GPIO Specification

Introduction

The General Purpose Input/Output (GPIO) module is designed to provide configurable input and output capabilities for a system-on-chip (SoC). It supports a variety of operations including reading and writing data, configuring direction, and handling interrupts. The module adheres to a standard register-based interface for configuration and data transfer, making it suitable for integration into a wide range of digital systems.

Architecture

The GPIO module is structured around a register-based architecture, allowing for flexible configuration and control. Key components of the architecture include:

- **Top-Level Module**: The `gpio` module serves as the top-level entity, encapsulating all functionality related to GPIO operations.
- **Register Interface**: The module uses a register interface for configuration and data operations, supporting both read and write transactions.
- **Data Flow**: The module handles data flow through a combination of input synchronization, output control, and interrupt management.
- **Interrupt Handling**: The module supports configurable interrupt generation based on input changes, with options for level or edge-triggered interrupts.

Key Operations

- **Write Operations**: Managed through address and data registers, supporting delayed data capture and write enable signals.
- **Read Operations**: Implemented via a multiplexer that selects data based on the requested address.
- **Interrupt Management**: Configurable through registers to enable, set, clear, and mask interrupts.

Interface

Signal	Width	In/Out	Description
`clk_i`	1	In	Clock input signal
`rst_i`	1	In	Reset input signal, active high
`cfg_awvalid_i`	1	In	Write address valid signal
`cfg_awaddr_i`	32	In	Write address input
`cfg_wvalid_i`	1	In	Write data valid signal
`cfg_wdata_i`	32	In	Write data input
`cfg_wstrb_i`	4	In	Write strobe input

Signal	Width	In/Out	Description
cfg_bready_i	1	In	Write response ready signal
cfg_arvalid_i	1	In	Read address valid signal
cfg_araddr_i	32	In	Read address input
cfg_rready_i	1	In	Read data ready signal
gpio_input_i	32	In	GPIO input data
cfg_awready_o	1	Out	Write address ready signal
cfg_wready_o	1	Out	Write data ready signal
cfg_bvalid_o	1	Out	Write response valid signal
cfg_bresp_o	2	Out	Write response
cfg_arready_o	1	Out	Read address ready signal
cfg_rvalid_o	1	Out	Read data valid signal
cfg_rdata_o	32	Out	Read data output
cfg_rresp_o	2	Out	Read response
gpio_output_o	32	Out	GPIO output data
gpio_output_enable_o	32	Out	GPIO output enable
intr_o	1	Out	Interrupt output signal

Timing

The GPIO module operates synchronously with the input clock `clk_i`. Key timing characteristics include:

- **Latency**: The module introduces a latency of one clock cycle for read and write operations due to internal register synchronization.
- **Signal Sampling**: Inputs are sampled on the rising edge of `clk_i`, and outputs are updated on the same edge.
- **Reset Behavior**: The module resets all internal registers and outputs to their default states when `rst_i` is asserted high.

Usage

To use the GPIO module, follow these steps:

1. **Initialization**: Ensure the module is reset by asserting `rst_i` high, then deassert it to begin normal operation.
2. **Configuration**: Use the configuration interface to set up GPIO direction, output values, and interrupt settings.
3. **Data Operations**: Perform read and write operations by asserting the appropriate valid signals (`cfg_awvalid_i`, `cfg_wvalid_i`, `cfg_arvalid_i`) and providing the necessary addresses and data.
4. **Interrupt Handling**: Configure interrupt settings through the relevant registers, and monitor `intr_o` for interrupt notifications.

The module supports flexible configuration and operation, making it suitable for a wide range of applications requiring general-purpose I/O capabilities.

Functional Description (Generated by funcgen)

gpio Module Description

Module Name

- **Name**: gpio
- **Source File**: gpio.v

Purpose

The GPIO (General Purpose Input/Output) module is designed to interface with external digital signals, providing read and write capabilities. It supports configurable data directions, and includes interrupt generation and handling capabilities.

Parameters

The module does not define explicit parameters in the provided code; however, it uses constants for register addresses and defaults from an included file (`gpio_defs.v`).

Ports

Inputs - `clk_i` (1-bit): Clock signal - `rst_i` (1-bit): Reset signal, active high - `cfg_awvalid_i` (1-bit): Indicates valid write address - `cfg_awaddr_i` (32-bit): Write address input - `cfg_wvalid_i` (1-bit): Indicates valid write data - `cfg_wdata_i` (32-bit): Write data input - `cfg_wstrb_i` (4-bit): Write strobe input - `cfg_bready_i` (1-bit): Indicates ready to accept write response - `cfg_arvalid_i` (1-bit): Indicates valid read address - `cfg_araddr_i` (32-bit): Read address input - `cfg_rready_i` (1-bit): Indicates ready to accept read data - `gpio_input_i` (32-bit): Input data from GPIO pins

Outputs - `cfg_awready_o` (1-bit): Indicates module is ready for write address - `cfg_wready_o` (1-bit): Indicates module is ready for write data - `cfg_bvalid_o` (1-bit): Valid write response indication - `cfg_bresp_o` (2-bit): Write response status - `cfg_arready_o` (1-bit): Indicates module is ready for read address - `cfg_rvalid_o` (1-bit): Valid read response indication - `cfg_rdata_o` (32-bit): Read data output - `cfg_rresp_o` (2-bit): Read response status - `gpio_output_o` (32-bit): Output data to GPIO pins - `gpio_output_enable_o` (32-bit): GPIO pin output enable control - `intr_o` (1-bit): Interrupt output

Internal Signals

- **Sequencing and Synchronization Signals**:

  - `awvalid_q`, `wvalid_q`: Signals for synchronizing data/address acceptance
  - `input_ms`, `input_q`, `input_last_q`: Used for double buffering and edge detection of input signals
  - `interrupt_raw_q`, `intr_q`: Handle interrupt generation and raw interrupt status
- **Register Signals**:
  - `gpio_direction_output_q`, `gpio_int_mask_enable_q`, `gpio_int_level_active_high_q`, `gpio_int_mode_edge_q`: Registers controlling GPIO operation, interrupt mask, and levels
  - `wr_addr_q`, `wr_data_q`: Captures write address/data
- **Control and Status Signals**:
  - `bvalid_q`, `rvalid_q`: Handle read/write response status
  - `data_r`, `rd_data_q`: Used to store read data
  - `output_q`, `output_next_r`: Captures next state logic for outputs

Functionality

Sequential Logic - Use of flip-flops to store write address (`wr_addr_q`), write data (`wr_data_q`), and maintain read/write valid statuses (`rvalid_q`, `bvalid_q`). - Registers to control GPIO direction and interrupt settings executed on write.

Combinational Logic - Address decoding and data multiplexing for read operations, depending on the input address. - Output logic that computes the next state based on current register values and input commands for setting/clearing GPIO outputs.

Control Logic - FSM-like logic to handle various configurations like enabling write/read operations based on periphery readiness and data validity. - Generate internal "write enable" (`write_en_w`) and "read enable" (`read_en_w`) signals based on control ports.

Instantiations

- No sub-module instantiations; all functionality is implemented within the module.

Inter-Module Connections

- This description is standalone; hence there are no hierarchical or inter-module connections provided.

This description covers the internal workings and external interactions of the GPIO module, necessary for both the development of new features and for constructing effective verification strategies.