



# **Xilinx Netlist Format (XNF) Specification**

Version 6.1

June 1, 1995



# TABLE OF CONTENTS

Introduction .....	1
Designing LCAs .....	2
The Xilinx Netlist File Format .....	5
Definitions .....	5
XNF Syntax .....	6
LCANET Record .....	8
Program Record .....	9
Part Type Record .....	10
Symbol Record .....	11
Pin Record .....	12
Setup and Hold Time Record .....	13
Pulse Width Record .....	14
Design Editor Configuration Command Record .....	15
Model Record .....	16
Model End Record .....	17
Symbol End Record .....	18
Signal Record .....	19
Bus Record .....	20
Power Record .....	21
External I/O Record .....	22
User Program Record .....	24
Logical End-of-File Record .....	25
LCA Netlist Syntax Summary .....	26
Parameters .....	27
Parameter Definitions .....	28
Symbol and EXT Parameters .....	28
Symbol Parameters .....	31
I/O Block Parameters .....	40
Signal Parameters .....	42
Pin Parameters .....	44
User-defined Parameters .....	46
Xilinx's Use of User-defined Parameters .....	46
Parameter Summary .....	47
LCA Library .....	49
Symbol Definitions .....	51
Inverter/Buffer Symbols .....	52
Combinational Logic Symbols .....	53

## TABLE OF CONTENTS

Input/Output Pad Symbols .....	54
Input Buffer .....	55
Output Buffer .....	56
Three-State Output Buffer .....	57
Output Flip-Flop .....	59
Three-State Output Flip-Flop .....	61
Input Flip-Flop .....	63
Input Latch .....	65
Input Register .....	67
D-Type Flip-Flop .....	69
D-Type Latch .....	71
RAM/ROM .....	72
RAMS .....	74
RAMD .....	75
Internal Three-State Driver .....	76
Special Internal Three-State Drivers .....	77
Pull-Up Resistor .....	78
Pull-Down Resistor .....	79
Clock Drivers .....	80
Global Buffers .....	81
Unified Library Global Buffer .....	82
Equation Symbol .....	83
F5_MUX .....	85
Carry Logic Symbols .....	86
CLB Symbol .....	90
MAP Symbols .....	91
IOB Symbol .....	94
TIMESPEC Symbol .....	95
TIMEGRP Symbol .....	97
Oscillator .....	98
Special Function Access .....	100
Macro Symbol .....	103
Summary of Symbol Types .....	104
Sample LCA Xilinx Netlist File .....	107
Appendix A: LCA Naming Restrictions	
Appendix B: LCA Location Names	
Appendix C: Design Editor Configuration Commands	

## TABLE OF CONTENTS

Appendix D: Document Revision History

## LIST OF FIGURES

Figure 1.	LCA Design Flow Using Schematic Capture .....	2
Figure 2.	Sample LCA Schematic Design .....	109
Figure B-1.	LCA Location Naming Convention #1 .....	B-2
Figure B-2.	LCA Location Naming Convention #2 .....	B-3
Figure B-3.	LCA Location Naming Convention #3 .....	B-5

## LIST OF TABLES

Table 1.	Allowed Extensions for RLOC=value .....	34
Table 2.	CYMODE Values for XC4000 Carry Mode Symbols .....	88
Table B-1.	Examples of Single LOC Properties .....	B-6
Table B-2.	Examples of Area LOC Properties .....	B-6
Table B-3.	Examples of Prohibit LOC Properties .....	B-7
Table B-4.	Examples of Multiple LOC Properties .....	B-7







## Introduction

This document is a description of a netlist file format for describing Logic Cell Array (LCA) designs. Refer to the separate XNF XC7000 Specification for EPROM technology-based complex Programmable Logic Device (EPLD) designs. We use the term Programmable Logic Device (PLD) to include both LCA and EPLD devices. With the Xilinx Netlist Format (XNF) file, LCA designs may be done with any design tools for which a translator to the XNF file is available. Also, simulation of the LCA design can be done with any simulator for which a translator from the LCA Xilinx netlist to the simulator's input language is available.

The XNF format is supported by Xilinx-developed tools which handle all the details of the LCA architecture. The translators to be developed for a particular CAE system are simple programs with little or no knowledge of the details of the LCA architecture.

The objective of the Xilinx netlist format and the companion tools which implement the LCA design is to allow designers to use the CAE tools with which they are familiar for their LCA designs. Also, by using the same tools as those used for the rest of the design, system level verification of the design is simplified.

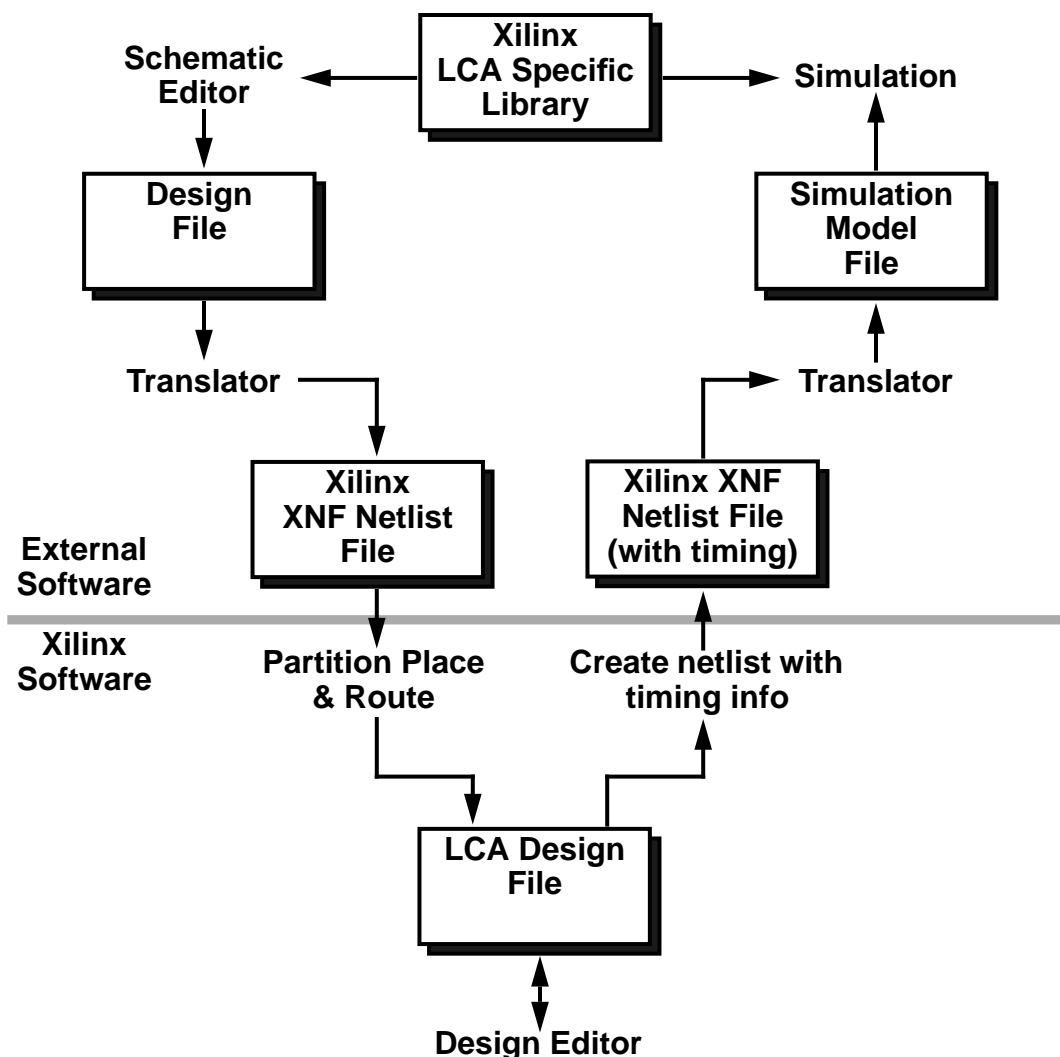
Throughout this document we assume that a schematic editor is being used for design entry. This is by far the most common form of design entry. However, the concepts described here could be applied to non-schematic design entry methods such as synthesis of state machine or register-transfer level descriptions.

The Xilinx netlist format described here was developed by Xilinx, Inc.

## Designing LCAs

Logic Cell Arrays (LCAs) are fully described in the Xilinx Programmable Logic Data Book. Before reading this document, the reader should familiarize himself with the basic architecture of the device by reading Section 2 of the Data Book. This describes the basic architecture of the device, including CLBs, IOBs and interconnect.

Currently there are two ways to design an LCA. One way is to use the Design Editor to configure each CLB and IOB and to interconnect them with nets. This methodology gives the designer complete control over his design, because he manually specifies every detail. In many cases, especially on larger designs, it can be very tedious and error prone to work at this level of detail. Therefore, most designers use a schematic editor to enter designs. This is shown in Figure 1.



**Figure 1. LCA Design Flow Using Schematic Capture**

The designer creates a schematic using symbols from an LCA specific library. This library contains elements such as combinational logic, flip-flops, clock buffers and I/O elements. If the schematic editor supports hierarchy, macros can be created which implement higher level logic functions. In this case the Xilinx merge software will flatten the hierarchy and generate a flat Xilinx netlist file. Full path names of symbols and signals will be preserved using the character “/” as the path separator.

The design file created with the schematic editor must be translated into the Xilinx netlist format file. This is done by a translator program developed for that particular schematic editor.

After the schematic design has been translated into an XNF file, Xilinx-supplied software automatically converts it into a LCA design. The Xilinx software handles all the details of the LCA architecture, including the mapping of the generic logic symbols from the schematic into the CLBs and IOBs of the LCA as well as placing and routing the design.

Simulation of the design requires a program to translate from the Xilinx netlist format into the format required by the simulator, and that simulation models of the symbols in the Xilinx netlist are available.

For LCA designs, a unit-delay simulation model should be used to verify the logic design before placement and routing is done on the LCA. Only a unit-delay simulation can be done at this time because, unlike traditional gate arrays, it is not possible to make a realistic estimate of the timing of the design until it is actually implemented in an LCA. LCA timing delays are dependent on the partitioning of the design into LCA elements. Also routing delays on the LCA are a much more significant part of the overall delay than in traditional gate arrays. It is important to do a unit delay simulation before placement and routing to avoid unnecessary design iterations.

Once the LCA design has been placed and routed, Xilinx software is used to create a delay simulation model of the design in the XNF format. Then the simulation translator can be used to create a worst case delay model for simulation.

The next two sections describe the XNF syntax and the logic symbols supported.



## The Xilinx Netlist File Format

This section defines the syntax of the Xilinx Netlist Format (XNF). It begins with a definition of the terms used followed by a definition of the syntax.

### Definitions

#### SYMBOLS

A symbol is an element in the schematic design, e.g., a NAND gate, a flip-flop, etc. All symbols have a type, which defines the symbol's logical function. For LCA designs, a symbol's type must be one of the legal symbol types defined in this document if the netlist is not hierarchical. Symbols also have a name, which may be defined by the user or may be automatically generated by the schematic editor or the translation program. A symbol's name must be unique across all other symbols in the design. Symbol names may be automatically assigned by the translator from the schematic. For hierarchical designs, the full path name of the symbol will be supported when the design is flattened using the "/" character as the path name separator.

A symbol may also define a macro. This means the symbol's type will not be one of the LCA primitive types defined here, but will refer to another XNF file which describes the logic. In this case, the design is hierarchical and the Xilinx merge software must be run to create a flat netlist before the design can be translated into a Xilinx LCA.

#### PINS

Pins are inputs to, and outputs from symbols. Each pin on a symbol has a name unique among all the pin names on that symbol. A pin name must be one of the legal names for the type of the symbol the pin is on. The legal symbol types and the pin names allowed for each symbol are defined later in this document.

For macro symbols, the Xilinx merge program uses pin names to determine the signal in the macro definition corresponding to the pin. If a PIN parameter matching the pin name is used on a signal then that signal is connected. Otherwise, a signal with the same name as the pin is sought to determine connectivity. If neither of these occurs, then it is an error.

#### SIGNALS

Signals define electrical connections between multiple symbol pins. Most signals connect to one symbol output pin and one or more symbol input pins. However, some signals may connect to multiple symbol output pins as well. Signals must have a name unique among all other signals in the design. For hierarchical designs, the full path name of the signal should be supported when the design is flattened using the "/" character as the path name separator. Signals may have the same name as a symbol, but this does not imply any relationship between them. Signal names must be legal Xilinx names. The Xilinx naming restrictions are described in Appendix A of this document.

## PARAMETERS

Parameters are PLD-specific design criteria provided by the user in the schematic that instruct the automatic tools to do particular things when implementing the schematic design in a PLD. Parameters may be specified for a symbol, a signal, or a pin. Some of the parameters are allowed only on certain symbol types.

Not all schematic editors support the concept of user-definable parameters, and others may support them for symbols but not signals or pins. Later in this document are some ideas on how to handle the concept of parameters in schematic editors that do not support them directly.

## VCC/GND SIGNALS

Different schematic editors use different conventions for identifying VCC and GND connections. Some use naming conventions, others use explicit schematic elements to identify these connections. In order to be as general as possible, the Xilinx netlist format does not use a naming convention for VCC and GND connections. Instead, any signal may be flagged as a VCC or GND signal by using the PWR record to identify it as a VCC or GND signal. The PWR record is defined in detail in the XNF Syntax section below.

## DELAYS

Pins have delays associated with them. Both symbol input pins and symbol output pins can have delays. Input pin delays are necessary to model the delays fully. This may cause problems for some simulators. In this case, delay buffers may be required to model these delays.

Delays are written into the Xilinx netlist file by a Xilinx-supplied program after the LCA design is placed and routed. The Xilinx supplied program reads in the LCA design file containing timing information and writes out the Xilinx netlist file with the delay information added.

Simulation issues and some ideas on how to handle them are discussed in more detail in later sections.

## XNF Syntax

The Xilinx netlist file is an ASCII text file. It consists of newline-separated lines, which are called records. Each record is terminated by an ASCII carriage return (CR) or carriage return-line feed (CR, LF), whichever is appropriate for the system it is on.

Each record is made up of a number of fields. Fields are separated from each other by commas. Note that this means commas are not allowed in the data. Certain fields contain optional information and may be empty. An empty field is represented by two consecutive commas. Leading and trailing blanks and tabs in a field are ignored. The maximum length of a field is 1024 characters. The maximum line length allowed is 2048.

The first field of any record is the record type and must be one of the legal record types defined in this section. The meaning of the remaining fields in the record is determined by the record type. Fields are of one of the following types:

#### char fields

A char field consists of a single, non-blank character. Leading and trailing blanks and tabs are ignored.

#### string fields

String fields are simply alphanumeric strings. They may contain any character except a comma, carriage return or line feed. Leading and trailing blanks and tabs are ignored. Strings may be made up entirely of digits. The maximum length of a string field is 1024 characters.

#### numeric fields

Numeric fields are made up of a string of digits (0 through 9), with an optional leading minus sign ('-') which specify a decimal integer value. Leading and trailing blanks and tabs are ignored. Floating point numbers, non-integers and non-decimal numbers are not allowed in numeric fields.

#### delay fields

Delay fields are floating point values of the form "<number>.<number>" (e.g., 0.5). Delay fields specify rising and falling delay times in nanoseconds. The rising time is specified first, optionally followed by a colon (':') and the falling time. The colon and the falling time need not be specified if it is the same as the rising time.

Delay fields are also used to specify setup and hold times. Empty delay fields with no rise and no fall times specified should always be assumed to have value zero. Note that it is possible for the delay to be negative, but only in the case of the setup and hold time record, described later in this section.

#### parm fields

Parm fields specify user defined parameters and optional values. The parameter type is specified first, optionally followed by an equals sign ('=') or the characters "<>", followed by the parameter value. Spaces and tabs immediately before or after the equals sign or "<>" are ignored.

If the parameter type is missing, then the parameter is assumed to be a user parameter. User parameters are usually ignored by the Xilinx software, but may be used to pass information from one translator to another.

## LCANET Record

Format:

LCANET, version

LCANET (string field)

The first field defines this record as an LCANET record.

version (numeric field)

The second field is a number specifying the version of the syntax which this file conforms to. The syntax described in this document is version 6.

The first record in the file must be an LCANET record, which defines the version of the netlist syntax being used. The version described in this document is version 6. Future revisions to this syntax may make it incompatible with parsers written for previous versions of the syntax. This record allows the parser to detect when it cannot read a file.

The LCANET record must be the first record in the file. No other LCANET records are allowed elsewhere in the file.

Please refer to Appendix D for a complete list of the changes from version 5 to 6. All programs that read XNF netlist files should read versions 5 and 6 to be backward compatible to recent versions.

Example:

LCANET,6



## Program Record

Format:

PROG, program name, rev, "comment"

PROG (string field)

The first field defines this record as a program record.

program name (string field)

The second field is the name of the program this record was created by.

program rev (string field)

The third field is the revision of the program.

comment (string field)

The last field is a comment which may contain any string and should be contained inside double quotes. It should contain the date and time the program was run as well as the command line options that were used.

The program records provide an audit trail of the programs used to create and process the file. A program that writes out an Xilinx netlist file should add a program record that specifies the name and version of the program. The comment field can be used for any other information the program developer feels is useful such as command line options used when running the specific program. The date and time that the program was run should be included in the comment field.

Programs that read in and then re-write a Xilinx netlist file, should preserve all PROG records from the input XNF file. The order of PROG records is not significant.

Example:

PROG,WIR2XNF,5.0, "Created from TEST.WIR at 10:18:27 on 4/27/93. -p 4005pg156"

## Part Type Record

Format:

PART, part type

PART (string field)

The first field defines this as a part type record.

part type (string field)

The second field defines the PLD part type for this design. It should be a legal PLD part type with an optional speed grade (such as -7) appended.

The PART record is used to specify the target PLD part type and speed grade. Legal part types and speed grades are listed in the data sheets; however, they should always begin with a number (i.e., 2, 3, 4, 5 or 7). If the part type string does not define a legal part type the program reading the file will abort.

Examples:

PART,3020PC68-70

PART, 5210PC84-5

## Symbol Record

Format:

Sym, name, type, parameters...

Sym (string field)

The first field defines this record as a symbol record.

name (string field)

The second field is the name of the symbol. This name must be unique among all symbols. See the naming conventions in Appendix A.

type (string field)

The third field specifies the type of the symbol. This type must be one of the legal symbol types described in this document, or may specify a macro type to be expanded by the Xilinx merge program.

parameters (parm fields)

The fourth and subsequent fields specify parameters for the symbol. Legal parameters for symbols are defined in the Parameter Definitions section.

The symbol record begins the definition of a symbol and its connections to signals. All the following records, up to an END record define the symbol.

In many cases symbol names are not defined by the user and must be generated by the translator program. Generated names should be constructed in such a manner as to facilitate the designer's finding the symbol in the schematic.

Examples:

Sym,Q<0>,DFF

Sym,4-AND5,AND

Sym,TOUT,OBUFT,FAST,BLKNM=OUT1,LOC=P17

## Pin Record

Format:

PIN, name, direction, signal, dly, parameters...

PIN (string field)

The first field identifies this record as a PIN record.

name (string field)

The second field is the name of the pin this record refers to. This name must be one of the legal pin names for the type of symbol it is on.

direction (char field)

The third field specifies the direction of the pin and must be I, O or B. I specifies an input to the symbol, O specifies an output from the symbol. The direction B (bidirectional) is allowed on only macro symbols.

signal (string field)

The fourth field is the name of the signal this pin is connected to. All pins with the same signal name in this field are connected together by that signal. The signal may also have a SIG record defined for it if there are parameters associated with the signal, but the PIN records are sufficient to define a signal and its connections.

dly (delay field)

The fifth field specifies the delay of this pin in nanoseconds. This field should be left blank by the translator from the schematic and is set by the Xilinx program which extracts the delays from the LCA file. If there are parameters for the pin, the blank delay field is delimited by two consecutive commas. (See the example below.)

parameters (parm field)

The sixth and subsequent fields specify parameters for the pin. The legal parameters for pins are defined in the Parameter Definitions section.

Pin records specify the signals connected to a symbol. Pin records must always be part of a symbol definition, i.e., between a SYM record and its corresponding END record.

When the delay fields are empty, the Xilinx program which generates an XNF netlist with timing information has not been run yet. In this case, simulation models extracted should be unit delay models.

Example:

PIN,C,I,CLOCK,,INV

## Setup and Hold Time Record

Format:

SETUP, input pin, clock pin, active edge, setup time, hold time

SETUP (string field)

The first field defines this record as a setup and hold time record.

input pin (string field)

The second field is the name of the input pin on the current symbol for which the setup and hold times are specified.

clk pin (string field)

The third field is the name of the clock pin on the current symbol against which the setup and hold requirements must be met.

active edge (char field)

The fourth field is a character, either '+' or '-', which specifies the active clock edge against which the setup and hold times are to be measured.

setup time (delay field)

The fifth field specifies the setup time in nanoseconds for the input pin with respect to the clock pin.

hold time (delay field)

The sixth field specifies the hold time in nanoseconds for the input pin with respect to the clock pin.

Setup and hold time records define setup and hold time requirements for a pair of pins on the current symbol. They are created by the Xilinx program which generates an XNF netlist with timing information.

Setup and hold time records must always occur as part of a symbol definition, that is between a SYM record and its corresponding END record. The pins referred to on the record are always pins on that symbol. Setup and hold times may not be specified between pins on two different symbols.

Multiple setup/hold time records may be specified on a single symbol for multiple input pins. For example, on the XC3000 family flip-flop, setup/hold times will be specified for both the data-to-clock and the clock enable-to-clock pins. Multiple setup/hold time records may also be specified on a single symbol for the same pair of pins. For example, on the XC4000 RAM, there is a setup requirement on the address pins against the rising edge of WE, and a hold requirement against the falling edge of WE.

Example:

SETUP,D,C,+,5,3

## Pulse Width Record

Format:

PULSE, pin, polarity, min width

PULSE (string field)

The first field defines this record as a pulse width record.

pin (string field)

The second field is the name of the pin on the current symbol for which the pulse width times are specified.

polarity (char field)

The third field is a character, '+' or '-', specifying the polarity of the pulse to be checked.

min width (delay field)

The fourth field specifies the minimum time allowed for a pulse in nanoseconds.

Pulse width records are created by the Xilinx program that generates an XNF netlist with timing information. Pulse width records always occur as part of a symbol definition, that is between a SYM record and corresponding END record. The pin referred to on the record is always a pin on that symbol.

Multiple pulse width records may be specified on a single symbol for multiple input pins.

Example:

PULSE,C,-,3

## Design Editor Configuration Command Record

Format:

CFG,<Design Editor configuration command>

CFG (string field)

The first field specifies that this is a configuration command record.

Design Editor configuration command (string field)

The second field must be a legal BASE, CONFIG or EQUATE command. The syntax for these commands is documented in Appendix C.

Design Editor configuration command records define commands which specify the configuration for CLB and IOB symbols. These symbols allow the user to manually specify the exact configuration desired for a CLB or IOB without relying on the partitioning algorithm to create it.

The configuration of the CLB or IOB is specified by placing Design Editor configuration commands right in the schematic. These commands are then passed along in this record. This gives the user absolute control over the LCA configuration, if desired. This record must be part of a symbol definition and is allowed only on symbols with type CLB or IOB.

Note that it is NOT necessary for the translation program to parse configuration commands. They should just be passed along, as is, to the LCA Xilinx netlist file.

Example:

```
CFG,BASE FGM
CFG,CONFIG G:Q Q:FF X:Q CLK:K:NOT
CFG,EQUATE F = (A*D) + (~A*C)
CFG,EQUATE G = Q
```

## Model Record

Format:

MODEL

MODEL (string field)

The first field specifies that this is a MODEL record.

The MODEL record defines the beginning of the definition of the simulation model for a CLB or IOB symbol. The records following the MODEL record, up to a matching ENDMOD record, define the simulation model for the symbol. MODEL record groups are always contained within a symbol definition, i.e., between a SYM record and its matching END record. MODEL records are present for ONLY CLB and IOB symbol types. The MODEL record groups are placed in the netlist by some of the Xilinx-developed programs.

The records within the MODEL and ENDMOD records are normal records just like those used to describe the rest of the network. When the simulation translator program reads the netlist and encounters a CLB or IOB symbol it should ignore the PIN and CFG and other records in that symbol group. The records in the model group define the network corresponding to the symbol. If there is no model group for a CLB or IOB symbol, then the program that generates an XNF netlist from an LCA design has not been run yet. The simulation model for the network cannot be built until this program is run.

Note that names in the model group may contain the '.' character which is normally not allowed in LCA names.

Example:

```
SYM,Q0,CLB
PIN,A,I,D<0>
PIN,K,I,CLK
PIN,X,O,Q<0>
CFG,BASE FG
CFG,CONFIG F:A CLK:K:NOT Q:FF X:Q
CFG,EQUATE F = A
MODEL
SYM,Q0.F,BUF
PIN,I,I,D<0>
PIN,O,O,Q0.F
END
SYM,Q0.FF,DFF
PIN,D,I,Q0.F
PIN,C,I,CLK,,INV
PIN,Q,O,Q<0>
END
ENDMOD
END
```



## **Model End Record**

Format:

ENDMOD

The model end record terminates the group of records which define a model. Every MODEL record must have a matching ENDMOD record.

## Symbol End Record

Format:

END

The symbol end record terminates the group of records which define a symbol. Every symbol record must have a matching end record.

Example:

```
SYM,U1,INV
PIN,I,I,CS
PIN,O,O,-CS
END
```

## Signal Record

Format:

SIG, signal name, parameters...

SIG (string field)

The first field defines this record as a signal record.

signal name (string field)

The second field defines the name of the signal. This name must match the name used on the PIN records specifying the connections for this signal. See the naming conventions in Appendix A.

parameters

The third and subsequent fields specify parameters for the signal. Legal signal parameters are defined in the Parameter Definitions section.

Signal records are used for parameters for a signal. If there are no parameters to be specified, then the signal record need not be included, because the signal and its connections are defined by PIN records, which are part of the symbol definition.

Example:

SIG,ADR<0>,X

## Bus Record

Format:

BUS, bus name, bus bits...

BUS (string field)

The first field defines this record as a bus record.

bus name (string field)

The second field defines the name of the bus.

bus bits (string field)

The third and subsequent fields specify the individual bit names of each bit of the bus signal.

Bus records are used for resolving bus/signal names when a design is merged using Xilinx software. For those schematic packages that treat busses the same as signals, it is not necessary to create separate bus records.

Note that the "<" and ">" characters are reserved for specifying a complete bus signal name. For example the signal corresponding to the "0" bit of bus "ADR" would be expressed as "ADR<0>". The complete bus signal names must adhere to the naming restrictions in Appendix A.

Examples:

BUS,ADR,0,1,2,3

{This record represents the 4 signals ADR<0>, ADR<1>,ADR<2> and ADR<3>}

BUS,CTRL, Read, Write, Select

{This record represents the 3 signals CTRL<Read>, CTRL<Write> and CTRL<Select>}

## Power Record

Format:

PWR, polarity, signal name...

PWR (string field)

The first field defines this record as a power record.

polarity (char field)

The second field is a character, '1' or '0' specifying whether this is a connection to VCC or GND respectively.

signal name (string field)

The third and subsequent fields are the names of signals which are VCC or GND connections. See the naming conventions in Appendix A.

The power record is used to specify signals which are VCC or GND connections. Multiple signal names may be specified on a single record.

Power connections are treated specially by the Xilinx netlist preparation program. Wherever possible, power connections are optimized when converting the design into the LCA architecture. For example, if an AND gate has an input tied low the AND gate will be removed and the signal connected to its output will be tied low.

It is a fatal error for a power signal to have a source pin driving it. It is also a fatal error to define both a 0 and a 1 power connection for the same signal.

A signal on a power record does not have to be used. For schematic editors that use naming conventions to identify power signals, the power record may be used to indicate the signal names that are, by convention, power connections.

Examples:

PWR,1,VCC,VDD

PWR,0,GND,VSS

## External I/O Record

Format:

EXT, signal name, direction, pin number, parameters...

EXT (string field)

The first field defines this record as an external I/O record.

signal name (string field)

The second field specifies the name of the signal which is being defined as an external I/O signal. See the naming conventions in Appendix A.

direction (char field)

The third field is a single character. It must be 'I' for an external input, 'O' for an external output, 'T' for a three-state external output, 'B' for a bidirectional connection, or 'U' for an unbonded IOB.

pin number (string field)

The fourth field is the pin number. It is added by the Xilinx program that creates an XNF file from an LCA and should be left blank by other programs.

parameters (parm field)

The fifth and subsequent fields define optional parameters for this external connection. Legal parameters are defined in the Parameter Definitions section.

The external I/O record is used to specify which signals in the schematic connect off chip, or correspond to the pad on an unbonded IOB.

The only signals which may be specified as external I/O signals are those which connect to the external pins of the I/O symbols: IBUF, OBUF, OBUFT, INFF, INLAT, INREG, OUTFF and OUTFFT.

In the XC3000 family, there are special buffers from an I/O pin directly to the input of each of the clock buffers (GCLK and ACLK and BUFG). To specify this schematically, the user connects the input pin of the BUFG, GCLK and ACLK symbols to the external signal corresponding to the I/O pin. The same is true for the BUFG symbol in the XC4000 and XC5200 families, and for the BUFGP and BUFGS symbols in the XC4000 family.

The pin number field is added by the Xilinx software that creates an XNF file from the LCA design. It should be left blank by all other programs. The pin number may be useful to the simulation translator in creating models or symbols to be used in board-level simulation. Note that the pin number is a string field because on some package types (specifically PGA packages) pin "numbers" are not numeric. Refer to the Xilinx Data Book to find out exactly what pin number will appear in this field for the specified part type. Examples are: K12, P4, 39, 7.

The parameters on the external I/O record may be used to specify any IOB parameters, including the location, logical name, INTERNAL, INIT, NODELAY, FAST, SLOW, etc. It is also possible to specify these parameters on the I/O symbols directly.

In general, the translator should generate an EXT record when the user specifies an off-chip connector symbol in his schematic such as a IPAD, OPAD, IOPAD or UPAD.

Example:

```
EXT,DATA<1>,O,,LOC=P22,FAST
```

## User Program Record

Format:

USER, record type, user-defined data

The USER record may be used by external programs to pass information which is not supported by the Xilinx tools. An example might be naming information passed from the schematic translator to the simulation translator. The record type and the format of the user defined data is to be defined by the programs which read and write these records. The only constraint is that the record be fewer than 1024 characters long and that it be terminated by a newline, like other record types.

Any program which reads and then rewrites a netlist file must preserve all USER records in the original file.

Note that the position of the records in the file may be changed. USER records should not be placed inside SYM groups because they may be moved outside of the SYM group when the file is rewritten.



## Logical End-of-File Record

Format:

EOF

The logical end-of-file record defines the logical end of the file. It is the last record in the file. There should not be any data following this record; if there is, it will be ignored.

Example:

EOF

## LCA Netlist Syntax Summary

The following is a summary of all the record types and their field definitions:

- LCANET, version
- PROG, program name, rev, "comment"
- PART, part type
- SYM, name, type, parameters...
- PIN, name, direction, signal, dly, parameters...
- SETUP, input pin, clock pin, active edge, setup time, hold time
- PULSE, pin, polarity, min width
- CFG, Design Editor configuration command
- MODEL
- ENDMOD
- END
- SIG, signal name, parameters...
- BUS, bus name, bus bits...
- PWR, polarity, signal names...
- EXT, signal name, direction, pin number, parameters...
- USER, record type, user-defined data
- EOF

## Parameters

Parameters are user-defined instructions for the automatic tools. Parameters fall into several categories: user-defined parameters, symbol parameters, I/O block parameters, pin parameters, and signal parameters. Each of these categories is defined in detail in the following sections.

Some schematic editors have full parameter definition capabilities, while others support only parameters, (a part type and a reference location), that make sense in board-level TTL-type designs. It is up to the developer of the symbol library and translator program to decide how to support parameters for a specific schematic editor. On editors where user-definable parameters are fully supported, the translator need only pass along to the Xilinx netlist file whatever parameters the user has defined. Full error checking of legal parameters will be performed by the Xilinx software.

On systems without user-definable parameters, the developer may decide not to support certain parameters, or to support them in a less general way. An example of this is the INV parameter for invertible pins. If the schematic editor does not easily support this then it may be omitted because the user may use inverters in the place of inverted pins without affecting the quality of the final LCA designs. This is because inverters are removed wherever possible by the Xilinx technology mapping programs. Alternatively, the library provided to the user could include symbols which imply the INV option on some pin. An example might be a combinational symbol with bubbled inputs. The symbol might be called AND2B1 where pin I0 is the bubbled input pin. The translator program would then automatically create an INV parameter for the pin record for I0.

Wherever possible, arbitrary parameters should be allowed. That is, the translator should not check the parameter type or value, but just pass along whatever the user has specified. The Xilinx software will check for errors. This will allow future additions to the parameter list without modifications to the translator programs.

XC7000 EPLD designs use some additional parameters not listed in this specification; refer to the XNF XC7000 Specification for details.

## Parameter Definitions

### Symbol and EXT Parameters

The following parameters may be specified on a symbol or on an EXT record:

#### LOC=nn and LOC<>nn

The LOC parameter is used to specify the LCA location at which to place a symbol. The location, “nn”, must be a legal LCA location for the target LCA part type. Likewise the LOC<> can be used to prohibit the placement of logic into a particular CLB or IOB. For symbols which map into single CLBs, the location must be a CLB name. For symbols which map into single IOBs, the location must be a valid IOB name. Location names are also allowed for three-state buffers, the decode logic in the XC4000 LCA and XC4000 global buffers.

Multiple LOC parameters may be specified for the same symbol by using a “;” to separate each LOC within the field (e.g., LOC = CLB\_R1C1; LOC <> CLB\_R5C3). This specifies that the symbol(s) be placed (or not placed if using LOC<>nn) in any of the locations specified. Also an area may be specified in which to place a symbol or group of symbols. Areas are specified by using a “:” or “;” (e.g., LOC = CLB\_R1C1:CLB\_R5C5). The colon is preferred; the semicolon is accepted to maintain compatibility with the previous definition. The location value on the left hand side of the colon specifies the top-left block for the area, while the value on the right hand side of the colon specifies the bottom-right block of the area. Note that the legal names are a function of the target LCA part type. Please see Appendix B for a discussion of legal values allowed for each family of LCAs and for examples of the different ways the LOC parameter can be used.

The LOC parameter may also be used for logic that uses multiple CLBs, IOBs or other LCA resources. To do this the LOC parameter can be used on a soft macro symbol; the Xilinx merge program passes the location information down to the logic on the lower level.

The LOC parameter may also be specified on the EXT signal record. In this case the IOB corresponding to the external signal will be placed in the specified location. If different LOC parameters are specified on an EXT record and on a symbol which connects to that signal, an error is generated.

LOC parameters are not allowed on symbols which already have a fixed location, such as GCLK, ACLK and OSC.

#### BLKNM=name

The BLKNM parameter is used to specify a logical block name to be assigned to the CLB, TBUF or IOB in which this symbol ends up. Designers may use this capability to make the resulting LCA file easier to understand by making the CLB,

TBUF, and IOB logical names meaningful with respect to the schematic. Hierarchical prefixes are not prefixed to BLKNM parameters. See the HBLKNM parameter if this feature is desired.

In the absence of a BLKNM parameter, the program which generates the LCA file will still try to create meaningful CLB and IOB logical names from the names of the signals which are connected to the block.

BLKNM parameters are ignored on combinational logic, and on symbols which have fixed locations, such as GCLK, ACLK and OSC.

Note that symbol names are not used for CLB and IOB names because symbol names may be computer generated and thus not meaningful to the user. The BLKNM parameter must be used to define a CLB or IOB logical block name. Of course, the BLKNM could be the same as the symbol name.

Although the primary purpose of this parameter is to name LCA blocks, a secondary purpose is to support the user-specified grouping of symbols into LCA blocks. If two or more symbols have the same BLKNM value, the logic mapping software will attempt to place those symbols in the same LCA block (CLB or IOB). For this reason, it is important to attach identical BLKNM parameters to only symbols which can, and are desired to be, partitioned into the same LCA block. Also note that BLKNM cannot be used to group TBUFs — because there is a one-to-one correspondence between the TBUF symbol and the TBUF LCA block — and a BLKNM on a TBUF must therefore be unique among all BLKNM parameters in the design.

The HBLKNM parameter, explained below, can also be used to specify grouping of symbols into LCA blocks. HBLKNM has the advantage that it can be used inside of a macro that is instantiated more than once.

## HBLKNM=name

The “hierarchical block name”, HBLKNM, parameter is similar to the BLKNM parameter. It is used to name the LCA block (CLB, IOB, TBUF) that includes the associated symbol. Unlike the name value of the BLKNM parameter that remains unchanged throughout the design process, the name value of the HBLKNM parameter will be hierarchically qualified during the design flattening process. The name value for HBLKNM= will have each level of the design hierarchy prefixed to the name, with each level separated by the forward slash character “/”. With this feature the designer may specify an HBLKNM parameter on a symbol in a macro that is instantiated several times in a design and rely on the design-flattening process to create unique hierarchical HBLKNM values for each instantiation of the symbol. The creation of unique HBLKNM values is important because LCA blocks are required to have unique names.

Although the HBLKNM parameter can be used simply to name LCA blocks, its primary purpose is to indicate user-specified grouping of symbols into common LCA blocks. Because HBLKNM is hierarchically qualified, this grouping can be

indicated within macros which are instantiated more than once in a design. If two or more symbols have the same hierarchically qualified HBLKNM value, the logic mapping software will attempt to place those symbols in the same LCA block. Note, however, that HBLKNM cannot be used to group TBUFs — because there is a one-to-one correspondence between the TBUF symbol and the TBUF LCA block — and a HBLKNM on a TBUF must therefore be unique among all HBLKNM parameters in a hierarchical block.

#### TNM=name\_list

The TNM parameter is used to define groups of path endpoints for use in TIMESPEC specifications. The name\_list consists of one or more logical names, separated by semicolon characters (;). Each logical name is a combination of letters, digits and underscores (\_). The TNM parameter is allowed on I/O flip-flops, CLB flip-flops, I/O latches, CLB latches, RAMs, RAMSs, RAMDs and EXT records, as well as on macro symbols, pins and signals.

When a TNM is specified on a macro symbol, it will be propagated down to all lower-level symbols of the appropriate type by the Xilinx merge program. To restrict the downward propagation to only specific types of symbols, each logical name may be prefaced with a qualifier followed by a colon (:) character. The valid qualifiers are: “ffs”, “rams”, “pads” and “latches”.

No qualifiers are allowed on TNM parameters on EXT records. All four qualifiers are allowed on TNM parameters on Symbol records.

#### Examples:

```
TNM=RDDATA
TNM=ENA;CNTENA
TNM=PADS:ABUS
TNM=FFS:REG01;LATCHES:ADDR
```

## Symbol Parameters

### DOUBLE

On the XC3000 and XC4000 family parts, there are internal bus structures with programmable pull-up resistors available for implementing bidirectional busses or wired-AND functions. Two pull-up resistors are available on each internal bus line. The designer may use both resistors for a fast, higher-power signal, or only one for a slower, lower-power signal. The DOUBLE parameter on the PULLUP symbol specifies that the fast, high-power option is to be used. The slower, lower-power option will be used by default.

Note that the DOUBLE parameter will be considered an error on PULLUP symbols which are used to specify a pull-up resistor on an external signal. It is also an error to specify the DOUBLE parameter on a PULLUP symbol on an internal bus line in a design targeted for the low-voltage ZERO+ parts (XC2000L and XC3000L) as well as for the XC5200 family.

### FILE=filename

The FILE parameter on a symbol may be used to specify the name of the XNF file which defines the logic for a macro symbol. Designs with macro symbols must be flattened with the Xilinx merge program before running the technology mapper.

An extension of **.xnf** will be assumed if no extension is present on the file name. If the FILE parameter is missing on a symbol with an unknown type, the symbol's type, with an extension of **.xnf**, will be used as the file name. The FILE parameter should not be used on primitive symbols.

### MAP=type

The MAP parameter is allowed on only the CLBMAP, FMAP, F5MAP, and HMAP symbols. The map symbol is used when explicit control over partitioning is required. This symbol tells the technology mapper how to partition logic into the LCA. For more information on how to use this symbol, refer to the section on symbol definitions.

The currently supported values for the "type" are: PLC (Pins Locked Closed), PLO (Pins Locked Open), PUC (Pins Unlocked Closed), and PUO (Pins Unlocked Open). The PLC and PLO types tell the mapping program to keep signals locked to the pins used on the map symbol. The PUC and PUO types are used when it isn't important to keep a particular signal locked to a specific pin. The "C" (Closed) tells the mapping program that no more logic other than what is explicitly specified can be put into a CLB or function generator. In this case, the user must specify all input and output signals. The "O" (Open) tells the mapping program that it should determine the logic to place in the CLB or function generator given the specified output signals. In this case, the user must specify the output signals, but not all (or any) of the input signals. FMAP and CLBMAP

symbols may have any of the four MAP parameter values, but MAP parameters of HMAP symbols may have only the PUC type. MAP parameters of F5MAP symbols may have only the PUC type.

#### INIT=value

The INIT parameter is allowed on all XC4000 flip-flop elements, the ROM symbols, and the XC4000E RAM, RAMS and RAMD symbols. When used on flip-flop symbols, the value will be either R (Reset, the default) or S (Set). This is the value the storage element will take on during power-on initialization and assertion of the global Set/Reset signal. A XC4000 CLB flip-flop with a reset direct or clear pin is always INIT=R and a CLB flip-flop with a set direct or preset pin is always INIT=S.

When used on a ROM or XC4000E RAM/RAMS/RAMD symbol, the value will be 4 or 8 hexadecimal digits which determine the bit pattern for a ROM or the initial bit pattern for a RAM/RAMS/RAMD. Leading zeros should be added if needed to achieve the correct number of hexadecimal digits, 4 digits for a 16X1 ROM, 16X1 RAM, 16X1 RAMS or RAMD and 8 digits for a 32X1 ROM, 32X1RAM, or 32X1 RAMS. The INIT property is required for the ROM symbol.

#### DECODE

The DECODE parameter is allowed on the XC4000 WAND symbol only. When this parameter is present, it indicates that the wired-AND should be implemented in edge decode logic.

#### EQN=equation

The EQN parameter is allowed on the EQN symbol only. It is used to specify the function of the symbol without having to use the equivalent gate level logic. It can be used for only combinational logic. The legal Boolean operation symbols allowed in this parameter are: "~" (not), "+" (or), "\*" (and), and "@" (xor) as well as "(" and ")" (parentheses).

The equation is expressed in terms of the symbol's input pin names. For example,  $EQN=((I0+I1)*I2)$ .

#### DEF=type

The DEF parameter is allowed on symbols that represent special logic. The currently supported types are "HM"\* (hard macro) and "BLOX" or "XBLOX". The software handles hard macros and X-BLOX symbols as special symbols. Any symbol that represents a hard macro or X-BLOX symbol should have this parameter on it.



\* The DEF=HM will be supported for existing designs but will no longer be supported with the Unified libraries. Hard macros are being replaced by Relationally Placed Macros (RPMs) in the Unified library.

#### CYMODE=carry\_mode

The CYMODE=carry\_mode parameter is a schematic library or program-defined parameter added to CY4 or Carry Mode SYM records. The carry\_mode string is a valid XNF string field.

The carry\_mode string identifies the mode for the dedicated carry logic in an XC4000 CLB. In a schematic library, it is typically defined on the Carry Mode symbol and is transferred to the CY4 symbol by the Xilinx netlist preparation program. It may also be assigned directly to a CY4 primitive by a non-schematic program. The netlist preparation program may also modify the carry\_mode string if CY4 outputs are unused.

Examples of valid carry\_mode strings are: ADD-FG-CI, SUB-G-F1, DEC-FG-CI, FORCE-1. A current list of valid carry modes is given with the Carry Mode symbol definition in this document. There is a one-to-one correspondence between Carry Mode symbol types and carry\_mode strings.

#### SCHNM=name

This parameter will carry forward the original symbol type name from the schematic library into the XNF file. The name must be a valid XNF string field.

Examples:

SYM, IS1, DFF, SCHNM=fdce

SYM, inst5, AND, SCHNM=and5b3

#### LIBVER=version\_string

This symbol parameter will be on all symbols of the Unified Library only. It will enable Xilinx software to distinguish between netlists created using the Unified Library and those created with earlier versions of the libraries.

This property will eventually be used by the Xilinx netlist checker (XNFPREP) to advise users to switch over to the Unified Libraries. The version\_string value will be a triplet of the form x.x.x. Each component of this triplet will be an integer. The first component of the string will be the library version. (Currently only 2 is valid.) The second component of the string will be the symbol version (starting with 0). The last component of the string will be the simulation model version if the symbol is a primitive; otherwise, if the symbol is a macro, this number will refer to the version of the underlying schematic (in either case, starting with 0). The absence of a LIBVER parameter will represent an implied version 1 library, which is anything before the Unified Library.

Only the first component of the version\_string (the library version) is significant to Xilinx programs. The other components may be used by translators to track library revisions, or simply left at 0.

Examples:

```
SYM, IS1, DFF, SCHNM=fdce, LIBVER=2.1.3
SYM, inst5, AND, SCHNM=and5b3, LIBVER=2.0.0
```

### TSidentifier=specification

The TS parameter is used to convey timing information to the Xilinx place and route program. It is to be attached to the TIMESPEC symbol and should be translated to a parameter on the SYM record of the TIMESPEC symbol. This symbol parameter is allowed on only the TIMESPEC symbol. The *identifier* is a name consisting of any combination of letters, digits and underscores. It should be kept short for convenience and clarity. The specification can be a combination of characters allowed in LCA names, plus colons (":"), equals ("="), asterisks ("\*"), parentheses "(" and ")" and question marks ("?").

Examples:

```
SYM, name1, TIMESPEC, TS01=DP2S:10, TS02=DC2P:25
SYM, name2, TIMESPEC, TS03=FROM:PADS(data*):TO:FFS=TS02/8
```

*The following parameters (RLOC, U\_SET, HU\_SET, H\_SET, RLOC\_ORIGIN, and RLOC\_RANGE) are all used to specify, name, and manipulate sets of symbols that have relative location constraints. These parameters may be used on only CLB flip-flops, CLB latches, memories, carry logic, TBUFs, non-DECODE WANDs, WORANDs, mapping symbols, F5\_MUXs, EQNs and macros. These parameters are allowed on only XC4000 and XC5200 family symbols.*

### RLOC=value

The RLOC=value parameter is a user-, system-, or library-defined parameter added to SYM records. The value field of the RLOC parameter has the following syntax:

RnCn[.extension]

The Rn and Cn denote the row and column numbers of the LCA grid array, and each must be a positive integer or 0. The values allowed for the [.extension] suffix appear below.

Table 1. Allowed Extensions for RLOC=value

Values	Families	Use
.0, .1, .2, .3	XC5200	Further denote positioning of TBUF symbol
.1, .2	XC4000	Further denote positioning of TBUF symbol; .0 and .3 are invalid for these families

Table 1. Allowed Extensions for RLOC=value

Values	Families	Use
.LC0, .LC1, .LC2, .LC3	XC5200	Further denote positioning of FMAPs, flip-flops, latches, and CY_MUXs within the appropriate Logic Cell of the CLB
.LC0, .LC2	XC5200	Further denote positioning of F5MAP and F5_MUX symbols within the appropriate Logic Cell of the CLB
.ffx, .ffy	XC4000	Further denote positioning of flip-flop symbols within the flip-flop locations of the CLB
.f, .g	XC4000	Further denote positioning of FMAPs, EQNs and 16x1 versions of ROM, RAM and RAMS symbols within the function generator locations of the CLB
.h	XC4000	Further denote positioning of HMAP symbol within the H-function generator of the CLB

The “relative location,” RLOC, parameter defines the desired row and column relationship between two or more design symbols. They do not define an absolute LCA location specification for symbols. Instead, they define the spatial relationships between two or more symbols. This spatial relationship will be respected by the Xilinx placement software.

The RLOC parameter will be added to some Xilinx-supplied schematic library macros to achieve the best spatial relationship/ordering of macro logic. The user will also be able to specify RLOC parameters on design logic symbols.

The simple RLOC=RnCn specification without any suffix can be used on TBUF, DFF, CY4, CY4\_XX, F5MAP, F5\_MUX, CY\_MUX, DLAT, WAND (without DECODE parameter), WORAND, RAM, RAMS, RAMD, ROM, FMAP, HMAP, EQN and macro symbols.

The RLOC parameter values can be propagated down the design hierarchy. An RLOC value on a non-primitive symbol in the design specifies the continuation of an existing RLOC “chain” down the design hierarchy. When the RLOC parameter is attached to a non-primitive symbol, the RLOC row and column values will be added to the RLOC row and column values attached to any symbols in the next lower level of hierarchy. This propagation of RLOC values is to symbols in only the same RLOC set (defined below). It is possible to “fix” the symbols with RLOC parameters to an absolute LCA location by using the RLOC\_ORIGIN parameter (defined below). It is also possible to specify an LCA range within which all the RLOC symbols in the set must remain by using the RLOC\_RANGE parameter (defined below).

**USE\_RLOC={TRUE | FALSE}**

This parameter can be attached to a macro symbol and will tell the Xilinx merge program whether or not the RLOC information inside the macro symbol should be used or ignored.

Some Xilinx macros will include RLOC parameters inside them. The user will have the option to ignore (USE\_RLOC=FALSE) or observe (USE\_RLOC=TRUE) all RLOC information inside a macro by adding this parameter to the macro symbol.

*The following parameters, U\_SET and HU\_SET, are used to collect design symbols into “sets” of symbols that have relative location parameters. The name associated with each parameter is the means by which each set is identified and named.*

**U\_SET=name**

The U\_SET=name parameter is a user-defined parameter added to SYM records that also have RLOC parameters. The name must be a valid XNF string field.

The “user-defined symbol set,” U\_SET, parameter is used to specify symbols that belong to a unique set of objects that have a relative location relationship. Therefore this parameter can be attached to only symbols that have an RLOC parameter. The name is specified by the user.

The name field of the U\_SET parameter will not change during the design flattening process. Because the name is left unchanged the designer can use the U\_SET parameter to collect together symbols in any part of the design hierarchy that have a relative location relationship. The user must ensure that the given name is unique across all the final U\_SET and HU\_SET names in the design.

**HU\_SET=name**

The HU\_SET=name parameter is a user-defined parameter added to SYM records. The name must be a valid XNF string field.

The “hierarchical user-defined symbol set”, HU\_SET, parameter is also used to specify symbols that belong to a unique set of objects that have a relative location relationship. Unlike the U\_SET parameter, the HU\_SET parameter works within the bounds of the design hierarchy. All of the symbols that belong to each unique HU\_SET must be located within the same branch of hierarchy. While the U\_SET name string is left unchanged by the design flattening process, the name field of the HU\_SET parameter will be hierarchically-qualified during design flattening. The name for the HU\_SET parameter will have each level of the design hierarchy prefixed to the name, with each level separated by the forward slash character “/”. The final hierarchically-qualified name must be unique across all the U\_SET and HU\_SET names in the design.

This parameter can be specified by the user in two ways. One method is to attach the HU\_SET parameter to each of the symbols in the desired set. The second method is to attach the HU\_SET parameter to a non-primitive (macro) symbol. The design flattening process will propagate the HU\_SET parameter to all of the lower level symbols in the macro that have RLOC parameters and are not members of another set. The propagation through the design hierarchy will stop at non-primitive symbols that do not have an RLOC parameter (a break in the RLOC chain), or at non-primitive symbols that have a different HU\_SET name, or have an RLOC\_ORIGIN or RLOC\_RANGE parameter. The HU\_SET parameter will not be propagated to RLOC symbols that have a U\_SET parameter.

#### RLOC\_ORIGIN=value

The RLOC\_ORIGIN=value parameter is a user-defined parameter added to SYM records. The RLOC\_ORIGIN parameter has the following syntax:

RLOC\_ORIGIN=RnCn

The Rn and Cn denote the row and column numbers of the LCA grid array which must be positive *non-zero* integers.

The “relative location origin,” RLOC\_ORIGIN, parameter fixes the absolute location origin for a set of symbols that have a relative location relationship. Without an RLOC\_ORIGIN parameter, a set of symbols can be located in any area of the LCA as long as the relative location relationships are maintained. When an RLOC\_ORIGIN value is specified, the value is added to the RLOC values of the set and then the RLOC values are effectively translated into absolute locations that “fix” each symbol in the set to a specific LCA location.

Usually, the RLOC\_ORIGIN parameter is attached to a non-primitive symbol that defines the “top” of a set of RLOC symbols, so that the relative locations are resolved in conjunction with the RLOC symbols for the set. Because a U\_SET is unrelated to hierarchy, the only way to specify an origin for a U\_SET is to attach the RLOC\_ORIGIN to one of the members in the U\_SET.

The RLOC\_ORIGIN must specify a single LCA location. It cannot include wildcard (\*) characters.

#### RLOC\_RANGE=value

The RLOC\_RANGE=value parameter is a user-defined parameter added to SYM records. The RLOC\_RANGE parameter has the following syntax:

RLOC\_RANGE=RnCn:RnCn

The Rn and Cn denote the row and column numbers of the LCA grid array.

Here “n” can be any *non-zero* positive integer, or the wildcard “\*” character.

The wildcard “\*” character can be associated with either the row OR column on both sides of the range separator character “:”. It cannot be used for the row character on one side and the column character on the other.

The “relative location range,” RLOC\_RANGE, parameter specifies the range of LCA locations that are allowed for a set of symbols with RLOC parameters. Unlike the RLOC\_ORIGIN parameter that “fixes” each member of a set at an absolute location, the RLOC\_RANGE parameter allows the members of the set to be located anywhere within the range, as long as the RLOC relationships are maintained. While the RLOC\_ORIGIN value is added to the RLOC values of the symbols to “fix” the placement, the RLOC\_RANGE value is kept as a separate parameter on each of the symbols in the set. During the placement process, the placer function must check that the two requirements are met for the symbol - that its relative location relationships are maintained, and that it is not placed outside of its allowed LCA range.

Usually, the RLOC\_RANGE parameter is attached to a non-primitive symbol that defines the “top” of a set of RLOC symbols. The RLOC\_RANGE will be propagated to every symbol in the RLOC set by the Xilinx merge program. Because a U\_SET is unrelated to hierarchy, the only way to specify a range for a U\_SET is to attach the RLOC\_RANGE to one of the members in the U\_SET.

## SYSTEM=value

The SYSTEM=value parameter is used to record and transfer information between programs. It is used by programs that generate XNF design files (i.e., schematic-to-XNF) and by specific XACT Development System programs. It should not be documented as available to the designer.

The value field may be any legal XNF string field. Multiple SYSTEM values may be specified for the same symbol by using a “;” to separate the values within the field (i.e., SYSTEM=XMACRO;FILEDEF).

XACT development system programs will add and delete information in the value field of the system parameter. Programs that read and write XNF files but do not operate on the SYSTEM parameters, should simply maintain the parameter as part of the SYM record in the output file if it existed in the input file.

A SYSTEM parameter value of “XMACRO” specifies that the symbol is a Xilinx macro symbol that should be flattened by the Xilinx merge program and not regarded as a distinct level of design hierarchy. In many design entry programs, some symbols that appear as “primitives” to the designer are actually macro symbols. Because designers see these symbols as logic primitives, they expect the logic for these symbols to be included in the user hierarchy level that includes the primitive symbols. Because design hierarchy can seriously affect design implementation, it is important to be able to flatten the Xilinx macros and leave only the user-defined macros as distinct levels of hierarchy. The “SYSTEM=XMACRO” parameter on a symbol will cause the Xilinx merge

program to flatten the logic of each Xilinx macro symbol into the encompassing level of design hierarchy.

DIVIDE1\_BY=value and DIVIDE2\_BY=value

DIVIDE1\_BY=value and DIVIDE2\_BY=value are user-defined parameters added to SYM records for the XC5200 OSC52 symbol.

The value field for these attributes has the following syntax:

DIVIDE1\_BY=[4, 16, 64, 256]

DIVIDE2\_BY=[2, 8, 32, 128, 1024, 4096, 16384, 65536]

These attributes define the amount by which the clock input to the OSC52 symbol is to be divided. If no clock input is specified, then the internal 16MHz clock will be used. The user may choose to use one or both of the outputs ('OSC1' and/or 'OSC2'), but the DIVIDE1\_BY=value attribute is required for 'OSC1' and the DIVIDE2\_BY=value is required for 'OSC2'.

OSC=value

The OSC=value parameter is used to define whether the source clock for the OSC52 is the internal 16 MHz oscillator or a user-defined clock signal. The value may be "INTERNAL," indicating that the internal 16 MHz oscillator is the source, or "USER," indicating that a user-defined signal will drive the oscillator clock pin.

Examples:

OSC=INTERNAL

OSC=USER

## I/O Block Parameters

I/O block parameters are allowed on only I/O symbols (such as IBUF, OBUF, INFF, etc.) or the EXT record that corresponds to the external connection of the I/O symbol. They apply to the IOB where the I/O symbol or external signal is placed. The TTL and CMOS parameters are not allowed on EXT records, because these parameters are ambiguous on an EXT record.

In addition to the LOC, BLKNM, and HBLKNM parameters described in the previous section, the following parameters are supported for I/O symbols and EXT records:

### INTERNAL

On certain LCA packages for XC2000, XC3000, XC4000, and XC5200 family LCAs there are more IOBs on the die than there are I/O pins on the package. In this case the unbonded IOBs may still be used, usually as extra flip-flops. Those IOBs which may be assigned to unbonded IOBs may be identified with the INTERNAL option on the I/O symbols or on the EXT record, or by specifying the 'U' direction on the EXT record. The information about internal IOBs is passed along to the Xilinx placement and routing programs, where it is necessary for determining optimal placement of the IOBs. This is allowed in the XC5200 family, even though there are no flip-flops in the XC5200 family IOBs.

### FAST or SLOW

On the XC3000, XC4000, and XC5200 families of LCAs the transition time of the output driver may be programmed to be either fast or slow. The SLOW option increases the transition time which reduces the noise level in the circuit, and is the default. The FAST option may be used to decrease the transition time of the output driver which also increases the noise in the system. These parameters are not allowed on XC4000H family parts. See the data sheets for the XC3000, XC4000 and XC5200 families for more information.

### MEDFAST or MEDSLOW

On the XC4000A family of LCAs the transition time of the output driver may be programmed to be either fast, slow, medfast or medslow.

### NODELAY

On the XC4000 and XC5200 family LCAs only, the NODELAY option may be used to reduce the Input Set-up Time requirement for the XC4000 INFF, INLAT and INREG, as well as for the XC5200 DFF and DLAT. For the 5000 family, the NODELAY parameter should be applied to the IBUF symbol feeding the DFF or DLAT. This attribute may be applied to the XC4000 INFF, INLAT, and INREG, or on the XC5200 IBUF symbol only. The default is with delay, which eliminates the hold-time requirement on the XC4000 INFF, INLAT, and INREG, and on the XC5200 DFF and DLAT.



## TTL, CMOS, RES or CAP

The RES and CAP parameters are allowed on XC4000H I/O symbols and EXT records only. The TTL and CMOS parameters are allowed on XC4000H I/O symbols only.

Examples:

```
SYM, name1, IBUF, TTL
PIN,...
...
END
EXT, sig1, O,, CAP
```

Legal combinations of these XC4000H properties are:

```
Output = {CAP or RES} {TTL or CMOS}
Input = {TTL or CMOS}
```

It is also illegal to specify Output = TTL and Input = CMOS on a single IOB, although the other combinations of TTL and CMOS are allowed.

## Signal Parameters

The following signal parameters may be defined on SIG records:

### X

The X parameter on a signal tells the partitioning algorithm to make this signal an explicit LCA net. Any signal which does not have the X parameter specified for it may be pulled into a CLB logic function during partitioning. The X parameter is useful for controlling the partitioning algorithm in certain cases. Any signal specified as an input or output signal of a CLBMAP symbol will automatically be treated as an explicit LCA net.

### W=weight

The W parameter indicates the routing order of the specified net. A net weight is used to specify the level of criticality for the net. Legal values are 1 - 99, where 99 indicates the most critical net (i.e., the highest in the routing order). This parameter affects the partitioning algorithm and is also passed along to the place and route program.

### C

The C parameter indicates that the signal is on a critical path and that all efforts should be made to minimize the delay through this signal. This gives the net a weight of 100. This parameter affects the partitioning algorithm and is also passed along to the place and route program.

### SC

The SC parameter indicates that the signal is Skew Critical and that the difference between load delays on this net must be minimized. This parameter may be used with the net weight (W) parameter.

### N

The N parameter indicates that the signal's timing is non-critical and that all other signals should take precedence over it. This gives the net a weight of 0. This affects the partitioning algorithm and is also passed along to the place and route program.

### L

The L parameter is passed along to the partition, place and route program and specifies that this signal should be routed using a longline on the LCA. This parameter is available for the XC2000 and XC3000 family only.

## S

The S parameter tells the netlist preparation program to save this signal and treat it as an external connection. This parameter is useful when processing an incomplete design.

### TNM=name\_list

The TNM parameter is used to define groups of path endpoints for use in TIMESPEC specifications. The name\_list consists of one or more logical names, separated by semicolon characters (;). Each logical name is a combination of letters, digits and underscores (\_). The TNM parameter is allowed on I/O flip-flops, CLB flip-flops, I/O latches, CLB latches, RAMs, RAMSs, RAMDs and EXT records, as well as on macro symbols, pins and signals.

When a TNM is specified on a signal, it will indicate to the Xilinx place and route program that flip-flops driven by that signal belong to the named group(s). To specify that a group name should be associated with a different type of storage element, each logical name may be prefaced with a qualifier followed by a colon (:) character. The valid qualifiers are: “ffs”, “rams” and “latches”. (An unqualified group name is equivalent to one qualified by “ffs”.)

Note that the “pads” qualifier is not allowed on TNM parameters on Signal records (although it is allowed on TNM parameters on Symbol records).

Examples:

TNM=ENA;CNTENA

TNM=FFS:REG01;LATCHES:ADDR

### TSidentifier

The TSidentifier parameter is used on SIG records in conjunction with the TIMESPEC symbol. It is used to flag a specific signal with a TS parameter which is also defined on the TIMESPEC symbol. (See TIMESPEC symbol for more information.) The identifier is a name consisting of any combination of letters, digits or underscores. It should be kept short for convenience and clarity.

Example:

SIG, name, TS01

### PIN=pinname

The PIN parameter is used in XNF files which define a macro to specify the pin name on the macro symbol which connects to the signal. If the pin name and signal name are identical, the PIN parameter is not necessary. The PIN parameter is ignored if found in a file which is not being expanded as a macro.

## Pin Parameters

The following parameters are allowed on the PIN records:

### INV

The INV parameter inverts the sense of the pin. For example, when specified for the clock pin of a DFF it makes the DFF negative edge-triggered instead of positive edge-triggered. Only certain pins may have the INV parameter specified for them. Those pins which are invertible are identified in the Symbol Definitions Section.

### I or K or G

In the XC2000 family of LCAs there are three ways to bring in a signal which drives the clock pin of the CLB flip-flop (or the enable pin of the CLB latch). The signal may be brought in on the dedicated CLB K pin (which connects to only a long line) or it may be brought in on the CLB C pin or it may be driven by the G logic function of the CLB. Each of these different options has significantly different timing characteristics. To allow the designer to minimize clock skew, he must be given complete control over which method is used.

By default, the K pin will be used for signals which go to multiple CLB clocks, and the C pin or the G function will be used for signals driving a single clock pin. The K, I and G parameters may be used to override these defaults. The K parameter tells the program to use the K pin of the CLB to bring in this signal. The I parameter forces the Input pin (C) to be used, and the G parameter forces the logic function G to be used. These parameters are not allowed on any pin which is not the clock pin of a CLB flip-flop or the enable pin of a CLB latch. In the XC3000 and XC4000 families, the clock may be driven by only the K pin of the CLB so these parameters are not allowed for XC3000, XC4000, and XC5200 series parts.

### P

The P (Pin lock) parameter is used with the CLBMAP, FMAP and CLB primitive symbol pins only. It allows a user to lock signals to the CLB pins on an individual basis. If a user wishes to lock all signals to the map symbol pins, they can either use a P pin parameter on all the pins or use the MAP=PLC parameter on the map symbol.

### TNM=name\_list

The TNM parameter is used to define groups of path endpoints for use in TIMESPEC specifications. The name\_list consists of one or more logical names, separated by semicolon characters (;). Each logical name is a combination of letters, digits and underscores (\_). The TNM parameter is allowed on I/O flip-

flops, CLB flip-flops, I/O latches, CLB latches, RAMs, RAMSs, RAMDs and EXT records, as well as on macro symbols, pins and signals.

For schematic packages that may not handle PIN parameters, a SIG TNM parameter may be used instead. The Xilinx software will then “push” the parameter from the SIG record onto input PIN records. Schematic → XNF translators must handle the SIG TNM parameter. However, for those schematic packages that accommodate pin properties, TNM parameters should be placed on load pins, and the schematic → XNF translators will simply output the TNM parameter on the input PIN record, where it will be appropriately handled by the Xilinx software.

When a TNM is specified on a pin, it will indicate to the Xilinx place and route program that flip-flops driven by that pin belong to the named group(s). To specify that a group name should be associated with a different type of storage element, each logical name may be prefaced with a qualifier followed by a colon (:) character. The valid qualifiers are: “ffs”, “rams” and “latches”. (An unqualified group name is equivalent to one qualified by “ffs”.)

Note that the “pads” qualifier is not allowed on TNM parameters on Pin records (although it is allowed on TNM parameters on Symbol records).

Examples:

TNM=ENA;CNTENA

TNM=FFS:REG01;LATCHES:ADDR

### TSidentifier

The *TSidentifier* parameter used in conjunction with TIMESPEC. The *identifier* is a name consisting of any combination of letters, digits or underscores. It should be kept short for convenience and clarity. For schematic packages that may not handle PIN parameters, a SIG *TSidentifier* parameter may be used instead. The Xilinx software will then “push” the parameter from the SIG record onto input PIN records. Schematic → XNF translators must handle the SIG *TSidentifier* parameter. In addition, for those schematic packages that accommodate pin properties, TS parameters can also be placed on load pins, in which case the schematic → XNF translators will simply output the parameter on the PIN record where it will be appropriately handled by the Xilinx software.

Example:

PIN, name, I, signal,,TS01

## User-defined Parameters

Parameters with no type (i.e., of the form “=val”) are user defined parameters. The val field may contain any characters except for a comma or a newline. Currently Xilinx software makes use of user parameters for defining X-BLOX parameters on symbols and for timing parameters on signals.

Programs which re-write the netlist file, such as the merge program, will maintain these parameters on the PIN, SIG, SYM or EXT records.

## Xilinx's Use of User-defined Parameters

=type[=value]

This is the format used for specifying attributes to be used by the Xilinx X-BLOX software. A complete list of supported X-BLOX properties can be obtained from Xilinx.

## Parameter Summary

The following is a summary of the parameters allowed on each record type (see previous section for definition of parameters):

### SYM Records

LOC=nn (Location specification)  
 LOC<>nn (Location prohibit)  
 RLOC=value (Relative location)  
 USE\_RLOC=value  
 U\_SET=name (User-defined set)  
 HU\_SET=name (Hierarchical user-defined set)  
 RLOC\_ORIGIN=value (Relative location origin)  
 RLOC\_RANGE=value (Relative location range)  
 BLKNM=name (Ignored on logic, clock, and OSC symbols)  
 HBLKNM=name (Ignored on logic, clock, and OSC symbols)  
 INTERNAL (I/O symbols only)  
 FAST (I/O symbols in XC3000, XC4000, and XC5200 only)  
 SLOW (I/O symbols in XC3000, XC4000, and XC5200 only)  
 MEDFAST (I/O symbols in XC4000A only)  
 MEDSLOW (I/O symbols in XC4000A only)  
 NODELAY (XC4000 and XC5200 I/O symbols only)  
 TTL (I/O symbols in XC4000H only)  
 CMOS (I/O symbols in XC4000H only)  
 RES (I/O symbols in XC4000H only)  
 CAP (I/O symbols in XC4000H only)  
 DOUBLE (PULLUP symbol in XC3000 and XC4000 only)  
 FILE=filename (Macro symbols only. Used by merge program)  
 MAP=type (CLBMAP, FMAP, HMAP and F5MAP symbols only)  
 INIT=value (XC4000 flip-flops and ROM only)  
 DEF=type  
 DECODE (WAND symbol in XC4000 only)  
 EQN=equation  
 TNM=name\_list (used with TIMESPEC)  
 TSidentifier=specification (used with TIMESPEC)  
 CYMODE=carry\_mode (XC4000 family CY4 and CY4\_XX carry mode symbols)  
 SCHNM=name (any symbol)  
 LIBVER=version\_string (all Unified Library symbols)  
 DIVIDE1\_BY=value (division value for XC5200 OSC52 symbol 'OSC1' pin)  
 DIVIDE2\_BY=value (division value for XC5200 OSC52 symbol 'OSC2' pin)  
 OSC=value (clock source for OSC52 symbol)

## PIN Records

INV (Allowed on invertible pins only)  
K (Ignored on non-clock pins, XC2000 only)  
I (Ignored on non-clock pins, XC2000 only)  
G (Ignored on non-clock pins, XC2000 only)  
P (CLBMAP, FMAP and CLB primitive symbol pins only)  
TNM=name\_list, "pads" qualifier not allowed (used with TIMESPEC)  
*TSidentifier* (used with TIMESPEC)

## SIG Records

X  
W=weight (Legal values are 1- 99 inclusive)  
C  
N  
SC  
L (XC2000 and XC3000 only)  
PIN=name (Macro files only. Used by merge program)  
S  
TNM=name\_list, "pads" qualifier not allowed (used with TIMESPEC)  
*TSidentifier* (used with TIMESPEC)

## EXT Records

LOC=nn (Location specification)  
LOC<>nn (Location prohibit)  
BLKNM=name  
HBLKNM=name  
INTERNAL  
FAST (XC3000, XC4000, and XC5200 family only)  
SLOW (XC3000, XC4000, and XC5200 family only)  
MEDFAST (XC4000A family only)  
MEDSLOW (XC4000A family only)  
NODELAY (XC4000 family only)  
RES (I/O symbols in XC4000H only)  
CAP (I/O symbols in XC4000H only)  
TNM=name\_list, no qualifiers allowed (used with TIMESPEC)

## USER Parameters

=val (allowed on PIN, SIG, SYM and EXT records)



## LCA Library

This section describes the symbol types supported in the Xilinx netlist file for LCA designs. A symbol needs to be created in the schematic editor's LCA library for each of these types. Complex functions may be built from these symbols if the schematic editor supports hierarchical design. The following terms are used in the descriptions of the library symbols:

### SYMBOL TYPE

The symbol type is the name which must be used in the type field of SYM records for this symbol.

### PIN NAMES

The pin names are the names which must be used in the pin name field of the PIN records. The names of each pin should be exactly the same name as specified for each symbol listed in the next section. Note that unconnected pins should be represented by simply not having a pin record for them in the file.

### INVERTIBLE PINS

Invertible pins are those pins which may be specified as inverted in the netlist by specifying the INV parameter on the PIN record. An inverted pin corresponds to a "bubbled" input or output pin on a schematic.

The LCA architecture is unique in that inverters almost always come for free. This is because of the nature of the Boolean logic generators which can implement ANY logic function of its inputs. There are also programmable inverters in strategic parts of the LCA, such as clock inputs and three-state output buffers which allow a free inversion. Invertible pins are those pins where a free inversion is available.

It is not necessary to use the invertible pins to take advantage of this feature of the LCA, however. The Xilinx technology mapping program will use these features of the LCA to implement inverters wherever possible. This means that the developer of the schematic library does not need to support the invertible pins. The designer could just use inverters without adversely affecting the ultimate LCA design.

Note that only certain pins may be inverted automatically. An INV parameter on a non-invertible pin is a fatal error which will cause the conversion program to abort with an appropriate message.

### XC2000, XC3000, XC4000, AND XC5200 FAMILIES

The term 'XC2000 family' refers to the first LCA architecture. The term 'XC3000 family' refers to the second generation of LCAs, which includes enhancements to the first LCA architecture. These enhancements include on-chip three-state drivers for implementing internal busses and wide input logic functions, two flip-

flops per CLB, functions of up to 5 inputs, 2 flip-flops per IOB and many other improvements. The term 'XC4000 family' refers to the third generation of LCAs which includes further enhancements to the XC3000 family. These enhancements include wide decode logic, on-chip RAM or ROM, fast carry logic, global three-state of I/O buffers, greater logic density and more extensive routing resources.

The term 'XC5200 family' refers to the fourth generation of LCAs, which features a new carry logic structure (CY\_MUX), a new primitive for combining two 4-input function generators to create a 5-input function (F5\_MUX, F5MAP), a new oscillator symbol (OSC52), and improved boundary scan (BSCAN).

There are some symbols such as DLAT that are available in only the XC2000 and XC5200 families but not in the XC3000 and XC4000 families. Other symbols such as OUTFF are available in the XC3000 and XC4000 families but not in the XC2000 and XC5200 families. Some symbols such as RAM/ROM and WAND exist in only the XC4000 family. Also, symbols like DFF and DLAT have different input pins depending on the family. The simplest approach for dealing with the different architectures is to create a separate library for each one. This technique restricts the designer to using only those symbols that are available for the selected LCA family.

There are variations on the XC3000 and XC4000 family such as XC3000A and XC4000H. Symbols and parameters that are specific to these architectures will be documented in this specification.

XC7000 EPLD designs use some additional symbols not listed in this specification; refer to the XNF XC7000 Specification for details.

## FLIP-FLOPS and LATCHES

The LCA contains D type flip-flops in the CLB (and sometimes IOB) blocks. The flip-flops in the CLBs have different capabilities and timing characteristics than those in the IOBs. The CLB flip-flops also differ depending on the LCA family. In addition, some flip-flops (namely, the CLB flip-flop in the XC2000 and XC5200 families, and the input flip-flop in the IOB of the XC3000 and XC4000 families) may optionally be configured as level-sensitive D-type latches rather than as edge-triggered D-type flip-flops.

Because of the differences in timing and capabilities of the various flip-flops, the designer must specify which type of flip-flop to use. In the symbol definitions below, there are symbols corresponding to the CLB flip-flops and others corresponding to the IOB flip-flops. There are still others corresponding to the latch options.

In addition to differences between the CLB and IOB flip-flops in a single family, each architecture has its own features and the symbols to support them. Some symbols are defined for one family and not another, while others may have different pins depending on the LCA family used. These differences may affect decisions about which architecture is appropriate for a particular application. For

a summary of the differences between the XC2000, XC3000, XC4000, and XC5200 family architectures, refer to Section 1 of the Xilinx Programmable Logic Data Book. For more detailed information about those architectures, refer to Section 2 of the Xilinx Programmable Logic Data Book.

There is one feature of all the LCA flip-flops that must be taken into account by the simulation translator. There is a dedicated input pin called RESET on the XC2000, XC3000, and XC5200 LCA that will asynchronously reset all flip-flops in a configured LCA. The RESET input is described in Section 2 of the Xilinx Programmable Logic Data Book.

Because it is a fixed, non-configurable feature, the global reset function is not shown in the schematic explicitly, but it must be modeled during simulation. To facilitate this, a GR pin is listed as an input pin on all the latch and flip-flop symbols described here. This pin should NOT be part of the normal schematic symbol, because its connection to the global reset input signal is fixed. The global reset signal will be created and connected to all GR pins when the delay information is written to the netlist file by the Xilinx program that creates an XNF file from the LCA.

For the XC4000 family flip-flops, the use of this dedicated pin is slightly different. It is a global Set/Reset function which will individually set or reset each flip-flop depending on how the flip-flop is configured. For the XC4000 family the name of this global set/reset pin is GSR and the connections to the GSR pins are created when the delay information is written to the netlist file by the Xilinx program that generates an XNF file from an LCA. The GSR input is described in Section 2 of the Xilinx Programmable Logic Data Book.

## Symbol Definitions

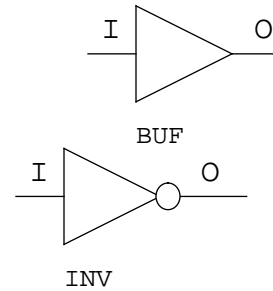
The following are detailed descriptions of each of the library symbols supported in the LCA Xilinx netlist file.

## Inverter/Buffer Symbols

Symbol Types: BUF INV

XC2000, XC3000, XC4000 and XC5200 Families

Input pins: I  
Output pins: O  
Invertible pins: All



Inverters are treated specially by the mapping program. Generally, a discrete inverter is an inefficient element in LCA designs. To generate the complement of a signal, an entire function generator must be used. This same function generator could have been used to implement a function of 3 or 4 variables.

The LCA architecture provides a number of features which generally make inverters unnecessary. The program which prepares a netlist file for mapping into an LCA design has an inverter optimization algorithm which makes full use of the LCA architectural features to eliminate inverters wherever possible. This means that designers may use inverters freely without increasing the LCA resources needed for a design.

Buffers will always be removed, because there is no reason to re-drive a signal in the LCA architecture. They can be used during schematic capture to break up nets to give different pin parameters to each pin on the same net. After pin parameters have been transferred to the XNF format, the Xilinx mapping software will remove any buffers in the design.

The pin names for these symbols must match those specified above for each symbol.

## Combinational Logic Symbols

Symbol Types: AND NAND OR NOR XOR XNOR

### XC2000 Family

Input pins: I0, I1, I2, I3 \*  
Output pins: O \*  
Invertible pins: All

### XC3000, XC4000, and XC5200 Families

Input pins: I0, I1, I2, I3, I4 \*  
Output pins: O \*  
Invertible pins: All

These symbols are used to implement combinational logic functions. The technology mapping programs group these symbols together and implement the resulting logic function of each group with a Boolean function generator in a CLB.

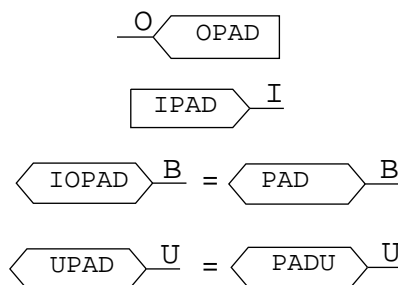
The Boolean function generator on the XC2000 family can implement any function of 4 inputs, so at most 4 inputs are allowed on a logic gate. When using the XC3000, XC4000, or XC5200 family of parts, up to 5 inputs may be used. In the schematic library, symbols with 2, 3 and 4 inputs should be provided, as well as 5 inputs for XC3000, XC4000, and XC5200 libraries. Combinational gates with only one input are not allowed.

Inversion may be specified on any pin, input or output. This may be useful if DeMorgan equivalents are supported by the schematic editor. Note that there are two ways to specify a NAND function: either with a NAND symbol type or with an AND symbol type with the output pin flagged as inverted. Also, a NAND symbol with the output flagged as inverted implements an AND function.

\* Unlike the other symbols defined here, the names of the inputs and outputs do not have to be the same as those listed above for XC2000, XC3000, XC4000, and XC5200 designs. Because all inputs are equivalent, and only one output is available, the direction field of the PIN record is sufficient to define the function of the pin. However, all pins must have a name which is unique from all the other pins on the symbol.

## Input/Output Pad Symbols

Symbol Types: IPAD  
OPAD  
IOPAD  
UPAD  
PAD\* (Obsolete)  
PADU\* (Obsolete)



XC2000, XC3000, XC4000, and XC5200 Families

Input pins: O (OPAD)  
Output pins: I (IPAD), B (IOPAD), U (UPAD)  
Invertible pins: None

The Pad symbols correspond to external connections of the LCA design. These symbols should be translated directly to an EXT record described earlier in this specification. They should not appear in SYM records in the XNF file.

\* The PAD symbol can be used as either a bidirectional pad, input pad or output pad. It has one pin that is bidirectional. This symbol should translate to an EXT record with a direction of “B”, “I” or “O” depending on whether it is bidirectional, input or output. Note that the direction can always be specified as “B” and the Xilinx software will handle this. This symbol will be obsolete in the Unified library. Instead there will be IOPAD, IPAD and OPAD symbols which correspond to the appropriate EXT records with direction B, I, and O respectively.

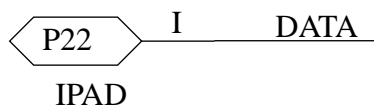
The UPAD symbol is used when the use of an unbonded pad is desired. This symbol should translate directly to an EXT record with a direction of “U”.

\* The PADU symbol will be obsolete in the Unified library and is replaced with the UPAD symbol.

Each of these symbols can connect to only the external pin of the I/O symbols described in the following sections. The IPAD symbol may also connect to the input pins of: the ACLK, GCLK and BUFG in the XC3000 family; the BUFGP and BUFGS in the XC4000 family; and the BUFG in the XC5200 family. Any parameters allowed on EXT records can be attached to these symbols in the schematic drawing.

Example:

EXT,DATA,I,,LOC=P22

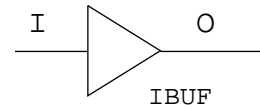


## Input Buffer

Symbol Type: IBUF

XC2000, XC3000, and XC4000 Families

External input pin: I  
Output pin: O  
Invertible pins: None



XC5200 Family

External input pin: I  
Output pin: O  
Invertible pins: O

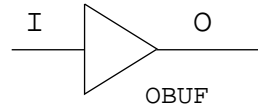
The IBUF symbol corresponds to the input buffer in the IOB of the LCA. The signal connected to the I pin is an external (off-chip) signal. The I pin may be connected to the output of an OBUF, OBUFT, OUTFF or OUTFFT symbol in addition to an IPAD, IOPAD or UPAD symbol. In the XC4000 family, the I pins of the special input pads (TDI, TMS, TCK, MD0 and MD2) may also connect to the I pin of this symbol. In the XC5200 family, the I pins of the special input pads (TDI, TMS, TCK, MD0, and MD2) and the O pins of the special output pads (TDO and MD1) may also connect to the I pin of this symbol. Only one of the symbols INFF, INLAT, INREG and IBUF may be used in a single IOB, except that an IBUF may be used to provide a buffered version of the D input of an INFF or INLAT in the XC3000 and XC4000 families.

## Output Buffer

Symbol Type: OBUF

XC2000 Family

Input pin: I  
External output pin: O  
Invertible pins: None



XC3000 Family

Input pin: I  
External output pin: O  
Invertible pin: I

XC4000 and XC5200 Families

Input pins: I, GTS  
External output pin: O  
Invertible pin: I\*, GTS

The OBUF symbol corresponds to the output buffer in the IOB of the LCA. The signal connected to the O pin is an external (off-chip) signal. The O pin may be connected to the I pin of an IBUF or the D pin of an INFF, INLAT or INREG symbol in addition to an OPAD, IOPAD or UPAD symbol. In the XC3000 family, the I pins of the clock drivers (ACLK, GCLK and BUFG) may also connect to the O pin of this symbol. In the XC4000 family, the I pins of the global clock buffers (BUFGS, BUFGP, and BUFG), the I pins of the TDI, TMS and TCK special input pads, and the O pins of the MD1 and TDO special output pads may also connect to the O pin of this symbol. In the XC5200 family, the I pins of the global clock buffers (BUFG) and the special input pads (TDI, TMS, TCK, MD0 and MD2) and the O pins of the special output pads (MD1 and TDO) may also connect to the O pin of this symbol. Only one of the symbols OBUF, OBUFT, OUTFF and OUTFFT may be used in a single IOB.

The I pin is invertible on only the XC3000, XC4000, and XC5200 families of LCAs.

\* For the XC4000 family, the I pin is not invertible if the O pin is connected to an MD1 or TDO special pad.

The connection to the GTS pin will be added by the Xilinx program which creates an XNF file from an LCA. The GTS pin is the Global Three-State pin on all the output symbols in the XC4000 and XC5200 families. When the input to this pin is high (logical 1), the output becomes high impedance. This pin should not be shown in the schematic but should be modeled by simulators.

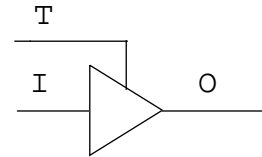


## Three-State Output Buffer

Symbol Type: OBUFT

XC2000 Family

Input pins: I, T  
 External output pin: O  
 Invertible pins: None



XC3000 Family

Input pins: I, T  
 External output pin: O  
 Invertible pins: I, T

XC4000 and XC5200 Families

Input pins: I, T, GTS  
 External output pin: O  
 Invertible pins: I\*, T\*, GTS

The OBUFT symbol is a three-state buffer in the IOB of the LCA. The signal connected to the O pin is an external (off-chip) signal. The O pin may be connected to the I pin of an IBUF or the D pin of an INFF, INLAT or INREG symbol in addition to an OPAD, IOPAD or UPAD symbol. In the XC3000 family, the I pins of the clock drivers (ACLK, GCLK and BUFG) may also connect to the O pin of this symbol. In the XC4000 family, the I pins of the global clock buffers (BUFGS, BUFGP, and BUFG), the I pins of the TDI, TMS and TCK special input pads, and the O pins of the MD1 and TDO special output pads may also connect to the O pin of this symbol. In the XC5200 family, the I pins of the global clock buffers (BUFG) and the special input pads (TDI, TMS, TCK, MD0 and MD2) and the O pins of the special output pads (MD1 and TDO) may also connect to the O pin of this symbol. Only one of the symbols OBUF, OBUFT, OUTFF and OUTFFT may be used in a single IOB.

Note that the T pin represents an active-high three-state enable. When the T pin is active (i.e., it is high or 1) the buffer's output is high impedance. The T pin could also be considered an active low output enable.

The I and T pins are invertible on only the XC3000, XC4000, and XC5200 families of LCAs.

\* For the XC4000 family, the I and T pins are not invertible if the O pin is connected to an MD1 or TDO special pad.

The TBUF symbol is also a three-state buffer, but it corresponds to the internal three-state buffers available on the XC3000, XC4000, and XC5200 families of parts. The OBUFT corresponds to the buffer in the IOB.

The connection to the GTS pin will be added by the Xilinx program which creates an XNF file from an LCA. GTS is the Global Three-State pin on all the output symbols in the XC4000 and XC5200 families. When the input to this pin is high (logical 1), the output becomes high impedance. GTS should not appear in the schematic but should be modeled by simulators.

## Output Flip-Flop

Symbol Type: OUTFF

XC2000 and XC5200 Families

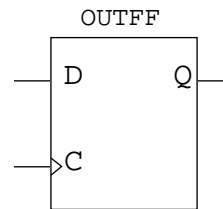
Not available

XC3000 Family

Input pins: D, C, GR

External output pin: Q

Invertible pins: D, C

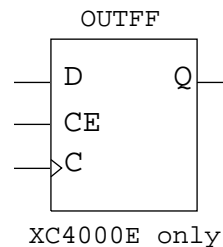


XC4000 Family

Input pins: D, C, GSR, GTS

External output pin: Q

Invertible pins: D, C, GSR, GTS



XC4000E Family

Input pins: D, CE, C, GSR, GTS

External output pin: Q

Invertible pins: D, C, GSR, GTS

The OUTFF symbol corresponds to the flip-flop in the output path of the IOB on the XC3000 and XC4000 family of LCAs. The OUTFF is not available in the XC2000 and XC5200 families. The signal connected to the Q pin is an external (off-chip) signal. The Q pin may be connected to the I pin of an IBUF or the D pin of an INFF, INLAT or INREG symbol in addition to an OPAD, IOPAD or UPAD symbol. In the XC3000 family, the I pins of the clock drivers (ACLK, GCLK and BUFG) may also connect to the Q pin of this symbol. In the XC4000 family, the I pins of the global clock buffers (BUFGS, BUFGP, and BUFG) and the I pins of the TDI, TMS and TCK special input pads may also connect to the Q pin of this symbol. Only one of the symbols OBUF, OBUFT, OUTFF and OUTFFT may be used in a single IOB.

The XC4000E family OUTFF has a CE pin. When the CE pin is low, the output pin Q does not change.

The OUTFFT may be used instead if a three-state output from the flip-flop output is desired.

The GR pin on the XC3000 family corresponds to the fixed connection to the global reset signal of the LCA.

The GSR pin in the XC4000 family corresponds to the fixed connection to the global set/reset signal of the LCA. The INIT property is used to determine whether the flip-flop will be set or

reset (INIT=S or INIT=R) upon power-on initialization and assertion of the GSR signal. If no INIT property is given, the default is reset.

The GTS pin is the Global Three-State pin on all the output symbols in the XC4000 family. When the input to this pin is high (logical 1), the output becomes high impedance.

The connections to the GR, GSR and GTS pins will be added by the Xilinx program that creates an XNF file from an LCA. These pins should not appear in a schematic or in an XNF file that is input to the Xilinx tools. They appear only after back-annotation is performed on the placed and routed LCA file. They must be modeled in all XNF simulators.

## Three-State Output Flip-Flop

Symbol Type: OUTFFT

XC2000 and XC5200 Families

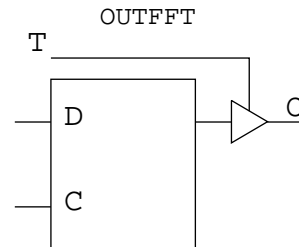
Not available

XC3000 Family

Input pins: D, T, C, GR

External output pin: O

Invertible pins: D, T, C

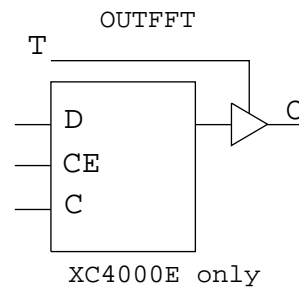


XC4000 Family

Input pins: D, T, C, GSR, GTS

External output pin: O

Invertible pins: D, T, C, GSR, GTS



XC4000E Family

Input pins: D, CE, T, C, GSR, GTS

External output pin: O

Invertible pins: D, T, C, GSR, GTS

The OUTFFT symbol corresponds to the flip-flop and the three-state buffer in the output path of an IOB on the XC3000 and XC4000 family of LCAs. The OUTFFT is not available in the XC2000 family. The signal connected to the O pin is an external (off-chip) signal. The O pin may be connected to the I pin of an IBUF or the D pin of an INFF, INLAT or INREG symbol in addition to an OPAD, IOPAD or UPAD symbol. In the XC3000 family, the I pins of the clock drivers (ACLK, GCLK and BUFG) may also connect to the O pin of this symbol. In the XC4000 family, the I pins of the global clock buffers (BUFGS, BUFGP, and BUFG) and the I pins of the TDI, TMS and TCK special input pads may also connect to the O pin of this symbol. Only one of the symbols OBUF, OBUFT, OUTFF and OUTFFT may be used in a single IOB.

The XC4000E family OUTFFT has a CE pin. When the CE pin is low, the output pin O does not change.

The OUTFF may be used instead, if the three-state output is not desired. Similarly, if the T pin is not used, or if it is tied low (output enabled), the OUTFFT functions the same as the OUTFF. Note that the T pin represents an active high three-state enable. When the T pin is active (i.e., it is high or 1) the buffer's output is high impedance. The T pin could also be considered an active low output enable. The GR pin on the XC3000 corresponds to the fixed connection to the global reset signal of the LCA. The GSR pin on the XC4000 corresponds to the fixed connection to the global set/reset signal of the LCA. The INIT property is used to

determine whether the flip-flop will be set or reset (INIT=S or INIT=R) upon power-on initialization and assertion of the GSR signal. If no INIT property is given, the default is reset.

The GTS pin is the Global Three-State pin on all the output symbols in the XC4000 family. When the input to this pin is high (logical 1), the output becomes high impedance.

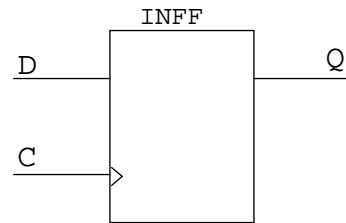
The connections to the GR, GSR and GTS pin will be added by the Xilinx program that creates an XNF file from an LCA. These pins should not appear in a schematic or in an XNF file that is input to the Xilinx tools. They appear only after back-annotation is performed on the placed and routed LCA file. They must be modeled in all XNF simulators.

## Input Flip-Flop

Symbol Type: INFF

### XC2000 Family

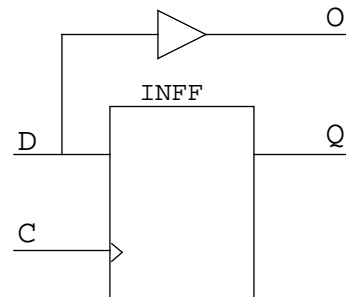
External input pin: D  
Input pins: C, GR  
Output pin: Q  
Invertible pins: None



XC2000 Version and Preferred  
XC3000 & XC4000 Versions

### XC3000 Family

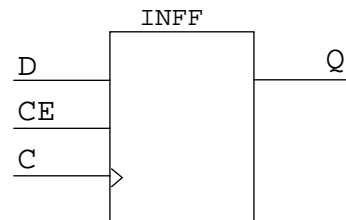
External input pin: D  
Input pins: C, GR  
Output pin: Q  
Invertible pin: C



XC3000 & XC4000 Versions  
(Old style)

### XC4000 Family

External input pin: D  
Input pins: C, GSR  
Output pin: Q  
Invertible pin: C, GSR



XC4000E Only

### XC4000E Family

External input pin: D  
Input pins: CE, C, GSR  
Output pin: Q  
Invertible pin: C, GSR

### XC5200 Family

Not available

The INFF symbol corresponds to the flip-flop in the input path of an IOB. The signal connected to the D pin is an external (off-chip) signal. The D pin may be connected to the output of an OBUF, OBUFT, OUTFF or OUTFFT symbol in addition to an IPAD, IOPAD or UPAD symbol. In the XC4000 family, the I pins of the special input pads (TDI, TMS and TCK) may also connect to the D pin of this symbol. Only one of the symbols INFF, INLAT, INREG and IBUF may be used in a single IOB, except that an IBUF may be used to provide a buffered version of the D input of an INFF or INLAT in the XC3000 and XC4000 families.

The XC4000E family INFF has a CE pin. When the CE pin is low, the output on Q does not change.

In the XC3000 and XC4000 family a buffered version of the D input signal is available on the O output pin. The preferred way to access the buffered output is to use an IBUF symbol in conjunction with this symbol and a signal should connect the D pin of the INFF to the I pin of the IBUF. To maintain compatibility with previous symbol libraries, the buffered output may also be accessed using the O pin of the INFF symbol.

In the XC3000 and XC4000 family, the INLAT symbol may be used instead if a level sensitive D type latch is desired. In the XC4000 family a default delay element exists on the data input so that no hold time is required. The NODELAY parameter can be used to remove this delay to reduce set-up time.

The GR pin in the XC2000 and XC3000 family corresponds to the fixed connection to the global reset signal of the LCA. The GSR pin in the XC4000 family corresponds to the fixed connection to the global set/reset signal of the LCA. The INIT property is used to determine whether the flip-flop will be set or reset (INIT=S or INIT=R) upon power-on initialization and assertion of the GSR signal. If no INIT property is given, the default is reset.

The connections to the GR and GSR pins will be added by the Xilinx program that creates an XNF file from an LCA. These pins should not appear in a schematic or in an XNF file that is input to the Xilinx tools. They appear only after back-annotation is performed on the placed and routed LCA file. They must be modeled in all XNF simulators.



## Input Latch

Symbol Type: INLAT

XC2000 and XC5200 Families

Not available

XC3000 Family

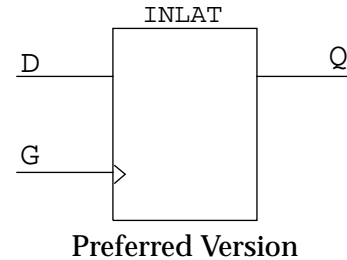
External input pin: D  
Input pins: G (or L)\*, GR  
Output pins: Q, (O)\*\*  
Invertible pins: G (or L)\*

XC4000 Family

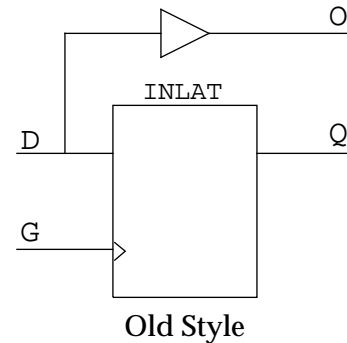
External input pin: D  
Input pins: G (or L)\*, GSR  
Output pins: Q, (O)\*\*  
Invertible pins: G (or L)\*, GSR

XC4000E Family

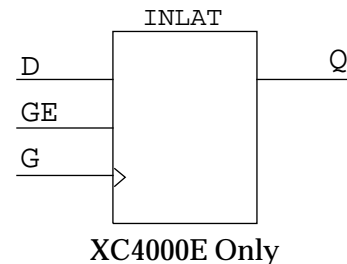
External input pin: D  
Input pins: GE, G, GSR  
Output pins: Q  
Invertible pins: G, GSR



Preferred Version



Old Style



XC4000E Only

The INLAT symbol corresponds to the flip-flop in the input path of an IOB configured as a transparent latch (HIGH for XC3000, LOW for XC4000). The INLAT function is not available in the XC2000 family. The signal connected to the D pin is an external (off-chip) signal. The D pin may be connected to the output of an OBUF, OBUFT, OUTFF or OUTFFT symbol in addition to an IPAD, IOPAD or UPAD symbol. In the XC4000 family, the I pins of the special input pads (TDI, TMS and TCK) may also connect to the D pin of this symbol. Only one of the symbols INFF, INLAT, INREG and IBUF may be used in a single IOB, except that an IBUF may be used to provide a buffered version of the D input of an INFF or INLAT in the XC3000 and XC4000 families.

The XC4000E family INLAT has a GE pin. When the GE pin is low, the output pin Q does not change.

\*\* A buffered version of the D input signal is available on the O output pin. The preferred way to access the buffered output is to use an IBUF symbol in conjunction with this symbol and a signal should connect the D pin of the INLAT to the I pin of the IBUF. To maintain compatibility with previous symbol libraries, the buffered output may also be accessed using the O pin of the INLAT symbol.

The INFF symbol may be used instead if an edge-triggered D type latch is desired. In the XC4000 family a default delay element exists on the data input so that no hold time is required. The NODELAY parameter can be used to remove this delay to reduce set-up time.

The GR pin in the XC2000 and XC3000 family corresponds to the fixed connection to the global reset signal of the LCA. The GSR pin in the XC4000 family corresponds to the fixed connection to the global set/reset signal of the LCA. The INIT property is used to determine whether the flip-flop will be set or reset (INIT=S or INIT=R) upon power-on initialization and assertion of the GSR signal. If no INIT property is given, the default is reset.

The connections to the GR and GSR pins will be added by the Xilinx program that creates an XNF file from an LCA. These pins should not appear in a schematic or in an XNF file that is input to the Xilinx tools. They appear only after back-annotation is performed on the placed and routed LCA file. They must be modeled in all XNF simulators.

\* Both pin names (G or L) are legal in version 6 XNF. If Unified libraries are used, a LIBVER parameter should exist for this symbol and the new pin name G should be used.

## Input Register

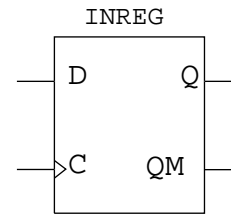
Symbol Type: INREG\* (Obsolete)

XC2000, XC3000, and XC5200 Families

Not available

XC4000 Family (Obsolete in Unified Library)

External input pin: D  
 Input pins: C, GSR  
 Output pins: Q, QM  
 Invertible pin: C, GSR



The INREG symbol corresponds to the flip-flop in the input path of a XC4000 family IOB only. It differs from the XC4000 family INFF primitive in that the output of the Master portion of the flip-flop is available in addition to the regular Slave flip-flop output. The INREG function is not available in the XC2000 and XC3000 families. The signal connected to the D pin is an external (off-chip) signal. The D pin may be connected to the output of an OBUF, OBUFT, OUTFF or OUTFFT symbol in addition to an IPAD, IOPAD or UPAD symbol. In the XC4000 family, the I pins of the special input pads (TDI, TMS and TCK) may also connect to the D pin of this symbol. Only one of the symbols INFF, INLAT, INREG and IBUF may be used in a single IOB, except that an IBUF may be used to provide a buffered version of the D input of an INFF or INLAT in the XC3000 and XC4000 families.

The Q output pin is identical to the Q output of the XC4000 family INFF primitive, i.e., it is the output of the flop-flop. The QM output pin is the Master output of the flip-flop in the IOB, i.e., it is the output of the Master latch. The INFF symbol may be used instead if only an edge-triggered D-type flip-flop is desired or the INLAT symbol may be used if a level-sensitive D-type latch is desired.

\* This symbol has been made obsolete in the Xilinx Unified Library. With the Unified Library, to get the functionality of an INREG, the C pin of an INFF and the G pin of an INLAT should be connected together, and the D pins of the INFF and INLAT symbols should also be connected together.

In the XC4000 family a default delay element exists on the data input so that no hold time is required. The NODELAY parameter can be used to get rid of this delay to reduce set-up time.

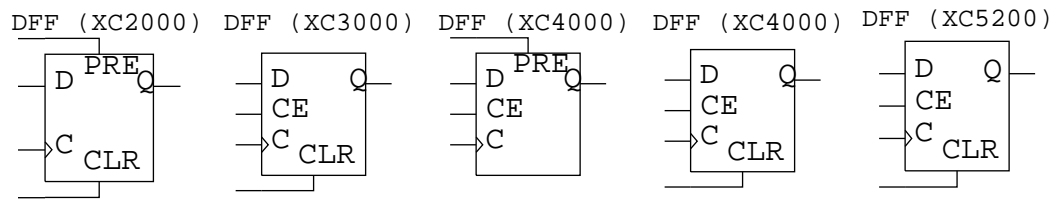
The GSR pin in the XC4000 family corresponds to the fixed connection to the global set/reset signal of the LCA. The INIT property is used to determine whether the flip-flop will be set or reset (INIT=S or INIT=R) upon power-on initialization and assertion of the GSR signal. If no INIT property is given, the default is reset.

The connection to the GSR pin will be added by the Xilinx program that creates an XNF file from an LCA. This pin should not appear in a schematic or in an XNF file that is input to the

Xilinx tools. This pin appears only after back-annotation is performed on the placed and routed LCA file. This pin must be modeled in all XNF simulators.

## D-Type Flip-Flop

Symbol Type: DFF



### XC2000 Family

Input pins: D, C, CLR (or RD)\*, PRE (or SD)\*, GR  
 Output pin: Q  
 Invertible pin: C

### XC3000 Family

Input pins: D, C, CE, CLR (or RD)\*, GR  
 Output pin: Q  
 Invertible pin: C

### XC4000 Family

Input pins: D, C, CE, CLR (RD)\* or PRE (SD)\*, GSR  
 Output pin: Q  
 Invertible pin: C, GSR

### XC5200 Family

Input pins: D, C, CE, CLR, GR  
 Output pin: Q  
 Invertible pins: C, GR

The DFF symbol corresponds to the edge-triggered D-type flip-flop in the CLB. Notice that the XC2000 version and the XC3000 version are quite different. The XC2000 version has asynchronous preset and clear and no clock enable. The XC3000 and XC5200 versions have a clock enable and asynchronous clear but no asynchronous preset. The XC4000 differs slightly from the XC3000 in that it can have either asynchronous preset or asynchronous clear. The best way to deal with this is to have two library symbols for the XC4000 DFF, one with a CLR pin and one with an PRE pin.

The clock pin on all four types may be inverted. The PRE and CLR inputs are active high.

The DLAT symbol defined below may be used for a level sensitive D type latch but is available in the XC2000 and XC5200 families only.

The GR pin in the XC2000, XC3000, and XC5200 families corresponds to the fixed connection to the global reset signal of the LCA.

The GSR pin in the XC4000 family corresponds to the fixed connection to the global set/reset signal of the LCA. Depending on whether the PRE or CLR pin is used, the flip-flop will be set or reset upon power-on initialization and assertion of the GSR signal. If neither PRE or CLR is used, the default is clear.

The connections to the GR and GSR pins will be added by the Xilinx program that creates an XNF file from an LCA. These pins should not appear in a schematic or in an XNF file that is input to the Xilinx tools. They appear only after back-annotation is performed on the placed and routed LCA file. They must be modeled in all XNF simulators.

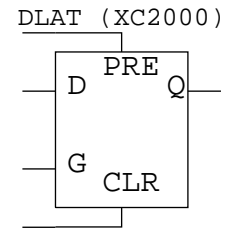
\* Both pin names (CLR or RD and PRE or SD) are legal in version 6 XNF. If Unified libraries are used, a LIBVER parameter should exist for this symbol and the new pin names CLR and PRE should be used.

## D-Type Latch

Symbol Type: DLAT

### XC2000 Family

Input pins: D, G (or L)\*, CLR (or RD)\*, PRE (or SD)\*, GR  
 Output pin: Q  
 Invertible pins: G (or L)\*

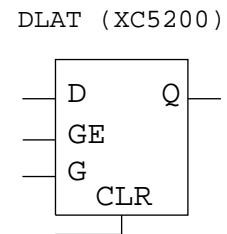


### XC3000 and XC4000 Families

Not available

### XC5200 Family

Input pins: D, G, CLR, GE, GR  
 Output pin: Q  
 Invertible pin: G, GR



The DLAT symbol corresponds to the CLB flip-flop configured as a level-sensitive D-type latch. Notice that the XC2000 version and the XC5200 version are quite different. The XC2000 version is transparent high; it has asynchronous preset and clear, but no clock enable. The XC5200 version is transparent high; it has a clock enable and an asynchronous clear, but no asynchronous preset.

The latch enable (G) pin on both types may be inverted. The PRE, CLR, and GE inputs are active high.

The DFF symbol described above may be used instead for an edge-triggered D-type flip-flop.

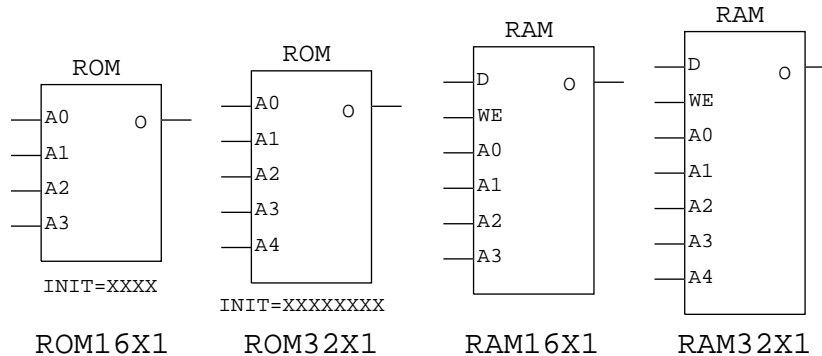
The GR pin corresponds to the fixed connection to the global reset signal of the LCA for both XC2000 and XC5200 families.

The connection to the GR pin will be added by the Xilinx program that creates an XNF file from an LCA. This pin should not appear in a schematic or in an XNF file that is input to the Xilinx tools. This pin appears only after back-annotation is performed on the placed and routed LCA file. This pin must be modeled in all XNF simulators.

\* Both pin names (G or L, CLR or RD, and PRE or SD) are legal in version 6 XNF. If Unified libraries are used, a LIBVER parameter should exist for this symbol and the new pin names G, CLR and PRE should be used.

## RAM/ROM

Symbol Types: RAM ROM



### XC2000, XC3000, and XC5200 Families

Not available

### XC4000 Family RAM

Input pins: A0, A1, A2, A3, A4, WE, D

Output pin: O

Invertible pins: None

### XC4000 Family ROM

Input pins: A0, A1, A2, A3, A4

Output pin: O

Invertible pins: None

In the XC4000 family, one or more function generators can be used as RAM or ROM. A 16x1 memory uses a single 4-input function generator (F or G). A 32x1 memory uses both 4-input function generators (F and G) plus the 3-input function generator (H). In the 32x1 case, the four lower address signals (A3-A0) are routed to both F and G, and the upper address signal (A4) is used to control a multiplexor in H.

The INIT parameter is required for the ROM symbol to define its pattern. The value must be a 4 (for 16x1) or 8 (for 32x1) digit hexadecimal number. INIT data is written from MSB (A<3:0>=1111) to LSB (A<3:0>=0000). For example a 16x1 bit ROM might have INIT=8F30, where the MSB is 1 and the LSB is 0. The INIT parameter is supported, although not required, for the XC4000E family RAM symbol; if no INIT parameter is specified, the RAM is initialized to all zeroes. The format is the same as for the INIT parameter on the ROM symbol.

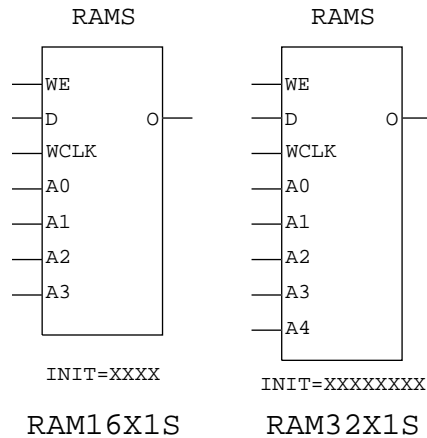
Note that although the name of the symbol in the library may be RAM16X1 or ROM32X1, the symbol type in the XNF file is ALWAYS simply RAM or ROM. The implementation software



determines whether a 16x1 or 32x1 memory is to be used based on the number of address inputs.

## RAMS

Symbol Types: RAMS



### XC2000, XC3000, XC4000 and XC5200 Families

Not available

### XC4000E Family

Input pins: A0, A1, A2, A3, A4, WE, WCLK, D

Output pin: O

Invertible pins: WCLK

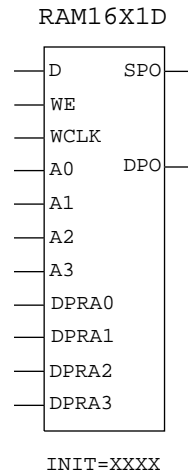
In the XC4000E family, one or more function generators can be used as synchronous RAM. A write to the RAM is triggered by the rising edge of the WCLK input (or falling edge if WCLK is inverted), whenever WE is high. A 16x1 RAMS uses a single 4-input function generator (F or G). A 32x1 RAMS uses both 4-input function generators (F and G) plus the 3-input function generator (H). In the 32x1 case, the four lower address signals (A3-A0) are routed to both F and G, and the upper address signal (A4) is used to control a multiplexor in H.

The INIT parameter is supported, although not required, for the RAMS symbol to define its initial pattern. The value must be a 4 (for 16x1) or 8 (for 32x1) digit hexadecimal number. INIT data is written from MSB (A<3:0>=1111) to LSB (A<3:0>=0000). For example, a 16x1 bit RAMS might have INIT=8F30, where the MSB is 1 and the LSB is 0. In the absence of the INIT parameter, the RAMS is initialized with zeros at every address.

Note that although the name of the symbol in the library may be RAM16X1S or RAM32X1S, the symbol type in the XNF file is ALWAYS simply RAMS. The implementation software determines whether a 16x1 or 32x1 memory is to be used based on the number of address inputs.

## RAMD

Symbol Types: RAMD



XC2000, XC3000, XC4000 and XC5200 Families

Not available

XC4000E Family

Input pins: A0, A1, A2, A3, DPRA0, DPRA1, DPRA2, DPRA3, WE, WCLK, D

Output pin: SPO, DPO

Invertible pins: WCLK

In the XC4000E family, both 4-input function generators (F and G) can be used as a 16x1 dual-port, synchronous RAM. There are two separate address ports, the read-only address (DPRA3-DPRA0) and the read/write address (A3-A0). These two address ports are completely asynchronous, with the read-only address controlling the location of data driven out of the output pin DPO, and the read/write address controlling the destination of a valid write transaction and the location of data driven out of the output pin SPO.

When the write enable (WE) is low, transitions on the write (WCLK) input are ignored and data stored in the RAMD is not affected. When WE is high, any positive going transition on WCLK loads the data on the data input (D) into the bit selected by the 4-bit read/write address (A3-A0).

The INIT parameter is supported, although not required, for the RAMD symbol to define its initial pattern. The value must be a 4 digit hexadecimal number. INIT data is written from MSB (A<3:0>=1111) to LSB (A<3:0>=0000). For example, a RAMD might have INIT=8F30, where the MSB is 1 and the LSB is 0. In the absence of the INIT parameter, the RAMD is initialized with zeros at every address.

## Internal Three-State Driver

Symbol Type: TBUF

XC2000 Family

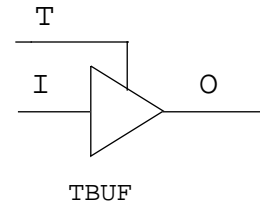
Not available

XC3000, XC4000, and XC5200 Families

Input pins: I, T

Output pin: O

Invertible pins: None



The TBUF symbol corresponds to the internal three-state drivers on the XC3000, XC4000 and XC5200 families of LCAs. The TBUF symbol is distinct from the OBUFT element defined earlier, which corresponds to the output buffer of the IOB. The long line connected to each TBUF is a “weak keeper” in the XC3000 and XC4000 families. The weak keeper will retain the last value driven onto the long line. The weak keeper operates only when no other circuitry (TBUF, PULLUP, or the output of a CLB or IOB) is driving the long line.

In the XC3000, XC4000, and XC5200 families of parts there are three-state drivers inside the LCA. In the XC3000 and XC4000 families they can be used to create internal bus structures or to create wide-input wired-AND functions. In the XC5200 family they can be used to create internal bus structures, but NOT to create wide-input wired-AND functions. The reason for this restriction is that, unlike the XC3000 and XC4000 families, the XC5200 family does not have PULLUPs at the ends of the long lines driven by the TBUFs. The number of internal three-state buffers and the maximum number which share a common output depend on the size of the LCA.

The maximum number of three-state drivers which can drive a single signal is limited, because only those drivers which connect to a single horizontal long line on the LCA may be tied together. Also the number of three-state signals is limited by the number of long lines with three-state capability on the chip. See the data sheets for the part being used to determine these limits.

When implementing a wired-AND function with three-state buffers in the XC3000 family, the T and I pins should be connected to the same signal or the I input should be grounded, and a pull-up resistor must be specified on the output signal. Pull-up resistors are specified with the PULLUP symbol.

When implementing a wired-AND function in the XC4000 family, the WAND symbol should be used because it is more efficient.

Note that the T pin represents an active high three-state enable. When the T pin is active (i.e., it is high or 1) the buffer’s output is high impedance. The T pin could also be considered an active low output enable.

## Special Internal Three-State Drivers

Symbol Types: WAND WORAND

XC2000, XC3000, and XC5200 Families

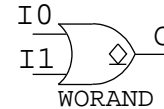
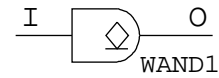
Not available

XC4000 Family

Input pins: I

Output pin: O

Invertible pins: I (only when DECODE property exists)



These two symbols are available in the XC4000 family only. They are very similar to the TBUF symbols except that they have open-drain outputs. The XC4000 architecture has more flexibility for routing wide-input functions and the use of these symbols will allow the automatic tools to take advantage of this flexibility.

The diamond symbol inside the WAND and WORAND indicate an open-drain output. A PULLUP symbol must be attached to the output net for a logical high state to exist.

The WAND symbol, along with the DECODE property, specifies that the special decode logic around the IOBs in the XC4000 family should be used to implement the wired-AND function. This decode logic is comprised of groups of three inputs whose outputs can be connected together to form a wired-AND function. There are special resources available in the XC4000 family to accommodate this function.

Note that the name of the WAND library symbol may be WAND or WAND1; however, the symbol type in the XNF file is always simply WAND. Similarly, the WORAND library symbol may be named WOR2AND, but the symbol type is always WORAND in the XNF file.

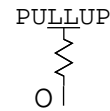
\* Either pin names I0 and I1 or I1 and I2 are legal in version 6 XNF. The new pin names I0 and I1 are preferred.

## Pull-Up Resistor

Symbol Type: PULLUP

XC2000 Family

Not available



XC3000, XC4000, and XC5200 Families

Input pins: None

Output pin: O

Invertible pins: None

The PULLUP symbol corresponds to the programmable pull-up resistors in the XC3000, XC4000, and XC5200 family LCAs. For the XC5200 family pull-up resistors are available on only the external I/O signals. For the XC3000 and XC4000 families, these resistors are available on the horizontal long lines that have three-state drivers on them, and on the external I/O signals. They are also available on the decode lines in the XC4000 family LCA only. The PULLUP symbol for the XC5200 family may be connected to only I/O primitives. For the XC3000 and XC4000 families, this symbol may connect to signals corresponding to only these elements of the LCA: TBUF, WAND, WORAND and I/O primitives.

On each horizontal longline there are two resistors available. If only one is configured, the rising transition will be slower, but will draw less power. If both are configured, the transition will be faster, but will draw more power. The parameter DOUBLE may be specified on the PULLUP symbol to specify that both resistors should be connected. The DOUBLE parameter is an error on PULLUP symbols connected to external signals. The DOUBLE parameter is not allowed on PULLUPS in XC3000L family LCAs.

In the XC3000 family I/O, the PULLUP may be attached to only an input. In the I/O of the XC4000 and XC5200 families it may be attached to an input or output.

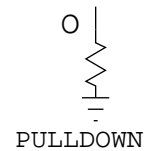
Connecting a pull-up symbol to a signal which is not an output from an internal three-state symbol (TBUF, WAND, WORAND), or an external signal is an error.

## Pull-Down Resistor

Symbol Type: PULLDOWN

XC2000 and XC3000 Families

Not available



XC4000 and XC5200 Families

Input pins: None

Output pin: O

Invertible pins: None

This resistor is available on only the external I/O signals for the XC4000 and XC5200 families. Connecting a pull-down symbol to a signal which is not an external signal is an error.

## Clock Drivers

Symbol Types: ACLK GCLK

XC2000 and XC3000 Families

Input pin: I  
Output pin: O  
Invertible pins: None



XC4000 Family

Not available.  
See Global Buffers in the next section.

XC5200 Family

Not available

The GCLK and ACLK symbols correspond to the global and alternate clock buffers respectively.

There is only one GCLK and one ACLK buffer per LCA. Using more than one ACLK or GCLK in a design is a fatal error.

In the XC3000 family there are special inputs to the ACLK and GCLK buffers which connect directly to specific I/O pins. The purpose of these inputs is to allow external clock signals to be brought inside the LCA with as little delay as possible.

The user specifies the use of this special input in the schematic by connecting the input pin of the clock buffer directly to the external signal which corresponds to the I/O pin. For example, the IPAD symbol is connected to the input of GCLK.

These two clocks are not available in the XC4000 family. The global buffers described in the next section are used instead. Also refer to the BUFG symbol which is available in the Unified library.



## Global Buffers

Symbol Types: BUFGP BUFGS

XC2000, XC3000, and XC5200 Families

Not available

XC4000 Family

Input pin: I  
Output pin: O  
Invertible pins: None



There are four of each of these two kinds of buffers in XC4000 family. They are used for high fan-out, clock and other control signals.

The input pins of these symbols may be connected directly to an external input signal (IPAD) to indicate a dedicated fast external input pin.

Also refer to the BUFG symbol which is available in the Unified library.

## Unified Library Global Buffer

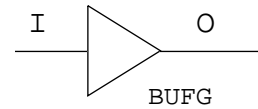
Symbol Type: BUFG

All Families

External input pin: I

Output pin: O

Invertible pins: None



This is a generic global buffer that is available in the Xilinx Unified Library. It can be used for any Xilinx architecture and the Xilinx software will translate it into the appropriate clock driver or global buffer for the targeted architecture. For example, for XC2000 and XC3000 family designs, the BUFG will turn into a GCLK or ACLK. For XC4000 family designs, the BUFG will become a BUFGS or BUFGP. For XC5200 family designs, the BUFG is the clock buffer primitive.

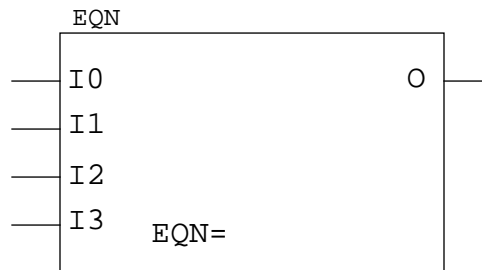
There are four of these buffers in the XC5200 family LCA. They are used for high fan-out clock and other control signals. The input pins of these symbols may be connected directly to an external input signal (IPAD) to indicate a dedicated fast external input pin.

## Equation Symbol

Symbol Type: EQN

XC3000 Family

Input pins: I0, I1, I2, I3, I4  
 Output pin: O  
 Invertible pins: None



XC2000, XC4000, and XC5200 Families

Input pins: I0, I1, I2, I3  
 Output pin: O  
 Invertible pins: None

This is a special purpose symbol that can be used in place of combinational logic symbols. Xilinx does not provide an EQN symbol in its schematic libraries; they are supported primarily for the use of synthesis tools. The function of the symbol is specified using the EQN parameter. In addition to parentheses, the following Boolean operators are allowed in the EQN parameter: “~” (not), “+” (or), “\*” (and), and “@” (xor). The EQN parameter is used with this symbol to specify its function in terms of its input pin names.

For example:

```
SYM, name, EQN, EQN=((I0+I1)*I2)
PIN, I0, I, sig1
PIN, I1, I, sig2
PIN, I2, I, sig3
PIN, O, O, outsig
END
```

When there are no parentheses in the equation, the precedence is in the following order:

1. ~ (not)
2. \* (and)
3. + (or), @ (xor)

Terms with the same precedence are evaluated left to right.

Up to 100 input pins will be allowed on the EQN symbol although the Xilinx core partitioning tools will accept EQN symbols with up to only 4 or 5 input pins (4 inputs for XC2000, XC4000, and XC5200 LCA designs, 5 input pins for XC3000 LCA designs). The allowance of wide fan-in EQN symbols is primarily for the support of third party and synthesis tools. The names of the input pins must begin with “I” followed by a number from “0” to “99”. No leading “0” after the “I” character will be accepted (i.e., “I09” is invalid).

The EQN symbols equation string (EQN=) should be no larger than the currently defined length for signal and symbol names, 1024 characters. Only input pin names and the “0” and

“1” characters (representing VCC and GND) will be allowed in the equation string. Note that this is a change from the Version 4.00 XNF Specification which allowed signal names in the equation string.

**F5\_MUX**

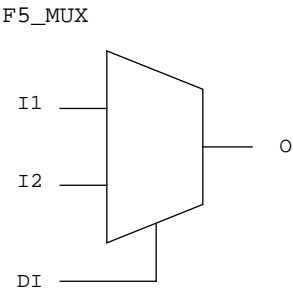
Symbol Type: F5\_MUX

XC2000, XC3000, and XC4000 Families

Not available

XC5200 Family

Input pins: I1, I2, DI  
Output pin: O  
Invertible pins: None



The F5\_MUX symbol represents a dedicated 2-1 multiplexor in the XC5200 family. It is used to combine two adjacent function generators to implement any 5-input function, and limited functions of 6-9 inputs.

When the Direct Input signal (DI) is zero, the I1 input will be passed to the output (O). When the Direct Input signal is a one, the I2 input will be passed to the output.

DI	O
0	I1
1	I2

## Carry Logic Symbols

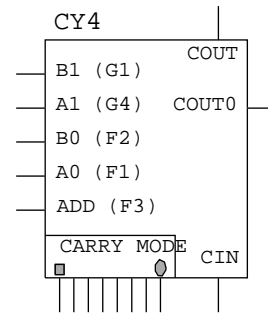
Symbol Type: Carry Logic (CY4)

XC2000, XC3000, and XC5200 Families

Not available

XC4000 Family

Input pins: A0, B0, A1, B1, ADD, CIN,  
C0, C1, C2, C3, C4, C5, C6, C7  
Output pins: COUT0, COUT  
Invertible pins: None



The Carry Logic symbol represents the carry logic available within a XC4000 CLB. The A0 and B0 pins are the operands for the arithmetic operation in the F function generator. The A1 and B1 pins are the operands for the arithmetic operation in the G function generator. The ADD pin is the UP/DOWN or ADD/SUBTRACT control signal. The CIN pin is the dedicated carry-in signal from the previous CLB in the chain. The COUT0 pin is the carry-out signal from the F function generator. The COUT pin is the dedicated carry-out signal from the CLB.

The C0–C7 pins may be connected to only the output signals of the Carry Mode symbols. During the design preparation process (DRC, logic trimming), the Carry Mode symbol is trimmed from the design and the Carry Mode parameter is transferred to the Carry Logic symbol itself. At this time the C0–C7 pins are removed.

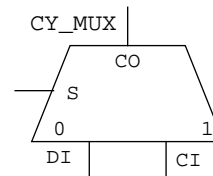
Symbol Type: Carry Logic (CY\_MUX)

XC2000, XC3000, and XC4000 Families

Not available

XC5200 Family

Input pins: S, DI, CI  
Output pin: CO  
Invertible pins: None



The CY\_MUX symbol represents the carry/cascade logic available within the XC5200 CLB. The CY\_MUX is functionally identical to a 2-1 multiplexor, but is a special architectural feature dedicated to carry-propagate in the XC5200 CLB.

The CI pin may be sourced by only the CO pin of another CY\_MUX.

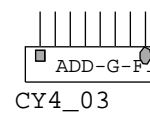
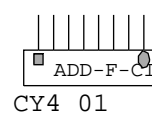
S	CO
0	DI
1	CI

Symbol Type: Carry Modes

XC2000, XC3000, and XC5200 Families

Not available

CY4\_01, CY4\_02,  
CY4\_03,..., CY4\_42

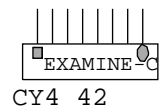


XC4000 Family

Input pins: None

Output pins: C0, C1, C2, C3, C4, C5, C6, C7

Invertible pins: None



There are 42 Carry Mode symbols that represent the 42 viable carry mode configurations identified for the XC4000 CLB. The Carry Mode symbol's output signals can be connected to only the C0-C7 input pins on a Carry Logic symbol. The underlying simulation model of VCC and GND signals for each Carry Mode symbol will provide the proper values to the Carry Mode symbol's simulation model to perform the specified carry function.

Each Carry Mode symbol will have a unique CYMODE=<carry\_mode> parameter that describes the mnemonic of the specific carry mode. During the design preparation process (DRC, logic trimming), the Carry Mode symbol is trimmed from the design and the CYMODE=<carry\_mode> parameter is transferred to the Carry Logic symbol itself.

Example:

```

SYM, $1_abc, CY4_03, CYMODE=ADD-G-F1
PIN, C0, O, sig0
PIN, C1, O, sig1
...
END

```

There is a one-to-one correspondence between the Carry Mode symbol type and the CYMODE=<carry\_mode> parameter that is attached to it. Table 2 shows the CYMODE value that should be attached to each Carry Mode symbol.

Table 2. CYMODE Values for XC4000 Carry Mode Symbols

<b>CYMODE Value</b>	<b>Symbol</b>
ADD_F_CI	CY4_01
ADD_FG_CI	CY4_02
ADD-G-F1	CY4_03
ADD-G-CI	CY4_04
ADD-G-F3-	CY4_05
ADDSUB-F-CI	CY4_12
ADDSUB-FG-CI	CY4_13
ADDSUB-G-F1	CY4_14
ADDSUB-G-CI	CY4_15
ADDSUB-G-F3-	CY4_16
FORCE-0	CY4_37
FORCE-1	CY4_38
FORCE-F1	CY4_39
FORCE-CI	CY4_40
FORCE-F3-	CY4_41
EXAMINE-CI	CY4_42
DEC-F-CI	CY4_24
DEC-FG-CI	CY4_25
DEC-FG-0	CY4_26
DEC-G-0	CY4_27
DEC-G-F1	CY4_28
DEC-G-CI	CY4_29
DEC-G-F3-	CY4_30
INC-F-CI	CY4_17
INC-FG-CI	CY4_18
INC-FG-1	CY4_19
INC-G-1	CY4_20
INC-G-F1	CY4_21
INC-G-CI	CY4_22
INC-G-F3-	CY4_23
SUB-F-CI	CY4_06
SUB-FG-CI	CY4_07
SUB-G-1	CY4_08



Table 2. CYMODE Values for XC4000 Carry Mode Symbols

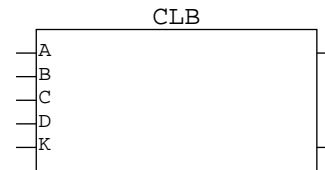
CYMODE Value	Symbol
SUB-G-F1	CY4_10
SUB-G-CI	CY4_09
SUB-G-F3-	CY4_11
INCDEC-F-CI	CY4_31
INCDEC-FG-CI	CY4_32
INCDEC-FG-1	CY4_33
INCDEC-G-0	CY4_34
INCDEC-G-F1	CY4_35
INCDEC-G-CI	CY4_36

## CLB Symbol

Symbol Type: CLB

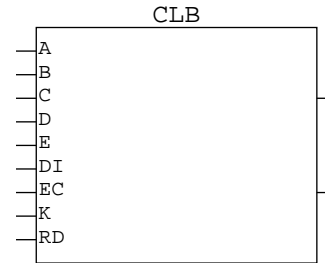
### XC2000 Family

Input pins: A, B, C, D, K  
Output pins: X, Y  
Invertible pins: None



### XC3000 Family

Input pins: A, B, C, D, E, K, EC, DI, RD  
Output pins: X, Y  
Invertible pins: None



### XC4000 and XC5200 Families

Not available

The CLB symbol is used to specify a CLB configuration manually. The configuration is specified with the Design Editor commands BASE, CONFIG and EQUATE. See Appendix C for a description of the syntax for these commands.

These commands are entered on the schematic by the designer and the translator puts them into the CFG records in the LCA Xilinx netlist file. It is not necessary for the translator program to parse the commands specifying the CLB configuration. The mapping program from the LCA Xilinx netlist to the LCA design will check these commands for errors.

The configuration commands must be consistent with the connections to the pins on the symbol. For example, if the A input is used in an equation, then a signal should be connected to the A pin.

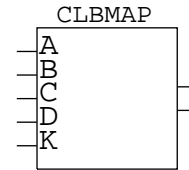
Note that this symbol is used in place of the other combinational logic gates, latches and flip-flops described in this section to specify the functionality of a CLB. This is unlike the CLBMAP symbol which is used for mapping only control and is used in addition to combinational logic gates, latches and flip-flops.

## MAP Symbols

Symbol Type: CLBMAP

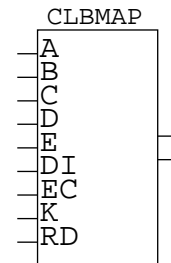
XC2000 Family

Input pins: A, B, C, D, K, X, Y  
 Output pins: None  
 Invertible pins: None



XC3000 Family

Input pins: A, B, C, D, E, K, EC, DI, RD, X, Y  
 Output pins: None  
 Invertible pins: None



XC4000 and XC5200 Families

Not available

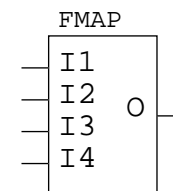
Symbol Type: FMAP

XC2000 and XC3000 Families

Not available

XC4000 and XC5200 Families

Input pins: I1, I2, I3, I4, O  
 Output pins: None  
 Invertible pins: None



Symbol Type: F5MAP

XC2000, XC3000, and XC4000 Families

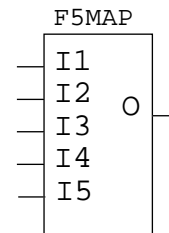
Not available

XC5200 Family

Input pins: I1, I2, I3, I4, I5, and O

Output pins: None

Invertible pins: None



Symbol Type: HMAP

XC2000, XC3000, and XC5200 Families

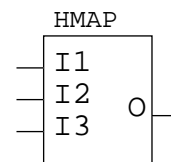
Not available

XC4000 Family

Input pins: I1, I2, I3, O

Output pins: None

Invertible pins: None



The CLBMAP symbol is used to control the partitioning of logic into XC2000 and XC3000 family CLBs. This symbol is used at the schematic level along with the other logic primitives described in this section. The user may implement a portion of logic using gates and flip-flops and specify that the logic be grouped into a single CLB by using this CLBMAP symbol. The user names the signals of the logic that are to be inputs and outputs of a CLB and then use the CLBMAP symbol, drawing signals to the pins of the symbol or naming the signals the same as the signals of the logic to be put into the CLB.

The FMAP symbol is used to control partitioning of logic into XC4000 and XC5200 family function generators. The F5MAP symbol is used to control partitioning of logic into XC5200 family function generators, in which two function generators can be combined via an F5\_MUX to form any function of five inputs. The HMAP symbol is used to control the partitioning of logic into XC4000 family H function generators. The user may implement a portion of logic using gates and flip-flops and specify the gate logic to be grouped into the F and G function generators using the FMAP and the H function generator using the HMAP. The user names the signals of the logic that are to be inputs and outputs of a function generator and then use the FMAP, F5MAP, and HMAP symbols, naming the signals connected to the pins of the FMAP/F5MAP/HMAP the same as the signals of the logic to be mapped into the function generators.

The MAP=type parameter can be used with the CLBMAP, FMAP, and F5MAP symbols to specify whether the pins are locked to signals and whether the CLB or function generator is “closed”. MAP=PLC is used to specify that the pins are locked to their signals and no more logic other than what is specified can be put into the CLB or function generator. MAP=PUC indicates that the pins are not locked to their signals but that the CLB or function generator is closed for more logic. MAP=PUC is the only type that is supported by the F5MAP symbol.

Two additional MAP types (MAP=PLO and MAP=PUO) can be used to specify that the CLB is “open”, meaning the mapping program will determine the logic to place in the CLB or function generator given the specified output signals. In this case the user need specify only the output signals. Individual pin locking can be specified by using the “P” (Pin lock) pin parameter on the CLBMAP and FMAP symbol’s pins. The only MAP parameter currently supported for the HMAP symbol is PUC. The only MAP parameter currently supported for the F5MAP symbol is PUC.

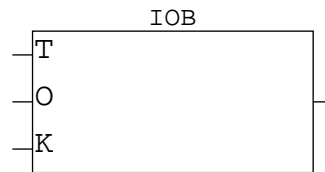
Note that this symbol is not a substitute for logic in the way that the CLB symbol is used. It is used for mapping only control and is used in addition to combinational logic gates, latches and flip-flops.

## IOB Symbol

Symbol Type: IOB

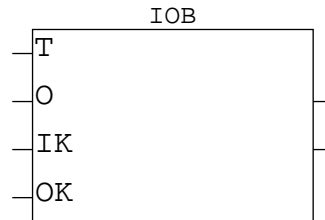
### XC2000 Family

Input pins: O, T, K  
Output pin: I  
Invertible pins: None



### XC3000 Family

Input pins: O, T, IK, OK  
Output pins: I, Q  
Invertible pins: IK, OK



### XC4000 and XC5200 Families

Not available

The IOB symbol is used to specify an IOB configuration manually. The configuration is specified with the Design Editor commands BASE and CONFIG. See Appendix C for a description of the syntax for these commands.

These commands are entered on the schematic by the designer and the translator puts them into the CFG records in the LCA Xilinx netlist file. It is not necessary for the translator program to parse the commands specifying the IOB configuration. The mapping program from the LCA Xilinx netlist to the LCA design will check these commands for errors.

Notice that the I pin is actually an output pin on the symbol and that the O pin is actually an input pin to the symbol. This is because the I pin refers to an input into the LCA, not into the IOB symbol. Similarly, the O pin refers to an output from the LCA, not the IOB symbol.

The configuration commands must be consistent with the connections to the pins on the symbol. For example, if the configuration commands specify this as a three-state buffer then the T and O pins should be connected to signals.

Note that this symbol is used in place of the other I/O primitives described in this section to specify the functionality of an IOB.

## TIMESPEC Symbol

Symbol Type: TIMESPEC

XC2000 and XC3000 Families

Not available

XC3000A/L, XC4000, and XC5200 Families

Input pins: None

Output pins: None

TIMESPEC

The TIMESPEC symbol along with TS parameters allows the user to specify timing requirements from schematic to the Xilinx place and route tools.

The TIMESPEC symbol serves as a place to specify TS parameters. The syntax for the TS parameters is:

*TSidentifier=specification*

The *identifier* is a name consisting of any combination of letters, numbers or underscores. It should be kept short for convenience and clarity. The *specification* is a combination of letters, digits, underscores (“\_”), equals (“=”), colons (“:”), asterisks (“\*”), forward slashes (“/”), parentheses (“(” and “)”), and question marks (“?”). The *specification* may also reference signal names, in which case any legal XNF character is valid (i.e., “<” and “>”). See the XACT Performance chapter of the Development System Reference Guide for information on usage of TS parameters.

An example of a TIMESPEC primitive with TS parameters:

TIMESPEC
TS01 = C2P:40
TS_wide=from:start:to:end3=200ns

The example of the TIMESPEC symbol with TS parameters would be translated as follows:

```
SYM, name1, TIMESPEC, TS01=C2P:40, TS_wide=from:start:to:end3=200ns
END
```

The TS\_01 parameter would have a corresponding signal parameter to specify the set of flip-flops to which this specification applies. This parameter would then be translated into the XNF format as follows:

SIG, sig1, TS01

The TS\_wide parameter is only an identifier; because this specification is defined in terms of groups (defined by TNM parameters or in TIMEGRP symbols), TS\_wide need not appear as a SIG parameter in the netlist.



## TIMEGRP Symbol

Symbol Type: TIMEGRP

XC2000 and XC3000 Families

Not available

XC3000A/L, XC4000, and XC5200 Families

Input pins: None

Output pins: None

TIMEGRP

The TIMEGRP symbol is provided to define groups in terms of the basic “tnm” groups, for use in TIMESPEC symbols. The TIMEGRP symbol consists of only user parameters, all of the form:

*=new\_group=construction*

Here *new\_group* is a name consisting of any combination of letters, digits and underscores, and *construction* may contain letters, digits, underscores (“\_”), colons (“:”), equals (“=”), asterisks (“\*”), parenthesis (“(” and “)”) and question marks (“?”). The *construction* may also reference signal names in which case any legal XNF character would be valid (i.e., “<” and “>”).

Example:

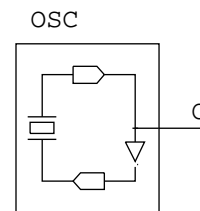
```
SYM, instname2, TIMEGRP, =xy=ffs(x*:y*), =new:group1:except:group2
END
```

## Oscillator

Symbol Type: OSC

XC2000 and XC3000 Families

Input pins: None  
Output pin: O  
Invertible pins: None



XC4000 and XC5200 Families

Not available

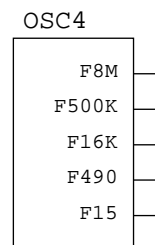
Symbol Type: OSC4

XC2000, XC3000, and XC5200 Families

Not available

XC4000 Family

Input pins: None  
Output pins: F8M, F500K, F16K, F490, F15  
Invertible pins: None



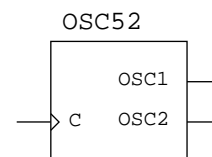
Symbol Type: OSC52

XC2000, XC3000, and XC4000 Families

Not available

XC5200 Family

Input pins: C  
Output pins: OSC1, OSC2  
Invertible pins: None



The OSC symbol corresponds to the oscillator driver on the XC2000 and XC3000 LCA, which is described in Section 2 of the Programmable Logic Data Book.

The oscillator driver is a special dedicated circuit on the LCA which can be used to drive an off-chip crystal to generate a high speed clock signal in the LCA.

When the oscillator circuit is used, two of the IOBs are dedicated to it. The pin numbers of the IOBs depends on the part and package type being used and is listed on the data sheets. When the oscillator is used, the program which maps the LCA Xilinx netlist to an LCA design will reserve the two IOBs required for the oscillator circuit. If the design uses those IOBs for another purpose, it is a fatal error. Note that the oscillator circuit is not explicitly shown in the design. The two IOBs and the inverting buffer are implicit in the OSC symbol.

In the XC2000 family the oscillator can be used to drive ONLY the input of the ACLK. No other connection is allowed. The XC3000 family allows connection to other symbols, but the ACLK is still the preferred way of distributing the signal.

The OSC4 symbol represents the XC4000 family internal oscillator. There are five output pins, providing the oscillator frequency (nominally 8 MHz) and four outputs from a frequency divider, nominally 500 KHz, 16 KHz, 490 Hz, and 15 Hz. At most two of the divided frequencies may be used.

The OSC52 symbol represents the XC5200 family internal oscillator, which runs at a nominal 16 MHz, as well as a divider circuit that can provide two of 12 frequencies divided down from the internal oscillator or from a signal specified by the user on the 'C' pin. The OSC parameter is required on this symbol. If no clock input is specified, then OSC=INTERNAL should be specified, and the internal oscillator will be used. Otherwise, the clock input should be connected to a user-defined clock signal and OSC=USER should be specified. The parameters DIVIDE1\_BY and DIVIDE2\_BY define the amount by which the clock input or internal oscillator is to be divided. The user may choose to use one or both of the outputs ('OSC1' and/or 'OSC2'), but the DIVIDE1\_BY=value parameter is required for 'OSC1' and DIVIDE2\_BY=value is required for 'OSC2'.

## Special Function Access

These symbols allow the user to control or monitor certain LCA functions such as Boundary Scan, Readback, and Configuration.

Symbol Type: BSCAN

XC2000 and XC3000 Families

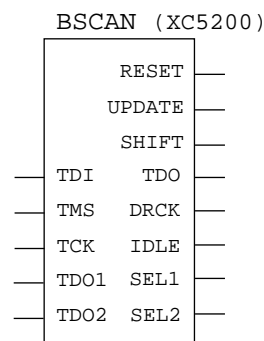
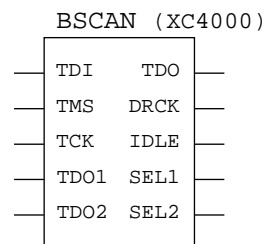
Not available

XC4000 Family

Input pins: TDI, TMS, TCK, TDO1, TDO2  
Output pins: TDO, DRCK, IDLE, SEL1, SEL2  
Invertible pins: None

XC5200 Family

Input pins: TDI, TMS, TCK, TDO1, TDO2  
Output pins: TDO, RESET, UPDATE, SHIFT, DRCK, IDLE, SEL1, SEL2  
Invertible pins: None



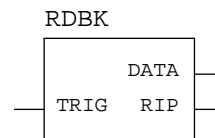
Symbol Type: RDBK

XC2000 and XC3000 Families

Not available

XC4000 and XC5200 Families

Input pins: TRIG  
Output pins: DATA, RIP  
Invertible pins: None



Symbol Type: RDCLK

XC2000 and XC3000 Families

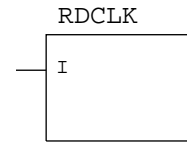
Not available

XC4000 and XC5200 Families

Input pin: I

Output pins: None

Invertible pins: None



Symbol Type: STARTUP

XC2000 and XC3000 Families

Not available

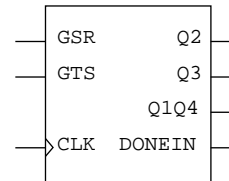
XC4000 Family

Input pins: GSR, GTS, CLK

Output pins: Q2, Q3, Q1Q4, DONEIN

Invertible pins: GSR, GTS

STARTUP (XC4000)



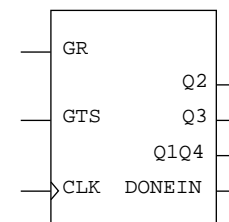
XC5200 Family

Input pins: GR, GTS, CLK

Output pins: Q2, Q3, Q1Q4, DONEIN

Invertible pins: GR, GTS

STARTUP (XC5200)



Symbol Types: Special Input Pads TDI, TMS, TCK, MD0, MD2

#### XC2000 and XC3000 Families

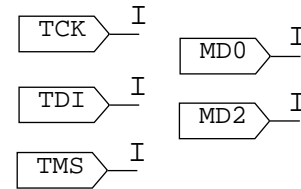
Not available

#### XC4000 and XC5200 Families

Input pins: None

Output pin: I

Invertible pins: None



In the XC4000 family, the MD0 and MD2 special pads may connect only to the I pin of an IBUF symbol. In the XC5200 family, the MD0 and MD2 may be used as unrestricted (input, output or bidirectional) pads after configuration. The TDI, TMS and TCK special pads may connect to the corresponding pins of the BSCAN symbol and they may also connect to the external pins of I/O symbols, in the same manner as PAD symbols. The TDI, TMS and TCK special pads may be used as unrestricted (input, output or bidirectional) pads in both the XC4000 and XC5200 families.

Symbol Types: Special Output Pads TDO, MD1

#### XC2000 and XC3000 Families

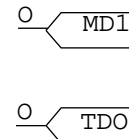
Not available

#### XC4000 and XC5200 Families

Input pin: O

Output pins: None

Invertible pins: None



The TDO special pad is the only pad that may be connected to the TDO pin of a BSCAN symbol. In the XC4000 family, the TDO pad may otherwise be connected only to an OBUF or an OBUFT symbol, while in the XC5200 family, the TDO pad may be used as an unrestricted (input, output or bidirectional) pad. The MD1 special pad may be connected only to the O pin of an OBUF or OBUFT symbol in the XC4000 family, but may be used as an unrestricted (input, output or bidirectional) pad in the XC5200 family.

## Macro Symbol

Macro symbols define an instantiation of logic which is described in another XNF file. This allows the user to instantiate logic in the schematic which was not defined with the schematic editor. Some examples include a design described in state machine syntax, in Boolean equations in PALASM2 format, or with a different schematic editor.

Macro symbols do not have any predefined symbol type. A symbol is assumed to be a macro if it has a symbol type different from any of those described here. As long as the translator program for a schematic editor does not try to do any error checking on the symbol types, macro symbols should not cause any problems.

The name of the XNF file which defines the logic for a macro symbol is defined by the FILE parameter on the symbol or, if no FILE parameter is specified, the symbol type, with an extension of .XNF, is used. All the pins on the macro symbol must correspond to a signal in the macro definition. This correspondence is made either by the pin and the signal having the same name or by a PIN parameter with the pin's name on a signal.

Designs with macro symbols in them should be flattened with the Xilinx merge program before running the technology mapper.

## Summary of Symbol Types

The following is a summary of the symbol types described above (note that not all symbols are available in all of the XC2000, XC3000 or XC4000 family of LCAs):

### Combinational Logic:

BUF	INV
AND	NAND
OR	NOR
XOR	XNOR

### I/O Symbols:

IBUF	OBUF	OBUFT
INFF	OUTFF	OUTFFT
INLAT	INREG	IPAD
OPAD	IOPAD	UPAD
PULLUP	PULLDOWN	

### CLB Flip-Flops and Latches:

DFF	DLAT
-----	------

### CLB RAM and ROM

RAM	ROM
RAMS	RAMD

### Internal 3-State Symbols:

TBUF	PULLUP
WAND	WORAND

### Clock Buffers and Oscillators:

ACLK	GCLK	
OSC	OSC4	OSC52
BUFG	BUFGP	BUFGS

### Manually Defined CLB and IOB:

CLB	IOB
-----	-----



## Mapping Control Symbols:

CLBMAP	FMAP
F5MAP	HMAP

## Equation Symbol:

EQN

## Timing Specification Symbols:

TIMESPEC	TIMEGRP
----------	---------

## Carry Logic Symbols:

CY4	CY_MUX	CY4_01, CY4_02, ..., CY4_42
-----	--------	-----------------------------

## Special Function Access Symbols:

F5_MUX	BSCAN	STARTUP
RDBK	RDCLK	TCK
TDO	TDI	TMS
MD0	MD1	MD2

## Macro Symbols:

Any type not previously defined.



## Sample LCA Xilinx Netlist File

A sample LCA Xilinx netlist file appears on the next page, followed by Figure 2, the sample schematic design that it describes.

The developer of the translator for a particular schematic editor must decide the best way to uniquely name symbols for that particular editor. In this example, the symbol “names” are actually numbers assigned by the schematic editor to the symbols. In general, symbol names do not have to be assigned by the user, but symbol names must all be unique and there must be some way for the user to identify a symbol in the schematic from the name. This is important because messages will use the symbol names to identify particular symbols.

This example uses explicit symbols for I/O connections. These symbols are at the far left and the far right of the schematic and include user defined LCA pin numbers such as P61 and P11. The external symbols don't map into a symbol record but rather map into an EXT record. The EXT records are not required for specifying I/O connections, however, they are recommended. Any signal connected to the external pin of an I/O primitive such as IBUF or OBUF is automatically assumed to be an external connection. In the absence of the EXT records, the I/O pin locations may be specified on the IBUF or OBUF symbols with the LOC parameter.

Different schematic editors have different conventions for specifying I/O connections. The translator developed for a particular schematic editor should support whatever conventions are generally used by that editor.

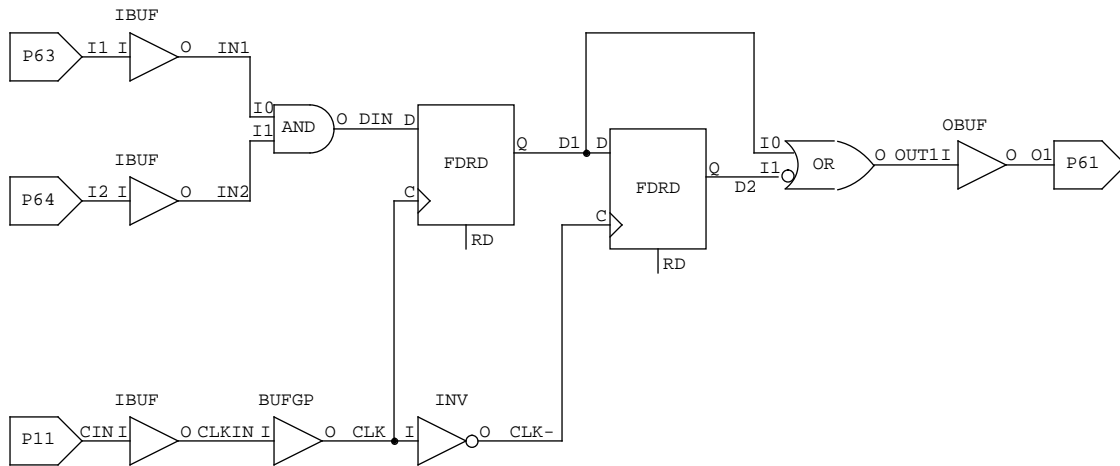
Sample LCA Xilinx Netlist (see design in Figure 2):

```
LCANET,6
PROG,WIR2XNF,4.00,"4-27-93,10:27:13; -p 4005pg156"
PART,4005PG156
SYM,S1,IBUF
PIN,I,I,I1
PIN,O,O,IN1
END
SYM,S2,IBUF
PIN,I,I,I2
PIN,O,O,IN2
END
SYM,S3,AND
PIN,I0,I,IN1
PIN,I1,I,IN2
PIN,O,O,DIN
END
SYM,S4,DFF,INIT=R
PIN,D,I,DIN
PIN,C,I,CLK
PIN,Q,O,D1
END
SYM,S5,DFF,INIT=R
PIN,D,I,D1
PIN,C,I,CLK-
PIN,Q,O,D2
END
SYM,S6,OR
PIN,I0,I,D1
PIN,I1,I,D2,,INV
PIN,O,O,OUT1
END
SYM,S7,OBUF
PIN,I,I,OUT1
PIN,O,O,O1
END
SYM,S8,IBUF
PIN,I,I,CIN
PIN,O,O,CLKIN
END
SYM,S9,BUFGP
PIN,I,I,CLKIN
PIN,O,O,CLK
END
SYM,S10,INV
PIN,I,I,CLK
```

```

PIN,O,O,CLK-
END
EXT,I1,I,,LOC=P63
EXT,I2,I,,LOC=P64
EXT,CIN,I,,LOC=P11
EXT,O1,O,,LOC=P61
EOF

```



**Figure 2. Sample LCA Schematic Design**



## Appendix A: LCA Naming Restrictions

LCA names may contain only the following characters:

A-Z a-z 0-9 \$ \_ - < > /

The “/” character is reserved as the hierarchy path name separator. Symbol and signal names can be up to 1024 characters in length. Hierarchical designs should contain full path names for all symbol and signal names and use the “/” character as the path name separator.

Names must contain at least one non-digit character. They may start with any legal character.

Names generated by the Xilinx program that creates an XNF file from an LCA have a ‘.’ in them. These names are used in only the simulation model. By using the ‘.’ character, which is illegal in user names, name collisions are guaranteed to be avoided.

In name comparisons, the case of letters is ignored. (That is, ABC and abc and AbC all refer to the same names.)

Certain block names are reserved and will be ignored if a BLKNM parameter tries to name a block with a reserved name. These names may still be used as signal names. The exact names which are reserved depends on the LCA part type being used.

Reserved names are:

### CLB LOCATIONS

See Appendix B for a description of CLB names.

### IOB LOCATIONS

The letters L (Left), R (Right), T (Top), B (Bottom), TL (Top-Left), LT (Left-Top), TR (Top-Right), RT (Right-Top), LB (Left-Bottom), BL (Bottom-Left), RB (Right-Bottom), and BR (Bottom-Right) are reserved for specifying an area in which to place an IOB.

### IOB PIN NAMES

IOB pin names are either one or two letters followed by a number.

### IOB PAD NAMES

IOB pad names are always of the form PADnn where nn is a number.

### NON-USER I/O PIN NAMES

The following block names are reserved as names for non-user I/O pins:

GND  
VCC  
CCLK  
DP  
M0RT  
M1RD  
PWRDN  
RST  
OSC  
BSCAN  
RDBK  
RDCLK  
STARTUP  
MD0  
MD1  
MD2  
TDO  
TDI  
TMS  
TCK



## Appendix B: LCA Location Names

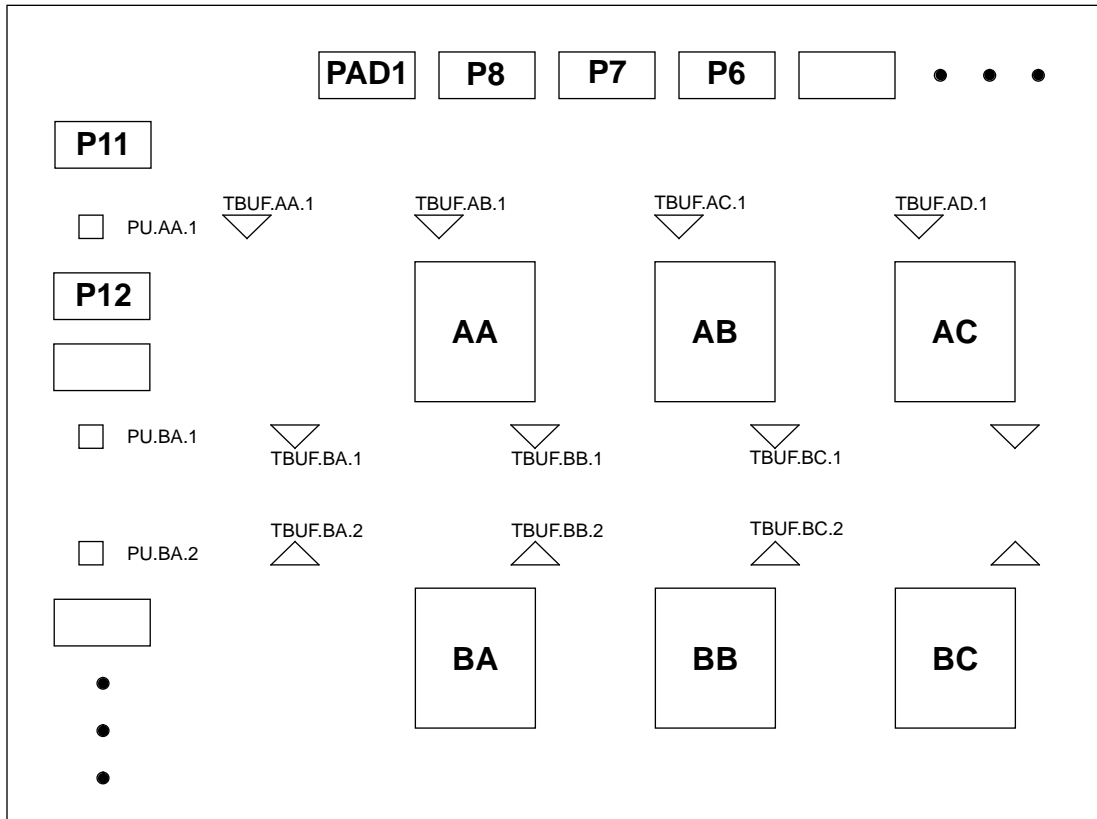
At the time this document was printed, the Xilinx software supports Naming Convention #1 for 2000, 3000 and 3000A family designs, and Naming Convention #2 for 4000 family designs. The syntax for specifying the exact locations for the decode logic (DEC\_R#C#.#) is not supported.

### Naming Convention #1

The format for CLB locations is two letters. The first letter indicates the row, the second letter indicates the column. The CLB with location AA is in the top left corner of the LCA. The wild card character (\*) may be used to indicate a non-specific row or column. Examples of CLB location names are: AA, FG, BC, A\*, \*D.

IOB location names may be a specific pad location which corresponds to an LCA package pin. They may also be of the form "PADnn" where nn is a number beginning at 1 (top left pad) and continues clockwise around the LCA. IOB locations can also refer to an edge, half edge or corner by specifying T (top edge), L (left edge), B (bottom edge), R (right edge), TL (top left half edge), TR (top right half edge), RT (right top half edge), RB (right bottom half edge), BR (bottom right half edge), BL (bottom left half edge), LB (left bottom half edge) and LT (left top half edge). Examples are: P12, PAD15, T, BL.

The format for TBUF and PULLUP locations is TBUF.nn.# and PU.nn.# respectively. The "nn" refers to the CLB location near the TBUF or PULLUP. The # is optional and can be a 1 or 2 to indicate which of the two TBUFs or PULLUPs in the given row and column is desired. Wild cards (\*) can be used for TBUFs to indicate a non-specific row or column. Examples are: TBUF.AA.1, PU.BB.2, TBUF.D\*.



**Figure B-1. LCA Location Naming Convention #1**

## Naming Convention #2

In this naming scheme, CLB locations are of the form `CLB_R#C#[.ffx, ffy, f, g or h]` where the # refers to a number to indicate the row and column. `CLB_R1C1` is the top left CLB in the LCA. The user may optionally specify the internal function of the CLB in which logic is to be placed in such as `ffx`, `ffy`, `f`, `g` or `h`. The wild card character (\*) may be used to indicate a non-specific row or column. Examples are: `CLB_R1C1`, `CLB_R*C12`, `CLB_R10C*`, `CLB_R2C2.ffy`.

IOB location names may be a specific pad location which corresponds to an LCA package pin. They may also be of the form "PADnn" where nn is a number beginning at 1 (top left pad) and continues clockwise around the LCA. IOB locations can also refer to an edge, half edge or corner by specifying T (top edge), L (left edge), B (bottom edge), R (right edge), TL (top left half edge), TR (top right half edge), RT (right top half edge), RB (right bottom half edge), BR (bottom right half edge), BL (bottom left half edge), LB (left bottom half edge) and LT (left top half edge). Examples are: `P12`, `PAD15`, `T`, `BL`.

TBUF location names are of the form `TBUF_R#C#.#` where the # following the R and C refer to the row and column number. The .# is optional and can be a 1 or 2 to indicate which TBUF

in the specified row or column. The wild card character (\*) may be used to indicate a non-specific row or column. Examples are: TBUF\_R\*C1, TBUF\_R3C5.2, TBUF\_R1C\*.1.

The syntax for specifying the exact locations for the decode logic is: DEC\_R#C#.# where the # following the R and C is the row and column number. The .# is optional and can be a 1, 2 or 3 to indicate which of the three decoders to use in the given row and column. Edges and half edges are legal decode location names as well (see description for IOB location names above). Examples are: DEC\_R0C5.3, T, BR, DEC\_R1C0.

Global buffer (BUFGP, BUFGS) location names can be TR (top right corner), TL (top left corner), BR (bottom right corner), BL (bottom left corner).

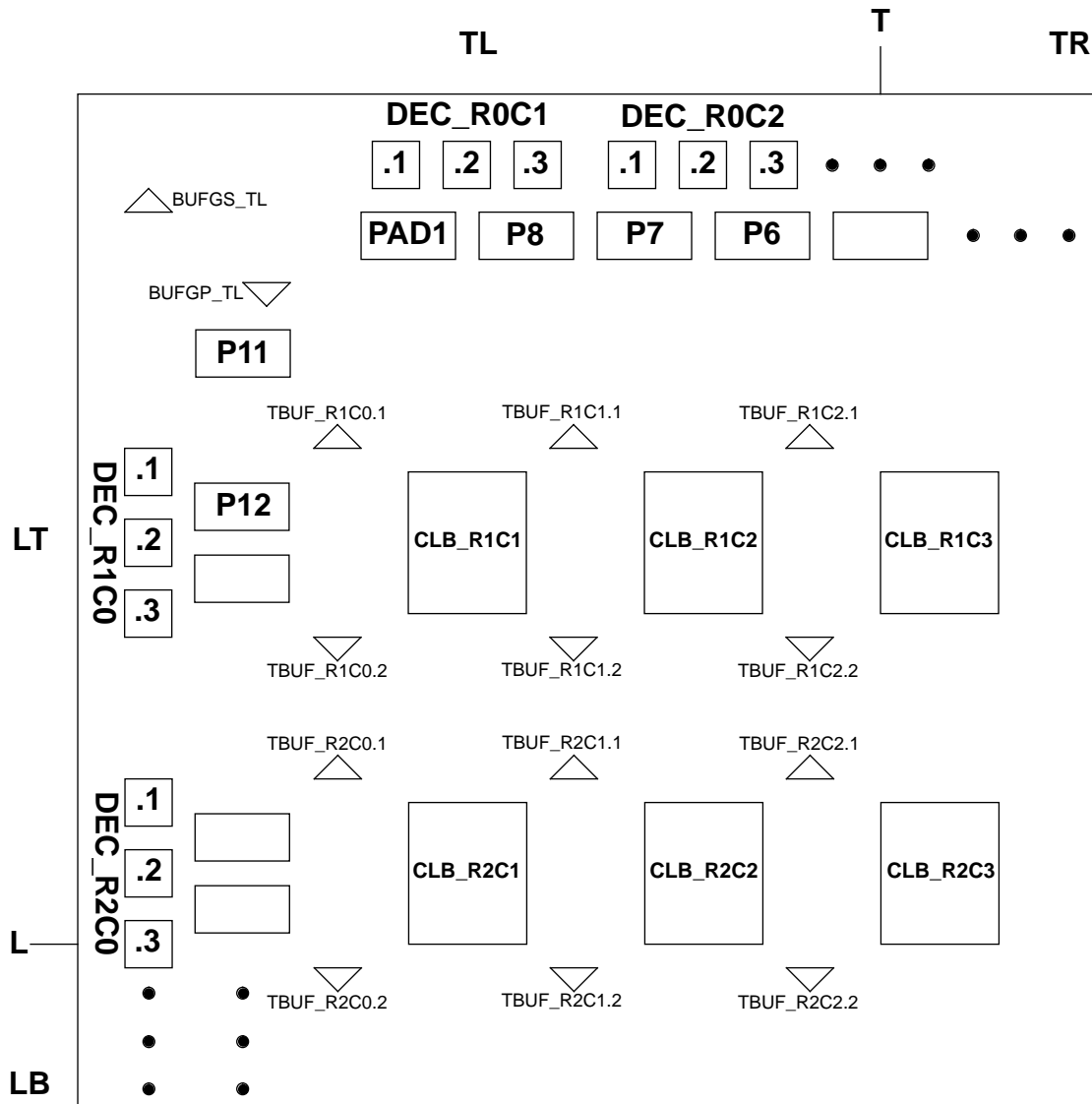


Figure B-2. LCA Location Naming Convention #2

### Naming Convention #3

This naming convention is very similar to Naming Convention #2. In this scheme, CLB locations take the form `CLB_R#C#[.LC0, .LC1, .LC2, .LC3]`, where # is a number indicating a row or column. `CLB_R1C1` is the top left CLB in the LCA. The user may optionally specify the internal location of the CLB in which logic is to be placed by using the `.LC#` suffix. The wild card character (\*) may be used to indicate a non-specific row or column. Examples are: `CLB_R1C1`, `CLB_R*C12`, `CLB_R10C*`, `CLB_R2C2.LC3`.

An IOB location name may be a specific pad location that corresponds to an LCA package pin. It may also be of the form "PADnn" where nn is a number beginning at 1 (top left pad) and continuing clockwise around the LCA. IOB locations can also refer to an edge, half edge or corner by specifying T (top edge), L (left edge), B (bottom edge), R (right edge), TL (top left half edge), TR (top right half edge), RT (right top half edge), RB (right bottom half edge), BR (bottom right half edge), BL (bottom left half edge), LB (left bottom half edge), and LT (left top half edge). Examples are: `P12`, `PAD15`, `T`, `BL`.

TBUF location names are of the form `TBUF_R#C#[.0, .1, .2, .3]` where # refers to a number indicating a row and column. The `.0`, `.1`, `.2`, or `.3` extension is optional and is used to indicate the TBUF in the specified row or column. Examples are: `TBUF_R*C1`, `TBUF_R3C5.0`, `TBUF_R1C*.2`.

Global buffer (BUFGP, BUFGS) location names can be `TR` (top right corner), `TL` (top left corner), `BR` (bottom right corner), and `BL` (bottom left corner).

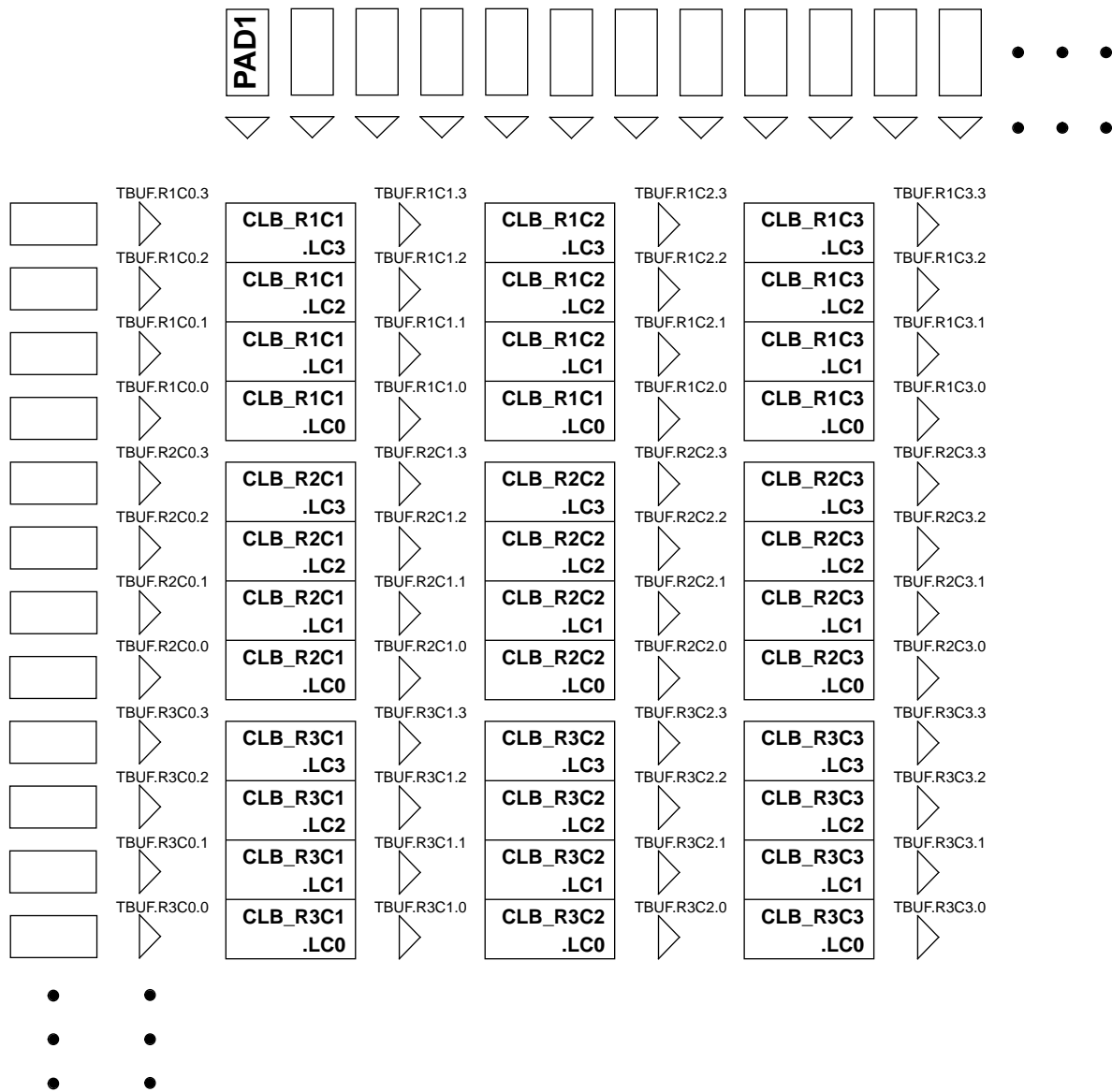


Figure B-3. LCA Location Naming Convention #3

### Single LOC Properties:

The syntax for specifying single LOC properties is simply LOC = name where name is a legal location name as described above. Examples of this are:

Table B-1. Examples of Single LOC Properties

Parameter	Meaning
LOC = AA	Place logic in CLB AA.
LOC = TBUF_R2C1.1	Place TBUF in row 2, column 1, along the top.
LOC = TBUF_R*C0.1	Place TBUF in any row in column 0.
LOC = TL	Place decode logic, I/O or global buffer on the top left edge.
LOC = P12	Place I/O at location P12.
LOC = DEC_R0C1.3	Place decode logic at the third location in the first group from the left (row 0, column 1).
LOC = B	Place decode logic or I/O on the bottom edge.
LOC = CLB_R4C5.ffx	Place CLB flip-flop in the X flip-flop of the CLB in row4, column 5.
LOC = CLB_R2C3.LC2	Place FMAP, flip-flop, latch, or CY_MUX in appropriate location within Logic Cell 2 of the CLB in row 2, column 3.

### Area LOC Properties:

Areas can be specified using the LOC property. This is done by specifying the upper left and lower right corners of an area in which logic is to be placed using the colon (:) to separate the two boundaries. Some examples are:

Table B-2. Examples of Area LOC Properties

Parameter	Meaning
LOC = CLB_R1C1: CLB_R5C5	Place logic in top left corner in a 5x5 area bounded by row 5, column 5.
LOC = TBUF_R1C1: TBUF_R2C8	Place TBUF anywhere in the area bounded by row 1, column 1 and row2, column 8.
LOC = AA: FF	Place CLB logic anywhere in the top left corner bounded by row F, column F.

### Prohibit LOC Properties:

In addition to specifying locations for logic to be placed, locations may also be prohibited by using the LOC <> (not equal) parameter. Single locations may be prohibited or an entire area may also be prohibited just as with the LOC = parameter. Examples of this are:

Table B-3. Examples of Prohibit LOC Properties

Parameter	Meaning
LOC <> A*	Do not place CLB logic anywhere on the top row.
LOC <> TBUF_R*C0	Do not place TBUF anywhere in column 0.
LOC <> CLB_R5C*.ffy	Do not place the CLB flip-flop in the Y flip-flop of any CLB in row 5.
LOC <> T	Do not place I/O or decoder on the top edge.
LOC <> CLB_R1C1:CLB_R5C5	Do not place the logic in any CLB in the top left corner extending to row 5, column 5.

### Multiple LOC Properties:

Multiple LOC = and LOC <> parameters may be used on the same symbol. If multiple LOC = parameters are placed on a single symbol or group of symbols (such as a macro), this is interpreted by the software as OR-ing each of the parameters together. Multiple LOC <> parameters are interpreted as AND-ing the parameters together. The syntax for using multiple LOC parameters is to separate each of them with a semi-colon (;). Some examples are:

Table B-4. Examples of Multiple LOC Properties

Parameter	Meaning
LOC = T; LOC = L	Place I/O or decoder on the top or left edge.
LOC = CLB_R1C1:CLB_R5C5; LOC <> CLB_R1C1; LOC <> CLB_R5C5	Place CLB logic in the top left corner in a 5x5 area but not in the CLB in row 1, column 1 and not in the CLB in row 5, column 5.





## Appendix C: Design Editor Configuration Commands

### XC2000 Family:

The allowable BASE commands for the XC2000 family are:

IO	for IOBs only
F	for CLBs, one function of up to 4 variables
FG	for CLBs, two functions of up to 3 variables each
FGM multiplexed	for CLBs, same as FG but the two function outputs are together and controlled by the B input.

The CONFIG statements for the XC2000 family are of the form <tag>:<option>.

Tag	Options
-----	---------

#### CLB:

X	F, G, M or Q
Y	F, G, M or Q
Q	FF or LATCH
SET	A, F or M
RES (RESET)	D, F, G or M
CLK	K, C, F, G or M and NOT
F	A, B, C, D and Q
G	A, B, C, D and Q
For base F	function G = F
For base FGM	functions F and G are not individually available, only M

Note: The F and G tags are not required but will always appear in post-layout XNF files.

#### IOB:

I	PAD or Q
BUF	ON or TRI

All of the options except for NOT and those for F and G are mutually exclusive. Unused tags may either be omitted or left without an option.

The EQUATE F and EQUATE G statements are expressed in terms of the variables A, B, C, D and Q and the Boolean operators:

~	NOT
*	AND
+	OR
@	XOR

**Example:**

```
SYM, clb2, CLB
CFG, CONFIG X:F Q:FF CLK:K:NOT
CFG, BASE F
CFG, EQUATE F = (B*(~C+D)
.
.
.
```

**XC3000 Family:**

The allowable BASE commands for the XC3000 family are:

IO	for IOBs only
F	for CLBs, one function of up to 5 variables
FG	for CLBs, two functions of up to 4 variables each
FGM	for CLBs, same as FG but the two function outputs are multiplexed together and controlled by the E input.

The CONFIG statements for the XC3000 family are of the form <tag>:<option>.

Tag	Options
<b>CLB:</b>	
X	E, M or QX
Y	G, M or QY
DX	DI, F, M or G
DY	DI, F, M or G
CLK	K and NOT
RSTDIR	RD
ENCLK	EC
F	A, B, C, D, E, QX and QY
G	A, B, C, D, E, QX and QY
For base F	function G = F
For base FGM	functions F and G are not individually available, only M
<b>IOB:</b>	
IN	I, IQ, IKNOT, FF, LATCH, PULLUP*
OUT	O, OQ, NOT, OKNOT, FAST
T	RIT and NOT

\* The IN : PULLUP option is not permitted when the IOB is configured as an output.

All of the options for CLB except for NOT and those for F and G are mutually exclusive. For IOB, FF and LATCH are mutually exclusive for the IN tag. The O and OQ are mutually exclusive for the OUT tag. Unused tags may either be omitted or left without an option.

The EQUATE F and EQUATE G statements are expressed in terms of the variables A, B, C, D, E, QX and QY and the Boolean operators:

~	NOT
*	AND
+	OR
@	XOR

Example:

```

SYM, clb3, CLB
CFG, CONFIG X:F G:QY DX:F CLK:K:NOT
CFG, BASE FG
CFG, EQUATE F = ((A @ B) *~C)
CFG, EQUATE G = (A + B) PIN,...
.
.
.
SYM, iob3, IOB
CFG, BASE IO
CFG, CONFIG IN:I:IQ:FF OUT:OQ:OKNOT:FAST
.
.
.
```

XC4000 Family:

Not available.



## Appendix D: Document Revision History

This appendix describes any functional changes made since previous versions of this document.

This is the thirteenth draft of the Xilinx Netlist Format (XNF) Specification. It is released as Version 6.1, dated June 1, 1995. Previous versions were released on these dates:

Draft 12, Version 6.0	—	November 30, 1994
Draft 11, Version 5.2	—	September 14, 1994
Draft 10, Version 5.1	—	August 5, 1994
Draft 9, Version 5.0	—	September 7, 1993
Draft 8	—	September 2, 1992
Draft 7	—	October 18, 1991
Draft 6, Version 4.0	—	December 6, 1990
Draft 5, Version 2.0	—	January 26, 1989
Draft 4	—	January 6, 1988
Draft 3, Version 1.0	—	May 29, 1987
Draft 2	—	March 27, 1987
Draft 1	—	February 17, 1987

The thirteenth draft, Version 6.1, includes the following changes to the twelfth draft, Version 6.0:

1. The XC5200 family is now mentioned in the description of the BUFG symbol in the External I/O Record description.
2. Descriptions of the allowed connectivity of the XC4000 and XC5200 special pads were edited for clarity. These changes were made in the sections on the OBUF, OUTFF, OUTFFT Symbols.
3. The XC4000E OUTFF, OUTFFT, and INFF symbols have a CE pin. The XC4000E INLAT Symbol has a GE pin.
4. The new XC4000E RAMS and RAMD symbols were added.
5. The description of the RAM/ROM symbols was changed to clarify how they are mapped into function generators and to describe the usage of the INIT parameter for the XC4000E family.
6. The description of the INIT parameter was modified to include the XC4000E RAM, RAMS and RAMD symbols.
7. The description of the RLOC and related parameters now indicates that they may also be applied to F5\_MUX and EQN symbols.
8. The entries in Table 1 for XC5200 F5MAP and F5\_MUX symbols have been clarified. The entries for the XC4000 FMAP, EQN, ROM, RAM and RAMS symbols have also been clarified.
9. The description of IOB pin names in Appendix A has been changed to reflect current usage.

10. A correction has been made to the changes to the eleventh draft in the twelfth draft. The TNM Signal and Symbol Parameters were modified, not new parameters.

The twelfth draft, Version 6.0, includes the following changes to the eleventh draft, Version 5.2:

1. All part families are now referenced with the XC prefix.
2. The XC7000 family is now described in a separate document, the XNF XC7000 Specification.
3. The LCANET version is now 6.
4. The example XNF file has been updated to use LCANET 6. Some symbol names were also changed.
5. The external connections of the I/O Symbols (IBUF, INFF, INREG, INLAT, OBUF, OBUFT, OUTFF and OUTFFT) were clarified.
6. References to the INIT and TNM parameters were removed from the I/O Block Parameters section.
7. The following new parameters were added:  
 DIVIDE1\_BY and DIVIDE2\_BY Symbol Parameters (OSC52 symbol only)  
 OSC Symbol Parameter (OSC52 symbol only)
8. Clarified or modified the descriptions of the following parameters:  
 BLKNM Symbol and EXT Parameter  
 TNM Symbol and EXT Parameter (qualifiers are not allowed on EXT Parameter)  
 DOUBLE Symbol Parameter  
 MAP Symbol Parameter  
 INIT Symbol Parameter  
 LIBVER Symbol Parameter  
 TSidentifier Symbol Parameter (additional characters are allowed in specification)  
 RLOC and related Symbol Parameters (only allowed for XC4000 and XC5200 families)  
 INTERNAL I/O Block Parameter  
 NODELAY I/O Block Parameter  
 S Signal Parameter  
 TNM Signal Parameter  
 TNM Pin Parameter
9. Clarified or modified the descriptions of the following symbols:  
 XC5200 IBUF Symbol (O pin is invertible)  
 TBUF Symbol (added description of “weak keeper” circuit)  
 OSC4 and OSC52 Symbols  
 Special Input Pads (TCI, TMS, TCK, MD0 and MD2)  
 Special Output Pads (TDO and MD1)  
 TIMESPEC Symbol  
 TIMEGRP Symbol
10. The GSR pins of all XC4000 family flip-flops, latches and registers are now invertible. The GR pins of all XC5200 family flip-flops are now invertible. The GTS pins of all XC4000 and XC5200 family three-state output symbols are now invertible.

Version 5.2 included many minor corrections to Version 5.1, which revised this document specifically to incorporate the 5200 family's architecture changes. The eleventh draft included the following revisions to the ninth draft (Version 5.0):

1. The entire document was re-formatted slightly to aid in the revision process.
2. Definitions of terms at the start of the LCA Library section were updated to include 5200 family information.
3. A new Carry Logic symbol was added, CY\_MUX.
4. New symbols were added to implement the 5200 family's 5-input function capability, F5\_MUX and F5MAP.
5. A new Oscillator Symbol was added, OSC52.
6. Seventeen symbol types were modified to indicate that they are not supported in the 5200 family. Those symbol types are: OUTFF, OUTFFT, INFF, INLAT, INREG, RAM/ROM, WAND/WORAND, ACLK/GCLK, BUFGP/BUFGS, OSC, CY4, Carry Modes, CLB primitive, CLBMAP, HMAP, IOB symbol, and OSC4.
7. All other symbols were modified as needed only to indicate clearly those changes required to support the 5200 family architecture.
8. LCA location naming conventions were defined for the 5200 family.

The ninth draft (Version 5.0) included the following changes to the eighth draft:

1. LCANET record is version 5.
2. The TIMEGRP symbol was added.
3. New Carry Logic symbols were added, CY4 and the 42 carry mode symbols, CY4\_01,..., CY4\_42.
4. The BUFG symbol from the Unified library was added.
5. The TS parameters associated with the TIMESPEC symbol are no longer "user" parameters (no preceding "=" necessary although the Xilinx software will support these as user parameters too).
6. The PAD and PADU symbols are obsolete with the Unified library and are replaced by IPAD, OPAD, IOPAD and UPAD.
7. The DOUBLE parameter on a PULLUP which is connected to an external is considered an error (it won't be ignored).
8. Signal names are no longer valid in EQN parameters, use pin names instead.
9. LOC ranges are specified using the colon (:) instead of a semi-colon (;), although both are supported.
10. The INFF and INLAT symbols no longer have the buffered input as part of the symbol.
11. The I pin is the invertible pin on a WAND with DECODE specified instead of the O pin.
12. Pin names have changed for the following (LCANET 4 pin names will still be supported):  
Combinatorial gates have changed input pin names from 1-5 to I0-I4.

WORAND input pin names have changed from I1 and I2 to I0 and I1.

INLAT and DLAT L pin name has changed to G.

DFF and DLAT RD pin name has changed to CLR (clear).

DFF and DLAT SD pin name has changed to PRE (preset).

13. New signal and pin parameters, TNM and TSidentifier were added.
14. New symbol and EXT parameters, TNM and HBLKNM were added. Also new I/O parameters for the 4000H Xilinx architecture, TTL, CMOS, RES and CAP were added.
15. New symbol parameters, CYMODE, SCHNM, LIBVER, TS, SYSTEM, RLOC, USE\_RLOC, U\_SET, HU\_SET, RLOC\_ORIGIN and RLOC\_RANGE were added.
16. The following symbol parameters are now obsolete: FREQ, PERIOD, THI, TLO.

The eighth draft included the following changes to the sixth draft:

11. Two new I/O properties, MEDFAST and MEDSLOW were added in order to support the Xilinx 4000A architecture.
12. Obsoleted the following symbols: 4000 family CLB and IOB, BITCFG and CARRY.
13. Obsoleted the following parameters: MODE, DS and DE. Also the SC (skew critical) parameter was modified so that it does not accept a delay value.
14. The output pin of the WAND symbol is invertible if the symbol has the DECODEparameter attached to it. Also wide input WAND gates will no longer be generated by Xilinx software since most simulators could not support this. Only the 1-input WAND gate will show up in an XNF file.
15. The new 4000 family mapping symbols, FMAP and HMAP were added.
16. The new TIMESPEC symbol was added.
17. New USER properties were added to support X-BLOX and TIMESPEC.
18. The DEF parameter now includes support for X-BLOX.
19. The INIT parameter is no longer allowed on the RAM symbol.
20. The invertible pins for the 3000 and 4000 family OBUFT have been corrected to read I and T instead of O and T.

The seventh draft included the following changes to the sixth draft:

11. The INLAT description on page 50 was changed to say that the latch is transparent high for the 3000 family and low for the 4000 family.
12. Some typos were fixed in the location name examples in Appendix B.

The sixth draft (Version 4.0) included the following changes to the fifth draft:

1. The LCANET record currently supported has changed to 4. Note that there never existed a version 3, this was intentionally skipped.
2. The BUS record was added.



3. The maximum line length of any one line in the XNF has increased from 1024 characters to 2048. The maximum field length of any record including name lengths has been increased from 512 characters to 1024.
4. All references to the “back-annotation” program were replaced by “the Xilinx program that creates an XNF file from an LCA”.
5. The HIER record is no longer supported.
6. The IOBMAP symbol is no longer supported.
7. The direction field for PIN records allows a direction of “B” (bi-directional) for macro symbols only.
8. New symbol properties: LOC<=>nn, INIT, DECODE, EQN, NODELAY, MODE, FREQ, PERIOD, THI, TLO, DEF.
9. New signal properties: W=weight, SC[=ns]. Also the new properties FREQ, PERIOD, THI, TLO and NODELAY are allowed on EXT (external) records.
10. New pin properties: DS-name[=ns], DE-name[=ns].
11. The OBUFZ symbol name was changed to OBUFT.
12. The OUTFFZ symbol name was changed to OUTFFT.
13. The width field of the PULSE record has changed from being a numeric field to a delay field (floating point).
14. The comment field of the PROG record was changed and should be contained within double quotes (“comment”).
15. New 4000 family primitives were added: INREG, EQN, CARRY, BITCFG, RAM, ROM, PULLDOWN, BUFGP, BUFGS, WAND, WORAND, all the special function access symbols (STARTUP, BSCAN, OSC4, RDBK, RDCLK, TDI, TDO, TMS, TCK, MD0, MD1, MD2) as well as the 4000 versions of DFF and description of the GSR pin.
16. The clock enable pin for the 3000 CLB and CLBMAP symbols was changed from CE to EC.
17. The addition of two new MAP= properties for CLBMAP symbols (PLO, PUO) which supports “open” CLBs.

The fifth draft (Version 2.0) included the following changes to the fourth draft:

1. The LCANET record was changed to 2.
2. Addition of the new “MAP=type” symbol parameter for use with the CLBMAP and IOBMAP symbols.
3. Addition of the “S” (Save) signal parameter.
4. Addition of the “P” (Pin lock) pin parameter for use with the CLBMAP and IOBMAP symbols.
5. Addition of two new mapping control symbols, CLBMAP and IOBMAP.
6. Support of full hierarchical path names was added along with the reserved character “/” to be used as the path name separator.

7. The maximum field length of any record including name lengths was increased from 255 characters to 512. (See section 3.2 and Appendix A)
8. Delay fields were changed to floating point values.

The fourth draft included the following changes to the third draft:

1. Support for hierarchy was added. Macro symbols were defined, which are just symbols with an unknown type. The Xilinx merge program must be used to flatten designs with macro symbols before running the technology mapper.
2. The FILE parameter was added for SYM records.
3. The PIN parameter was added for SIG records.
4. The HIER record type was added.
5. The connection of the input pin of the GCLK and ACLK symbols to EXT records was defined as legal in the 3000 family.
6. The DIV2 parameter was eliminated from the OSC symbol.

The third draft (Version 1.0) included the following changes to the second draft:

1. The revision number on the LCANET record has been changed from 0 to 1, to reflect the changes listed below.
2. Changed the name of the three-state output buffer from TBUF to OBUFZ. Changed the name of the internal three-state driver from TRI to TBUF.
3. A pin number field has been added to the EXT record. This information is useful for board level interfaces.
4. The VCC and GND parameters on signals were removed in favor of the new PWR record. This allows simulators to determine the power connections without parsing the signal parameters. It also simplifies the definition of power signals for schematic editors which use naming conventions to define signals tied high or low.
5. MODEL and ENDMOD record types were added for defining simulation models for CLB and IOB symbols.
6. The PULSE record type was added for defining minimum pulse widths on certain signals.
7. The USER record was added to allow external programs to include information not included in the standard syntax. Also, a user defined parameter type was added for PIN, SIG, SYM and EXT records.
8. The PROG record no longer defines separate fields for the date and time. The date and time should still be included, in the comment field, but explicit fields are no longer defined for them.
9. The GR pin has been added to all latch and flip-flop symbols. This pin corresponds to the global reset signal of the LCA and is used for simulation purposes only.
10. The DIV2 parameter has been defined for the OSC symbol in the 3000 family.
11. Use of the PULLUP symbol to indicate the programmable pull-up resistor on the external I/O pins of the 3000 family has been defined.

12. The DOUBLE parameter was added for the PULLUP symbol.

The following changes were made from the first draft to the second draft:

1. The user-definable parameters have been fully documented in the section called Parameter Definitions.
2. Clock pins have been added to the IOB symbol. They were left out of the previous version by mistake. In the 2000 family, the clock pin is called K. In the 3000 family, there are 2 clock pins, IK and OK, for the input and output flip-flops respectively.
3. The C pin on the flip-flops in the IOBs in the 3000 family are now listed as invertible.
4. The DIN pin on the CLB symbol has been changed to DI. This is to conform to the CLB definition in the Design Editor.
5. The EXF record name has been changed to LCANET. The name of the program which converts the Xilinx netlist file into the LCA file is called XNF2LCA. XNF2LCA supports a default suffix for the LCA Xilinx netlist file of XNF.
6. The VCC parameter has been changed to VDD.
7. The U direction has been added to EXT records to identify unbonded IOBs.
8. The LCA naming restrictions have been documented in Appendix A.



## Index

### Symbols

= Parameter 46

### A

ACLK Symbol 80

AND Symbol 53

### B

BLKNM Parameter 28

Block Name Parameter 28

    Hierarchical 29

Boundary Scan Symbol 100

BSCAN Symbol 100

BUF Symbol 52

Buffer

    Global 81

    Global for Unified Library 82

    Input 55

    Output 56

    Output Three-State 57

    Symbol 52

BUFG Symbol 82

BUFGP Symbol 81

BUFGS Symbol 81

Bus Record 20

### C

C Signal Parameter 42

CAP Parameter 41

Carry Logic

    CYMODE Parameter 33

    Symbol 86

Carry Mode Symbol 87

CFG Record 15

Char Field 7

CLB

Location Names B-1, B-2, B-4

MAP Parameter 31

Symbol 90

CLBMAP Symbol 91

Clock

    Drivers 80

    for Readback Symbol 101

CMOS Parameter 41

Combinational Logic Symbols 53

Critical Routing C Parameter 42

CY\_MUX Symbol 86

CY4 Symbol 86

CYMODE Parameter 33

### D

DECODE Parameter 32

DEF Parameter 32

Definition

    Delay 6

    Parameter 6

    Pin 5

    Signal 5

    Symbol 5

    VCC/GND Signal 6

Delay 6

    Field 7

Design Editor Configuration Command

    Discussion C-1

    Record 15

Designing LCAs 2

DFF Symbol 69

DIVIDE1\_BY Parameter 39

DIVIDE2\_BY Parameter 39

DLAT Symbol 71

DOUBLE Parameter 31

Driver

    Clock 80

D-Type

    Flip-Flop 69

    Latch 71

## E

- END Record 18
- ENDMOD Record 17
- EOF Record 25
- EPLD 1
- EQN
  - Parameter 32
  - Symbol 83
- Equation
  - Parameter 32
  - Symbol 83
- EXT
  - Parameter Summary 48
  - Parameters 28–41
  - Record 22
- External I/O Record 22

## F

- F5\_MUX Symbol 85
- F5MAP Symbol 92
- FAST Parameter 40
- F-Function Generator Symbol 91
- Field 6
  - Char 7
  - Delay 7
  - Numeric 7
  - Parm 7
  - String 7
- FILE Parameter 31
- Five-Input Functions 85, 92
- Flip-Flop
  - D-Type 69
  - INIT Parameter 32
  - Input 63
  - Output 59
  - Three-State Output 61
- FMAP Symbol 91

## G

- G Pin Parameter 44
- GCLK Symbol 80
- Global Buffers 81
  - Unified Library 82

## H

- HBLKNM Parameter 29
- H-Function Generator Symbol 92
- Hierarchical Block Name Parameter 29
- HMAP Symbol 92
- HU\_SET Parameter 36

## I

- I Pin Parameter 44
- I/O
  - Block Symbol 94
  - Pad Symbols 54
- IBUF Symbol 55
- INFF Symbol 63
- INIT Parameter 32
- INLAT Symbol 65
- Input
  - Buffer 55
  - Flip-Flop 63
  - Latch 65
    - Symbol 65
  - Pad Symbols 54
  - Pads, Special 102
  - Register (obsolete) 67
- INREG Symbol (obsolete) 67
- INTERNAL Parameter 40
- Internal Three-State Driver
  - TBUF 76
  - Wired-AND 77
  - Wired-OR 77
- Inter-Program Information Transfer 38
- INV Pin Parameter 44
- INV Symbol 52

Invert Pin Parameter 44

Inverter Symbol 52

IOB

Location Names A-1, B-1, B-2, B-4

Pad Names A-1

Pin Names A-1

Symbol 94

IOPAD Symbol 54

IPAD Symbol 54

## K

K Pin Parameter 44

## L

L Signal Parameter 42

Latch

D-Type 71

Input 65

LCA 1, 2

Location Names B-1

Names A-1

Netlist Syntax Summary 26

LCANET Record 8

Library Version Parameter 33

LIBVER Parameter 33

LOC Parameter 28, B-6

Location

Parameter 28

Relative Placement 34

Logic Symbols 53

Logical End-of-File Record 25

Longline Routing L Parameter 42

## M

Macro

PIN Parameter 43

Symbol 103

MAP Parameter 31

MD0 Symbol 102

MD1 Symbol 102

MD2 Symbol 102

MEDFAST Parameter 40

MEDSLOW Parameter 40

Memory

RAM 72

RAMD 74

RAMS 75

ROM 72

Model End Record 17

Model Record 16

## N

N Signal Parameter 42

Names A-1, B-1

CLB Locations B-1, B-2, B-4

IOB Locations A-1, B-1, B-2, B-4

IOB Pads A-1

IOB Pins A-1

Reserved A-1

Naming Restrictions A-1

NAND Symbol 53

NODELAY Parameter 40

Non-Critical Routing N Parameter 42

NOR Symbol 53

Numeric Field 7

## O

OBUF Symbol 56

OBUFF Symbol 57

OPAD Symbol 54

OR Symbol 53

OSC Parameter 39

OSC Symbol 98

OSC4 Symbol 98

OSC52 Symbol 98

Oscillator

4000 Symbol 98

5200 Symbol 98

On-Chip 98	I/O
OUTFF Symbol 59	CAP 41
OUTFFT Symbol 61	CMOS 41
Output	FAST 40
Buffer 56	INTERNAL 40
Three-State 57	MEDFAST 40
Flip-Flop 59	MEDSLOW 40
Three-State 61	NODELAY 40
Pad Symbol 54	RES 41
Pads, Special 102	SLOW 40
	TTL 41
<b>P</b>	INIT 32
P Pin Parameter 44	INTERNAL 40
PAD Symbol (obsolete) 54	INV 44
Pad Symbols 54	Invert Pin 44
PADU Symbol (obsolete) 54	K 44
Parameter 6, 44	L 42
= 46	Library Version 33
BLKNM 28	LIBVER 33
Block Name 28	LOC 28, B-6
C 42	Location 28
CAP 41	MAP= 31
Carry Mode (CYMODE) 33	MEDFAST 40
CMOS 41	MEDSLOW 40
CYMODE 33	N 42
DECODE 32	NODELAY 40
DEF 32	OSC 39
DIVIDE1_BY 39	P 44
DIVIDE2_BY 39	PIN 43
DOUBLE 31	Pin
EQN 32	G 44
EXT 28–41	I 44
EXT Summary 48	INV 44
FAST 40	K 44
Field 7	P 44
FILE= 31	TNM 44
G 44	TSidentifier 45
HBLKNM 29	Pin Summary 48
Hierarchical Block Name 29	RES 41
HU_SET 36	RLOC 34
I 44	RLOC_ORIGIN 37
	RLOC_RANGE 37
	S 43
	SC 42



Schematic Name 33	PLD 1
SCHNM 33	Power Record 21
Signal	PROG Record 9
C 42	Program Record 9
L 42	Pull-Down Resistor 79
N 42	PULLDOWN Symbol 79
S 43	Pull-Up
SC 42	DOUBLE Parameter 31
TNM 43	Resistor 78
TSidentifier 43	PULLUP Symbol 78
W 42	PULSE Record 14
X 42	Pulse Width Record 14
Signal Summary 48	PWR Record 21
SLOW 40	
Summary 47	<b>R</b>
Symbol 28–41	RAM Symbol 72
Symbol Summary 47, 48	RAMD Symbol 74
SYSTEM 38	RAMS Symbol 75
Timespec TSidentifier 34	RDBK Symbol 100
TNM	RDCLK Symbol 101
EXT 30	Readback
Pin 44	Clock Symbol 101
Signal 43	Symbol 100
Symbol 30	Record 6
TSidentifier	Bus 20
Pin 45	CFG 15, C-1
Signal 43	Design Editor Configuration
Symbol 34	Command 15, C-1
TTL 41	END 18
U_SET 36	ENDMOD 17
USER_RLOC 36	EOF 25
User-Defined 46	EXT 22
W 42	External I/O 22
X 42	LCANET 8
X-BLOX 46	Logical End-of-File 25
Parm Field 7	Model 16
PART Record 10	Model End 17
Part Type Record 10	PART 10
Pin 5	Part Type 10
Parameter Summary 48	Pin 12
Record 12	
Pin Lock Parameter (P) 44	
PIN Parameter 43	

Power 21	<b>S</b>
PROG 9	S Signal Parameter 43
Program 9	Save Signal Parameter 43
PULSE 14	SC Signal Parameter 42
Pulse Width 14	Schematic Name Parameter 33
PWR 21	SCHNM Parameter 33
SETUP 13	SETUP 13
Setup and Hold Time 13	Setup and Hold Time Record 13
SIG 19	SIG Record 19
Signal 19	Signal 5
SYM 11	Parameter Summary 48
Symbol 11	Record 19
Symbol End 18	VCC/GND 6
USER 24	Simulation 2, 3
User Program 24	Skew Critical Routing SC Parameter 42
Register	SLOW Parameter 40
Input (obsolete) 67	Special Pads
Relative Location Parameters	Input 102
HU_SET 36	Output 102
RLOC_ORIGIN 37	STARTUP Symbol 101
RLOC_RANGE 37	String Field 7
U_SET 36	Summary
USE_RLOC 36	LCA Netlist Syntax 26
Relative Placement 34	Parameters 47
RES Parameter 41	Symbol Types 104
Reserved Names A-1	SYM record 11
Resistor	Symbol 5
Pull-Down 79	End Record 18
Pull-Up 78	Parameter Summary 47, 48
RLOC Parameter 34	Parameters 28–41
RLOC_ORIGIN Parameter 37	Record 11
RLOC_RANGE Parameter 37	Type Definition 32
ROM Symbol 72	Syntax 6
Routing Parameter	Summary 26
C 42	SYSTEM Parameter 38
L 42	
N 42	<b>T</b>
SC 42	TBUF Symbol 76
W 42	TCK Symbol 102

TDI Symbol 102  
TDO Symbol 102  
Three-State  
    Driver  
        Internal Three-State 76  
        Wired-AND 77  
        Wired-OR 77  
    Output  
        Buffer 57  
        Flip-Flop 61  
TIMEGRP Symbol 97  
Timespec Parameters  
    TNM EXT 30  
    TNM Pin 44  
    TNM Signal 43  
    TNM Symbol 30  
    TSidentifier Pin 45  
    TSidentifier Signal 43  
    TSidentifier Symbol 34  
TIMESPEC Symbol 95  
TMS Symbol 102  
TNM EXT Parameter 30  
TNM Pin Parameter 44  
TNM Signal Parameter 43  
TNM Symbol Parameter 30  
TSidentifier Pin Parameter 45  
TSidentifier Signal Parameter 43  
TSidentifier Symbol Parameter 34  
TTL Parameter 41

## U

U\_SET Parameter 36  
UPAD Symbol 54  
USE\_RLOC Parameter 36  
User Program Record 24  
USER Record 24  
User-Defined Parameters 46

## V

VCC/GND Signal 6

Version  
    Library Parameter 33

## W

W Signal Parameter 42  
WAND  
    DECODE Parameter 32  
    Symbol 77  
Weights  
    Routing W Parameter 42  
Wired-AND  
    DECODE Parameter 32  
WORAND Symbol 77

## X

X Signal Parameter 42  
X-BLOX Parameters 46  
Xilinx Netlist Format, see XNF  
XNF 1, 5  
    File Format 1  
    Syntax 6  
XNOR Symbol 53  
XOR Symbol 53