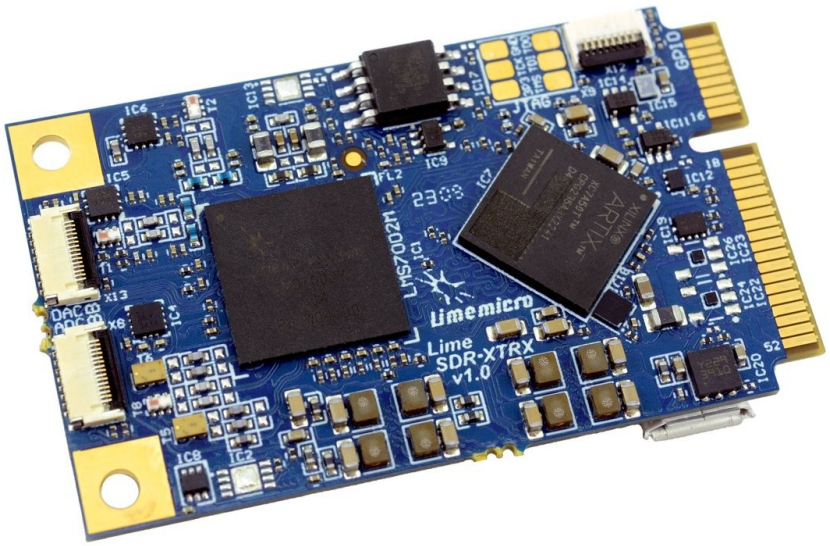
Lime Microsystems Limited

Surrey Technology Centre
Occam Road
The Surrey Research Park
Guildford, Surrey GU2 7YG
United Kingdom

Tel: +44 (0) 1483 685 063
e-mail: enquiries@limemicro.com

LimeSDR-XTRX v1.2
User Manual

- Gateware description-



Version: 1.01 Last modified: 05/02/2024

REVISION HISTORY

The following table shows the revision history of this document:

Date	Version	Description of Revisions
22/01/2024	1.00	Initial version

Table of Contents

1 Introduction

This is the user manual for LimeSDR-XTRX gateware. This document provides build instructions and information on features is implemented in this gateware. It contains the following features:

- PCIe connection to transfer data between host and FPGA
- LMS64 protocol implementation in soft-core CPU
- User accessible configuration registers
- Interface to LMS7002 LimeLightTM digital IQ interface in TRXIQ double data rate mode
- TX samples synchronization with RX samples time stamp
- Reconfigurable PLL blocks for LMS7002 clocking

It is assumed that user is familiar with hardware if not it is suggested to review available user guides and hardware description manuals first.

1.1 Software and hardware requirements

Required hardware:

- LimeSDR-XTRX v1.2 - Lime Microsystems mini PCIe expansion card SDR board

Required gateware:

- https://github.com/myriadrf/LimeSDR-XTRX_GW – gateware for LimeSDR-XTRX v1.2

Required software:

- Vivado 2022.1 Standard edition – Xilinx FPGA design software

2 Building the gateware

Instructions for building the gateware can be found in the README file of the gateware repository.

3 Board programming

Instructions for programming the board can be found in the README fileof the gateware repository.

4 Gateware description

Description for LimeSDR-XTRX gateware can be found in this chapter.

4.1 Main block diagram

Top level file LimeSDR_XTRX_top.vhd has all required external ports and lower level module instantiated. It also has some reset logic and clocks generated for rest of the instances.

LimeSDR-XTRX provides a DMA interface with a PCIe host. There are two channels (F2H_C0 and H2F_C0) implemented for control data and two channels for stream data (F2H_S0 and H2F_S0). Control endpoints are connected to the MicroBlaze soft-core processor, which provides SPI and I2C communication interfaces for the LMS7002M chip, XO DAC, FLASH, and EEPROM memories. MicroBlaze also provides access to internal SPI configuration registers. Stream channels are dedicated to receiving and sending IQ data from/to the LMS7002M.

High level description for instances can be found in Table 1 and main block diagram in Figure 1. More details for each instance can be found in the following sections.

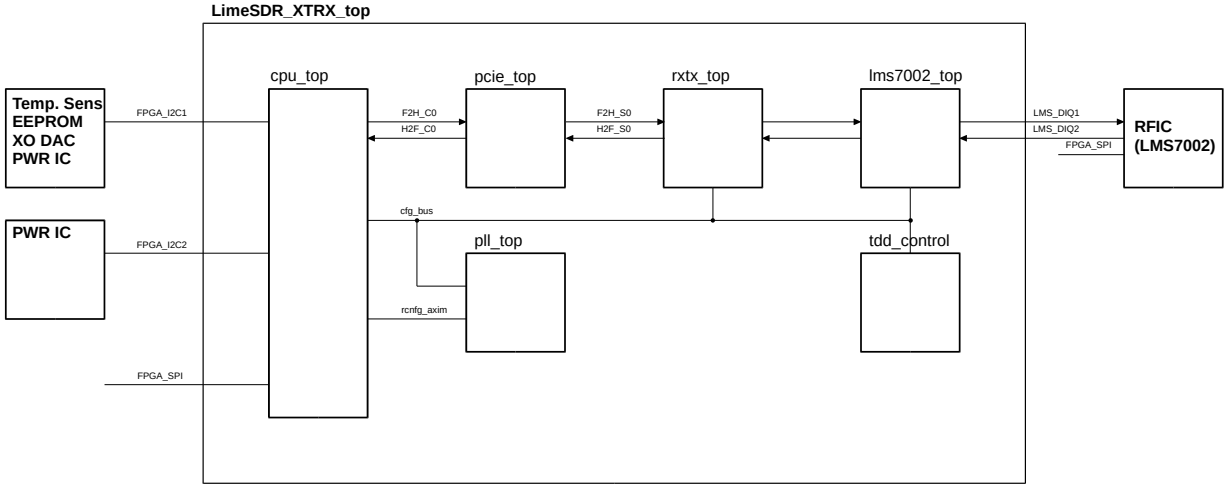


Figure 1: LimeMM-X8 main block diagram

Table 1: Instances of main block diagram

Instance name	Description
LimeSDR_XTRX_top	Top module
inst0_pcie_top	PCIe bus implementation
inst1_cpu	Microblaze soft-core CPU
inst2_pll_top	Clock control
inst3_rxtx_top	Receive and transmit logic between FPGA and external LMS7002 transceiver.
inst4_lms7002_top	Digital data and control interface for LMS7002 IC.
inst5_tdd_control	Control logic for TDD RF control
inst6_tst_top	Clock and GNSS test modules

4.1.1 PCIe interface with host - *inst0_pcie_top*

Module *pcie_top* implements PCIe x2 Gen2 endpoint. This block uses two GTP transceivers and AMD 7 Series Integrated Block for PCI Express v3.3 IP core as a physical layer. Module *pcie_top* handles TLP layer and provides one channel for sending/receiving LMS64 control packets and one DMA channel for sending/receiving stream data packets.

More information about litepcie_core can be found in Github repository: https://github.com/enjoy-digital/litepcie/tree/master.

Table 2: pcie_top module generics

Generic name	Type	Value	Description
g_DEV_FAMILY	string	""	Device family
g_S0_DATA_WIDTH	integer	32	Stream 0 data width
g_C0_DATA_WIDTH	integer	8	Control 0 data width
g_H2F_S0_0_RDUSEDW_WIDTH	integer	11	Stream 0_0 FIFO read used words width
g_H2F_S0_0_RWIDTH	integer	32	Stream 0_0 FIFO read data with
g_H2F_S0_1_RDUSEDW_WIDTH	integer	11	Stream 0_1 FIFO read used words width
g_H2F_S0_1_RWIDTH	integer	32	Stream 0_1 FIFO read data with
g_F2H_S0_WRUSEDW_WIDTH	integer	10	Stream 0 FIFO write used words width
g_F2H_S0_WWIDTH	integer	64	Stream 0 FIFO write width
g_H2F_C0_RDUSEDW_WIDTH	integer	11	Control 0 FIFO read used words width
g_H2F_C0_RWIDTH	integer	8	Control 0 FIFO read width
g_F2H_C0_WRUSEDW_WIDTH	integer	11	Control 0 FIFO write used words width
g_F2H_C0_WWIDTH	integer	8	Control 0 FIFO write width

pcie_top port descriptions can be found in Table 3.

Table 3 pcie_top port description

Port name	Direction	Type	Description
clk	out	std_logic	Internal logic clock (125Mhz)
reset_n	in	std_logic	Active low reset
pcie_perstn	in	std_logic	PCIe fundamental reset
pcie_refclk_p	in	std_logic	PCIe reference clock
pcie_refclk_n	in	std_logic	PCIe reference clock
pcie_rx_p	in	std_logic_vector (1 downto 0)	PCIe receiver
pcie_rx_n	in	std_logic_vector (1 downto 0)	PCIe receiver
pcie_tx_p	out	std_logic_vector (1 downto 0)	PCIe transmitter
pcie_tx_n	out	std_logic_vector (1 downto 0)	PCIe transmitter
H2F_S0_sel	in	std_logic	Stream select: 0 - S0_0, 1 - S0_1
H2F_S0_dma_en	out	std_logic	Host->FPGA stream ready
S0_rx_en	in	std_logic	Stream 0 enable
F2H_S0_open	out	std_logic	FPGA->Host stream 0 ready
H2F_S0_0	in	Virtual bus	Stream 0 endpoint FIFO 0 (Host->FPGA)
H2F_S0_1	in	Virtual bus	Stream 0 endpoint FIFO 1 (Host->FPGA)
F2H_S0	out	Virtual bus	Stream 0 endpoint FIFO (FPGA->Host)
H2F_C0	in	Virtual bus	Control endpoint FIFO (Host->FPGA)
F2H_C0	out	Virtual bus	Control endpoint FIFO (FPGA->Host)

Table 4 H2F_S0_0 virtual bus description

Port name	Direction	Type	Description
H2F_S0_0_rdclk	in	std_logic	Read clock
H2F_S0_0_aclrn	in	std_logic	Asynchronous clear
H2F_S0_0_rd	in	std_logic	Read enable
H2F_S0_0_rdata	out	std_logic_vector(g_H2F_S0_0_RWIDTH-1 downto 0)	Read data
H2F_S0_0_rempty	out	std_logic	Read empty
H2F_S0_0_rdusedw	out	std_logic_vector(g_H2F_S0_0_RDUSEDW_WIDTH-1 downto 0)	Read used words

Table 5 H2F_S0_1 virtual bus description

Port name	Direction	Type	Description
H2F_S0_1_rdclk	in	std_logic	Read clock
H2F_S0_1_aclrn	in	std_logic	Asynchronous clear
H2F_S0_1_rd	in	std_logic	Read enable
H2F_S0_1_rdata	out	std_logic_vector(g_H2F_S0_1_RWIDTH-1 downto 0)	Read data
H2F_S0_1_rempty	out	std_logic	Read empty
H2F_S0_1_rdusedw	out	std_logic_vector(g_H2F_S0_1_RDUSEDW_WIDTH-1 downto 0)	Read used words

Table 6 F2H_S0 virtual bus description

Port name	Direction	Type	Description
F2H_S0_wclk	in	std_logic	Write clock
F2H_S0_aclrn	in	std_logic	Asynchronous clear
F2H_S0_wr	in	std_logic	Write enable
F2H_S0_wdata	in	std_logic_vector(g_F2H_S0_WWIDTH-1 downto 0)	Write data
F2H_S0_wfull	out	std_logic	Write full
F2H_S0_wrusedw	out	std_logic_vector(g_F2H_S0_WRUSEDW_WIDTH-1 downto 0)	Write used words

Table 7 H2F_C0 virtual bus description

Port name	Direction	Type	Description
H2F_C0_rdclk	in	std_logic	Read clock
H2F_C0_aclrn	in	std_logic	Asynchronous clear
H2F_C0_rd	in	std_logic	Read enable
H2F_C0_rdata	out	std_logic_vector(g_H2F_C0_RWIDTH-1 downto 0)	Read data
H2F_C0_rempty	out	std_logic	Read empty

Table 8 F2H_C0 virtual bus description

Port name	Direction	Type	Description
F2H_C0_wclk	in	std_logic	Write clock
F2H_C0_aclrn	in	std_logic	Asynchronous clear
F2H_C0_wr	in	std_logic	Write enable
F2H_C0_wdata	in	std_logic_vector(g_F2H_C0_WWIDTH-1 downto 0)	Write data
F2H_C0_wfull	out	std_logic	Write full

4.1.2 Softcore CPU – inst1_*cpu_top*

Module *cpu_top* is a wrapper for AMD XILINX MicroBlaze Soft Processor Core and user accessible register blocks. Application code of MicroBlaze has following functions:

- LMS64 protocol decoder/encoder
- SPI access to user register blocks
- I2C device control

High level block diagram can be found in Figure 2

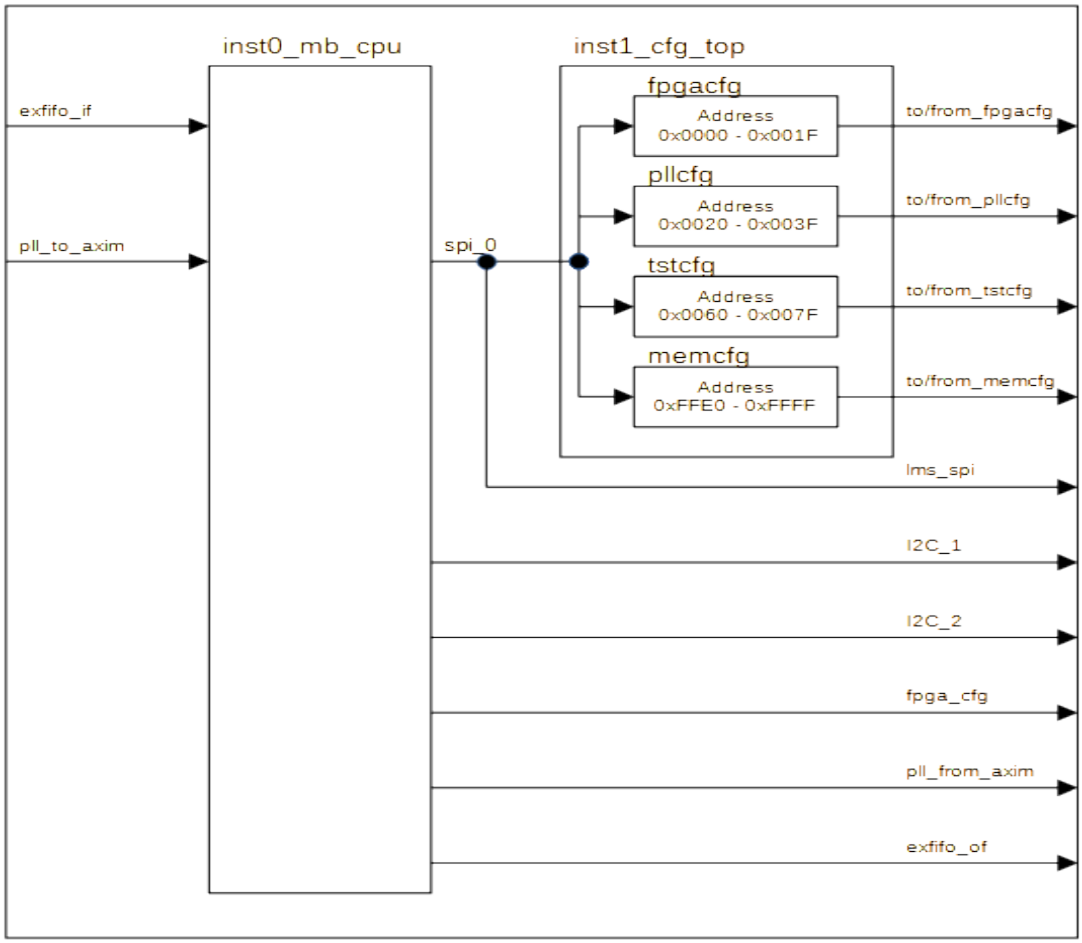


Figure 2: cpu_top block diagram

Module parameter descriptions can be found in Table 9.

Table 9: cpu_top module parameters

Generic name	Type	Value	Description
FPGACFG_START_ADDR	integer	0	FPGACFG register start address
PLLCFG_START_ADDR	integer	32	PLLCFG register start address
TSTCFG_START_ADDR	integer	96	TSTCFG register start address
MEMCFG_START_ADDR	integer	65504	MEMCFG register start address

Module port descriptions can be found in tables: Table 3, Table 10, Table 11, Table 12, Table 14, Table 15, Table 16, Table 28.

Table 10: exfifo_if virtual bus ports

Port name	Direction	Type	Description
EXFIFO_IF_D	in	std_logic_vector(31 downto 0)	Read data
EXFIFO_IF_RD	out	std_logic	Read enable
EXFIFO_IF_RDEMPTY	in	std_logic	Read empty

Table 11: exfifo_of virtual bus ports

Port name	Direction	Type	Description
EXFIFO_OF_D	out	std_logic_vector(31 downto 0)	Write data
EXFIFO_OF_WR	out	std_logic	Write enable
EXFIFO_OF_WRFULL	in	std_logic	Write full
EXFIFO_OF_RST	out	std_logic	Reset, active high

Table 12: spi_0 virtual bus ports

Port name	Direction	Type	Description
SPI_0_MISO	in	std_logic	Master In Slave Out
SPI_0_MOSI	out	std_logic	Master Out Slave In
SPI_0_SCLK	out	std_logic	Clock output
SPI_0_SS_N	out	std_logic_vector(1 downto 0)	Slave Select, active low [0] - internal registers, [1] - LMS7002

Table 13: cpu_top module ports

Port name	Dir.	Type	Description
CLK	in	std_logic	Clock
RESET_N	in	std_logic	Reset, active low
GPI	in	std_logic_vector(7 downto 0)	General purpose inputs (UNUSED)
GPO	out	std_logic_vector(7 downto 0)	General purpose outputs (UNUSED)
VCTCXO_TUNE_EN	in	std_logic	Unused
VCTCXO_IRQ	in	std_logic	Unused
PLL_RST	out	std_logic_vector(1 downto 0)	PLL reset
PLL_AXI_RESETN_OUT	out	std_logic_vector(0 to 0)	PLL Axi interface reset
PLL_FROM_AXIM	out	t_FROM_AXIM_32x32	PLL AXI interface CPU -> PLL
PLL_TO_AXIM	in	t_TO_AXIM_32x32	PLL AXI interface PLL -> CPU
PLL_AXI_SEL	out	std_logic_vector(3 downto 0)	PLL AXI slave select
FROM_FPGACFG	out	t_FROM_FPGACFG	FPGACFG register bus Registers -> Modules
TO_FPGACFG	in	t_TO_FPGACFG	FPGACFG register bus Modules -> Registers
FROM_PLLCFG	out	t_FROM_PLLCFG	PLLCFG register bus Registers -> Modules
TO_PLLCFG	in	t_TO_PLLCFG	PLLCFG register bus Modules -> Registers
FROM_TSTCFG	out	t_FROM_TSTCFG	TSTCFG register bus Registers -> Modules
TO_TSTCFG	in	t_TO_TSTCFG	TSTCFG register bus Modules -> Registers
TO_MEMCFG	in	t_TO_MEMCFG	MEMCFG register bus Registers -> Modules
FROM_MEMCFG	out	t_FROM_MEMCFG	MEMCFG register bus Modules -> Registers
SMPL_CMP_EN	out	std_logic_vector(0 downto 0)	Sample compare module enable
SMPL_CMP_STATUS	in	std_logic_vector(1 downto 0)	Sample compare module status
EXFIFO_IF	in	Virtual bus	Control packet fifo Host -> FPGA
EXFIFO_OF	out	Virtual bus	Control packet fifo FPGA -> HOST
SPI_0	out	Virtual bus	SPI interface 0, used for internal registers and LMS7002
I2C_1	out	Virtual bus	I2C interface 1, used for Temperature sensor, XO DAC, Switching voltage regulator 1 (IC22)
I2C_2	out	Virtual bus	I2C interface 2, used for Switching voltage regulator 2 (IC31)
FPGA_CFG	out	Virtual bus	SPI interface for configuration flash
AVMM_M0	out	Virtual bus	Avalon master interface (UNUSED)

Table 14: I2C_1 virtual bus ports

Port name	Direction	Type	Description
I2C_1_SCL	inout	std_logic	Clock signal
I2C_1_SDA	inout	std_logic	Data signal

Table 15: I2C_2 virtual bus ports

Port name	Direction	Type	Description
I2C_2_SCL	inout	std_logic	Clock signal
I2C_2_SDA	inout	std_logic	Data signal

Table 16: fpga_cfg virtual bus ports

Port name	Direction	Type	Description
FPGA_CFG_QSPI_MISO	in	std_logic	Master In Slave Out
FPGA_CFG_QSPI_MOSI	out	std_logic	Master Out Slave In
FPGA_CFG_QSPI_SS_N	out	std_logic	Slave Select, active low

Table 17: avmm_m0 virtual bus ports

Port name	Direction	Type	Description
AVMM_M0_ADDRESS	out	std_logic_vector(7 downto 0)	Adress
AVMM_M0_READ	out	std_logic	Read
AVMM_M0_WAITREQUEST	in	std_logic	Wait request
AVMM_M0_READDATA	in	std_logic_vector(7 downto 0)	Read data
AVMM_M0_READDATAVALID	in	std_logic	Read data valid
AVMM_M0_WRITE	out	std_logic	Write
AVMM_M0_WRITEDATA	out	std_logic_vector(7 downto 0)	Write data
AVMM_M0_CLK_CLK	out	std_logic	Clock
AVMM_M0_RESET_RESET	out	std_logic	Reset

4.1.2.1 FPGACFG module registers

Table 18: FPGACFG module registers (0x0000-0x0009)

Address	Def. Value	Bits	Name	Description
0x0000	0x001B	15-0	board_id	Board ID (read only)
0x0001	-	15-0	Major rev	Major revision (read only).
0x0002	-	15-0	compile_rev	Compile revision (read only).
0x0003	-	15-0	Reserved	
0x0004	0x0000	15-0	Reserved	
0x0005	0x0000	15-0	Reserved	
0x0006	0x0000	15-0	Reserved	
0x0007	0x0003	15-2	Reserved	
		1-0	ch_en	Channel enable 01 – Channel A 10 – Channel B 11 – Channels A and B
0x0008	0x0102	15-10	Reserved	
		9	synch_dis	Packets synchronization using timestamps: 0 - Enabled 1 - Disabled (Default)
		8	mimo_int_en	MIMO mode: 0 - Disabled 1 - Enabled (Default)
		7	trxiq_pulse	TRXIQ_pulse mode: 0 - OFF (Default) 1 – ON
		6	ddr_en	DIQ interface mode: 0 - SDR 1 - DDR (Default)
		5-2	Reserved	
		1-0	smpl_width	Interface sample width selection: "10" - 12bit (Default) "01" - Do not use "00" - 16bit
0x0009	0x0003	15-2	Reserved	
		1	txpct_loss_clr	TX packets dropping flag clear: 0 - Normal operation (Default) 1 - Rising edge clears flag
		0	smpl_nr_clr	Reset_timestamp: 0 - Normal operation (Default) 1 - Timestamp is cleared

Table 19: FPGACFG module registers (0x000A-0x0010)

Address	Def. Value	Bits	Name	Description
0x000A	0x0000	15-12	Reserved	
		11	rf_sw_auto_en	Control of RF switches by internal TDD signal: 0 – Disabled (Default) 1 – Enabled
		10	tx_cnt_en	Counter test pattern on TX: 0 – Disabled (Default) 1 – Enabled
		9	tx_ptrn_en	Test pattern on TX: 0 – Disabled (Default) 1 – Enabled
		8	rx_ptrn_en	Test pattern on RX: 0 – Disabled (Default) 1 – Enabled
		7	tdd_invert	Invert external TDD signal: 0 – Disabled (Default) 1 – Enabled
		6	tdd_auto_en	Control external TDD signal by internal TDD signal: 0 – Disabled 1 – Enabled
		5	tdd_manual	Manual value of external TDD signal
		4	tx_rf_sw	TX RF switch selection: 0 – TX 2 (Default) 1 – TX 1
		3-2	rx_rf_sw	RX RF switch selection: 00 – RX_W 01 – RX_L 10 – RX_H 11 – No connection
		1	Reserved	
		0	rx_en	RX/TX unified enable: 0 – Disabled (Default) 1 – Enabled
0x000B	0x0000	15-0	Reserved	
0x000C	0x0003	15-0	Reserved	
0x000D	0x0000	15-0	Reserved	
0x000E	0x0000	15-0	RX_PACKET_SAMPLES	RX packet size in samples
0x000F	0x03FC	15-0	Reserved	
0x0010	0x0001	15-0	txant_pre	How many samples to delay turning on internal TDD singal

Table 20: FPGACFG module registers (0x0011-0x0017)

Address	Def. Value	Bits	Name	Description
0x0011	0x0001	15-0	txant_post	How many samples to delay turning off internal TDD singal
0x0012	0xFFFF	15-0	Reserved	
0x0013	0x6F6B	15-8	Reserved	
		7	LMS_TXRXEN_MUX_SEL	Control LMS TX/RXEN signals by internal TDD signal 0 – Disabled (Default) 1 – Enabled
		6	LMS1_RXEN	RX hard enable: 0 - Disabled 1 - Enabled (Default)
		5	LMS1_TXEN	TX hard enable: 0 - Disabled 1 - Enabled (Default)
		4	LMS1_TXNRX2	Port 2 mode selection: 0 - TXIQ (Default) 1 - RXIQ
		3	LMS1_TXNRX1	Port 1 mode selection: 0 - TXIQ 1 - RXIQ (Default)
		2	LMS1_CORE_LDO_EN	Internal LDO control(UNUSED): 0 - Disabled (Default) 1 – Enabled
		1	LMS1_RESET	Hardware reset: 0 - Reset activated 1 - Reset inactive (Default)
		0	Reserved	
0x0014	0x0003	15-0	Reserved	
0x0015	0x0000	15-0	Reserved	
0x0016	0x0000	15-0	Reserved	
0x0017	0x2340	15-0	Reserved	

Table 21: FPGACFG module registers (0x0018-0x001F)

Address	Def. Value	Bits	Name	Description
0x0018	0x0003	15-4	Reserved	
		3	CORE_LDO_EN	LMS internal LDO control 0 – Disabled (Default) 1 – Enabled
		2	EXT_CLK	Clock source selection 0 – Onboard clock (Default) 1 – External clock
		1	TCXO_EN	Onboard clock enable 0 – Disabled 1 – Enabled (Default)
		0	LMS_RST	LMS hardware reset 0 - Reset activated 1 - Reset inactive (Default)
0x0019	0x1000	15-0	RX_PACKET_SIZE	RX packet size in bytes
0x001A	0x0000	15-0	Reserved	
0x001B	0x0000	15-0	Reserved	
0x001C	0x0000	15-0	Reserved	
0x001D	0x00FF	15-0	Reserved	
0x001E	0x0003	15-0	Reserved	
0x001F	0xD090	15-0	Reserved	

4.1.2.2 PLLCFG module registers

Table 22: PLLCFG module registers (0x0020-0x0025)

Address	Def. Value	Bits	Name	Description
0x0020	0x0000	0-15	C1 Phase	Phase value for PLL output clock 1
0x0021	0x0001	4-15	Reserved	
		3	phcfg_error	Phase config error (unused)
		2	phcfg_done	Phase config done (read only) 0 – Not done 1 – Done
		1	pllcfg_busy	PLL config busy (read only) 0 – Idle 1 – Busy
		0	pllcfg_done	PLL config done (read only) 0 – Not done 1 – Done
0x0022	0x0000	15-8	pllcfg_err	PLL config error (unused)
		7-2	pll_lock	Reserved
		1		RX PLL 0 – No Lock 1 – Locked
		0		TX PLL 0 – No Lock 1 – Locked
0x0023	0x0000	15	Reserved	
		14	phcfg_mode	PLL phase configuration mode: 0 – Manual 1 – Auto
		13-8	Reserved	
		7-3	pll_ind	PLL index for reconfiguration: 0000 - TX PLL 0001 - RX PLL Do not use other index values
		2	Reserved	
		1	phcfg_start	Start phase configuration 0 – Phase configuration inactive 0 to 1 transition – start configuration
		0	Reserved	
0x0024	0x0000	15-0	Reserved	
0x0025	0x01F0	15-0	Reserved	

Table 23: FPGACFG module registers (0x0026-0x003F)

Address	Def. Value	Bits	Name	Description
0x0026	0x000A	15-3	Reserved	
		2	m_byp	PLL multiplier bypass
		1	Reserved	
		0	n_byp	PLL divider bypass
0x0027	0xAAA	15-3	Reserved	
		2	c1_byp	PLL output 1 divider bypass
		1	Reserved	
		0	c0_byp	PLL output 0 divider bypass
0x0028	0xAAAA	15-0	Reserved	
0x0029	0xAAAA	15-0	Reserved	
0x002A	0x0000	15-0	n_cnt	PLL divider value
0x002B	0x0000	15-0	m_cnt	PLL multiplier value
0x002C	0x0000	15-0	m_frac(LSB)	PLL multiplier fractional value
0x002D	0x0000	15-0	m_frac(MSB)	PLL multiplier fractional value
0x002E	0x0000	15-0	c0_cnt	PLL output 0 divider value
0x002F	0x0000	15-0	c1_cnt	PLL output 1 divider value
0x0030	0xEFFF	15-0	auto_phcfg_smpls	Number of samples to use during auto phase configuration
0x0031	0x0000	15-0	Reserved	
0x0032	0x0000	15-0	Reserved	
0x0033	0x0000	15-0	Reserved	
0x0034	0x0000	15-0	Reserved	
0x0035	0x0000	15-0	Reserved	
0x0036	0x0000	15-0	Reserved	
0x0037	0x0000	15-0	Reserved	
0x0038	0x0000	15-0	Reserved	
0x0039	0x0000	15-0	Reserved	
0x003A	0x0000	15-0	Reserved	
0x003B	0x0000	15-0	Reserved	
0x003C	0x0000	15-0	Reserved	
0x003D	0x0000	15-0	Reserved	
0x003E	0x0000	15-0	Reserved	
0x003F	0x0000	15-0	Reserved	

4.1.2.3 TSTCFG module registers

Table 24: TSTCFG module registers (0x0060-0x006D)

Address	Def. Value	Bits	Name	Description
0x0060	0x0000	15-0	Reserved	
0x0061	0x0000	15-3	Reserved	
		2	test_en	GNSS test: 0 – Idle (Default) 1 – Test started
		1		LMS_TX_CLK test: 0 – Idle (Default) 1 – Test started
		0		sys_clk test: 0 – Idle (Default) 1 – Test started
0x0062	0x0000	15-0	Reserved	
0x0063	0x0000	15-0	Reserved	
0x0064	0x0000	15-0	Reserved	
0x0065	0x0000	15-3	Reserved	
		2	test_cmplt	GNSS test: 0 – Test not finished 1 – Test finished
		1		LMS_TX_CLK test: 0 – Test not finished 1 – Test finished
		0		sys_clk_test 0 – Test not finished 1 – Test finished
0x0066	0x0000	15-0	Reserved	
0x0067	0x0000	15-3	Reserved	
		2	test_rez	GNSS test result. (not implemented)
		1		LMS_TX_CLK test result. (not implemented)
		0		sys_clk test result. (not implemented)
0x0068	0x0000	15-0	Reserved	
0x0069	0x0000	15-0	sys_clk_cnt	Number of sys_clk cycles counted by sys_clk test. Different values on subsequent reads indicate clock is active
0x006A	0x0000	15-0	Reserved	
0x006B	0x0000	15-0	Reserved	
0x006C	0x0000	15-0	Reserved	
0x006D	0x0000	15-0	Reserved	

Table 25: TSTCFG module registers (0x006E-0x007F)

Address	Def. Value	Bits	Name	Description
0x006E	0x0000	15-0	Reserved	
0x006F	0x0000	15-0	Reserved	
0x0070	0x0000	15-0	Reserved	
0x0071	0x0000	15-0	Reserved	
0x0072	0x0000	15-0	lms_tx_clk_cnt	15-0 bits of cycles counted by lms_tx_clk test
0x0073	0x0000	15-8	Reserved	
		7-0	lms_tx_clk_cnt	23-16 bits of cycles counted by lms_tx_clk test.
0x0074-0x007C	0x0000	15-0	Reserved	
0x007D	0xAAAA	15-0	tx_tst_i	Test value for tx I channel
0x007E	0x5555	15-0	tx_tst_q	Test value for tx Q channel
0x007F	0x0000		Reserved	

4.1.2.4 MEMCFG module registers

Table 26: MEMCFG module registers (0xFFE0-0xFFFF)

Address	Def. Value	Bits	Name	Description
0xFFE0	0x0000	15-0	Reserved	
0xFFE1	0x0000	15-0	Reserved	
0xFFE2	0x0000	15-0	Reserved	
0xFFE3	0x0000	15-0	Reserved	
0xFFE4	0x0000	15-0	Reserved	
0xFFE5	0x0000	15-0	Reserved	
0xFFE6	0x0000	15-0	Reserved	
0xFFE7	0x0000	15-0	Reserved	
0xFFE8	0x0000	15-0	Reserved	
0xFFE9	0x0000	15-0	Reserved	
0xFFEA	0x0000	15-0	Reserved	
0xFFEB	0x0000	15-0	Reserved	
0xFFEC	0x0000	15-0	Reserved	
0xFFED	0x0000	15-0	Reserved	
0xFFEE	0x0000	15-0	Reserved	
0xFFEF	0x0000	15-0	Reserved	
0xFFF0	0x0000	15-0	Reserved	
0xFFF1	0x0000	15-0	Reserved	
0xFFF2	0x0000	15-0	Reserved	
0xFFF3	0x0000	15-0	Reserved	
0xFFF4	0x0000	15-0	Reserved	
0xFFF5	0x0000	15-0	Reserved	
0xFFF6	0x0000	15-0	Reserved	
0xFFF7	0x0000	15-0	Reserved	
0xFFF8	0x0000	15-0	Reserved	
0xFFF9	0x0000	15-0	Reserved	
0xFFFA	0x0000	15-0	Reserved	
0xFFFB	0x0000	15-0	Reserved	
0xFFFC	0x0000	15-0	Reserved	
0xFFFD	0x0000	15-0	Reserved	
0xFFFE	0x0000	15-0	Reserved	
0xFFFF	0x0000	15-0	Reserved	

4.1.3 Design clocks – inst2_*pll_top*

Module *pll_top* Figure 3 provides clocks for LMS7002 RX and TX digital interfaces. This module contains two dynamically reconfigurable MMCMs. Clock frequency and phase relationship can be changed while FPGA is in user mode.

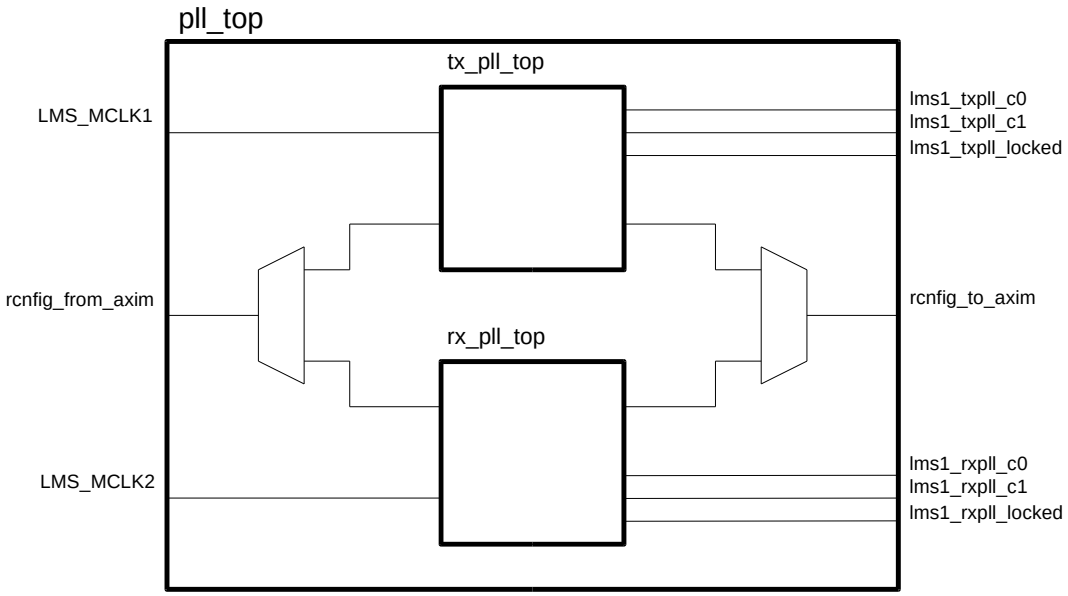


Figure 3: pll_top module block diagram

Table 27: pll_top generics

Generic name	Type	Value	Description
INTENDED_DEVICE_FAMILY	STRING	""	Device family
N_PLL	integer	2	Number of PLLs
LMS1_TXPLL_DRCT_C0_NDLY	integer	1	Direct TX clock delay (Obsolete)
LMS1_TXPLL_DRCT_C1_NDLY	integer	2	Direct TX clock delay (Obsolete)
LMS1_RXPLL_DRCT_C0_NDLY	integer	1	Direct RX clock delay (Obsolete)
LMS1_RXPLL_DRCT_C1_NDLY	integer	2	Direct RX clock delay (Obsolete)

Table 28: pll_top ports

Port name	Direction	Type	Description
lms1_smpl_cmp_en	out	std_logic	Sample compare enable
lms1_smpl_cmp_done	in	std_logic	Sample compare done
lms1_smpl_cmp_error	in	std_logic	Sample compare error
lms1_smpl_cmp_cnt	out	std_logic_vector(15 downto 0)	Number of samples to compare
rcnfg_axi_clk	in	std_logic	AXI bus reconfiguration clock
rcnfg_axi_reset_n	in	std_logic	AXI bus active low reset
rcnfg_from_axim	in	t_FROM_AXIM_32x32	AXI bus inputs
rcnfg_to_axim	out	t_TO_AXIM_32x32	AXI bus outputs
rcnfg_sel	in	std_logic_vector(3 downto 0)	Reconfiguration select
to_pllcfg	out	t_TO_PLLCFG	Output signals PLLCFG registers
from_pllcfg	in	t_FROM_PLLCFG	Input signals from PLLCFG registers
lms1_txpll	in	Virtual bus	LMS#1 TX PLL ports
lms1_rxpll	in	Virtual bus	LMS#1 RX PLL ports

Table 29: lms1_txpll virtual bus

Port name	Direction	Type	Description
lms1_txpll_inclk	in	std_logic	TXPLL input clock
lms1_txpll_reconfig_clk	in	std_logic	TXPLL reconfiguration clock
lms1_txpll_logic_reset_n	in	std_logic	TXPLL logic active low reset
lms1_txpll_clk_ena	in	std_logic_vector(1 downto 0)	TXPLL clock enable
lms1_txpll_drct_clk_en	in	std_logic_vector(1 downto 0)	TXPLL direct clock enable (Obsolete)
lms1_txpll_c0	out	std_logic	TXPLL clock output c0
lms1_txpll_c1	out	std_logic	TXPLL clock output c1
lms1_txpll_locked	out	std_logic	TXPLL locked output

Table 30: lms1_rxpll virtual bus

Port name	Direction	Type	Description
lms1_rxpll_inclk	in	std_logic	RXPLL input clock
lms1_rxpll_reconfig_clk	in	std_logic	RXPLL reconfiguration clock
lms1_rxpll_logic_reset_n	in	std_logic	RXPLL logic active low reset
lms1_rxpll_clk_ena	in	std_logic_vector(1 downto 0)	RXPLL clock enable
lms1_rxpll_drct_clk_en	in	std_logic_vector(1 downto 0)	RXPLL direct clock enable (Obsolete)
lms1_rxpll_c0	out	std_logic	RXPLL clock output c0
lms1_rxpll_c1	out	std_logic	RXPLL clock output c1
lms1_rxpll_locked	out	std_logics	RXPLL locked output

4.1.4 Receive and transmit interface – inst3_*rxtx_top*

Main function of *rxtx_top* module is to provide timestamp synchronization between receive and transmit packets. Module *tx_path_top* receives packets trough *tx_in_pct* bus, decodes them and writes decoded IQ samples to external FIFO buffer trough *tx_smpl_fifo* bus when decoded sample number is equal to number received from *rx_path_top* module.

Module *rx_path_top* receives IQ samples trough *rx_smpl_fifo* bus, packs them into packets together with sample number counter (64bit wide) and forwards packets to *rx_pct_fifo* bus.

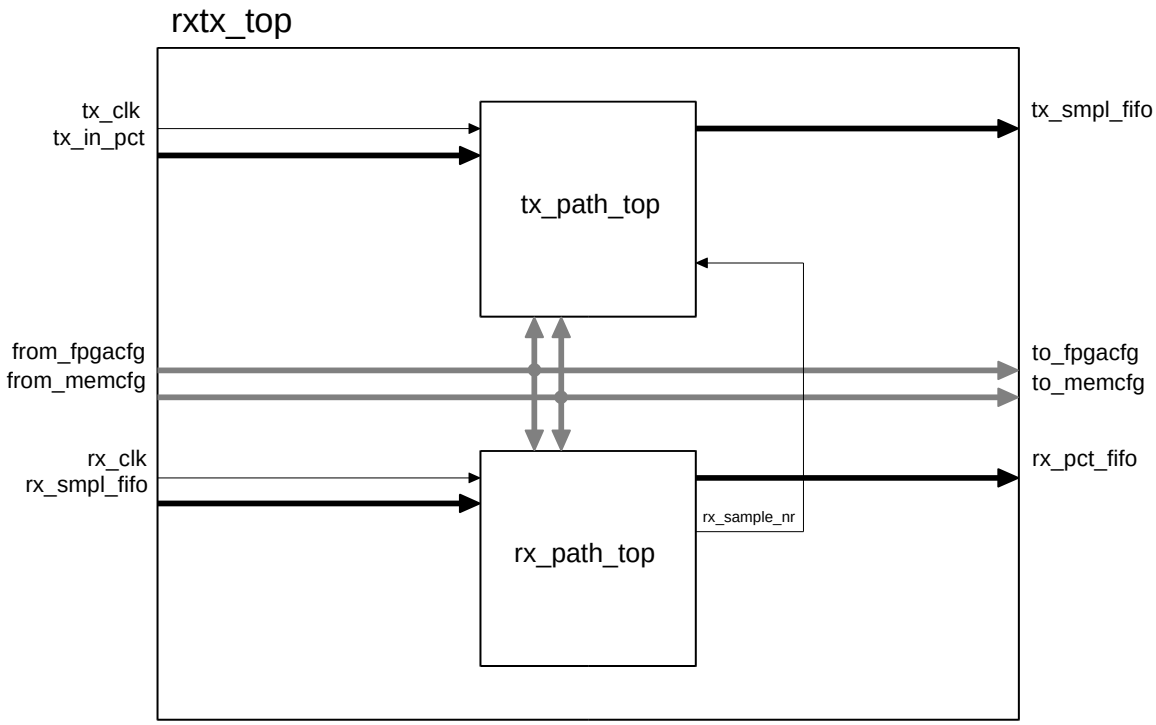


Figure 4: rxtx_top block diagram

Generics of *rxtx_top* module can be found in Table 31 and port descriptions in Table 32.

Table 31: rxtx_top generic parameters

Generic name	Type	Value	Description
index	integer	1	Module index, if more than one module exists in design
DEV_FAMILY	string	""	Device family
TX_EN	boolean	true	TX path enable
RX_EN	boolean	true	RX path enable
TX_IQ_WIDTH	integer	12	TX DIQ sample width
TX_N_BUFF	integer	4	2,4 valid values
TX_IN_PCT_SIZE	integer	4096	TX packet size in bytes
TX_IN_PCT_HDR_SIZE	integer	16	TX packet header size
TX_IN_PCT_DATA_W	integer	128	TX packet data width
TX_IN_PCT_RDUSEDW_W	integer	11	TX in packet read used word width
TX_OUT_PCT_DATA_W	integer	64	TX out packet data width
TX_SMPL_FIFO_WRUSEDW_W	integer	9	TX sample FIFO write used word size
TX_HIGHSPEED_BUS	boolean	false	Double TX sample FIFO size to
RX_DATABUS_WIDTH	integer	64	RX data width
RX_IQ_WIDTH	integer	12	RX IQ sample width
RX_INVERT_INPUT_CLOCKS	string	"OFF"	RX input clock inversion
RX_SMPL_BUFF_RDUSEDW_W	integer	11	RX sample bus read used words width in bits
RX_PCT_BUFF_WRUSEDW_W	integer	12	RX packet buffer write used words width in bits
RX_DISABLE_14B_SAMPLEPACKING	boolean	false	Disable RX sample packing in 14b words

Table 32: rxtx_top module ports

Port name	Direction	Type	Description
sys_clk	in	std_logic	System clock, free running
from_fpgacfg	in	t_FROM_FPGACFG	Signals from FPGACFG registers
to_fpgacfg	out	t_TO_FPGACFG	Signals to FPGACFG register
tx_clk	in	std_logic	TX path clock
tx_clk_reset_n	in	std_logic	TX path active low reset
tx_pct_loss_flg	out	std_logic	TX packet loss indication flag
rx_clk	in	std_logic	RX path clock
rx_clk_reset_n	in	std_logic	RX path active low reset
rx_smpl_nr_cnt_en	in	std_logic	RX path sample number count enable
to_memcfg	out	t_TO_MEMCFG	Signals to MEMCFG registers
from_memcfg	in	t_FROM_MEMCFG	Signals from MEMCFG registers
ext_rx_en	in	std_logic	External RX path enable
tx_dma_en	in	std_logic	TX DMA enble flag
tx_smpl_fifo	out	Virtual bus	
tx_in_pct	in	Virtual bus	
rx_smpl_fifo	in	Virtual bus	
rx_pct_fifo	out	Virtual bus	

Table 33: tx_smpl_fifo virtual bus

Port name	Direction	Type	Description
tx_smpl_fifo_wrreq	out	std_logic	Write request (data valid)
tx_smpl_fifo_wrfull	in	std_logic	Write full
tx_smpl_fifo_wrusedw	in	std_logic_vector(TX_SMPL_FIFO_WRUSEDW_W-1 downto 0)	Write used words
tx_smpl_fifo_data	out	std_logic_vector(127 downto 0)	Write data

Table 34: tx_in_pct virtual bus

Port name	Direction	Type	Description
tx_in_pct_reset_n_req	out	std_logic	Reset request
tx_in_pct_rdreq	out	std_logic	Read request
tx_in_pct_data	in	std_logic_vector(TX_IN_PCT_DATA_W-1 downto 0)	Packet data
tx_in_pct_rdempty	in	std_logic	Read empty
tx_in_pct_rdusedw	in	std_logic_vector(TX_IN_PCT_RDUSEDW_W-1 downto 0)	Read used words

Table 35: rx_smpl_fifo virtual bus

Port name	Direction	Type	Description
rx_smpl_fifo_wrreq	in	std_logic	Write request
rx_smpl_fifo_data	in	std_logic_vector(RX_IQ_WIDTH*4-1 downto 0)	Data
rx_smpl_fifo_wrfull	out	std_logic	Write full

Table 36: rx_pct_fifo virtual bus

Port name	Direction	Type	Description
rx_pct_fifo_aclrn_req	out	std_logic	Asynchronous clear request
rx_pct_fifo_wusedw	in	std_logic_vector(RX_PCT_BUFF_WRUSEDW_W-1 downto 0)	Write used words
rx_pct_fifo_wrreq	out	std_logic	Write request (data valid)
rx_pct_fifo_wdata	out	std_logic_vector(RX_DATABUS_WIDTH-1 downto 0)	Data

4.1.4.1 Receive interface – *rx_path_top*

Once *rx_path_top* Figure 5 is enabled it starts continuously packing IQ samples into 4kB packets. For packet structure see Stream protocol document.

Packets are written to 16kB FIFO buffer to maintain continuous data flow in short periods when host cannot accept data. If host halts data transfer for longer time period and four packets are buffered into 16kB buffer, FIFO full condition arises and other packets are dropped. When host starts to receive data after FIFO full condition, host should expect to receive those four buffered packets.

Module *rx_path_top* provides two 64bit sample counters. One is for TX logic – *tx_path_top*. TX logic uses this counter to synchronize transmitted LMS_DQ1 samples with received LMS_DIQ2 samples. Other is used for LMS_DIQ2 samples packing into 4kB packets.

When *rx_path_top* is enabled it starts to collect IQ samples from *rx_smpl_fifo* bus, collected samples are written to FIFO buffer and each write enables *smpl_cnt:inst4* module to increase its counter value. This means that counter value increases in same continuous rate as IQ sample rate.

Module *smpl_cnt:inst3* is used for LMS_DIQ2 samples packing into 4kB packets. Module *data2packets* reads IQ samples in bursts from FIFO buffer, each read enables *smpl_cnt:inst3* module to increase its counter value. One read burst fills one 4kB packet and there are some idle cycles between bursts.

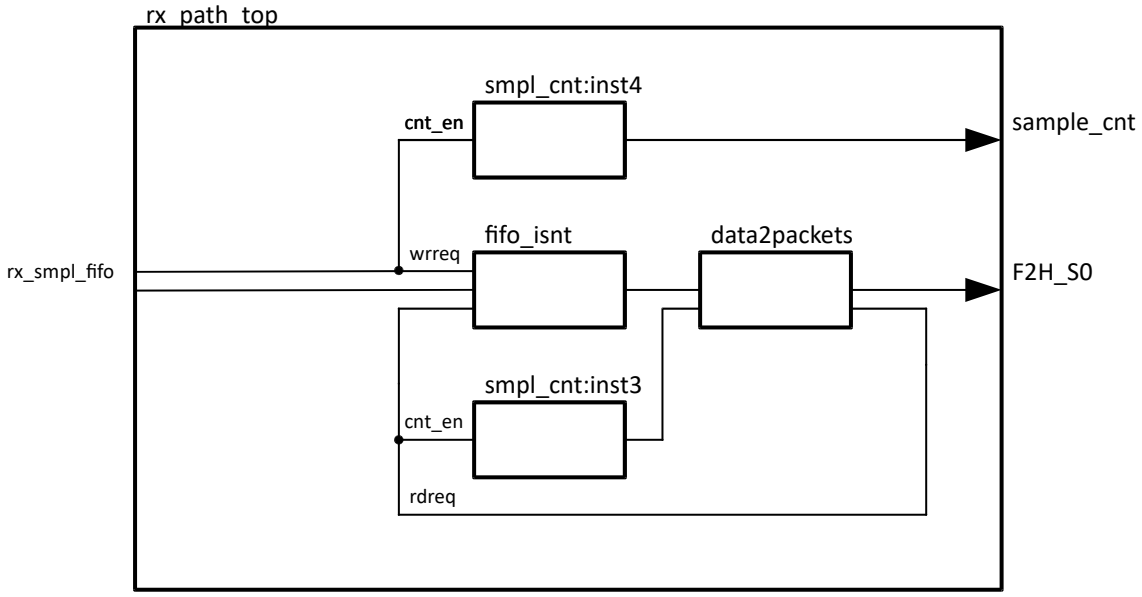


Figure 5: rx_path_top block diagram

Table 37: rx_path_top instance description

Instance	Description
fifo_inst	FIFO buffer for storing samples.
data2packets	Module for packing IQ samples to 4kB packets.
smpl_cnt:inst3	Sample counter for tx_path_top.
smpl_cnt:inst4	Sample counter for data2packets module.

4.1.4.2 Transmit interface – t*x_path_top*

Transmit module *tx_path_top* reads IQ samples from EP03 FIFO buffer packed in 4kB packets. Packet header (see Stream protocol document) contains sample number (or so-called time stamp) at which packet should be transmitted.

By using sample numbers from *rx_path_top* and received sample numbers in packet header transmitted IQ samples can be synchronized with received IQ samples.
Module *p2d_wr_fsm* separates packet header and payload. Packet payload is written into one of four 4kB FIFO buffers located in *packets2data* module and packet header is stored in *p2d_rd* module. This module can work in two modes:

- **Synchronization enabled** - module compares received sample number from packet header and sample number from *rx_path_top*. When sample number from received packet is equal to sample number of *rx_path_top* module (this means that it is time to send TX packet), read process begins and IQ samples are transmitted to LMS_DIQ1 interface. When sample number from received packet is greater than sample number of *rx_path_top* module (this means that received packet should be sent after some time) *p2d_rd* waits until those sample numbers will be equal. When sample number from received packet is less than sample number of *rx_path_top* module (this means that packet arrived too late) corresponding FIFO buffer is cleared.
- ***Synchronization disabled*** – module does not compare sample numbers and every received packet is transmitted to LMS_DIQ1 interface.

Block diagram can be found in **Figure 6** and instance description in **Table 38.**

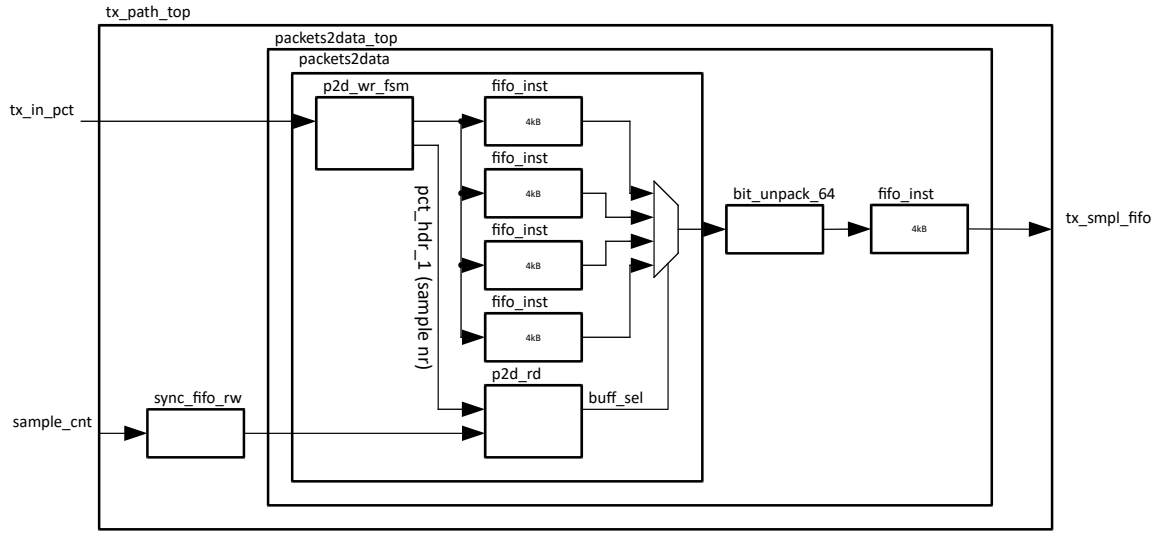


Figure 6: tx_path_top block diagram

Table 38: tx_path_top instance description

Instance	Description
packets2data_top	Wrapper file
packets2data	Wrapper file
p2d_wr_fsm	Module reads packets from EP03 buffer and places to one of the 4kB FIFO buffers in increasing order and stores corresponding sample number from packet header.
p2d_rd	Module checks one of the FIFO buffers if it is filled with samples in increasing order. When buffer is ready depending on received sample number from packet header and sample number from rx_path_top module buffer can be cleared or IQ sample reading begins.
fifo_inst	FIFO buffer
sync_fifo_rw	Dual clock FIFO buffer for clock domain crossing.
bit_unpack_64	Depending on mode selection samples are unpacked (see Stream protocol document).

4.1.5 *LMS7002 digital interface – inst4_lms7002_top*

Module lms7002_top implements various digital LimeLigth interface modes that LMS7002 IC can operate. Supported modes:

- TRXIQ MIMO DDR
- TRXIQ SISO DDR
- TRXIQ Pulse
- TRXIQ SDR

Instance *tx_path_top* is used for transmitting samples trough LMS_DIQ1 port and *diq2fifo* instance for receiving samples from LMS_DIQ2 port.

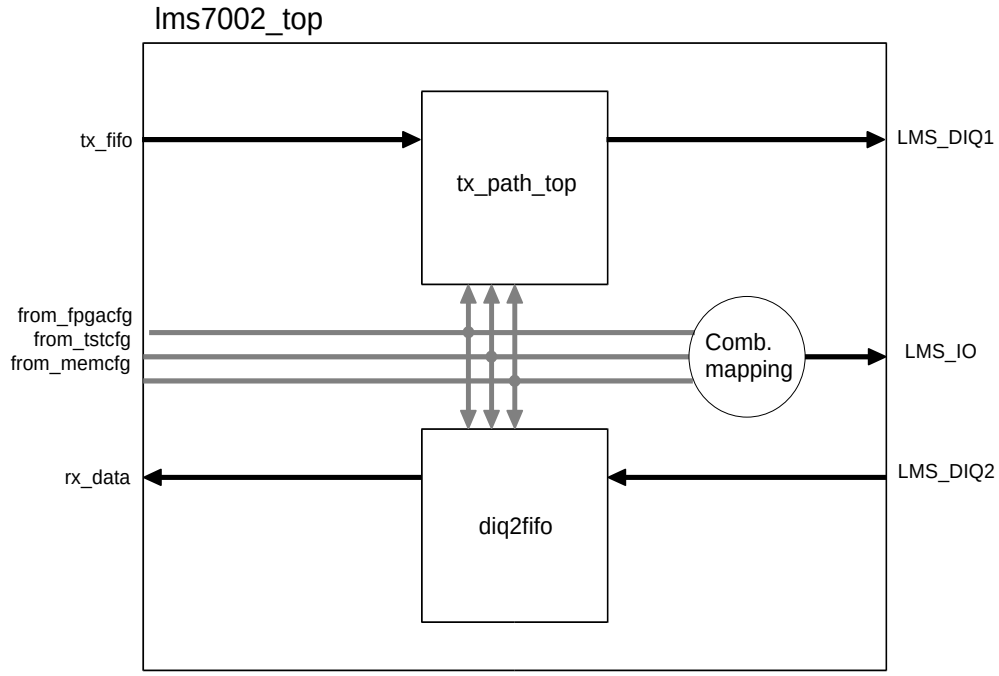


Figure 7: lms7002_top block diagram

Generic parameter description can be found in Table 39 and port description in Table 40.

Table 39: lms7002_top generic parameters

Generic name	Type	Value	Description
g_DEV_FAMILY	string	""	Device family
g_IQ_WIDTH	integer	12	IQ bus width
g_INV_INPUT_CLK	string	"ON"	Input clock inversion
g_TX_SMPL_FIFO_0_WRUSEDW	integer	9	TX sample FIFO 0 write used words width
g_TX_SMPL_FIFO_0_DATAW	integer	128	TX sample FIFO 0 data width
g_TX_SMPL_FIFO_1_WRUSEDW	integer	9	TX sample FIFO 1 write used words width (Not in use)
g_TX_SMPL_FIFO_1_DATAW	integer	128	TX sample FIFO 1 write used words width (Not in use)

Table 40: lms7002_top port description

Port name	Direction	Type	Description
from_fpgacfg	in	t_FROM_FPGACFG	Signals from FPGACFG registers
from_tstcfg	in	t_FROM_TSTCFG	Signals from TSTCFG registers
from_memcfg	in	t_FROM_MEMCFG	Signals from MEMCFG registers
mem_reset_n	in	std_logic	Memory module reset
tx_ant_en	out	std_logic	TX antenna enable flag
rx_reset_n	in	std_logic	RX interface active low reset
rx_diq_h	out	std_logic_vector(g_IQ_WIDTH downto 0)	Output of Direct capture on rising edge of DIQ2 port
rx_diq_l	out	std_logic_vector(g_IQ_WIDTH downto 0)	Output of Direct capture on falling edge of DIQ2 port
rx_data_valid	out	std_logic	Received data from DIQ2 port valid signal
rx_data	out	std_logic_vector(g_IQ_WIDTH*4-1 downto 0)	Received data from DIQ2 port
rx_smpl_cmp_start	in	std_logic	RX sample compare start
rx_smpl_cmp_length	in	std_logic_vector(15 downto 0)	RX sample compare length
rx_smpl_cmp_done	out	std_logic	RX sample compare done flag

Port name	Direction	Type	Description
rx_smpl_cmp_err	out	std_logic	RX sample compare error flag
rx_smpl_cnt_en	out	std_logic	RX sample counter enable
LMS_PORT1	out	Virtual bus	interface
LMS_PORT2	in	Virtual bus	interface
LMS_MISC	out	Virtual bus	LMS miscellaneous control ports
tx_fifo0	in	Virtual bus	Internal TX ports
tx_fifo1	in	Virtual bus	(not in use)

Table 41: LMS_PORT1 virtual port

Port name	Direction	Type	Description
MCLK1	in	std_logic	TX interface clock
MCLK1_2x	in	std_logic	(Not in use)
FCLK1	out	std_logic	TX interface feedback clock
DIQ1	out	std_logic_vector(g_IQ_WIDTH-1 downto 0)	DIQ1 data bus
ENABLE_IQSEL1	out	std_logic	IQ select flag for DIQ1 data
TXNRX1	out	std_logic	LMS_PORT1 direction select

Table 42: LMS_PORT2 virtual port

Port name	Direction	Type	Description
MCLK2	in	std_logic	RX interface clock
FCLK2	out	std_logic	RX interface feedback clock
DIQ2	in	std_logic_vector(g_IQ_WIDTH-1 downto 0)	DIQ2 data bus
ENABLE_IQSEL2	in	std_logic	IQ select flag for DIQ2 data
TXNRX2	out	std_logic	LMS_PORT2 direction select

Table 43: LMS_MISC virtual bus

Port name	Direction	Type	Description
RESET	out	std_logic	LMS hardware reset, active low
TXEN	out	std_logic	TX hard power off
RXEN	out	std_logic	RX hard power off
CORE_LDO_EN	out	std_logic	LMS internal LDO enable control

Table 44: tx_fifo_0 virtual bus

Port name	Direction	Type	Description
tx_reset_n	in	std_logic	TX interface active low reset
tx_fifo_0_wrclk	in	std_logic	TX FIFO write clock
tx_fifo_0_reset_n	in	std_logic	TX FIFO Reset
tx_fifo_0_wrreq	in	std_logic	TX FIFO write request
tx_fifo_0_data	in	std_logic_vector(g_TX_SMPL_FIFO_0_DATAW-1 downto 0)	TX FIFO data
tx_fifo_0_wrfull	out	std_logic	TX FIFO write full
tx_fifo_0_wrusedw	out	std_logic_vector(g_TX_SMPL_FIFO_0_WRUSEDW-1 downto 0)	TX FIFO write used words

4.1.6 RF control logic – inst5_tdd_control

Module tdd_control implements some simple logic to control RF switches automatically when TDD operation is required.

If automatic control is disabled, RF switches are set to the mode that is specified in register 0x000A (refer to Table 19 FPGACFG module registers (0x000A-0x0010)). If automatic control is enabled, RX switches are set to “No connection” when data is being transmitted from FPGA to LMS and to the value specified in 0x000A register when no data is being transmitted. TX switches are set to the value in 0x000A register when data is transmitted and to the inverse value when it is not.

This module contains only logic and no lower modules, so no block diagram is provides. Port descriptions are in Table 45.

Table 45: tdd_control port description

Port name	Direction	Type	Description
MANUAL_VALUE	in	std_logic	Value to be used for TDD_OUT if AUTO_ENABLE = '0'
AUTO_ENABLE	in	std_logic	Enable automatic TDD_OUT control
AUTO_IN	in	std_logic	RF status. '1': transmitting, '0': not transmitting
AUTO_INVERT	in	std_logic	Invert TDD_OUT signal
RX_RF_SW_IN	in	std_logic_vector(1 downto 0)	RF RX switch configuration to be used when receiving
TX_RF_SW_IN	in	std_logic	RF TX switch configuration to be used when transmitting
RF_SW_AUTO_ENANBLE	in	std_logic	Enable automatic TDD RF RX/TX switch control
TDD_OUT	out	std_logic	Output signal for external TDD modules
RX_RF_SW_OUT	out	std_logic_vector(1 downto 0)	RF RX switch control output
TX_RF_SW_OUT	out	std_logic	RF TX switch control output

4.1.7 Test modules – inst6_tst_top

Module tst_top implements basic test modules for clocks FPGA_CLK (refered to as lms_tx_clk in code), SYS_CLK, as well as for the GNSS module.

Clock tests work in the following way:
SYS_CLK – tested with no reference clock, the test can only answer if the clock is active or not.
FPGA_CLK – tested by counting the number of cycles in 0.1 seconds by using SYS_CLK as a reference.

GNSS is tested by sending a test command and waiting for an appropriate answer.

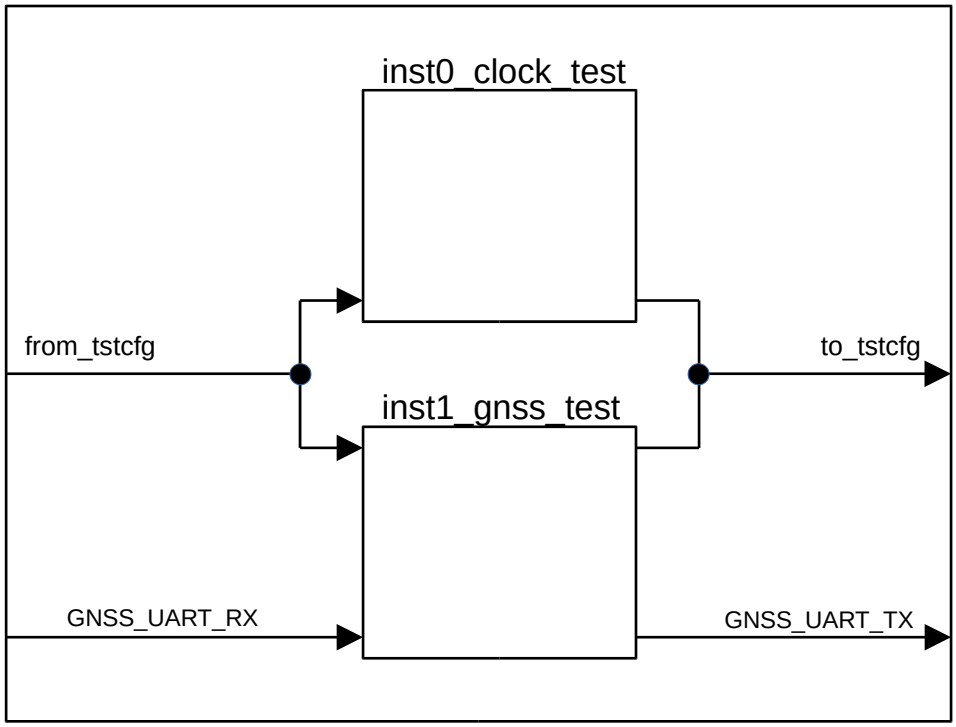


Figure 8: tst_top block diagram

Table 46: tst_top port description

Port name	Direction	Type	Description
SYS_CLK	in	std_logic	System clock
RESET_N	in	std_logic	Reset, active low
LMS_TX_CLK	in	std_logic	LMS7002 tx clock
GNSS_UART_RX	in	std_logic	GNSS chip's uart rx signal
GNSS_UART_TX	out	std_logic	GNSS chip's uart rx signal
TO_TSTCFG	out	t_TO_TSTCFG	Configuration register bus Module -> Registers

Port name	Direction	Type	Description
FROM_TSTCFG	in	t_FROM_TSTCFG	Configuration register bus Registers -> Module

4.2 Clock network

Figure 6 shows dataflow between main modules and clocking scheme. More details about design clocks can be found in Table 47.

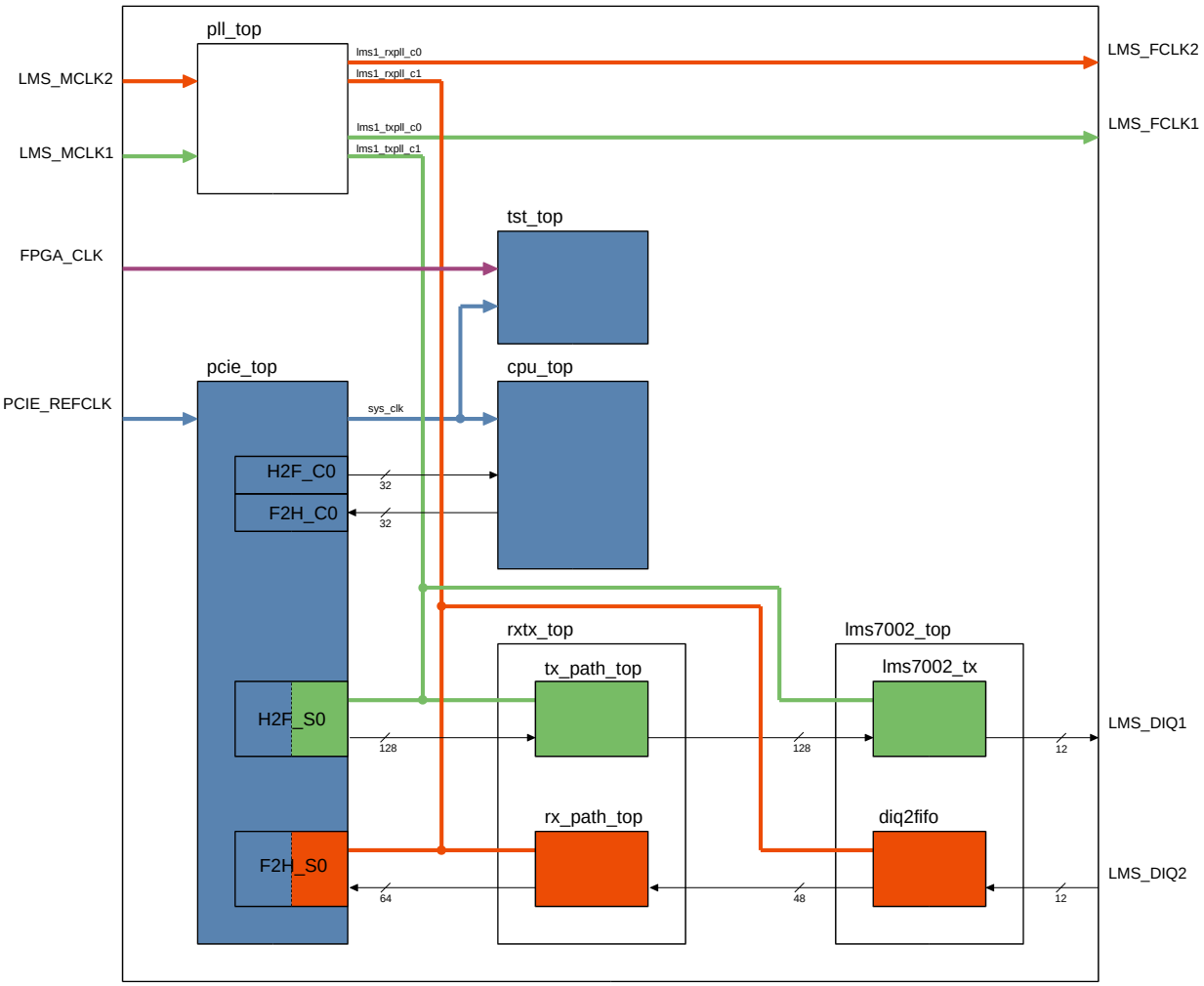


Figure 9: Clock network diagram

Table 47: Design clock details

Clock name	Frequency, MHz	Description
LMS_MCLK1	Configurable	Reference clock for LMS_DIQ1 data bus
LMS_MCLK2	Configurable	Reference clock for LMS_DIQ2 data bus.
LMS_FCLK1	Configurable	Feedback clock LMS7002 IC captures LMS_DIQ1 data using this clock
LMS_FCLK2	Configurable	Not used.
FPGA_CLK	26	Onboard XO clock
PCIE_REFCLK	100	Reference clock from PCIe host.
sys_clk	125	PCIe system clock synchronous to PCIE_REFCLK