Stratix II I/O WYSIWYG Description

Version 1.4

April 6, 2007

By

Altera Corporation

Table of Contents:

1. Overview

This document describes the Stratix II I/O WYSIWYG primitive. The Stratix II I/O element is similar to the Stratix I/O element. The main differences between the Stratix II I/O and Stratix I/O are:

1) DDR memory interface on VIO.
    a. In the Stratix II family, VIO data out driver can route the delayed DQS bus signal to core. This is not available in Stratix devices.
    b. In the Stratix II family, DQS delay chain is routed through general I/O routing interface. In the Stratix family, the DQS delay chain is dedicated from DLL. This will provide more flexibility to user in designing soft core to control the DQS delay.
    c. In the Stratix II family, there is extra phase offset control to fine tune the DQS delay. The extra phase offset control is a 6-bit bus and is driven by DLL.
2) Delay chains.
    a. ZBT delay chain is removed in the Stratix II family.
    b. DQ course delay (Δtc) and PAD-to-input register delay (Δt1) have been merged into a single delay (Δt1) in the Stratix II family. Also, it provides a finer control (5-bit) of the delay chain in the Stratix II family.
    c. Stratix DQ-DQS matching delay (Δt7) has been removed. The DQ-DOS matching will be done through the new Δt1 delay.
    d. PAD-to-Core delay has finer-grained control in Stratix II devices. It now has 8 settings vs 4 settings in Stratix devices. Also, the delays have been more evenly distributed across the 8 settings.
    e. All core to I/O registers delay chains are removed (Δt4 and Δt5). The 0 hold transfer will be guaranteed by fitter through placement and detour routing.
3) Unused I/O driver (IO_DATAIN) tie-off to VCC has been removed in the Stratix II architecture. This was deemed redundant since unused drivers can be allowed to toggle as long as the IODIMs they drive are disabled.
4) OE override logic has been removed in the Stratix II architecture. The override logic provides the method of permanently enabling/disabling the OE. This was deemed unnecessary since software can achieve the same functionality through program the existing OE IOIM path.

2. The Stratix II I/O Cell

The Stratix II I/O WYSIWYG is mostly similar to Stratix I/O WYSIWIG with the additional ports modeling the DDR memory interface functionality. Other than that, new buried ports have been added to aid in preserve register names for the buried registers.

Also, for the Stratix II architecture, WYSIWYG primitive should not have its 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. Thus, we will have more parameters in the Stratix II I/O WYSIWYG than the Stratix I/O even though the new parameter is not a new functionality.

2.1 The Stratix II I/O Primitive

```
stratixii_io <I/O name>
(
      // Inputs
      .datain(<output source>),
      .ddiodatain(<DDIO output source >),
      .oe(<oe source>),
      .outclk(<output clock source>),
      .outclkena(<output clock-enable source>),
      .inclk(<input register clock source>),
      .inclkena(<input register clock-enable source>),
      .areset(<asynchronous set/reset source>),
      .sreset(<synchronous set/reset source>),
      .ddioinclk(<negative-edge DDIO input register clock
                 source>),
      .delayctrlin[5..0](<DQS delay setting source>),
      .offsetctrlin[5..0](<DQS delay offset source>),
      .dqsupdateen(<DQS update enable source>),
      .linkin(<linkin source>),
      .terminationcontrol[13..0](<control bus>),

      // Bidirectional Port
      .padio(<pad I/O source/output>),

      // Outputs
      .combout(<combinational output>),
      .regout(<input register output>),
      .ddioregout(<DDIO register output>),
      .dqsbusout(<DQS bus output>),
      .linkout(<linkout output>),

      // PE testing output port
      .dftdelayctrlout(<DQS delay setting output>),

      // Buried ports: placeholder to keep buried register names
      .dffdataout(<buried output register>),
      .dffoe(<buried output-enable register>),
      .dffddiodataout(<buried DDIO output register>),
      .dffddiooe(<buried DDIO output-enable register>),
      .dffddiodatain(<buried DDIO input register>)
);
defparam <I/O name>.operation_mode = <operation mode>;
defparam <I/O name>.ddio_mode = <DDIO mode>;
defparam <I/O name>.open_drain_output = <open drain mode>;
defparam <I/O name>.bus_hold = <bus hold mode>;

defparam <I/O name>.output_register_mode = <output register
      mode>;
defparam <I/O name>.output_async_reset = <output register
      asynchronous reset mode>;
defparam <I/O name>.output_power_up = <output register power up
      mode>;
```

```
defparam <I/O name>.output_sync_reset = <output register
      synchronous reset mode>;
defparam <I/O name>.tie_off_output_clock_enable = <tie off output
      clock enable>;

defparam <I/O name>.oe_register_mode = <oe register mode>;
defparam <I/O name>.oe_async_reset = <oe register asynchronous
      reset mode>;
defparam <I/O name>.oe_power_up = <oe register power up mode>;
defparam <I/O name>.oe_sync_reset = <oe register synchronous
      reset mode>;
defparam <I/O name>.tie_off_oe_clock_enable = <tie off oe clock
      enable>;

defparam <I/O name>.input_register_mode = <input register mode>;
defparam <I/O name>.input_async_reset = <input register
      asynchronous reset mode>;
defparam <I/O name>.input_power_up = <input register power up
      mode>;
defparam <I/O name>.input_sync_reset = <input register
      synchronous reset mode>;

defparam <I/O name>.extend_oe_disable = <extend OE disable>;

defparam <I/O name>.dqs_input_frequency = <DQS input frequency>;
defparam <I/O name>.dqs_out_mode = <DQS out mode>;
defparam <I/O name>.dqs_delay_buffer_mode = <DQS delay buffer
      mode>;
defparam <I/O name>.dqs_phase_shift = <phase offset of the
      delayed DQS signal>;
defparam <I/O name>.inclk_input = <inclk input>;
defparam <I/O name>.ddioinclk_input = <ddioinclk input>;
defparam <I/O name>.dqs_offsetctrl_enable = <DQS offset control
      enable>;
defparam <I/O name>.dqs_ctrl_latches_enable = <DQS control
      latches enable>;
defparam <I/O name>.dqs_edge_detect_enable = <dqs edge detect
  enable>;
defparam <I/O name>.gated_dqs = <gated DQS>;


// PE test only parameters
defparam <I/O name>.dft_delayctrl_select = <DQS delay setting
      control output select>;


// Hidden simulation only parameters
defparam <I/O name>.sim_dqs_intrinsic_delay = <DQS intrinsic
      delay for simulation>;
defparam <I/O name>.sim_dqs_delay_increment = <DQS delay
      increment for simulation>;
defparam <I/O name>.sim_dqs_offset_increment = <DQS offset
      increment for simulation>;
```

2.2 The Stratix II I/O Input Ports

<*I/O name*> is the unique identifier for the I/O 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 I/O element.

.ddiodatain(<*DDIO output source*>**)** is the second data input to the I/O element when it is in DDIO “output” or “bidir” mode.

.oe(<*oe source*>**)** is the output enable signal for the I/O element. This should not be specified when the I/O element is in “input” mode. It defaults to VCC (permanently enabled) in output mode, and it should not be specifically set to VCC.

.outclk(<*output clock source*>**)** designates the output-clock input to the I/O element, which drives the clock input on the output and oe registers. If this signal is used, then at least one of the output and oe registers must be in “register” mode.

.outclkena(<*output clock-enable source*>**)** designates the clock-enable signal for the output and oe registers in the I/O element. Only allowed when the .outclk port is specified and the output or oe is in “register” mode.

.inclk(<*input clock source*>**)** designates the clock input to the input register(s) (there are two input registers used when in DDIO mode) in the I/O element. Only allowed when .regout is connected.

.inclkena(<*input clock-enable source*>**)** designates the clock-enable signal for the input register in the I/O element. Only allowed when the .inclk port on the logic element is specified and **.regout** is connected.

.areset(<*asynchronous set/reset source*>**)** designates the asynchronous set/reset signal for the I/O element. Each set of registers (input, output and OE) in the I/O element can independently choose to use the .areset signal as either a clear or a preset or neither. Only allowed when at least one of the .outclk or .inclk ports on the logic element is specified, and when at least one of the input, output, or oe register’s asynchronous reset mode is “preset” or “clear.”

.sreset(<*synchronous set/reset source*>**)** designates the synchronous set/reset signal for the I/O element. Each set of registers (input, output and OE) in the I/O element can independently choose to use the .sreset signal as either a clear, a preset or neither. Only allowed when at least one of the .outclk or .inclk ports on the logic element is specified, and when at least one of the input, output or oe register’s synchronous reset mode is “preset” or “clear.”

.ddioinclk(<*negative-edge DDIO input register clock source*>**),** designates the clock input to the negative-edge DDIO input register in the I/O element. Only allowed when **.regout** is connected and **ddioinclk_input** is set to *dqsb_bus*. The negative-edge DDIO input register can be clocked by **inclk** or **ddioinclk**. When **ddioinclk_input** is set to *dqsb_bus*, the negative-edge DDIO input register is clocked by **ddioinclk**. Otherwise, it’s clocked by **inclk**.

.delayctrlin[5..0](<*DQS delay setting source*>**),** designates the DQS delay chain setting for DQS pins. It’s a 6-bit bus that specifies the delay setting. It will cause delay of the incoming signal by the amount of the specified setting. This port is optional. If unconnected, the incoming signal will not be delayed. This port is only valid on DQS pins.

.offsetctrlin[5..0](<<*DQS delay offset source*>**),** designates the DQS delay offset setting for DQS pins. It’s a 6-bit bus that specifies the delay offset. The DQS delay offset is the extra control on 1 of the 4 DQS delay chain cells to allow a finer phase control of the DQS strobe. This port is optional. This port is only valid on DQS pins and when **dqs_offsetctrl_enable** is true.

.dqsupdateen(<*DQS update enable source*>*)*, designates the enable signal for the **delayctrlin** and **offsetctrlin** latches. This port is optional. It is only valid if the **dqs_ctrl_latches_enable** is true.

.linkin(<*linkin source***>*)*,** this port is mainly provided to allow association of a differential pin pair. A negative differential pin can have the linkin port connect to the linkout port of its positive counterpart.

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

2.3 The Stratix II I/O Output Ports

.combout(<*combinational output*>**)** is the combinatorial output of the I/O element. It always feeds directly from the .padio.

.regout(<*registered output*>**)** is the registered output of the I/O element. Only allowed when the .inclk port on the logic element is specified. It is always sourced from the input register.

.ddioregout(<*DDIO registered output*>**)** is the second registered output of the I/O element when in DDIO mode. Only allowed when the .inclk port on the logic element is specified and the ddio_mode is “input” or “bidir”.

.dqsbusout(<DQS *bus output*>**)** is the delayed DQS signal from DQS pin that drives onto the dedicated DQS clock network of the DQ pins. It can also drive out to the core by stealing outputs from unused DQS pins in its DQS group. This port is optional. It is only valid if the dqs_out_mode is set to *bypass* or *delay_chain*.

.dftdelayctrlout(<DQS *delay setting output*>**)** is the output of one of the six DQS delay controls. This port is optional and for PE test only. The one being output among the six DQS delay controls is specified by parameter **dft_delayctrl_select**.

.linkout(<*linkout output***>*)*,** this port is mainly provided to allow association of a differential pin pair. A positive differential pin can have the linkout port connect to the linkin port of its negative counterpart.

2.4 The Stratix II I/O Bidirectional Ports

.padio(<*pad I/O source/output*>**)** represents the physical pad of the I/O element. A bi-directional port, this signal should be connected directly to module inputs and outputs. Logic feeding out between these ports and the top-level module outputs is illegal.

The following Example demonstrates how the .padio signal is used to connect up to input, output and bidir signals.

Example 1 – The Stratix II .padio usage in different modes

```
module foo ( input_signal, output_signal, bidir_signal )
      input input_signal;
      output output_signal;
      inout bidir_signal;
      stratixii_io input_io (.padio(input_signal), ...);
            defparam input_io.operation_mode = “input”;
            ...
      stratixii_io bidir_io (.padio(bidir_signal), ...)
            defparam bidir_io.operation_mode = “bidir”;
            ...
      stratixii_io output_io (.padio(output_signal), ...)
```

```
            defparam output_io.operation_mode = “output”;
            ...
end module
```

2.5 The Stratix II I/O Buried Ports

The following ports are placeholder to keep names for buried registers. They should not have real fan-outs.

.dffdataout(<*buried output register*>**)** is the placeholder output to keep the name of buried output register.

.dffoe(<*buried output-enable register*>**)** is the placeholder output to keep the name of buried output-enable register.

.dffddiodataout(<*buried DDIO output register*>**)** is the placeholder output to keep the name of buried DDIO output register.

.dffddiooe(<*buried DDIO output-enable register*>**)** is the placeholder output to keep the name of buried DDIO output-enable register.

.dffddiodatain(<*buried DDIO input register*>**)** is the placeholder output to keep the name of buried DDIO input register.

2.6 I/O Modes

<*operation_mode*> is one of *{input, output, bidir}*. Determines the directionality of the I/O element. *This field is required.*

<*ddio_mode*> is one of *{input, output, bidir or none}*. Determines the functionality of the DDIO circuitry in the I/O element. This field is optional and defaults to none.

If the DDIO mode is “input” or “bidir” then the I/O element is setup to receive data on both clock edges, but pass it to the core only on the rising edge of the clock. The following must be true:

- The .inclk port must be specified.
- The .combout port cannot be connected.
- The I/O element can be in “input” or “bidir” *operation_mode*.
- The .ddioregout port passes the data received on the falling edge of the input clock cycle and the .regout port passes the data received on the rising edge of the input clock cycle.

If the DDIO mode is “output” or “bidir” then the I/O element is setup to transmit data on both clock edges, but the data is latched from the core only on the rising edge of the clock. The following must be true:

- The .outclk port must be specified.
- The *output register mode* must be “register”.
- The I/O element can be in “output” or “bidir” *operation_mode*.
- On the rising edge of the output clock cycle, the .padio port transmits the data that was received at the .datain port. On the falling edge of the output

clock cycle, the .padio port transmits the data that was received at the .ddiodatain port on the rising edge of the output clock cycle.

Note that it is possible for an I/O element in *operation_mode* = “bidir” to use DDIO on its output, its input, or both. For example, an I/O with *operation_mode* = “bidir” and *ddio_mode* = “input” uses the standard output circuitry to send one value per clock, and the DDIO input circuitry to receive two values per clock.

<*open_drain_mode*> is one of *{true, false}*. This field is optional and defaults to *false*. This field can only be set to *true* when the I/O is in output or bidir mode but not input mode.

<*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*.

<*output_register_mode*> is one of *{register or none}*. This field is optional, and defaults to none. This determines if the .datain signal should be registered or not. The .outclk port is required if this mode is *register*. If the *ddio_mode* is *output* or *bidir*, then this mode must be *register*.

<*output_async_reset*> 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 output register(s). The .areset port is required if this mode is *clear* or *preset*.

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

<*output_sync_reset*> 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 output register(s). The .sreset port is required if this mode is *clear* or *preset*.

<*tie off output clock enable*> is one of *{true, false}* and determines if the clock enable for the output register, controlled by **.outclkena**, should be tied off (has no effect on the register). This field is optional and defaults to false.

<*oe_register_mode*> is one of *{register or none}*. This field is optional, and defaults to none. This determines if the .oe signal (output enable source) is registered or not. The .outclk port is required if this mode is *register*.

<*oe_async_reset*> 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 oe register. The .areset port is required if this mode is *clear* or *preset*.

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

<*oe_sync_reset*> 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 oe register. The .sreset port is required if this mode is *clear* or *preset*.

<*tie off oe clock enable*> is one of *{true, false}* and determines if the clock enable for the oe register, controlled by **.outclkena** should be tied off (has no effect on the register). This field is optional and defaults to false.

<*input_register_mode*> is one of *{register or none}*. This field is optional, and defaults to none. This determines if the .regout signal is used or not. The .inclk port is required if this mode is *register*. Although the use of input register can be inferred from the connection of .regout port, we still want to have an explicit parameter to indicate if input register is used or not since formal verification tool can’t instantiate the model based on connections.

<*input_async_reset*> 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 input register(s). The .areset port is required if this mode is *clear* or *preset*.

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

<*input_sync_reset*> 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 input register(s). The .sreset port is required if this mode is *clear* or *preset*.

<*extend OE disable*> is one of *{true, false}* and describes whether the second OE register should be used. When the second OE register is used, the output drive is held at high impedance for an extra ½ clock cycle when OE goes low. This field is optional and defaults to false.

<*DQS input_frequency*> is either set to the frequency of the DQS strobe/clock input or set to unused. This parameter should be checked to ensure it falls within a legal range. It should not be specified if dqs_out_mode is set to none. This field is optional and defaults to *unused*.

<*DQS out mode*> is one of *{none, bypass, delay_chain1, delay_chain2, delay_chain3, delay_chain4}*. This sets the number of DQS delay buffers that should feed the **dqsbusout** port. When it is set to *none*, the **dqsbusout** port should not be connected. When it is set to *bypass*, the **dqsbusout** signal will bypass the DQS delay chain. When it is set to *delay_chain1, delay_chain2, delay_chain3,* or *delay_chain4*, the **dqsbusout** signal will go through 1, 2, 3, or 4 delay buffers, respectively, that are controlled by **delayctrlin[5..0]**. This field is optional and defaults to *none* if the *dqs_input_frequency* and *dqs_phase_shift* are not specified. If *dqs_input_frequency* and *dqs_phase_shift* are specified, the default setting will depends on the *dqs_input_frequency* and *dqs_phase_shift* settings. The following is the default setting table:

Commercial *input_frequency* range (MHz)	*dqs_phase_shift* (hundreds of degrees)	*dqs_delay_buffer_mode*	Default *dqs_out_mode*
[100..175]	3000	low	delay_chain1
[100..175]	6000	low	delay_chain2
[100..175]	9000	low	delay_chain3
[100..175]	12000	low	delay_chain4
[150..230]	2250	high	delay_chain1
[150..230]	4500	high	delay_chain2
[150..230]	6750	high	delay_chain3
[150..230]	9000	high	delay_chain4
[200..310]	3000	high	delay_chain1
[200..310]	6000	high	delay_chain2
[200..310]	9000	high	delay_chain3
[200..310]	12000	high	delay_chain4
[240..400]	3600	high	delay_chain1
[240..400]	7200	high	delay_chain2
[240..400]	10800	high	delay_chain3
[240..400]	14400	high	delay_chain4

<*DQS delay buffer mode*> is one of *{low, high, none}*. Determines whether the variable delay buffers are working in low-frequency mode(RLOWNHIGH CRAM should be set to 1) or high-frequency mode(RLOWNHIGH CRAM should be set to 0). This field is optional and defaults to *low*. See above table for recommended settings.

<*phase offset of the delayed DQS signal*> is an integer between 0 through 36000. This parameter indicates the phase difference between the delayed DQS signal and the input DQS signal in units of hundredth degree (so 90 degree phase shift would be represented as 9000). This parameter is used for static timing analysis only since timing analysis can't figure out the phase shift through the delayctrlin[5:0] ports and offsetctrlin[5:0] ports like simulation can. This field is optional and defaults to *0*.

<*inclk input*> is one of *{dqs_bus, normal}* and describes whether the **inclk** input ports should be fed by the DQS_BUS where possible or not. If set to *dqs_bus*, the fitter must use the dedicated DQS bus to connect to the **inclk** port, or give an error if the placement is out of the reach of the dedicated DQS bus. If it is set to *normal*, the **inclk** port has to be routed by fitter through normal routing resource. This parameter is optional and defaults to *normal*.

<*ddioinclk input*> is one of *{dqsb_bus, negated_inclk}* and describes whether the **ddioinclk** input ports should be fed by the DQS#_BUS where possible or not. If set to *dqsb_bus*, the fitter must use the dedicated DQS# bus to connect to the **ddioinclk** port, or give an error if the placement is out of the reach of the dedicated DQS# bus. If set to *negated_inclk*, the **ddioinclk** port is ignored and the negative-edge DDIO register will be fed by a negated signal from the **inclk** port. This parameter is optional and defaults to *negated_inclk*.

<*DQS offset control enable*> is one of *{true, false}* and describes whether the **offsetctrlin** input is used. This parameter is optional and defaults to *false*.

<*DQS control latches enable*> is one of *{true, false}* and describes whether the **delayctrlin** and **offsetctrlin** inputs are latched or not. This parameter is optional and defaults to *false*.

<*DQS edge detect enable*> is one of *{true, false}* and describes whether the edge detection circuit prevent updates to the delayctrlin and offsetctrlin latches during a DQS transition. This parameter is optional and defaults to *false*. This parameter cannot be true if *dqs_ctrl_latches_enable* is set to *false*.

<*gated DQS*> is one of *{true, false}* and describes whether the DQS input signal should be gated by an AND gate fed by an I/O input register before outputting on the **dqsbusout** port. The AND is used to hold the DQS input low after the read post-amble phase to ensure that the 0 to Z transition doesn't trigger the DQ registers. To use this feature, the *input_register_mode* parameter must be set to *register*, the *input_sync_reset* parameter must be set to *clear*, and the *input_async_reset* parameter must be set to *preset*. Also, the **sreset** input port must be tied to VCC, the **dqsbusout** output must be inverted and fed back to the **inclk** input port, and the **areset** input port should be used to control the input register feeding the AND gate. The **areset** input should normally be asserted, and since it is a preset, the input register is kept high and the gate is transparent. To switch the gate off, the **areset** input should be released to allow the register to enter the low state on the last falling DQS transition. This would keep the DQS low after the post-amble for a full cycle. This parameter is optional and defaults to *false*.

<*DQS delay setting control output select*> is an integer between 0 through 5. The parameter select the delay chain setting control output on the **dftdelayctrlout** port. This parameter is optional and should not be used unless **dftdelayctrlout** is connected.

<*DQS intrinsic delay for simulation*> is the intrinsic delay of the DQS delay chain in ps. This parameter should only use by the netlist writers to pass the delay loop parameter to the simulation model. This field is optional and defaults to 0. This parameter is only used for simulation.

<*DQS delay increment for simulation*> is the delay increment of the DQS delay chain in ps. This parameter should only use by the netlist writers to pass the delay loop parameter to the simulation model. This field is optional and defaults to 0. This parameter is only used for simulation.

<*DQS offset increment for simulation*> is the offset increment of the DQS delay chain in ps. This parameter should only use by the netlist writers to pass the delay loop parameter to the simulation model. This field is optional and defaults to 0. This parameter is only used for simulation.

Note: An I/O register can use only one of asynchronous clear or preset. If an I/O register uses asynchronous clear the power-up state must be low, if an I/O register uses asynchronous preset the power-up state must be high. If neither asynchronous preset nor clear is used the power-up state can be high or low.

An IO register can use both an asynchronous present or clear and a synchronous preset or clear. The setting of the synchronous preset or clear has no effect on the power-up state of the register.

2.7 I/O Polarities and Default Values

Table 1 – Polarity of I/O Inputs

Signal	*Polarity*	*Programmable Inversion*
.datain	--	Yes
.ddiodatain	--	Yes
.oe	Active high	Yes
.outclk	Rising edge	Yes
.outclkena	Active high	Yes
.inclk	Rising edge	Yes
.inclkena	Active high	Yes
.areset	Active high	Yes
.sreset	Active high	Yes
.ddioinclk	Rising edge	Yes
.delayctrlin	--	No
offsetctrlin	--	No
.dqsupdateen	Active high	No

If not explicitly set in the device primitive instantiation, signals default to unconnected on the I/O element. Output enable will default to GND when the I/O element is in “input” mode, and VCC when it is in “output” mode. It should be left unconnected in “input” mode, and it should not be set to VCC in “output “ mode.

All Stratix II I/O element inputs have programmable inversion, and hence can be provided in either polarity.

2.8 Basic I/O Mode Diagrams

The diagrams below show the basic configurations of the I/O element, showing all possible registers in each configuration. OE and output registers may be removed by changing the appropriate register mode to “none”, while the input register is bypassed by using the .combout port instead of the .regout port.

Any register can use one of asynchronous preset or clear (but not both) by setting the register's asynchronous reset mode. The asynchronous preset or clear is fed by the .areset port. Similarly, any register can use one of synchronous preset or clear by setting the register's synchronous reset mode. The synchronous preset or clear is fed by the .sreset port.

Any diagram showing a TRI primitive may have an OPNDRN primitive fed by, or in place of, the TRI primitve by setting open_drain_output = "true."

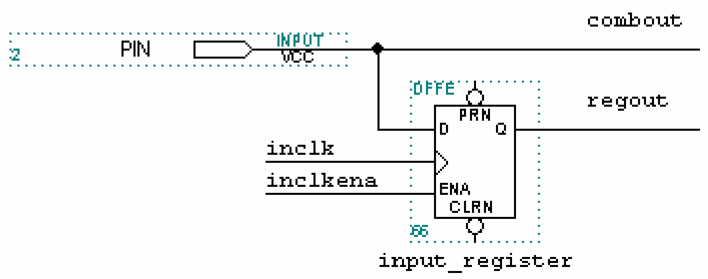

Figure 7: operation_mode = "input"; either or both of **.combout** and **.regout** can be used. When .regout is used, input_register_mode = "register". Otherwise, input_register_mode = "none".

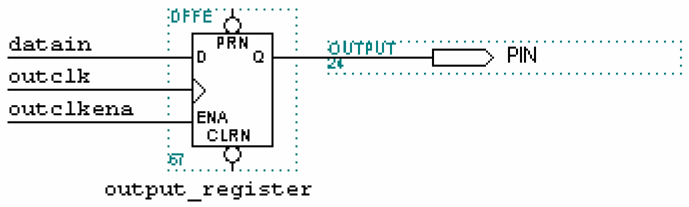

Figure 8: operation_mode = "output"; output_register_mode = "register." This is the non tri-stated version of the output mode (.oe is disconnected).

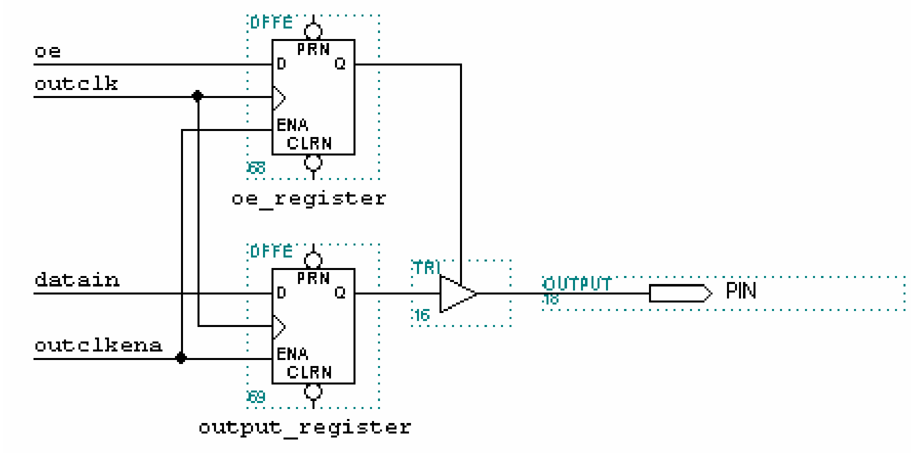

Figure 9: operation_mode = “output”; output_register_mode = “register”; oe_register_mode = “register.” This is the tri-state version of the output mode (**.oe** is connected).

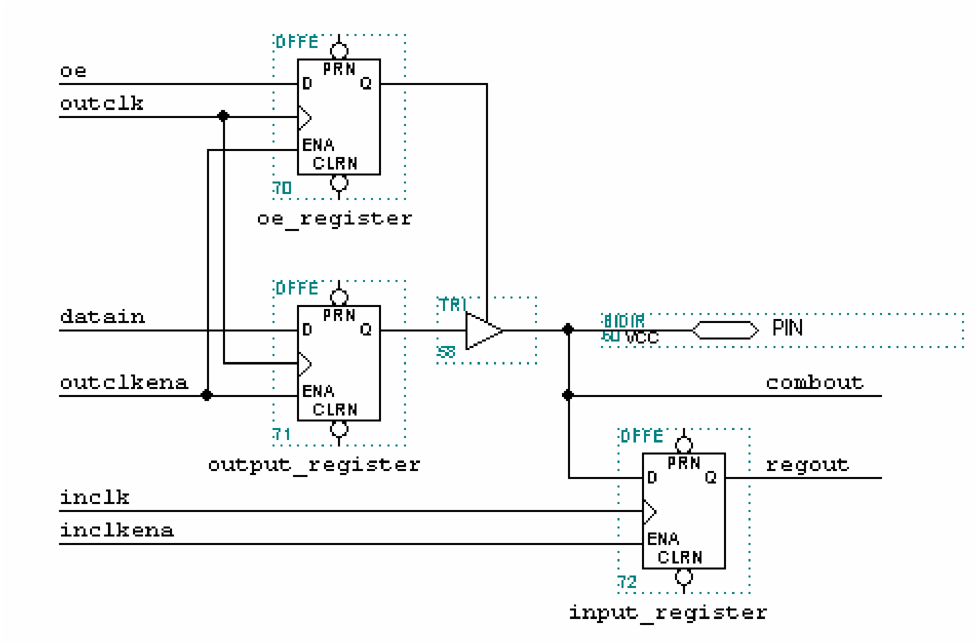

Figure 10: operation_mode = “bidir”; output_register_mode = “register”; oe_register_mode = “register”; either or both of .combout and .regout can be used. When .regout is used, input_register_mode = “register”. Otherwise, input_register_mode = “none”.

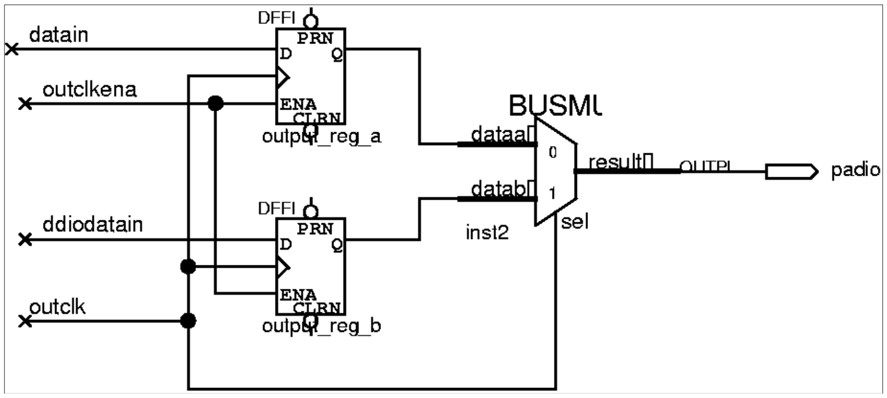

Figure 11: operation_mode = “output”; output_register_mode = “register”; ddio_mode = “output.”

3. Backward Compatiblity with Stratix Devices

The mapping of Stratix I/O ports and parameters to Stratix II I/O ports and parameters should be relatively simple. The sim_dll_phase_shift parameter is replaced by dqs_phase_shift. Also, the new dqs_out_mode should be set according to the Stratix I/O sim_dll_phase_shift parameter as follows: { unused -> none, 0 -> bypass, 72 or 90 -> delay_chain }. The sim_dqs_input_frequency has been renamed to dqs_input_frequency.

The Stratix DLL delayctrlout->Stratix I/O delayctrlin connection has now been expanded to a 6-bit bus in the Stratix II architecture. And the Stratix II I/O dqsbusout port should adopt the fanouts of Stratix I/O combout port if sim_dll_phase_shift parameter of the Stratix I/O is not “unused”. Otherwise, the dqsbusout port should be left unconnected.

The rest of new ports in the Stratix II I/O can be left unconnected. The rest of new parameters in the Stratix II I/O can be left at default settings.