Rate This Document

IGLOO, Fusion and ProASIC3 Macro Library Guide

for Software v8.4

Actel®

Actel Corporation, Mountain View, CA 94043

Part Number: 5-02-00029-11

Release: August 2008

Actel makes no warranties with respect to this documentation and disclaims any implied warranties of merchantability or fitness for a particular purpose. Information in this document is subject to change without notice. Actel assumes no responsibility for any errors that may appear in this document.

This document contains confidential proprietary information that is not to be disclosed to any unauthorized person without prior written consent of Actel Corporation.

Trademarks

Actel and the Actel logo are registered trademarks of Actel Corporation.

Adobe and Acrobat Reader are registered trademarks of Adobe Systems, Inc.

All other products or brand names mentioned are trademarks or registered trademarks of their respective holders.

Table of Contents

Introduction

Lists

Introduction

This macro library guide supports only the IGLOO, Fusion, and ProASIC3 familes.

IGLOO indicates IGLOO PLUS, IGLOO and IGLOOe families; ProASIC3 indicates ProASIC3, ProASIC3E, and ProASIC3L families.

For information on macros available in other familes, see the *Antifuse Macro Library Guide* (for the MX, eX, SX, SX-A, and Axcelerator devices) or the *ProASIC/ProASIC$^{\underline{PLUS}}$ MLG*, as appropriate.

The naming convention of previous antifuse families has been a source of confusion. With ProASIC3/E we are introducing a new naming convention for sequential macros that is unambiguous and extensible, making it possible to understand the function of the macros by their name alone.

The first two mandatory characters of the macro name will indicate the basic macro function:

- DF - D-type flip-flop
- TF - Toggle flip-flop
- JF - JK flip-flop
- DL - D-type latch

The next mandatory character indicates the output polarity:

- I - output inverted (QN with bubble)
- N - output non-inverted (Q without bubble)

The next mandatory number indicates the polarity of the clock or gate:

- 1 - rising edge triggered flip-flop or transparent high latch (non-bubbled)
- 0 - falling edge triggered flip-flop or transparent low latch (bubbled)

The next two optional characters indicate the polarity of the Enable pin, if present:

- E0 - active low enable (bubbled)
- E1 - active high enable (non-bubbled)

The next two optional characters indicate the polarity of the asynchronous Preset pin, if present:

- P0 - active low preset (bubbled)
- P1 - active high preset (non-bubbled)

The next two optional characters indicate the polarity of the asynchronous Clear pin, if present:

- C0 - active low preset (bubbled)
- C1 - active high preset (non-bubbled)

Combinatorial macros all use one tile in the Fusion and ProASIC3/E families.

List of Combinational Macros

List of Sequential Macros

List of RAM Macros

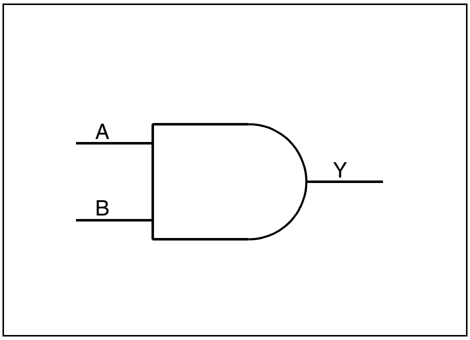
List of Input/Output Macros

Alphabetical List of Macros

Combinational/Sequential Macros

AND2

IGLOO, Fusion, ProASIC3



Function

2-Input AND

Truth Table

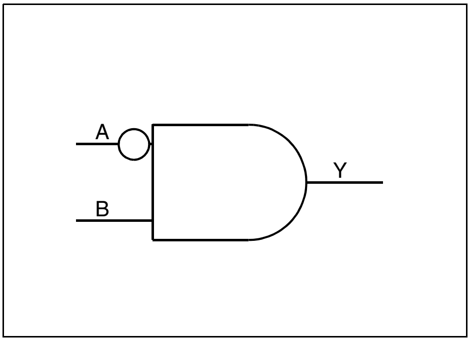
A	B	Y
X	0	0
0	X	0
1	1	1

Input	Output
A, B	Y

Family	Tiles
All	1

AND2A

IGLOO, Fusion, ProASIC3



Function

2-Input AND with active low A Input

Truth Table

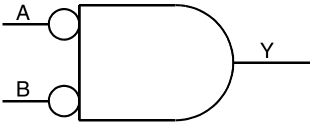
A	B	Y
X	0	0
0	1	1
1	X	0

Input	Output
A, B	Y

Family	Tiles
All	1

AND2B

IGLOO, Fusion, ProASIC3



Input	Output
A, B	Y

Function

2-Input AND with active low Inputs

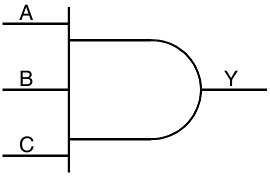
Truth Table

A	B	Y
0	0	1
X	1	0
1	X	0

Family	Tiles
All	1

AND3

IGLOO, Fusion, ProASIC3



Input	Output
A, B,C	Y

Function

3-Input AND

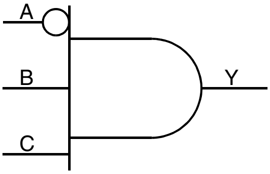
Truth Table

A	B	C	Y
X	X	0	0
X	0	X	0
0	X	X	0
1	1	1	1

Family	Tiles
All	1

AND3A

IGLOO, Fusion, ProASIC3



Function

3-Input AND with active low A-Input

Truth Table

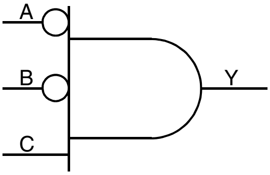
A	B	C	Y
X	X	0	0
X	0	X	0
0	1	1	1
1	X	X	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

AND3B

IGLOO, Fusion, ProASIC3



Function

3-Input AND with active low A- and B-Inputs

Truth Table

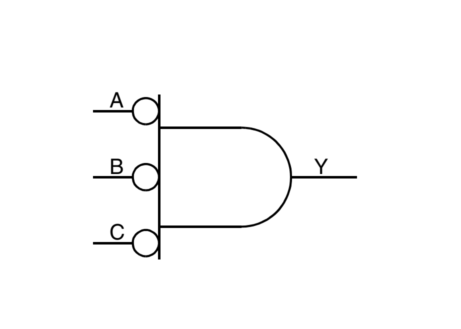
A	B	C	Y
X	X	0	0
0	0	1	1
X	1	X	0
1	X	X	0

Input	Output
A, B,C	Y

Family	Tiles
All	1

AND3C

IGLOO, Fusion, ProASIC3



Function

3-Input AND with active low Inputs

Truth Table

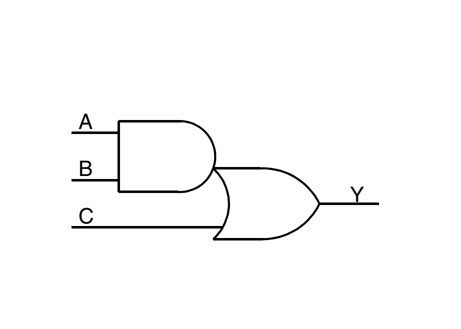
A	B	C	Y
0	0	0	1
X	X	1	0
X	1	X	0
1	X	X	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

AO1

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR

Truth Table

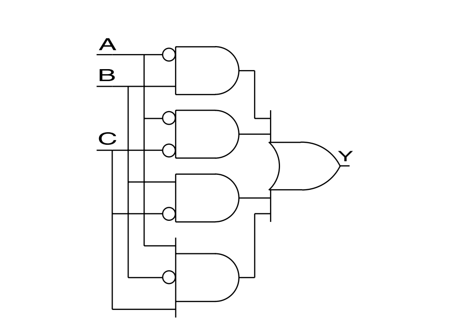
A	B	C	Y
X	0	0	0
X	X	1	1
0	X	0	0
1	1	X	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

AO12

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input AND-OR

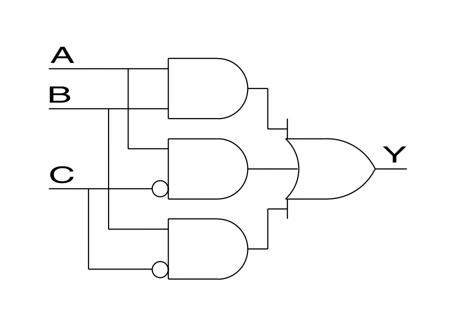
Truth Table

A	B	C	Y
0	0	0	1
1	0	0	0
0	1	0	1
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	0

Family	Tiles
All	1

AO13

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input AND-OR

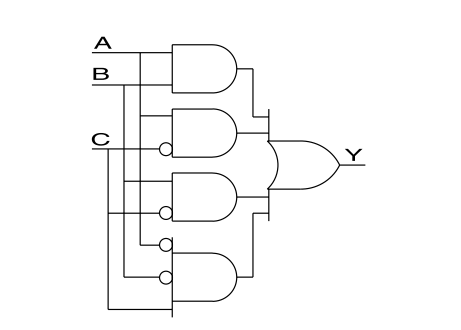
Truth Table

A	B	C	Y
0	0	0	0
1	0	0	1
0	1	0	1
1	1	0	1
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	1

Family	Tiles
All	1

AO14

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR

Input
A, B, C

Output
Y

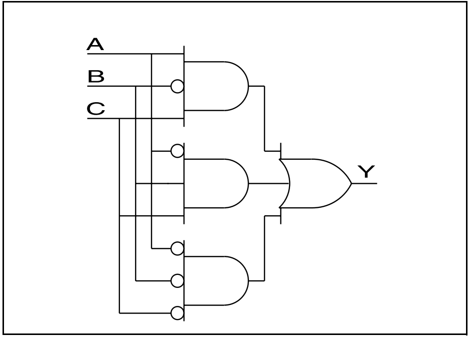
Truth Table

A	B	C	Y
0	0	0	0
1	0	0	1
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	1

Family	Tiles
All	1

AO15

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR

Input
A, B, C

Output
Y

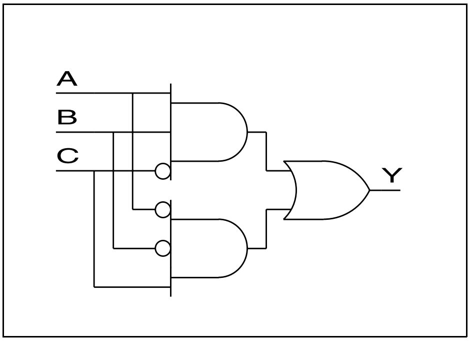
Truth Table

A	B	C	Y
0	0	0	1
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	0

Family	Tiles
All	1

AO16

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input AND-OR

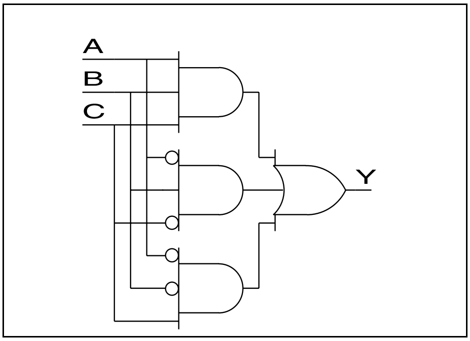
Truth Table

A	B	C	Y
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	0

Family	Tiles
All	1

AO17

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input AND-OR

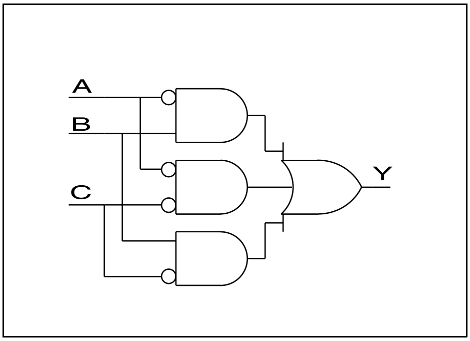
Truth Table

A	B	C	Y
0	0	0	0
1	0	0	0
0	1	0	1
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	1

Family	Tiles
All	1

AO18

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input AND-OR

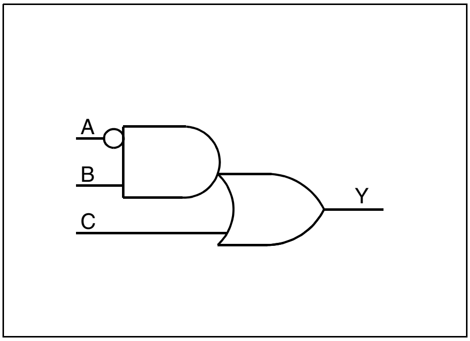
Truth Table

A	B	C	Y
0	0	0	1
1	0	0	0
0	1	0	1
1	1	0	1
0	0	1	0
1	0	1	0
0	1	1	1
1	1	1	0

Family	Tiles
All	1

AO1A

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input AND-OR with active low A-Input

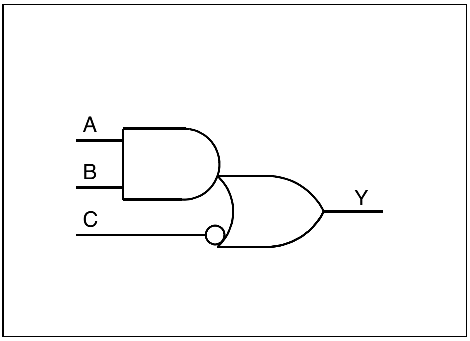
Truth Table

A	B	C	Y
X	0	0	0
X	X	1	1
0	1	X	1
1	X	0	0

Family	Tiles
All	1

AO1B

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR with active low C-Input

Truth Table

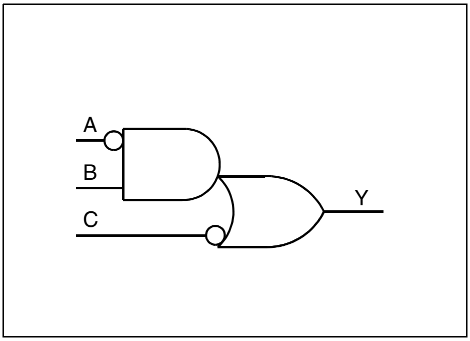
A	B	C	Y
X	X	0	1
X	0	1	0
0	X	1	0
1	1	X	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

AO1C

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR with active low A- and C-Inputs

Truth Table

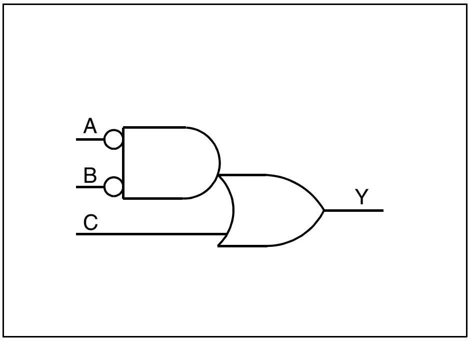
A	B	C	Y
X	X	0	1
X	0	1	0
0	1	X	1
1	X	1	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

AO1D

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR with active low A- and B-Inputs

Truth Table

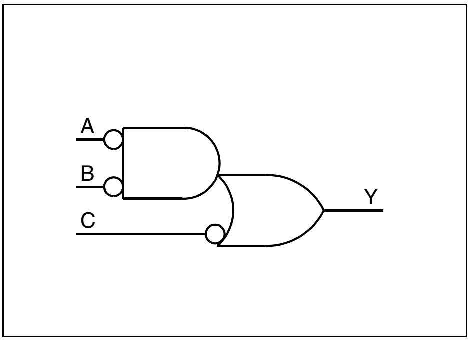
A	B	C	Y
0	0	X	1
X	1	0	0
X	X	1	1
1	X	0	0

Input	**Output**
A, B, C	Y

Family	Tiles
All	1

AO1E

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR with active low Inputs

Truth Table

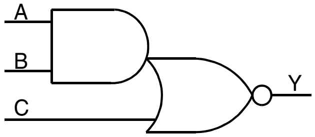
A	B	C	Y
X	X	0	1
0	0	X	1
X	1	1	0
1	X	1	0

Input	**Output**
A, B, C	Y

Family	Tiles
All	1

AOI1

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR-INVERT

Truth Table

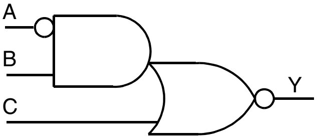
A	B	C	Y
X	0	0	1
X	X	1	0
0	X	0	1
1	1	X	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

AOI1A

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR-INVERT with active low A-Input

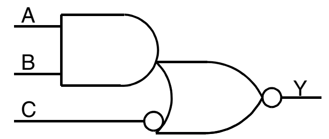
Truth Table

A	B	C	Y
X	0	0	1
X	X	1	0
0	1	X	0
1	X	0	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

AOI1B



Input
A, B, C

Output
Y

Function

3-Input AND-OR-INVERT with active low C-Input

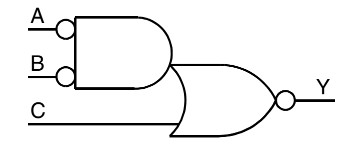
Truth Table

A	B	C	Y
X	X	0	0
X	0	1	1
0	X	1	1
1	1	X	0

Family	Tiles
All	1

AOI1C

IGLOO, Fusion, ProASIC3



Function

3 Input AND-OR-INVERT with active low A- and B-Inputs

Truth Table

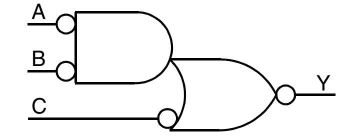
A	B	C	Y
0	0	X	0
X	1	0	1
X	X	1	0
1	X	0	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

AOI1D

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR-INVERT with active low Inputs

Truth Table

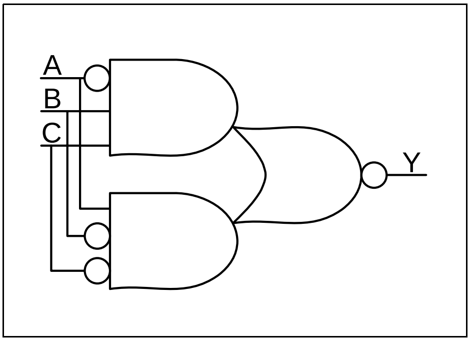
A	B	C	Y
X	X	0	0
0	0	X	0
X	1	1	1
1	X	1	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

AOI5

IGLOO, Fusion, ProASIC3



Function

3-Input AND-OR-INVERT

Truth Table

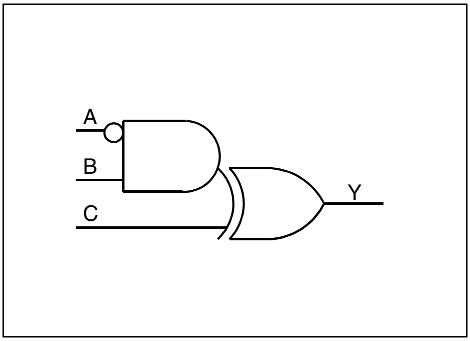
A	B	C	Y
0	0	0	1
1	0	0	0
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	1
0	1	1	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

AX1

IGLOO, Fusion, ProASIC3



Function

3-Input AND-XOR with active low A-Input

Truth Table

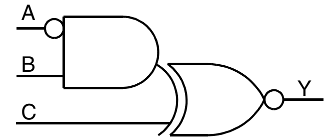
A	B	C	Y
X	0	0	0
X	0	1	1
0	1	0	1
0	1	1	0
1	X	0	0
1	X	1	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

AX1A

IGLOO, Fusion, ProASIC3



Input
A, B, C

Output
Y

Function

3-Input AND-XOR-INVERT with active low A-Input

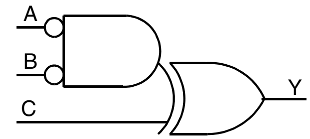
Truth Table

A	B	C	Y
X	0	0	1
X	0	1	0
0	1	0	0
0	1	1	1
1	X	0	1
1	X	1	0

Family	Tiles
All	1

AX1B

IGLOO, Fusion, ProASIC3



Input
A, B, C

Output
Y

Function

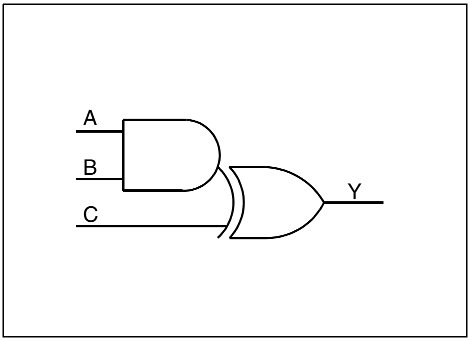
3-Input AND-XOR with active low A- and B-Inputs

Truth Table

A	B	C	Y
0	0	0	1
0	0	1	0
X	1	0	0
X	1	1	1
1	X	0	0
1	X	1	1

Family	Tiles
All	1

AX1C



Input	Output
A, B, C	Y

Function

3-Input AND-XOR

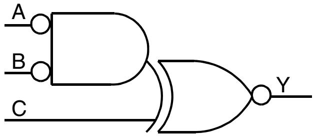
Truth Table

A	B	C	Y
X	0	0	0
X	0	1	1
0	X	0	0
0	X	1	1
1	1	0	1
1	1	1	0

Family	Tiles
All	1

AX1D

IGLOO, Fusion, ProASIC3



Input
A, B, C

Output
Y

Function

3-Input AND-XNOR

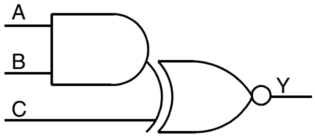
Truth Table

A	B	C	Y
0	0	0	0
1	0	0	1
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	0

Family	Tiles
All	1

AX1E

IGLOO, Fusion, ProASIC3



Input
A, B, C

Output
Y

Function

3-Input AND-XNOR

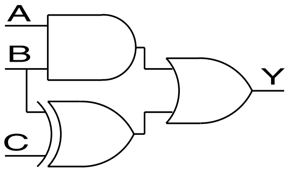
Truth Table

A	B	C	Y
0	0	0	1
1	0	0	1
0	1	0	1
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	1

Family	Tiles
All	1

AXO1

IGLOO, Fusion, ProASIC3



Function

3-Input Combinatorial Gate

Truth Table

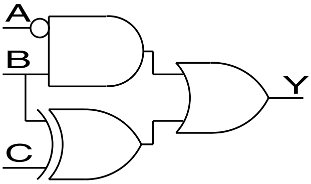
A	B	C	Y
0	0	0	0
1	0	0	0
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	1
0	1	1	0
1	1	1	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

AXO2

IGLOO, Fusion, ProASIC3



Function

3-Input Combinatorial Gate

Truth Table

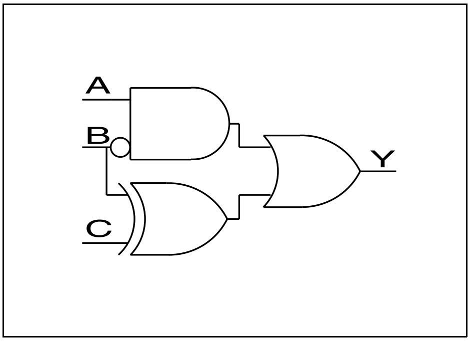
A	B	C	Y
0	0	0	0
1	0	0	0
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	1
0	1	1	1
1	1	1	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

AXO3

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input Combinatorial Gate

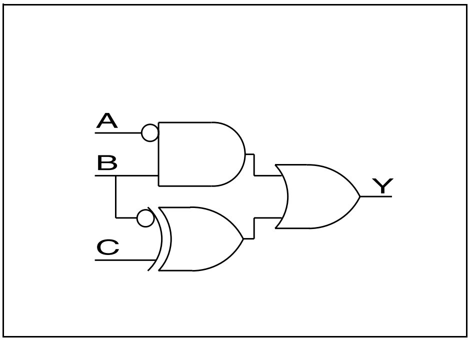
Truth Table

A	B	C	Y
0	0	0	0
1	0	0	1
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	1
0	1	1	0
1	1	1	0

Family	Tiles
All	1

AXO5

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

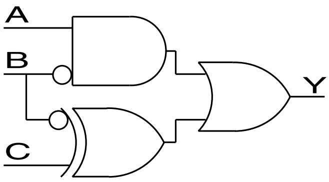
3-Input Combinatorial Gate

Truth Table

A	B	C	Y
0	0	0	1
1	0	0	1
0	1	0	1
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	1
1	1	1	1

Family	Tiles
All	1

AXO6 IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

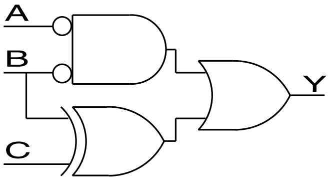
3-Input Combinatorial Gate

Truth Table

A	B	C	Y
0	0	0	1
1	0	0	1
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Family	Tiles
All	1

AXO7 IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input Combinatorial Gate

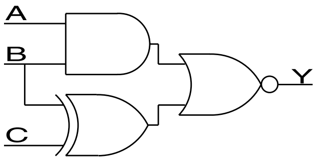
Truth Table

A	B	C	Y
0	0	0	1
1	0	0	0
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	1
0	1	1	0
1	1	1	0

Family	Tiles
All	1

AXOI1

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input Combinatorial Gate

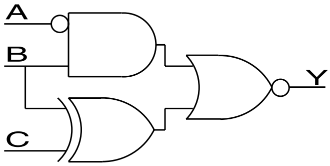
Truth Table

A	B	C	Y
0	0	0	1
1	0	0	1
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	1
1	1	1	0

Family	Tiles
All	1

AXOI2

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input Combinatorial Gate

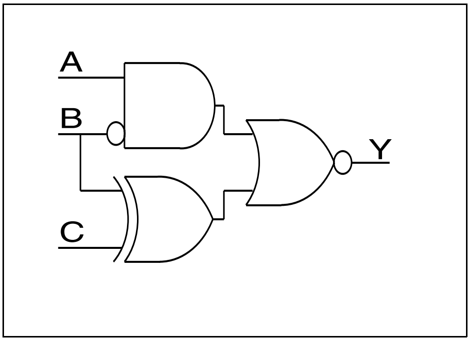
Truth Table

A	B	C	Y
0	0	0	1
1	0	0	1
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	1

Family	Tiles
All	1

AXOI3

IGLOO, Fusion, ProASIC3



Function

3-Input Combinatorial Gate

Truth Table

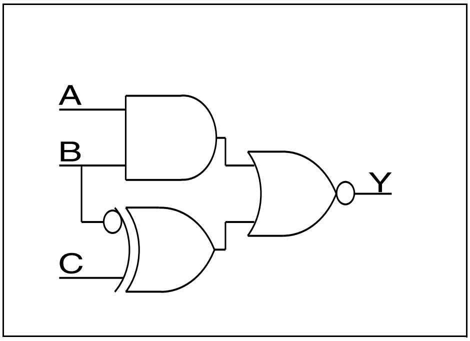
A	B	C	Y
0	0	0	1
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	1
1	1	1	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

AXOI4

IGLOO, Fusion, ProASIC3



Function

3-Input Combinatorial Gate

Truth Table

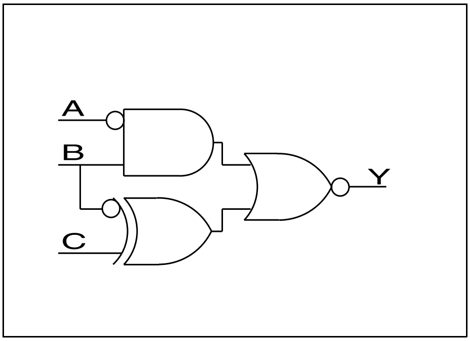
A	B	C	Y
0	0	0	0
1	0	0	0
0	1	0	1
1	1	0	0
0	0	1	1
1	0	1	1
0	1	1	0
1	1	1	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

AXOI5

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input Combinatorial Gate

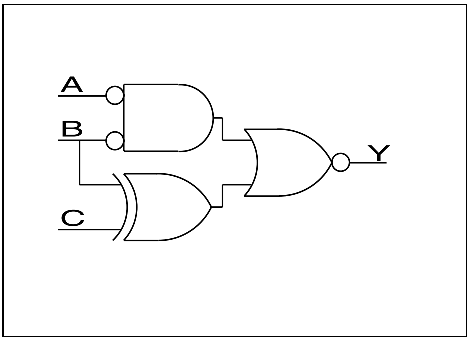
Truth Table

A	B	C	Y
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	1
1	0	1	1
0	1	1	0
1	1	1	0

Family	Tiles
All	1

AXOI7

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input Combinatorial Gate

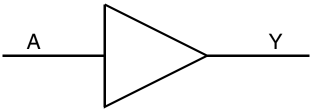
Truth Table

A	B	C	Y
0	0	0	0
1	0	0	1
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	1
1	1	1	1

Family	Tiles
All	1

BUFF

IGLOO, Fusion, ProASIC3



Input
A

Output
Y

Function

Buffer

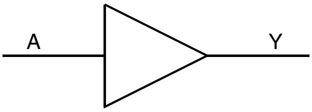
Truth Table

A	Y
0	0
1	1

Family	Tiles
All	1

BUFD

IGLOO, Fusion, ProASIC3



Input
A

Output
Y

Function

Buffer
NOTE: The Combiner will not remove this macro

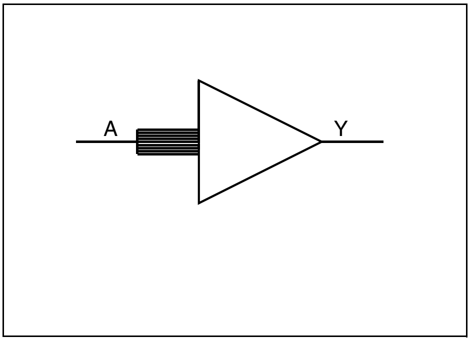
Truth Table

A	Y
0	0
1	1

Family	Tiles
All	1

CLKINT

IGLOO, Fusion, ProASIC3



Function

Internal Clock Interface

Truth Table

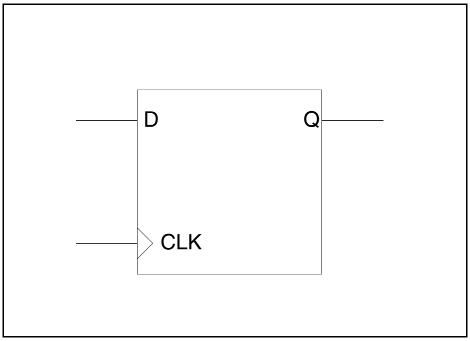
A	Y
0	0
1	1

Input	Output
A	Y

NOTE: CLKINT does not use any tiles. .
For more information on the Global Clock Network, refer to the latest Actel datasheet.

DFN1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop

Truth Table

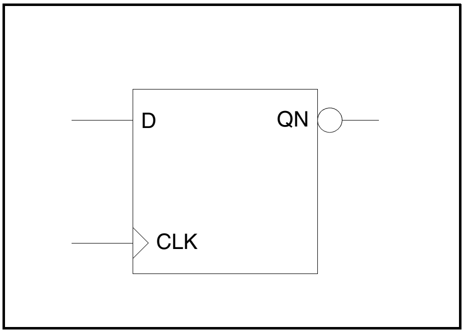
CLK	Q_{n+1}
↑	D

Input	Output
D, CLK	Q

Family	Tiles
All	1

DFI1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with inverted Output

Truth Table

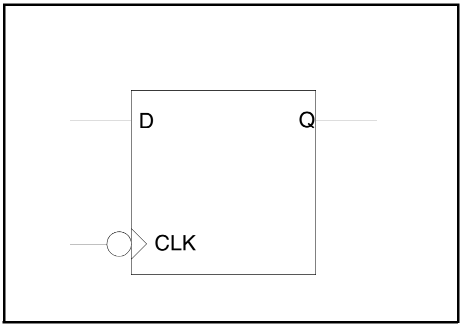
CLK	QN_{n+1}
↑	!D

Input	Output
D, CLK	QN

Family	Tiles
All	1

DFN0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Clock

Truth Table

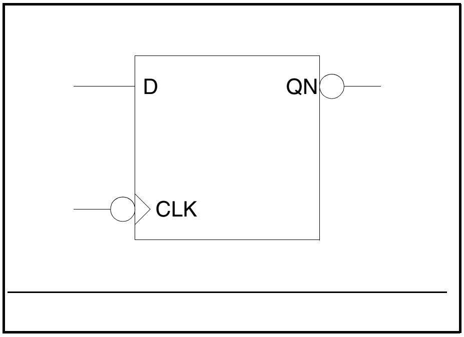
CLK	Q_{n+1}
↓	D

Input	**Output**
D, CLK	Q

Family	Tiles
All	1

DFI0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Clock and inverted Output

Truth Table

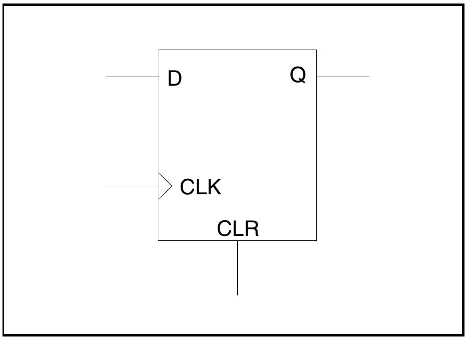
CLK	QN_{n+1}
↓	!D

Input	**Output**
D, CLK	QN

Family	Tiles
All	1

DFN1C1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Clear

Truth Table

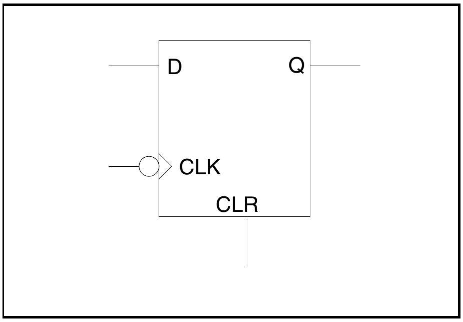
CLR	CLK	Q_{n+1}
1	X	0
0	↑	D

Input	**Output**
CLR, D, CLK	Q

Family	Tiles
All	1

DFN0C1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Clear and active low Clock

Truth Table

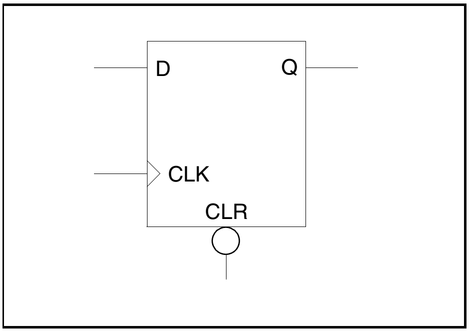
CLR	CLK	Q_{n+1}
1	X	0
0	↓	D

Input	**Output**
CLR, D, CLK	Q

Family	Tiles
All	1

DFN1C0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Clear

Truth Table

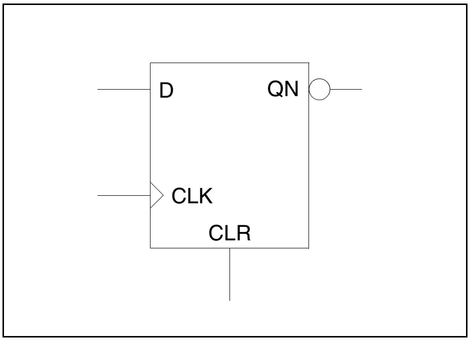
CLR	CLK	Q_{n+1}
0	X	0
1	↑	D

Input	Output
CLR, D, CLK	Q

Family	Tiles
All	1

DFI1C1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Clear and Clock

Truth Table

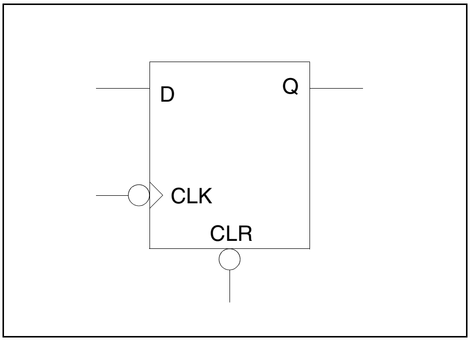
CLR	CLK	QN_{n+1}
1	X	1
0	↑	!D

Input	Output
CLR, D, CLK	QN

Family	Tiles
All	1

DFN0C0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Clear and Clock

Truth Table

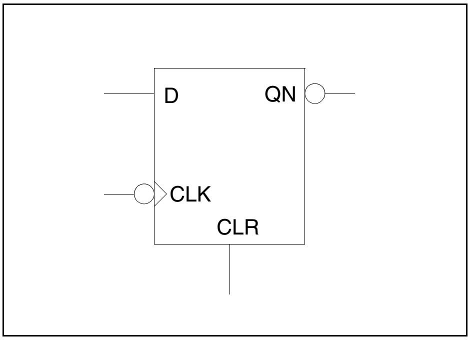
CLR	CLK	Q_{n+1}
0	X	0
1	↓	D

Input	Output
CLR, D, CLK	Q

Family	Tiles
All	1

DFI0C1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Clear, active low Clock and inverted Output

Truth Table

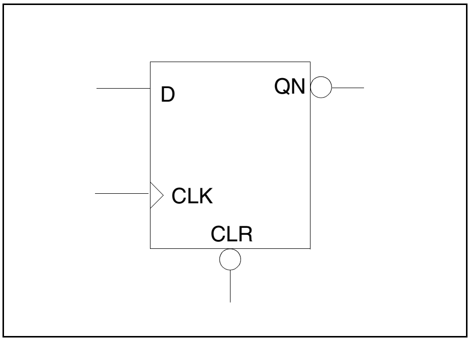
CLR	CLK	QN_{n+1}
1	X	1
0	↓	!D

Input	Output
CLR, D, CLK	QN

Family	Tiles
All	1

DFI1C0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Clear and inverted Output

Truth Table

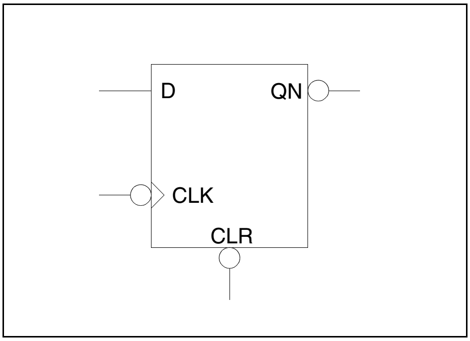
CLR	CLK	QN_{n+1}
0	X	1
1	↑	!D

Input	Output
CLR, D, CLK	QN

Family	Tiles
All	1

DFI0C0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Clear, Clock and inverted Output

Truth Table

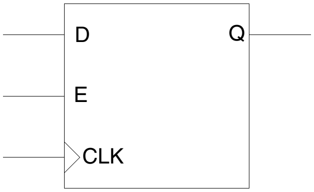
CLR	CLK	QN
0	X	1
1	↓	!D

Input	Output
CLR, D, CLK	QN

Family	Tiles
All	1

DFN1E1

IGLOO, Fusion, ProASIC3



Input	Output
D, E, CLK	Q

Function

D-Type Flip-Flop with active high Enable

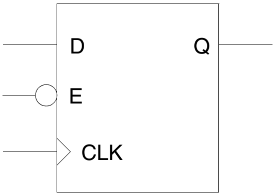
Truth Table

E	CLK	Q_{n+1}
0	X	Q
1	↑	D

Family	Tiles
All	1

DFN1E0

IGLOO, Fusion, ProASIC3



Input	Output
D, E, CLK	Q

Function

D-Type Flip-Flop with active low Enable

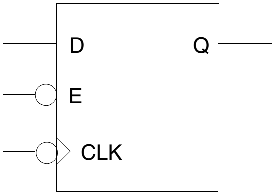
Truth Table

E	CLK	Q_{n+1}
1	X	Q
0	↑	D

Family	Tiles
All	1

DFN0E0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable and Clock

Truth Table

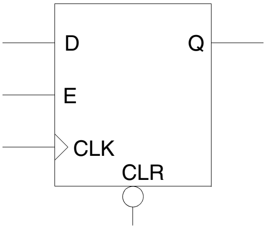
E	CLK	Q_{n+1}
1	X	Q
0	↓	D

Input	**Output**
D, E, CLK	Q

Family	Tiles
All	1

DFN1E1C0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop, with Enable and active low Clear

Truth Table

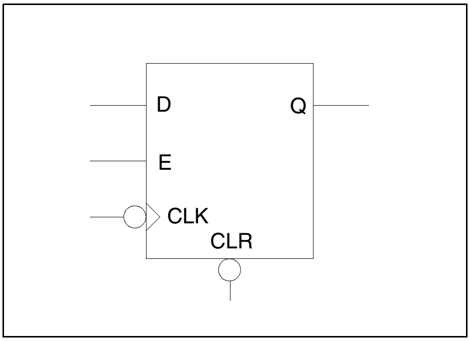
CLR	E	CLK	Q_{n+1}
0	X	X	0
1	0	X	Q
1	1	↑	D

Input	**Output**
CLR, D, E, CLK	Q

Family	Tiles
All	1

DFN0E1C0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with Enable and active low Clear and Clock

Truth Table

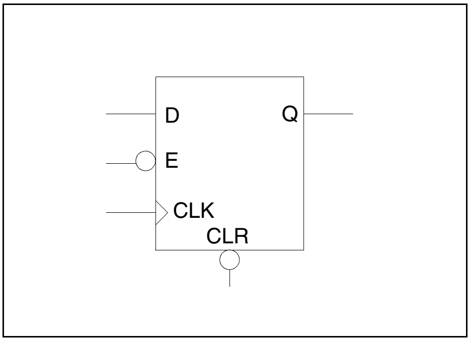
CLR	E	CLK	Q_{n+1}
0	X	X	0
1	0	X	Q
1	1	↓	D

Input	**Output**
CLR, D, E, CLK	Q

Family	Tiles
All	1

DFN1E0C0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with Active Low Enable and Clear

Truth Table

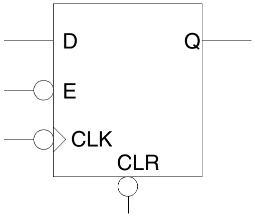
CLR	E	CLK	Q_{n+1}
0	X	X	0
1	1	X	Q
1	0	↑	D

Input	**Output**
CLR, D, E, CLK	Q

Family	Tiles
All	1

DFN0E0C0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable, Clear and Clock

Truth Table

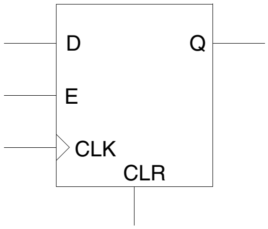
CLR	E	CLK	Q_{n+1}
0	X	X	0
1	1	X	Q
1	0	↓	D

Input	Output
CLR, D, E, CLK	Q

Family	Tiles
All	1

DFN1E1C1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop, with Enable and active high Clear

Truth Table

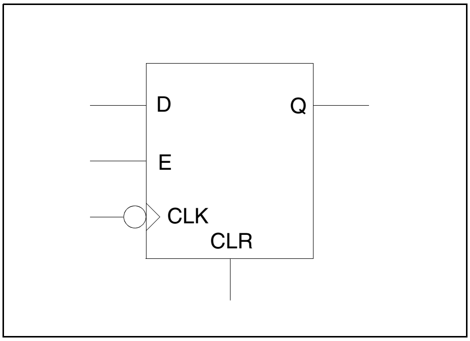
CLR	E	CLK	Q_{n+1}
1	X	X	0
0	0	X	Q
0	1	↑	D

Input	Output
CLR, D, E, CLK	Q

Family	Tiles
All	1

DFN0E1C1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with Enable and active high Clear and active low Clock

Truth Table

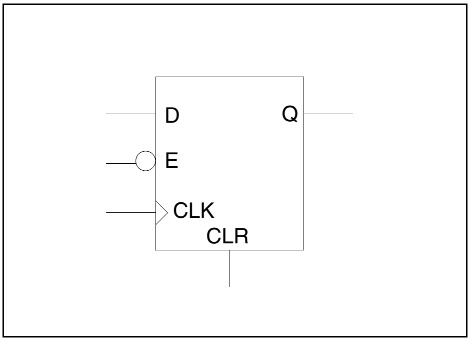
CLR	E	CLK	Q_{n+1}
1	X	X	0
0	0	X	Q
0	1	↓	D

Input	**Output**
CLR, D, E, CLK	Q

Family	Tiles
All	1

DFN1E0C1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with Active Low Enable and active high Clear

Truth Table

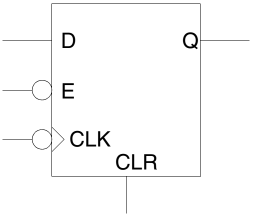
CLR	E	CLK	Q_{n+1}
1	X	X	0
0	1	X	Q
0	0	↑	D

Input	**Output**
CLR, D, E, CLK	Q

Family	Tiles
All	1

DFN0E0C1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable, Clock and active high Clear

Truth Table

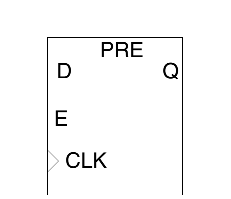
CLR	E	CLK	Q_{n+1}
1	X	X	0
0	1	X	Q
0	0	↓	D

Input	Output
CLR, D, E, CLK	Q

Family	Tiles
All	1

DFN1E1P1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Enable and Preset

Truth Table

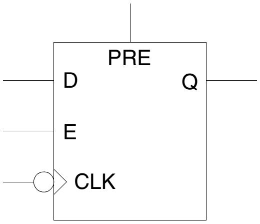
PRE	E	CLK	Q_{n+1}
1	X	X	1
0	0	X	Q
0	1	↑	D

Input	Output
D, E, PRE, CLK	Q

Family	Tiles
All	1

DFN0E1P1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Enable and Preset, and active low Clock

Truth Table

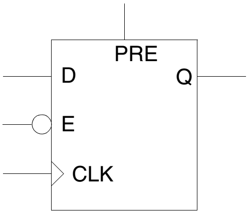
PRE	E	CLK	Q_{n+1}
1	X	X	1
0	0	X	Q
0	1	↓	D

Input	Output
D, E, PRE, CLK	Q

Family	Tiles
All	1

DFN1E0P1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable, and active high Preset

Truth Table

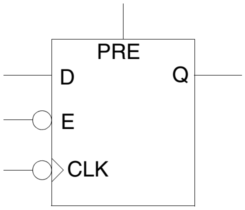
PRE	E	CLK	Q_{n+1}
1	X	X	1
0	1	X	Q
0	0	↑	D

Input	Output
D, E, PRE, CLK	Q

Family	Tiles
All	1

DFN0E0P1

IGLOO, Fusion, ProASIC3



Input	Output
D, E, PRE, CLK	Q

Function

D-Type Flip-Flop with active low Enable and Clock, and active high Preset

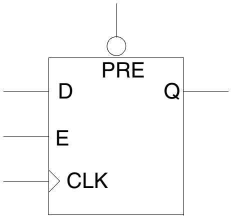
Truth Table

PRE	E	CLK	Q_{n+1}
1	X	X	1
0	1	X	Q
0	0	↓	D

Family	Tiles
All	1

DFN1E1P0

IGLOO, Fusion, ProASIC3



Input	Output
D, E, PRE, CLK	Q

Function

D-Type Flip-Flop with active high Enable and active low Preset

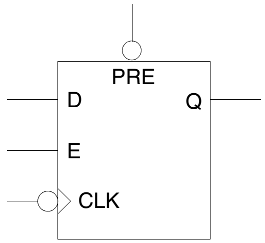
Truth Table

PRE	E	CLK	Q_{n+1}
0	X	X	1
1	0	X	Q
1	1	↑	D

Family	Tiles
All	1

DFN0E1P0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Enable and active low Preset and Clock

Truth Table

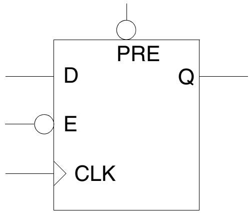
PRE	E	CLK	Q_{n+1}
0	X	X	1
1	0	X	Q
1	1	↓	D

Input	Output
D, E, PRE, CLK	Q

Family	Tiles
All	1

DFN1E0P0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable and Preset

Truth Table

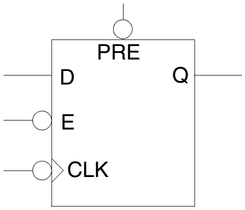
PRE	E	CLK	Q_{n+1}
0	X	X	1
1	1	X	Q
1	0	↑	D

Input	Output
D, E, PRE, CLK	Q

Family	Tiles
All	1

DFN0E0P0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable, Clock, and Preset

Truth Table

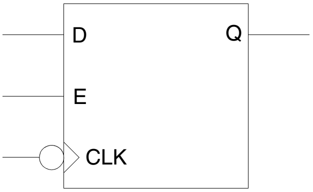
PRE	E	CLK	Q_{n+1}
0	X	X	1
1	1	X	Q
1	0	↓	D

Input	Output
D, E, PRE, CLK	Q

Family	Tiles
All	1

DFN0E1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop, with Enable, and active low Clock

Truth Table

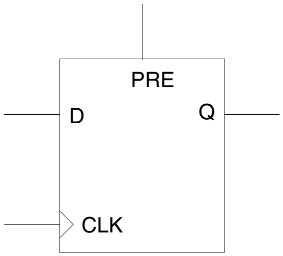
E	CLK	Q_{n+1}
0	X	Q
1	↓	D

Input	Output
D, E, CLK	Q

Family	Tiles
All	1

DFN1P1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Preset

Truth Table

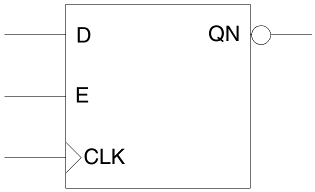
PRE	CLK	Q_{n+1}
1	X	1
0	↑	D

Input	**Output**
D, PRE, CLK	Q

Family	Tiles
All	1

DFI1E1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Enable and inverted output

Truth Table

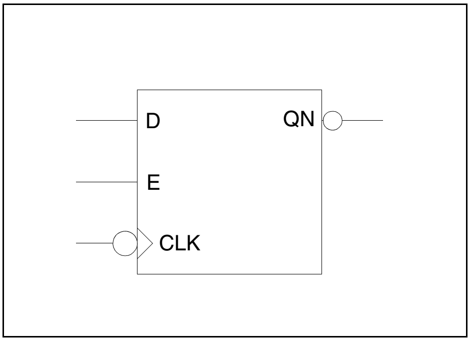
E	CLK	QN_{n+1}
0	X	QN
1	↑	!D

Input	**Output**
D, E, CLK	QN

Family	Tiles
All	1

DFI0E1

IGLOO, Fusion, ProASIC3



Input	Output
D, E, CLK	QN

Function

D-Type Flip-Flop, with Enable, and active low Clock and inverted ouput

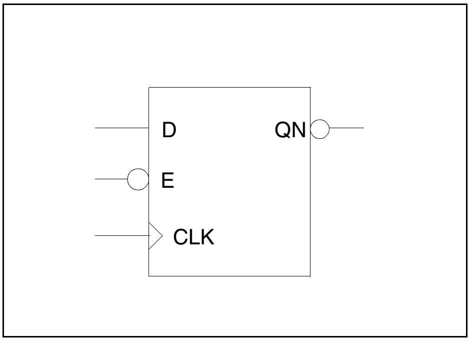
Truth Table

E	CLK	QN_{n+1}
0	X	QN
1	↓	!D

Family	Tiles
All	1

DFI1E0

IGLOO, Fusion, ProASIC3



Input	Output
D, E, CLK	QN

Function

D-Type Flip-Flop with active low Enable and inverted output

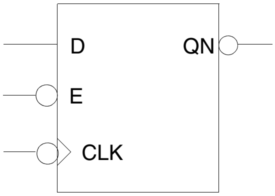
Truth Table

E	CLK	QN_{n+1}
1	X	QN
0	↑	!D

Family	Tiles
All	1

DFI0E0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable and Clock and inverted output

Truth Table

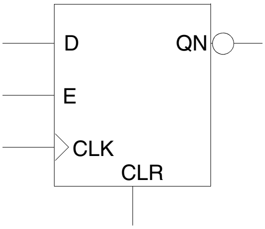
E	CLK	QN_{n+1}
1	X	QN
0	↓	!D

Input	**Output**
D, E, CLK	QN

Family	Tiles
All	1

DFI1E1C1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop, with Enable and active high Clear and inverted output

Truth Table

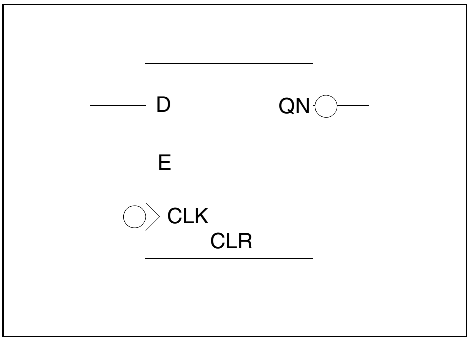
CLR	E	CLK	QN_{n+1}
1	X	X	1
0	0	X	QN
0	1	↑	!D

Input	**Output**
CLR, D, E, CLK	QN

Family	Tiles
All	1

DFI0E1C1

IGLOO, Fusion, ProASIC3



Input	Output
CLR, D, E, CLK	QN

Function

D-Type Flip-Flop with Enable and active high Clear and active low Clock and inverted output

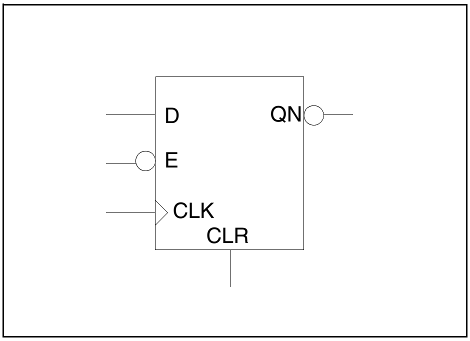
Truth Table

CLR	E	CLK	QN_{n+1}
1	X	X	1
0	0	X	QN
0	1	↓	!D

Family	Tiles
All	1

DFI1E0C1

IGLOO, Fusion, ProASIC3



Input	Output
CLR, D, E, CLK	QN

Function

D-Type Flip-Flop with Active Low Enable and anctive high Clear and inverted output

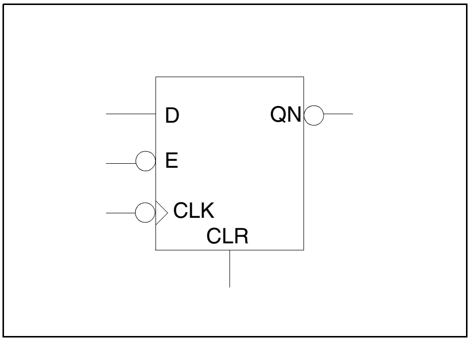
Truth Table

CLR	E	CLK	QN_{n+1}
1	X	X	1
0	1	X	QN
0	0	↑	!D

Family	Tiles
All	1

DFI0E0C1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable, Clock, active high Clear, and inverted output

Truth Table

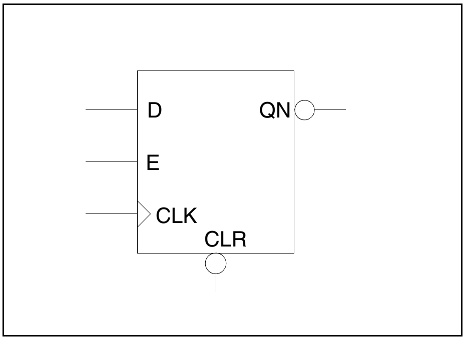
CLR	E	CLK	QN_{n+1}
1	X	X	1
0	1	X	QN
0	0	↓	!D

Input	Output
CLR, D, E, CLK	QN

Family	Tiles
All	1

DFI1E1C0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop, with Enable and active low Clear and inverted output

Truth Table

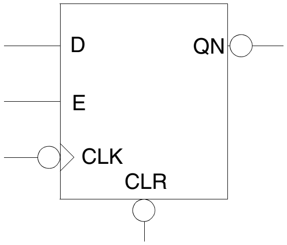
CLR	E	CLK	QN_{n+1}
0	X	X	1
1	0	X	QN
1	1	↑	!D

Input	Output
CLR, D, E, CLK	QN

Family	Tiles
All	1

DFI0E1C0

IGLOO, Fusion, ProASIC3



Input	Output
CLR, D, E, CLK	QN

Function

D-Type Flip-Flop with Enable and active low Clear and Clock and inverted output

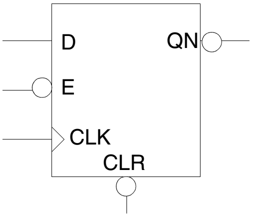
Truth Table

CLR	E	CLK	QN_{n+1}
0	X	X	1
1	0	X	QN
1	1	↓	!D

Family	Tiles
All	1

DFI1E0C0

IGLOO, Fusion, ProASIC3



Input	Output
CLR, D, E, CLK	QN

Function

D-Type Flip-Flop with Active Low Enable and Clear and inverted output

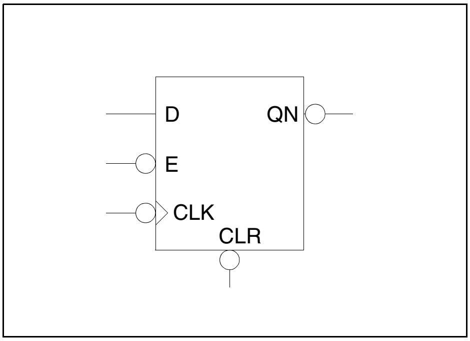
Truth Table

CLR	E	CLK	QN_{n+1}
0	X	X	1
1	1	X	QN
1	0	↑	!D

Family	Tiles
All	1

DFI0E0C0

IGLOO, Fusion, ProASIC3



Input	Output
CLR, D, E, CLK	QN

Function

D-Type Flip-Flop with active low Enable, Clear, Clock and inverted output

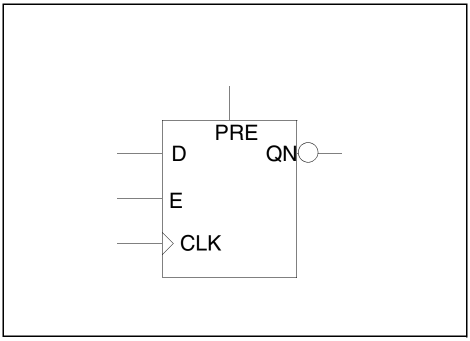
Truth Table

CLR	E	CLK	QN_{n+1}
0	X	X	1
1	1	X	QN
1	0	↓	!D

Family	Tiles
All	1

DFI1E1P1

IGLOO, Fusion, ProASIC3



Input	Output
D, E, PRE, CLK	QN

Function

D-Type Flip-Flop with active high Enable and Preset and inverted output

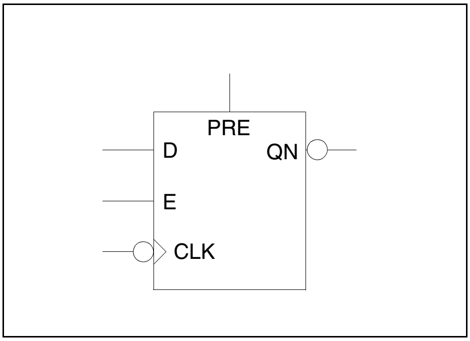
Truth Table

PRE	E	CLK	QN_{n+1}
1	X	X	0
0	0	X	QN
0	1	↑	!D

Family	Tiles
All	1

DFI0E1P1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Enable and Preset, active low Clock and inverted output

Truth Table

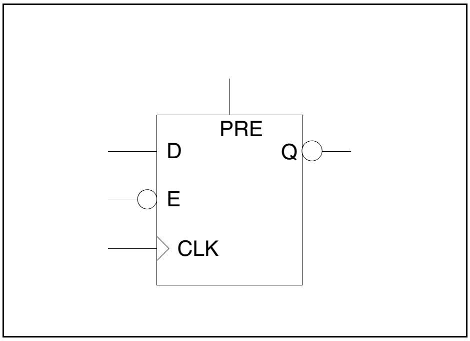
PRE	E	CLK	QN_{n+1}
1	X	X	0
0	0	X	QN
0	1	↓	!D

Input	**Output**
D, E, PRE, CLK	QN

Family	Tiles
All	1

DFI1E0P1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable, active high Preset and inverted output

Truth Table

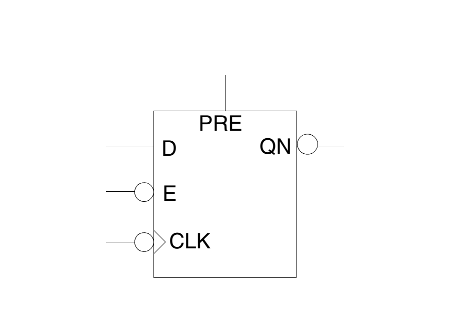
PRE	E	CLK	QN_{n+1}
1	X	X	0
0	1	X	QN
0	0	↑	!D

Input	**Output**
D, E, PRE, CLK	QN

Family	Tiles
All	1

DFI0E0P1

IGLOO, Fusion, ProASIC3



Input	Output
D, E, PRE, CLK	QN

Function

D-Type Flip-Flop with active low Enable and Clock, active high Preset and inverted output

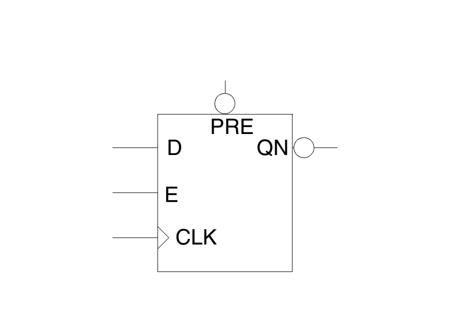
Truth Table

PRE	E	CLK	QN_{n+1}
1	X	X	0
0	1	X	QN
0	0	↓	!D

Family	Tiles
All	1

DFI1E1P0

IGLOO, Fusion, ProASIC3



Input	Output
D, E, PRE, CLK	QN

Function

D-Type Flip-Flop with active high Enable, active low Preset, and inverted output

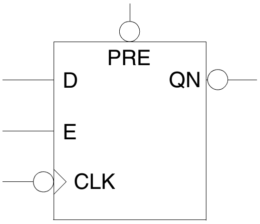
Truth Table

PRE	E	CLK	QN_{n+1}
0	X	X	0
1	0	X	QN
1	1	↑	!D

Family	Tiles
All	1

DFI0E1P0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Enable, active low Preset and Clock, and inverted output

Truth Table

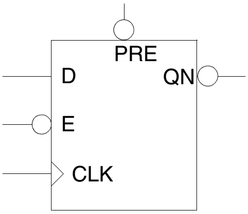
PRE	E	CLK	QN_{n+1}
0	X	X	0
1	0	X	QN
1	1	↓	!D

Input	Output
D, E, PRE, CLK	QN

Family	Tiles
All	1

DFI1E0P0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable and Preset, and inverted output

Truth Table

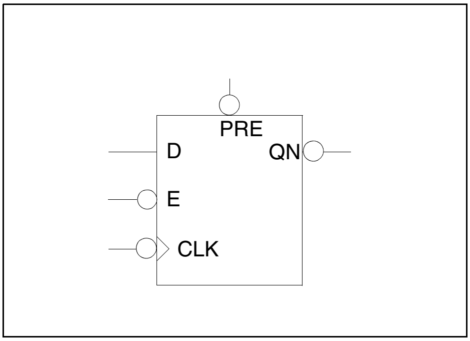
PRE	E	CLK	QN_{n+1}
0	X	X	0
1	1	X	QN
1	0	↑	!D

Input	Output
D, E, PRE, CLK	QN

Family	Tiles
All	1

DFI0E0P0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Enable, Clock, and Preset, and inverted output

Truth Table

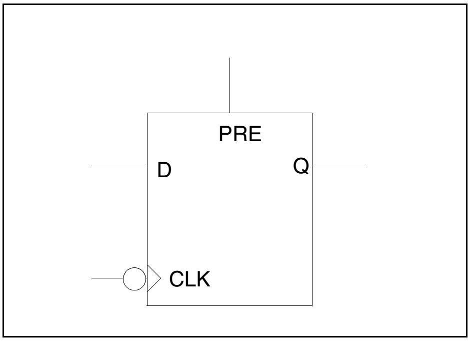
PRE	E	CLK	QN_{n+1}
0	X	X	0
1	1	X	QN
1	0	↓	!D

Input	Output
D, E, PRE, CLK	QN

Family	Tiles
All	1

DFN0P1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Preset, and active low Clock

Truth Table

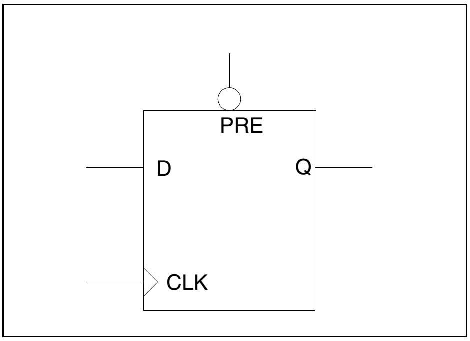
PRE	CLK	Q_{n+1}
1	X	1
0	↓	D

Input	Output
D, PRE, CLK	Q

Family	Tiles
All	1

DFN1P0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Preset

Truth Table

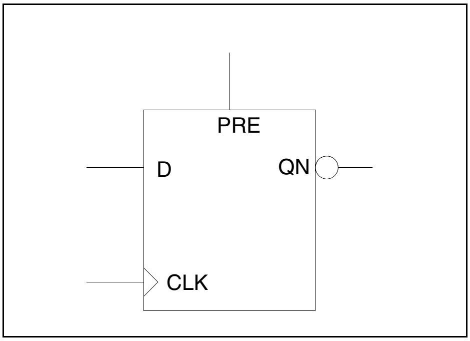
PRE	CLK	Q_{n+1}
0	X	1
1	↑	D

Input	Output
D, PRE, CLK	Q

Family	Tiles
All	1

DFI1P1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Preset, and inverted Output

Truth Table

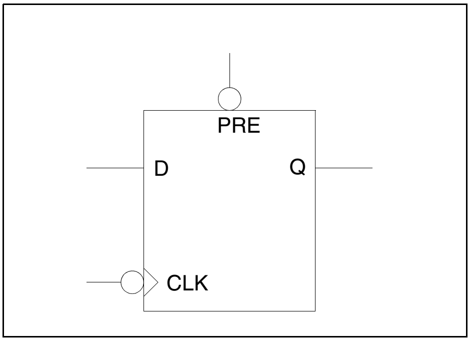
PRE	CLK	QN_{n+1}
1	X	0
0	↑	!D

Input	Output
D, PRE, CLK	QN

Family	Tiles
All	1

DFN0P0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Preset and Clock

Truth Table

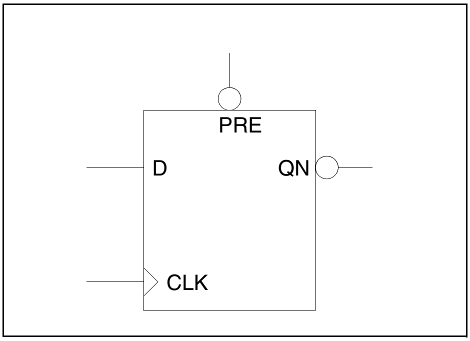
PRE	CLK	Q_{n+1}
0	X	1
1	↓	D

Input	**Output**
D, PRE, CLK	Q

Family	Tiles
All	1

DFI1P0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Preset and inverted Output

Truth Table

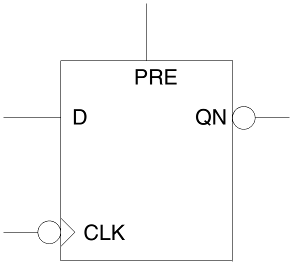
PRE	CLK	QN_{n+1}
0	X	0
1	↑	!D

Input	**Output**
D, PRE, CLK	QN

Family	Tiles
All	1

DFI0P1

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active high Preset, and active low Clock and inverted Output

Truth Table

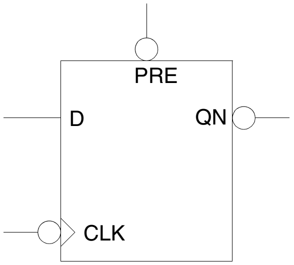
PRE	CLK	QN_{n+1}
1	X	0
0	↓	!D

Input	Output
D, PRE, CLK	QN

Family	Tiles
All	1

DFI0P0

IGLOO, Fusion, ProASIC3



Function

D-Type Flip-Flop with active low Preset, Clock and inverted Output

Truth Table

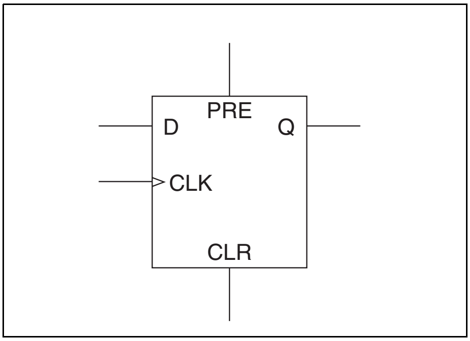
PRE	CLK	QN_{n+1}
0	X	0
1	↓	!D

Input	Output
D, PRE, CLK	QN

Family	Tiles
All	1

DFN1P1C1

IGLOO, Fusion, ProASIC3



Input	Output
CLR, PRE, CLK, D	Q

Function

Rising Edge Triggered D-Type Flip-Flop with Active High Preset and Clear

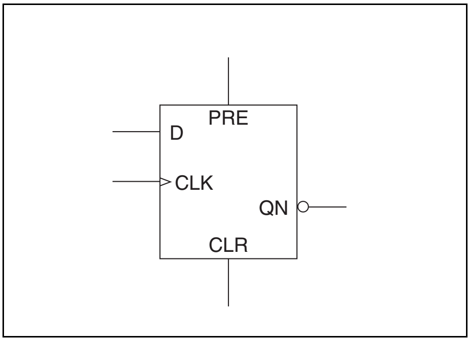
Truth Table

CLK	PRE	CLR	Q_{n+1}
X	1	0	1
X	X	1	0
↑	0	0	D

Family	Tiles
All	4

DFI1P1C1

IGLOO, Fusion, ProASIC3



Input	Output
CLR, PRE, CLK, D	QN

Function

Rising Edge Triggered D-Type Flip-Flop with Active High Preset and Clear and inverted Output

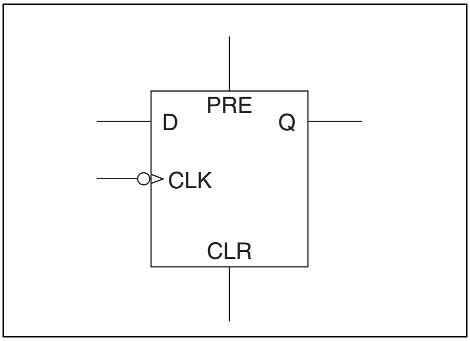
Truth Table

CLK	PRE	CLR	QN_{n+1}
X	1	0	0
X	X	1	1
↑	0	0	!D

Family	Tiles
All	4

DFN0P1C1

IGLOO, Fusion, ProASIC3



Function

Falling Edge Triggered D-Type Flip-Flop with Active High Preset and Clear

Truth Table

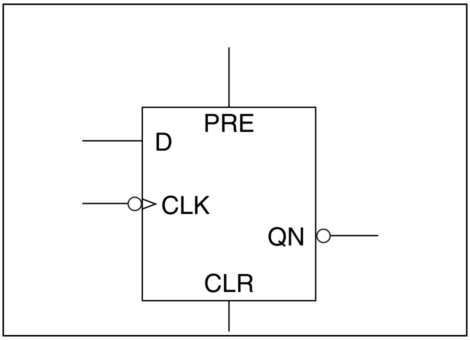
CLK	PRE	CLR	Q_{n+1}
X	1	0	1
X	X	1	0
↓	0	0	D

Input	**Output**
CLR, PRE, CLK, D	Q

Family	Tiles
All	4

DFI0P1C1

IGLOO, Fusion, ProASIC3



Function

Falling Edge Triggered D-Type Flip-Flop with Active High Preset and Clear and inverted Output

Truth Table

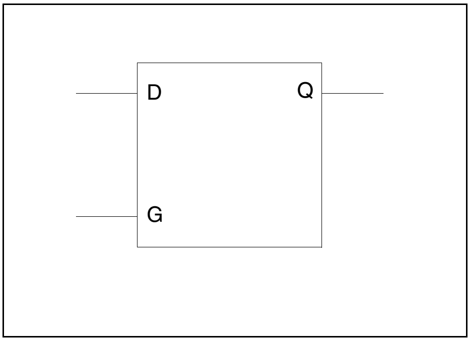
CLK	PRE	CLR	QN_{n+1}
X	1	0	0
X	X	1	1
↓	0	0	!D

Input	**Output**
CLR, PRE, CLK, D	QN

Family	Tiles
All	4

DLN1

IGLOO, Fusion, ProASIC3



Function

Data Latch

Truth Table

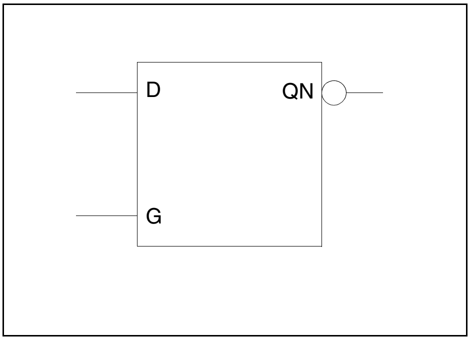
G	Q_{n+1}
0	Q
1	D

Input	Output
D, G	Q

Family	Tiles
All	1

DLI1

IGLOO, Fusion, ProASIC3



Function

Data Latch with inverted Output

Truth Table

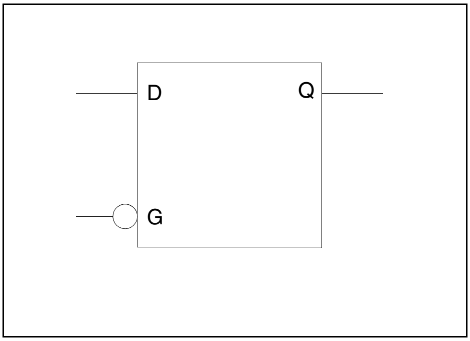
G	QN_{n+1}
0	QN
1	!D

Input	Output
D, G	QN

Family	Tiles
All	1

DLN0

IGLOO, Fusion, ProASIC3



Function

Data Latch with active low Clock

Truth Table

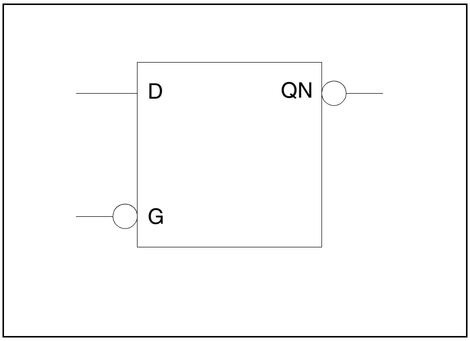
G	Q_{n+1}
1	Q
0	D

Input	**Output**
D, G	Q

Family	Tiles
All	1

DLI0

IGLOO, Fusion, ProASIC3



Function

Data Latch, with active low Clock and inverted Output

Truth Table

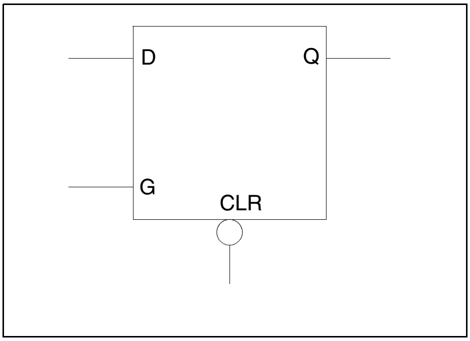
G	QN_{n+1}
1	QN
0	!D

Input	**Output**
D, G	QN

Family	Tiles
All	1

DLN1C0

IGLOO, Fusion, ProASIC3



Function

Data Latch with active low Clear

Truth Table

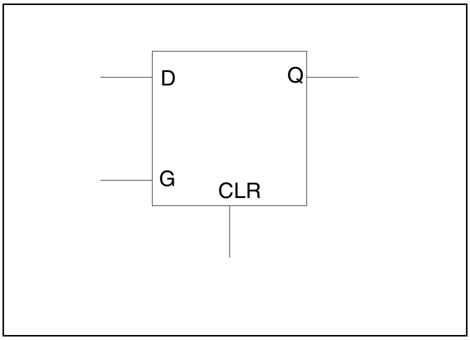
CLR	G	Q_{n+1}
0	X	0
1	0	Q
1	1	D

Input	**Output**
CLR, D, G	Q

Family	Tiles
All	1

DLN1C1

IGLOO, Fusion, ProASIC3



Function

Data Latch with active high Clear

Truth Table

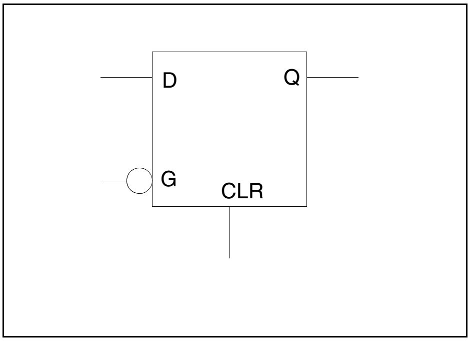
CLR	G	Q_{n+1}
1	X	0
0	0	Q
0	1	D

Input	**Output**
CLR, D, G	Q

Family	Tiles
All	1

DLN0C1

IGLOO, Fusion, ProASIC3



Function

Data Latch with active high Clear and active low Clock

Truth Table

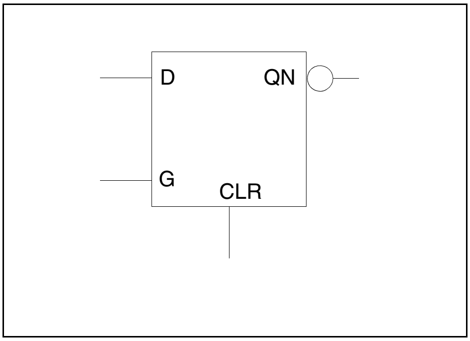
CLR	G	Q_{n+1}
1	X	0
0	1	Q
0	0	D

Input	Output
CLR, D, G	Q

Family	Tiles
All	1

DLI1C1

IGLOO, Fusion, ProASIC3



Function

Data Latch with active high Clear and inverted Output

Truth Table

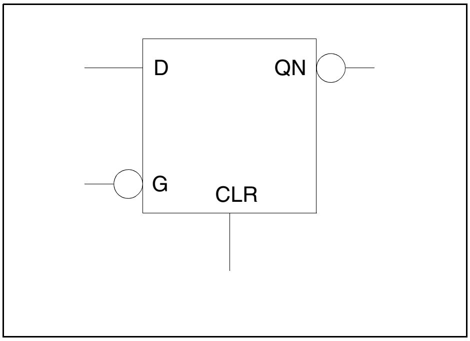
CLR	G	QN_{n+1}
1	X	1
0	0	QN
0	1	!D

Input	Output
CLR, D, G	QN

Family	Tiles
All	1

DLI0C1

IGLOO, Fusion, ProASIC3



Function

Data Latch with active high Clear and active low Clock and inverted Output

Truth Table

CLR	G	QN_{n+1}
1	X	1
0	1	QN
0	0	!D

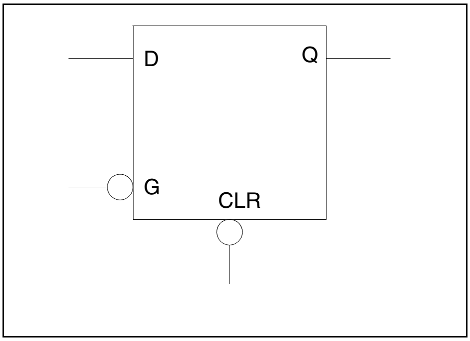
Input
CLR, D, G

Output
QN

Family	Tiles
All	1

DLN0C0

IGLOO, Fusion, ProASIC3



Function

Data Latch with active low Clear and Clock

Truth Table

CLR	G	Q_{n+1}
0	X	0
1	1	Q
1	0	D

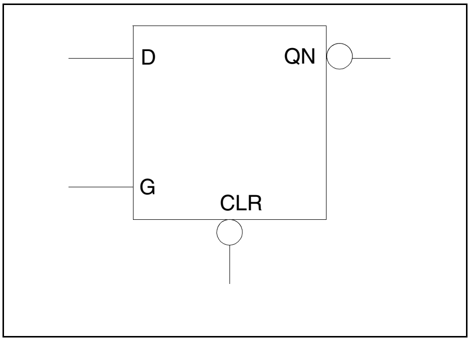
Input
CLR, D, G

Output
Q

Family	Tiles
All	1

DLI1C0

IGLOO, Fusion, ProASIC3



Function

Data Latch with active low Clear and inverted output

Truth Table

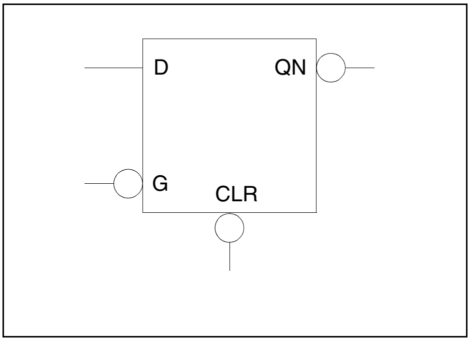
CLR	G	QN_{n+1}
0	X	1
1	0	QN
1	1	!D

Input	Output
CLR, D, G	QN

Family	Tiles
All	1

DLI0C0

IGLOO, Fusion, ProASIC3



Function

Data Latch with active low Clear, Clock, and inverted Output

Truth Table

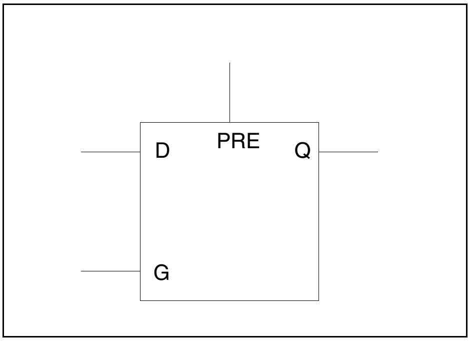
CLR	G	QN_{n+1}
0	X	1
1	1	QN
1	0	!D

Input	Output
CLR, D, G	QN

Family	Tiles
All	1

DLN1P1

IGLOO, Fusion, ProASIC3



Function

Data Latch with active high Preset and Clock

Truth Table

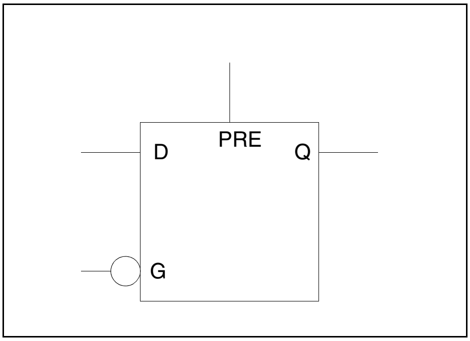
PRE	G	Q_{n+1}
1	X	1
0	0	Q
0	1	D

Input	Output
D, G, PRE	Q

Family	Tiles
All	1

DLN0P1

IGLOO, Fusion, ProASIC3



Function

Data Latch with active high Preset and active low Clock

Truth Table

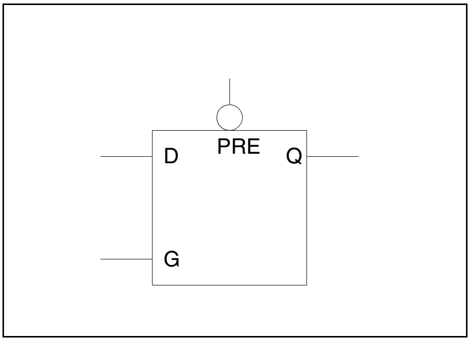
PRE	G	Q_{n+1}
1	X	1
0	1	Q
0	0	D

Input	Output
D, G, PRE	Q

Family	Tiles
All	1

DLN1P0

IGLOO, Fusion, ProASIC3



Function

Data Latch with active low Preset and active high Clock

Truth Table

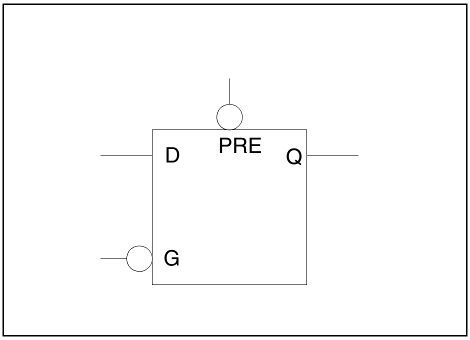
PRE	G	Q_{n+1}
0	X	1
1	0	Q
1	1	D

Input	**Output**
D, G, PRE	Q

Family	Tiles
All	1

DLN0P0

IGLOO, Fusion, ProASIC3



Function

Data Latch with active low Preset and Clock

Truth Table

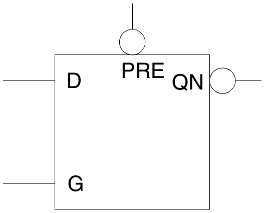
PRE	G	Q_{n+1}
0	X	1
1	1	Q
1	0	D

Input	**Output**
D, G, PRE	Q

Family	Tiles
All	1

DLI1P0

IGLOO, Fusion, ProASIC3



Function

Data Latch with active low Preset and Output, and active high Clock

Truth Table

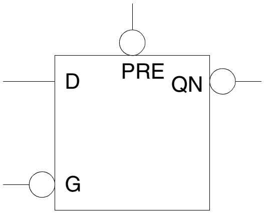
PRE	G	QN_{n+1}
0	X	0
1	0	QN
1	1	!D

Input	Output
D, G, PRE	QN

Family	Tiles
All	1

DLI0P0

IGLOO, Fusion, ProASIC3



Function

Data Latch with active low Preset, Clock, and inverted Output

Truth Table

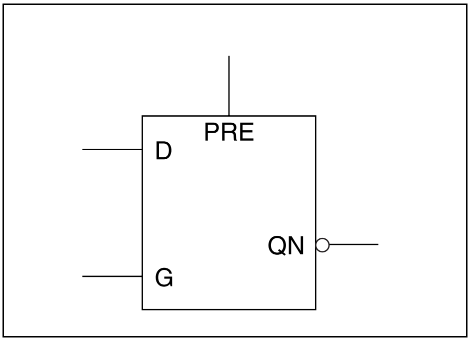
PRE	G	QN_{n+1}
0	X	0
1	0	!D
1	1	QN

Input	Output
D, G, PRE	QN

Family	Tiles
All	1

DLI1P1

IGLOO, Fusion, ProASIC3



Function

Active High Latch with Active High Preset and inverted Output

Truth Table

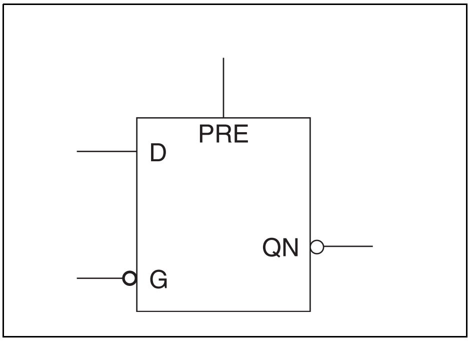
G	PRE	QN_{n+1}
X	1	0
0	0	QN
1	0	!D

Input	**Output**
PRE, G, D	QN

Family	Tiles
All	1

DLI0P1

IGLOO, Fusion, ProASIC3



Function

Active Low Latch with Active High Preset and inverted Output

Truth Table

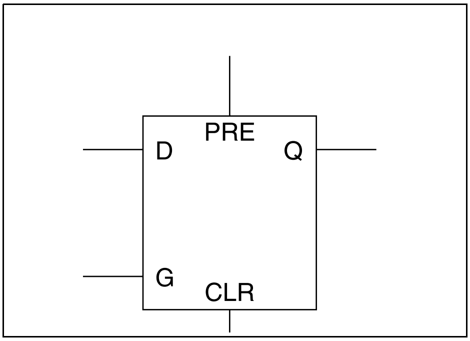
G	PRE	QN_{n+1}
X	1	0
0	0	!D
1	0	QN

Input	**Output**
PRE, G, D	QN

Family	Tiles
All	1

DLN1P1C1

IGLOO, Fusion, ProASIC3



Input	Output
CLR, PRE, G, D	Q

Function

Active High Latch with Active High Preset and Clear

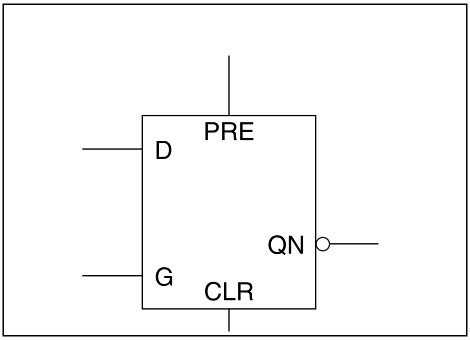
Truth Table

G	PRE	CLR	Q_{n+1}
X	1	0	1
X	X	1	0
1	0	0	D
0	0	0	Q

Family	Tiles
All	2

DLI1P1C1

IGLOO, Fusion, ProASIC3



Input	Output
CLR, PRE, G, D	QN

Function

Active High Latch with Active High Preset and Clear and inverted Output

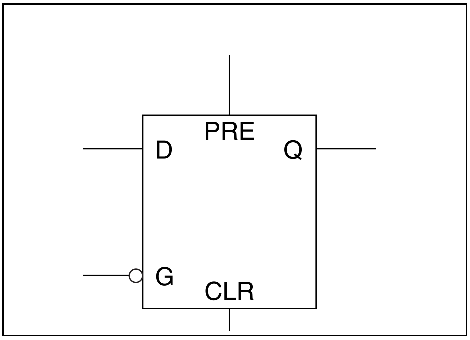
Truth Table

G	PRE	CLR	QN_{n+1}
X	1	0	0
X	X	1	1
1	0	0	!D
0	0	0	QN

Family	Tiles
All	2

DLN0P1C1

IGLOO, Fusion, ProASIC3



Function

Active Low Latch with Active High Preset and Clear

Truth Table

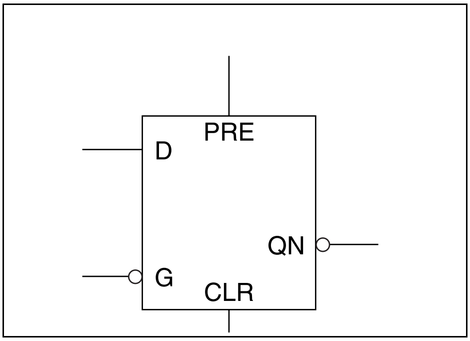
G	PRE	CLR	Q_{n+1}
X	1	0	1
X	X	1	0
0	0	0	D
1	0	0	Q

Input	**Output**
CLR, PRE, G, D	Q

Family	Tiles
All	2

DLI0P1C1

IGLOO, Fusion, ProASIC3



Function

Active Low Latch with Active High Preset and Clear and inverted Output

Truth Table

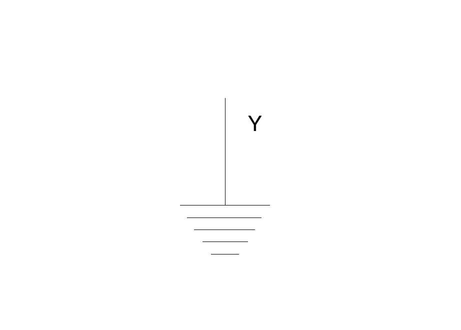
G	PRE	CLR	QN_{n+1}
X	1	0	0
X	X	1	1
0	0	0	!D
1	0	0	QN

Input	**Output**
CLR, PRE, G, D	QN

Family	Tiles
All	2

GND

IGLOO, Fusion, ProASIC3



Function

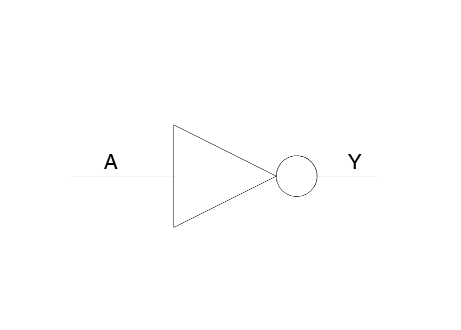
Ground

Input	**Output**
	Y

NOTE: Ground does not use any tiles.

INV

IGLOO, Fusion, ProASIC3



Function

Inverter with active low Output

Truth Table

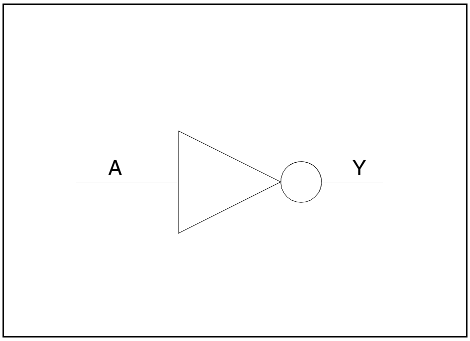
A	Y
0	1
1	0

Input	**Output**
A	Y

Family	Tiles
All	1

INVD

IGLOO, Fusion, ProASIC3



Function

Inverter with active low Output
NOTE: The Combiner will not remove this macro

Truth Table

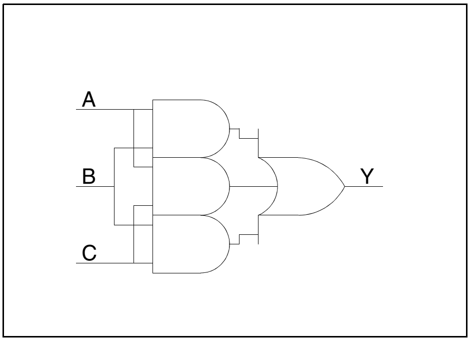
A	Y
0	1
1	0

Input	Output
A	Y

Family	Tiles
All	1

MAJ3

IGLOO, Fusion, ProASIC3



Function

3-Input majority function

Truth Table

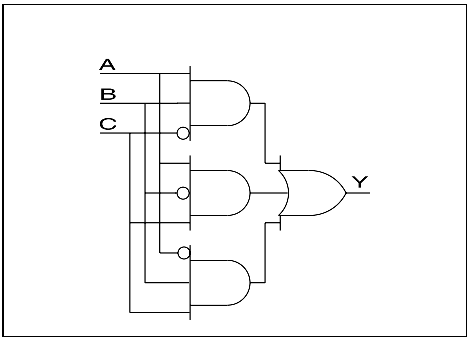
A	B	C	Y
X	0	0	0
0	0	X	0
0	X	0	0
X	1	1	1
1	X	1	1
1	1	X	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

MAJ3X

IGLOO, Fusion, ProASIC3



Function

2 of 3 function

Truth Table

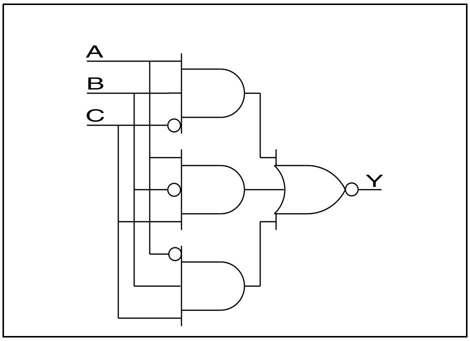
A	B	C	Y
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

MAJ3XI

IGLOO, Fusion, ProASIC3



Function

2 of 3 function with active low output

Truth Table

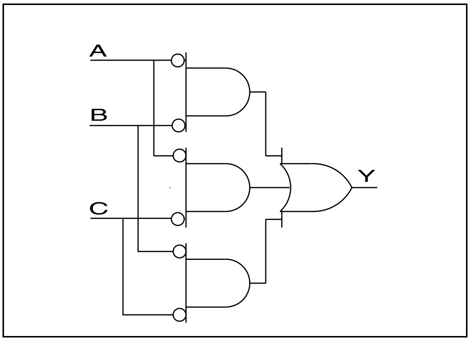
A	B	C	Y
0	0	0	1
1	0	0	1
0	1	0	1
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

MIN3

IGLOO, Fusion, ProASIC3



Input
A, B, C

Output
Y

Function

3-Input minority function

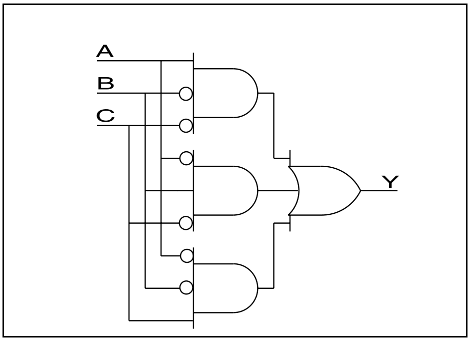
Truth Table

A	B	C	Y
X	0	0	1
0	0	X	1
0	X	0	1
X	1	1	0
1	X	1	0
1	1	X	0

Family	Tiles
All	1

MIN3X

IGLOO, Fusion, ProASIC3



Input
A, B, C

Output
Y

Function

1 of 3 function

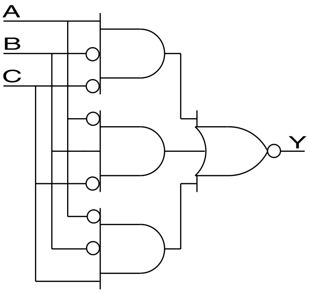
Truth Table

A	B	C	Y
0	0	0	0
1	0	0	1
0	1	0	1
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	0

Family	Modules	
	Seq	Comb
54SX, 54SX-A, 54SX-S, eX		1

MIN3XI

IGLOO, Fusion, ProASIC3



Function

1 of 3 function with active low output

Input	Output
A, B, C	Y

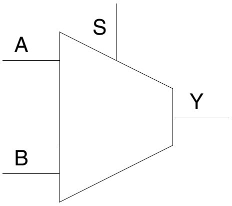
Truth Table

A	B	C	Y
0	0	0	1
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

Family	Tiles
All	1

MX2

IGLOO, Fusion, ProASIC3



Function

2 to 1 Multiplexer

Truth Table

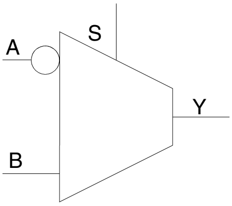
S	Y
0	A
1	B

Input	Output
A, B, S	Y

Family	Tiles
All	1

MX2A

IGLOO, Fusion, ProASIC3



Function

2 to 1 Multiplexer with active low A-Input

Truth Table

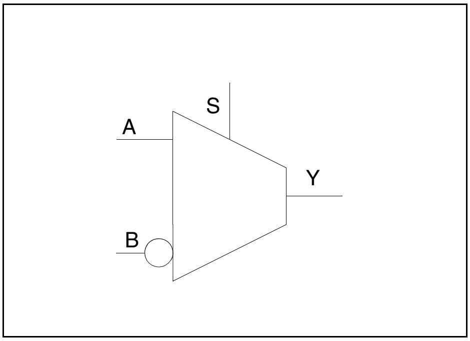
S	Y
0	!A
1	B

Input	Output
A, B, S	Y

Family	Tiles
All	1

MX2B

IGLOO, Fusion, ProASIC3



Function

2 to 1 Multiplexer with active low B-Input

Truth Table

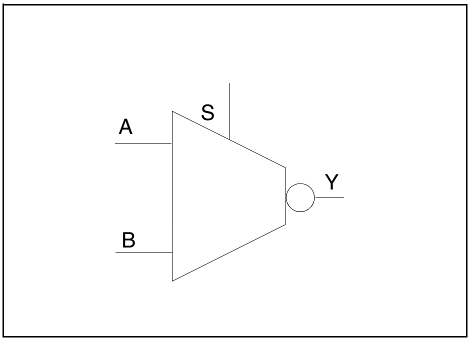
S	Y
0	A
1	!B

Input	Output
A, B, S	Y

Family	Tiles
All	1

MX2C

IGLOO, Fusion, ProASIC3



Function

2 to 1 Multiplexer with active low Output

Truth Table

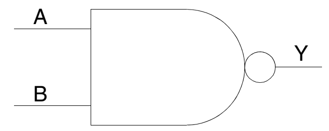
S	Y
0	!A
1	!B

Input	Output
A, B, S	Y

Family	Tiles
All	1

NAND2

IGLOO, Fusion, ProASIC3



Function

2-Input NAND

Truth Table

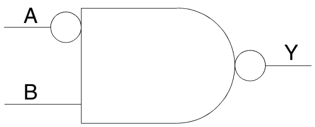
A	B	Y
X	0	1
0	X	1
1	1	0

Input	**Output**
A, B	Y

Family	Tiles
All	1

NAND2A

IGLOO, Fusion, ProASIC3



Function

2-Input NAND with active low A-Input

Truth Table

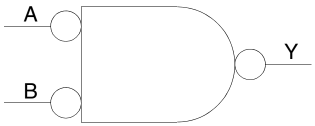
A	B	Y
X	0	1
0	1	0
1	X	1

Input	Output
A, B	Y

Family	Tiles
All	1

NAND2B

IGLOO, Fusion, ProASIC3



Function

2-Input NAND with active low Inputs

Truth Table

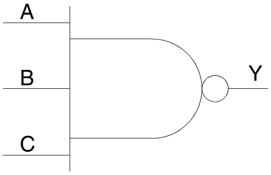
A	B	Y
0	0	0
X	1	1
1	X	1

Input	Output
A, B	Y

Family	Tiles
All	1

NAND3

IGLOO, Fusion, ProASIC3



Function

3-Input NAND

Truth Table

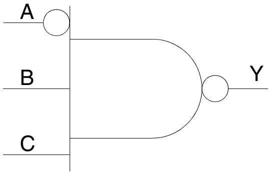
A	B	C	Y
X	X	0	1
X	0	X	1
0	X	X	1
1	1	1	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

NAND3A

IGLOO, Fusion, ProASIC3



Function

3-Input NAND with active low A-Input

Truth Table

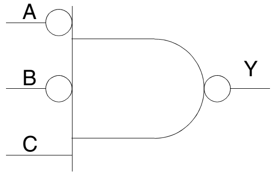
A	B	C	Y
X	X	0	1
X	0	X	1
0	1	1	0
1	X	X	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

NAND3B

IGLOO, Fusion, ProASIC3



Function

3-Input NAND with active low A- and B-Inputs

Truth Table

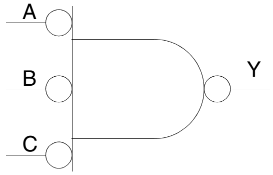
A	B	C	Y
X	X	0	1
0	0	1	0
X	1	X	1
1	X	X	1

Input	**Output**
A, B, C	Y

Family	Tiles
All	1

NAND3C

IGLOO, Fusion, ProASIC3



Function

3-Input NAND with active low Inputs

Truth Table

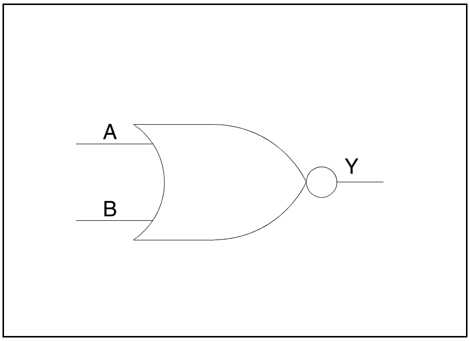
A	B	C	Y
0	0	0	0
X	X	1	1
X	1	X	1
1	X	X	1

Input	**Output**
A, B, C	Y

Family	Tiles
All	1

NOR2

IGLOO, Fusion, ProASIC3



Function

2-Input NOR

Truth Table

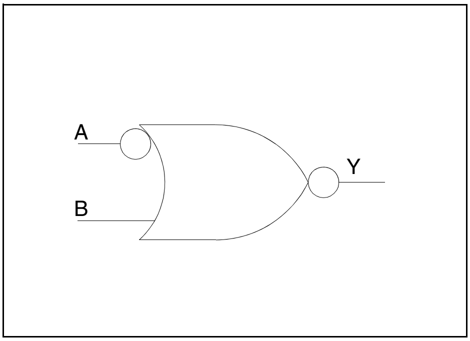
A	B	Y
0	0	1
X	1	0
1	X	0

Input	Output
A, B	Y

Family	Tiles
All	1

NOR2A

IGLOO, Fusion, ProASIC3



Function

2-Input NOR with active low A-Input

Truth Table

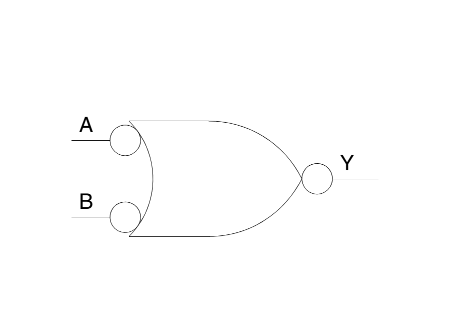
A	B	Y
0	X	0
1	0	1
X	1	0

Input	Output
A, B	Y

Family	Tiles
All	1

NOR2B

IGLOO, Fusion, ProASIC3



Function

2-Input NOR with active low Inputs

Truth Table

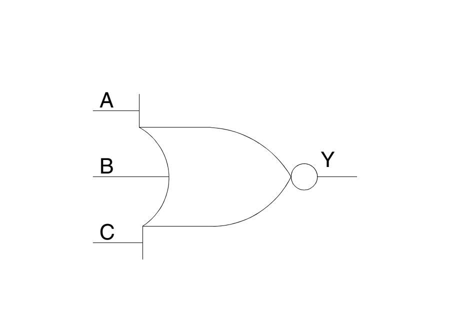
A	B	Y
X	0	0
0	X	0
1	1	1

Input	Output
A, B	Y

Family	Tiles
All	1

NOR3

IGLOO, Fusion, ProASIC3



Function

3-Input NOR

Truth Table

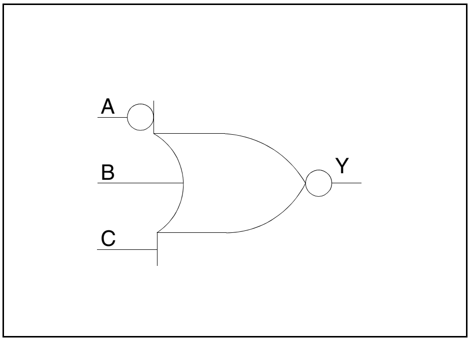
A	B	C	Y
0	0	0	1
X	X	1	0
X	1	X	0
1	X	X	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

NOR3A

IGLOO, Fusion, ProASIC3



Function

3-Input NOR with active low A-Input

Truth Table

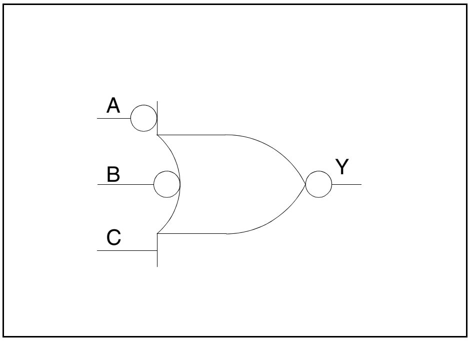
A	B	C	Y
0	X	X	0
1	0	0	1
X	X	1	0
X	1	X	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

NOR3B

IGLOO, Fusion, ProASIC3



Function

3-Input NOR with active low A- and B-Inputs

Truth Table

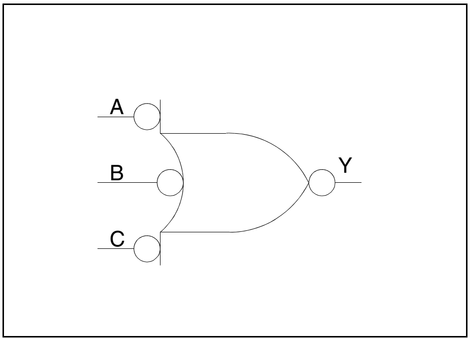
A	B	C	Y
X	0	X	0
0	X	X	0
1	1	0	1
X	X	1	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

NOR3C

IGLOO, Fusion, ProASIC3



Function

3-Input NOR with active low Inputs

Truth Table

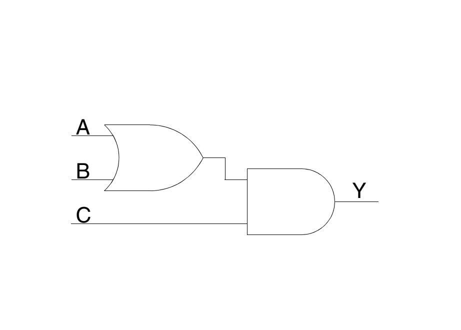
A	B	C	Y
X	X	0	0
X	0	X	0
0	X	X	0
1	1	1	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

OA1

IGLOO, Fusion, ProASIC3



Function

3 Input OR-AND

Truth Table

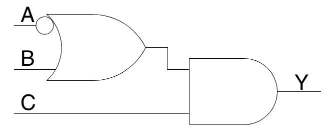
A	B	C	Y
X	X	0	0
0	0	X	0
X	1	1	1
1	X	1	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

OA1A

IGLOO, Fusion, ProASIC3



Function

3 Input OR-AND with active low A-Input

Truth Table

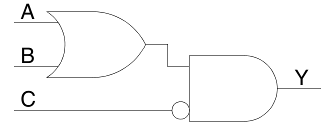
A	B	C	Y
X	X	0	0
0	X	1	1
1	0	X	0
X	1	1	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

OA1B

IGLOO, Fusion, ProASIC3



Function

3 Input OR-AND with active low C-Input

Truth Table

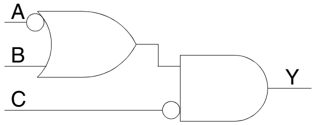
A	B	C	Y
0	0	X	0
X	1	0	1
X	X	1	0
1	X	0	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

OA1C

IGLOO, Fusion, ProASIC3



Function

3 Input OR-AND with active low A- and C-Inputs

Truth Table

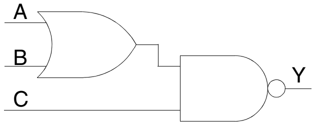
A	B	C	Y
0	X	0	1
X	X	1	0
1	0	X	0
X	1	0	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

OAI1

IGLOO, Fusion, ProASIC3



Function

3-Input OR-AND-INVERT

Truth Table

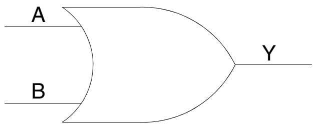
A	B	C	Y
X	X	0	1
0	0	X	1
X	1	1	0
1	X	1	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

OR2

IGLOO, Fusion, ProASIC3



Function

2-Input OR

Truth Table

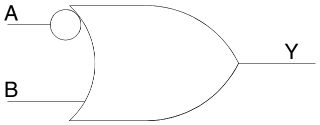
A	B	Y
0	0	0
X	1	1
1	X	1

Input	Output
A, B	Y

Family	Tiles
All	1

OR2A

IGLOO, Fusion, ProASIC3



Function

2-Input OR with active low A-Input

Truth Table

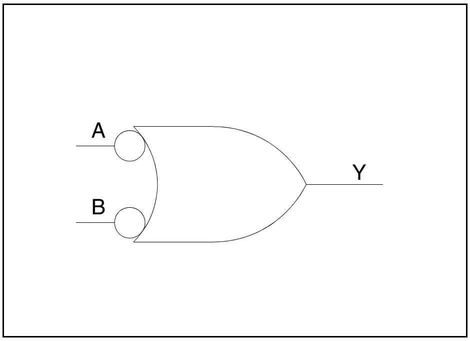
A	B	Y
0	X	1
1	0	0
X	1	1

Input	Output
A, B	Y

Family	Tiles
All	1

OR2B

IGLOO, Fusion, ProASIC3



Function

2-Input OR with active low Inputs

Truth Table

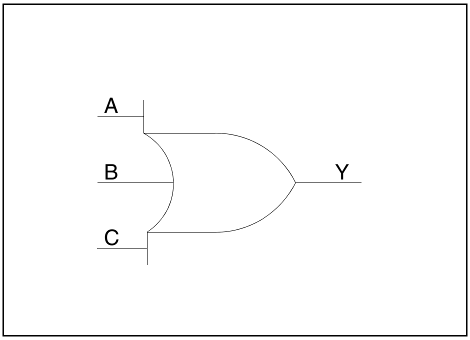
A	B	Y
X	0	1
0	X	1
1	1	0

Input	**Output**
A, B	Y

Family	Tiles
All	1

OR3

IGLOO, Fusion, ProASIC3



Function

3-Input OR

Truth Table

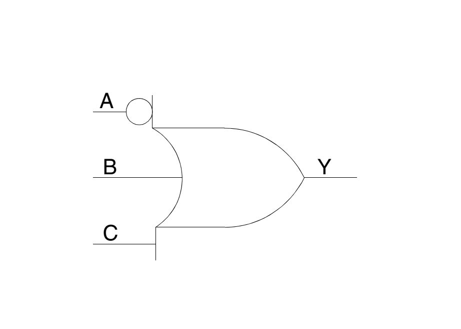
A	B	C	Y
0	0	0	0
X	X	1	1
X	1	X	1
1	X	X	1

Input	**Output**
A, B, C	Y

Family	Tiles
All	1

OR3A

IGLOO, Fusion, ProASIC3



Function

3-Input OR with active low A-Input

Truth Table

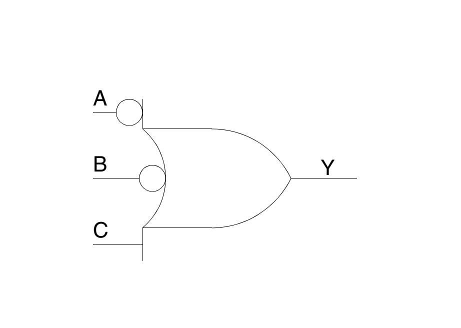
A	B	C	Y
0	X	X	1
1	0	0	0
X	X	1	1
X	1	X	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

OR3B

IGLOO, Fusion, ProASIC3



Function

3-Input OR with active low A- and B-Inputs

Truth Table

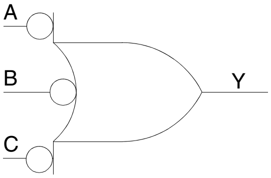
A	B	C	Y
X	0	X	1
0	X	X	1
1	1	0	0
X	X	1	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

OR3C

IGLOO, Fusion, ProASIC3



Function

3-Input OR with active low Inputs

Truth Table

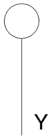
A	B	C	Y
X	X	0	1
X	0	X	1
0	X	X	1
1	1	1	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

VCC

IGLOO, Fusion, ProASIC3



Function

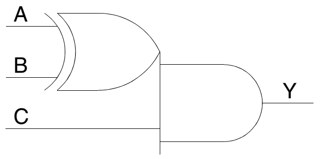
Power

Input	Output
	Y

NOTE: VCC does not use any modules.

XA1

IGLOO, Fusion, ProASIC3



Function

3-Input XOR-AND

Truth Table

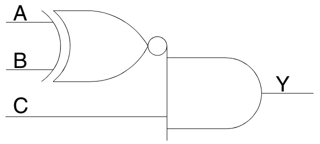
A	B	C	Y
X	X	0	0
0	0	X	0
0	1	1	1
1	0	1	1
1	1	X	0

Input	**Output**
A, B, C	Y

Family	Tiles
All	1

XA1A

IGLOO, Fusion, ProASIC3



Function

3-Input XNOR-AND

Truth Table

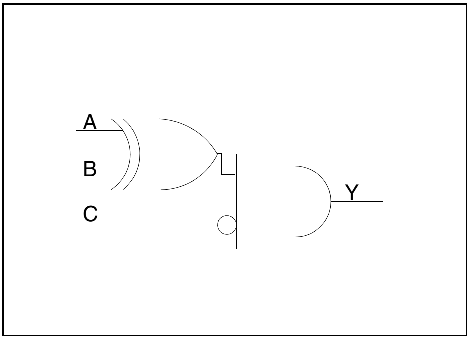
A	B	C	Y
X	X	0	0
0	0	1	1
0	1	X	0
1	0	X	0
1	1	1	1

Input	**Output**
A, B, C	Y

Family	Tiles
All	1

XA1B

IGLOO, Fusion, ProASIC3



Function

3-Input XNOR-AND with active low C-input

Truth Table

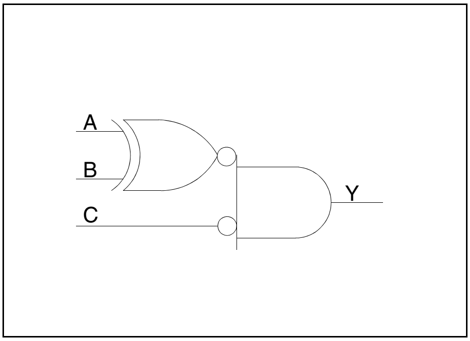
A	B	C	Y
X	X	1	0
0	0	X	0
1	0	0	1
0	1	0	1
1	1	X	0

Input	**Output**
A, B, C	Y

Family	Tiles
All	1

XA1C

IGLOO, Fusion, ProASIC3



Function

3-Input XNOR-AND with active low C-input

Truth Table

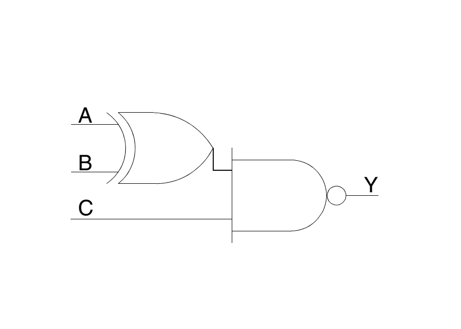
A	B	C	Y
X	X	1	0
0	0	0	1
1	0	X	0
0	1	X	0
1	1	0	1

Input	**Output**
A, B, C	Y

Family	Tiles
All	1

XAI1

IGLOO, Fusion, ProASIC3



Function

3-Input XNOR-NAND

Truth Table

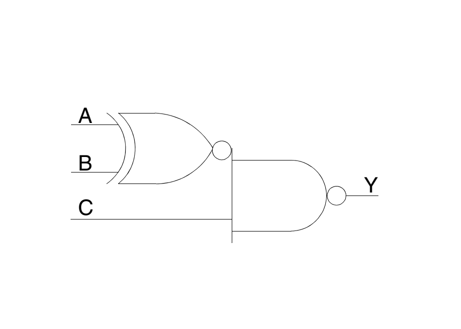
A	B	C	Y
X	X	0	1
0	0	X	1
1	0	1	0
0	1	1	0
1	1	X	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

XAI1A

IGLOO, Fusion, ProASIC3



Function

3-Input XNOR-NAND

Truth Table

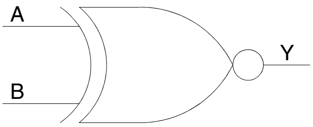
A	B	C	Y
X	X	0	1
0	0	1	0
1	0	X	1
0	1	X	1
1	1	1	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

XNOR2

IGLOO, Fusion, ProASIC3



Function

2- Input XNOR

Truth Table

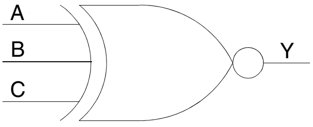
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

Input	**Output**
A, B	Y

Family	Tiles
All	1

XNOR3

IGLOO, Fusion, ProASIC3



Function

3-Input XNOR

Truth Table

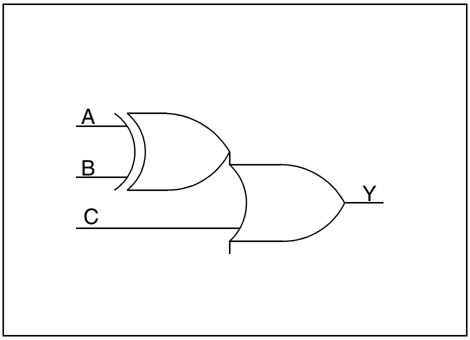
A	B	C	Y
0	0	0	1
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	0

Input	**Output**
A, B, C	Y

Family	Tiles
All	1

XO1

IGLOO, Fusion, ProASIC3



Function

3-Input XOR-OR

Truth Table

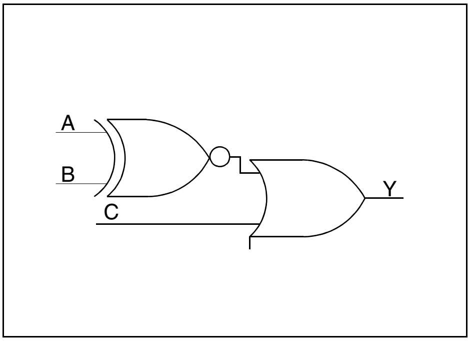
A	B	C	Y
0	0	0	0
X	X	1	1
0	1	X	1
1	0	X	1
1	1	0	0

Input	Output
A, B, C	Y

Family	Tiles
All	1

XO1A

IGLOO, Fusion, ProASIC3



Function

3-Input XNOR-OR

Truth Table

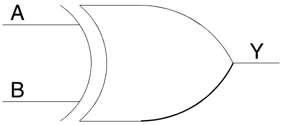
A	B	C	Y
0	0	0	1
X	X	1	1
0	1	0	0
1	0	0	0
1	1	0	1

Input	Output
A, B, C	Y

Family	Tiles
All	1

XOR2

IGLOO, Fusion, ProASIC3



Input	Output
A, B	Y

Function

2-Input XOR

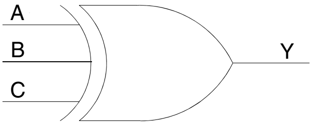
Truth Table

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Family	Tiles
All	1

XOR3

IGLOO, Fusion, ProASIC3



Input	Output
A, B, C	Y

Function

3-Input XOR

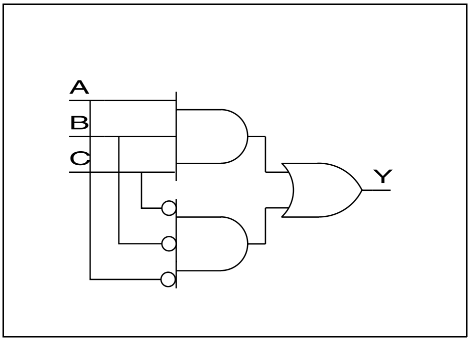
Truth Table

A	B	C	Y
0	0	0	0
1	0	0	1
0	1	0	1
1	1	0	0
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	1

Family	Tiles
All	1

ZOR3

IGLOO, Fusion, ProASIC3



Function

3-Input function

Input	Output
A, B, C	Y

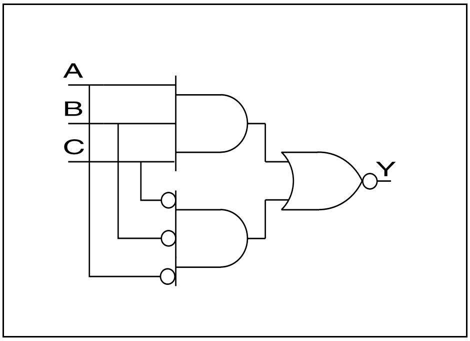
Truth Table

A	B	C	Y
0	0	0	1
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	1

Family	Tiles
All	1

ZOR3I

IGLOO, Fusion, ProASIC3



Function

3-Input function

Truth Table

A	B	C	Y
0	0	0	0
1	0	0	1
0	1	0	1
1	1	0	1
0	0	1	1
1	0	1	1
0	1	1	1
1	1	1	0

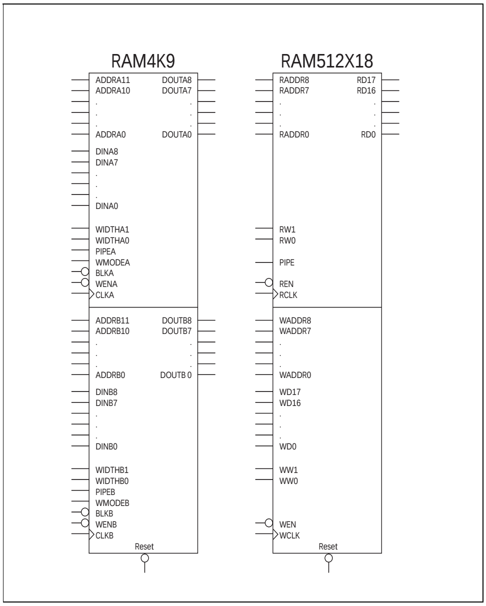
Input	Output
A, B, C	Y

Family	Tiles
All	1

RAM and FIFO Macros

RAM4K9 and RAM512X18

IGLOO, Fusion, ProASIC3



Function

RAM4K9 is a fully synchronous, true dual-port RAM with an optional pipeline stage; RAM512X18 is a fully synchronous, two-port RAM with an optional pipeline stage

Truth tables are listed below.

Input
Inputs are shown on the left of the diagrams. For example, ADDRA11, ADDRA10, ... , ADDRA0.

Output
Outputs are shown at the right on the diagrams. For example, DOUTA8, DOUT7, ... , DOUT0.

There are two RAM macros in the ProASIC3/ProASIC3E library: RAM4K9 and RAM512X18. The RAM4K9 is a fully synchronous, true dual-port RAM with an optional pipeline stage. It can be used for word widths up to 9 bits. Both ports are capable of reading and writing, making it possible to write with both ports or read with both ports simultaneously. You can also read from one port while writing to the other. Each port also has an optional pipeline stage that can be controlled separately via the PIPE pins. The RAM512X18 is a fully synchronous, two-port RAM with an optional pipeline stage. You can use it for word widths of 9 or 18 bits. It has one dedicated read port and one dedicated write port (you can read from one port while writing to the other). The read port also has an optional pipeline stage that you can control separately via the PIPE pin.

During the write operation of the RAM4K9, the WMODE pins control the data that appears on the read pins of the same port. When WMODE is high, the same data appears on the read and write ports at the rising CLK edge. When WMODE is low, the old data stored in the current memory location being addressed appears on the read port. There are no WMODE pins on the RAM512X18.

The aspect ratio of each port can be specified independently via the WIDTHA and WIDTHB pins. For the RAM512X18, the allowable values are 18 x 256 and 9 x 512. For the RAM4K9, the allowable values are 9 x 512, 4 x 1K, 2 x 2K, and 1 x 4K. Although it is possible to dynamically reconfigure the aspect ratios, the RAM was designed with only static configuration in mind, so the timing is unknown and you are discouraged from performing such operations. The same is true for the WMODE and PIPE configuration pins.

The RAM4K9 only needs 2 bits to configure the WIDTH. The allowable RAM4K9 WIDTHA and WIDTHB values are shown in the table below.

RAM4K9 WIDTHA and WIDTHB Values

WIDTHA1, WIDTHA0	WIDTHB1, WIDTHB0	W x D
00	00	1 x 4K
01	01	2 x 2K
10	10	4 x 1K
11	11	9 x 512

The RAM512X18 also needs 2 bits to configure the read and write widths. The allowable RAM512X18 WW and RW values are shown in the table below.

RAM512x18 WW and RW Values

WW1, WW0	RW1, RW0	W x D
01	01	9 x 512
10	10	18 x 256
00, 11	00, 11	Illegal

When specifying a width that is less than the maximum (e.g. 1), the upper unused data input pins (e.g. DINA8 - DINA1) must be connected to GND. When specifying a depth that is less than the maximum (e.g. 512), the upper unused address pins (e.g. ADDRA11 - ADDRA9) must also be connected to GND.

When widths of 1, 2, and 4 are used, the ninth bit is skipped. This can cause counter-intuitive effects when these widths are used for read operations and larger widths are used for write operations (or vice versa). For example, if a width of 9 is used for writing and a width of 1 for reading, every 9th bit will be dropped. This effect may be desirable for removing parity bits. If a write width of 4 and read width of 9 is used, the 9th bit may either contain garbage or remnants of previous write operations when a write width of 9 or higher was being used. For this reason, SmartGen only supports the following aspect ratio combinations when one of the ports is configured with a 1-, 2-, or 4-bit width using the RAM4K9.

SmartGen Supported Aspect Ratio Combinations for the RAM4K9

READ	WRITE
1 x 4K	1 x 4K
1 x 4K	2 x 2K
1 x 4K	4 x 1K
2 x 2K	1 x 4K
2 x 2K	2 x 2K
2 x 2K	4 x 1K
4 x 1K	1 x 4K
4 x 1K	2 x 2K
4 x 1K	4 x 1K

The RAM4K9 can still be used for 9-bit width applications, but no other bit-width can be used with it other than 9-bits.

SmartGen Supported Aspect Ratios for 9-bit Width Applications

READ	WRITE
9 x 512	9 x 512

There are several restrictions that apply when you use an 18 x 256 aspect ratio. For this reason, SmartGen uses the RAM512X18 whenever 18-bit widths are specified. The only allowable combinations of read and write configurations for the RAM512X18 are as follows:

RAM512X18 Read and Write Combinations

READ	WRITE
18 x 256	18 x 256
18 x 256	9 x 512
9 x 512	18 x 256

The RADDR pins are always used for the read address in the above configurations and the WADDR pins are used for the write address. The RW pin is used to specify the read width and the WW pin for the write width. The WD pins are used for writing data and the RD pins for reading data.

RAM4K9 Truth Table

Operation	Address	CLK	BLK	WMODE	WEN	RESET	DI	DO
Deselect	X	X	H	X	X	H	X	Data-Last
Reset	X	X	X	X	X	L	X	L
Read	ADDR	Rising Edge	L	L	H	H	X	Data
Write (0)	ADDR	Rising Edge	L	L	L	H	WData	Data-Last
Write (1)	ADDR	Rising Edge	L	H	L	H	WData	WData

When deserted, the BLK pins will cause the DO outputs to hold their last value. When asserted, the WEN pins can be used to switch each port between write and read mode. The RESET pin sets all outputs low but does not reset the memory. The WMODE pins are used to either allow the write data to appear immediately on the output pins or to hold the last value.

RAM512x18 Truth Table

Operation	Address	WCLK	REN	WEN	RESET	WD	RD
Reset	X	X	X	X	L	X	L
Read	RADDR	Rising Edge	L	X	H	X	Stored Data
Write	WADDR	Rising Edge	X	L	H	WData	Data-Last

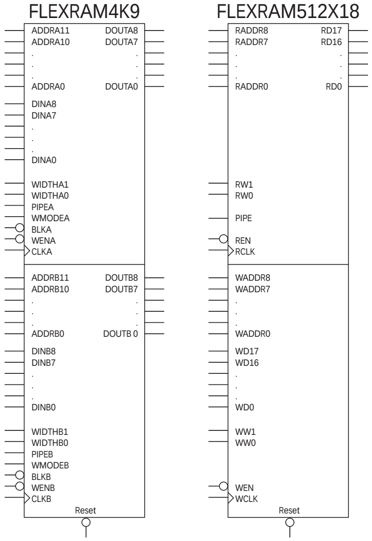
Use SmartGen to configure the RAM for typical use. SmartGen will not support dynamic reconfiguration or cascading width-wise. Customers who wish to use such features must instantiate and configure the RAM macro manually. You can configure your RAM dynamically if you use the FlexRAM macros in the next section.

Warnings

- Simultaneous write and read to same address is detected and if timing requirements are not met, read data is driven to X. Write operation is not affected.
- Simultaneous write to the same address from both ports is possible, but the results are undefined. Avoid writing to the same address simultaneously from both ports.
- Dynamic reconfiguration of any pins possible but not supported by SmartGen.
- Cascading is possible and limited only by the number of available RAM blocks in a row, which is device dependent. SmartGen prompts you for device type information in order to correctly calculate the maximum.
- RESET has priority over BLKA and BLKB.
- In read mode (i.e. when WEN high) WMODE is ignored.
- Dual-port operation not possible unless both ports have the same aspect ratio.

FLEXRAM4K9 and FLEXRAM512X18

Fusion



Function

FLEXRAM4K9 is a fully synchronous, true dual-port RAM with an optional pipeline stage; FLEXRAM512X18 is a fully synchronous, two-port RAM with an optional pipeline stage

Truth tables are listed below.

Input
Inputs are shown on the left of the diagrams. For example, ADDRA11, ADDRA10, ... , ADDRA0.

Output
Outputs are shown at the right on the diagrams. For example, DOUTA8, DOUT7, ... , DOUT0.

There are two dynamically reconfigurable RAM macros in the Fusion library: FLEXRAM4K9 and FLEXRAM512X18. The FLEXRAM4K9 is a fully synchronous, true dual-port RAM with an optional pipeline stage. It can be used for word widths up to 9 bits. Both ports are capable of reading and writing, making it possible to write with both ports or read with both ports simultaneously. You can also read from one port while writing to the other. Each port also has an optional pipeline stage that can be controlled separately via the PIPE pins. The FLEXRAM512X18 is a fully synchronous, two-port RAM with an optional pipeline stage. You can use it for word widths of 9 or 18 bits. It has one dedicated read port and one dedicated write port (you can read from one port while writing to the other). The read port also has an optional pipeline stage that you can control separately via the PIPE pin.

During the write operation of the FLEXRAM4K9, the WMODE pins control the data that appears on the read pins of the same port. When WMODE is high, the same data appears on the read and write ports at the rising CLK edge. When WMODE is low, the old data stored in the current memory location being addressed appears on the read port. There are no WMODE pins on the FLEXRAM512X18.

The aspect ratio of each port can be specified independently via the WIDTHA and WIDTHB pins. For the FLEXRAM512X18, the allowable values are 18 x 256 and 9 x 512. For the FLEXRAM4K9, the allowable values are 9 x 512, 4 x 1K, 2 x 2K, and 1 x 4K. Actel recommends that you do not change the WMODE and pipe configuration pins dynamically because the timing is unknown.

The FLEXRAM4K9 only needs 2 bits to configure the WIDTH. The allowable FLEXRAM4K9 WIDTHA and WIDTHB values are shown in the table below.

FLEXRAM4K9 WIDTHA and WIDTHB Values

WIDTHA1, WIDTHA0	WIDTHB1, WIDTHB0	W x D
00	00	1 x 4K
01	01	2 x 2K
10	10	4 x 1K
11	11	9 x 512

The FLEXRAM512X18 also needs 2 bits to configure the read and write widths. The allowable FLEXRAM512X18 WW and RW values are shown in the table below.

FLEXRAM512x18 WW and RW Values

WW1, WW0	RW1, RW0	W x D
01	01	9 x 512
10	10	18 x 256
00, 11	00, 11	Illegal

When specifying a width that is less than the maximum (e.g. 1), the upper unused data input pins (e.g. DINA8 - DINA1) must be connected to GND. When specifying a depth that is less than the maximum (e.g. 512), the lower unused address pins (e.g. ADDRA2 - ADDRA0) must also be connected to GND.

When widths of 1, 2, and 4 are used, the ninth bit is skipped. This can cause counter-intuitive effects when these widths are used for read operations and larger widths are used for write operations (or vice versa). For example, if a width of 9 is used for writing and a width of 1 for reading, every 9th bit will be dropped. This effect may be desirable for removing parity bits. If a write width of 4 and read width of 9 is used, the 9th bit may either contain garbage or remnants of previous write operations when a write width of 9 or higher was being used. For this reason, Actel recommends that you use only the following aspect ratio combinations when one of the ports is configured with a 1-, 2-, or 4-bit width using the FLEXRAM4K9.

Recommended Aspect Ratio Combinations for the FLEXRAM4K9

READ	WRITE
1 x 4K	1 x 4K
1 x 4K	2 x 2K
1 x 4K	4 x 1K
2 x 2K	1 x 4K
2 x 2K	2 x 2K
2 x 2K	4 x 1K
4 x 1K	1 x 4K
4 x 1K	2 x 2K
4 x 1K	4 x 1K

The FLEXRAM4K9 can still be used for 9-bit width applications, but no other bit-width can be reliably used with it other than 9-bits.

Recommended Aspect Ratios for 9-bit Width Applications

READ	WRITE
9 x 512	9 x 512

There are several restrictions that apply when you use an 18 x 256 aspect ratio. For this reason, Actel recommends that you use the FLEXRAM512X18 whenever 18-bit widths are specified. The only allowable combinations of read and write configurations for the FLEXRAM512X18 are as follows:

FLEXRAM512X18 Read and Write Combinations

READ	WRITE
18 x 256	18 x 256
18 x 256	9 x 512
9 x 512	18 x 256

The RADDR pins are always used for the read address in the above configurations and the WADDR pins are used for the write address. The RW pin is used to specify the read width and the WW pin for the write width. The WD pins are used for writing data and the RD pins for reading data.

FLEXRAM4K9 Truth Table

Operation	Address	CLK	BLK	WMODE	WEN	RESET	DI	DO
Deselect	X	X	H	X	X	H	X	Data-Last
Reset	X	X	X	X	X	L	X	L
Read	ADDR	Rising Edge	L	L	H	H	X	Data
Write (0)	ADDR	Rising Edge	L	L	L	H	WData	Data-Last
Write (1)	ADDR	Rising Edge	L	H	L	H	WData	WData

When deserted, the BLK pins will cause the DO outputs to hold their last value. When asserted, the WEN pins can be used to switch each port between write and read mode. The RESET pin sets all outputs low but does not reset the memory. The WMODE pins are used to either allow the write data to appear immediately on the output pins or to hold the last value.

FLEXRAM512x18 Truth Table

Operation	Address	WCLK	REN	WEN	RESET	WD	RD
Reset	X	X	X	X	L	X	L
Read	RADDR	Rising Edge	L	X	H	X	Stored Data
Write	WADDR	Rising Edge	X	L	H	WData	Data-Last

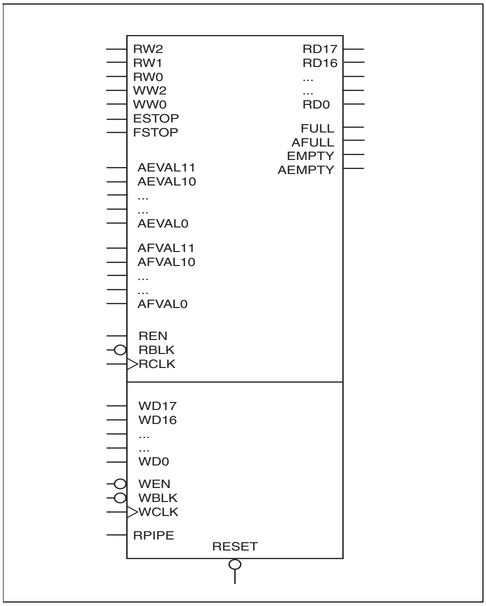
SmartGen does not support FlexRAM macros; you must instantiate and configure the FlexRAM macros manually.

Warnings

- Simultaneous write and read to same address is detected and if timing requirements are not met, read data is driven to X. Write operation is not affected.
- Simultaneous write to the same address from both ports is possible, but the results are undefined. Avoid writing to the same address simultaneously from both ports.
- RESET has priority over BLKA and BLKB.
- In read mode (i.e. when WEN high) WMODE is ignored.
- Dual-port operation not possible unless both ports have the same aspect ratio.

FIFO4K18

IGLOO, Fusion, ProASIC3



Function

FIFO4K18 is fully synchronous and has its own built-in controller, capable of variable aspect ratios.

Truth tables are listed below.

Input
Inputs are shown on the left of the diagram. For example, AEVAL11, AEVAL10, ... , AEVAL0.

Output
Outputs are shown at the right on the diagram. For example, RD17, RD16, ... , RD0.

FIFO4K18 is fully synchronous and has its own built-in controller. Like the RAM, the FIFO can have different write and read aspect ratios that can be configured dynamically. The WW and RW pins are used to specify one of five allowable aspect ratios, as shown below.

FIFO4K18 Aspect Ratios

WW2, WW1, WW0 and RW2, RW1, RW0	W x H
000	1 x 4K
001	2 x 2K
010	4 x 1K
011	9 x 512
100	18 x 256
101, 110, 111	Illegal

The AEVAL and AFVAL pins are used to specify the almost empty and almost full threshold values, respectively. In order to handle different read and write aspect ratios, the values specified by the AEVAL and AFVAL pins are to be interpreted as the address of the last word stored in the FIFO. The FIFO actually contains separate write address (WADDR) and read address (RADDR) counters. These counters calculate the 12-bit memory address that is a function of WW and RW, respectively. WADDR is incremented every time a write operation is performed and RADDR is incremented every time a read operation is performed. Whenever the difference between WADDR and RADDR is greater than or equal to AFVAL, the AFULL output is raised. Likewise, whenever the difference between WADDR and RADDR is less than or equal to AEVAL, the AEMPTY output is raised. Therefore AEVAL and AFVAL must be left-justified for widths greater than one (i.e. unused lsb's must be grounded).

Aspect Ratio and Related Bits to Ground

Aspect ratio	AEVAL/AFVAL bits to ground
1 x 4K	none
2 x 2K	0
4 x 1K	1:0
9 x 512	2:0
18 x 256	3:0

When the number of words stored in the FIFO reaches the amount specified by AEVAL while reading, the AEMPTY output will go high. Likewise when the number of words stored in the FIFO reaches the amount specified by AFVAL while writing, the AFULL output will go high. The FULL and EMPTY outputs will go high when the FIFO is completely full or empty, respectively.

It should be noted that the internal memory size is 512 X 9. When widths of 1, 2, and 4 are specified, the 9th bit is skipped.

The ESTOP pin is used to stop the read counter from counting any further once the FIFO is empty (i.e. the EMPTY flag goes high). Likewise, the FSTOP pin is used to stop the write counter from counting any further once the FIFO is full (i.e. the FULL flag goes high). These are configuration pins that should not be dynamically reconfigured. SmartGen treats them as static configuration pins and always ties them high.

Independent read and write operations are allowed, however only the read port can be pipelined. Data on the appropriate WD pins are written to the FIFO every rising WCLK edge as long as WEN and WBLK are low. Data is read from the FIFO and output on the appropriate RD pins every rising RCLK edge as long as REN is high and RBLK is low.

The active low RESET pin is used to asynchronously clear the outputs of the FIFO and reset the internal read and write address counters. It sets all the RD pins low, the FULL and AFULL pins low, and the EMPTY and AEMPTY pins high, however the contents of the memory remain unchanged. RESET has priority over RBLK and WBLK.

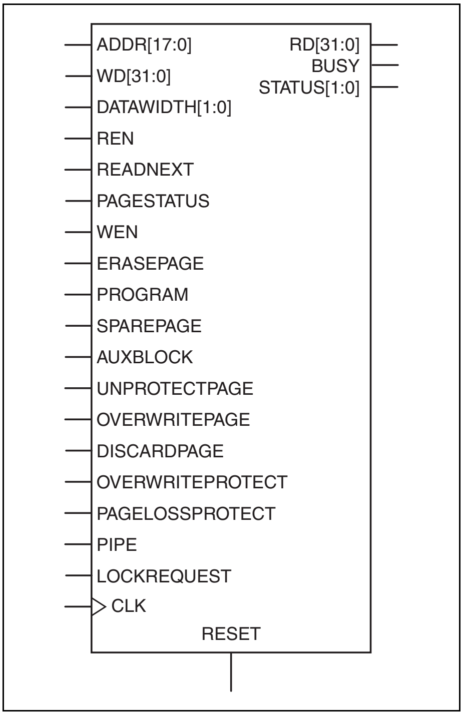
When instantiating the FIFO4K18, all unused input pins must be connected to GND.

Warnings

- The WW, RW, AEVAL, and AFVAL pins can be dynamically configured, but only static configuration will be supported by SmartGen.
- The RPIPE signal can be dynamically configured, but only static configuration will be supported by SmartGen.
- No pipeline on the write port.
- Cascading allowed and supported in the width direction only by SmartGen. Cascading in the depth direction requires the use of a soft controller (i.e. implemented with core logic).
- ESTOP and FSTOP applications not clear. The effect of activating ESTOP is to allow the read pointer to wrap around, allowing the memory contents to be read over and over again with rewriting after EMPTY. The effect of activating FSTOP is not clear, however, since the write pointer could wrap around allowing overwriting of data which is never read. Therefore SmartGen will always tie these pins off high.

Flash Memory Block Macro

Flash Memory Block



Function

Flash memory block builder for use with SmartGen and Fusion

Inputs / Outputs

See the description below for an explanation of the inputs and outputs available on the Flash memory block macro.

Each Flash Memory block holds 256 kb of data. Although it is functionally similar to a large single-port synchronous RAM, it has several significant differences, including:

1. Address bits are MSB justified, unlike RAM4K9 and RAM512X18 in which the address bits are LSB justified.
2. Write operation updates write data into the block buffer ONLY. To store data permanently into the Flash Memory Block array writes to a page must be followed by a program operation of the same page.
3. The simulation models always execute copy page from Flash memory block array (internal operation) in 65 clock cycles; in silicon the behavior is non-deterministic (63-67 clock cycles). This mismatch is reflected in the number of cycles BUSY is asserted.

Operations on Flash memory block are synchronous to rising-edge of CLK.

Flash Memory Block Pin Description

All Flash memory block signals are active high, except for RESET which is active low. The Flash memory block is a completely synchronous model sensitive to rising edge of CLK input.

NAME	FUNCTION
ADDR[17:0]	Byte-offset into the Flash memory block array or block buffer of page buffer
WD[31:0]	Write data
DATAWIDTH[31:0]	00 = 1-byte in data_in/out[7:0] 01 = 2-bytes in data_in/out[15:0] 10/11 = 4-bytes in data_in/out[31:0]
REN	When asserted, initiates a read operation
READNEXT	When asserted with REN, initiates a read from next address after read to current address is complete.
PAGESTATUS	When asserted with read, initiates a read page status operation
WEN	When asserted, interface data is stored into the assembly buffer.
ERASEPAGE	When asserted, erase addressed page (program all zeroes).
PROGRAM	When asserted, write the contents of the assembly buffer into the cell array page addressed.
SPAREPAGE	When asserted, the sector addressed is used to access the spare page within that sector.
AUXBLOCK	When asserted, the page addressed is used to access the auxiliary block within that page.
UNPROTECTPAGE	When asserted, the page addressed is copied into the AB and the AB made writeable.
OVERWRITEPAGE	When asserted, the page addressed is overwritten with the contents of the AB if the page is writeable.
DISCARDPAGE	When asserted, the contents of the AB are discarded so that a new page write can be started.
OVERWRITEPROTECT	When asserted, all program operations will set the overwrite protect bit in the auxiliary block of the page being programmed.
PAGELOSSPROTECT	When asserted, a modified assembly buffer must be programmed or discarded before accessing a new page.
PIPE	When asserted with REN, read operation completes in 6 cycles. Required to be asserted for CLK speeds above 50MHz.
LOCKREQUEST	Request to lock user access to Flash memory block array.
CLK	Input clock. All operations and status are synchronous to rising-edge of this clock.
RESET	When asserted resets the state of the Flash memory block.
RD[31:0]	Read data to be sampled when BUSY=0.
STATUS[1:0]	Status of the last operation completed: 00 = successful completion 01 = Read: single error detected and corrected Write: operation addressed a write-protected page Erase-Page/Program: AB is unmodified 10 = Read: two or more errors detected Erase-Page/Program: Compare operation failed 11 = Write: attempt to write to another page before Programming current page.

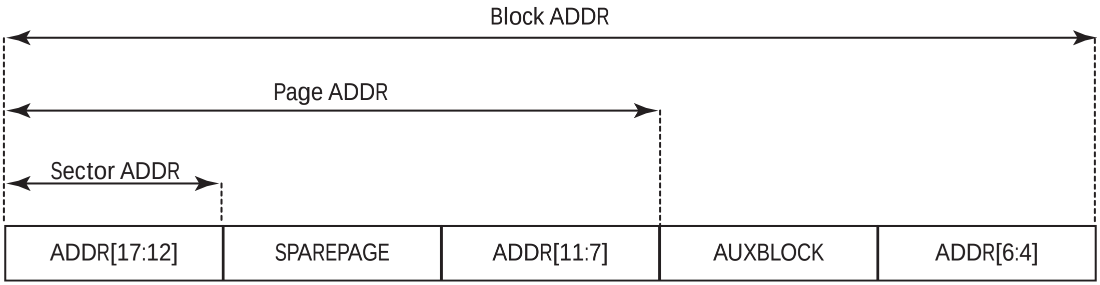
Functional Description

The Fusion datasheet, available here, http://www.actel.com/techdocs/ds/default.aspx, contains a detailed functional description of the entire Flash Memory Block.

Simulation Details for Flash Memory Block

The Flash memory block array can be pre-loaded with user-defined data. For simulation purposes, you can specify a memory initialization file by over-riding the parameter "MEMORYFILE" in the Verilog netlist and the generic "MEMORYFILE" in Vital netlist.

The memory array declared in simulation models stores data that is one block wide. It is 64k x 140 bits. The addressing scheme for accessing this array consists of 16 bits, as shown in the figure below.



Addressing Scheme for Accessing the Flash Memory Block Memory Array

ADDR[17:0] is the Flash memory block interface address, SPAREPAGE and AUXBLOCK are input signals.

The memory file for pre-loading an Flash memory block array consists of "*strings of address and data in hexadecimal notation with address delimiters (@)*" and MUST conform to the rules shown below:

1. Each line MUST contain a string of fixed length (=35 characters) and start with an "@" if it corresponds to an address.
2. Each line following the address line corresponds to a block of data starting at the block address specified in the address line. This applies until the next line with an address specifier (@) is encountered.
3. Each data block consists of 35 hex chars. Hex[31:0] are the data characters corresponding to 16 bytes of user data with Hex[1:0] corresponding to Byte0 and Hex[31:30] corresponding to Byte15. Hex[34:32] are ECC related bits and must be addressed manually.

Based on these rules, the format looks like:

```
@Block_Address_0
Block_Data_0 ( required )
Block_Data_1 ( optional )
Block_Data_2 ( optional )
...
...
Block_Data_8  ( Aux block data for this page,  optional )
@Block_Address_n
Block_Data_n      ( required )
Block_Data_n+1 ( optional )
Block_Data_n+2 ( optional )
...
...
...
Block_Data_n+8 ( Aux block data for this page, optional )
```

A typical memory file looks like:

```
@000…0000  // beginning with @, start address in hex. format. 0s to be padded
// between @ and hex address, to get a string of length 35.
ab101fd01…  // 35 hexadecimal characters corresponding to each block of Flash memory
block cell
eab9c4……
@000…4030  // start address for next data stream
```

```
c805489e….  // 35 hexadecimal characters corresponding to each block of Flash memory
block cell
96986391…
```

User Controlled Generics

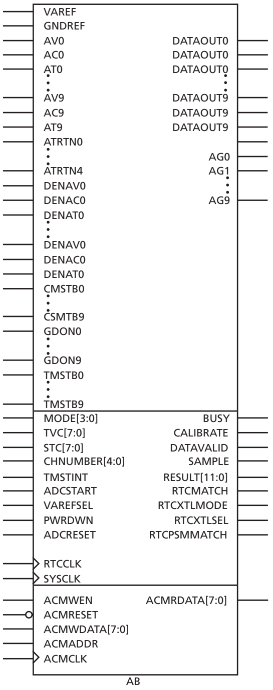
FAST_SIM - The generic/parameter FAST_SIM is included in pre-synthesis and pre-layout simulation models to reduce cycles wasted in executing the PROGRAM operation. The default is '1', which means the PROGRAM operation is executed with a 4 µs simulated delay. You can chose to deactivate the operation by overriding FAST_SIM to 0, in which case PROGRAM is executed with a delay close to real time of 8.4 ms. You can also choose this mode for post-layout simulations.

WR_THR - When the number of writes to a page in the Flash memory block array (program operation) exceeds the write threshold specified in the data sheet, the status returned is non-zero. Since the threshold is a huge value, WR_THR is provided to simulate this failure at a reduced number of writes. You can override the generic with any non-zero count (such as 10 or 12). The write threshold exceeded condition is produced on the 10th write to the same page in the Flash memory block array.

Note: The string assigned to the generic/parameter MEMORYFILE is preserved through synthesis and place-and-route. But for FAST_SIM and WR_THR generics/parameters, you must reassign the desired value each time after synthesis and place-and-route.

Analog System Builder Macro

Analog System Builder



Function

Analog system builder for use with SmartGen and Fusion. See the Fusion datasheet for a thorough description of the Analog System Builder.

Inputs / Outputs

Inputs are listed on the left, outputs on the right.

See the description below for an explanation of the inputs and outputs available on the Analog System Builder. For a complete description of the features in the ASB, see the Fusion datasheet.

Analog System Builder Pin Description

Analog System Builder Pin Description

Signal Name	Number of bits	Direction	Function
VAREF	1	Inout	External voltage ref.; used as either input or output, depending on VREFSEL
GNDREF	1	Input	External ground ref.
MODE[3:0]	4	Input	ADC operating mode
SYSCLK	1	Input	External system clock
TVC[7:0]	8	Input	Clock divide control
STC[7:0]	8	Input	Sample time control
CHNUMBER[4:0]	5	Input	Analog input channel select
ADCSTART	1	Input	Start of conversion
PWRDWN	1	Input	Comparator power-down if 1
ADCRESET	1	Input	ADC initialize if 1
BUSY	1	Output	1 – Running conversion
CALIBRATE	1	Output	1 – Power-up calibration
DATAVALID	1	Output	1 – Valid conversion result
RESULT[11:0]	12	Output	Conversion result - MSB justified
TMSTBINT	1	Input	Internal temp. monitor strobe
SAMPLE	1	Output	1 – Analog input is sampled
CMSTB0 to CMSTB9	10	Input	Current monitor strobe – 1 per quad, active high
GDON0 to GDON9	10	Input	Control to power MOS – 1 per quad
TMSTB0 to TMSTB9	10	Input	Temperature monitor strobe – 1 per quad; active high
DAVOUT0, DACOUT0, DATOUT0 to DAVOUT9, DACOUT9, DATOUT9	30	Output	Digital outputs – 3 per quad
DENAV0, DENAC0, DENAT0 to DENAV9, DENAC9, DENAT9	30	Input	Digital input enables – 3 per quad
ACMCLK	1	Input	ACM clock
ACMWEN	1	Input	ACM write enable – active high
ACMRESET	1	Input	ACM reset – active low
ACMWDATA[7:0]	8	Input	ACM write data
ACMRDATA[7:0]	8	Output	ACM read data
ACMADDR[7:0]	8	Input	ACM address
VAREFSEL	1	Input	0 = Output internal voltage reference (2.56 V) to VAREF 1 = Input external voltage reference from VAREF and GNDREF
AV0	1	Input	Analog Quad 0

Analog System Builder Pin Description (Continued)

Signal Name	Number of bits	Direction	Function
AC0	1	Input	
AG0	1	Output	
AT0	1	Input	
ATRTN0	1	Input	Temperature monitor return shared by Analog Quads 0 and 1
AV1	1	Input	Analog Quad 1
AC1	1	Input	
AG1	1	Output	
AT1	1	Input	
AV2	1	Input	Analog Quad 2
AC2	1	Input	
AG2	1	Output	
AT2	1	Input	
ATRTN1	1	Input	Temperature monitor return shared by Analog Quads 2 and 3
AV3	1	Input	Analog Quad 3
AC3	1	Input	
AG3	1	Output	
AT3	1	Input	
AV4	1	Input	Analog Quad 4
AC4	1	Input	
AG4	1	Output	
AT4	1	Input	
ATRTN2	1	Input	Temperature monitor return shared by Analog Quads 4 and 5
AV5	1	Input	Analog Quad 5
AC5	1	Input	
AG5	1	Output	
AT5	1	Input	
AV6	1	Input	Analog Quad 6
AC6	1	Input	
AG6	1	Output	
AT6	1	Input	
ATRTN3	1	Input	Temperature monitor return shared by Analog Quads 6 and 7
AV7	1	Input	Analog Quad 7
AC7	1	Input	
AG7	1	Output	

Analog System Builder Pin Description (Continued)

Signal Name	Number of bits	Direction	Function
AT7	1	Input	
AV8	1	Input	Analog Quad 8
AC8	1	Input	
AG8	1	Output	
AT8	1	Input	
ATRTN4	1	Input	Temperature monitor return shared by Analog Quads 8 and 9
AV9	1	Input	Analog Quad 9
AC9	1	Input	
AG9	1	Output	
AT9	1	Input	
RTCMATCH	1	Output	MATCH
RTCPSMMATCH	1	Output	MATCH connected to VRPSM
RTCXTLMODE[1:0]	2	Output	Drives XTLOSC RTCMODE[1:0] pins
RTCXTLSEL	1	Output	Drives XTLOSC MODESEL pin
RTCCLK	1	Input	RTC clock input

Functional Description

The Fusion datasheet, available at http://www.actel.com/techdocs/ds/default.aspx, contains a detailed functional description of the entire Analog System Builder.

Connecting Analog Ports

Each analog port must be connected to one of the following "virtual pads": INBUF_A. INBUF_DA or OUTBUF_A. AV0, AC0, AT0, …, AV9, AC9 and AT9 are analog inputs that can be used either as analog or digital inputs. When used as an analog input, the analog input signal (e.g. AV0) must be connected to an INBUF_A, and the corresponding digital input enable (e.g. DENAV0) must be tied to 0.

When used as a digital input, the analog input must be connected to an INBUF_DA, and the corresponding digital input enable must be tied to 1.

All other analog inputs (ATRETURN01, ATRETURN23, ATRETURN45, ATRETURN67, and ATRETURN89) must be connected to an INBUF_A.

Note: ATRETURN01, ATRETURN23, ATRETURN45, ATRETURN67, and ATRETURN89 must be connected to an INBUF_A, even though they have no function in the simulation model.

Analog outputs (AG0, …, AG9) must be connected to an OUTBUF_A instance.

VAREF is an inout pad and does not need to be connected to INBUF_A or OUTBUF_A.

Serialization

The analog ports are represented by a 1-bit wide port in both the Verilog and VHDL simulation models. Verilog modules and VHDL functions were developed to drive a real value through a 1-bit port and to read an analog value from a 1-bit port. The Analog System Builder macro contains embedded read and drive logic to read from the analog input and drive the analog output, respectively.

The drive module/function converts a real value into a 64-bit value, serializes it and streams it in zero simulation time, using delta delays. The read module/function deserializes a 64-bit stream into a 64-bit value and converts it into a real value.

Connecting Analog Ports with Verilog

Two Verilog modules (drive_analog_io and read_analog_io) are available to drive an analog input and read an analog output. You must instantiate a drive_analog_io for each analog input and a read_analog_io for each analog output. The read_analog_io starts as soon as there is a non 'Z' data bit on the module input pin. All read and drive operations happen in zero time (delta delays). drive_analog_input is a new module with the same functionality as drive_analog_io and can be used interchangeably to drive analog inputs.

Example: drive_analog_io with an INBUF_A and OUTBUF_A instantiation

```
wire    AV0_stream_pad, AV0_stream_y, AG0_d, AG0_pad;
wire [63:0] AG0_VECTOR;
real    AV0_real;
drive_analog_io drive_AV0 ( $realtobits(AV0_real), AV0_stream_pad );
INBUF_A inbuf_AV0 ( .Y(AV0_stream_y), .PAD (AV0_stream_pad) );

AB ab_inst (
    …
    .AV0 (AV0_stream_y),
    …
    .AG0 (AG0_d),
    …
    );
OUTBUF_A outbuf_AG0 ( .PAD(AG0_pad), . D (AG0_d) );
initial
begin
AV0_real <= 1.28;
end
```

Connecting Analog Ports with VHDL

Similarly, two VHDL functions (drive_analog_input and read_analog_input) are available to drive an analog input and read an analog output. These functions are part of the analog_io VHDL package. The read_analog_input starts as soon as there is a non 'Z' data bit on the function input pin. All read and drive operations happen in zero time (delta delays).

Example: drive_analog_input and read_analog_input with an INBUF_A instantiation

```
use work.analog_io.all;
…
signal varef_real : real
signal AV0_real : real
signal varef_serial_out : std_logic;
signal AV0_stream_pad : std_logic;
signal AV0_stream_y : std_logic;
component INBUF_A
    port(
    …
    );
end component;
component AB
    port(
    …
    );
```

```
end component;
read_varef : process
begin
  wait until varef_serial_out /= 'Z';
  read_analog_input( varef_serial_out, varef_real);
end process read_varef;
-- concurrent procedural call
drive_analog_input( AV0_real, AV0_stream_pad );
drive_quads : process
begin
  AV0_real <= 2.18;
end process drive_quads;
-- INBUF_A instance
inbuf_at0 : INBUF_A
port map (
  PAD => AV0_stream_pad,
  Y   => AV0_stream_y,
  …
);

-- AB instance
ab_inst: AB
port map (
  VAREF => varef_serial_out,
  …
  AV0 => AV0_stream_y,
  …
);
```

drive_temperature_quad and drive_current_monitor

In addition to the standard Verilog drive_analog_io modules and VHDL drive_analog_input procedures used to drive the analog quads, two Verilog modules and VHDL procedures are also available to drive the AC and AT quads.

The first Verilog module / VHDL procedure, drive_temperature_quad, takes a temperature in Celsius, converts it into a voltage, and drives it over the digital input. This can be used regardless of which input is selected by the Analog MUX - T-prescaler, temperature monitor or T-direct analog input. The equation is:

AT(V) = (Temperature(C) + 273.15) * (2.30258 * 0.000087248)

The second Verilog module / VHDL procedure, drive_current_monitor, can only be used when driving an AC

quad that will be used for Current Monitoring. As an input it takes the corresponding AV quad voltage (e.g. AV3's voltage, if driving AC3), the Resistor (in Ohm) and Current (in A) values, to calculate the voltage on the AC quad (see the Fusion datasheet for more information). The resistor must be between the 0.01 and 10 Ohm, and (AV - AC) has to be less than 250 mV. The equation is:

AC(V) = AV(V) - (Resistor(Ohm) * Current(A))

If the analog MUX selects the C-prescaler or the C-direct input, then use the standard VHDL procedure drive_analog_input or Verilog drive_analog_io modules to drive the AC quad.

drive_temperature_quad and drive_current_monitor Verilog Example

```
real voltage0, resistor0, current0, temperature0;
wire AV0_i, AC0_i, AT0_i;
wire AV0_o, AC0_o, AT0_o;

drive_analog_io drive_AV0 ( $realtobits(voltage0), AV0_i );
drive_current_monitor drive_CM_AC0 ( $realtobits(voltage0), $realtobits(resistor0),
$realtobits(current0), AC0_ i );
drive_temperature_quad drive_AT0 ( $realtobits(temperature0), AT0_ i );

INBUF_A inbuf_a_voltage0 ( .Y(AV0_o), .PAD (AV0_i) );
INBUF_A inbuf_a_current0 ( .Y(AC0_ o), .PAD (AC0_ i) );
INBUF_A inbuf_a_temperature0 ( .Y(AT0_ o), .PAD (AT0_ i) );

AB ab_inst (
…
.AV0 (AV0_ o),
.AC0 (AC0_ o),
.AT0 (AT0_ o),
…
);

initial
begin
  voltage0     <= 1.00032;
  resistor0    <= 1.0;
  current0     <= 1.031;
  temperature0 <= -70.0;
…
End
```

drive_temperature_quad and drive_current_monitor VHDL Example

```
component INBUF_A
port(
PAD : in STD_ULOGIC;
Y : out STD_ULOGIC);
end component;
component AB
port(
VAREF : INOUT STD_LOGIC ;
GNDREF : IN STD_LOGIC ;
AV0 : IN STD_LOGIC ;
…
RTCMATCH : OUT STD_LOGIC ;
ACMRDATA : OUT STD_LOGIC_VECTOR(7 DOWNTO 0)
) ;
end component;
signal voltage0_pad, current0_pad, temperature0_pad, voltage0_y, current0_y,
temperature0_y, : std_logic;
signal voltage0, current0, resistor0, temperature0 : real;

-- concurrent procedure calls for driving voltage0_pad, current0_pad, temperature0_pad
drive_analog_input( voltage0, voltage0_pad );
drive_current_monitor ( voltage0, resistor0, current0, current0_pad );
```

```
drive_temperature_quad(temperature0, temperature0_pad );

drive_aq : process
begin
voltage0 <= 1.00032;
resistor0 <= 1.0;
current0 <= 1.031;
temperature0 <= -70.0;
…
end process drive_aq;

inbuf_a_voltage0 : INBUF_A
port map (
PAD => voltage0_pad,
Y => voltage0_y
);

inbuf_a_current0 : INBUF_A
port map (
PAD => current0_pad,
Y => current0_y
);

inbuf_a_temperature0 : INBUF_A
port map (
PAD => ato_pad,
Y => temperature0_y
);

top: AB
port map (
…
AV0 => voltage0_y,
AC0 => current0_y,
AT0 => temperature0_y,
…
);
```

drive_current_inputs

This module/procedure has been added to both Verilog and VHDL libraries to drive both the pads of an AV/AC pair when configured as a current monitor. Voltage for AV, resistor and current are the inputs, serialized AV and AC are the outputs. You can use this module/procedure to replace a pair of drive_analog_input and drive_current_monitor instances, as shown in the example below.

drive_current_inputs Verilog Example

Two drivers for driving inputs to voltage and current pads:

```
drive_analog_io drive_AV0 ( $realtobits(voltage0), AV0_i );
drive_current_monitor drive_CM_AC0 ( $realtobits(voltage0), $realtobits(resistor0),
$realtobits(current0), AC0_i );
```

drive_current_inputs can drive inputs to both AV and AC pads. For example:

```
drive_current_inputs drive_AV0_AC0 ( $realtobits(voltage0), $realtobits(resistor0),
$realtobits(current0), AV0_i, AC0_i );
```

drive_current_inputs VHDL Example

Two drivers for driving inputs to voltage and current pads:

```
drive_analog_io( voltage0, voltage0_pad );
drive_current_monitor ( voltage0, resistor0, current0, current0_pad );
```

drive_current_inputs can drive inputs to both AV and AC pads. For example:

```
drive_current_inputs ( voltage0, resistor0, current0, voltage0_pad, current0_pad );
```

drive_differential_inputs

This module/procedure has been added to both Verilog and VHDL libraries to drive both the pads of an AV/AC pair when configured for differential inputs. Voltage for AV, AV-AC delta are the inputs, serialized AV and AC are the outputs.

drive_varef_out

Use this module to drive external VAREF onto VAREF inout pin. Since the applied voltage is serialized, you must ensure that the receiving logic in AB model is ready to receive the serial stream. Actel recommends that you wait until the RESET signal is deasserted (inactive) before assigning the external voltage to VAREF. This module is available in Verilog only. Use drive_analog_input in VHDL to drive external VAREF.

For example, driving VAREF externally by instantiating drive_varef_out module/procedure:

drive_varef_out Verilog Example

```
drive_varef_out   i_VAREF (
          .parallel_in($realtobits(val_varef)),
          .en_out(!SYS_RESET),
          .serial_out(VAREF)
     );

     initial
     begin
         wait(!SYS_RESET);
          val_varef = 3.00;
     end
```

drive_varef_out VHDL example

```
…
  use work.analog_io.all;
  …
  signal varef_real : real
  signal VAREF : std_logic;
component AB
  port(
  …
  );
-- concurrent procedure call
drive_analog_input( varef_real, varef_pad );

drive_quads : process
  begin
    wait until ( SYS_RESET = '1' );   -- active low reset
    varef_real <= 3.00;
    ...
    end process drive_quads;
ab_inst: AB
  port map (
  VAREF => varef_pad,
  …
  …
  );
```

drive_analog_input

A new module with the same functionality as drive_analog_io has been added to Verilog library (for naming consistency with VHDL).

The table below lists procedures/modules available for driving specific Analog System Builder analog input pins in both Verilog and VHDL libraries.

Analog input procedure-module	AV0…AV9	AC0…AC9	AT0…AT9	AV/AC pair	GNDREF	VAREF
drive_analog_input	X	X	X		X	X
drive_analog_io (Verilog only)	X	X	X		X	X
drive_temperature_quad			X			
drive_current_inputs				X		
drive_differential_inputs				X		
drive_current_monitor		X				
drive_varef_out (Verilog only)						X

Each of the AV, AC, and AT pins may be independently configured to be used as a digital input. An AV/AC pair refers to the AV and AC associated with a specific quad. When used as a pair these inputs will both be driven from one function/module. For example:

- AV0 and AC0 can both be driven using drive_current_monitor
- AV9 and AC9 can both be driven using drive_differential_voltage

Note: Make sure that the Flash Memory System Builder's INIT_DONE output is '1' before you drive the analog block (AB) analog quads (AV0-9, AC0-9 and AT0-9) in the stimulus file. For example:

```
ENTITY testbench IS
END testbench;
Architecture stimuli of testbench is
...
begin
...
drive_analog_input ( real(AV0_real), AV0_serial );
...
serial_AV0 : process
begin
  wait until ( INIT_DONE = '1' );
  AV0_real <= 1.2;
  …
  …
end process serial_AV0;
```

Polarity

Each quad has a polarity bit, Bx[6], (e.g. B0[6] for AV0-polarity and B4[6] for AV1-polarity). The default polarity is Positive, Bx[6] = '0'. Polarity error occurs when the polarity bit is inconsistent with the quad sign (e.g. AV0 > 0 and AV0-polarity = '1').

AT-quad can only be positive and therefore its polarity can only be set to Positive, Bx[6] = '0'. For AT-quad, polarity error occurs if AT is negative or if AT-polarity is set to Negative, Bx[6] = '1'.

Prescaler

Each quad has a Prescaler Opamp mode bit, Bx[7], (e.g. B0[7] for AV0- prescaler op-amp and B4[7] for AV1 prescaler op-amp). Default is Powerdown, Bx[7] = '0'.

If the factor of the prescaler input and scaling factor is greater than the internal reference voltage, the prescaler output will saturate and the prescaler output will be equal to the internal reference voltage (default 2.56V).

If a Polarity error occurs (e.g. AV0 > 0 and AV0-polarity = '1'), the prescaler output will be '0.0'.

Current Monitor

Each C-quad has a Current Monitor Switch bit (B0[4] for AC0, B4[4] for AC1, etc.). This switch needs to be 'ON' if the analog MUX selects the Current Monitor, otherwise the analog MUX output will be '0.0'. Default is Off, B0[4] = '0'.

The current monitor output is the difference between the AV and AC multiplied by a factor of 10. CMSTB-9 enables the current monitor for analog quads 0-9. Additionally, each C-quad has a Current-Monitor Switch (B0[4]) which enables you to switch the current monitor on or off. This switch needs to be 'ON' if the analog MUX selects the Current Monitor input, otherwise the analog MUX output will be '0.0'. The default setting is off.

The following requirements must be met in order to use the current monitor:

- ABS(AV) needs to be greater than ABS(AC), otherwise the Current Monitor returns a value of 0.0
- AV and AC must have the same sign and polarity. If not, they are invalid Current monitor inputs, and the current monitor output will be 0.0
- If a Polarity error occurs (e.g. AV0 > 0 and AV0-polarity = '1', or AC0 < 0 and AC0-polarity = '0'), the current monitor output will be 0.0
- If the difference between the AV and AC multiplied by a factor of 10 is greater than the internal reference voltage, the current monitor output saturates and the current monitor output is equal to the internal reference voltage (default 2.56V).

Temperature Monitor

The temperature monitor output is the AT-quad value multiplied by a factor of 12.5. TMSTB0-9 enables the temperature monitor for analog quads 0-9.

AT quad only accepts positive voltages, and T-pad polarity has to be set to 0 (Positive)

If the AT-quad value multiplied by 12.5 is greater than the internal reference voltage, the temperature monitor output saturates and the temperature monitor output is equal to the internal reference voltage (default 2.56V).

When using the temperature monitor, to reflect a temperature change, the value applied to AT should be a differential voltage.

AT (delta V) = T (K) * (0.0595 / 300)

AT (delta V) = T (K) * 1.983E-4

Using ADC in the Temperature Monitor

Using the previous equation, 300K (room temperature) should correspond to 59.5mV (0.0595 on AT, therefore 0.748 at the temperature monitor output / ADC input). When doing a 10-bit ADC conversion using a 2.56V reference voltage, 0.0595 on AT (T=300K) will give a RESULT of 300 (decimal). In this case, 1LSB change on RESULT corresponds to 1K temperature change.

Direct Analog Input

Each V, C and T-quad has a Direct Analog Input Switch (B0[5] for AV0, B1[5] for AC0, and B3[5] for AT0) which enables you to switch the direct analog input ON or OFF. This switch must be ON if the analog MUX selects the direct analog input, otherwise the analog MUX output will be 0.0. The default setting is OFF.

Analog Quad Switch Conditions

For the V-quad, the analog MUX can choose the V-prescaler or the V-direct analog input; for the C-quads, the analog MUX can select the C-prescaler, C-direct analog input or current monitor; for the T-quads, the analog MUX can choose the T-prescaler, the T-direct analog input or the Temperature Monitor.

Set the V, C and T-prescaler Opamp, V, C and T-direct analog input switches and Current Monitor switch need be set according to the table below for power efficiency and/or ADC conversion accuracy.

Selected MUX input Switch \	V-Prescaler	V-Direct analog input	C-Prescaler	C-Direct analog input	Current Monitor	T-Prescaler	T-Direct analog input	Temp. Monitor
V-Prescale Op Amp	ON	OFF	X	X	X	X	X	X
V-Direct analog input switch	OFF	ON	X	X	"OFF" or "ON and 0<AV<varef"	X	X	X
C-Prescale Op-Amp	X	X	ON	OFF	OFF	X	X	X
C-Direct analog input switch	X	X	OFF	ON	OFF	X	X	X
Current Monitor switch	X	X	OFF	OFF	ON	X	X	X
T-Prescaler Op-Amp	X	X	X	X	X	ON	OFF	OFF
T-Direct input switch	X	X	X	X	X	OFF	ON	OFF

If you do not meet the switch conditions above, the analog MUX output is 0.0 and an error message appears.

In addition, when you the AV direct analog input, the scaling factor for the V-prescaler (B0[2-0]) must be 000 to avoid accidental damage if high voltages are applied.

The same restriction applies when you select the AC direct input; the scaling factor for the C-prescaler (B1[2-0]) must be 000. This restriction does not apply to the AT direct analog input.

When you do not use an analog input quad as an analog input or a digital input, it must tied to GND (1'b0 in Verilog and '0' in VHDL). The corresponding configuration byte must be set to "00000000".

ADC

The Fusion datasheet, available at http://www.actel.com/techdocs/ds/default.aspx, contains a detailed description of the ADC.

The power-up calibration time after ADC comes out of reset is 3840 ADC_CLK cycles.

The conversion time can vary greatly depending on the SYSCLK frequency, ADCCLK frequency (determined by TVC), the STC settings, and the conversion bit-resolution (MODE).

t_conv = t_sync_read + t_sample + t_distrib + t_post_cal + t_sync_write

t_conv = SYSCLK period + ((2 + STC) * ADCCLK period) + (8, 10 or 12 * ADCCLK period) + (2 * ADCCLK period) + SYSCLK period

t_sync_read: Time for latching the input data

t_sample: Time for sampling the analog signal

t_distrib: Time for charge distribution

t_post_cal: Time for post-calibration

t_sync_write: Time for latching the output data

A Verilog parameter / VHDL generic enables faster conversion time. See the "User Parameter / Generics" on page 151 for more information.

RTC

When you use the RTC, you must first write the CTRL_STAT register. Refer to the Fusion datasheet for more information on the functionality of each CTRL_STAT register's bit. If do not use the RTC, Actel recommends that you write the CTRL_STAT register to "00000000".

When doing a byte-read from the counter, the match register, or the individual match bits, byte 0 must be read before other bytes can be read. A byte 0 read latches the 5-byte register into a 40-bit capture register. The following read operations are made from the 40-bit capture register. For the counter, the match register or the individual match bits, if bytes 1 through 4 are read before byte 0 is read, the read data (ACMRDATA) is irrelevant.

Example: Proper read sequence

```
Read COUNTER0
Read COUNTER4
Read COUNTER3
Read MATCHREG0
Read MATCHREG2
Read MATCHBITS0
Read MATCHBITS2
Read MATCHBITS3
```

Register Read and Write Conditions

Use ACMADDR, ACWEN and ACMCLK (and ACMWDATA for write) to control RTC read and write operations. Besides setting those control signals properly, the following conditions need to be met before the read or write on the chosen register byte can be executed.

```
COUNTER write: rstb_cnt = '1', cntr_en = '0' and rtm_rst = '0'
MATCHREG write: rtm_rst = '0'
MATCHBITS can not be overwritten
COUNTER read: rstb_cnt = '1' and rtm_rst = '0'
MATCHREG read: rtm_rst = '0'
MATCHBITS read: rstb_cnt = '1' and rtm_rst = '0'
```

User Parameter / Generics

MEMORYFILE

This feature enables loading of the memory initial values. This can especially help with the AQ-ACM and RTC configurations. In Verilog the MEMORY is defined as an array of 8 bits by 90, and the memory init file format needs to be of a similar type. In VHDL, the MEMORY is defined as a 720-bit vector, and the memory init file needs to be of a similar type. Default is an empty string (i.e. no memory init file).

WARNING_MSGS_ON

This feature enables you to disable the warning messages display. Default is ON ('True' in VHDL and '1' in Verilog).

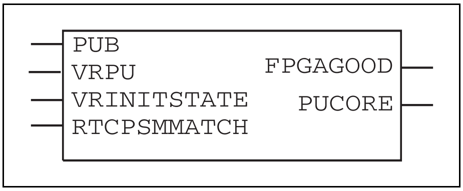
FAST_ADC_CONV_SIM

Setting FAST_ADC_CONV_SIM to True enables much faster ADC conversion time; In this fast simulation mode, the time for latching input and output data (one SYSCLK period each), and the time for sampling the analog input signal ((2 + STC) * ADCCLK period) are not accounted for. The default is OFF ('False' in VHDL and '0' in Verilog). Set FAST_ADC_CONV_SIM to 'True' in VHDL and to '1' in Verilog to enable this fast simulation mode.

Note: This is in simulation mode only. There is no equivalent mode on silicon.

Voltage Regulator and Power Supply Monitor Macro

Voltage Regulator and Power Supply Monitor (VRPSM)



Function

The Voltage Regulator and Power Supply Monitor were combined into one macro because the VR and power supply logic work together to control the power-up state of the FPGA core.

Inputs / Outputs

Inputs are listed on the left, outputs on the right.
See the VRPSM signal description below for an explanation of the inputs and outputs.

The VR generates a 1.5 V power supply (500 mA max) from the 3.3 V power supply. The 1.5 V output is intended to supply all 1.5 V needs of the Fusion product. This regulator requires an external bipolar pass transistor. Enable for this block is generated in the VR logic block, or from an external pad.

The 1.5 V is not supplied internally to the Fusion device. It must be routed externally to the VCC pins on the device. Therefore the user is not required to use the V-Reg and can use an off-chip 1.5 V supply if desired.

The VRPSM can be enabled from several sources: the PUB pin, RTCMATCH signal from the Analog Block's RTC, or triggered by the PUP0 (RTINIT1 and RTINIT1, PC bits). In the simulation library, PUP0 is represented by VRINITSTATE. VRINITSTATE is FPGAGOOD initial power-up value. It enables you to drive FPGAGOOD to '1' or '0', before the 3.3V is up. The PUCORE output is the Power-Up Bar (PUB) input inverted.

Once triggered the VRPSM remains on because of the latching functions of RS flip-flops. Only the FPGA fabric can reset these flip-flops and turn off the VRPSM. Once the FPGAGOOD signal is established, this VRPSM enable mechanism is no longer active. See the tables below for signal descriptions and recommended power-up sequences.

VRPSM Signal Description and Power-Up Sequences

The signals for the VPRSM macro are listed in the table below. The PUB input comes from the PUB pin on the device and can be pulled high by a signal external to the Fusion device. This can be used to wake up from a standby condition. The inputs VRINITSTATE and RTCPSMMTACH come from the VR Init and RTC blocks respectively and either can initiate a VR power up.

NAME	Number of Bits	Direction	FUNCTION
PUB	1	INPUT	Power-up bar
VRPU	1	INPUT	Voltage regulator power-up
VRINITSTATE	1	INPUT	FPGAGOOD initial value (set by 2 flash bits in the FPGA)
RTCPSMMATCH	1	INPUT	Connected to RTCMATCH signal from RTC
FPGAGOOD	1	OUTPUT	Indicates that the FPGA is logically functional
PUCORE	1	OUTPUT	Power-up to core

Recommended power up sequences are listed below. ? indicates a don't care value.

	PUB	VRINITSTATE	VRPU	RTCPSMMATCH	FPGAGOOD
Initial power-up	?	1	?	?	1
	?	0	?	?	0
Sequence	1	?	0	0	0
	0	?	0	0	1
	1	?	1	0	1
	1	?	0	0	0
Sequence	1	?	0	0	0
	1	?	0	1	1
	1	?	1	0	1
	1	?	0	0	0

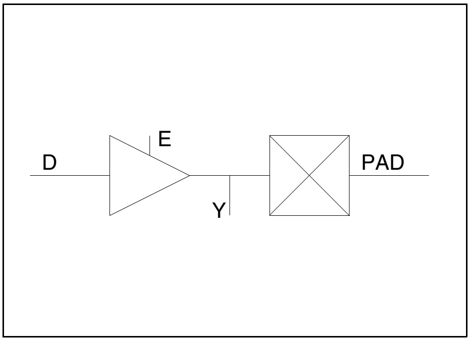
Functional Description

The Fusion datasheet, available at http://www.actel.com/techdocs/ds/default.aspx, contains a detailed functional description of the entire VRPSM and its uses.

I/O Macros

BIBUF

IGLOO, Fusion, ProASIC3



Function

Bidirectional Buffer, High Slew (with Hidden Buffer at Y pin)

Truth Table

MODE	E	D	PAD	Y
OUTPUT	1	X	D	D
INPUT	0	X	X	PAD

Input	Output
D, E, PAD	PAD, Y

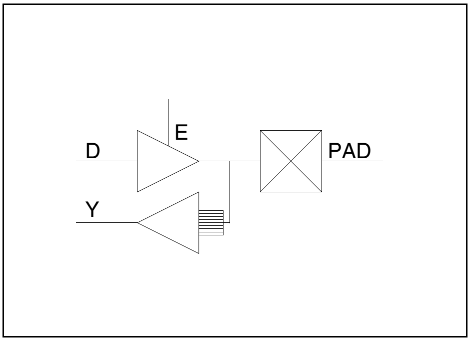
Family	I/O Tiles
All	1

Attribute Default Values

Attribute	Default Value	
	ProASIC3	ProASIC3E
IO_THRESH	LVTTL	LVTTL
OUT_DRIVE	12	12
SLEW	HIGH	HIGH
SKEW	OFF	OFF
IN_DELAY	N/A	OFF
SCHMITT_TRIGGER	N/A	OFF
RES_PULL	NONE	NONE

CLKBIBUF

IGLOO, Fusion, ProASIC3



Function

Bidirectional with Input Dedicated to routed Clock Network

Truth Table

D	E	PAD	Y
X	0	Z	X
X	0	0	0
X	0	1	1
0	1	0	0
1	1	1	1

Input	Output
D, E, PAD	PAD, Y

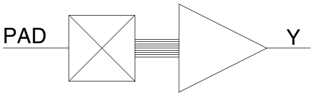
Family	I/O Tiles
All	1

Attribute Default Values

Attribute	Default Value	
	ProASIC3	ProASIC3E
IO_THRESH	LVTTL	LVTTL
OUT_DRIVE	12	12
SLEW	HIGH	HIGH
SKEW	OFF	OFF
IN_DELAY	N/A	OFF
SCHMITT_TRIGGER	N/A	OFF
RES_PULL	NONE	NONE

CLKBUF

IGLOO, Fusion, ProASIC3



Function

Input for Dedicated Routed Clock Network

Truth Table

PAD	Y
0	0
1	1

Attribute Default Values

Attribute	Default Value	
	ProASIC3	ProASIC3E
IO_THRESH	LVTTL	LVTTL
IN_DELAY	N/A	OFF
SCHMITT_TRIGGER	N/A	OFF
RES_PULL	NONE	NONE

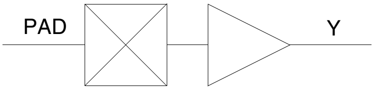
Input	Output
PAD	Y

Family	I/O Tiles
All	1

NOTE 1: For an internal Clock net, refer to the CLKINT macro.

INBUF

IGLOO, Fusion, ProASIC3



Function

Input Buffer

Truth Table

PAD	Y
0	0
1	1

Attribute Default Values

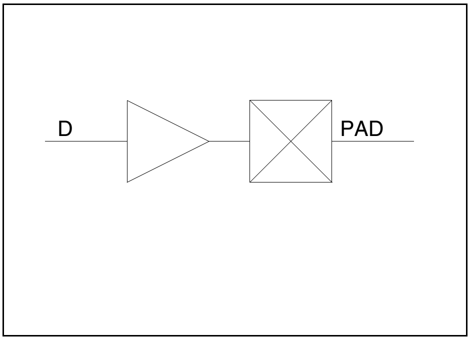
Attribute	Default Value	
	ProASIC3	ProASIC3E
IO_THRESH	LVTTL	LVTTL
IN_DELAY	N/A	OFF
SCHMITT_TRIGGER	N/A	OFF
RES_PULL	NONE	NONE

Input	Output
PAD	Y

Family	I/O Tiles
All	1

OUTBUFF

IGLOO, Fusion, ProASIC3



Input	Output
D	PAD

Family	I/O Tiles
All	1

Function

Output Buffer, High Slew

Truth Table

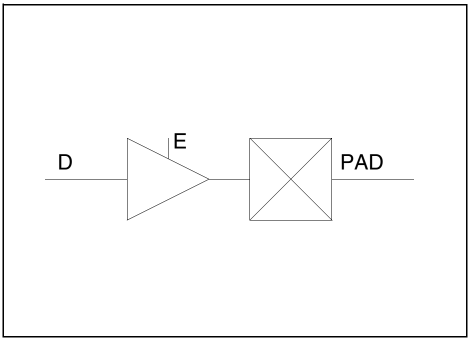
D	PAD
0	0
1	1

Attribute Default Values

Attribute	Default Value	
	ProASIC3	ProASIC3E
IO_THRESH	LVTTL	LVTTL
OUT_DRIVE	12	12
SLEW	HIGH	HIGH
RES_PULL	NONE	NONE

TRIBUFF

IGLOO, Fusion, ProASIC3



Input	Output
D, E	PAD

Family	I/O Tiles
All	1

Function

Tristate Output, High Slew

Truth Table

E	PAD
0	Z
1	D

Attribute Default Values

Attribute	Default Value	
	ProASIC3	ProASIC3E
IO_THRESH	LVTTL	LVTTL
OUT_DRIVE	12	12
SLEW	HIGH	HIGH
SKEW	OFF	OFF
RES_PULL	NONE	NONE

IGLOO, Fusion and ProASIC3 Input I/O Macros

Names for the input buffers are composed of up to 4 parts:

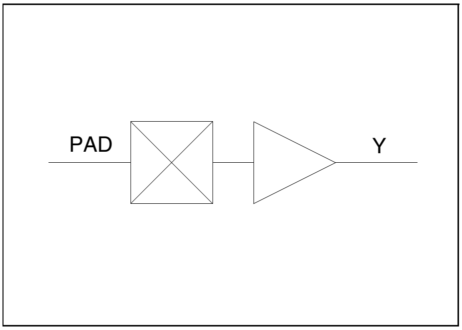
- A base name indicating the type of buffer: INBUF
- I/O Technology like LVCMOS
- An optional number code 33, 25, 18 or 15 indicating a 3.3, 2.5, 1.8 OR 1.5 voltage level.
- An optional one character code (U/D) designating a pull-up/down resistor. When the buffer has no resistor, this code is omitted.

For example:

- INBUF_LVCMOS25U - An input LVCMOS buffer with 2.5 CMOS voltage levels, pull-up resistor.
- INBUF_PCIX - An input PCIX buffer

INBUF_X

IGLOO, Fusion, ProASIC3



Function
Global Input Buffer

Truth Table

PAD	Y
0	0
1	1

Input
PAD

Output
Y

Family	I/O Tiles
All	1

Available INBUF_X Macro Types

Name	Description
INBUF_LVCMOS5	LVCMOS Input buffer with 2.5V CMOS voltage level, 5.0V tolerant †
INBUF_LVCMOS5D	LVCMOS Input buffer with 2.5V CMOS voltage level, pull-down resistor, 5.0V tolerant †
INBUF_LVCMOS5U	LVCMOS Input buffer with 2.5V CMOS voltage level, pull-up resistor, 5.0V tolerant †
INBUF_LVCMOS33	LVCMOS Input buffer with 3.3 CMOS voltage level
INBUF_LVCMOS33U	LVCMOS Input buffer with 3.3 CMOS voltage level, pull-up resistor
INBUF_LVCMOS33D	LVCMOS Input buffer with 3.3 CMOS voltage level, pull-down resistor
INBUF_LVCMOS25	LVCMOS Input buffer with 2.5 CMOS voltage level*
INBUF_LVCMOS25U	LVCMOS Input buffer with 2.5 CMOS voltage level, pull-up resistor*
INBUF_LVCMOS25D	LVCMOS Input buffer with 2.5 CMOS voltage level, pull-down resistor*
INBUF_LVCMOS18	LVCMOS Input buffer with 1.8 CMOS voltage level
INBUF_LVCMOS18U	LVCMOS Input buffer with 1.8 CMOS voltage level, pull-up resistor
INBUF_LVCMOS18D	LVCMOS Input buffer with 1.8 CMOS voltage level, pull-down resistor
INBUF_LVCMOS15	LVCMOS Input buffer with 1.5 CMOS voltage level
INBUF_LVCMOS15U	LVCMOS Input buffer with 1.5 CMOS voltage level, pull-up resistor
INBUF_LVCMOS15D	LVCMOS Input buffer with 1.5 CMOS voltage level, pull-down resistor
INBUF_PCI	PCI Input buffer
INBUF_PCIX	PCIX Input buffer
INBUF_GTL25	GTL Input buffer with 2.5 CMOS voltage level*
INBUF_GTL33	GTL Input buffer with 3.3 CMOS voltage level*
INBUF_GTLP25	GTLP Input buffer with 2.5 CMOS voltage level*

Available INBUF_X Macro Types (Continued)

Name	Description
INBUF_GTLP33	GTLP Input buffer with 3.3 CMOS voltage level*
INBUF_HSTL_I	HSTL Class I Input buffer*
INBUF_HSTL_II	HSTL Class II Input buffer*
INBUF_SSTL2_I	SSTL2 Class I Input buffer*
INBUF_SSTL2_II	SSTL2 Class II Input buffer*
INBUF_SSTL3_I	SSTL3 Class I Input buffer*
INBUF_SSTL3_II	SSTL3 Class II Input buffer*
INBUF_A	Analog input buffer; you must connect the GNDREF and ATRTN01 - ATRTN89 pads (in the Analog System Builder) to this buffer. You cannot use a generic INBUF in place of INBUF_A.
INBUF_DA	Digital or analog input buffer; you must connect the voltage, current, and temperature monitoring pads (from the Analog System Builder) to this macro. You cannot use a generic INBUF in place of INBUF_DA.
INBUF_FF	Flash*Freeze input buffer; Flash*Freeze is available only for low power devices: IGLOO PLUS, IGLOOe, IGLOO and ProASIC3L. See the Flash*Freeze section of the device handbook or the Libero IDE online help for more information on this macro and its implementation.

† = The A3P030 device does not support INBUF_LVCMOS5, INBUF_LVCMOS5D, or INBUF_LVCMOS5U

* = LVCMOS 2.5 V and LVCMOS 2.5 V / 5.0 V I/O standards are identical in the ProASIC3 family. For the A3P030 device, these standards have no clamp diode; therefore, they both behave like a LVCMOS 2.5 V standard. For other ProASIC3 devices, these standards have a clamp diode; therefore, they both behave like a LVCMOS 2.5 V / 5.0 V input standard.

Bi-Directional IO Macros

Names for the bi-directional buffers are composed of up to 4 parts:

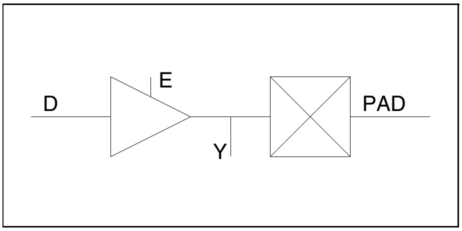
- A base name indicating the type of buffer: BIBUF
- Optional IO Technology like LVCMOS
- An optional number code indicating drive strength in milli-amps.
- An optional one character code (S/F) indicating high(F) slew or low(S) slew
- An optional one character code (U/D) designating a pull-up/down resistor. When the buffer has no resistor, this code is omitted.

For example:

- BIBUF_LVCMOS25U - A bi-directional LVCMOS buffer with 2.5 CMOS voltage levels, pull-up resistor
- BIBUF_S_8- A bi-directional buffer with low slew and 8 mA drive strength

BIBUF_X

IGLOO, Fusion, ProASIC3



Input
D, E, PAD

Output
Y, PAD

Function
Bidirectional Buffer (with Hidden Buffer at Y pin)

Truth Table

MODE	E	D	PAD	Y
OUTPUT	1	X	D	D
INPUT	0	X	X	PAD

Family

Family	I/O Tiles
All	1

BIBUF_X Macro Types

Name	Description
BIBUF_LVCMOS33	LVCMOS bi-directional buffer with 3.3 CMOS voltage level
BIBUF_LVCMOS33U	LVCMOS bi-directional buffer with 3.3 CMOS voltage level, pull-up resistor
BIBUF_LVCMOS33D	LVCMOS bi-directional buffer with 3.3 CMOS voltage level, pull-down resistor
BIBUF_LVCMOS25	LVCMOS Bi-directional buffer with 2.5 CMOS voltage level
BIBUF_LVCMOS25U	LVCMOS Bi-directional buffer with 2.5 CMOS voltage level, pull-up resistor
BIBUF_LVCMOS25D	LVCMOS Bi-directional buffer with 2.5 CMOS voltage level, pull-down resistor
BIBUF_LVCMOS18	LVCMOS Bi-directional buffer with 1.8 CMOS voltage level
BIBUF_LVCMOS18U	LVCMOS Bi-directional buffer with 1.8 CMOS voltage level, pull-up resistor
BIBUF_LVCMOS18D	LVCMOS Bi-directional buffer with 1.8 CMOS voltage level, pull-down resistor
BIBUF_LVCMOS15	LVCMOS Bi-directional buffer with 1.5 CMOS voltage level
BIBUF_LVCMOS15U	LVCMOS Bi-directional buffer with 1.5 CMOS voltage level, pull-up resistor
BIBUF_LVCMOS15D	LVCMOS Bi-directional buffer with 1.5 CMOS voltage level, pull-down resistor
BIBUF_PCI	PCI Bi-directional buffer
BIBUF_PCIX	PCIX Bi-directional buffer
BIBUF_SSTL2_I	SSTL2 class I bi-directional buffer*
BIBUF_SSTL2_II	SSTL2 class II bi-directional buffer*
BIBUF_SSTL3_I	SSTL3 class I bi-directional buffer
BIBUF_SSTL3_II	SSTL3 class II bi-directional buffer*
BIBUF_HSTL_I	HSTL class I bi-directional buffer*
BIBUF_HSTL_II	HSTL class II bi-directional buffer*
BIBUF_GTL25	GTL bi-directional buffer*
BIBUF_GTL33	GTL bi-directional buffer*
BIBUF_GTLP25	GTLP Bi-directional buffer with 2.5 CMOS voltage level*

BIBUF_X Macro Types (Continued)

Name	Description
BIBUF_GTLP33	GTLP Bi-directional buffer with 3.3 CMOS voltage level*
BIBUF_F_2	Bi-directional buffer with high slew
BIBUF_F_2U	Bi-directional buffer with high slew and pull-up resistor
BIBUF_F_2D	Bi-directional buffer with high slew and pull-down resistor
BIBUF_F_4	Bi-directional buffer with high slew
BIBUF_F_4U	Bi-directional buffer with high slew and pull-up resistor
BIBUF_F_4D	Bi-directional buffer with high slew and pull-down resistor
BIBUF_F_6	Bi-directional buffer with high slew
BIBUF_F_6U	Bi-directional buffer with high slew and pull-up resistor
BIBUF_F_6D	Bi-directional buffer with high slew and pull-down resistor
BIBUF_F_8	Bi-directional buffer with high slew
BIBUF_F_8U	Bi-directional buffer with high slew and pull-up resistor
BIBUF_F_8D	Bi-directional buffer with high slew and pull-down resistor
BIBUF_F_12	Bi-directional buffer with high slew
BIBUF_F_12U	Bi-directional buffer with high slew and pull-up resistor
BIBUF_F_12D	Bi-directional buffer with high slew and pull-down resistor
BIBUF_F_16	Bi-directional buffer with high slew
BIBUF_F_16U	Bi-directional buffer with high slew and pull-up resistor
BIBUF_F_16D	Bi-directional buffer with high slew and pull-down resistor
BIBUF_F_24	Bi-directional buffer with high slew
BIBUF_F_24U	Bi-directional buffer with high slew and pull-up resistor
BIBUF_F_24D	Bi-directional buffer with high slew and pull-down resistor
BIBUF_S_2	Bi-directional buffer with low slew
BIBUF_S_2U	Bi-directional buffer with low slew and pull-up resistor
BIBUF_S_2D	Bi-directional buffer with low slew and pull-down resistor
BIBUF_S_4	Bi-directional buffer with low slew
BIBUF_S_4U	Bi-directional buffer with low slew and pull-up resistor
BIBUF_S_4D	Bi-directional buffer with low slew and pull-down resistor
BIBUF_S_6	Bi-directional buffer with low slew
BIBUF_S_6U	Bi-directional buffer with low slew and pull-up resistor
BIBUF_S_6D	Bi-directional buffer with low slew and pull-down resistor
BIBUF_S_8	Bi-directional buffer with low slew
BIBUF_S_8U	Bi-directional buffer with low slew and pull-up resistor
BIBUF_S_8D	Bi-directional buffer with low slew and pull-down resistor
BIBUF_S_12	Bi-directional buffer with low slew
BIBUF_S_12U	Bi-directional buffer with low slew and pull-up resistor
BIBUF_S_12D	Bi-directional buffer with low slew and pull-down resistor
BIBUF_S_16	Bi-directional buffer with low slew
BIBUF_S_16U	Bi-directional buffer with low slew and pull-up resistor
BIBUF_S_16D	Bi-directional buffer with low slew and pull-down resistor
BIBUF_S_24	Bi-directional buffer with low slew
BIBUF_S_24U	Bi-directional buffer with low slew and pull-up resistor
BIBUF_S_24D	Bi-directional buffer with low slew and pull-down resistor

* = not supported in ProASIC3

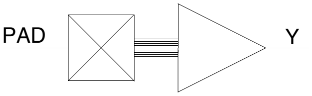
Clock Buffers

Names for the input buffers are composed of up to 3 parts:

- A base name indicating the type of buffer: CLKBUF
- IO Technology like LVCMOS
- An optional number code 33, 25, 18 or 15 indicating a 3.3, 2.5, 1.8 OR 1.5 voltage level

CLKBUF_X

IGLOO, Fusion, ProASIC3



Function
Input for Dedicated Routed Clock Network

Truth Table

PAD	Y
0	0
1	1

Input
PAD

Output
Y

Family

Family	I/O Tiles
All	1

NOTE 1: For an internal Clock net, refer to the CLKINT macro.

Available CLKBUF_X Macro Types

Name	Description
CLKBUF_LVCMOS5	LVCMOS Clock buffer with 2.5V CMOS voltage level, 5.0V tolerant
CLKBUF_LVCMOS33	LVCMOS Clock buffer with 3.3 CMOS voltage level
CLKBUF_LVCMOS25	LVCMOS Clock buffer with 2.5 CMOS voltage level *
CLKBUF_LVCMOS18	LVCMOS Clock buffer with 1.8 CMOS voltage level
CLKBUF_LVCMOS15	LVCMOS Clock buffer with 1.5 CMOS voltage level
CLKBUF_PCI	PCI Clock buffer
CLKBUF_PCIX	PCIX Clock buffer
CLKBUF_GTL25	GTL Clock buffer with 2.5 CMOS voltage level *
CLKBUF_GTL33	GTL Clock buffer with 3.3 CMOS voltage level*
CLKBUF_GTLP25	GTLP Clock buffer with 2.5 CMOS voltage level *
CLKBUF_GTLP33	GTLP Clock buffer with 3.3 CMOS voltage level *
CLKBUF_ HSTL _I	HSTL Class I Clock buffer *
CLKBUF_ HSTL _II	HSTL Class II Clock buffer *
CLKBUF_SSTL2_I	SSTL2 Class I Clock buffer *
CLKBUF_SSTL2_II	SSTL2 Class II Clock buffer *
CLKBUF_SSTL3_I	SSTL3 Class I Clock buffer *
CLKBUF_SSTL3_II	SSTL3 Class II Clock buffer *

† = The A3P030 device does not support CLKBUF_LVCMOS5

* = LVCMOS 2.5 V and LVCMOS 2.5 V / 5.0 V I/O standards are identical in the ProASIC3 family. For the A3P030 device, these standards have no clamp diode; therefore, they both behave like a LVCMOS 2.5 V standard. For other ProASIC3 devices, these standards have a clamp diode; therefore, they both behave like a LVCMOS 2.5 V / 5.0 V input standard.

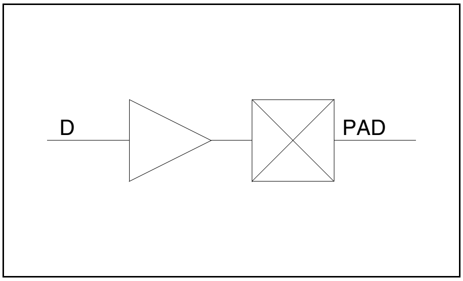
Output Buffers

Names for the bi-directional buffers are composed of up to 4 parts:

- A base name indicating the type of buffer: OUTBUF
- Optional IO Technology like LVCMOS
- An optional number code indicating drive strength in milli-amps.
- An optional one character code (S/F) indicating high (F) slew or low (S) slew

OUTBUF_X

IGLOO, Fusion, ProASIC3



Function
Output Buffer,

Truth Table

D	PAD
0	0
1	1

Family

Family	I/O Tiles
All	1

Input
D

Output
PAD

Available OUTBUF_X Macro Types

Name	Description
OUTBUF_LVCMOS33	LVCMOS Output buffer with 3.3 CMOS voltage level †
OUTBUF_LVCMOS25	LVCMOS Output buffer with 2.5 CMOS voltage level
OUTBUF_LVCMOS18	LVCMOS Output buffer with 1.8 CMOS voltage level
OUTBUF_LVCMOS15	LVCMOS Output buffer with 1.5 CMOS voltage level
OUTBUF_PCI	PCI Output buffer
OUTBUF_PCIX	PCIX Output buffer
OUTBUF_ HSTL _I	HSTL Class I Output buffer *
OUTBUF_ HSTL _II	HSTL Class II Output buffer *
OUTBUF_SSTL2_I	SSTL2 Class I Output buffer *
OUTBUF_SSTL2_II	SSTL2 Class II Output buffer *
OUTBUF_SSTL3_I	SSTL3 Class I Output buffer *
OUTBUF_SSTL3_II	SSTL3 Class II Output buffer *
OUTBUF_GTL25	GTL Output buffer with 2.5 CMOS voltage level *
OUTBUF_GTL33	GTL Output buffer with 3.3 CMOS voltage level *
OUTBUF_GTLP25	GTLP Output buffer with 2.5 CMOS voltage level *
OUTBUF_GTLP33	GTLP Output buffer with 3.3 CMOS voltage level *
OUTBUF_F_2	Output buffer with high slew
OUTBUF_F_4	Output buffer with high slew
OUTBUF_F_6	Output buffer with high slew
OUTBUF_F_8	Output buffer with high slew
OUTBUF_F_12	Output buffer with high slew
OUTBUF_F_16	Output buffer with high slew
OUTBUF_F_24	Output buffer with high slew
OUTBUF_S_2	Output buffer with low slew
OUTBUF_S_4	Output buffer with low slew
OUTBUF_S_6	Output buffer with low slew
OUTBUF_S_8	Output buffer with low slew
OUTBUF_S_12	Output buffer with low slew
OUTBUF_S_16	Output buffer with low slew
OUTBUF_S_24	Output buffer with low slew
OUTBUF_A	Analog output buffer. You must use this output buffer to indicate your analog outputs. You cannot use a generic OUTBUF in place of OUTBUF_A.

† = Actel recommends that you use this buffer to drive a 5.0V receiver

* = not supported in ProASIC3

Tri-State Buffer Macros

Names for the tri-state outputs are composed of up to 4 parts:

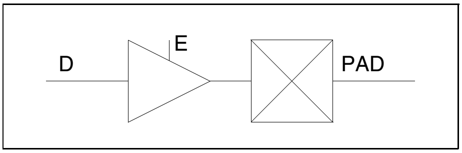
- A base name indicating the type of buffer: TRIBUFF
- Optional IO Technology like LVCMOS
- An optional number code indicating drive strength in milli-amps.
- An optional one character code (S/F) indicating high(F) slew or low(S) slew
- An optional one character code (U/D) designating a pull-up/down resistor. When the buffer has no resistor, this code is omitted.

For example:

- TRIBUFF_LVCMOS25U - A tri-state LVCMOS output with 2.5 CMOS voltage levels, pull-up resistor
- TRIBUFF_S_8- A tri-state output with low slew and 8 mA drive strength

TRIBUFF_X

IGLOO, Fusion, ProASIC3



Input
D, E

Output
PAD

Function
Tristate Output

Truth Table

E	PAD
0	Z
1	D

Family

Family	I/O Tiles
All	1

TRIBUFF_X Macro Types

Name	Description
TRIBUFF_LVCMOS33	LVCMOS tri-state output with 3.3 CMOS voltage level †
TRIBUFF_LVCMOS25	LVCMOS tri-state output with 2.5 CMOS voltage level
TRIBUFF_LVCMOS18	LVCMOS tri-state output with 1.8 CMOS voltage level
TRIBUFF_LVCMOS15	LVCMOS tri-state output with 1.5 CMOS voltage level
TRIBUFF_PCI	PCI tri-state output
TRIBUFF_PCIX	PCIX tri-state output
TRIBUFF_GTL25	GTL tri-state output with 2.5 CMOS voltage level *
TRIBUFF_GTL33	GTL tri-state output with 3.3 CMOS voltage level *
TRIBUFF_GTLP25	GTLP tri-state output with 2.5 CMOS voltage level *
TRIBUFF_GTLP33	GTLP tri-state output with 3.3 CMOS voltage level *
TRIBUFF_ HSTL _I	HSTL Class I tri-state output buffer *
TRIBUFF_ HSTL _II	HSTL Class II tri-state output buffer *
TRIBUFF_SSTL2_I	SSTL2 Class I tri-state output buffer *
TRIBUFF_SSTL2_II	SSTL2 Class II tri-state output buffer *
TRIBUFF_SSTL3_I	SSTL3 Class I tri-state output buffer *
TRIBUFF_SSTL3_II	SSTL3 Class II tri-state output buffer *
TRIBUFF_F_2	Tri-state output with high slew
TRIBUFF_F_4	Tri-state output with high slew
TRIBUFF_F_6	Tri-state output with high slew
TRIBUFF_F_8	Tri-state output with high slew
TRIBUFF_F_12	Tri-state output with high slew
TRIBUFF_F_16	Tri-state output with high slew
TRIBUFF_F_24	Tri-state output with high slew*
TRIBUFF_S_2	Tri-state output with low slew
TRIBUFF_S_4	Tri-state output with low slew
TRIBUFF_S_6	Tri-state output with low slew
TRIBUFF_S_8	Tri-state output with low slew
TRIBUFF_S_12	Tri-state output with low slew
TRIBUFF_S_16	Tri-state output with low slew
TRIBUFF_S_24	Tri-state output with low slew *

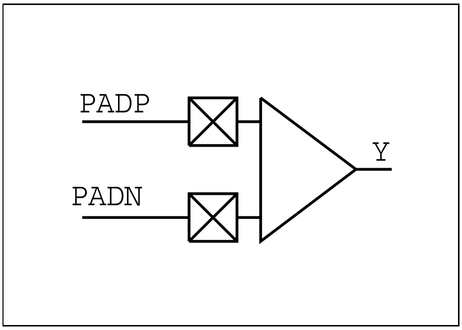
† = Actel recommends that you use this buffer to drive a 5.0V receiver

* = not supported in ProASIC3

Differential IO Macros

INBUF_LVDS; INBUF_LVPECL

IGLOO, Fusion, ProASIC3



Function
INBUF_LVDS and INBUF_LVPECL

Input
PADP; PADN

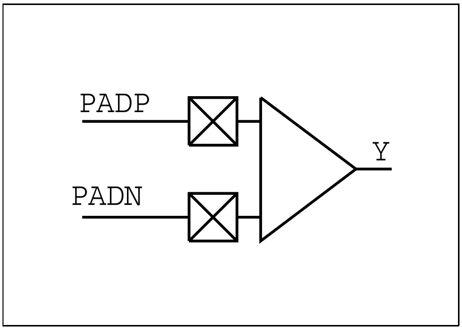
Output
Y

Available Differential Macro Types

Name	Description
INBUF_LVDS	
INBUF_LVPECL	

CLKBUF_LVDS; CLKBUF_LVPECL

IGLOO, Fusion, ProASIC3



Function
CLKBUF_LVDS and CLKBUF_LVPECL

Input
PADP; PADN

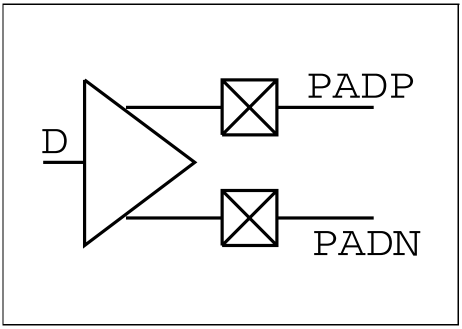
Output
Y

Available Differential Macro Types

Name	Description
CLKBUF_LVDS	
CLKBUF_LVPECL	

OUTBUF_LVDS; OUTBUF_LVPECL

IGLOO, Fusion, ProASIC3



Function

OUTBUF_LVDS and OUTBUF_LVPECL

Input

D

Output

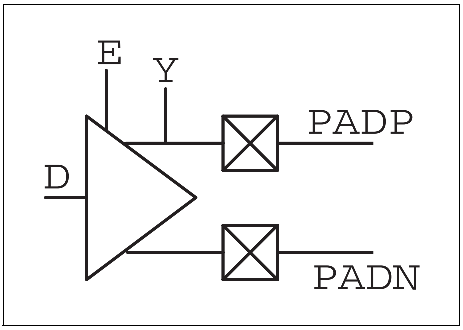
PADP, PADN

Available Differential Macro Types

Name	Description
OUTBUF_LVDS	
OUTBUF_LVPECL	

BIBUF_LVDS

IGLOO, Fusion, ProASIC3



Function

Bi-directional differential I/O, high slew

Truth Table

Mode	E	D	PADP	PADN	Y
Output	1	x	D	!D	D
Input	0	x	x	!PADP	PADP

Family

Family	I/O Tiles
All	2

Input

D, E, PADP, PADN

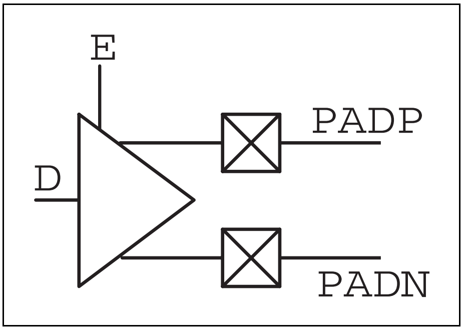
Output

PADP, PADN, Y

Attribute Default Values

Attribute	Default Value
IO_THRESH	LVDS
OUT_DRIVE	24
SLEW	HIGH
SKEW	OFF
IN_DELAY	OFF
SCHMITT_TRIGGER	NONE
RES_PULL	NONE

TRIBUFF_LVDS



Function
Tri-state differential output, high slew

Truth Table

E	D	PADP	PADN
0	x	Z	Z
1	x	D	!D

Family

Family	I/O Tiles
All	2

Input
D, E

Output
PADP, PADN,

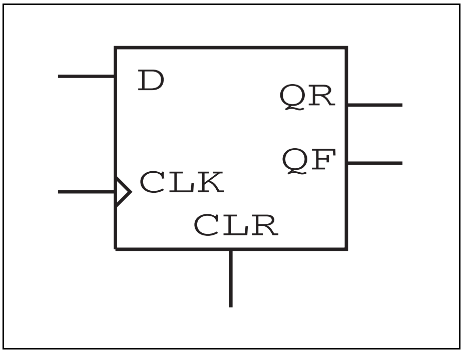
Attribute Default Values

Attribute	Default Value
IO_THRESH	LVDS
OUT_DRIVE	24
SLEW	HIGH
SKEW	OFF
RES_PULL	NONE

DDR Macros

DDR_REG

IGLOO, Fusion, ProASIC3



Function

DDR (DDR) Register; please refer to the Fusion or ProASIC3/E datasheet for more information on the DDR_REG

Truth Table

CLR	CLK	QR(n+1)	QF(n+1)
1	X	0	0
0	↑	D	QF(n)
0	↓	QR(n)	D

Input

D, CLK, CLR

Output

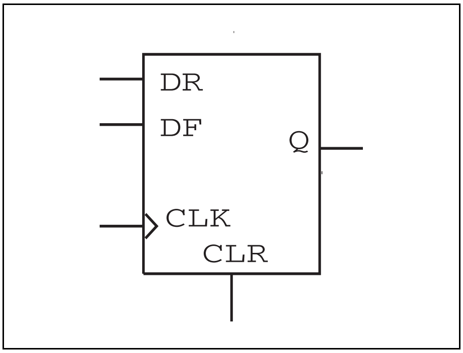
QR, QF

Family

Family	I/O Tiles
All	1

DDR_OUT

IGLOO, Fusion, ProASIC3



Function

DDR (DDR) output; please refer to the Fusion or ProASIC3/E datasheet for more information on the DDR_OUT

Truth Table

CLR	CLK	Q
1	X	0
0	↑	DR
0	↓	DF

Input

DR, DF, CLK, CLR

Output

Q

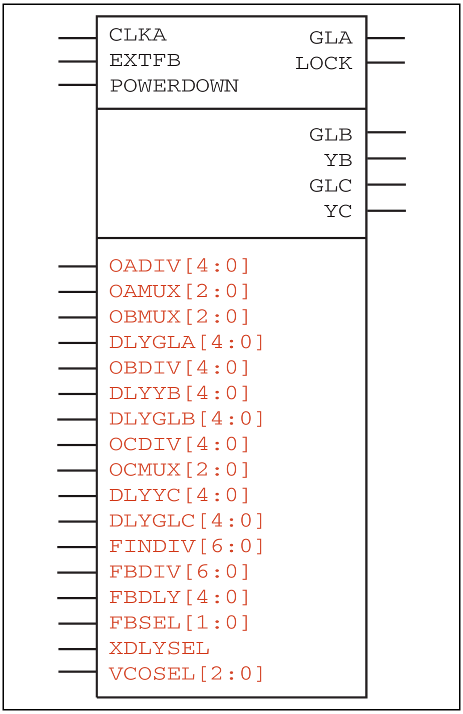
Family

Family	I/O Tiles
All	1

Clocking Resources

PLL for ProASIC3 / IGLOO

IGLOO, ProASIC3



Function

Static PLL

Actel recommends that you use SmartGen to generate your PLLs; SmartGen calculates the settings for all the pins in the PLL for the required input-output frequency combinations.

Refer to the latest Actel datasheets on PLLs for ProASIC3 / ProASIC3E for more information. They are available at http://www.actel.com.

Inputs / Outputs

See the description below for an explanation of the inputs and outputs available on the Static PLL for ProASIC3/E; all inputs are shown on the left, and ouputs are to the right.

The static PLL supports only a single input. The Combiner is able to combine the PLL with the regular CLKBUF macros and any of the CCC macros to utilize available unused globals.

In the symbol shown above, all the required user-accessible inputs and outputs are above the top horizontal line. Optional inputs and outputs are below the top line. The static configuration inputs are below the second line. These pins can only be connected to GND or VCC.

The table below summarizes the configuration control bits.

Configuration Control Bits Summary

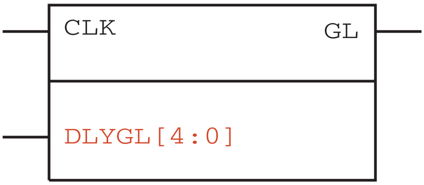
NAME	FUNCTION
FINDIV<6:0>	7-BIT INPUT DIVIDER (/N)
FBDIV<6:0>	7-BIT FEEDBACK DIVIDER (/M)
OADIV<4:0>	5-BIT OUTPUT DIVIDER (/U)
OBDIV<4:0>	5-BIT OUTPUT DIVIDER (/V)
OCDIV<4:0>	5-BIT OUTPUT DIVIDER (/W)
OAMUX<2:0>	3-BIT POST-PLL MUXA (BEFORE DIVIDER /U)
OBMUX<2:0>	3-BIT POST-PLL MUXB (BEFORE DIVIDER /V)

Configuration Control Bits Summary (Continued)

NAME	FUNCTION
OCMUX<2:0>	3-BIT POST-PLL MUXC (BEFORE DIVIDER /W)
FBSEL<1:0>	2-BIT PLL FEEDBACK MUX
FBDLY<4:0>	FEEDBACK DELAY
XDLYSEL	1-BIT PLL FEEDBACK MUX
DLYGLA<4:0>	DELAY ON GLOBAL A
DLYGLB<4:0>	DELAY ON GLOBAL B
DLYGLC<4:0>	DELAY ON GLOBAL C
DLYB<4:0>	DELAY ON YB
DLYC<4:0>	DELAY ON YC
VCOSEL<2:0>	3-BIT VCO GEAR CONTROL (4 FREQUENCY RANGES)

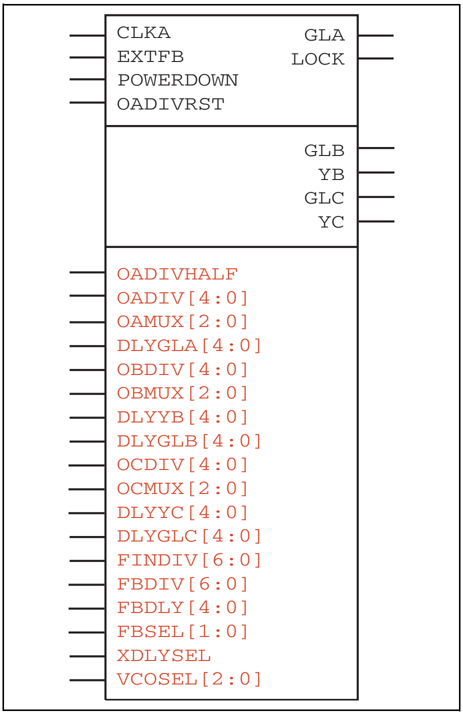
Static Clock with Divider and/or Delay

The Combiner is able to combine the clock conditioning circuit macro with the regular CLKBUF macros and the PLL to utilize available unused globals.



The CLKDLY is essentially a CLKBUF with a delay. The PLLINT macro is included to unambiguously show Designer which routing resources are required to connect the REFCLK input: The PLLINT is used when REFCLK is driven by a pad in a different I/O tile.

PLL for Fusion



Function

Static PLL

Actel recommends that you use SmartGen to generate your PLLs; SmartGen calculates the settings for all the pins in the PLL for the required input-output frequency combinations.

Refer to the latest Actel datasheets on Clocking Resources for Fusion for more information. They are available at http://www.actel.com.

Inputs / Outputs

See the description below for an explanation of the inputs and outputs available on the Static PLL for Fusion; all inputs are shown on the left, and ouputs are to the right.

The static PLL supports only a single input. The Combiner is able to combine the PLL with the regular CLKBUF macros and any of the CCC macros to utilize available unused globals.

In the symbol shown above, all the required user-accessible inputs and outputs are above the top horizontal line. The ones below the top line are optional inputs and outputs. The static configuration inputs are below the third line. These pins can only be connected to GND or VCC.

OADIVRST may only be used when you bypass the PLL core (i.e. OAMUX = 001).

The purpose of the O(A/B/C)DIVRDST signals is to reset the output of the final clock divider in order to synchronize it with the input to that divider when the PLL is bypassed. The signal is active on a low to high transition. The signal must be low for at least one divider input clock frequency, and then shifted high for at least 3 input clock cycles for the reset operation to work correctly.

The table below summarizes the configuration control bits.

Configuration Control Bits Summary

NAME	FUNCTION
FINDIV<6:0>	7-BIT INPUT DIVIDER (/N)
FBDIV<6:0>	7-BIT FEEDBACK DIVIDER (/M)
OADIVHALF*	Division by half (see Fusion datasheet for more information)

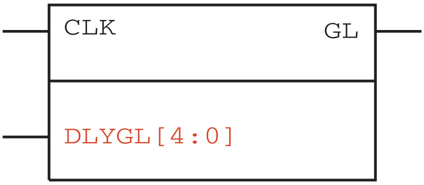
Configuration Control Bits Summary (Continued)

NAME	FUNCTION
OADIV<4:0>	5-BIT OUTPUT DIVIDER (/U)
OBDIV<4:0>	5-BIT OUTPUT DIVIDER (/V)
OCDIV<4:0>	5-BIT OUTPUT DIVIDER (/W)
OAMUX<2:0>	3-BIT POST-PLL MUXA (BEFORE DIVIDER /U)
OBMUX<2:0>	3-BIT POST-PLL MUXB (BEFORE DIVIDER /V)
OCMUX<2:0>	3-BIT POST-PLL MUXC (BEFORE DIVIDER /W)
FBSEL<1:0>	2-BIT PLL FEEDBACK MUX
FBDLY<4:0>	FEEDBACK DELAY
XDLYSEL	1-BIT PLL FEEDBACK MUX
DLYGLA<4:0>	DELAY ON GLOBAL A
DLYGLB<4:0>	DELAY ON GLOBAL B
DLYGLC<4:0>	DELAY ON GLOBAL C
DLYB<4:0>	DELAY ON YB
DLYC<4:0>	DELAY ON YC
VCOSEL<2:0>	3-BIT VCO GEAR CONTROL (4 FREQUENCY RANGES)

* OADIVHALF may only be used when you bypass the PLL core (i.e. OAMUX = 001) and the RC Oscillator (RCOSC) drives the CLKA input.

Static Clock with Divider and/or Delay

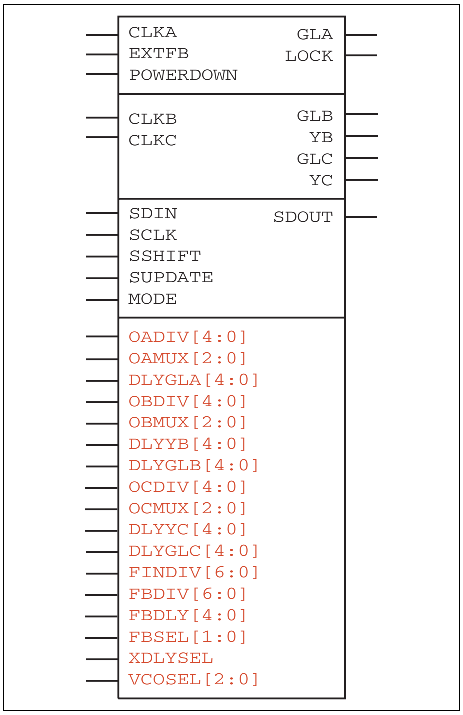
The Combiner is able to combine the clock conditioning circuit macro with the regular CLKBUF macros and the PLL to utilize available unused globals.



The CLKDLY is essentially a CLKBUF with a delay. The PLLINT macro is included to unambiguously show Designer which routing resources are required to connect the REFCLK input: The PLLINT is used when REFCLK is driven by a pad in a different I/O tile.

DYNCCC for IGLOO and ProASIC3

IGLOO, ProASIC3



Function

Dynamic PLL / Clock Conditioning Circuitry

Actel recommends that you use SmartGen to generate your DYNCCCs; SmartGen calculates the settings for all the pins in the DYNCCC for the required input-output frequency combinations.

Refer to the latest Actel datasheets on PLLs for ProASIC3 / ProASIC3E for more information. They are available at http://www.actel.com.

Inputs / Outputs

See the description below for an explanation of the inputs and outputs available on the Dynamic CCC for ProASIC3/E; all inputs are shown on the left, and ouputs are to the right.

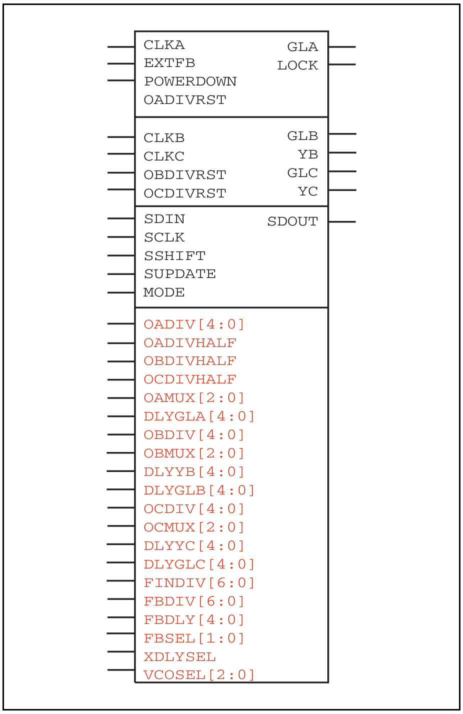
In the symbol shown above, all the required user-accessible inputs and outputs are above the top horizontal line. The ones below the top line are optional inputs and outputs. The elements below the second line (SDIN, SCLK, SDOUT, etc.) are associated with the dynamic shift register. The static configuration inputs are below the third line. Static configuration input pins can only be connected to GND or VCC.

There are seventy-four static configuration ports on the DYNCCC model, but ther are eighty dynamic configuration bits that may be shifted in. The extra bits control the STATSEL and DYNSEL input multiplexors, a silicon feature that is not reflected in the simulation model. These extra bits are ingored by the simulation model. See the PLL for ProASIC3 and ProASIC3E macro description on page 176 for a summary of the configuration control bits.

Please note the following during dynamic configuration:

- When the “MODE” pin is at logic high, the PLL control bits are the bits from the configuration register. Before “MODE” is switched from logic low to logic high, the desired dynamic configuration bits should already be shifted-in.
- Reconfiguration does not unlock the PLL as long as the new configuration does not change the state of input divider or feedback elements.

DYNCCC for Fusion



Function

Dynamic PLL / Clock Conditioning Circuitry

Actel recommends that you use SmartGen to generate your DYNCCCs; SmartGen calculates the settings for all the pins in the DYNCCC for the required input-output frequency combinations.

Refer to the latest Actel datasheets on PLLs for Fusion for more information. They are available at http://www.actel.com.

Inputs / Outputs

See the description below for an explanation of the inputs and outputs available on the Dynamic CCC for Fusion; all inputs are shown on the left, and ouputs are to the right.

In the symbol shown above, all the required user-accessible inputs and outputs are above the top horizontal line. The ones below the top line are optional inputs and outputs. The elements below the second line (SDIN, SCLK, SDOUT, etc.) are associated with the dynamic shift register. The static configuration inputs are below the third line. Static configuration input pins can only be connected to GND or VCC.

There are seventy-seven static configuration ports on the DYNCCC model, but ther are eighty-eight dynamic configuration bits that may be shifted in. The extra bits control the STATSEL, DYNSEL, and No-Glitch MUX input multiplexors, a silicon feature that is not reflected in the simulation model. These extra bits are ingored by the simulation model. See the PLL for Fusion macro description on page 178 for a summary of the configuration control bits.

Please note the following during dynamic configuration:

- When the “MODE” pin is at logic high, the PLL control bits are the bits from the configuration register. Before “MODE” is switched from logic low to logic high, the desired dynamic configuration bits should already be shifted-in.
- Reconfiguration does not unlock the PLL as long as the new configuration does not change the state of input divider or feedback elements.

External Feedback for PLL and DYNCCC

External feedback is implemented by relying on self-synchronization in-out on GLA. Since GLA is in the PLL loop, it automatically synchronizes to the incoming clock, and the additional synchronization circuits on divider U are disabled when external feedback is enabled.

GLB and GLC still have to rely on synchronization circuits for dividers V and W. As a result, GLA, GLB, and GLC are synchronized when LOCK goes high.

External feedback comes with the following restrictions:

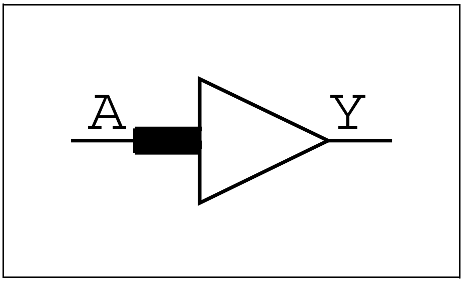
- Only GLA (the primary global) may be used as the signal for the external feedback loop.
- Division factor N is defined as: N = U*a where a= 1, 2, 3, ...
- Division factor M is defined as: M >= 5
- Total sum of delays in the feedback loop must be less than 1 VCO period and less than 1 CLKA (incoming clock) period. This restriction only applies to cases where V and or W dividers are used.
- M*U < 233

EXTFB must come from an I/O. This I/O is placed at one fixed location per CCC. Please refer to the Fusion datasheet, IGLOO/e datasheet, or ProASIC3/E datasheet for more information.

External feedback is supported on both PLL and DYNCCC cells for Fusion, IGLOO and ProASIC3 families.

PLLINT

Fusion, IGLOO, ProASIC3



Function

PLL Int

Truth Table

A	Y
0	0
1	1

Use PLLINT to connect a signal from the FPGA array to the PLL reference clock (CLKA). The input to PLLINT may come from an I/O (excluding the dedicated I/Os for the PLL being driven), local routing, or a global resource.

Input

A

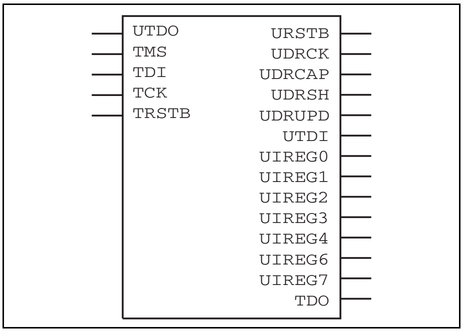
Output

Y

Refer to the latest Actel datasheets on PLLs and Clock Conditioning Circuits (CCC) application notes for more information. They are available at http://www.actel.com.

UJTAG

Fusion, IGLOO, ProASIC3



Function

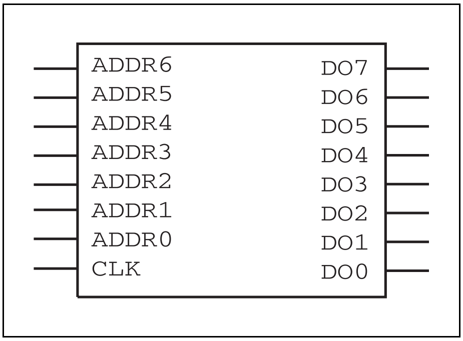
The UJTAG macro is a special purpose macro. It is provided to allow users access to the user JTAG circuitry on board the chip. You must instantiate a UJTAG macro in their design if they plan to make use of the user JTAG feature. It is identical to the APA and A500K UJTAG macro.

Input

Output

UFROM

Fusion, IGLOO, ProASIC3



Function

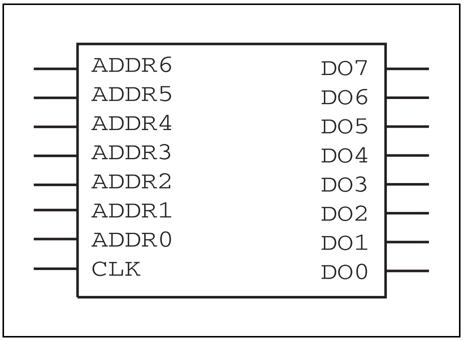
The UFROM is the USER FlashROM macro. It is a simple 128 X 8 synchronous read-only memory. There is only one UFROM per chip. New data appears on the DO pins after the falling edge of the clock pin. The UFROM can only be programmed by the user via the JTAG pins. There is currently no support for programming the UFROM in any of the CAE tools or libraries, however the simulation models will utilize a memory initialization file so users can specify the contents of the memory for simulation purposes. The memory initialization file will be an ASCII format text file containing exactly 128 lines of 8-character binary strings.

Input	Output
ADDR[0:6], CLK	DO[7:0]

Data outputs always transition to X on the rising edge of the input CLK. Please refer to the datasheet for more information.

UFROMH

Fusion, IGLOO, ProASIC3



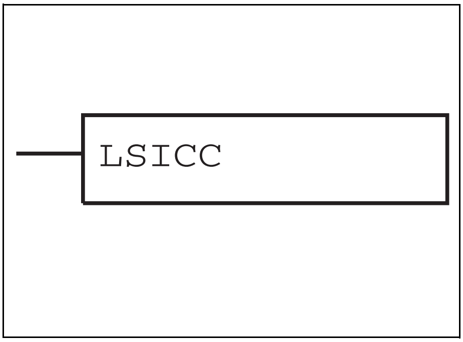
Function

UFROMH is a fixed placement version of the UFROM macro. Actel highly recommends that you use this macro instead of UFROM.

Input	Output
ADDR[0:6], CLK	DO[7:0]

ULSICC

Fusion, ProASIC3



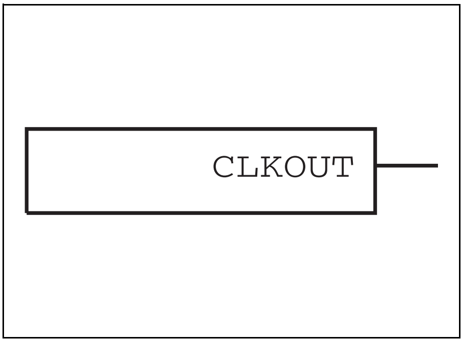
Function
Low standby ICC configuration macro.

Refer to the Fusion, ProASIC3, or ProASIC3E datasheet (at http://www.actel.com) for more information on this macro.

Input	**Output**
LSICC	

RCOSC

Fusion



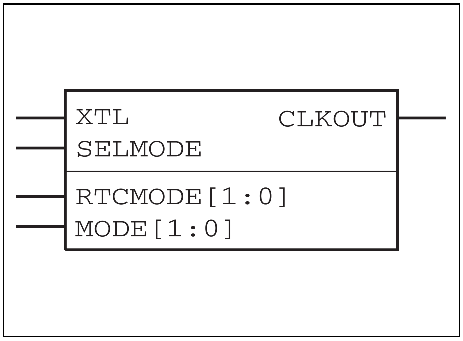
Function
On-chip free-running clock source that generates a 100 Mhz clock.

Refer to the Clocking Resources of the Fusion datasheet for more information on this macro. The Fusion datasheet is available at http://www.actel.com.

Input	**Output**
	CLKOUT

XTLOSC

Fusion



Function

On-chip crystal oscillator circuit that works with an off-chip crystal to generate a high-precision clock.

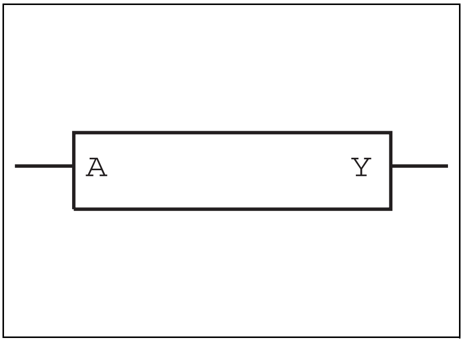
Refer to the Clocking Resources of the Fusion datasheet for more information on this macro. The Fusion datasheet is available at http://www.actel.com.

Input	Output
	CLKOUT

The XTLOSC requires a physical connection to an external crystal, ceramic resonator, or a resistor/capacitor network. For simulation purposes you can use the XTL pin to provide a clock signal running at the desired input frequency.

CLKSRC

Fusion



Function

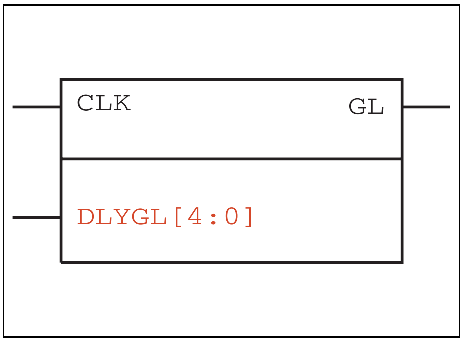
Clock buffer used to connect either the RCOSC or the XTLOSC to the core.

Refer to the Clocking Resources of the Fusion datasheet for more information on this macro. The Fusion datasheet is available at http://www.actel.com.

Input	Output
A	Y

CLKDLY

Fusion, IGLOO, ProASIC3



Function

Static clock with delay.

Refer to the Clocking Resources of the Fusion and ProASIC3/E datasheets for more information on this macro. They are available at http://www.actel.com.

Input

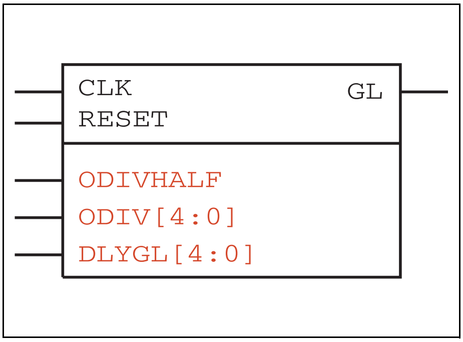
CLK, DLYGL[4:0]

Output

GL

CLKDIVDLY

Fusion



Function

Static clock with divider and/or delay with global output driver only.

Refer to the Clocking Resources of the Fusion and ProASIC3/E datasheets for more information on this macro. They are available at http://www.actel.com.

Input

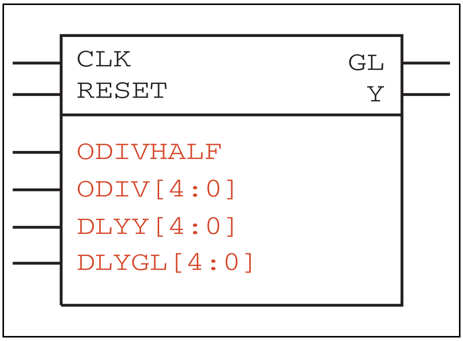
CLK, RESET, ODIVHALF, ODIV[4:0], DLYGL[4:0]

Output

GL

CLKDIVDLY1

Fusion



Function

Static clock with divider and/or delay with both global output driver and regular net driver.

Refer to the Clocking Resources of the Fusion datasheet for more information on this macro. It is available at http://www.actel.com.

Input

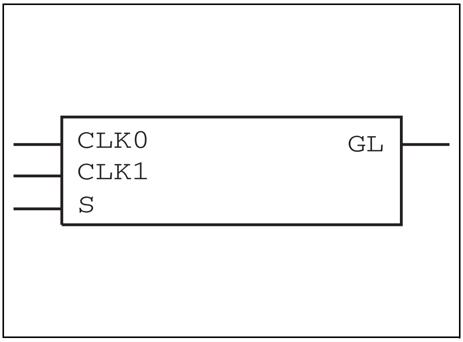
CLK, RESET, ODIVHALF, ODIV[4:0], DLYY[4:0], DLYGL[4:0]

Output

GL, Y

NGMUX

Fusion



Function

No-glitch MUX. NGMUX provides a special switching sequence between two asynchronous clock domains to avoid generating any unwanted narrow clock pulses.

Refer to the Clocking Resources of the Fusion datasheet for more information on this macro. It is available at http://www.actel.com.

Input

CLK0, CLK1, S

Output

GL

Transition S from high to low to initiate a switch to CLK0, and from low to high to initiate a switch to CLK1.

The output of NGMUX is undefined if S switches again before the previous switch operation has completed.

A

Product Support

Actel backs its products with various support services including Customer Service, a Customer Technical Support Center, a web site, an FTP site, electronic mail, and worldwide sales offices. This appendix contains information about contacting Actel and using these support services.

Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

From Northeast and North Central U.S.A., call **650.318.4480**
From Southeast and Southwest U.S.A., call **650. 318.4480**
From South Central U.S.A., call **650.318.4434**
From Northwest U.S.A., call **650.318.4434**
From Canada, call **650.318.4480**
From Europe, call **650.318.4252** or **+44 (0)1276.401500**
From Japan, call **650.318.4743**
From the rest of the world, call **650.318.4743**
Fax, from anywhere in the world **650. 318.8044**

Actel Customer Technical Support Center

Actel staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions. The Customer Technical Support Center spends a great deal of time creating application notes and answers to FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

Actel Technical Support

Visit the Actel Customer Support website (www.actelcom/.custsup/search.html) for more information and support. Many answers available on the searchable web resource include diagrams, illustrations, and links to other resources on the Actel web site.

Website

You can browse a variety of technical and non-technical information on Actel's home page, at www.actel.com.

Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center from 7:00 A.M. to 6:00 P.M., Pacific Time, Monday through Friday. Several ways of contacting the Center follow:

Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is tech@actel.com.

Phone

Our Technical Support Center answers all calls. The center retrieves information, such as your name, company name, phone number and your question, and then issues a case number. The Center then forwards the information to a queue where the first available application engineer receives the data and returns your call. The phone hours are from 7:00 A.M. to 6:00 P.M., Pacific Time, Monday through Friday. The Technical Support numbers are:

650.318.4460
800.262.1060

Customers needing assistance outside the US time zones can either contact technical support via email (tech@actel.com) or contact a local sales office. Sales office listings can be found at www.actel.com/contact/offices/index.html.

For more information about Actel's products, visit our website at http://www.actel.com

***Actel Corporation** • 2061 Stierlin Court • Mountain View, CA 94043 USA*
Customer Service: 650.318.1010 • Customer Applications Center: 800.262.1060

***Actel Europe Ltd**. • River Court, Meadows Business Park, Station Approach • Blackwater, Surrey GU17 9AB • UK*
Phone +44 (0) 1276 609 300 • Fax +44 (0) 1276 609 340

***Actel Japan** • EXOS Ebisu Bldg. 4F • 1-24-14 Ebisu Shibuya-ku • Tokyo 150 • Japan*
Phone +81.03.3445.7671 • Fax +81.03.3445.7668 • www.jp.actel.com

***Actel Hong Kong** • Suite 2114, Two Pacific Place • 88 Queensway, Admiralty Hong Kong*
Phone +852 2185 6460 • Fax +852 2185 6488 • www.actel.com.cn