Altera to Lattice Semiconductor
Design Conversion Utility Application Notes

These application notes describe how to install and use the Altera to Lattice Semiconductor Design Conversion Utility. The following topics are included:

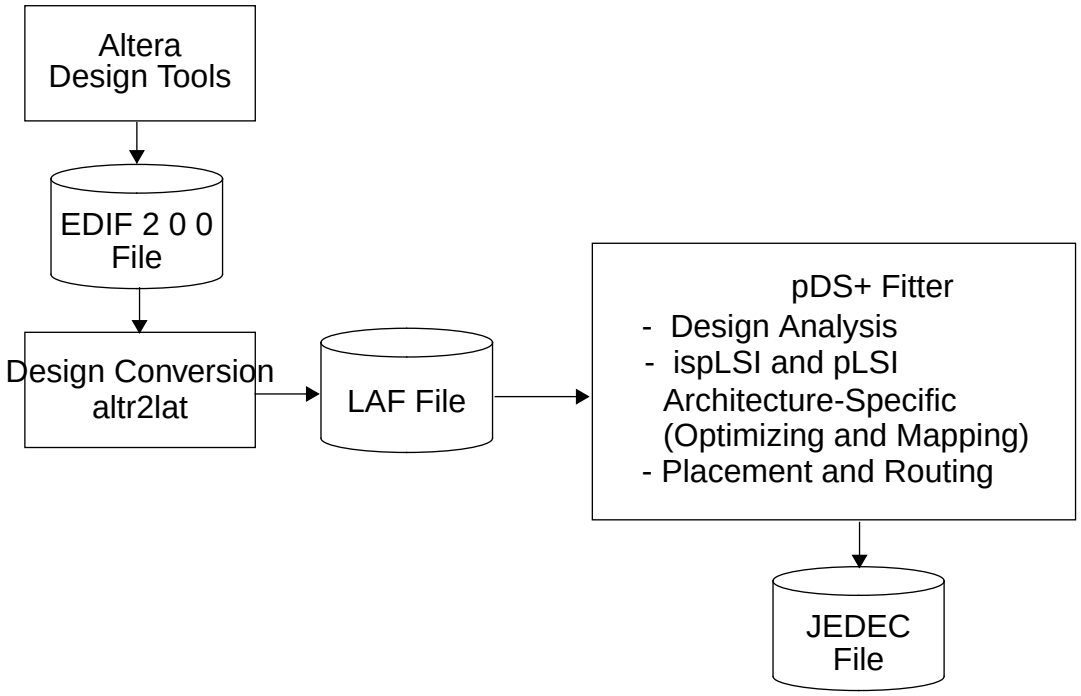
- Introduction
- Design Conversion Steps
- Prefixes, VCC, and GND
- Asynchronous Signal Handling
- Macro Support
- DFFE Mapping
- Command Line Options
- Issues and Solutions

Introduction

The Altera to Lattice Semiconductor Design Conversion Utility allows you to take an Altera fitted design directly into the Lattice Semiconductor (LSC) environment for fitting and device programming. This version of the conversion program (version 2.1) has the following enhancements:

- Supports EDIF array and bundle constructs
- DFFE, TRIBUF macro support
- Better mapping for DFFE macros
- Asynchronous clear and asynchronous set mapping options
- Cleans up dangling nets and blocks after mapping
- Mapping for wide input Boolean functions

The basic design flow is shown on below.



Design Conversion Steps

Installation

If the Altera to Lattice Semiconductor Design Conversion Utility **altr2lat** is not in the location specified below, copy it into the bin directory for pDS+.

UNIX

```
cp altr2lat <pdsplus_install_path>/bin
```

PC

```
copy altr2lat <pdsplus_install_path>\bin
```

Environment Setup

UNIX

Ensure that the pDS+ Fitter environment is set correctly in your `.cshrc` file before running the Altera to Lattice Semiconductor Design Conversion Utility. The pDS+ Fitter environment setup is shown below.

```
setenv PDSPLUS <pdsplus_install_path>

set path = ($path $PDSPLUS/bin)

setenv LM_LICENSE_FILE <path_license_file>/license.dat:
$LM_LICENSE_FILE
```

PC

Ensure that the pDS+ Fitter environment is set correctly in your `autoexec.bat` file before running the Altera to Lattice Semiconductor Design Conversion Utility. The pDS+ Fitter environment setup is shown below.

```
set PDSPLUS = <pdsplus_install_path>

set path = %path%;<pdsplus_install_path>\bin

set PDS_LIC = <pdsplus_license_path>
```

Running the Design Conversion Utility

To perform design conversion:

1. Fit your design using the Altera fitter and write out the fitted design in EDIF 2 0 0 format.

2. Perform design conversion by entering:

   altr2lat -edif *design_name*.edo

3. Invoke the pDS+ Fitter by entering:

   dpm -if laf -i *design_name*.laf -of *output_format*

   The `-of` *`output_format`* option specifies which output format you wish the pDS+ Fitter to create. The following output formats are available: `edif`, `ldf`, `orcad`, `preroute_ldf`, `verilog`, `vhdl`, or `viewlogic`.

4. Program an ispLSI or pLSI device with the JEDEC file that the pDS+ Fitter generates when the design has routed successfully. See the *ISP Daisy Chain Download Reference Manual* for information on device programming.

Prefixes, VCC, and GND

When writing out your EDIF 2 0 0 format design file from your Altera design tool, you have the option of specifying a vendor format, a VCC setting, and a GND setting from the EDIF Netlist Writer Settings menu. You may, or may not, need to use additional commands when running the Altera to Lattice Semiconductor Design Conversion Utility depending on the selections you make. To determine whether additional commands are necessary, check for the following in your file:

- Added prefixes
- VCC Setting
- GND Setting

If you specify a vendor format when writing out your EDIF 2 0 0 format design file, prefixes may be added to cell names in your file; for example, an `AND2` cell with an "`A_`" prefix appears as "`A_AND2`." To preserve the functionality and mapping of the cells that have prefixes, add the `-prefix` *`prefix_name`* option to the design conversion command line statement as shown below.

Syntax:

```
altr2lat -edif design_name.edo -prefix prefix_name
```

Example:

```
altr2lat -edif counter.edo -prefix A_
```

If you used a name other than VCC to specify power in your design (for example, VDD), you must add the `-vcc` *`your_vcc_name`* option to the design conversion command line statement as shown below.

Syntax:

```
altr2lat -edif design_name.edo -vcc your_vcc_name
```

Example:

```
altr2lat -edif counter.edo -vcc VDD
```

If you used a name other than GND to specify ground in your design (for example, GRND), you must add the `-gnd` *`your_gnd_name`* option to the design conversion command line statement as shown below.

Syntax:

```
altr2lat -edif design_name.edo -gnd your_gnd_name
```

Example:

```
altr2lat -edif counter.edo -gnd GRND
```

By default, the Altera to Lattice Semiconductor Design Conversion Utility assumes that VCC and GND are represented as nets in your EDIF design file. If they are represented as cells, you will receive an error message from the pDS+ Fitter. Add the `-vcc_gnd_cell` option to the design conversion command line statement as shown below.

Example:

```
altr2lat -edif design_name.edo -vcc_gnd_cell
```

NOTE

If your design requires more than one of these design conversion command line options, you can enter them in any order in the same command line statement.

Asynchronous Signal Handling

Due to architectural differences in the Altera and Lattice Semiconductor devices, certain designs may not map directly into the LSC environment. This is because LSC device architecture does not support multiple asynchronous signals in a single sequential element.

An Altera DFF or DFFE may have both asynchronous set and clear. A one-to-one mapping is not always possible when both of the asynchronous signals are used. In most cases, the asynchronous pin “CLRN” and PRN” are not directly connected to “VCC” or “GND” nets. Instead, both “CLRN” and “PRN” have associated logic functions, as shown in Figure 1.

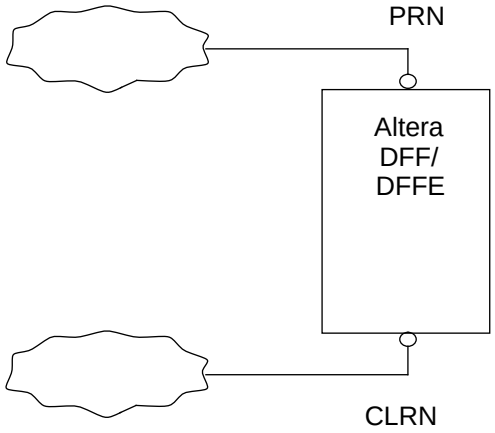


Figure 1. Original Block Before Mapping

Before a DFF or DFFE is mapped, the logic functions that drive the “CLRN” pin or “PRN” pin are evaluated to logic 1 and mapped as shown in Figure 2.

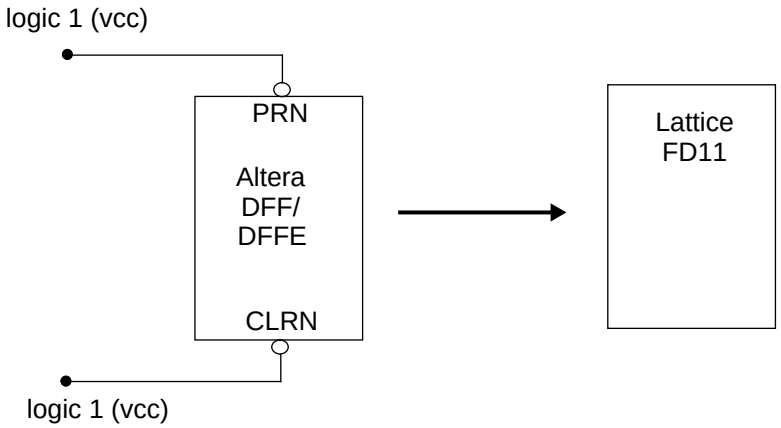


Figure 2. Mapping After Evaluation to Logic 1

Figure 3 shows the mapping where only the logic function connected to the “CLRN” pin evaluates to logic 1.

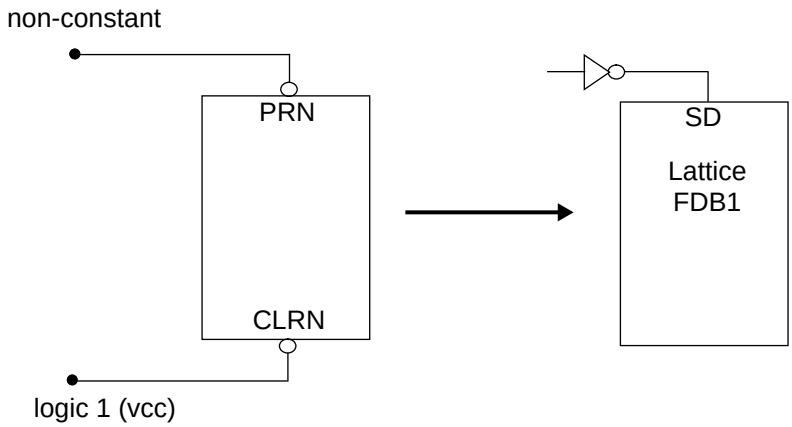


Figure 3. Logic Function Connected to “CLRN” pin Evaluates to Logic 1

Figure 4 shows the mapping where only the logic function connected to the “PRN” pin evaluates to logic 1.

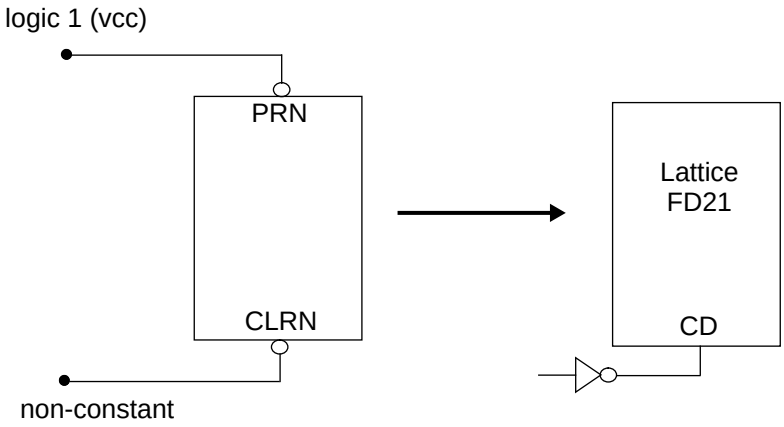


Figure 4. Logic Function Connected to “PRN” pin Evaluates to Logic 1

If both logic functions evaluate to non-constant logic values, you may need to change your original design to eliminate one of the asynchronous signals, or you may use `-no_clear` or `-no_preset` command line options to have one of the asynchronous signals automatically removed for you.

Figure 5 shows the mapping of a DFF or DFFE when the `-no_clear` option is used.

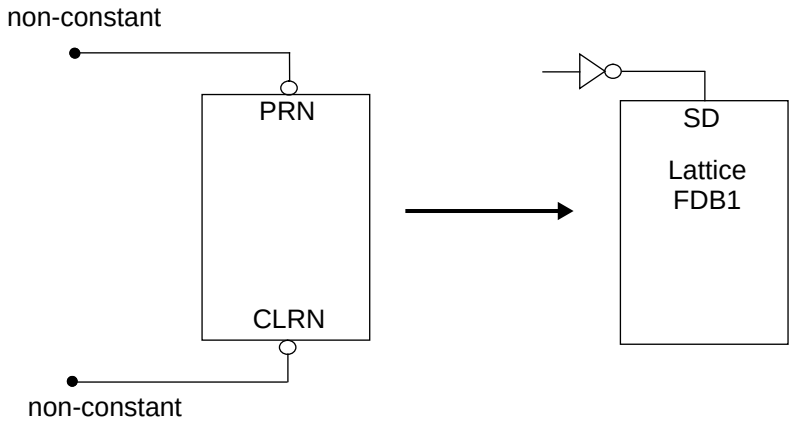


Figure 5. Mapping After Using `-no_clear`

Figure 6 shows the mapping of a DFF or DFFE when the `-no_preset` option is used.

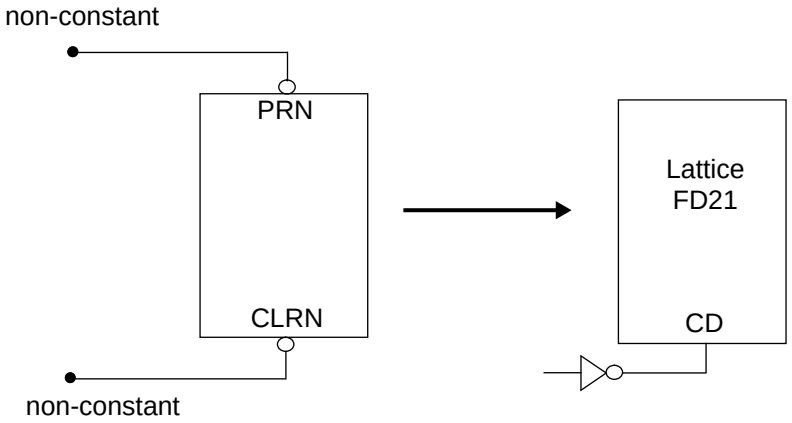
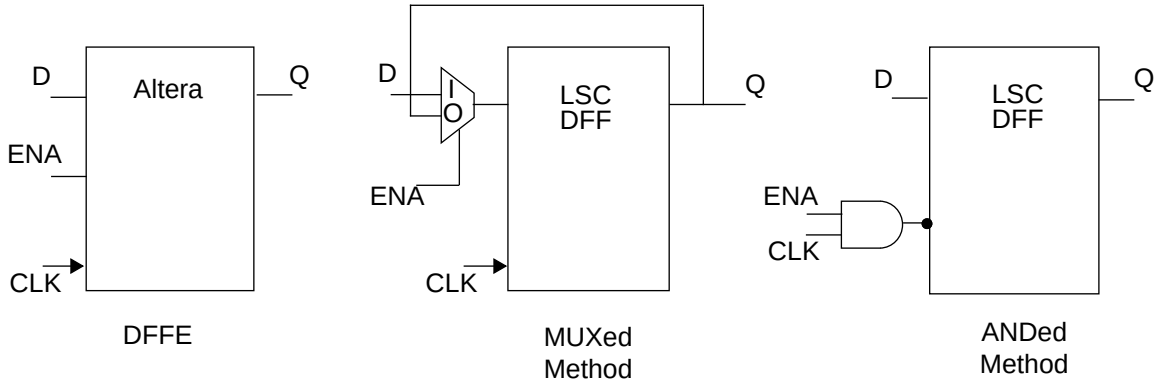


Figure 6. Mapping After Using `-no_preset`

In the logic evaluation phase for the signals connected to the “PRN” pin and “CLRN” pin, the conversion program must know the name of the power and ground signals. The default names are “VCC” and “GND.” If this is not the case, you must use `-vcc` *`VCC_name`* and `-gnd` *`GND_name`* when invoking the design conversion program so logic evaluation can be done correctly.

DFFE Mapping

DFFE can be mapped in two ways. The default mapping is to use a MUX to select between D and Q; the second method is to AND CLK and ENA signals.



If you need to use the ANDed method, use the `-ena_and` option at the command line. Please note that the MUXed mapping method gives significantly better device utilization.

Macro Support

The following list contains the macros which the Altera to Lattice Semiconductor Design Conversion Utility supports:

- AND functions: AND1 through AND128
- NAND functions: NAND1 through NAND128
- Delay buffer: DELAY
- D Flip-Flop: DFF, DFFE
- Filter: FILTER
- D Latch: LATCH
- NOR functions: NOR1 through NOR128
- OR functions: OR1 through OR128
- Inverter: NOT
- Risefall: RISEFALL
- Tristate: TRI, TRIBUF
- XNOR: XNOR2
- XOR: XOR2

The list contains a complete set of Altera physical primitives for all releases of Maxplus II up to version 6.0. If you use a Maxplus II version newer than 6.0, and you get an error from the conversion program or compiler on unknown primitives, contact your Lattice Semiconductor representative to receive an updated design conversion program.

Command Line Options

Multiple command line options may be specified when using **altr2lat**. For example:

```
altr2lat -edif design.edo -prop design.prp -no_preset.
```

A list of command line options can be obtained by typing **altr2lat** without any arguments, or **altr2lat -help** for more detailed descriptions.

File Formats and Naming

- **-edif** edif_file_name – Specifies the name of the EDIF netlist file in which **edif2laf** is to read.
- **-fname** laf_file_name – Specifies a name for the output LAF file from 1cb. Default is the input EDIF file name.
- **-fext** laf_file_extension – Specifies a different dot extension for the output LAF file from **edif2laf**. The default dot extension is laf.
- -**help** – Prints a detailed help message

VCC and GND Handling

- **-vcc** vcc_net or cell_name – Specifies a name for a Vcc net or cell. The default name is VCC.
- **-gnd** gnd_net or cell_name – Specifies a name for a GND net or cell. The default name is GND.
- **-vcc_gnd_cell** – Specifies the type of representation for Vcc or GND in the input EDIF file. The default representation is a net.
- -**prefix** – Altera cell prefix_name. There is no default for this option.

Attribute Handling

- **-prop** property_file_name – Directs **edif2laf** to read in a Property File containing Design Attributes. If you need to use LSC tabulates for pin locking and other device control options, refer to the appropriate “Third Party Vendor and pDS+ Design Environment User Manual” for your design environment.

DFF/Latch Mapping options

- **-no_clear** – Ignores all asynchronous clear.
- **-no_preset** – Ignores all asynchronous preset.
- **-ena_and** – Maps @DFFE with the ANDed CLK and ENA method.

Issues and Solutions

The following issues have been identified and the solutions determined.

Issue: The following message appears when running the design conversion process:

```
lafout:net U14x2 is not driving anything.
```

Solution: Disregard this message. The design conversion utility does not remove redundant logic; this task is performed by the pDS+ Fitter during the logic optimization process.

Issue: The following message appears when you run the pDS+ Fitter:

```
32269 ERROR: Primitive 'FDB1' is invalid.
```

Solution: This error message appears when the definition for a macro cannot be found. In this case, the definition for FDB1 is missing. Ensure that you have the $PDSPLUS environment variable set before rerunning the design conversion process.

Issue: The following message appears when you run the pDS+ Fitter:

```
32269 ERROR: Primitive 'VCC' is invalid.
```

Solution: The design conversion utility assumes that VCC and GND are represented as nets in your EDIF file. If you receive this error message, VCC is entered as a cell. You must use the **`-vcc_gnd_cell`** option so the design conversion program will recognize it and process it properly.

Issue: The following message appears when you run the pDS+ Fitter:

```
32269 ERROR: Primitive 'A_TRI' is invalid.
```

Solution: This message appears if you have specified a vendor format when writing your EDIF 2 0 0 format design file, which adds prefixes to the cell names. In this case, you must use prefix option **`-prefix A_`** when running the **`altr2lat`** command.

Issue: The following message appears when running the design conversion process:

```
ERROR: Block [DFF_662] of type [DFF] cannot be mapped. You
may want to modify your original design so that your FF/Latch
elements do not contain BOTH async Clear and async Set.
Or you may use the -no_clear or -no_preset options to have
the program automatically ignore one of the two async
signals.
```

Solution: Ensure that VCC and GND are identified correctly. If you have used names other than VCC and GND, the sequential element mapping may not be performed properly. Use the `-vcc` *`your_vcc`* and `-gnd` *`your_gnd`* options when rerunning the design conversion utility to correctly identify power and ground.

If VCC and GND are identified correctly, you must change your original design so that you do not use both asynchronous clear and asynchronous set on a single sequential element. LSC device architecture does not support multiple asynchronous signals in a single sequential element.