Supported Devices

You can simulate the ASMI block in your design for the following devices:

- Arria V, Arria V GZ, Arria 10
- Cyclone V
- Stratix V
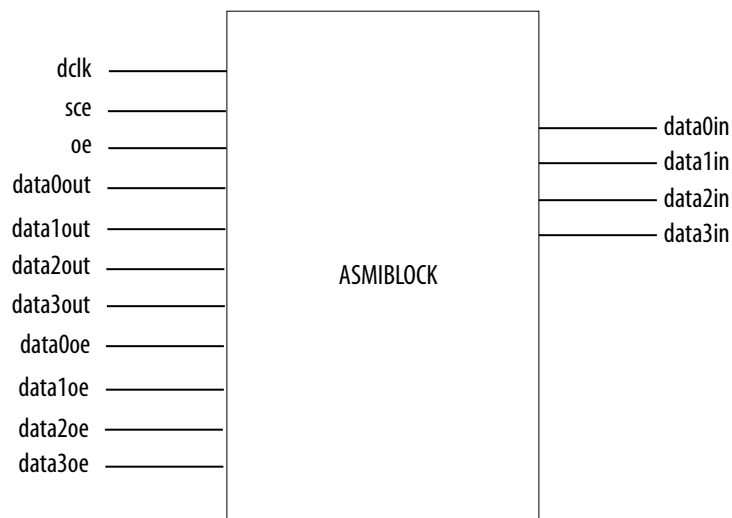
Overview

In the Quartus II software, the supported FPGA devices support the ASMI block, allowing you to access your EPCS/EPCQ flash devices.

Block Diagrams for ASMI Block

The Quartus II software allows the instantiation of different ASMI block atoms according to the core interface in your respective FPGA devices.

Figure 1: ASMI Block Diagram for Arria V, Arria V GZ, Cyclone V, and Stratix V Devices
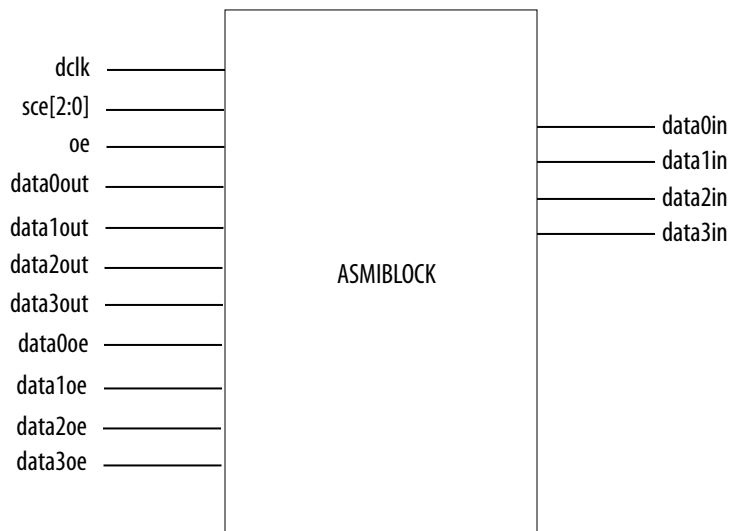
ISO 9001:2008 Registered

Figure 2: ASMI Block Diagram for Arria 10 Devices

Signals for ASMI Block

This table lists the ASMI block signals.

Table 1: Signals for ASMI Block

Signal	Direction	Width (bits)	Description
dclk	Input	1	Clock signal from your FPGA design to the external DCLK pin through the ASMI hard logic.
sce	Input	3	Chip select signal from your FPGA design to the external nCSO pin through the ASMI hard logic.
oe	Input	1	Active-low signal to enable DCLK and nCSO pins to reach the flash. The DCLK and nCSO are fixed to high when you set this signal to high, blocking the connection between FPGA and flash.
data0out data1out data2out data3out	Input of the ASMI block, which outputs the data from your FPGA design to the AS data pin	4	Control signal from your FPGA design to the AS data pin for sending data into the serial configuration device. If you want to connect your Arria 10, Arria V, Arria V GZ, Cyclone V, or Stratix V device to the EPCS device, Altera recommends setting the data out ports to the following: • data0out = FPGA design data to the EPCS through the AS_DATA0 pin. • data1out = 1'b0 • data2out = 1'b1 • data3out = 1'b1

Signal	Direction	Width (bits)	Description
data0oe data1oe data2oe data3oe	Input	4	Controls data pin either as input or output because the dedicated pins for active serial data is bidirectional. To set the AS data pin as input, set the desired data pin oe to 0. To set the AS data pin as output, set the desired data pin oe to 1. If you want to connect your Arria 10, Arria V, Arria V GZ, Cyclone V or Stratix V device to the EPCS device, then set the data pin oe to the following: • data0oe = 1'b1 • data1oe = 1'b0 • data2oe = 1'b1 • data3oe = 1'b1
data0in data1in data2in data3in	Output of the ASMI block, which receives input from the AS data pin and outputs to your FPGA design	4	Signal from the AS data pin to your FPGA design. If you want to connect your Arria 10, Arria V, Arria V GZ, Cyclone V or Stratix V device to the EPCS device, then set the data in pin to the following: • data0in = don't care • data1in = EPCS device data to your FPGA design through the AS_DATA1 pin. • data2in = don't care • data3in = don't care

WYSIWYG for ASMI Block

If you want to use ASMI in user mode, then you must instantiate the WYSIWYG of the ASMI block in your design. Instantiating the WYSIWYG of the ASMI block in your design allows you to access the active serial pins from the FPGA user design.

Example 1: Example of Verilog WYSIWYG for Arria V, Arria V GZ, Cyclone V, and Stratix V ASMI Block

```
<device>_asmiblock <name>
(
    .dclk(<clock source from user design>),
    .sce(<1 bit SCE from user design>),
    .oe(<output enable from user design>),
    .data0out(<AS_DATA0 from user design>),
    .data1out(<AS_DATA1 from user design>),
    .data2out(<AS_DATA2 from user design>),
    .data3out(<AS_DATA3 from user design>),

    .data0oe (<OE of data0out from user design>),
    .data1oe (<OE of data1out from user design>),
    .data2oe (<OE of data2out from user design>),
    .data3oe (<OE of data3out from user design>),
```

```
        .data0in(<AS_DATA0 to user design>),
        .data1in(<AS_DATA1 to user design>),
        .data2in(<AS_DATA2 to user design>),
        .data3in(<AS_DATA3 to user design>)
);
defparam <name>.enable_sim = "false";
```

Example 2: Example of VHDL WYSIWYG for Arria V, Arria V GZ, Cyclone V, and Stratix V ASMI Block

```
component <device>_asmiblock
    generic(
      enable_sim :      string    :=  "false"
    );
    port(
        dclk     :    in    std_logic;
        sce      :    in    std_logic;
        oe       :    in    std_logic;
        data0out :    in    std_logic;
        data1out :    in    std_logic;
        data2out :    in    std_logic;
        data3out :    in    std_logic;
        data0oe  :    in    std_logic;
        data1oe  :    in    std_logic;
        data2oe  :    in    std_logic;
        data3oe  :    in    std_logic;
        data0in  :    out  std_logic;
        data1in  :    out  std_logic;
        data2in  :    out  std_logic;
        data3in  :    out  std_logic
    );
  end component;
```

Example 3: Example of Verilog WYSIWYG for Arria 10 ASMI Block

```
<device>_asmiblock <name>
(
    .dclk(<clock source from user design>),
    .sce(<3 bit SCE from user design>),
    .oe(<output enable from user design>),
    .data0out(<AS_DATA0 from user design>),
    .data1out(<AS_DATA1 from user design>),
    .data2out(<AS_DATA2 from user design>),
    .data3out(<AS_DATA3 from user design>),

    .data0oe (<OE of data0out from user design>),
    .data1oe (<OE of data1out from user design>),
    .data2oe (<OE of data2out from user design>),
    .data3oe (<OE of data3out from user design>),

    .data0in(<AS_DATA0 to user design>),
    .data1in(<AS_DATA1 to user design>),
    .data2in(<AS_DATA2 to user design>),
    .data3in(<AS_DATA3 to user design>)
);
defparam <name>.enable_sim = "false";
```

Example 4: Example of VHDL WYSIWYG for Arria 10 ASMI Block

```
component <device>_asmiblock
    generic(
        enable_sim :      string    :=  "false"
    );
    port(
        dclk      :   in    std_logic;
        sce       :   in    std_logic_vector(2 downto 0);
        oe        :   in    std_logic;
        data0out  :   in    std_logic;
        data1out  :   in    std_logic;
        data2out  :   in    std_logic;
        data3out  :   in    std_logic;
        data0oe   :   in    std_logic;
        data1oe   :   in    std_logic;
        data2oe   :   in    std_logic;
        data3oe   :   in    std_logic;
        data0in   :   out   std_logic;
        data1in   :   out   std_logic;
        data2in   :   out   std_logic;
        data3in   :   out   std_logic
    );
end component;
```

Simulating the ASMI Block in Your Design

To simulate the ASMI block in your design with a flash simulation model, set the `enable_sim` parameter from `false` to `true`.

Then, create a wrapper with the same module name in the simulation project.

Example: Simulating the ASMI Block in a Stratix V Device

To simulate the ASMI block in a Stratix V device, follow these steps:

1. Create a design which instantiates the ASMI block, and set the `enable_sim` parameter to `true`.

Figure 3: Sample Code to Instantiate the ASMI Block

```
stratixv_asmiblock dut
(
    .dclk(),
    .sce(),
    .oe(),
    .data0out(),
    .data1out(),
    .data2out(),
    .data3out(),
    .data0oe (),
    .data1oe (),
    .data2oe (),
    .data3oe (),
    .data0in(),
    .data1in(),
    .data2in(),
    .data3in()
);
defparam dut.enable_sim = "true";
```

2. Compile the design in the Quartus II software and ensure that the design does not contain any syntax error.
3. Create a wrapper to connect a third-party flash simulation model to the ASMI block through the `asmi_sim_model` module. Note that the interface for the `asmi_sim_model` module varies according to devices.

Figure 4: Sample Code for Wrapper to Connect the Flash Model with the ASMI Block

```
'timescale 1ps/1ps

module asmi_sim_model(
    DCLK,
    nCS0,
    AS_DATA0,
    AS_DATA1,
    AS_DATA2,
    AS_DATA3
);

in put      DCLK, nCS0;
inout       AS_DATA0, AS_DATA1, AS_DATA2, AS_DATA3;

// third party flash simulation module
N25Qxxx device_model(
    .S(nCS0),
    .C(DCLK),
    .HOLD_DQ3(AS_DATA3),
    .DQ0(AS_DATA0),
    .DQ1(AS_DATA1),
    .Vcc('d3300),
    .Vpp_W_DQ2(AS_DATA2)
);

endmodule
```

4. In a simulation project, compile the following files to your working folder:

- Flash simulation model
- asmi_sim_model design wrapper
- **quartus/eda/sim_lib/stratixv_atoms.v** or **quartus/eda/sim_lib/stratixv_atoms.vhd** (if you are not using the ModelSim® -Altera software)

   Note: The ModelSim-Altera software contains all device atom libraries, so no compilation is needed.

5. Run simulation. The FPGA design is connected to the flash simulation model via the ASMI interface.

Document Revision History

This table lists the revision history for this application note.

Table 2: Document Revision History

Date	Version	Changes
August 2015	2015.08.03	• Added example of VHDL WYSIWYG for Arria V, Arria V GZ, Cyclone V, Stratix V, and Arria 10 ASMI Block. • Added working folder path for VHDL. • Removed statement stating compiling and simulating the active serial memory interface (ASMI) block is available from Quartus II version 14.0 onwards.
December 2014	2014.12.15	Initial release.