NATIONAL
OPTICAL
ASTRONOMY
OBSERVATORY

SYSTEM INSTRUMENTATION GROUP
950 N. Cherry Ave.
P. O. Box 26732
Tucson, Arizona 85726-6732
(520) 318-8000 FAX: (520) 318-8303

TORRENT
AFE Control Module
Firmware Description

NOAO Document TRNT-AD-08-0011
Revision: 0

Authored by:
Dave Sawyer

Please send comments:
pmoore@noao.edu

Revision History

Version	Date Approved	Sections Affected	Remarks
0	8/16/2010	All	Initial draft release - aro

Table of Contents

List of Figures

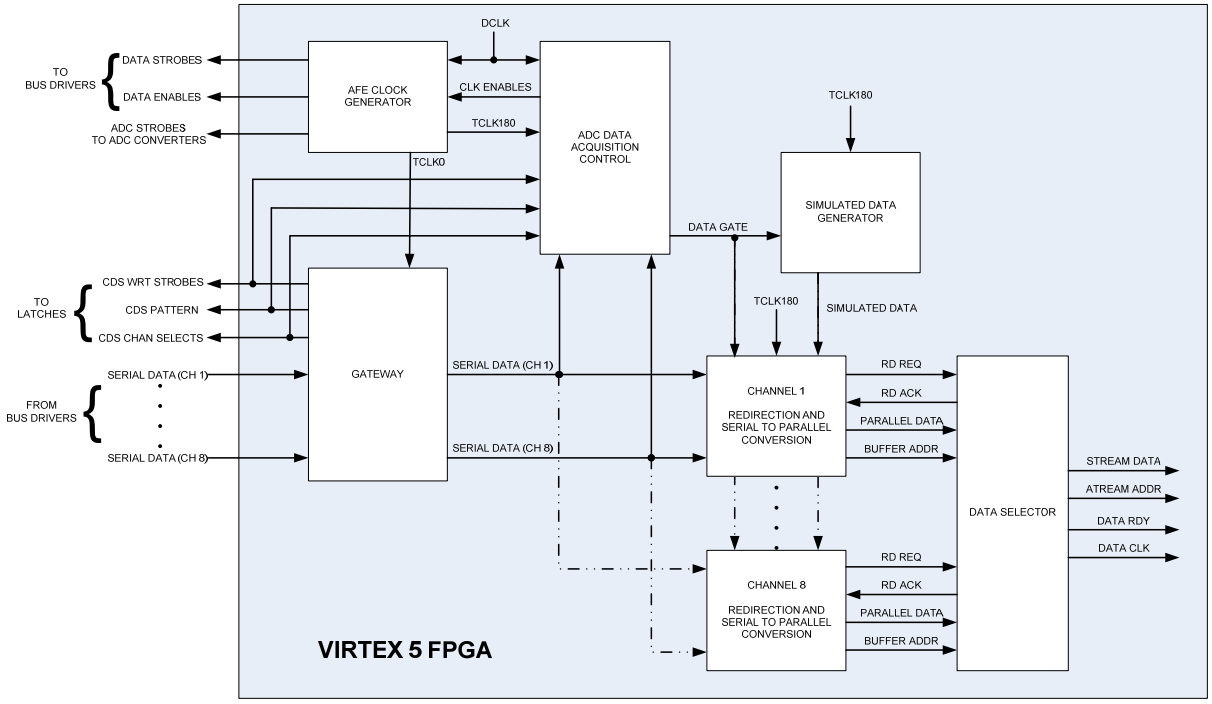
1.0 Introduction

This document is a technical and functional description of the firmware contained in the top-level AFE Control module of the TORRENT image acquisition system. Each major section of this document describes a different functional component of the AFE control firmware design. The functional components include communications and configuration via the Wishbone bus, data acquisition and flow, CDS and clock control via the microsequencer, and DAC configuration and control.

2.0 Wishbone Bus Communication and Configuration

3.0 DATA Acquisition and Flow

This data acquisition and flow component of the firmware describes the process to acquire the data from the ADC(s) and transfer it through the hardware bus drivers into the FPGA. Once the data are in the FPGA it is multiplexed (for channel redirection), converted from serial to parallel, and buffered through a FIFO that formats the data in bursts to match the DDR2 memory and reorders data for descrambling. Figure 3.1 shows a block diagram of the major firmware modules involved in the data acquisition process and their signal relationships. The buffering and processing of the stream data is beyond the scope of this document and described in the PIX Services Module Description document.



Block Diagram of The Firmware Modules Involved in Data Acquisition
Along with Their Signal Relationships
Figure 1

The data acquisition and flow process is controlled through the configuration of registers in the various modules. These registers are set through the Register Control module described in section 2. The table in figure 3.2 lists the registers pertinent to the data acquisition process with a description of their location and function.

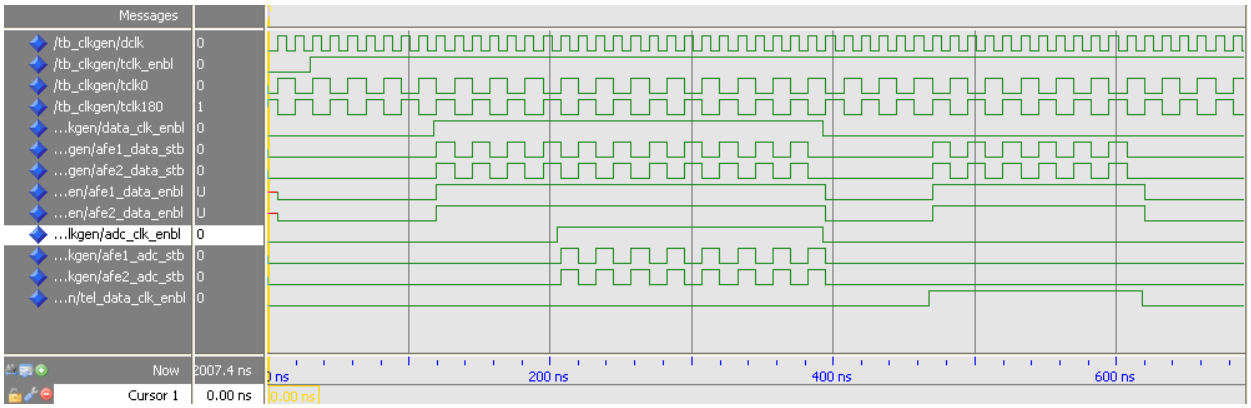
Table 1 - Configuration Registers Used in the Acquisition Control Module

Register Name	Register Address	Word Length	Module	Module Address	Description
AFE1_ChanSlct		[3:0]	ADC Data Acquisition Control		Four enable bits corresponding to four video channels on each AFE board. When the bits are true the CDS clocks are enabled for those channels and the acquisition process uses those bits to set up ADC conversion (busy) monitoring.
AFE2_ChanSlct		[3:0]			
PixelDataWriteSlct		[7:0]	AFE Pixel Data Converters	1, 2, 4, 8, 16, 32, 64, 128	Strobe to select data channel (1-8) while writing configuration data *ChanADCSourceSlct*, *ChanBaseAddr*, *ChanRowIncVal*, *ChanColIncVal*, and *ChanDataCfg*.
ChanADCSourceSlct		[3:0]	Channel Redirection Multiplexer	None	Selects the source for each logical data channel: Values of 1-8 selects the physical ADC channel to input, 9 selects channel ID, and 10 selects synthetic pixel data
ChanBaseAddr		[26:0]	FIFO Acquisition Control	00	27-bit base address of image buffer to use for each frame capture
ChanRowIncVal		[15:0]		01	Number of pixels in an amplifier output row (multiple of 4)
ChanColIncVal		[15:0]		10	Number of pixels in an amplifier output column
ChanDataConfig		[7:0]		11	Data format configuration: Bits[1:0] Amp 0=LL, 1=LR, 2=UL, 3=UR. Bit [2] Descramble mode enable Bits [7:3] Number of active channels
ChanSynthDataType		[1:0]	Synthetic Pixel Generator	None	Sets the synthetic pixel generator mode: 0 = disable the synthetic pixel generator 1 = generate incremental data 2 = generate random data
ChanXferCfg		[5:0]	Pixel Data	None	Configures data for stream data

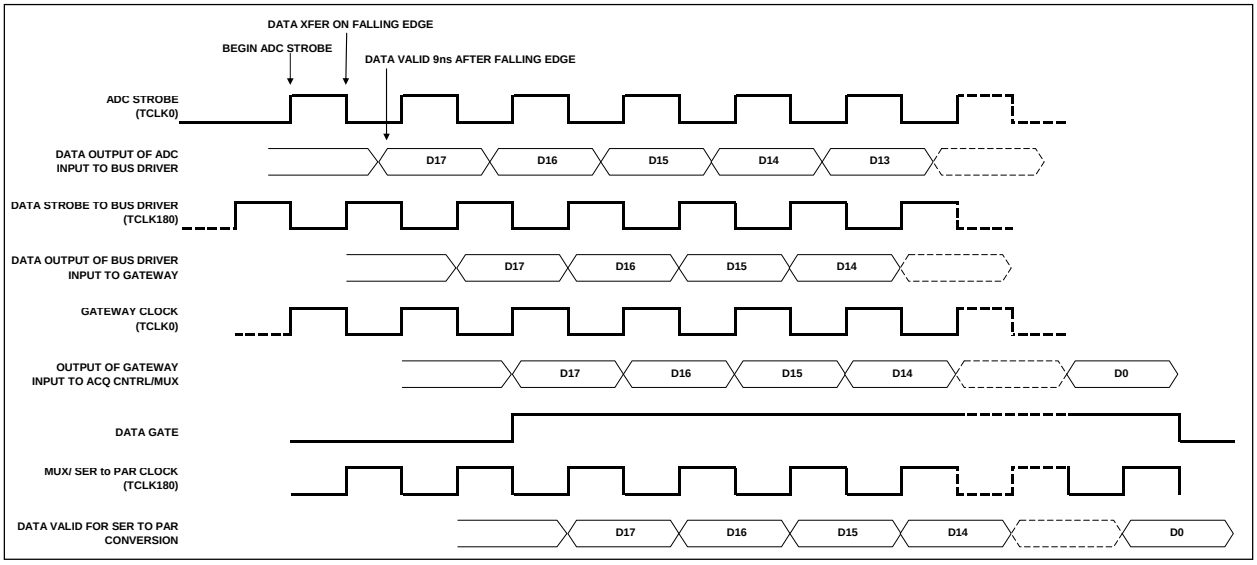
			Selector		bus Bit [0] Data mode: 0 = stream data [default] 1 = descramble data Bits [5:1] are used for the number of active channels (1 to 8 for CCDs)

3.1 AFE Clock Generator (AFE_DataClkGen.vhd)

All clocks are derived from the input clock DCLK (80 MHz). TCLK0 is derived from DCLK as a divide by 2 (40MHz). TCLK180 is also derived from DCLK as a divide by 2, but is 180 degrees out of phase with TCLK0. The out of phase timing sequence of the TCLKs will be used to compensate for the propagation delays of the PCB traces and the latching delays of the translators and is discussed in more detail later in this section. Clock enables from the ADC Data Acquisition Control module are used to control the clock outputs for the acquisition process. Refer to the waveform diagram of figure 3.3 for the clock timing and control. Note the *tclk_enable* input is included in the clock generator module in case there is some future reason (noise or heat reduction) to disable these clocks. In the current implementation, this feature is ignored and the TCLKs run whenever DCLK is active. As can be seen in the timing diagram, the *data_clk_enbl* input activates the bus driver enable signals, *afe#_data_enbl*, and the data strobes, *afe#_data_stb*, to control data transfer through the bus drivers. The data clocks are synchronized to TCLK180. The *adc_clk_enbl* input activates the adc strobes, *afe#_adc_stb*, to clock data out of the ADCs. The ADC clocks are synchronized to TCLK0. Figure 3.4 is a timing diagram showing the clocks and data transfer through the hardware bus drivers and onto the FPGA gateway.



Clock Generator Module Waveform Timing and Control
Figure 2



Timing diagram showing the data flow from the ADC through the bus driver and into the FPGA gateway and serial to parallel converter

Figure 3

The ADCs clock data out on the falling edge of the ADC strobe and the data are valid at the output after an internal 9 ns delay. Additional propagation delays from the PCB traces will further delay the arrival of the data at the bus driver. Thus, the bus driver is clocked 180 degrees out of phase with the ADC so that the rising edge (bus drivers are rising edge sensitive) of the data clock is guaranteed to occur while the data is valid. As can be seen in figure 3.4, the data can arrive at the bus driver up to 25 ns (one clock cycle) after the ADC transfer. Likewise, the gateway clock is 180 degrees out of phase with the data clock to allow propagation time between the bus driver and the FPGA. In this case the delay can be up to 12.5 ns (half a clock cycle).

3.2 AFE Gateway (AFE_IFCGateway.vhd)

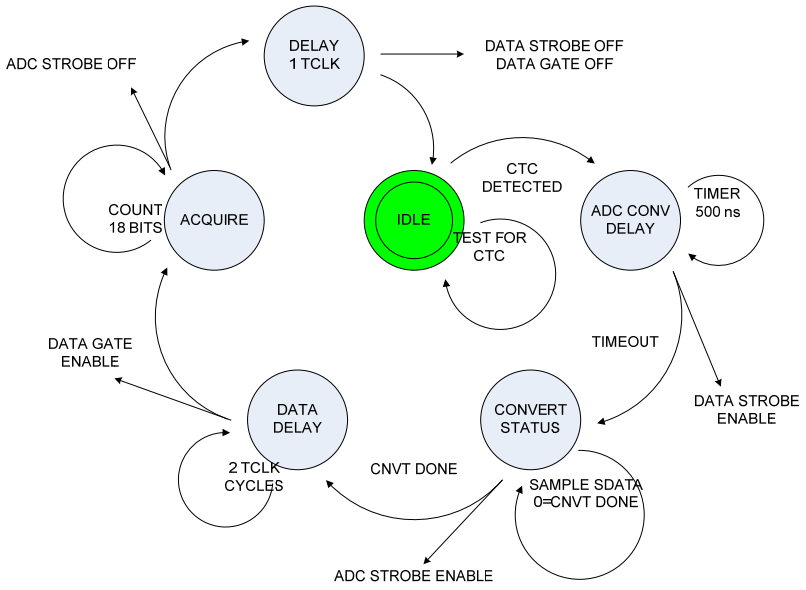
The AFE gateway is the main interface between the FPGA and the external hardware and synchronizes all transactions within the FPGA. The gateway services many signals and functions (synchronized to either DCLK or TCLK0) and will only be discussed here at the level it is involved with data acquisition and flow.

CDS control signals are generated by the microsequencer module (discussed elsewhere in this document) and are latched through the gateway (registered) on the rising edge of DCLK. These CDS signals (write strobes, pattern bus, and channel enables), output from the gateway to the hardware, are also fed back to the acquisition control module where they are used for sensing the command to convert (CTC) to enable the ADC read operation.

The data output from the ADCs are input to the gateway, registered on the rising edge of TCLK0, and output to the acquisition control module (for busy status monitoring) and to the redirection MUX/ serial to parallel converter modules. The details of the timing for the data transfer were described in section 3.1 and shown in figure 3.

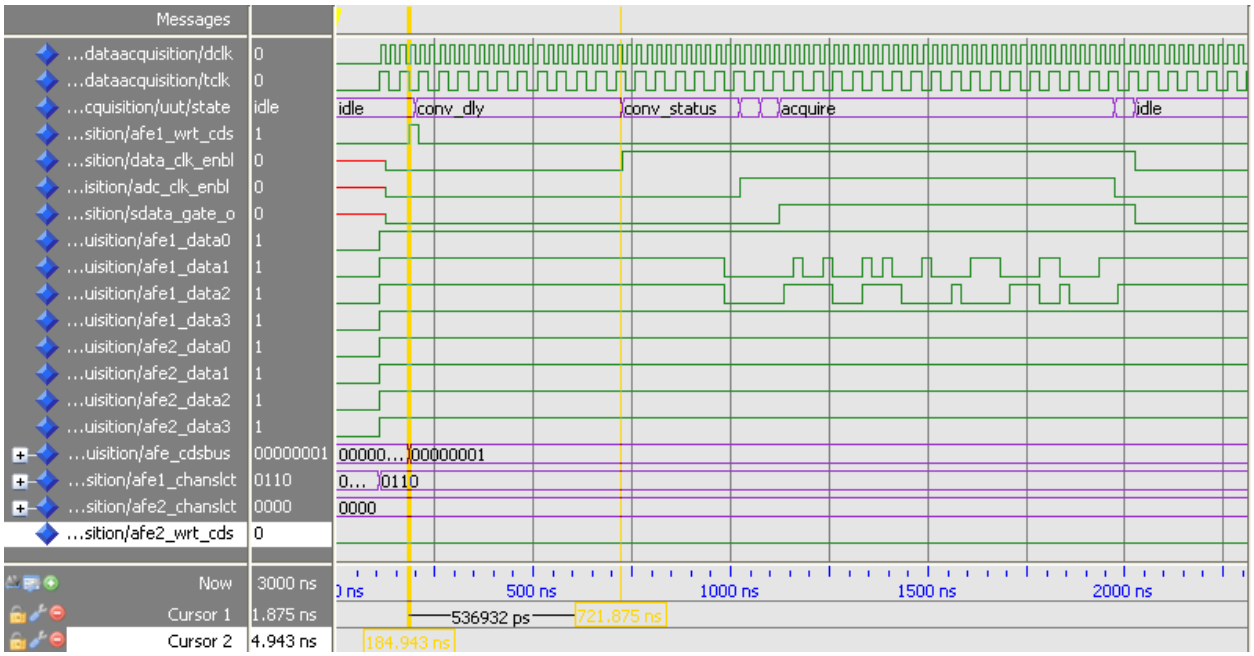
3.3 ADC Data Acquisition Control (AFE_AdcDataAcquisition)

This module incorporates a state machine to control the clocking of data from the ADC to the redirection MUX(es). The state machine scans on the falling edge of DCLK, and synchronizes the output clock enables and data gate to TCLK0 so that it matches the ADC clocking rate. Figure 3.5 shows the functional state machine (FSM) diagram. The FSM sits in an IDLE state and is triggered by the detection of a CTC pulse. After a CTC occurs, a DELAY state of 500 ns is invoked to allow the minimum time for the ADC to convert the data. After the conversion delay, the data strobe is enabled and the FSM changes to a CONVERT STATUS state where the ADC is tested for its busy status (only active channels are monitored per the *AFE_ChanSlct* registers). This is done by testing the active serial data which goes low (0V) when the conversion is done (note, when no channels are active this state is skipped, for instance if synthetic data are generated). When the ADC conversion is done, the ADC clock strobe is enabled and the FSM delays for two TCLK cycles to allow the data to transfer across the bus drivers and gateway. After the delay the serial data gate is enabled and the FSM changes to an ACQUIRE state where the ADC clock strobe is counted for 18 cycles (bits). Once all 18-bits have been read from the ADC, the ADC strobe is disabled and the FSM delays for one clock cycle before disabling the data strobe and data gate, and returning to the IDLE state.



The ADC Data Acquisition State Machine
Figure 4

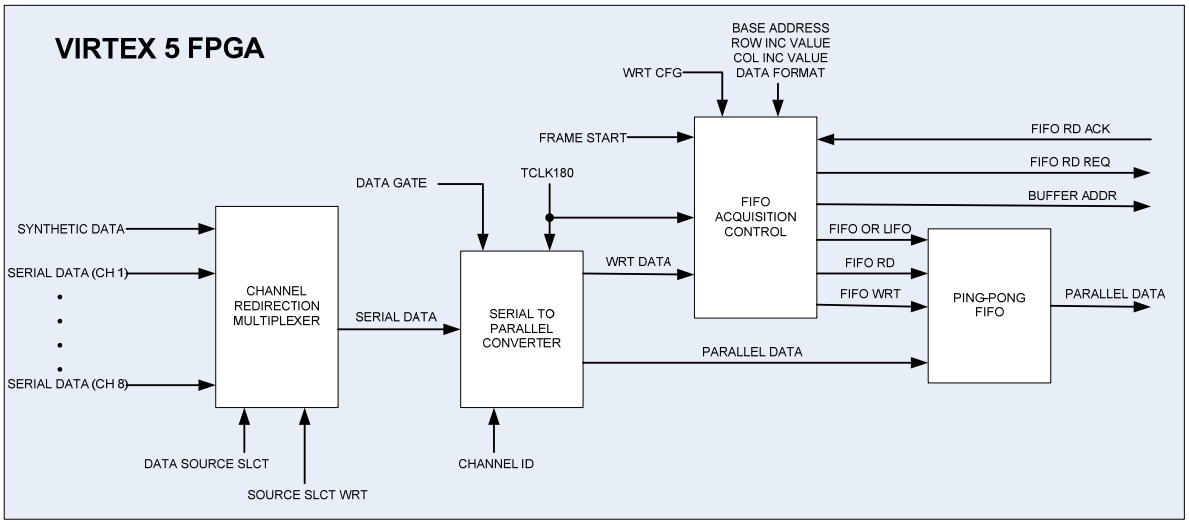
Figure 3.6 is the timing diagram for the ADC Data Acquisition module. The states shown correspond to the states described in the previous paragraph (the states not labeled are the delay states). In this simulated test bench, two channels were active (data1 and data2) and the conversion busy state (*sdata* is pulled high) was simulated to delay well beyond the minimum conversion time. The CTC pulse is generated when the CDS write strobe, *afe1_wrt_cds*, occurs when the CDS pattern, *afe_cdsbus*, includes bit(0)=true. As can be seen in the diagram the data clock is enabled at approximately 500 ns after the CTC pulse (conv_delay state), but the ADC clock is delayed until the conversion is done (conv_status state).



ADC Data Acquisition Module Waveform Timing and Control
Figure 5

3.4 Data Redirection and Serial to Parallel Conversion

There are eight redirection and serial-to-parallel conversion modules, as were shown in Figure 1, to correspond with the eight potential video channels in the Torrent CCD controller. Each one of these eight modules is composed of several sub modules as shown in figure 3.7.

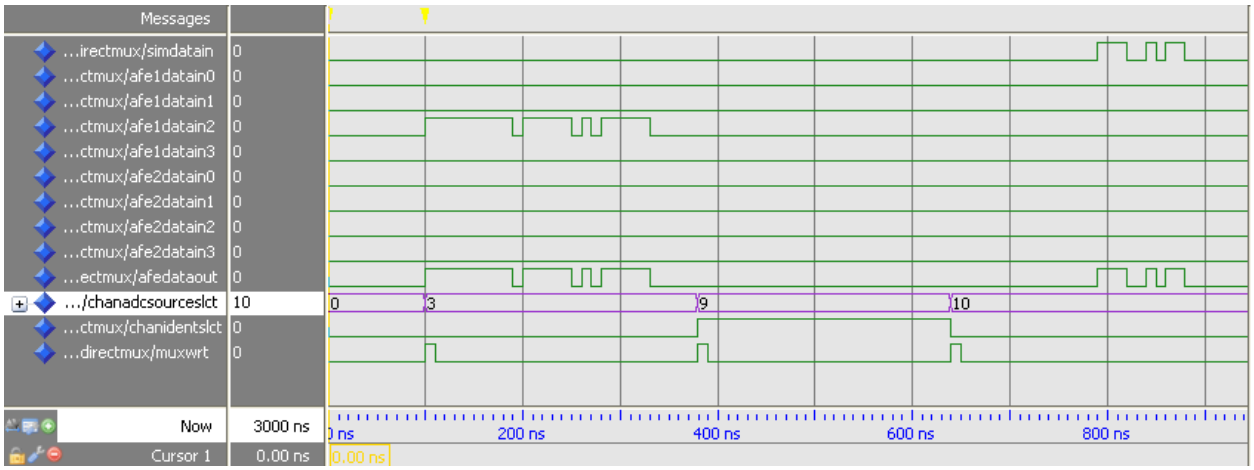


Block diagram of the firmware modules involved in data redirection and serial to parallel conversion along with their signal relationships
Figure 6

3.4.1 Channel Redirection Multiplexer (AFE_RedirectMux.vhd)

This module redirects one of eight AFE channel inputs to the serial to parallel converter. It also triggers a signal to the serial to parallel converter to send channel ID information instead of video data

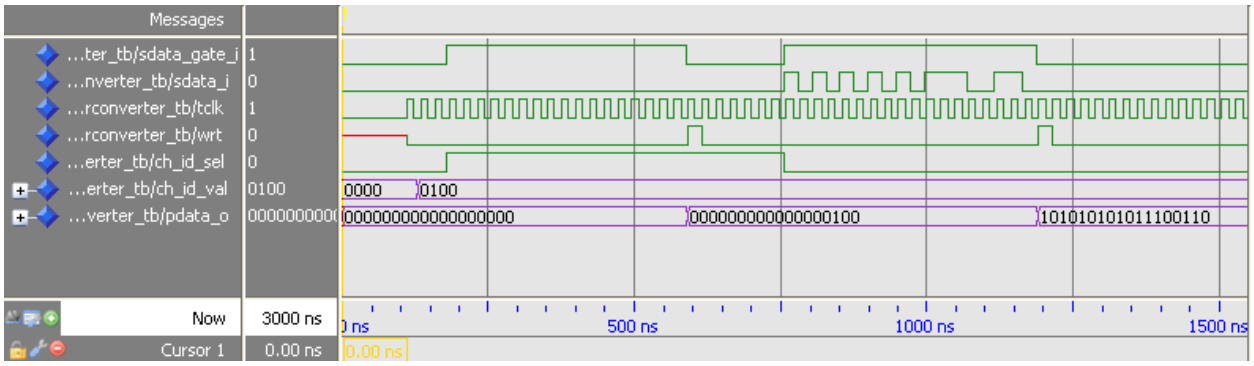
if requested. This module is sensitive to the source select write strobe, *MuxWrt*, where on the rising edge of that strobe one of the input data lines is mapped to the output per the data source select word. The data select word, *ChanAdcSourceSlct*, is 4-bits for selecting the source of data to carry on the logical data channel (module 1-8). Data select values of 1-8 correspond to the physical ADC channel, a value of 9 selects the logical data channel ID, and a value of 10 selects synthetic pixel data. The waveform of figure 3.8 shows the behavior of the module for source data selections of 3, 9, and 10. A value of 3 corresponds to the third data channel (*datain2*) which is mapped to the output (*afedataout*) as shown in the waveform when channel 3 is selected. A data source selection of 9 selects the channel ID which sets the signal, *ChanIdentSlct*, to the true state as shown. The data source selection of 10 selects synthetic data which is mapped to the output as shown.



AFE Redirection MUX Module Waveform Timing and Control
Figure 7

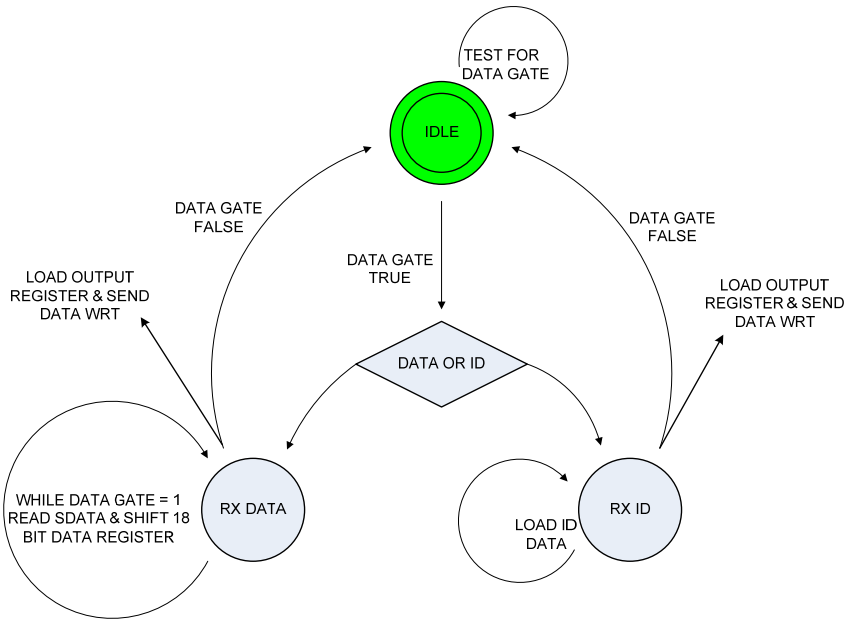
3.4.2 Serial to Parallel Converter (AFE_SerToParConverter.vhd)

The serial to parallel converter module converts the incoming serial data stream from the ADC to 18-bit parallel data. The functionality of this module is synchronized to TCLK180. Figure 3.9 shows the waveform timing of this module. The data gate generated by the ADC Data Acquisition module provides the signal to convert data arriving on the input of the module. As long as the data gate remains high serial data will be clocked in on every TCLK180 cycle. Thus the data gate is normally high for 18 clock cycles (18-bit data). Upon completion of a data read (data gate goes low) a data write signal is generated to signal the FIFO Acquisition Control module that new data are available to the FIFO. This module will substitute the video channel ID (available from Channel ID module on the *ch_id_val* input) as parallel data if requested by the Redirection MUX module, signal *ch_id_sel* set true. The waveform shows the output data set on the *Wrt* strobe for both serial data in and the channel ID modes.



Serial to Parallel Module Waveform Timing and Control
Figure 8

The serial to parallel converter module implements a state machine that is sensitive to the rising edge of TCLK180. Figure 3.10 shows the state machine diagram. The FSM loops in the IDLE state until the serial data gate goes true (signal from ADC data acquisition module). Once a data gate true is detected the FSM tests whether the data source is serial data or channel ID (signal from the redirection MUX module) and jumps to either the RX DATA state or the RX ID state accordingly. In the RX DATA state, the serial data is loaded into the LSB of an 18-bit register and then left-shifted (first data out of the ADC is MSB) and reloaded every TCLK180 rising edge until the data gate goes false. At that time, the FSM loads an 18-bit output parallel data register (dual buffering), sends the write data signal to the acquisition control module, and returns to the IDLE state. In the RX ID state the FSM loads the 18-bit output parallel data register with the channel ID value input to it, sends the write data signal to the acquisition control module, and returns to the IDLE state.



Serial to Parallel Converter State Machine Diagram
Figure 9

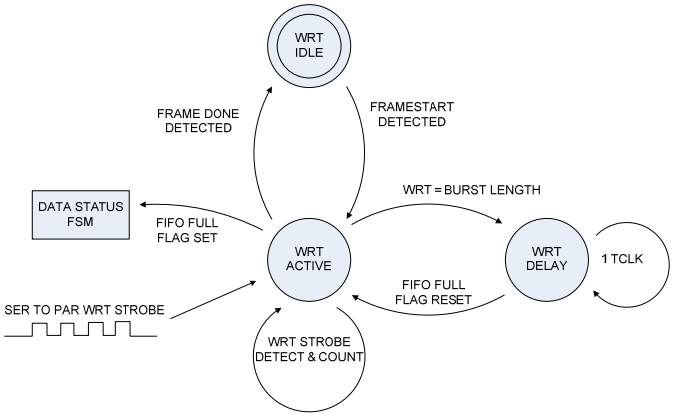
3.4.3 FIFO Acquisition Control (AFE_AcqControl.vhd)

This module coordinates the transfer of parallel pixel data from the Serial to Parallel Converter module, through the Ping-Pong Fifo module, to the Data Pixel Selector module. During this transfer,

the data can be descrambled, for various amplifier output configurations, into the memory area. The buffer memory address locations are determined from data format configuration variables supplied by the PAN (see table 3.2). These values are set to allow the data to be descrambled in memory. For example, an amplifier on the top right of a CCD device will have the first pixel read (top right corner) assigned with an address of [base address + total number of pixels] and the row and column increment values set negative so that the last pixel to arrive is the first in memory space (base address). Descrambling the image in memory will require all pixels to be read into memory before writing to the PAN. Only unscrambled data can be streamed to the PAN during readout. The selection of the DDR2 memory chip requires that data be written in burst-widths of 4 or 8, and the firmware requires that the row increment value be a multiple of the burst-width (the burst width is currently a constant in the firmware set to a value of 4). The PAN will be responsible for calculating/sending valid row increment values (i.e. multiples of 4). The initial implementation of this firmware does not include error checking and thus incorrect row increment values will result in operational errors. If stream mode is selected, via the data format variable, the PAN must set the base addresses for each channel in increments of 1 (e.g. ch0 base addr = 0, ch1 base addr = 1, ch2 base addr = 3, etc).

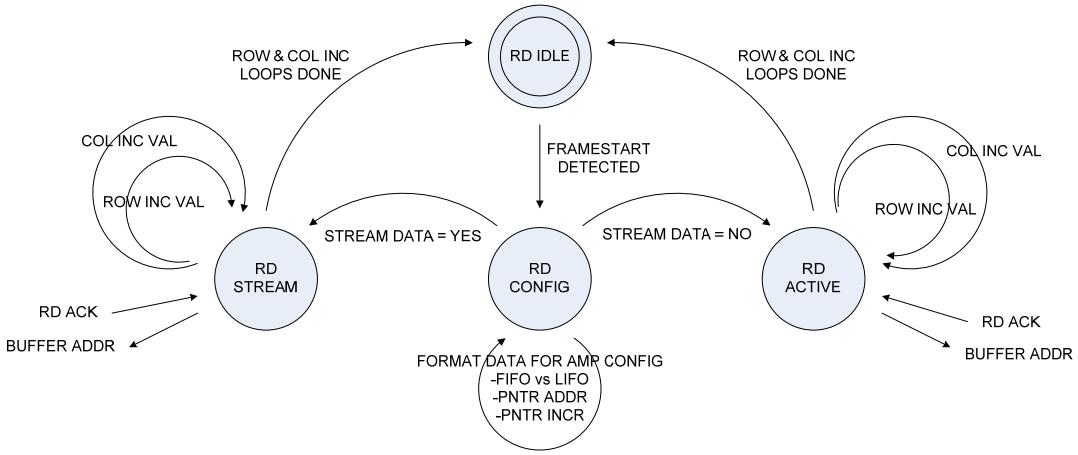
Three state machines are implemented in the FIFO Acquisition module and operate concurrently. One of the state machines monitors the "data write" status during an active exposure to determine when the burst length FIFO is full. Another state machine controls the “data read” process and formats the data being read by the Data Selector module for descrambling in the DDR2 memory. The third FSM monitors the “data status” by detecting the read acknowledge strobes from the Data Selector module to determine when the FIFO is empty.

The data write FSM is sensitive to the falling edge of TCLK180 because the write strobe it is monitoring, *WrtConvData*, is generated on the rising edge of TCLK180 in the Serial to Parallel Converter module. As shown in figure 3.11, the FSM loops in an IDLE state until a *FRAMESTART* pulse is detected (signals the start of an exposure). The state changes to WRT ACTIVE where the *WrtConvData* strobes are detected and counted. Once four write strobes are detected (the data burst-width) a *fifo_full* flag is set to notify the "data status" FSM. After setting the fifo_full flag, a delay state holds the flag true for one clock cycle before resetting the flag and returning to the IDLE state.



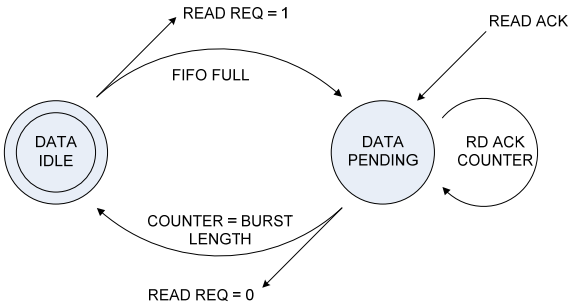
FIFO Acquisition Control “data write” State Machine Diagram
Figure 10

The "data read" state machine is sensitive to the rising edge of TCLK180 because the read acknowledgement strobe that triggers this FSM, *FifoRdAck*, is generated on the falling edge of TCLK180 in the Data Selector module. As shown in figure 3.12, the FSM loops in the IDLE state until a *FRAMESTART* pulse is detected. A RD CONFIG state is then implemented to configure the buffer starting address and address increment value depending on the amplifier configuration selected in the data format register (table 1). This information is also used to set the *FifoRdSlct* output signal to the FIFO module for FIFO vs LIFO operation. Once the read configuration parameters are set in the RD CONFIG state, the FSM changes to either the RD STREAM state or the RD ACTIVE state depending on the stream data mode setting in the data format register. While in either of these states, the FSM detects the *FifoRdAck* pulse from the Pixel Data Selector module and increments the buffer addressing according to the data format. The primary difference between the STREAM and ACTIVE states is that the buffer address is incremented by the number of active channels in the STREAM state, so that the data from multiple channels are properly interlaced (in this mode the base addresses of the different channels must be consecutive values). In the ACTIVE state the buffer addresses are incremented so that each channel is contiguous in memory (in this mode the base addresses for the different channels must be separated by at least the number of pixels in each channel). Once all pixels have been read (i.e. the *ChanRowIncVal* and the *ChanColIncVal* loops are reached, the state machine returns to the IDLE state and an internal *frame_done* pulse is generated to notify the data write FSM.



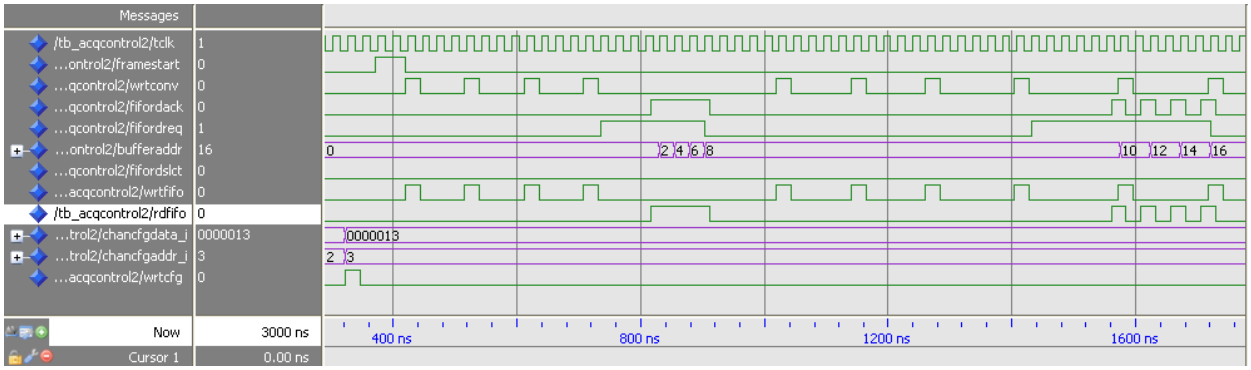
FIFO Acquisition Control “data read” State Machine Diagram
Figure 11

The "data status" state machine is sensitive to the rising edge of TCLK180 because the signals it detects, the *FifoRdAck* read acknowledgement strobe from the Data Selector module and the *fifo_full* flag set by data write FSM, are both generated on the falling edge of TCLK180. As shown in figure 3.13, the FSM waits in the IDLE state until the *fifo_full* flag is set by the data write FSM. Once the flag is set this FSM sets the *FifoRdReq* signal to the Data Selector module and changes to the PENDING state where the *FifoRdAck* signal is detected and counted. Once four *FifoRdAck* pulses are received (the data burst length), the *FifoRdReq* signal is reset (FIFO is empty) and the FSM returns to the IDLE state. The operation of this state machine ensures that the Data Selector module receives (and processes) all read requests from all active channels regardless of data mode (i.e. in stream mode one pixel is read per active channel in sequence whereas in descramble mode all four burst pixels per channel are read in sequence).



FIFO Acquisition Control "data status" State Machine Diagram
Figure 12

Figure 3.14 is a waveform diagram to show the timing of the FIFO read and acknowledge strobes. In this example the data format is configured for stream data with two active channels. This waveform is generated for one of the active channels and thus the buffer addresses are incremented by 2 (the odd number addresses would be assigned to the data from the other active channel). The waveform shows the read request signal, *FifoRdReq*, going true on the first rising edge of TCLK180 after four write strobes have been detected (*WrtConv* is generated by the Serial to Parallel module and mapped through the FIFO Acquisition module to the FIFO module as the signal *WrtFifo*). The buffer address increments on the rising edge of TCLK180 after the read acknowledgement strobe is received (*FifoRdAck* is generated by the Data Selector module on the falling edge of TCLK180 and is mapped through to the FIFO module as the signal *RdFifo*). Note in the example waveform that varying delays of receiving an acknowledgement strobe from the Data Selector module do not affect the operation and nor does receiving the acknowledgement as a four clock cycle pulse or four individual strobes.

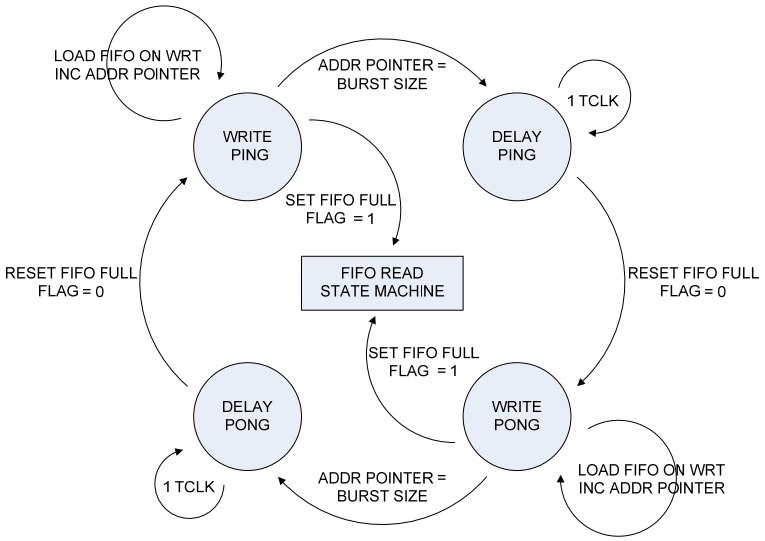


FIFO Acquisition Control Waveform Timing
Figure 13

3.4.4 Ping-Pong FIFO (AFE_Fifo.vhd)

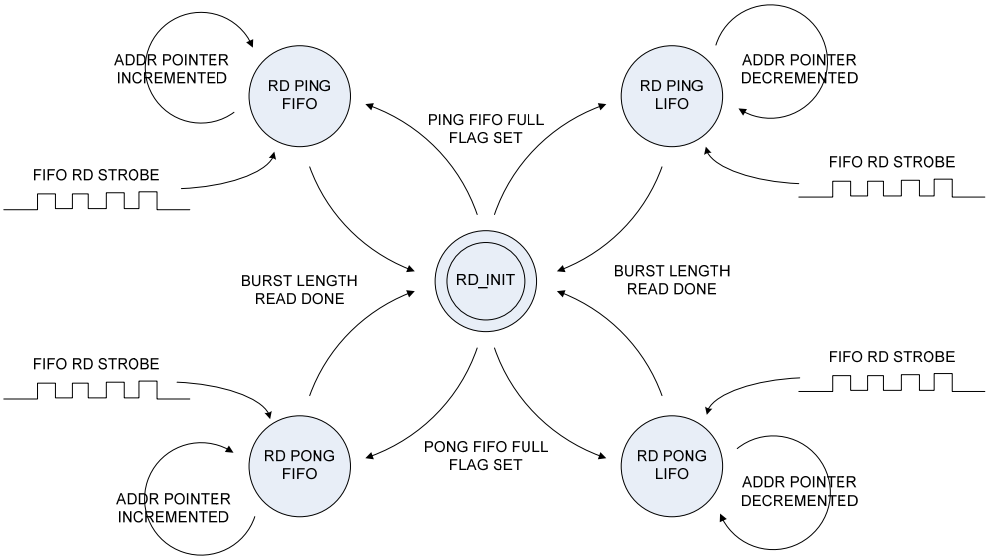
The ping-pong FIFO is used to match the image buffer memory burst size (burst widths supported are 4 and 8 and we have set the default value to 4). There are two 18 bit x 8 deep registers that are used for the ping-pong functionality. One FIFO is loaded with data from the serial to parallel converter module (at the pixel rate) while the other FIFO is read by the pixel data selector (at the memory data rate). Once one FIFO is loaded with a burst-width of data, a pointer swaps to the other FIFO for loading. The ping-pong timing of the FIFO is set by the pixel data rate under the assumption that the data are read by the memory at a higher rate and thus the data read FIFO is emptied before the data write FIFO is full.

The ping-pong operation is achieved by running two state machines concurrently. One state machine handles the write operations (pixel rate data from the serial to parallel converter) while the other handles the read operations (memory rate data to the data selector module). Figures 3.15 and 3.16 show the write FSM and read FSM respectively. The two state machines are coordinated by a *fifo_full* flags that are pulsed by the Write FSM whenever the write process to a FIFO is complete (i.e. the number of pixels in the FIFO equals the memory burst width).



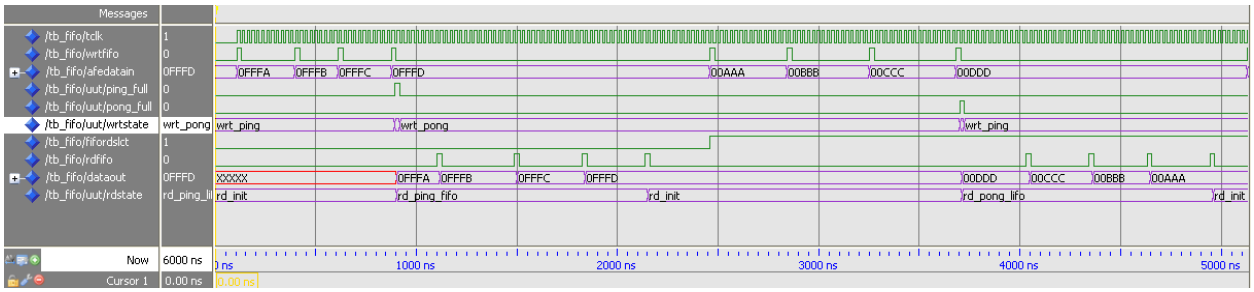
FIFO Write State Machine Diagram
Figure 14

The Read FSM loops in the RD_INIT state until a *fifo_full* flag is set by the Write FSM. The Read FSM then jumps to one of four read states depending on which *fifo_full* flag is set (ping or pong) and the sense of the *FifoRdSlct* signal (FIFO or LIFO) from the FIFO Acquisition module. During FIFO reads, the data are read in ascending order of the FIFO address. During LIFO reads the data are read in descending order of the FIFO address. The later is used for descrambling pixel data bursts as they are read (e.g. amplifiers located at the end of pixel rows). During the read process the data are registered to the output bus on the rising edge of TCLK180 whenever the FIFO RD strobe is true.

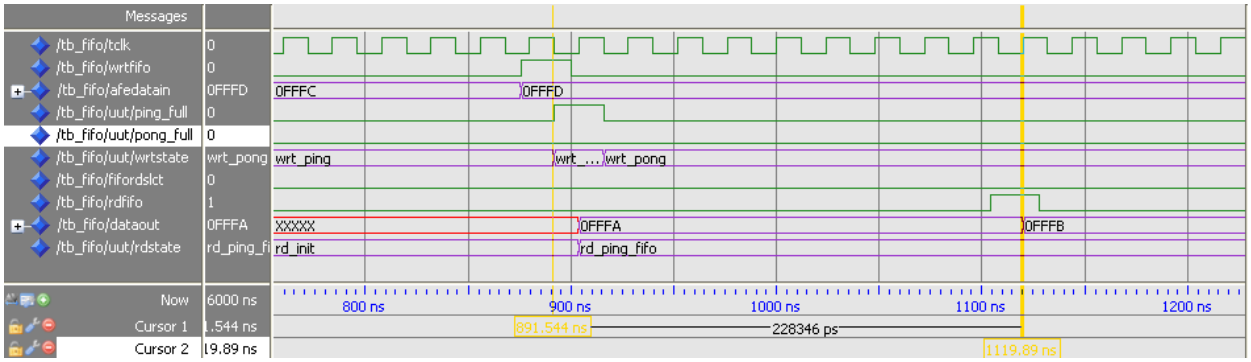


FIFO Read State Machine Diagram
Figure 15

Figures 3.17 and 3.18 illustrate the waveform timing of the FIFO module. In this example, two bursts of four pixels each are written to demonstrate the ping-pong operation. Note the state changes and the behavior of the FIFO full strobes, *ping_full* and *pong_full*. As shown in the serial to parallel converter section, the parallel data are valid before the WRT strobe is generated on the rising edge of TCLK180. The Write FSM is sensitive to the falling edge of TCLK180 to provide a half-cycle (12.5 ns) delay for propagation to ensure data are valid when written to the FIFO (this is highlighted by cursor 1 in figure 3.18 to show that the falling edge of TCLK occurs midway through the *wrtfifo* strobe). Note, the FIFO Acquisition Control module is responsible for sending FIFO read requests to the Data Selector module and thus the FIFO module does not generate any outgoing control signals.



FIFO Module Waveform Timing and Control
Figure 16



Zoomed Section of Figure 16 to Show Critical Timing
Figure 17

NOTE: Cursor 1 marks the detection of the *wrtfifo* strobe and cursor 2 marks the detection of the *rdfifo* acknowledge strobe.

During the read process, the *rdfifo* strobe is received as a read acknowledgement from the Data Selector module after the data are read, and thus the *dataout* bus from the FIFO is valid until the first rising edge of TCLK180 when *rdfifo* is true. The Read FSM operates on the rising edge of TCLK180 and the read acknowledge signal from the Data Selector module is generated on the falling edge of TCLK180 to allow a 12.5 ns propagation delay (this is highlighted by cursor 2 in figure 3.18 to show that the rising edge of TCLK occurs midway through the RD strobe).

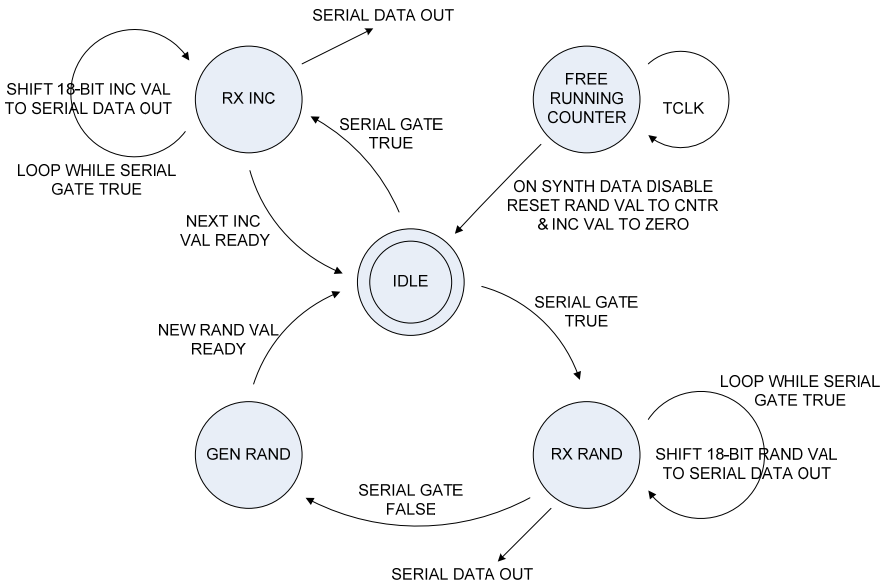
Figure 3.17 also demonstrates the descrambling functionality of the ping-pong FIFO. In this example the first burst of data are read while the the *fifordslct* input (set by the FIFO Acquisition Control

module, 0=FIFO, 1=LIFO) is set low and the second burst of data are read while the *fifordslct* input is set high. As shown the output data are read as FIFO and LIFO respectively.

3.5 Simulated Pixel Generator (AFE_SimPixelGenerator.vhd)

The Simulator Pixel Generator module generates two different types of synthetic data. One type of synthetic data is a ramp where each pixel value is incremented by 1. The other is random data that is generated using exclusive OR functions on the data. The type of synthetic data generated is selected through the 2-bit register *ChanSynthDataType*. A setting of 0 will disable the synthetic pixel generator, a setting of 1 will generate incremental data, and a setting of 2 will generate random data. When a synthetic data type is selected, this module will stream new serial data to the inputs of the Redirection Multiplexer modules during each serial data gate pulse generated by the ADC Data Acquisition module. To actually get synthetic data on the output of a give channel requires that the Redirection MUX be configured properly as described in section 3.4.1

This module incorporates a state machine that is sensitive to the falling edge of TCLK180 so that the data is synchronized with actual pixel data originating from the AFE Gateway module (the Gateway transfers data on the rising edge of TCLK0). As shown in figure 3.19, this FSM loops in the IDLE state until the serial gate signal, *sdata_gate*, goes high indicating that serial data transfer is active. At that time the FSM changes to the RX INC or the RX RAND state depending on the synthetic data type. In both RX states the FSM loops on TCLK180 and each time shifts one bit of the 18-bit data word (MSB first) onto the serial data output. In the RX INC state the data value is incremented after each serial word transfer, before returning to the IDLE state. In the RX RAND state the FSM changes to the GEN RAND state at the end of the serial word transfer where it generates a new random number before returning to the IDLE state. The initial value of the random data is determined by a free running counter that is loaded into the random number value each time the simulated pixel mode is set to the disable state. The initial value of the incremental is reset to zero each time the simulated pixel mode is set to the disable state.



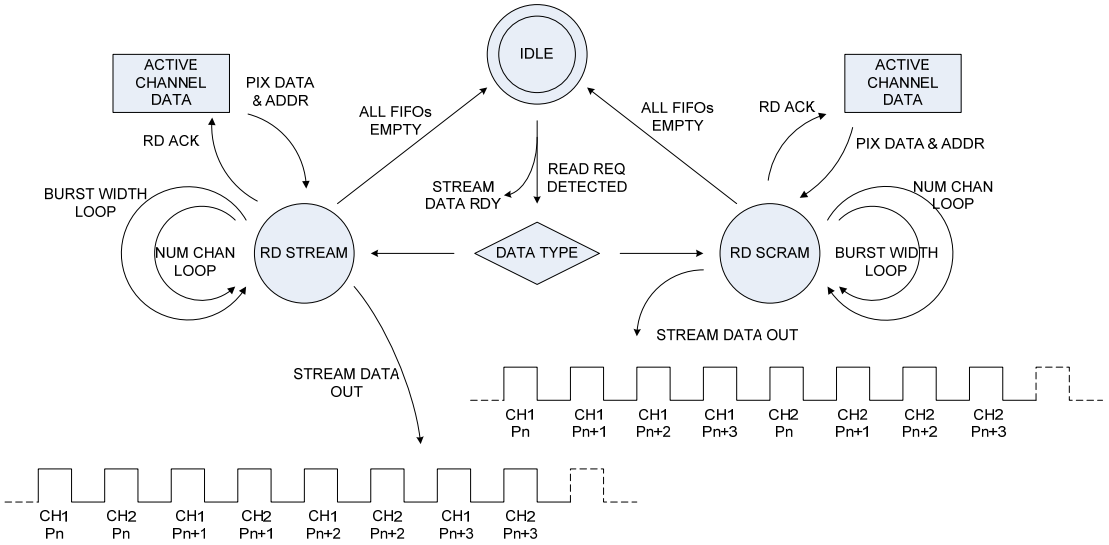
Simulated Pixel Generator State Machine Diagram
Figure 18

3.6 Pixel Data Selector (AFE_PixelDataSelector.vhd)

The Pixel Data Selector module monitors the read requests from the pixel data converters and, based on the data configuration (descrambled vs streamed), transfers the data to the dual port memory emulator in the proper sequence. The data configuration is set through a 6-bit register, *ChanXferCfg*, that contains information about the data mode and number of active channels as follows: Bit [0] stream data (not) enable, and bits [5:1] are used for the number of active channels (1 to 8 for CCDs)

The Pixel Data Selector module incorporates a state machine that is sensitive to the falling edge of TCLK180. This module interacts with the FIFO Acquisition Control module and receives data from the FIFO module, both of which operate on the rising edge of TCLK180, so that propagation delays up to 12.5 ns can be tolerated. As shown in figure 3.20, the FSM loops in the IDLE state until a read request signal is received from the FIFO Acquisition Control module. Upon receiving the read request, the *StreamDataRdy* signal is set true to notify other modules that stream data are ready and the state is changed to one of the read processes, RD STREAM or RD SCRAM, depending on the selected data mode. During this read process the *StreamDataRdy* signal will remain true, the *StreamDataClk* (synchronized to TCLK180) will become active, and one pixel will be transferred on each data clock cycle. In addition, as the data are read during the read process the *FifoRdAck* strobe is triggered to signal the FIFO Acquisition Control module that the data were read.

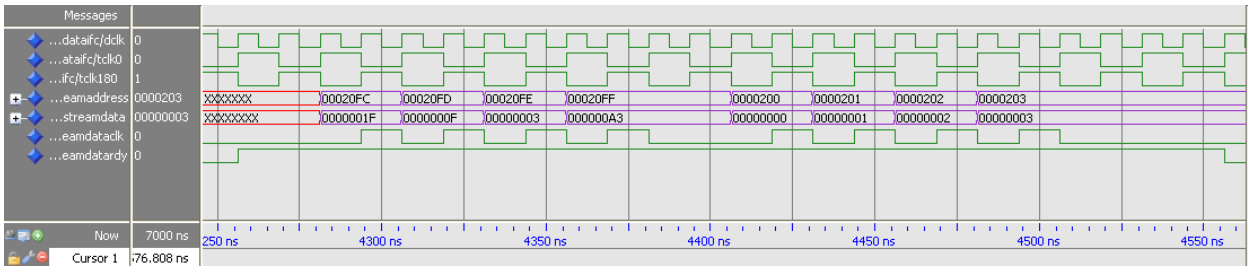
When the [default] stream data mode is selected, the FSM changes to the RD STREAM state where the FIFOs from the active channels are read one pixel at a time from each active channel in sequence and mapped to the stream data bus along with their associated address. In the descramble mode, the FSM changes to the RD SCRAM state where all four pixels from each active channel FIFO are read before moving onto the next channel.



Pixel Data Selector State Machine Diagram
Figure 19

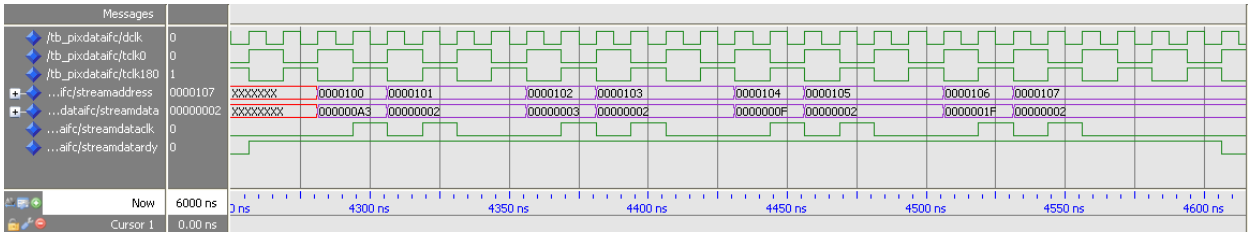
The ping-pong FIFO module feeding this module is configured as 18 bit x four-deep FIFO to match the buffer memory burst width (set to 4 in this implementation). The 18-bit data words that are received from the FIFO are converted by this module to 32-bit data (14 MSBs filled with zeros) and are transferred out as 32-bit stream data. Each pixel is transferred with a unique address even though the memory only associates one address per burst read (four pixels). This is done to ensure that the burst addressing is valid regardless of the data mode (stream or descrambled) and the number of active channels. The Dual Port Memory Emulator module will reconfigure the addressing for the DDR2 memory by only reading the address associated with the first pixel of a burst read and ignoring the rest. The DDR2 uses a 25-bit address as follows: [24:23] bank address, [22:10] row address, [9:0] column address.

Figure 3.21 is a sample output from the Data Selector module showing the timing of the *StreamDataClk* and the valid *StreamData* and *StreamAddress*. Note that the data are valid on the rising edge of the *StreamDataClk* which is synchronized with *TCLK180*. This example shows two bursts of data read in the descramble mode. The first burst is random data from the AFE channel 1 input configured for a UR amplifier, a base address of x100, a row increment of x10, and a column increment of x200. Thus the first pixel read (the last in the first burst) is placed at the last location in memory (x20FF). The second burst is from the pixel data simulator set to the incremental mode with a base address of x200.



Data Selector module waveform timing showing the stream data output in the “descramble” mode
Figure 20

Figure 3.22 shows two bursts of data read in the stream data mode from two channels. The first channel is random data from the AFE 1 input configured for a base address of x100, a row increment of x10, and a column increment of x200. The second channel is configured for Channel ID with a base address of x101. This example demonstrates how the stream data are written in pixel sequence (i.e. the pixel data alternates between the two channels).



Data Selector module waveform timing showing the stream data output in the “stream” mode.
Figure 21

4.0 Micro-Sequencer CDS and Clock Control

5.0 DAC Configuration and Control