LPM Quick Reference Guide

December 1996

About this Quick Reference Guide

December 1996

The ***LPM Quick Reference Guide*** provides information on functions in the library of parameterized modules (LPM) and on custom parameterized functions created by Altera®.

How to Contact Altera

For additional information about Altera products, consult the sources shown in Table 1.

Table 1. How to Contact Altera

Information Type	Access	USA & Canada	All Other Locations
Literature	Altera Express	(800) 5-ALTERA	(408) 894-7850
	Altera Literature Services	(888) 3-ALTERA, *Note (1)*	(888) 3-ALTERA; lit_req@altera.com, *Note (1)*
Non-technical customer service	Telephone hotline	(800) SOS-EPLD	(408) 894-7000
	Fax	(408) 954-8186	(408) 954-8186
Technical support	Telephone hotline (8:00 a.m. to 5 p.m. Pacific Time)	(800) 800-EPLD	(408) 894-7000, *Note (1)*
	Fax	(408) 954-0348	(408) 954-0348, *Note (1)*
	Bulletin board service	(408) 954-0104	(408) 954-0104
	Electronic mail	sos@altera.com	sos@altera.com
	FTP site	ftp.altera.com	ftp.altera.com
	CompuServe	go altera	go altera
General product information	Telephone	(408) 894-7104	(408) 894-7104, *Note (1)*
	World-wide web	http://www.altera.com	http://www.altera.com

Note:

(1) You can also contact your local Altera sales office or sales representative. See "Altera Sales Offices" in this quick reference guide.

Typographic Conventions

This ***LPM Quick Reference Guide*** uses the typographic conventions shown in Table 2.

Table 2. LPM Quick Reference Guide Conventions

Visual Cue	Meaning
Bold italics	Book titles are shown in bold italics, with initial capital letters. Example: ***LPM Quick Reference Guide***
"Subheading Title"	Subheadings within a quick reference guide section and titles of MAX+PLUS II Help topics are shown in quotation marks. Example: "Altera Sales Offices"
`Courier font`	Function names and port names are shown in lowercase Courier. For example: `lpm_and`, `data[]` Parameter names are shown in uppercase Courier. For example: `LPM_WIDTH`, `LPM_DIRECTION`

Contents

December 1996

Introduction

December 1996

Overview

Digital logic designers today must create designs consisting of tens-of-thousands of gates while meeting increased pressure to shorten time-to-market. At the same time, designers must maintain architecture-independence without sacrificing silicon efficiency.

Meeting these requirements with today's EDA software tools is not easy. Schematic-based design entry provides superior efficiency but implements architecture-dependent, low-level functions. High-level hardware description languages (HDLs) offer architecture-independence, but offer reduced silicon efficiency and performance.

Because a standard set of functions supported by all EDA and integrated circuit (IC) vendors was not previously available, bridging the gap between architecture-independence and efficiency was difficult. However, with the introduction of EDA software tools that support the library of parameterized modules (LPM), designers can now create architecture-independent designs that have high silicon efficiency.

History of LPM

The LPM standard was proposed in 1990 to enable efficient mapping of digital designs to diverse architectures such as programmable logic devices (PLDs), gate arrays, and standard cells. The LPM was accepted as an Electronic Industries Association (EIA) Interim standard in April 1993 as an adjunct standard to the Electronic Design Interface Format (EDIF), an industry-standard syntax that describes a structural netlist. EDIF can be used to transfer designs between the different software tools of EDA vendors and from EDA tools to IC tools. LPM functions describe the logical operation of the netlist. LPM functions used in a design can be directly passed to the IC vendor's design implementation software through an EDIF netlist file. Before the arrival of the LPM standard, each EDIF netlist would typically contain architecture-specific logic functions, which made architecture-independent design impossible.

LPM functions are compatible with any text or graphic design entry tool, and are supported by Altera® through MAX+PLUS® II and major EDA tool vendors, including Cadence, Exemplar, Mentor Graphics, MINC, Summit Design, Synopsys, VeriBest, and Viewlogic. Altera has supported the standard since 1993, and many other silicon companies will support the LPM standard by the end of 1997.

LPM Features

The primary objective for the LPM is to enable architecture-independent design without sacrificing efficiency. The LPM meets the following key criteria:

- *Architecture-independent design entry*—Designers can work with LPM functions during design entry and verification without specifying the target architecture. Design entry and simulation tools remain architecture-independent, relying on the synthesis or fitting tools to efficiently map the design to various architectures.
- *Efficient design mapping*—The LPM allows designers to create architecture-independent designs without sacrificing efficiency. The IC vendor is responsible for the mapping of LPM functions; thus, optimum solutions are guaranteed.
- *Tool-independent design entry*—The LPM enables designers to migrate designs between EDA tools while maintaining a high-level logic description of the functions. For example, designers can use one vendor's tool for logic synthesis and another vendor's tool for logic simulation.
- *Specification of a complete design*—LPM functions completely specify the digital logic for any design. Designers can create new functions with LPM functions.

LPM Functions

The LPM presently contains 25 functions. Despite its small size, the LPM can duplicate the functionality of other design libraries that contain many more functions; each function contains parameters that allow it to expand in many dimensions. For example, the `lpm_counter` function allows the user to create counters with widths ranging from 1 to 256 bits.

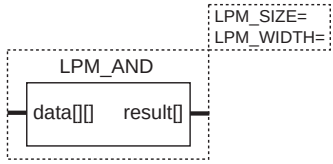
In addition to width, the user can specify the features and functionality of the counter. For example, parameters indicate whether the counter counts up or down, or loads synchronously or asynchronously. Thus, the single `lpm_counter` function can replace over thirty 74-series counters.

Gate Functions

December 1996

lpm_and

Parameterized AND Gate



Ports

Name	Type	Required	Description
`data[][]`	Input	Yes	Data input to the AND gate(s). This port consists of `LPM_SIZE` buses, each `LPM_WIDTH` wide.
`result[]`	Output	Yes	Each `result[]` bit is the AND of all `data[LPM_SIZE-1..0][]` inputs. This port is `LPM_WIDTH` wide.

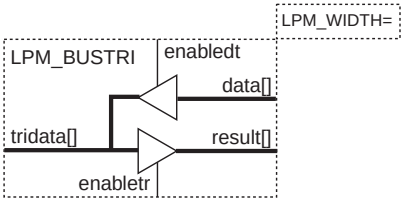
Parameters

Name	Required	Value	Description
`LPM_SIZE`	Yes	Integer > 0	Number of inputs to each AND gate.
`LPM_WIDTH`	Yes	Integer > 0	Width of the `result[]` port. Number of AND gates.

lpm_bustri

Parameterized Tri-State Buffer

The `lpm_bustri` function can be used to create both unidirectional and bidirectional tri-state bus controllers.



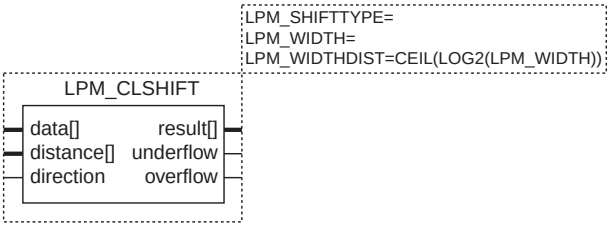
Ports

Name	Type	Required	Description
`data[]`	Input	Yes	Data input to the `tridata[]` bus. This port is `LPM_WIDTH` wide.
`enabletr`	Input	No	If high, enables `tridata[]` onto the `result[]` bus. Required if `result[]` is present (default = 0).
`enabledt`	Input	Yes	If high, enables `data[]` onto the `tridata[]` bus.
`result[]`	Output	No	This port is `LPM_WIDTH` wide.
`tridata[]`	Bidirectional	Yes	Bidirectional bus signal. This port is `LPM_WIDTH` wide.

Parameters

Name	Required	Value	Description
`LPM_WIDTH`	Yes	Integer > 0	Width of the `data[]`, `result[]`, and `tridata[]` ports.

lpm_clshift



Parameterized Combinatorial Logic Shifter or Barrel Shifter

The `lpm_clshift` function performs logical, rotational, or arithmetic combinatorial shifting. The direction and distance of the shifting are user-controllable.

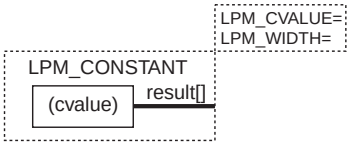
Ports

Name	Type	Required	Description
`data[]`	Input	Yes	Data to be shifted. This port is `LPM_WIDTH` wide.
`distance[]`	Input	Yes	Number of positions to shift `data[]` in the direction specified by the `direction` port. This port is `LPM_WIDTHDIST` wide.
`direction`	Input	No	Direction of shift. Low = left (toward the most significant bit (MSB)), high = right (toward the least significant bit (LSB)). Default value is 0 (low) = left (toward the MSB).
`result[]`	Output	Yes	Shifted data. This port is `LPM_WIDTH` wide.
`underflow`	Output	No	Logical or arithmetic underflow. If `"ROTATE"` is specified as the `LPM_SHIFTTYPE` parameter value and `overflow` and `underflow` are connected, the output of `overflow` and `underflow` will be undefined logic levels.
`overflow`	Output	No	Logical or arithmetic overflow. If `"ROTATE"` is specified as the `LPM_SHIFTTYPE` parameter value and `overflow` and `underflow` are connected, the output of `overflow` and `underflow` will be undefined logic levels.

Parameters

Name	Required	Value	Description
`LPM_SHIFTTYPE`	No	`"LOGICAL"` `"ROTATE"` `"ARITHMETIC"`	The sign bit is extended for `"ARITHMETIC"` right shifts. For a `"LOGICAL"` right shift, 0s are always shifted into the MSB or LSB. The default value is `"LOGICAL"`.
`LPM_WIDTH`	Yes	Integer > 1	Width of the `data[]` and `result[]` ports.
`LPM_WIDTHDIST`	Yes	Integer > 0	Width of the `distance[]` port. The `distance[]` port values normally range from 0, which is “no shift,” to (`LPM_WIDTH` – 1), which is the maximum meaningful shift. The typical value assigned to `LPM_WIDTHDIST` is “the smallest integer not less than $\log_2$(`LPM_WIDTH`)” or ceil($\log_2$(`LPM_WIDTH`)). Any `distance[]` port value greater than (`LPM_WIDTH` – 1) results in an undefined logic level.

lpm_constant



Parameterized Constant Generator

The `lpm_constant` function applies a constant to a bus. This function is useful for comparisons and arithmetic functions that operate on a constant value.

Ports

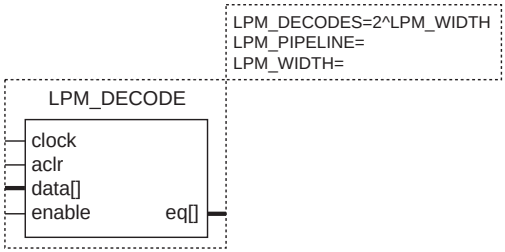
Name	Type	Required	Description
`result[]`	Output	Yes	Value specified by the argument to `LPM_CVALUE`. This port is `LPM_WIDTH` wide. The `LPM_CVALUE` parameter is truncated or zero-padded to `LPM_WIDTH` bits.

Parameters

Name	Required	Value	Description
`LPM_CVALUE`	Yes	Integer ≥ 0	Constant value to be driven out on the `result[]` port. If `LPM_CVALUE` cannot be represented in `LPM_WIDTH` bits, the `result[]` port drives the value `LPM_CVALUE` mod $2^{\texttt{LPM\_WIDTH}}$.
`LPM_WIDTH`	Yes	Integer > 0	Width of the `result[]` port.

lpm_decode

Parameterized Decoder



Ports

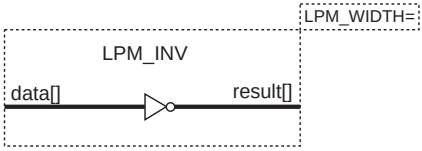
Name	Type	Required	Description
clock	Input	No	Clock for pipelined usage. The clock port provides pipelined operation for lpm_decode. For LPM_PIPELINE values other than 0 (default value), the clock port must be connected.
aclr	Input	No	Asynchronous clear for pipelined usage. The pipeline initializes to undefined. The aclr port can be used at any time to reset the pipeline to all 0s, asynchronously to the clock.
data[]	Input	Yes	Data input. Treated as an unsigned binary-encoded number. This port is LPM_WIDTH wide.
enable	Input	No	Enable. All outputs are low when this port is inactive. If absent, the default value is active (high).
eq[]	Output	Yes	Output of the decoder. This port is LPM_DECODES wide. If data[] ≥ LPM_DECODES, all eq[] outputs are 0.

Parameters

Name	Required	Value	Description
LPM_DECODES	Yes	$2^{LPM\_WIDTH} \geq$ Integer > 0	Number of explicit decoder outputs. The default value is 2^{LPM_WIDTH}.
LPM_PIPELINE	No	Integer ≥ 0	Specifies the number of clock cycles of latency associated with the eq[] output. A value of zero (0) indicates that no latency exists, and that a purely combinatorial function will be instantiated. The default value is 0 (non-pipelined).
LPM_WIDTH	Yes	Integer > 0	Width of the input value to be decoded.

lpm_inv

Parameterized Inverter



Ports

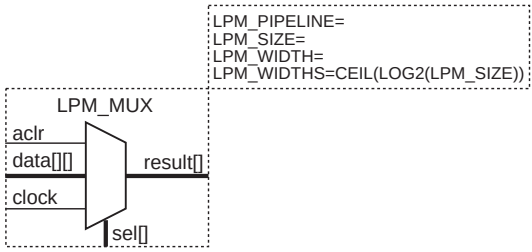
Name	Type	Required	Description
data[]	Input	Yes	Data input to the lpm_inv function. This port is LPM_WIDTH wide.
result[]	Output	Yes	Inverted result. This port is LPM_WIDTH wide.

Parameters

Name	Required	Value	Description
LPM_WIDTH	Yes	Integer > 0	Width of the data[] and result[] ports.

lpm_mux

Parameterized Multiplexer



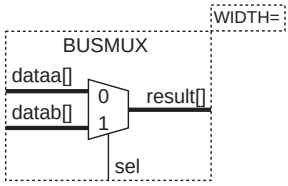
Ports

Name	Type	Required	Description
aclr	Input	No	Asynchronous clear for pipelined usage. The pipeline initializes to undefined. The aclr port can be used at any time to reset the pipeline to all 0s, asynchronously to the clock.
data[][]	Input	Yes	Data input. This port consists of LPM_SIZE buses, each LPM_WIDTH wide.
clock	Input	Yes	Clock for pipelined usage. The clock port provides pipelined operation for lpm_mux. For LPM_PIPELINE values other than 0 (default value), the clock port must be connected.
sel[]	Input	Yes	Selects one of the input buses. This port is LPM_WIDTHS wide.
result[]	Output	Yes	Selected input port. This port is LPM_WIDTH wide.

Parameters

Name	Required	Value	Description
LPM_PIPELINE	No	Integer ≥ 0	Specifies the number of clock cycles of latency associated with the result[] output. A value of zero (0) indicates that no latency exists, and that a purely combinatorial function will be instantiated. The default value is 0 (non-pipelined).
LPM_SIZE	Yes	2^{LPM_WIDTHS} ≥ Integer > 1	Number of inputs to each multiplexer. Number of input buses.
LPM_WIDTH	Yes	Integer > 0	Width of the data[][] and result[] ports.
LPM_WIDTHS	Yes	Integer > 0	Width of the sel[] port. The default value is ceil($\log_2$(LPM_SIZE)).

busmux



Parameterized Multiplexer

The busmux function is an Altera-provided function derived from lpm_mux and is intended to simplify the use of lpm_mux in Graphic Design Files (**.gdf**). The busmux function is an instance of lpm_mux with LPM_SIZE set to 2.

Ports

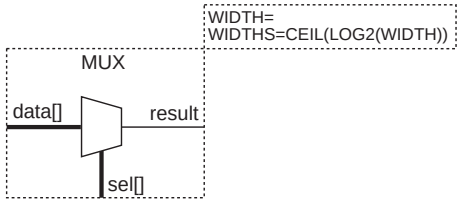
Name	Type	Required	Description
dataa[]	Input	Yes	Data input to the busmux. This port is WIDTH wide.
datab[]	Input	Yes	Data input to the busmux. This port is WIDTH wide.
sel	Input	Yes	Selects one of the ports.
result[]	Output	Yes	The selected input port. This port is WIDTH wide.

Parameters

Name	Required	Value	Description
WIDTH	Yes	Integer > 0	Width of the dataa[], datab[], and result[] ports.

mux

Parameterized Multiplexer



The mux function is an Altera-provided function derived from lpm_mux and is intended to simplify the use of lpm_mux in GDFs. The mux function is an instance of lpm_mux with LPM_WIDTH set to 1.

Ports

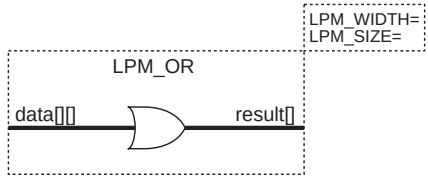
Name	Type	Required	Description
data[]	Input	Yes	Data input to the mux. This port is WIDTH wide.
sel[]	Input	Yes	Selects one of the ports. This port is WIDTHS wide.
result	Output	Yes	The selected input port. This port is 1 bit wide.

Parameters

Name	Required	Value	Description
WIDTH	Yes	Integer > 0	Width of the data[] port.
WIDTHS	Yes	Integer > 0	Width of the sel[] port. The default value is ceil($\log_2$(WIDTH)).

lpm_or

Parameterized OR Gate



Ports

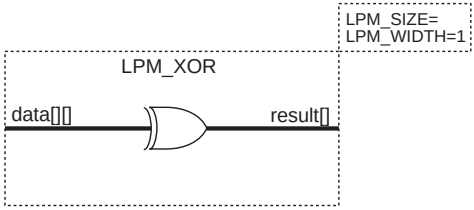
Name	Type	Required	Description
data[][]	Input	Yes	Data input to the OR gate(s). This port consists of LPM_SIZE buses, each LPM_WIDTH wide.
result[]	Output	Yes	Result of OR operators. This port is LPM_WIDTH wide.

Parameters

Name	Required	Value	Description
LPM_WIDTH	Yes	Integer > 0	Width of the data[][] and result[] ports. Number of OR gates.
LPM_SIZE	Yes	Integer > 0	Number of inputs to each OR gate. Number of input buses.

lpm_xor

Parameterized XOR Gate



Ports

Name	Type	Required	Description
data[][]	Input	Yes	Data input to the XOR gate(s). This port consists of LPM_SIZE buses, each LPM_WIDTH wide.
result[]	Output	Yes	Each result[] bit is the XOR of LPM_SIZE bits. This port is LPM_WIDTH wide.

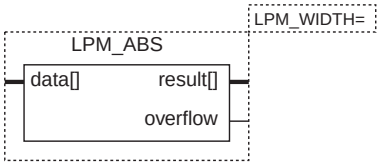
Parameters

Name	Required	Value	Description
LPM_SIZE	Yes	Integer > 0	Number of inputs to each XOR gate. Number of input buses.
LPM_WIDTH	Yes	Integer > 0	Width of the data[][] and result[] ports. Number of XOR gates.

December 1996

lpm_abs

Parameterized Absolute Value



Ports

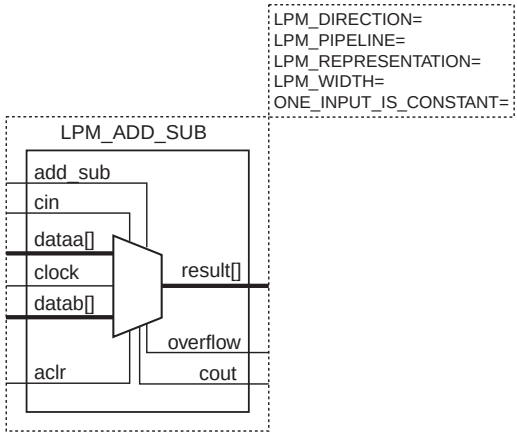
Name	Type	Required	Description
data[]	Input	Yes	Signed number. This port is LPM_WIDTH wide.
result[]	Output	Yes	Absolute value of data[]. This port is LPM_WIDTH wide.
overflow	Output	No	High if data[] = $-2^{(LPM\_WIDTH - 1)}$. Two's complement allows one more negative number than positive. The overflow port detects that singular instance and goes high to indicate that no positive equivalent exists.

Parameters

Name	Required	Value	Description
LPM_WIDTH	Yes	Integer ≥ 0	Width of the data[] and result[] ports.

lpm_add_sub

Parameterized Adder/Subtractor



Ports

Name	Type	Required	Description
add_sub	Input	No	If the input is high, the operation = dataa[] + datab[]. If the input is low, the operation = dataa[] – datab[]. This port cannot be used if the LPM_DIRECTION parameter is used. If both the add_sub port and the LPM_DIRECTION parameter are omitted, the operation defaults to "ADD".
cin	Input	No	Carry-in to the low-order bit. If the operation is "ADD", Low = 0 and High = +1. If the operation is "SUB", Low = –1 and High = 0. If omitted, the default is 0 (i.e., no affect on "ADD" or "SUB" operations).
dataa[]	Input	Yes	Augend/Minuend. This port is LPM_WIDTH wide.
clock	Input	No	Clock for pipelined usage. The clock port provides pipelined operation of lpm_add_sub. For LPM_PIPELINE values other than 0 (default value), the clock port must be connected.
datab[]	Input	Yes	Addend/Subtrahend. This port is LPM_WIDTH wide.
aclr	Input	No	Asynchronous clear for pipelined usage. The pipeline initializes to undefined. The aclr port can be used at any time to reset the pipeline to all 0s, asynchronously to clock.
result[]	Output	Yes	dataa[] + or – datab[] + or – cin. This port is LPM_WIDTH wide.
overflow	Output	No	Result exceeds available precision. If overflow is used, cout cannot be used. The overflow signal has a physical interpretation as the XOR of the carry into the MSB with the carry out of the MSB. The overflow signal is only meaningful when the LPM_REPRESENTATION parameter is set to "SIGNED". *Note (1)*
cout	Output	No	Carry-out (borrow-in) of the MSB. If overflow is used, cout cannot be used. The cout signal has a physical interpretation as the carry-out (borrow-in) of the MSB and is most meaningful for detecting overflow in "UNSIGNED" operations. *Note (2)*

Parameters

Name	Required	Value	Description
LPM_DIRECTION	No	"ADD" "SUB" "DEFAULT"	The add_sub port cannot be used if the LPM_DIRECTION parameter is has a value other than "DEFAULT". The default value is "DEFAULT".
LPM_PIPELINE	No	Integer ≥ 0	Specifies the number of clock cycles of latency associated with the result[] output. A value of zero (0) indicates that no latency exists, and that a purely combinatorial function will be instantiated. The default value is 0 (non-pipelined).
LPM_REPRESENTATION	No	"SIGNED" "UNSIGNED"	Type of addition performed. The default value is "SIGNED".
LPM_WIDTH	Yes	Integer > 0	Width of the dataa[], datab[], and result[] ports.
ONE_INPUT_IS_CONSTANT	No	"YES" "NO"	Provides greater optimization if an input is constant. The default value is "NO".

Notes:

(1) The following table describes the overflow port during "ADD" and "SUB" operations.

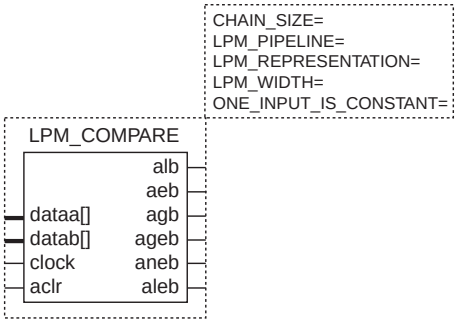
Value	ADD Operation	SUB Operation
"UNSIGNED"	Not meaningful.	Not meaningful.
"SIGNED"	(dataa + datab + cin) > $2^{(LPM\_WIDTH - 1) - 1}$ or (dataa + datab + cin) > $2^{(LPM\_WIDTH - 1)}$	(dataa – datab – cin) > $2^{(LPM\_WIDTH - 1) - 1}$ or (dataa – datab – cin) > $2^{(LPM\_WIDTH - 1)}$

(2) The following table describes the cout port during "ADD" and "SUB" operations.

Value	ADD Operation	SUB Operation
"UNSIGNED"	dataa + datab + cin > $2^{(LPM\_WIDTH - 1) - 1}$	Normal subtract. However, if cout = 0, then (dataa – datab – cin) < 0.
"SIGNED"	Normal result of adding two negative numbers, or possible overflow.	Normal result when subtracting a positive number from a negative number, or possible overflow.

lpm_compare

Parameterized Comparator



Ports

Name	Type	Required	Description
dataa[]	Input	Yes	The datab[] signal is compared to this value. This port is LPM_WIDTH wide.
datab[]	Input	Yes	Value to be compared to dataa[]. This port is LPM_WIDTH wide.
clock	Input	No	Clock for pipelined usage. The clock port provides pipelined operation for lpm_compare. For LPM_PIPELINE values other than 0 (default value), the clock port must be connected.
aclr	Input	No	Asynchronous clear for pipelined usage. The pipeline initializes to undefined. The aclr port can be used at any time to reset the pipeline to all 0s, asynchronously to clock.
alb	Output	No	High (1) if dataa[] < datab[]. One of the alb, aeb, agb, ageb, aleb, or aneb outputs must be present.
aeb	Output	No	High (1) if dataa[] = datab[]. One of the alb, aeb, agb, ageb, aleb, or aneb outputs must be present.
agb	Output	No	High (1) if dataa[] > datab[]. One of the alb, aeb, agb, ageb, aleb, or aneb outputs must be present.
ageb	Output	No	High (1) if dataa[] ≥ datab[]. One of the alb, aeb, agb, ageb, aleb, or aneb outputs must be present.
aneb	Output	No	High (1) if dataa[] ≠ datab[]. One of the alb, aeb, agb, ageb, aleb, or aneb outputs must be present.
aleb	Output	No	High (1) if dataa[] ≤ datab[]. One of the alb, aeb, agb, ageb, aleb, or aneb outputs must be present.

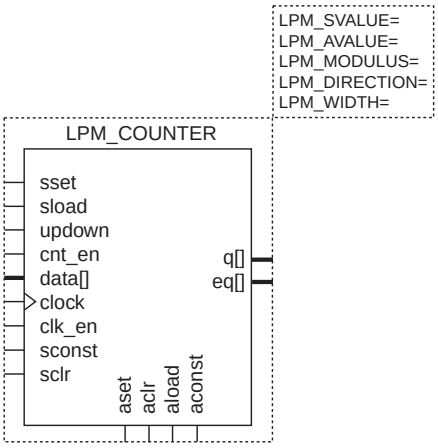
Parameters

Name	Required	Value	Description
CHAIN_SIZE	No	Integer > 0	Maximum allowable length of carry chains or cascade chains in FLEX 10K and FLEX 8000 devices. The default value is 8. In MAX+PLUS II, this value overrides the value of the *Max. Auto Length* option for the Carry Chain or Cascade Chain logic option(s) in the global project logic synthesis style, which is specified with the **Global Project Logic Synthesis** dialog box (Assign menu). For other device families, varying the CHAIN_SIZE parameter will provide different speed/size combinations—setting smaller values for CHAIN_SIZE generally results in faster and larger comparators and vice versa. For more information, contact Altera Applications.
LPM_PIPELINE	No	Integer ≥ 0	Specifies the number of clock cycles of latency associated with the alb, aeb,agb, ageb, aneb, and aleb outputs. A value of zero (0) indicates that no latency exists, and that a purely combinatorial function will be instantiated. Default value is 0 (non-pipelined).
LPM_REPRESENTATION	No	"SIGNED" "UNSIGNED"	Type of comparison performed. The default value is "UNSIGNED".
LPM_WIDTH	Yes	Integer > 0	Width of the dataa[] and datab[] ports.
ONE_INPUT_IS_CONSTANT	No	"YES" "NO"	Provides greater optimization if an input is constant. The default value is "NO".

lpm_counter

Parameterized Counter

The lpm_counter function is a full-featured counter with loading, up / down control, clock, and count enabling and clearing.



Ports (Part 1 of 2)

Name	Type	Required	Description
sset	Input	No	Synchronous set input. Sets the counter on the next active clock edge. Default = 0. Sets q[] outputs to all 1s, or to the value specified by LPM_SVALUE. If both sset and sclr are used and both are asserted, sclr is dominant. For outputs such as q[] and eq[], sset affects the output before polarity is applied.
sload	Input	No	Synchronous load input. Loads the counter with data on the next active clock edge. Default = 0. If sload is used, data[] must be connected.
updown	Input	No	Controls the direction of the count. High (1) = count up. Low (0) = count down. Default = up (1). If the LPM_DIRECTION parameter is used, the updown port cannot be connected. If LPM_DIRECTION is not used, the updown port is optional.
cnt_en	Input	No	Count enable input. Disables count when low (0) without affecting sload, sset, or sclr. Default = 1.
data[]	Input	No	Parallel data input to the counter. This port is LPM_WIDTH wide. Uses aload and/or sload.
clock	Input	Yes	Positive-edge-triggered clock.
clk_en	Input	No	Clock enable input. Enables all synchronous activities. Default = 1.
sconst	Input	No	This port is provided only for backwards-compatibility in MAX+PLUS II pre-version 6.0 designs. Altera does not recommend using this port for new designs.
sclr	Input	No	Synchronous clear input. Clears the counter on the next active clock edge. Default = 0. If both sset and sclr are used and both are asserted, sclr is dominant. For outputs such as q[] and eq[], sclr affects the output before polarity is applied.

Ports (Part 2 of 2)

Name	Type	Required	Description
aset	Input	No	Asynchronous set input. Default = 0. Sets q[] outputs to all 1s, or to the value specified by LPM_AVALUE. If both aset and aclr are used and both are asserted, aclr is dominant. For outputs such as q[] and eq[], aset affects the output before polarity is applied.
aclr	Input	No	Asynchronous clear input. Default = 0. If both aset and aclr are used and both are asserted, aclr is dominant. For outputs such as q[] and eq[], aclr affects the output before polarity is applied.
aload	Input	No	Asynchronous load input. Asynchronously loads the counter with the value on the data input. Default = 0. If aload is used, data[] must be used.
aconst	Input	No	This port is provided only for backwards-compatibility in MAX+PLUS II pre-version 6.0 designs. Altera does not recommend using this port in new designs.
eq[]	Output	No	Counter decode output. Active high when the counter reaches the specified count value. Either the q[] port or eq[] port must be connected. Up to *c* eq ports can be used ($c \leq 15$). Only the 16 lowest count values are decoded. When the count value is *c*, the eq*c* output is set high (1). For example, when the count is 0, eq0 = 1, when the count is 1, eq1 = 1, and when the count is 15, eq15 = 1. Decoded outputs for count values of 16 or greater require external decoding. The eq*c* outputs are asynchronous.
q[]	Output	No	Data output from the counter. This port is LPM_WIDTH wide. Either q[] or at least one of the eq[] ports must be connected.

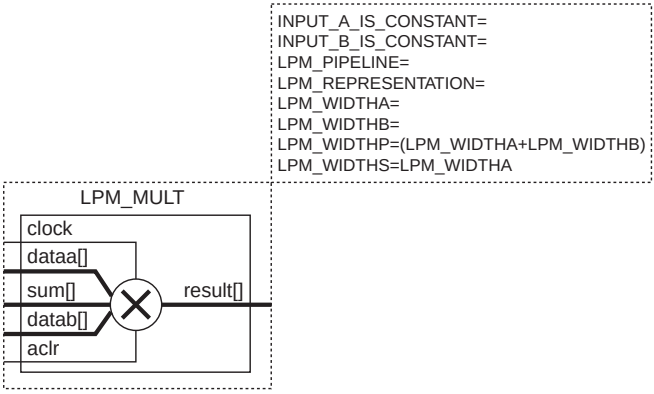
Parameters

Name	Required	Value	Description
LPM_SVALUE	No	Integer ≥ 0	Constant value that is loaded on the rising edge of clock when sset is high. Must be used if sset is used.
LPM_AVALUE	No	Integer ≥ 0	Constant value that is loaded when aset is high. This parameter must be used if aset is used. If the value specified is larger than the <*modulus*>, the behavior of the counter is an undefined logic level. The <*modulus*> is LPM_MODULUS, if present, or 2^{LPM_WIDTH}.
LPM_MODULUS	No	Integer > 1	The maximum count, plus one. Number of unique states in the counter's cycle. If the load value is larger than the LPM_MODULUS parameter, the behavior of the counter is not specified. The default value is 2^{LPM_WIDTH}.
LPM_DIRECTION	No	"UP" "DOWN" "DEFAULT"	If the LPM_DIRECTION parameter is used, the updown port cannot be connected. When the updown port is not connected, the count direction is "UP". In all other cases, the default value is "DEFAULT".
LPM_WIDTH	Yes	Integer > 0	Width of the input and output ports. If no output ports are specified, the value is the number of bits in the count.

lpm_mult

Parameterized Multiplier

The lpm_mult function allows two signed or unsigned numbers to be multiplied. In addition, the result of the multiplication can be added to a third number.



Ports

Name	Type	Required	Description
clock	Input	No	Clock for pipelined usage. The clock port provides pipelined operation for lpm_mult. For LPM_PIPELINE values other than 0 (default value), the clock port must be connected.
dataa[]	Input	Yes	Multiplicand. This port is LPM_WIDTHA wide.
sum[]	Input	No	Partial sum. This port is LPM_WIDTHS wide.
datab[]	Input	Yes	Multiplier. This port is LPM_WIDTHB wide.
aclr	Input	No	Asynchronous clear for pipelined usage. The pipeline initializes to undefined. The aclr port can be used at any time to reset the pipeline to all 0s, asynchronously to clock.
result[]	Output	Yes	This port is LPM_WIDTHP wide. If LPM_WIDTHP < max (LPM_WIDTHA + LPM_WIDTHB, LPM_WIDTHS) or (LPM_WIDTHS), only the LPM_WIDTHP MSBs are present.

Parameters

Name	Required	Value	Description
INPUT_A_IS_CONSTANT	No	"YES" "NO"	If dataa[] is connected to a constant value, setting the value of INPUT_A_IS_CONSTANT to "YES" optimizes the multiplier for resource usage and speed. The default value is "NO".
INPUT_B_IS_CONSTANT	No	"YES" "NO"	If datab[] is connected to a constant value, setting the value of INPUT_B_IS_CONSTANT to "YES" optimizes the multiplier for resource usage and speed. The default value is "NO".
LPM_PIPELINE	No	Integer ≥ 0	Specifies the number of clock cycles of latency associated with the result[] output. The default value of zero (0) indicates that no latency exists, and that a purely combinatorial function will be instantiated. The default value is 0 (non-pipelined).
LPM_REPRESENTATION	No	"SIGNED" "UNSIGNED"	Type of multiplication performed. The default value is "UNSIGNED".
LPM_WIDTHA	Yes	Integer > 0	Width of the dataa[] port.
LPM_WIDTHB	Yes	Integer > 0	Width of the datab[] port.
LPM_WIDTHP	Yes	Integer > 0	Width of the result[] port. The default is LPM_WIDTHA+LPM_WIDTHB.
LPM_WIDTHS	No	Integer > 0	Width of the sum[] port. If the sum[] port is not used, LPM_WIDTHS must be set to a value between 1 and LPM_WIDTHP, inclusive. The default value is LPM_WIDTHA.
LATENCY, *Note (1)*	No	Integer ≥ 0	Same as LPM_PIPELINE.

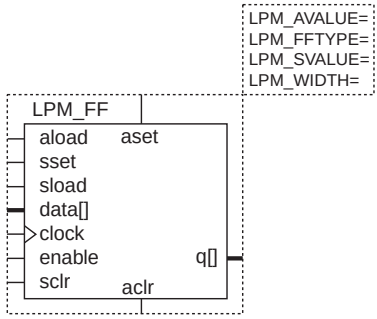
Note:

(1) The LATENCY parameter is provided only for backwards-compatibility with MAX+PLUS II pre-version 7.0 designs. For all new designs, you should use the LPM_PIPELINE parameter instead.

December 1996

lpm_ff

Parameterized D or T Flipflop



Ports

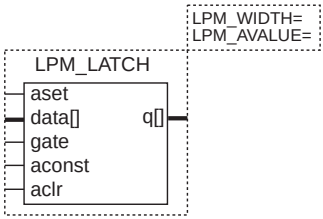
Name	Type	Required	Description
aload	Input	No	Asynchronous load input. Asynchronously loads the flipflop with the value on the data[] input. Default = 0. If aload is used, data[] must be used.
sset	Input	No	Synchronous set input. Sets the q[] outputs to the value specified by LPM_SVALUE, if that value is present, or sets the q[] outputs to all 1s. If both sset and sclr are used and both are asserted, sclr is dominant. The sset input affects the output q[] values before polarity is applied to the ports.
sload	Input	No	Synchronous load input. Loads the flipflop with the value on the data[] input on the next active clock edge. Default = 0. If sload is used, data[] must be used. For load operation, sload must be high (1) and enable must be high (1) or unconnected.
data[]	Input	No	T flipflop: toggle enable; D flipflop: data input. This port is LPM_WIDTH wide. If the data[] port is not used, at least one of the aset, aclr, sset, or sclr ports must be used.
clock	Input	Yes	Positive-edge-triggered clock.
enable	Input	No	Clock enable input. Default = 1.
sclr	Input	No	Synchronous clear input. If both sset and sclr are used and both are asserted, sclr is dominant. The sclr input affects the output q[] values before polarity is applied to the ports.
aset	Input	No	Asynchronous set input. Sets q[] outputs to the value specified by LPM_AVALUE, if that value is present, or sets the q[] outputs to all 1s.
aclr	Input	No	Asynchronous clear input. If both aset and aclr are used and both are asserted, aclr is dominant. The aclr input affects the output q[] values before polarity is applied to the ports.
q[]	Output	Yes	Data output from D flipflops. This port is LPM_WIDTH wide.

Parameters

Name	Required	Value	Description
LPM_AVALUE	Yes	Integer ≥ 0	Constant value that is loaded when aset is high. The default value is all 1s.
LPM_FFTYPE	No	"DFF" "TFF"	Type of flipflop. The default value is "DFF".
LPM_SVALUE	No	Integer ≥ 0	Constant value that is loaded on the rising edge of clock when sset is high. The default value is all 1s.
LPM_WIDTH	Yes	Integer > 0	Width of the data[] and q[] ports.

lpm_latch

Parameterized Latch



Ports

Name	Type	Required	Description
aset	Input	No	Asynchronous set input. Default = 0. Sets q[] outputs to the value specified by LPM_AVALUE, if that value is present. If no LPM_AVALUE is specified, aset will set the count to all 1s. If both aset and aclr are used and both are asserted, aclr is dominant. The aset and aclr inputs affect the output q[] values before polarity is applied to the ports.
data[]	Input	No	Data input to the D-type latch. This port is LPM_WIDTH wide. If the data[] port is not used, either aset or aclr must be used.
gate	Input	Yes	Latch enable input. High = flow-through, low = latch.
aconst	Input	No	This port is provided only for backwards-compatibility in MAX+PLUS II pre-version 6.0 designs. Altera does not recommend using this port in new designs.
aclr	Input	No	Asynchronous clear input. Default = 0. Sets the latch to all 0s. If both aset and aclr are used and both are asserted, aclr is dominant. The aset and aclr inputs affect the output q[] values before the polarity is applied to the ports.
q[]	Output	Yes	Data output from the latches. This port is LPM_WIDTH wide.

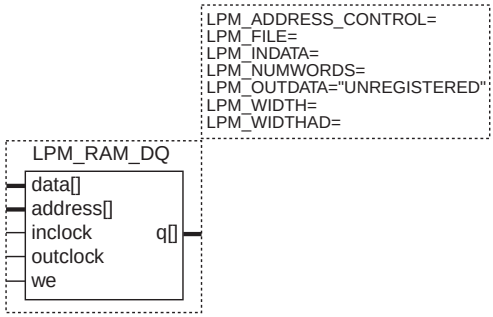
Parameters

Name	Required	Value	Description
LPM_WIDTH	Yes	Integer > 0	Width of the data[] and q[] ports.
LPM_AVALUE	No	Integer ≥ 0	Constant value that is loaded when aset is high. The default value of LPM_AVALUE is all 1s.

lpm_ram_dq

Parameterized Random Access Memory with Separate Input and Output Ports

The lpm_ram_dq function can be used as either synchronous or asynchronous random access memory. The lpm_ram_dq function has separate input and output data buses.



Ports

Name	Type	Required	Description
data[]	Input	Yes	Data input to memory. This port is LPM_WIDTH wide.
address[]	Input	Yes	Address input to the memory. This port is LPM_WIDTHAD wide.
inclock	Input	No	Synchronizes memory loading. If the inclock port is used, the we port acts as an enable for write operations synchronized to the rising edge of the inclock input. If the inclock port is not used, the we port acts as an enable for asynchronous write operations.
outclock	Input	No	Synchronizes q[] outputs from memory. The addressed memory content → q[] response is synchronous when the outclock port is connected, and asynchronous when it is not connected.
we	Input	Yes	Write enable input. When high, enables write operation to the memory. Required if inclock is not present. If only we is used, the data on the address[] port should not change while we is high. If the data on the address[] port changes while we is high, all memory locations that are addressed are overwritten with data[].
q[]	Output	Yes	Data output from the memory. This port is LPM_WIDTH wide.

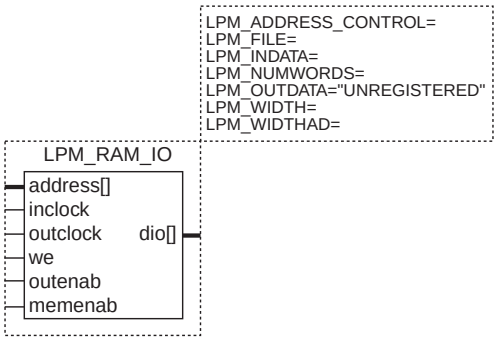
Parameters

Name	Required	Value	Description
LPM_ADDRESS_CONTROL	No	"REGISTERED" "UNREGISTERED"	Controls whether the address[] and we ports are registered. The default value is "REGISTERED".
LPM_FILE	No	"<*filename*>"	Name of the Memory Initialization File (**.mif**) or Hexadecimal File (**.hex**) containing RAM initialization data ("<*filename*>"). If omitted, contents default to all 0s.
LPM_INDATA	No	"REGISTERED" "UNREGISTERED"	Controls whether the data[] port is registered. The default value is "REGISTERED".
LPM_NUMWORDS	No	Integer > 0	Number of words stored in memory. In general, this value should be (but is not required to be): $2^{LPM\_WIDTHAD-1} < LPM\_NUMWORDS \leq 2^{LPM\_WIDTHAD}$. The default value is $2^{LPM\_WIDTHAD}$.
LPM_OUTDATA	No	"REGISTERED" "UNREGISTERED"	Controls whether the q[] and internal eq ports are registered. The default value is "REGISTERED".
LPM_WIDTH	Yes	Integer > 0	Width of the data[] and q[] ports.
LPM_WIDTHAD	Yes	Integer > 0	Width of the address[] port. LPM_WIDTHAD should be (but is not required to be): $\log_2$(LPM_NUMWORDS). If LPM_WIDTHAD is too small, some memory locations will not be addressable. If LPM_WIDTHAD is too large, the addresses that are too high will return undefined logic levels.

lpm_ram_io

Parameterized Random Access Memory with a Single I/O Port

The `lpm_ram_io` function can be used as either synchronous or asynchronous random access memory. The `lpm_ram_io` function has a bidirectional bus.



Ports

Name	Type	Required	Description
`address[]`	Input	Yes	Address input to the memory. This port is `LPM_WIDTHAD` wide. If `memenab` is used, it should be inactive when `address[]` is changing.
`inclock`	Input	No	Synchronizes memory loading. If the `inclock` port is used, the `we` port acts as an enable for write operations synchronized to the rising edge of the `inclock` input. If the `inclock` port is not used, the `we` port acts as an enable for asynchronous write operations.
`outclock`	Input	No	Synchronizes `dio[]` from memory. The addressed memory content → `q[]` response is synchronous when the `outclock` port is connected, and asynchronous when it is not connected.
`we`	Input	Yes	Write enable input. Either `we` or `outenab` should be used. When high, enables write operations to the memory. If no clock ports are used, the data on the `address[]` port should not change when `we` is high (1). Required if `clock` is not present. If `we` is absent, the default value is enabled.
`outenab`	Input	No	Output enable input. High (1): `dio[]` from memory `address[]`. Low (0): Memory `address[]` from `dio[]`. Either `memenab` or `outenab` must be present.
`memenab`	Input	No	Memory output tri-state enable. Either `memenab` or `outenab` must be connected. If `memenab` is present, it should be inactive when `address[]` is changing.
`dio[]`	Bidirectional	Yes	Data port for the memory. This port is `LPM_WIDTH` wide.

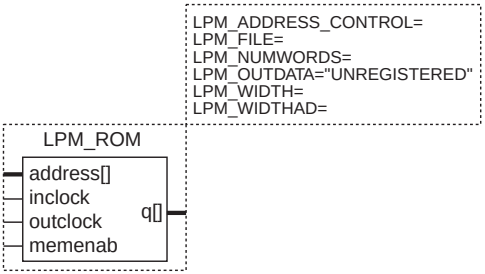
Parameters

Name	Required	Value	Description
LPM_ADDRESS_CONTROL	No	"REGISTERED" "UNREGISTERED"	Controls whether the address[], memenab, and we ports are registered. The default value is "REGISTERED".
LPM_FILE	No	"<*filename*>"	Name of the MIF or Hex File containing ROM initialization data ("<*filename*>"). If omitted, contents default to all 0s.
LPM_INDATA	No	"REGISTERED" "UNREGISTERED"	Controls whether the internal data port is registered. The default value is "REGISTERED".
LPM_NUMWORDS	No	Integer > 0	Number of words stored in memory. In general, this value should be (but is not required to be): $2^{LPM\_WIDTHAD-1} < LPM\_NUMWORDS \leq 2^{LPM\_WIDTHAD}$. The default value is $2^{LPM\_WIDTHAD}$.
LPM_OUTDATA	No	"REGISTERED" "UNREGISTERED"	Controls whether the dio[] port is registered. The default value is "REGISTERED".
LPM_WIDTH	Yes	Integer > 0	Width of dio[] and internal data and q ports.
LPM_WIDTHAD	Yes	Integer > 0	Width of the address[] port. LPM_WIDTHAD should be (but is not required to be) equal to $\log_2$(LPM_NUMWORDS). If LPM_WIDTHAD is too small, some memory locations will not be addressable. If it is too large, the addresses that are too high will return undefined logic levels.

lpm_rom

Parameterized Read-Only Memory

The lpm_rom function can be used as either synchronous or asynchronous read-only memory.



Ports

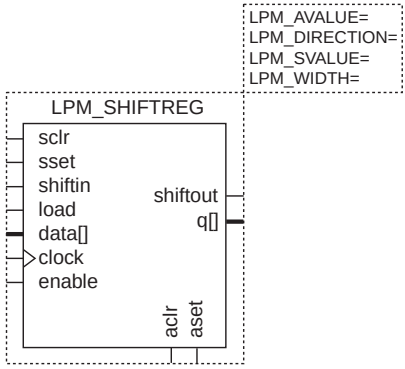
Name	Type	Required	Description
address[]	Input	Yes	Address input to the memory. This port is LPM_WIDTHAD wide.
inclock	Input	No	Clock for input registers. The address[] port is synchronous (registered) when the inclock port is connected, and is asynchronous (unregistered) when the inclock port is not connected.
outclock	Input	No	Clock for output registers. The addressed memory content → q[] response is synchronous when the outclock port is connected, and is asynchronous when it is not connected.
memenab	Input	No	Memory enable input. High = data output on q[], Low = high-impedance outputs.
q[]	Output	Yes	Memory output. This port is LPM_WIDTH wide.

Parameters

Name	Required	Value	Description
LPM_ADDRESS_CONTROL	No	"REGISTERED" "UNREGISTERED"	Controls whether the address[] port is registered. The default value is "REGISTERED".
LPM_FILE	Yes	"<*filename*>"	Name of the MIF or Hex File containing ROM initialization data ("<*filename*>").
LPM_NUMWORDS	No	Integer > 0	In general, this value should be (but is not required to be) $2^{LPM\_WIDTHAD-1} < LPM\_NUMWORDS \leq 2^{LPM\_WIDTHAD}$. The default value is $2^{LPM\_WIDTHAD}$.
LPM_OUTDATA	No	"REGISTERED" "UNREGISTERED"	Controls whether the q[] port is registered. The default value is "REGISTERED".
LPM_WIDTH	Yes	Integer > 0	Width of the q[] port.
LPM_WIDTHAD	Yes	Integer > 0	Width of the address[] port. LPM_WIDTHAD should be (but is not required to be) equal to $\log_2$(LPM_NUMWORDS). If LPM_WIDTHAD is too small, some memory locations will not be addressable. If it is too large, the addresses that are too high will return undefined logic levels.

lpm_shiftreg

Parameterized Shift Register



Ports

Name	Type	Required	Description
sclr	Input	No	Synchronous clear input. If both sset and sclr are used and both are asserted, sclr is dominant. The sclr input affects the output q[] values before polarity is applied to the ports.
sset	Input	No	Synchronous set input. Sets q[] outputs to the value specified by LPM_SVALUE, if that value is present, or sets the q[] outputs to all 1s. If both sset and sclr are used and both are asserted, sclr is dominant. The sset input affects the output q[] values before polarity is applied to the ports.
shiftin	Input	No	Serial shift data input. At least one of the data[], aset, aclr, sset, sclr, and/or shiftin ports must be used.
load	Input	No	Synchronous parallel load. High (1): load operation; low (0): shift operation. Default is low (0) shift operation. For parallel load operation, load must be high (1) and enable must be high or unconnected.
data[]	Input	No	Data input to the shift register. This port is LPM_WIDTH wide. At least one of the data[], aset, aclr, sset, sclr and/or shiftin ports must be used.
clock	Input	Yes	Positive-edge-triggered clock. Default = 1.
enable	Input	No	Clock enable input. The shift options also use the enable input for the clock enable. For serial operation, both shiftin and enable must be high.
aclr	Input	No	Asynchronous clear input. If both aset and aclr are used and both are asserted, aclr is dominant. The aclr input affects the output q[] values before polarity is applied to the ports.
aset	Input	No	Asynchronous set input. Sets q[] outputs to the value specified by LPM_AVALUE, if that value is present, or sets the q[] outputs to all 1s. If both aset and aclr are used and both are asserted, aclr is dominant. The aset input affects the output q[] values before polarity is applied to the ports.
shiftout	Output	No	Serial shift data output. This port is LPM_WIDTH wide. Either q[] or shiftout or both must be used.
q[]	Output	No	Data output from the shift register. This port is LPM_WIDTH wide. Either q[] or shiftout or both must be used.

Parameters

Name	Required	Value	Description
LPM_AVALUE	No	Integer > 0	Constant value that is loaded when aset is high.
LPM_DIRECTION	No	"LEFT" "RIGHT"	Direction of the shift register. The default value is "LEFT".
LPM_SVALUE	No	Integer ≥ 0	Constant value that is loaded on the rising edge of clock when sset is high. The default value is all 1s.
LPM_WIDTH	Yes	Integer > 0	Width of the data[] and q[] ports.

Notes:

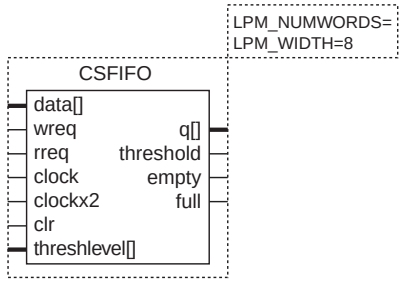
Custom Parameterized Functions

December 1996

csfifo

Cycle-Shared FIFO

The csfifo function is a custom function that provides a cycle-shared FIFO with both dynamic and static user threshold level controls. It also offers empty and full flag outputs.



Ports

Name	Type	Required	Description
data[]	Input	Yes	Data input to the csfifo. This port is LPM_WIDTH wide.
wreq	Input	Yes	Write request.
rreq	Input	Yes	Read request.
clock	Input	Yes	Positive-edge-triggered clock.
clockx2	Input	Yes	Positive-edge-triggered clock.
clr	Input	No	Resets csfifo to empty.
threshlevel[]	Input	No	Level (number of words) that the threshold output signal asserts.
q[]	Output	Yes	Data output from csfifo. This port is LPM_WIDTH wide.
threshold	Output	No	Indicates that csfifo contains greater than the threshlevel[] number of words.
empty	Output	No	Indicates that csfifo is empty.
full	Output	No	Indicates that csfifo is full.

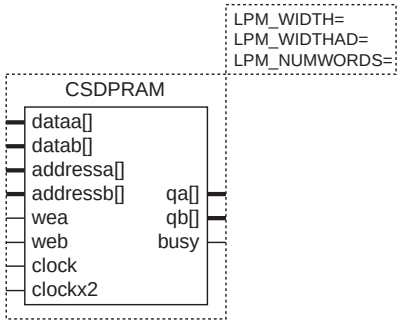
Parameters

Name	Required	Value	Description
LPM_NUMWORDS	No	Integer > 0	Number of words stored in memory. In general, this value should be (but is not required to be): $2^{LPM\_WIDTH-1} < LPM\_NUMWORDS \leq 2^{LPM\_WIDTHAD}$. The default value is 2^{LPM_WIDTH}.
LPM_WIDTH	Yes	Integer > 0	Width of the data[] and q[] ports. The default value is 8.

csdpram

Parameterized Cycle-Shared Dual-Port RAM

The `csdpram` function is a custom function that has two address and two data ports that are cycle-shared. This function also provides a busy flag to indicate that the address on both ports is pointing to the same location and that a port will have priority.



Ports

Name	Type	Required	Description
`dataa[]`	Input	Yes	Data input to the memory. This port is `LPM_WIDTH` wide.
`datab[]`	Input	Yes	Data input to the memory. This port is `LPM_WIDTH` wide.
`addressa[]`	Input	Yes	Address input to the memory. This port is `LPM_WIDTHAD` wide.
`addressb[]`	Input	Yes	Address input to the memory. This port is `LPM_WIDTHAD` wide.
`wea`	Input	Yes	Write enable input.
`web`	Input	Yes	Write enable input.
`clock`	Input	Yes	Positive-edge-triggered clock.
`clockx2`	Input	Yes	Positive-edge-triggered clock.
`qa[]`	Output	Yes	Data output from the memory. This port is `LPM_WIDTH` wide.
`qb[]`	Output	Yes	Data output from the memory. This port is `LPM_WIDTH` wide.
`busy`	Output	No	Indicates that `addressa[]` = `addressb[]` and that `dataa[]` is writing data.

Parameters

Name	Required	Value	Description
`LPM_WIDTH`	Yes	Integer > 0	Width of the `dataa[]`, `datab[]`, `qa[]`, and `qb[]` ports.
`LPM_WIDTHAD`	Yes	Integer > 0	Width of the `addressa[]` and `addressb[]` ports. `LPM_WIDTHAD` should be (but is not required to be) equal to $\log_2$(`LPM_NUMWORDS`). If `LPM_WIDTHAD` is too small, some memory locations will not be addressable. If it is too large, the addresses that are too high will return undefined logic levels.
`LPM_NUMWORDS`	No	Integer > 0	Number of words stored in memory. In general, this value should be (but is not required to be): $2^{\texttt{LPM\_WIDTHAD}-1} < \texttt{LPM\_NUMWORDS} \leq 2^{\texttt{LPM\_WIDTHAD}}$. The default value is $2^{\texttt{LPM\_WIDTHAD}}$.

Notes:

Altera Sales Offices

December 1996

Altera Regional Offices

NORTHERN CALIFORNIA (CORPORATE HEADQUARTERS)
Altera Corporation
2610 Orchard Parkway
San Jose, CA 95134-2020
TEL: (408) 894-7000
FAX: (408) 433-3943
(408) 894-7755

Altera Corporation
2290 N. First Street, Suite 212
San Jose, CA 95131
TEL: (408) 894-7900
FAX: (408) 894-7979

SOUTHERN CALIFORNIA
Altera Corporation
15375 Barranca Parkway, Suite B-201
Irvine, CA 92618
TEL: (714) 450-0262
FAX: (714) 450-0263

Altera Corporation
Olympic Plaza Executive Center
11500 West Olympic Boulevard, Suite 400
Los Angeles, CA 90064
TEL: (310) 312-4507
FAX: (310) 312-4508

Altera Corporation
5355 Mira Sorrento Place, Suite 100
San Diego, CA 92121
TEL: (619) 597-7518
FAX: (619) 597-7418

ARIZONA
Altera Corporation
2390 E.Camelback Road, Suite 300
Phoenix, Arizona 85016
TEL: (602) 553-1090
FAX: (602) 553-1198

COLORADO
Altera Corporation
Denver Technology Center
7900 East Union Avenue, #1100
Denver, CO 80237
TEL: (303) 694-5352
FAX: (303) 694-5351

Altera Corporation
14142 Denver West Parkway, #200
Golden, CO 80401
TEL: (303) 216-0167
FAX: (303) 277-0429

GEORGIA
Altera Corporation
3675 Crestwood Parkway, Suite 400
Duluth, GA 30136
TEL: (770) 935-6070
FAX: (770) 935-6073

ILLINOIS
Altera Corporation
475 N. Martingale Road, Suite 420
Schaumburg, IL 60173
TEL: (847) 240-0313
FAX: (847) 240-0266

MARYLAND
Altera Corporation
9891 Broken Land Parkway, Suite 300
Columbia, MD 21046
TEL: (410) 312-5708
FAX: (410) 309-0720

MASSACHUSETTS
Altera Corporation
238 Littleton Road, Suite 207
Westford, MA 01886
TEL: (508) 392-1100
FAX: (508) 392-1157

MINNESOTA
Altera Corporation
2850 Metro Drive, Suite 250
Bloomington, MN 55425
TEL: (612) 851-7861
FAX: (612) 858-7258

NEW JERSEY
Altera Corporation
575 State Highway 28
Raritan, NJ 08869
TEL: (908) 526-9400
FAX: (908) 526-5471

NORTH CAROLINA
Altera Corporation
5511 Capital Center Drive, Suite 110
Raleigh, NC 27606
TEL: (919) 852-1004
FAX: (919) 852-0809

OHIO
Altera Corporation
7784 Reynolds Road
Mentor, OH 44060
TEL: (216) 946-8211
FAX: (216) 946-9411

Altera Regional Offices

(Continued)

OREGON
Altera Corporation
11000 S.W. Stratus Street, Suite 330-17
Beaverton, OR 97008
TEL: (503) 643-8447
FAX: (503) 644-2345

TEXAS
Altera Corporation
5080 Spectrum Drive, Suite 812W
Dallas, TX 75248
TEL: (972) 701-2330
FAX: (972) 701-2331

TEXAS *(continued)*
Altera Corporation
9430 Research Boulevard
Echelon IV, Suite 400
Austin, TX 78759
TEL: (512) 343-4542
FAX: (512) 418-1186

CANADA
Altera Corporation
300 March Road, Suite 601B
Kanata, Ontario K2K 2E2
Canada
TEL: (613) 599-3141
FAX: (613) 599-3143

Altera International Sales Offices

UNITED STATES
(CORPORATE HEADQUARTERS)
Altera Corporation
2610 Orchard Parkway
San Jose, CA 95134-2020
USA
TEL: (408) 894-7000
FAX: (408) 433-3943
(408) 894-7755

UNITED KINGDOM
(EUROPEAN HEADQUARTERS)
Altera UK Limited
Holmers Farm Way
High Wycombe
Buckinghamshire
HP12 4XF
United Kingdom
TEL: (44) 1 494 602 000
FAX: (44) 1 494 602 001

NORTHERN EUROPE
Altera UK Limited
Holmers Farm Way
High Wycombe
Buckinghamshire
HP12 4XF
United Kingdom
TEL: (44) 1 494 602 020
FAX: (44) 1 494 602 021

BELGIUM
Altera Belgium
Katwilgweg 7B
2050 Antwerpen
Belgium
TEL: (32) 3 254 0420|
FAX: (32) 3 254 0320

FRANCE
Altera France S.A.R.L.
Le Mermoz
13 Avenue Morane Saulnier
78140 Velizy
France
TEL: (33) 1 34 63 07 50
FAX: (33) 1 34 63 07 51

GERMANY
Altera GmbH
Max-Planck-Str. 5
D-85716 Unterschleissheim
Germany
TEL: (49) 89 3218 250
FAX: (49) 89 3218 2579

HONG KONG
Altera Hong Kong
Suite 1008, Tower 1
China Hong Kong City
33 Canton Road, Tsimshatsui
Kowloon, Hong Kong
TEL: (852) 2377-0218
FAX: (852) 2377-2811

ITALY
Altera Italia S.R.L.
Corso Lombardia 75
Autoporto Pescarito
10099 San Mauro, Torinese (Torino)
Italy
TEL: (39) 11 223 8588
FAX: (39) 11 223 8589

JAPAN
Altera Japan, Ltd.
Shinjuku Mitsui Building
2-1-1, Nishi-Shinjuku
Shinjuku-ku, Tokyo 162-04
Japan
TEL: (81) 3 3340 9480
FAX: (81) 3 3340 9487

KOREA
Altera Korea
Youndang Building, Suite 501
144-23 Samsung-dong, Kangnam-Ku
Seoul, Korea 135-090
TEL: (82) 2 538-6895
FAX: (82) 2 538-6896

SWEDEN
Altera AB
Sjoangsvagen 15
192 72 Sollentuna
Sweden
TEL: (46) 8 626 60 91
FAX: (46) 8 626 60 99

LPM Quick Reference Guide
December 1996 A-CAT-LPM-01

I.S. EN ISO 9001