

up_apb3.v

AUTHORS

JAY CONVERTINO

DATES

2024/03/19

INFORMATION

Brief

APB3 slave to uP interface

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_apb3

```
module up_apb3 #(
    parameter
    ADDRESS_WIDTH
    =
    16,
    parameter
    BUS_WIDTH
    =
    4
) ( input clk, input rst, input [ADDRESS_WIDTH-1:0] s_apb_paddr, input [0:0]
```

APB3 slave to uP interface

Parameters

ADDRESS_WIDTH parameter	Width of the APB3 address port in bits.
BUS_WIDTH parameter	Width of the APB3 bus data port in bytes.

Ports

clk	Clock
rst	Positive reset
s_apb_paddr	APB3 address bus, up to 32 bits wide.
s_apb_psel	APB3 select per slave (1 for this core).
s_apb_penable	APB3 enable device for multiple transfers after first.
s_apb_pready	APB3 ready is a output from the slave to indicate its able to process the request.
s_apb_pwrite	APB3 Direction signal, active high is a write access. Active low is a read access.
s_apb_pwdata	APB3 write data port.
s_apb_prdata	APB3 read data port.
s_apb_pslverror	APB3 error indicates transfer failure, not implimented.
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

VARIABLES

valid

```
assign valid = s_apb_psel & s_apb_penable
```

This will add an extra clock cycle. since enable happens after select. both are needed to use the device.

s_apb_pslverror

```
assign s_apb_pslverror = 1'b0
```

APB3 error is always 0, no error.

up_waddr

```
assign up_waddr = s_apb_paddr
```

up_waddr and s_apb_addr are a direct mapping.

up_raddr

up_raddr and s_apb_addr are a direct mapping.

up_wdata

```
assign up_wdata = s_apb_pdata
```

up_wdata and s_apb_pdata are a direct mapping.

s_apb_prdata

```
assign s_apb_prdata = up_rdata
```

s_apb_prdata and up_rdata are a direct mapping.

up_wreq

```
assign up_wreq = valid & s_apb_pwrite
```

uP write request is a combination of the APB3 valid and APB3 write select (active high is write).

up_rreq

```
assign up_rreq = valid & ~s_apb_pwrite
```

uP read request is a combination of the APB3 valid and APB3 write select (active low is read).

s_apb_pready

```
assign s_apb_pready = up_wack | up_rack | ~valid
```

Diagrams seem to indicate that we should indicate ready when not sel and enable, which is why valid is complimented.