AHB-Lite APB4 Bridge Datasheet

© 2017 Roa Logic, All rights reserved
Introduction

The Roa Logic AHB-Lite APB4 Bridge is a fully parameterized soft IP interconnect bridge between the AMBA 3 AHB-Lite v1.0 and AMBA APB v2.0 bus protocols.

The AHB-Lite APB4 Bridge natively supports a single peripheral, however multiple APB4 peripherals may be connected to a single bridge by including supporting multiplexer logic – See the AMBA APB v2.0 Protocol specification. An APB4 Multiplexer IP implementing this capability is available from Roa Logic.

Features

- Full support for AMBA 3 AHB-Lite and APB version 2.0 (APB4) protocol
- Fully parameterized
- Unlimited APB4 address and data widths supported
- Configurable number of peripheral-side byte lanes with automatic handling of burst transfers
- Support for separate clock domain per interface with automatic handling of cross-domain timing.
Table of Contents

Introduction ..................................................................................................................2
Features .......................................................................................................................2

1 Getting Started ........................................................................................................5
  1.1 Deliverables .......................................................................................................5
  1.2 Running the testbench .....................................................................................6
    1.2.1 Self-checking testbench ...........................................................................6
    1.2.2 Makefile setup .........................................................................................6
    1.2.3 Makefile backup .......................................................................................6
    1.2.4 No Makefile .............................................................................................6

2 Specifications ..........................................................................................................7
  2.1 Functional Description .....................................................................................7
  2.2 AHB-Lite Interface ..........................................................................................8
  2.3 APB4 Interface ................................................................................................8

3 Configurations .........................................................................................................9
  3.1 Introduction .......................................................................................................9
  3.1.1 HADDR_SIZE ..............................................................................................9
  3.1.2 HDATA_SIZE .............................................................................................9
  3.1.3 PADDR_SIZE .............................................................................................9
  3.1.4 PDATA_SIZE .............................................................................................9
  3.1.5 SYNCDEPTH ............................................................................................10
  3.1.6 Limits to APB4 Address & Data Sizes .....................................................10

4 Interfaces ................................................................................................................11
  4.1 AHB-Lite Interface ..........................................................................................11
    4.1.1 HRESETn .................................................................................................11
    4.1.2 HCLK ......................................................................................................11
    4.1.3 HSEL ........................................................................................................11
    4.1.4 HTRANS ................................................................................................12
    4.1.5 HADDR ....................................................................................................12
    4.1.6 HWDATA ................................................................................................12
    4.1.7 HRDATA ................................................................................................12
    4.1.8 HWRITE .................................................................................................12
    4.1.9 HSIZE ......................................................................................................12
    4.1.10 HBURST ................................................................................................13
    4.1.11 HPROT ..................................................................................................13
    4.1.12 HREADYOUT .......................................................................................13
    4.1.13 HMASTLOCK .......................................................................................13
    4.1.14 HREADY ...............................................................................................13
    4.1.15 HRESP ..................................................................................................14

  4.2 APB4 (Peripheral) Interface .............................................................................14
    4.2.1 PRESETn .................................................................................................14
    4.2.2 PCLK .......................................................................................................14
    4.2.3 PSEL .......................................................................................................14
    4.2.4 PENABLE ...............................................................................................15
    4.2.5 PPROT ....................................................................................................15
    4.2.6 PWRITE ..................................................................................................15
    4.2.7 PSTRB ....................................................................................................15

© 2017 Roa Logic, All rights reserved
4.2.8  PADDR........................................................................................................15
4.2.9  PWDATA...................................................................................................15
4.2.10 PRDATA....................................................................................................15
4.2.11 PREADY....................................................................................................16
4.2.12 PSLVERR..................................................................................................16

5  Resources .......................................................................................................17

6  Revision History .............................................................................................18
1 Getting Started

1.1 Deliverables

All IP is delivered as a zipped tarball, which can be unzipped with all common compression tools (like unzip, winrar, tar, ...).

The tarball contains a directory structure as outlined below.

![Figure 1-1: IP Directory Structure](image)

The `doc` directory contains relevant documents like user guides, application notes, and datasheets.

The `rtl` directory contains the actual IP design files. Depending on the license agreement the AHB-Lite APB4 Bridge is delivered as either encrypted Verilog-HDL or as plain SystemVerilog source files. Encrypted files have the extension “.enc.sv”, plain source files have the extension “.sv”. The files are encryption according to the IEEE-P1735 encryption standard. Encryption keys for Mentor Graphics (Modelsim, Questasim, Precision), Synplicity (Synplify, Synplify-Pro), and Aldec (Active-HDL, Riviera-Pro) are provided. As such there should be no issue targeting any existing FPGA technology.

If any other synthesis or analysis tool is used then a plain source RTL delivery may be needed. A separate license agreement and NDA is required for such a delivery.

The `bench` directory contains the (encrypted) source files for the testbench.

© 2017 Roa Logic, All rights reserved
The sim directory contains the files/structure to run the simulations. Section 1.2 ‘Running the testbench’ provides for instructions on how to use the makefile.

1.2 Running the testbench

The AHB-Lite APB4 Bridge IP comes with a dedicated testbench that tests all features of the design and finally runs a full random test. The testbench is started from a Makefile that is provided with the IP.

The Makefile is located in the <install_dir>/sim/rtlsim/run directory. The Makefile supports most commonly used simulators; Modelsim/Questasim, Cadence ncsim, Aldec Riviera, and Synopsys VCS.

To start the simulation, enter the <install_dir>/sim/rtlsim/run directory and type: make <simulator>. Where simulator is any of: msim (for modelsim/questasim), ncsim (for Cadence ncsim), riviera (for Aldec Riviera-Pro), or vcs (for Synopsys VCS). For example type make msim to start the testbench in Modelsim/Questasim.

1.2.1 Self-checking testbench

The testbenches is a self-checking testbench intended to be executed from the command line. There is no need for a GUI or a waveform viewer. Once the testbench completes it displays a summary and closes the simulator.

1.2.2 Makefile setup

The simulator is executed in its associated directory. Inside this directory is another Makefile that contains simulator specific commands to start and execute the simulation. The <install_dir>/sim/rtlsim/run/Makefile enters the correct directory and calls the simulator specific Makefile.

For example modelsim is executed in the <install_dir>/sim/rtlsim/run/msim directory. Typing make msim loads the main Makefile, which then enters the msim sub-directory and calls its Makefile. This Makefile contains commands to compile the RTL and testbench sources with Modelsim, start the Modelsim simulator, and run the simulation.

1.2.3 Makefile backup

The <install_dir>/sim/rtlsim/bin directory contains backups of the original Makefiles. It may be desirable to modify or extend the Makefiles or to completely clean the run directory. Use the backups to restore the original setup.

1.2.4 No Makefile

For users unfamiliar with Makefiles or those on systems that do not natively support make (e.g. Windows) a run.do file is provided that can be used with Modelsim/Questasim and Riviera-Pro.
2 Specifications

2.1 Functional Description

The Roa Logic AHB-Lite APB4 Bridge is a highly configurable, fully parameterized soft IP interconnect bridge between the AMBA 3 AHB-Lite v1.0 and AMBA APB v2.0 bus protocols.

These protocols are commonly referred to as AHB-Lite and APB4 respectively – these terms will be used throughout this datasheet. All signals defined in the AHB-Lite and APB4 specifications are fully supported.

The IP contains 2 interfaces; an AHB-Lite Slave Interface and an APB4 Master Interface. Transactions received on the AHB-Lite Slave Interface are translated into APB4 transactions on the APB4 Master Interface. The IP automatically generates APB4 burst transactions if the APB4 data width is less than the AHB-Lite data width.

Each interface can operate on a separate clock domain and the IP automatically handles all cross clock domain synchronization requirements.

![Figure 2-1: Bridge Signaling](image)

**Notes:**

1. The APB4 Interface clock frequency must be less than or equal to the AHB-Lite interface clock frequency
2. The APB4 Interface data width must be less than or equal to the data width of the AHB-Lite interface
3. AHB-Lite and APB4 Interface data widths must be an integer multiple of bytes.
2.2 AHB-Lite Interface

An AHB-Lite Bus Master connects to the AHB interface of the AHB-Lite APB4 Bridge. The AHB interface is implemented as a regular AHB-Lite Slave Interface, supporting all signals in the AMBA 3 AHB-Lite v1.0 protocol specification.

2.3 APB4 Interface

An APB4 Bus Slave connects to the APB interface of the Bridge IP. The APB port is implemented as a regular APB4 Master Interface supporting all signals of the AMBA APB v2.0 protocol specification. This allows a single APB4 Peripheral to be connected directly to the Interface without further logic requirements.

Multiple peripherals can share the APB4 Interface through appropriate decoding and multiplexing of the interface signals. Roa Logic provides an additional APB4 Multiplexer IP to implement this capability.

Figure 2-2: APB4 Multiplexing Peripherals
3 Configurations

3.1 Introduction

The Roa Logic AHB-Lite AP4 Bridge is a fully configurable bridge IP to enable AHB-Lite based hosts to communicate with AP4 based peripherals. The core parameters and configuration options are described in this section.

3.1 Core Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HADDR_SIZE</td>
<td>Integer</td>
<td>32</td>
<td>AHB-Lite Address Bus Size</td>
</tr>
<tr>
<td>HDATA_SIZE</td>
<td>Integer</td>
<td>32</td>
<td>AHB-Lite Data Bus Size</td>
</tr>
<tr>
<td>PADDR_SIZE</td>
<td>Integer</td>
<td>10</td>
<td>APB4 Address Bus Size</td>
</tr>
<tr>
<td>PDATA_SIZE</td>
<td>Integer</td>
<td>8</td>
<td>APB4 Data Bus Size</td>
</tr>
<tr>
<td>SYNC_DEPTH</td>
<td>Integer</td>
<td>3</td>
<td>Clock Domain Crossing Sync Stages</td>
</tr>
</tbody>
</table>

Table 3-1: Core Parameters

3.1.1 HADDR_SIZE

The HADDR_SIZE parameter specifies the width of the address bus for the AHB-Lite interface.

3.1.2 HDATA_SIZE

The HDATA_SIZE parameter specifies the width of the data bus for the AHB-Lite interface. This parameter must equal an integer multiple of bytes and also be greater or equal to PDATA_SIZE:

Conditions:

\[ (\text{HDATA\_SIZE} \geq \text{PDATA\_SIZE}) \]
\[ (\text{HDATA\_SIZE} \mod 8 = 0) \]

3.1.3 PADDR_SIZE

The PADDR_SIZE parameter specifies the width of the address bus for the APB4 (i.e. peripheral) interface.

3.1.4 PDATA_SIZE

The PDATA_SIZE parameter specifies the width of the data bus for APB4 (i.e. peripheral) interface. This parameter must equal an integer multiple of bytes and also be less than or equal to HDATA_SIZE.

Conditions:

\[ (\text{PDATA\_SIZE} \leq \text{HDATA\_SIZE}) \]
\[ (\text{PDATA\_SIZE} \mod 8 = 0) \]
3.1.5 Sync_Depth

The APB4 Bridge IP supports operating the AHB-Lite and APB4 interfaces in separate, unrelated clock domains. The IP automatically handles cross-domain synchronization and the Sync_Depth parameter determines the number of synchronization stages between these clock domains.

Increasing this parameter reduces the possibility of metastability for signals crossing between the two domains, but at the cost of increased latency.

The minimum and default value of the Sync_Depth parameter is 3.

3.1.6 Limits to APB4 Address & Data Sizes

The AMBA APB v2.0 Protocol specification limits the widths of both Address (PADDR_SIZE) and Data (PDATA_SIZE) buses to 32 bits. However the AHB-Lite APB4 Bridge IP Address and Data sizes are not similarly constrained – any Address width and any byte-aligned Data width is supported by the IP.
## 4 Interfaces

### 4.1 AHB-Lite Interface

The AHB-Lite interface is a regular AHB-Lite Slave Interface. All signals are supported. See the *AMBA 3 AHB-Lite Specification* for a complete description of the signals.

<table>
<thead>
<tr>
<th>Port</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRESETn</td>
<td>1</td>
<td>Input</td>
<td>Asynchronous active low reset</td>
</tr>
<tr>
<td>HCLK</td>
<td>1</td>
<td>Input</td>
<td>Clock Input</td>
</tr>
<tr>
<td>HSEL</td>
<td>1</td>
<td>Input</td>
<td>Bus Select</td>
</tr>
<tr>
<td>HTRANS</td>
<td>2</td>
<td>Input</td>
<td>Transfer Type</td>
</tr>
<tr>
<td>HADDR</td>
<td>HADDR_SIZE</td>
<td>Input</td>
<td>Address Bus</td>
</tr>
<tr>
<td>HDATA</td>
<td>HDATA_SIZE</td>
<td>Input</td>
<td>Write Data Bus</td>
</tr>
<tr>
<td>HRDATA</td>
<td>HDATA_SIZE</td>
<td>Output</td>
<td>Read Data Bus</td>
</tr>
<tr>
<td>HWRITE</td>
<td>1</td>
<td>Input</td>
<td>Write Select</td>
</tr>
<tr>
<td>HSIZE</td>
<td>3</td>
<td>Input</td>
<td>Transfer Size</td>
</tr>
<tr>
<td>HBURST</td>
<td>3</td>
<td>Input</td>
<td>Transfer Burst Size</td>
</tr>
<tr>
<td>HPROT</td>
<td>4</td>
<td>Input</td>
<td>Transfer Protection Level</td>
</tr>
<tr>
<td>HMASTLOCK</td>
<td>1</td>
<td>Input</td>
<td>Transfer Master Lock</td>
</tr>
<tr>
<td>HREADYOUT</td>
<td>1</td>
<td>Output</td>
<td>Transfer Ready Output</td>
</tr>
<tr>
<td>HREADY</td>
<td>1</td>
<td>Input</td>
<td>Transfer Ready Input</td>
</tr>
<tr>
<td>HRESP</td>
<td>1</td>
<td>Input</td>
<td>Transfer Response</td>
</tr>
</tbody>
</table>

**Table 4-1: AHB-Lite Interface Ports**

#### 4.1.1 HRESETn

When the active low asynchronous HRESETn input is asserted (‘0’), the interface is put into its initial reset state.

#### 4.1.2 HCLK

**HCLK** is the interface system clock. All internal logic for the AMB3-Lite interface operates at the rising edge of this system clock and AHB bus timings are related to the rising edge of **HCLK**.

The frequency of **HCLK** must be greater than or equal to that of the APB4 (Peripheral) Interface clock **PCLK**:

*Conditions:* \( \text{Freq} (\text{HCLK}) \geq \text{Freq} (\text{PCLK}) \)

#### 4.1.3 HSEL

The AHB-Lite interface only responds to other signals on its bus – with the exception of the global asynchronous reset signal HRESETn – when **HSEL** is
asserted ('1'). When HSEL is negated ('0') the interface considers the bus IDLE and negates HREADYOUT ('0').

### 4.1.4 HTRANS

HTRANS indicates the type of the current transfer.

<table>
<thead>
<tr>
<th>HTRANS</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>IDLE</td>
<td>No transfer required</td>
</tr>
<tr>
<td>01</td>
<td>BUSY</td>
<td>Connected master is not ready to accept data, but intends to continue the current burst.</td>
</tr>
<tr>
<td>10</td>
<td>NONSEQ</td>
<td>First transfer of a burst or a single transfer</td>
</tr>
<tr>
<td>11</td>
<td>SEQ</td>
<td>Remaining transfers of a burst</td>
</tr>
</tbody>
</table>

Table 4-2: AHB-Lite Transfer Type (HTRANS)

### 4.1.5 HADDR

HADDR is the address bus. Its size is determined by the HADDR_SIZE parameter and is driven to the connected peripheral.

### 4.1.6 HWDATA

HWDATA is the write data bus. Its size is determined by the HDATA_SIZE parameter and is driven to the connected peripheral.

### 4.1.7 HRDATA

HRDATA is the read data bus. Its size is determined by HDATA_SIZE parameter and is sourced by the APB4 peripheral.

### 4.1.8 HWRITE

HWRITE is the read/write signal. HWRITE asserted ('1') indicates a write transfer.

### 4.1.9 HSIZE

HSIZE indicates the size of the current transfer.

<table>
<thead>
<tr>
<th>HSIZE</th>
<th>Size</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>8bit</td>
<td>Byte</td>
</tr>
<tr>
<td>001</td>
<td>16bit</td>
<td>Half Word</td>
</tr>
<tr>
<td>010</td>
<td>32bit</td>
<td>Word</td>
</tr>
<tr>
<td>011</td>
<td>64bits</td>
<td>Double Word</td>
</tr>
<tr>
<td>100</td>
<td>128bit</td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>256bit</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>512bit</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>1024bit</td>
<td></td>
</tr>
</tbody>
</table>

Table 4-3: Transfer Size Values (HSIZE)
4.1.10 HBURST

HBURST indicates the transaction burst type - a single transfer or part of a burst.

<table>
<thead>
<tr>
<th>HBURST</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>SINGLE</td>
<td>Single access</td>
</tr>
<tr>
<td>001</td>
<td>INCR</td>
<td>Continuous incremental burst</td>
</tr>
<tr>
<td>010</td>
<td>WRAP4</td>
<td>4-beat wrapping burst</td>
</tr>
<tr>
<td>011</td>
<td>INCR4</td>
<td>4-beat incrementing burst</td>
</tr>
<tr>
<td>100</td>
<td>WRAP8</td>
<td>8-beat wrapping burst</td>
</tr>
<tr>
<td>101</td>
<td>INCR8</td>
<td>8-beat incrementing burst</td>
</tr>
<tr>
<td>110</td>
<td>WRAP16</td>
<td>16-beat wrapping burst</td>
</tr>
<tr>
<td>111</td>
<td>INCR16</td>
<td>16-beat incrementing burst</td>
</tr>
</tbody>
</table>

Table 4-4: AHB-Lite Burst Types (HBURST)

4.1.11 HPROT

The HPROT signals provide additional information about the bus transfer and are intended to implement a level of protection.

<table>
<thead>
<tr>
<th>Bit#</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>Cacheable region addressed</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Non-cacheable region addressed</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Bufferable</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Non-bufferable</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Privileged Access</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>User Access</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Data Access</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>Opcode fetch</td>
</tr>
</tbody>
</table>

Table 4-5: AHB-Lite Transaction Protection Signals (HPROT)

4.1.12 HREADYOUT

HREADYOUT indicates that the current transfer has finished.

4.1.13 HMASTLOCK

HMASTLOCK, the instruction master lock signal, indicates if the current transfer is part of a locked sequence, commonly used for Read-Modify-Write cycles.

4.1.14 HREADY

HREADY indicates whether or not the addressed peripheral is ready to transfer data. When HREADY is negated ('0') the peripheral is not ready, forcing wait states. When HREADY is asserted ('1') the peripheral is ready and the transfer completed.
4.1.15 HRESP

HRESP is the instruction transfer response and indicates OKAY ('0') or ERROR ('1'). An error response causes an Instruction Bus Error Interrupt.

4.2 APB4 (Peripheral) Interface

The APB4 Interface is a regular APB4 Master Interface. All signals defined in the protocol are supported as described below. See the AMBA APB Protocol v2.0 Specifications for a complete description of the signals.

<table>
<thead>
<tr>
<th>Port</th>
<th>Size</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRESETn</td>
<td>1</td>
<td>Input</td>
<td>Asynchronous active low reset</td>
</tr>
<tr>
<td>PCLK</td>
<td>1</td>
<td>Input</td>
<td>Clock Input</td>
</tr>
<tr>
<td>PSEL</td>
<td>1</td>
<td>Output</td>
<td>Peripheral Select</td>
</tr>
<tr>
<td>PENABLE</td>
<td>1</td>
<td>Output</td>
<td>Peripheral Enable Control</td>
</tr>
<tr>
<td>PPROT</td>
<td>3</td>
<td>Output</td>
<td>Transfer Protection Level</td>
</tr>
<tr>
<td>PWRITE</td>
<td>1</td>
<td>Output</td>
<td>Write Select</td>
</tr>
<tr>
<td>PSTRB</td>
<td>PDATA_SIZE/8</td>
<td>Output</td>
<td>Byte Lane Indicator</td>
</tr>
<tr>
<td>PADDR</td>
<td>PADDR_SIZE</td>
<td>Output</td>
<td>Address Bus</td>
</tr>
<tr>
<td>PWDATA</td>
<td>PDATA_SIZE</td>
<td>Output</td>
<td>Write Data Bus</td>
</tr>
<tr>
<td>PRDATA</td>
<td>PDATA_SIZE</td>
<td>Input</td>
<td>Read Data Bus</td>
</tr>
<tr>
<td>PREADY</td>
<td>1</td>
<td>Input</td>
<td>Transfer Ready Input</td>
</tr>
<tr>
<td>PSLVERR</td>
<td>1</td>
<td>Input</td>
<td>Transfer Error Indicator</td>
</tr>
</tbody>
</table>

Table 4-6: APB4 Peripheral Interface Ports

4.2.1 PRESETn

When the active low asynchronous PRESETn input is asserted ('0'), the APB4 interface is put into its initial reset state.

4.2.2 PCLK

PCLK is the APB4 interface system clock. All internal logic for the APB4 interface operates at the rising edge of this system clock and APB4 bus timings are related to the rising edge of PCLK.

The frequency of PCLK must be less than or equal to that of the AHB-Lite Interface clock HCLK:

Conditions: \( \text{Freq}(\text{PCLK}) \leq \text{Freq}(\text{HCLK}) \)

4.2.3 PSEL

The APB4 Bridge generates PSEL, signaling to an attached peripheral that it is selected and a data transfer is pending.

Note: To support multiple APB4 peripherals, individual PSEL signals must be generated per peripheral – Roa Logic provides an
additional APB4 Multiplexer IP to support this requirement

### 4.2.4 PENABLE

The APB4 Bridge asserts PENABLE during the 2nd and subsequent cycles of an APB4 data transfer.

### 4.2.5 PPROT

PPROT[2:0] indicates the protection type of the data transfer, with 3 levels of protection supported as follows:

<table>
<thead>
<tr>
<th>Bit#</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>Instruction Access</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>Data Access</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Non-Secure Access</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>Secure Access</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Privileged Access</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>Normal Access</td>
</tr>
</tbody>
</table>

Table 4-7: APB4 Transaction Protection Types

### 4.2.6 PWRITE

PWRITE indicates a data write access when asserted high (‘1’) and a read data access when de-asserted (‘0’)

### 4.2.7 PSTRB

There is one PSTRB signal per byte lane of the APB4 write data bus (PWDATA). These signals indicate which byte lane to update during a write transfer such that PSTRB[n] corresponds to PWDATA[(8n+7):8n].

### 4.2.8 PADDR

PADDR is the APB4 address bus. The bus width is defined by the PADDR_SIZE parameter and is driven by the APB4 Bridge core.

### 4.2.9 PWDATA

PWDATA is the APB4 write data bus and is driven by the APB4 Bridge core during write cycles, indicated when PWRITE is asserted (‘1’). The bus width must be byte-aligned and is defined by the PDATA_SIZE parameter.

### 4.2.10 PRDATA

PRDATA is the APB4 read data bus. An attached peripheral drives this bus during read cycles, indicated when PWRITE is de-asserted (‘0’). The bus width must be byte-aligned and is defined by the PDATA_SIZE parameter.
4.2.11 PREADY

PREADY is driven by the attached peripheral. It is used to extend an APB4 transfer.

4.2.12 PSLVERR

PSLVERR indicates a failed data transfer when asserted ('1'). As APB4 peripherals are not required to support this signal it must be tied LOW ('0') when unused.
5 Resources

Below are some example implementations for various platforms.

All implementations are push button, no effort has been undertaken to reduce area or improve performance.

<table>
<thead>
<tr>
<th>Platform</th>
<th>DFF</th>
<th>Logic Cells</th>
<th>Memory</th>
<th>Performance (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Table 5-1: Resource Utilization Examples*
6 Revision History

<table>
<thead>
<tr>
<th>Date</th>
<th>Rev.</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>01-Feb-2017</td>
<td>1.0</td>
<td>Initial Release</td>
</tr>
</tbody>
</table>

Table 6-1: Revision History