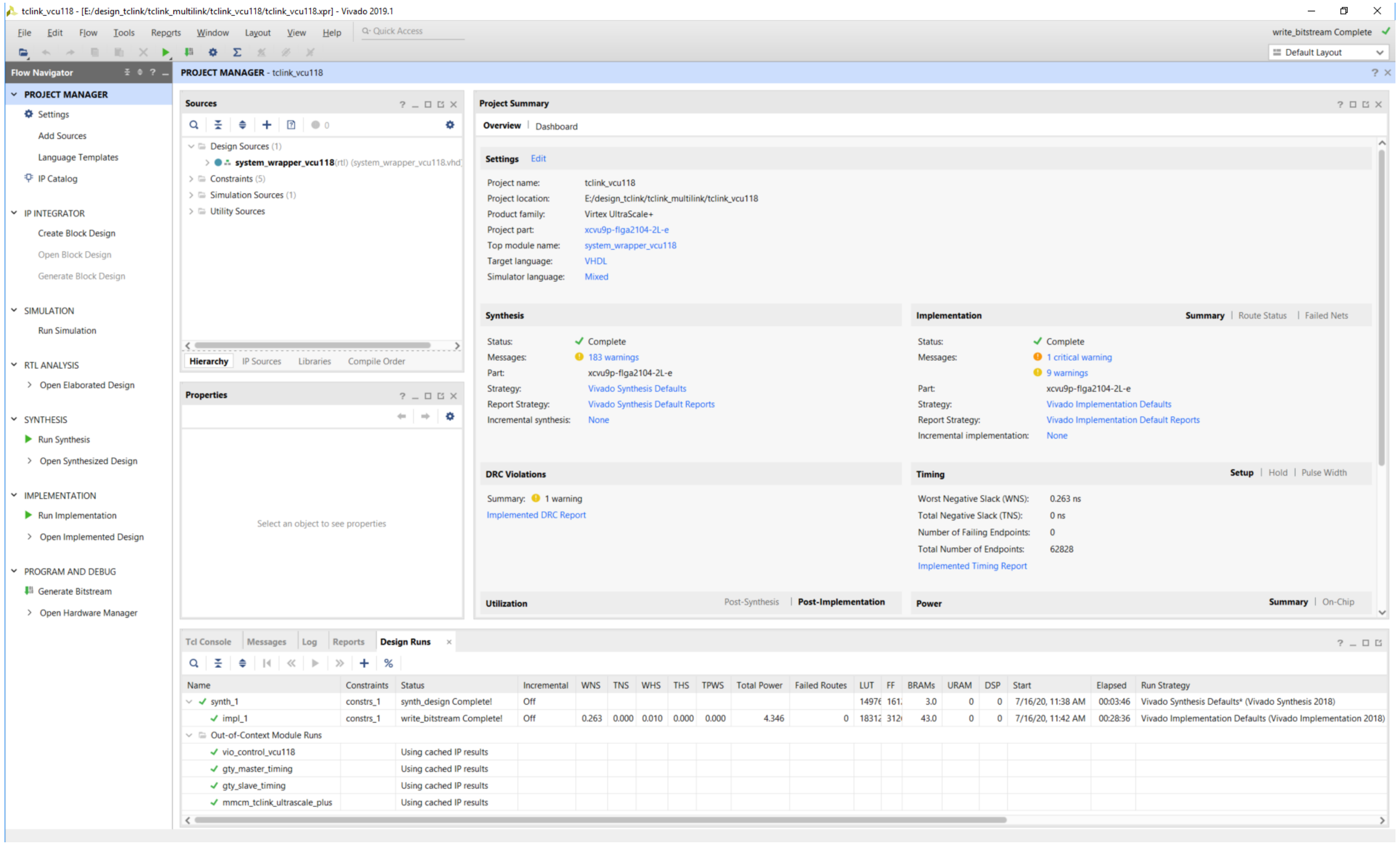
TCLink

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



Settings Edit

Project name: tclink_vcu118
Project location: E:/design_tclink/tclink_multilink/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: 183 warnings
Part: xcvu9p-flga2104-2L-e
Strategy: Vivado Synthesis Defaults
Report Strategy: Vivado Synthesis Default Reports
Incremental synthesis: None

Implementation — Summary | Route Status | Failed Nets

Status: ✓ Complete
Messages: 1 critical warning
9 warnings
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 — Setup | Hold | Pulse Width

Worst Negative Slack (WNS): 0.263 ns
Total Negative Slack (TNS): 0 ns
Number of Failing Endpoints: 0
Total Number of Endpoints: 62828
Implemented Timing Report

Utilization — Post-Synthesis | Post-Implementation

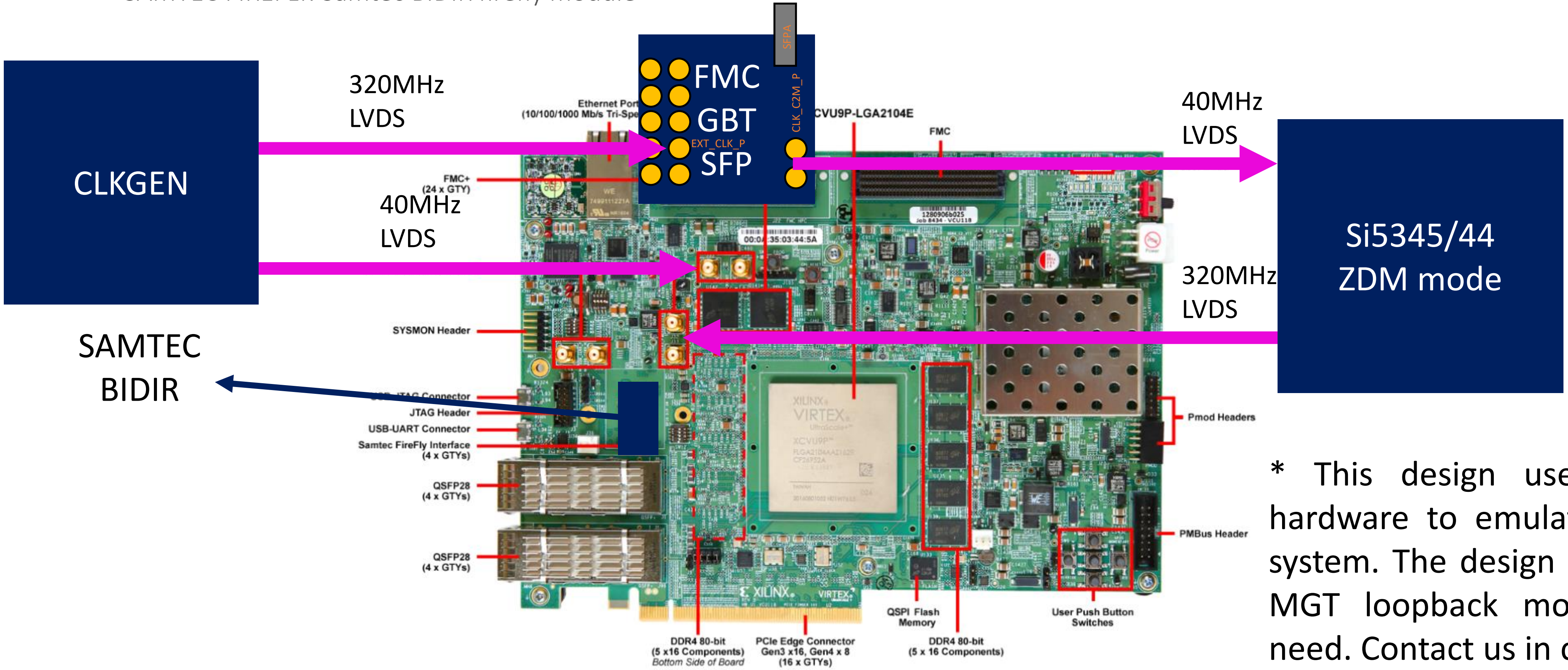
Power — Summary | On-Chip

Tcl Console | Messages | Log | Reports | **Design Runs**

Name	Constraints	Status	Incremental	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP	Start	Elapsed	Run Strategy
✓ synth_1	constrs_1	synth_design Complete!	Off								1497	161	3.0	0	0	7/16/20, 11:38 AM	00:03:46	Vivado Synthesis Defaults* (Vivado Synthesis 2018)
✓ impl_1	constrs_1	write_bitstream Complete!	Off	0.263	0.000	0.010	0.000	0.000	4.346	0	1831	312	43.0	0	0	7/16/20, 11:42 AM	00:28:36	Vivado Implementation Defaults (Vivado Implementation 2018)
Out-of-Context Module Runs																		
✓ vio_control_vcu118		Using cached IP results																
✓ gty_master_timing		Using cached IP results																
✓ gty_slave_timing		Using cached IP results																
✓ mmcm_tclink_ultrascale_plus		Using cached IP results																

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

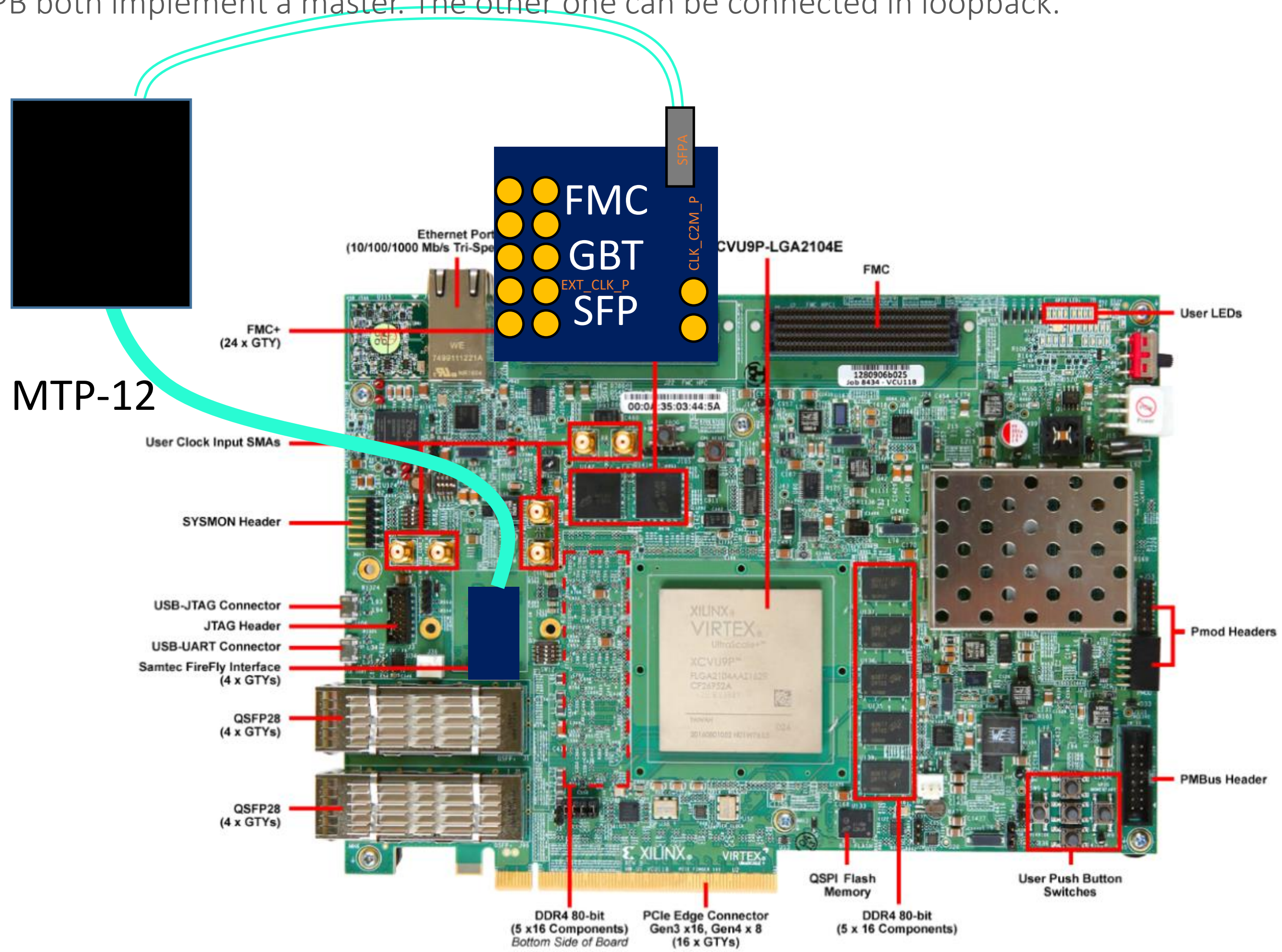


* This design uses very specific hardware to emulate a real TCLink system. The design can also work in MGT loopback modes in case of need. Contact us in case of need!

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.

Cassette
MTP-12
to LC



MTP-12

FMC
GBT
SFP

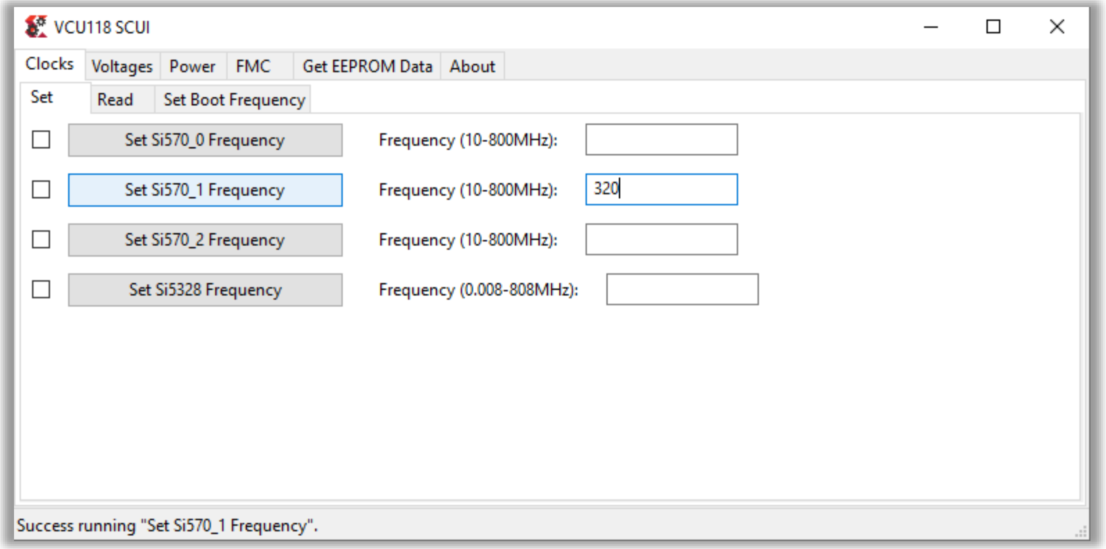
SFPA

CLK_C2M_P

EXT_CLK_P

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

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

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

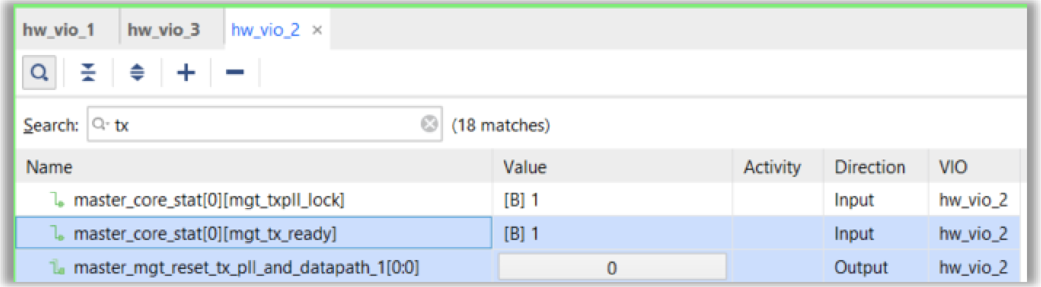
Success running "Set Si570_1 Frequency".

7. program FPGA ☺

Basic configuration

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

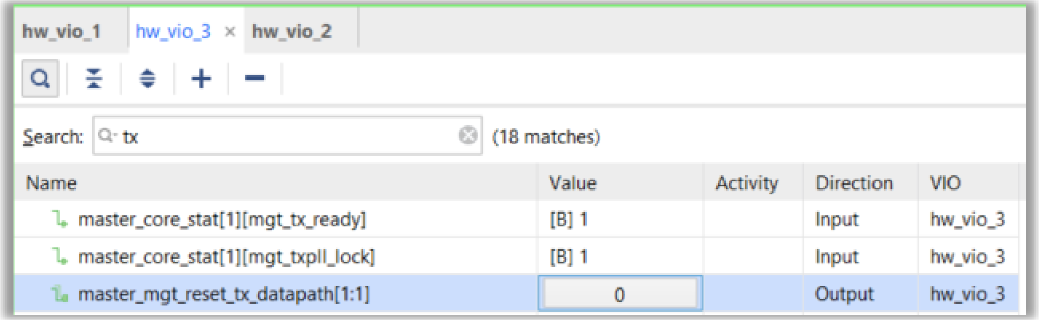
9. Reset Master0 Tx (Reset Tx PLL and datapath)
   - Check tx_ready



Search: tx (18 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[0][mgt_txpll_lock]	[B] 1		Input	hw_vio_2
master_core_stat[0][mgt_tx_ready]	[B] 1		Input	hw_vio_2
master_mgt_reset_tx_pll_and_datapath_1[0:0]	0		Output	hw_vio_2

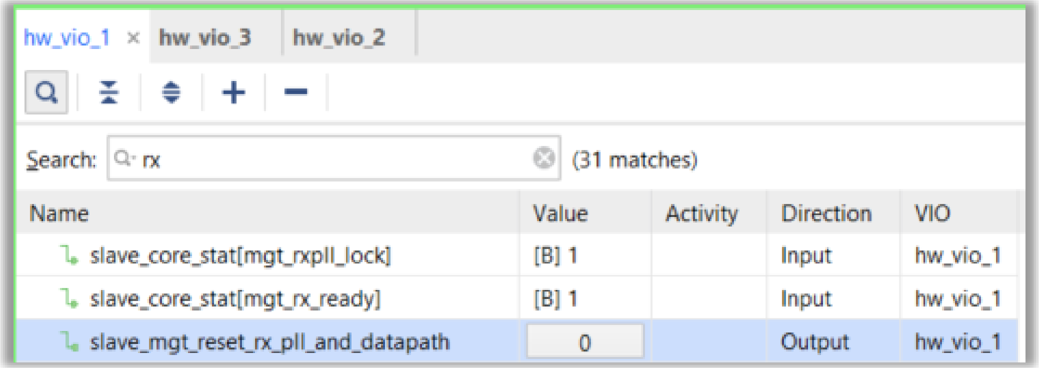
10. Reset Master1 Tx (Reset Tx datapath)
    - OBS: Master0 and Master1 share a QPLL and therefore only the reset of Tx PLL and datapath of Master0 is connected to PLL
    - Check tx_ready



Search: tx (18 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[1][mgt_tx_ready]	[B] 1		Input	hw_vio_3
master_core_stat[1][mgt_txpll_lock]	[B] 1		Input	hw_vio_3
master_mgt_reset_tx_datapath[1:1]	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.

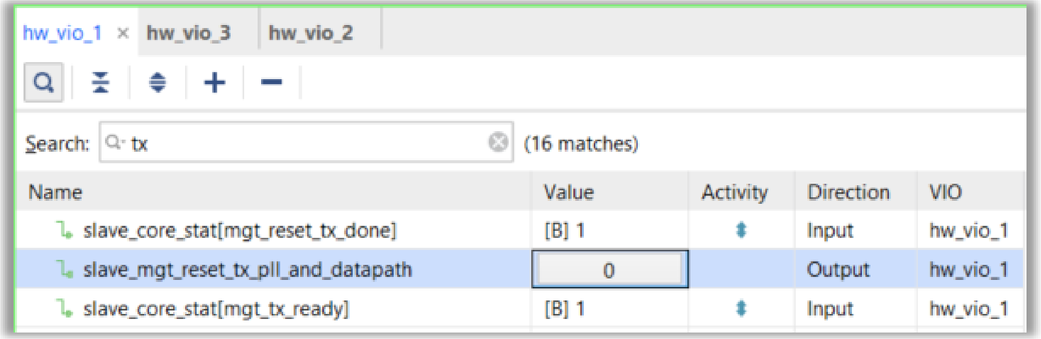


Search: rx (31 matches)

Name	Value	Activity	Direction	VIO
slave_core_stat[mgt_rxpll_lock]	[B] 1		Input	hw_vio_1
slave_core_stat[mgt_rx_ready]	[B] 1		Input	hw_vio_1
slave_mgt_reset_rx_pll_and_datapath	0		Output	hw_vio_1

Basic configuration

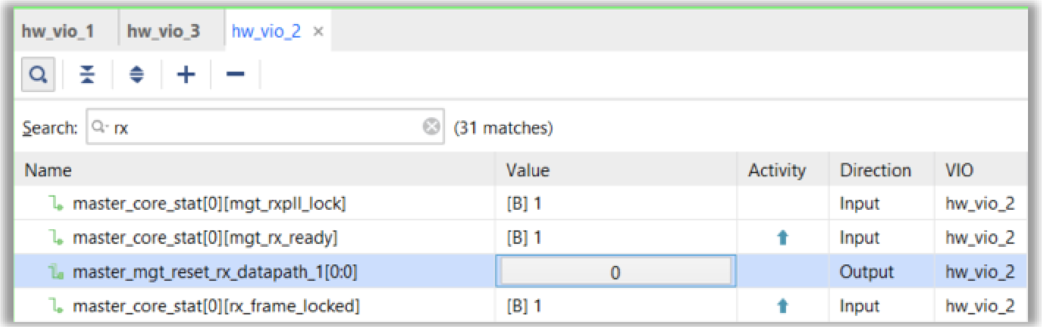
12. Reset Slave Tx (Reset Tx PLL and datapath)
    - Check tx_ready



Search: tx (16 matches)

Name	Value	Activity	Direction	VIO
slave_core_stat[mgt_reset_tx_done]	[B] 1	↕	Input	hw_vio_1
slave_mgt_reset_tx_pll_and_datapath	0		Output	hw_vio_1
slave_core_stat[mgt_tx_ready]	[B] 1	↕	Input	hw_vio_1

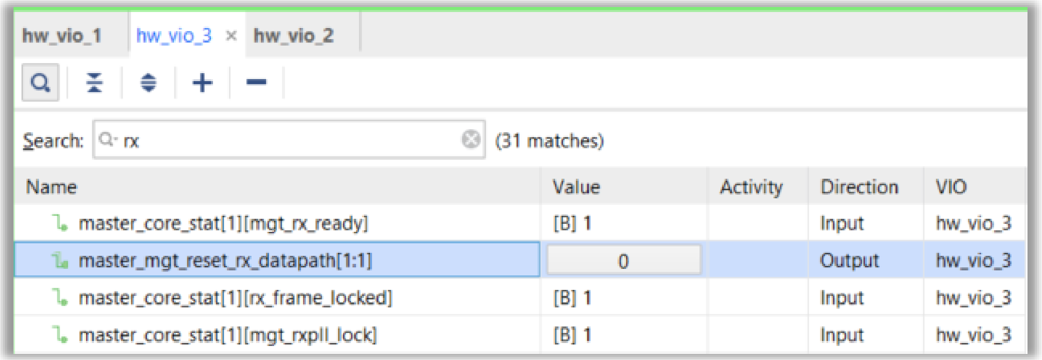
13. Reset Master0 Rx (Reset Rx datapath)
    - Check rx frame is locked
    - OBS: since master has Tx and Rx PLL shared, do not use Reset Rx PLL and datapath



Search: rx (31 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[0][mgt_rxpll_lock]	[B] 1		Input	hw_vio_2
master_core_stat[0][mgt_rx_ready]	[B] 1	↑	Input	hw_vio_2
master_mgt_reset_rx_datapath_1[0:0]	0		Output	hw_vio_2
master_core_stat[0][rx_frame_locked]	[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

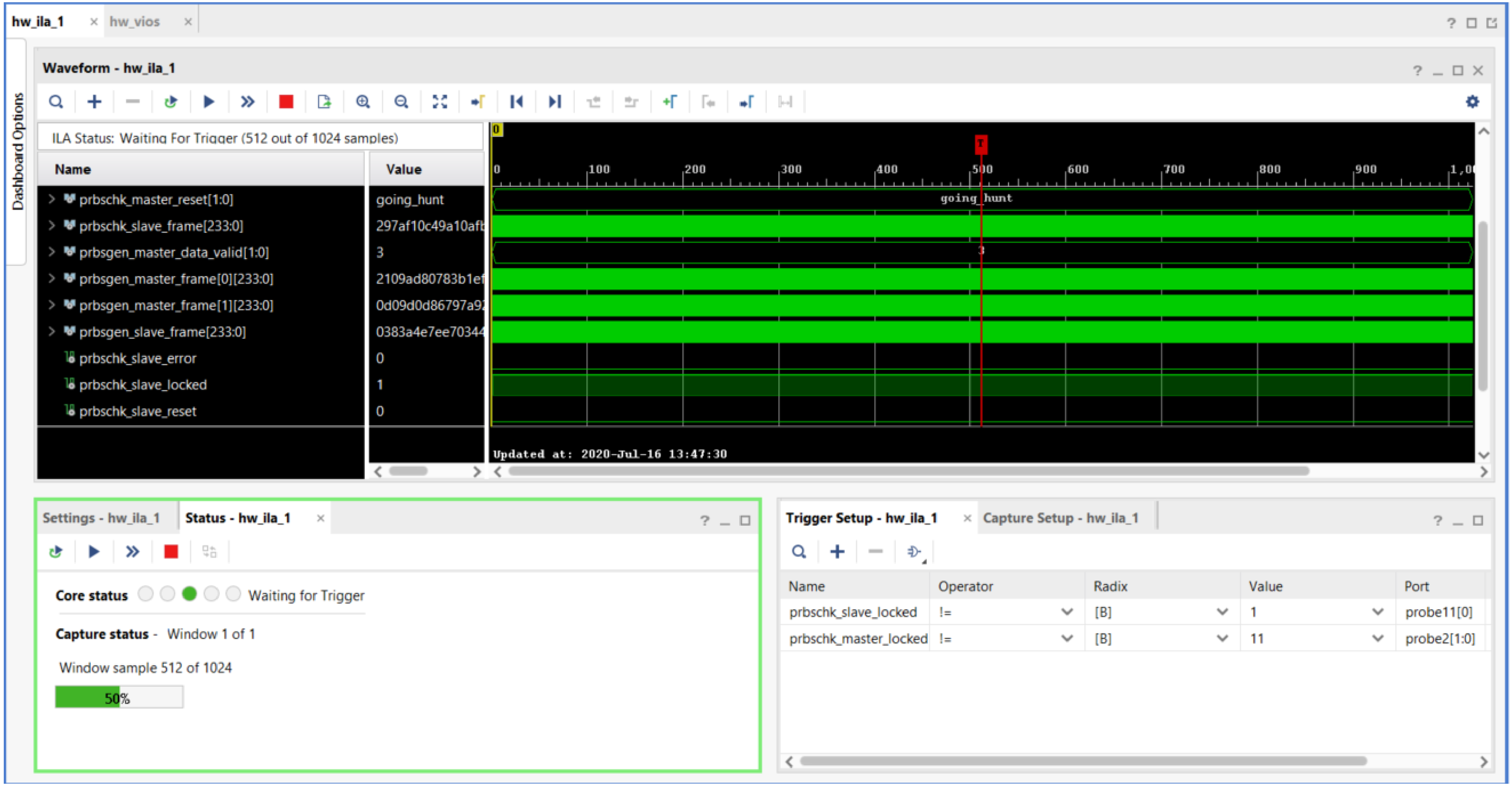


Search: rx (31 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[1][mgt_rx_ready]	[B] 1		Input	hw_vio_3
master_mgt_reset_rx_datapath[1:1]	0		Output	hw_vio_3
master_core_stat[1][rx_frame_locked]	[B] 1		Input	hw_vio_3
master_core_stat[1][mgt_rxpll_lock]	[B] 1		Input	hw_vio_3

Basic configuration

15. Check that all links are locked using the ILA



ILA Status: Waiting For Trigger (512 out of 1024 samples)

Name	Value
prbschk_master_reset[1:0]	going_hunt
prbschk_slave_frame[233:0]	297af10c49a10afb
prbsgen_master_data_valid[1:0]	3
prbsgen_master_frame[0][233:0]	2109ad80783b1ef
prbsgen_master_frame[1][233:0]	0d09d0d86797a92
prbsgen_slave_frame[233:0]	0383a4e7ee70344
prbschk_slave_error	0
prbschk_slave_locked	1
prbschk_slave_reset	0

Updated at: 2020-Jul-16 13:47:30

Core status ○ ○ ● ○ ○ Waiting for Trigger

Capture status - Window 1 of 1

Window sample 512 of 1024

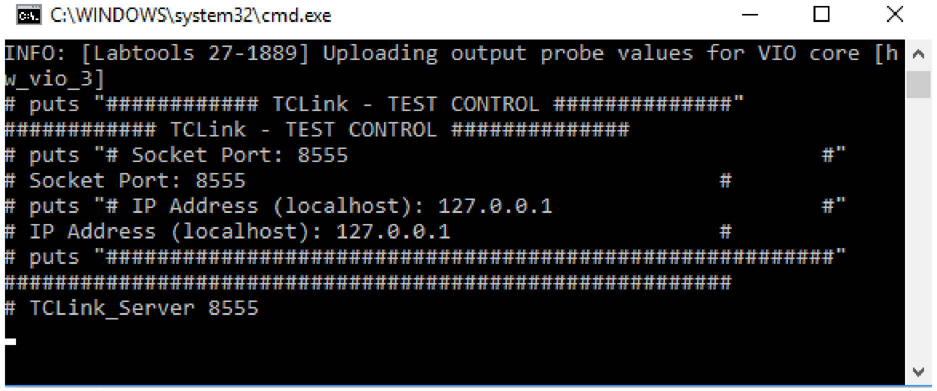
50%

Name	Operator	Radix	Value	Port
prbschk_slave_locked	!=	[B]	1	probe11[0]
prbschk_master_locked	!=	[B]	11	probe2[1:0]

Basic configuration

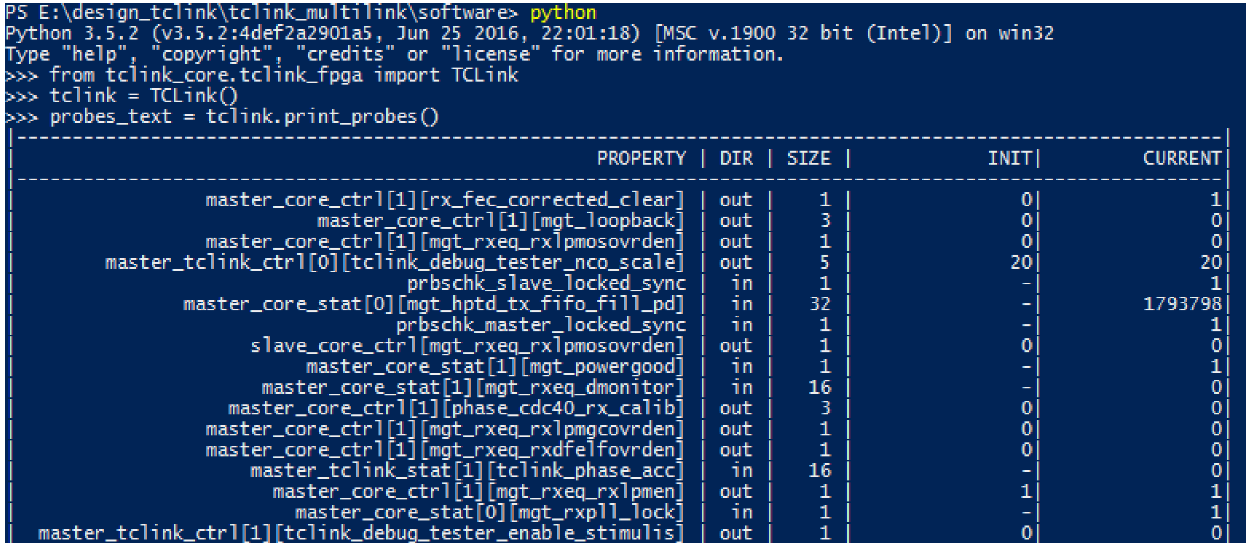
16. The design can be also controlled in **Python3** (for repetitive tests)
    a. Execute **software/jtag_server/jtag_server_vcu118.tcl** in Vivado batch mode (an example for Windows in software/jtag_server/execute_jtag_server_vcu118.bat)

Jtag server example


C:\WINDOWS\system32\cmd.exe

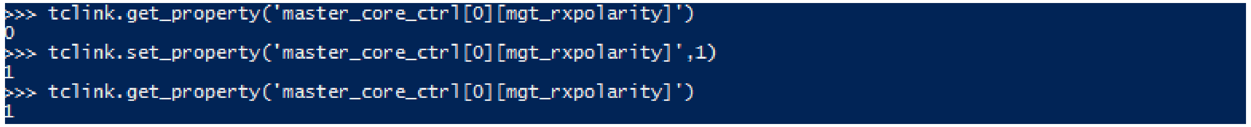
```
INFO: [Labtools 27-1889] Uploading output probe values for VIO core [h
w_vio_3]
# puts "############ TCLink - TEST CONTROL ##############"
############ TCLink - TEST CONTROL ##############
# puts "# Socket Port: 8555                                     #"
# Socket Port: 8555                                     #
# puts "# IP Address (localhost): 127.0.0.1                     #"
# IP Address (localhost): 127.0.0.1                     #
# puts "#########################################################"
#########################################################
# TCLink_Server 8555
```

Python example



```
PS E:\design_tclink\tclink_multilink\software> python
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> from tclink_core.tclink_fpga import TCLink
>>> tclink = TCLink()
>>> probes_text = tclink.print_probes()
|------------------------------------------------------------------------------------------------------------|
|                                                    PROPERTY | DIR | SIZE |            INIT|         CURRENT|
|------------------------------------------------------------------------------------------------------------|
|                 master_core_ctrl[1][rx_fec_corrected_clear] | out |    1 |               0|               1|
|                           master_core_ctrl[1][mgt_loopback] | out |    3 |               0|               0|
|                 master_core_ctrl[1][mgt_rxeq_rxlpmosovrden] | out |    1 |               0|               0|
|        master_tclink_ctrl[0][tclink_debug_tester_nco_scale] | out |    5 |              20|              20|
|                                   prbschk_slave_locked_sync |  in |    1 |               -|               1|
|               master_core_stat[0][mgt_hptd_tx_fifo_fill_pd] |  in |   32 |               -|         1793798|
|                                  prbschk_master_locked_sync |  in |    1 |               -|               1|
|                     slave_core_ctrl[mgt_rxeq_rxlpmosovrden] | out |    1 |               0|               0|
|                          master_core_stat[1][mgt_powergood] |  in |    1 |               -|               1|
|                      master_core_stat[1][mgt_rxeq_dmonitor] |  in |   16 |               -|               0|
|                   master_core_ctrl[1][phase_cdc40_rx_calib] | out |    3 |               0|               0|
|                 master_core_ctrl[1][mgt_rxeq_rxlpmgcovrden] | out |    1 |               0|               0|
|                 master_core_ctrl[1][mgt_rxeq_rxdfelfovrden] | out |    1 |               0|               0|
|                     master_tclink_stat[1][tclink_phase_acc] |  in |   16 |               -|               0|
|                       master_core_ctrl[1][mgt_rxeq_rxlpmen] | out |    1 |               1|               1|
|                         master_core_stat[0][mgt_rxpll_lock] |  in |    1 |               -|               1|
|  master_tclink_ctrl[1][tclink_debug_tester_enable_stimulis] | out |    1 |               0|               0|
```

...



```
>>> 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

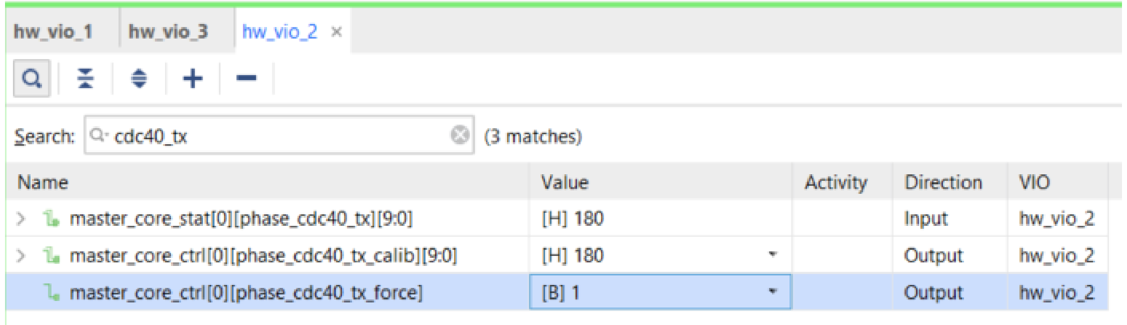
d. Set/get a probe value

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)
```

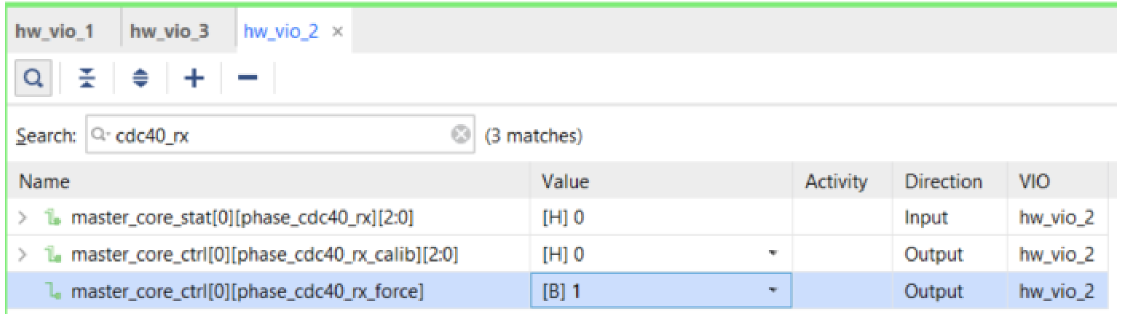


Search: cdc40_tx (3 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[0][phase_cdc40_tx][9:0]	[H] 180		Input	hw_vio_2
master_core_ctrl[0][phase_cdc40_tx_calib][9:0]	[H] 180		Output	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)
```



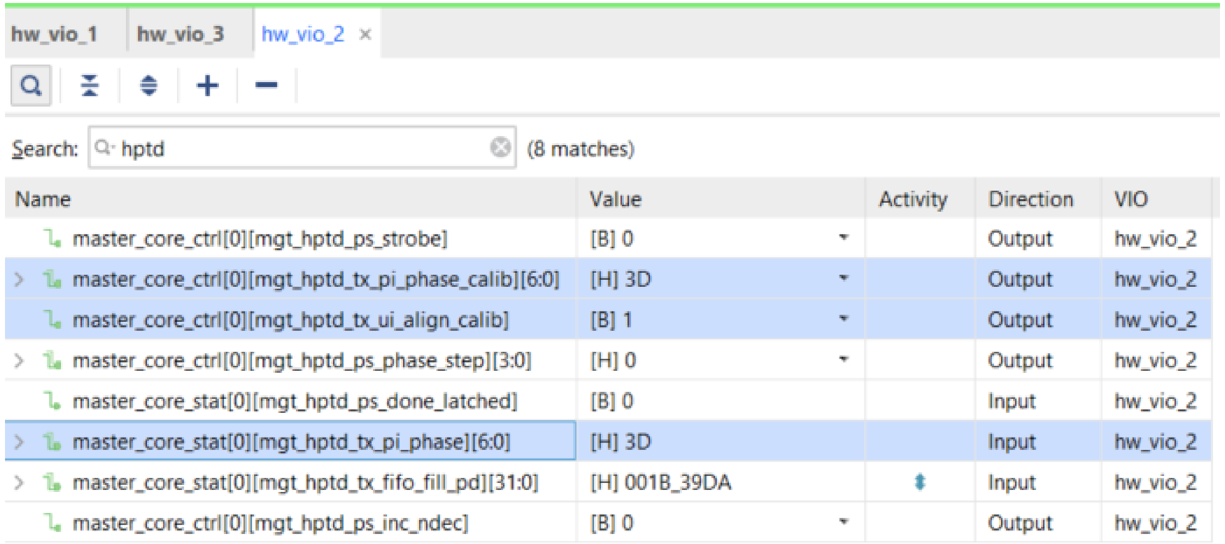
Search: cdc40_rx (3 matches)

Name	Value	Activity	Direction	VIO
master_core_stat[0][phase_cdc40_rx][2:0]	[H] 0		Input	hw_vio_2
master_core_ctrl[0][phase_cdc40_rx_calib][2:0]	[H] 0		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_ctrl[0][mgt_hptd_tx_pi_phase_calib][6:0]	[H] 3D		Output	hw_vio_2
master_core_ctrl[0][mgt_hptd_tx_ui_align_calib]	[B] 1		Output	hw_vio_2
master_core_ctrl[0][mgt_hptd_ps_phase_step][3:0]	[H] 0		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_pi_phase][6:0]	[H] 3D		Input	hw_vio_2
master_core_stat[0][mgt_hptd_tx_fifo_fill_pd][31:0]	[H] 001B_39DA	↕	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:

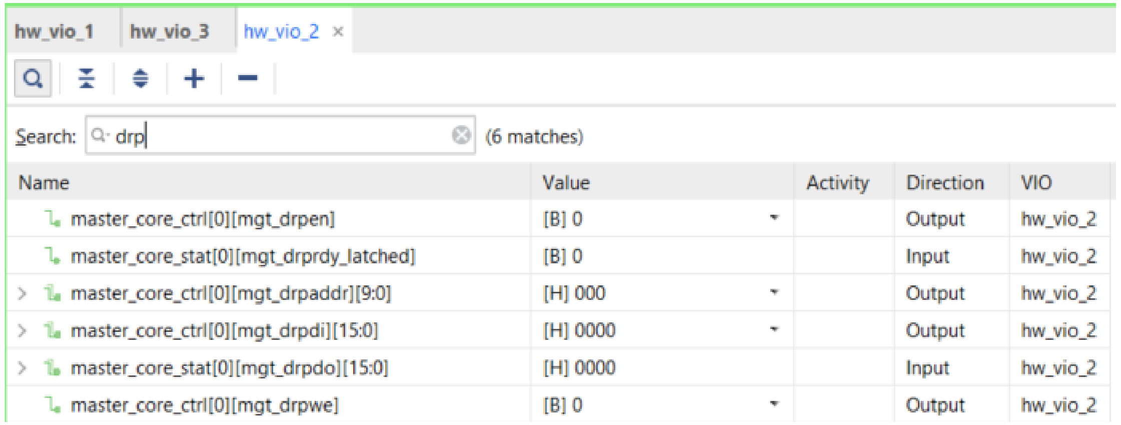
```
dut.set_property(‘master_core_ctrl[0][tclink_close_loop]’, 1)
```

- 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

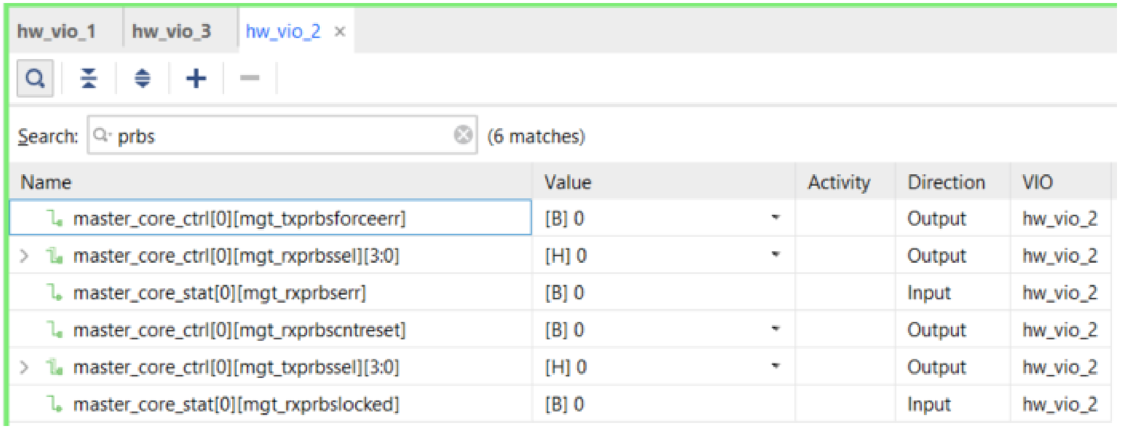


Search: drp (6 matches)

Name	Value	Activity	Direction	VIO
master_core_ctrl[0][mgt_drpen]	[B] 0		Output	hw_vio_2
master_core_stat[0][mgt_drprdy_latched]	[B] 0		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_stat[0][mgt_drpdo][15:0]	[H] 0000		Input	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

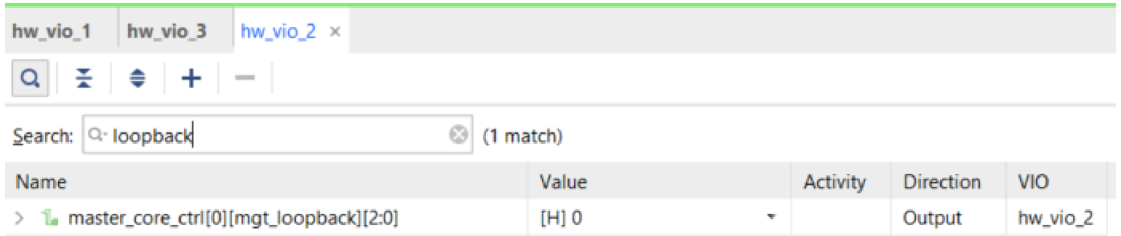


Search: prbs (6 matches)

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

- Transceiver Loopback

  000 = Normal operation
  001 = Near-End PCS Loopback
  010 = Near-End PMA Loopback
  100 = Far-End PMA Loopback
  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