

## **Table of Contents**

Purpose
References
CorePCIF Overview
CorePCIF Design Implementation
Recommendations for 33 MHz CorePCIF Design (32-bit and 64-bit)
Recommendation for 66 MHz CorePCIF Design (32-bit and 64-bit)
Conclusion
Appendix A: SDC Constraint and SmartTime Timing Analysis for 32-bit 33 MHz PCI Design 17
Appendix B: SDC Constraint and SmartTime Timing Analysis for 64-bit 66 MHz PCI Design 19
Appendix C: CorePCIF Sample PDC Constraint File for SmartFusion2 and IGLOO2
List of Changes

## **Purpose**

This application note describes the timing optimization challenges on implementing CorePCIF in SmartFusion<sup>®</sup>2 and IGLOO<sup>®</sup>2 devices and provides detailed information to overcome these challenges. It also provides implementation hints to follow in different stages of FPGA design, including I/O placement, synthesis, and place and route guidelines.

Note: It is recommended to have a basic understanding about CorePCIF and to be familiar with the Libero<sup>®</sup> System-on-Chip (SoC) v11.3 software tools and design flow when reading this application note.

## Introduction

CorePCIF is a Microsemi<sup>®</sup> supported intellectual property (IP) that provides a simple and flexible interface to the PCI bus. The CorePCIF, as a DirectCore, is designed, verified, supported, and maintained by Microsemi for Microsemi FPGA devices. It supports several Microsemi FPGA device families such as SmartFusion2, and IGLOO2. Refer to the *CorePCIF Handbook* for more information about device support and utilization.



## References

The following is the reference used:

CorePCIF Handbook

## **CorePCIF Overview**

Microsemi CorePCIF allows to connect a memory, a FIFO, and a processor subsystem resources to the main system PCI bus. The CorePCIF is used with a variety of peripherals where high-performance data transactions are required. Figure 1 shows the CorePCIF System Block Diagram.



#### Figure 1 • CorePCIF System Block Diagram

The CorePCIF enables implementing the PCI target and/or master functions and can be customized through parameters in the code. Several parameters are provided to easily change the following features:

- PCI vendor and device IDs
- BAR mapping
- · Memory depth
- I/O space

The CorePCIF has the below three major functional blocks:

- Target Controller
- Master Controller
- Datapath



Refer to the *CorePCIF Handbook* for more information. For a Target and Master, all the three blocks are required. Otherwise, only the Datapath and either the Target or Master function are required. The CorePCIF supports both 32-bit and 64-bit PCI implementations and also supports both 33 MHz and 66 MHz operations. The back-end interface of CorePCIF can be directly used to interface the memory, registers, or peripherals with the user fabric design.

Microsemi also provides another variation of CorePCIF DirectCore IP, called CorePCIF\_AHB. The CorePCIF\_AHB allows an AHB bus system to be connected to a PCI bus. It allows both the AHB bus and PCI bus to initiate data transfers between the two buses. It is built on top of the CorePCIF core.

Refer to the *CorePCIF\_AHB Handbook* for more information. This application note mainly focuses on timing optimization of CorePCIF IP, but can also be used for CorePCIF\_AHB DirectCore IP.

Table 1 lists the PCI specification timing requirements. While implementing the CorePCIF IP, apply these requirements as timing constraints by following the guidelines from this application note.

PCI Ports		33 MHz			66 MHz	
	Setup	Hold	Clock to output	Setup	Hold	Clock to output
	(ns)	(ns)	(ns)	(ns)	(ns)	(ns)
AD	7	0	11	3	0	6
CBEN						
DEVSELN						
FRAMEN						
IRDYN						
TRDYN						
PAR						
PERRN						
SERRN						
STOPN						
TRDYN						
PAR64						
ACK64N						
REQ64N						
IDSEL	10	0	-	5	0	-
GNTN						
INTAN	-	-	11	-	-	6
REQN						

Table 1 • PCI Specification Timing Requirements

Because of the tight timing margins for the external setup and clock to out timing checks, most of the timing critical paths are through the Datapath block. Assigning the PCI pins carefully reduces the workload for the place and route algorithms and makes it easier to achieve timing closure for these timing paths, especially in the larger SmartFusion2 and IGLOO2 devices.





Figure 2 shows the datapath for the 32-bit mode.

Figure 2 • Datapath Block Showing Critical Path for clk-out and External Setup



The following sections describe the design implementation guidelines for CorePCIF in the SmartFusion2 and IGLOO2 devices:

- SmartFusion2 and IGLOO2 Families- PCI I/O Assignments
- Bank Assignments 33 MHz PCI Design
- Bank Assignments 66 MHz PCI Design
- Bank and PCI I/O Assignment in Libero SoC
- Guidelines for Assigning the Special PCI I/O Signals
- Guidelines for RESET, TRDYn, and IRDYn Signals

#### SmartFusion2 and IGLOO2 Families- PCI I/O Assignments

In the SmartFusion2 and IGLOO2 devices, I/Os are grouped into I/O banks. Each I/O bank has its own voltage supply. For the signals, the compatible voltage supply can only be assigned to the same bank. Following are the three types of I/O banks in the SmartFusion2 and IGLOO2 families:

- MSIO
- MSIOD
- DDRIO

The MSIOD and DDRIO types can only operate at voltages up to 2.5V and do not support direct connection to the PCI bus. The MSIO banks, which operate at 3.3V, can be configured to the PCI I/O standard. So, when implementing a specific core variation of CorePCIF, ensure that the target device has a sufficient number of MSIO available. Table 2 shows the I/O count requirements for different CorePCIF variations. For example, implementing a 32-bit target only version of CorePCIF, requires 48 PCI I/Os. Implementing a 64-bit master and target version of CorePCIF requires 89 PCI I/Os.

#### Table 2 • PCI I/O Requirements for CorePCIF

CorePCIF Variation	Number of PCI I/O
32-bit Target	48
64-bit Target	88
32-bit Target and Master	50
64-bit Target and Master	89



#### Bank Assignments - 33 MHz PCI Design

Achieving timing closure for a 33 MHz PCI design for the SmartFusion2 and IGLOO2 devices is easy as long as the I/O locations are selected that are in adjacent banks. An exception is when a JTAG bank appears between the MSIO banks. The MSIO banks can be selected that have a JTAG bank between them.

For example, on the M2S050T-FG896 device, assign the PCI I/Os to banks 1, 2, and 3 only, as shown in Figure 3. Though, the bank 8 is available for PCI I/O assignment, it is recommended not to use it with banks 1, 2, and 3. Assigning the bank 8 for the PCI I/O signals causes the design placement to spread out on both the sides, which makes the die making timing closure more difficult to achieve.



Figure 3 • PCI I/O Assignment in M2S050T Device for 33 MHz PCI Design

#### Bank Assignments - 66 MHz PCI Design

Achieving the timing closure for 66 MHz design for the SmartFusion2 and IGLOO2 devices is difficult. Select the PCI I/O locations and banks more carefully. Selecting the adjacent I/O banks is the first step. An exception is when a JTAG bank appears between the MSIO banks, and to select the MSIO banks that have a JTAG bank between them. Table 3 lists the recommended bank assignments for PCI I/Os for SmartFusion2 and IGLOO2 devices. It also lists the number of I/Os available in those banks.



PCI Variatio n	Required PCI I/Os	FC1152	FG896	FG676			FG484		
		M2S150T M2GL150 T	M2S050T M2GL050T	M2S090T M2GL090T	M2S090T M2GL090T	M2S050T M2GL050T	M2S025T M2GL025T	M2S010T M2GL010T	M2S005T M2GL005
32-bit Target	48	Bank3,4,5, 8 (174) Or Bank17,18 (82)	Bank1,2,3 (92)	Bank2,3,5 (198) Or Bank0,8 (110)	Bank2,3,5 (116)	Bank1,3 (64)	Bank1,2,4 (116)	Bank1,2,4 (82)	Bank1,2,4 (56)
64-bit Target	88	Bank3,4,5, 8 (174)	Bank1,2,3 (92)	Bank2,3,5 (198) Or Bank0,8 (110))	Bank2,3,5 (116)	-	Bank1,2,4 (116)	-	-
32-bit Target and Master	50	Bank3,4,5, 8 (174) Or Bank17,18 (41 pairs)	Bank1,2,3 (92)	Bank2,3,5 (198) Or Bank0,8 (110)	Bank2,3,5 (116)	Bank1,3 (64)	Bank1,2,4 (116)	Bank1,2,4 (82)	Bank1,2,4 (56)
64-bit Target and Master	89	Bank3,4,5, 8 (174)	Bank1,2,3 (92)	Bank2,3,5 (198) Or Bank0,8 (110)	Bank2,3,5 (116)	-	Bank1,2,4 (116)	-	-

#### Table 3 • Recommended Bank Assignments for SmartFusion2 and IGLOO2 Devices

Table 4 •	<b>Recommended Bank</b>	Assignments for	r SmartFsuion2 a	nd IGLOO2 Devices
-----------	-------------------------	-----------------	------------------	-------------------

PCI Variation	Required PCI I/Os		v	F400			FC325	
		M2S050T M2GL050T	M2S025T M2GL025T	M2S010T M2GL010T	M2S005T M2GL005T	M2S090T M2GL090T	M2S050T M2GL050T	M2S025T M2GL025T
32-bit Target	48	Bank1,3 (64)	Bank1,2,4 (88)	Bank1,2,4 (82)	Bank1,2,4 (56)	Bank2,3 (54)	Bank1,2,3 (54)	Bank1,2,4 (66)
64-bit Target	88	-	Bank1,2,4 (88)	-	-	-	-	-
32-bit Target and Master	50	Bank1,3 (64)	Bank1,2,4 (88)	Bank1,2,4 (82)	Bank1,2,4 (56)	Bank2,3 (54)	Bank1,2,3 (54)	Bank1,2,4 (66)
64-bit Target and Master	89	-	-	-	-	-	-	-



### Bank and PCI I/O Assignment in Libero SoC

In the Libero SoC software, the multiview navigator (MVN) GUI can be used to configure the MSIO banks for PCI I/O standard, and assign the PCI I/Os to the selected I/O banks. The same task can be achieved by using a placement constraints (PDC) file. The following PDC constraint example shows assigning the AD[0] I/O to bank 3:

# Set Bank3 with a 3.3V setting (3.3V setting is required for the PCI IO standard)
set\_iobank Bank3 -vcci 3.30 -fixed yes
# Set AD[0] signal to Bank3 and enables IO register combining
set\_io {AD[0]} -iostd PCI -IN\_DELAY Off -OUT\_LOAD 10 -REGISTER Yes -OUT\_REG Yes DIRECTION INOUT

Refer to the "Appendix C: CorePCIF Sample PDC Constraint File for SmartFusion2 and IGLOO2" on page 21 for many examples of PDC file constraints for PCI I/O assignments.

### **Guidelines for Assigning the Special PCI I/O Signals**

The following sections describe the guidelines for assigning the PCI clock, reset, irdyn, and trdyn signals:

#### PCI CLK Assignment for 33MHz Design

To meet the PCI setup, hold, and clock-to-out timing requirements for a 33 MHz operating frequency, the PCI CLK must be placed at a chip global I/O location. But one of the dedicated global clock pins should be used for a direct access to a global clock buffer, called GBx.Refer to the SmartFusion2 and IGLOO2 FPGA Clocking Resources User Guide for more information. Alternatively, this can be done by setting the parameter USE\_GLOBAL\_CLK=1in the CorePCIF configurator. This setting uses a CLKBUF macro for the PCI CLK signal, and the layout software automatically places the PCI CLK at a global I/O location.

#### PCI CLK Assignment for 66 MHz Design

To meet the PCI setup, hold, and clock-to-out timing requirements for a 66 MHz operating frequency, the clocking scheme must be modified. The CorePCIF PCI CLK input signal must be driven by a fabric CCC (FCCC) with a negative programmable delay to meet the 6ns clock-to-out timing requirement.



Figure 4 shows the approximate worst-case timing delay for clock-to-out path in the SmartFusion2 and IGLOO2 devices with and without the FCCC macro.



Figure 4 • Clock-to-out Delay in SmartFusion2 With and Without FCCC

Ensure that the FCCC is instantiated in the design and uses the PCI CLK input as a reference clock. Configure the FCCC feedback option to CCC Internal and apply a negative delay on the programmable delay element path as shown in Figure 5. This FCCC configuration allows the clock to be shifted earlier,



which increases the available margin on the clk-to-out path. Either the FCCC output clock (GL0) or the CLK\_OUT signal from CorePCIF can be used to drive the back end logic.



Figure 5 • FCCC Configuration in SmartFusion2

Perform the following steps for a 66 MHz operating frequency:

- 1. Use a FCCC to de-skew the PCI clock.
- 2. Apply a negative delay (For example, -20 steps) on the programmable delay element path. Refer to the *SmartFusion2 and IGLOO2 Datasheet* for more information on the delay settings.
- 3. Connect the PCI CLK to the reference clock input of FCCC (CLK0\_PAD).
- 4. Connect the GL0 output of FCCC to the CLK input port of CorePCIF.



5. Use GL0 or CLK\_OUT signal from CorePCIF to drive the back end logic.





### Guidelines for RESET, TRDYn, and IRDYn Signals

The CorePCIF configurable parameters use the regular or global networks for routing the PCI reset, TRDYn, and IDRYn signals. Follow the guidelines listed in Table 5 for the PCI reset, TRDYn, and IDRYn signals.

Table 5 •	PCI Reset,	TRDYn,	and IDRYn	Signal	Guideline
Tuble 0				orginar	Guiacinic

PCI Signal	Parameter	Value	Description
PCI reset	USE_GLOBAL_RESET	0,1	0: Normal routing resources are used for PCI reset. However, due to the high fanout of the reset network, a buffer tree is created by synthesis.
			1: A global buffer is used for PCI reset.
			Microsemi recommends to set this parameter as 1.
TRDYn	USE_GLOBAL_TRDY	0,1	0: Normal routing resources are used for TRDYn.
			1: When MASTER = 1, a global buffer is used to drive the internal TRDY network in the core.
			Microsemi recommends to set this parameter as 0 for 66 MHz master or target and master version of CoerPCIF.
IRDYn	USE_GLOBAL_IRDY	0,1	0: Normal routing resources are used for IRDYn.
			1: When MASTER = 1, a global buffer is used to drive the internal IRDY network in the core.
			Microsemi recommends to set this parameter as 0 for 66 MHz master or target and master version of CoerPCIF.



## **CorePCIF Design Implementation**

This section describes the guidelines that must be followed for the synthesis and layout (place and route) software of the CorePCIF design.

### Synthesis Recommendations for CorePCIF Design

CorePCIF includes the synthesis timing requirements in the timing constraint file. Four timing constraints files are made available during synthesis such as 33 MHz, 66 MHz, 32-bit, and 64-bit operation.

Note: The same timing constraint files are used for target, master, or master and target core.

- pcitiming32\_33\_synplicity.sdc: 32-bit 33 MHz timing constraint
- pcitiming32\_66\_synplicity.sdc: 32-bit 66MHz timing constraint
- pcitiming64\_33\_synplicity.sdc: 64-bit 33 MHz timing constraint
- pcitiming64\_66\_synplicity.sdc: 64-bit 66 MHz timing constraint

The constraint files can be found under

<Libero\_prj>\component\Actel\DirectCore\COREPCIF\4.0.135\constraints folder. Note that the supplied timing constraints files assume a typical configuration for the core. So, some configurations may cause the timing constraint files to cause an error during compile operation in Libero SoC. For example, the ONCHIP\_IDSEL function is enabled, but the IDSEL input is not used in the configuration. The synthesis does remove the IDSEL input during configuration. In this case, the timing constraints for the IDSEL input should be manually removed from the SDC files using a text editor. During synthesis, import the appropriate timing constraint files and associate them with the synthesis project.

For 66 MHz PCI design, add the following synthesis constraints in the generated SDC file. The constraints impose a maximum fanout limit of 1 for output registers.

```
define attribute
{{i:UCORE.MAKE TARGET.DATAPATHE64.MAKE DATAPATH REGISTERS.AD REGS[31:0]}}{syn prese
rve} {1}
define attribute {{i:UCORE.MAKE\.UDMA.CBEN PAD[7:0]}} {syn replicate} {1}
define attribute {{i:UCORE.MAKE\.UDMA.CBEN PAD[7:0]}} {syn maxfan} {1}
define attribute
{{i:ucore.make target.datapathe.make datapath registers.ad regs[31:0]}}
{syn_preserve} {1}
define attribute{{i:UCORE.MAKE TARGET.BurstE.UA1\.MAKE ACK64 OUT.Q INT}}
{syn_replicate} {1}
define attribute {{i:UCORE.MAKE\.UDMA.MAKE REQ64N1.Q INT}} {syn replicate} {1}
define attribute {{i:UCORE.MAKE\.UDMA.MAKE REQ64N1.Q INT}} {syn maxfan} {1}
define attribute {{i:UCORE.MAKE TARGET.BurstE.UA1\.MAKE ACK64 OUT.Q INT}}
{syn maxfan} {1}
define attribute{{i:UCORE.MAKE TARGET.BurstE.UM1\.MAKE IRDY OUT.Q INT}}
{syn replicate} {1}
define attribute {{i:UCORE.MAKE TARGET.BurstE.UM1\.MAKE IRDY OUT.Q INT}}
{syn maxfan} {1}
```

The instance names used in this example must match with the names in the design. These constraints required to be applied as Synplify FPGA design constraint (FDC) file. Refer to the *Synopsys FPGA Synthesis Pro ME I-2013.09M SP1 User Guide* for more information.



Figure 7 shows the synthesis GUI with the required timing constraints for 66 MHz PCI design. The pcitiming64\_66\_synplicity.sdc is the 64-bit 66 MHz SDC timing constraint and PCISYSTEM\_syn.fdc is the Synplify FDC file that imposes the fanout limit.



Figure 7 • Synthesis GUI with Constraint

#### Layout (Place and Route) Recommendations for CorePCIF Design

This section describes the place and route recommendations for meeting the setup and hold timing requirements for external input, register-to-register, and clock-to-out paths. CorePCIF generates the SDC timing constraints for the place and route tool. Ensure that the timing constraint files are associated with the place and route tool. Following are the multiple timing constraint and physical constraint files made available for the various PCI versions:

- 33 MHz operation
- 66 MHz operation
- 32-bit operation
- 64-bit operation

Table 6 lists the CorePCIF generated SDC files. These files can be found under <*Libero\_prj*>\component\Actel\DirectCore\COREPCIF\4.0.135\constraints folder.

#### Table 6 • CorePCIF Generated SDC File Information

SDC File	Description
T_pcitiming32_33_designer.sdc	32-bit 33 MHz target only pci design
T_pcitiming32_66_designer.sdc	32-bit 66 MHz target only pci design



Table 6 •	CorePCIF	Generated	SDC F	File	Information	(continued)
-----------	----------	-----------	-------	------	-------------	-------------

SDC File	Description
T_pcitiming64_33_designer.sdc	64-bit 33 MHz target only pci design
T_pcitiming64_66_designer.sdc	66-bit 66 MHz target only pci design
TM_pcitiming32_33_designer.sdc	32-bit 33 MHz target and master pci design
TM_pcitiming32_66_designer.sdc	32-bit 66 MHz target and master pci design
TM_pcitiming64_33_designer.sdc	64-bit 33 MHz target and master pci design
TM_pcitiming64_66_designer.sdc	64-bit 66 MHz target and master pci design
M_pcitiming32_33_designer.sdc	32-bit 33 MHz master only pci design
M_pcitiming32_66_designer.sdc	32-bit 66 MHz master only pci design
M_pcitiming64_33_designer.sdc	64 bit 33 MHz master only pci design
M_pcitiming64_66_designer.sdc	64-bit 66 MHz master only pci design

Ensure that the correct timing constraint files are used during the place and route operation. The following steps describe how to import the timing constraint files:

- 1. Double-click the Timing Constraints to open the Import Files dialog box.
- 2. Browse to the Libero\_prj>\component\Actel\DirectCore\COREPCIF\4.0.135\constraints folder and import the timing constraint (\*.sdc) files.
- Right-click the selected constraint file and select Use for Compile. The selected constraint file is used during compilation for checking and passed on to the layout for timing driven place and route (TDPR).

To meet the timing for 33 MHz and 66 MHz PCI design, it is required to run a layout by selecting the High Effort Layout option.

# Recommendations for 33 MHz CorePCIF Design (32-bit and 64-bit)

This section provides guideline for the following 33 MHz CorePCIF design:

- Core Configuration
- I/O Assignment
- Design Implementation
- Note: The timing constraints are easily met for 33 MHz design using standard FPGA design flow, but for 66 MHz design it is recommended to follow the special guidelines.

#### **Core Configuration**

For RESET, TRDYn, and IRDYn signals assignment, Refer to the "Guidelines for RESET, TRDYn, and IRDYn Signals" section on page 11.

#### **I/O Assignment**

- Assign 3.3V to MSIO banks.
  - This ensures that PCI I/Os can be assigned to the MSIO bank. Use the MSIO bank as suggested in Figure 3 and Figure 4. Here is an example of PDC command:

set\_iobank Bank1 -vcci 3.30 -fixed yes

Use the PDC command to enable I/O register combining wherever possible.



 Use the -REGISTER Yes switch and the -OUT\_REG Yes or -IN\_REG Yes depending on whether the register resides in the input or output path for a particular I/O. If both Master and Target modes are enabled, they can function as both. Here is an example of PDC command:

```
set_io {AD[0]} -iostd PCI -IN_DELAY Off -OUT_LOAD 10 -REGISTER Yes\
-OUT_REG Yes -DIRECTION INOUT
```

- Do not pre-assign the PCI I/Os to any pin locations.
  - It is recommended to let the Place and Route tool perform the I/O placement. In some cases, where the boad-level schematic must be completed before FPGA progresses to the point of layout. Refer to "Appendix C: CorePCIF Sample PDC Constraint File for SmartFusion2 and IGLOO2" section on page 21.

#### **Design Implementation**

- 1. Enable the PCI timing SDC files for synthesis and layout steps. Refer to "Synthesis Recommendations for CorePCIF Design" section on page 12 and "Layout (Place and Route) Recommendations for CorePCIF Design" section on page 13.
- 2. Run Layout by selecting the High Effort Layout option.
- 3. Open SmartTime and perform static timing analysis for both MAX and MIN corners to verify that both setup and hold time requirements are met.

# Recommendation for 66 MHz CorePCIF Design (32-bit and 64-bit)

This section provides guideline for 66 MHz CorePCIF design. For 66 MHz design, it is required to apply the PCI timing constraints during synthesis and run layout with high-effort level option. It is also required to apply the additional maximum delay constraints to the input-register paths during layout. Here are the guidelines:

- Core Configuration
- I/O Assignment
- Design Implementation

#### **Core Configuration**

 Use global buffer for RESET. Use regular buffer for TRDYn and IRDYn signals to reduce the global network insertion delay. Refer to the "Guidelines for RESET, TRDYn, and IRDYn Signals" section on page 11.

#### I/O Assignment

- Assign 3.3V to MSIO banks.
  - This ensures that PCI I/Os can be assigned to MSIO bank. Use the MSIO bank suggested in Figure 3 and Figure 4. Here is an example of PDC command:

```
set_iobank Bank1 -vcci 3.30 -fixed yes
```

- Use PDC command to enable I/O register combining wherever possible.
  - Use the -REGISTER Yes switch and the -OUT\_REG Yes or -IN\_REG Yes depending on whether the register resides in the input or output path for a particular I/O. If both Master and Target modes are enabled, they can function as both. Here is an example of PDC command:

```
set_io {AD[0]} -iostd PCI -IN_DELAY Off -OUT_LOAD 10 -REGISTER Yes\
-OUT_REG Yes -DIRECTION INOUT
```



- Do not pre-assign PCI I/Os to any pin locations.
  - It is recommended to let the Place and Route tool perform the I/O placement. In some cases where the boad-level schematic must be completed before FPGA progresses to the point of layout. It is recommended to use the suggested PCI pinout available in "Appendix C: CorePCIF Sample PDC Constraint File for SmartFusion2 and IGLOO2" section on page 21.

#### **Design Implementation**

- Apply the PCI SDC and FDC constraints during synthesis and SDC constraints for layout. Refer to the "Synthesis Recommendations for CorePCIF Design" section on page 12 and "Layout (Place and Route) Recommendations for CorePCIF Design" section on page 13.
- 2. Run the layout by selecting the High Effort Layout option.
- 3. Run SmartTime to perform static timing analysis to check setup and hold time.

The setup and clk-to-out path are displayed as cross clock domain path due to the addition of FCCC as shown in "Appendix C: CorePCIF Sample PDC Constraint File for SmartFusion2 and IGLOO2" section on page 21.

4. Apply a 4.00 ns maximum delay constraint from PCI input to registers if the timing is not meet.

set\_max\_delay 4.000 -from [get\_ports { ACK64N AD[\*] CBEN[\*] CBEN[7] \
DEVSELN FRAMEN GNTN IDSEL IRDYN PAR PAR64 PERRN REQ64N RSTN STOPN TRDYN }] \
-to [get\_cells { \* }]

- 5. Run the layout again by selecting the High Effort Level option.
- 6. Open SmartTime and perform the static timing analysis at MAX and MIN corners to verify that both setup and hold time requirements are met.
- 7. If timing is not met, run Layout with multi-pass option using the tighter max delay constraint.

## Conclusion

This application note describes how to implement the CorePCIF IP for SmartFusion2 and IGLOO2 families. It also provides the guidelines and recommendations to meet the PCI timing requirements, especially for external setup and clock-to-out paths. The standard FPGA tool can be used if the operating frequency is 33 MHz. It is required to follow the specific guidelines and recommendations mentioned in this application note for 66 MHz operating frequency. The techniques mentioned in this application note for design optimization can be applied to other critical timing designs where meeting the I/O interface timing is a challenge.



Appendix A: SDC Constraint and SmartTime Timing Analysis for 32-bit 33 MHz PCI Design

## Appendix A: SDC Constraint and SmartTime Timing Analysis for 32-bit 33 MHz PCI Design

The following section shows SDC for 32-bit 33 MHz design and SmartTime analysis GUI.

SDC for 32-bit 33 MHz:

```
set period 30.0
set setup 23.0
set setupPP 20.0
set hold 0.0
set clkout 19.0
create clock -period $period { CLK }
*****
set input delay -max $setup -clock {CLK} { PAR }
set input delay -max $setup -clock {CLK} { PERRN }
set_input_delay -max $setup -clock {CLK} { FRAMEN }
set_input_delay -max $setup -clock {CLK} { IDSEL
                                                    }
set_input_delay -max $setup -clock {CLK} { STOPN
set_input_delay -max $setup -clock {CLK} { DEVSELN
set_input_delay -max $setup -clock {CLK} { IRDYN }
                             -clock {CLK} { DEVSELN }
set input delay -max $setup -clock {CLK} { TRDYN }
set input delay -max $setupPP -clock {CLK} { GNTN }
set_input_delay -max $setup -clock {CLK} { CBEN* }
set input delay -max $setup -clock {CLK} { AD* }
*****
set_input_delay -min $hold -clock {CLK} { PAR }
set_input_delay -min $hold -clock {CLK} { PERRN }
set_input_delay -min $hold -clock {CLK} { FRAMEN }
set_input_delay -min $hold -clock {CLK} { IDSEL }
set input delay -min $hold -clock {CLK} { STOPN }
set input delay -min $hold -clock {CLK} { DEVSELN }
set input delay -min $hold -clock {CLK} { IRDYN }
set input delay -min $hold -clock {CLK} { TRDYN }
set_input_delay -min $hold -clock {CLK} { GNTN }
set_input_delay -min $hold -clock {CLK} { CBEN* }
set input delay -min $hold -clock {CLK} { AD* }
*****
set output delay -max $clkout -clock {CLK} { FRAMEN }
set_output_delay -max $clkout -clock {CLK} { IRDYN
                                                    }
set output delay -max $clkout -clock {CLK} { TRDYN }
set output delay -max $clkout -clock {CLK} { STOPN }
set output delay -max $clkout -clock {CLK} { DEVSELN }
set output delay -max $clkout -clock {CLK} { PERRN }
set_output_delay -max $clkout -clock {CLK} { SERRN }
set_output_delay -max $clkout -clock {CLK} { REQN }
set_output_delay -max $clkout -clock {CLK} { PAR }
set output delay -max $clkout -clock {CLK} { INTAN }
set_output_delay -max $clkout -clock {CLK} { CBEN* }
set output delay -max $clkout -clock {CLK} { AD* }
***
#set false path -through { *UPAD*:E* }
set false path -through { *iobuf*:E*}
set false path -through { *obuft*:E*}
```



Figure 8 shows the SmartTime timing analysis GUI for 32-bit 33 MHz PCI design.



Figure 8 • SmartTime Timing Analysis GUI for 32-bit 33 MHz PCI Design



Appendix B: SDC Constraint and SmartTime Timing Analysis for 64-bit 66 MHz PCI Design

# Appendix B: SDC Constraint and SmartTime Timing Analysis for 64-bit 66 MHz PCI Design

The following section shows SDC for 64-bit 66 MHz design and SmartTime analysis GUI.

SDC for 64-bit 66 MHz:

```
set period 15.0
set setup 12.0
set setupPP 10.0
set hold 0.0
set clkout 9.0
create clock -period $period { CLK }
*****
set input delay -max $setup -clock {CLK} { PAR }
set input delay -max $setup -clock {CLK} { PERRN }
set_input_delay -max $setup -clock {CLK} { FRAMEN }
set_input_delay -max $setup -clock {CLK} { IDSEL
                                                     }
set_input_delay -max $setup -clock {CLK} { STOPN
set_input_delay -max $setup -clock {CLK} { DEVSELN
set_input_delay -max $setup -clock {CLK} { IRDYN }
                              -clock {CLK} { STOPN }
-clock {CLK} { DEVSELN }
set input delay -max $setup -clock {CLK} { TRDYN }
set input delay -max $setupPP -clock {CLK} { GNTN }
set_input_delay -max $setup -clock {CLK} { CBEN* }
set input delay -max $setup -clock {CLK} { AD* }
*****
set_input_delay -min $hold -clock {CLK} { PAR }
set_input_delay -min $hold -clock {CLK} { PERRN }
set_input_delay -min $hold -clock {CLK} { FRAMEN }
set_input_delay -min $hold -clock {CLK} { IDSEL }
set input delay -min $hold -clock {CLK} { STOPN }
set input delay -min $hold -clock {CLK} { DEVSELN }
set input delay -min $hold -clock {CLK} { IRDYN }
set input delay -min $hold -clock {CLK} { TRDYN }
set_input_delay -min $hold -clock {CLK} { GNTN }
set_input_delay -min $hold -clock {CLK} { CBEN* }
set input delay -min $hold -clock {CLK} { AD* }
*****
set output delay -max $clkout -clock {CLK} { FRAMEN }
set output delay -max $clkout -clock {CLK} { IRDYN }
set output delay -max $clkout -clock {CLK} { TRDYN }
set output delay -max $clkout -clock {CLK} { STOPN }
set output delay -max $clkout -clock {CLK} { DEVSELN }
set output delay -max $clkout -clock {CLK} { PERRN }
set_output_delay -max $clkout -clock {CLK} { SERRN }
set_output_delay -max $clkout -clock {CLK} { REQN }
set_output_delay -max $clkout -clock {CLK} { PAR }
set output delay -max $clkout -clock {CLK} { INTAN }
set_output_delay -max $clkout -clock {CLK} { CBEN* }
set output delay -max $clkout -clock {CLK} { AD* }
***
#set false path -through { *UPAD*:E* }
set false path -through { *iobuf*:E*}
set false path -through { *obuft*:E*}
```



Figure 9 shows the SmartTime timing analysis GUI.



#### Figure 9 • SmartTime Timing Analysis GUI for 64-bit 66 MHz PCI Design

Figure 10 shows the clock-to-out (Reg-lo) path and Figure 11 shows the input to register (lo-Reg) path for a typical 66-bit 66 MHz design in M2S150-FC1152 device.

Name	Туре	Net	Macro	Op D	lay Tot	al Fanout	Edge
Data_arrival_time_calculation							
UCORE/FCCC 0/CCC INST/INST CCC IP:GL0				0.0	00 0.00	0	
UCORE/FCCC 0/CCC INST/INST CCC IP:GL0	Clock source			+ 0.0	00 0.00	0	r .
	Clock generation			+ 1.3	67 1.26	7	
UCORE/FCCC 0/GL0 INST:An	net	UCORE/FCCC 0/GL0 net		+ 0.0	00 1.26	7	r .
UCORE/FCCC 0/GL0 INST:YEn	cell		ADLIB:GBM	+ 0.1	43 1.41	0 42	f
UCORE/FCCC 0/GL0 INST/U0 RGB1 RGB1:An	net	UCORE/FCCC 0/GL0 INST/U0 YWn GEast		+ 0.4	48 1.85	8	f
UCORE/FCCC 0/GL0 INST/U0 RGB1 RGB1;YR	cell		ADLIB:RGB	+ 0.2	50 2.10	8 6	r.
UCORE/my sd 0/COREPCIF 0/MAKE TARGET/DATAPATHE64/MAKE DATAPATH REGISTERS/AD REGSI151:	CLK net	UCORE/FCCC 0/GL0 INST/U0 RGB1 RGB1 rabr net 1		+ 0.6	00 2.70	8	r
UCORE/my sd 0/COREPCIE 0/MAKE TARGET/DATAPATHE64/MAKE DATAPATH REGISTERS/AD REGSI151:	) cell		ADLIB:IOOEFF	+ 0.3	63 2.97	1 1	f
AD jebuf(47)/U0/U IQQUTEEA	net	AD REG[47]		+ 0.3	57 3.22	8	f
AD inbuf(47)/10/11 IOOUTEEY	cell		ADUB-IOOUTEE BYPASS	+ 03	43 3 57	1 1	f
	net	AD iobuf(471/U0/DOUT	Abcabaccoon1_011455	+ 00	00 3 57	1	-
	cell	AD_10001(47)/00/0001	ADUB-IOPAD RI	+ 23	30 5.81	0 1	-
AD_000((4)) 500 (5)(0) AD: AD	net	AD[47]	ADDIDIO AD_DI	- 00	00 5.81	0	-
data arrival time	inex	A0(41)			5.81	0	
A Data required time calculation					5.01	•	
	Clock Constraint			15	000 15 0	00	
	Clock constraint				00 15.0	00	
	Output Dalay Constants			- 0.0	00 10.0	00	4
AD[47]	Output Delay Constrain			- 54	00 0.00	0	1
data required time					6.00	0	
	TARGET/DATAPA	THE64/MAKE DATAPATH REGISTED	RS/AD REGS[15]		6.00	0	
data required time  CLK UCORE/my_sd_0/COREPCIF_0/MAKE_		THE64/MAKE_DATAPATH_REGISTE	RS/AD_REGS[15]		6.00	0	
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE_ UCORE/FCCC_0/GL0_INST/U0_RGB:	TARGET/DATAPA	THE64/MAKE_DATAPATH_REGISTER	RS/AD_REGS[15]		6.00	0	
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE_ UCORE/FCCC_0/GL0_INST/U0_RGB: 		THE64/MAKE_DATAPATH_REGISTE	RS/AD_REGS[15]		6.00	0	
data required time  CLK UCORE/my_sd_0/COREPCIF_0/MAKE UCORE/FCCC_0/GL0_IDISORE/FCCC_0/GL0_INST/U0_RGB:  An YEn An YE	_TARGET/DATAPA L_RGB1 — ADn — ALn	THE64/MAKE_DATAPATH_REGISTER	RS/AD_REGS[15]		6.00	0	
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE_ UCORE/FCCC_0/GL0_INEGRE/FCCC_0/GL0_INST/U0_RGB: An YEn An YEN	TARGET/DATAPA L_RGB1 — ADn — ALn — CLK	THE64/MAKE_DATAPATH_REGISTER	RS/AD_REGS[15]		6.00	0	
data required time	TARGET/DATAPA L_RGB1 - ADn - ALn CLK	THE64/MAKE_DATAPATH_REGISTE	RS/AD_REGS[15]		6.00	0	
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE_ UCORE/FCCC_0/GL0_INEGTRE/FCCC_0/GL0_INST/U0_RGB: An YEn -ENn YWn -ENn YWn -ENn RGB	TARGET/DATAPA L_RGB1	THE64/MAKE_DATAPATH_REGISTER	RS/AD_REGS[15]	AD id	buf[4	7]/U0/	U IOPAD
data required time	TARGET/DATAPA' L_RGB1 - ADn - ALn - CLK - Dn	THE64/MAKE_DATAPATH_REGISTEI AD_iobuf[ <u>47]/U0/U</u>	RS/AD_REGS[15]	AD_io	buf[4	.7]/U0/	<u>u</u> Iopad
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE_ UCORE/FCCC_0/GL0_INST/U0_RGB: An YEnAn YL RGB RGB	TARGET/DATAPA L_RGB1	THE64/MAKE_DATAPATH_REGISTER AD_iobuf[47]/U0/U	RS/AD_REGS[15]	AD_io	buf[4	-7]/U0/	U_IOPAD
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE UCORE/FCCC_0/GL0_IDISORE/FCCC_0/GL0_INST/U0_RGB: An YEn ENn YWn GBM RGB	TARGET/DATAPA L_RGB1 - ADn - ALn - CLK - Dn - EN	THE64/MAKE_DATAPATH_REGISTER	RS/AD_REGS[15]	AD_io	buf[4		U_IOPAD
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE UCORE/FCCC_0/GL0_INST/U0_RGB An YEn -ENn YWn GBM RGB	TARGET/DATAPA L_RGB1 — ADn ALn CLK — Dn — EN — LAT	THE64/MAKE_DATAPATH_REGISTER AD_iobuf[47]/U0/U Q IOOUTFF_BYF	ICOUTFF	AD_io	buf[4 D E	7]/U0/	
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE UCORE/FCCC_0/GL0_IDISORE/FCCC_0/GL0_INST/U0_RGB: An YEn -ENn YWn -ENn YWn -ENn YR -RGB	TARGET/DATAPA L_RGB1 - ADn - ALn - CLK - Dn - EN - LAT	THE64/MAKE_DATAPATH_REGISTER AD_iobuf[47]/U0/U Q IOOUTFF_BYF	rs/AD_REGS[15] _IOOUTFF / VASS	AD_io	buf[4 D E	-7]/U0/	U_IOPAD
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE UCORE/FCCC_0/GL0_INST/U0_RGB An YEn -ENn YWn GBM RGB	TARGET/DATAPA L_RGB1 — ADn — ALn — CLK — Dn — EN — LAT — SD	THE64/MAKE_DATAPATH_REGISTER AD_iobuf[47]/U0/U Q IOOUTFF_BYF	rs/AD_REGS[15] 100UTFF	AD_io	buf[4 D E PAI	• <b>-7]/U0/</b>	U_IOPAD
dda required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE UCORE/FCCC_0/GL0_IDISORE/FCCC_0/GL0_INST/U0_RGB: An YEn ENn YWn GBM RGB	TARGET/DATAPA L_RGB1	THE64/MAKE_DATAPATH_REGISTER AD_iobuf[47]/U0/U Q IOOUTFF_BYF	IOOUTFF	AD_ic	buf[4 D E PAI	7]/U0/	U_IOPAD
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE_ UCORE/FCCC_0/GL0_INEGTRE/FCCC_0/GL0_INST/U0_RGB: AnYEnAnYL GBM RGB	TARGET/DATAPA L_RGB1 — ADn — ALn — CLK — Dn — EN — LAT — SD — SDR	THE64/MAKE_DATAPATH_REGISTER AD_iobuf[47]/U0/U Q IOOUTFF_BYF	IOOUTFF	AD_ic	buf[4 D E PAI	0 7]/U0/ 0 PAD_B]	
dda required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE UCORE/FCCC_0/GL0_IDISOTRE/FCCC_0/GL0_INST/U0_RGB An YEn -ENn YWn -ENn YR GBM RGB	TARGET/DATAPA L_RGB1 - ADn - ALn - CLK - Dn - EN - LAT - SD - SDR - SIR	THE64/MAKE_DATAPATH_REGISTER AD_iobuf[47]/U0/U Q IOOUTFF_BYF	IOOUTFF	AD_ic	buf[4 D E PAI	• <b>7]/U0/</b> • • • •	
data required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE_ UCORE/FCCC_0/GL0_INST/U0_RGB:	TARGET/DATAPA L_RGB1 — ADn — ALn — CLK — EN — LAT — SD — SDR — SLn	THE64/MAKE_DATAPATH_REGISTER AD_iobuf[47]/U0/U Q A IOOUTFF_BYF	IOOUTFF	AD_ic	buf[4 D E IO	• <b>7]/U0/</b> • • • •	U_IOPAD y  L AD[47]
dda required time CLK UCORE/my_sd_0/COREPCIF_0/MAKE UCORE/FCCC_0/GL0_IDISORE/FCCC_0/GL0_INST/U0_RGB: An YEn ENn YWn GBM RGB	TARGET/DATAPAT L_RGB1 - ADn - ALn - CLK - Dn - EN - LAT - SD - SDR - SLN IOOE	THE64/MAKE_DATAPATH_REGISTER AD_iobuf[47]/U0/U Q IOOUTFF_BYF	RS/AD_REGS[15]	AD_ic	bbuf[4 D E PAI	7]/U0/ D PAD_B]	U_IOPAD Y [ 

Figure 10 • SmartTime Timing Analysis GUI- clock-to-out Path for a 64-bit 66 MHz PCI Design



Appendix C: CorePCIF Sample PDC Constraint File for SmartFusion2 and IGLOO2

Name	Туре	Net	Macro	Op	Delay Total F
4 Data_arrival_time_calculation					
CLK					0.000 0.000
IRDYN	Input Delay Constrain	t		+	12.000 12.000
IRDYN_iobuf/U0/U_IOPAD:PAD	net	IRDYN		+	0.000 12.000
IRDYN_iobuf/U0/U_IOPAD:Y	cell		ADLIB:IOPAD_BI	+	2.124 14.124 1
IRDYN_iobuf/U0/U_IOINFF:A	net	IRDYN_iobuf/U0/YIN1		+	0.000 14.124
IRDYN_iobuf/U0/U_IOINFF:Y	cell		ADLIB:IOINFF_BYPASS	+	0.083 14.207 2
UCORE/my_sd_0/COREPCIF_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/CLK_GATE_PAR_in	B net	UCORE_my_sd_0_COREPCIF_0_MAKE_TARGET_ADD_PHASEE_IRDYN_BUFF		+	0.951 15.158
UCORE/my sd 0/COREPCIF 0/MAKE TARGET/DATAPATHE/MAKE DATAPATH REGISTERS/CLK GATE PAR i:	Y cell		ADLIB:CFG4	+	0.100 15.258 3
UCORE/my sd 0/COREPCIF 0/MAKE TARGET/DATAPATHE/MAKE DATAPATH REGISTERS/AD REGS[15]:EN	net	UCORE/my sd 0/COREPCIF 0/MAKE TARGET/DATAPATHE/MAKE DATAPATH REGISTERS/CLK GATE PAR i	0	+	1.381 16.639
data arrival time					16.639
4 Data required time calculation					
UCORE/FCCC 0/CCC INST/INST CCC IP:GL0	Clock Constraint				15,000 15,000
UCORE/FCCC 0/CCC INST/INST CCC IP:GL0	Clock source			+	0.000 15.000
	Clock generation			+	1.267 16.267
UCORE/FCCC 0/GL0 INST:An	net	UCORE/FCCC 0/GL0 net		+	0.000 16.267
UCORE/ECCC 0/GL0 INST:VEn	cell		ADLIB:GBM	+	0.129 16.396 4
UCORE/ECCC 0/GL0_INST/LI0_RGB1_RGB43:An	net	UCORE/ECCC 0/GL0 INST/U0 YWn GEast		+	0.380 16.776
UCORF/ECCC 0/GL0 INST/I/0 RGB1 RGB43:VR	cell		ADI IB:RGB	+	0.225 17.001 3
UCORE/my sd 0/COREPCIE 0/MAKE TARGET/DATAPATHE/MAKE DATAPATH REGISTERS/AD REGISTO/C/K	net	UCORF/ECCC 0/GL0 INST/U0 RGR1 RGR43 robr net 1		+	0.536 17.537
THE REAL PROPERTY AND A RE					
UCORE/my_sd_0/COREPCIE_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/AD_REGS[15]:EN	Library setun time		ADI IB:TOOFFF	-	0.419 17.118
UCORE/my sd 0/COREPCE_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/AD_REGS[15]EN data required time	Library setup time	-	ADLIB:IOOEFF	-	0.419 17.118 17.118
UCOBE/my_sd SUCOREPCIF_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATHEREGISTERS/AD_REGS[15]EN data required time	Library setup time	17	ADLIB:100EFF		0.419 17.118 17.118
UCORE/my_sd_9/COREPCIF_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/AD_REGS[15];EN data required time	Library setup time	" L <u>ej Ingstjoherre: ۵/Rar#2</u> target/datapathe/make_datapath_re	ADLIB:100EFF	EGS	0.419 17.118 17.118
UCORE//// JCOREPCE_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/AD_REGS[15]:EN data required time		" <u>لفا الإكتراطة المحمد المحم المحمد المحمد المحم </u>	ADLIB:100EFF	EG	0.419 17.118 17.118
UCORE/W_3d_0/COREPCE_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/aD_REGS[15];EN data required time		"  Led_IN(ST)BHPRUED/ROBE_TARGET/DATAPATHE/MAKE_DATAPATH_RE n YL ADn ALn	ADLIB:JOOEFF	EG	0.419 17.118 17.118
UCORE//rguired time	CIRE/IEIORE///G	Image: State	GISTERS/AD_R	EG	0.419 1/118 17.118
UCORE/my_sd SUCOREPCIP_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATHE/REGISTERS/AD_REGS[15]EN data required time UCORE/FCCC_0/GL0_IDXB An YEn ENn YWn GGBM		Image: State	GISTERS/AD_R	EG	0.419 1/118 17.118
UCORE/my_sd_9/COREPCIP_0/MAKE_TARGET/DATAPATH-E/MAKE_DATAPATH-[REGISTERS/AD_REGS[15];EN data required time UCORE/FCCC_0/GL0_IDUS An YEn ENn YWn GBM IRDYN_jobuf/U0/U_IDUBATRE/my_sd_0/COREPCIF_0/MAKE		Igitigstored provided	GISTERS/AD_R	EGS	5[15]
UCORE///UCOREPCIP_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/aD_REGS[15];EN data required time UCORE/FCCC_0/GL0_10(9) An YEn ENn YWn GBM IRDYN_jobuf/U0/U_I0/BAIRE/my_sd_0/COREPCIF_0/MAK 	COREVECTORE		GISTERS/AD_R	EGS	5[15]
UCORE/my_sd_9/COREPCIP_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/AD_REGS[15]:EN data required time UCORE/FCCC0/GL0IDEG An YEn 	ORE/ECIRE_/0/G	Igitigstoredeprices     International Additional Additiona Additional Additional Additiona Additional Additional Addi	ADUBJOOEFF	EGS	0.419 1/118 17.118
UCORE/VICY_Sd_UCOREPCE_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/aD_REGS[15]EN data required time UCORE/FCCC_0/GL0_10(9) An YEn ENn YWn GBM IRDYN_iobuf/U0/U_IOPATRE/rmy_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/rmy_sd_0/COREPCIF_0/MAKE A Y	CORE/ECORE_0/6 KE_TARGET/DA FF A B	Led <u>IN</u> SCTJRIEPREDD/RABE TARGET/DATAPATHE/MAKE_DATAPATH_RE n YL ADn Nn YR ALn RGB CLK TAPATHE/MAKE_DATAPATH_REGISTERS/CLK_GATE_PAR_i 	ADUBJOOEFF	EGS	17.118 17.118
UCORE/INV_S4_DUCOREPCE_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/AD_REGS[15]EN data required time UCORE/FCCC0/GL0_INVS Arr YEn 	URB//EORE///GA	Igitig     Igitig <td>ADLIBJOOEFF</td> <td>EGS</td> <td>17.118 17.118</td>	ADLIBJOOEFF	EGS	17.118 17.118
UCORE/VICUSED COREPCE DIMAKE TARGET/DATAPATHE/MAKE DATAPATH REGISTERS/AD_REGS[15]EN data required time C UCORE/FCCC_0/GL0_ING An YEn ENn YWn GBM IRDYN_iobuf/U0/U_IOPAIRE/rmy_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPAIRE/rmy_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPAIRE/rmy_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPAIRE/rmy_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPAIRE/rmy_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPAIRE/rmy_sd_0/COREPCIF_0/MAKE IRDYN_IOPAID	CORE/ECORE_0/66	LEATING.CORREPCTING D/RGB/G3 TARGET/DATAPATHE/MAKE_DATAPATH_RE n YL	ADLIBJOOEFF	EGS	17.118 17.118
UCORE///UCORE/CCC_0/GLO_INAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/AD_REGS[15]EN data required time UCORE/FCCC_0/GLO_INCS An YEn ENn YWn GBM IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_IOPATRE/MY_SS IOPAD_RI	URB//EORE///A	Igitigstoriggepred by Invasional Alling       n     YL       ADn       Nn     YR       RGB     CLK       TAPATHE/MAKE_DATAPATH_REGISTERS/CLK_GATE_PAR_i       Y     Lat       SO     SO	ADLIBJOOEFF	EGS	0.419 17.118 17.118
UCORE/VICUSED OF MAKE TARGET/DATAPATHE/MAKE DATAPATH/REGISTERS/AD_REGS[15]EN data required time UCORE/FCCC_0/GL0_IDUS	Library setup time	LEATING.CORREPCTING D/RGB/RGB/RGB/RGB/RGB/RGB/RGB/RGB/RGB/RGB	ADLIBJOOEFF	EGS	17.118 17.118
UCORE///UCORE/CCORE/CF_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/AD_REGS[15]EN data required time UCORE/FCCC_0/GL0_ING An YEn ENn YWn GBM IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_IOPATRE/MY_SS_IOPATRE/MY_SS_IOPATRE/MAKE_DATAPATHERS/AD_REGS[15]EN IRDYN_IOPATRE/MAKE_TARGET/DATAPATHE/MAKE_DATAPATHE/REGISTERS/AD_REGS[15]EN IRDYN_IOPATRE/MAKE_TARGET/DATAPATHE/MAKE_DATAPATHE/REGISTERS/AD_REGS[15]EN IRDYN_IOPATRE/MAKE_TARGET/DATAPATHE/RAME/INFO/INFO/INFO/INFO/INFO/INFO/INFO/INFO	CORE//ECORE///CA	Igitigstore     Initial State     Initia	ADLIBJOOEFF	EGS	17.118 17.118
UCORE/my_sd 9/COREPCE_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/AD_REGS[15]EN disrequied time UCORE/FCCC_0/GL0_INNE 	URE/FORE_OFG	LEATINGSTORHEPREFBD/RGB&2 TARGET/DATAPATHE/MAKE_DATAPATH_RE	ADLIBJOOEFF	tEGS	17.118 17.118
UCORE///UCORE/CCORPCE_0/MAKE_TARGET/DATAPATHE/MAKE_DATAPATH_REGISTERS/AD_REGS[15]EN dar required time UCORE/FCCC_0/GL0_IMS An YEn ENn YWn GBM IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_iobuf/U0/U_IOPATRE/my_sd_0/COREPCIF_0/MAKE IRDYN_IOPATRE/MY_SG_0/COREPCIF_0/MAKE IRDYN_IOPATRE/INF_IOPAT	CORE //ECORE /// A A KE_TARGET/DA FF A C C D	Led INSCUREPRED/RABE2 TARGET/DATAPATHE/MAKE_DATAPATH_RE         n       YL         ADn         Nn       YR         RGB       CLK         TAPATHE/MAKE_DATAPATH_REGISTERS/CLK_GATE_PAR_i         Y       Lat         SD       SDR         CFG4       SDR         IOQEFF       IOQEFF	ADLIBJOOEFF GISTERS/AD_R	EGS	17.118 17.118

Figure 11 • SmartTime Timing Analysis GUI- in-register Path for a 64-bit 66 MHz PCI Design

## Appendix C: CorePCIF Sample PDC Constraint File for SmartFusion2 and IGLOO2

You can download the sample SmartFusion2 PDC file design files from

http://soc.microsemi.com/download/rsc/?f=SF2\_IGL2\_COREPCIF\_PDC

The zipped file has the sample PDC files for the device package combination listed in Table 7. Refer to the *Readme.txt* file included in the zipped file for the directory structure and description. The bank assignment for SmartFusion2 and IGLOO2 are similar, so the same PDC file can be used for IGLOO2 also.

Table 7 •	PDC Files and De	vice Package	Combinations
-----------	------------------	--------------	--------------

PDC Files	Descriptions
M2S025_FC325_PCI_66_32_ios.pdc	M2S025-FC325 32-bit 66 MHz target and master pci design
M2S025_FG484_PCI_66_64_ios.pdc	M2S025-FG484 64-bit 66 MHz target and master pci design
M2S050_FG896_PCI_66_64_ios.pdc	M2S050-FG896 64-bit 66 MHz target and master pci design
M2S090_FG676_PCI_66_64_ios.pdc	M2S090-FG676 64-bit 66 MHz target and master pci design
M2S090_FG896_PCI_66_64_ios.pdc	M2S090-FG896 64-bit 66 MHz target and master pci design
M2S150_FC1152_PCI_33_32_ios.pdc	M2S150-FC1152 32-bit 33 MHz target and master pci design
M2S150_FC1152_PCI_33_64_ios.pdc	M2S150-FC1152 64-bit 33 MHz target and master pci design
M2S150_FC1152_PCI_66_64_ios.pdc	M2S150-FC1152 64-bit 66 MHz target and master pci design



## **List of Changes**

The following table lists the critical changes that are made in the current version:

Date	Changes	Page
Revision 1 (October, 2014)	First Release.	NA



Microsemi Corporate Headquarters One Enterprise, Aliso Viejo CA 92656 USA Within the USA: +1 (800) 713-4113 Outside the USA: +1 (949) 380-6100 Sales: +1 (949) 380-6136 Fax: +1 (949) 215-4996 E-mail: sales.support@microsemi.com Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense and security, aerospace, and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs, and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif. and has approximately 3,400 employees globally. Learn more at **www.microsemi.com**.

© 2014 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.