525.742 SoC FPGA Design Lab

Laboratory Project 1B – VHDL Review

# Laboratory Assignment 1B - VHDL Review (DAC Interface)

**Introduction**

Now that you are familiar with the tools, you will be asked to design a small component in synthesizable VHDL and simulate it to verify that it is functional. The entirety of what you need to complete the lab should have been covered in 525.642; however, depending on how much time that has passed since you took that class, you may need to review some of the material. Emphasis here should be placed on good, simple, synchronous design practices. This component will be reused for many of the following labs this semester.

### Goals

* Additional practice/re-familiarization with Vivado for FPGA design and simulation
* VHDL syntax refresher
* Synchronous design refresher
* To prepare a component which will be responsible for playing audio in many of our future labs.

**Lab Description:**

Consider the entity described below, which will be an interface to a serial DAC :

entity lowlevel\_dac\_intfc is

port (

resetn  : in std\_logic; -- active low synchronous reset

clk125 : in std\_logic; -- the clock for all flops in your design

data\_word : in std\_logic\_vector(31 downto 0); -- 32 bit input data

sdata : out std\_logic; -- serial data out to the DAC

lrck : out std\_logic;  -- a 50% duty cycle signal aligned as shown below

bclk   : out std\_logic; -- the dac clocks sdata on the rising edge of this clock

mclk : out std\_logic; -- a 12.5MHz clock with arbitrary phase, runs all the time

latched\_data  : out std\_logic -- 1 clk125 wide pulse which indicates when the current

-- value of data\_word has been read by this component

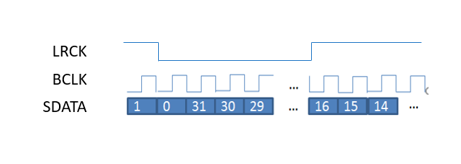
-- (and can be safely changed) );

end lowlevel\_dac\_intfc;

This device is essentially a parallel to serial converter with some special timing levied upon it. This component will generate the following “clocks” to be used by the external device:

1. MCLK : a 12.5MHz (125MHz/10) clock with 50% duty cycle and arbitrary phase.
2. LRCK : a 48.828125 kHz (125Mhz/(10\*256)) clock with 50% duty cycle.
3. BCLK : a 1.5625MHz clock (125MHz/(10\*8)) with 50% duty cycle

The component will also generate a signal SDATA, which will send one bit of the 32-bit data word to our external DAC each BCLK. The alignment of LRCK, BCLK, and SDATA are shown in the figure below.



When not in reset, this component will blindly run all the time, taking whatever data happens to be on the “data\_word” input, and shifting it out via the above pattern. **Please note : the 32 data bits of a word are shifted out in order, 31 (the leftmost) first, and 0 last.**

Signal Description :

**Resetn** : Active low, synchronous Reset for the module

**data\_word** : this is the 32 bit parallel data word which gets sent out serially to the external DAC.

**LRCK :** a signal with 50% duty cycle and 48.828125 kHz frequency. Changes at the same time as BCLK falls. Should be low when bits 0 and 31-17 are on SDATA, and high at all other times (ie. when bits 16-1 are on SDATA)

**BCLK :** a 1.5625MHz signal used as a clock by the external DAC. The external device will look at SDATA and LRCK on the rising edge of BCLK. Therefore, you should change those signals at the same time you create falling edges of BCLK.

**sdata** : this is the 1 bit serial data that goes to the DAC (see timing diagram)

**CLK125 :** this is your 125MHz system clock, and the clock to all flipflops in your component.

**MCLK :** 12.5 MHz clock, there are no alignment requirements with the rest of the signals.

**latched\_data** : your module creates this signal to tell the source of the parallel data : “I have latched this data, you can go ahead and advance to the next one”. The data source (some other module in the FPGA) will change to the next data\_word to be shifted on a rising edge of clk125 when this signal is high, so make sure to make this signal 1 clk125 wide and assert it when you don’t mind data\_word changing on the next clock. Remember clk125 is the only clock for any flipflop in your design.

In future labs we will use this component to stream audio samples in I2S format to the audio codec on the Zybo Z7 in order to play audio from your FPGA. This device is the SSM2603; the datasheet is freely available online, and is also included as a resource in the assignment on Canvas. In addition to this component, we will need to build a capability to write to the configuration registers on the codec over I2C. To find out why and prepare for Lab 2 please review the Zybo Hardware Users Guide and/or schematic and the SSM2603 datasheet in order to answer the following questions:

Question 1: Which of the signals in your dac interface component need to be routed to the codec? What FPGA pin should each of those signals be assigned to in your constraints file in order to accomplish this?

Question 2: Assume you have an I2C interface to write to the CODEC configuration registers (we’ll get to that next week). What values (in binary “0b…” or hex “0x…”) should we put in R4 and R5 in order to enable digital samples coming from your component to be converted to analog by the Digital to Analog Converter (DAC) and played over the line out port of your board? How do these values compare to the register defaults? Note there are other registers we will need to configure before this will work, but we’re only asking you to set R4 and R5.

### Requirements

* Zipped project directory named lab1\_yourlastname.zip, containing all files necessary for simulation, including a simple testbench). Instructor should be able to open the project in Vivado 2021.1 and simulate the behavioral model of the testbench
* Good simple synchronous design practices
  + single clock for all flipflops
  + no inferred latches
* Proper operation, meets all requirements. See rubric for details
* testbench exists and works. Nothing fancy required here, just something that creates data to be shifted out
* coding style -- comments, sensitivity lists...etc.
* A .doc(x), .pdf, or .txt document with answers to the Lab 2 prep questions

### Late Submissions

No late assignments will be accepted, so be sure to submit what you have (even if incomplete) by the due date specified.