

Zynq UltraScale+ RFSoc ZCU208 and ZCU216 RF Data Converter Evaluation Tool

User Guide

UG1433 (v1.2) October 27, 2021

Xilinx is creating an environment where employees, customers, and partners feel welcome and included. To that end, we're removing non-inclusive language from our products and related collateral. We've launched an internal initiative to remove language that could exclude people or reinforce historical biases, including terms embedded in our software and IPs. You may still find examples of non-inclusive language in our older products as we work to make these changes and align with evolving industry standards. Follow this [link](#) for more information.



Revision History

The following table shows the revision history for this document.

Section	Revision Summary
10/27/2021 Version 1.2	
Chapter 3: Hardware Design	Added time division duplex (TDD) power up/down block information.
TDD Control Block	Added new section.
Appendix B: Command List	Added new commands.
06/23/2020 Version 1.1	
General updates.	Added ZCU208 board information throughout.
RF-ADC DDR	Added information on dual RF-ADC tiles.
User Space Components	Added note to refer to appendix.
Software Build	Revised the steps for making the SD card images and copying files to the SD card.
Appendix A: Software Design Notes — Multimaster Access of CLK104 on TCA9548 Multiplexer	Added appendix.
Appendix B: Command List	Removed SetSignalDetector and GetSignalDetector commands. Revised description for RfclkWriteReg. Added GetExtParentClkList, GetExtParentClkConfig, SetMMCMFin, GetMMCMFin, and GetMTS_Setup commands.
03/23/2020 Version 1.0	
Initial release.	N/A

Table of Contents

Revision History	2
Chapter 1: Introduction	5
Chapter 2: Overview	6
Installation.....	7
Chapter 3: Hardware Design	8
Block RAM Generation and Capture.....	9
RF-DAC DDR.....	10
RF-ADC DDR.....	10
Clocking Scheme.....	12
TDD Control Block.....	13
Chapter 4: Software Design and Build	18
Software Architecture.....	18
User Space Components.....	19
Kernel Space Components.....	20
Software Build.....	21
Chapter 5: Protocol Specifications	23
Protocol Overview.....	23
Protocol Rules.....	24
Socket Interface.....	25
Appendix A: Software Design Notes — Multimaster Access of CLK104 on TCA9548 Multiplexer	26
Appendix B: Command List	27
Appendix C: Additional Resources and Legal Notices	34
Xilinx Resources.....	34
Documentation Navigator and Design Hubs.....	34



References.....	34
Please Read: Important Legal Notices.....	35

Introduction

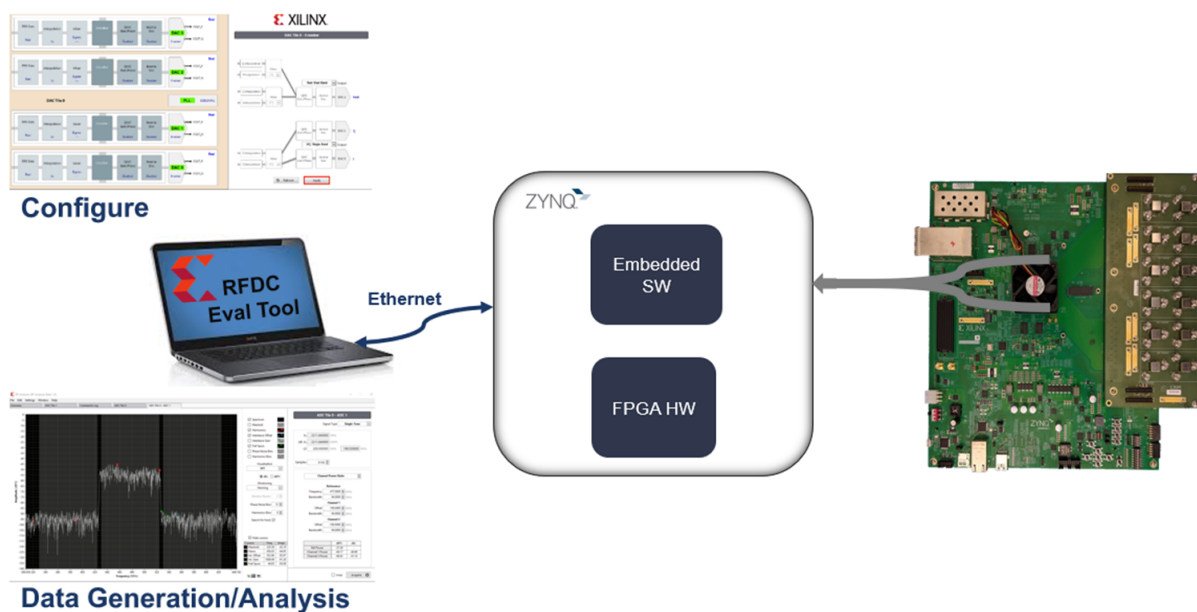
The Zynq[®] UltraScale+[™] RFSoc RF Data Converter (DC) Evaluation Tool and the ZCU208 and ZCU216 Evaluation Kits are the ideal combination of evaluation software and test platform to facilitate cutting edge application development. The evaluation tool can be used to jump start RF-class analog designs and to demonstrate the capabilities and excellent performance of the Gen 3 RF data converters. Advantages of using this tool include acceleration of the product design cycle, reduction of product go-to-market expense, and quicker revenue realization.

Overview

The evaluation tool enables control of the ZCU208 and ZCU216 RF DC IPs (see *Zynq UltraScale+ RFSoc RF Data Converter LogiCORE IP Product Guide (PG269)*) and associated designs from a host computer. The tool allows the exploration of RF configurations, generation and capture of RF data, and observation of key RF metrics.

The following figure shows the evaluation tool and test platform overview.

Figure 1: Evaluation Tool and Test Platform Overview



X24062-062220

The tool is built from these components:

- *ZCU208 Evaluation Board User Guide (UG1410)*
- *ZCU216 Evaluation Board User Guide (UG1390)*
- FPGA hardware design (see [Chapter 3: Hardware Design](#))
- FPGA embedded software design (see [Chapter 4: Software Design and Build](#))
- GUI (see *RF Data Converter Interface User Guide (UG1309)*)

- Protocol used to communicate between the host computer (GUI) and the software design (see [Chapter 5: Protocol Specifications](#))

Installation

See the [RF DC Evaluation Tool for ZCU208 Board Quick Start](#) or the [RF DC Evaluation Tool for ZCU216 Board Quick Start](#) website for installation and folder structure information.

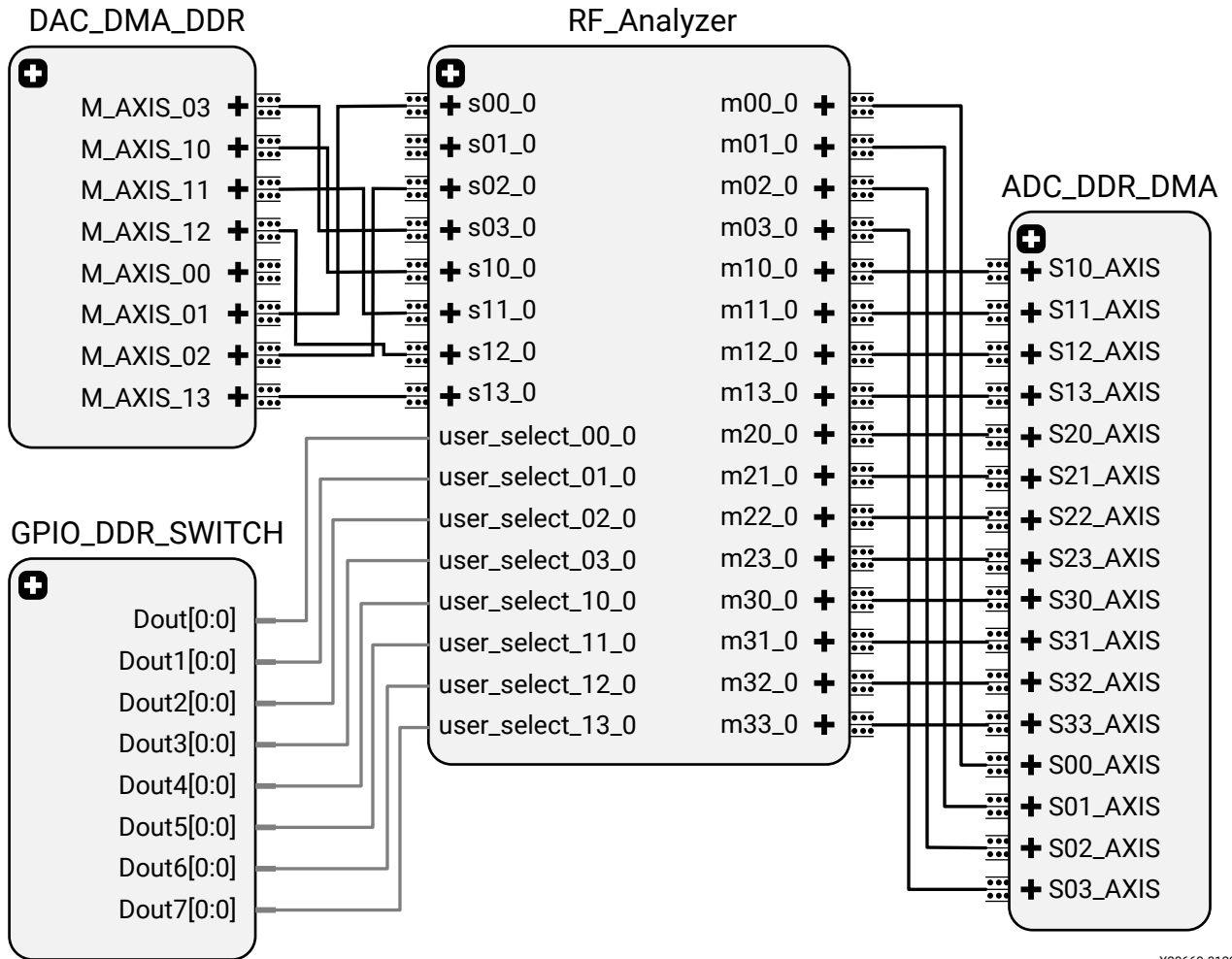
Hardware Design

The Vivado[®] Design Suite project used to build this programmable logic (PL) design is located in the install directory in the `p1` folder. The hardware design architecture is based on the RF analyzer architecture (see *Zynq UltraScale+ RFSoc RF Data Converter LogiCORE IP Product Guide (PG269)*). Notable additions to this architecture include:

- MicroBlaze[™] is replaced by the Cortex[®]-A53
- PL DDR4 support, including DMA, AXI4-Stream broadcaster and switch, and GPIO control
- Clocking scheme to support DDR4 and multi-tile synchronization
- I2C connection to control external clocks (CLK104, see *ZCU208 Evaluation Board User Guide (UG1410)* and *ZCU216 Evaluation Board User Guide (UG1390)*)
- Time division duplex (TDD) power up/down block to control the TDD real-time signal (RTS) pins of ADCs and DACs

The following figure shows the high-level hardware architecture.

Figure 2: High-level Hardware Architecture



X23663-012320

Block RAM Generation and Capture

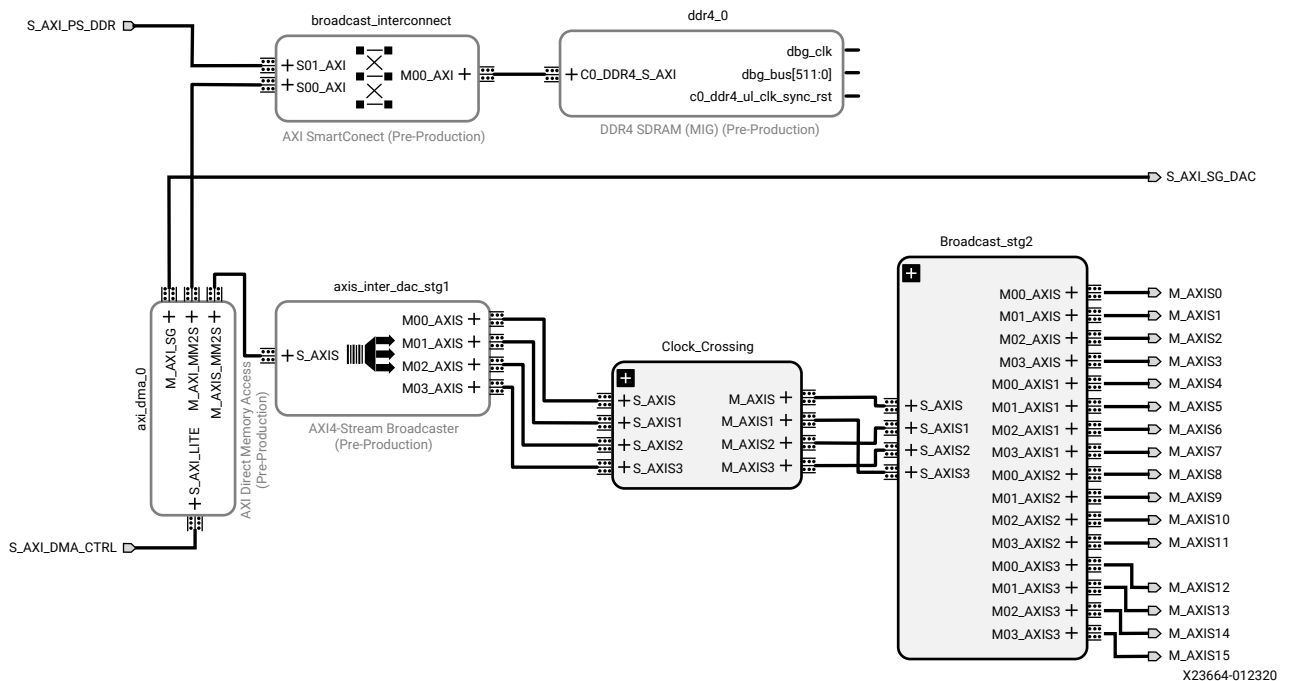
The block RAM generation and capture are described in the "RF Analyzer" section in the *Zynq UltraScale+ RFSoc RF Data Converter LogiCORE IP Product Guide* (PG269).

RF-DAC DDR

A DDR4 memory controller (see *UltraScale Architecture-Based FPGAs Memory IP LogiCORE IP Product Guide (PG150)*) is instantiated to control the external DDR4 memory. It connects to an AXI DMA controller (see *AXI DMA LogiCORE IP Product Guide (PG021)*). The waveform loaded into the DDR4 memory is then broadcast to the RF-DAC tiles selected by the AXI GPIO (see *AXI GPIO LogiCORE IP Product Guide (PG144)*). The GPIO connects to the `user_select` input of the block RAM generation and effectively bypasses this block when `user_select` is High.

For crossing clock domains, the broadcasting is done in two steps. The first step is part of the DDR clock domain and the second step is part of the RF-DAC tile clock domain. The RF-DAC DDR block architecture is illustrated in the following figure.

Figure 3: RF-DAC DDR Block Architecture

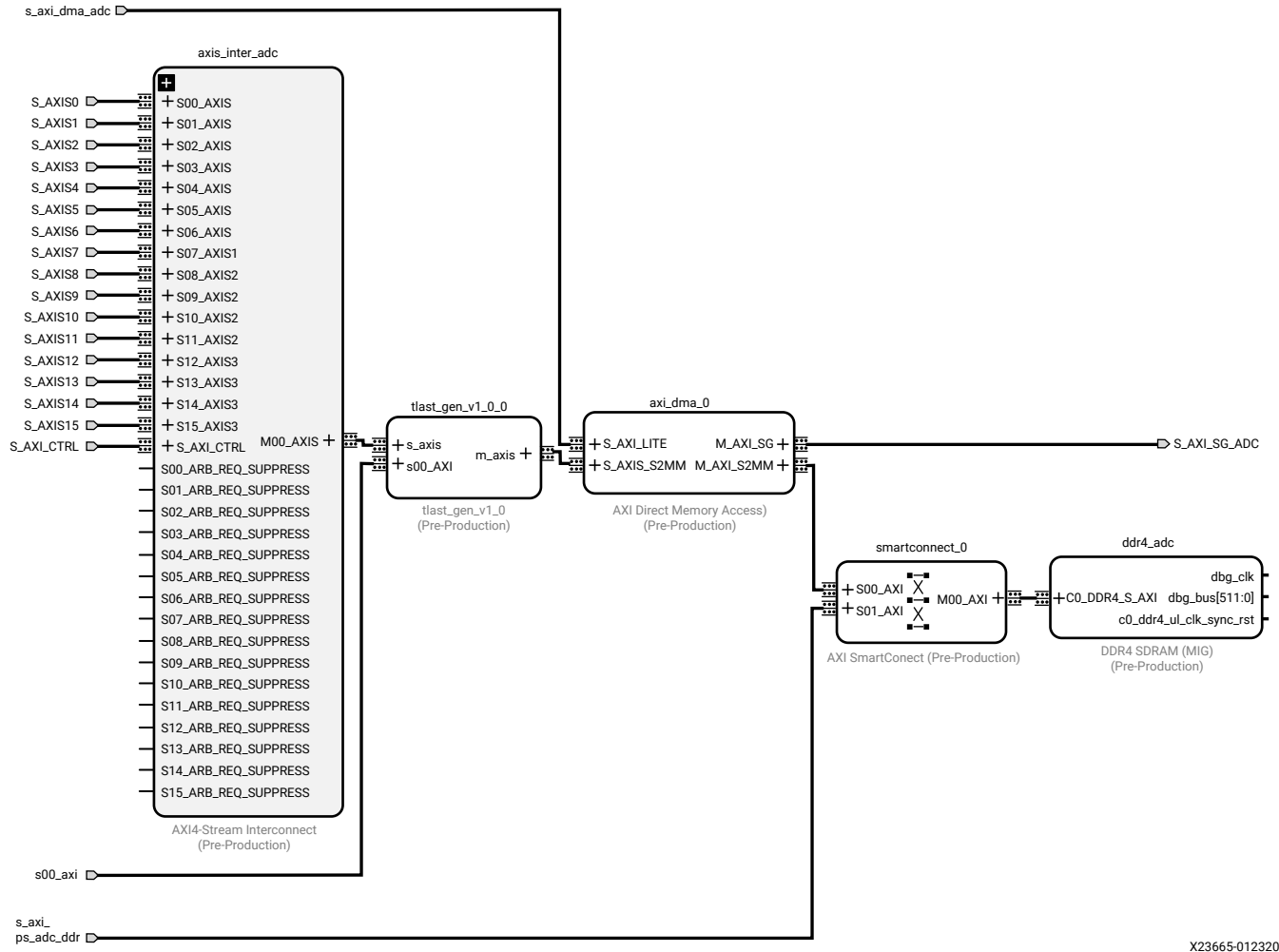


RF-ADC DDR

A DDR4 memory controller (see *UltraScale Architecture-Based FPGAs Memory IP LogiCORE IP Product Guide (PG150)*) is instantiated to control the external DDR4 memory. The RF-ADC output is duplicated on the output of the RF analyzer block RAM capture block. An AXI4-Stream interconnect is then used as a switch to select which RF-ADC waveform is fed into the DMA and captured into the DDR memory. To allow the software to control the DMA accurately, a TLAST

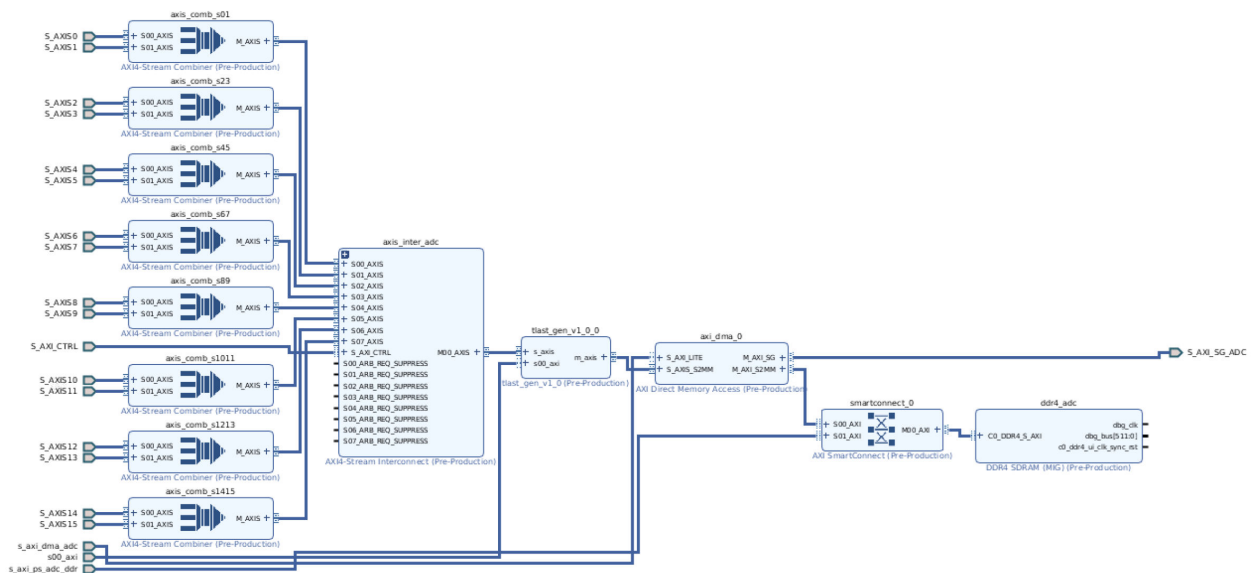
block is created. This block generates a TLAST pulse so that the DMA generates an interrupt after the correct number of samples. Because the dual RF-ADC tiles have two AXI4-Stream outputs in I/Q mode, an AXI4-Stream combiner is used to write both streams in the DDR memory. Consequently, the real mode cannot be used to capture the dual RF-ADC tiles in the DDR. The RF-ADC DDR block architecture is shown in the following figures.

Figure 4: RF-ADC DDR Block Architecture for Quad RF-ADC Tiles



X23665-012320

Figure 5: RF-ADC DDR Block Architecture for Dual RF-ADC Tiles



X23874-042220

Clocking Scheme

The clocking scheme is built to support both individual tile clocks and independent RF-ADC or RF-DAC multi-tile synchronization (MTS). Consequently, two of the typical clocking schemes from the *Zynq UltraScale+ RFSoc RF Data Converter LogiCORE IP Product Guide (PG269)* are merged. The first clocking scheme uses the output of each tile to connect to a mixed-mode clock manager (MMCM) or phase-locked loop (PLL), which gives an independent clock per tile. The second clocking scheme supports MTS, takes its input from a fabric pin, and generates an output for the tiles selected by the MTS functions. BUFGMUX and MMCM dual input allows merging these two clocking schemes. For the RF-ADC, a mixture of MMCM and PLL are used. The RF-ADC clocking scheme figure shows a representation for two tiles. The RF-DAC clocking scheme figure shows a representation for two tiles. For the RF-DAC, only the MMCM is used. The path used in the MTS case is shown in red.

Figure 6: RF-ADC Clocking Scheme

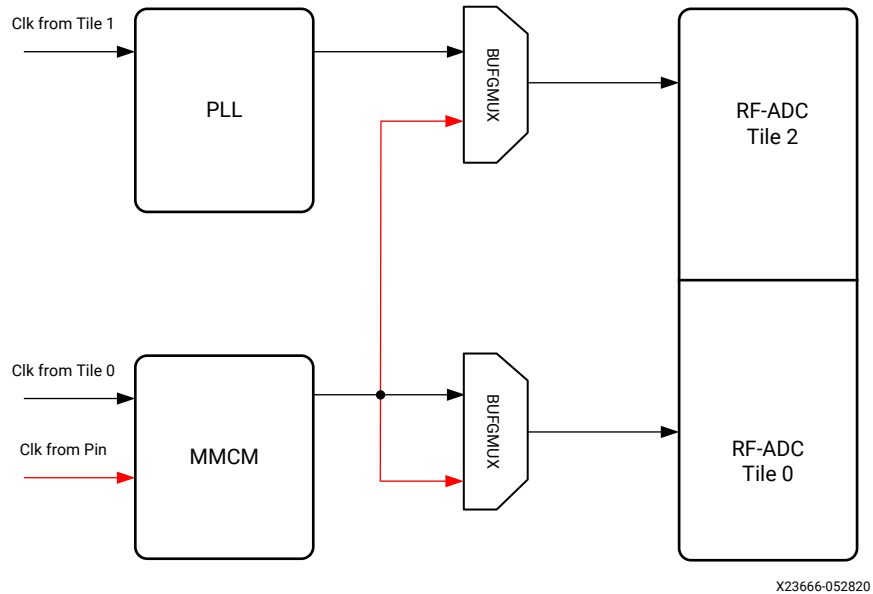
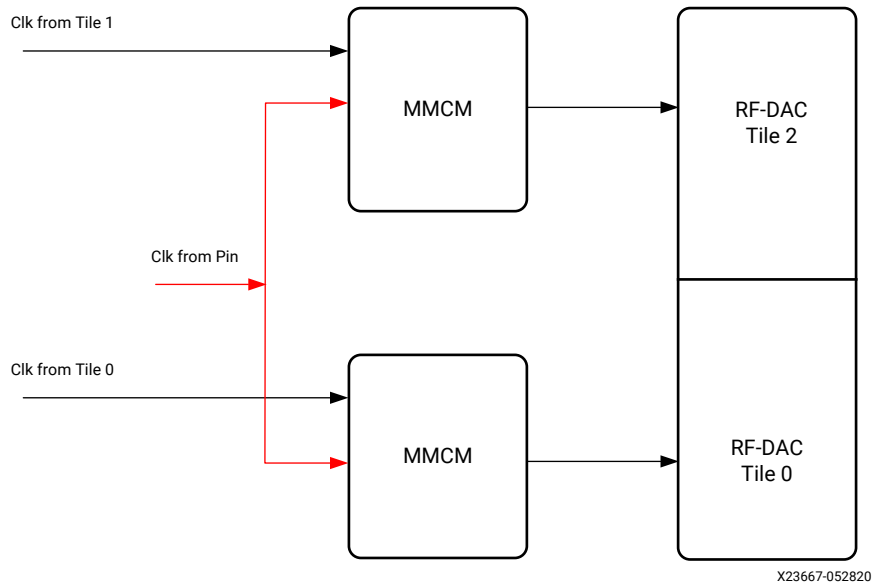


Figure 7: RF-DAC Clocking Scheme



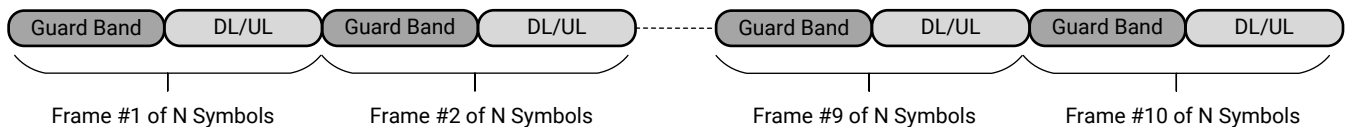
TDD Control Block

The TDD control hardware block demonstrates the TDD power up/down of each ADC and DAC channel. The block can be used as a reference on how to drive the real-time signal TDD pins, power and performance measurements, and debugging. The block has the following features:

- A set of two counters determine the number of symbols per frame and the number of frames
- The number of symbols is programmable, and the number of frames is always 10
- A map of downlink/uplink determine which TDD pins are asserted (DAC or ADC)
- A programmable guard band is possible
- Two trigger modes for the ADC: immediate or programmable
- The ADC programmable trigger can select the frame and symbol
- A programmable delay can be added on the trigger, available in each tile

The following figure shows a representation of the frame and symbol composition.

Figure 8: Frames and Symbols Representation

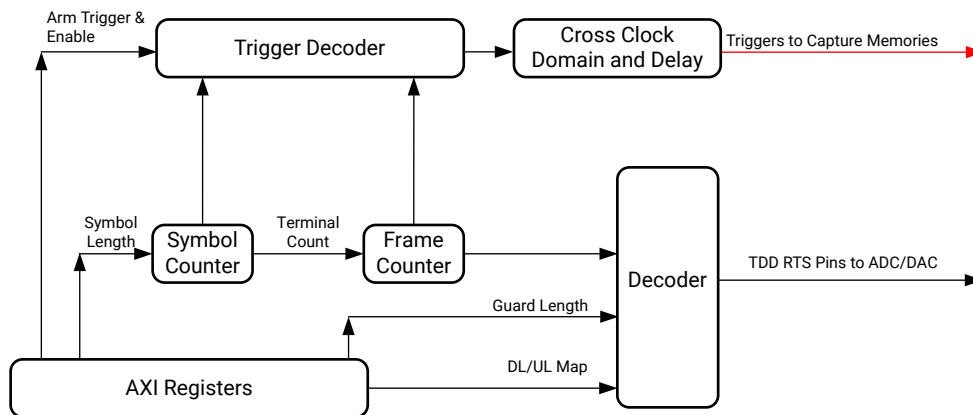


X25676-083021

Block Diagram

The TDD control block diagram is shown in the following figure.

Figure 9: TDD Control Block



X25660-081321

Ports

The following table lists the TDD control block ports.

Table 1: Ports

Port Name	I/O	Clock	Description
S00_AXI	AXI port	S00_AXI_ACLK	AXI control port
DACx_CLK	In		DAC AXI4-Stream clock
ADCx_CLK	In		ADC AXI4-Stream clock
DACxx_tdd_mode	Out	S00_AXI_ACLK	Connect to DACxx TDD mode pin
ADCxx_tdd_mode	Out	S00_AXI_ACLK	Connect to ADCxx TDD mode pin
Hw_trigger_en_x	Out	ADC AXI4-Stream clock	Enable external trigger on capture block
Trigger_x	Out	ADC AXI4-Stream clock	External trigger
Trigger_ext	Out	Dac0_clk	External trigger for DAC

Register Map

The TDD control block register map is listed in the following table.

Table 2: Register Map

Address	Description
0x0	DAC TDD mode pin, bit n: DAC channel n
0x4	ADC TDD mode pin, bit n: ADC channel n
0x8	Bit 0: reset, others: reserved
0xC	Bit 0 to 3: ADC hw_trigger_en control
0x10	Symbol to trigger on
0x14	Frame to trigger on
0x18	Arm the trigger
0x30	Tile 0 trigger delay
0x34	Tile 1 trigger delay
0x38	Tile 2 trigger delay
0x3C	Tile 3 trigger delay
0x44	Slot length (unused)
0x48	Guard band length
0x4C	Symbol length
0x50	Symbol type

Example Usage

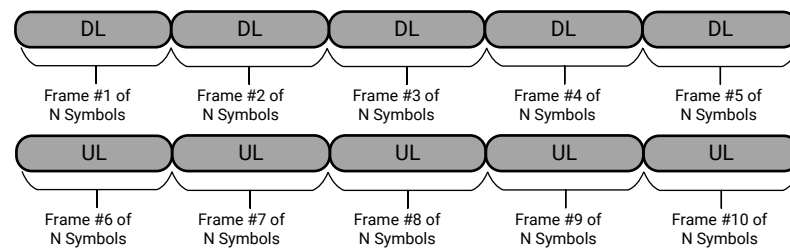
The software commands are documented in [Appendix B: Command List](#). The commands can be entered into the Command log window of the RF Evaluation tool GUI (*RF Data Converter Interface User Guide* (UG1309)). The following examples show the command sequences for DAC and ADC, respectively.

DAC Example

1. Write 1 into the DAC/ADC TDD mode pin register.
 - SetTDDRTSPinCtrl 1 0x0001 enables the TDD on DAC tile 0 channel 0.
 - SetTDDRTDPinCtrl DAC/ADC Channel_mask to 1 for DAC and 0 for ADC. Channel mask bit 0 = DAC0, bit 1 DAC1, etc.
2. Write the symbol length, guard band length, and symbol type registers.
 - SetTDDRTSSlot 0 1000 0x01f: guard band 0, symbol length 1000, half DL, half UL.
3. At this stage, the channels selected in step 1 are operating in TDD mode.
4. An immediate trigger can be issued.
5. To stop the TDD RTS pin assertion/deassertion, SetTDDRTSPinCtrl 10x0000

The following figure shows a representation of the frames in the DAC example.

Figure 10: DAC Example Frames and Symbols Representation



X25677-083021

Dual ADC Example

1. Write the symbol length, guard band length, and symbol type registers.
 - SetTDDRTSSlot 0 1000 0x01f: guard band 0, symbol length 1000, half DL, half UL.
2. Write 1 into the ADC TDD mode pin register.
 - SetTDDRTSPinCtrl 0 0x0030 enables the TDD on ADC tile 1 channel 0 for dual ADC.
 - As the ADC is a dual ADC, the two TDD RTS pins need to be driven High.
3. At this stage, the channels selected in step 1 are operating in TDD mode.
4. To set up an ADC trigger on a specific frame and symbol:
 - Write symbol and frame trigger registers.
 - SetTDDRTSTrigSlot 5 5.
 - SetTDDRTSTrigSlot symbol frame.

- Write 1 in the corresponding bit of the enable register.
 - SetTDDRTSEnables 0x0002.
 - This controls the hw_trigger_en pins. In this example, it enables the trigger on tile 1.
- Write 1 in the trigger register.
 - SetTDDRTSTrig 1.
 - This controls the hw_trigger pins.

Software Design and Build

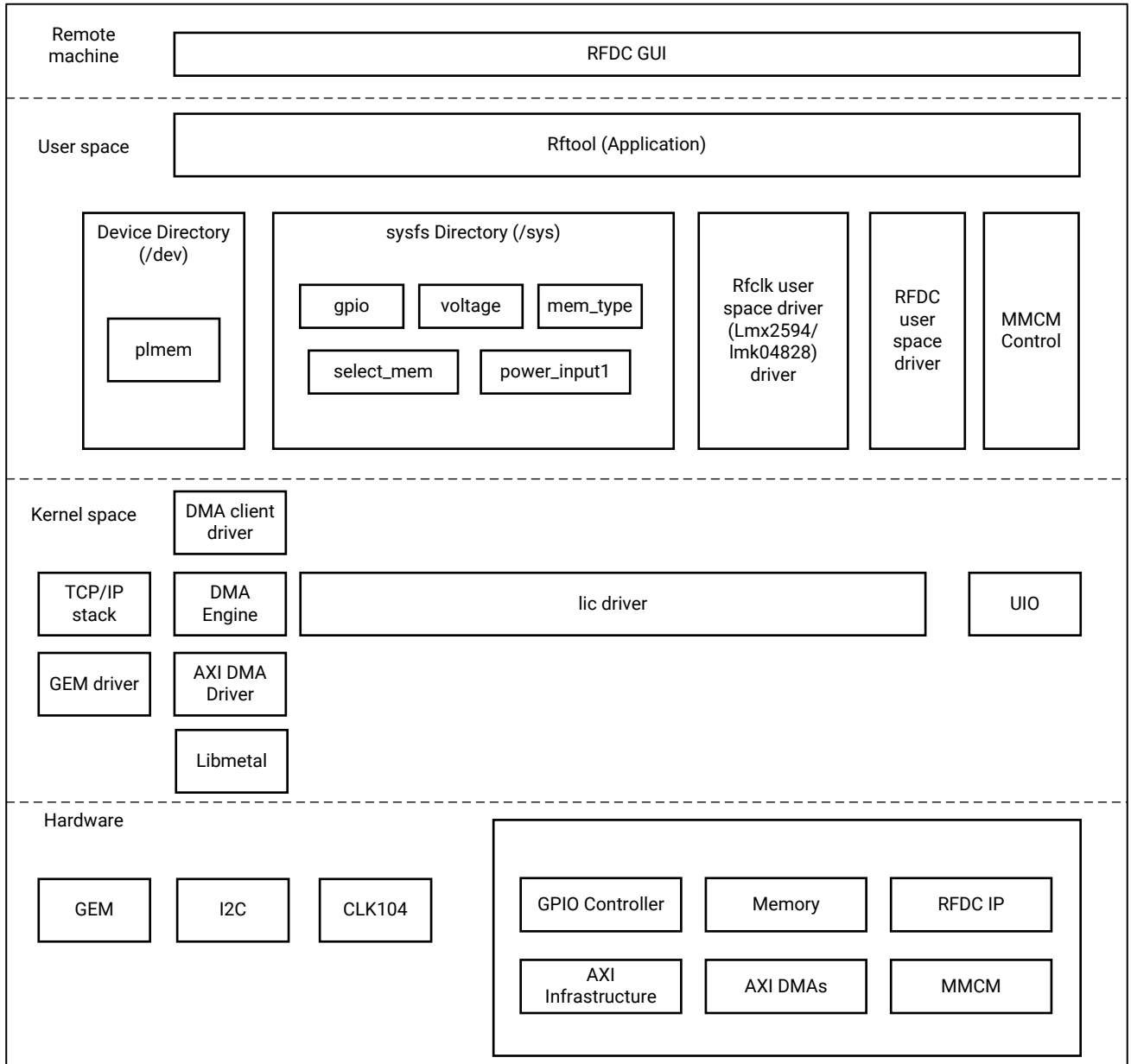
This chapter describes the software platform running on the application processing unit (APU), which is logically further subdivided into user and kernel space components (see [Figure 11: APU Linux Software Platform](#)). The Linux application `rftool` in the user space receives the command over the Ethernet and performs the appropriate action. This application is the main interface to the GUI and uses a string-based communication protocol described in [Chapter 5: Protocol Specifications](#). Device drivers expose a systematic interface to control hardware. These interfaces are used by the user space application to control hardware.

All configurations of the RF-ADCs and RF-DACs are done by the Linux application `rftool` running on the processing system (PS). The application supports all the configuration options supported by the GUI. The GUI sends configuration details via the Ethernet interface to the Linux application, which are implemented by their respective software APIs.

Software Architecture

The following figure shows the APU Linux software platform, which includes two logical software flows: the control path and the datapath. The control path and datapath are implemented using two different TCP sockets. The components in the software flows are implemented in the user space and the kernel space.

Figure 11: APU Linux Software Platform



X23668-031920

User Space Components

The Linux user space components used are:

- The rftool is the Linux application that receives commands over the Ethernet from the PC GUI and performs appropriate actions.

- RFDC user space drivers provide APIs for communication with the RFDC hardware.
- RFCLK user space drivers provide APIs for communications with external CLK104 daughter boards.
Note: See [Appendix A: Software Design Notes – Multimaster Access of CLK104 on TCA9548 Multiplexer](#).
- DMA client driver interface `/dev/pl_mem` is used to allocate a buffer from the PL DDR, and is also used to trigger a DMA transaction from the user space.

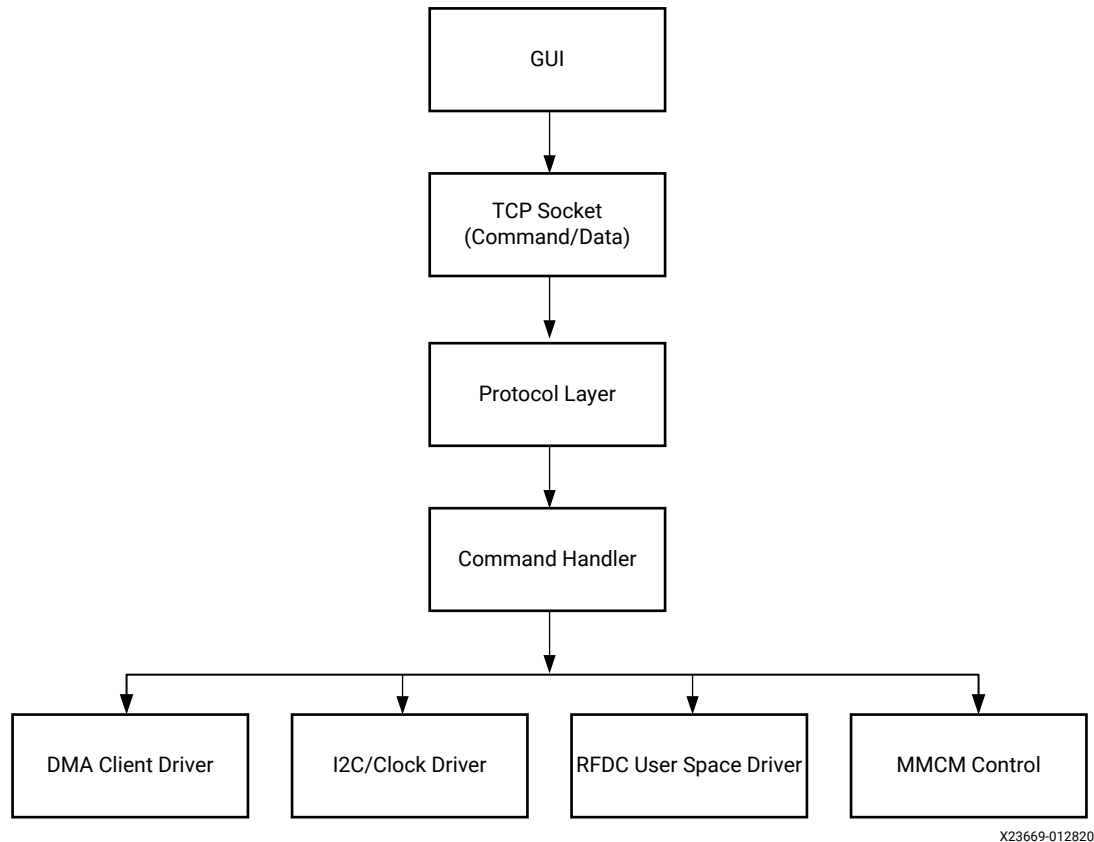
Kernel Space Components

The Linux kernel space components used are:

- The GEM driver provides the interface to send and receive packets over the GEM Ethernet controller.
- The TCP/IP stack provides the interface to create a transmission control protocol (TCP) socket. It also provides the interface to send and receive data over the TCP socket. The stack uses the GEM driver to send and receive packets from a network interface card (NIC).
- The DMA client driver provides the interface for a user application to trigger DMA transactions.
- The AXI DMA driver and DMA engine provide a driver interface to the AXI DMA. The DMA client driver uses APIs for this DMA engine. The DMA engine uses the AXI DMA driver to control hardware.
- The libmetal driver is a kernel space driver that allows the execution of user space drivers.
- Userspace I/O (UIO) is a Linux framework used to export a hardware space to a user space. It is used by the RFDC user space driver libmetal to configure RFDC hardware.
- MMCM control is reconfiguring the MMCM/PLL depending on tile clock requirement.

The following figure shows the application execution flow.

Figure 12: Application Execution Flow



Software Build

To build the software, the board support package (BSP) and build tools are needed. For the tools, see the 2019.2 version of the *PetaLinux Tools Documentation: PetaLinux Command Line Reference (UG1157)*.

1. Ensure the PetaLinux tools are available in the PATH environment variable.
2. To create the project, enter:

```
petalinux-create -t project -s <bsp> -n <project_name>
```

3. To build the project, enter:

```
cd <project_name>
petalinux-build -v
```

4. To make the images for the SD card, enter:

```
petalinux-package --force --boot --fsbl ./pre-built/linux/images/  
zynqmp_fsbl.elf --fpga ./images/linux/system.bit --pmufw ./pre-  
built/linux/images/pmufw.elf --u-boot ./images/linux/u-boot.elf
```

5. Copy these files to the SD card:

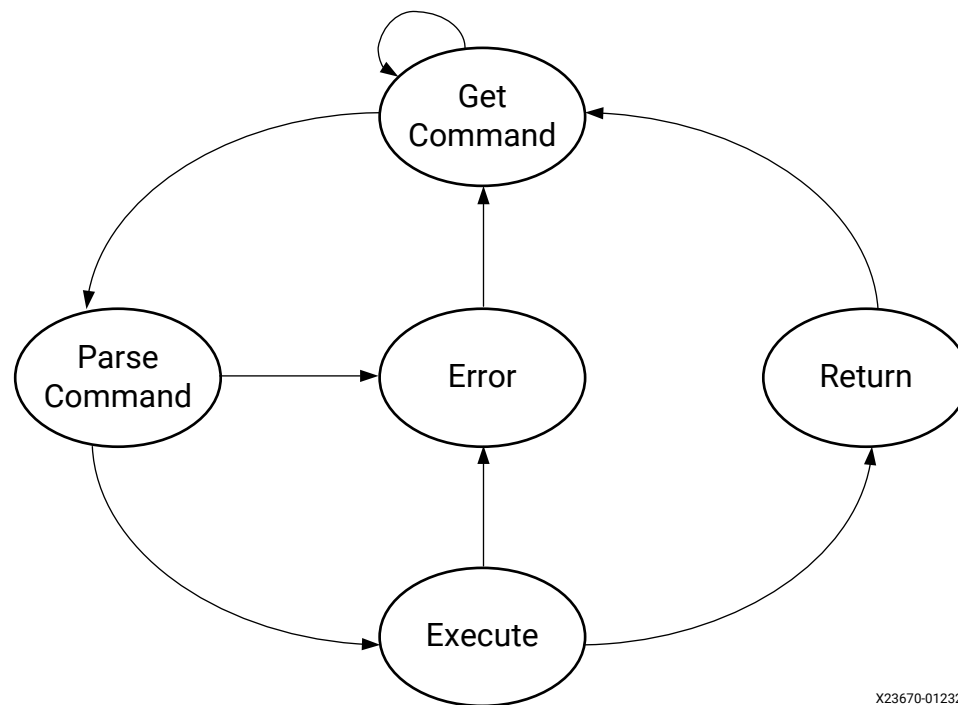
```
<project_name>/images/linux/BOOT.BIN  
<project_name>/images/linux/image.ub  
<project_name>/images/linux/boot.scr  
autostart.sh (copy from the Externals\image\216 folder)
```

Protocol Specifications

Protocol Overview

To communicate between the host and the design, a simple but robust protocol is used. This protocol is a string-based, space-separated command and response protocol. The state machine is represented in the following figure.

Figure 13: Finite State Machine Representing the Protocol



X23670-012320

Each command is sent in text (ASCII) format, parsed to check that the command is known and the number of arguments is correct, and then executed. The design returns the command name plus some parameters or the error type without any parameter if an error condition occurs. The host can get information on the error by using the `getlog` command, which reads the metal log buffer (see *Zynq UltraScale+ RFSoc RF Data Converter LogiCORE IP Product Guide (PG269)*).

The command can be split into these categories.

- RFDC commands are usually a direct translation of the RFDC driver with an addition of error checking when necessary
- PL design commands control the PL side of the design (MMCM, GPIO, capture and generation memories, etc.)
- Board control commands control some of the board components such as external clocks

Protocol Rules

The full list of commands is provided in [Appendix B: Command List](#).

The command syntax for input is:

- `CMD PARAM1 PARAM2 PARAM3` (with any number of parameters)
- Commands and parameters are separated by an ASCII space
- Commands and parameters are human readable, i.e., sent in ASCII format
- Receive end of line is `\n`

The output return format is:

- If an input command is not recognized, the parser errors out and sends `ERROR: CMD: Invalid Command\n`
- If an input command is recognized, the parser checks the number of arguments, and sends `ERROR: CMD: Invalid Number of Arguments\n`
- If execution succeeds, the result is returned `CMD param1 param2 ... paramX value1 value2 ... valueY\n`
- Commands that are not expected to return values return their name if execution succeeds: `[CMD]`
- CMD and values are separated by a space
- Receive end of line is `\n`
- Any log or messages from metal-log can be returned via the `getlog` command; the `\r\n` characters are replaced from any log messages with "|" and a single `\n` appended at the end

Socket Interface

The TCP/IP socket protocol is used for the datapath and control path. The GUI running on the host machine has TCP clients, and the TCP servers are running under the Linux application on the Zynq® UltraScale+™ RFSoc PS. The control path works on TCP port 8081 and the datapath works on TCP port 8082. The TCP socket uses a Linux TCP/IP stack, which uses the GEM Ethernet controller and driver for sending packets. Because the TCP socket is used, it is possible to run the GUI on any networked machines.

Software Design Notes — Multimaster Access of CLK104 on TCA9548 Multiplexer

The TCA9548 multiplexer is shared between the I2C masters MSP430 and APU on the Gen 3 RFSoc. The operation to select the CLK104 device on the multiplexer and read/write operation is completed in three steps in the I2C driver. When one master is accessing the device, the other master should not access it.

Command List

The commands supported by the evaluation tool are classified as basic commands, RFDC IP commands, board control commands, and PL data and control commands. The following table lists the supported commands.

Table 3: Command List

Command	Input Parameters	Output Parameters	Description
Basic Commands			
GUI_Title	None	Rftool Version	Rftool version, for example, (RFEvalTool v1.8).
RfdcVersion	None	Version Number(s)	RFdc API version.
Version	None	Rftool Version	Rftool version.
TermMode	Mode	None	Switch between UI mode (mode=0) and Interactive mode (mode=1). Interactive mode prints more status information to the terminal and should not be used with the evaluation tool GUI.
GetLog	None	Logged Strings	Uses metal_log to return any error, warning, or informational messages returned from the API or command interface.
JtagIdcode	None	Idcode	Return the device JTAG IDCODE.
Help	None	None	Print help on command interface.
Disconnect	None	None	
RFDC IP Commands			
GetBlockStatus	Type, Tile, Block	Type, Tile, Block, BlockStatus.SamplingFreq, BlockStatus.AnalogDataPathStatus, BlockStatus.DigitalDataPathStatus, BlockStatus.DataPathClocksStatus, BlockStatus.IsFIFOFlagsAsserted, BlockStatus.IsFIFOFlagsEnabled	Cf. PG269
GetCalFreeze	Tile, Block	Tile, Block, CalFreezePtr.CalFrozen, CalFreezePtr.FreezeCalibration, CalFreezePtr.DisableFreezePin	Cf. PG269
GetCalibrationMode	Tile, Block	Tile, Block, CalibrationMode	Cf. PG269

Table 3: Command List (cont'd)

Command	Input Parameters	Output Parameters	Description
GetClkDistribution	None	DistributionSettings.Distributions[i].SourceType DistributionSettings.Distributions[i].SourceTileId DistributionSettings.Distributions[i].EdgeTypes[0] DistributionSettings.Distributions[i].EdgeTypes[1] DistributionSettings.Distributions[i].EdgeTileIds[0] DistributionSettings.Distributions[i].EdgeTileIds[1] DistributionSettings.Distributions[i].DistRefClkFreq DistributionSettings.Distributions[i].DistributedClock DistributionSettings.Distributions[i].SampleRates[0][0] DistributionSettings.Distributions[i].SampleRates[0][1] DistributionSettings.Distributions[i].SampleRates[0][2] DistributionSettings.Distributions[i].SampleRates[0][3] DistributionSettings.Distributions[i].SampleRates[1][0] DistributionSettings.Distributions[i].SampleRates[1][1] DistributionSettings.Distributions[i].SampleRates[1][2] DistributionSettings.Distributions[i].SampleRates[1][3] DistributionSettings.Distributions[i].ShutdownMode DistributionSettings.Distributions[i].Info.MaxDelay DistributionSettings.Distributions[i].Info.MinDelay DistributionSettings.Distributions[i].Info.IsDelayBalanced DistributionSettings.Distributions[i].Info.Source DistributionSettings.Distributions[i].Info.UpperBound DistributionSettings.Distributions[i].Info.LowerBound DistributionSettings.Distributions[i].Info.ClkSettings[0][0].PLLEnable DistributionSettings.Distributions[i].Info.ClkSettings[0][1].PLLEnable DistributionSettings.Distributions[i].Info.ClkSettings[0][2].PLLEnable DistributionSettings.Distributions[i].Info.ClkSettings[0][3].PLLEnable DistributionSettings.Distributions[i].Info.ClkSettings[1][0].PLLEnable DistributionSettings.Distributions[i].Info.ClkSettings[1][1].PLLEnable DistributionSettings.Distributions[i].Info.ClkSettings[1][2].PLLEnable DistributionSettings.Distributions[i].Info.ClkSettings[1][3].PLLEnable With "i" the distribution number	Cf. PG269
GetClockSource	Type, Tile	Type, Tile, ClockSource	Cf. PG269
GetCoarseDelaySettings	Type, Tile, Block	Type, Tile, Block, Settings.CoarseDelay, Settings.EventSource	Cf. PG269
GetConnectedData	Type, Tile, Block	Type, Tile, Block, ConnectedIData, ConnectedQData	Cf. PG269
GetDACCompMode	Tile, Block	Tile, Block, EnablePtr	Cf. PG269
GetDataPathMode	Tile, Block	Tile, Block, Mode	Cf. PG269
GetDecimationFactor	Tile, Block	Tile, Block, DecimationFactor	Cf. PG269
GetDecimationFactorObs	Tile, Block	Tile, Block, DecimationFactor	Cf. PG269
GetDecoderMode	Tile, Block	Tile, Block, DecoderMode	Cf. PG269
GetDither	Tile, Block	Tile, Block, Mode	Cf. PG269
GetDSA	Tile, Block	Tile, Block, SettingsPtr.DisableRTS, SettingsPtr.Attenuation	Cf. PG269
GetFabClkOutDiv	Type, Tile	Type, Tile, FabClkDiv	Cf. PG269
GetFabRdVldWords	Type, Tile, Block	Type, Tile, Block, FabricDataRate	Cf. PG269
GetFabWrVldWords	Type, Tile, Block	Type, Tile, Block, FabricDataRate	Cf. PG269
GetFabWrVldWordsObs	Type, Tile, Block	Type, Tile, Block, FabricDataRate	Cf. PG269
GetFabRdVldWordsObs	Type, Tile, Block	Type, Tile, Block, FabricDataRate	Cf. PG269
GetFIFOStatus	Type, Tile	Type, Tile, Enable	Cf. PG269
GetFIFOStatusObs	Type, Tile	Type, Tile, Enable	Cf. PG269
GetIMRPassMode	Tile, Block	Tile, Block, Mode	Cf. PG269
GetInterpolationFactor	Tile, Block	Tile, Block, InterpolationFactor	Cf. PG269
GetIntrStatus	Type, Tile, Block	Type, Tile, Block, IntrStatus	Cf. PG269
GetInvSincFIR	Tile, Block	Tile, Block, Enable	Cf. PG269
GetIPStatus	None	IpStatus.DACTileStatus[Tile].IsEnabled, IpStatus.DACTileStatus[Tile].BlockStatusMask, IpStatus.DACTileStatus[Tile].TileState, IpStatus.DACTileStatus[Tile].PowerUpState, IpStatus.DACTileStatus[Tile].PLLState, IpStatus.ADCTileStatus[Tile].IsEnabled, IpStatus.ADCTileStatus[Tile].BlockStatusMask, IpStatus.ADCTileStatus[Tile].TileState, IpStatus.ADCTileStatus[Tile].PowerUpState, IpStatus.ADCTileStatus[Tile].PLLState, IpStatus.State	Cf. PG269
GetLinkCoupling	Tile, Block	Tile, Block, Mode	Cf. PG269
GetMixerSettings	Type, Tile, Block	Type, Tile, Block, Mixer_Settings.Freq, Mixer_Settings.PhaseOffset, Mixer_Settings.EventSource, Mixer_Settings.MixerType, Mixer_Settings.CoarseMixFreq, Mixer_Settings.MixerMode, Mixer_Settings.FineMixerScale	Cf. PG269
GetMTSEnable	Type, Tile	Type, Tile, Enable	Cf. PG269

Table 3: Command List (cont'd)

Command	Input Parameters	Output Parameters	Description
GetNyquistZone	Type, Tile, Block	Type, Tile, Block, NyquistZone	Cf. PG269
GetOutputCurr	Tile, Block	Tile, Block, OutputCurr	Cf. PG269
GetPLLConfig	Type, Tile	Type, Tile, PLLSettings.Enabled, PLLSettings.RefClkFreq, PLLSettings.SampleRate, PLLSettings.RefClkDivider, PLLSettings.FeedbackDivider, PLLSettings.OutputDivider	Cf. PG269
GetPLLLockStatus	Type, Tile	Type, Tile, LockStatus	Cf. PG269
GetQMCSettings	Type, Tile, Block	Type, Tile, Block, QMC_Settings.GainCorrectionFactor, QMC_Settings.PhaseCorrectionFactor, QMC_Settings.EnablePhase, QMC_Settings.EnableGain, QMC_Settings.OffsetCorrectionFactor, QMC_Settings.EventSource	Cf. PG269
GetThresholdSettings	Tile, Block	Tile, Block, ThresholdSettings.UpdateThreshold, ThresholdSettings.ThresholdMode[0], ThresholdSettings.ThresholdMode[1], ThresholdSettings.ThresholdAvgVal[0], ThresholdSettings.ThresholdAvgVal[1], ThresholdSettings.ThresholdUnderVal[0], ThresholdSettings.ThresholdUnderVal[1], ThresholdSettings.ThresholdOverVal[0], ThresholdSettings.ThresholdOverVal[1]	Cf. PG269
GetSignalDetector	Tile, Block	Tile, Block, Mode, TimeConstant, Flush, EnableIntegrator, Threshold, ThreshOnTriggerCnt, ThreshOffTriggerCnt, HysteresisEnable	Cf. PG269
GetDACPower	Board_id, Tile	Board_id, Tile, value_1, value_2, value_3, value_4, value_5	Cf. PG269
IntrClr	Type, Tile, Block, IntrMask	None	Cf. PG269
IntrDisable	Type, Tile, Block, IntrMask	None	Cf. PG269
IntrEnable	Type, Tile, Block, IntrMask	None	Cf. PG269
MTS_Sysref_Config	Enable, dac_tiles, adc_tiles	DAC_Sync_Config.Target_Latency, DAC_Sync_Config.Offset[Tile], DAC_Sync_Config.Latency[Tile], DAC_Sync_Config.Marker_Delay, DAC_Sync_Config.SysRef_Enable, ADC_Sync_Config.Target_Latency, ADC_Sync_Config.Offset[Tile], ADC_Sync_Config.Latency[Tile], ADC_Sync_Config.Marker_Delay, ADC_Sync_Config.SysRef_Enable	Cf. PG269
MultiConverter_Init	Type, Tile_ref	None	Cf. PG269
MultiConverter_Sync	Type, ADC_Sync_Config.Target_Latency/ DAC_Sync_Config.Target_Latency, Tile	Sync_config.Target_Latency, Sync_config.Offset[Tile], Sync_config.Latency[Tile], Sync_config.Marker_Delay, Sync_config.SysRef_Enable	Cf. PG269
Reset	Type, Tile	None	Cf. PG269
ResetNCOPhase	Type, Tile, Block	None	Cf. PG269
RF_ReadReg16	Offset	Value	Cf. PG269
RF_ReadReg32	Offset	Value	Cf. PG269
RF_WriteReg16	Offset, Value	None	Cf. PG269
RF_WriteReg32	Offset, Value	None	Cf. PG269
SetCalFreeze	Tile, Block, CalFreezePtr.FreezeCalibration, CalFreezePtr.DisableFreezePin	None	Cf. PG269
SetCalibrationMode	Tile, Block, Calibration Mode	None	Cf. PG269

Table 3: Command List (cont'd)

Command	Input Parameters	Output Parameters	Description
SetClkDistribution	Distribution_Settings.SourceTileId Distribution_Settings.SourceType Distribution_Settings.EdgeTileIds[0] Distribution_Settings.EdgeTileIds[1] Distribution_Settings.EdgeTypes[0] Distribution_Settings.EdgeTypes[1] Distribution_Settings.DistRefClkFreq Distribution_Settings.DistributedClock Distribution_Settings.SampleRates[0][0] Distribution_Settings.SampleRates[0][1] Distribution_Settings.SampleRates[0][2] Distribution_Settings.SampleRates[0][3] Distribution_Settings.EdgeTypes[0] Distribution_Settings.EdgeTypes[1] Distribution_Settings.DistRefClkFreq Distribution_Settings.DistributedClock Distribution_Settings.SampleRates[0][0] Distribution_Settings.SampleRates[0][1] Distribution_Settings.SampleRates[0][2] Distribution_Settings.SampleRates[0][3] Distribution_Settings.SampleRates[1][0] Distribution_Settings.SampleRates[1][1] Distribution_Settings.SampleRates[1][2] Distribution_Settings.SampleRates[1][3] Distribution_Settings.ShutdownMode	Distribution_Settings.SourceType Distribution_Settings.SourceTileId Distribution_Settings.EdgeTypes[0] Distribution_Settings.EdgeTypes[1] Distribution_Settings.EdgeTileIds[0] Distribution_Settings.EdgeTileIds[1] Distribution_Settings.DistRefClkFreq Distribution_Settings.DistributedClock Distribution_Settings.SampleRates[0][0] Distribution_Settings.SampleRates[0][1] Distribution_Settings.SampleRates[0][2] Distribution_Settings.SampleRates[0][3] Distribution_Settings.SampleRates[1][0] Distribution_Settings.SampleRates[1][1] Distribution_Settings.SampleRates[1][2] Distribution_Settings.SampleRates[1][3] Distribution_Settings.ShutdownMode Distribution_Settings.Info.MaxDelay Distribution_Settings.Info.MinDelay Distribution_Settings.Info.IsDelayBalanced Distribution_Settings.Info.Source Distribution_Settings.Info.UpperBound Distribution_Settings.Info.LowerBound Distribution_Settings.Info.ClkSettings[Type][Tile_Id].SourceType Distribution_Settings.Info.ClkSettings[Type][Tile_Id].SourceTile Distribution_Settings.Info.ClkSettings[Type][Tile_Id].PLLEnable Distribution_Settings.Info.ClkSettings[Type][Tile_Id].RefClkFreq Distribution_Settings.Info.ClkSettings[Type][Tile_Id].SampleRate Distribution_Settings.Info.ClkSettings[Type][Tile_Id].DivisionFactor Distribution_Settings.Info.ClkSettings[Type][Tile_Id].DistributedClock Distribution_Settings.Info.ClkSettings[Type][Tile_Id].Delay With Type 0 for ADC, Type 1 for DAC. Tile_Id from 0 to 3	Cf. PG269
SetCoarseDelaySettings	Type, Tile, Block, Settings.CoarseDelay, Settings.EventSource	None	Cf. PG269
SetDACPowerMode	Board_id, Tile, Block, op_current	Board_id, Tile, Block, op_current	Cf. PG269
SetDACVOP	Tile, Block, uACurrent	None	Cf. PG269
SetDataPathMode	Tile, Block, Mode	Tile, Block, Mode	Cf. PG269
SetDecimationFactor	Tile, Block, Decimation Factor	None	Cf. PG269
SetDecimationFactorObs	Tile, Block, Decimation Factor	None	Cf. PG269
SetDecoderMode	Tile, Block, Decoder Mode	None	Cf. PG269
SetDither	Tile, Block, Mode	Tile, Block, Mode	Cf. PG269
SetDSA	Tile, Block, DisableRTS, Attenuation	Tile, Block, Attenuation	Cf. PG269
SetFabClkOutDiv	Type, Tile, FabClkDiv	None	Cf. PG269
SetFabRdVldWords	Tile, Block, FabricDataRate	None	Cf. PG269
SetFabRdVldWordsObs	Tile, Block, FabricDataRate	None	Cf. PG269
SetFabWrVldWords	Tile, Block, FabricDataRate	None	Cf. PG269
SetIMRPassMode	Tile, Block, Mode	Tile, Block, Mode	Cf. PG269
SetInterpolationFactor	Tile, Block, InterpolationFactor	None	Cf. PG269
SetInvSincFIR	Tile, Block, Enable	None	Cf. PG269
SetMixerSettings	Type, Tile, Block, Mixer_Settings.Freq, Mixer_Settings.PhaseOffset, Mixer_Settings.EventSource, Mixer_Settings.MixerType, Mixer_Settings.CoarseMixFreq, Mixer_Settings.MixerMode, Mixer_Settings.FineMixerScale	None	Cf. PG269
SetMMCMReg	Type, Tile, Mult, Mult_frac, Div, clkout0_div, clkout0_frac	None	Cf. PG269
SetNyquistZone	Type, Tile, Block, NyquistZone	None	Cf. PG269
SetQMCSettings	Type, Tile, Block, QMC_Settings.EnablePhase, QMC_Settings.EnableGain, QMC_Settings.GainCorrectionFactor, QMC_Settings.PhaseCorrectionFactor, QMC_Settings.OffsetCorrectionFactor, QMC_Settings.EventSource	None	Cf. PG269

Table 3: Command List (cont'd)

Command	Input Parameters	Output Parameters	Description
SetThresholdSettings	Tile, Block, ThresholdSettings.UpdateThreshold, ThresholdSettings.ThresholdMode[0], ThresholdSettings.ThresholdMode[1], ThresholdSettings.ThresholdAvgVal[0], ThresholdSettings.ThresholdAvgVal[1], ThresholdSettings.ThresholdUnderVal[0], ThresholdSettings.ThresholdUnderVal[1], ThresholdSettings.ThresholdOverVal[0], ThresholdSettings.ThresholdOverVal[1]	None	Cf. PG269
SetSignalDetector	Tile, Block, Mode, TimeConstant, Flush, EnableIntegrator, Threshold, ThreshOnTriggerCnt, ThreshOffTriggerCnt, HysteresisEnable	Tile, Block	Cf. PG269
SetupFIFO	Type, Tile, Enable	None	Cf. PG269
SetupFIFOObs	Type, Tile, Enable	None	Cf. PG269
Shutdown	Type, Tile	None	Cf. PG269
StartUp	Type, Tile	None	Cf. PG269
CustomStartup	Type, Tile, StartState, EndState	None	Cf. PG269
UpdateEvent	Type, Tile, Block, Event	None	Cf. PG269
DynamicPLLConfig	Type, Tile, Source, RefClkFreq, SamplingRate	RefClkDivider, FeedbackDivider, OutputDivider	Cf. PG269
MultiBand	Type, Tile, DigitalDataPathMask, DataType, DataConverterMask	Type, Tile, DigitalDataPathMask, DataType, DataConverterMask	Cf. PG269
SetCalCoefficients	Tile, Block, CalBlock, Coeffs.Coeff0, Coeffs.Coeff1, Coeffs.Coeff2, Coeffs.Coeff3, Coeffs.Coeff4, Coeffs.Coeff5, Coeffs.Coeff6, Coeffs.Coeff7	None	Cf. PG269
GetCalCoefficients	Tile, Block, CalBlock	Tile, Block, CalBlock, Coeffs.Coeff0, Coeffs.Coeff1, Coeffs.Coeff2, Coeffs.Coeff3, Coeffs.Coeff4, Coeffs.Coeff5, Coeffs.Coeff6, Coeffs.Coeff7	Cf. PG269
DisableCoefficientsOverride	Tile, Block, CalBlock	None	Cf. PG269
SetDACCompMode	Tile, Block, Enable	Tile, Block, Enable	Cf. PG269
GetEnabledInterrupts	Type, Tile, Block	Type, Tile, Block, IntrMask	Cf. PG269
SetDACDataScaler	Tile, Block, Enabled	None	Cf. PG269
GetDACDataScaler	Tile, Block	Tile, Block, Enabled	Cf. PG269
Board Control Commands			
GetExtPllConfig	Board_id, Pll_src	Freq	Return the current frequency of the LMX.
GetExtParentClkList	Board_id	LMK_FREQ_LIST[LMK_FREQ_NUM]	Return a list of frequencies available for the LMK.
GetExtParentClkConfig	Board_id	LMKCurrentFreq	Return the current LMK frequency.
GetExtPllFreqList	Board_id, Pll_src	ADC_FREQ_LIST[LMX_ADC_NUM] or DAC_FREQ_LIST[LMX_DAC_NUM] depending on Pll_Src	Return a list of allowed values for the LMX.
RfclkReadReg	Board_id, chip_id, reg_addr	Board_id, chip_id, reg_addr, data	Read a register from the LMK/LMX.
RfclkWriteReg	Board_id, chip_id, reg_addr, data	Board_id, chip_id, reg_addr, data	Write a register from the LMK/LMX with the following parameters: Board_id (always 0) chip_id (0 :PLL ADC, 1 PLL DAC, 2: LMK) reg_addr (register address from the TCS file) data (data from the TCS file)
SetExtParentclk	Board_id, freq	Board_id, freq	Configure LMK clock with requested frequency.
SetExtPllClkRate	Board_id, Pll_src, freq	Board_id, Pll_Src, freq	Configure PLL (LMX) with requested frequency.

Table 3: Command List (cont'd)

Command	Input Parameters	Output Parameters	Description
PL Data and Control Commands			
GetMemtype	Type, Tile	Memtype	Get selected memory type (block RAM or DDR).
SetLocalMemSample	Type, Tile, Block, numsamples	None	Set the number of samples to be generated or captured.
SetMemtype	Type, Tile, mem_type	Error code on failure	Set memory type to DDR (0) or block RAM (1).
SetMMCM	Type, Tile	MMCM_Lock, Mult, Div, clkout0_div, Clk0DivFrac	Reconfigure the MMCM according to the tile settings, block 0 of this tile, and Fs.
MTS_Setup	Type, Enable	None	Setup the clocking scheme for multi-tile synchronization feature.
SetMMCMFin	Type, Tile, Fplin	MMCM_Lock, Mult, Div, clkout0_div, Clk0DivFrac	Reconfigure the MMCM based on tile settings, block 0 of this tile, and Fplin (user fabric clock input).
GetMMCMFin	Type, Tile	MMCMFin	Returns the current MMCM frequency input.
GetMTS_Setup	Type	Type Tile_Mask	Get the current clocking scheme for all tiles (standard or MTS).
GetMMCMReg	Type, Tile	Lock, Mult, MultFrac, Div, Clk0Div, Clk0DivFrac, Clk1Div	Get the MMCM settings.
MMCM_Rst	Type, Tile	None	Reset the MMCM.
LocalMemInfo	Type	Type, memBaseAddr, numTiles, numMem, memSize, numWords, mem_enable, mem_clksel	Get information on the memory.
LocalMemTrigger	Type, clksel, numsamples, rfdc_ch	None	Trigger the memory channel according to the mask rfdc_ch.
LocalMemAddr	Type, Tile, Block	Type, Tile, Block, Addr_I, Addr_Q	Get the memory addresses associated with the tile and block.
WriteDataToMemory	Tile, Block, number of bytes, interleaved pair	None	In this case, the buffer address is allocated by Linux (CMA pool) from PS DDR and firmware writes the data to the buffer by reading data from a socket.
ReadDataFromMemory	Tile, Block, number of bytes, interleaved pair	None	In this case, the buffer address is allocated by Linux (CMA pool) from PL DDR and firmware reads the data from the buffer and sends it to a socket. In this command, the number of bytes should be aligned to 32.
SetTDDRTPinCtrl	Type, TDD_ModePin	None	Set the channel to control, LSB channel 0, MSB channel 15.
GetTDDRTPinCtrl	Type	Type, TDD_ModePin	Get the channel to control, LSB channel 0, MSB channel 15.

Table 3: Command List (cont'd)

Command	Input Parameters	Output Parameters	Description
GetTDDRTSTrigDelay	Tile	Trig_Delay	Get the trigger delay for ADC per tile. This is the delay between the TDD trigger (from SetTDDRTSTrigSlot) to the hw_trigger of the capture memory. Delay in AXI control clock cycles (slow clock), then resync to tile clock.
SetTDDRTSTrigDelay	Tile, Trig_Delay	None	Set the trigger delay for ADC per tile. This is the delay between the TDD trigger (from SetTDDRTSTrigSlot) to the hw_trigger of the capture memory. Delay in AXI control clock cycles (slow clock), then resync to tile clock.
SetTDDRTSTrig	Trig	None	Set the trigger. 0: stop triggering the memory automatically via hw_trigger. 1: enable triggering of capture memories.
GetTDDRTSTrig	None	Trig	Get the trigger. 0: stop triggering the memory automatically via hw_trigger. 1: enable triggering of capture memories.
SetTDDRTSTrigSlot	Trig_symbol, trig_frame	None	Set the frame and symbol to trigger on.
GetTDDRTSTrigSlot	None	Trig_symbol, trig_frame	Get the frame and symbol to trigger on.
GetTDDRTSSlot	None	Guard_Length, Symbol_Length, Slot_Config	Get configuration of guard band, symbol length, and slot configuration (UL/DL).
SetTDDRTSSlot	Guard_Length, Symbol_Length, Slot_Config	None	Set configuration of guard band, symbol length, and slot configuration (UL/DL).
GetTDDRTSEnables	None	Enables	Read the enable/disable hw_trigger_en on capture mem, per tile.
SetTDDRTSEnables	Enables	None	Set the enable/disable hw_trigger_en on capture mem, per tile.
SetTDDRTSRst	Reset	None	Reset the counters in the TDD block.
GetTDDRTSRst	None	Reset	Read the reset register.

Additional Resources and Legal Notices

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see [Xilinx Support](#).

Documentation Navigator and Design Hubs

Xilinx[®] Documentation Navigator (DocNav) provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the Vivado[®] IDE, select **Help** → **Documentation and Tutorials**.
- On Windows, select **Start** → **All Programs** → **Xilinx Design Tools** → **DocNav**.
- At the Linux command prompt, enter `docnav`.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- On the Xilinx website, see the [Design Hubs](#) page.

Note: For more information on DocNav, see the [Documentation Navigator](#) page on the Xilinx website.

References

These documents provide supplemental material useful with this guide:

1. *Zynq UltraScale+ RFSoc RF Data Converter LogiCORE IP Product Guide* ([PG269](#))
2. *ZCU208 Evaluation Board User Guide* ([UG1410](#))
3. *ZCU216 Evaluation Board User Guide* ([UG1390](#))
4. *RF Data Converter Interface User Guide* ([UG1309](#))
5. *UltraScale Architecture-Based FPGAs Memory IP LogiCORE IP Product Guide* ([PG150](#))
6. *AXI DMA LogiCORE IP Product Guide* ([PG021](#))
7. *AXI GPIO LogiCORE IP Product Guide* ([PG144](#))

Please Read: Important Legal Notices

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at <https://www.xilinx.com/legal.htm#tos>; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at <https://www.xilinx.com/legal.htm#tos>.

Copyright

© Copyright 2020-2021 Xilinx, Inc. Xilinx, the Xilinx logo, Alveo, Artix, Kintex, Kria, Spartan, Versal, Vitis, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. AMBA, AMBA Designer, Arm, ARM1176JZ-S, CoreSight, Cortex, PrimeCell, Mali, and MPCore are trademarks of Arm Limited in the EU and other countries. All other trademarks are the property of their respective owners.