



# Gowin 存储器(B-SRAM & S-SRAM) 用户指南

UG285-1.2,2020-08-17

## **版权所有© 2020 广东高云半导体科技股份有限公司**

未经本公司书面许可，任何单位和个人不得擅自摘抄、复制、翻译本档内容的部分或全部，并不得以任何形式传播。

### **免责声明**

本档并未授予任何知识产权的许可，并未以明示或暗示，或以禁止发言或其它方式授予任何知识产权许可。除高云半导体在其产品的销售条款和条件中声明的责任之外，高云半导体概不承担任何法律或非法律责任。高云半导体对高云半导体产品的销售和 / 或使用不作任何明示或暗示的担保，包括对产品的特定用途适用性、适销性或对任何专利权、版权或其它知识产权的侵权责任等，均不作担保。高云半导体对档中包含的文字、图片及其它内容的准确性和完整性不承担任何法律或非法律责任，高云半导体保留修改档中任何内容的权利，恕不另行通知。高云半导体不承诺对这些档进行适时的更新。

## 版本信息

日期	版本	说明
2016/05/17	1.05	初始版本。
2016/07/15	1.06	标准化插图。
2016/10/27	1.07	适用 GW2AR 系列 FPGA 产品。
2017/05/03	1.08	<ul style="list-style-type: none"><li>● 更新 B-SRAM 操作时序图，添加 ROM、字节使能信号、字节校验、上电情况、输出寄存器复位、位置约束；</li><li>● 增加附录 A 注意事项。</li></ul>
2018/05/31	1.09	<ul style="list-style-type: none"><li>● 添加第三章端口和参数介绍；</li><li>● 添加存储扩展；</li><li>● 更新 A.3 读写注意。</li></ul>
2019/04/03	1.1	更新表 A.1 写操作注意列表。
2020/08/17	1.2	章节调整及内容优化。

# 目录

目录 .....	<b>i</b>
图目录 .....	<b>iii</b>
表目录 .....	<b>v</b>
<b>1 关于本手册 .....</b>	<b>1</b>
1.1 手册内容 .....	1
1.2 相关文档 .....	1
1.3 术语、缩略语 .....	1
1.4 技术支持与反馈 .....	2
<b>2 概述 .....</b>	<b>3</b>
2.1 B-SRAM 特性介绍 .....	3
2.2 B-SRAM 配置模式 .....	3
<b>3 B-SRAM 原语 .....</b>	<b>6</b>
3.1 双端口模式 .....	6
3.2 单端口模式 .....	18
3.3 伪双端口模式 .....	22
3.4 只读模式 .....	28
<b>4 B-SRAM 输出复位 .....</b>	<b>32</b>
<b>5 S-SRAM 原语 .....</b>	<b>35</b>
5.1 RAM16S1 .....	35
5.2 RAM16S2 .....	38
5.3 RAM16S4 .....	40
5.4 RAM16SDP1 .....	42
5.5 RAM16SDP2 .....	44
5.6 RAM16SDP4 .....	47
5.7 ROM16 .....	49
<b>6 IP 调用 .....</b>	<b>51</b>
6.1 B-SRAM 双端口模式 .....	51
6.2 S-SRAM 单端口模式 .....	54

---

<b>7 初始化文件 .....</b>	<b>56</b>
7.1 二进制格式 (Bin File) .....	56
7.2 十六进制格式 (Hex File) .....	56
7.3 带地址十六进制格式 (Address-Hex File) .....	57

# 图目录

图 3-1 DPB/DPX9B Normal 写模式时序波形图 (Bypass 读模式) .....	7
图 3-2 DPB/DPX9B Normal 写模式时序波形图 (Pipeline 读模式) .....	8
图 3-3 DPB/DPX9B Write-through 写模式时序波形图 (Bypass 读模式) .....	9
图 3-4 DPB/DPX9B Write-through 写模式时序波形图 (Pipeline 读模式) .....	10
图 3-5 DPB/DPX9B Read-before-write 写模式时序波形图 (Bypass 读模式) .....	11
图 3-6 DPB/DPX9B Read-before-write 写模式时序波形图 (Pipeline 读模式) .....	12
图 3-7 DPB/DPX9B 端口示意图 .....	13
图 3-8 SP/SPX9 端口示意图 .....	19
图 3-9 伪双端口 B-SRAM Normal 写模式时序波形图 (Bypass 读模式) .....	23
图 3-10 伪双端口 B-SRAM Normal 写模式时序波形图 (Pipeline 读模式) .....	23
图 3-11 SDPB/SDPX9B 端口示意图 .....	24
图 3-12 只读 ROM 时序波形图 (Bypass 模式) .....	28
图 3-13 只读 ROM 时序波形图 (Pipeline 模式) .....	28
图 3-14 pROM/pROMX9 端口示意图 .....	29
图 4-1 复位输出结构框图 .....	32
图 4-2 同步复位时序图 (Pipeline 模式) .....	33
图 4-3 同步复位时序图 (Bypass 输出模式) .....	33
图 4-4 异步复位时序图 (Pipeline 模式) .....	33
图 4-5 异步复位时序图 (Bypass 输出模式) .....	34
图 5-1 RAM16S1 模式时序波形图 .....	36
图 5-2 RAM16S1 端口示意图 .....	36
图 5-3 RAM16S2 端口示意图 .....	38
图 5-4 RAM16S4 端口示意图 .....	40
图 5-5 RAM16SDP1 模式时序波形图 .....	42
图 5-6 RAM16SDP1 端口示意图 .....	43
图 5-7 RAM16SDP2 端口示意图 .....	45
图 5-8 RAM16SDP4 端口示意图 .....	47
图 5-9 ROM16 模式时序波形图 .....	49
图 5-10 ROM16 端口示意图 .....	49

图 6-1 DPB 的 IP Customization 窗口结构 ..... 52

图 6-2 RAM16S 的 IP Customization 窗口结构 ..... 54

# 表目录

表 1-1 术语、缩略语 .....	1
表 2-1 B-SRAM 配置模式列表 .....	3
表 2-2 B-SRAM 数据和地址位宽对应关系 .....	4
表 2-3 双端口模式数据宽度配置列表 .....	4
表 2-4 伪双端口模式数据宽度配置列表 .....	5
表 3-1 DPB/DPX9B 数据宽度和地址深度配置关系 .....	12
表 3-2 DPB/DPX9B 端口介绍 .....	13
表 3-3 DPB/DPX9B 参数介绍 .....	14
表 3-4 SP/SPX9 数据宽度和地址深度配置关系 .....	18
表 3-5 SP/SPX9 端口介绍 .....	19
表 3-6 SP/SPX9 参数介绍 .....	19
表 3-7 SDPB/SDPX9B 数据宽度和地址深度配置关系 .....	23
表 3-8 SDPB/SDPX9B 端口介绍 .....	24
表 3-9 SDPB/SDPX9B 参数介绍 .....	25
表 3-10 pROM/pROMX9 配置关系 .....	29
表 3-11 pROM/pROMX9 端口介绍 .....	29
表 3-12 pROM/pROMX9 参数介绍 .....	30
表 5-1 S-SRAM 模式 .....	35
表 5-2 RAM16S1 端口介绍 .....	36
表 5-3 RAM16S1 参数介绍 .....	37
表 5-4 RAM16S2 端口介绍 .....	38
表 5-5 RAM16S2 参数介绍 .....	38
表 5-6 RAM16S4 端口介绍 .....	40
表 5-7 RAM16S4 参数介绍 .....	41
表 5-8 RAM16SDP1 端口介绍 .....	43
表 5-9 RAM16SDP1 参数介绍 .....	43
表 5-10 RAM16SDP2 端口介绍 .....	45
表 5-11 RAM16SDP2 参数介绍 .....	45
表 5-12 RAM16SDP4 端口介绍 .....	47



---

表 5-13 RAM16SDP4 参数介绍.....	47
表 5-14 ROM16 端口介绍.....	50
表 5-15 ROM16 参数介绍.....	50

# 1 关于本手册

## 1.1 手册内容

Gowin 存储器（B-SRAM&S-SRAM）用户手册主要描述高云半导体 B-SRAM 和 S-SRAM 的特性、工作模式、原语介绍、IP 调用等旨在给用户 提供应用说明。

## 1.2 相关文档

通过登录高云半导体网站 [www.gowinsemi.com.cn](http://www.gowinsemi.com.cn) 可以下载、查看以下 相关文档：

- [DS100](#), GW1N 系列 FPGA 产品数据手册
- [DS117](#), GW1NR 系列 FPGA 产品数据手册
- [DS102](#), GW2A 系列 FPGA 产品数据手册
- [DS226](#), GW2AR 系列 FPGA 产品数据手册
- [SUG100](#), Gowin 云源软件用户手册

## 1.3 术语、缩略语

表 1-1 中列出了本手册中出现的相关术语、缩略语及相关释义。

**表 1-1 术语、缩略语**

术语、缩略语	全称	含义
B-SRAM	Block SRAM	块状静态随机存储器
S-SRAM	Shadow SRAM	附加静态随机存储器
SP	Single Port	单端口
DP	Dual Port	双端口
SDP	Semi Dual Port	伪双端口
CFU	Configurable Function Unit	可配置功能单元
CST	Constraints	物理约束文件

## 1.4 技术支持与反馈

高云半导体提供全方位技术支持，在使用过程中如有任何疑问或建议，可直接与公司联系：

网址：[www.gowinsemi.com.cn](http://www.gowinsemi.com.cn)

E-mail: [support@gowinsemi.com](mailto:support@gowinsemi.com)

Tel: +86 755 8262 0391

# 2 概述

高云半导体 FPGA 产品提供了丰富的存储器资源，包括块状静态随机存储器（B-SRAM）和附加静态随机存储器（S-SRAM）。

每个 B-SRAM 可配置最高 18Kbits，数据位宽和地址深度均可配置。每个 B-SRAM 具有独立的 A、B 两个端口，具有独立的时钟、地址、数据和控制信号，可以独立的进行读写操作，且两个端口共享一块存储空间。

可配置功能单元(CFU) 是构成高云半导体 FPGA 产品内核的基本单元，可根据应用场景配置成 S-SRAM，包括 16 x 4 位的静态随机存储器(SRAM) 或只读存储器（ROM16）。

## 2.1 B-SRAM 特性介绍

- 一块 B-SRAM 最大容量为 18Kbits
- 时钟频率达到 380MHz(在 Read-before-write 模式下 230MHz)
- 支持单端口模式（SP）
- 支持双端口模式模式（DP）
- 支持伪双端口模式（SDP）
- 支持只读模式（ROM）
- 数据位宽最大支持 36 bits
- 双端口模式和伪双端口模式支持读写时钟独立、数据位宽独立
- 读模式支持寄存器输出或旁路输出
- 写模式支持 Normal 模式、read-before-write 模式和 write-through 模式

## 2.2 B-SRAM 配置模式

每个 B-SRAM 可配置成 16Kbits 或 18Kbits 大小，四种模式可配置的数据宽度和地址深度如表 2-1 所示。

**表 2-1 B-SRAM 配置模式列表**

存储容量	单端口模式	双端口模式	伪双端口模式	只读模式
16Kbits	16K x 1	16K x 1	16K x 1	16K x 1
	8K x 2	8K x 2	8K x 2	8K x 2

存储容量	单端口模式	双端口模式	伪双端口模式	只读模式
	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
18Kbits	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

每个 B-SRAM 的地址线位宽是 14 位, 即 AD[13:0], 最大地址深度 16,384。不同数据位宽使用的地址线不一样, 对应关系如表 2-2 所示。

表 2-2 B-SRAM 数据和地址位宽对应关系

存储容量	配置模式	数据位宽	地址深度	地址位宽
16Kbits	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]
18Kbits	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]

双端口和伪双端口模式写时钟和读时钟独立, 支持读/写操作数据位宽独立。在双端口模式下, A 端口和 B 端口支持的数据位宽如表 2-3 所示。在伪双端口模式下, A 端口和 B 端口支持的数据位宽如

表 2-4 所示。

表 2-3 双端口模式数据宽度配置列表

存储容量	B 端口	A 端口						
		16K x 1	8K x 2	4K x 4	2K x 8	1K x 16	2K x 9	1K x 18
16Kbits	16K x 1	*	*	*	*	*	N/A	N/A
	8K x 2	*	*	*	*	*	N/A	N/A
	4K x 4	*	*	*	*	*	N/A	N/A
	2K x 8	*	*	*	*	*	N/A	N/A
	1K x 16	*	*	*	*	*	N/A	N/A
18Kbits	2K x 9	N/A	N/A	N/A	N/A	N/A	*	*
	1K x 18	N/A	N/A	N/A	N/A	N/A	*	*

表 2-4 伪双端口模式数据宽度配置列表

存储容量	B 端口	A 端口								
		16K x 1	8K x 2	4K x 4	2K x 8	1K x 16	512x32	2K x 9	1K x 18	512 x 36
16Kbits	16K x 1	*	*	*	*	*	*	N/A	N/A	N/A
	8K x 2	*	*	*	*	*	*	N/A	N/A	N/A
	4K x 4	*	*	*	*	*	*	N/A	N/A	N/A
	2K x 8	*	*	*	*	*	*	N/A	N/A	N/A
	1K x 16	*	*	*	*	*	*	N/A	N/A	N/A
	512 x 32	*	*	*	*	*	*	N/A	N/A	N/A
18Kbits	2K x 9	N/A	N/A	N/A	N/A	N/A	N/A	*	*	*
	1K x 18	N/A	N/A	N/A	N/A	N/A	N/A	*	*	*

注!

- 标注为“\*”的表示支持的模式。

# 3 B-SRAM 原语

**Block Memory** 是块状静态随机存储器，具有静态存取功能。根据 **B-SRAM** 的特性建立软件模型，可分为单端口模式（**SP/SPX9**）、双端口模式（**DPB/DPX9B**）、伪双端口模式（**SDPB/SDPX9B**）和只读模式（**pROM/pROMX9**）。

注！

- **GW1N-9/GW1NR-9/GW1NS-2 系列/GW1NS-4 系列**，不支持双端口模式；
- **GW1N-9/GW1NR-9/GW1NS-2 系列/GW1NS-4 系列**，32/36 位宽的 **SP/SPX9** 被拆成两个 **SP/SPX9** 实现，因此会占用两个 **B-SRAM** 的位置；
- **GW1NZ-1** 不支持位宽为 1/2/4/8/9 的双端口模式。

## 3.1 双端口模式

### 原语介绍

**DPB/DPX9B**(True Dual Port 16K Block SRAM/True Dual Port 18K Block SRAM)，16K/18K 双端口 **B-SRAM**。

### 功能描述

**DPB/DPX9B** 的存储空间分别为 16K bit/18K bit，其工作模式为双端口模式，端口 **A** 和端口 **B** 均可分别独立实现读/写操作，可支持 2 种读模式（**bypass 模式**和 **pipeline 模式**）和 3 种写模式（**normal 模式**、**write-through 模式**和 **read-before-write 模式**）。

- 读模式

通过参数 **READ\_MODE0**、**READ\_MODE1** 来启用或禁用 **A 端**、**B 端** 输出 **pipeline 寄存器**，使用输出 **pipeline 寄存器**时，读操作需要额外的延迟周期。

- 写模式

包括 **normal 模式**、**write-through 模式**和 **read-before-write 模式**，**A 端**、**B 端**写模式通过参数 **WRITE\_MODE0**、**WRITE\_MODE1** 来分别配置使用，不同模式对应的内部时序波形图如图 3-1 到图 3-6 所示。

图 3-1 DPB/DPX9B Normal 写模式时序波形图 (Bypass 读模式)

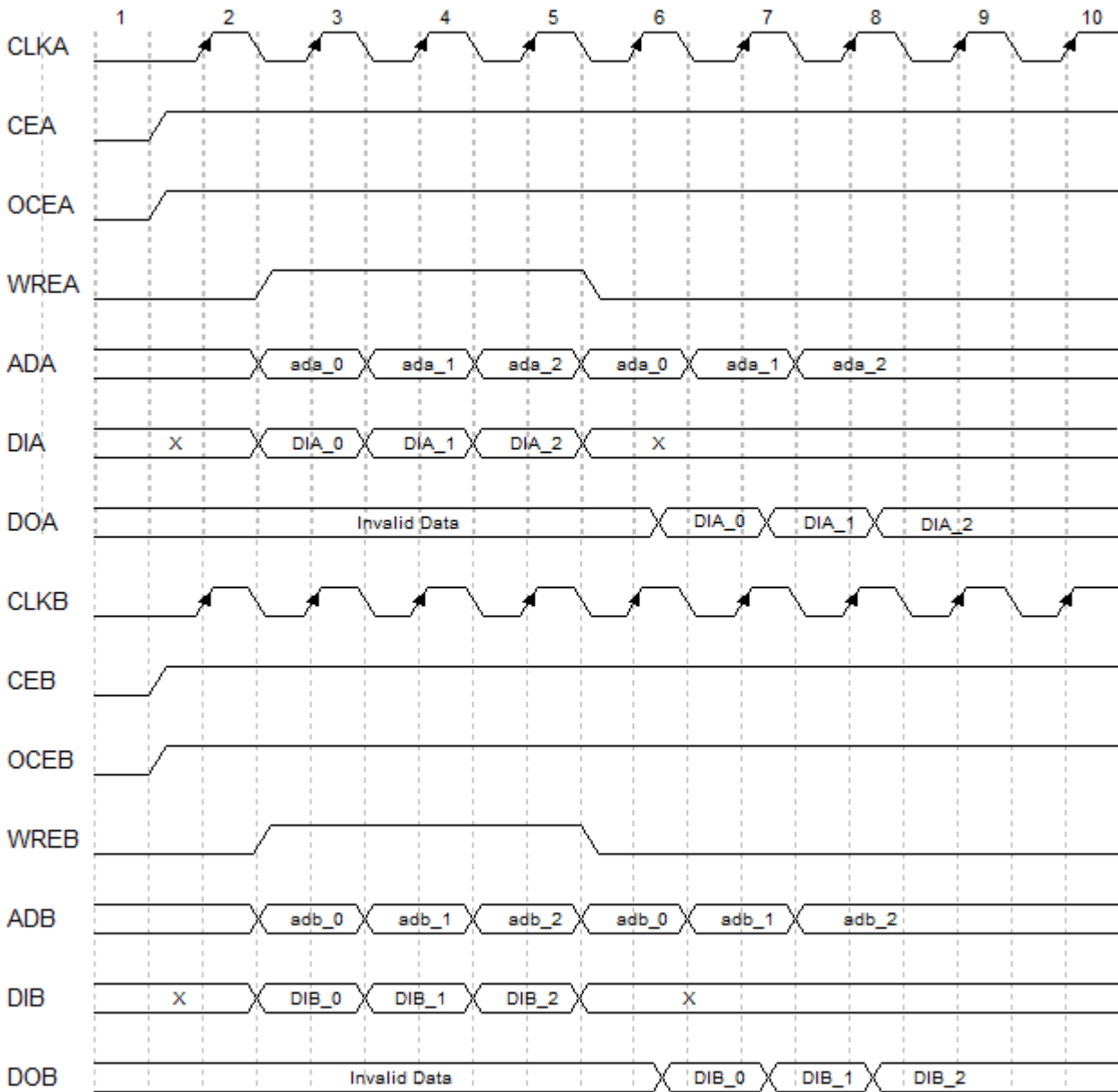




图 3-2 DPB/DPX9B Normal 写模式时序波形图 (Pipeline 读模式)

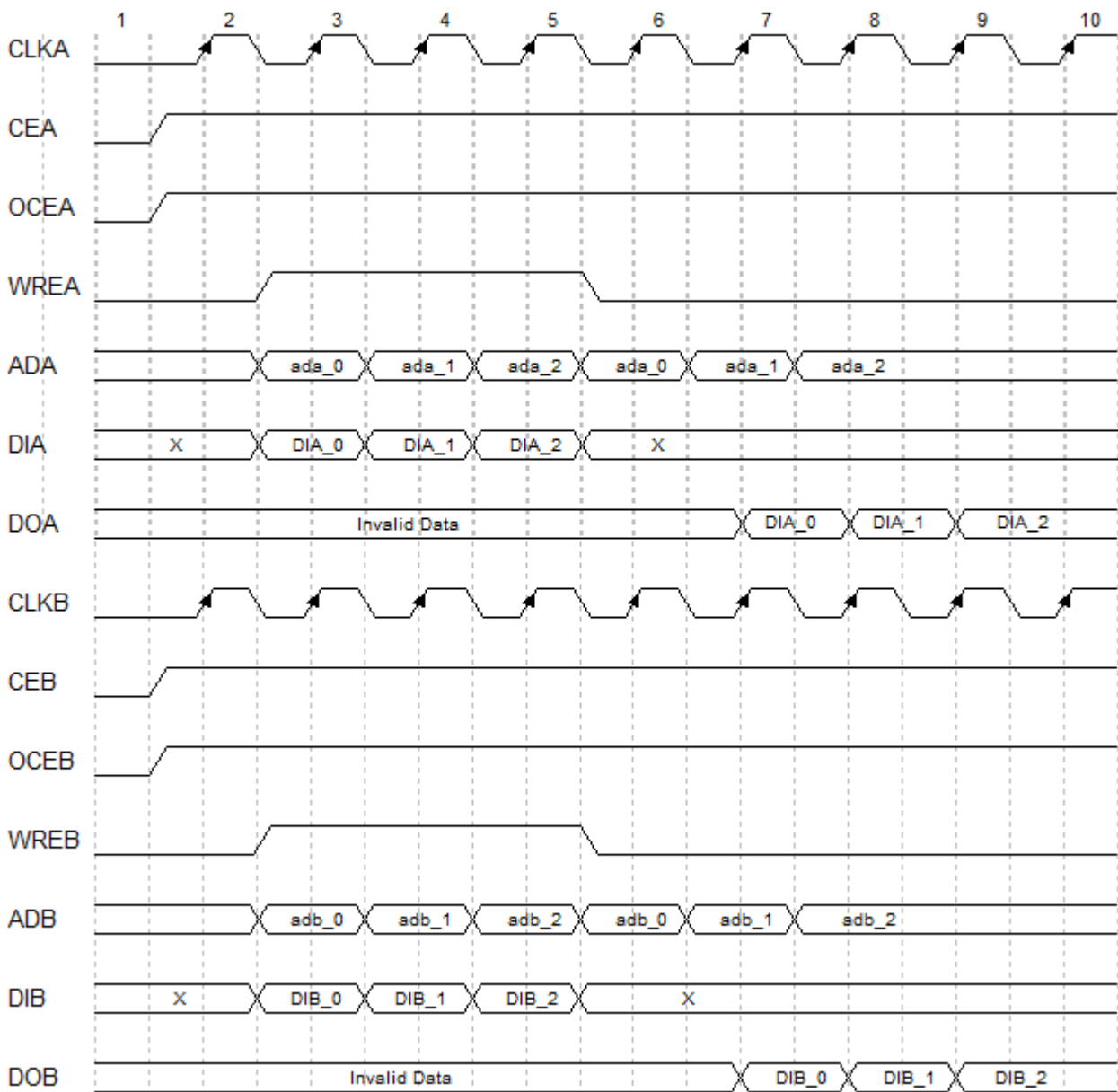


图 3-3 DPB/DPX9B Write-through 写模式时序波形图 (Bypass 读模式)

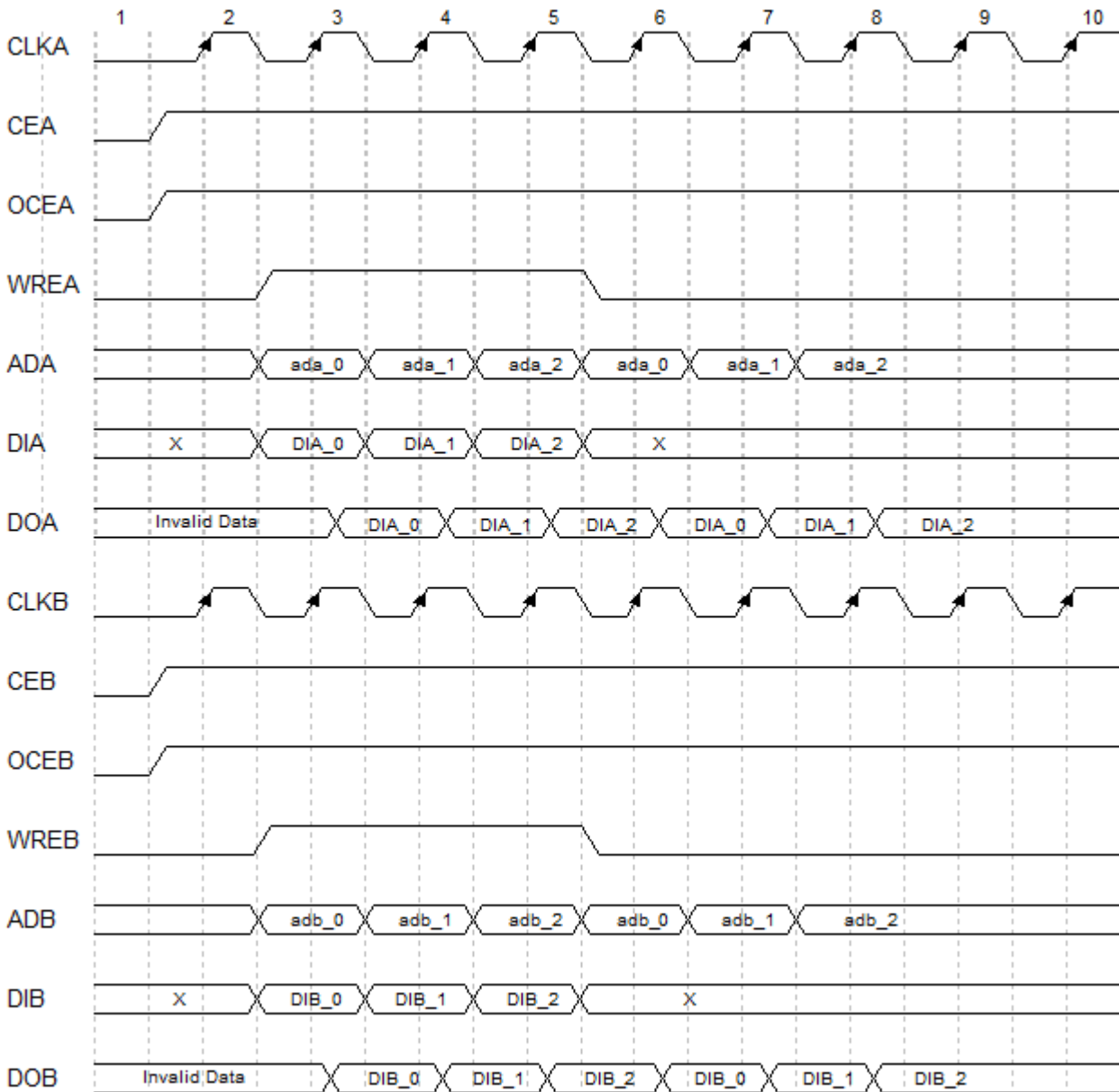


图 3-4 DPB/DPX9B Write-through 写模式时序波形图 (Pipeline 读模式)

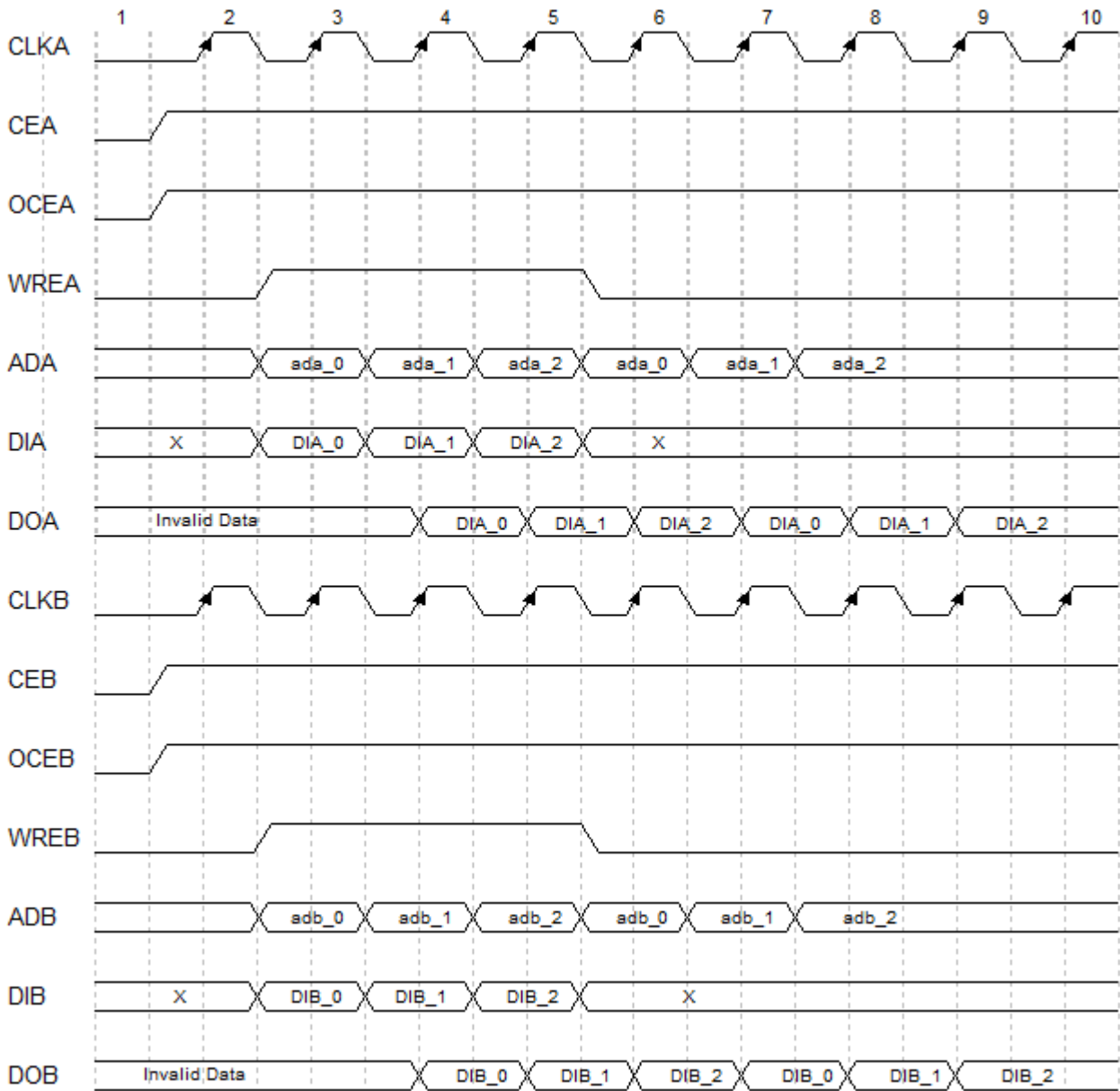


图 3-5 DPB/DPX9B Read-before-write 写模式时序波形图 (Bypass 读模式)

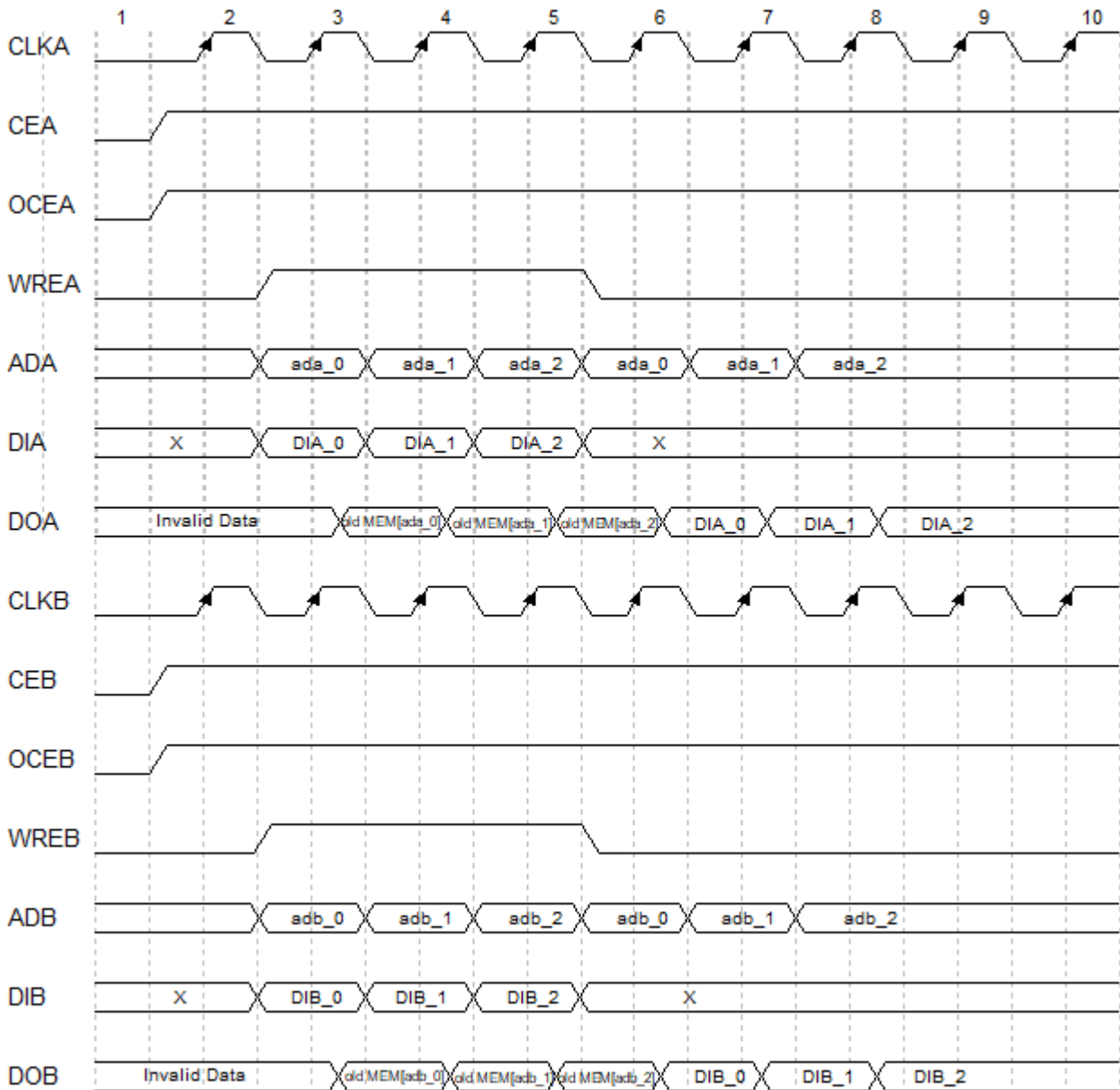
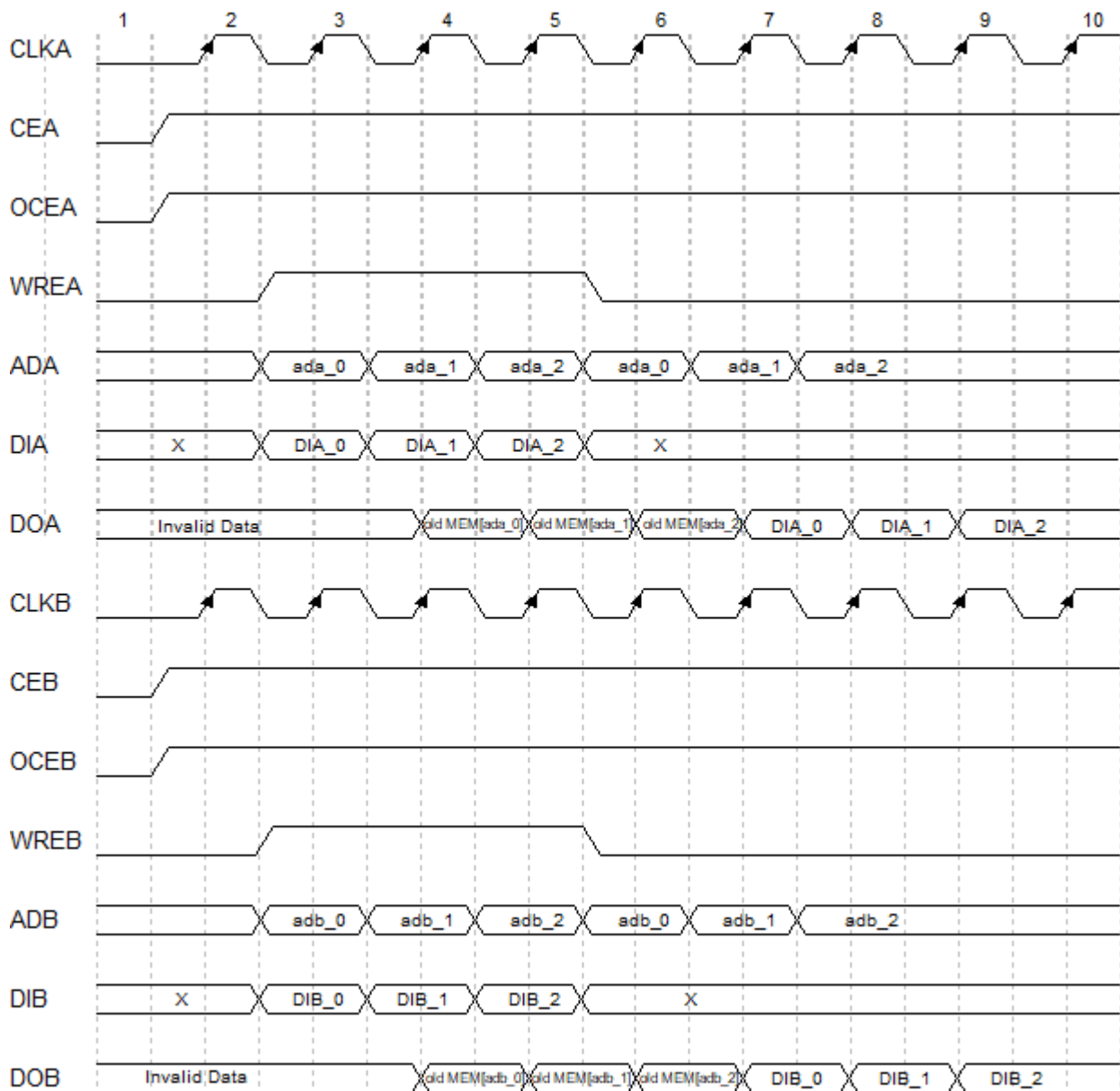


图 3-6 DPB/DPX9B Read-before-write 写模式时序波形图 (Pipeline 读模式)



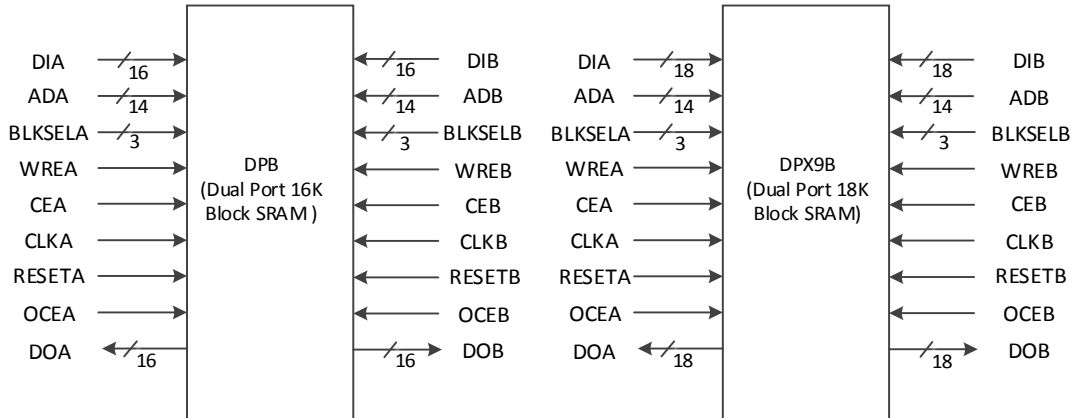
配置关系

表 3-1 DPB/DPX9B 数据宽度和地址深度配置关系

双端口模式	B-SRAM 容量	数据宽度	地址深度
DPB	16Kbits	1	14
		2	13
		4	12
		8	11
		16	10
DPX9B	18Kbits	9	11
		18	10

## 端口示意图

图 3-7 DPB/DPX9B 端口示意图



## 端口介绍

表 3-2 DPB/DPX9B 端口介绍

端口名	I/O	描述
DOA[15:0]/DOA[17:0]	Output	A 端数据输出信号
DOB[15:0]/DOB[17:0]	Output	B 端数据输出信号
DIA[15:0]/DIA[17:0]	Input	A 端数据输入信号
DIB[15:0]/DIB[17:0]	Input	B 端数据输入信号
ADA[13:0]	Input	A 端地址输入信号
ADB[13:0]	Input	B 端地址输入信号
WREA	Input	A 端写使能输入信号 1: 写入 0: 读出
WREB	Input	B 端写使能输入信号 1: 写入 0: 读出
CEA	Input	A 端时钟使能信号, 高电平有效
CEB	Input	B 端时钟使能信号, 高电平有效
CLKA	Input	A 端时钟输入信号
CLKB	Input	B 端时钟输入信号
RESETA	Input	A 端复位输入信号, 支持同步复位和异步复位, 高电平有效
RESETB	Input	B 端复位输入信号, 支持同步复位和异步复位, 高电平有效
OCEA	Input	A 端输出时钟使能信号, 用于 A 端 pipeline 模式, 对 bypass 模式无效
OCEB	Input	B 端输出时钟使能信号, 用于 B 端 pipeline 模式, 对 bypass 模式无效
BLKSELA[2:0]	Input	B-SRAM A 端口块选择信号, 用于需要多个 B-SRAM 存储单元级联实现容量扩展

端口名	I/O	描述
BLKSELB[2:0]	Input	B-SRAM B 端口块选择信号, 用于需要多个 B-SRAM 存储单元级联实现容量扩展

## 参数介绍

表 3-3 DPB/DPX9B 参数介绍

参数名	参数类型	取值范围	默认值	描述
READ_MODE0	Integer	1'b0,1'b1	1'b0	A 端读模式配置 1'b0:bypass 模式 1'b1:pipeline 模式
READ_MODE1	Integer	1'b0,1'b1	1'b0	B 端读模式配置 1'b0:bypass 模式 1'b1:pipeline 模式
WRITE_MODE0	Integer	2'b00,2'b01,2'b10	2'b00	A 端写模式配置 2'b00: normal 模式 2'b01: write-through 模式 2'b10: read-before-write 模式
WRITE_MODE1	Integer	2'b00,2'b01,2'b10	2'b00	B 端写模式配置 2'b00: normal 模式 2'b01: write-through 模式 2'b10: read-before-write 模式
BIT_WIDTH_0	Integer	DPB:1,2,4,8,16 DPX9B:9,18	DPB:16 DPX9B:18	A 端数据宽度配置
BIT_WIDTH_1	Integer	DPB:1,2,4,8,16 DPX9B:9,18	DPB:16 DPB:18	B 端数据宽度配置
BLK_SEL_0	Integer	3'b000~3'b111	3'b000	B-SRAM A 端口块选择参数设置, 与端口 BLKSELA 相等时该 B-SRAM 被选中。使用 IP Core Generator 进行存储扩展时软件自动进行扩展处理
BLK_SEL_1	Integer	3'b000~3'b111	3'b000	B-SRAM B 端口块选择参数设置, 与端口 BLKSELB 相等时该 B-SRAM 被选中。使用 IP Core Generator 进行存储扩展时软件自动进行扩展处理
RESET_MODE	String	"SYNC","ASYNC"	"SYNC"	复位模式配置 SYNC: 同步复位 ASYNC: 异步复位
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	用于设置 B-SRAM 存储单元的初始化数据

## 原语例化

可以直接实例化原语, 也可以通过 IP Core Generator 工具产生, 具体可参考第 6 章 IP 调用。

原语例化以 DPB 为例介绍:

### Verilog 例化:

```

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'h00A0000000000000B00A000000000000B00A000000000000B00A00
0000000000B;
defparam bram_dpb_0.INIT_RAM_3E =
256'h00A0000000000000B00A000000000000B00A000000000000B00A00

```



```

0000000000B;
    defparam bram_dpb_0.INIT_RAM_3F =
256'h00A0000000000000B00A000000000000B00A000000000000B00A00
0000000000B;

```

**Vhdl 例化:**

```

    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"0000000000000000
000000000000000000000000000000000000000000000000000000000000000";
            INIT_RAM_01:bit_vector:=X"0000000000000000
000000000000000000000000000000000000000000000000000000000000000";
            INIT_RAM_3F:bit_vector:=X"0000000000000000
000000000000000000000000000000000000000000000000000000000000000"
        );
        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"00000000000000000000000000000000000000000000",
        INIT_RAM_01=>X"00000000000000000000000000000000000000000000",
        INIT_RAM_3F=>X"00000000000000000000000000000000000000000000"
    )
    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=>blkseleb,
        DIA=>dia,
        DIB=>dib
    );
```

## 3.2 单端口模式

### 原语介绍

SP/SPX9(Single Port 16K B-SRAM/Single Port 18K B-SRAM),16K/18K 单端口 B-SRAM。

### 功能描述

SP/SPX9 存储空间为 16K bit/18K bit，其工作模式为单端口模式，由一个时钟控制单端口的读/写操作，可支持 2 种读模式（bypass 模式和 pipeline 模式）和 3 种写模式（normal 模式、write-through 模式和 read-before-write 模式）。

- 读模式

通过参数 READ\_MODE 来启用或禁用输出 pipeline 寄存器，使用输出 pipeline 寄存器时，读操作需要额外的延迟周期。

- 写模式

包括 normal 模式、write-through 模式和 read-before-write 模式，通过参数 WRITE\_MODE 来配置使用。

单端口 B-SRAM 不同读写模式对应的内部时序波形图可参考双端口 B-SRAM A 端/B 端时序图 3-1 到图 3-6。

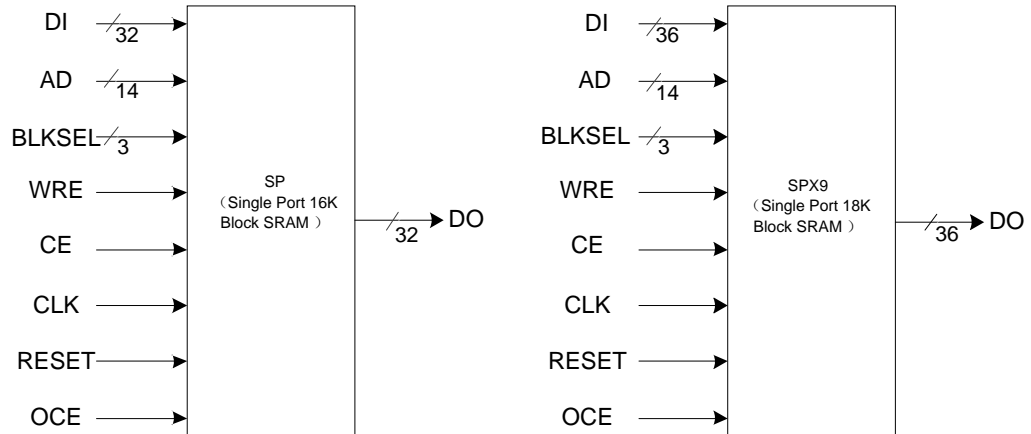
### 配置关系

表 3-4 SP/SPX9 数据宽度和地址深度配置关系

单端口模式	B-SRAM 容量	数据宽度	地址深度
SP	16Kbits	1	14
		2	13
		4	12
		8	11
		16	10
		32	9
SPX9	18Kbits	9	11
		18	10
		36	9

## 端口示意图

图 3-8 SP/SPX9 端口示意图



## 端口介绍

表 3-5 SP/SPX9 端口介绍

端口名	I/O	描述
DO[31:0]/DO[35:0]	Output	数据输出信号
DI[31:0]/DI[35:0]	Input	数据输入信号
AD[13:0]	Input	地址输入信号
WRE	Input	写使能输入信号 1: 写入 0: 读出
CE	Input	时钟使能输入信号, 高电平有效
CLK	Input	时钟输入信号
RESET	Input	复位输入信号, 支持同步复位和异步复位, 高电平有效
OCE	Input	输出时钟使能信号, 用于 pipeline 模式, 对 bypass 模式无效
BLKSEL[2:0]	Input	B-SRAM 块选择信号, 用于需要多个 B-SRAM 存储单元级联实现容量扩展

## 参数介绍

表 3-6 SP/SPX9 参数介绍

参数名	参数类型	取值范围	默认值	描述
READ_MODE	Integer	1'b0, 1'b1	1'b0	读模式配置 1'b0: bypass 模式 1'b1: pipeline 模式
WRITE_MODE	Integer	2'b00, 2'b01, 2'b10	2'b00	写模式配置 2'b00: normal 模式 2'b01: write-through 模式;

参数名	参数类型	取值范围	默认值	描述
				2'b10: read-before-write 模式
BIT_WIDTH	Integer	SP:1,2,4,8,16,32 SPX9:9,18,36	SP:32 SPX9:36	数据宽度配置
BLK_SEL	Integer	3'b000~3'b111	3'b000	B-SRAM 块选择参数设置，与端口 BLKSEL 相等时该 B-SRAM 被选中。使用 IP Core Generator 进行存储扩展时软件自动进行扩展处理
RESET_MODE	String	"SYNC","ASYN"	"SYNC"	复位模式配置 SYNC: 同步复位 ASYN: 异步复位
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	用于设置 B-SRAM 存储单元的初始化数据

### 原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生，具体可参考第 6 章 IP 调用。原语例化以 SP 为例介绍，

#### Verilog 例化:

```

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'h00A0000000000000B00A000000000000B00A000000000000B00
A0000000000000B;

```

```
defparam bram_sp_0.INIT_RAM_01 =
256'h00A00000000000B00A000000000000B00A000000000000B00
A0000000000000B;
```

```
defparam bram_sp_0.INIT_RAM_3F =
256'h00A00000000000B00A000000000000B00A000000000000B00
A0000000000000B;
```

**Vhdl 例化:**

```
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"00A0000000000000B
00A0000000000000B00A0000000000000B00A0000000000000B ";
    INIT_RAM_01:bit_vector:=X"00A0000000000000B
00A0000000000000B00A0000000000000B00A0000000000000B ";
    INIT_RAM_3F:bit_vector:=X"00A0000000000000B
00A0000000000000B00A0000000000000B00A0000000000000B "
  );
  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"00A0000000000000B00A00
0000000000B00A00000000000000B00A000000000000B ",
        INIT_RAM_01=>X"00A0000000000000B00A00
0000000000B00A00000000000000B00A000000000000B ",
        INIT_RAM_02=>X"00A0000000000000B00A00
0000000000B00A00000000000000B00A000000000000B ",
        INIT_RAM_3F=>X"00A0000000000000B00A00
0000000000B00A00000000000000B00A000000000000B "
    )
    PORT MAP (
        DO=>dout,
        CLK=>clk,
        OCE=>oce,
        CE=>ce,
        RESET=>reset,
        WRE=>wre,
        BLKSEL=>blkssel,
        AD=>ad,
        DI=>din
    );

```

### 3.3 伪双端口模式

#### 原语介绍

SDPB/SDPX9B(Semi Dual Port 16K Block SRAM /Semi Dual Port 18K Block SRAM ),16K/18K 伪双端口 B-SRAM。

#### 功能描述

SDPB/SDPX9B 存储空间分别为 16K bit/18K bit，其工作模式为伪双端口模式，端口 A 进行写操作，端口 B 进行读操作，可支持 2 种读模式（bypass 模式和 pipeline 模式）和 1 种写模式（normal 模式）。

- 读模式

通过参数 READ\_MODE 来启用或禁用输出 pipeline 寄存器，使用输出 pipeline 寄存器时，读操作需要额外的延迟周期。

- 写模式

SDPB/SDPX9B 端口 A 进行写操作，端口 B 进行读操作，支持 normal 模式。

伪双端口 B-SRAM 不同读模式对应的内部时序波形图如图 3-9 和图 3-10 所示。

图 3-9 伪双端口 B-SRAM Normal 写模式时序波形图 (Bypass 读模式)

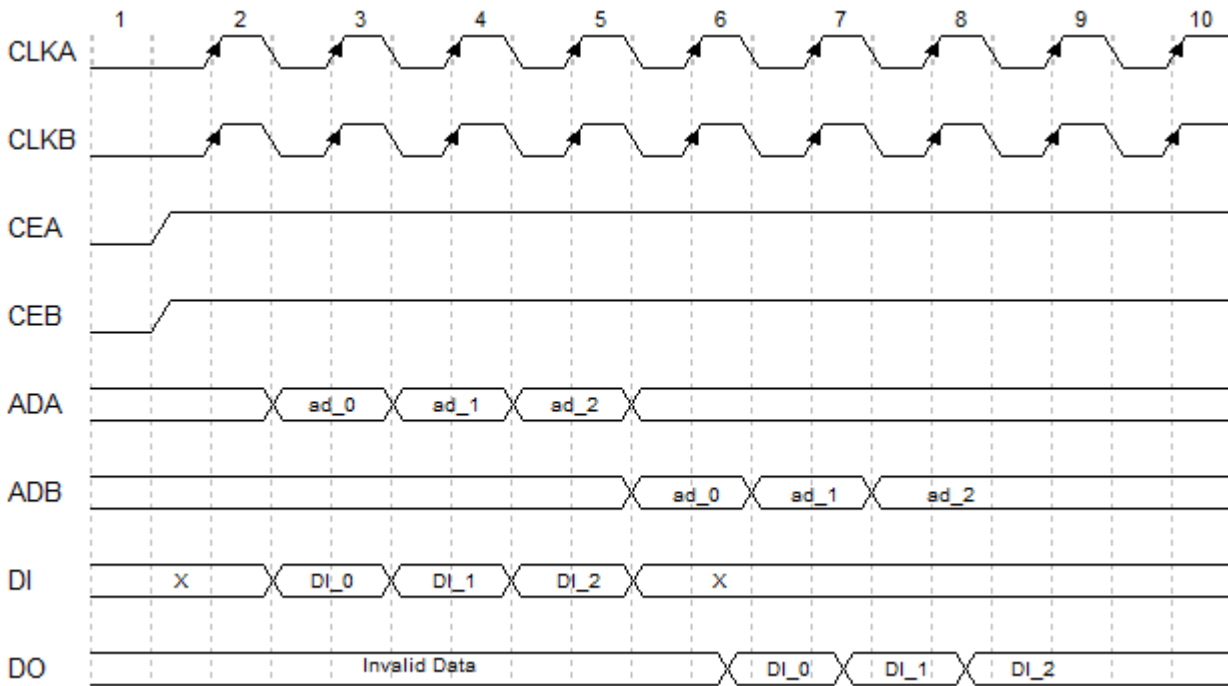
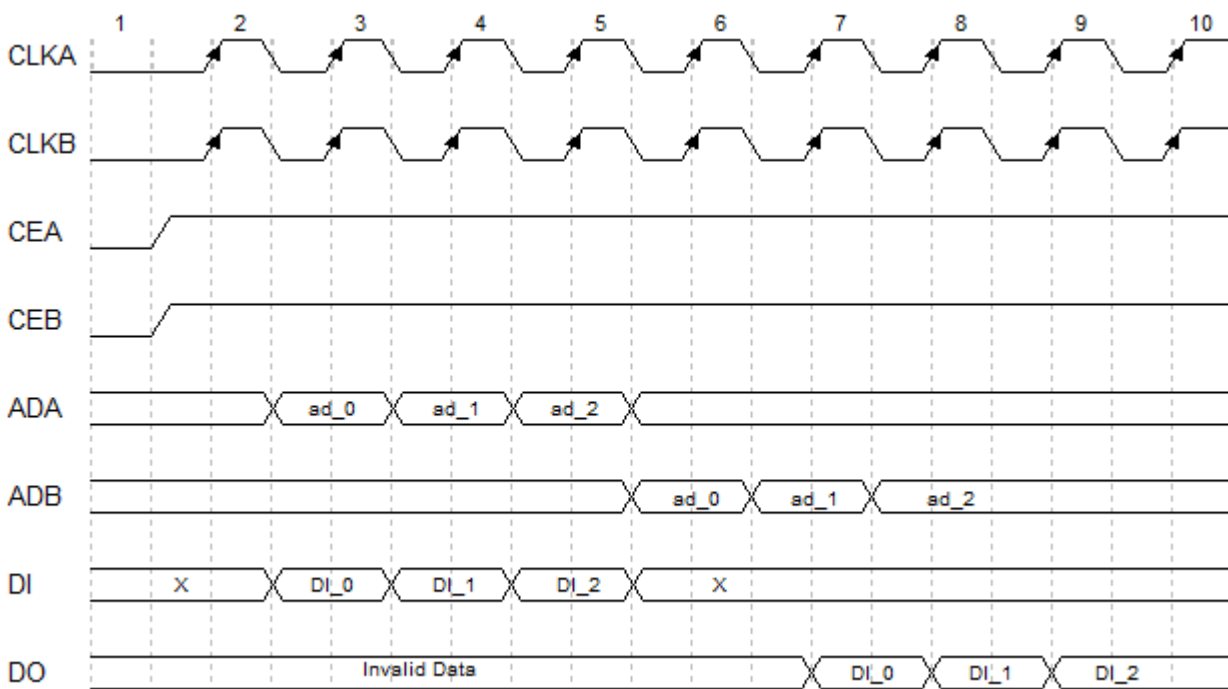


图 3-10 伪双端口 B-SRAM Normal 写模式时序波形图 (Pipeline 读模式)



配置关系

表 3-7 SDPB/SDPX9B 数据宽度和地址深度配置关系

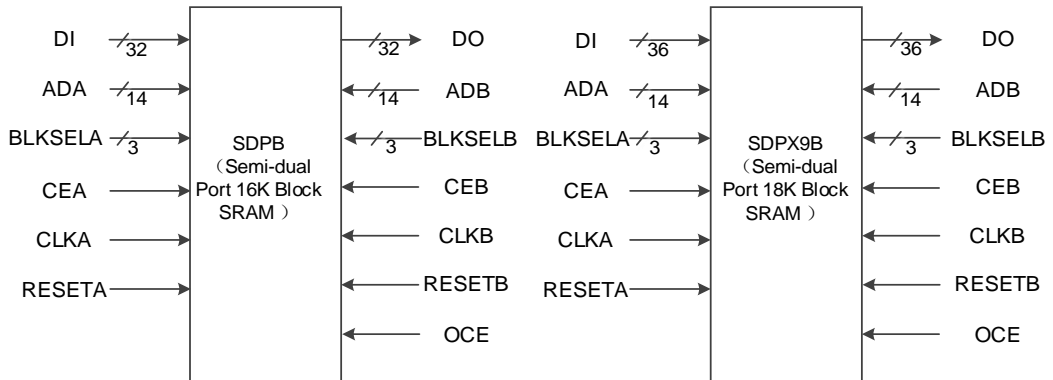
伪双端口模式	B-SRAM 容量	数据宽度	地址深度
SDPB	16Kbits	1	14
		2	13
		4	12



伪双端口模式	B-SRAM 容量	数据宽度	地址深度
		8	11
		16	10
		32	9
SDPX9B	18Kbits	9	11
		18	10
		36	9

端口示意图

图 3-11 SDPB/SDPX9B 端口示意图



端口介绍

表 3-8 SDPB/SDPX9B 端口介绍

端口名	I/O	描述
DO[31:0]/DO[35:0]	Output	数据输出信号
DI[31:0]/DI[35:0]	Input	数据输入信号
ADA[13:0]	Input	A 端地址输入信号
ADB[13:0]	Input	B 端地址输入信号
CEA	Input	A 端时钟使能信号，高电平有效
CEB	Input	B 端时钟使能信号，高电平有效
CLKA	Input	A 端时钟输入信号
CLKB	Input	B 端时钟输入信号
RESETA	Input	A 端复位输入信号，支持同步复位和异步复位，高电平有效
RESETB	Input	B 端复位输入信号，支持同步复位和异步复位，高电平有效
OCE	Input	输出时钟使能信号，用于 pipeline 模式，对 bypass 模式无效
BLKSELA[2:0]	Input	B-SRAM A 端口块选择信号，用于需要多个 B-SRAM 存储单元级联实现容量扩展

端口名	I/O	描述
BLKSELB[2:0]	Input	B-SRAM B 端口块选择信号, 用于需要多个 B-SRAM 存储单元级联实现容量扩展

## 参数介绍

表 3-9 SDPB/SDPX9B 参数介绍

参数名	参数类型	取值范围	默认值	描述
READ_MODE	Integer	1'b0,1'b1	1'b0	读模式配置 1'b0:bypass 模式 1'b1:pipeline 模式
BIT_WIDTH_0	Integer	SDPB:1,2,4,8,16,32 SDPX9B:9,18,36	SDPB:32 SDPX9B:36	A 端数据宽度配置
BIT_WIDTH_1	Integer	SDPB:1,2,4,8,16,32 SDPX9B:9,18,36	SDPB:32 SDPX9B:36	B 端数据宽度配置
BLK_SEL_0	Integer	3'b000~3'b111	3'b000	B-SRAM A 端口块选择参数设置, 与端口 BLKSEL 相等时该 B-SRAM 被选中。使用 IP Core Generator 进行存储扩展时软件自动进行扩展处理。
BLK_SEL_1	Integer	3'b000~3'b111	3'b000	B-SRAM B 端口块选择参数设置, 与端口 BLKSEL 相等时该 B-SRAM 被选中。使用 IP Core Generator 进行存储扩展时软件自动进行扩展处理
RESET_MODE	String	"SYNC","ASYNC"	"SYNC"	复位模式配置 SYNC: 同步复位 ASYNC: 异步复位
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'h 0...0	用于设置 B-SRAM 存储单元的初始化数据

## 原语例化

可以直接实例化原语, 也可以通过 IP Core Generator 工具产生, 具体可参考第 6 章 IP 调用。原语例化以 SDPB 为例介绍,

### Verilog 例化:

```
SDPB bram_sdpb_0 (
    .DO({dout[31:16],dout[15:0]}),
    .CLKA(clka),
    .CEA(cea),
    .RESETA(reseta),
```

```

        .CLKB(clkb),
        .CEB(ceb),
        .RESETB(resetb),
        .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'h00A00000000000B00A000000000000B00A000000000000B00
    A000000000000B;
    defparam bram_sdpb_0.INIT_RAM_3F =
    256'h00A00000000000B00A000000000000B00A000000000000B00
    A000000000000B;

```

**Vhdl 例化:**

```

    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"00A00000000000
            B00A000000000000B00A000000000000B00A000000000000B";
            INIT_RAM_01:bit_vector:=X"00A00000000000
            B00A000000000000B00A000000000000B00A000000000000B";
            INIT_RAM_3F:bit_vector:=X"00A00000000000
            B00A000000000000B00A000000000000B00A000000000000B"

```

```

);
PORT(
    DO:OUT std_logic_vector(31 downto 0):=conv_
std_logic_vector(0,32);
    CLKA,CLKB,CEA,CEB:IN std_logic;
    OCE,RESETA,RESETB: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"00A0000000000000B00A00
0000000000B00A00000000000000B00A000000000000B",
        INIT_RAM_01=>X"00A0000000000000B00A00
0000000000B00A00000000000000B00A000000000000B",
        INIT_RAM_3F=>X"00A0000000000000B00A00
0000000000B00A00000000000000B00A000000000000B"
    )
    PORT MAP(
        DO=>dout,
        CLKA=>clka,
        CEA=>cea,
        RESETA=>reseta,
        CLKB=>clkb,
        CEB=>ceb,
        RESETB=>resetb,
        OCE=>oce,

```

```

        BLKSELA=>blksela,
        BLKSELB=>blkselb,
        ADA=>ada,
        DI=>din,
        ADB=>adb
    );

```

## 3.4 只读模式

### 原语介绍

pROM/pROMX9(16K/18K Block ROM),16K/18K 块状只读存储器。

### 功能描述

pROM/pROMX9 存储空间分别为 16K bit/18K bit，其工作模式为只读模式，可支持 2 种读模式（bypass 模式和 pipeline 模式）。

通过参数 READ\_MODE 来启用或禁用输出 pipeline 寄存器，使用输出 pipeline 寄存器时，读操作需要额外的延迟周期。

只读 ROM 不同读模式对应的内部时序波形图可参考伪双端口 B-SRAM 的 B 端口时序，如图 3-12 和图 3-13 所示。

图 3-12 只读 ROM 时序波形图（Bypass 模式）

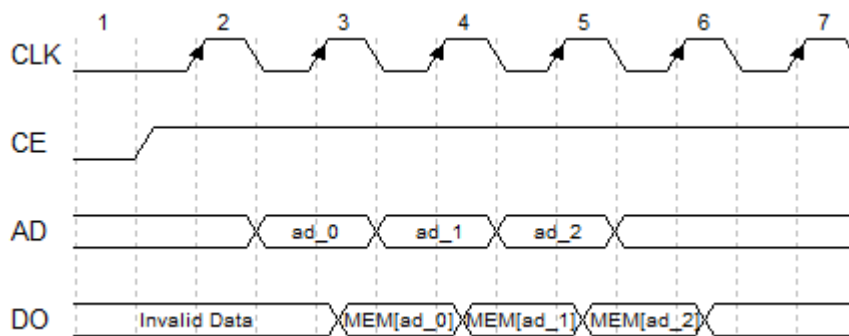
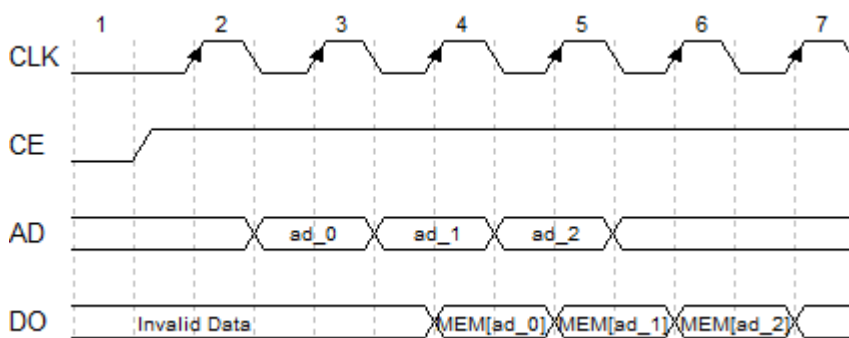


图 3-13 只读 ROM 时序波形图（Pipeline 模式）



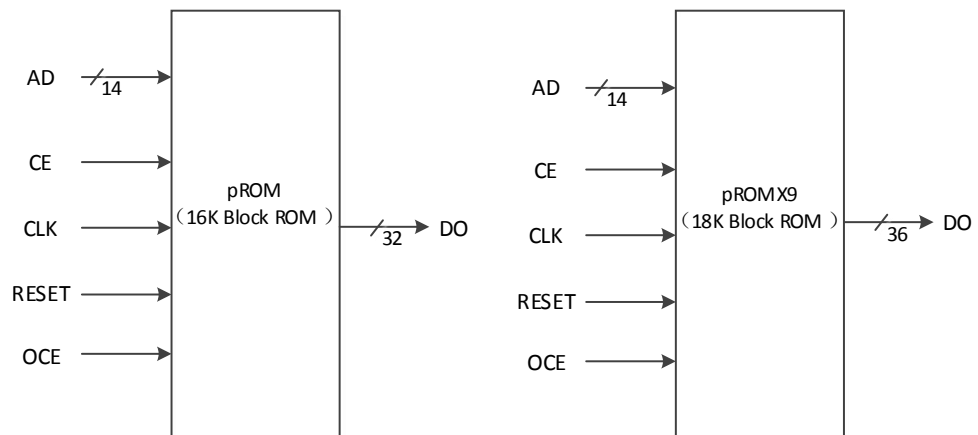
## 配置关系

表 3-10 pROM/pROMX9 配置关系

只读模式	B-SRAM 容量	数据宽度	地址深度
pROM	16Kbits	1	14
		2	13
		4	12
		8	11
		16	10
		32	9
pROMX9	18Kbits	9	11
		18	10
		36	9

## 端口示意图

图 3-14 pROM/pROMX9 端口示意图



## 端口介绍

表 3-11 pROM/pROMX9 端口介绍

端口名	I/O	描述
DO[31:0]/DO[35:0]	Output	数据输出信号
AD[13:0]	Input	地址输入信号
CE	Input	时钟使能输入信号，高电平有效
CLK	Input	时钟输入信号
RESET	Input	复位输入信号，支持同步复位和异步复位，高电平有效
OCE	Input	输出时钟使能信号，用于 pipeline 模式，对 bypass 模式无效

## 参数介绍

表 3-12 pROM/pROMX9 参数介绍

参数名	参数类型	取值范围	默认值	描述
READ_MODE	Integer	1'b0,1'b1	1'b0	读模式配置 1'b0:bypass 模式 1'b1:pipeline 模式
BIT_WIDTH	Integer	pROM:1,2,4,8,16,32 pROMX9:9,18,36	pROM:32 pROMX9:36	数据宽度配置
RESET_MODE	String	"SYNC","ASYNC"	"SYNC"	复位模式配置 SYNC: 同步复位 ASYNC: 异步复位
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	用于设置 B-SRAM 存储 单元的初始化数据

## 原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生，具体可参考第 6 章 IP 调用。原语例化以 pROM 为例介绍，

### Verilog 例化:

```
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'h9C23645D0F78986FFC3E36E141541B95C19F2F7164085E63
1A819860D8FF0000;
defparam bram_prom_0.INIT_RAM_01 =
256'h00000000000000000000000000000000000000000000000000
00FFFFFFBDCF;
```

### Vhdl 例化:

```
COMPONENT pROM
    GENERIC(
```

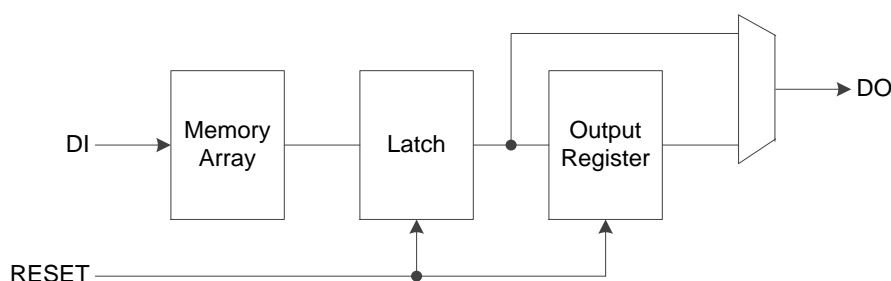




# 4 B-SRAM 输出复位

RESET 信号作用于输出模块,输出复位数据 0,结构框图如图 4-1 所示。

图 4-1 复位输出结构框图



RESET 信号高电平有效时输出端口输出 0。

RESET 支持同步复位和异步复位,当用户直接调用库原语时,通过参数 RESET\_MODE 设置。当用户使用 IP Core Generator 时,可通过窗口选择复位模式,详细资料请参考第 6 章 IP 调用。

RESET 信号复位锁存器和输出寄存器,因此,当设置 RESET 信号有效时,不管用户使用的是寄存器输出模式还是旁路输出模式,端口都输出 0。

注!

写操作过程中 RESET 信号须置为 0 (无效状态)。

图 4-2、图 4-3、图 4-4 和图 4-5 为不同模式下复位时序图,其中,DO\_RAM 表示存储阵列中的数据,DO 表示输出端口的数据。

寄存器输出模式如下所示:

- 同步复位有效时, DO 在 CLK 上升沿复位为 0;
- 异步复位有效时, DO 随之复位为 0,不需要等到 CLK 上升沿;
- 复位无效,且 OCE 信号有效时, DO 输出 DO\_RAM;
- 复位无效,且 OCE 信号无效时, DO 保持上一次输出的数据

旁路输出模式如下所示:

- 同步复位有效时, DO 在 CLK 上升沿复位为 0;

- 异步复位有效时，DO 随之复位为 0，不需要等到 CLK 上升沿；
- 复位无效时，不管 OCE 信号是否有效，DO 输出 DO\_RAM。

图 4-2 同步复位时序图（Pipeline 模式）

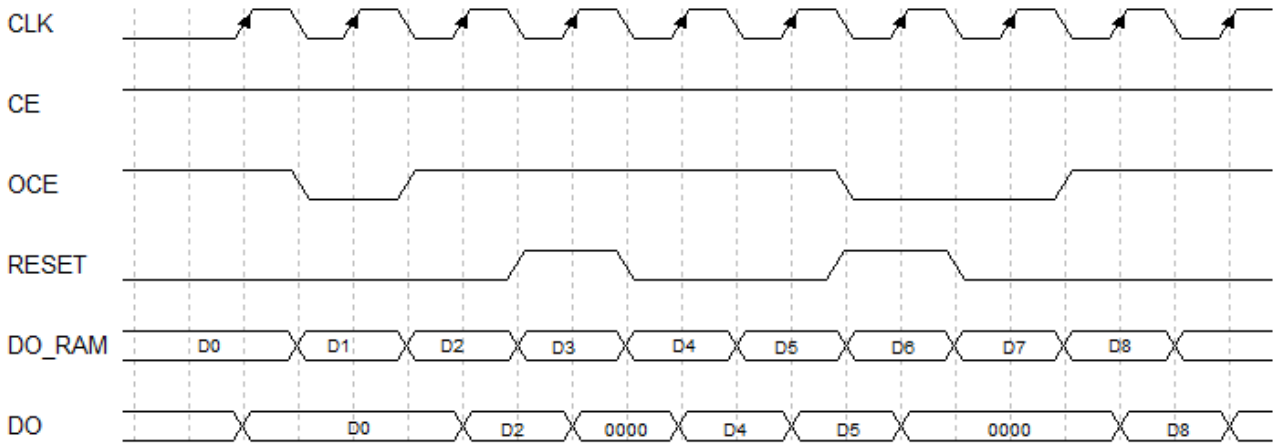


图 4-3 同步复位时序图（Bypass 输出模式）

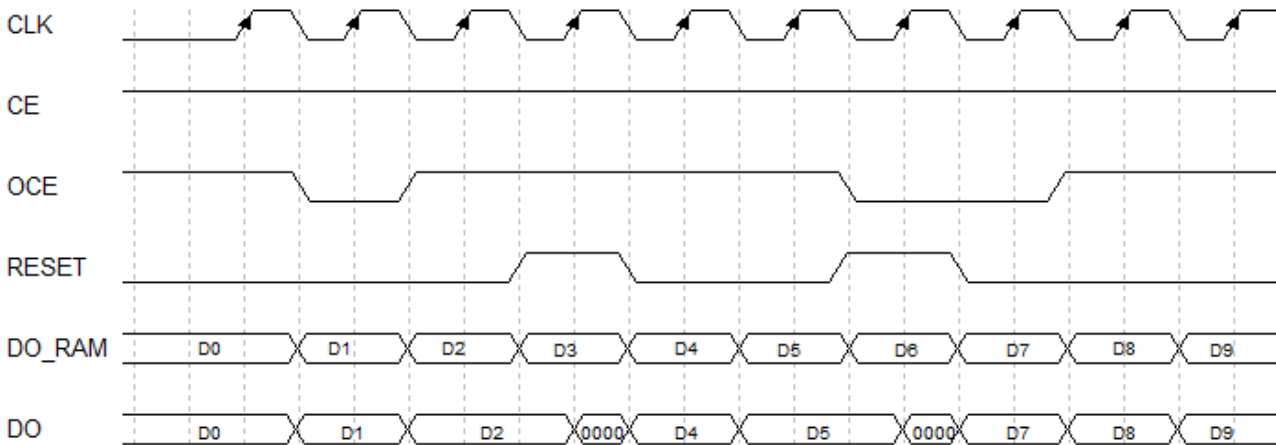


图 4-4 异步复位时序图（Pipeline 模式）

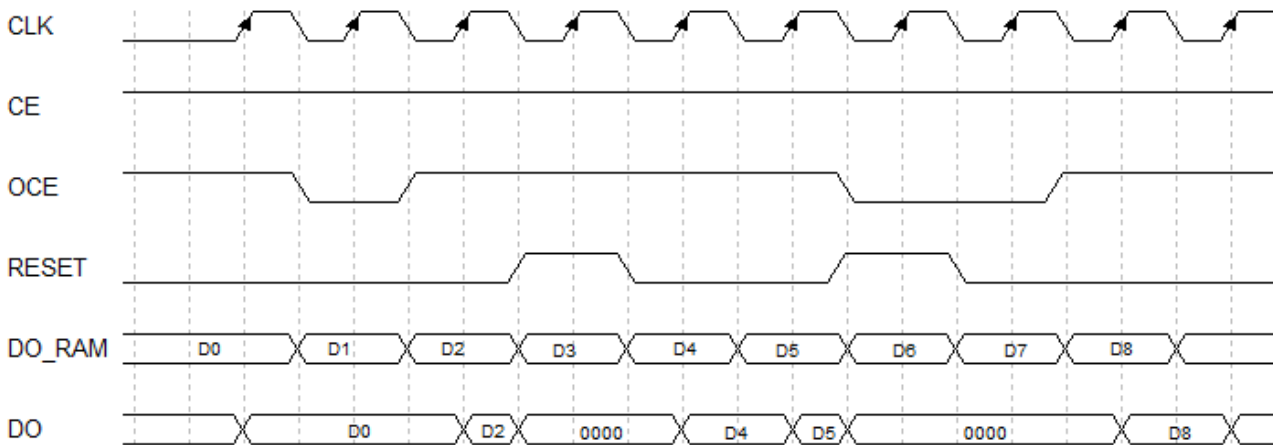
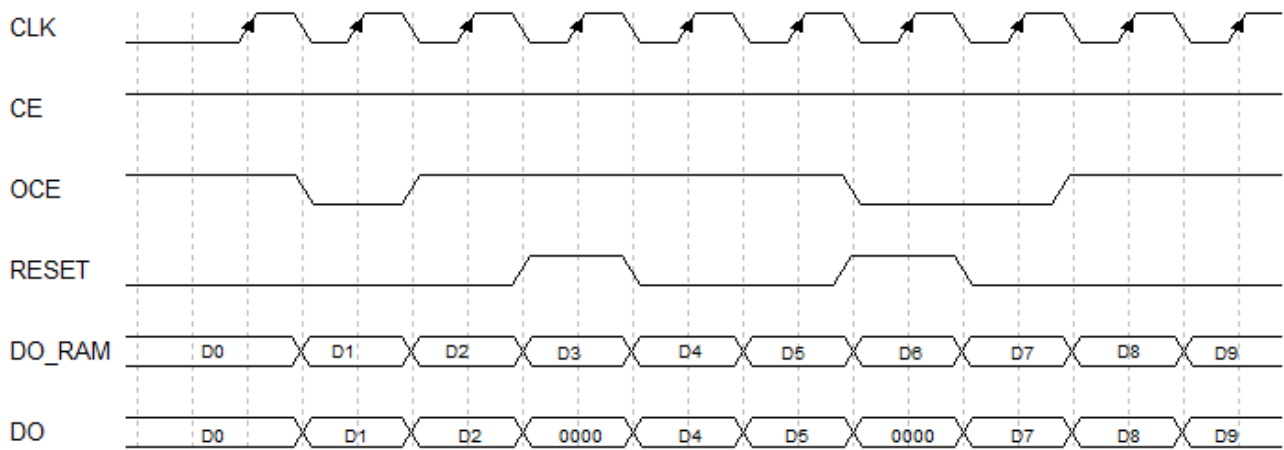


图 4-5 异步复位时序图 (Bypass 输出模式)



# 5 S-SRAM 原语

Shadow SRAM 是附加静态随机存储器，可配置成单端口模式，伪双端口模式和只读模式，如表 5-1 所示。

**表 5-1 S-SRAM 模式**

原语	描述
RAM16S1	地址深度 16，数据宽度为 1 的单端口 S-SRAM
RAM16S2	地址深度 16，数据宽度为 2 的单端口 S-SRAM
RAM16S4	地址深度 16，数据宽度为 4 的单端口 S-SRAM
RAM16SDP1	地址深度 16，数据宽度为 1 的伪双端口 S-SRAM
RAM16SDP2	地址深度 16，数据宽度为 2 的伪双端口 S-SRAM
RAM16SDP4	地址深度 16，数据宽度为 4 的伪双端口 S-SRAM
ROM16	地址深度 16，数据宽度为 1 的只读 ROM

注！

GW1N-1、GW1N-1S、GW1N-4、GW1N-4B、GW1NR-1、GW1NR-4、GW1NR-4B、GW1NRF-4B、GW1NS-4、GW1NS-4C、GW1NSER-4C、GW1NSR-4、GW1NSR-4C、GW1N-4C、GW1NR-4C 器件，不支持 S-SRAM。

## 5.1 RAM16S1

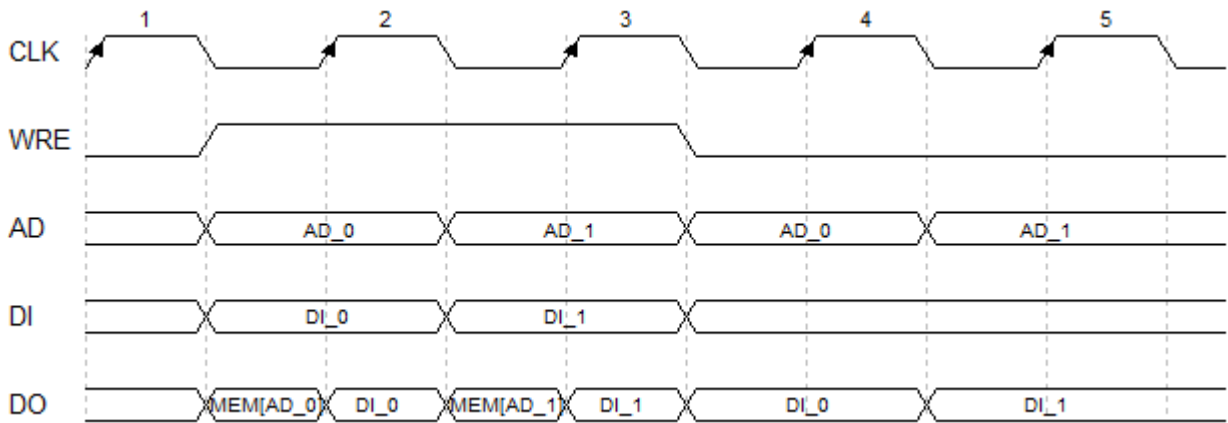
### 原语介绍

RAM16S1(16-Deep by 1-Wide Single-port S-SRAM)是地址深度为 16，数据位宽为 1 的单端口 S-SRAM。

### 功能描述

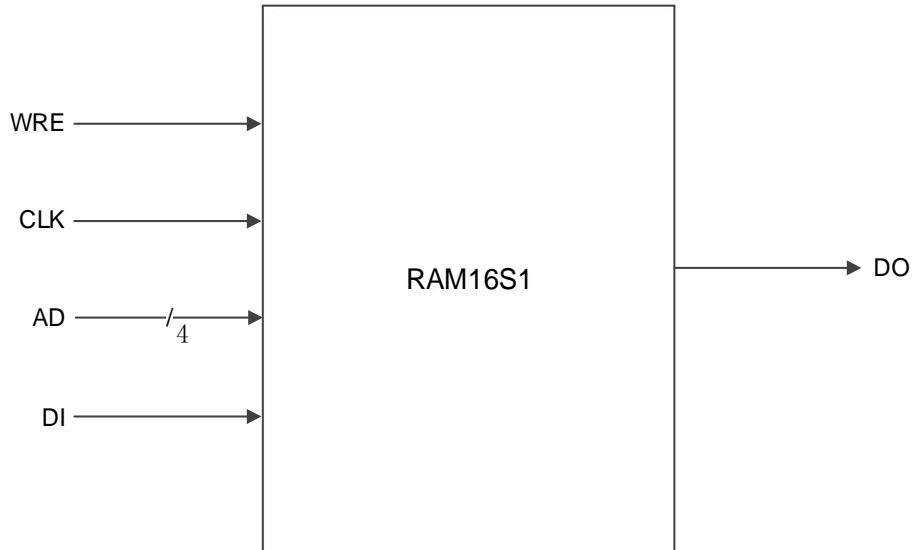
RAM16S1 是数据位宽为 1 的单端口 S-SRAM，读写地址相同，WRE 为高电平时进行写操作，此时会在 CLK 的上升沿将数据加载到存储器对应地址。读操作由地址确定输出 RAM 对应位置的数据。即 S-SRAM 由 CFU 的 LUT 配置实现，同步写入，异步读取。但如果应用需要，可使用与每个 LUT 关联的寄存器来实现同步读取功能。其时序波形图如图 5-1 所示。

图 5-1 RAM16S1 模式时序波形图



端口示意图

图 5-2 RAM16S1 端口示意图



端口介绍

表 5-2 RAM16S1 端口介绍

端口	I/O	描述
DI	Input	数据输入信号
CLK	Input	时钟输入信号
WRE	Input	写使能输入信号
AD[3:0]	Input	地址输入信号
DO	Output	数据输出信号

## 参数介绍

表 5-3 RAM16S1 参数介绍

参数	范围	默认	描述
INIT_0	16'h0000~16'hffff	16'h0000	RAM16S1 初始值

## 原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生，具体可参考第 6 章 IP 调用。

### Verilog 例化:

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

### Vhdl 例化:

```
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

### 原语介绍

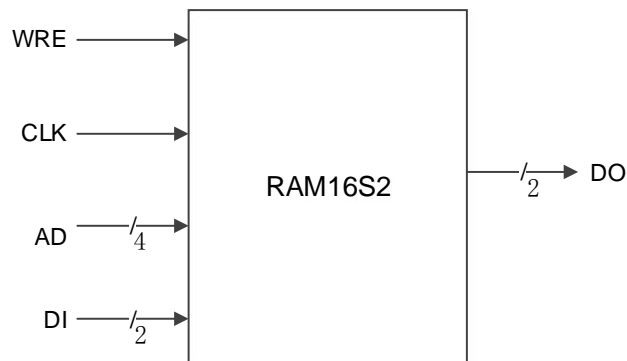
RAM16S2(16-Deep by 2-Wide Single-port S-SRAM)是地址深度为 16，数据位宽为 2 的单端口 S-SRAM。

### 功能描述

RAM16S2 是数据位宽为 2 的单端口 S-SRAM，读写地址相同，WRE 为高电平时进行写操作，此时会在 CLK 的上升沿将数据加载到存储器对应地址。读操作由地址确定输出 RAM 对应位置的数据。即 S-SRAM 由 CFU 的 LUT 配置实现，同步写入，异步读取。但如果应用需要，可使用与每个 LUT 关联的寄存器来实现同步读取功能。其时序波形图如图 5-1 所示。

### 端口示意图

图 5-3 RAM16S2 端口示意图



### 端口介绍

表 5-4 RAM16S2 端口介绍

端口	I/O	描述
DI[1:0]	Input	数据输入信号
CLK	Input	时钟输入信号
WRE	Input	写使能输入信号
AD[3:0]	Input	地址输入信号
DO[1:0]	Output	数据输出信号

### 参数介绍

表 5-5 RAM16S2 参数介绍

参数	范围	默认	描述
INIT_0~ INIT_1	16'h0000~16'hffff	16'h0000	RAM16S2 初始值

## 原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生，具体可参考第 6 章 IP 调用。

### Verilog 例化:

```
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 例化:

```
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

### 原语介绍

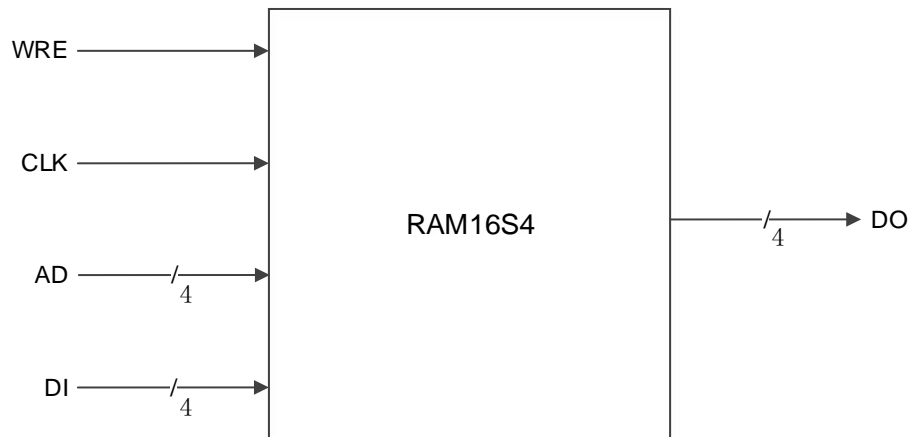
RAM16S4(16-Deep by 4-Wide Single-port S-SRAM)是地址深度为 16 ，数据位宽为 4 的单端口 S-SRAM。

### 功能描述

RAM16S4 是数据位宽为 4 的单端口 S-SRAM，读写地址相同，WRE 为高电平时进行写操作，此时会在 CLK 的上升沿将数据加载到存储器对应地址。读操作由地址确定输出 RAM 对应位置的数据。即 S-SRAM 由 CFU 的 LUT 配置实现，同步写入，异步读取。但如果应用需要，可使用与每个 LUT 关联的寄存器来实现同步读取功能。其时序波形图如图 5-1 所示。

### 端口示意图

图 5-4 RAM16S4 端口示意图



### 端口介绍

表 5-6 RAM16S4 端口介绍

端口	I/O	描述
DI[3:0]	Input	数据输入信号
CLK	Input	时钟输入信号
WRE	Input	写使能输入信号
AD[3:0]	Input	地址输入信号
DO[3:0]	Output	数据输出信号

## 参数介绍

表 5-7 RAM16S4 参数介绍

参数	范围	默认	描述
INIT_0~ INIT_3	16'h0000~16'hffff	16'h0000	RAM16S4 初始值

## 原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生，具体可参考第 6 章 IP 调用。

### Verilog 例化:

```
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 例化:

```
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

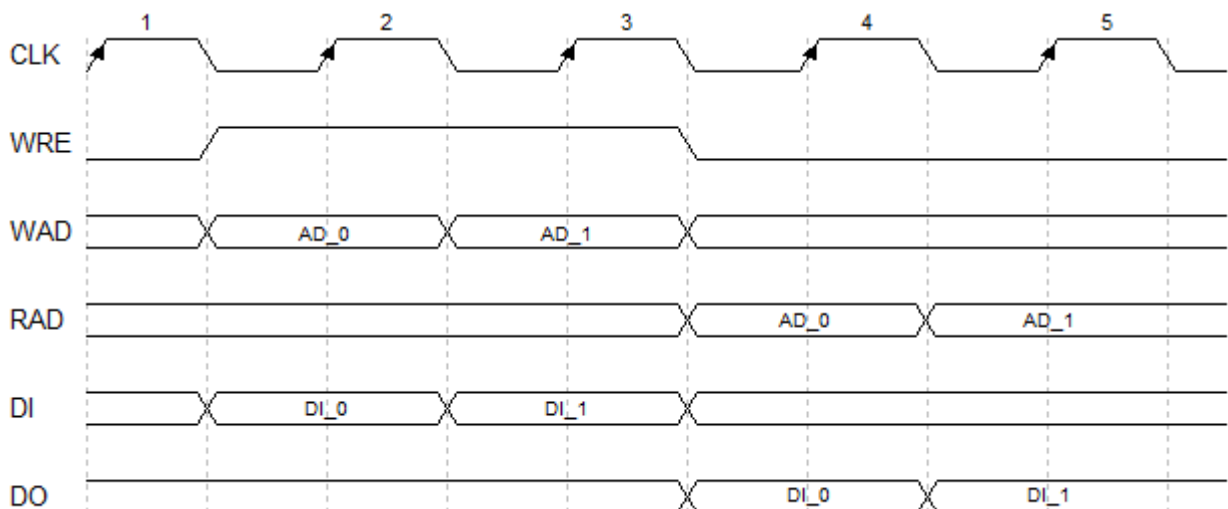
### 原语介绍

RAM16SDP1(16-Deep by 1-Wide Semi Dual-port S-SRAM)是地址深度为 16 ，数据位宽为 1 的伪双端口 S-SRAM。

### 功能描述

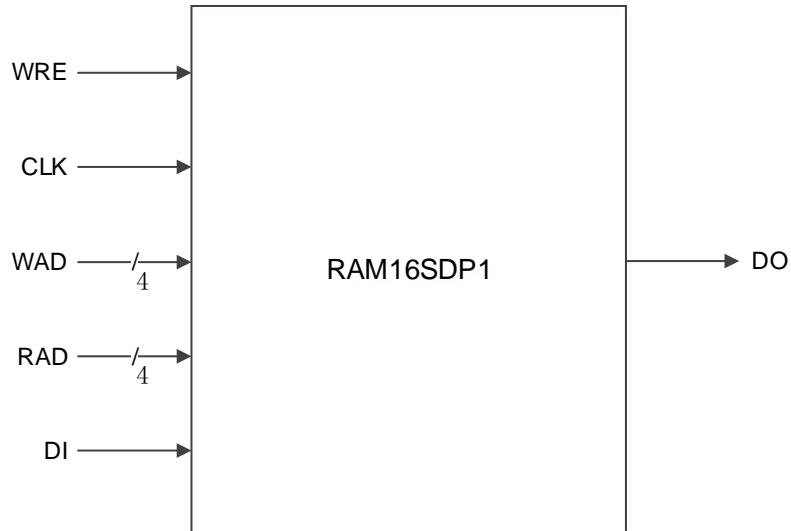
RAM16SDP1 是数据位宽为 1 的伪双端口 S-SRAM，具有两个地址，写地址 WAD 和读地址 RAD，这两个地址端口是异步的。WRE 为高电平时进行写操作，此时会在 CLK 的上升沿将数据加载到存储器对应写地址。读操作则由读地址确定输出 RAM 对应位置的数据。其时序波形图如图 5-5 所示。

图 5-5 RAM16SDP1 模式时序波形图



## 端口示意图

图 5-6 RAM16SDP1 端口示意图



## 端口介绍

表 5-8 RAM16SDP1 端口介绍

端口	I/O	描述
DI	Input	数据输入信号
CLK	Input	时钟输入信号
WRE	Input	写使能输入信号
WAD[3:0]	Input	写地址信号
RAD[3:0]	Input	读地址信号
DO	Output	数据输出信号

## 参数介绍

表 5-9 RAM16SDP1 参数介绍

参数	范围	默认	描述
INIT_0	16'h0000~16'hffff	16'h0000	RAM16SDP1 初始值

## 原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生，具体可参考第 6 章 IP 调用。

### Verilog 例化:

```
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 例化:**

```

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

### 原语介绍

RAM16SDP2(16-Deep by 2-Wide Semi Dual-port S-SRAM)是地址深度为 16 ，数据位宽为 2 的伪双端口 S-SRAM。

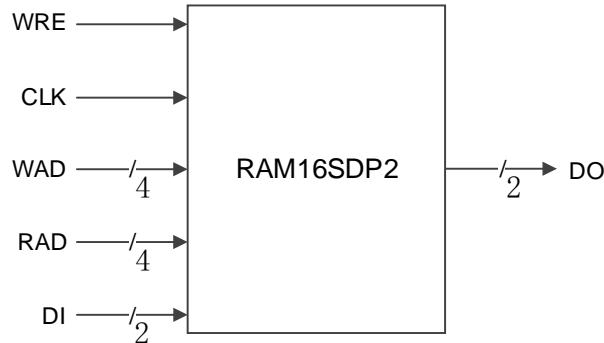
### 功能描述

RAM16SDP2 是数据位宽为 2 的伪双端口 S-SRAM，具有两个地址，写地址 WAD 和读地址 RAD，这两个地址端口是异步的。WRE 为高电平时进

行写操作，此时会在 CLK 的上升沿将数据加载到存储器对应写地址。读操作则由读地址确定输出 RAM 对应位置的数据。其时序波形图如图 5-5 所示。

### 端口示意图

图 5-7 RAM16SDP2 端口示意图



### 端口介绍

表 5-10 RAM16SDP2 端口介绍

端口	I/O	描述
DI[1:0]	Input	数据输入信号
CLK	Input	时钟输入信号
WRE	Input	写使能输入信号
WAD[3:0]	Input	写地址信号
RAD[3:0]	Input	读地址信号
DO[1:0]	Output	数据输出信号

### 参数介绍

表 5-11 RAM16SDP2 参数介绍

参数	范围	默认	描述
INIT_0~ INIT_1	16'h0000~16'hffff	16'h0000	RAM16SDP2 初始值

### 原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生，具体可参考第 6 章 IP 调用。

#### Verilog 例化:

```
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 例化:**

```

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

### 原语介绍

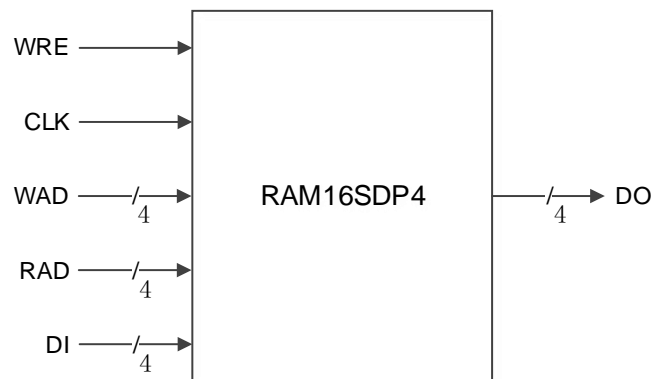
RAM16SDP4(16-Deep by 4-Wide Semi Dual-port S-SRAM)是地址深度为 16 ，数据位宽为 4 的伪双端口 S-SRAM。

### 功能描述

RAM16SDP4 是数据位宽为 4 的伪双端口 S-SRAM，具有两个地址，写地址 WAD 和读地址 RAD，这两个地址端口是异步的。WRE 为高电平时进行写操作，此时会在 CLK 的上升沿将数据加载到存储器对应写地址。读操作则由读地址确定输出 RAM 对应位置的数据。其时序波形图如图 5-5 所示。

### 端口示意图

图 5-8 RAM16SDP4 端口示意图



### 端口介绍

表 5-12 RAM16SDP4 端口介绍

端口	I/O	描述
DI[3:0]	Input	数据输入信号
CLK	Input	时钟输入信号
WRE	Input	写使能输入信号
WAD[3:0]	Input	写地址信号
RAD[3:0]	Input	读地址信号
DO[3:0]	Output	数据输出信号

### 参数介绍

表 5-13 RAM16SDP4 参数介绍

参数	范围	默认	描述
INIT_0~ INIT_3	16'h0000~16'hffff	16'h0000	RAM16SDP4 初始值



## 原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生，具体可参考第 6 章 IP 调用。

### Verilog 例化:

```
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 例化:

```
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

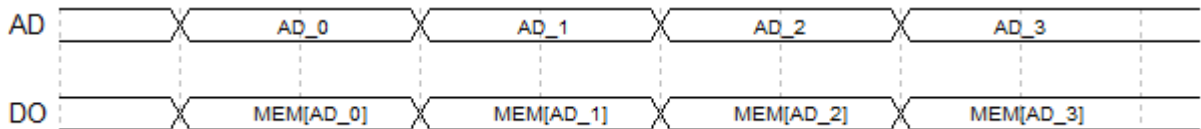
### 原语介绍

ROM16 是地址深度为 16，数据位宽为 1 的只读存储器，存储器的内容通过 INIT 进行初始化。

### 功能描述

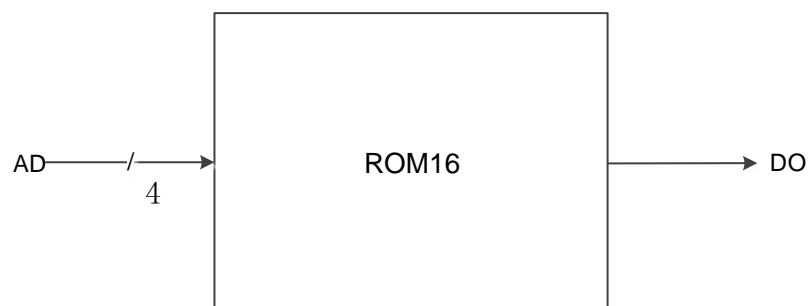
ROM16 是数据位宽为 1 的只读存储器，由地址确定输出存储在 ROM 对应位置的数据。其时序波形图如图 5-9 所示。

图 5-9 ROM16 模式时序波形图



### 端口示意图

图 5-10 ROM16 端口示意图



## 端口介绍

表 5-14 ROM16 端口介绍

端口	I/O	描述
AD[3:0]	Input	地址输入信号
DO	Output	数据输出信号

## 参数介绍

表 5-15 ROM16 参数介绍

参数	范围	默认	描述
INIT_0	16'h0000~16'hffff	16'h0000	ROM16 初始值

## 原语例化

可以直接实例化原语，也可以通过 IP Core Generator 工具产生，具体可参考第 6 章 IP 调用。

### Verilog 例化:

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

### Vhdl 例化:

```
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 调用

高云半导体 Gowin 云源软件的 IP Core Generator 支持 IP 核的界面调用，用户在界面中设置数据宽度、地址深度、写模式和读模式，Gowin 云源软件生成对应的 IP 模块，用户在使用中调用模块即可。此外，还有两种方式实现 B-SRAM、S-SRAM 的功能。一是用户可以通过调用高云半导体 Gowin 云源软件库文件，设置端口和参数生成需要的 IP 模块。二是代码综合时选择综合工具自动综合成 B-SRAM、S-SRAM 模式。

IP Core Generator 中，B-SRAM 模块可实现单端口模式、伪双端口模式、双端口模式以及只读模式，S-SRAM 模块可实现单端口模式、伪双端口模式和只读模式。下面 B-SRAM 以双端口模式，S-SRAM 以单端口模式为例来介绍 IP 调用。

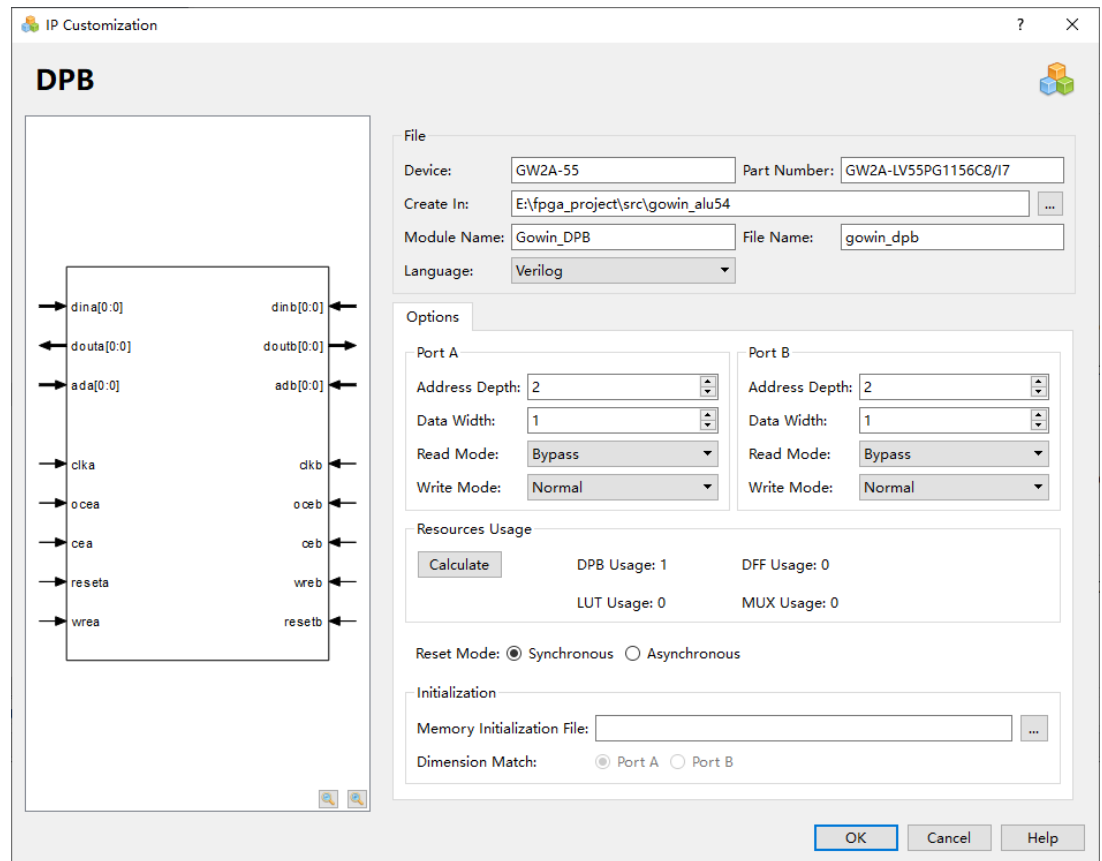
## 6.1 B-SRAM 双端口模式

B-SRAM 双端口工作模式 (DP)，可通过 DPB、DPX9B 原语实现。在 IP Core Generator 界面中，单击“DPB”，界面右侧会显示 DPB 的相关信息概要。

### IP 配置

在 IP Core Generator 界面中，双击“DPB”，弹出 DPB 的 IP Customization 窗口。该窗口包括“File”配置框、“Options”配置框、端口显示框图和“Help”按钮，如图 6-1 所示。

图 6-1 DPB 的 IP Customization 窗口结构



1. File 配置框。File 配置框用于配置产生的 IP 设计文件的相关信息。
  - Device: 显示已配置的 Device 信息;
  - Part Number: 显示已配置的 Part Number 信息;
  - Language: 配置产生的 IP 设计文件的硬件描述语言。选择右侧下拉列表框, 选择目标语言, 支持 Verilog 和 VHDL;
  - Module Name: 配置产生的 IP 设计文件的 module name。在右侧文本框可重新编辑模块名称。Module Name 不能与原语名称相同, 若相同, 则报出 Error 提示;
  - File Name: 配置产生的 IP 设计文件的文件名。在右侧文本框可重新编辑文件名称;
  - Create In: 配置产生的 IP 设计文件的目标路径。可在右侧文本框中重新编辑目标路径, 也可通过文本框右侧选择按钮选择目标路径。
2. Options 配置框。Options 配置框用于用户自定义配置 IP, 双端口模式分为 A、B 两个端口, Options 配置框如图 6-1 所示。
  - Data Width & Address Depth: 配置地址深度 (Address Depth) 和数据宽度 (Data Width)。当配置的地址深度和数据宽度无法通过单个模块实现时, IP Core 会实例化多个模块组合实现;
  - Resource Usage: 计算并显示当前容量配置上占用的 Block Ram、DFF、LUT、MUX 的资源情况;
  - Read/Write Mode: 配置读写模式。DPB 支持以下模式:

- 两种读模式: Bypass 和 Pipeline;
- 三种写模式: Normal、Write-Through、Read-before-Write;
- **Reset Mode:** 配置复位模式, 支持同步模式 “Synchronous” 和异步模式 “Asynchronous”;
- **Initialization:** 配置初始值。初始值以二进制、十六进制或带地址十六进制的格式写在初始化文件中。“Memory Initialization File” 选取的初始化文件可通过手写或者 IDE 菜单栏 “File > New > Memory Initialization File” 产生, 具体产生方式请参考文档 [SUG100](#), Gowin 云源软件用户指南, 初始化文件的格式请参考 [7](#) 初始化文件。

#### 注!

- Options 配置框中可独立配置 DPB 的 Port A 和 Port B 的地址深度、数据宽度和读写模式。
- DPB 的 Port A 和 Port B 是对同一块 memory 进行读写, 因此 Port A 和 Port B 的 Address Depth\*Data Width 的结果必须相同。
- Options 配置中的初始化文件 (Memory initialization File) 中的数据宽度应与 Dimension Match 选择的 Port 数据宽度一致。
- 如 Port A 和 Port B 的 Address Depth\*Data Width 的结果不同, 则会弹出 Error 提示信息。
- 如数据宽度不一致, 则产生的 DPB 实例 Init 值默认初始化为 0, 并且在 Output 窗口中, 会弹出如下提示信息: Error (MG2105): Initial values' width is unequal to user's width。

### 3. 端口显示框图

- 端口显示框图显示当前 IP Core 的配置结果示例框图, 输入输出端口的位宽根据 Options 配置实时更新, 如图 6-1 所示;
- Options 配置中的 Port A 和 Port B 的地址深度 Address Depth 配置影响地址的位宽, 数据位宽 Data Width 配置影响输入数据和输出数据的位宽。

### 4. Help 按钮

单击 “Help”, 显示 IP Core 的配置信息的页面。Help 页面包括当前 IP Core 的概要介绍, 以及 Options 各项配置的简要说明。

### IP 生成文件

IP 窗口配置完成后, 产生以配置文件 “File Name” 命名的三个文件, 以默认配置为例进行介绍:

- IP 设计文件 “gowin\_dpb.v” 为完整的 verilog 模块, 根据用户的 IP 配置, 产生实例化的 DPB;
- IP 设计使用模板文件 gowin\_dpb\_tmp.v, 为用户提供 IP 设计使用模板文件;
- IP 配置文件: “gowin\_dpb.ipc”, 用户可加载该文件对 IP 进行配置。

#### 注!

如配置中选择的语言是 VHDL, 则产生的前两个文件名后缀为.vhd。

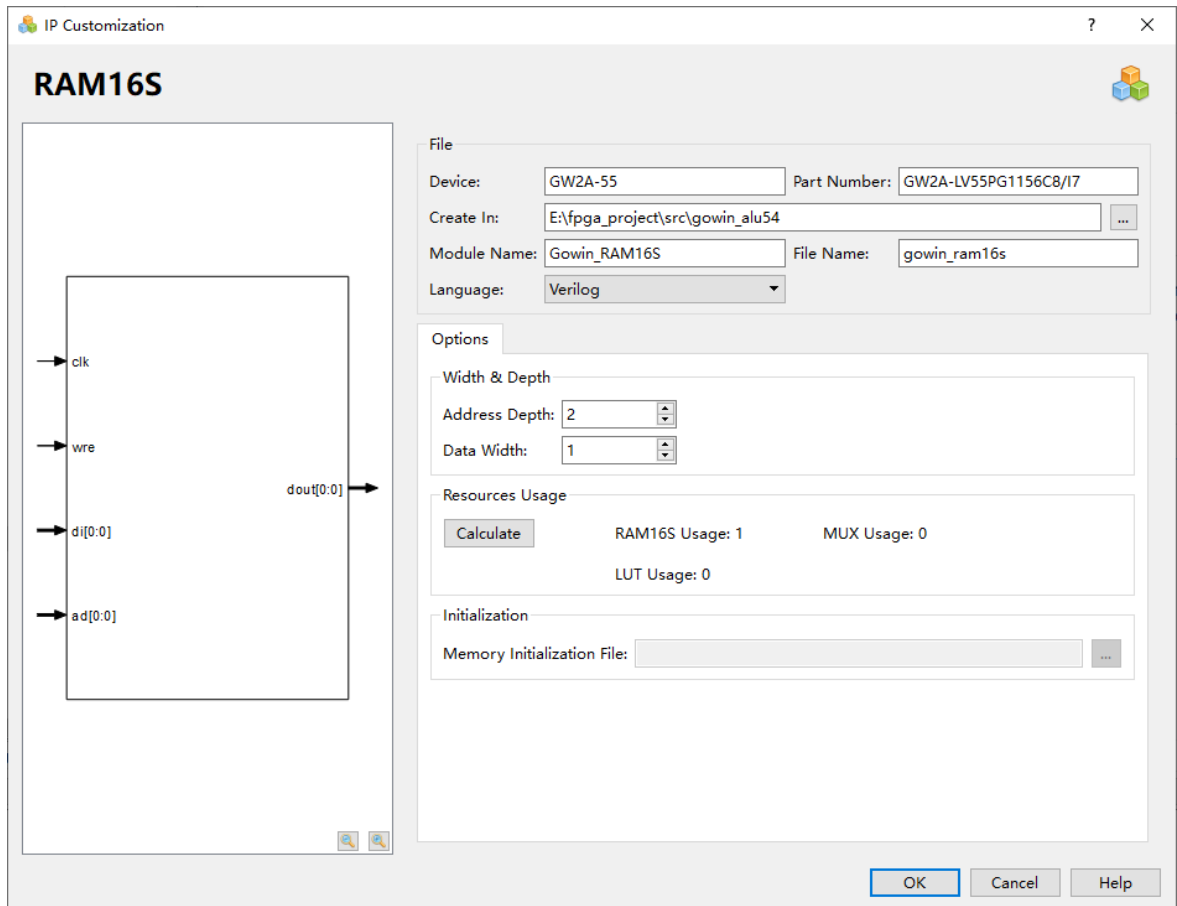
## 6.2 S-SRAM 单端口模式

RAM16S 为 S-SRAM 单端口工作模式, 可以通过 RAM16S1、RAM16S2、RAM16S4 原语实现。在 IP Core Generator 界面中, 单击“RAM16S”, 界面右侧会显示 RAM16S 的相关信息概要。

### IP 配置

在 IP Core Generator 界面中, 双击“RAM16S”, 弹出 RAM16S 的“IP Customization”窗口。该窗口包括“File”配置框、“Options”配置框、端口显示框图和“Help”按钮, 如图 6-2 所示。

图 6-2 RAM16S 的 IP Customization 窗口结构



1. **File 配置框。** File 配置框用于配置产生的 IP 设计文件的相关信息。RAM16S 的 File 配置框的使用和 B-SRAM 双端口模式类似, 具体请参考 [6.1B-SRAM 双端口模式的 File 配置框](#)。
2. **Options 配置框。** Options 配置框用于用户自定义配置 IP。Options 配置框如图 6-2 所示。RAM16S 的 Options 配置框的使用和 B-SRAM 双端口模式类似, 具体请参考 [6.1B-SRAM 双端口模式的 Options 配置框](#)。
3. **端口显示框图**
  - 端口显示框图显示当前 IP Core 的配置结果示例框图, 输入输出端口的位宽根据 Options 配置实时更新, 如图 6-2 所示;

- Options 配置中的地址深度 “Address Depth” 配置影响地址数据的位宽, 数据位宽 “Data Width” 配置影响输入数据和输出数据的位宽。

#### 4. Help 按钮

单击“Help”, 显示 IP Core 的配置信息的页面。Help 页面包括当前 IP Core 的概要介绍, 以及 Options 各项配置的简要说明。

#### IP 生成文件

IP 窗口配置完成后, 产生以配置文件 “File Name” 命名的三个文件, 以默认配置为例进行介绍:

- IP 设计文件 “gowin\_ram16s.v” 为完整的 verilog 模块, 根据用户的 IP 配置, 产生实例化的 RAM16S;
- IP 设计使用模板文件 gowin\_ram16s\_tmp.v, 为用户提供 IP 设计使用模板文件;
- IP 配置文件: “gowin\_ram16s.ipc”, 用户可加载该文件对 IP 进行配置。

#### 注!

如配置中选择的语言是 VHDL, 则产生的前两个文件名后缀为.vhd。



# 7 初始化文件

在 B-SRAM、S-SRAM 模式中,可以将存储器的每一位初始化为 0 或 1。初始值以二进制、十六进制或带地址十六进制的格式写在初始化文件中。

## 7.1 二进制格式 (Bin File)

Bin 文件是由二进制数 0 和 1 组成的文本文件,行数代表存储器的地址深度,列数代表存储器的数据宽度。

```
#File_format=Bin
#Address_depth=16
#Data_width=32
00001100000100000000100100010000
10000000010010000100000001000000
01000000100000001000000010000000
00100000100001001100000011000000
```

## 7.2 十六进制格式 (Hex File)

Hex 文件与 Bin 文件格式类似,由十六进制数 0~F 组成,行数代表存储器的地址深度,每一行数据的二进制位数,代表存储器的数据宽度。

```
#File_format=Hex
#Address_depth=8
#Data_width=16
3A40
A28E
0B52
1C49
D602
0801
03E6
```

4C18

## 7.3 带地址十六进制格式 (Address-Hex File)

Address-Hex 文件是在文件中对有数据记录的地址和数据都进行记录，地址和数据都是由十六进制数 0~F 组成，每行中冒号前面是地址，冒号后面是数据，文件中只对写入数据的地址和数据进行记录，没有记录的地址默认数据为 0。

```
#File_format=AddrHex
#Address_depth=256
#Data_width=16
9:FFFF
23:00E0
2a:001F
30:1E00
```

