SALEAGLE® Series FPGA

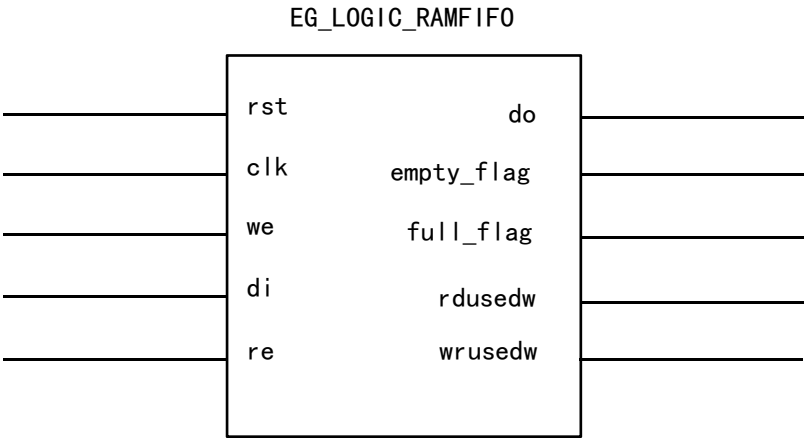
Libraries Guide for HDL Designs

UG301（v1.4）2022 年 10 月

Confidential

目 录

1 SALEAGLE®（以下简称为 EG）_LOGIC_RAMFIFO



1.1 简介

EG_LOGIC_RAMFIFO 是 TD 软件集成的软核 FIFO 控制器，与硬核 FIFO 控制器相比，额外支持实时深度指示与 Showahead 模式。

1.2 端口描述

Name	Direction	Width	Function
rst	input	1	异步复位信号，高有效
clk	input	1	读写时钟
we	input	1	写使能
di	input	DATA_WIDTH	数据输入
re	input	1	读使能
do	output	DATA_WIDTH	数据输出
empty_flag	output	1	空信号指示
full_flag	output	1	满信号指示
rdusedw	output	DATA_WIDTH+1	可读数据个数
wrusedw	output	DATA_WIDTH+1	已写数据个数

1.3 参数描述

Attribute	Allowed Values	Default	Descriptions
DATA_WIDTH	1-432	8	fifo 读写数据位宽
ADDR_WIDTH	2-15	8	fifo 深度位宽
SHOWAHEAD	0,1	0	1 表示 SHOWAHEAD 模式，0 表示普通模式

1.4 使用方式

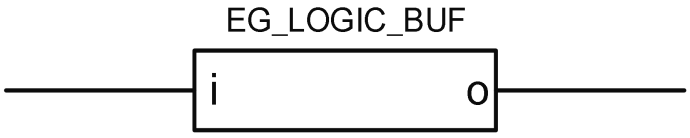
使用方式	是否支持
原语例化	是
IP Generator	是

1.5 Verilog 例化示例

```
EG_LOGIC_RAMFIFO #(
    .DATA_WIDTH(8),
    .ADDR_WIDTH(8),
    .SHOWAHEAD(1)
)
logic_ramfifo(
    .rst(rst),
    .di(di),
    .clk(clk),
    .we(we),
    .do(do),
    .re(re),
    .empty_flag(empty_flag),
    .full_flag(full_flag),
    .rdusedw(rdusedw),
    .wrusedw(wrusedw)
);
```

注：原语库中 gray2bin、gray_count、showahead_buffer 为 RAMFIFO 子模块，不建议用户例化使用。

2 EG_LOGIC_BUF



2.1 简介

LOGIC BUF 是 FPGA 内部的低延时布线资源。

2.2 端口描述

Name	Direction	Width	Function
i	I	1	Buffer 输入端口
o	O	1	Buffer 输出端口

2.3 参数描述

无

2.4 使用方式

使用方式	是否支持
原语例化	是
IP Generator	否

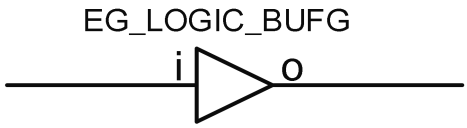
2.5 Verilog 例化模板

```
EG_LOGIC_BUF BUF_inst(
    .o(O),
    .i(I)
);
```

2.6 VHDL 例化模板

```
BUF_inst : EG_LOGIC_BUF
port map (
    o => O, -- 1-bit output: Clock buffer output
    i=> I  -- 1-bit input: Clock buffer input
   );
```

3 EG_LOGIC_BUFG



3.1 简介

BUFG 是具有高扇出的全局时钟缓冲器，延时和抖动小。常用于时钟网络和其他高扇出的网路（如复位信号、时钟使能等）。本 IP 用于将普通信号走线到 BUFG 上。

3.2 端口描述

Name	Direction	Width	Function
i	I	1	BUFG 输入端口
o	O	1	BUFG 输出端口

3.3 参数描述

无

3.4 使用方式

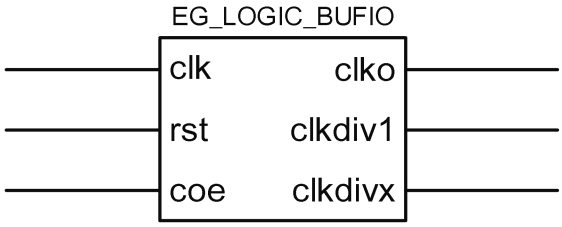
使用方式	是否支持
原语例化	是
IP Generator	是

3.5 Verilog 例化模板

```
EG_LOGIC_BUFG BUFG_inst(
    .o(O),
    .i(I)
);
```

3.6 VHDL 例化模板

4 EG_LOGIC_BUFIO



4.1 简介

BUFIO 是 IO 上的一种时钟分频器，每个 I/O 组有两个该分频器。Clko 输出时钟只能供 IO 使用，clkdiv1 和 clkdivx 可以供全局时钟网使用。

4.2 端口描述

Name	Direction	Width	Function
clki	I	1	输入时钟信号；
rst	I	1	复位信号，高有效；
coe	I	1	时钟使能信号,高有效；
clko	O	1	时钟输出信号,和 clk_i 同频不同 ioclk bank；
clkdiv1	O	1	输出时钟等于输入时钟频率；
clkdivx	O	1	clk_i x 分频输出,x=1,2,4；

4.3 参数描述

Attribute	Allowed Values	Default	Descriptions
GSR	DISABLE/ENABLE	"DISABLE"	全局复位
DIV	2,4	2	分频系数
STOPCLK	ENABLE,DISABLE	"DISABLE"	时钟关闭

4.4 使用方式

使用方式	是否支持
原语例化	是
IP Generator	否

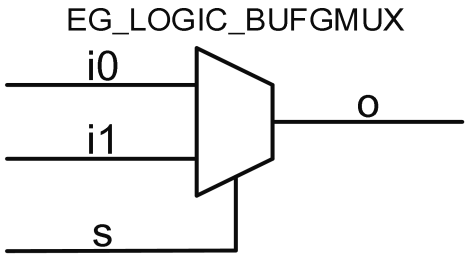
4.5 Verilog 例化模板

```
EG_LOGIC_BUFIO BUFIO_inst(
    .clki(clki),
```

```
        .rst(rst),
        .coe(coe),
        .clko(clko),
        .clkdiv1(clkdiv1),
        .clkdivx(clkdivx)
    );
```

4.6 VHDL 例化模板

5 EG_LOGIC_BUFGMUX



5.1 简介

BUFGMUX 是一个全局时钟缓冲选择器，输出时钟可以在两路输入信号中进行选择。当选择信号(s)为高时，选择 i1 作为输入；当选择信号(s)为低时，选择 i0 作为输入信号。

5.2 端口描述

Name	Direction	Width	Function
i0	I	1	BUFG 输入端口 0
i1	I	1	BUFG 输入端口 1
s	I	1	选择信号
o	O	1	BUFG 输出端口

5.3 真值表

Input			Output
i0	i1	s	o
i0	X	0	i0
X	i1	1	i1

5.4 参数描述

Attribute	Allowed Values	Default	Descriptions
INIT_OUT	"0","1"	"0"	输出端口初始化值
PRESELECT_I0	"TRUE","FALSE"	"TRUE"	预先选择端口 i0 为输入
PRESELECT_I1	"TRUE","FALSE"	"FALSE"	预先选择端口 i1 为输入

5.5 使用方式

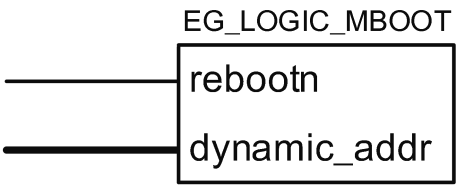
使用方式	是否支持
原语例化	是
IP Generator	否

5.6 Verilog 例化模板

```
EG_LOGIC_BUFGMUX #(
    .INIT_OUT     ("0"),
    .PRESELECT_I0 ("TRUE"),
    .PRESELECT_I1 ("FALSE"))
BUFGMUX_inst(
    .o(o),
    .i0(i0),
    .i1(i1),
    .s(s)
);
```

5.7 VHDL 例化模板

6 EG_LOGIC_MBOOT



6.1 简介

EG_LOGIC_MBOOT 是 EAGLE 系列 FPGA 多重启动控制模块。当地址模式为 STATIC 时，将使用参数设置的静态启动地址，当地址模式为 DYNAMIC 时，将使用 dynamic_addr 端口动态设置的地址。

6.2 端口描述

Name	Direction	Width	Function
rebootn	I	1	重启控制信号，低有效
dynamic_addr	I	8	动态重启地址,SPI Flash 高 8 位地址

6.3 参数描述

Attribute	Allowed Values	Default	Descriptions
ADDR_SOURCE_SEL	"STATIC" " DYNAMIC"	"STATIC"	启动模式，动态或者静态
STATIC_ADDR	8 位地址	8'b00000000	静态启动地址,该 8 位地址为 Flash 地址的高 8 位

6.4 使用方式

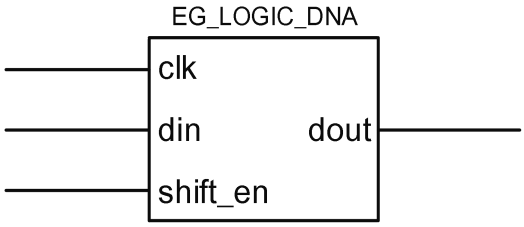
使用方式	是否支持
原语例化	是
IP Generator	否

6.5 Verilog 例化模板

```
EG_LOGIC_MBOOT #(
    .ADDR_SOURCE_SEL("STATIC"),
    .STATIC_ADDR(8'b00000000))
MBOOT_inst (
    .rebootn(reboot_n),
    .dynamic_addr(addr)
);
```

6.6 VHDL 例化模板

7 EG_LOGIC_DNA



7.1 简介

EG_LOGIC_DNA 是 EAGLE 系列 FPGA 64 位 DNA 信息读取接口模块。

7.2 端口描述

Name	Direction	Width	Function
clk	I	1	时钟输入
din	I	1	用户数据输入
shift_en	I	1	移位使能信号，高有效
dout	O	1	DNA 数据输出，LSB 优先

7.3 参数描述

无

7.4 使用方式

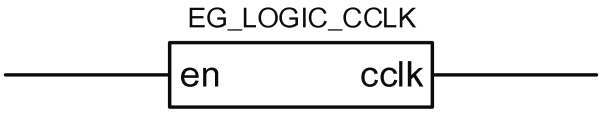
使用方式	是否支持
原语例化	是
IP Generator	否

7.5 Verilog 例化模板

```
EG_LOGIC_DNA DNA_inst(
    .dout(dout),
    .clk(clk),
    .din(din),
    .shift_en(shift_en)
);
```

7.6 VHDL 例化模板

8 EG_LOGIC_CCLK



8.1 简介

EG_LOGIC_CCLK 模块，EAGLE 系列 FPGA 的配置时钟控制模块，用于配置 FPGA 的配置时钟信号频率。

8.2 端口描述

Name	Direction	Width	Function
en	I	1	使能
cclk	O	1	输入时钟

8.3 参数描述

Attribute	Allowed Values	Default	Descriptions
FREQ	"4.5", "6.5", "10.0", "30.0"	"4.5"	设置输出时钟的频率，单位 MHz

8.4 使用方式

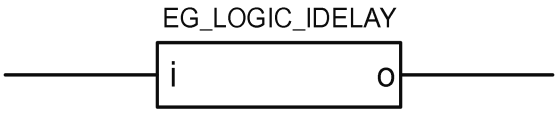
使用方式	是否支持
原语例化	是
IP Generator	否

8.5 Verilog 例化模板

```
EG_LOGIC_CCLK #(
    .FREQ("4.5"))
CCLK_inst (
    .cclk(cclk),
    .en(en)
);
```

8.6 VHDL 例化模板

9 EG_LOGIC_IDELAY



9.1 简介

EG_LOGIC_IDELAY 是 EAGLE FPGA 延时控制单元。4.2.266 软件版本后，EG 系列器件可以不用再调用该 IP，在 IO CONSTRAINT 图形界面处设置。

9.2 端口描述

Name	Direction	Width	Function
i	I	1	信号输入端
o	O	1	信号输出端

9.3 参数描述

Attribute	Allowed Values	Default	Descriptions
INDEL	0~31	0	初始化延时为 0.2ns，最大延时 3.8ns

9.4 使用方式

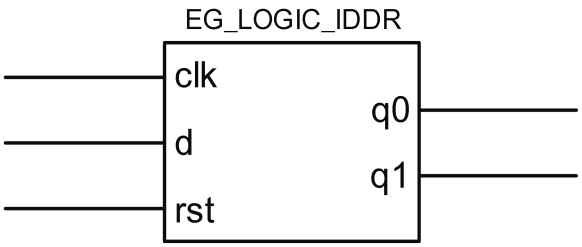
使用方式	是否支持
原语例化	是，4.2.266 版本以后可以直接在 IO CONSTRAINT 图形界面设置延时长度
IP Generator	否

9.5 Verilog 例化模板

```
EG_LOGIC_IDELAY #(
    .INDEL(0))
IDELAY_inst(
    .o(o),
    .i(i)
);
```

9.6 VHDL 例化模板

10 EG_LOGIC_IDDR



10.1 简介

EG_LOGIC_IDDR 是 EAGLE 系列 FPGA IO 中将双沿采样时钟数据变换为单沿数据的单元。，具有数据对齐的双速率 D 触发器，并具有同步/异步复位功能。可以将输入的 DDR 数据转为 2 位数据同步到内部时钟网络。

10.2 端口描述

Name	Direction	Width	Function
clk	I	1	采样时钟
d	I	1	IO 上输入的 DDR 数据
rst	I	1	复位信号
q0	O	1	1 位上升沿数据输出
q1	O	1	1 位下降沿数据输出

10.3 参数描述

Attribute	Type	Allowed Values	Default	Descriptions
ASYNCRST	String	"ENABLE","DISABLE"	"ENABLE"	默认异步复位使能
PIPEMODE	String	"PIPED","NONE"	"PIPED"	Q0,Q1 数据对齐模式

10.4 使用方式

使用方式	是否支持
原语例化	是
IP Generator	是

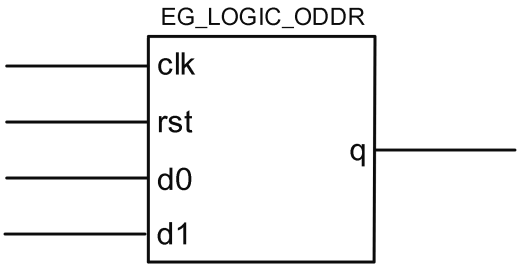
10.5 Verilog 例化模板

```
EG_LOGIC_IDDR#(
    .ASYNCRST("ENABLE"),
```

```
        .PIPEMODE("PIPED"))

    IDDR_inst(

        .q1(q1),

        .q0(q0),

        .clk(clk),

        .d(d),

        .rst(rst)

    );
```

10.6 VHDL 例化模板

11 EG_LOGIC_ODDR



11.1 简介

EG_LOGIC_ODDR 是EAGLE 系列FPGA IO 中将内部单沿数据变换为双沿数据后输出的单元,并具有同步/异步复位功能。可以将数据双倍率输出到 IO 口上，输出时钟取决于原语 CLK。

11.2 端口描述

Name	Direction	Width	Function
q	O	1	1 位 DDR 双沿输出数据
clk	I	1	同步时钟
d0	I	1	1 位上升沿输入数据
d1	I	1	1 位下降沿输入数据
rst	I	1	复位，高有效

11.3 参数描述

Attribute	Type	Allowed Values	Default	Descriptions
ASYNCRST	String	"ENABLE","DISABLE"	"ENABLE"	异步复位使能

11.4 使用方式

使用方式	是否支持
原语例化	是
IP Generator	是

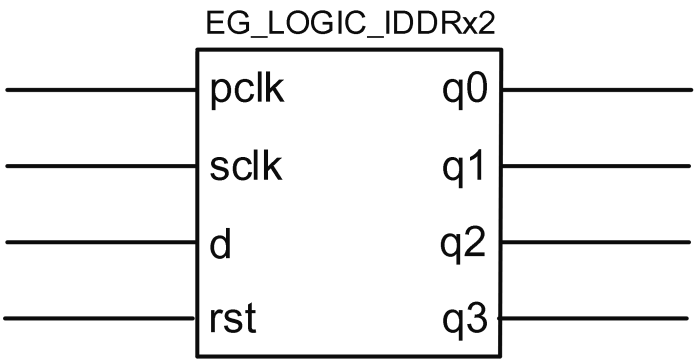
11.5 Verilog 例化模板

```
EG_LOGIC_ODDR#(
    .ASYNCRST("ENABLE"))
ODDR_inst(
    .q(q),
```

```
        .d1(d1),
        .d0(d0),
        .clk(clk),
        .rst(rst)
    );
```

11.6 VHDL 例化模板

12 EG_LOGIC_IDDRx2



12.1 简介

EG_LOGIC_IDDR x2 是 EAGLE 系列 FPGA IO IO 上的 1：4 串转并单元。功能与 IDDR 相同，但 IDDR x2 支持更高的 IO 速度。

12.2 端口描述

Name	Direction	Width	Function
pclk	I	1	1/2 sclk 的时钟输入
sclk	I	1	1/2 数据率的时钟输入
rst	I	1	复位，高有效
d	I	1	串行数据输入
q0	O	1	并行输出的第 0bit
q1	O	1	并行输出的第 1bit
q2	O	1	并行输出的第 2bit
q3	O	1	并行输出的第 3bit

12.3 参数描述

Attribute	Type	Allowed Values	Default	Descriptions
ASYNCRST	String	"ENABLE","DISABLE"	"ENABLE"	复位使能

12.4 使用方式

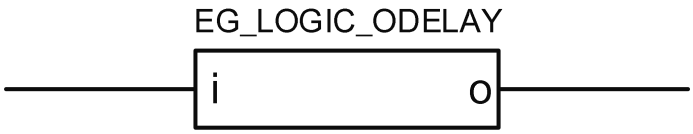
使用方式	是否支持
原语例化	是
IP Generator	否

12.5 Verilog 例化模板

```
EG_LOGIC_IDDRx2#(
    .ASYNCRST("ENABLE"))
IDDRx2_inst(
    .q3(q3),
    .q2(q2),
    .q1(q1),
    .q0(q0),
    .pclk(pclk),
    .sclk(sclk),
    .d(d),
    .rst(rst)
);
```

12.6 VHDL 例化模板

13 EG_LOGIC_ODELAY



13.1 简介

EG_LOGIC_ODELAY 是 EAGLE 系列 FPGA 的输出管脚延时控制模块。使用 4.2.266 软件版本以后，可以不再调用该单元，而直接在 IO CONSTRAINT 界面设置延时时间。

13.2 端口描述

Name	Direction	Width	Function
i	I	1	信号输入端
o	O	1	信号输出端

13.3 参数描述

Attribute	Allowed Values	Default	Descriptions
OUTDEL	0~3	0	4 挡可调

13.4 使用方式

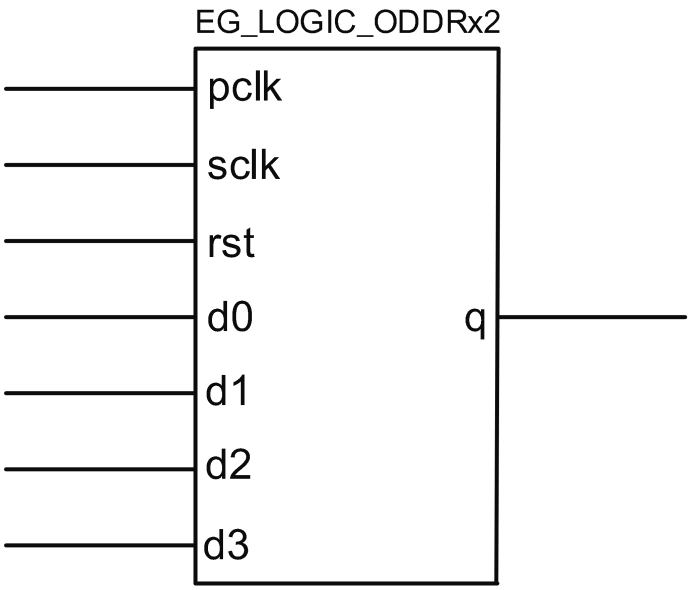
使用方式	是否支持
原语例化	是，4.2.266 软件版本以后，可以不再调用该单元，而直接在 IO CONSTRAINT 界面设置延时时间
IP Generator	否

13.5 Verilog 例化模板

```
EG_LOGIC_ODELAY #(
    .OUTDEL(0))
ODELAY_inst(
    .o(o),
    .i(i)
);
```

13.6 VHDL 例化模板

14 EG_LOGIC_ODDRx2



14.1 简介

EG_LOGIC_ODDR x2 是 EAGLE 系列 FPGA IO 内部的 4：1 并转串单元。功能与 ODDR 相同，但 ODDR x2 支持更高的 IO 速度。

14.2 端口描述

Name	Direction	Width	Function
pclk	I	1	1/2 sclk 的时钟输入
sclk	I	1	1/2 数据率的时钟输入
rst	I	1	复位，高有效
d0	I	1	并行数据第 0 bit
d1	I	1	并行数据第 1 bit
d2	I	1	并行数据第 2 bit
d3	I	1	并行数据第 3 bit
q	O	1	从 IO 上输出的串行数据

14.3 参数描述

Attribute	Allowed Values	Default	Descriptions
ASYNCRST	"ENABLE", "DISABLE"	"ENABLE"	默认异步复位

14.4 使用方式

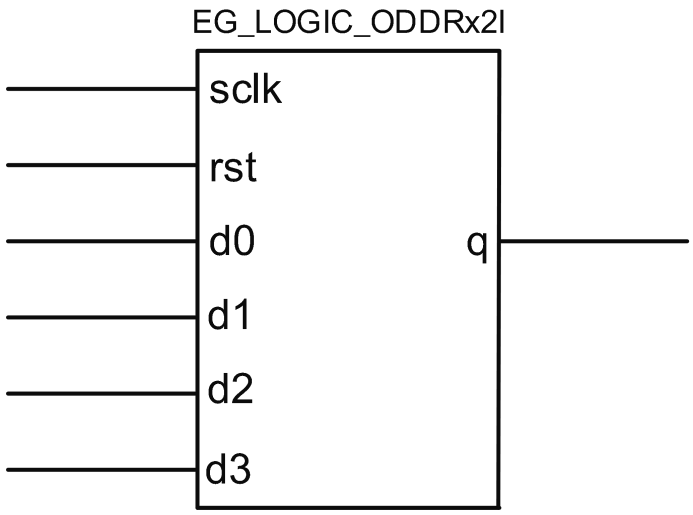
使用方式	是否支持
原语例化	是
IP Generator	否

14.5 Verilog 例化模板

```
EG_LOGIC_ODDRx2 #(
    .ASYNCRST("ENABLE"))
ODDRx2_inst(
    .q(q),
    .pclk(pclk),
    .sclk(sclk),
    .d3(d3),
    .d2(d2),
    .d1(d1),
    .d0(d0),
    .rst(rst)
);
```

14.6 VHDL 例化模板

15 EG_LOGIC_ODDRx2I



15.1 简介

EG_LOGIC_ODDRx2I 是 FPGA IO 内部的 4：1 并转串单元。与 ODDRx2 相比，ODDRx2L 模式直接使用内部 sclk 的 2 分频作为 pclk，节省 1 个 clk。数据输出比 ODDRx2 模式晚一个 sclk 时钟周期。

15.2 端口描述

Name	Direction	Width	Function
sclk	I	1	1/2 数据率的时钟输入
rst	I	1	复位，高有效
d0	I	1	第 0bit 并行数据
d1	I	1	第 1bit 并行数据
d2	I	1	第 2bit 并行数据
d3	I	1	第 3bit 并行数据
q	O	1	串行输出数据

15.3 参数描述

Attribute	Type	Allowed Values	Default	Descriptions
ASYNCRST	String	"ENABLE","DISABLE"	"ENABLE"	默认异步复位

15.4 使用方式

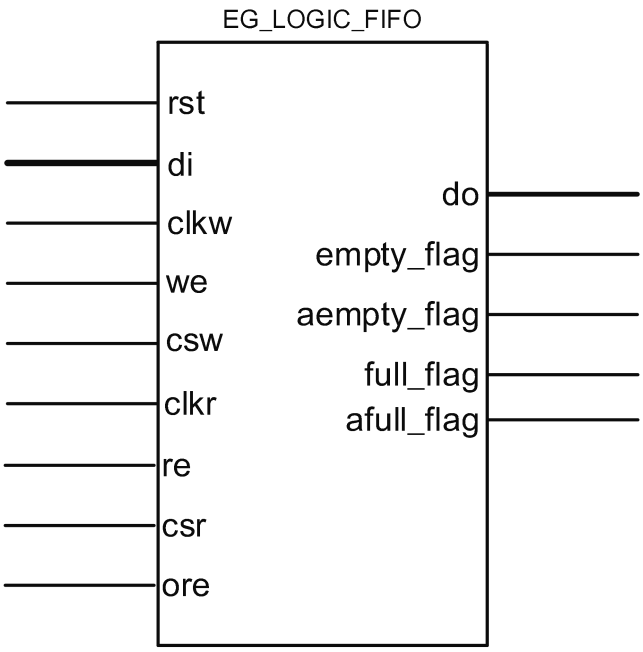
使用方式	是否支持
原语例化	是
IP Generator	否

15.5 Verilog 例化模板

```
EG_LOGIC_ODDRx2I  #(
    .ASYNCRST("ENABLE"))
ODDRx2I_inst(
    .q(q),
    .sclk(sclk),
    .d3(d3),
    .d2(d2),
    .d1(d1),
    .d0(d0),
    .rst(rst)
);
```

15.6 VHDL 例化模板

16 EG_LOGIC_ FIFO



16.1 简介

EG_LOGIC_FIFO 是 Eagle FPGA 的 FIFO 控制器模块，可以支持同步/异步以及混合端口的 FIFO 工作模式。

16.2 端口描述

Name	Direction	Width	Function
rst	I	1	复位，高有效
di	I	DATA_WIDTH_W	FIFO 写入数据
clkw	I	1	FIFO 写端口时钟输入，默认上升沿有效
we	I	1	FIFO 写使能， 1 为写入操作， 0 无操作
csw	I	3	写端口片选信号
do	O	DATA_WIDTH_R	FIFO 数据输出
clkr	I	1	读端口时钟输入，默认上升沿有效
re	I	1	FIFO 读使能， 1 为读操作， 0 无操作
csr	I	3	读端口片选信号
ore	I	1	读输出使能信号
empty_flag	O	1	FIFO 读空标志，和 clkr 同步
aempty_flag	O	1	FIFO 几乎空标识，可设定，和 clkr 同步
full_flag	O	1	FIFO 满标识，和 clkw 同步
afull_flag	O	1	FIFO 几乎满标识，可设定，和 clkw 同步

16.3 参数描述

Attribute	Allowed Values	Default	Descriptions
DATA_WIDTH_W	1…..9*48	9	FIFO 写入数据宽度
DATA_DEPTH_W	512, 1024, 2048, 4096, 8192	1024	FIFO 写入深度
DATA_WIDTH_R	1…..9*48	9	FIFO 读出数据宽度
DATA_DEPTH_R	512, 1024, 2048, 4096, 8192	1024	FIFO 读出深度
MODE	"FIFO8K"	"FIFO8K"	FIFO 类型
REGMODE_W	"NOREG" , OUTREG	"NOREG"	FIFO 写入寄存器
REGMODE_R	"NOREG" OUTREG	"NOREG"	FIFO 读出寄存器
E	0	0	FIFO 空标志深度设置
AE	1- DATA_DEPTH_W-1	6	FIFO 快空标志深度设置
AF	1- DATA_DEPTH_W-1	DATA_DEPTH_W-6	FIFO 快满标志深度设置
F	DATA_DEPTH_W	DATA_DEPTH_W	FIFO 满标志深度设置
GSR	"ENABLE" DISABLE	"ENABLE"	全局复位使能
RESETMODE	"ASYNC" SYNC	"ASYNC"	FIFO 复位模式
ASYNC_RESET_RELEASE	"SYNC" ASYNC	"SYNC"	FIFO 复位模式
ENDIAN	"LITTLE" BIG	"LITTLE"	FIFO 大小端对齐模式

16.4 使用方式

使用方式	是否支持
原语例化	是
IP Generator	是

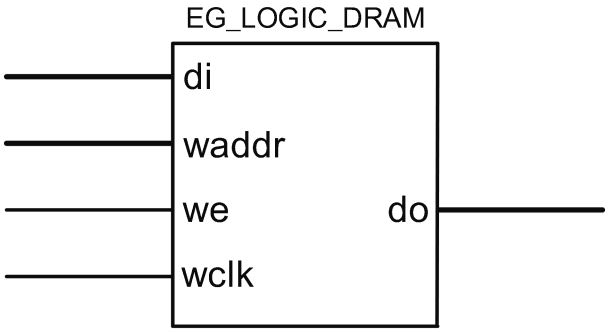
16.5 Verilog 例化模板

```
EG_LOGIC_FIFO#(
    .DATA_WIDTH_W(9),
    .DATA_DEPTH_W(1024),
    .MODE("FIFO8K"),
    .REGMODE_W("NOREG"),
    .REGMODE_R("NOREG"),
    .GSR("ENABLE"),
    .RESETMODE("ASYNC"),
```

```
        .ASYNC_RESET_RELEASE("SYNC"),
        .ENDIAN("LITTLE"),
    )
    FIFO_inst(
        .rst(rst),
        .di(di),
        .clkw(clkw),
        .we(we),
        .csw(csw),
        .do(do),
        .clkr(clkr),
        .re(re),
        .csr(csr),
        .ore(ore),
        .empty_flag(empty_flag),
        .aempty_flag(aempty_flag),
        .full_flag(full_flag),
        .afull_flag(afull_flag)
    );
```

16.6 VHDL 例化模板

17 EG_LOGIC_DRAM



17.1 简介

EG_LOGIC_DRAM 是 FPGA 内部的分布式 RAM 例化模块，可以通过配置作为单口、单双口 RAM 使用。

17.2 端口描述

Name	Direction	Width	Function
di	I	DATA_WIDTH_W	写数据
waddr	I	ADDR_WIDTH_W	写地址
we	I	1	写使能，高有效
wclk	I	1	时钟，高有效
do	O	DATA_WIDTH_W	读数据
raddr	I	ADDR_WIDTH_W	读地址

17.3 参数描述

Attribute	Allowed Values	Default	Descriptions
DATA_WIDTH_W	>0	4	写数据位宽
ADDR_WIDTH_W	>0	DATA_WIDTH_W	写地址位宽
DATA_WIDTH_R	>0	4	读数据位宽
ADDR_WIDTH_R	>0	DATA_WIDTH_W	读地址位宽
DATA_DEPTH_W	2 ** ADDR_WIDTH_R	2 ** ADDR_WIDTH_R	写数据深度
DATA_DEPTH_R	2 ** ADDR_WIDTH_R	2 ** ADDR_WIDTH_R	读数据深度
INIT_FILE	文件路径或“NONE”	“NONE”	预加载的初始值文件

17.4 使用方式

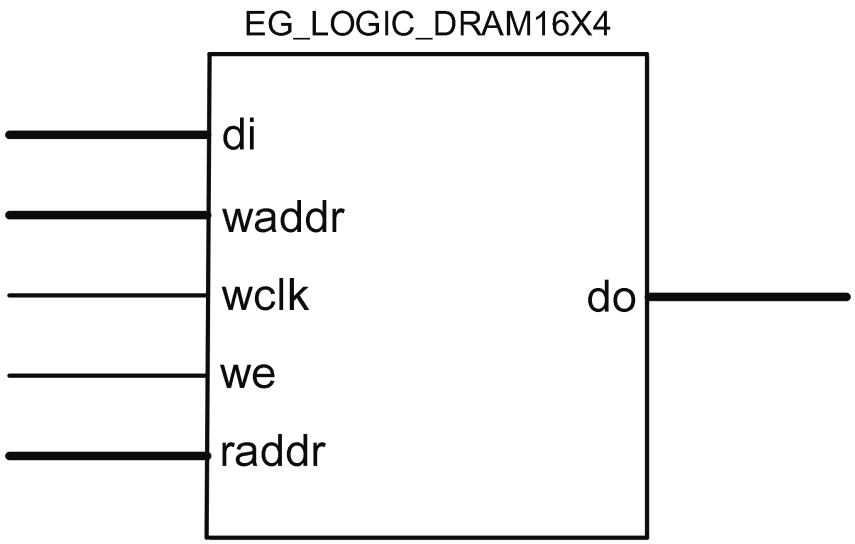
使用方式	是否支持
原语例化	是
IP Generator	是

17.5 Verilog 例化模板

```
EG_LOGIC_DRAM#(
    . DATA_WIDTH_W (9),
    . ADDR_WIDTH_W(10),
    . DATA_DEPTH_W(1024),
    . DATA_WIDTH_R (9),
    . ADDR_WIDTH_R(10),
    . DATA_DEPTH_R(1024),
    . INIT_FILE("NONE"))
DRAM_inst(
    .di(di),
    .waddr(waddr),
    .we(we),
    .wclk(wclk),
    .do(do),
    .raddr(raddr)
);
```

17.6 VHDL 例化模板

18 EG_LOGIC_ DRAM16X4



18.1 简介

EG_LOGIC_DRAM16X4 是 FPGA 内部的分布式 RAM 例化基本单元，可以通过配置作为单双口 RAM 使用。

18.2 端口描述

Name	Direction	Width	Function
di	I	4	写数据
waddr	I	4	写地址
we	I	1	写使能
wclk	I	1	时钟
raddr	I	4	读地址
do	O	4	读出数据

18.3 参数描述

Attribute	Allowed Values	Default	Descriptions
INIT_D0	16'h0000	16'h0000	初始化数据
INIT_D1	16'h0000	16'h0000	初始化数据
INIT_D2	16'h0000	16'h0000	初始化数据
INIT_D3	16'h0000	16'h0000	初始化数据

18.4 使用方式

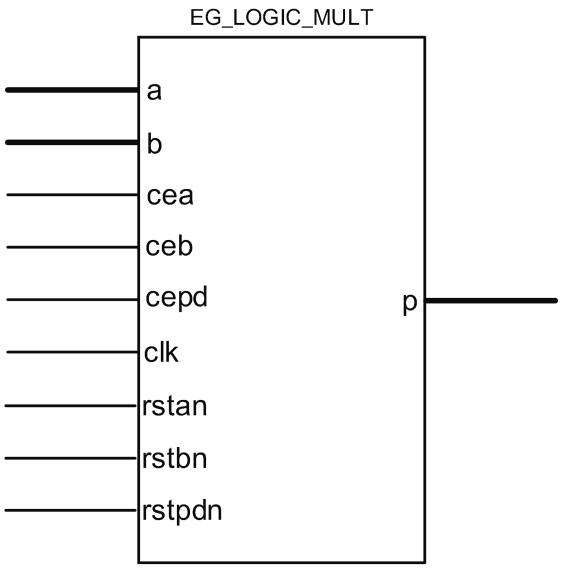
使用方式	是否支持
原语例化	是
IP Generator	否

18.5 Verilog 例化模板

```
EG_LOGIC_DRAM16X4(
    .INIT_D0(16'h0000),
    .INIT_D1(16'h0000),
    .INIT_D2(16'h0000),
    .INIT_D3(16'h0000),
DRAM16X4_inst(
    .di(di),
    .waddr(waddr),
    .we(we),
    .wclk(wclk),
    .raddr(raddr),
    .do(do)
);
```

18.6 VHDL 例化模板

19 EG_LOGIC_ MULT



19.1 简介

EG_LOGIC_MULT 是 FPGA 内部 18X18 硬件乘法器。

19.2 端口描述

Name	Direction	Width	Function
p	O	OUTPUT_WIDTH	乘积输出
a	I	INPUT_WIDTH_A	乘数 a
b	I	INPUT_WIDTH_B	乘数 b
cea	I	1	A 口片选，高有效
ceb	I	1	B 口片选，高有效
cepd	I	1	乘积输出片选，高有效
clk	I	1	同步时钟
rstan	I	1	A 口复位，低有效
rstbn	I	1	B 口复位，低有效
rstpdn	I	1	乘积输出复位，低有效

19.3 参数描述

Attribute	Allowed Values	Default	Descriptions
INPUT_WIDTH_A	18	18	A 口输入数据位宽
INPUT_WIDTH_B	18	18	B 口输入数据位宽
OUTPUT_WIDTH	36	36	输出数据位宽

INPUTFORMAT	SIGNED, UNSIGNED	"SIGNED"	乘法器类型
INPUTREGA	ENABLE, DISABLE	"ENABLE"	A 口寄存器使能
INPUTREGB	ENABLE, DISABLE	"ENABLE"	B 口寄存器使能
OUTPUTREG	ENABLE, DISABLE	"ENABLE"	乘积输出寄存器使能
SRMODE	ASYNC, SYNC	"ASYNC"	复位模式
IMPLEMENT	AUTO, DSP, GATE	"AUTO"	乘法器实现方式

19.4 使用方式

使用方式	是否支持
原语例化	是
IP Generator	否

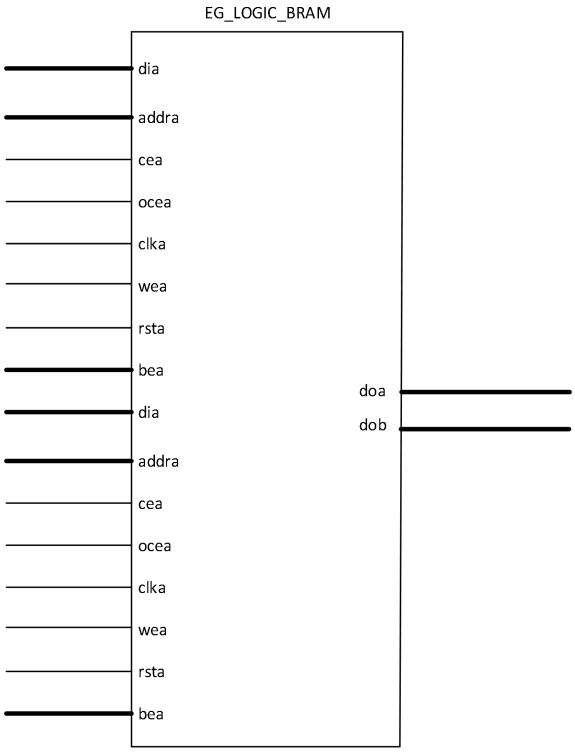
19.5 Verilog 例化模板

```
EG_LOGIC_MULT#(
    .INPUT_WIDTH_A(18),
    .INPUT_WIDTH_B(18),
    .OUTPUT_WIDTH(36),
    .INPUTFORMAT("SIGNED"),
    .INPUTREGA("ENABLE"),
    .INPUTREGB("ENABLE"),
    .OUTPUTREG("ENABLE"),
    .SRMODE("ASYNC"),
    .IMPLEMENT("AUTO"))
MULT_inst(
    .p(p),
    .a(a),
    .b(b),
    .cea(cea),
    .ceb(ceb),
    .cepd(cepd),
```

```
        .clk(clk),
        .rstan(rstan),
        .rstbn(rstbn),
        .rstpdn(rstpdn)
    );
```

19.6 VHDL 例化模板

20 EG_LOGIC_BRAM



20.1 简介

EG_LOGIC_BRAM 是 FPGA 内部 BRAM9K 或者 BRAM32K 搭建的 BRAM 资源。

20.2 端口描述

Name	Direction	Width	Function
dia	I	DATA_WIDTH_A	A 口写入数据
addra	I	ADDR_WIDTH_B	A 口写入地址
cea	I	1	A 口片选，高有效
ocea	I	1	A 口输出使能，在输出有寄存器时才有该信号，高有效
clka	I	1	A 口同步时钟
wea	I	1	A 口写使能
rsta	I	1	A 口复位，高有效
bea	I	BYTE_A	A 口字节使能，高有效
dib	I	DATA_WIDTH_A	B 口写入数据
addrb	I	ADDR_WIDTH_B	B 口写入地址
ceb	I	1	B 口读使能，在输出有寄存器时才有该信号，高有效

oceb	I	1	B 口输出使能
clkb	I	1	B 口同步时钟
web	I	1	B 口写使能，高有效
rstb	I	1	B 口复位，高有效
beb	I	BYTE_B	B 口字节使能，高有效
doa	O	DATA_WIDTH_A	A 口读出数据
dob	O	DATA_WIDTH_B	B 口读出数据

20.3 参数描述

Attribute	Allowed Values	Default	Descriptions
MODE	DP, SP, PDPW, FIFO		BRAM 类型设置
REGMODE_A	"NOREG", "OUTREG"	"NOREG"	A 口输出寄存器模式
REGMODE_B	"NOREG", "OUTREG"	"NOREG"	B 口输出寄存器模式
WRITEMODE_A	"NORMAL", "READBEFOREWRITE", "WRITETHROUGH"	"NORMAL"	A 口写数据模式
WRITEMODE_B	"NORMAL", "READBEFOREWRITE", "WRITETHROUGH"	"NORMAL"	B 口写数据模式
RESETMODE	"SYNC", "ASYNC"	"SYNC"	复位模式
DEBUGGABLE	"YES", "NO"	"NO"	是否使能调试，只有 SP 模式支持
PACKABLE	"YES", "NO"	"NO"	是否支持打包
FORCE_KEEP	"ON", "OFF"	"OFF"	是否强制编绎保留
INIT_FILE	"NONE",” path/file”	"NONE"	添加初始化文件
"NONE"	"NONE" or some binary string like "0101", not wider than the wider port		未初始化空间填充
IMPLEMENT	“9K”, “9K(FAST)” ,” 32K (all capitalized)”	"9K"	实现物理单元选择

20.4 使用方式

使用方式	是否支持
原语例化	是
IP Generator	是

20.5 Verilog 例化模板

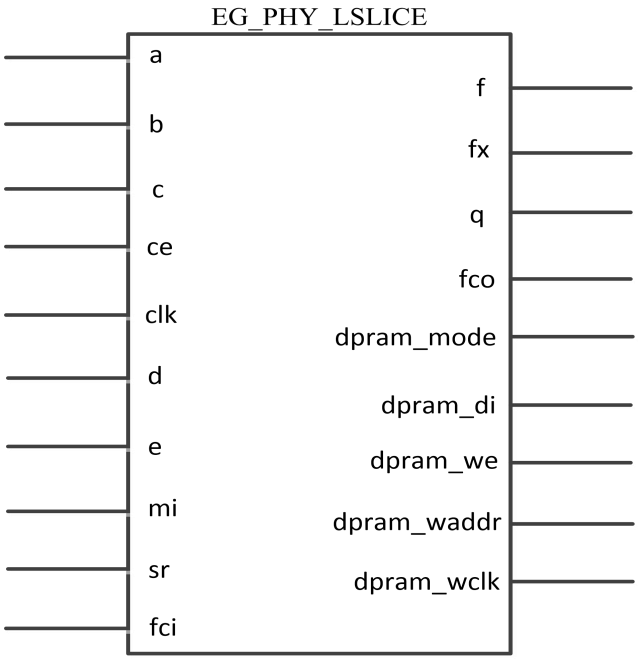
```
EG_PHY_BRAM #(
    . DATA_WIDTH_A(9),
```

```
        .DATA_WIDTH_B(9),
        .ADDR_WIDTH_A(10),
        .ADDR_WIDTH_B(10),
        .DATA_DEPTH_A(1024),
        .DATA_DEPTH_B(1024),
        .BYTE_ENABLE(0),
        .BYTE_A(1),
        .BYTE_B(1),
        .MODE(),
        .REGMODE_A( “DP” ),
        .REGMODE_B("NOREG"),
        .WRITEMODE_A("NORMAL"),
        .WRITEMODE_B("NORMAL"),
        .RESETMODE("SYNC"),
        .DEBUGGABLE"NO" (),
        .PACKABLE("NO"),
        .FORCE_KEEP("OFF"),
        .INIT_FILE("NONE"),
        .FILL_ALL("NONE"),
        .IMPLEMEN("9K")
    )
    PHY_BRAM_inst(
        .dia(dia),
        .addra(addra),
        .cea(cea),
        .ocea(ocea),
        .clka(clka),
```

```
    .wea(wea),
    .rsta(rsta),
    .bea(bea),
    .dib(dib),
    .addrb(addrb),
    .ceb(ceb),
    .oceb(oceb),
    .clkb(clkb),
    .web(web),
    .rstb(rstb),
    .beb(beb),
    .doa(doa),
    .dob(dob));
```

20.6 VHDL 例化模板

21 EG_PHY_LSLICE



21.1 简介

EG_PHY_LSLICE是FPGA内部的LSLICE模块,LSLICE 包含2 个增强型LUT5s 和两个寄存器以及4 级进位链

21.2 端口描述

Name	Direction	Width	Function
a	input	2	组合逻辑输入
b	input	2	组合逻辑输入
c	input	2	组合逻辑输入
d	input	2	组合逻辑输入
e	input	2	组合逻辑输入
mi	input	2	寄存器输入
clk	input	1	时钟输入
ce	input	1	时钟使能输入
sr	input	1	DFF 复位/置位控制
fci	input	1	进位输入
f	output	2	lut 主输出
fx	output	2	lut 辅助输出
q	output	2	lslice 输出

fco	output	1	进位输出
dpram_di	output	4	disram 写数据输出
dpram_waddr	output	4	disram 写地址输出
dpram_wclk	output	1	disram 写时钟输出
dpram_we	output	1	disram 写使能输出
dpram_mode	output	1	disram 模式控制输出

21.3 参数描述

Attribute	Allowed Values	Default	Descriptions
INIT_LUTF0	16'h0000~16'h1111	16'h0000	lut4 初始值
INIT_LUTG0	16'h0000~16'h1111	16'h0000	lut4 初始值
INIT_LUTF1	16'h0000~16'h1111	16'h0000	lut4 初始值
INIT_LUTG1	16'h0000~16'h1111	16'h0000	lut4 初始值
MODE	"LOGIC""RIPPLE""RAMW"	"LOGIC"	模式选择
GSR	"ENABLE""DISABLE"	"ENABLE"	全局 GSR 作为此 slice 用户异步复位，0 使能
TESTMODE	"OFF""ON"	"OFF"	测试模式
CEMUX	"CE""INV""1""0"	"1"	DFF 时钟使能选择
SRMUX	"SR""INV""1" "0"	"SR"	复位选择（复位，取反，1.0）
CLKMUX	"CLK""INV（取反）""1" "0"	"CLK"	时钟选择
SRMODE	"ASYNC""SYNC"	"ASYNC"	复位置位异步/同步模式选择
DFFMODE	"FF" "LATCH"	"FF"	DFF 模式选择
REG0_SD	"MI""F""FX"	"MI"	寄存器 0 的输入选择
REG1_SD	"MI""F""FX"	"MI"	寄存器 1 的输入选择
REG0_REGSET	"RESET""SET"	"SET"	寄存器 0 的 set or reset 选择
REG1_REGSET	"RESET""SET"	"SET"	寄存器 1 的 set or reset 选择
DEMUX0	"D""E"	"D"	LUT-G/D 输入 MUX 配置，"D"选择 D 输入，"E"选择 E 输入
DEMUX1	"D""E"	"D"	LUT-F/D 输入 MUX 配置，"D"选择 D 输入，"E"选择 E 输入
CMIMUX0	"C""MI"	"C"	LUT-G/C 输入前面 MUX 配置，"C"选择 C 输入，"MI"选择 MI 输入
CMIMUX1	"C""MI"	"C"	LUT-F/C 输入 MUX 配置，"C"选择 C 输入，"MI"选择 MI 输入
LSFMUX0	"LUTF""FUNC5""SUM"	"LUTF"	LSFMUX 输出选择，LUT4_F 输出，FUNC5 是 LUT5 输出，SUM 是加法和 sum[0]输出

LSFXMUX0	"LUTG""FUNC6""SUM"	"LUTG"	LSFXMUX 输出选择，LUT4_G 输出，FUNC6 是 LUT6 输出，SUM 是加法和 sum[1]输出
LSFMUX1	"LUTF""FUNC5""SUM"	"LUTF"	LSFMUX 输出选择，LUT4_F 输出，FUNC5 是 LUT5 输出，SUM 是加法和 sum[2]输出
LSFXMUX1	"LUTG""FUNC7""SUM"	"LUTG"	LSFXMUX 输出选择，LUT4_G 输出，FUNC7 是 LUT7 输出，SUM 是加法和 sum[3]输出

21.4 使用方式

使用方式	是否支持
原语例化	是
IP Generator	否

21.5 Verilog 例化示例

1） LSLICE 配置成 LUT5 的功能：

```
EG_PHY_LSLICE
#(
.REG0_SD ("F" ),
.REG1_SD ("F" ),
.CEMUX ("CE" ),
.DEMUX0 ("D" ),
.DEMUX1 ("D" ),
.LSFMUX0 ("FUNC5" ),
.LSFMUX1 ("FUNC5" ),
.INIT_LUTF0 (INIT_LUTF0 ),
.INIT_LUTF1 (INIT_LUTF1 ),
.INIT_LUTG0 (INIT_LUTG0 ),
.INIT_LUTG1 (INIT_LUTG1 )
)
```

```
u_lslice
(
.a ({din[5],din[0]} ),
.b ({din[6],din[1]} ),
.c ({din[7],din[2]} ),
.d ({din[8],din[3]} ),
.e ({din[9],din[4]} ),
.ce (ce ),
.clk (clk ),
.mi (2'h3 ),
.sr (sr ),
.fci ( ),
.f ( ),
.fx ( ),
.q (dout ),
.fco ( ),
.dpram_mode ( ),
.dpram_di ( ),
.dpram_we ( ),
.dpram_waddr ( ),
.dpram_wclk ( )
);
```

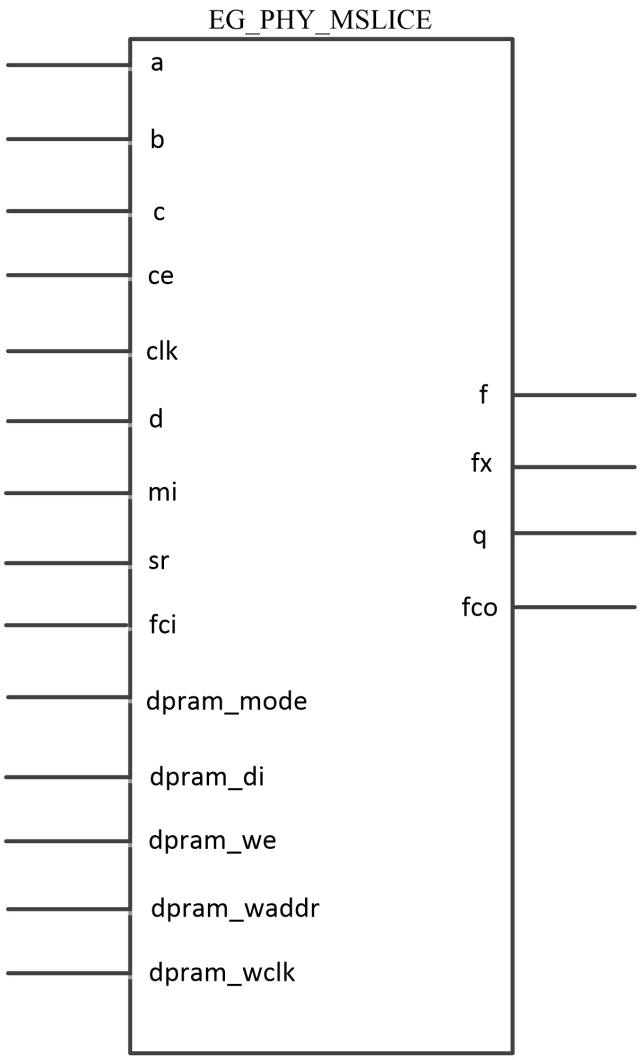
2) LSLICE 配置成 LUT6 的功能：

```
EG_PHY_LSLICE
#(
.DEMUX0 ("D" ),
.DEMUX1 ("D" ),
```

```
    .LSFMUX0 ("FUNC5" ),
    .LSFMUX1 ("FUNC5" ),
    .LSFXMUX0 ("FUNC6" ),
    .INIT_LUTF0 (INIT_LUTF0 ),
    .INIT_LUTG0 (INIT_LUTG0 ),
    .INIT_LUTF1 (INIT_LUTF1 ),
    .INIT_LUTG1 (INIT_LUTG1 )
    )
    u_lslice
    (
    .a ({din [0], din [0]}),
    .b ({din [1], din [1]}),
    .c ({din [2], din [2]}),
    .d ({din [3], din [3]}),
    .e ({din [4], din [4]}),
    .ce (1'b1 ),
    .mi ({1'b1, din [5]}),
    .sr (1'b0 ),
    .fci ( ),
    .f ( ),
    .fx (dout_6 ),
    .q ( ),
    .fco ( ),
    .dpram_mode ( ),
    .dpram_di ( ),
    .dpram_we ( ),
    .dpram_waddr ( ),
```

```
    .dpram_wclk ( )

    );
```

22 EG_PHY_MSLICE



22.1 简介

EG_PHY_MSLICE 是 FPGA 内部的 MSLICE 模块,MSLICE 包含 2 个 LUT4s 和两个寄存器以及 2 级进位链。

Name	Direction	Width	Function
a	input	2	组合逻辑输入
b	input	2	组合逻辑输入
c	input	2	组合逻辑输入
d	input	2	组合逻辑输入
mi	input	2	寄存器输入
clk	input	1	时钟输入
ce	input	1	时钟使能输入

sr	input	1	DFF 复位/置位控制
fci	input	1	进位输入
f	output	2	lut 主输出
fx	output	2	lut 辅助输出
q	output	2	mslice 输出
fco	output	1	进位输出
dpram_di	input	2	disram 写数据输入
dpram_waddr	input	4	disram 写地址输入
dpram_wclk	input	1	disram 写时钟输入
dpram_we	input	1	disram 写使能输入
dpram_mode	input	1	disram 模式控制输入

22.2 参数描述

Attribute	Allowed Values	Default	Descriptions
INIT_LUT0	16'h0000~16'h1111	16'h0000	lut4 初始值
INIT_LUT1	16'h0000~16'h1111	16'h0000	lut4 初始值
MODE	"LOGIC""RIPPLE""DPRAM"	"LOGIC"	模式选择
ALUTYPE	""	""	无
TESTMODE	"OFF""ON"	"OFF"	测试模式
MSFXMUX	"OFF""ON"	"OFF"	FX 输出是否使用
GSR	"ENABLE""DISABLE"	"ENABLE"	全局 GSR 作为此 slice 用户异步复位，0 使能
CEMUX	"CE""INV""1""0"	"CE"	DFF 时钟使能选择
SRMUX	"SR""INV""1" "0"	"SR"	复位选择（复位，取反，1.0）
CLKMUX	"CLK""INV（取反）""1" "0"	"CLK"	时钟选择
SRMODE	"ASYNC""SYNC"	"ASYNC"	复位置位异步/同步模式选择
DFFMODE	"FF" "LATCH"	"FF"	DFF 模式选择
REG0_SD	"MI""F""FX"	"MI"	寄存器 0 的输入选择
REG1_SD	"MI""F""FX"	"MI"	寄存器 1 的输入选择
REG0_REGSET	"RESET""SET"	"SET"	寄存器 0 的 set or reset 选择
REG1_REGSET	"RESET""SET"	"SET"	寄存器 1 的 set or reset 选择

22.3 使用方式

使用方式	是否支持
原语例化	是
IP Generator	否

22.4 Verilog 例化示例

MSLICE 配置成 LUT4 的功能：

```
EG_PHY_MSLICE
#(
.REG0_SD ("F" ),
.REG1_SD ("F" ),
.INIT_LUT0 (INIT_LUT0 ),
.INIT_LUT1 (INIT_LUT1 )
)
u_mslice
(
.a ({2{din[0]}} ),
.b ({2{din[1]}} ),
.c ({2{din[2]}} ),
.ce (ce ),
.clk (clk ),
.d ({2{din[3]}} ),
.mi ( ),
.sr (sr ),
.fci ( ),
.f ( ),
.fx ( ),
.q (dout ),
.fco ( ),
.dpram_mode ( ),
.dpram_di ( ),
```

```
    .dpram_we ( ),
    .dpram_waddr ( ),
    .dpram_wclk ( )
    );
```

版本信息

日期	版本	修订记录
2018/8/2	1.0	初版建立
2018/9/14	1.1	文档格式调整
2019/4/12	1.2	更正 EG_LOGIC_BUFIO 示意图中管脚信息 调整文档格式
2020/2/18	1.3	增加 SLICE 部分原语说明
2022/10/18	1.4	1. 更新 EG_LOGIC_RAMFIFO 模块，删除 clkr,clkw 端口，修正为 clk,增加 IMPLEMENT 参数描述，修正实例中的错误描述，更新模块图 2. 更新 EG_LOGIC_ODDRx2l：更新示例中端口错误 psclk 修改为 sclk； 3. 更新 EG_LOGIC_DRAM:更新参数介绍章节 4. 删除 EG_LOGIC_ CLOCKED_DIV 模块介绍 5. 更新 11.2 EG_LOGIC_ODDR 端口描述中的 rst 方向 6. 更新文档免责声明

免责声明

本文档并未授予任何知识产权的许可，并未以明示或暗示，或以禁止发言或其他方式授予任何知识产权许可；本文档仅为向用户提供使用器件的参考，协助用户正确地使用安路科技产品之用，其著作权归安路科技所有；本文档所展示的任何产品信息均不构成安路科技对所涉产品或服务作出任何明示或默示的声明或保证。

安路科技将不定期地对本文档进行更新、修订。用户如需获取最新版本的文档，可通过安路科技的官方网站（网址为：https://www.anlogic.com）自行查询下载，也可联系安路科技的销售人员咨询获取。