wishbone_classic_1553.v

AUTHORS

JAY CONVERTINO

DATES

2024/10/17

INFORMATION

Brief

wishbone classic to uP core for 1553 comms.

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.

wishbone_standard_1553

```
module wishbone_standard_1553 #(
parameter
ADDRESS_WIDTH
 =
32,
parameter
BUS_WIDTH
 =
4,
parameter
CLOCK_SPEED
 =
100000000
)
```

```
                                                                (
 input
 clk,
 input
 rst,
 input
 s_wb_cyc,
 input
 s_wb_stb,
 input
 s_wb_we,
 input
  [ADDRESS_WIDTH-1:0]
 s_wb_addr,
 input
  [BUS_WIDTH*8-1:0]
 s_wb_data_i,
 input
  [BUS_WIDTH-1:0]
 s_wb_sel,
 output
 s_wb_ack,
 output
  [BUS_WIDTH*8-1:0]
 s_wb_data_o,
 output
 s_wb_err,
 input
  [1:0]
 rx_diff,
 output
  [1:0]
 tx_diff,
 output
 tx_active,
 output
 irq
)
```

Wishbone Stanard based 1553 communications device.

Parameters

ADDRESS_WIDTH
parameter
Width of the address bus in bits, max 32 bit.

BUS_WIDTH
parameter
Width of the data bus in bytes.

CLOCK_SPEED
parameter
This is the aclk frequency in Hz

Ports

clk
input
Clock for all devices in the core

rst
input
Positive reset

s_wb_cyc
input
Bus Cycle in process

s_wb_stb
input
Valid data transfer cycle

s_wb_we
input
Active High write, low read

s_wb_addr `input [ADDRESS_WIDTH- 1:0]`	Bus address
s_wb_data_i `input [BUS_WIDTH* 8- 1:0]`	Input data
s_wb_sel `input [BUS_WIDTH- 1:0]`	Device Select
s_wb_ack `output [BUS_WIDTH- 1:0]`	Bus transaction terminated
s_wb_data_o `output [BUS_WIDTH* 8- 1:0]`	Output data
s_wb_err `output [BUS_WIDTH* 8- 1:0]`	Active high when a bus error is present
rx_diff `input [1:0]`	Input differential signal for 1553 bus
tx_diff `output [1:0]`	Output differential signal for 1553 bus
tx_active `output [1:0]`	Enable output of differential signal (for signal switching on 1553 module)
irq `output [1:0]`	Interrupt when data is received

up_rreq

```
wire up_rreq
```

uP read bus request

up_rack

```
wire up_rack
```

uP read bus acknowledge

up_raddr

```
wire [ADDRESS_WIDTH-(
BUS_WIDTH
                                                                    /
2
)-1:0] up_raddr
```

uP read bus address

up_rdata

```
wire [31:0] up_rdata
```

uP read bus request

up_wreq

```
wire up_wreq
```

uP write bus request

up_wack

```
wire up_wack
```

uP write bus acknowledge

up_waddr

```
wire [ADDRESS_WIDTH-(
BUS_WIDTH
                                                                    /
2
)-1:0] up_waddr
```

uP write bus address

up_wdata

```
wire [31:0] up_wdata
```

uP write bus data

INSTANTIANTED MODULES

inst_up_wishbone_standard

Module instance of up_wishbone_standard for the Wishbone Classic Standard bus to the uP bus.

inst_up_1553

Module instance of up_1553 creating a Logic wrapper for 1553 bus cores to interface with uP bus.