UART_PMOD1553

AIR FORCE RESEARCH LABORATORY

November 25, 2024

Jay Convertino

Contents

1 Usage **2**
1.1 Introduction 2
1.2 Interface 2
1.3 Dependencies 3
1.3.1 fusesoc_info Depenecies 3

2 Architecture **3**

3 Building **4**
3.1 fusesoc 4
3.2 Source Files 4
3.2.1 fusesoc_info File List 4
3.3 Targets 4
3.3.1 fusesoc_info Targets 4
3.4 Directory Guide 5

4 Simulation **6**

5 Module Documentation **7**
5.1 uart 1553 core 8
5.2 axis 1553 string decoder 15
5.3 axis 1553 string encoder 17
5.4 axis char to string converter 19
5.5 ARTY system 21
5.6 CMOD system 24
5.7 NEXYS system 27

1 Usage

1.1 Introduction

UART pmod1553 is a fusesoc 1553 project that has all input/output data through the UART. The 1553 is for avaiation bus connections. The UART is a simple interface for sending and receiving data, and does not emulate a bus endpoint or controller. Software that reads and writes from the UART is responsible for that response.

1.2 Interface

For xilinx fifo and uart the format is the same. Data is received in the following ASCII string format:

```
$ CMDS;D1;P1;I0;Hx5555\r
$ DATA;D0;P1;I0;HxAAAA\r
```

The fields are seperated by ; .

- The first is the sync type, Command/Status = CMDS, Data = DATA
- The second is if there is a 4us delay, 1 = delay over 4us, 0 = no delay or less then 4us.
- The third is parity, 1 = parity good, 0 = parity bad
- The fourth is invert, 1 = core is inverting data, 0 = core is not inverting data
- The fifth is the data in hex format, Hx???? where ? = 4 characters representing the data (16 bits in total).
- The carrige return is the string terminator. This works well with serial consoles with local newline addition enabled.

Data is sent in the following ASCII string format:

```
$ CMDS;D1;P1;I0;Hx5555\r
$ DATA;D0;P1;I0;HxAAAA\r
```

The fields are seperated by ; .

- The first is the sync type, Command/Status = CMDS, Data = DATA
- The second is to enable a 4us delay, 1 = delay of at least 4us, 0 = no delay or less then 4us.
- The third is parity, 1 = parity odd (default), 0 = parity even

- The fourth is invert, 1 = invert data, 0 = don't invert data
- The fifth is the data in hex format, Hx???? where ? = 4 characters representing the data (16 bits in total).
- The carrige return is the string terminator.

1.3 Dependencies

The following are the dependencies of the cores.

- fusesoc 2.X
- iverilog (simulation)
- cocotb (simulation)

1.3.1 fusesoc_info Depenecies

- nexys-a7-100t
  - AFRL:utility:digilent_nexys-a7-100t_board_base:1.0.0
- cmod-s7-25
  - AFRL:utility:digilent_cmod-s7-25_board_base:1.0.0
- arty-a7-35
  - AFRL:utility:digilent_arty-a7-35_board_base:1.0.0
- dep
  - AFRL:project_ip:uart_1553_core:1.0.0
  - AFRL:utility:helper:1.0.0
  - AFRL:utility:vivado_board_support_packages

2 Architecture

The project contains wrappers for the top level projects, and a common uart 1553 module for all.

- **system_wrapper** Contains the top level project module and contains uart_1553_core and any other logic needed.
- **uart_1555_core** Contains the PMOD1553, UART, and string generation interfaces.

Please see 5 for more information per target.

3 Building

The all UART pmod1553 is written in Verilog 2001. They should synthesize in any modern FPGA software. The core comes as a fusesoc packaged core and can be included in any other core. Be sure to make sure you have meet the dependencies listed in the previous section.

3.1 fusesoc

Fusesoc is a system for building FPGA software without relying on the internal project management of the tool. Avoiding vendor lock in to Vivado or Quartus. These cores, when included in a project, can be easily integrated and targets created based upon the end developer needs. The core by itself is not a part of a system and should be integrated into a fusesoc based system. Simulations are setup to use fusesoc and are a part of its targets.

3.2 Source Files

3.2.1 fusesoc_info File List

- src_xilinx
  - ’common/xilinx/system_gen.tcl’: ’file_type’: ’tclSource’
- nexys-a7-100t
  - ’nexys-a7-100t/system_constr.xdc’: ’file_type’: ’xdc’
  - ’nexys-a7-100t/system_wrapper.v’: ’file_type’: ’verilogSource’
- cmod-s7-25
  - ’cmod-s7-25/system_constr.xdc’: ’file_type’: ’xdc’
  - ’cmod-s7-25/system_wrapper.v’: ’file_type’: ’verilogSource’
- arty-a7-35
  - ’arty-a7-35/system_constr.xdc’: ’file_type’: ’xdc’
  - ’arty-a7-35/system_wrapper.v’: ’file_type’: ’verilogSource’

3.3 Targets

3.3.1 fusesoc_info Targets

- default

  Info: Default files, invalid gen target.

- nexys-a7-100t

  Info: Nexyx a7 100t dev board target.

- cmod-s7-25

  Info: cmod s7 25 dev board target.

- arty-a7-35

  Info: arty a7 35 dev board target.

3.4 Directory Guide

Below highlights important folders from the root of the directory.

1. **docs** Contains all documentation related to this project.
   - **manual** Contains user manual and github page that are generated from the latex sources.
2. **arty-a7-35** Contains source files for arty-a7-35
3. **common** Contains source file wrapper and helper cores for uart PMOD1553 core
4. **cmod-s7-25** Contains source files for cmod-s7-25
5. **nexys-a7-100t** Contains source files for nexys-a7-100t

4 Simulation

There is no simulation at the moment. This is dues to the AD9361 and ARM subsystems. Maybe a future addition with Vexriscv?

5 Module Documentation

There project has multiple modules. The targets are the top system wrappers.

- **arty-a7-35 system wrapper**
- **cmod-s7-25 system wrapper**
- **nexys-a7-100t system wrapper**
- **uart 1553 core wrapper**
- **axis 1553 string decoder**
- **axis 1553 string encoder**
- **axis char to string converter**

The next sections document the modules.

uart_1553_core.v

AUTHORS

JAY CONVERTINO

DATES

2021/06/28

INFORMATION

Brief

Core that ties together all ips into a single uart to 1553 core.

License MIT

uart_1553_core

```
module uart_1553_core #(
parameter
clock_speed
 =
2000000,
parameter
uart_baud_clock_speed
 =
2000000,
parameter
uart_baud_rate
```

```
 =
2000000,
parameter
uart_parity_ena
 =
0,
parameter
uart_parity_type
 =
0,
parameter
uart_stop_bits
 =
1,
parameter
uart_data_bits
 =
8,
parameter
uart_rx_delay
 =
0,
parameter
uart_tx_delay
 =
0,
parameter
mil1553_sample_rate
 =
2000000,
parameter
mil1553_rx_bit_slice_offset
 =
0,
parameter
mil1553_rx_invert_data
 =
0,
parameter
mil1553_rx_sample_select
 =
0
) ( input aclk, input arstn, input uart_clk, input uart_rstn, input rx_UART,
```

Core that ties together all ips into a single uart to 1553 core.

Parameters

clock_speed = 2000000	Requested Master Clock Speed from clk wiz
uart_baud_clock_speed parameter	UART Master Clock Speed
uart_baud_rate parameter	UART BAUD rate
uart_parity_ena parameter	UART Parity enable, active high.
uart_parity_type parameter	UART Parity type
uart_stop_bits parameter	UART Number of stop bits.
uart_data_bits parameter	UART Number of data bits.

uart_rx_delay parameter	UART RX Delay to align data.
uart_tx_delay parameter	UART TX Delay to align data.
mil1553_sample_rate parameter	Sample rate for 1553, must be 2 MHz or above, and divide evenly into clock_speed.
mil1553_rx_bit_slice_offset parameter	1553 change the offset of the receive bit taken from the inital sampling.
mil1553_rx_invert_data parameter	Invert 1553 data received.
mil1553_rx_sample_select parameter	1553 select sample from initial sampling.

Ports

aclk	Master Clock
arstn	Base Reset
uart_clk	UART Master Clock
uart_rstn	UART reset
rx_UART	UART RX input
tx_UART	UART TX output
rts_UART	UART request to send
cts_UART	UART clear to send
rx0_1553	PMOD1553 RX diff
rx1_1553	PMOD1553 RX diff
tx0_1553	PMOD1553 TX diff
tx1_1553	PMOD1553 TX diff
en_tx_1553	PMOD1553 enable transmit on mux.

INSTANTIANTED MODULES

mil1553_decoder

```
 axis_1553_decoder #(
                                                                        .
 CLOCK_SPEED(clock_speed),
                                                                        .
 SAMPLE_RATE(mil1553_sample_rate),
                                                                        .
 BIT_SLICE_OFFSET(mil1553_rx_bit_slice_offset),
                                                                        .
 INVERT_DATA(mil1553_rx_invert_data),
                                                                        .
 SAMPLE_SELECT(mil1553_rx_sample_select)
) mil1553_decoder ( .aclk(aclk), .arstn(arstn), .m_axis_tdata(m1553_decoder_
```

Module mil-std-1553 decoder capable of any clock rate at or above 2 MHz

decoder_fifo

```
axis_fifo #(
                                                                                              .
FIFO_DEPTH
                                                                                              (
256),
                                                                                              .
COUNT_WIDTH
                                                                                              (
0),
                                                                                              .
BUS_WIDTH
                                                                                              (
2),
                                                                                              .
USER_WIDTH
                                                                                              (
8),
                                                                                              .
DEST_WIDTH
                                                                                              (
1),
                                                                                              .
RAM_TYPE
                                                                                             ("
block"),
                                                                                              .
PACKET_MODE
                                                                                              (
0),
                                                                                              .
COUNT_DELAY
                                                                                              (
0),
                                                                                              .
COUNT_ENA
                                                                                              (
0)
) decoder_fifo ( .s_axis_aclk(aclk), .s_axis_arstn(arstn), .s_axis_tvalid(m
```

FIFO for decoder data output

string_encoder

```
axis_1553_string_encoder string_encoder (
                                                                                              .
aclk(aclk),
                                                                                              .
arstn(arstn),
                                                                                              .
s_axis_tdata(mfifo_decoder_data),
                                                                                              .
s_axis_tvalid(mfifo_decoder_valid),
                                                                                              .
s_axis_tuser(mfifo_decoder_user),
                                                                                              .
s_axis_tready(mfifo_decoder_ready),
                                                                                              .
m_axis_tdata(mstring_encoder_data),
                                                                                              .
m_axis_tvalid(mstring_encoder_valid),
                                                                                   .
 m_axis_tready(mstring_encoder_ready)
)
```

1553 to string core

string_to_char

```
 axis_data_width_converter #(
                                                                                   .
 SLAVE_WIDTH(21),
                                                                                   .
 MASTER_WIDTH(1),
                                                                                   .
 REVERSE(1)
) string_to_char ( .aclk(aclk), .arstn(arstn), .s_axis_tdata(mstring_encoder
```

data width converter

outgoing_char_fifo

```
 axis_tiny_fifo #(
                                                                                   .
 FIFO_DEPTH(4),
                                                                                   .
 BUS_WIDTH(8)
) outgoing_char_fifo ( .aclk(aclk), .arstn(arstn), .s_axis_tdata(mstring_to_
```

fifo for 1553 encoded into character string.

string_to_char

AXIS UART

incomming_char_fifo

```
 axis_tiny_fifo #(
                                                                                   .
 FIFO_DEPTH(4),
                                                                                   .
 BUS_WIDTH(8)
) incomming_char_fifo ( .aclk(aclk), .arstn(arstn), .s_axis_tdata(muart_char
```

fifo for chars to be decoded into 1553 data.

char_to_string

```
 axis_char_to_string_converter #(
                                                                                   .
 master_width(21)
 ) char_to_string ( .aclk(aclk), .arstn(arstn), .s_axis_tdata(min_char_fifo_
```

data width converter

char_to_string

```
axis_1553_string_decoder string_decoder (
                                                                       .
aclk(aclk),
                                                                       .
arstn(arstn),
                                                                       .
s_axis_tdata(mchar_to_string_data),
                                                                       .
s_axis_tvalid(mchar_to_string_valid),
                                                                       .
s_axis_tready(mchar_to_string_ready),
                                                                       .
m_axis_tdata(mstring_decoder_data),
                                                                       .
m_axis_tvalid(mstring_decoder_valid),
                                                                       .
m_axis_tuser(mstring_decoder_user),
                                                                       .
m_axis_tready(mstring_decoder_ready)
)
```

string to 1553

encoder_fifo

```
axis_fifo #(
                                                                       .
FIFO_DEPTH
                                                                       (
256),
                                                                       .
COUNT_WIDTH
                                                                       (
0),
                                                                       .
BUS_WIDTH
                                                                       (
2),
                                                                       .
USER_WIDTH
                                                                       (
8),
                                                                       .
DEST_WIDTH
                                                                       (
1),
                                                                       .
RAM_TYPE
                                                                      ("
block"),
                                                                       .
PACKET_MODE
                                                                       (
0),
                                                                       .
COUNT_DELAY
                                                                       (
 0),
                                                                            .
 COUNT_ENA
                                                                            (
 0)
) encoder_fifo ( .s_axis_aclk(aclk), .s_axis_arstn(arstn), .s_axis_tvalid(ms
```

fifo for decoder data

encoder_fifo

mil-std-1553 encoder capable of any clock rate at or over 2 MHz

axis_1553_string_decoder.v

AUTHORS

JAY CONVERTINO

DATES

2021/06/21

INFORMATION

Brief

Carrige return terminated string converted to 1553 data.

License MIT

axis_1553_string_decoder

```
module axis_1553_string_decoder #(
parameter
byte_swap
 =
0
) ( input aclk, input arstn, input [167:0] s_axis_tdata, input s_axis_tval
```

Carrige return terminated string converted to 1553 data.

Parameters

byte_swap parameter	swap input byte order.

Ports

aclk	Master Clock
arstn	Negative Reset
s_axis_tdata	Input raw data
s_axis_tvalid	Input raw data is valid
s_axis_tuser	Input raw user field
s_axis_tready	Input ready for characters
m_axis_tdata	Output string.
m_axis_tvalid	Output string is valid
m_axis_tlast	Indicates output string termination
m_axis_tkeep	What bytes are valid characters in the string.
m_axis_tready	Is the next device ready for output?

axis_1553_string_encoder.v

AUTHORS

JAY CONVERTINO

DATES

2021/06/21

INFORMATION

Brief

1553 data is converted to a string. The string is carrige return terminated.

License MIT

axis_1553_string_encoder

```
module axis_1553_string_encoder #(
parameter
byte_swap
 =
0
) ( input aclk, input arstn, input [ 15:0] s_axis_tdata, input s_axis_tval
```

1553 data is converted to a string. The string is carrige return terminated.

Parameters

byte_swap parameter	swap output byte order.

Ports

aclk	Master Clock
arstn	Negative Reset
s_axis_tdata	Input raw data
s_axis_tvalid	Input raw data is valid
s_axis_tuser	Input raw user field
s_axis_tready	Input ready for characters
m_axis_tdata	Output string.
m_axis_tvalid	Output string is valid
m_axis_tlast	Indicates output string termination
m_axis_tkeep	What bytes are valid characters in the string.
m_axis_tready	Is the next device ready for output?

axis_char_to_string_converter.v

AUTHORS

JAY CONVERTINO

DATES

2021/06/21

INFORMATION

Brief

Convert characters to a string. Output valid on carrige return or full.

License MIT

axis_char_to_string_converter

```
module axis_char_to_string_converter #(
parameter
master_width
 =
1
) ( input aclk, input arstn, input [ 7:0] s_axis_tdata, input s_axis_tvali
```

Convert characters to a string. Output valid on carrige return or full.

Parameters

master_width `parameter`	number of bytes for output data.

Ports

aclk	master clock
arstn	negative reset
s_axis_tdata	Input characters
s_axis_tvalid	Input character valid
s_axis_tready	Input ready for characters
m_axis_tdata	Output string built up of characters.
m_axis_tvalid	Output string is valid
m_axis_tready	Is the next device ready for output?

system_wrapper.v

AUTHORS

JAY CONVERTINO

DATES

2024/11/25

INFORMATION

Brief

System wrapper for pl

License MIT

system_wrapper

```
module system_wrapper #(
parameter
clock_speed
 =
2000000,
parameter
baud_rate
 =
1000000,
parameter
mil1553_sample_rate
```

```
  =
 2000000
) ( input clk, input [1:0] push_buttons, inout [7:0] pmod_ja, input ftdi_tx,
```

System wrapper for pl

Parameters

clock_speed parameter	Requested Master Clock Speed from clk wiz
baud_rate parameter	UART BAUD rate
mil1553_sample_rate parameter	Sample rate for 1553, must be 2 MHz or above, and divide evenly into clock_speed.

Ports

clk	Input clock for all clocks
push_buttons	Buttons used for reset (push button 0).
pmod_ja	1553 PMOD device port (JA)
ftdi_tx	FTDI UART input (TX)
ftdi_rx	FTDI UART output (RX)

INSTANTIANTED MODULES

inst_clk_wiz_1

```
 clk_wiz_1 inst_clk_wiz_1 (
                                                                          .
 clk_out1(sys_clk),
                                                                          .
 reset(push_buttons[0]),
                                                                          .
 clk_in1(clk)
)
```

Module instance of clock wizard to change input clock to requested clock speed.

inst_sys_rstgen

```
 sys_rstgen inst_sys_rstgen (
                                                                          .
 slowest_sync_clk(sys_clk),
                                                                          .
 ext_reset_in(1'b1),
                                                                          .
 aux_reset_in(push_buttons[0]),
                                                                          .
 mb_debug_sys_rst(1'b0),
                                                                          .
 dcm_locked(1'b1),
                                                                          .
 mb_reset(),
                                                                          .
 bus_struct_reset(),
                                                                          .
 peripheral_reset(reset),
                                                                          .
 interconnect_aresetn(),
                                                                          .
 peripheral_aresetn(resetn)
)
```

Module instance of reset gen to create system reset.

inst_uart_1553_core

```
 uart_1553_core #(
                                                                          .
 clock_speed(clock_speed),
                                                                          .
 uart_baud_clock_speed(clock_speed),
                                                                          .
 uart_baud_rate(baud_rate),
                                                                          .
 uart_parity_ena(0),
                                                                          .
 uart_parity_type(0),
                                                                          .
 uart_stop_bits(1),
                                                                          .
 uart_data_bits(8),
                                                                          .
 uart_rx_delay(0),
                                                                          .
 uart_tx_delay(0),
                                                                          .
 mil1553_sample_rate(mil1553_sample_rate),
                                                                          .
 mil1553_rx_bit_slice_offset(0),
                                                                          .
 mil1553_rx_invert_data(0),
                                                                          .
 mil1553_rx_sample_select(0)
) inst_uart_1553_core ( .aclk(sys_clk), .arstn(resetn), .uart_clk(sys_clk),
```

Module instance of the 1553 UART with all cores tied together as a common device.

system_wrapper.v

AUTHORS

JAY CONVERTINO

DATES

2024/11/25

INFORMATION

Brief

System wrapper for pl

License MIT

system_wrapper

```
module system_wrapper #(
parameter
clock_speed
 =
2000000,
parameter
baud_rate
 =
1000000,
parameter
mil1553_sample_rate
```

```
  =
 2000000
) ( input clk, input [1:0] push_buttons, inout [7:0] pmod_ja, input ftdi_tx,
```

System wrapper for pl

Parameters

clock_speed parameter	Requested Master Clock Speed from clk wiz
baud_rate parameter	UART BAUD rate
mil1553_sample_rate parameter	Sample rate for 1553, must be 2 MHz or above, and divide evenly into clock_speed.

Ports

clk	Input clock for all clocks
push_buttons	Buttons used for reset (push button 0).
pmod_ja	1553 PMOD device port (JA)
ftdi_tx	FTDI UART input (TX)
ftdi_rx	FTDI UART output (RX)

INSTANTIANTED MODULES

inst_clk_wiz_1

```
 clk_wiz_1 inst_clk_wiz_1 (
                                                                    .
 clk_out1(sys_clk),
                                                                    .
 reset(push_buttons[0]),
                                                                    .
 clk_in1(clk)
)
```

Module instance of clock wizard to change input clock to requested clock speed.

inst_sys_rstgen

```
 sys_rstgen inst_sys_rstgen (
                                                                    .
 slowest_sync_clk(sys_clk),
                                                                    .
 ext_reset_in(1'b1),
                                                                    .
 aux_reset_in(push_buttons[0]),
                                                                    .
 mb_debug_sys_rst(1'b0),
                                                                    .
 dcm_locked(1'b1),
                                                                    .
 mb_reset(),
                                                                    .
 bus_struct_reset(),
                                                                            .
 peripheral_reset(reset),
                                                                            .
 interconnect_aresetn(),
                                                                            .
 peripheral_aresetn(resetn)
)
```

Module instance of reset gen to create system reset.

inst_uart_1553_core

```
 uart_1553_core #(
                                                                            .
 clock_speed(clock_speed),
                                                                            .
 uart_baud_clock_speed(clock_speed),
                                                                            .
 uart_baud_rate(baud_rate),
                                                                            .
 uart_parity_ena(0),
                                                                            .
 uart_parity_type(0),
                                                                            .
 uart_stop_bits(1),
                                                                            .
 uart_data_bits(8),
                                                                            .
 uart_rx_delay(0),
                                                                            .
 uart_tx_delay(0),
                                                                            .
 mil1553_sample_rate(mil1553_sample_rate),
                                                                            .
 mil1553_rx_bit_slice_offset(0),
                                                                            .
 mil1553_rx_invert_data(0),
                                                                            .
 mil1553_rx_sample_select(0)
) inst_uart_1553_core ( .aclk(sys_clk), .arstn(resetn), .uart_clk(sys_clk),
```

Module instance of the 1553 UART with all cores tied together as a common device.

system_wrapper.v

AUTHORS

JAY CONVERTINO

DATES

2024/11/25

INFORMATION

Brief

System wrapper for pl

License MIT

system_wrapper

```
module system_wrapper #(
parameter
clock_speed
 =
2000000,
parameter
baud_rate
 =
1000000,
parameter
mil1553_sample_rate
```

```
  =
 2000000
) ( input clk, input [1:0] push_buttons, inout [7:0] pmod_ja, input ftdi_tx,
```

System wrapper for pl

Parameters

clock_speed parameter	Requested Master Clock Speed from clk wiz
baud_rate parameter	UART BAUD rate
mil1553_sample_rate parameter	Sample rate for 1553, must be 2 MHz or above, and divide evenly into clock_speed.

Ports

clk	Input clock for all clocks
push_buttons	Buttons used for reset (push button 0).
pmod_ja	1553 PMOD device port (JA)
ftdi_tx	FTDI UART input (TX)
ftdi_rx	FTDI UART output (RX)
ftdi_rts	FTDI Request To Send, input.
ftdi_cts	FTDI Clear to send, output.

INSTANTIANTED MODULES

inst_clk_wiz_1

```
 clk_wiz_1 inst_clk_wiz_1 (
                                                                    .
 clk_out1(sys_clk),
                                                                    .
 reset(push_buttons[0]),
                                                                    .
 clk_in1(clk)
)
```

Module instance of clock wizard to change input clock to requested clock speed.

inst_sys_rstgen

```
 sys_rstgen inst_sys_rstgen (
                                                                    .
 slowest_sync_clk(sys_clk),
                                                                    .
 ext_reset_in(push_buttons[0]),
                                                                    .
 aux_reset_in(1'b1),
                                                                    .
 mb_debug_sys_rst(1'b0),
                                                                    .
 dcm_locked(1'b1),
                                                                         .
 mb_reset(),
                                                                         .
 bus_struct_reset(),
                                                                         .
 peripheral_reset(),
                                                                         .
 interconnect_aresetn(),
                                                                         .
 peripheral_aresetn(resetn)
)
```

Module instance of reset gen to create system reset.

inst_uart_1553_core

```
 uart_1553_core #(
                                                                         .
 clock_speed(clock_speed),
                                                                         .
 uart_baud_clock_speed(clock_speed),
                                                                         .
 uart_baud_rate(baud_rate),
                                                                         .
 uart_parity_ena(0),
                                                                         .
 uart_parity_type(0),
                                                                         .
 uart_stop_bits(1),
                                                                         .
 uart_data_bits(8),
                                                                         .
 uart_rx_delay(0),
                                                                         .
 uart_tx_delay(0),
                                                                         .
 mil1553_sample_rate(mil1553_sample_rate),
                                                                         .
 mil1553_rx_bit_slice_offset(0),
                                                                         .
 mil1553_rx_invert_data(0),
                                                                         .
 mil1553_rx_sample_select(0)
) inst_uart_1553_core ( .aclk(sys_clk), .arstn(resetn), .uart_clk(sys_clk),
```

Module instance of the 1553 UART with all cores tied together as a common device.