Arora V 可配置功能单元(CFU)

用户指南

UG303-1.0,2023-04-20

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。高云半导体对文档中包含的文字、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文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文档进行适时的更新。

版本信息

日期	版本	说明
2023/04/20	1.0	初始版本。

目录

图目录

表目录

1 关于本手册

1.1 手册内容

Arora V 可配置功能单元(CFU)手册主要描述了 Arora V 产品可配置功能单元的结构、工作模式和原语。

1.2 相关文档

通过登录高云半导体网站 www.gowinsemi.com.cn 可以下载、查看以下相关文档：

- DS981, GW5AT 系列 FPGA 产品数据手册
- DS1103, GW5A 系列 FPGA 产品数据手册
- DS1104, GW5AST 系列 FPGA 产品数据手册
- SUG100，Gowin 云源软件用户指南
- UG300，Arora V 存储器(BSRAM & SSRAM)用户指南

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
ALU	Arithmetic Logic Unit	算术逻辑单元
BSRAM	Block Static Random Access Memory	块状静态随机存储器
CFU	Configurable Function Unit	可配置功能单元
CLS	Configurable Logic Section	可配置逻辑块
CRU	Configurable Routing Unit	可配置布线单元
DFF	D Flip Flop	D 触发器
DL	Data Latch	数据锁存器
LUT	Look-up Table	查找表
MUX2	Multiplexer 2:1	2 选 1 选择器
REG	Register	寄存器

术语、缩略语	全称	含义
SSRAM	Shadow Static Random Access Memory	分布式静态随机存储器

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

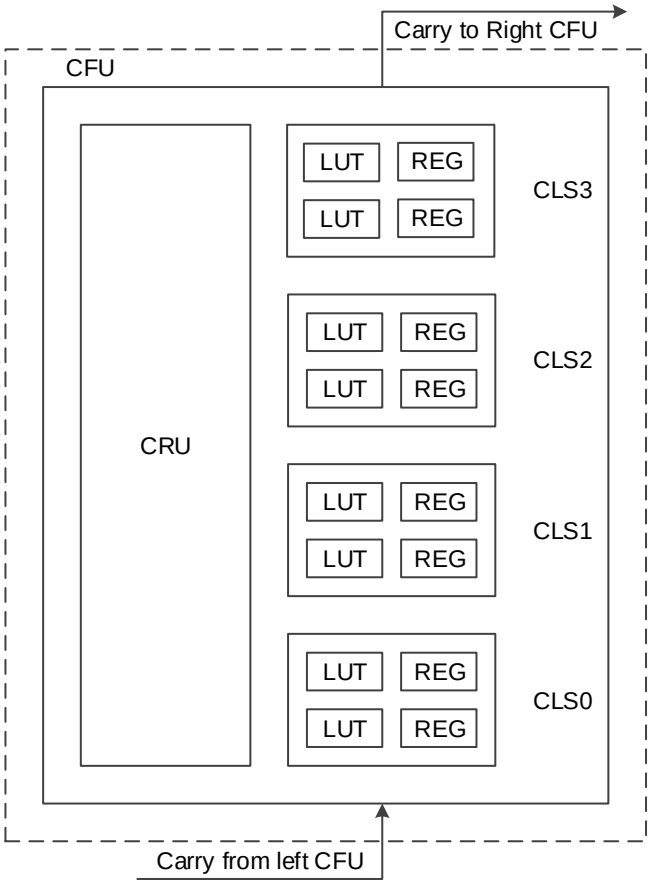
E-mail：support@gowinsemi.com

Tel：+86 755 8262 0391

2 CFU 结构

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，每个基本单元可由四个可配置逻辑块（CLS）以及相应的可配置布线单元（CRU）组成，其中四个可配置逻辑块各包含两个四输入查找表（LUT）和两个寄存器（REG），如图 2-1 所示。CFU 中的可配置逻辑块可根据应用场景配置成基本查找表、算术逻辑单元、静态随机存储器和只读存储器四种工作模式。

图 2-1 可配置功能单元结构示意图



注！

GW5AT 器件支持 CLS3 的 REG，且 CLS3/CLS2 的 CLK/CE/SR 信号同源。

2.1 CLS

2.1.1 CLS 模式

CLS 支持基本查找表、算术逻辑和存储器模式：

- 基本查找表模式

每个查找表可以被配置为一个 4 输入查找表（LUT4），可配置逻辑块可实现高阶查找表功能：

  - 一个可配置逻辑块可配置成一个 5 输入查找表（LUT5）。
  - 两个可配置逻辑块可配置成一个 6 输入查找表（LUT6）。
  - 四个可配置逻辑块可配置成一个 7 输入查找表（LUT7）。
  - 八个可配置逻辑块可配置成一个 8 输入查找表（LUT8）。

- 算术逻辑模式

结合进位链，查找表可配置成算术逻辑模式（ALU），用作实现以下功能：

  - 加法/减法运算
  - 计数器，包括加计数器和减计数器。
  - 比较器，包括大于比较、小于比较和不相等比较。
  - 乘法器

- 存储器模式

在此模式下，一个可配置功能单元可构成 16 x 4 位的静态随机存储器（SRAM）或只读存储器（ROM16）。

2.1.2 REG

可配置逻辑块（CLS0~CLS3）各含两个寄存器（REG），如图 2-2 所示。

图 2-2 CFU 中的寄存器示意图

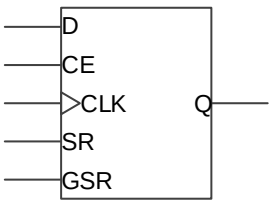


表 2-1 CFU 中寄存器模块信号说明

信号名	I/O	描述
D	I	寄存器数据输入[1]
CE	I	CLK 使能信号，可配置为高电平使能或低电平使能[2]。
CLK	I	时钟信号，可配置为上升沿触发或下降沿触发[2]。
SR	I	本地置复位输入，可配置为如下功能[2]： ● 同步复位 ● 同步置位

信号名	I/O	描述
		● 异步复位 ● 异步置位 ● 无本地置复位
GSR[3],[4]	I	全局复置位，可配置为如下功能[4]： ● 异步复位 ● 异步置位 ● 无全局复置位
Q	O	寄存器输出

注！

- [1]信号 D 的来源可以选择同一可配置逻辑块中任一查找表的输出，也可以选择来自于 CRU 的输入。因此在查找表被占用的情况下，寄存器仍可以单独使用。
- [2]CFU 中除 CLS2/CLS3 共线外，可配置逻辑块的 CE/CLK/SR 均可独立配置选择。
- [3]在高云半导体 FPGA 产品内部，GSR 通过直连线连接，不通过 CRU。
- [4]SR 与 GSR 同时有效时 GSR 有较高的优先级。

2.2 CRU

布线资源单元 CRU 的功能主要包括两个方面：

- 输入选择功能：为 CFU 的输入信号提供输入源选择。
- 布线资源功能：为 CFU 的输入/输出信号提供连接关系，包括 CFU 内部连接、CFU 之间连接以及 CFU 和 FPGA 内部其他功能模块之间的连接。

3 CFU 原语

3.1 LUT

输入查找表 LUT，常用的 LUT 结构有 LUT1、LUT2、LUT3、LUT4，其区别在于查找表输入位宽的不同。

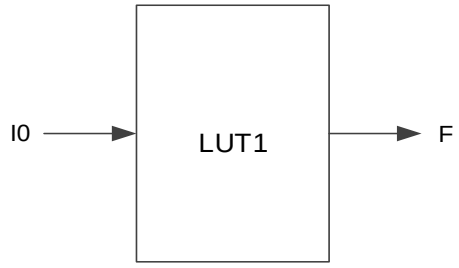
3.1.1 LUT1

原语介绍

LUT1（1-input Look-up Table）是其中最简单的一种，常用于实现缓冲器和反相器。LUT1 为 1 输入的查找表，通过 parameter 给 INIT 赋初值后，根据输入的地址查找对应的数据并输出结果。

端口示意图

图 3-1 LUT1 端口示意图



端口介绍

表 3-1 LUT1 端口介绍

端口	I/O	描述
I0	Input	数据输入信号
F	Output	数据输出信号

参数介绍

表 3-2 LUT1 参数介绍

参数	范围	默认	描述
INIT	2'h0~2'h3	2'h0	LUT1 初始值

真值表

表 3-3 LUT1 真值表

Input(I0)	Output(F)
0	INIT[0]
1	INIT[1]

原语例化

Verilog 例化：

```
LUT1 instName (
        .I0(I0),
        .F(F)
);
defparam instName.INIT=2'h1;
```

Vhdl 例化：

```
COMPONENT LUT1
        GENERIC (INIT:bit_vector:=X"0");
        PORT(
            F:OUT std_logic;
            I0:IN std_logic
        );
END COMPONENT;
uut:LUT1
        GENERIC MAP(INIT=>X"0")
        PORT MAP (
            F=>F,
            I0=>I0
        );
```

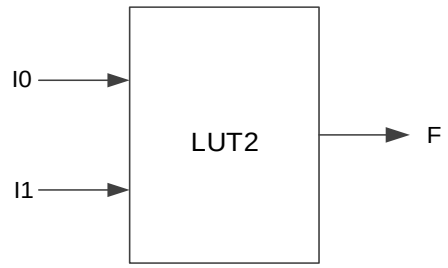
3.1.2 LUT2

原语介绍

LUT2（2-input Look-up Table）为 2 输入的查找表，通过 parameter 给 INIT 赋初值后，根据输入的地址查找对应的数据并输出结果。

端口示意图

图 3-2 LUT2 端口示意图



端口介绍

表 3-4 LUT2 端口介绍

端口	I/O	描述
I0	Input	数据输入信号
I1	Input	数据输入信号
F	Output	数据输出信号

参数介绍

表 3-5 LUT2 参数介绍

参数	范围	默认	描述
INIT	4'h0~4'hf	4'h0	LUT2 初始值

真值表

表 3-6 LUT2 真值表

Input(I1)	Input(I0)	Output(F)
0	0	INIT[0]
0	1	INIT[1]
1	0	INIT[2]
1	1	INIT[3]

原语例化

Verilog 例化：

```
LUT2 instName (
```

```
        .I0(I0),
        .I1(I1),
        .F(F)
    );
    defparam instName.INIT=4'h1;
```

Vhdl 例化：

```
    COMPONENT LUT2
        GENERIC (INIT:bit_vector:=X"0");
        PORT(
            F:OUT std_logic;
            I0:IN std_logic;
            I1:IN std_logic
        );
    END COMPONENT;
    uut:LUT2
        GENERIC MAP(INIT=>X"0")
        PORT MAP (
            F=>F,
            I0=>I0,
            I1=>I1
        );
```

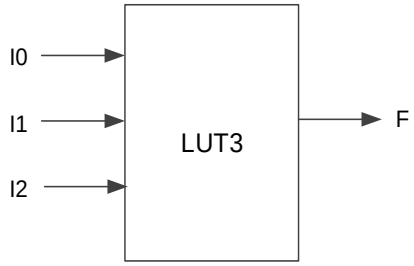
3.1.3 LUT3

原语介绍

LUT3（3-input Look-up Table）为 3 输入的查找表，通过 parameter 给 INIT 赋初值后，根据输入的地址查找对应的数据并输出结果。

端口示意图

图 3-3 LUT3 端口示意图



端口介绍

表 3-7 LUT3 端口介绍

端口	I/O	描述
I0	Input	数据输入信号
I1	Input	数据输入信号
I2	Input	数据输入信号
F	Output	数据输出信号

参数介绍

表 3-8 LUT3 参数介绍

参数	范围	默认	描述
INIT	8'h00~8'hff	8'h00	LUT3 初始值

真值表

表 3-9 LUT3 真值表

Input(I2)	Input(I1)	Input(I0)	Output(F)
0	0	0	INIT[0]
0	0	1	INIT[1]
0	1	0	INIT[2]
0	1	1	INIT[3]
1	0	0	INIT[4]
1	0	1	INIT[5]
1	1	0	INIT[6]
1	1	1	INIT[7]

原语例化

Verilog 例化：

```
LUT3 instName (
    .I0(I0),
    .I1(I1),
    .I2(I2),
    .F(F)
);
defparam instName.INIT=8'h10;
```

Vhdl 例化

```
COMPONENT LUT3
```

```
        GENERIC (INIT:bit_vector:=X"00");
        PORT(
            F:OUT std_logic;
            I0:IN std_logic;
            I1:IN std_logic;
            I2:IN std_logic
        );
END COMPONENT;
uut:LUT3
    GENERIC MAP(INIT=>X"00")
    PORT MAP (
        F=>F,
        I0=>I0,
        I1=>I1,
        I2=>I2
     );
```

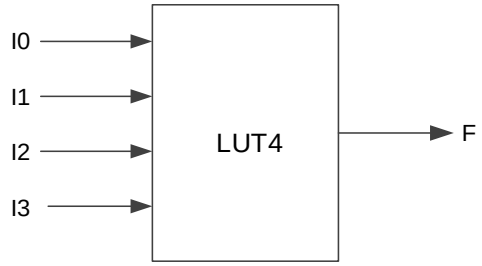
3.1.4 LUT4

原语介绍

LUT4（4-input Look-up Table）为 4 输入的查找表，通过 parameter 给 INIT 赋初值后，根据输入的地址查找对应的数据并输出结果。

端口示意图

图 3-4 LUT4 端口示意图



端口介绍

表 3-10 LUT4 端口介绍

端口	I/O	描述
I0	Input	数据输入信号
I1	Input	数据输入信号
I2	Input	数据输入信号
I3	Input	数据输入信号

端口	I/O	描述
F	Output	数据输出信号

参数介绍

表 3-11 LUT4 参数介绍

参数	范围	默认	描述
INIT	16'h0000~16'hffff	16'h0000	LUT4 初始值

真值表

表 3-12 LUT4 真值表

Input(I3)	Input(I2)	Input(I1)	Input(I0)	Output(F)
0	0	0	0	INIT[0]
0	0	0	1	INIT[1]
0	0	1	0	INIT[2]
0	0	1	1	INIT[3]
0	1	0	0	INIT[4]
0	1	0	1	INIT[5]
0	1	1	0	INIT[6]
0	1	1	1	INIT[7]
1	0	0	0	INIT[8]
1	0	0	1	INIT[9]
1	0	1	0	INIT[10]
1	0	1	1	INIT[11]
1	1	0	0	INIT[12]
1	1	0	1	INIT[13]
1	1	1	0	INIT[14]
1	1	1	1	INIT[15]

原语例化

Verilog 例化：

```
LUT4 instName (
    .I0(I0),
    .I1(I1),
    .I2(I2),
    .I3(I3),
    .F(F)
```

```
);
defparam instName.INIT=16'h1011;
```

Vhdl 例化：

```
COMPONENT LUT4
        GENERIC (INIT:bit_vector:=X"0000");
        PORT(
            F:OUT std_logic;
            I0:IN std_logic;
            I1:IN std_logic;
            I2:IN std_logic;
            I3:IN std_logic
        );
END COMPONENT;
uut:LUT4
        GENERIC MAP(INIT=>X"0000")
        PORT MAP (
            F=>F,
            I0=>I0,
            I1=>I1,
            I2=>I2,
            I3=>I3
        );
```

3.1.5 Wide LUT

原语介绍

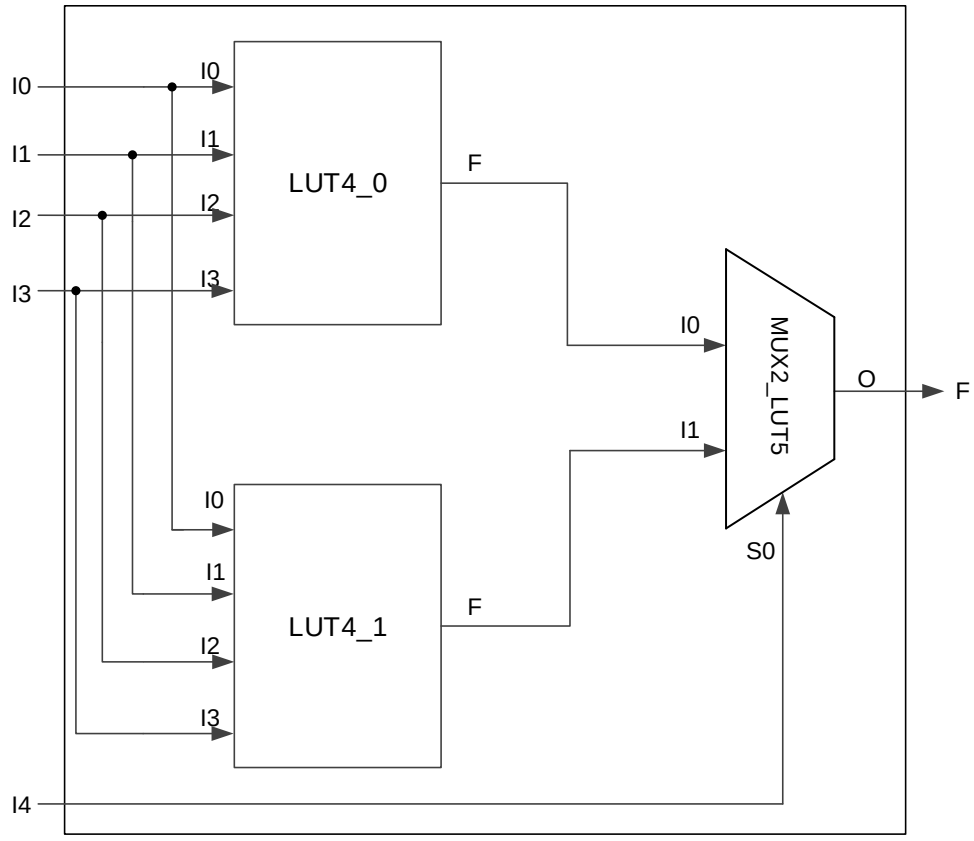
Wide LUT 是通过 LUT4 和 MUX2 构造高阶 LUT，高云 FPGA 目前支持的构造高阶 LUT 的 MUX2 有 MUX2_LUT5/MUX2_LUT6/MUX2_LUT7/MUX2_LUT8。

高阶 LUT 的构造方式如下：两个 LUT4 和 MUX2_LUT5 可组合实现 LUT5，两个组合实现的 LUT5 和 MUX2_LUT6 可组合实现 LUT6，两个组合实现的 LUT6 和 MUX2_LUT7 可组合实现 LUT7，两个组合实现的 LUT7 和 MUX2_LUT8 可组合实现 LUT8。

以 LUT5 为例介绍 Wide LUT 的使用。

端口示意图

图 3-5 LUT5 端口示意图



端口介绍

表 3-13 LUT5 端口介绍

端口名	I/O	描述
I0	Input	数据输入信号
I1	Input	数据输入信号
I2	Input	数据输入信号
I3	Input	数据输入信号
I4	Input	数据输入信号
F	Output	数据输出信号

参数介绍

表 3-14 LUT5 参数介绍

参数	范围	默认	描述
INIT	32'h00000~32'hfffff	32'h00000	LUT5 初始值

真值表

表 3-15 LUT5 真值表

Input(I4)	Input(I3)	Input(I2)	Input(I1)	Input(I0)	Output(F)
0	0	0	0	0	INIT[0]
0	0	0	0	1	INIT[1]
0	0	0	1	0	INIT[2]
0	0	0	1	1	INIT[3]
0	0	1	0	0	INIT[4]
0	0	1	0	1	INIT[5]
0	0	1	1	0	INIT[6]
0	0	1	1	1	INIT[7]
0	1	0	0	0	INIT[8]
0	1	0	0	1	INIT[9]
0	1	0	1	0	INIT[10]
0	1	0	1	1	INIT[11]
0	1	1	0	0	INIT[12]
0	1	1	0	1	INIT[13]
0	1	1	1	0	INIT[14]
0	1	1	1	1	INIT[15]
1	0	0	0	0	INIT[16]
1	0	0	0	1	INIT[17]
1	0	0	1	0	INIT[18]
1	0	0	1	1	INIT[19]
1	0	1	0	0	INIT[20]
1	0	1	0	1	INIT[21]
1	0	1	1	0	INIT[22]
1	0	1	1	1	INIT[23]
1	1	0	0	0	INIT[24]
1	1	0	0	1	INIT[25]
1	1	0	1	0	INIT[26]
1	1	0	1	1	INIT[27]
1	1	1	0	0	INIT[28]
1	1	1	0	1	INIT[29]
1	1	1	1	0	INIT[30]
1	1	1	1	1	INIT[31]

原语例化

Verilog 例化：

```
LUT5 instName (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .I4(i4),
    .F(f0)
);
defparam instName.INIT=32'h00000000;
```

Vhdl 例化：

```
COMPONENT LUT5
        PORT(
            F:OUT std_logic;
            I0:IN std_logic;
            I1:IN std_logic;
            I2:IN std_logic;
            I3:IN std_logic;
            I4:IN std_logic
        );
END COMPONENT;
uut:LUT5
        GENERIC MAP(INIT=>X"00000000")
        PORT MAP (
            F=>f0,
            I0=>i0,
            I1=>i1,
            I2=>i2,
            I3=>i3,
            I4=>i4
        );
```

3.2 MUX

MUX 是多路复用器，拥有多路输入，通过通道选择信号确定其中一路数据传送到输出端。高云原语中有 2 选 1 和 4 选 1 等多路复用器。

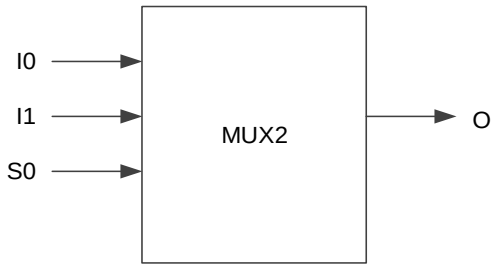
3.2.1 MUX2

原语介绍

MUX2（2-to-1 Multiplexer）是 2 选 1 的复用器，根据选择信号，从两个输入中选择其中一个作为输出。

端口示意图

图 3-6 MUX2 端口示意图



端口介绍

表 3-16 MUX2 端口介绍

端口	I/O	描述
I0	Input	数据输入信号
I1	Input	数据输入信号
S0	Input	数据选择信号
O	Output	数据输出信号

真值表

表 3-17 MUX2 真值表

Input(S0)	Output(O)
0	I0
1	I1

原语例化

Verilog 例化：

```
MUX2 instName (
    .I0(I0),
    .I1(I1),
    .S0(S0),
```

```
        .O(O)
    );
```

Vhdl 例化：

```
    COMPONENT MUX2
        PORT(
            O:OUT std_logic;
            I0:IN std_logic;
                I1:IN std_logic;
                S0:IN std_logic
        );
    END COMPONENT;
    uut:MUX2
        PORT MAP (
            O=>O,
            I0=>I0,
            I1=>I1,
            S0=>S0
        );
```

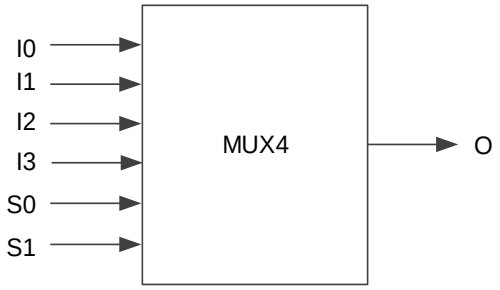
3.2.2 MUX4

原语介绍

MUX4（4-to-1 Multiplexer）是 4 选 1 的多路复用器，根据选择信号，从四个输入中选择其中一个作为输出。

端口示意图

图 3-7 MUX4 端口示意图



端口介绍

表 3-18 MUX4 端口介绍

端口	I/O	描述
I0	Input	数据输入信号

端口	I/O	描述
I1	Input	数据输入信号
I2	Input	数据输入信号
I3	Input	数据输入信号
S0	Input	数据选择信号
S1	Input	数据选择信号
O	Output	数据输出信号

真值表

表 3-19 MUX4 真值表

Input(S1)	Input(S0)	Output(O)
0	0	I0
0	1	I1
1	0	I2
1	1	I3

原语例化

Verilog 例化：

```
MUX4 instName (
    .I0(I0),
    .I1(I1),
    .I2(I2),
    .I3(I3),
    .S0(S0),
    .S1(S1),
    .O(O)
);
```

Vhdl 例化：

```
COMPONENT MUX4
    PORT(
        O:OUT std_logic;
        I0:IN std_logic;
        I1:IN std_logic;
        I2:IN std_logic;
        I3:IN std_logic;
```

```
                S0:IN std_logic;
                S1:IN std_logic
        );
END COMPONENT;
uut:MUX4
    PORT MAP (
        O=>O,
        I0=>I0,
        I1=>I1,
        I2=>I2,
        I3=>I3,
        S0=>S0,
        S1=>S1
    );
```

3.2.3 Wide MUX

原语介绍

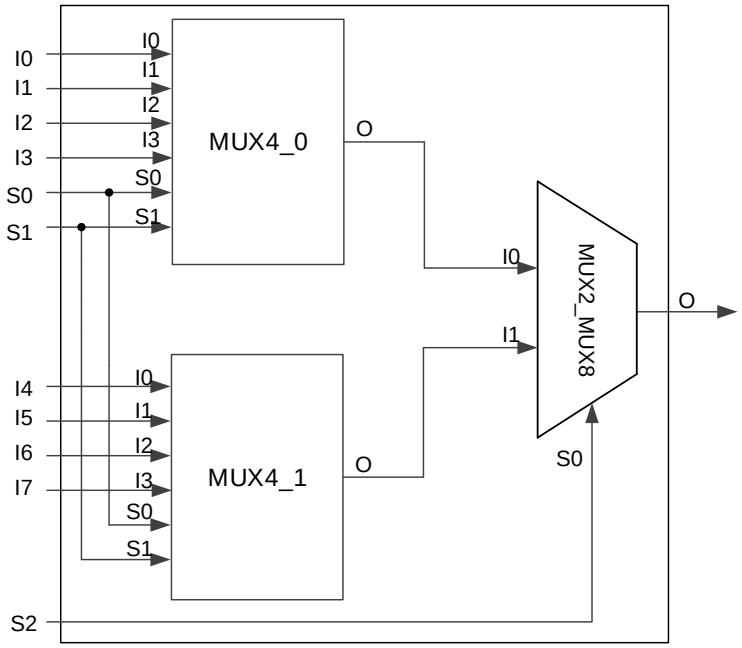
Wide MUX 是通过 MUX4 和 MUX2 构造高阶 MUX，高云 FPGA 目前支持的构造高阶 MUX 的 MUX2 有 MUX2_MUX8/ MUX2_MUX16/ MUX2_MUX32。

高阶 MUX 的构造方式如下：两个 MUX4 和 MUX2_MUX8 可组合实现 MUX8，两个组合实现的 MUX8 和 MUX2_MUX16 可组合实现 MUX16，两个组合实现的 MUX16 和 MUX2_MUX32 可组合实现 MUX32。

以 MUX8 为例介绍 Wide MUX 的使用。

端口示意图

图 3-8 MUX8 端口示意图



端口介绍

表 3-20 MUX8 端口介绍

端口	输入/输出	描述
I0	Input	数据输入信号
I1	Input	数据输入信号
I2	Input	数据输入信号
I3	Input	数据输入信号
I4	Input	数据输入信号
I5	Input	数据输入信号
I6	Input	数据输入信号
I7	Input	数据输入信号
S0	Input	数据选择信号
S1	Input	数据选择信号
S2	Input	数据选择信号
O	Output	数据输出信号

真值表

表 3-21 MUX8 真值表

Input(S2)	Input(S1)	Input(S0)	Output(O)
0	0	0	I0
0	0	1	I1

Input(S2)	Input(S1)	Input(S0)	Output(O)
0	1	0	I2
0	1	1	I3
1	0	0	I4
1	0	1	I5
1	1	0	I6
1	1	1	I7

原语例化

Verilog 例化：

```
MUX8 instName (
    .I0(i0),
    .I1(i1),
    .I2(i2),
    .I3(i3),
    .I4(i4),
    .I5(i5),
    .I6(i6),
    .I7(i7),
    .S0(s0),
    .S1(s1),
    .S2(s2),
    .O(o0)
);
```

Vhdl 例化：

```
COMPONENT MUX8
    PORT(
        O:OUT std_logic;
        I0:IN std_logic;
        I1:IN std_logic;
        I2:IN std_logic;
        I3:IN std_logic;
        I4:IN std_logic;
        I5:IN std_logic;
        I6:IN std_logic;
```

```
                I7:IN std_logic;
                S0:IN std_logic;
                S1:IN std_logic;
                S2:IN std_logic
        );
END COMPONENT;
uut:MUX8
        PORT MAP (
            O=>o0,
            I0=>I0,
            I1=>I1,
            I2=>I2,
            I3=>I3,
            I4=>I4,
            I5=>I5,
            I6=>I6,
            I7=>I7,
            S0=>S0,
            S1=>S1,
            S2=>S2
         );
```

3.3 ALU

原语介绍

ALU（2-input Arithmetic Logic Unit）2 输入算术逻辑单元，实现了 ADD/SUB/ADDSUB 等功能，具体功能如表 3-22 所示。

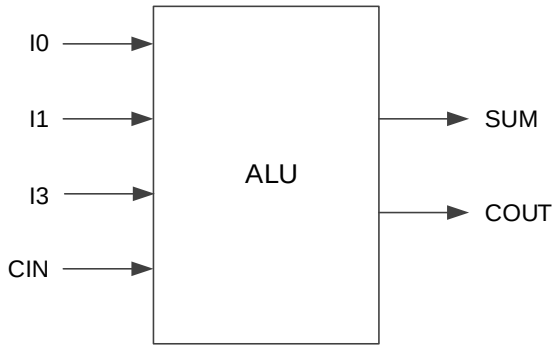
表 3-22 ALU 功能

项目	描述
ADD	加法运算
SUB	减法运算
ADDSUB	加/减法运算，由 I3 选择： ● 1：加法； ● 0：减法
CUP	加计数器
CDN	减计数器
CUPCDN	加/减计数器，由 I3 选择： ● 1：加计数器；

项目	描述
	● 0；减计数器。
GE	大于等于比较器
NE	不等于比较器
LE	小于等于比较器

端口示意图

图 3-9 ALU 端口示意图



注！

GW5AT 器件的 CIN 除了来自前一个 ALU 的 COUT，也可来自逻辑或常量。

端口介绍

表 3-23 ALU 端口介绍

端口	Input/Output	描述
I0	Input	数据输入信号
I1	Input	数据输入信号
I3	Input	数据选择信号，用于ADDSUB加减选择或CUPCDN的加减计数器选择。
CIN	Input	数据进位输入信号
COUT	Output	数据进位输出信号
SUM	Output	数据输出信号

参数介绍

表 3-24 ALU 参数介绍

参数	范围	默认	描述
ALU_MODE	0,1,2,3,4,5,6,7,8	0	Select the function of arithmetic. ● 0: ADD ● 1: SUB

参数	范围	默认	描述
			● 2: ADDSUB ● 3: NE ● 4: GE ● 5: LE ● 6: CUP ● 7: CDN ● 8: CUPCDN

原语例化

Verilog 例化：

```
ALU instName (
    .I0(I0),
    .I1(I1),
    .I3(I3),
    .CIN(CIN),
    .COUT(COUT),
    .SUM(SUM)
);
defparam instName.ALU_MODE=1;
```

Vhdl 例化：

```
COMPONENT ALU
    GENERIC (ALU_MODE:integer:=0);
        PORT(
            COUT:OUT std_logic;
                SUM:OUT std_logic;
            I0:IN std_logic;
                I1:IN std_logic;
                I3:IN std_logic;
                CIN:IN std_logic
        );
END COMPONENT;
uut:ALU
    GENERIC MAP(ALU_MODE=>1)
      PORT MAP (
          COUT=>COUT,
          SUM=>SUM,
```

```
        I0=>I0,
        I1=>I1,
        I3=>I3,
        CIN=>CIN
    );
```

3.4 FF

触发器是时序电路中常用的基本元件，FPGA 内部的时序逻辑都可通过 FF 结构实现，常用的 FF 有 DFFSE、DFFRE、DFFPE、DFFCE，其区别在于复位方式等方面。

与 FF 相关的原语有 4 个，如表 3-25 所示。

表 3-25 与 FF 相关的原语

原语	描述
DFFSE	带时钟使能、同步置位 D 触发器
DFFRE	带时钟使能、同步复位 D 触发器
DFFPE	带时钟使能、异步置位 D 触发器
DFFCE	带时钟使能、异步清零 D 触发器

放置规则

表 3-26 FF 类型

编号	类型 1	类型 2
1	DFFSE	DFFRE
2	DFFPE	DFFCE

- 相同类型的 DFF，可以放置在同一个 CLS 的 2 个 FF 上，除数据输入 pin 外的其它输入必须共线；
- 不同类型的 DFF，表 3-26 中同一编号的两种类型可以放置在同一个 CLS 的 2 个 FF 上，除数据输入 pin 外的其它输入必须共线；
- 可以约束 DFF 和 ALU 在同一个 CLS 的相同或不同位置；
- 可以约束 DFF 和 LUT 在同一个 CLS 的相同或不同位置。

注！

共线是指必须是同一条 net，经过反相器前后的两条 net 为不共线，不可放置在同一个 CLS。

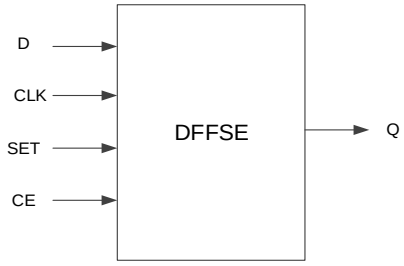
3.4.1 DFFSE

原语介绍

DFFSE（D Flip-Flop with Clock Enable and Synchronous Set）是上升沿触发的 D 触发器，具有同步置位和时钟使能功能。

端口示意图

图 3-10 DFFSE 端口示意图



端口介绍

表 3-27 DFFSE 端口介绍

端口	I/O	描述
D	Input	数据输入信号
CLK	Input	时钟输入信号
SET	Input	同步置位信号，高电平有效。
CE	Input	时钟使能信号
Q	Output	数据输出信号

参数介绍

表 3-28 DFFSE 参数介绍

参数	范围	默认	描述
INIT	1'b1	1'b1	DFFSE 初始值

原语例化

Verilog 例化：

```
DFFSE instName (
        .D(D),
        .CLK(CLK),
        .SET(SET),
        .CE(CE),
        .Q(Q)
```

```
);
defparam instName.INIT=1'b1;
```

Vhdl 例化：

```
COMPONENT DFFSE
        GENERIC (INIT:bit:='1');
        PORT(
            Q:OUT std_logic;
            D:IN std_logic;
                CLK:IN std_logic;
                SET:IN std_logic;
                CE:IN std_logic
        );
END COMPONENT;
uut:DFFSE
        GENERIC MAP(INIT=>'1')
        PORT MAP (
            Q=>Q,
            D=>D,
            CLK=>CLK,
            SET=>SET,
            CE=>CE
        );
```

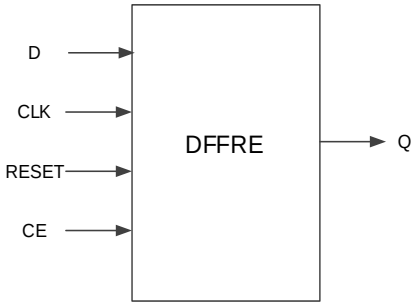
3.4.2 DFFRE

原语介绍

DFFRE（D Flip-Flop with Clock Enable and Synchronous Reset）是上升沿触发的 D 触发器，具有同步复位和时钟使能功能。

端口示意图

图 3-11 DFFRE 端口示意图



端口介绍

表 3-29 DFFRE 端口介绍

端口	I/O	描述
D	Input	数据输入信号
CLK	Input	时钟输入信号
RESET	Input	同步复位信号，高电平有效。
CE	Input	时钟使能信号
Q	Output	数据输出信号

参数介绍

表 3-30 DFFRE 参数介绍

参数	范围	默认	描述
INIT	1'b0	1'b0	DFFRE 初始值

原语例化

Verilog 例化：

```
DFFRE instName (
    .D(D),
    .CLK(CLK),
    .RESET(RESET),
    .CE(CE),
    .Q(Q)
);
defparam instName.INIT=1'b0;
```

Vhdl 例化：

```
COMPONENT DFFRE
        GENERIC (INIT:bit:='0');
        PORT(
            Q:OUT std_logic;
            D:IN std_logic;
                CLK:IN std_logic;
                RESET:IN std_logic;
                CE:IN std_logic
          );
END COMPONENT;
```

```
uut:DFFRE
        GENERIC MAP(INIT=>'0')
        PORT MAP (
            Q=>Q,
            D=>D,
            CLK=>CLK,
            RESET=>RESET,
            CE=>CE
         );
```

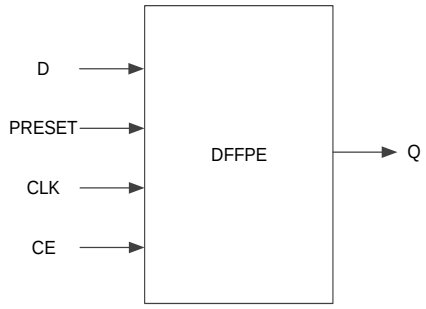
3.4.3 DFFPE

原语介绍

DFFPE（D Flip-Flop with Clock Enable and Asynchronous Preset）是上升沿触发的 D 触发器，具有异步置位和时钟使能功能。

端口示意图

图 3-12 DFFPE 端口示意图



端口介绍

表 3-31 DFFPE 端口介绍

端口	I/O	描述
D	Input	数据输入信号
CLK	Input	时钟输入信号
PRESET	Input	异步置位信号，高电平有效。
CE	Input	时钟使能信号
Q	Output	数据输出信号

参数介绍

表 3-32 DFFPE 参数介绍

参数	范围	默认	描述
INIT	1'b1	1'b1	DFFPE 初始值

原语例化

Verilog 例化：

```
DFFPE instName (
        .D(D),
        .CLK(CLK),
        .PRESET(PRESET),
        .CE(CE),
        .Q(Q)
);
defparam instName.INIT=1'b1;
```

Vhdl 例化：

```
COMPONENT DFFPE
        GENERIC (INIT:bit:='1');
        PORT(
            Q:OUT std_logic;
            D:IN std_logic;
                CLK:IN std_logic;
                PRESET:IN std_logic;
                CE:IN std_logic
        );
END COMPONENT;
uut:DFFPE
        GENERIC MAP(INIT=>'1')
        PORT MAP (
            Q=>Q,
            D=>D,
            CLK=>CLK,
            PRESET=>PRESET,
            CE=>CE
        );
```

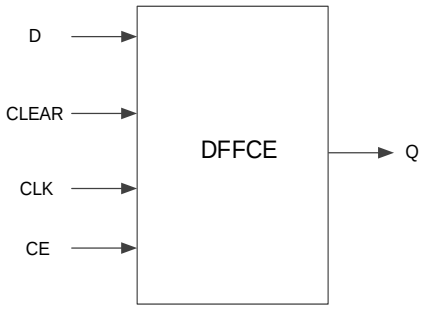
3.4.4 DFFCE

原语介绍

DFFCE（D Flip-Flop with Clock Enable and Asynchronous Clear）是上升沿触发的 D 触发器，具有异步清零和时钟使能功能。

端口示意图

图 3-13 DFFCE 端口示意图



端口介绍

表 3-33 DFFCE 端口介绍

端口	I/O	描述
D	Input	数据输入信号
CLK	Input	时钟输入信号
CLEAR	Input	异步清零信号，高电平有效。
CE	Input	时钟使能信号
Q	Output	数据输出信号

参数介绍

表 3-34 DFFCE 参数介绍

参数	范围	默认	描述
INIT	1'b0	1'b0	DFFCE 初始值

原语例化

Verilog 例化：

```
DFFCE instName (
    .D(D),
    .CLK(CLK),
    .CLEAR(CLEAR),
    .CE(CE),
    .Q(Q)
```

```
);
defparam instName.INIT=1'b0;
```

Vhdl 例化：

```
COMPONENT DFFCE
        GENERIC (INIT:bit:='0');
        PORT(
            Q:OUT std_logic;
            D:IN std_logic;
                CLK:IN std_logic;
                CLEAR:IN std_logic;
                CE:IN std_logic
        );
END COMPONENT;
uut:DFFCE
        GENERIC MAP(INIT=>'0')
        PORT MAP (
            Q=>Q,
            D=>D,
            CLK=>CLK,
            CLEAR=>CLEAR,
            CE=>CE
        );
```

3.5 LATCH

锁存器是一种对电平触发的存储单元电路，其可在特定输入电平作用下改变状态。与 LATCH 相关的原语有 2 个，如表 3-35 所示。

表 3-35 与 LATCH 相关的原语

原语	描述
DLCE	带异步清零和锁存使能的数据锁存器
DLPE	带异步预置位和锁存使能的数据锁存器

放置规则

表 3-36 LATCH 类型

编号	类型 1	类型 2
1	DLCE	DLPE

- 相同类型的 DL，可以放置在同一个 CLS 的 2 个 FF 上，除数据输入 pin 外的其它输入必须共线；
- 不同类型的 DL，表 3-36 中同一编号的两种类型可以放置在同一个 CLS 的 2 个 FF 上，除数据输入 pin 外的其它输入必须共线；
- 可以约束 DL 和 ALU 在同一个 CLS 的相同或不同位置；
- 可以约束 DL 和 LUT 在同一个 CLS 的相同或不同位置。

注！

共线是指必须是同一条 net，经过反相器前后的两条 net 为不共线，不可放置在同一个 CLS。

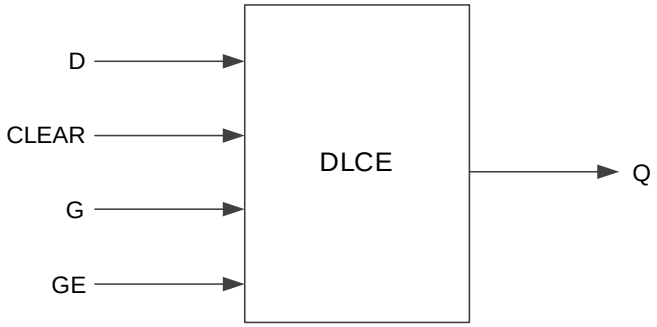
3.5.1 DLCE

原语介绍

DLCE（Data Latch with Asynchronous Clear and Latch Enable）是具有使能控制和异步清零功能的一种锁存器，控制信号 G 高电平有效。

端口示意图

图 3-14 DLCE 端口示意图



端口介绍

表 3-37 DLCE 端口介绍

端口	I/O	描述
D	Input	数据输入信号
CLEAR	Input	异步清零信号，高电平有效。
G	Input	数据控制信号，高电平有效。
GE	Input	电平使能信号
Q	Output	数据输出信号

参数介绍

表 3-38 DLCE 参数介绍

参数	范围	默认	描述
INIT	1'b0	1'b0	DLCE 初始值

原语例化

Verilog 例化：

```
DLCE instName (
        .D(D),
        .CLEAR(CLEAR),
        .G(G),
        .GE(GE),
        .Q(Q)
);
defparam instName.INIT=1'b0;
```

Vhdl 例化：

```
COMPONENT DLCE
        GENERIC (INIT:bit:='0');
        PORT(
            Q:OUT std_logic;
            D:IN std_logic;
            G:IN std_logic;
            GE:IN std_logic;
            CLEAR:IN std_logic
        );
END COMPONENT;
```

```
uut:DLCE
    GENERIC MAP(INIT=>'0')
    PORT MAP (
        Q=>Q,
        D=>D,
        G=>G,
        GE=>GE,
        CLEAR=>CLEAR
    );
```

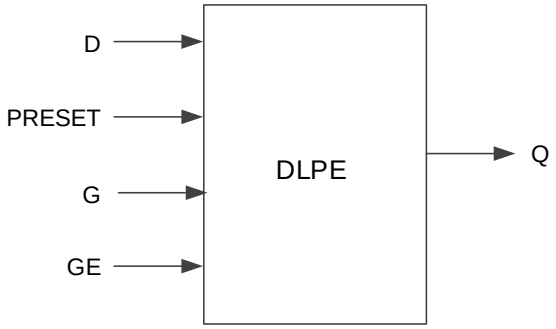
3.5.2 DLPE

原语介绍

DLPE（Data Latch with Asynchronous Preset and Latch Enable）是具有使能控制和置位功能的一种锁存器，控制信号 G 高电平有效。

端口示意图

图 3-15 DLPE 端口示意图



端口介绍

表 3-39 DLPE 端口介绍

端口	I/O	描述
D	Input	数据输入信号
PRESET	Input	异步置位信号，高电平有效。
G	Input	数据控制信号，高电平有效。
GE	Input	电平使能信号
Q	Output	数据输出信号

参数介绍

表 3-40 DLPE 参数介绍

参数	范围	默认	描述
INIT	1'b1	1'b1	DLPE 初始值

原语例化

Verilog 例化：

```
DLPE instName (
        .D(D),
        .PRESET(PRESET),
        .G(G),
        .GE(GE),
        .Q(Q)
);
defparam instName.INIT=1'b1;
```

Vhdl 例化：

```
COMPONENT DLPE
        GENERIC (INIT:bit:='1');
        PORT(
            Q:OUT std_logic;
            D:IN std_logic;
            G:IN std_logic;
            GE:IN std_logic;
            PRESET:IN std_logic
        );
END COMPONENT;
uut:DLPE
        GENERIC MAP(INIT=>'1')
        PORT MAP (
            Q=>Q,
            D=>D,
            G=>G,
            GE=>GE
            PRESET =>PRESET
        );
```

3.6 SSRAM

SSRAM 原语可参考 *UG300, Arora V 存储器(BSRAM & SSRAM)用户指南*。


GOWIN高云
智慧逻辑 定制未来