# Connecting an AMBA™ 2.0 AHB Slave to an AMBA 3 AXI™ Subsystem

By David Ptak, Corporate Applications Engineer

## Introduction

With the transition of new designs to AMBA™ 3 AXI™, there is considerable existing AMBA 2.0 AHB-based intellectual property (IP) which continues to be very useful. To quickly integrate legacy AHB designs into AXI subsystems, without requiring time-consuming and risky re-design, Synopsys provides an efficient and reliable method of reuse.

This application note focuses on a specific example of connecting one AHB slave to an AXI subsystem, however, any AHB-based subsystem could be connected in a similar way. It reviews the steps required to integrate multiple DesignWare® Synthesizable IP using coreAssembler, as well as how to include non-DesignWare IP in the subsystem. Finally, it discusses various integration considerations, including how to configure the DesignWare Synthesizable IP used in this type of subsystem.

All DesignWare IP components used in this application note, as well as coreAssembler usage, are made available by a standard DesignWare Library license.

## Example System Block Diagram

Figure 1 illustrates an example subsystem where an AXI interconnect connect to an AHB interconnect, using DW_axi_x2h.

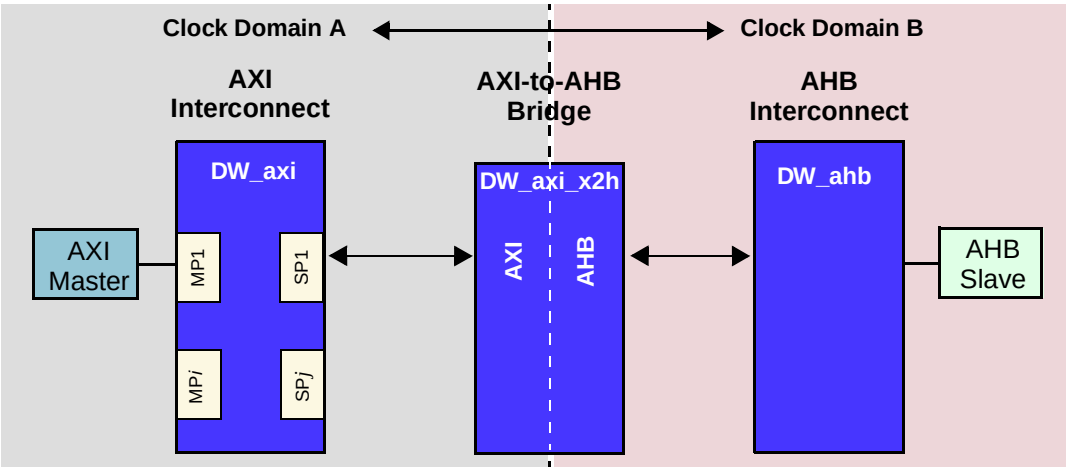


**Figure 1: Example AHB Slave connected to AXI Subsystem**

The components illustrated in Figure 1 on page 1 are described as follows:

- **AXI Master** – A component that generates new transactions to AXI slaves (for example, a microprocessor)
- **AXI Interconnect** (DW_axi) – DesignWare Synthesizable IP component that routes AXI requests/responses between AXI masters and AXI slaves
- **AXI-to-AHB Bridge** (DW_axi_x2h) – DesignWare Synthesizable IP component that processes AXI requests, translates them to the AHB bus and returns the AHB response to the AXI response channels
- **AHB Interconnect** (DW_ahb) – DesignWare Synthesizable IP component that is responsible for AHB bus arbitration, address decoding, and data multiplexing between AHB masters and AHB slaves
- **APB Slave** – A component that responds to transactions from DW_apb (for example, an interrupt controller or UART)

# Creating the Subsystem

When you use the Synopsys coreAssembler tool with DesignWare Synthesizable IP, you can construct and simulate any single- or multi-layer AMBA-based subsystem that you can conceive. coreAssembler is a highly flexible, integrated and feature-rich design environment that allows you to select, configure, interconnect, simulate, and synthesize IP. This application note describes the specific coreAssembler steps required to connect an AHB slave to an AXI subsystem.

## Required Tools, Components, and Licenses

- coreAssembler 5.1.1 or later (5.2 or later recommended)
- DesignWare Synthesizable IP for AMBA (*more information*)
  - DW_axi
  - DW_axi_x2h
  - DW_ahb
- DesignWare license

For detailed information about downloading and installing DesignWare Library Synthesizable IP, *click here*.

If you are not already familiar with using coreAssembler with DesignWare Synthesizable IP, *click here*. In that document, the tutorials in Chapter 6 provide step-by-step details of the process in the following subsection.

## coreAssembler Steps

Once you have coreAssembler and DesignWare Library Synthesizable IP installed on your system, you can get started with creating your subsystem. In coreAssembler, create a new workspace, then complete the following instructions:

### Add Subsystem Components and Interface Configuration

With any new coreAssembler workspace, you see a blank schematic, an Activity List, and a console for text input/output. The first activity to complete in the Activity List is Add Subsystem Components.

1. Insert components.

   The minimum set of components required are DW_axi, DW_axi_x2h, and DW_ahb. Use the menu item for **Schematic -> Add New Component**.

2. For each added component, right-click on each and use **Change Connection** to verify the interface connections are correct. Make sure the DW_axi_x2h component is connected to both DW_axi and DW_ahb. Any interface marked in red must be connected or marked as unused.

3. For each added component, right-click on each and use **Edit Interface Parameters** to verify the values of the parameters that affect that component's interfaces. Certain components (for example, DW_axi_x2h) do not have any interface parameters that are configurable, because their interfaces are automatically configured based on how attached components (for example, DW_axi and DW_ahb) are configured.

4. For any AXI masters or AHB slaves, if you are not using a DesignWare Synthesizable IP for those components, you can use **Export Interface**. You can export an AXI Master interface from DW_axi and an AHB Slave interface from DW_ahb. Export Interface creates top-level ports in your subsystem RTL for all signals in the interface, which you can then use to connect to the RTL for your AXI master and AHB slave RTL outside coreAssembler. Alternatively, with a coreAssembler license you can import your own IP RTL directly into coreAssembler using Import Component instead of Export Interface.

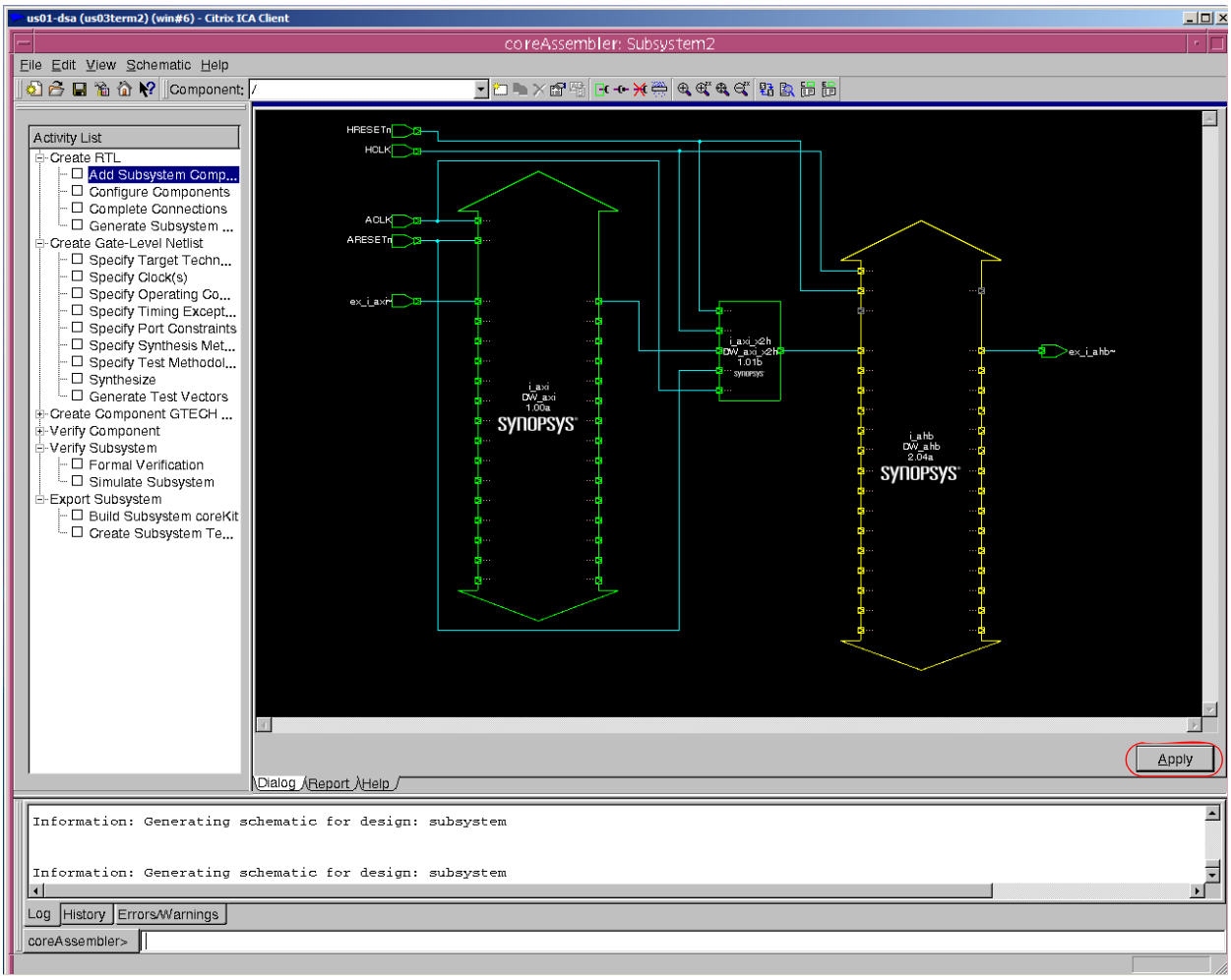


**Figure 2: coreAssembler Example Subsystem**

After completing these steps, click Apply in the lower right corner of the schematic. If you've configured a legal subsystem, the Add Subsystem Components activity will complete successfully. If the subsystem has errors, correct those errors and click Apply again.

## Configure Components

To move to the next activity in coreAssembler, click on Configure Components in the Activity List. For this activity, you configure each individual component separately. For each component, if you are not sure what a specific configuration parameter controls, details on that parameter can be found in the databook for that component. The databook for every component in the subsystem is available from the **Help** menu. Another way to retrieve brief information on a parameter is to right-click on the parameter and choose **What's This?** from the menu.

Further discussion of specific configuration details for this subsystem are discussed in Integration Considerations later in this application note.

To complete this activity, click Apply in the lower right corner, or simply click on the next activity in the Activity List. When this activity completes, the RTL for each individual component, but not the top level of the subsystem, is written to the coreAssembler workspace directory.

### Complete Connections

In the Complete Connections activity, you have the opportunity to manually connect/disconnect wires in the subsystem. In general, no manual changes are required. Complete this activity by clicking Apply or clicking on the next activity in the Activity List.

### Generate Subsystem RTL

This is the last activity in the Create RTL activity group. Choose the RTL language you want the top level RTL to be written in and click Apply to complete the activity. When this activity completes processing the configured RTL for the entire subsystem has been written to the coreAssembler workspace.

### Additional Optional Activities in coreAssembler

- **Create Gate-Level Netlist** – This activity group is used to synthesize the subsystem. Support is provided for Design Compiler, DFT Compiler, Power Compiler, Physical Compiler, PrimeTime (timing model generation), and TetraMax (ATPG). Licenses are required for each tool used.
- **Verify Component** – For each DesignWare component, tests and component-specific testbenches are provided to demonstrate the functionality of the component. The tests and testbenches used in these activities are not reusable, however, the simulation waveforms are useful to examine detailed signal behavior.
- **Verify Subsystem** – The Formal Verification activity runs Formality (license required) on the entire synthesized subsystem. The Simulate Subsystem activity generates a custom testbench for your specific subsystem configuration and executes basic connectivity tests (also called “ping” tests). The testbench and tests created by this activity can be reused as a starter testbench for a larger subsystem or more detailed tests.
- **Create Component GTECH Simulation Model** – If VCS is the simulator used for the Subsystem Simulation or Verify Component activities, this activity group is not required. However, if a simulator other than VCS is selected, and you do not own a source license for the RTL of the DesignWare component, a GTECH simulation model is required and these activities must be completed before simulation can be run.
- **Export Subsystem** – If you intend to package the subsystem for re-use in different designs or to package your own subsystem IP for delivery to your customers, these activities can be used. They require a coreBuilder license. For more information, see the *coreBuilder User Guide*.

# Integration Considerations

When integrating multiple DesignWare IP components, much of the details are handled automatically by coreAssembler:

- Component inputs/outputs that belong to standard interfaces (for example, the AXI protocol) are connected automatically.
- Component inputs/outputs that do not belong to standard interfaces (for example, interrupt pins) are automatically handled with default connections that are pre-defined by the DesignWare IP designer.
- Interface parameters (for example, AXI address bus width) are defined one time and then propagated to connected components automatically.

- Configured RTL generation for both components and the top level of the subsystem occurs automatically.
- Subsystem-level synthesis and verification are also highly automated.

For the specific configuration discussed in this application note, connecting an AHB slave to an AXI subsystem, there are a few integration considerations that deserve additional examination.

## AMBA Lite

For the specific design example provided in this application note, if the connected AHB slave does not issue Split or Retry, the AHB subsystem could be configured as AMBA Lite, since the only connected AHB master is DW_axi_x2h.

To configure DW_ahb for AMBA Lite mode, the AMBA Lite interface parameter should be checked in the Add Subsystem Components activity. The parameter value chosen for AMBA Lite in DW_ahb automatically propagates to DW_axi_x2h. Both these components are optimized if the AMBA Lite mode parameter is enabled, resulting in smaller designs.

If more than one AHB master were connected, or if any connected AHB slave uses Split or Retry, then AMBA Lite mode could not be used.

## Address Map

To ensure access for AXI masters to AHB slaves, take care in defining both the DW_axi and DW_ahb address maps. There is no address translation provided in DW_axi_x2h, so the address used in the AXI transaction is passed directly to the AHB bus.

In coreAssembler 5.2 or later, it is possible to allow coreAssembler to automatically initialize the address maps of both DW_axi and DW_ahb to a valid configuration. This initialization is enabled with a pop-up dialogue during the completion of the Add Subsystem Components activity. If any changes are made to the automatically initialized address map, carefully review all address maps before completing the Configure Components activity, to make sure all masters have access to appropriate slaves.

## AXI/AHB Data Bus Widths

DW_axi_x2h requires the AXI data bus width to be greater than or equal to the width of the AHB data bus width. These interface parameters are defined on DW_axi and DW_ahb during the Add Subsystem Components activity. If the AHB data bus width is defined to be wider than the AXI data bus width, an error occurs when completing the Configure Components activity.

Discussion of how DW_axi_x2h adapts data widths between AXI and AHB can be found in Chapter 3 of the *DesignWare DW_axi_x2h Databook*, under Data Width Adaption.

## Endianness

DW_axi_x2h currently only supports Little Endian on both the AXI and AHB buses. If DW_ahb is configured as Big Endian, an error occurs when completing the Add Subsystem Components activity.

## AXI/AHB Lock Transactions

DW_axi_x2h does not support lock passing to the AHB bus by default, but this can be enabled by changing the configuration parameter X2H_PASS_LOCK. DW_axi supports for locked transactions is scheduled to be supported in version 1.02a of the component.

For detailed information on locking transaction support, see the databooks for the DW_axi, DW_axi_x2h, and DW_ahb components.

### INCR Burst Support on AHB

For AHB buses with more than one AHB master, DW_ahb by default early burst terminates unspecified length INCR bursts from a lower-priority master, if a higher priority master requests the bus. The DW_axi_x2h supports using INCR bursts by default, and will correctly resume the remaining INCR transfers if interrupted by the DW_ahb.

This functionality is configurable in both the DW_ahb and the DW_axi_x2h. The DW_ahb can be configured to not interrupt unspecified length INCR bursts from a lower priority master (see DW_ahb parameter AHB_FULL_INCR). Also, the DW_axi_x2h can be configured to never use unspecified length INCR bursts (see DW_axi_x2h parameter X2H_USE_DEFINED_ONLY).

## Reference Documentation

All of the following documents can be found in the *Guide to DesignWare AMBA IP Components Documentation*. This document can also be found on your local system, after installing the DesignWare Synthesizable IP for AMBA, at $DESIGNWARE_HOME/doc/amba/latest/intro.pdf.

- DW_axi_x2h Databook and Release Notes – For detailed information on how commands are transferred from AXI to AHB, refer to Chapter 3 (Functional Description) and Chapter 7 (Integration Considerations) in the DW_axi_x2h Databook.
- DW_axi Databook and Release Notes – Provides AXI interconnect details.
- DW_ahb Databook and Release Notes – Provides AHB interconnect details.
- Using coreAssembler with DesignWare Synthesizable IP – Chapter 6 provides tutorials for getting started with coreAssembler and DesignWare IP.
- *coreAssembler User Guide* – Provides general coreAssembler usage.
- AMBA 2.0 Specification – *Download here*.
- AMBA 3 Specification – *Download here*.