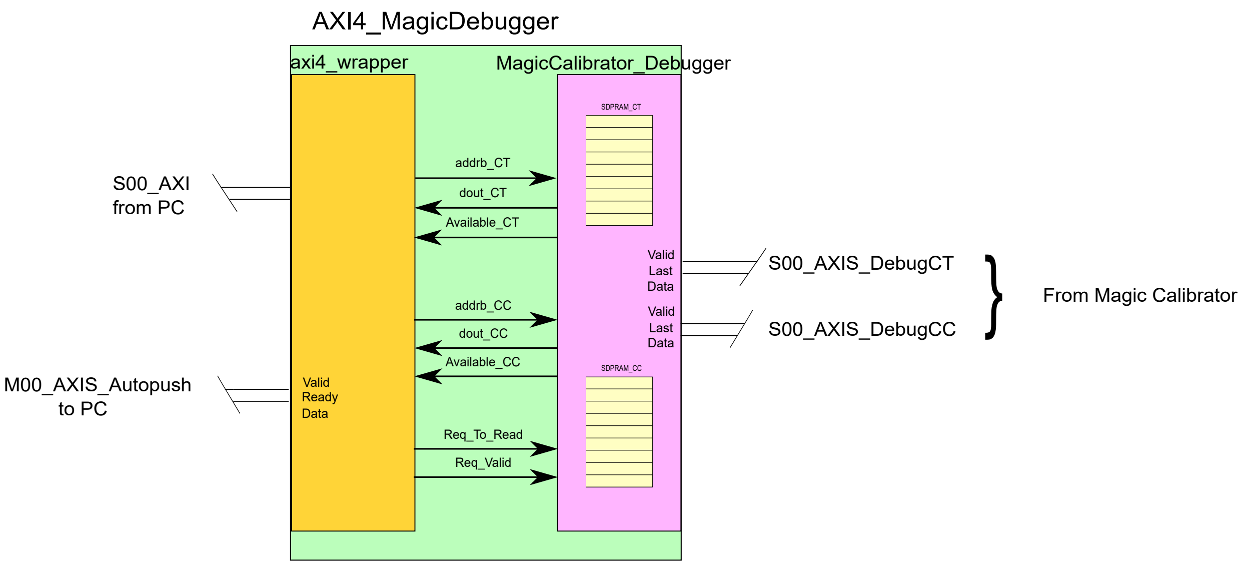
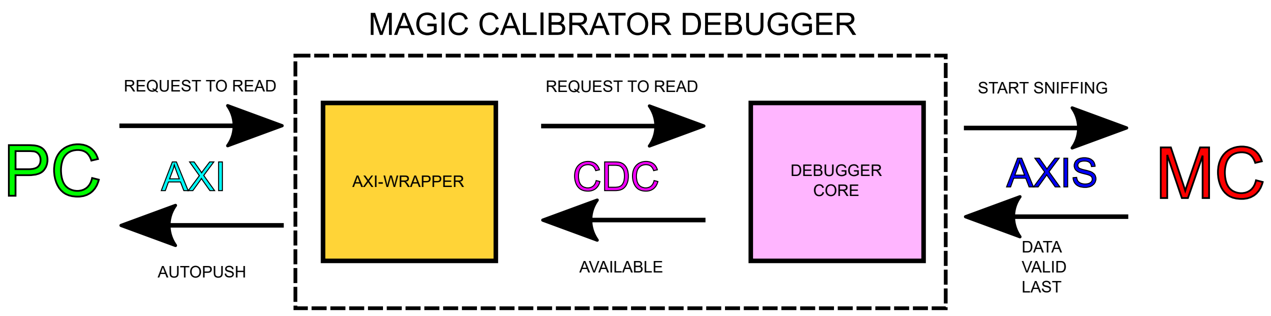
MagicCalibrator_Debugger

This is a Vivado 2017.3 Project, used to make possible the debugging of the BRAM memories that make up the Calibration Table and Characteristic Curve of the AXI4Stream_MagicCalibrator.
This module manages the duplication of the DualPort XPM SDPRAM (*Calibration Table*), where we compute an histogram of the incoming data, and the integrated SinglePort XPM SPRAM (*CharacteristicCurve*) which corresponds to the integration of the histogram (*CalibrationTable*).

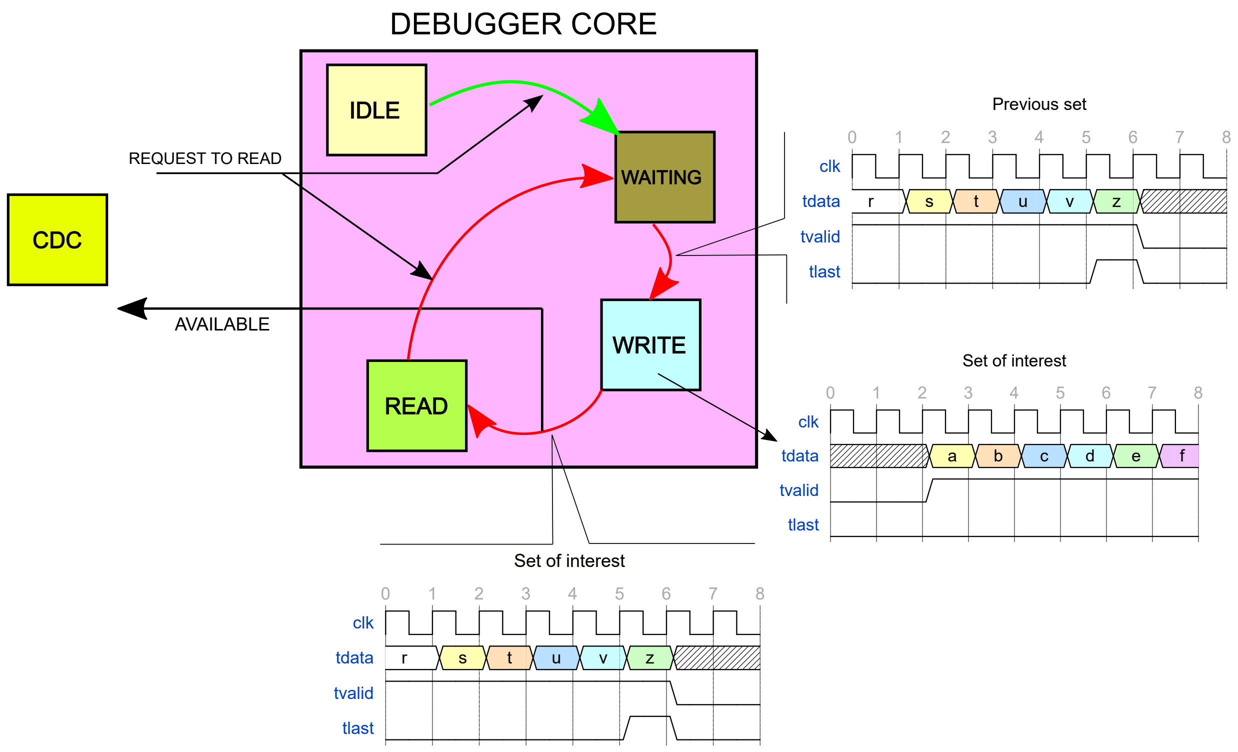


The content of the 2 memories is sniffed during the integration process of the MagicCalibrator, from the 1st to the last address of the memories, bringing out to the Debugger, for each memory, the data, valid and last ports.

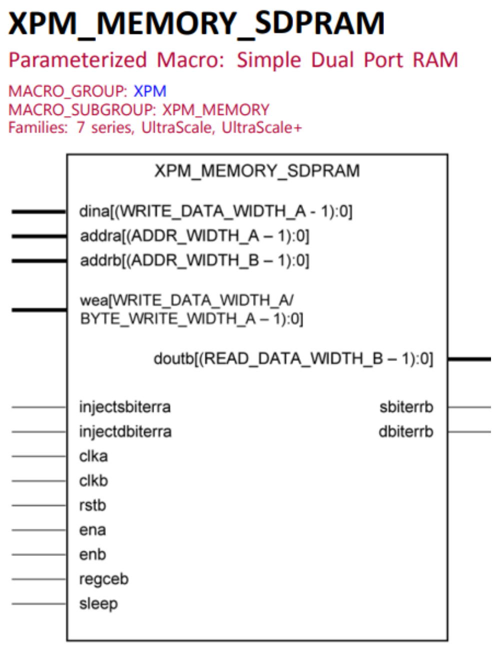
The sniffing of the data is triggered by software, and then a CDC allows the communication between the axi4-wrapper (which talks to the PC) and the Debugger core, since they work at different clock speeds.

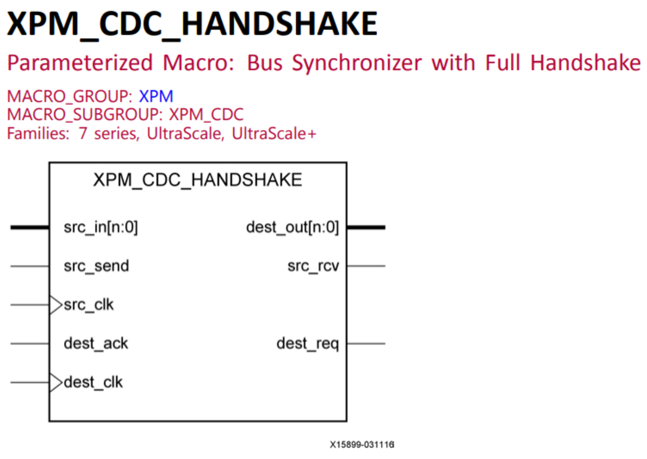


When a memory is requested to be read, the module changes its state from IDLE (the state at start up of the system) to WAITING, where it stays until the set of data being transmitted in that moment is ended, not to mistakenly take just some of the data of a certain set. When there is no more data being transmitted, the state of the debugger goes to WRITE, where it enables the flags to actually write the memories with the data that will come up next. Once the memories are completely copied, the debugger core changes state to READ, rises an Available flag, in order to communicate the presence of the requested data, deasserts the write enable flag and enables the EnB flag, making possible to the axi4-wrapper to directly read the memory through the B Port, by clocking itself the SDPRAM and sending it the address to be read.
Whenever a new Req_To_Read arrives, the debugger core will go back to WAITING state, and the process can start all over again.



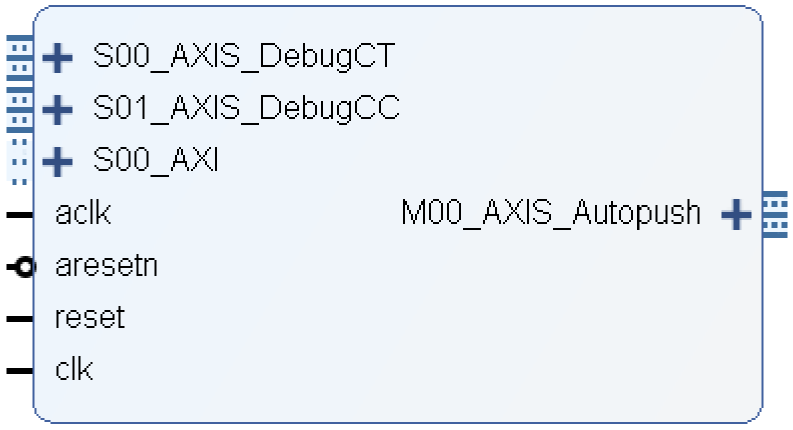
XPM primitives from Xilinx are used, here you can see how they are represented


XPM_MEMORY_SDPRAM
Parameterized Macro: Simple Dual Port RAM
MACRO_GROUP: XPM
MACRO_SUBGROUP: XPM_MEMORY
Families: 7 series, UltraScale, UltraScale+
XPM_MEMORY_SDPRAM
dina[(WRITE_DATA_WIDTH_A - 1):0]
addra[(ADDR_WIDTH_A – 1):0]
addrb[(ADDR_WIDTH_B – 1):0]
wea[WRITE_DATA_WIDTH_A/ BYTE_WRITE_WIDTH_A – 1):0]
doutb[(READ_DATA_WIDTH_B – 1):0]
injectsbiterra
injectdbiterra
clka
clkb
rstb
ena
enb
regceb
sleep
sbiterrb
dbiterrb



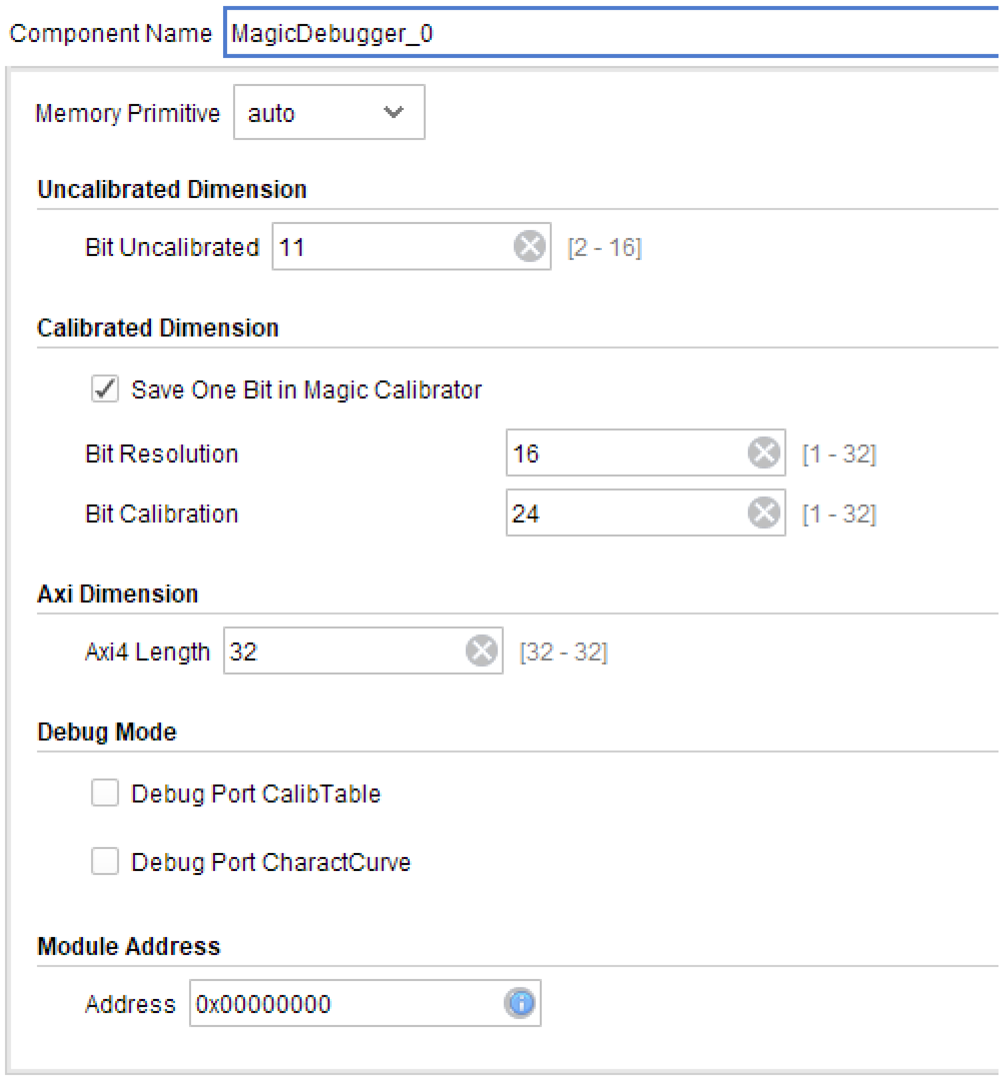
IP-CORE

This is an image of the AXI4_MagicDebugger wrapper for the AXI4 interface



Generic

- **DEBUG_MODE_CT**: True Activates the Debug of CalibTable, *Boolean* type, default "TRUE".
- **DEBUG_MODE_CC**: True Activates the Debug of CharactCurve, *Boolean* type, default "TRUE".
- **BIT_UNCALIBRATED**: Bit Dimension of Uncalibrated_TDL, *POSITIVE* type *RANGE 2 TO 16*. $2^{BIT\_UNCALIBRATED}-1$ is the memory length.
- **SAVE_BIT**: Bit used to decide whether to save one bit in *CharactCurveSPRAM*, *BOOLEAN* type, default **TRUE**.
- **BIT_CALIBRATION**: Number of Bits used to Calibrate, *POSITIVE* type *RANGE 1 TO 32*. *BIT_CALIBRATION* is the memory width.
- **BIT_RESOLUTION**: Number of Bits of the Calibrated_TDL, *POSITIVE* type *RANGE 1 TO 32*.
- **AXI4_LENGTH**: Length of the axi4 ports to set the debug ports length, *POSITIVE* type, default "32".
- **MEMORY_PRIMITIVE**: Length of the axi4 ports to set the debug ports length, *STRING* type, default "auto".



Component Name MagicDebugger_0

Memory Primitive auto

Uncalibrated Dimension

Bit Uncalibrated 11 [2 - 16]

Calibrated Dimension

[✓] Save One Bit in Magic Calibrator

Bit Resolution 16 [1 - 32]

Bit Calibration 24 [1 - 32]

Axi Dimension

Axi4 Length 32 [32 - 32]

Debug Mode

[ ] Debug Port CalibTable

[ ] Debug Port CharactCurve

Module Address

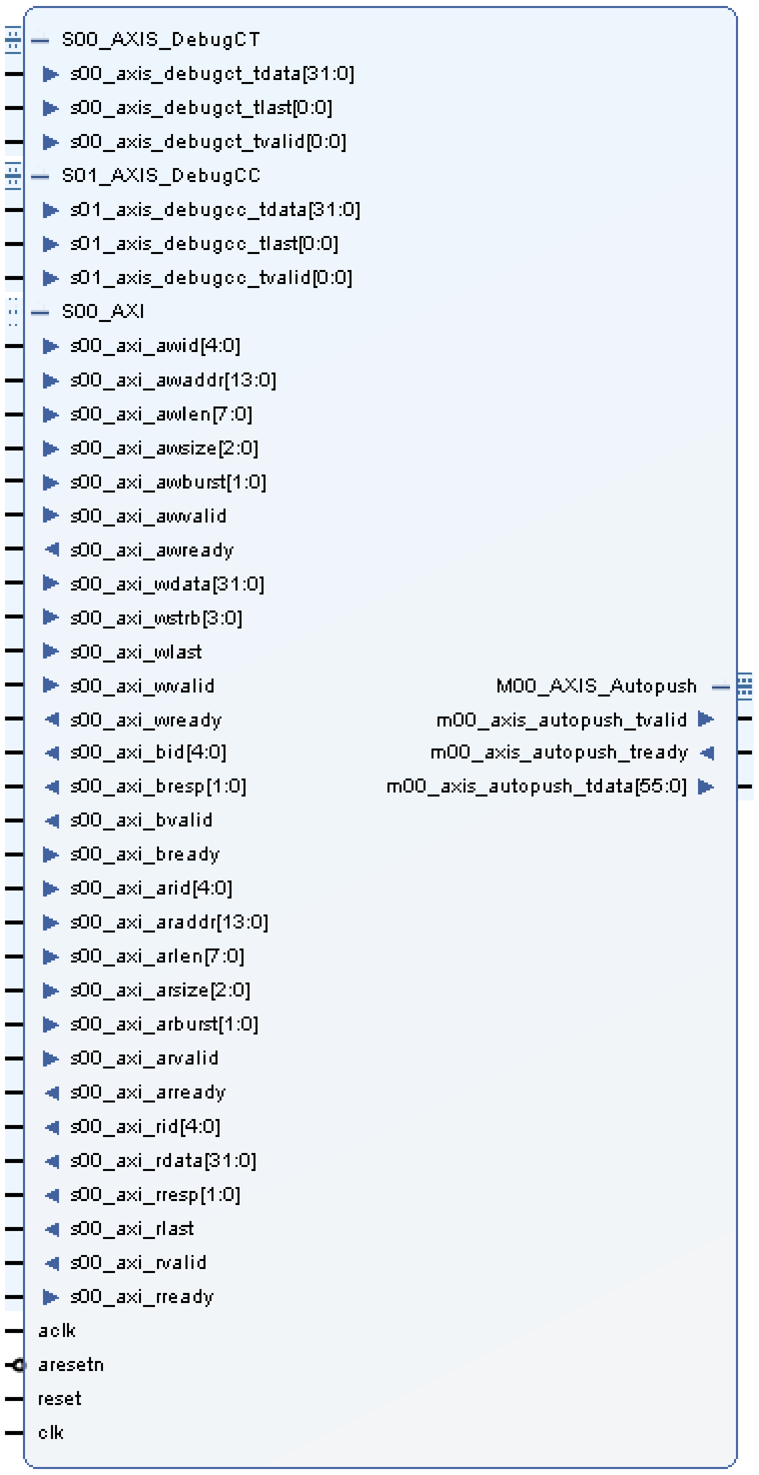
Address 0x00000000

Port

Only ports relative to debugger core are taken in consideration, the AXI slave port is a standard axi slave interface.

- **reset**: Asynchronous reset active high (of Debugger core).
- **clk**: System clock (of Debugger core).
- **s00_axis_debugct**: AXI4 Stream Slave (Input) interface, CT data coming from MagicCalibrator.
  - **s00_axis_debugct_tvalid**: Valid of debug port for CalibTable, *STD_LOGIC* type.
  - **s00_axis_debugct_tdata**: Actual data of CalibTable (same of data written in calib table), *STD_LOGIC_VECTOR(AXI4_LENGTH-1 DOWNTO 0)* type.
  - **s00_axis_debugct_tlast**: tlast of Axis input debug port for CalibTable to signal end of data set, *STD_LOGIC* type.
- **s00_axis_debugcc**: AXI4 Stream Slave (Input) interface, CC data coming from MagicCalibrator.
  - **s00_axis_debugcc_tvalid**: Valid of debug port for CharactCurve, *STD_LOGIC* type.
  - **s00_axis_debugcc_tdata**: Actual data of CharactCurve (same of data written in calib table), *STD_LOGIC_VECTOR(AXI4_LENGTH-1 DOWNTO 0)* type.
  - **s00_axis_debugcc_tlast**: tlast of Axis input debug port for CharactCurve to signal end of data set, *STD_LOGIC* type.
- **m00_axis_autopush**: AXI4 Stream Master interface, to command the autopush request to the host.

- **m00_axis_autopush_tvalid**: Valid of Autopush, *STD_LOGIC* type.
- **m00_axis_autopush_tdata**: Autopush to be sent if *dest_out* of MagicCal_Debugger is 1, 2 or 3, *STD_LOGIC_VECTOR(55 downto 0)* type.
- **m00_axis_autopush_tready**: Ready for Autopush from host, *STD_LOGIC* type.



Sources

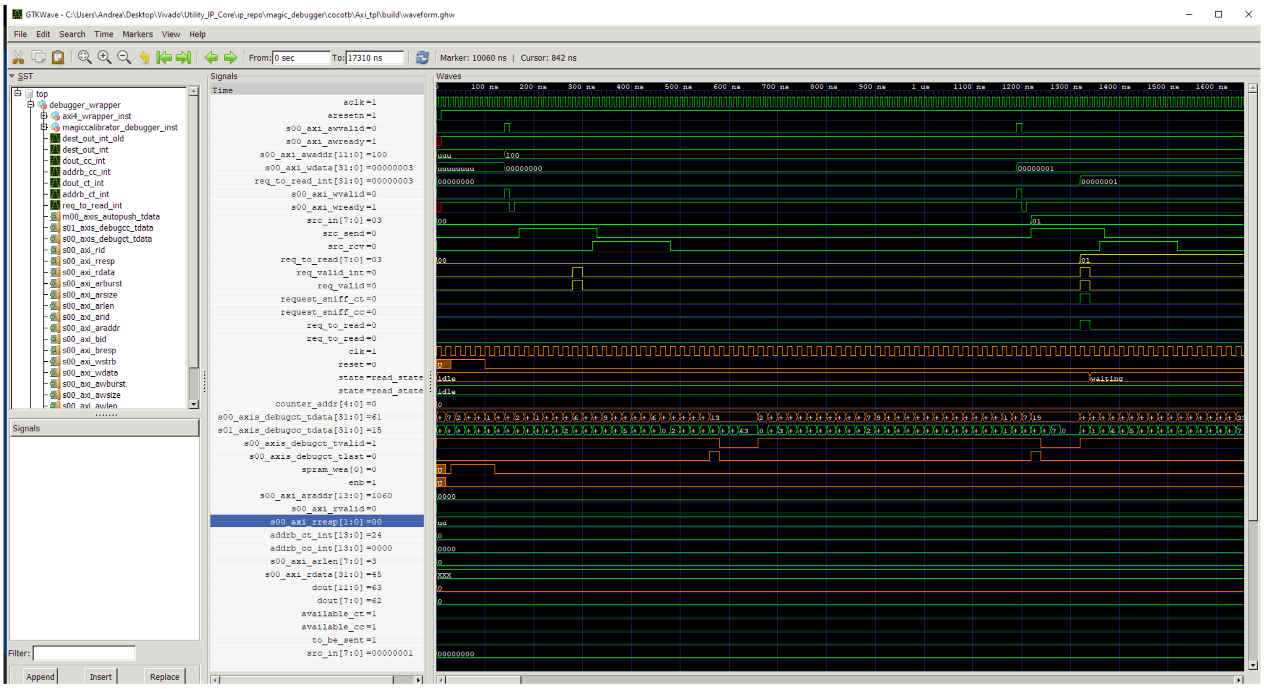
We can find the files of *MagicCalibrator_Debugger* in VHDL in *hdl/*:

- **AXI4_MagicDebugger**: AXI4 Wrapper wrapping axi4-wrapper and MagicCalibrator_Debugger.
  in the nested folder *hdl/axi4-wrapper* we can find:
- **axi4-wrapper**: AXI4 Wrapper for the AXI template module, managing the AXI protocol communication.
  - **axi4-interface**: This module is a simple AXI interface with a XPM_CDC added for communicating with the Debugger core.
  - **axi_channel_slice_register**: This module implements a channel slice register, it can be generated or not depending on a generic in axi4-wrapper
    in the nested folder *hdl/MagicCalibrator_Debugger* we can find:
- **MagicCalibrator_Debugger**: This module manages the instances of 1 or 2 DebugMemoryUnit (one for CT, one for CC) with a XPM_CDC added for communicating with the axi4-wrapper.
  - **DebugMemoryUnit**: This module manages the actual duplication of one of the memories of the MagicCalibrator
  - **MemorySDPRAM**: This module simply instanciates an SDPRAM with the proper dimensions to copy the MC memories.
  - **LocalPackage_MC**: In this package are contained the functions and procedures used by the MagicCalibrator and the MagicDebugger

Simulation

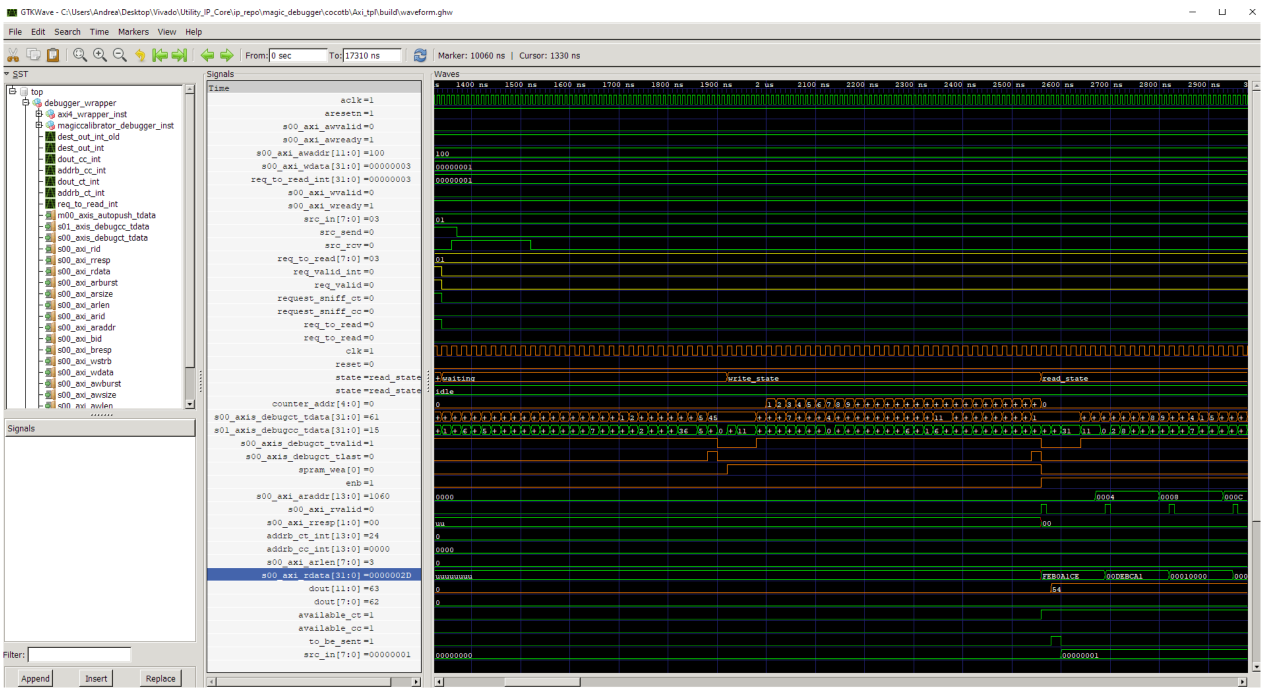
We can find the VHDL simulation and relative Waveforms of *MagicCalibrator_Debugger* in the directory *cocotb/axi_tpl/*.

- **DebuggerWaveforms**: Waveforms of *tb_axi4memory*.


GTKWave - C:\Users\Andrea\Desktop\Vivado\Utility_IP_Core\ip_repo\magic_debugger\cocotb\Axi_tpl\build\waveform.ghw
File
Edit
Search
Time
Markers
View
Help
From: 0 sec
To: 17310 ns
Marker: 10060 ns | Cursor: 842 ns
SST
Signals
Waves
Filter:
Append
Insert
Replace

```
top
  debugger_wrapper
    axi4_wrapper_inst
    magiccalibrator_debugger_inst
    dest_out_int_old
    dest_out_int
    dout_cc_int
    addrb_cc_int
    dout_ct_int
    addrb_ct_int
    req_to_read_int
    m00_axis_autopush_tdata
    s01_axis_debugcc_tdata
    s00_axis_debugct_tdata
    s00_axi_rid
    s00_axi_rresp
    s00_axi_rdata
    s00_axi_arburst
    s00_axi_arsize
    s00_axi_arlen
    s00_axi_arid
    s00_axi_araddr
    s00_axi_bid
    s00_axi_bresp
    s00_axi_wstrb
    s00_axi_wdata
    s00_axi_awburst
    s00_axi_awsize
    s00_axi_awlen
```

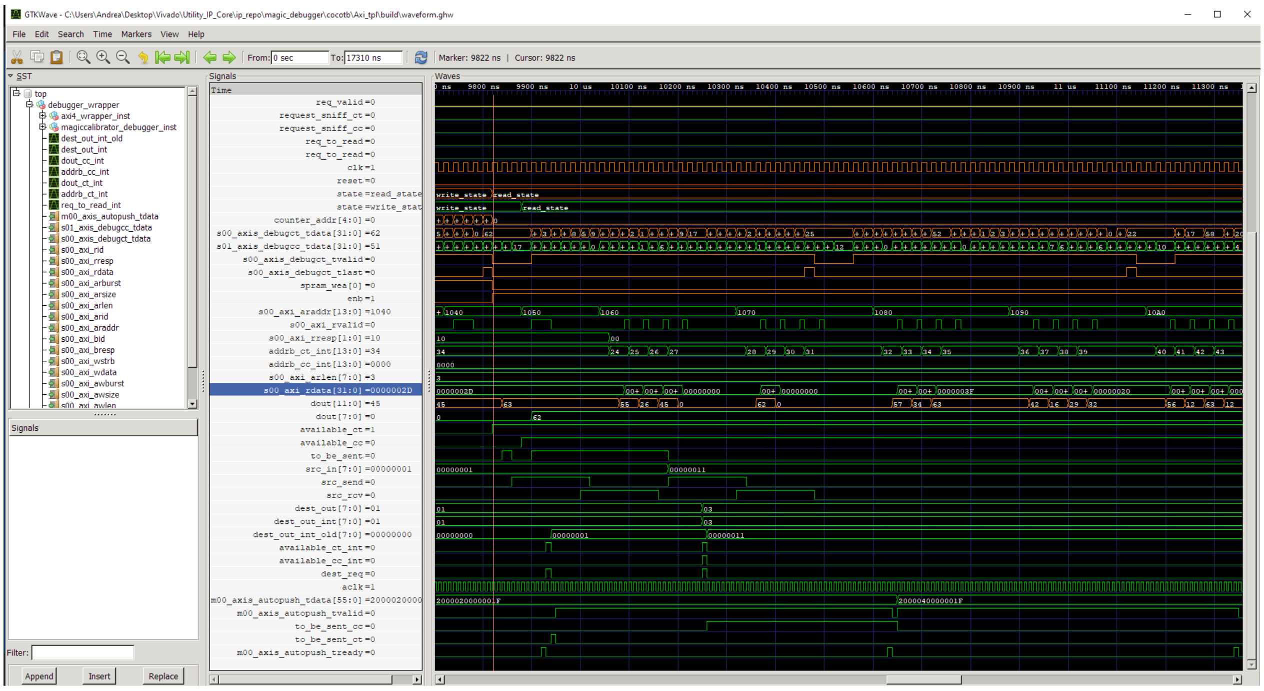
```
Time
                      aclk =1
                   aresetn =1
              s00_axi_awvalid =0
              s00_axi_awready =1
        s00_axi_awaddr[11:0] =100
         s00_axi_wdata[31:0] =00000003
     req_to_read_int[31:0] =00000003
               s00_axi_wvalid =0
               s00_axi_wready =1
               src_in[7:0] =03
                  src_send =0
                   src_rcv =0
           req_to_read[7:0] =03
             req_valid_int =0
                 req_valid =0
          request_sniff_ct =0
          request_sniff_cc =0
               req_to_read =0
               req_to_read =0
                       clk =1
                     reset =0
                     state =read_state
                     state =read_state
         counter_addr[4:0] =0
s00_axis_debugct_tdata[31:0] =61
s01_axis_debugcc_tdata[31:0] =15
     s00_axis_debugct_tvalid =1
      s00_axis_debugct_tlast =0
              spram_wea[0] =0
                       enb =1
       s00_axi_araddr[13:0] =1060
              s00_axi_rvalid =0
         s00_axi_rresp[1:0] =00
        addrb_ct_int[13:0] =24
        addrb_cc_int[13:0] =0000
         s00_axi_arlen[7:0] =3
        s00_axi_rdata[31:0] =45
               dout[11:0] =63
                dout[7:0] =62
              available_ct =1
              available_cc =1
                to_be_sent =1
               src_in[7:0] =00000001
```



GTKWave - C:\Users\Andrea\Desktop\Vivado\Utility_IP_Core\ip_repo\magic_debugger\cocotb\Axi_tpl\build\waveform.ghw

From: 0 sec To: 17310 ns Marker: 10060 ns | Cursor: 1330 ns

```
Time
aclk=1
aresetn=1
s00_axi_awvalid=0
s00_axi_awready=1
s00_axi_awaddr[11:0]=100
s00_axi_wdata[31:0]=00000003
req_to_read_int[31:0]=00000003
s00_axi_wvalid=0
s00_axi_wready=1
src_in[7:0]=03
src_send=0
src_rcv=0
req_to_read[7:0]=03
req_valid_int=0
req_valid=0
request_sniff_ct=0
request_sniff_cc=0
req_to_read=0
req_to_read=0
clk=1
reset=0
state=read_state
state=read_state
counter_addr[4:0]=0
s00_axis_debugct_tdata[31:0]=61
s01_axis_debugcc_tdata[31:0]=15
s00_axis_debugct_tvalid=1
s00_axis_debugct_tlast=0
spram_wea[0]=0
enb=1
s00_axi_araddr[13:0]=1060
s00_axi_rvalid=0
s00_axi_rresp[1:0]=00
addrb_ct_int[13:0]=24
addrb_cc_int[13:0]=0000
s00_axi_arlen[7:0]=3
s00_axi_rdata[31:0]=0000002D
dout[11:0]=63
dout[7:0]=62
available_ct=1
available_cc=1
to_be_sent=1
src_in[7:0]=00000001
```



GTKWave - C:\Users\Andrea\Desktop\Vivado\Utility_IP_Core\ip_repo\magic_debugger\cocotb\Axi_tpl\build\waveform.ghw

From: 0 sec To: 17310 ns Marker: 9822 ns | Cursor: 9822 ns

```
Time
req_valid=0
request_sniff_ct=0
request_sniff_cc=0
req_to_read=0
req_to_read=0
clk=1
reset=0
state=read_state
state=write_stat
counter_addr[4:0]=0
s00_axis_debugct_tdata[31:0]=62
s01_axis_debugcc_tdata[31:0]=51
s00_axis_debugct_tvalid=0
s00_axis_debugct_tlast=0
spram_wea[0]=0
enb=1
s00_axi_araddr[13:0]=1040
s00_axi_rvalid=0
s00_axi_rresp[1:0]=10
addrb_ct_int[13:0]=34
addrb_cc_int[13:0]=0000
s00_axi_arlen[7:0]=3
s00_axi_rdata[31:0]=0000002D
dout[11:0]=45
dout[7:0]=0
available_ct=1
available_cc=0
to_be_sent=0
src_in[7:0]=00000001
src_send=0
src_rcv=0
dest_out[7:0]=01
dest_out_int[7:0]=01
dest_out_int_old[7:0]=00000000
available_ct_int=0
available_cc_int=0
dest_req=0
aclk=1
m00_axis_autopush_tdata[55:0]=2000020000
m00_axis_autopush_tvalid=0
to_be_sent_cc=0
to_be_sent_ct=0
m00_axis_autopush_tready=0
```