GENERAL PURPOSE INPUT/OUTPUT (APB_gpio)

Interface Handbook

VERSION 1.0

February 9, 2021

Contents

List of Figures

List of Tables

1 Introduction

General purpose Input/Outputs (GPIOs) IP are the pins that can be configured as either input or output by the user. There are 32 GPIO pins that can be accessed through AMBA APB interface. When any pin is configured as output, write operation can be performed on it by writing in the output registers. One can also read the status of any GPIO pin by reading input register. Block diagram of above stated IP is shown in figure 1.

1.1 Block Diagram

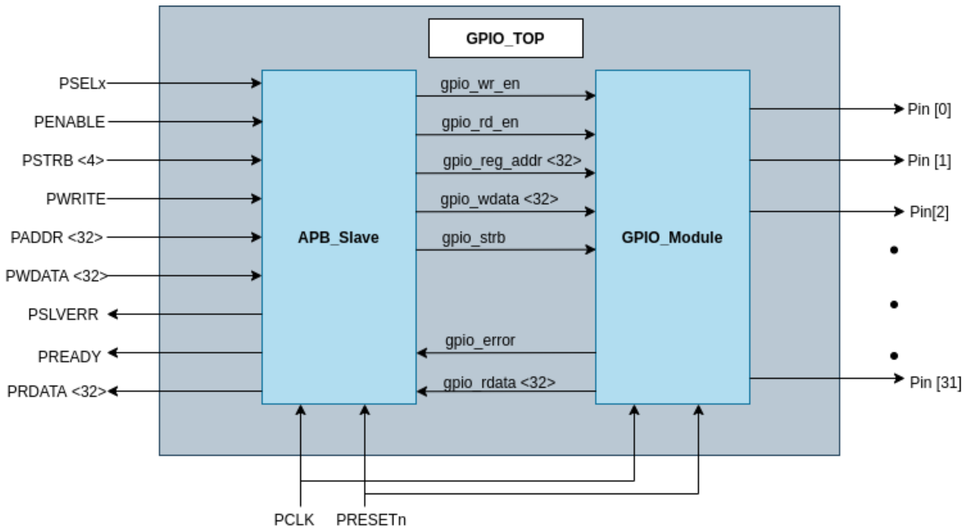

Figure 1: Block diagram of APB_GPIO

APB slave module is an interface that follows APB protocol to drive GPIO module. GPIO module performs read/write on pins, configuration and control registers as per instructions given by APB slave module.

1.2 Features

- GPIOs IP follows AMBA APB protocol
- 32 GPIO pins that can be configured as input/output
- Two operating modes: pushpull and opendrain
- Separate input and output registers
- Input register contains the data of GPIO pins
- Output register to write data on GPIOpins
- Direction register to set the direction of GPIO as input/output

- Set and Clear registers to drive GPIOs additionally
- Mode register for selection of mode

2 Signals Description

Table 1: Port description of GPIO module

Sr. No.	Signal name	Direction	Polarity	Signal Width	Description
1	PRESETn	Input	Active Low	1	Reset is active low. It resets the internal registers of GPIO block.
2	PCLK	Input	Active High	1	System clock
3	PSELx	Input	Active High	1	IP starts its operations of read/write if PSELx & PENABLE are set to HIGH.
4	PENABLE	Input	Active High	1	IP starts its operations of read/write if PSELx & PENABLE are set to HIGH.
5	PSTRB	Input		4	Signal used for modified write operation. Particular byte chunk of data is written whose corresponding PSTRB bit is HIGH.
6	PWRITE	Input	Active High	1	PWRITE is HIGH for write operation and LOW for read operation
7	PADDR	Input		32	Address for register that you want to read/write
8	PWDATA	Input		32	Data to be written on respective register.
9	PSLVERR	Output	Active High	1	The signal is HIGH if any error occurred during operation

11	PRDATA	Output		32	Read data, 32-bit output of the slave.
10	PREADY	Output	Active High	1	The signal becomes HIGH for one cycle after respective operation has been performed. The signal is also used to wait the master, e.g. it is set to 0, if IP require more cycles to perform operation. Master should keep particular signals unchanged unless PREADY is HIGH.
12	PIN[0]... PIN[31]	In-Out		32 pins	32 GPIO pins, can be configured as input/output.

3 GPIO MODULE

GPIO module performs read/write operations on GPIO pins. This module contains different configuration and control registers that are used to perform operations on GPIO pins. Read/Write operations on these registers are performed in this module. Moreover, the logic for the operations on GPIO pins according to the values of registers are also defined in this module.

3.1 Registers in GPIO module

All these registers are 32 bit wide. Each bit corresponds to one gpio pin. These registers and their functionality are mentioned below.

1. **GPIO Direction Register**
   This register configures the behavior of GPIO pins, i.e. input or output. When any particular bit of register is 1 the respective pin is set as OUTPUT, and vice versa. The register can be read/written.

2. **GPIO Output Data Register**
   This register drives the output pins which are configured as Output. Output register holds the data given by the user that should appear at the output pins. This register can be read/written.

3. **GPIO Input Data Register**
   This register takes the input from the all the pins. This register reflects the value on GPIO pins and continue to update itself. This register can only be read.

4. **GPIO Set Register**
   In order to set any output pin to logic 1, we can set the bit of that particular pin to 1 in the GPIO set register. This register can be read/written.

5. **GPIO Clear Register**
   In order to set any output pin to logic 0, we can set the bit of that particular pin to 1 in the GPIO clear register. This register can be read/written.

6. **GPIO Mode Register**
   Mode register is used to operate GPIOs as push-pull mode or open-drain mode. Each bit of mode register corresponds to particular pin of the IP. If a bit of the register is set to 1 then the respective pin of IP works in open drain mode. When set to 0, it works in push-pull mode. The register can be read/written.

3.2 GPIO Registers memory map

Follwing table shows internal addresses of gpio registers.

Table 2: Memory mapping of GPIO registers

Sr. No.	Register name	Address
1	dir	0
2	in	1
3	out	2
4	set	3
5	clr	4
6	mode	5

3.3 Logics in GPIO module

1. **Output Logic** GPIO_oe[n] is the enable of a tristate buffer and is driven by the direction register (dir_reg[n]). GPIO_oe[n] is used to configure the pin as input/output. If GPIO_oe[n] is HIGH then the particular pin will act as output and derived by GPIO_o[n], otherwise the pin will act as input. GPIO_o[n] and GPIO_oe[n] are driven by output and direction registers respectively according to the selected mode.

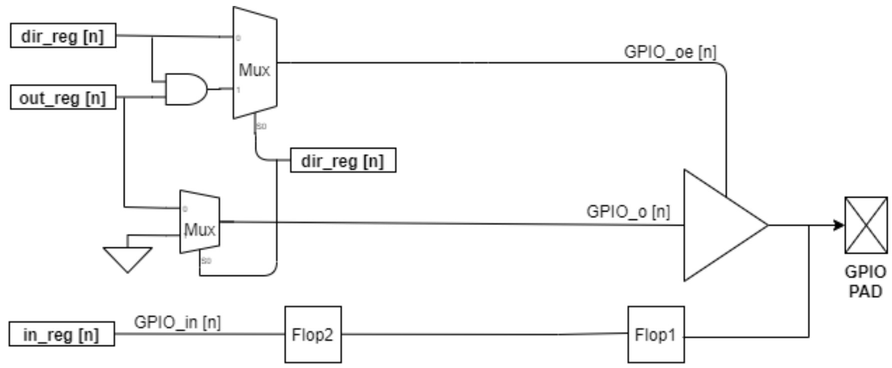

Figure 2: Buffer to drive output logic

2. **Mode selection Logic**
   GPIO pins can be configured in two modes i.e open drain and push-pull mode. The modes can be configured only if the pin is configured as an output and has no effect on input pins. Modes are selected on the basis of value in "mode_reg[n]". If mode_reg[n] is asserted, then pin[n] will be configured in open-drain mode otherwise if mode_reg[n] is set to "0" then pin[n] will be used in push-pull mode.

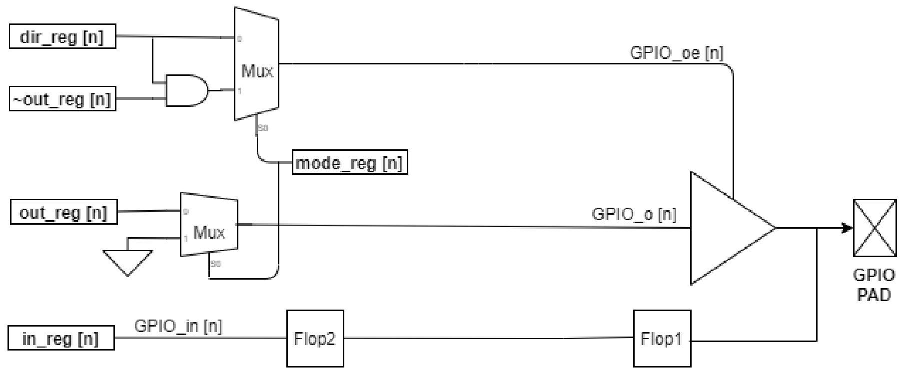

Figure 3: Buffer to drive output logic

Figure 3 shows the complete logic for mode selection of a specific pin. Mode_reg[n] is a select line for two muxes. In push-pull mode, mode_reg[n] is asserted and out_reg[n] drives the pin[n]. However, in open-drain mode mode_reg[n] is deasserted and if the out_reg[n] is "1" then it won't enable the tri-state buffer and pin[n] will not be connected to ground. If out_reg[n] is "0" which means the user wants to drive that pin[n] low, therefore in this case tri-state buffer will be enabled and pin[n] will be pulled to ground i.e open drain.

4 FUTURE WORK AND IMPROVEMENTS

Addition of Interrupt feature
In order to implement interrupts, we have to design general interrupt registers for interrupt

detection. First of all, we set the GPIO DIRECTION REGISTER as an input because we detect interrupts on input signals. Next, we have SET/CLEAR data registers in order to write the particular value. After that we assign the type of the register in GPIO INTERRUPT TYPE REGISTER i-e when this register is 0, we detect low level or high level. When this register is 1, we detect a rising edge and falling edge. The final STATUS will be taken from GPIO STATUS register i-e when we have any interrupt then status register will be HIGH otherwise LOW. The whole setup of interrupt detection is enabled by GPIO INTERRUPT ENABLE REGISTER. When this register is 1, we detect the interrupts and when this register is 0, we ignore the interrupts.
We can design our system for following interrupts signals which are mentioned below

i. **Rising Edge**
In order to detect this interrupt, the GPIO INTERRUPT TYPE Register and GPIO RISING/level1 EDGE Register are set high.

ii. **Falling Edge**
In order to detect this interrupt, the GPIO INTERRUPT TYPE Register is set High and GPIO Falling/level0 EDGE Register is set low.

iii. **Low Level**
In order to detect this interrupt, the GPIO INTERRUPT TYPE Register is set low and GPIO FALLING/LEVEL0 EDGE Register is set high.

iv. **High Level**
In order to detect this interrupt, the GPIO INTERRUPT TYPE Register is set low and GPIO RISING/LEVEL1 EDGE Register is set high.

Following will be the required registers used for implementation of interrupt logic.

i. **GPIO Interrupt Type Register**
When the value of this register is 0, we will detect LEVEL and when it is 1, we will detect the edge. We have separate registers for further classification of interrupts i-e high/low level or rising/falling edge.

ii. **GPIO LEVEL0/FALLING edge Register**
When the interrupt type is 0 and we have to detect low level then we set this register to 1, else if we have to detect high level then we set this register to 0. Moreover, if the interrupt type is 1, this means that we are detecting edges. If we have to detect a falling edge then this register value is set to 1 else is 0 for rising edge.

iii. **GPIO LEVEL1/RISING edge Register**
When the interrupt type is 0 and we have to detect high level then we set this register to 1, else if we have to detect low level then we set this register to 0. Moreover, if the interrupt type is 1, this means that we are detecting edges. If we have to detect a rising edge then this register value is set to 1 else 0.

iv. **GPIO Interrupt Status Register**
This register tells the status of interrupts i-e when there is any interrupt the status register will set high else 0.

v. **GPIO Interrupt Enable Register**
In order to implement the functionality of interrupts this register value must be set high else 0.