1300 Henley Court
Pullman, WA 99163
509.334.6306
www.digilentinc.com

Zmod Scope Controller IP Core User Guide

Revised March 22, 2022; Author Tudor Gherman

1 Introduction

This user guide describes the Digilent **Zmod Scope Controller** Intellectual Property. This IP interfaces directly with one of the Zmod Scope 1410 - 105, Zmod Scope 1010 - 40, Zmod Scope 1010 - 125, Zmod Scope 1210 – 40, Zmod Scope 1210 – 125, Zmod Scope 1410 - 40, or the Zmod Scope 1410 – 125 configuring the gain and coupling select relays, writing an initial configuration to the analog to digital converter (ADC) featured by these modules, demultiplexing the data received over the ADC's parallel interface and forwarding it to the user logic. The **Zmod Scope 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.

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 and AD92xx SPI interface; **Yes** for relays; **No** for he IP itself (it can be simulated using the design fies)
Constraints file	XDC
Software driver	**N**/A
Tested design flows	
Design entry	Vivado™ Design Suite 2019.1
Synthesis	Vivado Synthesis 2019.1

2 Features

- Initializes the hardware on the Zmod Scope 1410 - 105, Zmod Scope 1010 - 40, Zmod Scope 1010 - 125, Zmod Scope 1210 – 40, Zmod Scope 1210 – 125, Zmod Scope 1410 - 40, or the Zmod Scope 1410 – 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 gain and coupling select relays.
- Performs offset and gain calibration based on coefficients specified by the user/upper level IPs.

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 can be configured to be between 10 to 16 bit wide, while the sampling rate can be configured between 10 and 125 MSPS. The **Zmod Scope Controller** further exports to the user logic two distinct SDR channels synchronized in the ADC sampling clock domain. The latency added on the data path by this IP is of 13 sampling clock cycles

+1 sapling clock cycle uncertainty. The value of the latency obtained is mostly caused by the FIFO used to synchronize the incoming samples in the sampling clock domain. The 1 sampling clock cycle uncertainty is also introduced by the FIFO [1].

The Zmod Scope synchronization input (SYNC) signal is also generated by **the Zmod Scope Controller** in the input clock domain (*ADC_InClk*), allowing the ADC's sampling instant to be determined with a resolution equal to the input clock period. The relation between the input clock frequency (F_{ADC_InClk}) and the sampling clock frequency (F_{sample}) is determined by the *ADC_ClkDiv* parameter (and is further discussed in section 4.3):

$$F_{sample} = \frac{F_{ADC\_InClk}}{ADC\_ClkDiv} \quad (1)$$

4 Overview

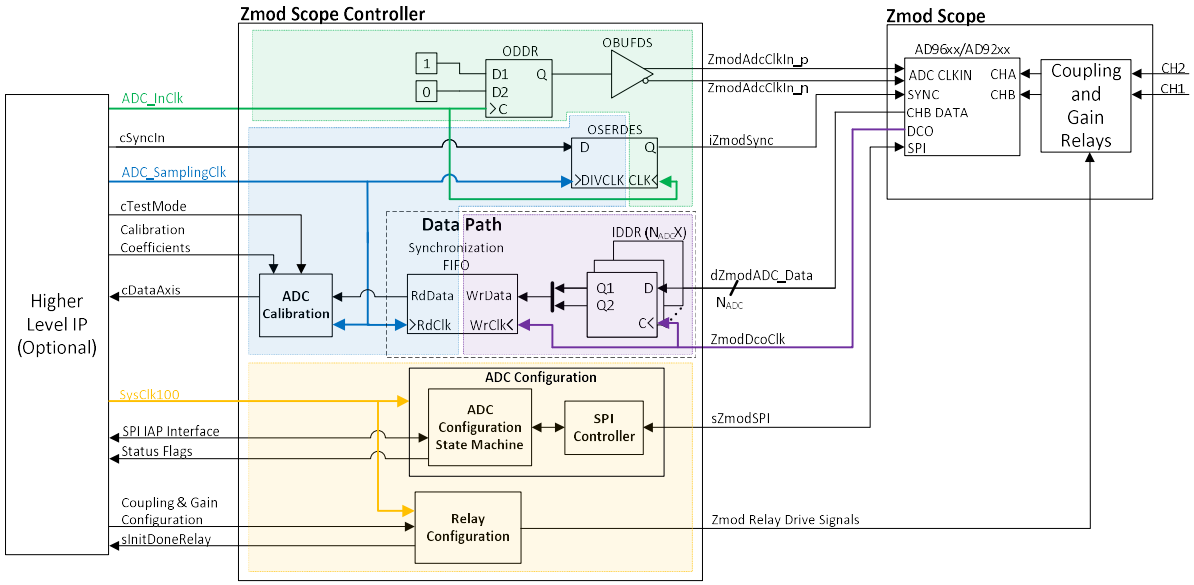

Figure 1. Zmod Scope Controller block diagram.

The structure of the IP is presented in Figure 1. The main functionalities are divided as ADC input clock generation, SYNC 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 relay configuration.

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 relay configuration module 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 sampling clock (*ADC_SamplingClk*) clocks the synchronization FIFO read port (of the Data Path module) and the ADC Calibration module. All signals synchronous with *ADC_SamplingClk* have the prefix "c".
3. The ADC input clock (*ADC_InClk*) is used to clock the primitives responsible with generating the SYNC signal and with passing the input clock to the Zmod Scope. The relation between the ADC input clock and the ADC sampling clock is described by equation (1). All signals synchronous with *ADC_InClk* have the prefix "i".
4. The DCO clock (*ZmodDcoClk*) is generated by the ADC and is used to clock the IDDR primitives that demultiplex the ADC incoming DDR data bus and the write port of the data FIFO in the Data Path module. All signals synchronous with the de-skewed version of *ZmodDcoClk* have the prefix "d".

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 Scope Controller**.

4.3 ADC Input Clock Generation

The ADC ICs (AD96xx and AD92xx) on the supported Zmod Scope modules require a differential clock as input. This IP allows a wide range of frequencies for this clock. The minimum frequency is limited to 10 MHz, while for the maximum frequency the limit is imposed by the FPGA's clocking network limitations [2] and by the ADC capabilities. The relation between the ADC input clock frequency and the sampling clock frequency defined by equation (1) must be respected. This IP does not generate these clocks internally (it only instantiates the IO primitives required to generate the differential clock forwarded to the ADC as input), instead it requires them as an input (*ADC_ClkIn* and *ADC_SamplingClk*), expecting them to be generated in the top-level design. It is the user's responsibility to properly configure the ADC input clock and sampling clock frequencies. The maximum tested input clock frequency is 400MHz with a value of 4 for the *ADC_ClkDiv* parameter.

4.4 SYNC Generation

The SYNC signal allows multiple ADC devices (AD96xx, AD92xx) to have their sample clock synchronized by resetting their internal clock dividers. An OSERDES primitive [3] with a ratio between the divided input clock and the high-speed clock of *ADC_ClkDiv* is used by the IP core to generate the SYNC

signal. The *kExtSyncEn* parameter enables the user to control the SYNC signal through the optional cSyncIn port which represents the *ADC_ClkDiv* bit wide parallel input of the OSERDES primitive. Only one of the bits of the cSyncIn port is allowed to be '1'. For example, with *ADC_ClkDiv* set to 4, the valid values that the cSyncIn signal can take are "0001", "0010", "0100", "1000". If the cSyncIn port is disabled, the default value of the OSERDES input will be "0…01".

The SYNC signal can also be used for oversampling purposes.

4.5 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. Each channel is further connected to a shallow FIFO used only to synchronize the received data with the IP core's system clock.

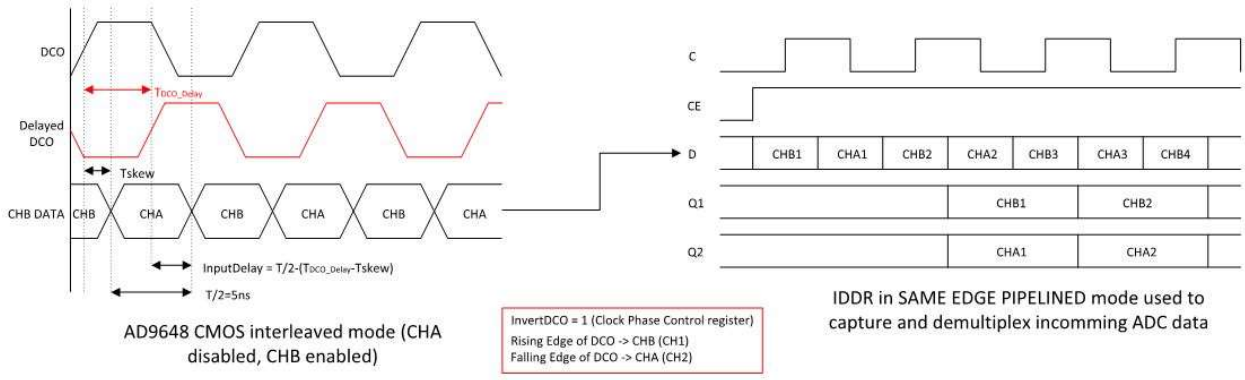

Figure 2. Input data channel demultiplexing

The synchronization FIFO is not designed to store samples. The upper levels should always be able to accept incoming samples. The output of this IP should be processed in real time.

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 synchronization FIFO write enable is conditioned by the ADC and relay initialization completion, making sure no corrupt samples are loaded into the FIFO or read by the user. After any relay state modification, the FIFO is reset, thus it is the user's responsibility to make sure all relevant samples received before the relay state change request have been processed. 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
ADC_SamplingClk	-	I	N/A	ADC sampling clock.
acRst_n	-	I	N/A	Active low reset (can be asynchronously asserted but synchronously de-asserted in the ADC sampling clock domain).
DcoClkIn	-	I	N/A	Connected directly to the Zmod's AD92xx/AD96xx DCO output clock.
DcoClkOut	-	O	N/A	The Zmod's AD92xx/AD96xx DCO output clock forwarded to the IP's top-level module (the de-skew block output).
dEnableAcquisition	-	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.
dADC_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.
cChannelA[kADC_Width-1 : 0]	-	O	N/A	The demultiplexed ADC's Channel A data output synchronized in the ADC sampling clock domain.
cChannelB[kADC_Width-1 : 0]	-	O	N/A	The demultiplexed ADC's Channel B data output synchronized in the ADC sampling clock domain.
cDataOutValid	-	O	N/A	Channel A & B data valid indicator.
cFIFO_RdEn	-	I	N/A	Synchronization FIFO read enable signal. Expected to be asserted by the module's upper levels.
dFIFO_WrRstBusy	-	O	N/A	Signal indicating when it is safe to assert *acRst_n* (when *dFIFO_WrRstBusy* is '1', it is not safe to assert *acRst_n*)
dDataOverflow	-	O	N/A	*dDataOverflow* indicates that the shallow synchronization FIFO in the Data Path module is full and an additional write operation has been attempted. There are two cases in which this signal is asserted: 1. The ratio between the ADC input clock and ADC sampling clock frequencies is different from *kADC_ClkDiv*. 2. The upper levels cannot accept data (*cDataAxisTready* is not asserted, resulting in *cFIFO_RdEn being also de-asserted*). However, this IP is not designed to store data, the upper levels should always be able to

				accept incoming samples. The output of this IP should be processed in real.
cInitDone	-	I	N/A	Input indicating when both the ADC and relay initialization is complete. This signal is synchronized with the ADC sampling clock.
dInitDone	-	I	N/A	Input indicating when both the ADC and relay initialization is complete. This signal is synchronized with the DCO clock.

4.6 ADC Calibration (GainOffsetCalib.vhd)

The analog front end of the Zmod Scope 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.
kLgMultCoefStatic[17:0]	Low 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.
kLgAddCoefStatic[17:0]	Low 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.
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 10MHz 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 Scope 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 (for low range: ±1V) or
  - 26.25 (for high range: ±25V)

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

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

While the real range of the Zmod Scope is the one defined above, the specified (ideal) range of the product is +-25V for the low gain option and +-1V for the high gain option. Having the calibration applied, the outputs of the **Zmod Scope Controller** should be those of an ideal ADC having these ideal ranges:

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

were $Range_{ideal}$ is 1 for the high gain option and 25 for the low gain option. From (2) and (3) one can obtain:

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

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

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

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:

$$cDataCalib = cDataRaw * CoefMult + CoefAdd \quad (6)$$

where:

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

$$CoefAdd = \frac{CA}{Range_{ideal}} \quad (8)$$

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:

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

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

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.

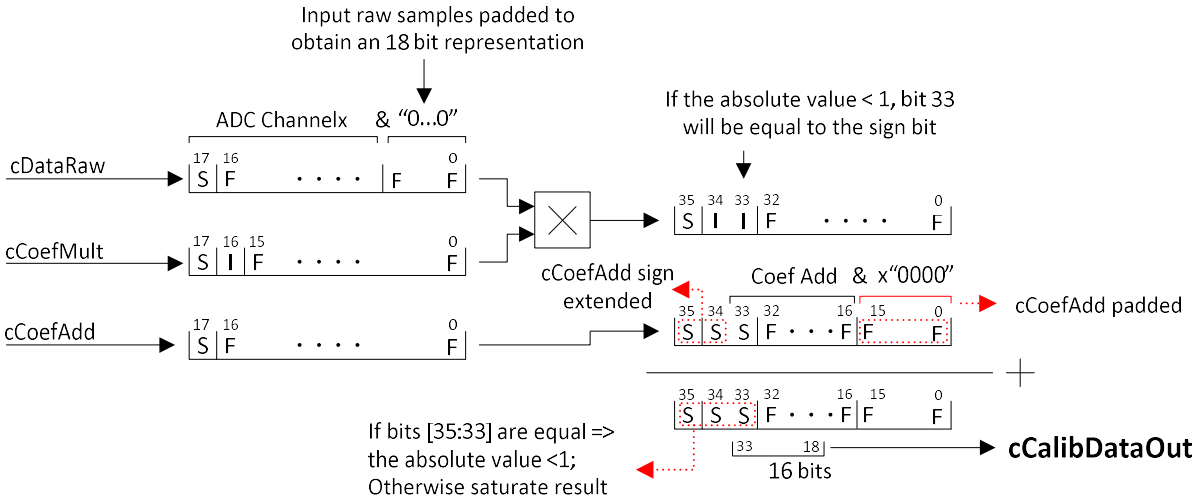

Figure 3: Calibration process

The multiplicative and additive coefficients are passed by the upper layer IP Core if the **Zmod Scope 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.

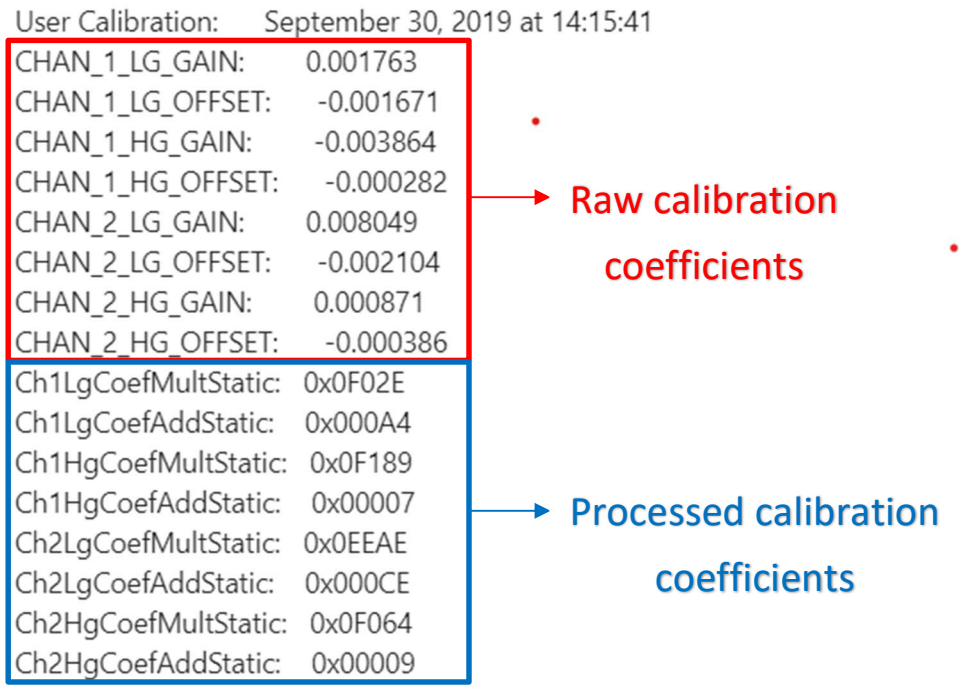

Figure 4: Obtaining calibration parameters from decutil

4.7 Relay Configuration (ConfigRelays.vhd)

The **Zmod Scope Controller** IP core provides the choice of selecting the coupling and gain options statically or dynamically for the supported Zmod Scope modules. For static configuration, the *kExtRelayConfigEn* needs to be set as "false" and the relay configuration parameters (See *Table 5*) need to be configured in the IP core GUI. For dynamic relay control, the *kExtRelayConfigEn* needs to be set as "true" and the configuration state machine will constantly monitor the external relay configuration ports and will reconfigure the AC/DC coupling select relays or the gain relays in response to user request.

The relay configuration logic is implemented as a distinct VHDL module having its parameters and ports described below:

Table 5. Relay configuration parameter description

Parameter Name	Description
kExtRelayConfigEn	Enables the external relay configuration port. Set to "true" when dynamic relay configuration is required. Set to "false" when static relay configuration is sufficient.
kCh1CouplingStatic	Channel1 AC DC coupling select static configuration parameter. If the *kExtRelayConfigEn* parameter is set to "false", the relay configuration state machine will configure the coupling select relay according to this parameter's value at initialization time. The state of the relay cannot be changed afterwards. If the value of *kExtRelayConfigEn* parameter is "true", this parameter is ignored. • 1 = DC coupling. • 0 = AC coupling.

kCh2CouplingStatic	Channel2 AC DC coupling select static configuration parameter. If the *kExtRelayConfigEn* parameter is set to "false", the relay configuration state machine will configure the coupling select relay according to this parameter's value at initialization time. The state of the relay cannot be changed afterwards. If the value of *kExtRelayConfigEn* parameter is "true", this parameter is ignored. • 1 = DC coupling. • 0 = AC coupling.
kCh1GainStatic	Channel1 gain select static configuration parameter. If the *kExtRelayConfigEn* parameter is set to "false", the relay configuration state machine will configure the gain select relay according to this parameter's value at initialization time. The state of the relay cannot be changed afterwards. If the value of *kExtRelayConfigEn* parameter is "true", this parameter is ignored. • 1 = High Gain. • 0 = Low Gain.
kCh2GainStatic	Channel2 gain select static configuration parameter. If the *kExtRelayConfigEn* parameter is set to "false", the relay configuration state machine will configure the gain select relay according to this parameter's value at initialization time. The state of the relay cannot be changed afterwards. If the value of *kExtRelayConfigEn* parameter is "true", this parameter is ignored. • 1 = High Gain. • 0 = Low Gain.

Table 6. Relay 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).
sCh1CouplingConfig	-	I	N/A	Channel1 AC DC coupling select configuration port. • 1 = DC coupling (relay set). • 0 = AC coupling (relay reset).
sCh2CouplingConfig	-	I	N/A	Channel2 AC DC coupling select external port. • 1 = DC coupling (relay set). • 0 = AC coupling (relay reset).
sCh1GainConfig	-	I	N/A	Channel1 gain select configuration port. • 1 = High Gain (relay set). • 0 = Low Gain (relay reset).
sCh2GainConfig	-	I	N/A	Channel2 gain select configuration port. • 1 = High Gain (relay set). • 0 = Low Gain (relay reset).
sCh1CouplingState	-	O	N/A	Output reflecting the current state of the channel 1 coupling select relay. • 1 = DC coupling (relay set). • 0 = AC coupling (relay reset).

sCh2CouplingState	-	O	N/A	Output reflecting the current state of the channel 2 coupling select relay. • 1 = DC coupling (relay set). • 0 = AC coupling (relay reset).
sCh1GainState	-	O	N/A	Output reflecting the current state of the channel 1 gain select relay. This output is used by the ADC Calibration block to determine the correct set of parameters to be applied for the gain and offset calibration process.
sCh2GainState	-	O	N/A	Output reflecting the current state of the channel 2 gain select relay. This output is used by the ADC Calibration block to determine the correct set of parameters to be applied for the gain and offset calibration process.
sInitDoneRelay	-	O	N/A	Flag indicating when the Zmod's relay initialization is complete. Whenever one of the Zmod's relays is requested to change state by one of the configuration ports, *sInitDoneRelay* is de-asserted until the relay is configured in the requested state.
sInitDoneRelayPush	-	O	N/A	The Data Path module uses the *sInitDoneRelay* signal to control the synchronization FIFO write enable and reset behavior. However, *sInitDoneRelay* needs to be pushed to the ADC sampling clock domain. This signal represents the HandshakeData clock domain crossing module (used for this purpose) *iPush* signal.
sInitDoneRelayRdy	-	I	N/A	HandshakeData clock domain crossing iRdy signal used to determine when sInitDoneRelay has propagated to the ADC sampling clock domain. The relay configuration can safely begin only after the sInitDoneRelay has propagated in the destination clock domain and write enable of the Data Path module's FIFO has been disabled.
sCh1CouplingH	Zmod	O	N/A	Channel1 AC DC coupling select relay driver control input. Connected directly to the corresponding Zmod port.
sCh1CouplingL	Zmod	O	N/A	Channel1 AC DC coupling select relay driver control input. Connected directly to the corresponding Zmod port.
sCh2CouplingH	Zmod	O	N/A	Channel2 AC DC coupling select relay driver control input. Connected directly to the corresponding Zmod port.
sCh2CouplingL	Zmod	O	N/A	Channel2 AC DC coupling select relay driver control input. Connected directly to the corresponding Zmod port.

sCh1GainH	Zmod	O	N/A	Channel1 gain select relay driver control input. Connected directly to the corresponding Zmod port.
sCh1GainL	Zmod	O	N/A	Channel1 gain select relay driver control input. Connected directly to the corresponding Zmod port.
sCh2GainH	Zmod	O	N/A	Channel2 gain select relay driver control input. Connected directly to the corresponding Zmod port.
sCh2GainL	Zmod	O	N/A	Channel2 gain select relay driver control input. Connected directly to the corresponding Zmod port.
sRelayComH	Zmod	O	N/A	Common relay terminal driver control input. Connected directly to the corresponding Zmod port.
sRelayComL	Zmod	O	N/A	Common relay terminal driver control input. Connected directly to the corresponding Zmod port.

4.8 ADC Configuration (ConfigADC.vhd)

The ADC Configuration block sends a predefined list of SPI commands to the Zmod Scope, 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 7. ADC Configuration parameter description

Parameter Name	Description
kZmodID	Parameter identifying the targeted Zmod. See Table 13 for more details.
kADC_ClkDiv	ADC Clock divider ratio. This parameter is passed to the ADC's Clock Divide register (address 0x0B for AD96xx and AD92xx) as part of the initialization sequence. *kADC_ClkDiv* also determines the ratio between the frequency of the ADC sampling clock and the ADC input clock as described by equation (1). However, it is the user's responsibility to correctly configure the frequency of these clock sources.
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. 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 4.9.
sADC_SDIO	SPI	IO	N/A	2 Wire SPI interface SDIO signal. Signal managed by the SPI controller. More details in section 4.9.
sADC_CS	SPI	O	N/A	2 Wire SPI interface CS signal. Signal managed by the SPI controller. More details in section 4.9.
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.
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. In addition (not represented in the list below), for Zmods featuring AD92xx devices the Transfer Register is set after each write operation [4-9].

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).

4.9 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 9. 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 10. 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 [4-9] divided from *SysClk100*. Connected directly to the corresponding Zmod port.
sSDIO	SPI	IO	N/A	2 Wire SPI interface SDIO signal [4-9]. Connected directly to the corresponding Zmod port.
sCS	SPI	O	N/A	2 Wire SPI interface CS signal [4-9]. 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 [4-9]. 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 [4-9].

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.
kSamplingPeriod	The sampling clock period expressed in ps. The maximum value is limited to 100000, 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. The resolution of the sampling clock period is 1000 ps. For kSamplingPeriod values smaller than 10000 ps (i.e. sampling frequencies larger than 100 MHz), you must edit the IP timing constraints for timing closure to be achieved. Please read the "7 IP Core customization" section below for more details.
kADC_ClkDiv	ADC Clock divider ratio. This parameter is passed to the ADC's Clock Divide register (address 0x0B for AD96xx and AD92xx) as part of the initialization sequence. *kADC_ClkDiv* also determines the ratio between the frequency of the ADC sampling clock and the ADC input clock as described by equation (1). However, it is the user's responsibility to correctly configure the frequency of these clock sources.
kADC_Width	ADC resolution (number of bits). It is automatically computed based on *kZmodID*.
kExtRelayConfigEn	Enables the external relay configuration port. Set to "true" when dynamic relay configuration is required. Set to "false" when static relay configuration is sufficient.
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 Scope 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 Scope SPI interface and to change the ADC settings if desired. Set to "false" when initial configuration described in Section 4.8 is sufficient.
kExtSyncEn	Enables the cSync port. When set to "true" *cSyncIn* controls the *iZmodSync* output signal behavior. When set to "false" the *cSyncIn* port is disabled and *iZmodSync* has the default behavior (see section 4.4 for more details).
kCh1CouplingStatic	Channel1 AC DC coupling select static configuration parameter. If the *kExtRelayConfigEn* parameter is set to "false", the relay configuration state machine will configure the coupling select relay according to this parameter's value at initialization time. The state of the relay cannot be changed afterwards. If the value of *kExtRelayConfigEn* parameter is "true", this parameter is ignored. • 1 = DC coupling. • 0 = AC coupling.
kCh2CouplingStatic	Channel2 AC DC coupling select static configuration parameter. If the *kExtRelayConfigEn* parameter is set to "false", the relay configuration state machine will configure the coupling select relay according to this parameter's value at initialization time. The state of the relay cannot be changed afterwards. If the value of *kExtRelayConfigEn* parameter is "true", this parameter is ignored. • 1 = DC coupling. • 0 = AC coupling.
kCh1GainStatic	Channel1 gain select static configuration parameter. If the *kExtRelayConfigEn* parameter is set to "false", the relay configuration state machine will configure the gain select relay according to this parameter's value at initialization time. The state of the relay cannot be changed afterwards. If the value of *kExtRelayConfigEn* parameter is "true", this parameter is ignored. • 1 = High Gain. • 0 = Low Gain.
kCh2GainStatic	Channel2 gain select static configuration parameter. If the *kExtRelayConfigEn* parameter is set to "false", the relay configuration state machine will configure the gain select relay according to this parameter's value at initialization time. The state of the relay cannot be changed afterwards. If the value of *kExtRelayConfigEn* parameter is "true", this parameter is ignored. • 1 = High Gain. • 0 = Low Gain.
kCh1LgMultCoefStatic[17:0]	Channel1 low 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.
kCh1LgAddCoefStatic[17:0]	Channel1 low 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.

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.
kCh2LgMultCoefStatic[17:0]	Channel2 low 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.
kCh2LgAddCoefStatic[17:0]	Channel2 low 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.

6 IP Top-Level Port Description

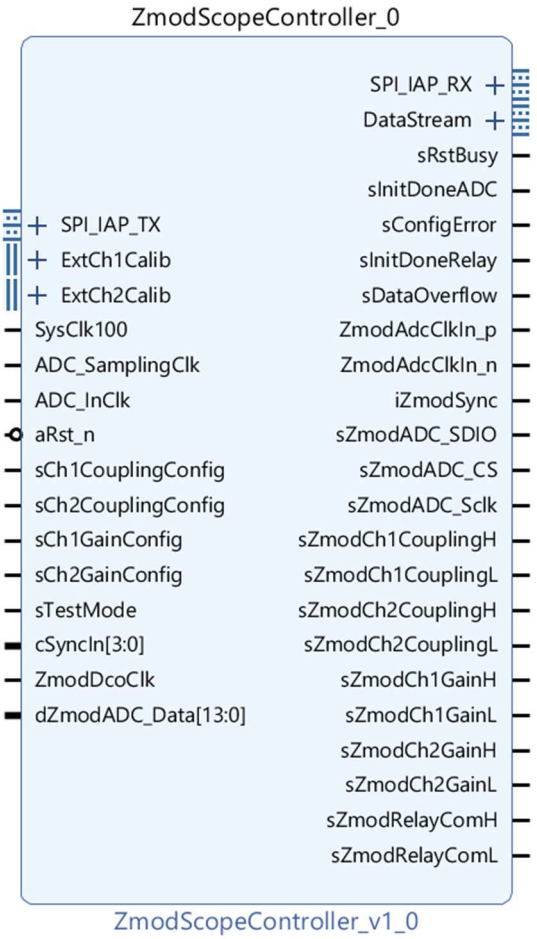

Figure 5: Zmod Scope Controller IP

Table 12. IP core port description

Signal Name	Interface	Signal Type	Init State	Description
SysClk100	-	I	N/A	100MHz input clock signal.
ADC_SamplingClk	-	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.
ADC_InClk	-	I	N/A	ADC input clock signal. The ratio between *ADC_InClk* and *ADC_SamplingClk* must be equal to *kADC_ClkDiv*.
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.
sInitDoneRelay	-	O	N/A	Flag indicating when the Zmod's relay initialization is complete. If relay dynamic configuration is enabled through the *kExtRelayConfigEn* parameter, *sInitDoneRelay* is de-asserted until the relay is configured in the requested state.
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.
sDataOverflow	-	O	N/A	Flag indicating that the shallow synchronization FIFO in the DataPath module is full. There are two cases in which this signal is asserted: 1. The ratio between the *ADC_InClk* and *ADC_SamplingClk* clock frequencies is different from *kADC_ClkDiv*. 2. The upper levels cannot accept data (*cDataAxisTready* is not asserted). This IP is not designed to store data, the upper levels should always be able to accept incoming samples. The output of this IP should be processed in real time.
cDataAxisTdata[31:0]	Data interface	O	N/A	(Master) AXI Stream Data interface TDATA port. Channel1 and Channel2 data are concatenated as follows: Channel1 -> cDataAxisTdata[31:16]. Channel2 -> cDataAxisTdata[15:0].
cDataAxisTvalid	Data interface	O	N/A	(Master) AXI Stream Data interface TVALID signal.
cDataAxisTready	Data interface	I	N/A	(Master) AXI Stream Data interface TREADY signal.
cExtCh1LgMultCoef[17:0]	ExtCh1Calib	I	N/A	Channel1 low gain multiplicative coefficient external port. This port is enabled by setting the *kExtCalibEn* parameter to "true".
cExtCh1LgAddCoef[17:0]	ExtCh1Calib	I	N/A	Channel1 low gain additive coefficient external port. This port is enabled by setting the *kExtCalibEn* parameter to "true".
cExtCh1HgMultCoef[17:0]	ExtCh1Calib	I	N/A	Channel1 high gain multiplicative coefficient external port. This port is enabled by setting the *kExtCalibEn* parameter to "true".
cExtCh1HgAddCoef[17:0]	ExtCh1Calib	I	N/A	Channel1 high gain additive coefficient external port. This port is enabled by setting the *kExtCalibEn* parameter to "true".

cExtCh2LgMultCoef[17:0]	ExtCh2Calib	I	N/A	Channel2 low gain multiplicative coefficient external port. This port is enabled by setting the *kExtCalibEn* parameter to “true”.
cExtCh2LgAddCoef[17:0]	ExtCh2Calib	I	N/A	Channel2 low gain additive coefficient external port. This port is enabled by setting the *kExtCalibEn* parameter to “true”.
cExtCh2HgMultCoef[17:0]	ExtCh2Calib	I	N/A	Channel2 high gain multiplicative coefficient external port. This port is enabled by setting the *kExtCalibEn* parameter to “true”.
cExtCh2HgAddCoef[17:0]	ExtCh2Calib	I	N/A	Channel2 high gain additive coefficient external port. This port is enabled by setting the *kExtCalibEn* parameter to “true”.
sCh1CouplingConfig	-	I	N/A	Channel1 AC DC coupling select external port. This port is enabled by setting the k*ExtRelayConfigEn* parameter to “true”. • 1 = DC coupling. • 0 = AC coupling.
sCh2CouplingConfig	-	I	N/A	Channel2 AC DC coupling select external port. This port is enabled by setting the *kExtRelayConfigEn* parameter to “true”. • 1 = DC coupling. • 0 = AC coupling.
sCh1GainConfig	-	I	N/A	Channel1 gain select external port. This port is enabled by setting the *kExtRelayConfigEn* parameter to “true”. • 1 = High Gain. • 0 = Low Gain.
sCh2GainConfig	-	I	N/A	Channel2 gain select external port. This port is enabled by setting the *kExtRelayConfigEn* parameter to “true”. • 1 = High Gain. • 0 = Low Gain.
sTestMode	-	I	N/A	*sTestMode* is used to bypass the calibration block. When asserted, raw samples are provided on the output Data interface.
cSyncIn	-	I	N/A	*SYNC* signal control input (functionality described in section 4.4). This port is enabled by setting the *ExtSyncEn* parameter to “true”.
sCmdTxAxisTdata[31:0]	SPI IAP	I	N/A	IAP command TX interface (Slave AXI Stream) data port. For more information see section 4.8.
sCmdTxAxisTvalid	SPI IAP	I	N/A	IAP command TX interface (Slave AXI Stream) valid signal. For more information see section 4.8.
sCmdTxAxisTready	SPI IAP	O	N/A	IAP command TX interface (Slave AXI Stream) ready signal. For more information see section 4.8.
sCmdRxAxisTdata[31:0]	SPI IAP	O	N/A	IAP command RX interface (Master AXI Stream) data port. For more information see section 4.8.
sCmdRxAxisTvalid	SPI IAP	O	N/A	IAP command RX interface (Master AXI Stream) valid signal. For more information see section 4.8.

sCmdRxAxisTready	SPI IAP	I	N/A	IAP command RX interface (Master AXI Stream) ready signal. For more information see section 4.8.
ZmodAdcClkIn_p	Zmod	O	N/A	ADC positive differential clock input. For more details see [4-9].
ZmodAdcClkIn_n	Zmod	O	N/A	ADC negative differential clock input. For more details see [4-9].
iZmodSync	Zmod	O	N/A	Synchronization signal connected to the ADC SYNC input. For more details see [4-9].
ZmodDcoClk	Zmod	I	N/A	Clock generated by the ADC synchronous with dADC_Data [4-9].
dZmodADC_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 [4-9].
sZmodADC_SDIO	Zmod	IO	N/A	SPI SDIO signal [4-9].
sZmodADC_CS	Zmod	O	N/A	SPI CS signal [4-9].
sZmodADC_Sclk	Zmod	O	N/A	SPI output clock [4-9].
sZmodCh1CouplingH	Zmod	O	N/A	Channel1 AC DC coupling select relay driver control input.
sZmodCh1CouplingL	Zmod	O	N/A	Channel1 AC DC coupling select relay driver control input.
sZmodCh2CouplingH	Zmod	O	N/A	Channel2 AC DC coupling select relay driver control input.
sZmodCh2CouplingL	Zmod	O	N/A	Channel2 AC DC coupling select relay driver control input.
sZmodCh1GainH	Zmod	O	N/A	Channel1 gain select relay driver control input.
sZmodCh1GainL	Zmod	O	N/A	Channel1 gain select relay driver control input.
sZmodCh2GainH	Zmod	O	N/A	Channel2 gain select relay driver control input.
sZmodCh2GainL	Zmod	O	N/A	Channel2 gain select relay driver control input.
sZmodRelayComH	Zmod	O	N/A	Common relay terminal driver control input.
sZmodRelayComL	Zmod	O	N/A	Common relay terminal driver control input.

7 IP Core customization

The customization parameters allow:

1. Selecting the Zmod Scope 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 (min)
Zmod Scope 1410 - 105	AD9648	0	14	9524 (105MSPS)

Zmod Scope 1010 - 40	AD9204	1	10	25000 (40MSPS)
Zmod Scope 1010 - 125	AD9608	2	10	8000 (125MSPS)
Zmod Scope 1210 – 40	AD9231	3	12	25000 (40MSPS)
Zmod Scope 1210 – 125	AD9628	4	12	8000 (125MSPS)
Zmod Scope 1410 - 40	AD9251	5	14	25000 (40MSPS)
Zmod Scope 1410 - 125	AD9648	6	14	8000 (125MSPS)

2. Selecting the sampling rate through the kSamplingPeriod parameter. The kSamplingPeriod parameter should be configured with the sampling period expressed in ps. It is the user's responsibility to correctly configure this parameter based on the project requirements and the Zmod used (limitations are shown in Table 13).
    - For kSamplingPeriod values smaller than 10000 ps (i.e. sampling frequencies larger than 100 MHz), you must edit the IP timing constraints for timing closure to be achieved.
    - To do this, please right click on the ZmodScopeController IP in the block diagram and select "Edit in IP Packager". In the Vivado project that gets opened, go to the Constraints section -> constr_1 -> open "ConstrsZmodADC.xdc". Comment lines 22 and 26, and uncomment lines 24 and 28 (i.e. replace tskew_max with 0.750 and tskew_min with -0.900, respectively).
    - Since the AD9648 DCO to Data Skew parameter is specified for the full operating temperature range (−40°C to +85°C ambient) and the Zmod Scope works in a much narrower temperature range, the above changes should work fine.
3. 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.
4. Enabling/disabling dynamic configuration of relays and calibration coefficients through the *kExtRelayConfigEn*, *kExtCalibEn* and *kExtSyncEn* parameters. These parameters enable the user to opt for a more basic design with minimal external ports or a more configurable but also more complex design with several ports that enable dynamic configuration of several hardware features. When using the IP Core with the higher level IPs provided by Digilent, the hardware configuration features are expected to be controlled by the processing system, so all interfaces should be enabled and connected to the upper layer IP Core.

8 References

The following documents provide additional information on the subjects discussed:

1. Xilinx Inc., PG057: FIFO Generator, v13.1, April 5, 2017.
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.
5. Analog Devices, AD9204 Datasheet, Rev A.
6. Analog Devices, AD9231 Datasheet, Rev B.
7. Analog Devices, AD9251 Datasheet, Rev B.

8. Analog Devices, AD9608 Datasheet, Rev C.
9. Analog Devices, AD9628 Datasheet, Rev C.