Stratix III and Cyclone III I/O WYSIWYG User Guide

Version 1.0

March 9, 2009

By

Altera Corporation

Table of Contents:

1. Overview

The Stratix III I/O is very similar to the Stratix II I/O. The most important change for Stratix III is that we have chosen to model the I/O functionality with a collection of atoms rather than a single monolithic cell. The following sections break down the descriptions of the different atoms used to represent the I/O.

As for Stratix II, the Stratix III WYSIWYG primitives should not have their functionality depend on port connectivity. This is because 3rd party simulators and formal verification tools cannot use connectivity to determine its functionality. All functionality should clearly be indicated by parameters.

Note: Cyclone III I/O WYSIWYGs are mostly the same as Stratix III. The only exception is that Cyclone III does not have a DDIO_IN primitive. You can simply replace the word Stratix III with Cyclone III for the Cyclone III family.

1.1 Differences from Stratix II

The Stratix III I/O element is similar to the Stratix II I/O element. The main difference between them is that the DDIO input latch is changed to a register to improve QDR II timing. There are other changes related to DDR and changes in delay chains that are not covered in this document.

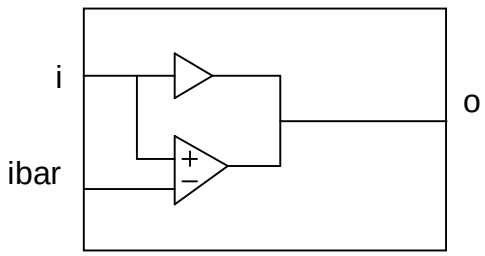
Note: Cyclone III will also use the new split IO model. The main difference between Cyclone III and Cyclone II is the addition of dedicated DDIO output and output-enable circuits.

2. The Stratix III I/O Pad

The Stratix III I/O pad does not have a WYSIWYG representation. It is simply a port on an HDL entity. Quartus II will map the port into a pad atom during technology mapping.

3. The Stratix III I/O Input Buffer

The Stratix III I/O Input Buffer WYSIWYG represents only the I/O input buffer. It can be used for a single-ended input buffer or a differential input buffer.



I/O Input Buffer

3.1 The Stratix III I/O Input Buffer Primitive

```
stratixiii_io_ibuf <I/O input buffer name>
(
      // Inputs
      .i(<buffer input>),
      .ibar(<differential buffer negative-input>),

      // Outputs
      .out(<buffer output>)
);

defparam <I/O input buffer name>.bus_hold = <bus hold mode>;
// The following is simulation-only parameter
defparam <I/O input buffer name>.differential_mode =
<differential mode>;
```

3.2 The Stratix III I/O Input Buffer – Input Ports

<*I/O input buffer name*> is the unique identifier for the I/O input buffer element. This is any identifier name which is legal for the given description language (e.g. Verilog, VHDL, AHDL, etc.). *This field is required.*

.i (<*buffer input*>**)** is the normal data input(or positive signal input for differential signals) to the I/O input buffer element. It is always fed by a pad/port atom.

.ibar(<*differential buffer negative-input*>**)** is the negative signal input of a differential signal to the I/O input buffer element. When it's connected, it is always fed by a pad/port atom.

3.3 The Stratix III I/O Input Buffer – Output Ports

.o(<*buffer output*>**)** is the output of the I/O input buffer element.

3.4 The Stratix III I/O Input Buffer – Modes

<*bus_hold_mode*> is one of *{true, false}*. This option is used to always enable the bus hold circuitry in user mode. This field is optional and defaults to *false*.

<*differential_mode*> is one of *{true, false}*. This option is used to indicate whether the input buffer is a differential input buffer or not. It's a simulation-only parameter, and used by simulator whether negative input should be used for simulation.

3.5 Port Polarities and Default Values

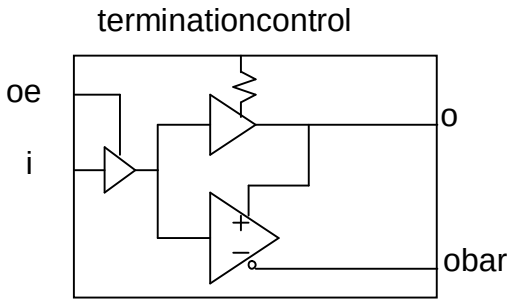
Table 1 – Port Polarity of IO Input Buffer

Signal	*Polarity*	*Programmable Inversion*
.i	Active high	No
.ibar	Active high	No

If not explicitly set in the device primitive instantiation, signals default to unconnected on the I/O input buffer element.

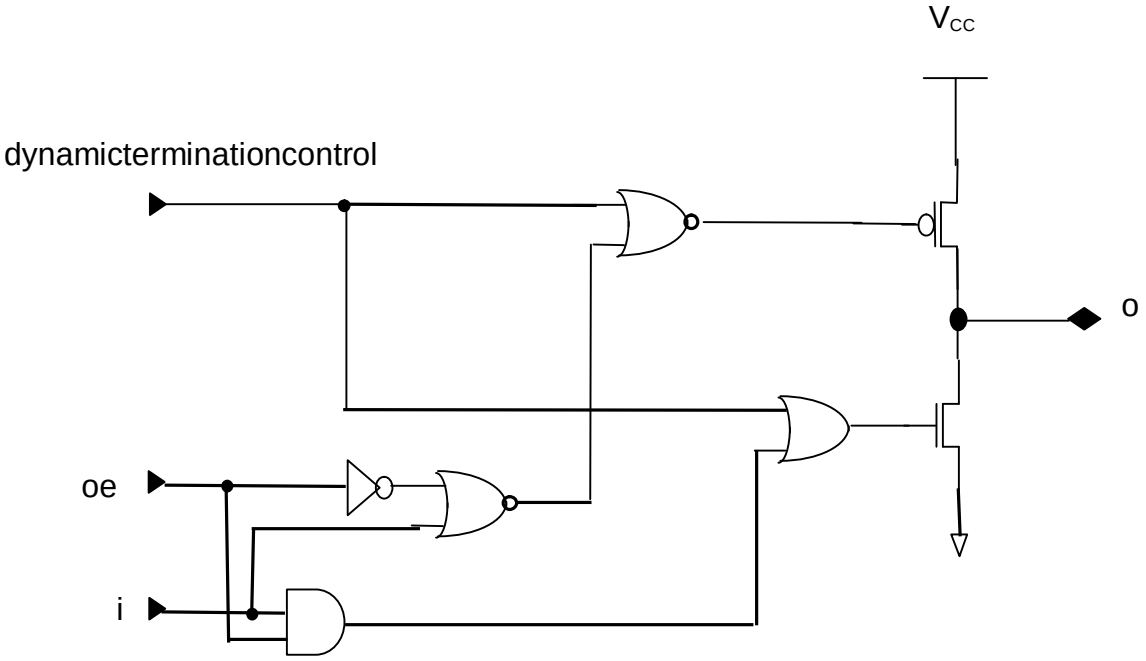
4. The Stratix III I/O Output Buffer

The Stratix III I/O Output Buffer WYSIWYG represents the I/O output buffer and the tri-state buffer. It can be used for single-ended pins or differential pins.



I/O Output Buffer

Note that in the previous diagram seriesterminationcontrol and parallelterminationcontrol are represented by the terminationcontrol signal.



Logic for the dynamicterminationcontrol input

4.1 The Stratix III I/O Output Buffer Primitive

```
stratixiii_io_obuf <I/O output buffer name>
(
      // Inputs
      .i(<buffer input>),
      .oe(<oe source>),
```

```
      .dynamicterminationcontrol(<dynamicterminationcontrol
source>),

      .seriesterminationcontrol[13..0](<Rs control bus source>),
      .parallelterminationcontrol[13..0](<Rt control bus
source>),

      // Outputs
      .o(<buffer output>),
      .obar(<differential buffer negative-output>)
);
defparam <I/O output buffer name>.open_drain_output = <open drain
mode>;
defparam <I/O output buffer name>.bus_hold = <bus hold mode>;
defparam <block name>.shift_series_termination_control = <shift
      series termination control>;
```

4.2 The Stratix III I/O Output Buffer – Input Ports

<*I/O output buffer name*> is the unique identifier for the I/O output buffer element. This is any identifier name which is legal for the given description language (e.g. Verilog, VHDL, AHDL, etc.). *This field is required.*

. **i(**<*buffer input*>**)** is the input signal to the I/O output buffer element.

.oe(<*oe source*>**)** is the output-enable source to the tri-state buffer in the I/O output buffer element.

.dynamicterminationcontrol(<*dynamicterminationcontrol source*>), receives the command to select either Rs code or Rt code from the core. When high it enables the Rt and when low it enables the Rs code. This control is meant for use in bi-directional I/Os. Rt should be disabled when driving out to maximize output performance and minimize power dissipation. Rt should only be enabled when the bi-directional I/O is receiving input. This signal can be fed by the core. This input does not need to match the polarity of its source and can be inverted.

.seriesterminationcontrol[13..0](<*control bus*>), receives the current state of the pullup and pulldown Rs control buses from a termination logic block. This signal is a dedicated route in the hardware, and does not connect to core routing.

.parallelterminationcontrol[13..0](<*control bus*>), receives the current state of the pullup and pulldown Rt control buses from a termination logic block. This signal is a dedicated route in the hardware, and does not connect to core routing.

4.3 The Stratix III I/O Output Buffer – Output Ports

.o(<*buffer output*>**)** is the output of the I/O output buffer element. It must only feed a pad.

.obar(<*differential buffer negative-output*>**)** is the negative-output of the I/O output buffer element. When it is connected, it must only feed the negative pad of a differential pad pair.

4.4 The Stratix III I/O Output Buffer – Modes

<*open_drain_mode*> is one of *{true, false}*. This field is optional and defaults to *false*.

<*bus_hold_mode*> is one of *{true, false}*. This option is used to always enable the bus hold circuitry in user mode. This field is optional and defaults to *false*.

< *shift_series_termination_control* > is one of *{true, false}*. When it's true, the block will shift the selected termination control code. This field is optional and defaults to *false*.

4.5 Port Polarities and Default Values

Table 2 – Port Polarity of IO Output Buffer

Signal	*Polarity*	*Programmable Inversion*
.i	Active high	Yes
.oe	Active high	Yes
dynamicterminationcontrol	Active high	Yes
.parallelterminationcontrol	Active high	No
.seriesterminationcontrol	Active high	No

If not explicitly set in the device primitive instantiation, signals default to unconnected on the I/O output buffer element. Output enable should be default to VCC to enable the tri-state buffer output by default.

Note that we label the input and oe signals having programmable inversion. However, the input path and oe path actually share their programmable inversions with the registers. The programmable inversion is physically located before the registers and the registers can be bypassed. So if the buffer is fed by the I/O registers, it actually can't have its inputs inverted. The placer has to know this restriction and do a proper netlist manipulation (not-gate push back) when placing the registers to the I/O register location. In certain circumstances (e.g. register has a special power up high requirement); it can't place the register in an I/O location if it feeds the output buffer with inversion.

4.6 The Cyclone III I/O Output Buffer Primitive

Cyclone III does not support parallel termination. The Cyclone III I/O output buffer primitive differs from the Stratix III primitive in the following ways:

1) The seriesterminationcontrol signal is width 16 on Cyclone III instead of 14 on Stratix III
2) There is no parallelterminationcontrol signal
3) There is no dynamicterminationcontrol signal
4) There is no shift_series_termination_control parameter

```
cycloneiii_io_obuf <I/O output buffer name>
(
      // Inputs
      .i(<buffer input>),
      .oe(<oe source>),

      .seriesterminationcontrol[15..0](<Rs control bus source>),

      // Outputs
      .o(<buffer output>),
      .obar(<differential buffer negative-output>)
);
```

```
defparam <I/O output buffer name>.open_drain_output = <open drain
mode>;
defparam <I/O output buffer name>.bus_hold = <bus hold mode>;
```

5. The Stratix III I/O Registers

Like the Stratix II & Stratix I/O cells the Stratix III I/O cell contains registers that can be used to register the OE, output data and input data in the cell. Unlike Stratix II & Stratix, these registers are represented using the same WYSIWYG primitive as the LE. See the *Stratix III FF WYSIWYG* document for a description of the register primitive and it's usage in the Stratix III I/O cell.

5.1 The Stratix III I/O Registers Restriction

The asynchronous clear (aclr) and asynchronous preset (aload) ports of I/O registers share the same resource. Therefore, the aclr and aload feature of I/O registers cannot be used at the same time. Similarly, the synchronous clear (sclr) and synchronous preset (sload) ports of I/O registers share the same resource and cannot be used at the same time. In addition, the input register, output register and output-enable register use the same input multiplexer (mux) to generate aclr or aload signals, so we cannot have distinct aclr and aload signals (GND aclr is allowed since aclr has independent GND tie-off control) among those registers. Similarly, sclr and sload share the same input mux, we can't have distinct sclr and sload signals among the 3 registers.

The power up setting can be high or low for IO registers. However, when ACLR is used, it can only support power-up low. And when ALOAD is used, it can only support power-up high. In addition, the IO register cannot implement an arbitrary ALOAD value. The ADATA must be VCC when ALOAD is used.

6. The Stratix III DDIO Output Cell

The Stratix III DDIO Output WYSIWYG represents the DDIO output portion of the I/O cell. The DDIO Output primitive can be placed in the same location as the regular output register and has the same secondary signal sharing restrictions as the output register.

The functionality of the DDIO Output primitive is to capture two data signals on the same clock edge and then pass them to the output on high and low phases of the clock, doubling the data output rate.

6.1 The Stratix III DDIO Output Primitive

```
stratixiii_ddio_out <DDIO Output name>
(
      // Inputs
      .datainlo(<DDIO low output source>),
      .datainhi(<DDIO high output source>),
      .clk(<clock source>),
      .ena(<clock-enable source>),
      .areset(<asynchronous set/reset source>),
      .sreset(<synchronous set/reset source>),
```

```
        // Outputs
        .dataout(<output of DDIO_OUT>),

        // Buried ports: placeholder to keep buried register names
        .dfflo(<buried low DDIO register>),
        .dffhi(<buried high DDIO register>),

    );
    defparam <DDIO Output name>.power_up = <register power up mode>;
    defparam <DDIO Output name>.async_mode = <register asynchronous
    mode>;
    defparam <DDIO Output name>.sync_mode = <register synchronous
    mode>;
```

6.2 The Stratix III DDIO Output -- Input Ports

<*DDIO Output name*> is the unique identifier for the DDIO output element. This is any identifier name which is legal for the given description language (e.g. Verilog, VHDL, AHDL, etc.). *This field is required.*

.**datainlo (**<*DDIO low output source*>**)** is the data input to the DDIO output element, which will be output on the low clock edge.

.**datainhi (**<*DDIO high output source*>**)** is the data input to the DDIO output element, which will be output on the high clock edge.

.clk(<*clock source*>**)** designates the clock input to the DDIO output element, which drives the clock input on the two DDIO output registers.

.ena(<*clock-enable source*>**)** designates the clock-enable signal for the DDIO output registers in the I/O element.

.areset(<*asynchronous set/reset source*>**)** designates the asynchronous set/reset signal for the DDIO output registers.

.sreset(<*synchronous set/reset source*>**)** designates the synchronous set/reset signal for the DDIO output registers.

6.3 The Stratix III DDIO Output – Output Ports

.dataout(<*output of DDIO_OUT*>**)** is the mux output of the DDIO output element.

6.4 The Stratix III DDIO Output – Buried Ports

These ports are placeholders for buried DDIO register names. They should not have fan-outs.

.dfflo(<low *buried DDIO output register*>**)** is the placeholder output to keep the name of low buried DDIO output register which sends out the data during the clock low period.

.dffhi(<high *buried DDIO output register*>**)** is the placeholder output to keep the name of high buried DDIO output register which sends out the data during the clock high period.

6.5 DDIO Output – Modes

<*power_up*> is one of *{high, low}* and describes the power-up condition of the DDIO register(s). This field is optional and defaults to low.

<*async_mode*> is one of *{clear, preset or none}*. This field is optional, and defaults to none. This determines if the .areset port clears, presets, or has no effect on the DDIO register(s). The .areset port is required if this mode is *clear* or *preset*.

<*sync_mode*> is one of *{clear, preset or none}*. This field is optional, and defaults to none. This determines if the .sreset port clears, presets, or has no effect on the DDIO register(s). The .sreset port is required if this mode is *clear* or *preset*.

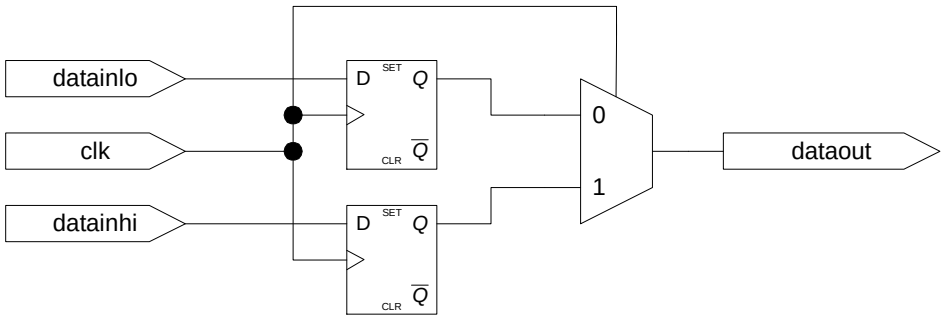
6.6 DDIO Output – Polarities and Default Values

Table 3 – Polarity of I/O Inputs

Signal	*Polarity*	*Programmable Inversion*
.datainlo	--	Yes
.datainhi	--	Yes
.clk	Rising edge	Yes
.ena	Active high	Yes
.areset	Active high	Yes
.sreset	Active high	Yes

If not explicitly set in the device primitive instantiation, signals default to unconnected on the I/O element. All Stratix III DDIO output primitive inputs have programmable inversion, and hence can be provided in either polarity.

6.7 Basic DDIO Output Diagrams



7. The Stratix III DDIO OE Cell

The Stratix III DDIO OE WYSIWYG represents the DDIO output-enable portion of the I/O cell. The DDIO OE primitive can be placed in the same location as the regular OE register and has the same secondary signal sharing restrictions as the OE register.

The functionality of the DDIO OE primitive is to generate a registered OE signal that the output drive is held at high impedance for an extra ½ clock cycle when OE goes low.

7.1 The Stratix III DDIO OE Primitive

```
stratixiii_ddio_oe <DDIO OE name>
(
      // Inputs
      .oe(<oe source>),
      .clk(<clock source>),
      .ena(<clock-enable source>),
      .areset(<asynchronous set/reset source>),
      .sreset(<synchronous set/reset source>),

      // Outputs
      .dataout(<output of DDIO_OE>),

      // Buried ports: placeholder to keep buried register names
      .dfflo(<buried low DDIO output-enable register>),
      .dffhi(<buried high DDIO output-enable register>)
);
defparam <DDIO OE name>.power_up = < register power up mode>;
defparam <DDIO OE name>.async_mode = <register asynchronous
      mode>;
defparam <DDIO OE name>.sync_mode = <register synchronous mode>;
```

7.2 The Stratix III DDIO OE – Input Ports

<*DDIO OE name*> is the unique identifier for the DDIO OE element. This is any identifier name which is legal for the given description language (e.g. Verilog, VHDL, AHDL, etc.). *This field is required.*

.oe(<*oe source*>**)** is the output enable signal for the DDIO OE element.

.clk(<*clock source*>**)** designates the clock input to the DDIO OE element, which drives the clock input on the DDIO oe registers.

.ena(<*clock-enable source*>**)** designates the clock-enable signal for the DDIO oe registers.

.areset(<*asynchronous set/reset source*>**)** designates the asynchronous set/reset signal for the DDIO OE element.

.sreset(<*synchronous set/reset source*>**)** designates the synchronous set/reset signal for the DDIO OE element.

7.3 The Stratix III DDIO OE – Output Ports

.dataout(<*output of DDIO_OE*>**)** is the output of the DDIO OE element.

7.4 The Stratix III DDIO OE – Buried Ports

These ports are placeholder for buried register names. They should not have real fan-outs.

.dfflo<*buried low DDIO OE register*>**)** is the placeholder output to keep the name of buried extended output-enable register that extends the output enable signal for extra half clock cycle.

.dffhi<*buried high DDIO OE register*>**)** is the placeholder output to keep the name of buried output-enable register

7.5 DDIO OE – Modes

<*power_up*> is one of *{high, low}* and describes the power-up condition of the DDIO register(s). This field is optional and defaults to low.

<*async_mode*> is one of *{clear, preset or none}*. This field is optional, and defaults to none. This determines if the .areset port clears, presets, or has no effect on the DDIO register(s). The .areset port is required if this mode is *clear* or *preset*.

<*sync_mode*> is one of *{clear, preset or none}*. This field is optional, and defaults to none. This determines if the .sreset port clears, presets, or has no effect on the DDIO register(s). The .sreset port is required if this mode is *clear* or *preset*.

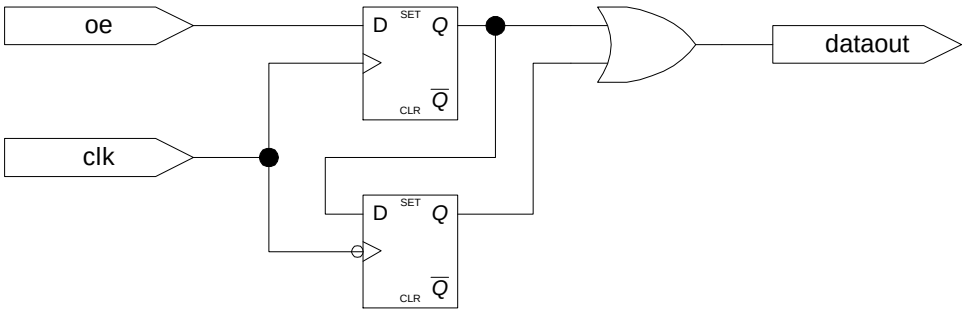
7.6 DDIO OE – Polarities and Default Values

Table 4 – Polarity of I/O Inputs

Signal	*Polarity*	*Programmable Inversion*
.oe	--	Yes
.clk	Rising edge	Yes
.ena	Active high	Yes
.areset	Active high	Yes
.sreset	Active high	Yes

If not explicitly set in the device primitive instantiation, signals default to unconnected on the I/O element. All Stratix III DDIO OE primitive inputs have programmable inversion, and hence can be provided in either polarity.

7.7 Basic DDIO OE Diagrams



8. The Stratix III DDIO Input Cell

The Stratix III DDIO input WYSIWYG represents the DDIO input portion of the I/O cell. It can be placed in the same location and has the same secondary signal sharing restrictions as the input register.

The functionality of the DDIO input primitive is to capture data on both the low and high phases of the clock and pass the two bits of data to the core synchronously with the clock signal. It can support both DDR and QDR methodologies. When DDR methodology is used (use_clkn is set to false), it only requires a single clock which is internally inverted to capture data on the negative

edge. If QDR methodology is used (use_clkn is set to true), it will require two clocks, one of which is 180 degrees out-of-phase with the other to capture data on the negative edge.

8.1 The Stratix III DDIO Input Primitive

```
stratixiii_ddio_in <DDIO Input name>
(
      // Inputs
      .datain(<output source>),
      .clk(<clock source>),
      .clkn(<clock source for regout_low register>),
      .ena(<clock-enable source>),
      .areset(<asynchronous set/reset source>),
      .sreset(<synchronous set/reset source>),

      // Outputs
      .regoutlo(<low register output>),
      .regouthi(<high register output>),


      // Buried ports: placeholder to keep buried register names
      .dfflo(<buried low DDIO register>)
);
defparam <DDIO Input name>.power_up = <register power up mode>;
defparam <DDIO Input name>.async_mode = <register asynchronous
      mode>;
defparam <DDIO Input name>.sync_mode = <register synchronous
      mode>;
defparam <DDIO Input name>.use_clkn = <use clkn for regout_low
      register>;
```

8.2 The Stratix III DDIO Input – Input Ports

<*DDR input name*> is the unique identifier for the DDIO input element. This is any identifier name which is legal for the given description language (e.g. Verilog, VHDL, AHDL, etc.). *This field is required.*

.**datain (**<*output source*>**)** is the data input to the DDIO input element.

.clk(< *clock source*>**)** designates the clock input to the DDIO input element, which drives the clock input on the regout_hi register. When use_clkn is false, a negated version of this clock also feeds the regout_low register.

.clkn(< *clock source*>**)** designates the clock input to the DDIO input element, which drives the clock input on the regout_low register. This port is optional. It is needed only for QDR application. When this port is connected, the use_clkn parameter must be set to true.

.ena(< *clock-enable source*>**)** designates the clock-enable signal for the registers in the DDIO input element.

.areset(<*asynchronous set/reset source*>**)** designates the asynchronous set/reset signal for the DDIO input element.

.sreset(<*synchronous set/reset source*>**)** designates the synchronous set/reset signal for the DDIO input element.

8.3 The Stratix III DDIO Input – Output Ports

.regoutlo(<*low registered output*>**)** is the registered output of the DDIO input element capturing data on the low clock phase.

.regouthi(<*high registered output*>**)** is the registered output of the DDIO input element capturing data on the high clock phase.

8.4 The Stratix III DDIO Input – Buried Ports

These ports are placeholder for buried register names. They should not have real fan-outs.

.dfflo<*buried low DDIO OE register*>**)** is the placeholder output to keep the name of buried input register that captures the data during the clock-low period.

Note: We don't need the buried ports for dffhi and the resynchronize registers since they are real outputs that drive into the core. Therefore, they already have the oterm names.

8.5 Stratix III DDIO Input – Modes

<*power_up*> is one of *{high, low}* and describes the power-up condition of the register(s). This field is optional and defaults to low.

<*async_mode*> is one of *{clear, preset or none}*. This field is optional, and defaults to none. This determines if the .areset port clears, presets, or has no effect on the register(s). The .areset port is required if this mode is *clear* or *preset*.

<*sync_mode*> is one of *{clear, preset or none}*. This field is optional, and defaults to none. This determines if the .sreset port clears, presets, or has no effect on the register(s). The .sreset port is required if this mode is *clear* or *preset*.

<*use_clkn*> is one of *{true or false}*. This field is optional, and defaults to false. This determines the clock source for the regout_low register. If this parameter is set to false, the regout_low register will use the negated version of the clk input as clock source. Otherwise, it will use the clkn input as clock source.

8.6 Stratix III DDIO Input – Polarities and Default Values

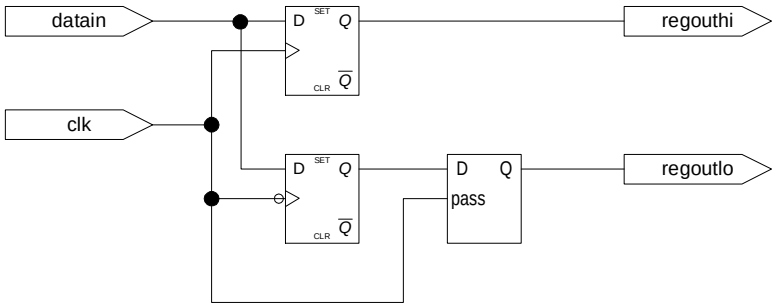
Table 5 – Polarity of I/O Inputs

Signal	*Polarity*	*Programmable Inversion*
.datain	--	Yes
.clk	Rising edge	Yes
.clkn	Rising edge	Yes
.ena	Active high	Yes
.areset	Active high	Yes
.sreset	Active high	Yes

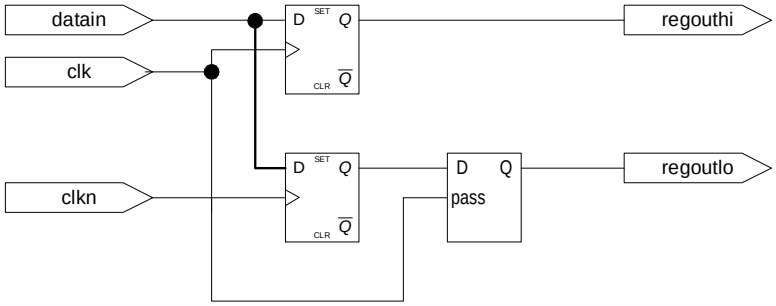
If not explicitly set in the device primitive instantiation, signals default to unconnected on the DDIO input element.

All Stratix III DDIO input element inputs have programmable inversion, and hence can be provided in either polarity.

8.7 Basic DDIO Input Diagrams



DDR



QDR