Zmod Digitizer Controller IP Core User Guide

Revised September 29, 2022; Author Robert Bocos

1   Introduction

This user guide describes the Digilent **Zmod Digitizer Controller** Intellectual Property. This IP interfaces directly with the Zmod Digitizer 1430 – 125 writing an initial configuration to the analog to digital converter (ADC) featured by these modules and one of the selected frequency configurations of the Clock Generator, demultiplexing the data received over the ADC's parallel interface and forwarding it to the user logic. The **Zmod Digitizer Controller** is intended to be used as a stand-alone IP (the stand alone mode) in projects that do not require processor interaction or it can be used in conjunction with higher level IPs that may provide connectivity with the processing system.

2   Features

- Initializes the hardware on the Zmod Digitizer 1430 – 125.
- Demultiplexes the double data rate (DDR) parallel interface outputted by the Zmod and exports two single data rate (SDR) channels to the user logic in the user clock domain.
- Provides the possibility of overwriting the initial ADC configuration by providing an optional upper level interface that allows indirect access to the ADC's SPI interface.
- Configures the Clock Generator to output a clock signal to the ADC with a frequency from a set list
- Performs offset and gain calibration based on coefficients specified by the user/upper level IPs.

IP quick facts	
Supported device families	Zynq®-7000, 7 series
Supported user interfaces	Custom, AXI Stream
Provided with core	
Design files	VHDL
Simulation model	**Yes** for AD96xx SPI interface; **No** for the IP itself (it can be simulated using the design files)
Constraints file	XDC
Software driver	**N**/A
Tested design flows	
Design entry	Vivado Design Suite - ML Editions 2021.1
Synthesis	Vivado Synthesis 2021.1
Implementation	Vivado Implementation 2021.1

3   Performance

This IP is designed to have a configurable width for the data interface and a configurable sampling rate so that it can manage the low-level communication with the Zmods enumerated in chapter 1 (Introduction). The parallel DDR data interface used to communicate with the ADC is 14 bit wide and the sampling rate has a set of 7 predefined values: 50MHz, 80MHz, 100MHz, 110MHz, 120MHz, 122.88MHz and 125MHz. The **Zmod Digitizer Controller** further exports to the user logic two distinct SDR channels synchronized in the ADC sampling clock domain (*DcoClkOut*).
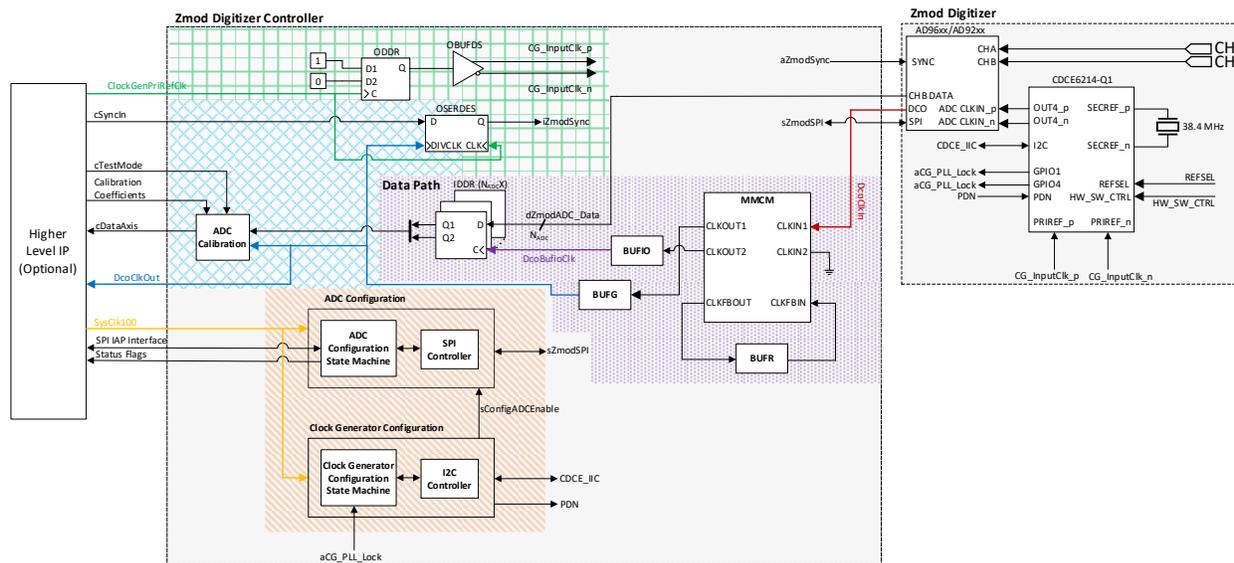
4   Overview

Figure 1. Zmod Digitizer Controller block diagram.

The structure of the IP is presented in Figure 1. The main functionalities are divided as ADC input clock generation, data capture (data path), ADC calibration, ADC configuration (two items related to configuration functionalities will be detailed separately: the configuration state machine and the SPI controller) and Clock Generator configuration (state machine and I2C controller).

4.1   Clocking

The IP is divided in 4 clock domains as shown in Figure 1:

1.   The system clock domain (*SysClk100*) clocks the ADC configuration module, the Clock Generator configuration module together with its I2C controller and the SPI controller. The frequency of this clock is expected to be 100MHz. All signals synchronous with *SysClk100* have the prefix "s".

2.   The ADC input clock (*ClockGenPriRefClk*) is used to clock the primitives responsible with passing the input clock to the Clock Generator which then feeds the ADC with a clock signal. All signals synchronous with *ClockGenPriRefClk* have the prefix "i".

3.   The DCO clock (*DcoClkIn*) is generated by the ADC and is used to clock the IDDR primitives that demultiplex the ADC incoming DDR data bus through a BUFIO primitive (*DcoBufioClk*) and the ADC Calibration module using a BUFG primitive (*DcoClkOut*). All signals synchronous with the de-skewed version of *DcoClkIn* have the prefix "do" for *DcoClkOut* or "db" for *DcoBufioClk*.

The IP does not constrain the clocks it requires as inputs, except for the DCO clock, therefore clocks need to be constrained in the top-level design either manually or by relying on the auto-derived constraints, if using clock modifying blocks. It is the user's responsibility to correctly configure the input clocks of this IP.

4.2   Reset

This IP has a single asynchronous reset input with negative polarity (aRst_n) which resets the logic in all four clock domains. To assure that recovery/removal time of sequential logic is respected, the reset input is distributed to the different clock domains throughout the IP by ResetBridge modules. The ResetBridge modules are responsible with converting the asynchronous reset input in reset signals with synchronous de-assertion (RSD) for each clock domain.

The input reset (aRst_n) must be asserted for at least $2*T_{slowest}$, where $T_{slowest}$ represents the period of the slowest clock input of the IP. After applying a reset, the *sRstBusy* output is asserted by the IP. The user/upper level IP has to wait for *sRstBusy* to be de-asserted in order to safely apply a new reset to the **Zmod Digitizer Controller**.

4.3   ADC Input Clock Generation

The ADC ICs (AD96xx) on the supported Zmod Digitizer modules require a differential clock as input. This IP allows a set range of frequencies for this clock while also allowing custom frequency configurations thanks to the Clock Generator that is used to feed a differential clocking signal directly to the ADC. The minimum frequency is limited to 10 MHz, while for the maximum frequency the limit is imposed by the Clock Generator's maximum output frequency [1], the FPGA's clocking network limitations [2] and by the ADC capabilities. This IP does not generate these clock frequencies internally, it only instantiates the IO primitives required to pass it further into the FPGA.

The Clock Generator can be configured to use one of two possible differential input reference clocks, a primary reference clock that is meant to come from the FPGA and a secondary reference clock that is hardwired to an XTAL source.

4.4   Data Path (DataPath.vhd)

The data path consists of two stages. First, each bit of the input parallel data bus is passed to an IDDR primitive used to demultiplex the DDR interleaved format exported by the ADC (Figure 2), obtaining two SDR data channels.
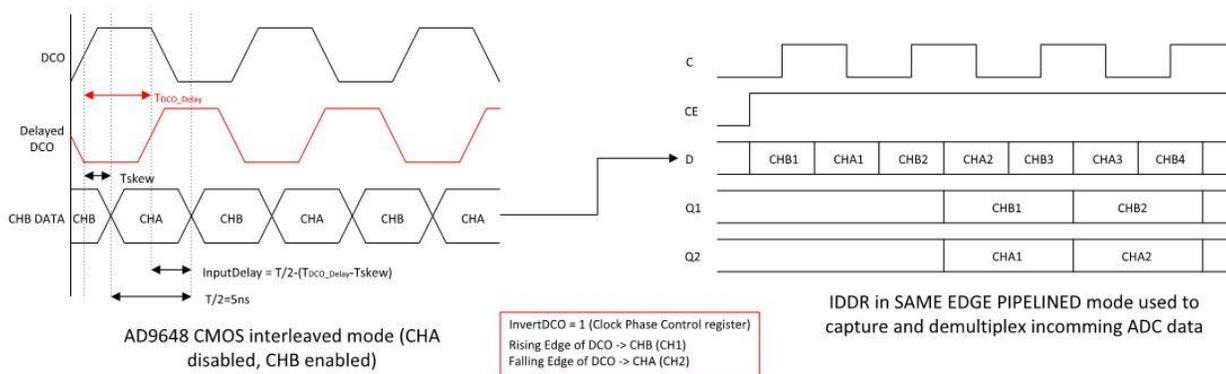

Figure 2. Input data channel demultiplexing

To meet timing requirements a MMCM is used for de-skew purposes. The MMCM is also used to detect the ADC's DCO clock loss. The Data Path module is implemented as a distinct VHDL module and its parameters and ports are described below:

Table 1. Data Path module parameter description

Parameter Name	Description
kSamplingPeriod	The sampling clock period expressed in ns. The maximum value is limited to 100, while it is the user's responsibility to determine the minimum value based on the targeted Zmod's capabilities and the targeted FPGA's clocking distribution network capabilities.
kADC_Width	ADC resolution (number of bits). This parameter ranges from 10 to 16.

Table 2. Data Path port description

Signal Name	Interface	Signal Type	Init State	Description
RefClk	-	I	N/A	Reference clock.
arRst	-	I	N/A	Active high reset (can be asynchronously de-asserted but synchronously asserted in the RefClk domain).
DcoClkIn	-	I	N/A	Connected directly to the Zmod's AD96xx DCO output clock.
DcoClkOut	-	O	N/A	The Zmod's AD96xx DCO output clock forwarded to the IP's top-level module (the de-skew block output).
doEnableAcquisition	-	I	N/A	When logic '1', this signal enables data acquisition from the ADC. This signal should be kept in logic '0' until the downstream IP (e.g. DMA controller) is ready to receive the ADC data. Once this signal has been set to logic '1', it should be kept in this state and never de-asserted.
adoRst	-	I	N/A	Reset signal asynchronously de-asserted and synchronously asserted (in the DcoClkOut domain).
rDcoMMCM_LockState	-	O	N/A	MMCM Locked output signal synchronized on RefClk.
diADC_Data[kADC_Width-1 : 0]	-	I	N/A	kADC_Width bit wide DDR parallel data bus exported by ADC containing Channel1 and Channel 2 interleaved samples.

doChannelA[kADC_Width-1 : 0]	-	O	N/A	The demultiplexed ADC's Channel A data output synchronized in the DcoClkOut domain.
doChannelB[kADC_Width-1 : 0]	-	O	N/A	The demultiplexed ADC's Channel B data output synchronized in the DcoClkOut domain.
doDataOutValid	-	O	N/A	Channel A & B data valid indicator.

4.5   ADC Calibration (GainOffsetCalib.vhd)

The analog front end of the Zmod Digitizer and the ADC itself inevitably distort in a certain degree the input signal. Out of these errors, the gain and offset errors are relatively easy to compensate.

Table 3. ADC Calibration module parameter descriptions.

Signal Name	Description
kWidth	ADC/DAC resolution.
kExtCalibEn	Enables the external calibration interface. Set to "true" when the IP core is expected to be interfaced with the processing system through a high level IP. Set to "false" when the core operates in stand alone mode.
kInvert	When asserted, *kInvert* determines the sign inversion of the data samples received. Used to compensate the physical inversion of some of the channels on the PCB at the ADC/DAC input/output on the Zmod.
kHgMultCoefStatic[17:0]	High gain multiplicative calibration coefficient. If the *kExtCalibEn* parameter is set to "false", the calibration block will expect the value of the multiplicative coefficient to be passed through this parameter. If the value of *kExtCalibEn* parameter is "true", this parameter is ignored, and the high level IP is expected to update the corresponding external port.
kHgAddCoefStatic[17:0]	High gain additive calibration coefficient. If the *kExtCalibEn* parameter is set to "false", the calibration block will expect the value of the additive coefficient to be passed through this parameter. If the value of *kExtCalibEn* parameter is "true", this parameter is ignored, and the high level IP is expected to update the corresponding external port.

Table 4. ADC Calibration module port description

Signal Name	Interface	Signal Type	Init State	Description
SamplingClk	-	I	N/A	Sampling clock. The frequency range supported is between 10 MHz and the maximum frequency supported by the Zmod/target FPGA clock distribution network.
acRst_n	-	I	N/A	Active low reset (can be asynchronously asserted but synchronously de-asserted in the sampling clock domain).

cTestMode	-	I	N/A	cTestMode is used to bypass the calibration block. When asserted, raw samples are provided on the data interface.
cExtLgMultCoef[17:0]	-	I	N/A	Low gain multiplicative coefficient external port. This port is enabled by setting the ExtCalibEn parameter to "true".
cExtLgAddCoef[17:0]	-	I	N/A	Low gain additive coefficient external port. This port is enabled by setting the ExtCalibEn parameter to "true".
cExtHgMultCoef[17:0]	-	I	N/A	High gain multiplicative coefficient external port. This port is enabled by setting the ExtCalibEn parameter to "true".
cExtHgAddCoef[17:0]	-	I	N/A	High gain additive coefficient external port. This port is enabled by setting the ExtCalibEn parameter to "true".
cGainState	-	I	N/A	Corresponding channel gain relay state. The ADC /DAC calibration module requires the gain relay state in order to apply the appropriate calibration coefficients, which are different between the low gain option and the high gain option. The relay gain state is provided by the Relay Configuration module. • 1 = High Gain (relay set). • 0 = Low Gain (relay reset).
cDataRaw[Width-1 : 0]	-	I	N/A	Input raw samples provided by the Data Path module.
cDataInValid	-	I	N/A	Data valid indicator provided by the Data Path module.
cCalibDataOut[15 : 0]	-	O	N/A	Calibrated data output. Regardless of the ADC/DAC resolution, the output of the calibration process is represented on 16 bits.
cDataCalibValid	-	O	N/A	Delayed version of cDataInValid. The same latency obtained for the calibration process (3 sampling clock cycles) is added to the input valid indicator to obtain cDataCalibValid.

In order to compensate the gain and offset errors, at manufacturing, a calibration additive constant and a calibration gain constant are computed and saved in the Zmod's calibration memory, as described in the Zmod Digitizer Reference Manual:

$$V_{in} = \frac{N_{raw} * Range * (1 + CG)}{2^{13}} + CA \qquad (1)$$

were:
- N_{Raw} = the 14 bit, 2's complement integer number returned by the ADC.
- V_{in} = the corrected value of the input voltage.
- CA = factory calibration raw additive constant stored in the Zmod's calibration memory (for the appropriate channel and gain).
- CG = factory calibration raw gain constant stored in the Zmod's calibration memory (for the appropriate channel and gain).
- $Range$ = the theoretical range of the ADC channel input stage:
  - 1.086 (-1V…+1V)

Based on equation (1), the ADC raw (uncalibrated) digital code output corresponding to an input voltage Vin is:

$$N_{raw} = \frac{V_{in} * 2^{13}}{Range * (1 + CG)} - \frac{CA * 2^{13}}{Range * (1 + CG)} \tag{2}$$

While the real range of the Zmod Digitizer is the one defined above, the specified (ideal) range of the product is ±1V. Having the calibration applied, the outputs of the **Zmod Digitizer Controller** should be those of an ideal ADC having these ideal ranges:

$$N_{calib} = \frac{V_{in} * 2^{13}}{Range_{ideal}} \tag{3}$$

were *Range*$_{ideal}$ is 1. From (2) and (3) one can obtain:

$$N_{calib} = N_{raw} * \frac{Range * (1 + CG)}{Range_{ideal}} + \frac{CA * 2^{13}}{Range_{ideal}} \tag{4}$$

This IP works, however, with a fractional representation of the ADC's output code. Equation (4) becomes:

$$N_{calib_{fract}} = N_{raw_{fract}} * \frac{Range * (1 + CG)}{Range_{ideal}} + \frac{CA}{Range_{ideal}} \tag{5}$$

Considering that the inputs of the GainOffsetCalib module are labeled *cDataRaw*, while the outputs are labeled *cDataCalib*, based on (5), the relation between the outputs and the inputs is:

$$cDataCalib = cDataRaw * CoefMult + CoefAdd \tag{6}$$

where:

$$CoefMult = \frac{Range * (1 + CG)}{Range_{ideal}} \tag{7}$$

$$CoefAdd = \frac{CA}{Range_{ideal}} \tag{8}$$

CoefMult and *CoefAdd* are fractional numbers computed for each channel and each gain option. *CoefMult* and *CoefAdd* are further scaled so that they can be passed as integer numbers to the IP:

$$CoefMult_{scaled} = \frac{Range * (1 + CG) * 2^{16}}{Range_{ideal}} \tag{9}$$

$$CoefAdd_{scaled} = \frac{CA * 2^{17}}{Range_{ideal}} \tag{10}$$

The numerical representation of the operands of the calibration process is illustrated in Figure 3. The scaled version of CoefMult and CoefAdd are named "processed calibration coefficients" and these are the constants required by the GainOffsetCalib module, not the "raw coefficients" stored in the Zmod's calibration memory. One can note (Figure 3) that the GainOffsetCalib converts the processed calibration coefficients back to their fractional representation.
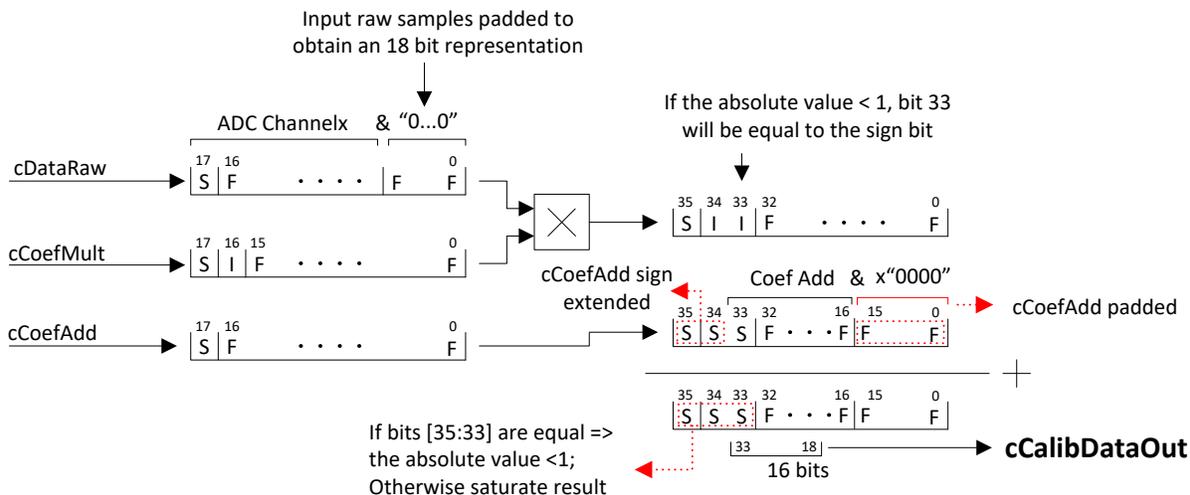
Figure 3: Calibration process

The multiplicative and additive coefficients are passed by the upper layer IP Core if the **Zmod Digitizer Controller** is used in a processor system (the external calibration interface is enabled) or they can be passed as IP core parameters otherwise. The user can obtain the calibration coefficients by booting the Eclypse board with the Linux image provided at https://github.com/Digilent/Eclypse-Z7/releases and run the "decutil enum" command in the command line. The Zmod has to be plugged in one of the Eclypse's board SYZYGY ports. The Digilent Eclypse Utility (decutil) provides a command line interface for discovering information about the features and configuration of an Eclypse platform board. Decutil reports the factory raw calibration parameters and processed calibration parameters as show in Figure 4. This IP works with the **processed calibration coefficients**. More information about the Digilent Eclypse Utility can be found at https://reference.digilentinc.com/_media/reference/programmable-logic/eclypse-z7/decutil.1.pdf.

For simplicity in Figure 1 only one ADC Calibration module is represented. In the actual implementation, a distinct ADC Calibration module is used for each channel. The outputs of the calibration module are concatenated on the AXI Stream output (master) Data interface in the IP's top-level module. Since the data port of the Data interface is 32 bit wide, regardless of the ADC resolution (*kADC_Width*), the most significant 16 relevant bits are selected out of the calibration result, as shown in Figure 3.
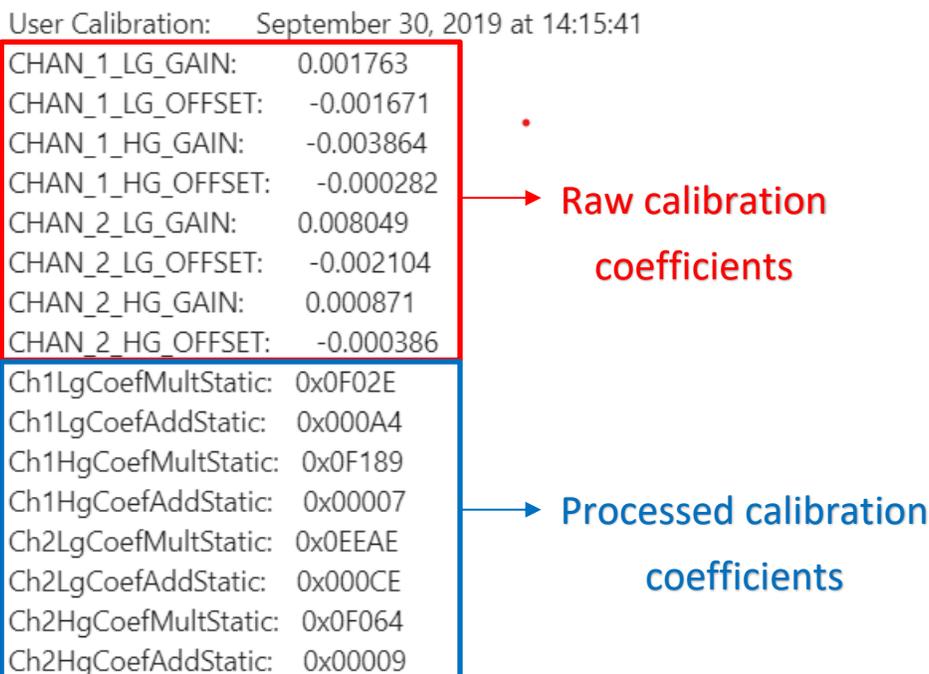
User Calibration: September 30, 2019 at 14:15:41
CHAN_1_LG_GAIN: 0.001763
CHAN_1_LG_OFFSET: -0.001671
CHAN_1_HG_GAIN: -0.003864
CHAN_1_HG_OFFSET: -0.000282
CHAN_2_LG_GAIN: 0.008049
CHAN_2_LG_OFFSET: -0.002104
CHAN_2_HG_GAIN: 0.000871
CHAN_2_HG_OFFSET: -0.000386
→ Raw calibration coefficients

Ch1LgCoefMultStatic: 0x0F02E
Ch1LgCoefAddStatic: 0x000A4
Ch1HgCoefMultStatic: 0x0F189
Ch1HgCoefAddStatic: 0x00007
Ch2LgCoefMultStatic: 0x0EEAE
Ch2LgCoefAddStatic: 0x000CE
Ch2HgCoefMultStatic: 0x0F064
Ch2HgCoefAddStatic: 0x00009
→ Processed calibration coefficients

Figure 4: Obtaining calibration parameters from decutil

4.6   ADC Configuration (ConfigADC.vhd)

The ADC Configuration block sends a predefined list of SPI commands to the Zmod Digitizer, performing the ADC initialization, and manages the SPI Indirect Access Port (IAP). The ports and parameters of the ADC configuration block, which is implemented as a distinct VHDL module are listed below:

Table 5. ADC Configuration parameter description

Parameter Name	Description
kZmodID	Parameter identifying the targeted Zmod. See Table 13 for more details.
kDataWidth	The number of data bits for the data phase of the transaction: only 8 data bits currently supported, parameter provided for future development.
kCommandWidth	The number of bits of the command phase of the SPI transaction.

Table 6. ADC Configuration port description

Signal Name	Interface	Signal Type	Init State	Description
SysClk100	-	I	N/A	100MHz input clock signal.
asRst_n	-	I	N/A	Active low reset (can be asynchronously asserted but synchronously de-asserted).
sADC_Sclk	SPI	O	N/A	Output SPI clock. Signal managed by the SPI controller. More details in section 4.7.

sADC_SDIO	SPI	IO	N/A	2 Wire SPI interface SDIO signal. Signal managed by the SPI controller. More details in section 4.7.
sADC_CS	SPI	O	N/A	2 Wire SPI interface CS signal. Signal managed by the SPI controller. More details in section 4.7.
sInitDoneADC	-	O	N/A	Flag indicating when the Zmod's ADC initialization is complete.
sConfigError	-	O	N/A	This flag is asserted if the ADC initialization fails. An IP reset is required to clear this flag.
sConfigADCEnable	-	I	N/A	ADC initialization enable signal. Configuration of the ADC over SPI should be done after sConfigADCEnable is asserted which only happens after the CDCE clock generator is configured and the PLL inside the CDCE is locked, otherwise the ADC configuration state machine should be kept in reset.
sCmdTxAxisTdata[31:0]	SPI IAP	I	N/A	IAP command TX interface (AXI Stream) data port.
sCmdTxAxisTvalid	SPI IAP	I	N/A	IAP command TX interface (AXI Stream) valid signal.
sCmdTxAxisTready	SPI IAP	O	N/A	IAP command TX interface (AXI Stream) ready signal.
sCmdRxAxisTdata[31:0]	SPI IAP	O	N/A	IAP command RX interface (AXI Stream) data port.
sCmdRxAxisTvalid	SPI IAP	O	N/A	IAP command RX interface (AXI Stream) valid signal.
sCmdRxAxisTready	SPI IAP	I	N/A	IAP command RX interface (AXI Stream) ready signal.

Once the initialization is complete, the state machine of the ADC configuration block enters the idle state where it monitors if there is any valid data on the upper level SPI command interface named the SPI Indirect Access port (IAP). After executing the requested SPI transfers, the state machine passes the received SPI data (for read commands) and returns to the idle state. The SPI IAP is optional, and it can be enabled by setting the *kExtCmdInterfaceEn* to "true". It is designed to interface with 2 AXI StreamFIFOs, one that stores commands to be transmitted (command queue TX FIFO) and one to store the received data (command RX FIFO). Thus, the IAP consists of two AXI Stream interfaces: the IAP command TX interface and the IAP RX command interface. Each element of the command queue should be represented on 32 bits, having the following format:

31 24	23	22 21	20 8	7 0
-	Read/Write	Width	Address	Data

Bits	Field Name	Description
23	Read/Write	Write this bit to 1 for a read command and to 0 for a write command
22-21	Width	Only 1 byte SPI transfers are supported. This field should be always 0h.
20-8	Address	ADC SPI register address
7-0	Data	Data byte to be sent to the AD9648. Ignored for read operations

The ADC Configuration state machine, when in the idle state, monitors the valid signal of the IAP command TX interface (*sCmdTxAxisTvalid*) and, if asserted and if the SPI controller can process a new command (not busy), the ready signal of the IAP command TX interface is asserted for one SysClk100 clock cycle. Write commands only consist of a single transaction on the IAP command TX interface. For read commands, after passing the register read request to the SPI controller and the request successfully completes, the ADC Configuration state machine asserts the valid signal of the IAP command RX interface (*sCmdRxAxisTvalid*) and places the register read data on the data port of the interface (*sCmdRxAxisTdata*). The ADC Configuration state machine waits for the upper layer IP to assert the ready signal on the IAP command interface (*sCmdRxAxisTready*) before it transitions to the idle state.

The initialization configuration command sequence is listed below. After configuring each register, the register data is read back and checked against the expected value in order to determine any SPI transaction error.

1. **SPI Port config register**: Soft Reset (Address: 00h; Data: 3C).
2. **Chip ID Register**: Check ID (Read, Address: 01h; Data: -).
3. **Device Index Register**: Select CHA (Address: 05h; Data: 01h).
4. **Power modes Register**: Digital reset (Address: 08h; Data: 03h).
5. **Output mode register**: Output port logic type -> CMOS 1.8V, output interleave enable, disable port (port A), output invert disable, 2's complement (Address: 14h; Data: 31h).
6. **Device Index Register**: Select CHB (Address: 05h; Data: 01h).
7. **Power modes Register**: Digital reset (Address: 08h; Data: 03h).
8. **Output mode register**: Output port logic type -> CMOS 1.8V, output interleave enable, enable port (port B), output invert disable, 2's complement (Address: 14h; Data: 21h).
9. **Device Index Register**: Select none (Address: 05h; Data: 00h).
10. **Clock Phase Control register**: Invert DCO (Address: 16h; Data: 80h).
11. **Clock Divide register**: Divide input clock by ADC_ClkDiv (Address: 0Bh; Data: ADC_ClkDiv).
12. **Overrange Control register**: Disable overrange output (Address: 2Ah; Data: 00h).
13. **Output Adjust register**: DCO and DATA 2X drive strength (Address: 15h; Data: 00h).
14. **Output delay register**: 1.12ns delay added to DCO (Address: 17h; Data: 81h).
15. **Sync control register**: SYNC enable, continuous SYNC (Address: 3Ah; Data: 02h).
16. **Device Index Register:** Select CHA (Address: 05h; Data: 01h).
17. **Power modes:** Normal operation (Address: 08h; Data: 00h).
18. **Device Index Register:** Select CHB (Address: 05h; Data: 02h).
19. **Power modes:** Normal operation (Address: 08h; Data: 00h).
20. **Device Index Register**: Select none (Address: 05h; Data: 00h).

4.7   SPI Controller (ADI_SPI.vhd)

The SPI controller is designed to carry out basic register access over the Analog Devices 2 wire SPI interface. The controller's parameters and ports are listed below:

Table 7. SPI controller parameter description

Parameter Name	Description
kSysClkDiv	The *sSPI_Clk* signal is obtained by dividing SysClk100 to $2^{kSysClkDiv}$.
kDataWidth	The number of data bits for the data phase of the transaction: only 8 data bits currently supported, parameter provided for future development.
kCommandWidth	The number of bits of the command phase of the SPI transaction.

Table 8. SPI controller port description

Signal Name	Interface	Signal Type	Init State	Description
SysClk100	-	I	N/A	100MHz input clock signal.
asRst_n	-	I	N/A	Active low reset (can be asynchronously asserted but synchronously de-asserted).
sSPI_Clk	SPI	O	N/A	Output SPI clock [4] divided from *SysClk100*. Connected directly to the corresponding Zmod port.
sSDIO	SPI	IO	N/A	2 Wire SPI interface SDIO signal [4]. Connected directly to the corresponding Zmod port.
sCS	SPI	O	N/A	2 Wire SPI interface CS signal [4]. Connected directly to the corresponding Zmod port.
sApStart	-	I	N/A	A pulse on this input initiates the transfers. When asserted, the inputs of the upper layer interface are also registered.
sRdData[kDataWidth-1 : 0]	-	O	N/A	SPI register read received data
sWrData[kDataWidth-1 : 0]	-	I	N/A	SPI register write data.
sAddr[kCommandWidth - 4:0]	-	I	N/A	SPI instruction phase address.
sWidth[1:0]	-	I	N/A	SPI instruction phase word length. The only value currently supported is 0 (1 data byte transferred).
sRdWr	-	I	N/A	Encodes the requested transaction type: '1' - read transaction. '0' – write transaction.
sDone	-	O	N/A	Indicates that the operation requested has completed and that the data placed on the *sRdData* port is valid.

sBusy	-	O	N/A	Indicates when a new command can be processed. It is de-asserted only when the internal state machine of the SPI Controller is in the idle state and it is asserted at any other time. *sApStart* is ignored while this signal is asserted.

An SPI transaction is triggered by generating a pulse on the *sApStart* input. When the pulse is detected, the inputs of the upper layer interface are also registered. The *sRdWr* signal encodes the type of the transaction (read/write). The SPI controller further constructs the command word and the data word based on the module's inputs and formats them on the SPI bus as requested by the Analog Devices' specifications [4]. For read operations, the received data is outputted on the *sRdData* port and can be read by the upper layer module when the *sDone* signal pulses high. When the SPI controller returns to the idle state and is ready to process new commands, the *sBusy* output signal is de-asserted. The *sApStart* signal is ignored while *sBusy* is asserted. The width of the command word is configurable through the *kCommandWidth* generic. The frequency of the SPI output clock is obtained by dividing the controller's input clock frequency by a factor of $2^{kSysClkDiv}$ (*kSysClkDiv* is also a generic). Only single byte data transfers are currently supported. More details about the SPI interface of the supported ADCs can be found in [4].

4.8 Clock Generator Configuration (ConfigClockGen.vhd)

The Clock Generator Configuration block sends a predefined list of I2C commands to the Clock Generator that feeds the ADC a differential clocking signal source. From the IP GUI the user can select from a set of frequency configurations. The possibility of adding custom frequency configurations is open to the user, however, it requires advanced knowledge of the IP, the clock generator and the proprietary software that must be used to generate said custom frequency configurations.

It is the user's responsibility to ensure that after changing the sampling frequency from inside the IP GUI they also they also change the calibration coefficients to match the desired sampling frequency. Each supported frequency configuration has its own calibration coefficients stored inside an EEPROM present on the Zmod. For more information please consult the Zmod Digitizer's Reference Manual, more specifically the paragraph about Calibration.

The REFSEL signal is used to select between the two reference clock inputs of the Clock Generator, a value of '0' selects the secondary reference input while a value of '1' selects the primary one which comes from the FPGA and is a tri-level output signal. This reference clock selection can also be done irrespective of the value of the signal by writing 0x02 for SECREF or 0x03 for PRIREF to the REFSEL_SW register bit field of R2.

The set values of output frequencies provided all use the XTAL input as reference clock and that is not dependent on the value of the REFSEL signal because the Clock Generator's REFSEL_SW register bit field is set to 0x02 and the REFSEL signal is kept in tristate.

The configuration of the Clock Generator registers can also be done via its internal 2 Page EEPROM and the pages can be written and rewritten over I2C. The signal that acts as page select is called HW_SW_CTRL and is also a tri-level output pin.

By default, the Clock Generator is configured to run in Fall-Back Mode (0x67 in 7-bit format or 0xCE in 8-bit format), a mode in which the I2C serial interface is always enabled and the configurations written inside the EEPROM are ignored. This is done by forcing the REFSEL and HW_SW_CTRL output signals to be in High-Z state through the use of an OBUFT FPGA primitive. **This means that the user should**

not add any kind of pull-up or pull-down on those signals as it would disrupt their functionality. The other modes involve its configuration being read from an internal factory programmed 2 Page EEPROM which can have frequency configurations unsupported by this IP.

Table 9. Clock Generator Configuration parameter description

Parameter Name	Description
kCDCE_SimulationConfig	Clock Generator I2C parameter for enabling a shortened configuration for simulation purposes.
kCDCE_SimulationCmdTotal	Clock Generator I2C shortened configuration number of commands to send over I2C for simulation purposes.
kCDCEI2C_Addr	Clock Generator 7-bit I2C address (0x67 (Fall-Back Mode), 0x68(Default Mode), 0x69).
kRefSel	Clock Generator input reference clock selection parameter ('0' selects SECREF(XTAL) and '1' selects PRIREF(FPGA)). Only matters when the Clock Generator is not in Fall-Back Mode.
kHwSwCtrlSel	Clock Generator EEPROM Page selection parameter ('0' selects Page 0 and '1' selects Page 1). Only matters when the Clock Generator is not in Fall-Back Mode.
kFreqSel	Parameter identifying the CDCE output frequency with SECREF(XTAL) as reference frequency: -- 0 -> 122.88MHz -- 1 -> 50MHz -- 2 -> 80MHz -- 3 -> 100MHz -- 4 -> 110MHz -- 5 -> 120MHz -- 6 -> 125MHz

Table 10. Clock Generator Configuration port description

Signal Name	Interface	Signal Type	Init State	Description
RefClk	-	I	N/A	100MHz input clock signal.
arRst	-	I	N/A	Active high reset (can be synchronously asserted but asynchronously de-asserted).
rInitConfigDoneClockGen	-	O	N/A	Clock Generator configuration done successfully signal.
aCG_PLL_Lock	-	I	N/A	Clock Generator PLL lock signal sent via the GPIO1 or GPIO4 port.
rPLL_LockClockGen	-	O	N/A	Clock Generator PLL lock signal sent via the GPIO1 or GPIO4 port and synchronized in the RefClk domain.
rConfigADCEnable	-	O	N/A	rConfigADCEnable is used to hold the ConfigADC module in reset until the Clock Generator is configured and locked.
REFSEL	-	O	Z	Clock Generator reference selection signal (low selects SECREF (XTAL) and high selects PRIREF

				(FPGA)). When High-Z selection is determined by register value.
HW_SW_CTRL	-	O	Z	Clock Generator EEPROM Page selection signal (low selects Page 0 and high selects Page 1). When High-Z selection is determined by register value.
rPDNout_n	-	O	L	Clock Generator power down signal. Resets the internal circuitry and is used in the initial power-up sequence.
s_scl_i	I2C	I	Z	I2C Serial Clock Input from 3-state buffer.
s_scl_o	I2C	O	L	I2C Serial Clock Output to 3-state buffer.
s_scl_t	I2C	T	N/A	I2C Serial Clock Output Enable to 3-state buffer.
s_sda_i	I2C	I	Z	I2C Serial Data Input from 3-state buffer.
s_sda_o	I2C	O	L	I2C Serial Data Output to 3-state buffer.
s_sda_t	I2C	T	N/A	I2C Serial Data Output Enable to 3-state buffer.

5   IP Top-Level Parameter Description

Table 11. IP core parameter descriptions.

Signal Name	Description
kZmodID	Parameter identifying the targeted Zmod. See Table 13 for more details.
kADC_Width	ADC resolution (number of bits). It is automatically computed based on *kZmodID*.
kExtCalibEn	Enables the external calibration interface. Set to "true" when the IP core is expected to be interfaced with the processing system through a high level IP which will send the calibration constants to the Zmod Digitizer Controller. Set to "false" when the core operates in stand alone mode.
kExtCmdInterfaceEn	Enables the upper layer IP SPI configuration interface. Set to "true" when the IP core is expected to be interfaced with the processing system through a high level IP. This will enable the processor to access the Zmod Digitizer SPI interface and to change the ADC settings if desired. Set to "false" when initial configuration described in Section 4.6 is sufficient.
kCh1HgMultCoefStatic[17:0]	Channel1 high gain multiplicative calibration coefficient. If the *kExtCalibEn* parameter is set to "false", the ADC Calibration block will expect the value of the multiplicative coefficient to be passed through this parameter. If the value of *kExtCalibEn* parameter is "true", this parameter is ignored, and the high level IP is expected to update the corresponding external port.
kCh1HgAddCoefStatic[17:0]	Channel1 high gain additive calibration coefficient. If the *kExtCalibEn* parameter is set to "false", the ADC Calibration block will expect the value of the additive coefficient to be passed through this parameter. If the value of *kExtCalibEn* parameter is "true", this parameter is ignored, and the high level IP is expected to update the corresponding external port.

|	
kCh2HgMultCoefStatic[17:0]	Channel2 high gain multiplicative calibration coefficient. If the *kExtCalibEn* parameter is set to "false", the ADC Calibration block will expect the value of the multiplicative coefficient to be passed through this parameter. If the value of *kExtCalibEn* parameter is "true", this parameter is ignored, and the high level IP is expected to update the corresponding external port.
kCh2HgAddCoefStatic[17:0]	Channel2 high gain additive calibration coefficient. If the *kExtCalibEn* parameter is set to "false", the ADC Calibration block will expect the value of the additive coefficient to be passed through this parameter. If the value of *kExtCalibEn* parameter is "true", this parameter is ignored, and the high level IP is expected to update the corresponding external port.
kCG_SimulationConfig	Clock Generator I2C shortened config for simulation purposes.
kCG_SimulationCmdTotal	Clock Generator I2C shortened configuration number of commands to send over I2C for simulation purposes, range should have been 0 to kCDCE_RegNrZeroBased := kCDCE_RegNrZeroBased, however Vivado IP GUI does not accept expressions.
kCGI2C_Addr	Clock Generator 7-bit I2C config address (0x67 (Fall-Back Mode), 0x68(Default Mode), 0x69).
kRefSel	Clock Generator input reference clock selection parameter ('0' selects SECREF(XTAL) and '1' selects PRIREF(FPGA)). Only relevant when the Clock Generator is not in Fall-Back Mode.
kHwSwCtrlSel	Clock Generator EEPROM Page selection parameter ('0' selects Page 0 and '1' selects Page 1). Only relevant when the Clock Generator is not in Fall-Back Mode.

kCDCEFreqSel	Parameter identifying the CDCE output frequency with SECREF(XTAL) as reference frequency, range should have been 0 to CDCE_I2C_Cmds'length, however Vivado IP GUI does not accept expressions:
	    -- 0 -> 122.88MHz
	    -- 1 -> 50MHz
	    -- 2 -> 80MHz
	    -- 3 -> 100MHz
	    -- 4 -> 110MHz
	    -- 5 -> 120MHz
	    -- 6 -> 125MHz

6   IP Top-Level Port Description
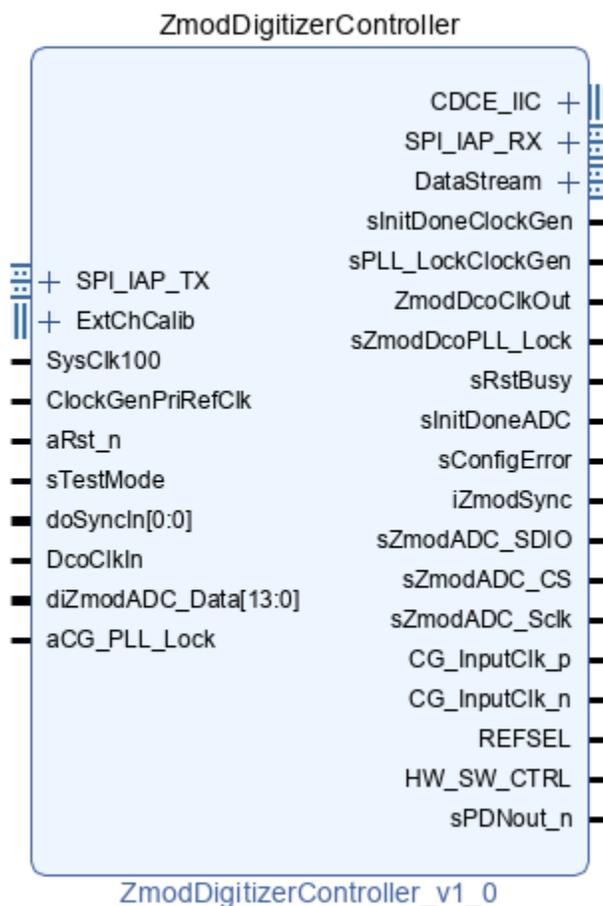
Figure 5: Zmod Digitizer Controller IP

Table 12. IP core port description

Signal Name	Interface	Signal Type	Init State	Description
SysClk100	-	I	N/A	100MHz input clock signal.
DcoClkIn	-	I	N/A	Sampling clock. The frequency range supported is between 10MHz and the maximum frequency supported by the Zmod/target FPGA clock distribution network. Clock generated by the ADC synchronous with diZmodADC_Data [4].
ClockGenPriRefClk	-	I	N/A	Clock Generator primary input reference clock signal. By default, the Clock Generator uses its secondary input reference clock. This input clock signal is completely optional.
sInitDoneClockGen	-	O	L	Clock Generator config done successful signal.
sPLL_LockClockGen	-	O	L	Clock Generator PLL lock signal sent via the GPIO1 or GPIO4 port and synchronized in the SysClock100 domain.

aRst_n	-	I	N/A	Asynchronous reset of negative polarity which resets the logic in all four clock domains. Must be asserted for at least 2*$T_{slowest}$- see section 4.2 for details.
sRstBusy	-	O	N/A	Reset busy flag. While this signal is asserted it is not recommended to apply a new reset to the IP. The user/upper level IP must wait for this signal to de-assert in order to apply a new reset.
sInitDoneADC	-	O	N/A	Flag indicating when the Zmod's ADC initialization is complete.
sConfigError	-	O	N/A	This flag is asserted if the ADC initialization fails. An IP reset is required to clear this flag.
sEnableAcquisition	-	I	N/A	When logic '1', this signal enables data acquisition from the ADC. This signal should be kept in logic '0' until the downstream IP (e.g. DMA controller) is ready to receive the ADC data. Once this signal has been set to logic '1', it should be kept in this state and never de-asserted.
doDataAxisTdata[31:0]	Data interface	O	N/A	(Master) AXI Stream Data interface TDATA port. Channel1 and Channel2 data are concatenated as follows: Channel1 -> doDataAxisTdata[31:16]. Channel2 -> doDataAxisTdata[15:0].
doDataAxisTvalid	Data interface	O	N/A	(Master) AXI Stream Data interface TVALID signal.
doDataAxisTready	Data interface	I	N/A	(Master) AXI Stream Data interface TREADY signal.
doExtCh1HgMultCoef[17:0]	ExtChCalib	I	N/A	Channel1 high gain multiplicative coefficient external port. This port is enabled by setting the kExtCalibEn parameter to "true".
doExtCh1HgAddCoef[17:0]	ExtChCalib	I	N/A	Channel1 high gain additive coefficient external port. This port is enabled by setting the kExtCalibEn parameter to "true".
doExtCh2HgMultCoef[17:0]	ExtChCalib	I	N/A	Channel2 high gain multiplicative coefficient external port. This port is enabled by setting the kExtCalibEn parameter to "true".
doExtCh2HgAddCoef[17:0]	ExtChCalib	I	N/A	Channel2 high gain additive coefficient external port. This port is enabled by setting the kExtCalibEn parameter to "true".
sTestMode	-	I	N/A	sTestMode is used to bypass the calibration block. When asserted, raw samples are provided on the output Data interface.
sCmdTxAxisTdata[31:0]	SPI IAP	I	N/A	IAP command TX interface (Slave AXI Stream) data port. For more information see section 4.6.
sCmdTxAxisTvalid	SPI IAP	I	N/A	IAP command TX interface (Slave AXI Stream) valid signal. For more information see section 4.6.

sCmdTxAxisTready	SPI IAP	O	N/A	IAP command TX interface (Slave AXI Stream) ready signal. For more information see section 4.6.
sCmdRxAxisTdata[31:0]	SPI IAP	O	N/A	IAP command RX interface (Master AXI Stream) data port. For more information see section 4.6.
sCmdRxAxisTvalid	SPI IAP	O	N/A	IAP command RX interface (Master AXI Stream) valid signal. For more information see section 4.6.
sCmdRxAxisTready	SPI IAP	I	N/A	IAP command RX interface (Master AXI Stream) ready signal. For more information see section 4.6.
CG_InputClk_p	Zmod	O	N/A	Clock Generator positive differential primary clock input. For more details see [1].
CG_InputClk_n	Zmod	O	N/A	Clock Generator negative differential primary clock input. For more details see [1].
aCG_PLL_Lock	Zmod	I	N/A	Clock Generator PLL lock signal sent via the GPIO1 or GPIO4 port.
aREFSEL	Zmod	O	N/A	Clock Generator reference selection signal ('0' selects SECREF(XTAL) and '1' selects PRIREF(FPGA)).
aHW_SW_CTRL	Zmod	O	N/A	Clock Generator EEPROM Page selection signal.
sPDNout_n	Zmod	O	L	Clock Generator power down signal, passthrough output.
ZmodDcoClkOut	-	O	N/A	De-skewed version of DcoClkIn through MMCM and BUFG primitives.
sZmodDcoPLL_Lock	-	O	L	Lock signal of the MMCM used for de-skewing purposes. Signals that ZmodDcoClkOut is stable.
diZmodADC_Data[kADC_Width-1 : 0]	Zmod	I	N/A	kADC_Width bit wide DDR parallel data bus exported by ADC containing Channel1 and Channel 2 interleaved samples [4].
sZmodADC_SDIO	Zmod	IO	N/A	SPI SDIO signal [4].
sZmodADC_CS	Zmod	O	N/A	SPI CS signal [4].
sZmodADC_Sclk	Zmod	O	N/A	SPI output clock [4].
s_scl_i	CDCE IIC	I	Z	I2C Serial Clock Input from 3-state buffer.
s_scl_o	CDCE IIC	O	L	I2C Serial Clock Output to 3-state buffer.
s_scl_t	CDCE IIC	T	N/A	I2C Serial Clock Output Enable to 3-state buffer.
s_sda_i	CDCE IIC	I	Z	I2C Serial Data Input from 3-state buffer.
s_sda_o	CDCE IIC	O	L	I2C Serial Data Output to 3-state buffer.
s_sda_t	CDCE IIC	T	N/A	I2C Serial Data Output Enable to 3-state buffer.

7   IP Core customization

The customization parameters allow:

1.   Selecting the Zmod Digitizer targeted through the *kZmodID* parameter. The *kZmodID* parameter also configures the *kADC_Width* generic. Table 13 details how kZmodID should be configured.

Table 13: kZmodID configuration

Zmod	ADC	kZmodID	kADC_Width	kSamplingPeriod (max)
Zmod Scope 1430 - 125	AD9648	6	14	8000 (125 MSPS)

2.   Enabling/disabling the external SPI IAP interface through the *kExtCmdInterfaceEn* parameter. The SPI IAP interface needs to be enabled (*kExtCmdInterfaceEn* = true) to modify the ADC's configuration registers after initialization.
3.   Enabling/disabling dynamic external calibration (*kExtCalibEn*).
4.   Selecting the ADC Sampling frequency from a set of supported frequencies that also have calibration coefficients written into the Zmod's EEPROM through the *kCDCEFreqSel* parameter.

Table 14: ADC Sampling Clock Frequency Selection

Sampling Frequency	kCDCEFreqSel	kSamplingPeriod
122.88 MHz	0	8138 ps
50 MHz	1	20000 ps
80 MHz	2	12500 ps
100 MHz	3	10000 ps
110 MHz	4	9090 ps
120 MHz	5	8333 ps
125 MHz	6	8000 ps

5.   Changing CDCE6214-Q1 I2C address (*kCGI2C_Addr*) and thus potentially changing the Clock Generator operating mode, the primary mode is called Fall-Back Mode [1] where the I2C interface is always enabled, the other modes involve its configuration being read from an internal factory programmed 2 Page EEPROM which can have frequency configurations unsupported by this IP.
6.   Changing CDCE6214-Q1 clock reference selection between SECREF (XTAL) and PRIREF (generated inside the FPGA).
7.   Selecting CDCE6124-Q1 internal EEPROM Page (Page 0 has a factory configuration that disables the I2C serial interface).

8   References

The following documents provide additional information on the subjects discussed:

1. Texas Instruments, CDCE6214-Q1 Datasheet, SNAS786B, Rev B.
2. Xilinx Inc., *UG472: 7 Series FPGAs Clocking Resources*, v1.6, October 2, 2012.
3. Xilinx Inc., *UG471: 7 Series FPGAs SelectIO Resources*, v1.4, May 13, 2014.
4. Analog Devices, AD9648 Datasheet, Rev C.