ADQ214/ADQ114 Development Kit

User's Guide

Development kit for data acquisition boards from SP Devices

Document Number 11-0724

Rev D, October 2017

Important Information

Signal Processing Devices Sweden AB (SP Devices) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to SP Devices' general terms and conditions supplied at the time of order acknowledgment.

SP Devices warrants that each product will be free of defects in materials and workmanship, and conform to specifications set forth in published data sheets, for a period of one (1) year. The warranty commences on the date the product is shipped by SP Devices. SP Devices' sole liability and responsibility under this warranty is to repair or replace any product which is returned to it by Buyer and which SP Devices determines does not conform to the warranty. Product returned to SP Devices for warranty service will be shipped to SP Devices at Buyer's expense and will be returned to Buyer at SP Devices' expense. SP Devices will have no obligation under this warranty for any products which (i) has been improperly installed; (ii) has been used other than as recommended in SP Devices' installation or operation instructions or specifications; or (iii) has been repaired, altered or modified by entities other than SP Devices. The warranty of replacement products shall terminate with the warranty of the product. Buyer shall not return any products for any reason without the prior written authorization of SP Devices.

In no event shall SP Devices be liable for any damages arising out of or related to this document or the information contained in it.

SP DEVICES' EXPRESS WARRANTY TO BUYER CONSTITUTES SP DEVICES' SOLE LIABILITY AND THE BUYER'S SOLE REMEDY WITH RESPECT TO THE PRODUCTS AND IS IN LIEU OF ALL OTHER WARRANTIES, LIABILITIES AND REMEDIES. EXCEPT AS THUS PROVIDED, SP DEVICES DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING ANY WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT.

SP DEVICES DOES NOT INDEMNIFY, NOR HOLD THE BUYER HARMLESS, AGAINST ANY LIABILITIES, LOSSES, DAMAGES AND EXPENSES (INCLUDING ATTORNEY'S FEES) RELATING TO ANY CLAIMS WHATSOEVER. IN NO EVENT SHALL SP DEVICES BE LIABLE FOR SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFIT, LOST DATA AND THE LIKE, DUE TO ANY CAUSE WHATSOEVER. NO SUIT OR ACTION SHALL BE BROUGHT AGAINST SP DEVICES MORE THAN ONE YEAR AFTER THE RELATED CAUSE OF ACTION HAS ACCRUED. IN NO EVENT SHALL THE ACCRUED TOTAL LIABILITY OF SP DEVICES FROM ANY LAWSUIT, CLAIM, WARRANTY OR INDEMNITY EXCEED THE AGGREGATE SUM PAID TO SP BY BUYER UNDER THE ORDER THAT GIVES RISE TO SUCH LAWSUIT, CLAIM, WARRANTY OR INDEMNITY.

Worldwide Sales and Technical Support

www.spdevices.com

Teledyne SP Devices Corporate Headquarters

Teknikringen 6
SE-583 30 Linköping
Sweden

Phone: +46 (0)13 465 0600
Fax: +46 (0)13 991 3044
Email: info@spdevices.com

Contents

1 INTRODUCTION

This user's guide documents the features of the ADQ Development Kit. There are two sections in this document, the first section describing deployment of features for FPGA #1, also called algorithm FPGA or alg_FPGA in some contexts, and the second section describes the use of software.

FPGA #1 is directly connected to the ADC output.

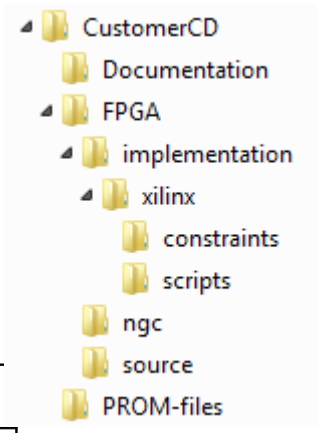
2 REQUIREMENTS

To fully utilize all examples in this guide, the following software and hardware tools are needed

- Xilinx ISE 12.4
- SP Devices ADQ Development Kit CD
- Xilinx Platform Cable II – optional for use with Xilinx debugging tool ChipScope

2.1 File structure

The SP Devices ADQ Development Kit comes with a CD with the file structure displayed to the right. At the top level the structure is divided into FPGA, Documentation and Software. FPGA contains all FPGA code and examples. Documentation includes papers describing how to use the Development Kit, how to program the FPGA. The PROMFiles folder contains the generated PROM files for the DevKit. In the Software folder one can find Matlab example scripts to use with the DevKit examples.



2.1.1 Algorithm FPGA

The FPGA folder contains the code for the algorithm FPGA (alg_**fpga**). Table 1 describes the folder contents of the directory.

Name	**Content**
implementation	Xilinx ISE project folder with script files
ngc	Precompiled netlists
source	Verilog source code

Table 1 Algorithm FPGA's folders

3 SYSTEM OVERVIEW

Figure 1 & Figure 2 gives a schematic overview of how the signal processing FPGA is connected to the ADCs and the communication FPGA. The ADQ series of boards features a set of ADCs and the number of channels depends on the specific board.

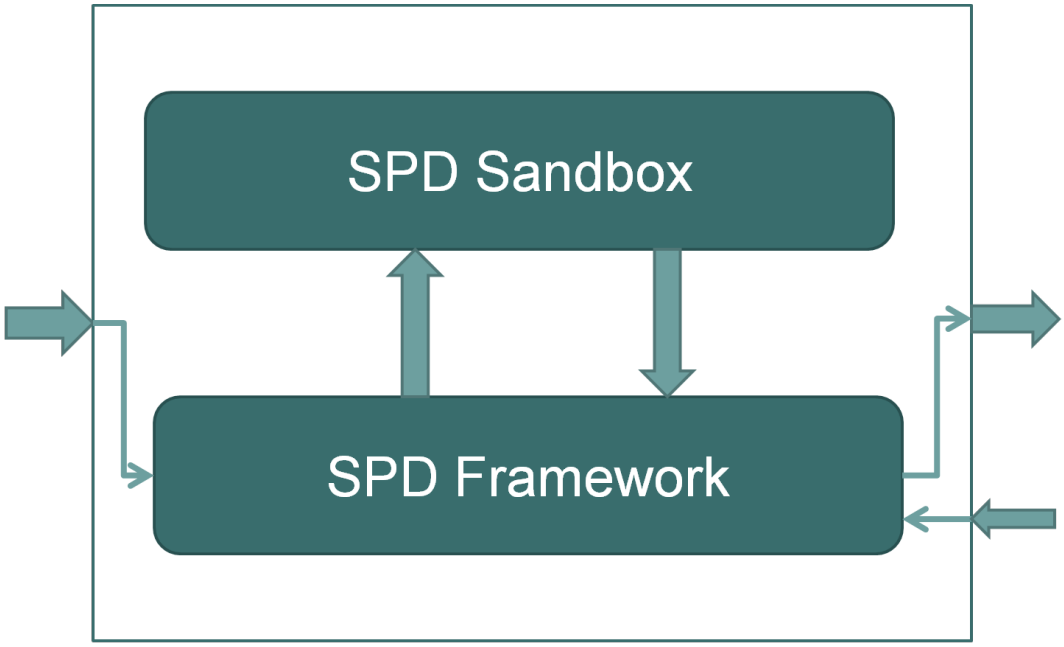


Figure 1. System level overview of an ADQ V5

SHAPE * MERGEFORMAT

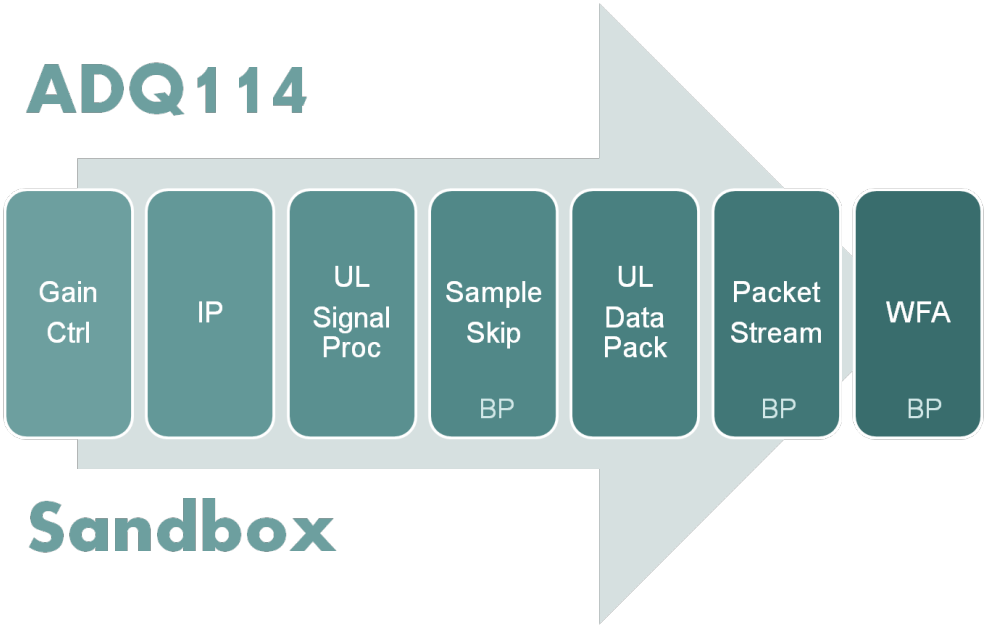


Figure 2. System level overview of an ADQ114. Other V5 products may differentiate slightly.

The ADQ Development Kit gives the user direct access to data from the ADC which can be used for various signal processing, such as decimation, FFT, filtering, or any custom algorithm. There are two user logic blocks where the user can insert their code. The following sections will cover specifications of the interfaces and give an example of an implementation. The modules marked with BP are supplied in two versions, one with full functionality and one with bypass functionality. This gives the user a possibility to save some resources by bypassing a module which function is not needed.

4 FPGA #1: ALGORITHM FPGA

4.1 Interface Specifications

The following specification is valid for ADQ114 and differences may occur between different ADQ products. Currently only customization for FPGA #1 is possible.

4.1.1 Signal Assignments

The data format of the incoming and outgoing data is signed two's complement.

Input	
Name	**Description**
x0	Data from ADC channel A
x0z	Data from ADC channel A one cycle delayed
x1	Data from ADC channel B
x1z	Data from ADC channel B one cycle delayed
clk	Data rate clock
user_register_i	User defined registers to be set by the API
trigger_vector_i	Information about the trigger from FPGA #2

Table 2 Input signals

Output	
Name	**Description**
y0	Data output channel A
y0z	Data output channel A one cycle delayed
y1	Data output channel B
y1z	Data output channel B one cycle delayed
user_register_o	User defined registers to be read by the API

Table 3 Output signals

The developer has access to a 8 user registers to set parameters in the FPGA and 8 registers to output status from the custom design. The default operation of the ADQ2147 ADQ212 is to forward the ADC data to the communication FPGA together with a data valid signal.

4.1.2 Data rates

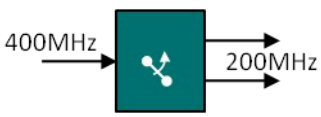


Figure 3. Data rate conversion on ADQ214.

Each channel has an effective data rate of 400MSPS on ADQ214 and 550 MSPS on ADQ212. It is divided into two data streams operating at 200MHz for ADQ214 and 275 MHz for ADQ212. That is, the ADC data arrives with two samples per channel into the development environment at half rate.

4.2 Structure

The following table describes the structure of the project and points out interesting areas for modification of the code.

Name	Description
adq_alg_fpga	Top module of the algorithm FPGA
spd_framework	Precompiled Netlist for external comunication
spd_sandbox	Configurable Area
Gain Control	LVDS outputs to communication FPGA
Counter Test Pattern	SPD's internal debugging module (optional)
ADX-2 IP	SPD's interleaving algorithm
UL Signal Processing	User code area
Sample Skipper	Skips a configurable number of samples between each sample to keep
UL Data Packaging	User code area
Packet Streaming	Compress data for more efficient data streaming
Waveform Averaging	Calculates the average during a configurable data length and number of repetitions
SPI_registers	SPI communication with FPGA #2

Table 4 FPGA #1 block hierarchy. Note that some block might not be present for some V5 product.

The two user logic (UL) blocks are the instances where the customer can access ADC data and perform custom signal processing. Data is formatted into 2's complement and ready for signal processing using for example custom algorithms or functionalities from Xilinx CORE Generator.

4.2.1 SPI registers

The SPI slave communicates with the communication FPGA, which makes it possible to communicate between the PC and the algorithm FPGA. The SPI interface block has eight 16 bits input registers and eight 16 bits output registers that are free to use in the design. The registers can be controlled from the ADQAPI, using either C++/C or Matlab.

As a default four registers are routed from signal processing UL and four from data packaging UL to the SPI registers. But all eight registers from SPI registers are routed into both UL.

In C use the ReadAlgoRegister and the WriteAlgoRegister functions to communicate with the user registers. The addresses are stated in Table 5 and Table 6. The functions are further explained in the ADQAPI documentation.

It is important to note that the ADC clock and the SPI register clock are uncorrelated. The signals passing in between must have a correct design of the clock domain crossing, for the functionality to be stable and reliable.

Name	Address	Routed from block
ur_0_i	0x20	UL-SP
ur_1_i	0x21	UL-SP
ur_2_i	0x22	UL-SP
ur_3_i	0x23	UL-SP
ur_4_i	0x24	UL-DP
ur_5_i	0x25	UL-DP
ur_6_i	0x26	UL-DP
ur_7_i	0x27	UL-DP

Table 5 Addresses for input user registers

Name	Address	Routed to block
ur_0_o	0x30	UL-SP, UL-DP
ur_1_o	0x31	UL-SP, UL-DP
ur_2_o	0x32	UL-SP, UL-DP
ur_3_o	0x33	UL-SP, UL-DP
ur_4_o	0x34	UL-SP, UL-DP
ur_5_o	0x35	UL-SP, UL-DP
ur_6_o	0x36	UL-SP, UL-DP
ur_7_o	0x37	UL-SP, UL-DP

Table 6 Addresses for output user registers

4.2.2 Setup

To setup a pass through example, please follow these steps.

- Copy the file structure from the Development Kit CD to the local hard drive
- Launch **Xilinx ISE Project Navigator**
- If an existing project is loaded, click **File – Close Project**
- Go to the **TCL Shell** tab
- Navigate to the **FPGA/implementation/xilinx/** folder with the help of the **cd** commands. For example **cd c:/FPGA/alg_fpga/implementation/xilinx/**
- For ADQ212 type **source scripts/ADQ214_devkit.tcl** or for ADQ212 type **source scripts/ADQ212_devkit.tcl** to load the different build options
- Type **rebuild_project** to build the default project. This will take a couple of minutes.
- Type **configure_devkit arg1 arg2 arg3** to configure the project to bypass a module Default setting is **configure_devkit 1 1 1** , which will disable all blocks.
    - Arg1 – bypass packet streamer

- arg2 – bypass waveform averaging
- arg3 – bypass decimation extended (only in ADQ2xx)

Make sure no errors were reported from the build script. The project structure should now look according to Figure 4.

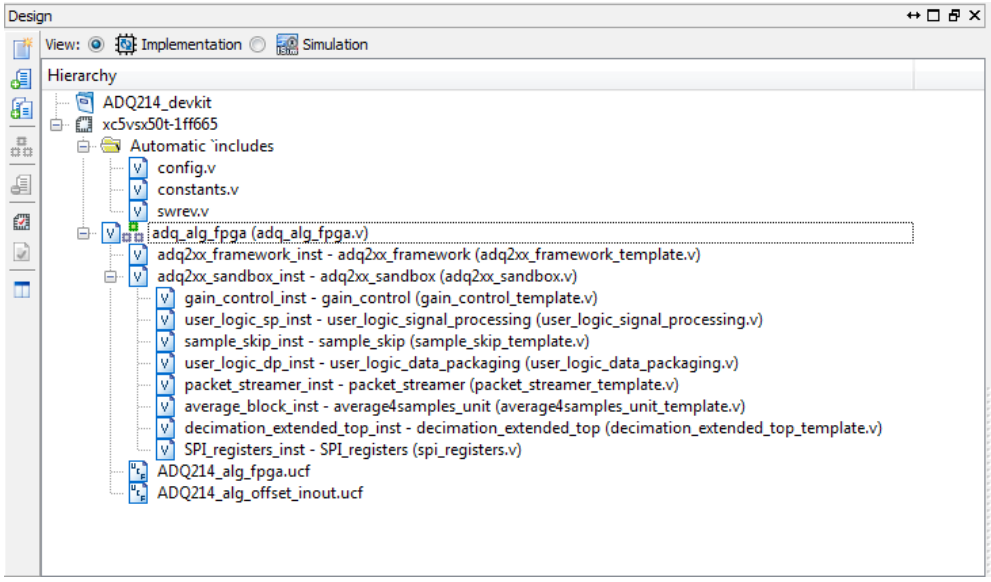


Figure 4. Project structure. Note that this might look slighly different depending on the product.

To verify that the project was built successfully and that there are no errors, please use the **Check Syntax** command in the following way.

- Select the top module **adq_alg_fpga** in the **Sources** tab
- In the **processes** tab, expand **Synthesize – XST** and double click **Check Syntax**

Verify that the check syntax process is successfully completed in the console tab and that the **Check Syntax** icon is green.

4.2.3 Generate Programming files

- One will have to do the following steps from the previous section every time one has restarted Xilinx ISE
  - Go to the **TCL Shell** tab
  - Navigate to the **FPGA/implementation/xilinx/** folder with the help of the **cd** commands. For example **cd c:/FPGA/alg_fpga/implementation/xilinx/**
  - Type **source scripts/ADQ214_devkit.tcl** ot **source scripts/ADQ212_devkit.tcl** to load the different build options
- Type **run_process** to generate the programming files of the chosen example. The running time of this process may vary depending on the size of the project.
- When this step is done there will be a new folder under **DevKit\alg_fpga\implementation\xilinx\Logfiles** with the time and date of the generation. This folder contains the generated programming files and reports.

The Dev Kit uses netlists (*.ngc) for the included modules which the user cannot modify. These modules are stored in the folder ngc, and are copied to the project directory using **configure_devkit**. If the copy fails, then the user will get a lot of errors in the Translate step, which complains that, there are constraints that cannot be mapped.

```
ERROR:NgdBuild:981 - Could not find any associations for the following constraint:
 '<INST "adq2xx_framework_inst/lvds_top_inst/pll_ad_inst_a/PLL_ADV_INST"  CLKOUT0_PHASE = 56.25;>
```

If this happens, rerun the **configure_devkit** and make sure that there are no errors and all ngcs are present in the project folder.

If a user decides to e.g. switch to a bypass module, it is not always that Xilinx ISE will notice the change and the user has to force a rerun of the Translate step before using the **process_run** command.

5 FPGA PROGRAMMING

To try the DevKit firmware out in hardware, the ADQUpdater software which is included in the ADQ SDK, may be used to reprogram the onboard firmware FLASH memory. Please refer to the document **ADQ Firmware Updater Guide,** which is included in the Documentation folder of the ADQ SDK installation directory.

Example firmware which has been built from the DevKit is included in the PROMFiles directory of the DevKit folder or CD and may be used as desired.

6 FUNCTIONAL VERIFICATION

Depending on the extent of the customization, the included SP Devices software ADCaptureLab might not be able to collect relevant data from the card. However, a simple way of testing the communication with the device is to do the following steps.

- Install the ADQ software including ADCaptureLab
- Start ADCaptureLab and verify that the board is found when pressing "**Find USB Devices**"
- Press **F2** to view the revision of the board

If the above tests are successful, basic communication with the board is working. To collect raw data from the board, MATLAB is a suitable tool. To test the MATLAB interface, please refer to the **ADQAPI User Guide**, in the Documentation folder, and the example scripts located in the Matlab_examples folder of the SDK installation directory.

Worldwide Sales and Technical Support

www.spdevices.com

Teledyne Signal Processing Devices Corporate Headquarters

Teknikringen 6
SE-583 30 Linköping
Sweden

Phone:	+46	(0)13	465	0600
Fax:	+46	(0)13	991	3044
Email:	info@spdevices.com			