

fast_axis_uart.v

AUTHORS

JAY CONVERTINO

DATES

2025/06/11

INFORMATION

Brief

Fast UART AXIS core that allows for back to back transmissions.

License MIT

Copyright 2025 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.

fast_axis_uart

```
module fast_axis_uart #(
    parameter
    CLOCK_SPEED
    =
    20000000,
    parameter
    BAUD_RATE
    =
    20000000,
    parameter
    PARITY_TYPE
    =
    0,
    parameter
```

```

STOP_BITS
=
1,
parameter
DATA_BITS
=
8,
parameter
RX_BAUD_DELAY
=
0,
parameter
TX_BAUD_DELAY
=
0
) ( input aclk, input arstn, output parity_err, output frame_err, input [ 7:

```

AXIS UART, fast simple UART with AXI Streaming interface.

Parameters

CLOCK_SPEED parameter	This is the aclk frequency in Hz
BAUD_RATE parameter	Serial Baud, this can be any value including non-standard.
PARITY_TYPE parameter	Set the parity type, 0 = none, 1 = odd, 2 = even, 3 = mark, 4 = space.
STOP_BITS parameter	Number of stop bits, 0 to crazy non-standard amounts.
DATA_BITS parameter	Number of data bits, 1 to 8.
RX_BAUD_DELAY parameter	Delay in rx baud enable. This will delay when we sample a bit (default is midpoint when rx delay is 0).
TX_BAUD_DELAY parameter	Delay in tx baud enable. This will delay the time the bit output starts.

Ports

aclk	Clock for AXIS
arstn	Negative reset for AXIS
parity_err	Indicates error with parity check (active high)
frame_err	Indicates error with frame (active high)
s_axis_tdata	Input data for UART TX.
s_axis_tvalid	When set active high the input data is valid
s_axis_tready	When active high the device is ready for input data.
m_axis_tdata	Output data from UART RX
m_axis_tvalid	When active high the output data is valid
m_axis_tready	When set active high the output device is ready for data.
tx	transmit for UART (output to RX)
rx	receive for UART (input from TX)

INSTANTIATED MODULES

uart_baud_gen_tx

```
mod_clock_ena_gen #(
    CLOCK_SPEED(CLOCK_SPEED),
    DELAY(TX_BAUD_DELAY)
) uart_baud_gen_tx ( .clk(aclk), .rstn(arstn), .start0(1'b1), .clr(1'b0), .f
```

Generates TX BAUD rate for UART modules using modulo divide method.

uart_baud_gen_rx

```
mod_clock_ena_gen #(
    CLOCK_SPEED(CLOCK_SPEED),
    DELAY(RX_BAUD_DELAY)
) uart_baud_gen_rx ( .clk(aclk), .rstn(arstn), .start0(1'b0), .clr(r_rx_clr)
```

Generates RX BAUD rate for UART modules using modulo divide method.

inst_sipo

```
sipo #(
    BUS_WIDTH(32),
    COUNT_AMOUNT(BITS_PER_TRANS)
) inst_sipo ( .clk(aclk), .rstn(arstn), .ena(uart_ena_rx), .rev(1'b1), .load
```

Captures RX data for uart receive

inst_piso

```
piso #(
    BUS_WIDTH(32),
    COUNT_AMOUNT(BITS_PER_TRANS),
    DEFAULT_RESET_VAL(1),
    DEFAULT_SHIFT_VAL(1)
) inst_piso ( .clk(aclk), .rstn(arstn), .ena(uart_ena_tx), .rev(1'b1), .load
```

Generates TX data for uart transmit