Altera GPIO IP Core User Guide

ug-altera_gpio
2017.05.08

Last updated for Intel® Quartus® Prime Design Suite: 17.0

Subscribe
Send Feedback

Contents

Altera GPIO IP Core User Guide.. 3
- Altera GPIO IP Core Features.. 3
- Altera GPIO IP Core Data Paths.. 4
  - Input Path.. 5
  - Output and Output Enable Paths... 6
- Altera GPIO Interface Signals... 8
  - Shared Signals... 10
  - Data Bit-Order for Data Interface.. 11
  - Input and Output Bus High and Low Bits.. 11
  - Data Interface Signals and Corresponding Clocks... 12
- Verifying Resource Utilization and Design Performance.. 12
- Altera GPIO Parameter Settings.. 13
- Register Packing.. 15
- Altera GPIO Timing... 15
  - Timing Components... 15
  - Delay Elements... 17
  - Timing Analysis... 17
  - Timing Closure Guidelines... 20
- Altera GPIO IP Core Design Examples... 20
  - Altera GPIO Synthesizable Quartus Prime Design Example................................... 21
  - Altera GPIO IP Core Simulation Design Example.. 21
- IP Migration Flow for Arria V, Cyclone V, and Stratix V Devices................................. 22
  - Migrating Your ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, and ALTIOBUF IP Cores.. 22
  - Guideline: Swap `datain_h` and `datain_l` Ports in Migrated IP............................ 23
- Altera GPIO IP Core User Guide Archives... 23
- Document Revision History... 23

Altera GPIO IP Core User Guide

The Altera GPIO IP core supports the general purpose I/O (GPIO) features and components. You can use GPIOs in general applications that are not specific to transceivers, memory interfaces, or LVDS.

The Altera GPIO IP core is available for Intel® Arria® 10 devices only. If you are migrating designs from Stratix® V, Arria V, or Cyclone® IV devices, you must migrate the ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, or ALTIOBUF IP cores.

Related Links

- IP Migration Flow for Arria V, Cyclone V, and Stratix V Devices on page 22
  The IP migration flow allows you to migrate the ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, and ALTIOBUF IP cores of Arria V, Cyclone V, and Stratix V devices to the Altera GPIO IP core of Arria 10 devices.
- Introduction to Intel FPGA IP Cores
  Provides general information about all Intel FPGA IP cores, including parameterizing, generating, upgrading, and simulating IP cores.
- Creating Version-Independent IP and Qsys Simulation Scripts
  Create simulation scripts that do not require manual updates for software or IP version upgrades.
- Project Management Best Practices
  Guidelines for efficient management and portability of your project and IP files.
- Altera GPIO IP Core User Guide Archives on page 23
  Provides a list of user guides for previous versions of the Altera GPIO IP core.
- Double Data Rate I/O (ALTDDIO_IN, ALTDDIO_OUT, and ALTDDIO_BIDIR) IP Cores User Guide
- I/O Buffer (ALTIOBUF) IP Core User Guide

Altera GPIO IP Core Features

The Altera GPIO IP core includes features to support the device I/O blocks. You can use the Quartus® Prime parameter editor to configure the Altera GPIO IP core.

The Altera GPIO IP core provides these components:

- Double data rate input/output (DDIO)—a digital component that doubles or halves the data rate of a communication channel.
- Delay chains—configure the delay chains to perform specific delay and assist in I/O timing closure.
- I/O buffers—connect the pads to the FPGA.

Intel Corporation. All rights reserved. Intel, the Intel logo, Altera, Arria, Cyclone, Enpirion, MAX, Nios, Quartus and Stratix words and logos are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries. Intel warrants performance of its FPGA and semiconductor products to current specifications in accordance with Intel's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Intel assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Intel. Intel customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.
*Other names and brands may be claimed as the property of others.

ISO 9001:2008 Registered

Altera GPIO IP Core Data Paths

Figure 1. High-Level View of Single-Ended GPIO

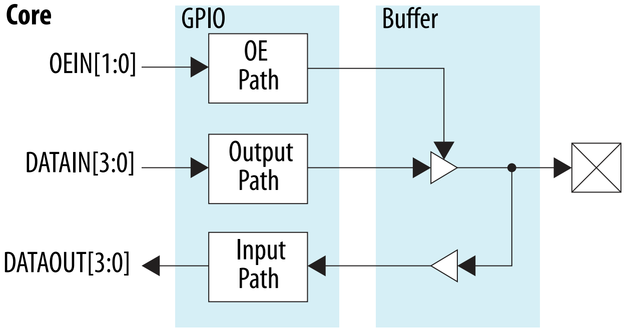


Table 1. Altera GPIO Data Path Modes

Data Path	Register Mode			
	Bypass	Simple Register	DDR I/O	
			Full-Rate	Half-Rate
Input	Data goes from the delay element to the core, bypassing all double data rate I/Os (DDIOs).	The full-rate DDIO operates as a simple register, bypassing half-rate DDIOs. The Fitter chooses whether to pack the register in the I/O or implement the register in the core, depending on the area and timing trade-offs.	The full-rate DDIO operates as a regular DDIO, bypassing the half-rate DDIOs.	The full-rate DDIO operates as a regular DDIO. The half-rate DDIOs convert full-rate data to half-rate data.
Output	Data goes from the core straight to the delay element, bypassing all DDIOs.	The full-rate DDIO operates as a simple register, bypassing half-rate DDIOs. The Fitter chooses whether to pack the register in the I/O or implement the register in the core, depending on the area and timing trade-offs.	The full-rate DDIO operates as a regular DDIO, bypassing the half-rate DDIOs.	The full-rate DDIO operates as a regular DDIO. The half-rate DDIOs convert full-rate data to half-rate data.
Bidirectional	The output buffer drives both an output pin and an input buffer.	The full-rate DDIO operates as a simple register. The output buffer drives both an output pin and an input buffer.	The full-rate DDIO operates as a regular DDIO. The output buffer drives both an output pin and an input buffer. The input buffer drives a set of three flip-flops.	The full-rate DDIO operates as a regular DDIO. The half-rate DDIOs convert full-rate data to half-rate. The output buffer drives both an output pin and an input buffer. The input buffer drives a set of three flip-flops.

If you use asynchronous clear and preset signals, all DDIOs share these same signals.

Half-rate and full-rate DDIOs connect to separate clocks. When you use half-rate and full-rate DDIOs, the full-rate clock must run at twice the half-rate frequency. You can use different phase relationships to meet timing requirements.

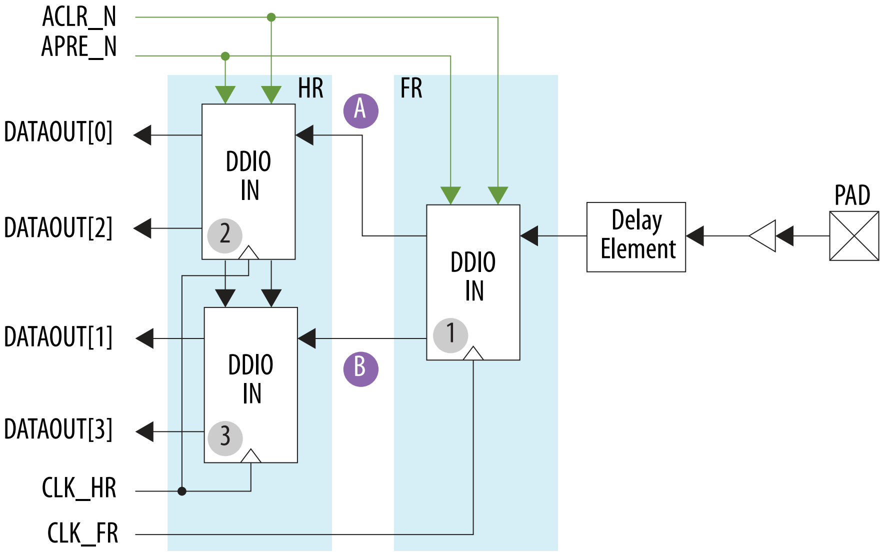
Related Links

Input and Output Bus High and Low Bits on page 11
The high and low bits in the input or output signals are included in the `din` and `dout` input and output buses.

Input Path

The pad sends data to the input buffer, and the input buffer feeds the delay element. After the data goes to the output of the delay element, the programmable bypass multiplexers select the features and paths to use.Each input path contains two stages of DDIOs, which are full-rate and half-rate.

Figure 2. Simplified View of Single-Ended GPIO Input Path

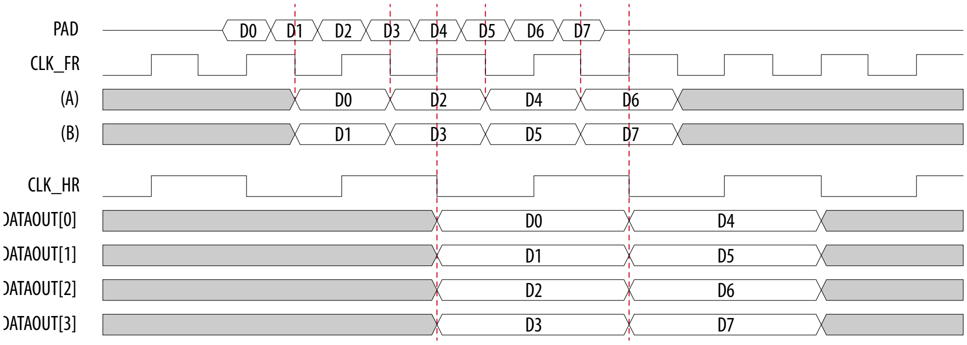

ACLR_N
APRE_N
HR
FR
A
DATAOUT[0]
DDIO IN
2
DATAOUT[2]
DDIO IN
1
Delay Element
PAD
DATAOUT[1]
DDIO IN
3
B
DATAOUT[3]
CLK_HR
CLK_FR

1. The pad receives data.
2. DDIO IN (1) captures data on the rising and falling edges of `CLK_FR` and sends the data, signals (A) and (B) in the following waveform figure, at single data rate.
3. DDIO IN (2) and DDIO IN (3) halve the data rate.
4. `DATAOUT[3:0]` presents the data as a half-rate bus.

Figure 3. Input Path Waveform in DDIO Mode with Half-Rate Conversion

In this figure, the data goes from full-rate clock at double data rate to half-rate clock at single data rate. The data rate is divided by four and the bus size is increased by the same ratio. The overall throughput through the Altera GPIO IP core remains unchanged.

The actual timing relationship between different signals may vary depending on the specific design, delays, and phases that you choose for the full-rate and half-rate clocks.


PAD
D0
D1
D2
D3
D4
D5
D6
D7
CLK_FR
(A)
D0
D2
D4
D6
(B)
D1
D3
D5
D7
CLK_HR
DATAOUT[0]
D0
D4
DATAOUT[1]
D1
D5
DATAOUT[2]
D2
D6
DATAOUT[3]
D3
D7

Note: The Altera GPIO IP core does not support dynamic calibration of the input path. For applications that require dynamic calibration of the input path, refer to the related information.

Related Links

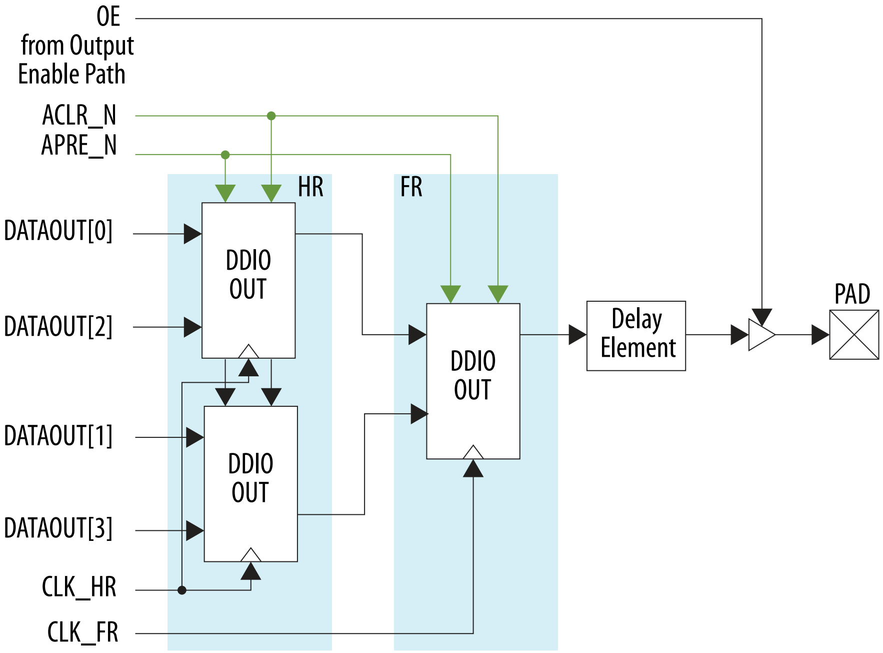
Altera PHYLite for Parallel Interfaces IP Core User Guide
Provides more information for applications that require dynamic calibration of the input path.

Output and Output Enable Paths

The output delay element sends data to the pad through the output buffer.

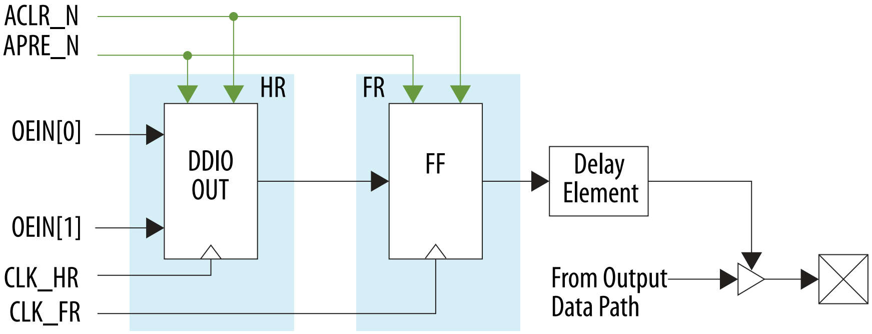
Each output path contains two stages of DDIOs, which are half-rate and full-rate.

Figure 4. Simplified View of Single-Ended GPIO Output Path


OE
from Output
Enable Path
ACLR_N
APRE_N
HR
FR
DATAOUT[0]
DDIO
OUT
DATAOUT[2]
DATAOUT[1]
DDIO
OUT
DATAOUT[3]
DDIO
OUT
Delay
Element
PAD
CLK_HR
CLK_FR

The output path waveform is symmetrical to the input path waveform.

Figure 5. Simplified View of Output Enable Path


ACLR_N
APRE_N
HR
FR
OEIN[0]
DDIO
OUT
FF
Delay
Element
OEIN[1]
CLK_HR
CLK_FR
From Output
Data Path

The difference between the output path and output enable (OE) path is that the OE path does not contain full-rate DDIO. To support packed-register implementations in the OE path, a simple register operates as full-rate DDIO. For the same reason, only one half-rate DDIO is present.

The OE path operates in the following three fundamental modes:

- Bypass—the core sends data directly to the delay element, bypassing all DDIOs.
- Packed Register—bypasses half-rate DDIO.
- SDR output at half-rate—half-rate DDIOs convert data from full-rate to half-rate.

Note: The Altera GPIO IP core does not support dynamic calibration of the output path. For applications that require dynamic calibration of the output path, refer to related information.

Related Links

- Altera PHYLite for Parallel Interfaces IP Core User Guide
  Provides more information for applications that require dynamic calibration of the output path.
- Input Path on page 5
  The pad sends data to the input buffer, and the input buffer feeds the delay element. After the data goes to the output of the delay element, the programmable bypass multiplexers select the features and paths to use.

Altera GPIO Interface Signals

Depending on parameter settings you specify, different interface signals are available for the Altera GPIO IP core.

Figure 6. Altera GPIO IP Core Interfaces

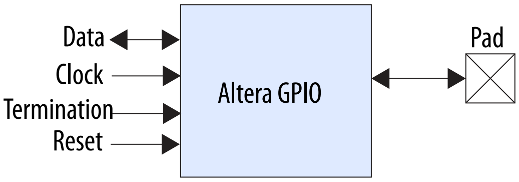

Data
Clock
Termination
Reset
Altera GPIO
Pad

Figure 7. Altera GPIO Interface Signals

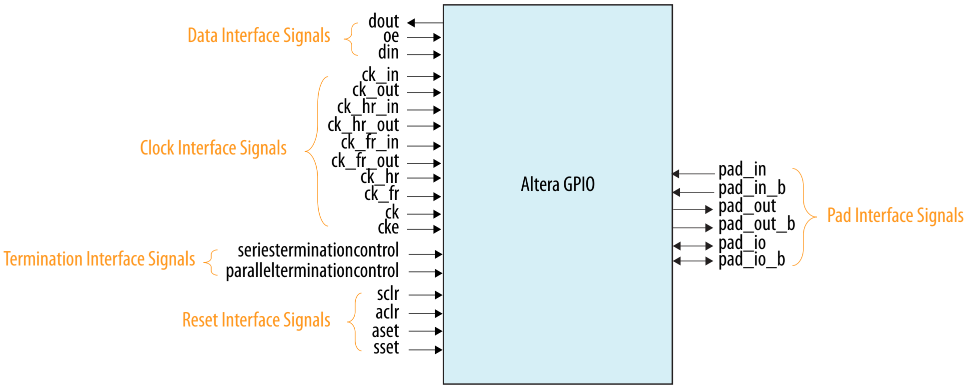

Data Interface Signals
dout
oe
din
Clock Interface Signals
ck_in
ck_out
ck_hr_in
ck_hr_out
ck_fr_in
ck_fr_out
ck_hr
ck_fr
ck
cke
Termination Interface Signals
seriesterminationcontrol
parallelterminationcontrol
Reset Interface Signals
sclr
aclr
aset
sset
Altera GPIO
pad_in
pad_in_b
pad_out
pad_out_b
pad_io
pad_io_b
Pad Interface Signals

Table 2. Pad Interface Signals

The pad interface is the physical connection from the Altera GPIO IP core to the pad. This interface can be an input, output or bidirectional interface, depending on the IP core configuration. In this table, *SIZE* is the data width specified in the IP core parameter editor.

Signal Name	Direction	Description
`pad_in[SIZE-1:0]`	Input	Input signal from the pad.
`pad_in_b[SIZE-1:0]`	Input	Negative node of the differential input signal from the pad. This port is available if you turn on the **Use differential buffer** option.
`pad_out[SIZE-1:0]`	Output	Output signal to the pad.

continued...

Signal Name	Direction	Description
`pad_out_b[`*SIZE*`-1:0]`	Output	Negative node of the differential output signal to the pad. This port is available if you turn on the **Use differential buffer** option.
`pad_io[`*SIZE*`-1:0]`	Bidirectional	Bidirectional signal connection with the pad.
`pad_io_b[`*SIZE*`-1:0]`	Bidirectional	Negative node of the differential bidirectional signal connection with the pad. This port is available if you turn on the **Use differential buffer** option.

Table 3. Data Interface Signals

The data interface is an input or output interface from the Altera GPIO IP core to the FPGA core. In this table, *SIZE* is the data width specified in the IP core parameter editor.

Signal Name	Direction	Description
`din[`*DATA_SIZE*`-1:0]`	Input	Data input from the FPGA core in output or bidirectional mode. *DATA_SIZE* depends on the register mode: • Bypass or simple register—*DATA_SIZE* = *SIZE* • DDIO without half-rate logic—*DATA_SIZE* = 2 × *SIZE* • DDIO with half-rate logic—*DATA_SIZE* = 4 × *SIZE*
`dout[`*DATA_SIZE*`-1:0]`	Output	Data output to the FPGA core in input or bidirectional mode, *DATA_SIZE* depends on the register mode: • Bypass or simple register—*DATA_SIZE* = *SIZE* • DDIO without half-rate logic—*DATA_SIZE* = 2 × *SIZE* • DDIO with half-rate logic—*DATA_SIZE* = 4 × *SIZE*
`oe[`*OE_SIZE*`-1:0]`	Input	OE input from the FPGA core in output mode with **Enable output enable port** turned on, or bidirectional mode. OE is active high. When transmitting data, set this signal to 1. When receiving data, set this signal to 0. *OE_SIZE* depends on the register mode: • Bypass or simple register—*DATA_SIZE* = *SIZE* • DDIO without half-rate logic—*DATA_SIZE* = *SIZE* • DDIO with half-rate logic—*DATA_SIZE* = 2 × *SIZE*

Table 4. Clock Interface Signals

The clock interface is an input clock interface. It consists of different signals, depending on the configuration. The Altera GPIO IP core can have zero, one, two, or four clock inputs. Clock ports appear differently in different configurations to reflect the actual function performed by the clock signal.

Signal Name	Direction	Description
`ck`	Input	In input and output paths, this clock feeds a packed register or DDIO if you turn off the **Half Rate logic** parameter. In bidirectional mode, this clock is the unique clock for the input and output paths if you turn off the **Separate input/output Clocks** parameter.
`ck_fr`	Input	In input and output paths, these clocks feed the full-rate and half-rate DDIOs if your turn on the **Half Rate logic** parameter. In bidirectional mode, the input and output paths use these clocks if you turn off the **Separate input/output Clocks** parameter.
`ck_hr`		
`ck_in`	Input	In bidirectional mode, these clocks feed a packed register or DDIO in the input and output paths if you specify both these settings: • Turn off the **Half Rate logic** parameter. • Turn on the **Separate input/output Clocks** parameter.
`ck_out`		

continued...

Signal Name	Direction	Description

`ck_fr_in`
`ck_fr_out`
`ck_hr_in`
`ck_hr_out`
	Input	In bidirectional mode, these clocks feed a full-rate and half-rate DDIOS in the input and output paths if you specify both these settings
		• Turn on the **Half Rate logic** parameter.
		• Turn on the **Separate input/output Clocks** parameter.
		For example, `ck_fr_out` feeds the full-rate DDIO in the output path.

`cke`	Input	Clock enable.

Table 5. Termination Interface Signals

The termination interface connects the Altera GPIO IP core to the I/O buffers.

Signal Name	Direction	Description
`seriesterminationcontrol`	Input	Input from the termination control block (OCT) to the buffers. It sets the buffer series impedance value.
`parallelterminationcontrol`	Input	Input from the termination control block (OCT) to the buffers. It sets the buffer parallel impedance value.

Table 6. Reset Interface Signals

The reset interface connects the Altera GPIO IP core to the DDIOs.

Signal Name	Direction	Description
`sclr`	Input	Synchronous clear.
`aclr`	Input	Asynchronous clear.
`aset`	Input	Asynchronous set.
`sset`	Input	Synchronous set.

Related Links

Input and Output Bus High and Low Bits on page 11

The high and low bits in the input or output signals are included in the `din` and `dout` input and output buses.

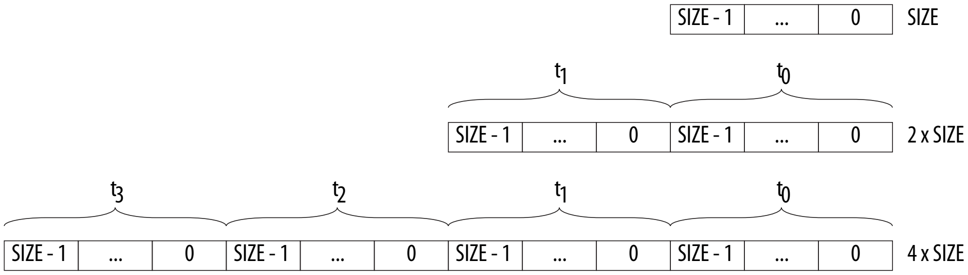
Shared Signals

- The input, output, and OE paths share the same clear and preset signals.
- The output and OE path shares the same clock signals.

Data Bit-Order for Data Interface

Figure 8. Data Bit-Order Convention

This figure shows the bit-order convention for the `din`, `dout` and `oe` data signals.



- If the data bus size value is `SIZE`, the LSB is at the right-most position.
- If the data bus size value is 2 × `SIZE`, the bus is made of two words of `SIZE` .
- If the data bus size value 4 × `SIZE`, the bus is made of four words of `SIZE`.
- The LSB is in the right-most position of each word.
- The right-most word specifies the first word going out for output buses and the first word coming in for input buses.

Related Links

Input Path on page 5

The pad sends data to the input buffer, and the input buffer feeds the delay element. After the data goes to the output of the delay element, the programmable bypass multiplexers select the features and paths to use.

Input and Output Bus High and Low Bits

The high and low bits in the input or output signals are included in the `din` and `dout` input and output buses.

Input Bus

For the `din` bus, if `datain_h` and `datain_l` are the high and low bits, with each width being `datain_width`:

- `datain_h` = `din[(2 × datain_width - 1):datain_width]`
- `datain_l` = `din[(datain_width - 1):0]`

For example, for `din[7:0]` = 8'b11001010:

- `datain_h` = 4'b1100
- `datain_l` = 4'b1010

Output Bus

For the `dout` bus, if `dataout_h` and `dataout_l` are the high and low bits, with each width being `dataout_width`:

- `dataout_h` = `dout[(2 × dataout_width - 1):dataout_width]`
- `dataout_l` = `dout[(dataout_width - 1):0]`

For example, for `dout[7:0]` = 8'b11001010:

- `dataout_h` = 4'b1100
- `dataout_l` = 4'b1010

Data Interface Signals and Corresponding Clocks

Table 7. Data Interface Signals and Corresponding Clocks

Signal Name	Parameter Configuration			Clock
	Register Mode	Half Rate	Separate Clocks	
`din`	• Simple Register • DDIO	Off	Off	ck
	DDIO	On	Off	ck_hr
	• Simple Register • DDIO	Off	On	ck_in
	DDIO	On	On	ck_hr_in
• `dout` • `oe`	• Simple Register • DDIO	Off	Off	ck
	DDIO	On	Off	ck_hr
	• Simple Register • DDIO	Off	On	ck_out
	DDIO	On	On	ck_hr_out
• `sclr` • `sset` • All pad signals	• Simple Register • DDIO	Off	Off	ck
	DDIO	On	Off	ck_fr
	• Simple Register • DDIO	Off	On	• Input path: ck_in • Output path: ck_out
	DDIO	On	On	• Input path: ck_fr_in • Output path: ck_fr_out

Verifying Resource Utilization and Design Performance

You can refer to the Quartus Prime compilation reports to get details about the resource usage and performance of your design.

1. On the menu, click **Processing ➤ Start Compilation** to run a full compilation.
2. After compiling the design, click **Processing ➤ Compilation Report**.
3. Using the **Table of Contents**, navigate to **Fitter ➤ Resource Section**.
    a. To view the resource usage information, select **Resource Usage Summary**.
    b. To view the resource utilization information, select **Resource Utilization by Entity**.

Altera GPIO Parameter Settings

You can set the parameter settings for the Altera GPIO IP core in the Quartus Prime software. There are three groups of options: **General**, **Buffer**, and **Registers**.

Table 8. Altera GPIO Parameters - General

Parameter	Condition	Allowed Values	Description
Data Direction	—	• Input • Output • Bidir	Specifies the data direction for the GPIO.
Data width	—	1 to 128	Specifies the data width.
Use legacy top-level port names	—	• On • Off	Use same port names as in Stratix V, Arria V, and Cyclone V devices. For example, `dout` becomes `dataout_h` and `dataout_l`, and `din` becomes `datain_h` and `datain_l`. *Note:* The behavior of these ports are different than in the Stratix V, Arria V, and Cyclone V devices. For the migration guideline, refer to the related information.

Table 9. Altera GPIO Parameters - Buffer

Parameter	Condition	Allowed Values	Description
Use differential buffer	—	• On • Off	If turned on, enables differential I/O buffers.
Use pseudo differential buffer	• Data Direction = Output • Use differential buffer = On	• On • Off	If turned on in output mode, enables pseudo differential output buffers. This option is automatically turned on for bidirectional mode if you turn on **Use differential buffer**.
Use bus-hold circuitry	• Data Direction = Input or Bidir • Use differential buffer = Off	• On • Off	If turned on, the bus hold circuitry can weakly hold the signal on an I/O pin at its last-driven state where the output buffer state will be 1 or 0 but not high-impedance.

continued...

Parameter	Condition	Allowed Values	Description
Use open drain output	- Data Direction = Output or Bidir - Use differential buffer = Off	- On - Off	If turned on, the open drain output enables the device to provide system-level control signals such as interrupt and write enable signals that can be asserted by multiple devices in your system.
Enable output enable port	Data Direction = Output	- On - Off	If turned on, enables user input to the OE port. This option is automatically turned on for bidirectional mode.
Enable seriestermination / paralleltermination ports	—	- On - Off	If turned on, enables the `seriesterminationcontrol` and `parallelterminationcontrol` ports of the output buffer.

Table 10. Altera GPIO Parameters - Registers

Parameter	Condition	Allowed Values	Description
Register mode	—	- None - Simple register - DDIO	Specifies the register mode for the Altera GPIO IP core: - **None**—specifies a simple wire connection from/to the buffer. - **Simple register**—specifies that the DDIO is used as a simple register in single data-rate mode (SDR). The Fitter may pack this register in the I/O. - **DDIO**— specifies that the IP core uses the DDIO.
Enable synchronous clear / preset port	- Register mode = DDIO	- None - Clear - Preset	Specifies how to implement synchronous reset port. - **None**—Disables synchronous reset port. - **Clear**—Enables the SCLR port for synchronous clears. - **Preset**—Enables the SSET port for synchronous preset.
Enable asynchronous clear / preset port	- Register mode = DDIO	- None - Clear - Preset	Specifies how to implement asynchronous reset port. - **None**—Disables asynchronous reset port. - **Clear**—Enables the ACLR port for asynchronous clears. - **Preset**—Enables the ASET port for asynchronous preset.
Enable clock enable ports	Register mode = DDIO	- On - Off	- **On**—exposes the clock enable (CKE) port to allow you to control when data is clocked in or out. This signal prevents data from being passed through without your control. - **Off**—clock enable port is not exposed and data always pass through the register automatically.
Half Rate logic	Register mode = DDIO	- On - Off	If turned on, enables half-rate DDIO.
Separate input / output Clocks	- Data Direction = Bidir - Register mode = Simple register or DDIO	- On - Off	If turned on, enables separate clocks (CK_IN and CK_OUT) for the input and output paths in bidirectional mode.

Related Links

- Input and Output Bus High and Low Bits on page 11
  The high and low bits in the input or output signals are included in the `din` and `dout` input and output buses.
- Guideline: Swap datain_h and datain_l Ports in Migrated IP on page 23
  When you migrate your GPIO IP to the Altera GPIO IP core, you can turn on **Use legacy top-level port names** option in the Altera GPIO IP core parameter editor. However, the behavior of these ports in the Altera GPIO IP core is different than in the IP cores used for the Stratix V, Arria V, and Cyclone V devices.

Register Packing

The Altera GPIO IP core allows you to pack register into the periphery to save area and resource utilization.

You can configure the full-rate DDIO on the input and output path as a flip flop. To do so, add the `.qsf` assignments listed in this table.

Table 11. Register Packing QSF Assignments

Path	QSF Assignment
Input register packing	`set_instance_assignment -name FAST_INPUT_REGISTER ON -to` *`<path to register>`*
Output register packing	`set_instance_assignment -name FAST_OUTPUT_REGISTER ON -to` *`<path to register>`*
Output enable register packing	`set_instance_assignment -name FAST_OUTPUT_ENABLE_REGISTER ON -to` *`<path to register>`*

Note: These assignments do not guarantee register packing. However, these assignments enable the Fitter to find a legal placement. Otherwise, the Fitter will keep the flip flop in the core.

Altera GPIO Timing

The performance of the Altera GPIO IP core depends on the I/O constraints and clock phases. To validate the timing for your Altera GPIO configuration, Intel recommends that you use the TimeQuest Timing Analyzer.

Related Links

The Quartus Prime TimeQuest Timing Analyzer

Timing Components

The Altera GPIO IP core timing components consist of three paths.

- I/O interface paths—from the FPGA to external receiving devices and from external transmitting devices to the FPGA.
- Core interface paths of data and clock—from the I/O to the core and from the core to I/O.
- Transfer paths—from half-rate to full-rate DDIO, and from full-rate to half-rate DDIO.

Note: The TimeQuest Timing Analyzer treats the path inside the DDIO_IN and DDIO_OUT blocks as black boxes.

Figure 9. Input Path Timing Components

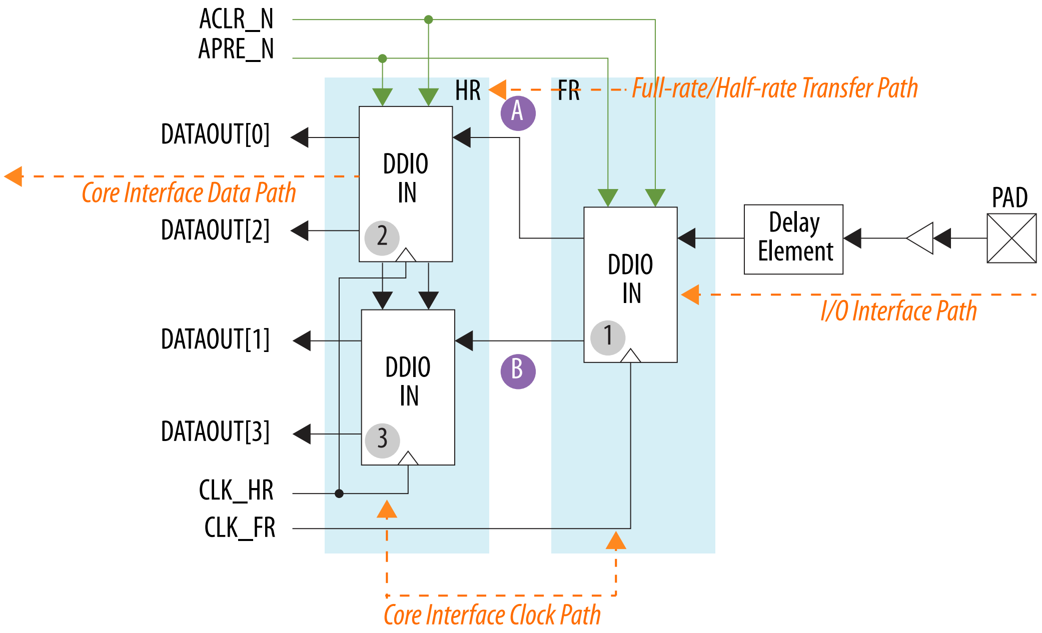

ACLR_N
APRE_N
HR
FR
Full-rate/Half-rate Transfer Path
A
DATAOUT[0]
DDIO
IN
Core Interface Data Path
DATAOUT[2]
2
DDIO
IN
1
Delay
Element
PAD
I/O Interface Path
DATAOUT[1]
DDIO
IN
B
DATAOUT[3]
3
CLK_HR
CLK_FR
Core Interface Clock Path

Figure 10. Output Path Timing Components

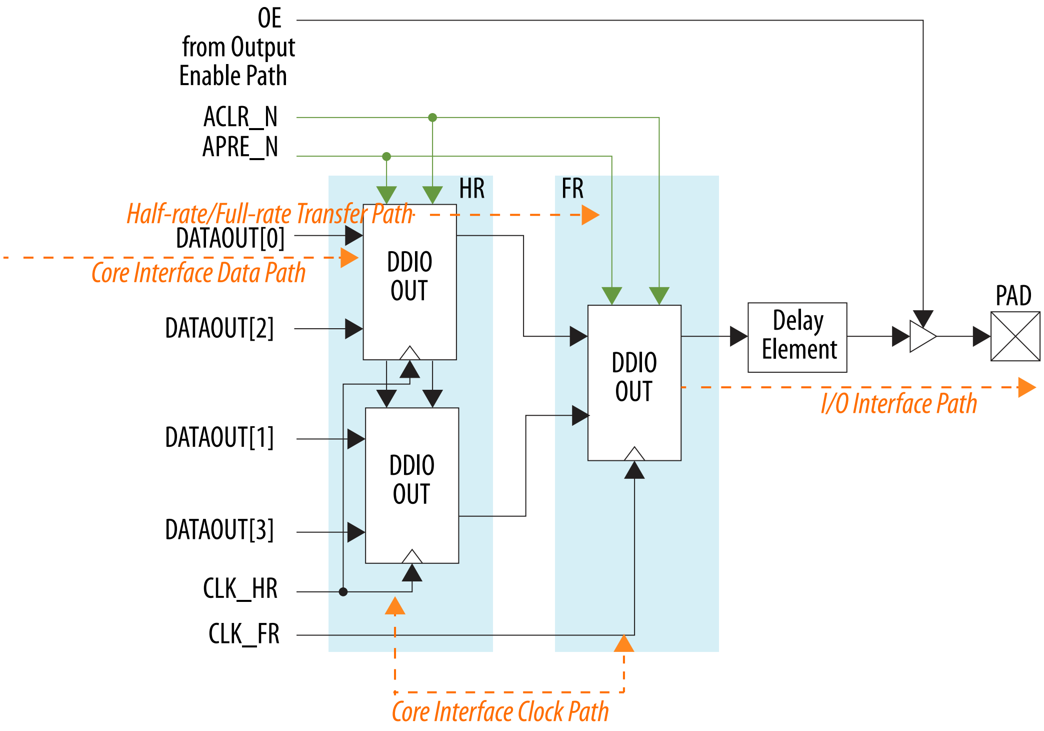
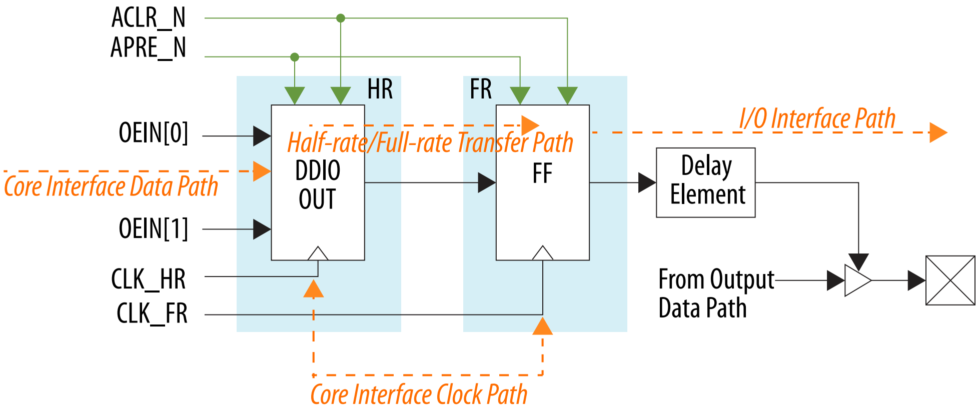

OE
from Output
Enable Path
ACLR_N
APRE_N
HR
FR
Half-rate/Full-rate Transfer Path
DATAOUT[0]
DDIO
OUT
Core Interface Data Path
DATAOUT[2]
DDIO
OUT
Delay
Element
PAD
I/O Interface Path
DATAOUT[1]
DDIO
OUT
DATAOUT[3]
CLK_HR
CLK_FR
Core Interface Clock Path

Figure 11. Output Enable Path Timing Components


ACLR_N
APRE_N
HR
FR
OEIN[0]
Core Interface Data Path
OEIN[1]
CLK_HR
CLK_FR
DDIO OUT
Half-rate/Full-rate Transfer Path
FF
I/O Interface Path
Delay Element
From Output Data Path
Core Interface Clock Path

Delay Elements

The Quartus Prime software does not automatically set delay elements to maximize slack in the I/O timing analysis. To close the timing or maximize slack, set the delay elements manually in the Quartus Prime settings file (`.qsf`).

Table 12. Delay Elements `.qsf` Assignments

Specify these assignments in the `.qsf` to access the delay elements.

Delay Element	`.qsf` Assignment
Input Delay Element	`set_intance_assignment –to <PIN> -name INPUT_DELAY_CHAIN <0..63>`
Output Delay Element	`set_intance_assignment –to <PIN> -name OUTPUT_DELAY_CHAIN <0..15>`
Output Enable Delay Element	`set_intance_assignment –to <PIN> -name OE_DELAY_CHAIN <0..15>`

Timing Analysis

The Quartus Prime software does not automatically generate the SDC timing constraints for the Altera GPIO IP core. You must manually enter the timing constraints.

Follow the timing guidelines and examples to ensure that the TimeQuest Timing Analyzer analyzes the I/O timing correctly.

- To perform proper timing analysis for the I/O interface paths, specify the system level constraints of the data pins against the system clock pin in the `.sdc` file.
- To perform proper timing analysis for the core interface paths, define these clock settings in the `.sdc` file:
  - Clock to the core registers
  - Clock to the I/O registers for the simple register and DDIO modes

Related Links

AN 433: Constraining and Analyzing Source-Synchronous Interfaces
Describes techniques for constraining and analyzing source-synchronous interfaces.

Single Data Rate Input Register

Figure 12. Single Data Rate Input Register

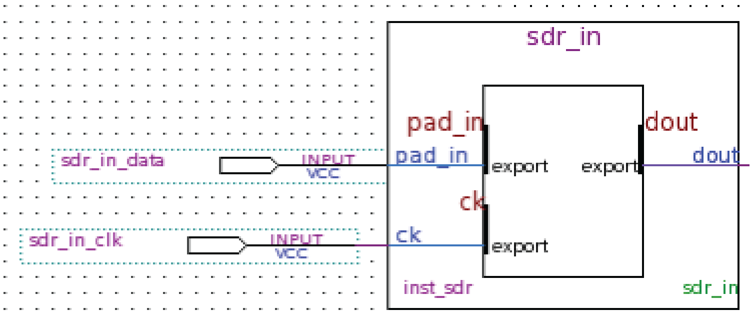

sdr_in
pad_in
dout
sdr_in_data
INPUT
VCC
pad_in
export
export
dout
ck
sdr_in_clk
INPUT
VCC
ck
export
inst_sdr
sdr_in

Table 13. Single Data Rate Input Register .sdc Command Examples

Command	Command Example	Description
`create_clock`	`create_clock -name sdr_in_clk -period "100 MHz" sdr_in_clk`	Creates clock setting for the input clock.
`set_input_delay`	`set_input_delay -clock sdr_in_clk 0.15 sdr_in_data`	Instructs the TimeQuest timing analyzer to analyze the timing of the input I/O with a 0.15 ns input delay.

Full-Rate or Half-Rate DDIO Input Register

The input side of the full-rate and half-rate DDIO input registers are the same. You can properly constrain the system by using a virtual clock to model the off-chip transmitter to the FPGA.

Figure 13. Full-Rate or Half-Rate DDIO Input Register

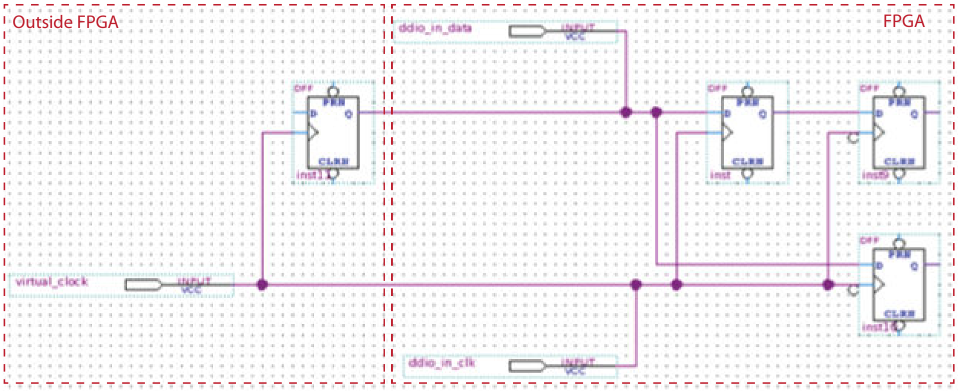

Outside FPGA
FPGA
ddio_in_data
INPUT
VCC
DFF
PRN
D
Q
CLRN
inst11
inst
inst9
inst10
virtual_clock
INPUT
VCC
ddio_in_clk
INPUT
VCC

Table 14. Full-Rate or Half-Rate DDIO Input Register `.sdc` Command Examples

Command	Command Example	Description
`create_clock`	`create_clock -name virtual_clock -period "200 MHz"` `create_clock -name ddio_in_clk -period "200 MHz" ddio_in_clk`	Create clock setting for the virtual clock and the DDIO clock.
`set_input_delay`	`set_input_delay -clock virtual_clock 0.25 ddio_in_data` `set_input_delay -add_delay -clock_fall -clock virtual_clock 0.25 ddio_in_data`	Instruct the TimeQuest Timing Analyzer to analyze the positive clock edge and the negative clock edge of the transfer. Note the `-add_delay` in the second `set_input_delay` command.
`set_false_path`	`set_false_path -fall_from virtual_clock -rise_to ddio_in_clk` `set_false_path -rise_from virtual_clock -fall_to ddio_in_clk`	Instruct the TimeQuest Timing Analyzer to ignore the positive clock edge to the negative edge triggered register, and the negative clock edge to the positive edge triggered register. *Note:* The CLK_HR frequency must be half the CLK_FR frequency. If the I/O PLL drives the clocks, you can consider using the `derive_pll_clocks` .sdc command.

Single Data Rate Output Register

Figure 14. Single Data Rate Output Register

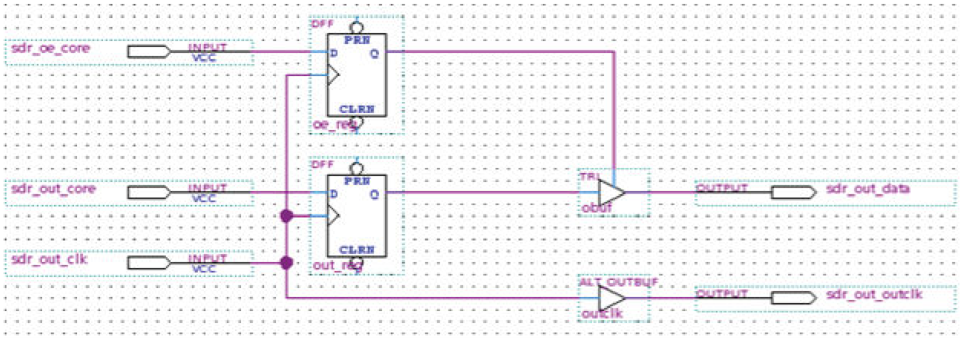


Table 15. Single Data Rate Output Register `.sdc` Command Examples

Command	Command Example	Description
`create_clock` and `create_generated_clock`	`create_clock -name sdr_out_clk -period "100 MHz" sdr_out_clk` `create_generated_clock -source sdr_out_clk -name sdr_out_outclk sdr_out_outclk`	Generate the source clock and the output clock to transmit.
`set_output_delay`	`set_output_delay -clock sdr_out_clk 0.45 sdr_out_data`	Instructs the TimeQuest Timing Analyzer to analyze the output data to transmit against the output clock to transmit.

Full-Rate or Half-Rate DDIO Output Register

The output side of the full-rate and half-rate DDIO output registers are the same.

Table 16. DDIO Output Register `.sdc` Command Examples

Command	Command Example	Description
`create_clock` and `create_generated_clock`	`create_clock -name ddio_out_fr_clk -period "200 MHz" ddio_out_fr_clk` `create_generated_clock -source ddio_out_fr_clk -name ddio_out_fr_outclk ddio_out_fr_outclk`	Generate the clocks to the DDIO and the clock to transmit.
`set_output_delay`	`set_output_delay -clock ddio_out_fr_outclk 0.55 ddio_out_fr_data` `set_output_delay -add_delay -clock_fall -clock ddio_out_fr_outclk 0.55 ddio_out_fr_data`	Instruct the TimeQuest Timing Analyzer to analyze the positive and negative data against the output clock.
`set_false_path`	`set_false_path -rise_from ddio_out_fr_clk -fall_to ddio_out_fr_outclk` `set_false_path -fall_from ddio_out_fr_clk -rise_to ddio_out_fr_outclk`	Instruct the TimeQuest Timing Analyzer to ignore the rising edge of the source clock against the falling edge of the output clock, and the falling edge of source clock against rising edge of output clock

Timing Closure Guidelines

For the Altera GPIO input registers, the input I/O transfer is likely to fail the hold time if you do not set the input delay chain. This failure is caused by the clock delay being larger than the data delay.

To meet the hold time, add delay to the input data path using the input delay chain. In general, the input delay chain is around 60 ps per step at the –1 speed grade. To get an approximate input delay chain setting to pass the timing, divide the negative hold slack by 60 ps.

However, if the I/O PLL drives the clocks of the GPIO input registers (simple register or DDIO mode), you can set the compensation mode to source synchronous mode. The Fitter will attempt to configure the I/O PLL for a better setup and hold slack for the input I/O timing analysis.

For the Altera GPIO output and output enable registers, you can add delay to the output data and clock using the output and output enable delay chains.

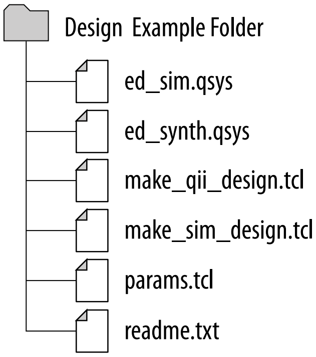
- If you observe setup time violation, you can increase the output clock delay chain setting.
- If you observe hold time violation, you can increase the output data delay chain setting.

Altera GPIO IP Core Design Examples

The Altera GPIO IP core can generate design examples that match your IP configuration in the parameter editor. You can use these design examples as references for instantiating the IP core and the expected behavior in simulations.

You can generate the design examples from the Altera GPIO IP core parameter editor. After you have set the parameters that you want, click **Generate Example Design**. The IP core generates the design example source files in the directory you specify.

Figure 15. Source Files in the Generated Design Example Directory



Note: The `.qsys` files are for internal use during design example generation only. You cannot edit these `.qsys` files.

Altera GPIO Synthesizable Quartus Prime Design Example

The synthesizable design example is a compilation-ready Qsys system that you can include in a Quartus Prime project.

Generating and Using the Design Example

To generate the synthesizable Quartus Prime design example from the source files, run the following command in the design example directory:

```
quartus_sh -t make_qii_design.tcl
```

To specify an exact device to use, run the following command:

```
quartus_sh -t make_qii_design.tcl [device_name]
```

The TCL script creates a `qii` directory that contains the `ed_synth.qpf` project file. You can open and compile this project in the Quartus Prime software.

Altera GPIO IP Core Simulation Design Example

The simulation design example uses your Altera GPIO IP core parameter settings to build the IP instance connected to a simulation driver. The driver generates random traffic and internally checks the legality of the out going data.

Using the design example, you can run a simulation using a single command, depending on the simulator that you use. The simulation demonstrates how you can use the Altera GPIO IP core.

Generating and Using the Design Example

To generate the simulation design example from the source files for a Verilog simulator, run the following command in the design example directory:

```
quartus_sh -t make_sim_design.tcl
```

To generate the simulation design example from the source files for a VHDL simulator, run the following command in the design example directory:

```
quartus_sh -t make_sim_design.tcl VHDL
```

The TCL script creates a `sim` directory that contains subdirectories—one for each supported simulation tool. You can find the scripts for each simulation tool in the corresponding directories.

IP Migration Flow for Arria V, Cyclone V, and Stratix V Devices

The IP migration flow allows you to migrate the ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, and ALTIOBUF IP cores of Arria V, Cyclone V, and Stratix V devices to the Altera GPIO IP core of Arria 10 devices.

This IP migration flow configures the Altera GPIO IP core to match the settings of the ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, and ALTIOBUF IP cores, allowing you to regenerate the IP core.

Note: Some IP cores support the IP migration flow in specific modes only. If your IP core is in a mode that is not supported, you may need to run the IP Parameter Editor for the Altera GPIO IP core and configure the IP core manually.

Migrating Your ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, and ALTIOBUF IP Cores

To migrate your ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, and ALTIOBUF IP cores to the Altera GPIO IP core, follow these steps:

1. Open your ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, or ALTIOBUF IP core in the IP Parameter Editor.
2. In the **Currently selected device family**, select **Arria 10**.
3. Click **Finish** to open the Altera GPIO IP Parameter Editor.

   The IP Parameter Editor configures the Altera GPIO IP core settings similar to the ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, or ALTIOBUF IP core settings.
4. If there are any incompatible settings between the two, select **new supported settings**.
5. Click **Finish** to regenerate the IP core.
6. Replace your ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, or ALTIOBUF IP core instantiation in RTL with the Altera GPIO IP core.

Note: The Altera GPIO IP core port names may not match the ALTDDIO_IN, ALTDDIO_OUT, ALTDDIO_BIDIR, or ALTIOBUF IP core port names. Therefore, simply changing the IP core name in the instantiation may not be sufficient.

Related Links

Input and Output Bus High and Low Bits on page 11

The high and low bits in the input or output signals are included in the `din` and `dout` input and output buses.

Guideline: Swap `datain_h` and `datain_l` Ports in Migrated IP

When you migrate your GPIO IP to the Altera GPIO IP core, you can turn on **Use legacy top-level port names** option in the Altera GPIO IP core parameter editor. However, the behavior of these ports in the Altera GPIO IP core is different than in the IP cores used for the Stratix V, Arria V, and Cyclone V devices.

The Altera GPIO IP core drives these ports to the output registers on these clock edges:

- `datain_h`—on the falling edge of `outclock`
- `datain_l`—on the rising edge of `outclock`

If you migrated your GPIO IP from Stratix V, Arria V, and Cyclone V devices, swap the `datain_h` and `datain_l` ports when you instantiate the IP generated by the Altera GPIO IP core.

Related Links

Input and Output Bus High and Low Bits on page 11
The high and low bits in the input or output signals are included in the `din` and `dout` input and output buses.

Altera GPIO IP Core User Guide Archives

If an IP core version is not listed, the user guide for the previous IP core version applies.

IP Core Version	User Guide
16.1	Altera GPIO IP Core User Guide
16.0	Altera GPIO IP Core User Guide
14.1	Altera GPIO Megafunction User Guide
13.1	Altera GPIO Megafunction User Guide

Document Revision History

Date	Version	Changes
May 2017	2017.05.08	• Updated the table listing the Altera GPIO buffer parameters to specify the conditions for the **Use bus-hold circuitry** parameter option. • Rebranded as Intel.
October 2016	2016.10.31	• Updated the input path waveform. • Added a topic describing the high and low bits in the `din` and `dout` buses.

continued...

Date	Version	Changes
August 2016	2016.08.05	• Added notes about dynamic OCT support in the Altera GPIO IP core. • Updated the topic about parameter settings to improve accuracy and clarity. • Updated the section about generating the design example. • Added a guideline topic about behavior of the legacy ports when you migrate to the Altera GPIO IP core from Stratix V, Arria V, and Cyclone V devices. • Rewrote and restructured the document to improve clarity and for ease of reference. • Changed instances of *Quartus II* to *Quartus Prime*.
August 2014	2014.08.18	• Added timing information. • Added register packing information. • Added **Use legacy top-level port names** parameter. This is a new parameter. • Added register packing information. • Replaced the term megafunction with IP core.
November 2013	2013.11.29	Initial release.