One Technology Way · P.O. Box 9106 · Norwood, MA 02062-9106 · Tel: 781.329.4700 · Fax: 781.461.3113 · www.analog.com

AD5252 FMC-SDP Interposer & Evaluation Board / Xilinx KC705 Reference Design

Supported Devices

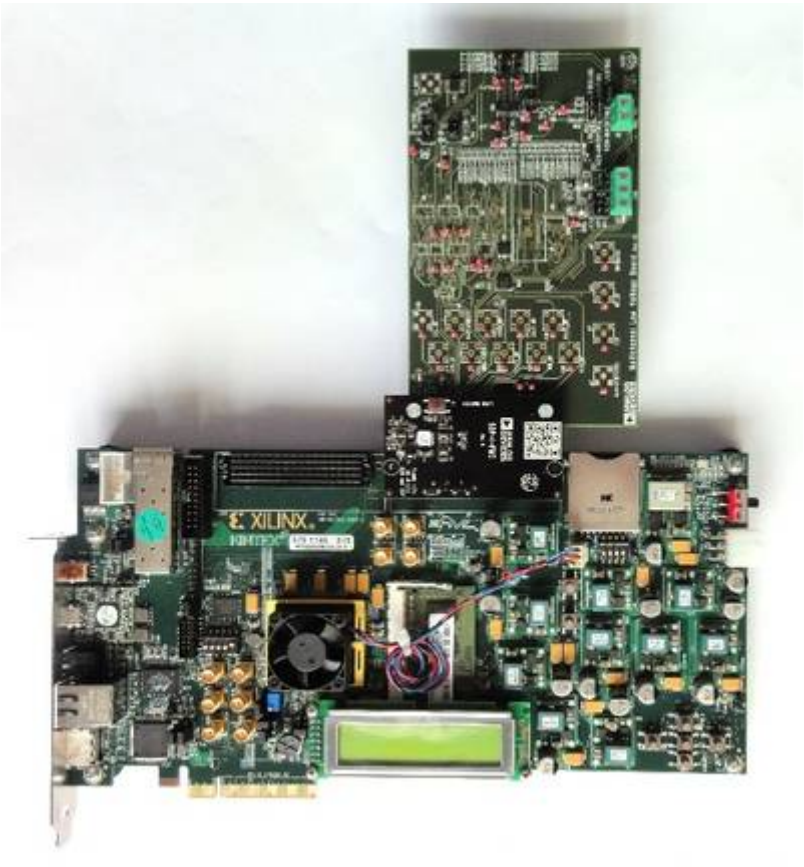
- AD5252

Evaluation Boards

- EVAL-AD5252SDZ

Overview

This document presents the steps to setup an environment for using the **EVAL-AD5252SDZ** evaluation board together with the Xilinx KC705 FPGA board and the Xilinx Embedded Development Kit (EDK). Below is presented a picture of the EVAL-AD5252SDZ Evaluation Board with the Xilinx KC705 board.



For component evaluation and performance purposes, as opposed to quick prototyping, the user is directed to use the part evaluation setup. This consists of:

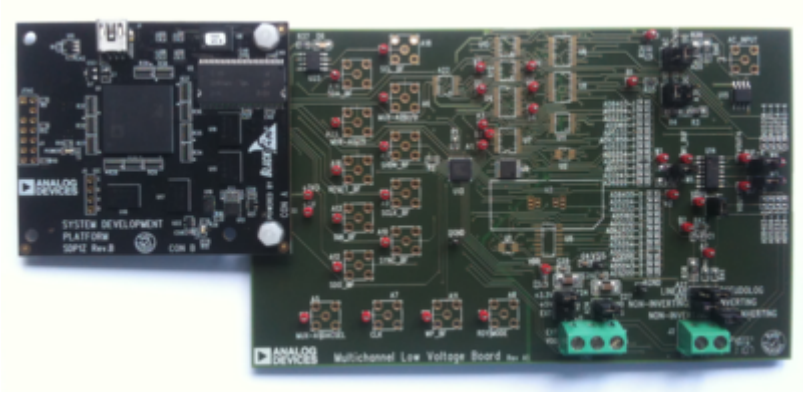
- 1. A controller board like the SDP-B (EVAL-SDP-CS1Z)
- 2. The component SDP compatible product evaluation board
- 3. Corresponding PC software (shipped with the product evaluation board)

The SDP-B controller board is part of Analog Devices System Demonstration Platform (SDP). It provides a high speed USB 2.0 connection from the PC to the component evaluation board. The PC runs the evaluation software. Each evaluation board, which is an SDP compatible daughter board, includes the necessary installation file required for performance testing.

Note: it is expected that the analog performance on the two platforms may differ.

28 Sep 2012 09:32 · Adrian Costina

Below is presented a picture of **SDP-B** Controller Board with the **EVAL-AD5252SDZ** Evaluation Board.



The AD5252 is dual-channel, I2C, nonvolatile memory, digitally controlled potentiometer with 256 positions, respectively. This device performs the same electronic adjustment functions as mechanical potentiometers, trimmers, and variable resistors. The part's versatile programmability allows multiple modes of operation, including read/write access in the RDAC and EEMEM registers, increment/decrement of resistance, resistance changes in ±6 dB scales, wiper setting readback, and extra EEMEM for storing user-defined information, such as memory data for other components, look-up table, or system identification information.

The **EVAL-AD5252SDZ** evaluation board is designed to help customers quickly prototype new AD5252 circuits and reduce design time. The EVAL-AD5252SDZ incorporates several test circuits to evaluate the AD5252 performance.

More information

- AD5252 Product Info - pricing, samples, datasheet
- EVAL-AD5252SDZ evaluation board user guide
- Xilinx KC705 FPGA board

Getting Started

The first objective is to ensure that you have all of the items needed and to install the software tools so that you are ready to create and run the evaluation project.

Required Hardware

- Xilinx KC705 FPGA board
- FMC-SDP adapter board
- **EVAL-AD5252** evaluation board

Required Software

- Xilinx ISE 14.6.
- UART Terminal (Termite/Tera Term/Hyperterminal), baud rate 115200.

Downloads

- **AD5252 Driver:** https://github.com/analogdevicesinc/no-OS/tree/master/device_drivers/AD525x
- **AD5252 Commands:** https://github.com/analogdevicesinc/no-OS/tree/master/device_commands/AD525x
- **Xilinx Boards Common Drivers:** https://github.com/analogdevicesinc/no-OS/tree/master/platform_drivers/Xilinx/SDP_Common
- **EDK KC705 Reference project:** https://github.com/analogdevicesinc/fpgahdl_xilinx/tree/master/cf_sdp_kc705

Hardware setup

Before connecting the ADI evaluation board to the Xilinx KC705 make sure that the VADJ_FPGA voltage of the KC705 is set to 3.3V. For more details on how to change the setting for VADJ_FPGA visit the Xilinx KC705 product page.

- Use the FMC-SDP interposer to connect the ADI evaluation board to the Xilinx KC705 board on the FMC LPC connector.
- Connect the JTAG and UART cables to the KC705 and power up the FPGA board.

Reference Project Overview

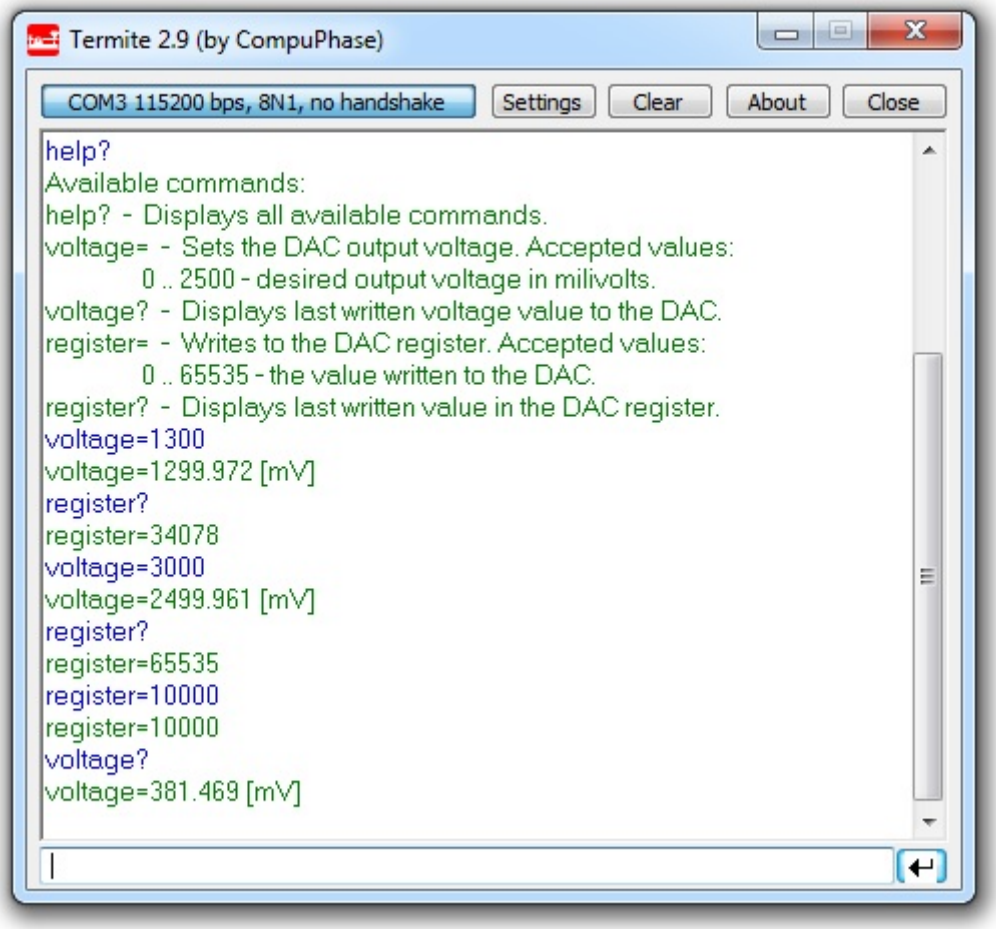
The following commands were implemented in this version of EVAL-AD5252 reference project for Xilinx KC705 FPGA board.

Command	Description
help?	Displays all available commands.
rdac=	Load the wiper register with a give value. Accepted values: channel: 0 - select RDAC 1 wiper register. 1 - select RDAC 2 wiper register. value: 0 .. 255 - value to be written in register.
rdac?	Read back the value of the wiper register. Accepted values: channel: 0 - select RDAC 1 wiper register. 1 - select RDAC 2 wiper register.
reset!	Reset all wiper register to its stored values
restore=	Restore the specified wiper register setting form the memory. Accepted value: channel: 0 - select RDAC 1 wiper register. 1 - select RDAC 2 wiper register.
save=	Save the given wiper register settings to the memory. Accepted value: channel: 0 - select RDAC 1 wiper register. 1 - select RDAC 2 wiper register.
writemem=	Write to one of the user memory address. Accepted value: address: a value between 0x2 and 0xE. data: a value between 0 and 255.
readmem=	Read data from the EEMEM memory. Accepted value: address: a value between 0x2 and 0xE.
decrdacdb=	Decrement a given wiper register by 6dB. Accepted value: channel: 0 - select RDAC 1 wiper register. 1 - select RDAC 2 wiper register.
decrdacdball!	Decrement all wiper register by 6dB.
decrdac=	Decrement a given wiper register by one. Accepted value: channel: 0 - select RDAC 1 wiper register. 1 - select RDAC 2 wiper register.
decrdacall!	Decrement all wiper register by one.
incrdacdb=	Increment a given wiper register by 6dB. Accepted value: channel: 0 - select RDAC 1 wiper register. 1 - select RDAC 2 wiper register.
incrdacdball!	Increment all wiper register by 6dB.
incrdac=	Increment a given wiper register by one. Accepted value: channel: 0 - select RDAC 1 wiper register. 1 - select RDAC 2 wiper register.
incrdacall!	Increment all wiper register by one.
setwp=	Set the state of the Write Protect (WP) pin. Accepted value: desired state: 0 - inactive 1 - active

Command	Description
getwp?	Return the current value of the Write Protect (WP) pin
sethwreset=	Set the state of the Hardware Override Preset (PR) pin. Accepted value: 0 - inactive 1 - active
gethwreset?	Return the current value of the Hardware Override Preset (PR) pin
tolerance=	Read one of the Tolerance register. Accepted value: 0x0 .. 0x3 - virtual address of the tolerance register

Commands can be executed using a serial terminal connected to the UART peripheral of Xilinx KC705 FPGA.

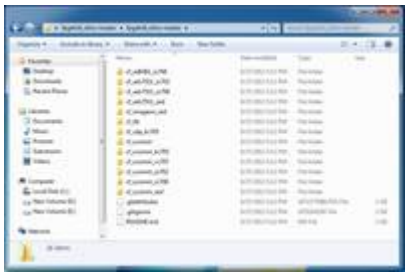
The following image shows a generic list of commands in a serial terminal connected to Xilinx KC705 FPGA's UART peripheral.



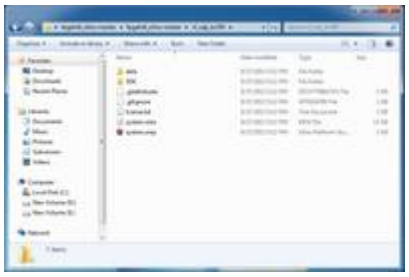
Software Project Setup

The hardware platform for each reference projects with FMC-SDP interposer and KC705 evaluation board is common. The next steps should be followed to recreate the software project of the reference design:

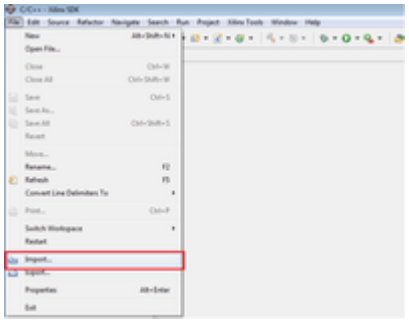
- First download the **KC705 Reference project** from Github on your computer. You can do this by clonning this repository: https://github.com/analogdevicesinc/fpgahdl_xilinx.



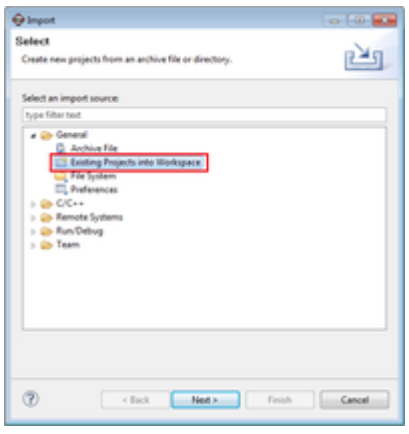
- From this entire repository you will use **cf_sdp_kc705** folder. This is common for all KC705 projects.



- Open the Xilinx SDK. When the SDK starts, it asks you to provide a folder where to store the workspace. Any folder can be provided. Make sure that the path where it is located does not contain any spaces.
- In the SDK select the ***File→Import*** menu option to import the software projects into the workspace.

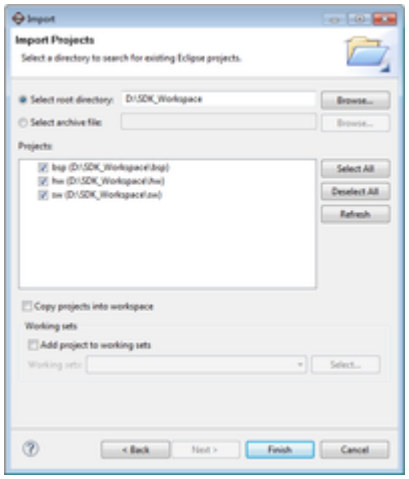


- In the *Import* window select the ***General→Existing Projects into Workspace*** option.



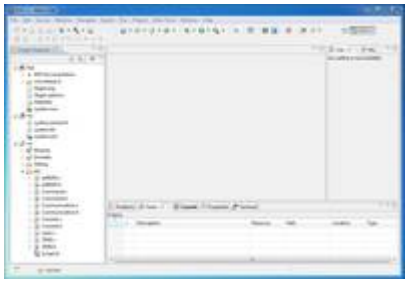
- In the *Import Projects* window select the **cf_sdp_kc705** folder as root directory and check the **Copy projects into workspace** option. After the root directory is chosen the projects that reside in that

directory will appear in the *Projects* list. Press *Finish* to finalize the import process.

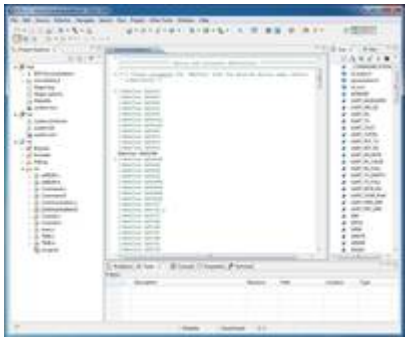


- The *Project Explorer* window now shows the projects that exist in the workspace without software files.

- Now the software must be added in your project. For downloading the software, you must use 3 links from Github given in **Downloads** section. From there you'll download the specific driver, the specific commands and the Xilinx Boards Common Drivers(which are commons for all Xilinx boards). All the software files downloaded must be copied in **src** folder from sw folder.



- Before compilation in the file called **Communication.h** you have to uncomment the name of the device that you currently use. In the picture below there is an example of this, which works only with AD5629R project. For another device, uncomment only the respective name. You can have one driver working on multiple devices, so the drivers's name and the uncommented name may not be the same for every project.



- The SDK should automatically build the project and the *Console* window will display the result of the build. If the build is not done automatically, select the ***Project→Build Automatically*** menu option.
- If the project was built without any errors, you can program the FPGA and run the software application.

13 Aug 2013 08:22 · Lucian Sin

More information

- AD5252 Digital Potentiometer Linux Driver

- ask questions about the FPGA reference design
- Example questions:
  - FMCOMMS1 AD9548 clock derivation explanation by cherif.chibane@ll.mit.edu
  - FMCDAQ2 Arria10GX Nios Sourcecode by kairue
  - Using ZC706 and AD-fmcomms3 by 85083074@qq.com
  - FM-COMMS3 and FM-COMMS5 with VC707 vs Zync ZC706 by dr8
  - How Xps ip update to vivado ? by huanmolb@163.com

28 May 2012 14:18