PDC Commands Reference Guide for SmartFusion 2, IGLOO 2, and RTG4 FPGAs

Libero SoC v2025.1

Introduction (Ask a Question)

In the FPGA design world, constraint files are as important as design source files. Physical Design Constraints (PDC) are used to constrain the I/O attributes, placement, and routing during the physical layout phase.

You can enter PDC commands manually using the Libero® SoC Text Editor. PDC commands can also be generated using Libero SoC's interactive tools. The I/O Attribute Editor is the interactive tool for making I/O attribute changes, and the Chip Planner is the interactive tool for making floorplanning changes.

When changes are made in the I/O Attribute Editor or the Chip Planner, the PDC file(s) are updated to reflect the changes. These PDC commands can be used as part of a script file to constrain the Place and Route step of your design.

Supported Families (Ask a Question)

This document describes the PDC commands applicable to SmartFusion® 2, IGLOO® 2, and RTG4™ families.

Table of Contents

1. PDC Syntax Conventions (Ask a Question)

The following table shows the typographical conventions that are used for the PDC command syntax.

Table 1-1. PDC Syntax Conventions

Syntax Notation	Description
`command` `-argument`	Commands and arguments appear in Courier New typeface.
`variable`	Variables appear in Courier New typeface. You must substitute an appropriate value for the variable.
`[-argument value] [variable]+`	Optional arguments begin and end with a square bracket with one exception: if the square bracket is followed by a plus sign (+), users must specify at least one argument. The plus sign (+) indicates that items within the square brackets can be repeated. Do not enter the plus sign character.

Note: PDC commands are case sensitive. However, their arguments are not.

1.1. Examples (Ask a Question)

Syntax for the `set_io` command followed by a sample command:

```
set_io portname [-iostd value][-register value][-out_drive value][-slew value][- res_pull
value][-out_load value][-pinname value][-fixed value][-in_delay value] set_io ADDOUT2 \
-iostd PCI \
-register yes \
-out_drive 16 \ -slew high \
-out_load 10 \
-pinname T21 \
-fixed yes
```

1.2. Wildcard Characters (Ask a Question)

You can use the following wildcard characters in names used in PDC commands:

Table 1-2. Wildcard Characters

Wildcard	What It Does
\	Interprets the next character literally
?	Matches any single character
*	Matches any string

Note: The matching function requires that you add a slash (\) before each slash in the port, instance, or net name when using wildcards in a PDC command.

1.3. Special Characters ([ ], { }, and \) (Ask a Question)

Sometimes square brackets are part of the command syntax. In these cases, you must either enclose the open and closed square brackets characters with curly brackets or precede the open and closed square brackets characters with a backslash (\). If you do not, you will get an error message.

For example:

```
set_iobank {mem_data_in[57]} -fixed no 7 2
```

```
or set_iobank mem_data_in\[57\] -fixed no 7 2
```

1.4. Entering Arguments on Separate Lines (Ask a Question)

To enter an argument on a separate line, you must enter a backslash (\) character at the end of the preceding line of the command as shown in the following example:

```
set_io ADDOUT2 \
-iostd PCI \
-register Yes \
-out_drive 16 \
-slew High \
-out_load 10 \
-pinname T21 \
-fixed yes
```

2. PDC Naming Conventions (Ask a Question)

Note: The names of ports, instances, and nets in an imported netlist are sometimes referred to as their original names.

2.1. Rules for Displaying Original Names (Ask a Question)

Port names appear exactly as they are defined in a netlist.

Instances and nets display the original names plus an escape character (\) before each backslash (/) and each slash (\) that is not a hierarchy separator. For example, the instance named A/\B is displayed as A\/\\B.

2.2. Which Name Do I Use in PDC Commands? (Ask a Question)

When writing PDC commands, follow these rules:

- Always use the macro name as it appears in the netlist
- Names from a netlist: For port names, use the names exactly as they appear in the netlist. For instance and net names, add an escape character (\) before each backslash (\) and each slash (/) that is not a hierarchy separator.
- For wildcard names, always add an extra backslash (\) before each backslash
- Always apply the PDC syntax conventions to any name in a PDC command

The following table provides examples of names as they appear in an imported netlist and as they must appear in a PDC file.

Table 2-1. PDC Command Names with Examples

Type of Name and its Location	Name in the Imported netlist	Name to use in PDC File
Port name in netlist	A/:B1	A/:B1
Instance name in a netlist	A/:B1 A$(1)	A\\/:B1 A$(1)
Instance name in the netlist but using a wildcard character in a PDC file	A/:B1	A\\\\/:B*
Net name in a netlist	Net1/:net1	Net1\\/:net1

When exporting PDC commands, the software always exports names using the PDC rules described in this section.

2.3. Case Sensitivity When Importing PDC Files (Ask a Question)

The following table shows the case sensitivity in the PDC file based on the source netlist.

Table 2-2. Case Sensitivity in the PDC File

File Type	Case Sensitivity
Verilog	Names in the netlist are case sensitive.
EDIF (SmartFusion® 2, IGLOO® 2, and RTG4™)	Names in the netlist are always case sensitive because you use the Rename clause, which is case sensitive.
VHDL	Names in the netlist are not case sensitive unless those names appear between slashes (\).

For example, in VHDL, capital "A" and lowercase "a" are the same name, but \A\ and \a\ are two different names. However, in a Verilog netlist, an instance named "A10" fails, if spelled as "a10" in the `set_location` command:

`set_location A10` (This command will succeed.)

`set_location a10` (This command will fail.)

3. I/O PDC Commands (Ask a Question)

I/O PDC commands are used to set and reset I/O standards, voltages values, and attributes.

3.1. set_iobank (Ask a Question)

This PDC command sets the input/output supply voltage (vcci) and the input reference voltage (vref) for the specified I/O bank. DDRIO banks have a dedicated vref pin and you do not need to set any pin on these banks. (See the device datasheet to see which banks are of type DDRIO.)

Diff I/Os do not need a vref pin.

```
set_iobank bankname \
[-vcci vcci_voltage]\ [-vref vref_voltage]\ [-fixed value]\
[-vrefpins value]\
[-updateiostd value]\
```

Arguments

The following sections describe the `set_iobank` arguments.

bankname

Specifies the name of the bank. I/O banks are numbered 0 through N (bank0, bank1,...bankN). See the datasheet for your device to determine the number of banks.

-vcci vcci_voltage

Sets the input/output supply voltage. You can enter one of the following values:

Table 3-1. -Vcci Values

Vcci Voltage	Compatible Standards
3.3V	LVTTL, LVCMOS 3.3, PCI 3.3, LVPECL
2.5V	LVCMOS 2.5, SSTL2 (Class I and II), LVDS, BUSLVDS, MLVDS, MINILVDS, RSDS
1.8V	LVCMOS 1.8, LPDDRI, LPDDRII, SSTL18I
1.5V	LVCMOS 1.5, SSTL 1.5 (Class I and II), HSTL (Class I and II)
1.2V	LVCMOS 1.2

-vref vref_voltage

Sets the input reference voltage. You can enter one of the following values:

Table 3-2. -vref Values

Vref Voltage	Compatible Standards
1.25V	SSTL2 (Class I and II)
1.0V	SSTL18 (Class I and II), LPDDR (Class I and II)
0.75V	SSTL15 (Class I and II), HSTL (Class I and Class II)

-fixed Value

Specifies if the I/O technologies (vcci and vccr voltage) assigned to the bank are locked. You can enter one of the following values:

Table 3-3. -fixed Values

Value	Description
yes	The technologies are locked.
no	The technologies are not locked.

-vrefpins value

Specifies, if the I/O technologies (vcci and vccr voltage) assigned to the bank are locked. You can enter one of the following values:

Table 3-4. -vrefpins Values

Value	Description
default	Because the VREF pins are not locked, the I/O Bank Assigner can assign a VREF pin.

Table 3-4. -vrefpins Values (continued)

Value	Description
pinnum	The specified VREF pin(s) are locked if the -fixed option is yes. The I/O Bank Assigner cannot remove locked VREF pins.

-updateiostd Value Specifies, if the I/O technologies (vcci and vccr voltage) assigned to the bank are locked. You can enter one of the following values:

Table 3-5. -updateiostd Values

Value	Description
yes	If there are I/Os placed on the bank, we keep the placement and change the host to one which is compatible with this bank setting. Check the I/O Attributes to see the one used by the tool.
no	If there are I/Os placed and locked on the bank, the command will fail. If they are placed I/Os they will be unplaced.

Exceptions

Any pins assigned to the specified I/O bank that are incompatible with the default technology are unassigned.

Examples

The following example assigns 3.3V to the input/output supply voltage (vcci) and 1.5V to the input reference voltage (vref) for I/O bank 0.

```
set_iobank bank0 -vcci 3.3 -vref 1.5
```

The following example shows that even though you can import a `set_iobank` command with the `-vrefpins` argument set to “default”, the exported PDC file shows the specific default pins instead of "default".

Imported PDC file contains:

```
set_iobank bank3 -vcci 3.3 -vref 1.8 -fixed yes -vrefpins {default}
```

Exported PDC file contains:

```
set_iobank bank3 -vcci 3.3 -vref 1.8 -fixed yes -vrefpins {N3 P8 M8}
```

See Also

- "reset_io"
- "reset_iobank"

3.2. reset_io (Ask a Question)

This PDC command restores all attributes of an I/O macro to its default values. Also, if the port is assigned, it becomes unassigned.

```
reset_io portname -attributes value
```

Arguments

The following sections describe `reset_io` arguments.

portname Specifies the port name of the I/O macro to be reset. You can use the following wild card characters in port names:

Table 3-6. portname Characteristics

Wild Card	What It Does
\	Interprets the next character as a non-special character
?	Matches any single character
*	Matches any string

-attributes Value Preserve or not preserve the I/O attributes during incremental flow. The following table shows the acceptable values for this argument:

Table 3-7. -attributes Values

Value	Description
yes	Unassigns all of the I/O attributes and resets them to their default values.
no	Unassigns only the port.

Exceptions
None.

Examples
Resets the I/O macro "a" to the default I/O attributes and unassigns it.

```
reset_io a
```
Resets all I/O macros beginning with "b" to the default I/O attributes and unassigns them.

```
reset_io b_*
```
Only unassigns port b from its location.

```
reset_io b -attributes no
```

See Also
- "set_io (SmartFusion 2 and IGLOO 2)"
- "set_io (RTG4 only)"

3.3. reset_iobank (Ask a Question)
This PDC command resets an I/O bank's technology to the default technology.

```
reset_iobank bankname
```

Arguments
The following section describes `reset_iobank` arguments.

bankname Specify, if the I/O bank must be reset to the default technology. I/O banks are numbered 0-7 (bank0, bank1, ..., bank7).

Exceptions
Any pins that are assigned to the specified I/O bank but are incompatible with the default technology are unassigned.

Examples
The following example resets the I/O bank 4 to the default technology:

```
reset_iobank bank4
```

See Also
- "set_iobank"

3.4. reserve (Ask a Question)

This PDC command reserves the named pins in the current device package.

```
reserve -pinname "list of package pins"
```

Arguments

The following section describes `reserve` arguments.

-pinname "list of package pins"	Specify the package pin name(s) to reserve. You can reserve one or more pins.

Exceptions

None.

Examples

```
reserve -pinname "F2"
reserve -pinname "F2 B4 B3"
reserve -pinname "124 17"
```

See Also

- "unreserve"

3.5. set_io (SmartFusion 2 and IGLOO 2) (Ask a Question)

This PDC command sets the attributes of an I/O.

You can use the `set_io` command to assign an I/O technology, the I/O attributes, place, or lock the I/O at a given pin location. There are three I/O bank types available in SmartFusion 2 and IGLOO 2; MSIOD, MSIO, and DDRIO.

Important: For the I/O Register combining option, use the `set_ioff` command. See the set_ioff command for more information.

```
set_io portname\
[-iostd value]\
[-pre_emphasis value]\
[-lpe value]\
[-ff_io_state value]\
[-out_drive value]\
[-slew value]\
[-res_pull value]\
[-schmitt_trigger value]\
[-in_delay value]\
[-odt_static value]\
[-odt_imp value]\
[-ff_io_avail value]\
[-pinname package_pin] \
[-fixed value] \
[-out_load value]
```

Arguments

The following section describes `set_io` (SmartFusion 2 and IGLOO 2) arguments. The attributes below are case sensitive.

portname	Specify the portname of the I/O macro.
-iostd Value	Sets the I/O standard for this macro. Choosing a standard allows the software to set other attributes, such as the slew rate and output loading. If the voltage standard used with the I/O is not compatible with other I/Os in the I/O bank, then assigning an I/O standard to a port will invalidate its location and automatically unassign the I/O.

The following table lists the supported I/Os by Bank type.

Table 3-8. -iostd Values

MSIOD	MSIO	DDRIO
—	LVTTL	—
—	LVCMOS33	—
—	PCI	—
—	LVPECL (Input ONLY)	—
—	LVDS33	—
LVCMOS12	LVCMOS12	LVCMOS12
LVCMOS15	LVCMOS15	LVCMOS15
LVCMOS18	LVCMOS18	LVCMOS18
LVCMOS25	LVCMOS25	LVCMOS25
SSTL2I	SSTL2I	SSTL2I (DDR1)
—	STL2II	SSTL2II (DDR1)
SSTL18I	SSTL18I	SSTL18I (DDR2)
—	SSTL18II	SSTL18II (DDR2)
HSTLI	HSTLI	HSTLI
—	—	HSTLII
—	—	SSTL15I (DDR3)
—	—	SSTL15II (DDR3)
—	—	LPDDRI
—	—	LPDDRII
LVDS	LVDS	—
RSDS	RSDS	—
MINILVDS	MINILVDS	—
BUSLVDS (Input ONLY)	BUSLVDS	—
MLVDS (Input ONLY)	MLVDS	—

I/O standards support for single and differential I/Os is shown in the following table.

Table 3-9. I/O Standards for Single and Differential I/Os

Value	Single	Differential	Description
LVTTL	X	—	(Low-Voltage TTL) A general purpose standard (EIA/JESDSA) for 3.3V applications. It uses an LVTTL input buffer and a push-pull output buffer.
LVCMOS33	X	—	(Low-Voltage CMOS for 3.3V) An extension of the LVCMOS standard (JESD8-5) used for general purpose 3.3V applications.
LVCMOS25	X	—	(Low-Voltage CMOS for 2.5V) An extension of the LVCMOS standard (JESD8-5) used for general purpose 2.5V applications.
LVCMOS18	X	—	(Low-Voltage CMOS for 1.8V) An extension of the LVCMOS standard (JESD8-5) used for general purpose 1.8V applications. It uses a 3.3V-tolerant CMOS input buffer and a push-pull output buffer.
LVCMOS15	X	—	(Low-Voltage CMOS for 1.5V) An extension of the LVCMOS standard (JESD8-5) used for general purpose 1.5V applications. It uses a 3.3V-tolerant CMOS input buffer and a push-pull output buffer.

Table 3-9. I/O Standards for Single and Differential I/Os (continued)

Value	Single	Differential	Description
LVCMOS12	X	—	(Low-Voltage CMOS for 1.2V) An extension of the LVCMOS standard (JESD8-5) used for general purpose 1.2V applications.
LVDS	—	X	A moderate-speed differential signaling system, in which the transmitter generates two different voltages which are compared at the receiver. It requires that one data bit be carried through two signal lines; therefore, you need two pins per input or output. It also requires an external resistor termination. The voltage swing between these two signal lines is approximately 350 mV.
LVDS33	—	X	LVDS for 3.3V
BUSLVDS	—	X	2.5V BUSLVDS
MLVDS	—	X	—
MINILVDS	—	X	—
RSDS	—	X	—
LVPECL (only for inputs)	—	X	PECL is another differential I/O standard. It requires that one data bit is carried through two signal lines; therefore, two pins are needed per input or output. It also requires an external resistor termination. The voltage swing between these two signal lines is approximately 850 mV. When the power supply is 3.3V, it is commonly referred to as Low-Voltage PECL (LVPECL).
PCI	—	—	(Peripheral Component Interface) Specifies support for both 33 MHz and 66 MHz PCI bus applications. It uses an LVTTL input buffer and a push-pull output buffer. With the aid of an external resistor, this I/O standard can be 5V compliant for most families.
PCIX	—	—	(Peripheral Component Interface Extended) An enhanced version of the PCI specification that can support higher average bandwidth; it increases the speed that data can move within a computer from 66 MHz to 133 MHz. PCI-X is backward-compatible, which means that devices can operate at conventional PCI frequencies (33 MHz and 66 MHz). PCI-X is also more Fault tolerant than PCI.
HSTLI	X	X	(High-Speed Transceiver Logic) A general purpose, high-speed 1.5V bus standard (EIA/JESD8-6). It has four classes; Microchip® SoC supports Class I and II. It requires a differential amplifier input buffer and a push- pull output buffer.
HSTLII	X	X	(High-Speed Transceiver Logic) A general purpose, high-speed 1.5V bus standard (EIA/JESD8-6). It has four classes; Microchip SoC supports Class I and II. It requires a differential amplifier input buffer and a push-pull output buffer.
SSTL2I	X	X	(Stub Series Terminated Logic for 2.5V) A general purpose 2.5V memory bus standard (JESD8-9). It has two classes; Microchip SoC supports both. It requires a differential amplifier input buffer and a push-pull output buffer.
SSTL2II	X	X	See SSTL2I above.

Table 3-9. I/O Standards for Single and Differential I/Os (continued)

Value	Single	Differential	Description
SSTL15I	X	X	(Stub Series Terminated Logic for 1.5V) A general purpose 1.5V memory bus standard (JESD8-9). It has two classes; Microchip SoC supports both. It requires a differential amplifier input buffer and a push-pull output buffer.
SSTL15II	X	X	See SSTL15I
SSTL18II	X	X	(Stub Series Terminated Logic for 1.8V) A general-purpose 1.8V memory bus standard (JESD8-9). It has two classes; Microchip SoC supports both. It requires a differential amplifier input buffer and a push-pull output buffer.

-pre_emphasis Value

The pre-emphasis rate is the amount of rise or fall time an input signal takes to get from logic low to logic high or vice versa. It is commonly defined to be the propagation delay between 10% and 90% of the signal's voltage swing. Possible values are shown in the following table. The output buffer has a programmable slew rate for both high-to-low and low-to-high transitions. The low rate is incompatible with 3.3V PCI requirements.

Table 3-10. -pre_emphasis Value

Value	Description
NONE	Sets to none (default)
MIN	Sets to minimum
MEDIUM	Sets to medium
MAX	Sets to maximum

-lpe Value

Sets the state at which your device exits from Low Power mode. Possible values are shown in the following table.

Table 3-11. -lpe Value

Value	Description
OFF	Default; no LPE set
Wake_on_Change	Exits from Low Power mode on change
Wake_on_0	Exits from Low Power mode on 0
Wake_on_1	Exits from Low Power mode on 1

-ff_io_state Value

Preserves the previous state of the I/O. By default, all the I/Os become striated, when the device goes into Flash*Freeze mode. (A tristatable I/O is an I/O with three output states: high, low, and high-impedance.) You can override this default using the`FF_IO_STATE` attributes. When you set this attribute to `LAST_VALUE`, the I/O remains in the same state in which it was functioning before the device went into Flash*Freeze mode. Possible values are shown in the following table.

Table 3-12. -ff_io_state Value

Value	Description
TRISTATE	Sets the I/O to tri-state (default).
LAST_VALUE	Preserves the previous state of the I/O.

-out_drive Value

Sets the strength of the output buffer to 2, 4, 6, 8, 10, 12, 16, or 20 in mA, weakest to strongest. The list of I/O standards for which you can change the output drive and the list of values you can assign for each I/O standard is family-specific. Not all I/O standards have a selectable output drive strength. Also, each I/O standard has a different range of legal output drive strength values. The values you can choose from depend on which

I/O standard you have specified for this command. See the Slew and Out_drive Settings table under "Exceptions" in this topic for possible values. The following table lists the acceptable values.

Table 3-13. -out_drive Value

Value	Description
2	Sets the output drive strength to 2 mA
4	Sets the output drive strength to 4 mA
6	Sets the output drive strength to 6 mA
8	Sets the output drive strength to 8 mA
10	Sets the output drive strength to 10 mA
12	Sets the output drive strength to 12 mA
16	Sets the output drive strength to 16 mA
20	Sets the output drive strength to 20 mA

-slew Value

Sets the output slew rate. Slew control affects only the falling edges for some families. Slew control affects both rising and falling edges. Not all I/O standards have a selectable slew. Whether you can use the slew attribute depends on which I/O standard you have specified for this command.

See the Slew and Out_drive Settings table under Exceptions in this topic. The following table lists the acceptable values for the -slew attribute.

Table 3-14. -slew Value

Value	Description
SLOW	Sets the I/O slew to slow.
MEDIUM	Sets the I/O slew to medium.
MEDIUM_FAST	Sets the I/O slew to medium fast.
FAST	Sets the I/O slew to fast.

-res_pull Value

Allows you to include a weak resistor for either pull-up or pull-down of the input buffer or the output buffer. Not all I/O standards have a selectable resistor pull option. The following table shows the acceptable values for the -res_pull attribute.

Table 3-15. -res_pull Value

Value	Description
up	Includes a weak resistor for pull-up of the input buffer.
down	Includes a weak resistor for pull-down of the input buffer.
none	Does not include a weak resistor. This is the default value.

-schmitt_trigger Value

Specifies whether this I/O has an input Schmitt Trigger. The Schmitt Trigger introduces hysteresis on the I/O input. This allows very slow moving or noisy input signals to be used with the part without false or multiple I/O transitions taking place in the I/O. The following table lists the acceptable values for the - schmitt_trigger attribute.

Table 3-16. -schmitt_trigger Value

Value	Description
ON	Turns the Schmitt Trigger ON.
OFF	Turns the Schmitt Trigger OFF.

-in_delay Value

Specifies whether this I/O has an input delay. You can specify an input delay between 0 and 63. The input delay is not a delay value but rather a selection from 0 to 63. The actual value is a function of the operating conditions and is automatically computed

by the delay extractor when a timing report is generated. The following table lists the acceptable values for the -in_delay attribute.

Table 3-17. -in_delay Value

Value	Description
OFF	This I/O does not have an input delay.
0	Sets the input delay to 0.
1	Sets the input delay to 1.
2	Sets the input delay to 2.
...	...
63	Sets the input delay to 63.

-odt_static Value

On-Die Termination (ODT) is the technology where the termination resistor for impedance matching in transmission lines is located inside a semiconductor chip instead of on a printed circuit board. The following table lists the possible values.

Table 3-18. -odt_static Value

Value	Description
ON	Yes, the termination resistor for impedance matching is located inside the chip
OFF	No, the termination resistor is on the printed circuit board

-odt_imp Value

ODT is the technology where the termination resistor for impedance matching in transmission lines is located inside a semiconductor chip instead of on a printed circuit board.

Port Configuration (PC) bits are static configuration bits set during programming to configure the I/O(s) as per your choice. See your device data sheet for a full range of possible values.

-ff_io_avail Value

Indicates the I/O is available in Flash*Freeze mode. The following table lists the possible values.

Table 3-19. -ff_io_avail Value

Value	Description
yes	I/O is available in Flash*Freeze mode.
no	By default, I/O is unavailable in Flash*Freeze mode.

-pinname package_pin

Specifies the package pin name(s) on which to place the I/O.

-fixed Value

Specifies whether the pin is locked or unlocked.

Table 3-20. -fixed Value

Value	Description
yes	The location of this port is locked.
no	The location of this port is unlocked.

-out_load Value

Sets the output load (in pF) of output signals. The default is 5.

Direction: Output

Examples

```
set_io PAD \
-PINNAME A2 \
-FIXED yes \
-FF_IO_STATE LAST_VALUE \
set_io PAD_0 \
```

```
-pinname A8 \
-fixed yes \
-IN_DELAY 6 \
-LPE Wake_On_Change \
-RES_PULL Down \
-SCHMITT_TRIGGER On \
set_io PAD_3 \
-OUT_DRIVE 6 \
-OUT_LOAD 52 \
```

See Also

- "reset_io"

3.6. set_io (RTG4 only) (Ask a Question)

This PDC command sets the attributes of an I/O for RTG4 devices. You can use the `set_io` command to assign an I/O technology, the I/O attributes, place, or lock the I/O at a given pin location. There are three I/O Bank types available in RTG4; MSIOD, MSIO, and DDRIO.

Important: For the I/O Register combining option, use the set_ioff command. See the set_ioff command for more information.

```
set_io portname\
[-direction input | output]\
[-iostd value]\
[-pre_emphasis value]\
[-out_drive value]\
[-out_load value]\
[-slew value]\
[-res_pull value]\
[-schmitt_trigger value]\
[-input_delay value]\
[-odt_static value]\
[-odt_dynamic value]\
[-odt_imp value]\
```

Arguments

The following section describes `set_io` (RTG4 only) arguments.

portname — Specifies the portname of the I/O macro.

-direction Value — Specifies the direction of the I/O ports. Valid values are input, output, and inout.

-iostd Value — Sets the I/O standard for this macro. Choosing a standard allows the software to set other attributes, such as the slew rate and output loading. If the voltage standard used with the I/O is not compatible with other I/Os in the I/O bank, then assigning an I/O standard to a port will invalidate its location and automatically unassign the I/O.

The following table lists a list of supported I/Os by Bank type.

Table 3-21. -iostd Value

MSIOD	MSIO	DDRIO
—	LVTTL	—
—	LVCMOS33	—
—	PCI	—
—	LVPECL (Input ONLY)	—
—	LVDS33	—
LVCMOS12	LVCMOS12	LVCMOS12
LVCMOS15	LVCMOS15	LVCMOS15
LVCMOS18	LVCMOS18	LVCMOS18
LVCMOS25	LVCMOS25	LVCMOS25

Table 3-21. -iostd Value (continued)

MSIOD	MSIO	DDRIO
SSTL2I	SSTL2I	SSTL2I (DDR1)
SSTL2II	STL2II	SSTL2II (DDR1)
SSTL18I	SSTL18I	SSTL18I (DDR2)
SSTL18II	SSTL18II	SSTL18II (DDR2)
HSTLI	HSTLI	HSTLI
—	—	HSTLII
—	—	SSTL15I (DDR3) Only for IOs used by FDDR
—	—	SSTL15II (DDR3) Only for I/OS used by FDDR
—	—	LPDDRI
—	—	LPDDRII
LVDS	LVDS	—
RSDS	RSDS	—
MINILVDS	MINILVDS	—
BUSLVDS (Input ONLY)	BUSLVDS	—
MLVDS (Input ONLY)	MLVDS	—

I/O standards support for single and differential I/Os is shown in the following table.

Table 3-22. I/O Standards for Single and Differential I/Os

Value	Single	Differential	Description
LVTTL	X	—	(Low-Voltage TTL) A general purpose standard (EIA/JESDSA) for 3.3V applications. It uses an LVTTL input buffer and a push-pull output buffer.
LVCMOS33	X	—	(Low-Voltage CMOS for 3.3V) An extension of the LVCMOS standard (JESD8-5) used for general purpose 3.3V applications.
LVCMOS25	X	—	(Low-Voltage CMOS for 2.5V) An extension of the LVCMOS standard (JESD8-5) used for general purpose 2.5V applications.
LVCMOS18	X	—	(Low-Voltage CMOS for 1.8V) An extension of the LVCMOS standard (JESD8-5) used for general purpose 1.8V applications. It uses a 3.3V-tolerant CMOS input buffer and a push-pull output buffer.
LVCMOS15	X	—	(Low-Voltage CMOS for 1.5V) An extension of the LVCMOS standard (JESD8-5) used for general purpose 1.5V applications. It uses a 3.3V-tolerant CMOS input buffer and a push-pull output buffer.
LVCMOS12	X	—	(Low-Voltage CMOS for 1.2V) An extension of the LVCMOS standard (JESD8-5) used for general purpose 1.2V applications. This I/O standard is supported only in ProASIC®3L and the IGLOO® family of devices.
LVDS	—	X	A moderate-speed differential signaling system, in which the transmitter generates two different voltages which are compared at the receiver. It requires that one data bit be carried through two signal lines. Therefore, you need two pins per input or output. It also requires an external resistor termination. The voltage swing between these two signal lines is approximately 350 mV.
LVDS33	—	X	LVDS for 3.3V

Table 3-22. I/O Standards for Single and Differential I/Os (continued)

Value	Single	Differential	Description
BUSLVDS	—	X	2.5V BUSLVDS
MLVDS	—	X	—
MINILVDS	—	X	—
RSDS	—	X	—
LVPECL (only for inputs)	—	X	PECL is another differential I/O standard. It requires that one data bit is carried through two signal lines; therefore, two pins are needed per input or output. It also requires an external resistor termination. The voltage swing between these two signal lines is approximately 850 mV. When the power supply is 3.3V, it is commonly referred to as LVPECL.
HSTLI	X	X	High-Speed Transceiver Logic Class I. A general-purpose, high-speed 1.5V bus standard (EIA/JESD 8-6). It has four classes; Microchip® SoC supports Class I and Class II. It requires a differential amplifier input buffer and a push-pull output buffer.
HSTLII	X	X	High-Speed Transceiver Logic Class II. A general-purpose, high-speed 1.5V bus standard (EIA/JESD8-6). It has four classes; Microchip SoC supports Class I and Class II. It requires a differential amplifier input buffer and a push-pull output buffer.
HSTL18I	X	X	High-Speed Transceiver Logic 1.8V Class I. A general-purpose, high-speed 1.8V bus. It has four classes; Microchip SoC supports Class I and Class II. It requires a differential amplifier input buffer and a push-pull output buffer.
HSTL18II	X	X	High-Speed Transceiver Logic 1.8V Class II. A general-purpose, high-speed 1.8V bus. It has four classes; Microchip SoC supports Class I and Class II. It requires a differential amplifier input buffer and a push-pull output buffer.
SSTL2I	X	X	(Stub Series Terminated Logic for 2.5V) A general-purpose 2.5V memory bus standard (JESD8-9). It has two classes: Class I and Class II; Microchip SoC supports both. It requires a differential amplifier input buffer and a push-pull output buffer.
SSTL2II	X	X	See SSTL2I above.
SSTL15I	X	X	Stub Series Terminated Logic for 1.5V Class I. A general purpose 1.5V memory bus standard (JESD8-9). It has two classes: Class I and Class II; Microchip supports both. It requires a differential amplifier input buffer and a push-pull output buffer.
SSTL15II	X	X	Stub Series Terminated Logic for 1.5V Class II. See SSTL15I above.
SSTL18I	X	X	Stub Series Terminated Logic for 1.8V Class I. A general purpose 1.8V memory bus standard (JESD8-9). It has two classes; Microchip SoC supports both. It requires a differential amplifier input buffer and a push- pull output buffer.
SSTL18II	X	X	Stub Series Terminated Logic for 1.8V Class II. A general purpose 1.8V memory bus standard (JESD8-9). It has two classes; Microchip SoC supports both. It requires a differential amplifier input buffer and a push- pull output buffer.
LPDDRI	X	X	—

Table 3-22. I/O Standards for Single and Differential I/Os (continued)

Value	Single	Differential	Description
LPDDRII	X	X	—

-pre_emphasis Value

The pre-emphasis rate is the amount of rise or fall time an input signal takes to get from logic low to logic high or vice versa. It is commonly defined to be the propagation delay between 10% and 90% of the signal's voltage swing. Possible values are shown in the following table. The output buffer has a programmable slew rate for both high-to-low and low-to-high transitions.

Table 3-23. -pre_emphasis Value

Value	Description	Applicable to I/O Standards
NONE	Sets to none (default)	LVDS, RSDS
MIN	Sets to minimum	LVDS, RSDS
MEDIUM	Sets to medium	RSDS only
MAX	Sets to maximum	LVDS, RSDS

-out_drive Value

Sets the strength of the output buffer to 2, 4, 6, 8, 10, 12, 16, or 20 in mA, weakest to strongest. The list of I/O standards for which you can change the output drive and the list of values you can assign for each I/O standard is family-specific and I/O Bank Type -specific. Not all I/O standards have a selectable output drive strength. Also, each I/O standard has a different range of legal output drive strength values. The values you can choose from depend on which I/O standard you have specified for this command. The following table lists the acceptable values.

Table 3-24. -out_drive Value

Value (mA)	Description
2	Sets the output drive strength to 2 mA
4	Sets the output drive strength to 4 mA
6	Sets the output drive strength to 6 mA
8	Sets the output drive strength to 8 mA
10	Sets the output drive strength to 10 mA
12	Sets the output drive strength to 12 mA
16	Sets the output drive strength to 16 mA
20	Sets the output drive strength to 20 mA

Table 3-25. I/O Standards

I/O Standard	User -set Valid Output Drive Values (mA) Per I/O Bank Type			Valid Output Drive Value for Die
LVTTL	MSIO	MSIOD	DDRIO	—
	2	—	—	2
	4	—	—	4
	8	—	—	8
	12	—	—	12
	16	—	—	16
LVCMOS33	2	—	—	2
	4	—	—	4
	8	—	—	8
	12	—	—	12
	16	—	—	16

Table 3-25. I/O Standards (continued)

I/O Standard	User -set Valid Output Drive Values (mA) Per I/O Bank Type			Valid Output Drive Value for Die
LVCMOS12	2	2	2	2
	4	4	4	4
	—	6	6	6
LVCMOS15	2	2	2	2
	4	4	4	4
	6	6	6	6
	8	—	8	8
	—	—	10	10
	—	—	12	12
LVCMOS18	2	2	2	2
	4	4	4	4
	6	6	6	6
	8	8	8	8
	10	—	10	10
	12	—	12	12
	—	—	16	16

-out_load Value

Sets the output load (in pF) of output signals.

-slew Value

Sets the output slew rate. Slew control affects only the falling edges for some families. Slew control affects both rising and falling edges. Not all I/O standards have a selectable slew. Whether you can use the slew attribute depends on which I/O standard you have specified for this command.

The following table lists the acceptable values for the -slew attribute.

Table 3-26. -slew Value

Value	Description	I/O Standard	I/O Bank Type
SLOW	Sets the I/O slew to slow	LVCMOS12, LVCMOS15, LVCMOS18	MSIO, MSIOD, DDRIO
MEDIUM	Sets the I/O slew to medium	LVCMOS12, LVCMOS15, LVCMOS18	DDRIO
FAST	Sets the I/O slew to fast	LVCMOS12, LVCMOS15	DDRIO

-res_pull Value

Allows you to include a weak resistor for either pull-up or pull-down of the input buffer or the output buffer. Not all I/O standards have a selectable resistor pull option. The following table shows the acceptable values for the -res_pull attribute for different I/O Standard and I/O Bank combinations.

Table 3-27. -res_pull Value

Value	I/O Standard	I/O Bank Type	Description
up	LVTTL, LVCMOS33 PCI	MSIO	Includes a weak resistor for pull-up of the input buffer
	LVCMOS12, LVCMOS15, LVCMOS18, LVCMOS25	MSIO/MSIOD/ DDRIO	—
down	LVTTL, LVCMOS33 PCI	MSIO	Includes a weak resistor for pull-down of the input buffer
	LVCMOS12, LVCMOS15, LVCMOS18, LVCMOS25	MSIO/MSIOD/ DDRIO	—

Table 3-27. -res_pull Value (continued)

Value	I/O Standard	I/O Bank Type	Description
none	LVTTL, LVCMOS33 PCI	MSIO	Does not include a weak resistor (Default value)
	LVCMOS12, LVCMOS15, LVCMOS18, LVCMOS25	MSIO/MSIOD/ DDRIO	—

-schmitt_trigger Value

Specifies whether this I/O has an input Schmitt Trigger. The Schmitt Trigger introduces hysteresis on the I/O input. This allows very slow moving or noisy input signals to be used with the part without false or multiple I/O transitions taking place in the I/O. The following table shows the acceptable values for the -schmitt_trigger attribute.

Table 3-28. -schmitt_trigger Value

Value	Description
ON	Turns the Schmitt Trigger ON.
OFF	Turns the Schmitt Trigger OFF (Default value).

The applicable valid values are dependent on the I/O Standard and the I/O Bank Type.

Table 3-29. I/O Standard and the I/O Bank Type

I/O Standard	I/O Bank Type		
LVTTL	MSIO	MSIOD	DDRIO
	OFF/ON	N/A	N/A
LVCMOS33	OFF/ON	N/A	N/A
PCI	OFF/ON	N/A	N/A
LVCMOS12	OFF/ON	OFF/ON	OFF/ON
LVCMOS15	OFF/ON	OFF/ON	OFF/ON
LVCMOS18	OFF/ON	OFF/ON	OFF/ON
LVCMOS25	OFF/ON	OFF/ON	OFF/ON

-input_delay Value

Specifies whether this I/O has an input delay. You can specify an input delay between 0 and 63. The input delay is not an absolute delay value but rather a selection from 0 to 63. The actual value is a function of the operating conditions and is automatically computed by the delay extractor when a timing report is generated. The following table shows the acceptable values for the -input_delay attribute.

Table 3-30. -input_delay Value

Value	Description
OFF	This I/O does not have an input delay (Default value)
0	Sets the input delay to 0
1	Sets the input delay to 1
2	Sets the input delay to 2
...	...
63	Sets the input delay to 63

-odt_static Value

ODT is the technology where the termination resistor for impedance matching in transmission lines is located inside a semiconductor chip instead of on a printed circuit board. Possible value are listed in the following table.

Note: ODT is not allowed for 2.5V or higher single-ended signals. It is allowed for differential signals.

Table 3-31. -odt_static Value

Value	Description
ON	Yes, the termination resistor for impedance matching is located inside the chip.
OFF	No, the termination resistor is on the printed circuit board (Default value).

The valid value for each I/O Standard and I/O Bank Type combination is listed in the following table.

Table 3-32. I/O Standard and I/O Bank Type

I/O Standard	I/O Bank Type		
	MSIO	MSIOD	DDRIO
LVPECL	OFF/ON	N/A	N/A
LVDS33	OFF/ON	N/A	N/A
SSTL18I (DDR2)	OFF/ON	OFF/ON	OFF/ON
SSTL18II (DDR2)	OFF/ON	OFF/ON	OFF/ON
HSTL18I	OFF/ON	OFF/ON	OFF/ON
HSTL18II	N/A	N/A	OFF/ON
HSTLI	OFF	OFF	OFF/ON
HSTLII	OFF	OFF	OFF/ON
SSTL15I (DDR3)	N/A	N/A	OFF/ON
SSTL15II (DDR3)	N/A	N/A	OFF/ON
LPDDRI	N/A	N/A	OFF/ON
LPDDRII	N/A	N/A	OFF/ON
LVDS	OFF/ON	OFF/ON	N/A
RSDS	OFF/ON	OFF/ON	N/A
MINILVDS	OFF/ON	OFF/ON	N/A
BUSLVDS	OFF/ON	OFF/ON	N/A
MLVDS	OFF/ON	OFF/ON	N/A

-odt_dynamic Value

This option is not supported for “ES” devices. It is supported only for production devices. This option is used to opt in or out of the dynamic odt set on a bank. Possible value are listed in the table below.

Value	Description
ODT_STATIC = ON ODT_DYNAMIC = ON	Illegal
ODT_STATIC = ON ODT_DYNAMIC = OFF	The ODT resistor is always turned ON.
ODT_STATIC = OFF ODT_DYNAMIC = OFF	The ODT resistor is always turned OFF.
ODT_STATIC = OFF ODT_DYNAMIC = ON	The ODT resistor is ON or OFF based on the ODT Dynamic bank setting.

The following I/O standards are supported:

LVDS, RSDS, MINILVDS, LVPECL, HSTLI, HSTLII, SSTL15I, SSTL15II, SSTL18I, SSTL18II, HSTL18I, HSTL18II, LPDDRI, LPDDRII

-odt_imp Value

ODT is the technology where the termination resistor for impedance matching in transmission lines is located inside a semiconductor chip instead of on a printed circuit board. The valid value for each I/O Standard and I/O Bank type is listed in the table

below. When the value for an I/O standard is not listed, the impedance value is fixed for the specific I/O standard and is not user- selectable.

Table 3-33. -odt_imp Value

I/O Standard	I/O Bank Type		
	MSIO	MSIOD	DDRIO
SSTL18I (DDR2)	50 75 150	50 75 150	50 75 150
SSTL18II (DDR2)	50 75 150	50 75 150	50 75 150
HSTL18I	50 75 150	50 75 150	50 75 150
HSTL18II	—	—	50 75 150
LPDDRI	—	—	50 75 150
LPDDRII	—	—	50 75 150
SSTL15I (DDR3)	—	—	20 30 40 60 120
SSTL15II (DDR3)	—	—	20 30 40 60 120

PC bits are static configuration bits set during programming to configure the IO(s) as per your choice. See your device data sheet for a full range of possible values.

Examples

```
set_io IO_in\[2\] -iostd LVCMOS25 \
-slew slow \
-schmitt_trigger off \
-input_delay off \
```

See Also

- "reset_io"

3.7. unreserve (Ask a Question)

This PDC command resets the named pins in the current device. Therefore, they are no longer reserved. You can use these pins in your design.

```
unreserve -pinname "list of package pins"
```

Arguments

The following section describes `unreserve` arguments.

-pinname "list of package pins"	Specifies the package pin name(s) to unreserve.

Exceptions

None.

Examples

```
unreserve -pinname "F2"
unreserve -pinname "F2 B4 B3"
unreserve -pinname "124 63"
```

See Also

- "reserve"

4. Netlist Attributes NDC Commands (Ask a Question)

To set netlist-specific constraints, use Netlist attributes Netlist Design Constraint (NDC) commands. These commands are placed in a Compile Netlist Constraint (`*.ndc`) file and used by the Libero SoC Compile engine to optimize the post-synthesis netlist.

4.1. set_mitigation (RTG4 Devices Only) (Ask a Question)

This command sets the mitigation option on a per-instance basis for RTG4 devices. For the Enhanced Constraint flow, import this NDC command as a Netlist Attributes constraint file (`*.ndc`) and associate it with Synthesis in the **Constraint Manager**.

For the Classic Constraint Flow, save the command as a PDC file (`*.pdc`) and import the `*.pdc` file into the Project (**Design Flow Window** > **Floorplan Constraints** > **Import Files**). After import, associate the file with Compile (Right-click the PDC file and select **Use for Compile**).

Important: This NDC/PDC command overrides the project-wide global setting for mitigation option (**Project Settings** > **Device settings** > **Enable Single Event Transient Mitigation**). The global mitigation setting is project-wide and applies to ALL instances in the design that the mitigation option is valid. If you want to override the global setting for certain specific instances in the design (and allow the global setting to remain valid on all other instances), use the set_mitigation command for the specific instances because this command sets the mitigation option on a per-instance basis.

```
set_mitigation -inst_name <instance_name> -mitigated <value>
```

Arguments

The following sections describe `set_mitigation` (RTG4 Devices Only) arguments.

-inst_name <instance_name>
Specifies the name of the instance in the netlist to set the mitigation option. A hierarchical instance name is allowed. The wildcard character "*" in the instance name is also supported. When the mitigation option is set on an instance that contains design elements that can have the mitigation option, all the design elements within that instance are set with this option. This applies to MACC blocks, µSRAM blocks, LSRAM blocks, I/O Flip-Flops, and regular Flip-Flops within that instance.

-mitigated Value
Sets the mitigation value for the instance. The acceptable values for this argument are listed in the following table.

Table 4-1. -mitigated Value

Value	Description
Yes | ON | true | 1	The mitigation option is enabled on this instance. “Yes”, “ON”, “True” are case-insensitive.
No | OFF | false | 0	The mitigation option is disabled on this instance. No, OFF, and False are case-insensitive.

Return Value

Returns "0" on success and "1" on failure.

Examples

This example sets the mitigation option on the instance blk1/* globally in the netlist:

```
set_mitigation -inst_name blk1/* -mitigated yes
```

After the first command above, to set a different mitigation value on a specific instance, for example, instance FF_1 inside the blk1 instance, use the <-mitigated No> argument as follows:

```
set_mitigation -inst_name blk1/FF_1 -mitigated No
```

Note: The last `set_mitigation` command overrides a previous set_mitigation command, if there is a conflict.

4.2. set_ioff (Ask a Question)

This command specifies whether or not a register is combined with an I/O during synthesis/compile. This command is placed in a Compile Netlist Constraint (`*.ndc`) file and passed to the Libero SoC Compile engine for netlist optimization after synthesis.

```
set_ioff {<portname>}
\[-in_reg yes|no] \
[-out_reg yes|no] \
[-en_reg yes|no]
```

Arguments

The following sections describe `set_ioff` arguments.

portname	Specifies the name of the I/O port to be combined with a register. The port can be an input, output, or inout port.
-in_reg	Specifies whether the input register is combined into the port <portname>. Valid values are “yes” or “no”.
-out_reg	Specifies whether the output register is combined into the port <portname>. Valid values are “yes” or “no”.
-en_reg	Specifies whether the enable register is combined into the port <portname>. Valid values are “yes” or “no”.

Return Value

The command returns “0” on success and “1” on failure.

I/O Register Combining Rules

I/Os are combined with a register to achieve better clock-to-out and input-to-clock timing. When combining these registers at the I/O buffer, some design rules must be met. This feature is supported by all I/O standards. The I/O register combining rules are as follows:

- Registers combined on the Output and Output Enable must have the same configuration. Example: If the output register is DFN1C0, the Output Enable register must also be DFN1C0.
- All registers (Input, Output, and Output Enable) connected to an I/O must share the same clear (CLR) or preset (PRE) pin
- If one of the registers has a CLR pin, all the other registers that are candidates for combining in the I/O must have a CLR pin
- If one of the registers has a PRE pin, all the other registers that are candidates for combining in the I/O must have a PRE pin
- If one of the registers has neither a CLR nor a PRE pin, all the other registers that are candidates for combining must have neither a CLR nor a PRE pin
- If the CLR or PRE pins are present, they must have the same polarity
- If the CLR or PRE pins are present, they must be driven by the same signal (net)
- The fan-out between an I/O pin (D, Y, or E) and a register must be equal to 1
- The register pin connected to the I/O must be the 'D' or 'Q' pin

- Registers connected to an I/O on the Output and Output Enable pins must have the same clock and enable function
- Both the Output and Output Enable registers must have an E pin (clock enable) or none at all
- If the E pins are present, they must have the same polarity. The CLK pins must also have the same polarity
- Input Registers—The I/O must drive the D pin of a register with a fanout of 1
- Output Registers—The Q pin of the register must drive the D pin of the I/O with a fanout of 1
- Enable Registers—The Q pin of the register must drive the E pin of the I/O with a fanout of 1
- Output and Enable Register CLK pins must be driven by the same net with the same polarity to be combined in the same I/O
- Input, Output and Enable register ALn and SLn must be driven by the same net and the same polarity to combine them into the same I/O

Examples

The following command specifies that for the port my_in_out[1], the output register is combined into the port, but it is not combined into the input register nor the enable register:

```
set_ioff {my_in_out[1]} -in_reg no -out_reg yes -en_reg no
```

The following command specifies that for the port my_in_out[19], the enable register is combined into the port, but is not combined into the input register nor the output register:

```
set_ioff {my_in_out[19]} -in_reg no -out_reg no -en_reg yes
```

The set_ioff command applies to scalar I/Os only. For an I/O bus, use the for-loop available in Tcl. The following command combines each scalar member of the 32-bit I/O bus DataA with input registers:

```
for { set i 0 } { i < 32 } { incr i } { set_ioff "DataA\[$i\]" -in_reg yes }
```

Alternatively, you can use a wild card to include all scalar signals of an I/O bus:

```
set_ioff {DataA[*]} -in_reg yes
```

4.3. set_preserve (Ask a Question)

This command sets a preserve property on instances before compile. Therefore, the compile preserves these instances and does not combine them.

```
set_preserve <hier_inst_name>
```

Arguments

The following sections describe `set_preserve` arguments.

hier_inst_name Specifies the full hierarchical name of the macro in the netlist to preserve.

Exceptions

This constraint must be defined in the NDC file and associated with Synthesis.

Examples

In some cases, you might be required to preserve some instances for timing purposes. For example, you might combine registers with input of a bibuf and keep the output unchanged.

If the outbuf of a bidirectional signal test[1] must be preserved while inbuf is required to combine with the registers, use the following PDC commands:

```
set_io test\[1\] -register yes set_preserve test\[31\]
```

If any internal instance is required to be preserved, use the `set_preserve` command as shown in the following example:

```
set_preserve top/inst1 top/inst2
```

5. Floorplanning FDC Commands (Ask a Question)

This Floorplanning (FDC) commands are used to create and edit user regions and to assign/unassign logic to these regions.

5.1. assign_region (Ask a Question)

PDC command; constrains a set of macros to a specified region.

```
assign_region region_name [ macro_name]+
```

Arguments

The following sections describe `assign_region` arguments.

region_name Specifies the region to which the macros are assigned. The macros are constrained to this region. Because the define_region command returns a region object, you can write a simpler command such as `assign_region [define_region]+ [macro_name]+`.

macro_name Specifies the macro(s) to assign to the region. You must specify at least one macro name. You can use the following wild card characters in macro names.

Table 5-1. Wild Card Characters in Macro Names

Wild Card	What it Does
\	Interprets the next character as a non-special character
?	Matches any single character
*	Matches any string

Exceptions

- The region must be created before you can assign macros to it. If the region creation PDC command and the macro assignment command are in different PDC files, the order of the PDC files is important.
- You can assign only hard macros or their instances to a region. You cannot assign a group name. A hard macro is a logic cell consisting of one or more silicon modules with locked relative placement.
- You can assign a collection of macros by providing a prefix to their names.

Examples

In the following example, two macros are assigned to a region:

```
assign_region cluster_region1 des01/G_2722_0_and2 des01/data1_53/U0
```

In the following example, all macros whose names have the prefix des01/Counter_1 (or all macros whose names match the expression des01/Counter_1/*) are assigned to a region:

```
assign_region User_region2 des01/Counter_1/*
```

See Also

- "unassign_macro_from_region"

5.2. assign_net_macros (Ask a Question)

This PDC command assigns to a user-defined region all the macros that are connected to a net.

```
assign_net_macros region_name [net1]+ [-include_driver value]
```

Arguments

The following sections describe `assign_net_macros` arguments.

region_name

Specifies the name of the region to which you are assigning macros. The region must exist before you use this command. See define_region (rectangular) or define_region (rectilinear). Because the define_region command returns a region object, you can write a simple command such as `assign_net_macros [define_region]+ [net]+`.

net1

You must specify at least one net name. Net names are AFL-level (flattened netlist) names. These names match your netlist names most of the time. When they do not, you must export AFL and use the AFL names. Net names are case insensitive. Hierarchical net names from ADL are not allowed. You can use the following wild card characters in net names:net1.

Table 5-2. net1

Wild Card	What it Does
\	Interprets the next character as a non-special character
?	Matches any single character
*	Matches any string

-include_driver

Specifies whether to add the driver of the net(s) to the region. You can enter one of the following values.

Table 5-3. -include_driver Value

Value	Descriptions
Yes	Include the driver in the list of macros assigned to the region (default).
No	Do not assign the driver to the region.

Exceptions

- Placed macros (not connected to the net) that are inside the area occupied by the net region are automatically unplaced
- Net region constraints are internally converted into constraints on macros. PDC export results as a series of assign_region <region_name> macro1 statements for all the connected macros.
- If the region does not have enough space for all of the macros, or if the region constraint is impossible, the constraint is rejected and a warning message appears in the **Log** window
- For overlapping regions, the intersection must be at least as big as the overlapping macro count
- If a macro on the net cannot legally be placed in the region, it is not placed and a warning message appears in the **Log** window
- Net region constraints may result in a single macro being assigned to multiple regions. These net region constraints result in constraining the macro to the intersection of all the regions affected by the constraint.

Examples

```
assign_net_macros cluster_region1 keyin1intZ0Z_62 -include_driver no
```

See Also

- "unassign_net_macros"

5.3. define_region (Ask a Question)

This PDC command defines either a rectangular region or a rectilinear region.

```
define_region [-name region_name ] -type region_type [x1 y1 x2 y2]+ [-color value]\ [-route
value] [-push_place value]
```

Arguments

The following sections describe `define_region` arguments.

-name region_name

Specifies the region name. The name must be unique. Do not use reserved names such as "bank0" and "bank<N>" for region names. If the region cannot be created, the name is empty. A default name is generated if a name is not specified in this argument.

-type region_type

Specifies the region type. The default is inclusive. The following table lists the acceptable values for this argument.

Table 5-4. -type region_type

Region Type	Description
Empty	Empty regions cannot contain macros.
Exclusive	Only contains macros assigned to the region.
Inclusive	Can contain macros both assigned and unassigned to the region.

x1, y1, x2, and y2

Specifies the series of coordinate pairs that constitute the region. These rectangles may or may not overlap. They are given as x1, y1, x2 and y2 where x1, y1 is the lower left and x2, y2 is the upper right corner in row/column coordinates. You must specify at least one set of coordinates.

-color Value

Specifies the color of the region.

Important: This option is deprecated.

-route Value

Specifies whether the routing of all nets internal to a region that is constrained within that region. A net is internal to a region if its source and destination pins are assigned to the region. You can enter one of the following values.

Table 5-5. -route Value

Constrain Routing Value	Description
Yes	Constrain the routing of nets within the region as well as the placement.
No	Do not constrain the routing of nets within the region. Only constrain the placement. This is the default value.

Important: Local clocks and global clocks are excluded from the `-route` option. Also, interface nets are excluded from the `-route` option because they cross region boundaries.

An empty routing region is an empty placement region. If -route is "yes", then no routing is allowed inside the empty region. However, local clocks and globals can cross empty regions. An exclusive routing region is an exclusive placement region (rectilinear area with assigned macros) along with the following additional constraints:

- For all nets internal to the region (the source and all destinations belong to the region), routing must be inside the region (that is, such nets cannot be assigned any routing resource which is outside the region or crosses the region boundaries)
- Nets without pins inside the region cannot be assigned any routing resource which is inside the region or crosses any region boundaries

An inclusive routing region is an inclusive placement region (rectilinear area with assigned macros) along with the following additional constraints:

- For all nets internal to the region (the source and all destinations belong to the region), routing must be inside the region (that is, such nets cannot be assigned any routing resource which is outside the region or crosses the region boundaries)
- Nets not internal to the region can be assigned routing resources within the region.

-push_place

Specifies to prevent the placer from positioning macros at the periphery of the region, thereby ensuring that routing remains within the region (that is, routing between macros assigned to the region) and preventing routing failures.

Important:

- This option is supported only for the SmartFusion 2 and IGLOO 2 family of devices. This is due to their use of routing switch primitive modeling, as opposed to the RTG4, which utilizes mux-based modeling.
- This option is applicable when the `-route` option is enabled. If `-route` is not set or is set to false, the `-push_place` option will have no effect. By default, when `-route` is enabled, `-push_place` is internally set to true to ensure that the router confines the routing within the specified region.

Description

Unlocked macros in empty or exclusive regions are unassigned from that region. You cannot create empty regions in areas that contain locked macros. Use inclusive or exclusive region constraints, if you intend to assign logic to a region. An inclusive region constraint with no macros assigned to it has no effect. An exclusive region constraint with no macros assigned to it is equivalent to an empty region.

Exceptions

If macros assigned to a region exceed the area's capacity, the region's **Properties** window displays the overbooked resources (over 100% resource usage) in red.

Examples

The following example defines an empty rectangular region.

```
define_region -name cluster_region1 -type empty 100 46 102 46
```

The following example defines a rectilinear region with the name RecRegion. This region contains two rectangular areas.

```
define_region -name RecRegion -type Exclusive 0 40 3 42 0 77 7 79
```

The following examples define three regions with three different colors:

```
define_region -name UserRegion0 -color 128 50 19 60 25
define_region -name UserRegion1 -color 16711935 11 2 55 29
define_region -name UserRegion2 -color 8388736 61 6 69 19
```

See Also

- "assign_region"

5.4. undefine_region (Ask a Question)

This PDC command removes the specified region. All macros assigned to the region are unassigned.

```
undefine_region region_name
```

Arguments

The following sections describe `undefine_region` arguments.

region_name Specifies the region to be removed.

Exceptions

To use this command, the region must be previously defined.

Examples

```
undefine_region cluster_region1
```

See Also

- "define_region"

5.5. move_region (Ask a Question)

This PDC command moves the named region to the coordinates specified.

```
move_region region_name [x1 y1 x2 y2]+
```

Arguments

The following sections describe `move_region` arguments.

region_name Specifies the name of the region to move. This name must be unique.

x1 y1 x2 y2 Specifies the series of coordinate pairs representing the location in which to move the named region. These rectangles can overlap. They are given as x1 y1 x2 y2, where x1, y1 represents the lower-left corner of the rectangle and x2 y2 represents the upper-right corner. You must specify at least one set of coordinates.

Exceptions

None.

Examples

This example moves the region named RecRegion to a new region which is made up of two rectangular areas:

```
move_region RecRegion 0 40 3 42 0 77 7 79
```

See Also

- "move_region"

5.6. unassign_net_macros (Ask a Question)

This PDC command unassigns macros connected to a specified net.

```
unassign_net_macros region_name [net1]+
```

Arguments

The following sections describe `unassign_net_macros` arguments.

region_name Specifies the name of the region containing the macros in the net(s) to unassign.

net1 Specifies the name of the net(s) that contain the macros to unassign from the specified region. You must specify at least one net name. Optionally, you can specify additional nets to unassign.

Exceptions

If the region is currently not assigned, an error message appears in the **Log** window, if you try to unassign it.

Examples

```
unassign_net_macros cluster_region1 keyin1intZ0Z_62
```

See Also

- "unassign_macro_from_region"
- "assign_net_macros"

5.7. unassign_macro_from_region (Ask a Question)

This PDC command specifies the name of the macro to be unassigned.

```
unassign_macro_from_region [region_name] macro_name
```

Arguments

The following sections describe `unassign_macro_from_region` arguments.

region_name Specifies the region where the macro or macros are to be removed.

macro_name Specifies the macro to be unassigned from the region. Macro names are case sensitive. You can unassign a collection of macros by assigning a prefix to their names. You cannot use hierarchical net names from ADL. However, you can use the following wild card characters in macro names.

Table 5-6. Wild Card Characters in Macro Names

Wild card	What It Does
\	Interprets the next character as a non-special character
?	Matches any single character
*	Matches any string

Exceptions

If the macro was not previously assigned, an error message is generated.

Examples

```
unassign_macro_from_region macro21
```

See Also

- "unassign_macro_from_region"
- "assign_net_macros"

5.8. set_location (Ask a Question)

This PDC command assigns the specified macro to a particular location on the chip.

```
set_location macro_name -fixed value x y
```

Note: This command may not honour placing globals to physical locations on the die. Instead of placing globals on die locations, let the Libero Design Suite decide where to place the global buffers.

Arguments

The following sections describe `set_location` arguments.

macro_name Specifies the name of the macro in the netlist to assign to a particular location on the chip.

-fixed Value Sets whether the location of this instance is fixed (that is, locked). Locked instances are not moved during layout. The default is yes. The following table shows the acceptable values for this argument.

Table 5-7. -fixed Value

Value	Description
yes	The location of this instance is locked.
no	The location of this instance is unlocked.

x y The x and y coordinates specify where to place the macro on the chip. Use the Chip Planner tool to determine the x and y coordinates of the location.

Exceptions

None.

Examples

This example assigns and locks the macro with the name "`mem_data_in\[57\]`" at the location x=7, y=2:

```
set_location mem_data_in\[57\] -fixed yes 7 2
```

5.9. move_block (Ask a Question)

This PDC command moves a Block from its original, locked placement by preserving the relative placement between the instances. You can move the Block to the left, right, up, or down.

Note: If possible, routing is preserved when you move the blocks.

```
move_block -inst_name instance_name -up y -down y -left x -right x -non_logic value
```

Arguments

The following sections describe `move_block` arguments.

-inst_name instance_name Specifies the name of the instance to move. Refer to the Logical Veiw of Chip Planner for the instance name to use.

-up y Moves the block up the specified number of rows. The value must be a positive integer.

-down y Moves the block down the specified number of rows. The value must be a positive integer.

-left x Moves the block left the specified number of columns. The value must be a positive integer.

-right x Moves the block right the specified number of columns. The value must be a positive integer.

-non_logic Value Specifies what to do with the non-logic part of the block, if one exists. The following table shows the acceptable values for this argument.

Table 5-8. -non_logic Value

Value	Description
move	Move the entire block.
keep	Move only the logic portion of the block (COMB/SEQ) and keep the rest locked in the same previous location, if there is no conflict with other blocks.
unplace	Move only the logic portion of the block (COMB/SEQ) and unplace the rest of it, such as I/Os and RAM.

Description

This command moves a block from its original, locked position to a new position. You can move the entire block or just the logic part of it. You must use the -non_logic argument to specify what to do with the non-logic part of the block.

The -up, -down, -left, and -right arguments enable you to specify how to move the block from its original placement. You cannot rotate the block, but the relative placement of macros within the block will be preserved and the placement will be locked. However, routing will be lost. You can either use the Chip Planner tool or run a Block report to determine the location of the block.

The -non_logic argument enables you to move a block that includes non-logic instances, such as RAM or I/Os that are difficult to move. Once you have moved a part of a block, you can unplace the remaining parts of the block and then place them manually as necessary.

Note: Microchip recommends that you move the block left or right by increments of 12. If not, placement may fail because it violates clustering constraints. Also, Microchip recommends that you move the block up or down by increments of three.

Exceptions

- You must associate this PDC constraint file to Place and Route
- You must use this PDC command if you want to preserve the relative placement and routing (if possible) of a block you are instantiating many times in your design. Only one instance will be preserved by default. To preserve other instances, you must move them using this command.

Examples

The following example moves the entire block (instance name instA) 12 columns to the right and 3 rows up:

```
move_block -inst_name instA -right 12 -up 3 -non_logic move
```

The following example moves only the logic portion of the block and unplaces the rest by 24 columns to the right and 6 rows up.

```
move_block -inst_name instA -right 24 -up 6 -non_logic unplace
```

See Also

- "set_block_options"

5.10. set_port_block (Ask a Question)

This PDC command sets properties on a port in the block flow. This PDC command applies to only one I/O.

```
set_port_block -name portName -remove_ios value -add_interface value
```

Arguments

The following sections describe `set_port_block` arguments.

-name portName Specify the name of the port.

-remove_ios Value Sets whether or not to remove I/Os connected to the specified port from the netlist. The following table shows the acceptable values for this argument.

Table 5-9. -remove_ios Value

Value	Description
yes	Remove I/Os connected to the specified port from the netlist.
no	Do not remove I/Os connected to the specified port from the netlist.

-add_interface Value Adds an interface macro each time the fanout of the net connected to the port is greater than the value specified. The value must be a positive integer.

Exceptions

- You must import this PDC command as a constraint file
- TRIBUFF and BIBUF macros cannot be removed even if you specify "-remove_ios yes"
- You must enable the block flow before calling this command. To enable the block flow, either select the "Enable block creation" option in the New Project wizard, or use the -block argument in the new_design Tcl command to enable block mode.

Examples

This example removes any I/Os connected to portA, excluding TRIBUFF and BIBUF I/Os.

```
set_port_block -name portA -remove_ios yes
```

5.11. set_block_options (Ask a Question)

This PDC command overrides the compile option for placement or routing conflicts for an instance of a block.

```
set_block_options -inst_name instance_name -placement_conflicts value \
-routing_conflicts value
```

Arguments

The following sections describe `set_block_options` arguments.

-inst_name instance_name Specifies the block instance name. If you do not know the name of the instance, run a **Block Report** (**Design** > **Reports** > **Blocks** > **Interface**) or look at the names shown in the **Block View** tab of the Chip Planner.

-placement_conflicts Value Specifies what to do when the software encounters a placement conflict. The following table shows the acceptable values for this argument.

Table 5-10. -placement_conflicts Value

Value	Description
error	Compile errors out if any instance from a Designer block becomes unplaced or its routing is deleted. This is the default compile option.
resolve	If some instances get unplaced for any reason, the non-conflicting elements remaining are also unplaced. Basically, if there are any conflicts, nothing from the block is kept.
keep	If some instances get unplaced for any reason, the non-conflicting elements remaining are preserved but not locked. Therefore, the placer can move them into another location if necessary.
lock	If some instances get unplaced for any reason, the non-conflicting elements remaining are preserved and locked.
discard	Discards any placement from the block, even if there are no conflicts.

-routing_conflicts Value

Specifies what to do when the software encounters a routing conflict. The following table shows the acceptable values for this argument.

Table 5-11. -routing_conflicts Value

Value	Description
error	Compile errors out if any route in any preserved net from a Designer block is deleted.
resolve	If a route is removed from a net for any reason, the routing for the non-conflicting nets is also deleted. Basically, if there are any conflicts, no routes from the block are kept.
keep	If a route is removed from a net for any reason, the routing for the non-conflicting nets is kept unlocked. Therefore, the router can re-route these nets.
lock	If routing is removed from a net for any reason, the routing for the non-conflicting nets is kept as locked, and the router will not change them. This is the default compile option.
discard	Discards any routing from the block, even if there are no conflicts.

Description

This command enables you to override the compile option for placement or routing conflicts for an instance of a block.

Exceptions

You must put this PDC command in a constraint PDC file and associate it to Place and Route. If placement is discarded, the routing is automatically discarded too.

Examples

This example makes the Libero Soc software display an error if any instance from a block becomes unplaced or the routing is deleted:

```
set_block_options -inst_name instA -placement_conflicts ERROR -routing_conflicts ERROR
```

See Also

- "move_block"

6. Post Layout Edit PDC Commands (Ask a Question)

Post Layout Edit PDC Commands are used when the design is in post-layout state.

6.1. edit_io (Ask a Question)

Use this PDC command to make the changes related to GPIO and HSIO type I/Os in the `edit_post_layout_design` tool.

Tip: This command is also supported by PolarFire and PolarFire SoC family of devices.

```
edit_io -port_name <port_name>\
        [-OUT_LOAD <value>]\
        [-RES_PULL <value>]\
        [-LOCK_DOWN <value>]\
        [-CLAMP_DIODE <value>]\
        [-SCHMITT_TRIGGER <value>]\
        [-SLEW <value>]\
        [-ODT <value>]\
        [-ODT_VALUE <value>]\
        [-OUT_DRIVE <value>]\
        [-IN_DELAY <value>]\
        [-OUT_DELAY <value>]
```

For PolarFire Transceiver type I/Os, `edit_io` supports the following attributes.

```
edit_io  -port_name <port_name>\
         [-TX_EMPHASIS_AMPLITUDE <value>]\
         [-TX_IMPEDANCE <value>]\
         [-TX_TRANSMIT_COMMON_MODE_ADJUSTMENT <value>]\
         [-RX_INSERTION_LOSS <value>]\
         [-RX_CALIBRATION <value>]\
         [-RX_CTLE <value>]\
         [-RX_CDR_GAIN <value>]\
         [-RX_TERMINATION <value>]\
         [-RX_PN_BOARD_CONNECTION]\
         [-RX_LOSS_OF_SIGNAL_DETECTOR_LOW <value>]\
         [-RX_LOSS_OF_SIGNAL_DETECTOR_HIGH <value>]
         [-RX_DFE_COEFFICIENT_H1 <value>]\
         [-RX_DFE_COEFFICIENT_H2 <value>]\
         [-RX_DFE_COEFFICIENT_H3 <value>]\
         [-RX_DFE_COEFFICIENT_H4 <value>]\
         [-RX_DFE_COEFFICIENT_H5 <value>]\
         [-RX_POLARITY <value>]
```

Arguments

For arguments related to GPIO and HSIO type I/Os, see set_io (RTG4 only).

Note: The arguments **-pin_name**, **-fixed**, and **-io_std** are not supported by **edit_io** PDC command.

The following are the arguments for RTG4 Transceiver type I/Os.

-TX_EMPHASIS_AMPLITUDE <value>	Adjusts the transmit emphasis and DC amplitude settings of the transmitter output drivers. The default value is 400mV_with_-1.0dB. Direction: Output

Table 6-1. TX Emphasis Amplitude Values

Name	Values
TX_EMPHASIS_AMPLITUDE	100mV_with_0dB
	200mV_with_0dB
	200mV_with_-1.0dB
	200mV_with_-2.5dB
	200mV_with_-3.5dB
	200mV_with_-4.4dB
	200mV_with_-6.0dB
	300mV_with_0dB
	400mV_with_0dB
	400mV_with_-1.0dB
	400mV_with_-2.5dB
	400mV_with_-3.5dB
	400mV_with_-4.4dB
	400mV_with_-6.0dB
	500mV_with_0dB
	600mV_with_-3.5dB
	600mV_with_-6.0dB
	800mV_with_0dB
	800mV_with_-1.0dB
	800mV_with_-2.5dB
	800mV_with_-3.5dB
	800mV_with_-4.4dB
	800mV_with_-6.0dB
	1000mV_with_0dB
	1000mV_with_-1.0dB
	1000mV_with_-2.5dB
	1000mV_with_-3.5dB
	1000mV_with_-4.4dB
	1000mV_with_-6.0dB

-TX_IMPEDANCE <value>

Adds calibrated internal impedance onto the differential outputs. The default value is 100.

Direction: Output

Table 6-2. TX_IMPEDANCE Values

Name	Values
TX_IMPEDANCE	150
	100
	85
	180

-TX_TRANSMIT_COMMON_MODE_ADJUSTMENT <value>

Transmit Common-mode level is used as a percentage of full Common-mode level or VDDA. It is only adjusted when DC coupled. For AC coupled systems, the level must remain as default. The default value is 50.

Direction: Output

Table 6-3. TX_TRANSMIT_COMMON_MODE_ ADJUSTMENT Values

Name	Values
TX_TRANSMIT_COMMON_MODE_ ADJUSTMENT	50
	60
	70
	80

-RX_INSERTION_LOSS <value>

Sets the predefined settings used to statically adjust the receiver CDR and DFE. The default value is 6.5 dB.

Direction: Input

Table 6-4. RX_INSERTION_LOSS Values

Name	Values
RX_INSERTION_LOSS	6.5 dB
	17.0 dB
	25.0 dB

-RX_CALIBRATION <value>

For more information about Receiver Calibration, see PolarFire Family Transceiver User Guide.

Direction: Input

Table 6-5. RX_CALIBRATION Values

Name	Values
RX_CALIBRATION	None_CDR
	On Demand
	On Demand and First Lock
	None_DFE

-RX_CTLE

Sets the receiver equalization settings used to reduce the low-frequency component of the signal while boosting the high frequency component. The default value is set based on data-rate and Rx insertion loss model.

Direction: Input

For Rx CTLE Settings table, see AC483: PolarFire FPGA Transceiver Signal Integrity Application Note.

-RX_CDR_GAIN <value>

CDR Gain denotes the effect of Gain on Jitter. Low CDR gain denotes low CDR lock time and better jitter tolerance whereas High CDR gain denotes faster CDR lock time and high jitter.

Direction: Input

Table 6-6. RX_CDR_GAIN Values

Name	Values
RX_CDR_GAIN	Low
	High

-RX_TERMINATION <value>

Sets a calibrated input termination for available differential impedances within the Rx buffer. The default value is 100.

Direction: Input

Table 6-7. RX_TERMINATION Values

Name	Values
RX_TERMINATION	150
	100
	85

-RX_PN_BOARD_CONNECTION <value>

Sets the coupling type for PCB. The default value is AC_COUPLED_WITH_EXT _CAP.

Direction: Input

Table 6-8. RX_PN_BOARD_CONNECTION Values

Name	Values
RX_PN_BOARD_CONNECTION	AC_COUPLED_WITH_EXT_CAP
	DC_COUPLED

-RX_LOSS_OF_SIGNAL_DETECTOR_LOW <value>

Sets the lower set point for a Loss-of-signal (LOS) detector to ensure that a good signal is applied to the receiver. The default value is OFF.

Direction: Input

Table 6-9. RX_LOSS_OF_SIGNAL_DETECTOR_LOW Values

Name	Values
RX_LOSS_OF_SIGNAL_DETECTOR_LOW	OFF
	PCIE
	SATA
	BMR
	1
	2
	3
	4
	5
	6
	7

-RX_LOSS_OF_SIGNAL_DETECTOR_HIGH <value>

Sets the higher set point for an LOS detector to ensure that a good signal is applied to the receiver. The default value is OFF.

Direction: Input

Table 6-10. RX_LOSS_OF_SIGNAL_DETECTOR_HIGH Values

Name	Values
RX_LOSS_OF_SIGNAL_DETECTOR_HIGH	OFF
	PCIE
	SATA
	BMR
	1
	2
	3
	4
	5
	6
	7

-RX_DFE_COEFFICIENT_H1 <value>

Sets the DFE coefficients for a design set in static mode. These attributes are optional and take integer values between 0 and 15. The corresponding register fields are 5 bits wide in all cases with the MSB bit reserved for sign bit.

The same values are applicable for `-RX_DFE_COEFFICIENT_H2`, `-RX_DFE_COEFFICIENT_H3`, `-RX_DFE_COEFFICIENT_H4` and `-RX_DFE_COEFFICIENT_H5`.

-RX_POLARITY <value>

This attribute is used to swap the P and N receiver pins, which provide flexible PCB routing by interchanging the devices physical pin to the logical signal. The default value is Normal.

Direction: Input

Table 6-11. RX_POLARITY Values

Name	Value
RX_POLARITY	Normal
	Inverted

Example

```
edit_io A -RES_PULL Down
-CLAMP_DIODE LVCMOS15 \
-OUT_DRIVE 12
```

7. Revision History (Ask a Question)

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

Revision	Date	Description
M	05/2025	The following changes are made in this revision: • Updated the set_preserve command. • Updated the define_region command.
L	08/2024	This document is released with Libero SoC Design Suite v2024.2 without changes from v2024.1.
K	02/2024	This document is released with Libero SoC Design Suite v2024.1 without changes from v2023.2.
J	08/2023	This document is released with Libero SoC Design Suite v2023.2 without changes from v2023.1.
H	04/2023	The following changes are made in this revision: • Updated section Netlist Attributes NDC Commands. • Added edit_io command to Post Layout Edit PDC Commands.
G	12/2022	This document is released with Libero SoC Design Suite v2022.3 without changes from v2022.2.
F	08/2022	The following changes are made in this revision: • Updated the set_io (SmartFusion 2 and IGLOO 2) command. • Updated the set_io (RTG4 only) command. • Updated the set_ioff command.
E	04/2022	This document is released with Libero SoC Design Suite v2022.1 without changes from v2021.3.
D	12/2021	Editorial updates only. No technical content updates.
C	08/2021	This document is released with Libero SoC Design Suite v2021.2 without changes from v2021.1.
B	04/2021	Editorial updates only. No technical content updates.
A	11/2020	Document converted to Microchip® template. Initial Revision.

Microchip FPGA Support

Microchip FPGA products group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, and worldwide sales offices. Customers are suggested to visit Microchip online resources prior to contacting support as it is very likely that their queries have been already answered.

Contact Technical Support Center through the website at www.microchip.com/support. Mention the FPGA Device Part number, select appropriate case category, and upload design files while creating a technical support case.

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

- From North America, call **800.262.1060**
- From the rest of the world, call **650.318.4460**
- Fax, from anywhere in the world, **650.318.8044**

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at https://www.microchip.com/en-us/about/legal-information/microchip-trademarks.

ISBN: 979-8-3371-1214-5

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.