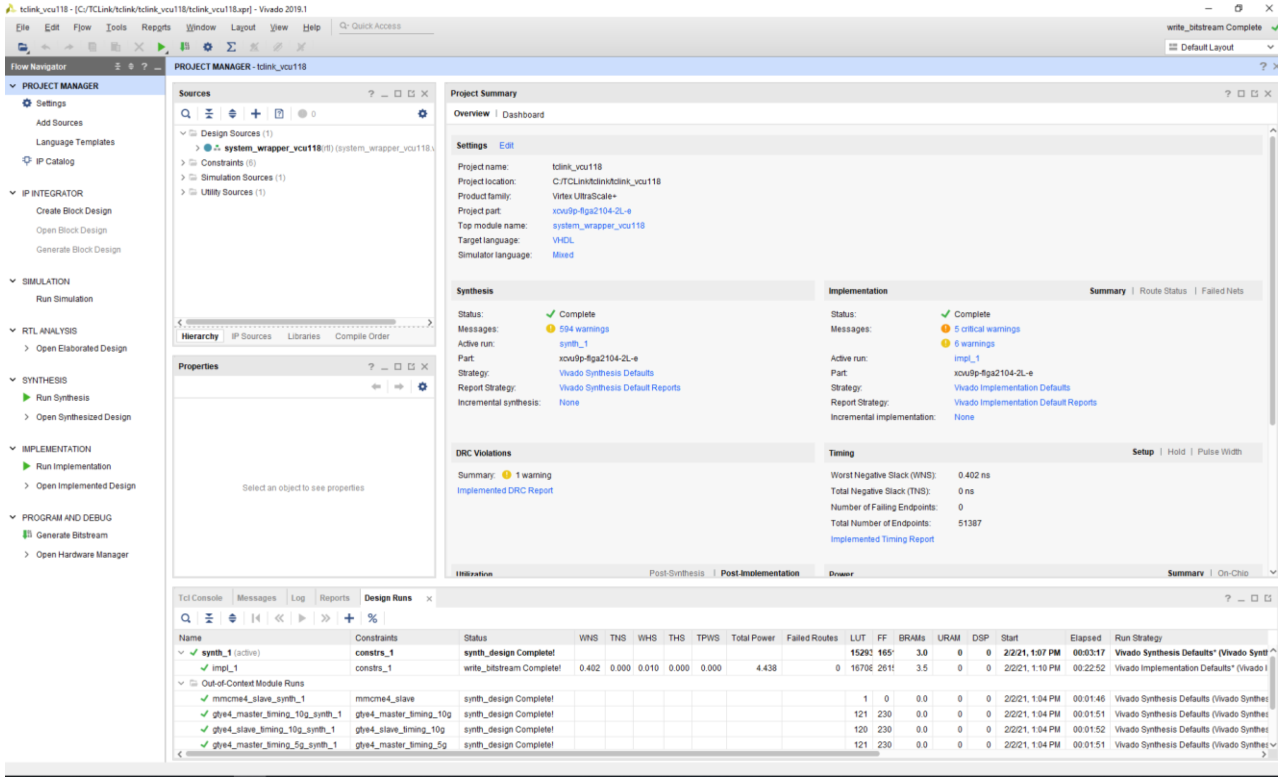
TCLink

VCU118 example design
quick start guide

Eduardo Mendes

Basic configuration

1. clone GIT repository
2. run **tclink_vcu118.tcl** script in **Vivado 2019.1** in batch mode (an example in **tclink_vcu118.bat** for Windows)
3. once the project is created and the physical implementation is finished, you can open the Vivado project



tclink_vcu118 - [C:/TCLink/tclink/tclink_vcu118/tclink_vcu118.xpr] - Vivado 2019.1

Design Sources (1)
- system_wrapper_vcu118(rtl) (system_wrapper_vcu118.v
Constraints (6)
Simulation Sources (1)
Utility Sources (1)

Settings Edit

Project name: tclink_vcu118
Project location: C:/TCLink/tclink/tclink_vcu118
Product family: Virtex UltraScale+
Project part: xcvu9p-flga2104-2L-e
Top module name: system_wrapper_vcu118
Target language: VHDL
Simulator language: Mixed

Synthesis

Status: ✓ Complete
Messages: 594 warnings
Active run: synth_1
Part: xcvu9p-flga2104-2L-e
Strategy: Vivado Synthesis Defaults
Report Strategy: Vivado Synthesis Default Reports
Incremental synthesis: None

Implementation

Status: ✓ Complete
Messages: 5 critical warnings
6 warnings
Active run: impl_1
Part: xcvu9p-flga2104-2L-e
Strategy: Vivado Implementation Defaults
Report Strategy: Vivado Implementation Default Reports
Incremental implementation: None

DRC Violations

Summary: 1 warning
Implemented DRC Report

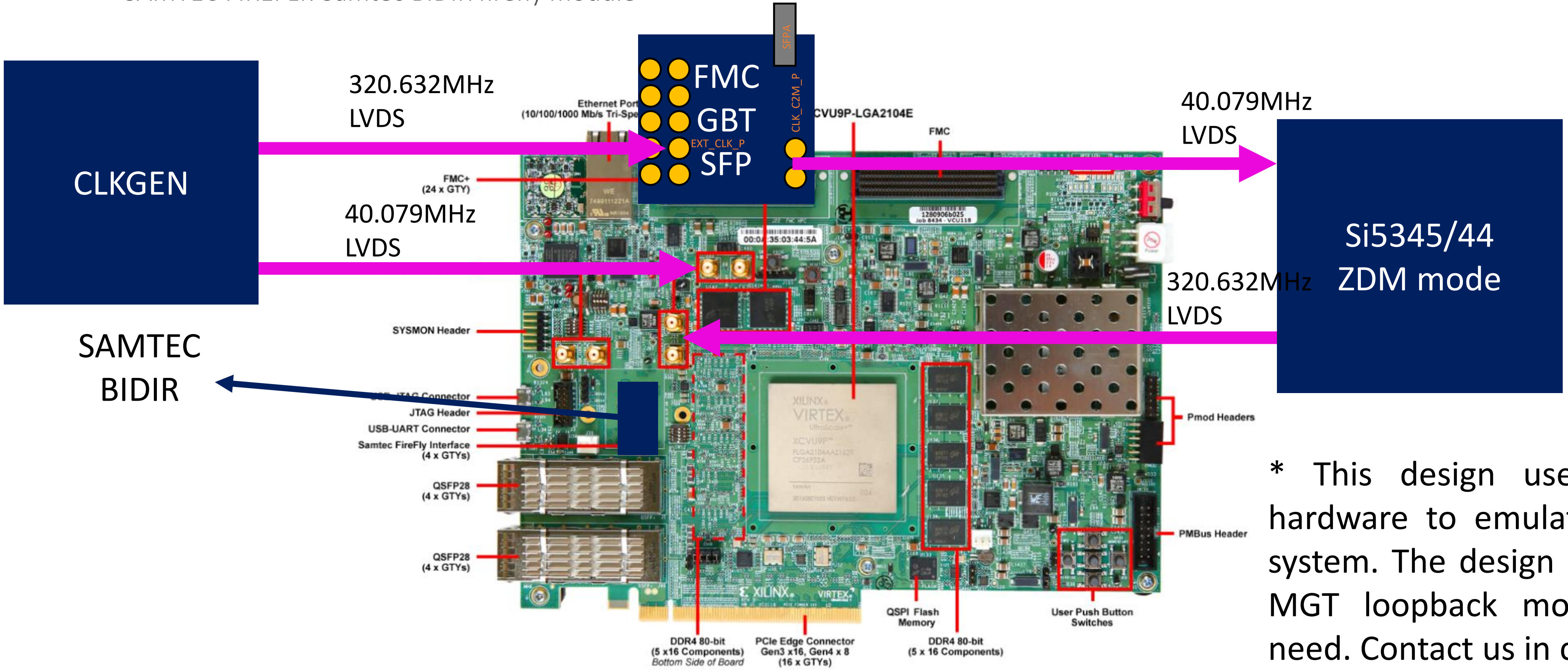
Timing

Worst Negative Slack (WNS): 0.402 ns
Total Negative Slack (TNS): 0 ns
Number of Failing Endpoints: 0
Total Number of Endpoints: 51387
Implemented Timing Report

Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP	Start	Elapsed	Run Strategy
✓ synth_1 (active)	constrs_1	synth_design Complete!								15293	165	3.0	0	0	2/2/21, 1:07 PM	00:03:17	Vivado Synthesis Defaults* (Vivado Synth
✓ impl_1	constrs_1	write_bitstream Complete!	0.402	0.000	0.010	0.000	0.000	4.438	0	16708	261	3.5	0	0	2/2/21, 1:10 PM	00:22:52	Vivado Implementation Defaults* (Vivado I
Out-of-Context Module Runs																	
✓ mmcme4_slave_synth_1	mmcme4_slave	synth_design Complete!								1	0	0.0	0	0	2/2/21, 1:04 PM	00:01:46	Vivado Synthesis Defaults (Vivado Synthes
✓ gtye4_master_timing_10g_synth_1	gtye4_master_timing_10g	synth_design Complete!								121	230	0.0	0	0	2/2/21, 1:04 PM	00:01:51	Vivado Synthesis Defaults (Vivado Synthes
✓ gtye4_slave_timing_10g_synth_1	gtye4_slave_timing_10g	synth_design Complete!								120	230	0.0	0	0	2/2/21, 1:04 PM	00:01:52	Vivado Synthesis Defaults (Vivado Synthes
✓ gtye4_master_timing_5g_synth_1	gtye4_master_timing_5g	synth_design Complete!								121	230	0.0	0	0	2/2/21, 1:04 PM	00:01:51	Vivado Synthesis Defaults (Vivado Synthes

Basic configuration

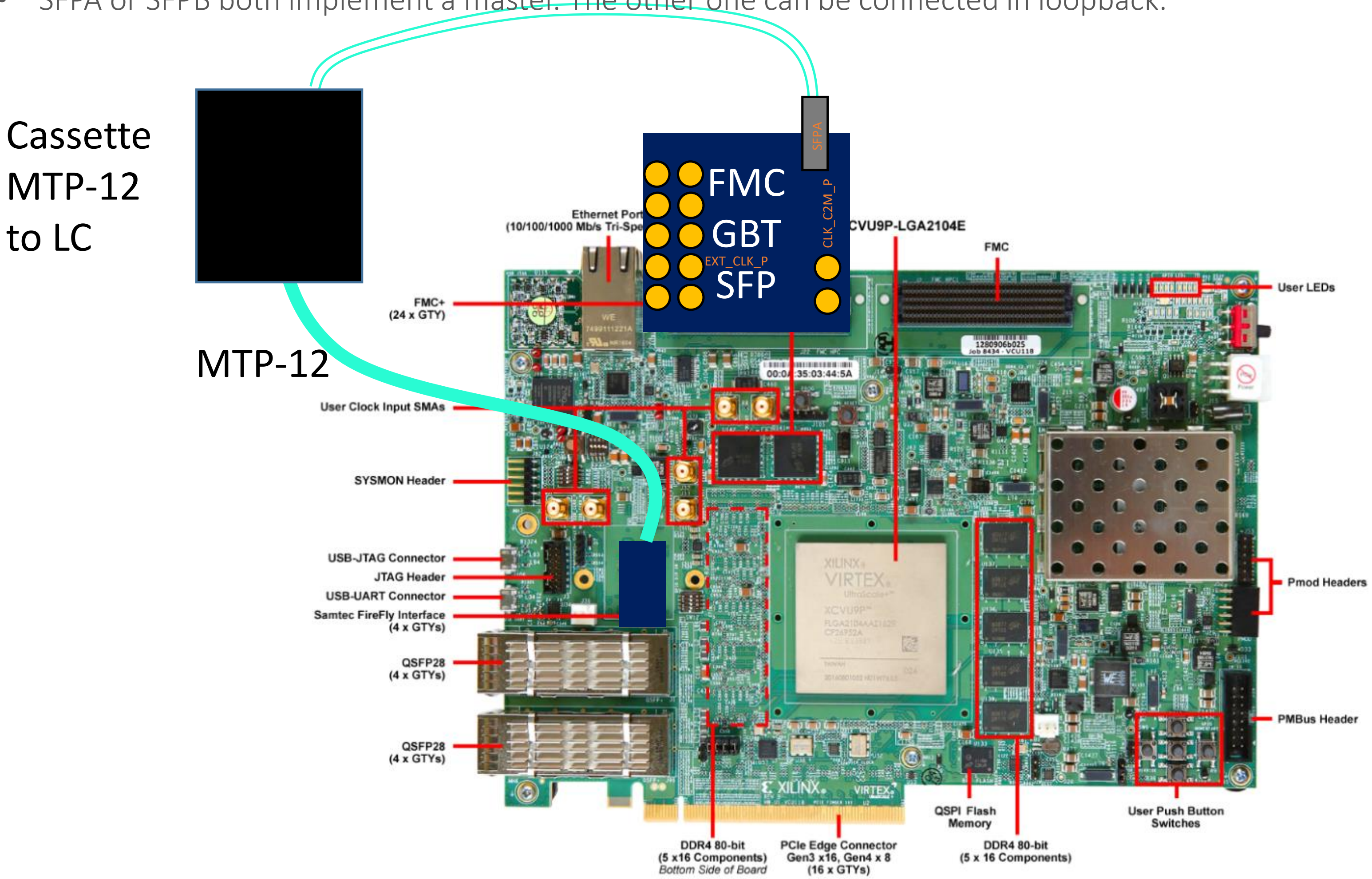
4. Hardware (clocks, modules)*
   - CLKGEN: Free-running reference clock providing 320MHz and 40MHz for masters
   - FMC GBT SFP: custom design from Stephane Detraz (CERN EP-ESE), the design uses GBTCLK0 for the master, an SFP in DP0 for the master and FMC_LA00_CC_P for the slave recovered clock. Obs: connect jumper to W1.
   - Si5345/44: Silicon Labs PLL featuring fixed-latency in ZDM mode
   - SAMTEC FIREFLY: Samtec BIDIR firefly module



* This design uses very specific hardware to emulate a real TCLink system. The design can also work in MGT loopback modes in case of need. Contact us in case of need!

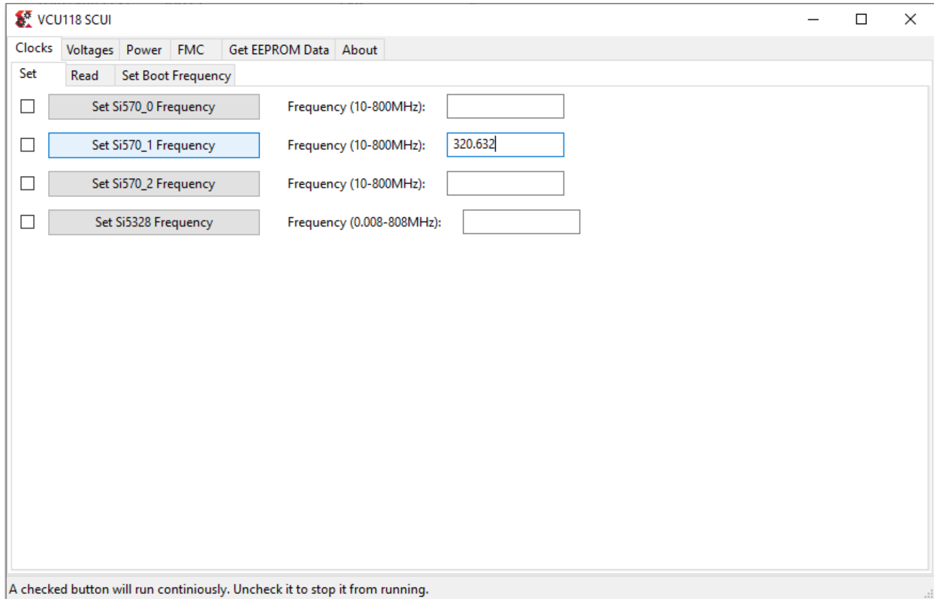
Basic configuration

5. Hardware (fiber)
   - SFPA or SFPB TX to Firefly channel 0 Rx
   - SFPA or SFPB RX to Firefly channel 0 Tx
   - SFPA or SFPB both implement a master. The other one can be connected in loopback.



Basic configuration

6. program VCU118 free-running MGT clock using **system controller** user interface (documentation VCU118)
   - This clock is used as a reference for the slave MGT Rx



□ Set Si570_0 Frequency — Frequency (10-800MHz):

□ Set Si570_1 Frequency — Frequency (10-800MHz): 320.632

□ Set Si570_2 Frequency — Frequency (10-800MHz):

□ Set Si5328 Frequency — Frequency (0.008-808MHz):

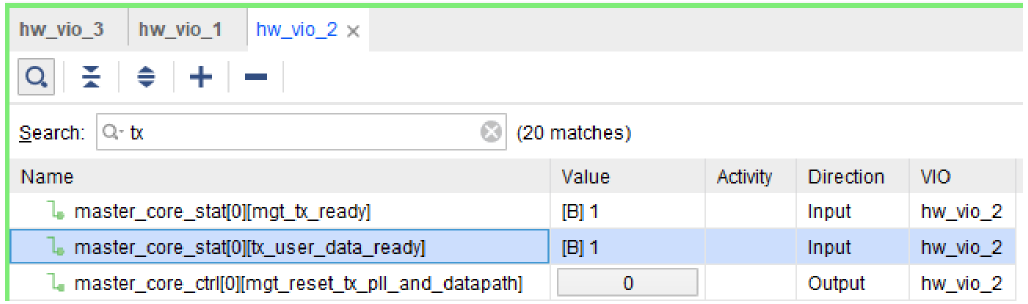
A checked button will run continiously. Uncheck it to stop it from running.

7. program FPGA ☺

Basic configuration

8. The design can be controlled using the VIO graphical interface in Vivado

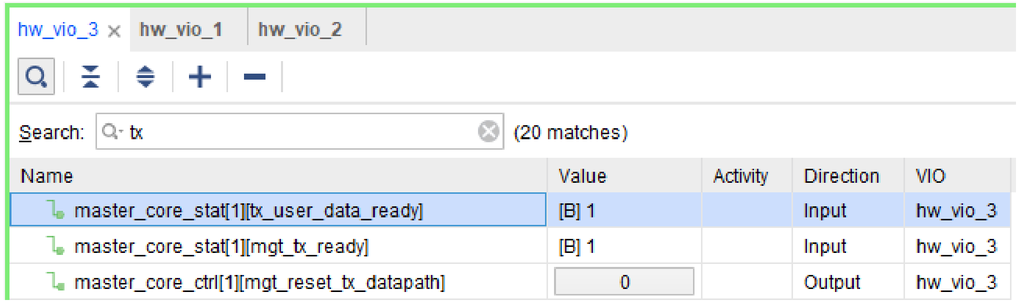
9. Reset Master0 Tx (Reset Tx PLL and datapath)
   - Check tx_ready



Search: tx (20 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[0][mgt_tx_ready]	[B] 1		Input	hw_vio_2
master_core_stat[0][tx_user_data_ready]	[B] 1		Input	hw_vio_2
master_core_ctrl[0][mgt_reset_tx_pll_and_datapath]	0		Output	hw_vio_2

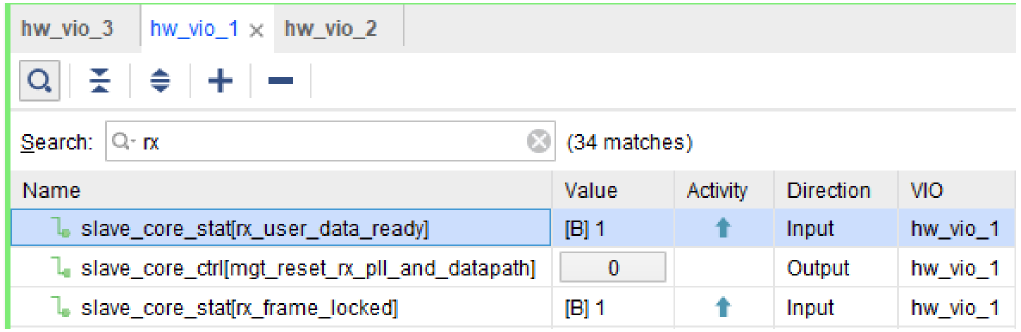
10. Reset Master1 Tx (Reset Tx datapath)
    - OBS: Master0 and Master1 share a QPLL and therefore only the reset of Tx PLL and datapath of Master0 is connected to PLL
    - Check tx_ready



Search: tx (20 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[1][tx_user_data_ready]	[B] 1		Input	hw_vio_3
master_core_stat[1][mgt_tx_ready]	[B] 1		Input	hw_vio_3
master_core_ctrl[1][mgt_reset_tx_datapath]	0		Output	hw_vio_3

11. Reset Slave Rx (Reset Rx PLL and datapath)
    - Check rx frame is locked
    - A common issue if the frame is not locked is some polarity inversion. Try to invert the Rx polarity in case the link does not lock.

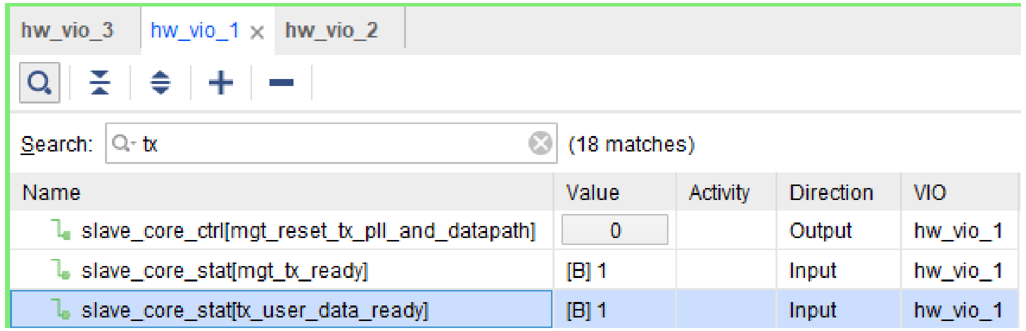


Search: rx (34 matches)

Name	Value	Activity	Direction	VIO
slave_core_stat[rx_user_data_ready]	[B] 1	↑	Input	hw_vio_1
slave_core_ctrl[mgt_reset_rx_pll_and_datapath]	0		Output	hw_vio_1
slave_core_stat[rx_frame_locked]	[B] 1	↑	Input	hw_vio_1

Basic configuration

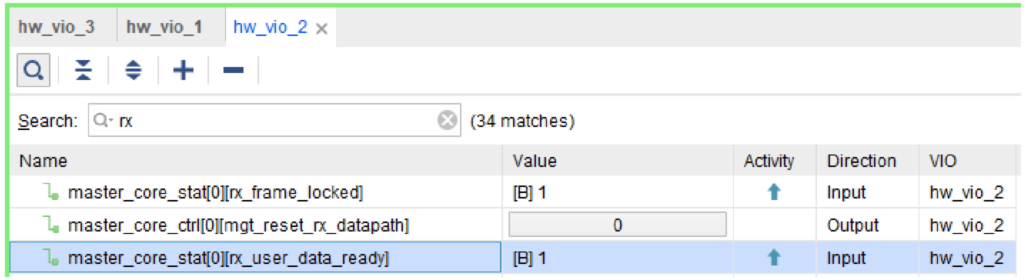
12. Reset Slave Tx (Reset Tx PLL and datapath)
    - Check tx_ready



Search: tx (18 matches)

Name	Value	Activity	Direction	VIO
slave_core_ctrl[mgt_reset_tx_pll_and_datapath]	0		Output	hw_vio_1
slave_core_stat[mgt_tx_ready]	[B] 1		Input	hw_vio_1
slave_core_stat[tx_user_data_ready]	[B] 1		Input	hw_vio_1

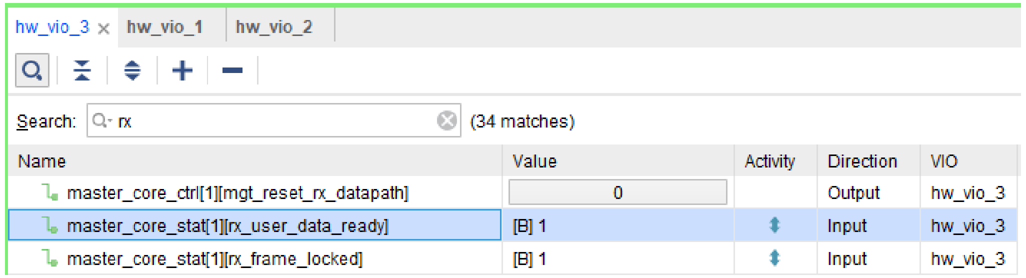
13. Reset Master0 Rx (Reset Rx datapath)
    - Check rx frame is locked
    - OBS: since master has Tx and Rx PLL shared, do not use Reset Rx PLL and datapath



Search: rx (34 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[0][rx_frame_locked]	[B] 1	↑	Input	hw_vio_2
master_core_ctrl[0][mgt_reset_rx_datapath]	0		Output	hw_vio_2
master_core_stat[0][rx_user_data_ready]	[B] 1	↑	Input	hw_vio_2

14. Reset Master1 Rx (Reset Rx datapath)
    - Check rx frame is locked
    - OBS: since master has Tx and Rx PLL shared, do not use Reset Rx PLL and datapath



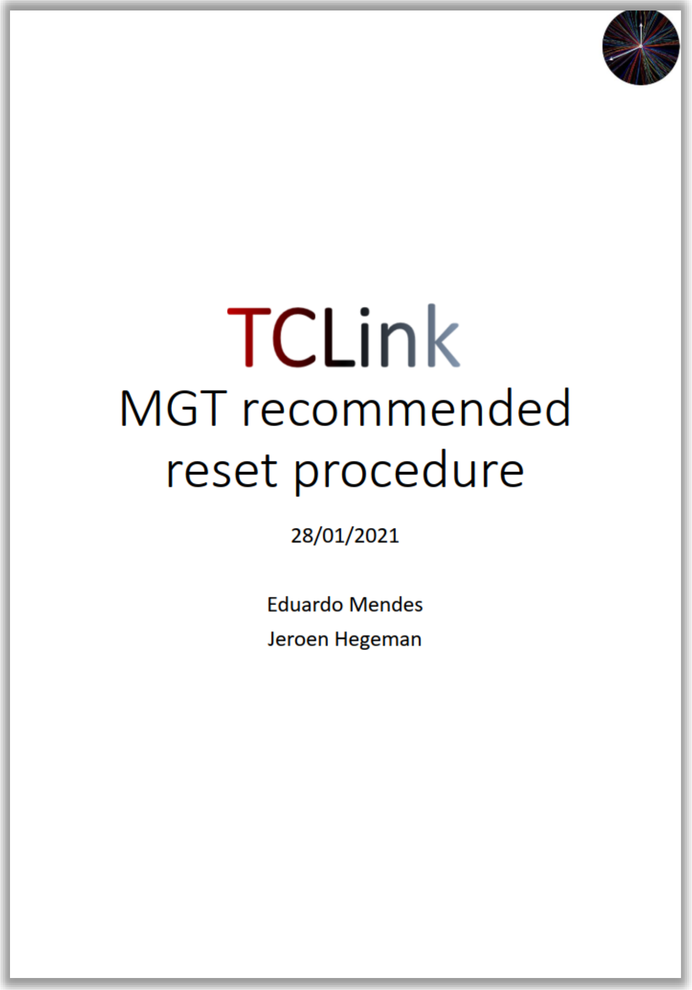
Search: rx (34 matches)

Name	Value	Activity	Direction	VIO
master_core_ctrl[1][mgt_reset_rx_datapath]	0		Output	hw_vio_3
master_core_stat[1][rx_user_data_ready]	[B] 1	↕	Input	hw_vio_3
master_core_stat[1][rx_frame_locked]	[B] 1	↕	Input	hw_vio_3

Basic configuration

-> An example of FSM implementation for the reset scheme is integrated in the core

- Read the design choice documents for the MGT recommended reset procedure to know more



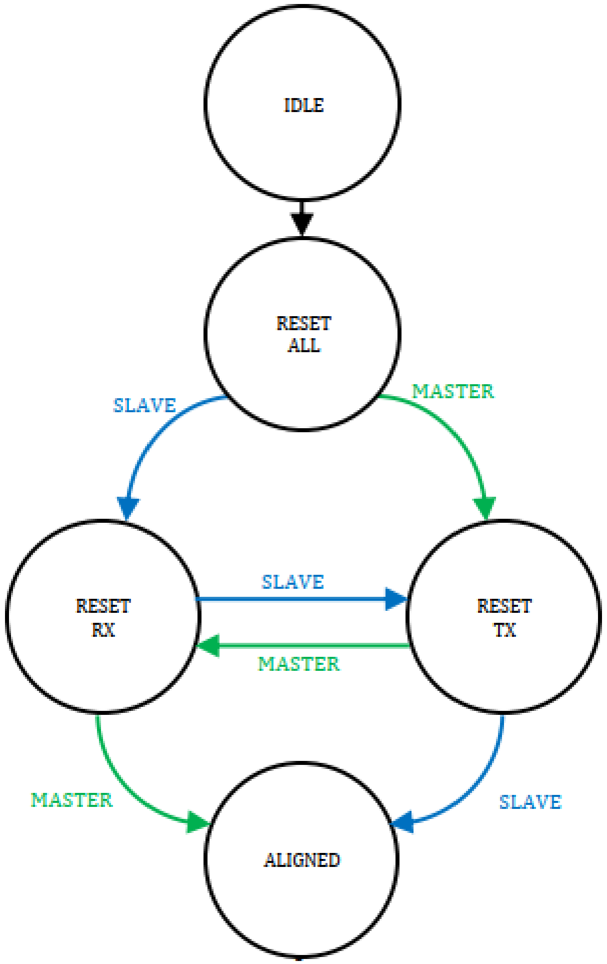
TCLink

MGT recommended reset procedure

28/01/2021

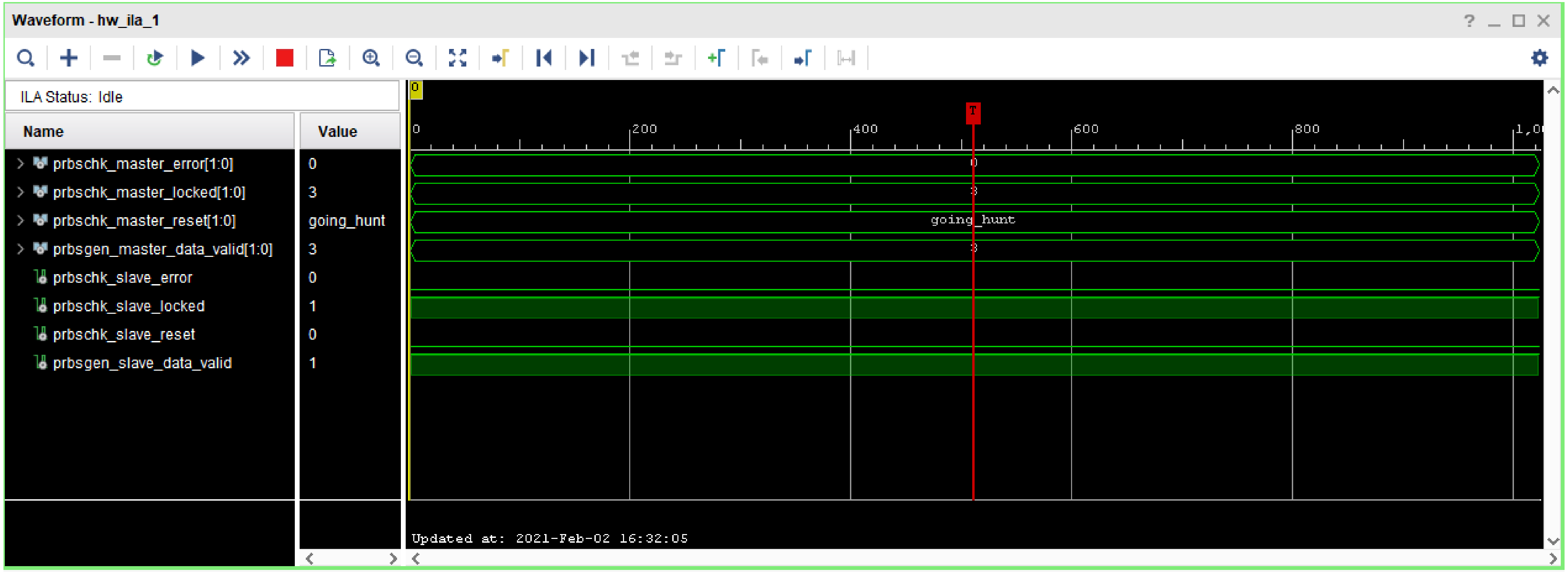
Eduardo Mendes

Jeroen Hegeman



Basic configuration

15. Check that all links are locked using the ILA



Waveform - hw_ila_1

ILA Status: Idle

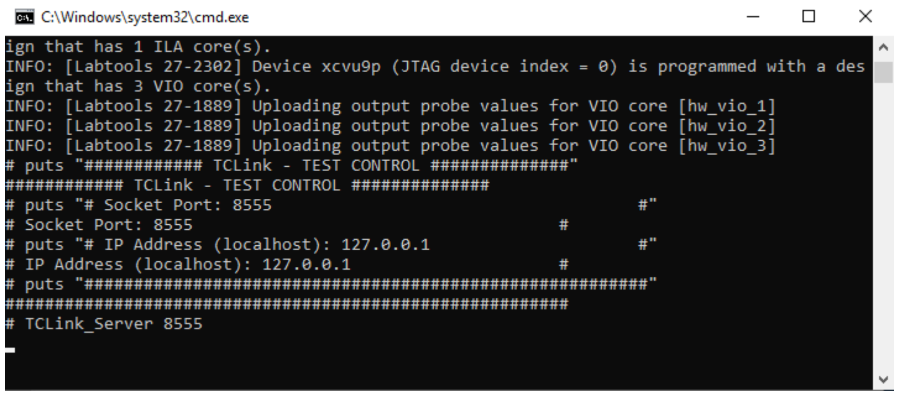
Name	Value
prbschk_master_error[1:0]	0
prbschk_master_locked[1:0]	3
prbschk_master_reset[1:0]	going_hunt
prbsgen_master_data_valid[1:0]	3
prbschk_slave_error	0
prbschk_slave_locked	1
prbschk_slave_reset	0
prbsgen_slave_data_valid	1

```
Updated at: 2021-Feb-02 16:32:05
```

Basic configuration

16. The design can be also controlled in **Python3** (for repetitive tests)
    a. Execute **software/jtag_server/jtag_server_vcu118.tcl** in Vivado batch mode (an example for Windows in software/jtag_server/execute_jtag_server_vcu118.bat)

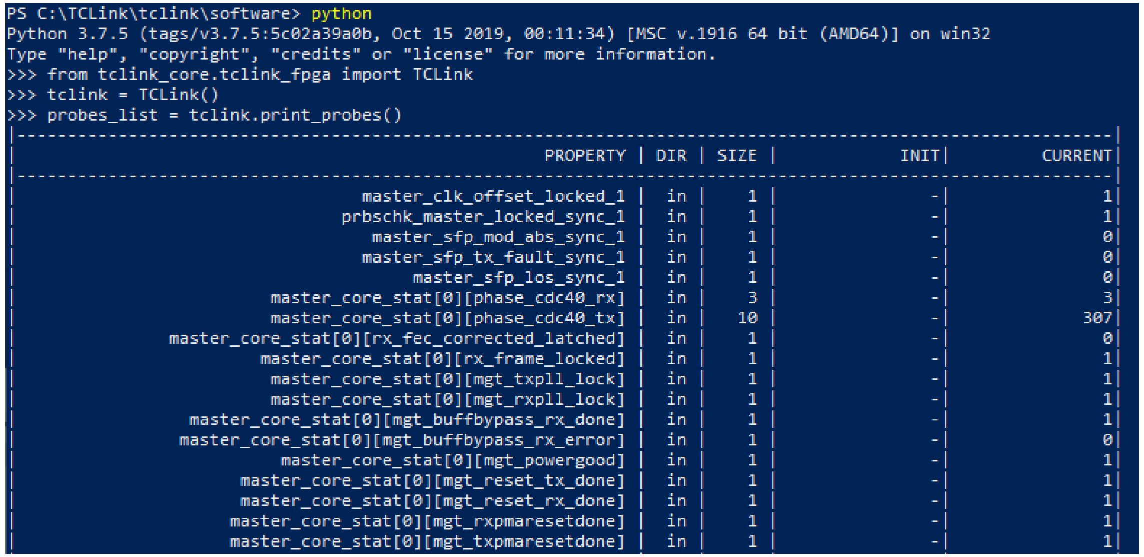
Jtag server example



C:\Windows\system32\cmd.exe

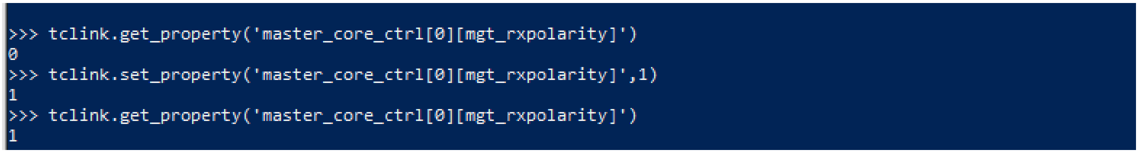
```
ign that has 1 ILA core(s).
INFO: [Labtools 27-2302] Device xcvu9p (JTAG device index = 0) is programmed with a des
ign that has 3 VIO core(s).
INFO: [Labtools 27-1889] Uploading output probe values for VIO core [hw_vio_1]
INFO: [Labtools 27-1889] Uploading output probe values for VIO core [hw_vio_2]
INFO: [Labtools 27-1889] Uploading output probe values for VIO core [hw_vio_3]
# puts "############ TCLink - TEST CONTROL ##############"
############ TCLink - TEST CONTROL ##############
# puts "# Socket Port: 8555                                #"
# Socket Port: 8555                                #
# puts "# IP Address (localhost): 127.0.0.1                #"
# IP Address (localhost): 127.0.0.1                #
# puts "##################################################"
##################################################
# TCLink_Server 8555
```

Python example



```
PS C:\TCLink\tclink\software> python
Python 3.7.5 (tags/v3.7.5:5c02a39a0b, Oct 15 2019, 00:11:34) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from tclink_core.tclink_fpga import TCLink
>>> tclink = TCLink()
>>> probes_list = tclink.print_probes()
|-------------------------------------------------------------------------------------------------------------|
|                                                    PROPERTY | DIR | SIZE |            INIT|         CURRENT|
|-------------------------------------------------------------------------------------------------------------|
|                                  master_clk_offset_locked_1 |  in |    1 |               -|               1|
|                                prbschk_master_locked_sync_1 |  in |    1 |               -|               1|
|                                   master_sfp_mod_abs_sync_1 |  in |    1 |               -|               0|
|                                  master_sfp_tx_fault_sync_1 |  in |    1 |               -|               0|
|                                       master_sfp_los_sync_1 |  in |    1 |               -|               0|
|                         master_core_stat[0][phase_cdc40_rx] |  in |    3 |               -|               3|
|                         master_core_stat[0][phase_cdc40_tx] |  in |   10 |               -|             307|
|               master_core_stat[0][rx_fec_corrected_latched] |  in |    1 |               -|               0|
|                        master_core_stat[0][rx_frame_locked] |  in |    1 |               -|               1|
|                         master_core_stat[0][mgt_txpll_lock] |  in |    1 |               -|               1|
|                         master_core_stat[0][mgt_rxpll_lock] |  in |    1 |               -|               1|
|                 master_core_stat[0][mgt_buffbypass_rx_done] |  in |    1 |               -|               1|
|                master_core_stat[0][mgt_buffbypass_rx_error] |  in |    1 |               -|               0|
|                          master_core_stat[0][mgt_powergood] |  in |    1 |               -|               1|
|                      master_core_stat[0][mgt_reset_tx_done] |  in |    1 |               -|               1|
|                      master_core_stat[0][mgt_reset_rx_done] |  in |    1 |               -|               1|
|                     master_core_stat[0][mgt_rxpmaresetdone] |  in |    1 |               -|               1|
|                     master_core_stat[0][mgt_txpmaresetdone] |  in |    1 |               -|               1|
```

•••



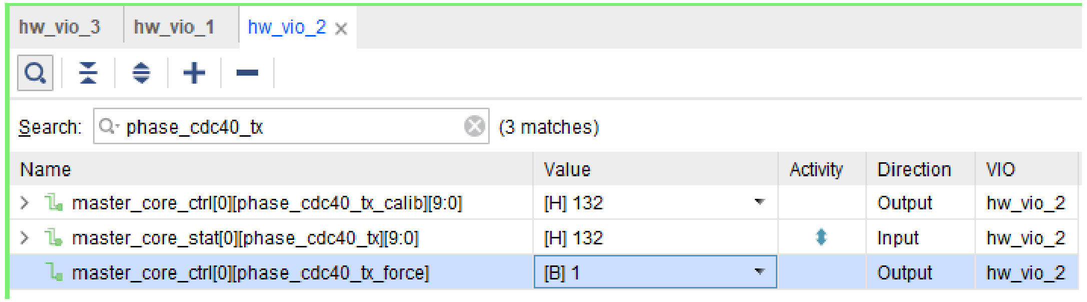
```
>>> tclink.get_property('master_core_ctrl[0][mgt_rxpolarity]')
0
>>> tclink.set_property('master_core_ctrl[0][mgt_rxpolarity]',1)
1
>>> tclink.get_property('master_core_ctrl[0][mgt_rxpolarity]')
1
```

    b. Open socket in python to the local JTAG server
    c. Check VIO probes available
    d. Set/get a probe value
    e. Additional features:
        a. Preset design (function **preset()**)
        b. Reprogram FPGA (function **fpga_program()**)
        c. Sysmon monitoring (function **save_sysmon_state(name_file)**)

Fixed latency CDC 40-320 (for master and slave)

- If required, the CDC for Tx (40MHz to 320MHz) and Rx (320MHz to 40MHz) allow a fixed-latency operation after resets
  - In order to achieve fixed-latency, the phase has to be forced after the first reset
  - Example for master0 Tx

```
tx40_phase = dut.get_property(‘master_core_stat[0][phase_cdc40_tx]’)
dut.set_property(‘master_core_ctrl[0][phase_cdc40_tx_calib]’, tx40_phase)
dut.set_property(‘master_core_ctrl[0][phase_cdc40_tx_force]’, 1)
```



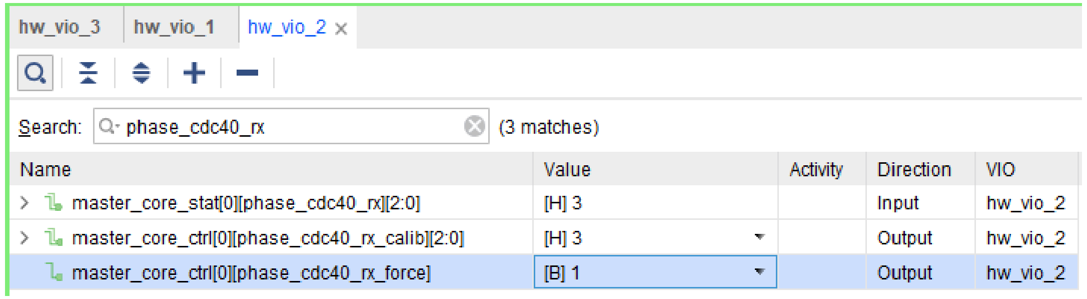
hw_vio_3 | hw_vio_1 | hw_vio_2 ×

Search: phase_cdc40_tx (3 matches)

Name	Value	Activity	Direction	VIO
master_core_ctrl[0][phase_cdc40_tx_calib][9:0]	[H] 132		Output	hw_vio_2
master_core_stat[0][phase_cdc40_tx][9:0]	[H] 132	⬍	Input	hw_vio_2
master_core_ctrl[0][phase_cdc40_tx_force]	[B] 1		Output	hw_vio_2

  - Example for master0 Rx

```
rx40_phase = dut.get_property(‘master_core_stat[0][phase_cdc40_rx]’)
dut.set_property(‘master_core_ctrl[0][phase_cdc40_rx_calib]’, rx40_phase)
dut.set_property(‘master_core_ctrl[0][phase_cdc40_rx_force]’, 1)
```



hw_vio_3 | hw_vio_1 | hw_vio_2 ×

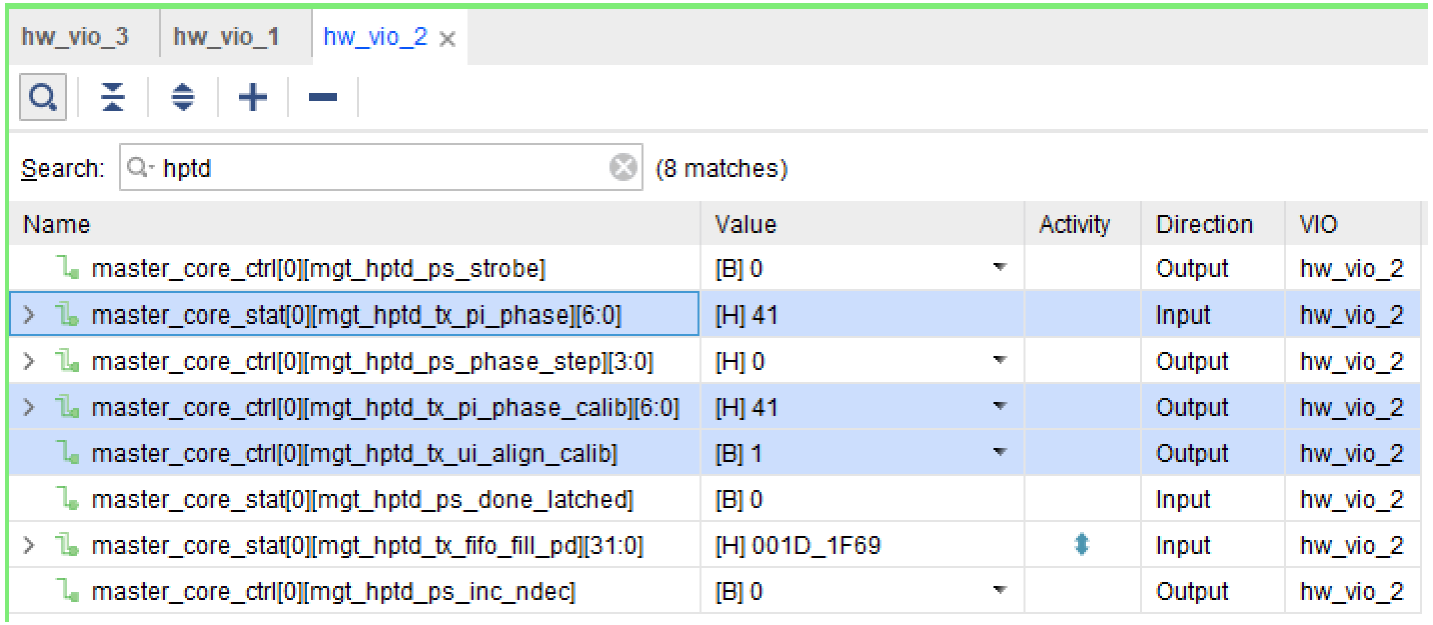
Search: phase_cdc40_rx (3 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[0][phase_cdc40_rx][2:0]	[H] 3		Input	hw_vio_2
master_core_ctrl[0][phase_cdc40_rx_calib][2:0]	[H] 3		Output	hw_vio_2
master_core_ctrl[0][phase_cdc40_rx_force]	[B] 1		Output	hw_vio_2

MGT Tx fixed-phase (for master and slave)

- HPTD IP core (more information on HPTD IP documentation)
  - In order to freeze a given Tx PI phase value (after first reset) – example for master0

```
tx_phase = dut.get_property('master_core_stat[0][mgt_hptd_tx_pi_phase]')
dut.set_property('master_core_ctrl[0][mgt_hptd_tx_pi_phase_calib]', tx_phase)
dut.set_property('master_core_ctrl[0][ mgt_hptd_tx_ui_align_calib]', 1)
```



Search: hptd (8 matches)

Name	Value	Activity	Direction	VIO
master_core_ctrl[0][mgt_hptd_ps_strobe]	[B] 0		Output	hw_vio_2
master_core_stat[0][mgt_hptd_tx_pi_phase][6:0]	[H] 41		Input	hw_vio_2
master_core_ctrl[0][mgt_hptd_ps_phase_step][3:0]	[H] 0		Output	hw_vio_2
master_core_ctrl[0][mgt_hptd_tx_pi_phase_calib][6:0]	[H] 41		Output	hw_vio_2
master_core_ctrl[0][mgt_hptd_tx_ui_align_calib]	[B] 1		Output	hw_vio_2
master_core_stat[0][mgt_hptd_ps_done_latched]	[B] 0		Input	hw_vio_2
master_core_stat[0][mgt_hptd_tx_fifo_fill_pd][31:0]	[H] 001D_1F69	⇕	Input	hw_vio_2
master_core_ctrl[0][mgt_hptd_ps_inc_ndec]	[B] 0		Output	hw_vio_2

  - The user can also shift the phase with o(ps) resolution using the mgt_hptd_ps_inc_ndec (increment or decrement), mgt_hptd_ps_phase_step and mgt_hptd_ps_strobe signals

MGT Rx fixed-phase

- Slave fixed-latency is only supported in buffer-bypass and rxslide in PMA mode (this is not recommended by Xilinx)
  - This is related to lpGBT-FPGA frame aligner design
  - Another potential mode is the roulette approach (reset until locked)

- Master fixed-latency is not necessary (TCLink takes into account, mathematically, number of Rxslide pulses in the master side)

- Rx equalizer adaptation does not seem to need to be frozen in LPM mode (Rx equalizer impact on fixed-phase report)

TCLink basic configuration

- Before closing the loop, the offset phase has to be measured (procedure to be done once after first reset):

```
dut.set_property('master_core_ctrl[0][tclink_offset_error]', 0)
Wait for at least one second…
offset = dut.get_property('master_core_stat[0][tclink_error_controller]')
dut.set_property('master_core_ctrl[0][tclink_offset_error]', offset)
```

- Close-loop:

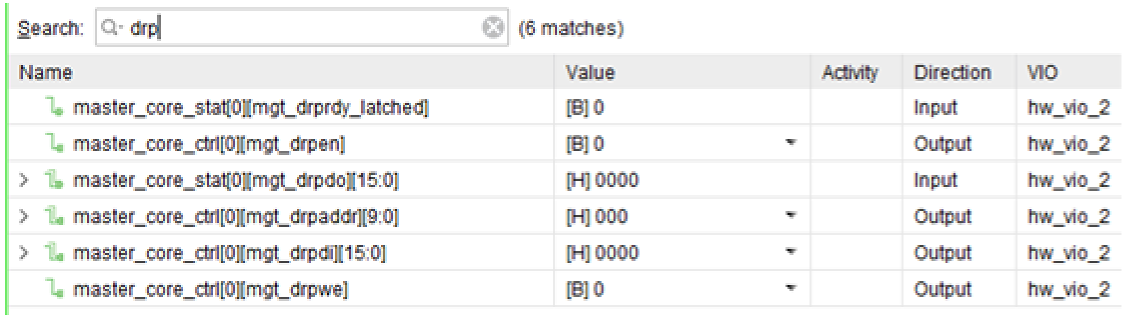
```
dut.set_property('master_core_ctrl[0][tclink_close_loop]', 1)
```

- An example of how to configure TCLink in the example design and run a transfer function in the FPGA (using the TCLink tester) is given in the script **software/fpga_transfer_function_vcu118.py**

- **Read official tclink reference note on how to convert phase measurement values to ps**

Additional MGT features

- Dynamic reconfiguration port

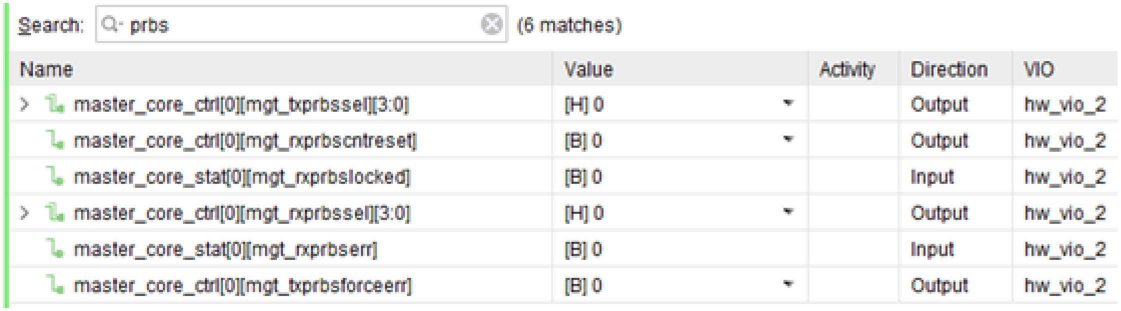


Search: drp (6 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[0][mgt_drprdy_latched]	[B] 0		Input	hw_vio_2
master_core_ctrl[0][mgt_drpen]	[B] 0		Output	hw_vio_2
master_core_stat[0][mgt_drpdo][15:0]	[H] 0000		Input	hw_vio_2
master_core_ctrl[0][mgt_drpaddr][9:0]	[H] 000		Output	hw_vio_2
master_core_ctrl[0][mgt_drpdi][15:0]	[H] 0000		Output	hw_vio_2
master_core_ctrl[0][mgt_drpwe]	[B] 0		Output	hw_vio_2

- Internal MGT PRBS
  - lpGBT-FPGA Rx is kept reset when internal PRBS is selected

  0000 = Normal operation
  0001 = PRBS-7
  0010 = PRBS-9
  0011 = PRBS-15
  0100 = PRBS-23
  0101 = PRBS-31
  1001 = Square wave with 2 UI (alternating 0s/1s)
  1010 = Square wave with 32 UI

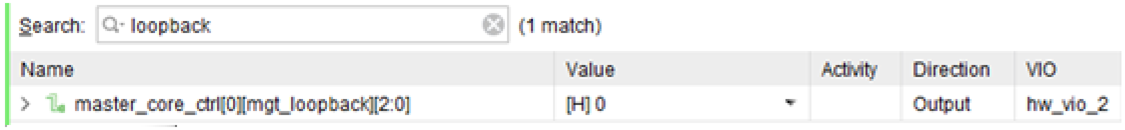


Search: prbs (6 matches)

Name	Value	Activity	Direction	VIO
master_core_ctrl[0][mgt_txprbssel][3:0]	[H] 0		Output	hw_vio_2
master_core_ctrl[0][mgt_rxprbscntreset]	[B] 0		Output	hw_vio_2
master_core_stat[0][mgt_rxprbslocked]	[B] 0		Input	hw_vio_2
master_core_ctrl[0][mgt_rxprbssel][3:0]	[H] 0		Output	hw_vio_2
master_core_stat[0][mgt_rxprbserr]	[B] 0		Input	hw_vio_2
master_core_ctrl[0][mgt_txprbsforceerr]	[B] 0		Output	hw_vio_2

- Transceiver Loopback

  000 = Normal operation
  001 = Near-End PCS Loopback
  010 = Near-End PMA Loopback
  100 = Far-End PMA Loopback (do not use - problem when using Tx PI)
  110 = Far-End PCS Loopback



Search: loopback (1 match)

Name	Value	Activity	Direction	VIO
master_core_ctrl[0][mgt_loopback][2:0]	[H] 0		Output	hw_vio_2