Cyclone II MAC WYSIWYG Description

Version 1.1
October 16, 2006

by
Altera Corporation

Author:	Altera Corporation
System:	
Subsystem:	
Keywords:	Cyclone II, MAC, WYSIWYG, DSP, Multiplier

Table of Contents:

1 MAC WYSIWYG primitive

This document describes the WYSIWYG primitive for the Cyclone II MAC. The Cyclone II MAC is a subset of the Stratix MAC: only simple 9-bit and 18-bit multipliers are supported. Consequently, since the Stratix II MAC is a superset of the Stratix MAC, the Cyclone II MAC can also be said to be a subset of the Stratix II MAC. See yeager_mac_wys_doc.doc for more info on the Stratix MAC WYSIWYG, and armstrong_mac_wys.doc for more info on the Stratix II MAC WYSIWYG.

The following is a summary of the differences in the Cyclone II MAC, compared to the Stratix MAC:

1. Only simple multiplier mode is supported (i.e. MAC_OUT can only be OUTPUT_ONLY mode)
2. One clk/ena/aclr signal set instead of four
3. No scan chains
4. No output register in the MAC_MULT

The following diagram is an overview of the Megallan MAC WYSIWYGs (one MAC_MULT feeding one MAC_OUT).

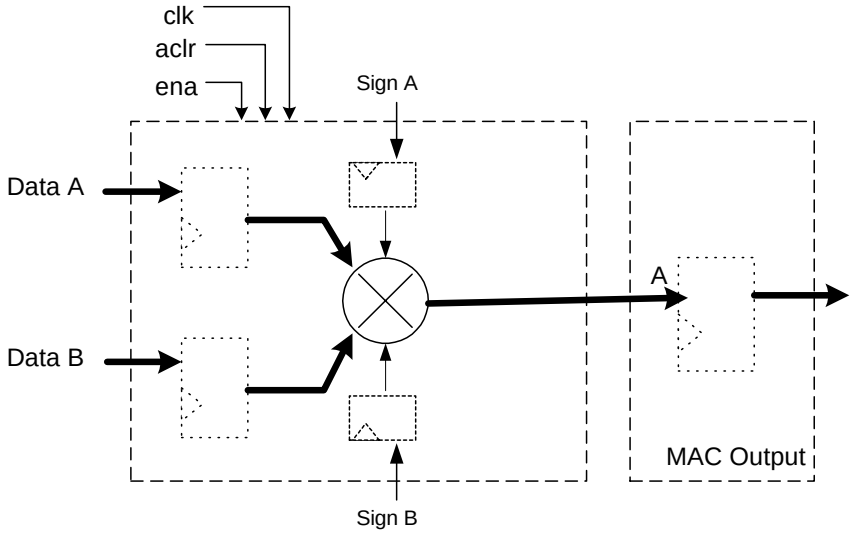

Figure 1: Cyclone II MAC Slice

1.1 Cyclone II MAC Multiplier

1.1.1 MAC Multiplier Cell Primitives

```
Cyclone II_mac_mult <mac_mult_name>
(
      .dataa(<data_a source bus>),
      .datab(<data_b source bus>),
      .signa(<signed/unsigned a source>),
      .signb(<signed/unsigned b source>),
      .clk(<clock source>),
      .aclr(<asynchronous clear source>),
```

```
        .ena(<clock enable source>),

        .dataout(<data output bus>),

        .observabledataa_regout(<observable bus for dataa>),
        .observabledatab_regout(<observable bus for datab>),

        .observablesigna_regout(<observable port for signa>),
        .observablesignb_regout(<observable port for signb>),
    );
    defparam <mac_mult_name>.dataa_width = <dataa width>;
    defparam <mac_mult_name>.datab_width = <datab width>;
    defparam <mac_mult_name>.dataa_clock = <clk for data_a>;
    defparam <mac_mult_name>.datab_clock = <clk for data_b>;
    defparam <mac_mult_name>.signa_clock = <clk for sign_a>;
    defparam <mac_mult_name>.signb_clock = <clk for sign_b>;

    // simulation-only parameters
    defparam <mac_mult_name>.dataout_width = <width of dataout port>;
```

1.1.2 MAC Multiplier Cell Input Ports

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

.**dataa(**<*data_a source bus*>**), .datab(**<*data_b source bus*>**):** are the two input buses for the multiplier. The inputs can be registered by specifying the dataa/b_clock option. Each input can be of a different width, however internally this is implemented by 0 padding the low-order unused bits. This port is required.

.signa(<*signed/unsigned a source*>***),* .signb(**<*signed/unsigned b source*>***)*:** designates the two sign signals for each input of the multiplier. High implies the input is a signed integer (i.e. MSB bit is used as sign bit), low implies the input is an unsigned integer. Each input can have a different signed/unsigned value since the multiplier will take this into account. This can be set to a constant value by tying the input to VCC or GND. The signal can also be registered by specifying the signa/b_clock option. This port is not required and defaults to VCC if left unconnected.

.clk(<*clock source* >**):** designates the single clock input that drives the multiplier. Every register bank in the multiplier can independently choose whether to be registered or not, and are all fed by this single clock if used. This port must be connected if it is specified to be used by one of the parameters, otherwise it must not be connected.

.aclr(<*aclr source*>**):** designates the single asynchronous clear input that drives the multiplier. Every register bank in the multiplier is fed by this input. This port must be connected if it is specified to be used by one of the parameters, otherwise it must not be connected.

.ena(<*clock enable source* >**):** designates the single clock enable input that drives the multiplier. This port must be connected if it is specified to be used by one of the parameters, otherwise it must not be connected.

1.1.3 MAC Multiplier Cell Output Ports

.dataout(<*data output bus*>**)**: the output bus of the multiplier (could be registered if using output registers). Note that this bus can only feed the data inputs of a mac_out atom through dedicated

internal routing – it is not visible external to the MAC. However, since the MAC_OUT is only a wire, the outputs of the MAC_MULT are technically immediately visible. This output must be connected – it cannot be left unconnected. The outputs can be registered by specifying the output_clock option.

.observabledataa_regout(<*observable bus*>***), .observabledatab_regout(**<*observable bus*>***):** assigns names for the observable bus for the data inputs. The observable bus will only be created if its corresponding data input bus is registered. These cannot be connected to other routing and are used for simulation and timing purposes only. By connecting wires to the observable bus, the user is able to rename the observable bus to the name of the connected wires.

.observablesigna_regout(<*observable port*>***), .observablesignb_regout(**<*observable port*>**):** assigns a name for the observable port for the two sign ports. The observable port will only be created if its corresponding port is registered. These cannot be connected to other routing and are used for simulation and timing purposes only. By connecting wires to the observable ports, the user is able to rename the observable ports to the name of the connected wires. **This feature has not been implemented yet for these ports.**

1.1.4 MAC Multiplier Cell Modes

<*dataa width*>, <*datab width*> is an integer in the range *{1, 2, …, 18}*. It specifies the width of each operand of the multiplier. The combination of the dataa width and the datab width implies the output width: the output width is the sum of both input widths (e.g. for dataa_width = 13 and datab_width = 15, dataout_width = 28). Internally there are only 9-bit and 18-bit multipliers, so multipliers of smaller width are implemented by tying the unused low-order bits to 0 (not the high-order bits since the MSB bit is still needed when doing signed multiplication). *This field is required.*

<*clk for …*> is one of *{none, 0}*. Specifies whether a given register is used or not. If "none" is specified, then the corresponding signal is not registered. Otherwise, the corresponding signal will be registered and will be driven by the clk, ena and aclr signals, which must be connected. This field is not required and defaults to "none" if unspecified.

<*width of dataout port*> specifies the width of the dataout port. This is redundant information since the width is derived from the input port widths, but is needed for sim models that cannot do the calculation. This is only a simulation parameter and is ignored by the compiler.

1.1.5 MAC Multiplier Cell Polarities and Default Values

All signals are active high, except for the aclr signal which is active low. All inputs have programmable inverts.

1.2 Cyclone II MAC Output

1.2.1 MAC Output Cell Primitives

```
Cyclone II_mac_out <mac_out_name>
(
        .dataa(<data_a source bus>),
        .clk(<clock source>),
        .aclr(<asynchronous clear source>),
        .ena(<clock enable source>),
```

```
        .dataout(<data output bus>)
    );
    defparam <mac_out_name>.dataa_width = <dataa width>;
    defparam <mac_mult_name>.output_clock = <clk for output>;

    // simulation-only parameters
    defparam <mac_mult_name>.dataout_width = <width of dataout port>;
```

1.2.2 MAC Output Cell Input Ports

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

.**dataa(**<*data_a source bus*>**):** is the input data bus for the output block, which feeds the dataout bus through an output register. This input can only be driven by the output of a mac_mult atom through dedicated internal routing – it is not visible external to the MAC. This input is required.

.clk(<*clock source* >**):** designates the single clock input that drives the multiplier. The output register in the multiplier can independently choose whether to be registered or not, and is fed by the same clock as other registers in the multiplier if used. This port must be connected if it is specified to be used by one of the parameters, otherwise it must not be connected.

.aclr(<*aclr source*>**):** designates the single asynchronous clear input that drives the multiplier. Every register bank in the multiplier is fed by this input. This port must be connected if it is specified to be used by one of the parameters, otherwise it must not be connected.

.ena(<*clock enable source* >**):** designates the single clock enable input that drives the multiplier. This port must be connected if it is specified to be used by one of the parameters, otherwise it must not be connected.

1.2.3 MAC Output Cell Output Ports

.dataout(<*data output bus*>**)**: the output bus of the MAC output cell. The MAC_OUT is just a wire so the dataout is just fed by the dataa bus and has the same width as the dataa bus.

1.2.4 MAC Output Cell Modes

<*dataa width*> is an integer in the range {1, 2, …, 36}. It specifies the width of the input ports for this output cell. This width must be the same as the MAC_MULT's output width. This parameter also specifies the output width. *This field is required.*

<*clk for …*> is one of *{none, 0}*. Specifies whether a given register is used or not. If "none" is specified, then the corresponding signal is not registered. Otherwise, the corresponding signal will be registered and will be driven by the clk, ena and aclr signals, which must be connected. This field is not required and defaults to "none" if unspecified.

<*width of dataout port*> specifies the width of the dataout port. This is redundant information since the width is derived from the input port widths, but is needed for sim models that cannot do the calculation. This is only a simulation parameter and is ignored by the compiler.

1.2.5 MAC Output Cell Polarities and Default Values

All inputs are active high. MAC_MULT data inputs have programmable inverts, while MAC_OUT data inputs do not have programmable inverts.