table 5-11 RAM16SDP2 Parameter Description

Parameter	Range	Default Value	Description
INIT_0~INIT_1	16'h0000~16'hffff	16'h0000	RAM16SDP2 initialization value

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation.

Verilog Instantiation:

```
RAM16SDP2 instName(
    .DI(DI[1:0]),
    .WRE(WRE),
    .CLK(CLK),
    .WAD(WAD[3:0]),
    .RAD(RAD[3:0]),
    .DO(DOUT[1:0])
);
defparam instName.INIT_0=16'h5600;
defparam instName.INIT_1=16'h0af0;
```

Vhdl Instantiation:

```
COMPONENT RAM16SDP2
        GENERIC (INIT_0:bit_vector:=X"0000";
                 INIT_1:bit_vector:=X"0000"
       );
        PORT(
             DO:OUT std_logic_vector(1 downto 0);
             DI:IN std_logic_vector(1 downto 0);
             CLK:IN std_logic;
             WRE:IN std_logic;
             WAD:IN std_logic_vector(3 downto 0);
             RAD:IN std_logic_vector(3 downto 0)
       );
END COMPONENT;
uut:RAM16SDP2
       GENERIC MAP(INIT_0=>X"0000",
```

```
            INIT_1=>X"0000"
)
 PORT MAP (
      DO=>DOUT,
      DI=>DI,
      CLK=>CLK,
      WRE=>WRE,
      WAD=>WAD,
      RAD=>RAD
);
```

5.6 RAM16SDP4

Primitive Introduction

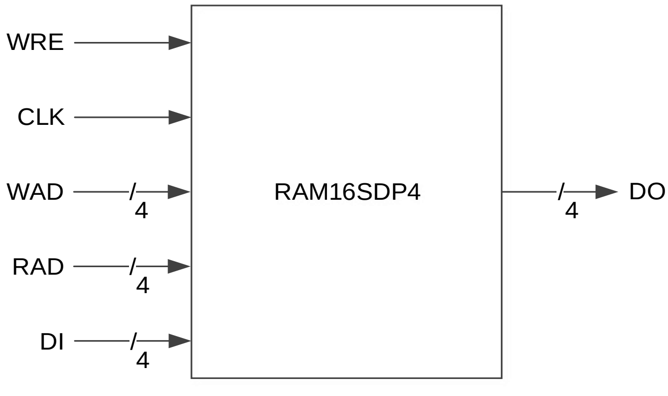
RAM16SDP4 (16-Deep by 4-Wide Semi Dual-port SSRAM) is a semi dual port SSRAM with 16 address depth and 4-bit data width.

Functional Description

RAM16SDP4 is a semi dual port SSRAM with 4-bit data width. It has two asynchronous addresses, write address WAD and read address RAD. And the write operation is performed when WRE is high, at which time the data is loaded into the corresponding write address of the memory at the rising edge of CLK. The read operation is based on the read address to determine the data in the corresponding location of the output RAM. The timing diagram is as shown in Figure 5-5.

Port Diagram

Figure 5-8 RAM16SDP4 Port Diagram



Port Description

Table 5-12 RAM16SDP4 Port Description

Port	I/O	Description
DI[3:0]	Input	Data input signal
CLK	Input	Clock input signal
WRE	Input	Write enable input signal
WAD[3:0]	Input	Write address signal
RAD[3:0]	Input	Read address signal
DO[3:0]	Output	Data output signal

Parameter Description

Table 5-13 RAM16SDP4 Parameter Description

Parameter	Range	Default Value	Description
INIT_0~INIT_3	16'h0000~16'hffff	16'h0000	RAM16SDP4 initialization value

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation.

Verilog Instantiation:

```
RAM16SDP4 instName(
    .DI(DI[3:0]),
    .WRE(WRE),
    .CLK(CLK),
    .WAD(WAD[3:0]),
    .RAD(RAD[3:0]),
    .DO(DOUT[3:0])
);
defparam instName.INIT_0=16'h0340;
defparam instName.INIT_1=16'h9065;
defparam instName.INIT_2=16'hac12;
defparam instName.INIT_3=16'h034c;
```

Vhdl Instantiation:

```
COMPONENT RAM16SDP2
        GENERIC (INIT_0:bit_vector:=X"0000";
                 INIT_1:bit_vector:=X"0000";
                 INIT_2:bit_vector:=X"0000";
                 INIT_3:bit_vector:=X"0000";
```

```
        );
        PORT(
                DO:OUT std_logic_vector(3 downto 0);
                DI:IN std_logic_vector(3 downto 0);
                CLK:IN std_logic;
                WRE:IN std_logic;
                WAD:IN std_logic_vector(3 downto 0);
                RAD:IN std_logic_vector(3 downto 0)
        );
END COMPONENT;
uut:RAM16SDP2
        GENERIC MAP(INIT_0=>X"0000",
                    INIT_1=>X"0000",
                    INIT_2=>X"0000",
                    INIT_3=>X"0000"
        )
        PORT MAP (
            DO=>DOUT,
            DI=>DI,
            CLK=>CLK,
            WRE=>WRE,
            WAD=>WAD,
            RAD=>RAD
        );
```

5.7 ROM16

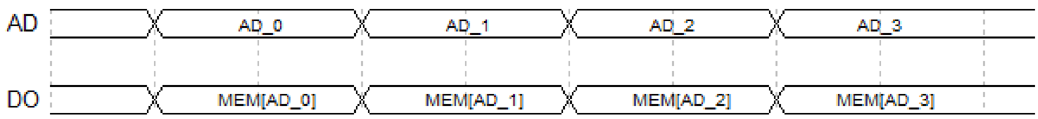
Primitive Introduction

ROM16 is a read only memory with 16 address depth and 1-bit data width. The memory is initialized using INIT.

Functional Description

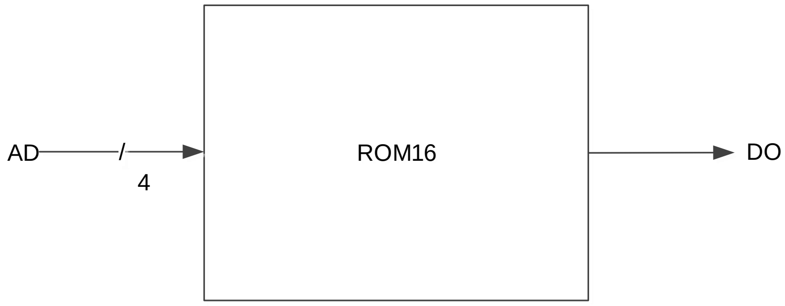
ROM16 is the 1-bit read only memory and read operation reads the data specified by the address corresponding to the data stored in ROM. The timing diagram is as shown in Figure 5-9.

Figure 5-9 ROM16 Timing Diagram



Port Diagram

Figure 5-10 ROM16 Port Diagram



Port Description

Table 5-14 ROM16 Port Description

Port	I/O	Description
AD[3:0]	Input	Address input signal
DO	Output	Data output signal

Parameter Description

Table 5-15 ROM16 Parameter Description

Parameter	Range	Default Value	Description
INIT_0	16'h0000~16'hffff	16'h0000	ROM16 initialization value

Primitive Instantiation

The primitives can be instantiated directly, or generated by the IP Core Generator. For the details, you can refer to Chapter 6 IP Generation.

Verilog Instantiation:

```
ROM16 instName(
      .AD(AD[3:0]),
      .DO(DOUT)
);
defparam instName.INIT_0=16'hfc00;
```

Vhdl Instantiation:

```
COMPONENT ROM16
          GENERIC (INIT:bit_vector:=X"0000");
          PORT(
                DO:OUT std_logic;
                AD:IN std_logic_vector(3 downto 0)
```

```
    );
END COMPONENT;
uut:ROM16
    GENERIC MAP(INIT=>X"0000")
    PORT MAP (
        DO=>DOUT,
        AD=>AD
    );
```

6 IP Generation

The IP core generator that is integrated in the GOWIN FPGA Designer can be used to generate IP cores. You can set data width, address depth, and write/read mode to generate the corresponding IP modules. In addition, there are two more ways to implement BSRAM and SSRAM functions. You can call the library file of GOWIN FPGA Designer and set the ports and parameters to generate the required IP modules. You can also use Gowin Synthesis to synthesize BSRAM and SSRAM automatically during code synthesis.

In IP Core Generator, BSRAM module includes Single Port mode, Semi Dual Port mode, Semi Dual Port mode with ECC function, Dual Port mode, and Read Only mode; SSRAM module includes Single Port mode, Semi Dual Port mode, and Read Only mode. Here BSRAM is introduced in Dual Port mode and Semi Dual Port mode with ECC function; SSRAM is introduced in Single Port mode as an example for IP calls, other modes refer to BSRAM Dual Port mode and SSRAM Single Port mode.

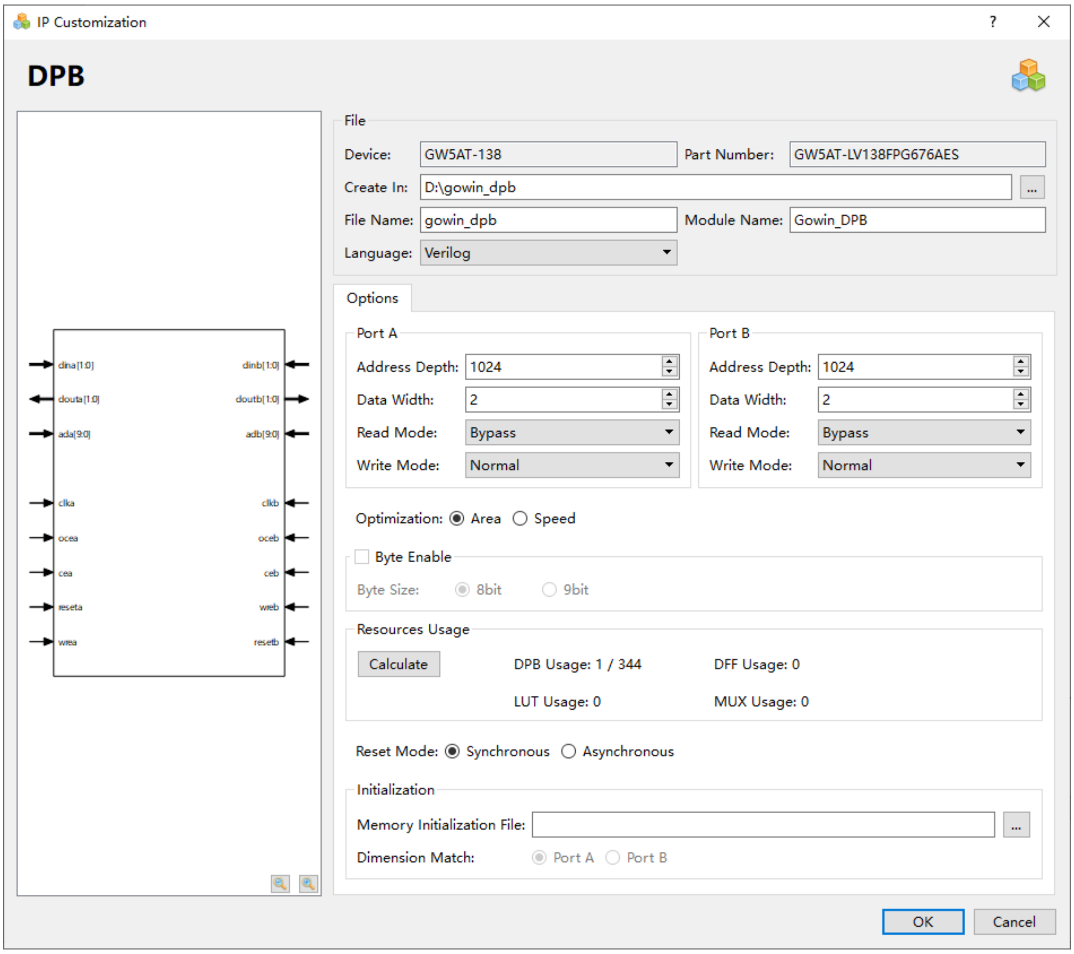
6.1 BSRAM Dual Port Mode

BSRAM Dual Port mode (DP) can be implemented by DPB and DPX9B primitives. Click "DPB" on the IP Core Generator, and a brief introduction to the DPB will be displayed.

IP Configuration

Double-click "DPB" to open the "IP Customization" window. This window includes the "File", "Options", port diagram, as shown in Figure 6-1.

Figure 6-1 IP Customization of DPB



1. File Configuration Box
   The File Configuration box is used to configure the information about the generated IP design file.

- Device: Displays information about the configured Device.
- Part Number: Display the configured Part Number.
- Language: Hardware description language used to generate the IP design files. Click the drop-down list to select the language, including Verilog and VHDL.
- Module Name: The module name of the generated IP design files. Enter the module name in the text box. Module name cannot be the same as the primitive name. If it is the same, an error will be reported.
- File Name: The name of the generated IP design files. Enter the file name in the text box.
- Create In: The path in which the generated IP files will be stored. Enter the target path in the box or select the target path by clicking the option.

2. Options Configurations
   The Options Configurations are used to configure IP by users. The

Dual Port mode includes Port A and Port B, as shown in Figure 6-1.

- Data Width & Address Depth: Configure address depth and data width. If the configuration cannot be implemented by one module, IP Core will instantiate multiple modules to implement the current configuration.
- Resource Usage: Calculate and display the resource usage of Block Ram, DFF, LUT, and MUX.
- Read/Write Mode: Configure Read/Write mode. DPB supports the following modes:
  - Two Read modes: Bypass and Pipeline
  - Three Write modes: Normal, Write-Through, and Read-before-Write
- Reset Mode: support synchronous or asynchronous
- Initialization: Configure the INIT value of SP. Initialization value is written in the initialization file in Binary, Hex or Address Hex formats. The initialization file can be generated by manual writing or clicking "File > New > Memory Initialization File" in the IDE menu bar. For the details, see *SUG100, GOWIN FPGA Designer User Guide*. For the format of initialization file, see Chapter 7 Initialization File.

Note!

- The address depth, data width, and read/write mode of DPB Port A and Port B can be configured independently.
- The address depth and data width of DPB Port A and Port B must be same because Port A and Port B read or write the same memory.
- The data width in the Memory Initialization File should be consistent with the data width of the port specified in the "Dimension Match".
- If the address depth*data width of Port A and Port B is different, an error prompt will pop up.
- If the data width is different, the Init value of generated DPB instance is 0 by default, and an Error will pop up:
Error (MG2105): Initial.'width is unequal to user's width.

3. Port Diagram

- The port diagram is based on the current IP Core configuration. The input/output bit-width updates in real time based on the "Options" configuration, as shown in Figure 6-1;
- "Address Depth" of port A and port B affects the bit-width of Address; "Data Width" affects the bit-width of input and output.

IP Generation Files

After configuration, it will generate three files that are named after the "File Name".

- "gowin_dpb.v" file is a complete Verilog module to generate instance DPB, and it is generated according to the IP configuration;
- "gowin_dpb_tmp.v" is the instance template file;

- "gowin_dpb.ipc" file is IP configuration file. The user can load the file to configure the IP.

Note!

If VHDL is selected as the hardware description language, the first two files will be named with .vhd suffix.

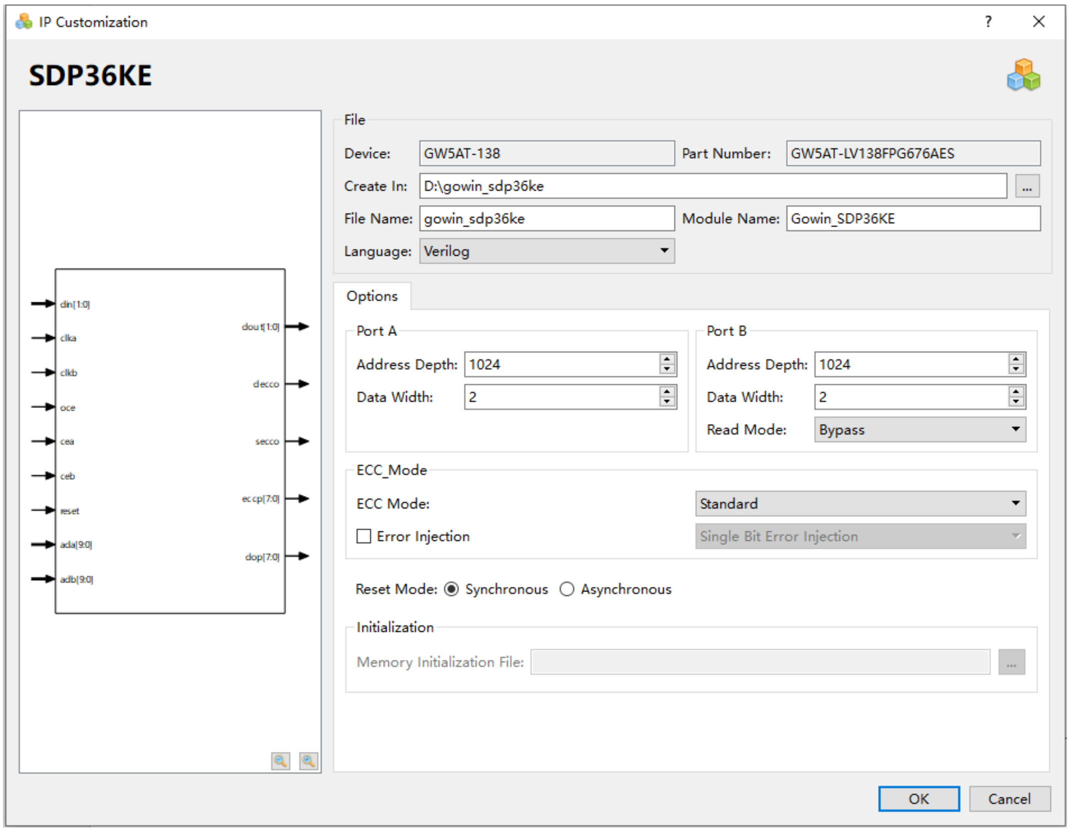
6.2 BSRAM Semi Dual Port Mode with ECC Function

BSRAM Semi Dual Port mode with ECC function (SDP36KE) can be implemented by SDP36KE primitives. Click "SDP36KE" on the IP Core Generator, and a brief introduction to "ALU54" will be displayed.

IP Configuration

Double-click "SDP36KE" in IP Core Generator to open the "IP Customization" window. This window includes the "File", "Options", port diagram, as shown in Figure 6-2.

Figure 6-2 IP Customization of SDP36KE



1. File Configuration Box
   The File Configuration box is used to configure the information about the generated IP design file. The SDP36KE file configuration is similar to that of BSRAM Dual Port mode. For the details, see the descriptions of file configuration box in 6.1 BSRAM Dual Port Mode .
2. Options Configuration
   The Options Configuration is used to configure IP by users. The Dual Port mode includes Port A and Port B, as shown in Figure 6-1.

- Data Width & Address Depth: Configure address depth and data width. If the configuration cannot be implemented by one module, IP Core will instantiate multiple modules to implement the current configuration.
- ECC Mode supports:
  - Standard: Supports Encode and Decode
  - Encode-Only: Only supports Encode
  - Decode-Only: Only supports Decode
- Error Injection: Configure the bit of error injection SDP36KE supports the following bits of error injection
  - Single Bit Error Injection: Inject 1-bit error
  - Double Bit Error Injection: Inject 2-bit error
  - Single and Double Bit Error Injection: Inject 1-bit and 2-bit error
- Reset Mode: support synchronous or asynchronous

3. Port Diagram

- The port diagram is based on the current IP Core configuration. The input/output bit-width updates in real time based on the "Options" configuration, as shown in Figure 6-2.
- "Address Depth" of port A and port B affects the bit-width of Address; "Data Width" affects the bit-width of input and output.

IP Generation Files

After configuration, it will generate three files that are named after the "File Name".

- "gowin_sdp36ke.v" file is a complete Verilog module to generate instance I3C, and it is generated according to the IP configuration;
- "gowin_sdp36ke_tmp.v" is the instance template file;
- "gowin_sdp36ke.ipc" file is IP configuration file. The user can load the file to configure the IP.

Note!

If VHDL is selected as the hardware description language, the first two files will be named with a .vhd suffix.

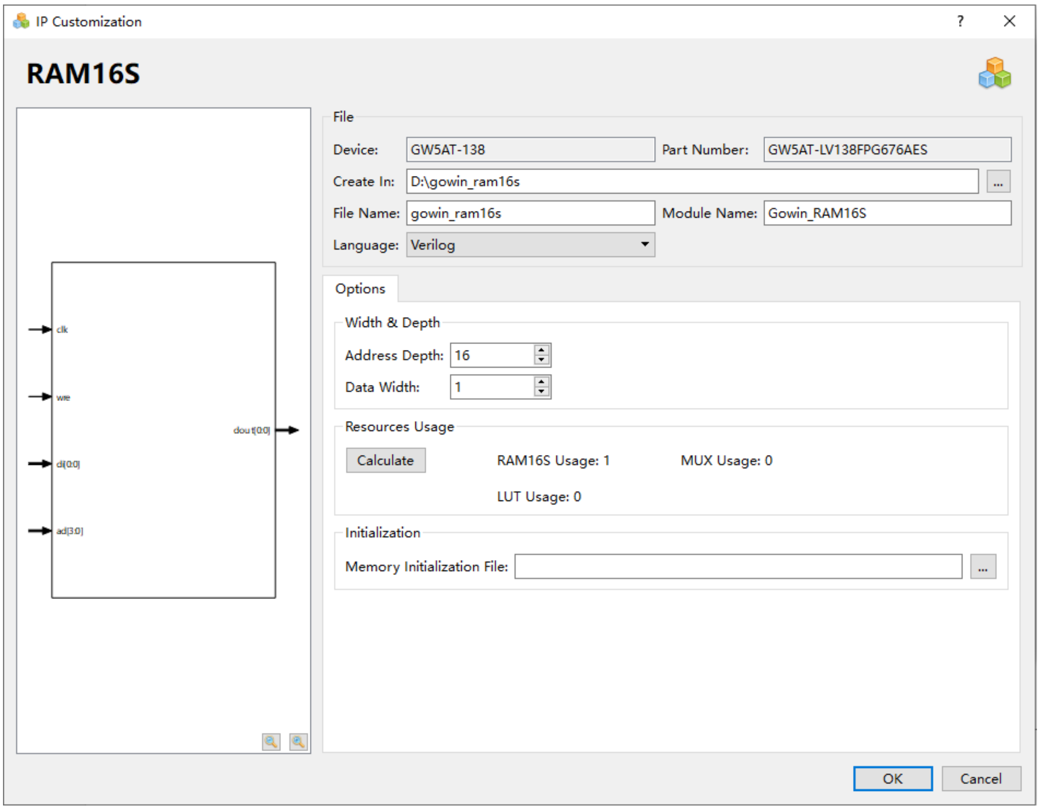
6.3 SSRAM Single Port Mode

RAM16S is a single port SSRAM that can be implemented by RAM16S1, RAM16S2, and RAM16S4 primitives. Click "RAM16S" on the IP Core Generator, and a brief introduction to RAM16S will be displayed.

IP Configuration

Double-click "RAM16S" to open the "IP Customization" window. This window includes the "File", "Options", port diagram, as shown in Figure 6-3.

Figure 6-3 IP Customization of RAM16S



1. File Configuration Box
   The File Configuration box is used to configure the information about the generated IP design file. The RAM16S file configuration is similar to that of BSRAM Dual Port mode. For the details, see see the descriptions of file configuration box in 6.1 BSRAM Dual Port Mode.
2. Options Configurations
   The Options is used to configure IPs by users. The Options configuration is as shown in . The RAM16S Options configuration is similar to that of BSRAM Dual Port mode. For the details, see the descriptions of options configurations in 6.1 BSRAM Dual Port Mode.
3. Port Diagram

- The port diagram is based on the current IP Core configuration. The input/output bit-width updates in real time based on the "Options" configuration, as shown in Figure 6-3;

- "Address Depth" affects the bit-width of Address; "Data Width" affects the bit-width of the input and output.

IP Generation Files

After configuration, it will generate three files that are named after the "File Name".

- "gowin_ram16s.v" file is a complete Verilog module to generate instance ROM16S, and it is generated according to the IP configuration;
- "gowin_ram16s_tmp.v" is the instance template file;
- "gowin_ram16s.ipc" file is IP configuration file. The user can load the file to configure the IP.

Note!

If VHDL is selected as the hardware description language, the first two files will be named with a .vhd suffix.

7 Initialization File

In BSRAM and SSRAM modes, you can initialize each bit in memory to 0 or 1. Initialization value is written in the initialization file in Binary, Hex or Address Hex formats.

7.1 Bin File

Bin file is a text file that consists of the 0 and 1 binary numbers. The line represents address depth, and the column represents data width.

```
#File_format=Bin
#Address_depth=16
#Data_width=32
00001100000100000000100100010000
10000000010010000100000001000000
01000000100000001000000010000000
001000000100001001100000011000000
```

7.2 Hex File

The Hex file is similar to the Bin file. It consists of hexadecimal numbers 0~F. The line represents the address depth, and the binary bits in each line represents data width.

```
#File_format=Hex
#Address_depth=8
#Data_width=16
3A40
A28E
0B52
1C49
D602
0801
```

03E6

4C18

7.3 Address-Hex File

Address-Hex file is the file that includes both the data and the address with data record. The address and the data is composed of the hexadecimal number of 0~F. In each line, the address is located before the colon, and the data is located after the colon. The address with no data record is 0 by default.

#File_format=AddrHex

#Address_depth=256

#Data_width=16

9:FFFF

23:00E0

2a:001F

30: 1E00

PROGRAMMING FOR THE FUTURE