Important information, pls. read carefully before taking the IE4-DS lab!

General Introduction to IE4-DS Digital Circuits Lab

Prof. Dr.-Ing. Lutz Leutelt

The following document give hints how to make most out of the lab. The lab and its preparation and documentation will give you a lot of practical experience that supports the theory in the lecture, hence, it will consume a significant amount of workload hours (approx. 45…50 hours per semester).

There are four lab sessions this semester which have three phases each: preparation, lab exercises and report.

This semester, you will be given **3 lab tasks** which you have to **complete in 4 lab sessions**.

Preparation

Since only 4 teaching hours (@ 45 min) are available to do the lab exercises in the lab room, it is essential to prepare them very well at home. In the lab you have the assistance from a lab assistant and a prof – when you do not finish on time due to a lack of preparation you have no such assistance afterwards and work in lab will be less effective and progress slower.
In order to help you prepare, prep sheets are available for most of the lab session which raise the most important questions to be solved beforehand. It is mandatory for everyone to upload a completed copy to EMIL **in PDF format** per session. In addition, one hardcopy per team has to be handed to the lab assistant/prof at the beginning of the lab session **(copies must be ready to be collected at the beginning of the session)**. Pls. expect the preparation to take half a day, depending on your pre-knowledge maybe even more. So, starting the night before the lab session is not a good idea, hoping that the team mates will have properly prepared the tasks is not a good idea either.

Lab exercises

It is essential that you studied at home already what has to be done in the lab exercises. Is something unclear? Talk to lab assistant or prof as soon as possible in order not to lose time. Make sure that you mind the safety regulation and other warnings that you do not bring yourself or hardware devices in danger. It is easy to destroy a lab setup by

- **applying wrong voltage levels to a device or**
- **touching pins of integrated circuits with fingers or**
- **connecting outputs of CMOS circuits (e.g. by a wrong pin assignment in a UCF file)**.

During the lab session, take down any information that might be later relevant for the lab report. Save screenshots of the oscilloscope, Xilinx projects, VHDL code changes (version management!), wiring diagrams, synthesis reports, measured values etc. Better log more than necessary – getting information that was forgotten in a lab session is time-consuming.
BTW: Be on time, because it is unfair towards your team mates when they cannot start properly because of your absence. At the beginning of a lab session, important announcements are made. The more you miss of them, the longer you sit afterwards on the lab chair.

Report

We distinguish between a full lab report and a compact lab protocol. One full lab report is required from every student per semester. If you are a team of two/three students you have to write two/three full reports, each with a different corresponding author who will receive the grade. For the other one or two sessions, a lab protocol is to be submitted preferably at the end of the session. Reports and protocols are to be submitted as hardcopies to the prof's yellow mailbox 2 weeks after the lab session.

A lab report shall contain the following information:

- **introduction to the objective of the lab session and the lab setup**
- **description of the lab tasks**
- **results/outcome of the lab task including a discussion**
- **answer to all questions of the lab task**
- **summary of the main results at the end of the report**
- **references to foreign sources (pictures, figures, calculations, values, source code, even when public domain or from internet) have to be given (reference in text and list of references at the end)**

- **figures and tables have number and a caption. The caption contains what the reader is supposed to see in the figure.**

What background can you expect from a reader of your report (who are you writing for)? Just imagine, you were the reader of the report before you have read the lab instructions and prepared the lab. What would have been useful information for you to understand purpose and results of the lab session?

Digital design by VHDL and verification

In all lab sessions, digital hardware is implemented that has been defined by a VHDL model before. The modelling in VHDL and functional verification is major part of the work and has to be done during preparation **before the lab session.** Implementation on a Programmable Logic Device, the test of the real system and corresponding measurements are the minor part if the previous step was done thoroughly. The lab time is primarily reserved for the second part (implementation and test on the real system). During lab preparation you have to create the VHDL model of the digital system and you have to make sure by functional verification that the desired function is fulfilled 100%.
"Really 100%?"
"Yes, 100%. A 'I tried a bit, look here I have some line of code, but somehow it does not work' attitude will result in a highly inefficient lab session and will not be accepted."
You won't be able to finish within lab time. You can ask your class mates, the lab assistant and the prof beforehand to make sure that your VHDL model meets 100% of the functional requirements.

"How can I check 100% correct function at home?"

"By simulations of your model with Modelsim or VHDL test benches and appropriate test cases."

"What means 'appropriate'?"

To give an example: you are developing a synchronous digital counter which shall count from 0 to 15, upwards and downwards. You have to test with your test bench that

- the counter counts with every rising edge of the clock signal
- the count direction matches the input signal which controls the direction of counting
- the counter wraps around correctly at 15 (when counting up) and at 0 (when counting down).

OK?
"What is Modelsim?"

It is a simulation tool which is available in lab room 8.01, PC pool (13th floor) and can be downloaded as a student edition:
http://www.mentor.com/company/higher_ed/modelsim-student-edition

Grading

The lab is passed if you receive **60 out of 100 points** which are awarded as follows:

task	*points*
correct lab preparation (to be uploaded before lab task by every participant plus 1 hard copy per team that will be collected during lab)	**3 x 20 pts**
lab tasks successfully completed (proven by protocol)	**3 x 5 pts.**
1 detailed lab report per participant; (if team of 2, only a brief but complete results summary "protocol" required)	**25 pts.**
punctual arrival for lab session	**- 5 pts. per late arrival**

The criterions for assessment of the lab reports are as follows:

criterion	*points*
purpose and set-up of the lab tasks are properly introduced and supported by sketches, figures, flowcharts, etc. results are summarized at the end	**8**
all tasks have been correctly completed	**4**
results are proven by e.g. measurements, simulations, calculations, source code etc. and have been discussed	**8**
formal aspects are correct, e.g. figures with numbers and subtitles, citation of foreign sources, structure, spelling and style	**5**

Lab Task 1

Serial Interface Module to an ADC and DAC

Estimated Preparation Time: 6-8 hrs

Objective of this Lab Session

A common task in development of digital system is the design of input-output modules (IO modules) that allow interaction between digital devices. Whereas the signal communication inside an integrated chip is mostly parallel because it is faster and the necessary resources for parallel communication is available and comparably cheap. The communication between digital devices is often serial, i.e. the bits of a data word are sent one after another. This saves wires and is for many applications fast enough. To give an example how powerful serial communication can be, the S-ATA (Serial Advanced Technology Attachment) standard revision 3.2 communicates with up to 16 Gbit/s net raw transmission rate between processor system and (solid state) hard drives.
In this lab task, you will develop a module for communication with a digital-to-analog converter (Burr Brown DAC8801) that shall be connected via a 3-wire serial data interface to the FPGA. Before the data can be sent to the DAC, the data has to be serialized. In order to test the design, 12bit data words are read from switches connected to the FPGA and sent from there to the DAC via the serial bus. In a second step, the DAC module is connected to an ADC module (ADS7947) and the entire signal processing chain ADC-DAC can be tested.

Function of the DAC8801

The DAC8801 is a serial 3-wire-input 14-bit DAC with current output. The explanation of the internal operation (R-2R ladder DAC) can be taken from the data sheet [1.1]. In the following, the features are summarized that are relevant for designing serial communication module on the FPGA (see Figure 1):

- input SDI is the serial data stream from FPGA with digital values to be converted
- the CLK signal from FPGA gives the timing of bits
- the falling edge of the low active $\overline{CS}$ signal determines the start of the serial 14-bit data word (MSB first)
- with a rising edge of the $\overline{CS}$ signal the value is copied from shift register to DAC register and conversion is conducted
- the current output I_{OUT} is connected to an I/V converting OpAmp with voltage output range -1.024V (0x0000) to +1.024V (0x3FFF)
- the shift register can latch (store) up to 16 bits, however only the 14 bits sent last are used for conversion

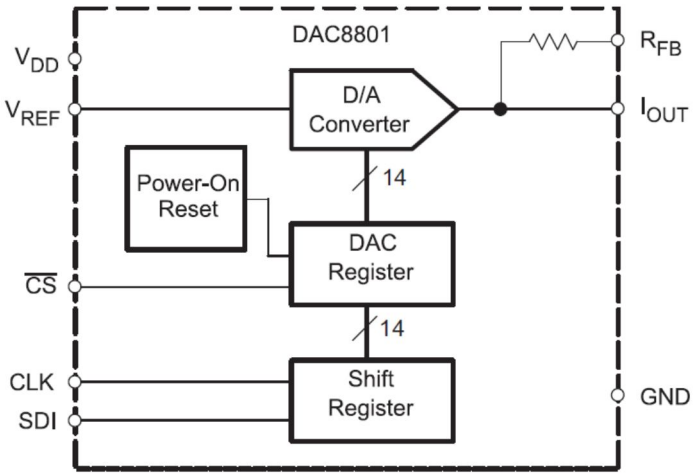
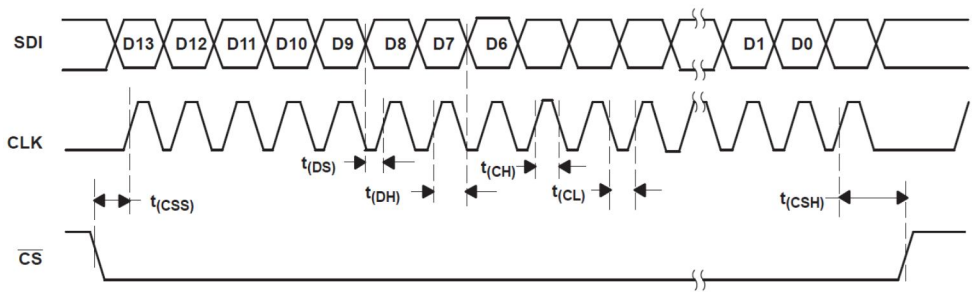


Figure 1: Terminals and main function blocks of DAC8801 (from [1.1]).

The timing requirements can be derived from Figure 2. The serial data can be provided with a maximum clock of 50 MHz. The data have to be valid 5 ns before the rising clock edge *CLK*. For the first bit D13 (MSB), the falling edge of the $\overline{CS}$ signal and the rising edge of the *CLK* signal can occur simultaneously. After the clock edge of the last bit D0 we have to wait for 10 ns before we can pull-up the $\overline{CS}$ signal to start the conversion.



PARAMETER		DAC8801 MIN	DAC8801 TYP	DAC8801 MAX	UNITS
INTERFACE TIMING					
f_{CLK}	Clock input frequency			50	MHz
$t_{(CH)}$	Clock pulse width high	10			ns
$t_{(CL)}$	Clock pulse width low	10			ns
$t_{(CSS)}$	$\overline{CS}$ to Clock setup time	0			ns
$t_{(CSH)}$	Clock to $\overline{CS}$ hold time	10			ns
$t_{(DS)}$	Data setup time	5			ns
$t_{(DH)}$	Data hold time	10			ns

Figure 2: Timing diagram for transmission of a 14-bit word from FPGA to DAC (from [1.1]).

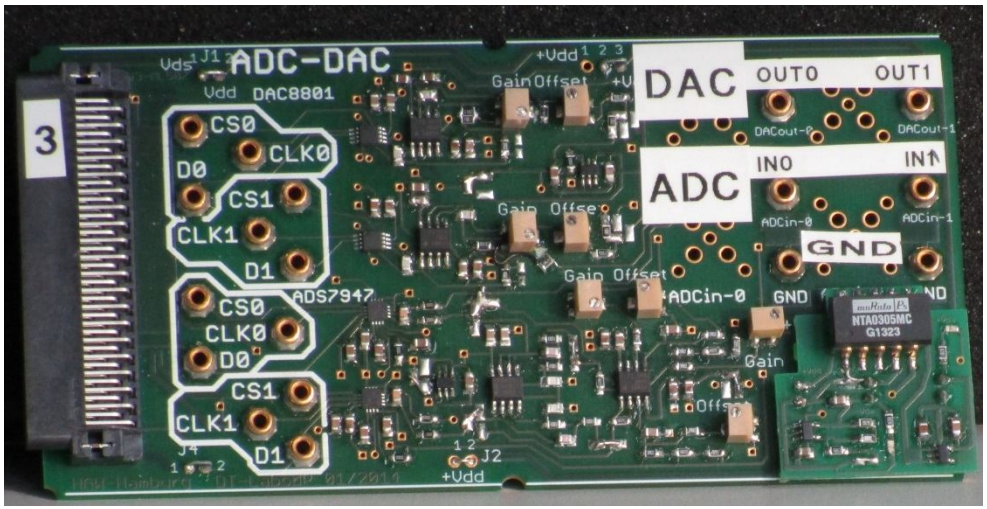


Figure 3: ADCDAC board with DAC8801 and ADS7947 [1.2].

Design of the DAC control module on FPGA

The DAC8801is part of the ADCDDAC board [1.2] that can be connected to the FPGA board and is depicted in Figure 3. It allows 2-channel ADC and DAC operation. In the following, for the design of the DAC subsystem on the FPGA, only 1-channel operation is considered. The DAC subsystem DAC_001 (red) is shown in the block diagram (Figure 4) together with the other modules of the ADCDAC subsystem. The signals DA_SD0, DA_CLK0 and DA_CS0 are connected directly to SDI, CLK and $\overline{CS}$ of the DAC8801. The input DA_IN is a 12 bit digital value to be sent to the DAC (2 LSB will be filled up with "00"). The DA_IN signal is selected by signal M_ODE via a multiplexer and can either be

- for M_ODE=0 the output of the 12 switches of the APB board ([1.2] DA_IN(11:0) or
- for M_ODE=1 the output of the ADC submodule (AD_OUT(11:0)).

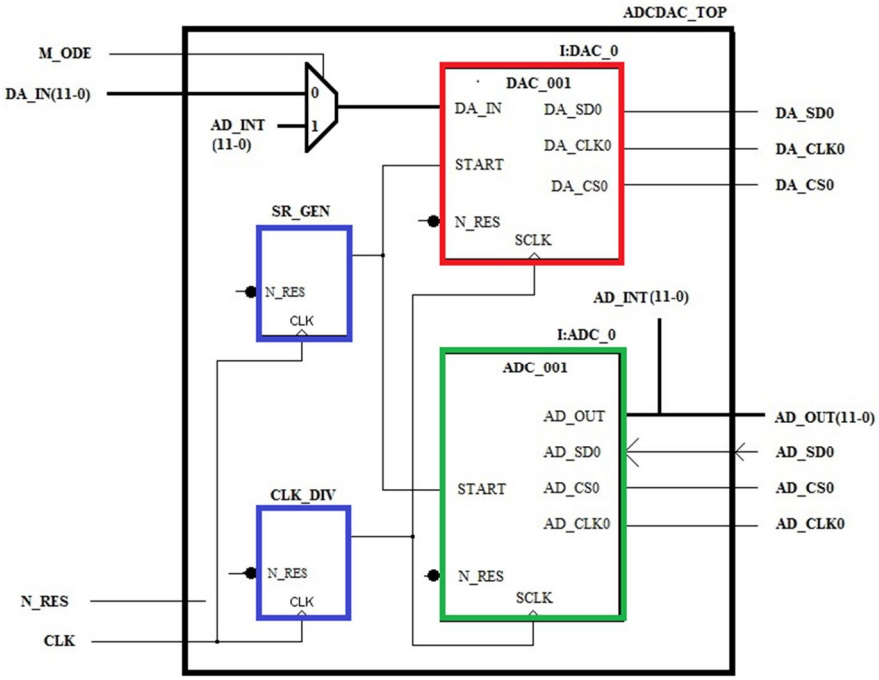


Figure 4: Block diagram of the overall system that connects to ADC and DAC of the ADDA board.

To provide the right timing and clock signal DA_CLK0 of serial bit stream, a clock divider generates the signal SCLK with a frequency of 3.125MHz (=100MHz/32). Another clock divider SR_GEN generates the sampling frequency of 48 kHz (which is common for high-end audio applications). The sampling is realized by asserting a START pulse every 1/48kHz = 208,333 µs.

PREP TASK 1.1: Design the module DAC_001 that communicates with the DAC8801. Use an FSM that sends the bits serially to the DAC after the START=1 pulse occurred. The bits are read from a 16bit shift register SR_DAC0.

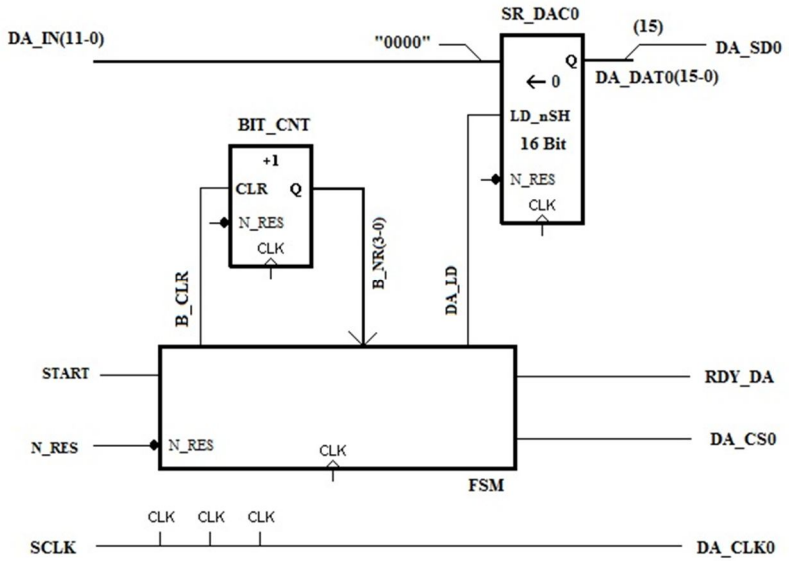


Figure 5: Structural diagram of the DAC communication module

PREP TASK 1.2: Create an entity ADCDAC_TOP that instantiates the module DAC_001 SR_GEN and adds the sample rate generator and the clock divider.

LAB TASK 1.1 (TEST OF DAC MODULE): Implement the top level module ADCDAC_TOP including your DAC module from Prep Task 1.2 on the ML507 board. The SLB board is placed on the upper connector, the ADCDAC board on the lower.

- Test your DAC circuit with 12 switches of the SLB board and apply different digital values to your DAC module and compare the value to the analog output voltage of the DAC. Hint: which value of M_ODE is required for this test? Acc. to the .UCF file, to which switch is M_ODE connected?
- Take screen shots of transmission frames and compare them to the pattern applied by the switches B0 SW0 to SW7 and B1SW0 to SW3 .

Hint: Use the following project settings to connect to the right device (Virtex 5):

Project Settings

Property Name	Value
Top-Level Source Type	HDL
Evaluation Development Board	None Specified
Product Category	All
Family	Virtex5
Device	XC5VFX70T
Package	FF1136
Speed	-1
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	VHDL
Property Specification in Project File	Store all values
Manual Compile Order	
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	

LAB TASK 1.2 (TEST OF ADC-DAC-PORCESSING CHAIN): Implement the top-level module ADCDAC_TOP including DAC and ADC (given from the lec-

ture) and do following measurements (oscilloscope screen shots required in report):

- Apply a voltage in the range -1V to +1V (not more/less!) to the ADC and measure the serial data signals from ADC and to DAC. Compare the results!
- Apply a sinusoidal signal (amplitude max. 1V, use a termination resistor!) to the ADC and determine the amplitude and delay of the DAC output.

References

[1.1] Texas Instuments (Burr Brown), Data sheet DAC8801, 14-Bit, Serial Input Multiplying Digital-to-Analog Converter, downloaded from http://www.ti.com/lit/ds/symlink/dac8801.pdf (last access: April 5, 2015).

[1.2] D. Palme, Documentation ADCDAC board, http://www.haw-hamburg.de/ti-ie/labore/digitaltechnik-digitale-systeme/download.html

[1.3] D. Palme, Documentation APB board, http://www.haw-hamburg.de/ti-ie/labore/digitaltechnik-digitale-systeme/download.html