

up_gpio.v

AUTHORS

JAY CONVERTINO

DATES

2024/07/25

INFORMATION

Brief

uP Core for interfacing with general purpose input/output.

License MIT

Copyright 2024 Jay Convertino

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

up_gpio

```
module up_gpio #(
    parameter
    ADDRESS_WIDTH
    =
    32,
    parameter
    BUS_WIDTH
    =
    4,
    parameter
    GPIO_WIDTH
```

```

    =
    32,
    parameter
    IRQ_ENABLE
    =
    0
) ( input clk, input rstn, input up_rreq, output up_rack, input [ADDRESS_WID

```

uP based GPIO device.

Parameters

ADDRESS_WIDTH parameter	Width of the uP address port.
BUS_WIDTH parameter	Width of the uP bus data port.
GPIO_WIDTH parameter	Width of the GPIO for inputs and outputs
IRQ_ENABLE parameter	Enable interrupt

Ports

clk	Clock for all devices in the core
rstn	Negative reset
up_rreq	uP bus read request
up_rack	uP bus read ack
up_raddr	uP bus read address
up_rdata	uP bus read data
up_wreq	uP bus write request
up_wack	uP bus write ack
up_waddr	uP bus write address
up_wdata	uP bus write data
irq	Interrupt when data is received
gpio_io_i	Input for GPIO
gpio_io_o	Output for GPIO
gpio_io_t	Tristate for GPIO

REGISTER INFORMATION

Core has 4 registers at the offsets that follow.

GPIO_DATA	h000
GPIO_TRI	h004
GPIO2_DATA	h008 N/A
GPIO2_TRI	h00C N/A
GIER	h11C
IP_ISR	h120
IP_IER	h128

GPIO_DATA

```
localparam GPIO_DATA = 12'h000
```

Defines the address offset for GPIO DATA

GPIO DATA
31:0
GPIO_WIDTH:0

Valid bits are from GPIO_WIDTH:0, input or output data.

GPIO_TRI

```
localparam GPIO_TRI = 12'h004
```

Defines the address offset for GPIO TRI.

GPIO TRI
31:0
GPIO_WIDTH:0

Valid bits are from GPIO_WIDTH:0, 1 indicates input, 0 is output.

GPIO2_DATA

```
localparam GPIO2_DATA = 12'h008
```

Defines the address offset for GPIO2 DATA

GPIO2 DATA
31:0
UNUSED

Valid bits are from GPIO2_WIDTH:0, input or output data. This Register is not implimented in this design.

GPIO2_TRI

```
localparam GPIO2_TRI = 12'h00C
```

Defines the address offset for GPIO2 TRI.

GPIO2 TRI
31:0
UNUSED

Valid bits are from GPIO2_WIDTH:0, 1 indicates input, 0 is output. This register is not implimented in this design.

GIER

```
localparam GIER = 12'h11C
```

Defines the address offset for GIER.

GIER	
31	30:0
Global IRQ Ena	UNUSED

Bit 31 is the Global interrupt enable. Write a 1 to enable interrupts.

IP_ISR

```
localparam IP_ISR = 12'h120
```

Defines the address offset for IP_ISR.

IP ISR	
31:1	0
UNUSED	IRQ Status

Bit 0 is GPIO IRQ status, On write this will toggle(acknowledge) the interrupt.

IP_IER

```
localparam IP_IER = 12'h128
```

Defines the address offset to set the control bits.

IP_IER	
31:1	0
UNUSED	IRQ Ena

Bit 0 is GPIO IRQ enable interrupt. Write a 1 to bit 0 to enable interrupt.