DE0-Nano
ADC Controller

For Quartus Prime 16.0

1 Core Overview

The DE0-Nano ADC Controller IP Core provides an interface between a processor and the ADC128S022 Analog-to-Digital Converter present on the DE0-Nano board.

2 Functional Description

The DE0-Nano ADC Controller IP Core provides access to all 8 input channels of the ADC128S002 Analog-to-Digital Converter. It is designed for use with the 50 MHz system clock, and controls all required digital signals both to and from the ADC.

3 Instantiating the Core

The DE0-Nano ADC Controller IP core can be instantiated in a system using Qsys or as a standalone component from the IP Catalog within the Quartus II software. Designers use the core's configuration wizard to specify the number of channels used by the ADC Controller. The ADC will only update channels 0 to *n*-1, where *n* is the number of channels in use. By default, all eight channels are used.

4 Software Programming Model

4.1 Register Map

The DE0-Nano ADC IP Core provides eight registers for reading and three for writing, as shown in Table 1. The eight readable registers contain the outputs from the ADC for the eight analog inputs. The three writable registers are used to control the ADC. Writing to the *Update* register triggers an update of the stored conversions, writing to *Auto-Update* enables or disables the automatic update feature, and writing to the *SCLK Counter* sets the frequency of SCLK.

4.1.1 Channel Registers

These eight registers hold the 12-bit output from the ADC for each channel. They are refreshed upon completion of an update cycle and are stored until another update occurs. These registers are only available for reading.

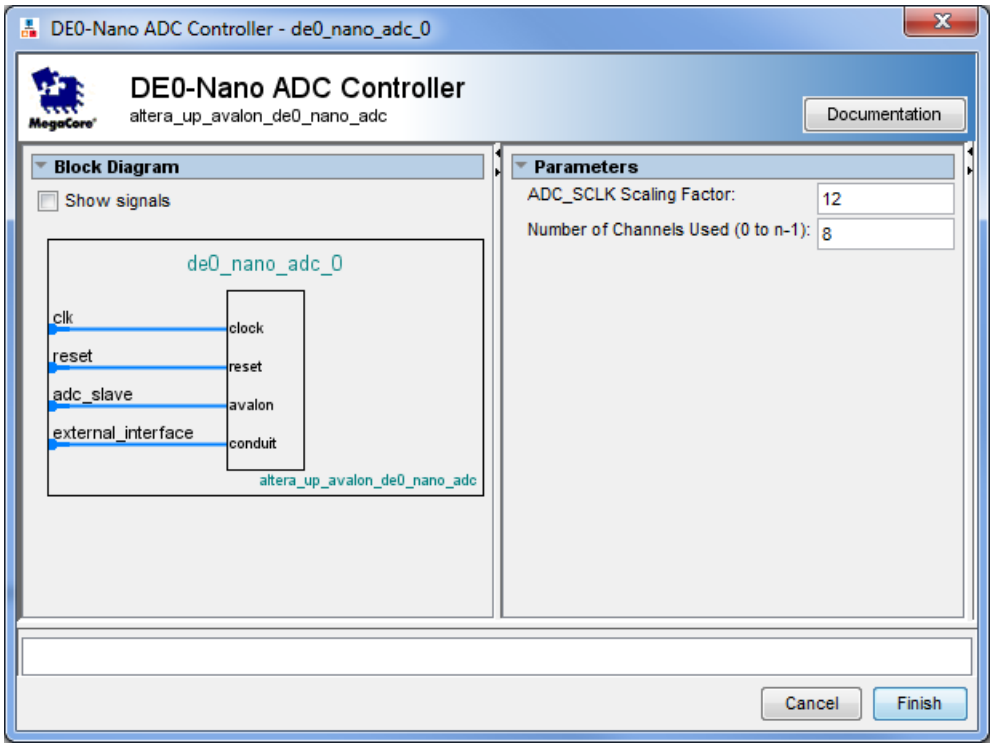

Figure 1. ADC Controller's Configuration Wizard.

Table 1. DE0-Nano ADC Controller register map

Offset in bytes	Register name	Read/Write	Purpose
0	CH_0	R	Converted value of channel 0
	Update	W	Update the converted values
4	CH_1	R	Converted value of channel 1
	Auto-Update	W	Enables or disables auto-updating
8	CH_2	R	Converted value of channel 2
	SCLK Counter	W	Set the SCLK frequency
12	CH_3	R	Converted value of channel 3
16	CH_4	R	Converted value of channel 4
20	CH_5	R	Converted value of channel 5
24	CH_6	R	Converted value of channel 6
28	CH_7	R	Converted value of channel 7

4.1.2 *Update* Register

Performing a write to the *Update* register begins an conversion cycle on the ADC. During this time, all desired channels will be sampled. The new values will become available on the Channel registers once the entire update operation has finished. If reads to the channel registers are attempted while a conversion is taking place, then the *wait_request* signal will be raised, causing the processor to stall until the update has finished.

4.1.3 *Auto-Update* Register

On system startup, this register will be loaded with a zero value. Writing a '1' to this register will enable auto-updating, while writing a '0' will disable it.

When auto-update is enabled, the system will automatically begin another update cycle after the previous one finishes. Additionally, channel values will be available during an update cycle, allowing them to be accessed any time.

4.2 *SCLK Counter* Register

The *SCLK Counter* register is used to control the frequency of the SCLK signal to the ADC. One cycle of SCLK will correspond to the number of system clock cycles equal to the value stored in this register.

The user must ensure that the value of SCLK remains within the ADC's limit of 0.8 MHz to 3.2 MHz. For a standard system clock frequency of 50 MHz, the value in *SCLK Counter* should be between 16 and 62. By default, *SCLK Counter* stores a value of 25, which equates to an SCLK frequency of 2 MHz.

4.3 Device Driver for the Nios II Processor

The DE0-Nano ADC Controller core is packaged with C-language functions accessible through the hardware abstraction layer (HAL). These functions implement basic operations for the ADC Controller.

To use the functions, the C code must include the statement:

```
#include "altera_up_avalon_de0_nano_adc.h"
```

4.3.1 alt_up_de0_nano_adc_open_dev

Prototype: `alt_up_de0_nano_adc_dev* alt_up_de0_nano_adc_open_dev(const char *name)`
Include: `<altera_up_avalon_de0_nano_adc.h>`
Parameters: `name` – the ADC Controller name. For example, if the ADC controller name in Qsys is "ADC", then *name* should be "/dev/ADC"
Returns: The corresponding device structure, or NULL if the device is not found.
Description: Open the ADC controller device specified by *name* .

4.3.2 alt_up_de0_nano_adc_read

Prototype: `unsigned int alt_up_de0_nano_adc_read ( alt_up_de0_nano_adc_dev *adc, unsigned channel)`
Include: `<altera_up_avalon_de0_nano_adc.h>`
Parameters: `adc` – struct for the ADC controller device .
`channel` – the channel to be read, from 0 to 7.
Returns: data – The converted value from the desired channel.
Description: Read from a channel of the ADC.

4.3.3 alt_up_de0_nano_adc_update

Prototype: `void alt_up_de0_nano_adc_update( alt_up_de0_nano_adc_dev *adc)`
Include: `<altera_up_avalon_de0_nano_adc.h>`
Parameters: `adc` – struct for the ADC controller device .
Description: Trigger the controller to convert all channels and store the values.

4.3.4 alt_up_de0_nano_adc_auto_enable

Prototype: `void alt_up_de0_nano_adc_auto_enable( alt_up_de0_nano_adc_dev *adc)`
Include: `<altera_up_avalon_de0_nano_adc.h>`
Parameters: `adc` – struct for the ADC controller device .
Description: Enable automatic converting of channels.

4.3.5 alt_up_de0_nano_adc_auto_disable

Prototype: `void alt_up_de0_nano_adc_auto_disable( alt_up_de0_nano_adc_dev *adc)`
Include: `<altera_up_avalon_de0_nano_adc.h>`
Parameters: `adc` – struct for the ADC controller device .
Description: Disable automatic converting of channels.

4.3.6 alt_up_de0_nano_adc_set_sclk_counter

Prototype: `void alt_up_de0_nano_adc_set_sclk_counter( alt_up_de0_nano_adc_dev *adc, unsigned sclk_count)`
Include: `<altera_up_avalon_de0_nano_adc.h>`
Parameters: `adc` – struct for the ADC controller device .
`sclk_count` – the new value for the SCLK counter.
Description: Set the frequency of SCLK by changing the ratio of system clock cycles per SCLK cycle.

4.3.7 alt_up_de0_nano_adc_set_sclk_freq

Prototype: `void alt_up_de0_nano_adc_set_sclk_freq( alt_up_de0_nano_adc_dev *adc, float sclk_freq, float sysclk_freq)`

Include: `<altera_up_avalon_de0_nano_adc.h>`

Parameters: `adc` – struct for the ADC controller device .
`sclk_freq` – the desired value for SCLK, in MHz.
`sysclk_freq` – the system clock frequency in MHz. Typically 50MHz on DE0-Nano boards.

Description: Set the frequency of SCLK. This function provids an alternate interface to the `alt_up_de0_nano_adc_set_sclk_freq` function.