

# axi\_lite\_1553.v

---

## AUTHORS

---

JAY CONVERTINO

---

## DATES

---

2024/10/17

---

## INFORMATION

---

### Brief

---

AXI Lite 1553 is a core for interfacing with 1553 devices over the AXI lite bus.

### 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.

## axi\_lite\_1553

---

```
module axi_lite_1553 #(
  parameter
  ADDRESS_WIDTH
  =
  32,
  parameter
  CLOCK_SPEED
  =
  100000000,
  parameter
  SAMPLE_RATE
  =
  2000000,
  parameter
```

```

    BIT_SLICE_OFFSET
    =
    0,
    parameter
    INVERT_DATA
    =
    0,
    parameter
    SAMPLE_SELECT
    =
    0
) ( input aclk, input arstn, input s_axi_awvalid, input [ADDRESS_WIDTH-1:0]

```

AXI Lite based 1553 communications device.

## Parameters

<b>ADDRESS_WIDTH</b> parameter	Width of the axi address bus, max 32 bit.
<b>CLOCK_SPEED</b> parameter	This is the aclk frequency in Hz
<b>SAMPLE_RATE</b> parameter	Rate of in which to sample the 1553 bus. Must be 2 MHz or more and less than aclk. This is in Hz. BIT_SLICE_OFFSET- Adjust where the sample is taken from the input.
<b>INVERT_DATA</b> parameter	Invert all 1553 bits coming in and out.
<b>SAMPLE_SELECT</b> parameter	Adjust where in the array of samples to select a bit.

## Ports

<b>aclk</b>	Clock for all devices in the core
<b>arstn</b>	Negative reset
<b>s_axi_awvalid</b>	Axi Lite aw valid
<b>s_axi_awaddr</b>	Axi Lite aw addr
<b>s_axi_awprot</b>	Axi Lite aw prot
<b>s_axi_awready</b>	Axi Lite aw ready
<b>s_axi_wvalid</b>	Axi Lite w valid
<b>s_axi_wdata</b>	Axi Lite w data
<b>s_axi_wstrb</b>	Axi Lite w strb
<b>s_axi_wready</b>	Axi Lite w ready
<b>s_axi_bvalid</b>	Axi Lite b valid
<b>s_axi_bresp</b>	Axi Lite b resp
<b>s_axi_bready</b>	Axi Lite b ready
<b>s_axi_arvalid</b>	Axi Lite ar valid
<b>s_axi_araddr</b>	Axi Lite ar addr
<b>s_axi_arprot</b>	Axi Lite ar prot
<b>s_axi_arready</b>	Axi Lite ar ready
<b>s_axi_rvalid</b>	Axi Lite r valid
<b>s_axi_rdata</b>	Axi Lite r data
<b>s_axi_rresp</b>	Axi Lite r resp
<b>s_axi_rready</b>	Axi Lite r ready
<b>i_diff</b>	Input differential signal for 1553 bus

<b>o_diff</b>	Output differential signal for 1553 bus
<b>en_o_diff</b>	Enable output of differential signal (for signal switching on 1553 module)
<b>irq</b>	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

## INSTANTIATED MODULES

### inst\_up\_axi

```
up_axi #(
    AXI_ADDRESS_WIDTH(ADDRESS_WIDTH)
) inst_up_axi ( .up_rstn (arstn), .up_clk (aclk), .up_axi_awvalid(s_axi_awv
```

Module instance of up\_axi for the AXI Lite bus to the uP bus.

### inst\_up\_1553

```
up_1553 #(
    ADDRESS_WIDTH(ADDRESS_WIDTH),
    CLOCK_SPEED(CLOCK_SPEED),
    SAMPLE_RATE(SAMPLE_RATE),
    BIT_SLICE_OFFSET(BIT_SLICE_OFFSET),
    INVERT_DATA(INVERT_DATA),
    SAMPLE_SELECT(SAMPLE_SELECT)
) inst_up_1553 ( .clk(aclk), .rstn(arstn), .up_rreq(up_rreq), .up_rack(up_rack
```

Module instance of up\_1553 creating a Logic wrapper for 1553 bus cores to interface with uP bus.