AXI to APB Bridge Specification

Introduction

The AXI to APB Bridge is designed to interface between an AXI bus and an APB bus, facilitating communication between AXI masters and APB slaves. This bridge adheres to the AMBA AXI and APB protocols, providing a seamless translation of AXI transactions into APB transactions. The bridge supports single 32-bit data transfers and ensures proper handling of read and write operations.

Architecture

The design consists of several interconnected modules, each responsible for a specific aspect of the AXI to APB translation process:

1. **Top-Level Module (PREFIX)**: This module orchestrates the overall operation of the bridge, connecting various sub-modules and managing the flow of data and control signals.
2. **Command Module (PREFIX_cmd)**: Handles the command queue, managing incoming AXI commands and determining whether they are read or write operations.
3. **Control Module (PREFIX_ctrl)**: Manages the APB control signals, ensuring proper sequencing of `psel`, `penable`, and `pwrite` signals based on the command type and readiness of the APB slave.
4. **Multiplexer Module (PREFIX_mux)**: Handles the selection of the appropriate APB slave based on the address decoding logic, ensuring that the correct slave is addressed for each transaction.
5. **Read Module (PREFIX_rd)**: Manages the read data path, capturing data from the APB slave and forwarding it to the AXI master.
6. **Write Module (PREFIX_wr)**: Manages the write data path, forwarding data from the AXI master to the APB slave and handling write responses.

Interface

Top-Level Module (PREFIX)

Signal	Width	In/Out	Description
clk	1	In	Clock signal for synchronous operations
reset	1	In	Asynchronous reset signal
penable	1	Out	APB enable signal
pwrite	1	Out	APB write signal
paddr	ADDR_BITS	Out	APB address bus

Signal	Width	In/Out	Description
pwdata	32	Out	APB write data bus
cmd_empty	1	Wire	Indicates if the command queue is empty
cmd_read	1	Wire	Indicates if the current command is a read
cmd_id	ID_BITS	Wire	Command identifier
cmd_addr	ADDR_BITS	Wire	Command address
cmd_err	1	Wire	Command error indicator
finish_wr	1	Wire	Write operation completion indicator
finish_rd	1	Wire	Read operation completion indicator

Command Module (PREFIX_cmd)

Signal	Width	In/Out	Description
clk	1	In	Clock signal for synchronous operations
reset	1	In	Asynchronous reset signal
cmd_empty	1	Out	Indicates if the command queue is empty
cmd_read	1	Out	Indicates if the current command is a read
cmd_id	ID_BITS	Out	Command identifier
cmd_addr	ADDR_BITS	Out	Command address
cmd_err	1	Out	Command error indicator
AGROUP_APB_AXI_A		Wire	Command mux output
cmd_push	1	Wire	Command push signal
cmd_pop	1	Wire	Command pop signal
cmd_full	1	Wire	Indicates if the command queue is full
read	1	Reg	Read operation flag
wreq	1	Wire	Write request signal
rreq	1	Wire	Read request signal
wack	1	Wire	Write acknowledge signal
rack	1	Wire	Read acknowledge signal
AERR	1	Wire	Address error indicator

Control Module (PREFIX_ctrl)

Signal	Width	In/Out	Description
clk	1	In	Clock signal for synchronous operations
reset	1	In	Asynchronous reset signal
psel	1	Out	APB select signal
penable	1	Out	APB enable signal
pwrite	1	Out	APB write signal
wstart	1	Wire	Write start signal
rstart	1	Wire	Read start signal
busy	1	Reg	Busy flag indicating ongoing operation

Signal	Width	In/Out	Description
pack	1	Wire	APB acknowledge signal
cmd_ready	1	Wire	Command ready signal

Multiplexer Module (PREFIX_mux)

Signal	Width	In/Out	Description
clk	1	In	Clock signal for synchronous operations
reset	1	In	Asynchronous reset signal
cmd_addr	ADDR_BITS	In	Command address
psel	1	In	APB select signal
prdata	32	Out	APB read data bus
pready	1	Out	APB ready signal
pslverr	1	Out	APB slave error signal
pselSX	1	Out	APB select signal for specific slave
preadySX	1	In	APB ready signal from specific slave
pslverrSX	1	In	APB slave error signal from specific slave
prdataSX	32	In	APB read data from specific slave
pready	1	Reg	APB ready signal (internal)
pslverr_pre	1	Reg	APB slave error signal (internal)
pslverr	1	Reg	APB slave error signal (internal)
prdata_pre	32	Reg	APB read data (internal)
prdata	32	Reg	APB read data (internal)
slave_num	SLV_BITS	Reg	Slave number for address decoding

Read Module (PREFIX_rd)

Signal	Width	In/Out	Description
clk	1	In	Clock signal for synchronous operations
reset	1	In	Asynchronous reset signal
finish_rd	1	Out	Read operation completion indicator
RGROUP_APB_AXI_R_OUT	Reg	Out	Read data output register

Write Module (PREFIX_wr)

Signal	Width	In/Out	Description
clk	1	In	Clock signal for synchronous operations
reset	1	In	Asynchronous reset signal
finish_wr	1	Out	Write operation completion indicator

Signal	Width	In/Out	Description
BGROUP_APB_AXI_B_reg	2	OUT	Write response output register

Timing

The AXI to APB Bridge operates synchronously with the provided clock signal. The latency for read and write operations is determined by the APB slave's response time. The bridge ensures that all signals are sampled and updated on the rising edge of the clock. The `psel`, `penable`, and `pwrite` signals are controlled to ensure proper sequencing and timing of APB transactions.

Usage

To use the AXI to APB Bridge, follow these steps:

1. **Initialization**: Ensure that the `reset` signal is asserted to initialize the bridge. Deassert the `reset` signal to begin normal operation.
2. **Command Issuance**: Provide AXI commands through the appropriate AXI interface signals. The bridge will automatically handle the translation to APB transactions.
3. **Data Transfer**: For write operations, ensure that the `WVALID` signal is asserted along with valid data on the `WDATA` bus. For read operations, monitor the `RVALID` signal to capture read data.
4. **Completion**: Monitor the `finish_wr` and `finish_rd` signals to determine the completion of write and read operations, respectively.
5. **Error Handling**: Check the `cmd_err` and `pslverr` signals for any errors during the transaction process.

The bridge supports configuration for multiple APB slaves, with address decoding logic to select the appropriate slave based on the command address.

Functional Description (Generated by funcgen)

Functional Description of AXI to APB Bridge Design Modules

Module: PREFIX

Purpose

The `PREFIX` module acts as a top-level module for bridging AXI transactions to APB protocol slaves. It manages the handshake and state transitions to

coordinate data transfer between AXI and APB buses.

Parameters

- No explicit parameters in the provided code, though system-level parameters from included headers might affect its behavior.

Ports

- `clk` (input, 1-bit): Clock signal for synchronization.
- `reset` (input, 1-bit): Reset signal, active high.
- `penable` (output, 1-bit): APB enable signal indicating the second APB cycle.
- `pwrite` (output, 1-bit): APB write signal indicating write transaction.
- `pwdata` (output, 32-bit): Data to be written on the APB bus.
- `pselSX` (output, 1-bit): APB select signal for slave X.
- `prdataSX` (input, 32-bit): Data read from the APB slave.
- `preadySX` (input, 1-bit): Indicates slave-specific ready status.
- `pslverrSX` (input, 1-bit): Slave error signal for the APB interface.

Internal Signals

- `GROUP_APB3` (wire): Represents aggregated APB3 bus signals.
- `cmd_empty` (wire): Indicates whether the command FIFO is empty.
- `cmd_read` (wire): Signals if the current command is a read operation.
- `cmd_err` (wire): Error signal for command processing.
- `finish_wr` (wire): Indicates completion of a write operation.
- `finish_rd` (wire): Indicates completion of a read operation.

Functionality

1. **Sequential and Combinational Logic:**
   - Generates APB address and write data based on AXI write data.
   - Controls the APB transaction lifecycle through penable and pwrite signals.
2. **State Machines:**
   - Coordinates AXI to APB transfers, detecting command readiness and managing read/write paths.

Instantiations

- **`PREFIX_cmd:`** Manages AXI command processing including command FIFO.
- **`PREFIX_rd:`** Handles read path between AXI and APB, setting read responses.
- **`PREFIX_wr:`** Manages write path, asserting proper write responses.
- **`PREFIX_ctrl:`** Controls APB command sequencing and execution.

- **PREFIX_mux:** (Conditional) Handles selection logic in systems with multiple slaves.

Module: PREFIX_cmd

Purpose

The PREFIX_cmd module manages the command processing for AXI transactions, handling both read and write requests and interfacing with a command FIFO.

Parameters

- None explicitly defined in the provided code.

Ports

- clk (input, 1-bit): System clock.
- reset (input, 1-bit): System reset.
- finish_wr (input, 1-bit): Completion signal for write operations.
- finish_rd (input, 1-bit): Completion signal for read operations.
- cmd_empty (output, 1-bit): Command FIFO empty status.
- cmd_read (output, 1-bit): Indicates whether the current command is a read.
- cmd_err (output, 1-bit): Signals an error in command processing.

Internal Signals

- AGROUP_APB_AXI_A (wire): Represents active AXI address group selection.
- cmd_push/pop/full/empty (wire): Command FIFO control signals.
- read (reg): Determines if the ongoing transaction is read-oriented.

Functionality

- **Combinational Logic:**
  - Determines request type (read/write) and initiates command sequence.
- **Sequential Logic:**
  - Manages FIFO operations for command storage and retrieval based on completion signals.

Instantiations

- **prgen_fifo:** Implements command FIFO for storing and managing pending requests.

Module: PREFIX_ctrl

Purpose

The `PREFIX_ctrl` module controls APB transaction flow, enabling command start sequences and managing busy state during transfers.

Parameters

- None explicitly specified.

Ports

- `clk`, `reset`, `finish_wr`, `finish_rd`: As described above.
- `cmd_empty` (input, 1-bit): Command FIFO empty status.
- `cmd_read` (input, 1-bit): Indicates if the command in progress is read.
- `WVALID` (input, 1-bit): Write valid indicator from AXI.
- `psel`, `penable`, `pwrite` (outputs): Control signals for APB transaction management.
- `pready` (input, 1-bit): Ready status input affecting state transitions.

Internal Signals

- `wstart`, `rstart` (wire): Signals for start of write/read transactions.
- `busy` (reg): Indicates ongoing transaction activity and prohibits new command starts.
- `pack` (wire): Identifies completion of an APB operation cycle.
- `cmd_ready` (wire): Indicates readiness to start processing a new command.

Functionality

- **Sequential and Combinational Logic:**
  - Manages APB initiation based on command readiness, starts or continues transactions based on available commands.
- **State Management:**
  - Handles busy states to prevent command overlap and ensures proper signal assertions during transactions.

Instantiations

- None within the module itself.

Module: PREFIX_mux

Purpose

The `PREFIX_mux` module dynamically manages multiplexer operations for APB slaves based on decoded addresses, directing traffic to and from appropriate slaves.

Parameters

- `ADDR_MSB` and `ADDR_LSB`, representing the bit range used for address decoding.

Ports

- Includes control and data path signals such as `cmd_addr`, `pselSX`, `prdataSX`, etc.

Internal Signals

- Includes register signals for read data and error status.

Functionality

- **Combinational Logic:**
  - Decodes address bits to select the appropriate slave, controls the data path and acknowledges based on read/write operations.

Instantiations

- None, though address decoding logic is prominent.

Module: PREFIX_rd

Purpose

Manages the read operations from APB to AXI, ensuring data and response transportation complies with AXI protocol.

Parameters

- Response codes: `RESP_OK`, `RESP_SLVERR`, `RESP_DECERR`.

Ports

- Includes signals for data readiness and AXI response generation.

Internal Signals

- Implements response and data transfer synchronizations.

Functionality

- **Sequential Logic:**
  - Generates responses and captures read data, handling the state transitions for valid AXI read responses.
  - Manages read completions and response assertions based on APB and command status.

Instantiations

- None.

Module: PREFIX_wr

Purpose

The `PREFIX_wr` module manages write transactions from AXI to APB, coordinating data and status response signals.

Parameters

- Same response types as in `PREFIX_rd`.

Ports

- `cmd_err`, `cmd_id`, `finish_wr`, controls transaction lifecycle and completion signaling.

Internal Signals

- Manage write readiness and assertion of valid write responses.

Functionality

- **Sequential Logic:**
  - Asserts write completion, aligns data as per the AXI-B channel requirements, including handling slave errors and completion acknowledgements.

Instantiations

- None.

Inter-Module Connections

In the AXI to APB bridge design, the `PREFIX` module acts as the main control hub, interfacing with several submodules (`PREFIX_cmd`, `PREFIX_ctrl`, `PREFIX_rd`, `PREFIX_wr`) that handle command processing, control logic sequencing, and read/write path management. Each submodule operates on a specific task within the AXI to APB conversion, coordinating through shared signals defined across input/output ports and intrinsic state machines within each submodule. The command and control flow from `PREFIX_cmd` to `PREFIX_ctrl` plays a central role, propagating actions to the `PREFIX_rd` and `PREFIX_wr` modules which execute read and write operations respectively, completing AXI transactions to the connected APB slaves.

Each submodule is designed distinctly for handling specific transactional aspects, with the top-level module monitoring and controlling all interconnections to ensure consistent and accurate protocol bridging. This layered architectural approach facilitates a modular design, geared towards high integration flexibility and scalability in multi-slave APB environments.