Project-I

EE619A

09/Oct/2022

- This is a group project, wherein a group will consist of two designers.
- Please divide the work between the two designers, such that the roles/responsibilities of the two members can be clearly identified. Please divide the design into multiple modules and assign a designer to each module. The modules each designer worked on must be mentioned explicitly in the report.
- A testbench code with FPGA module will be provided to you shortly. You can expand the testbench code to test for all possible scenarios.
- A brief report of the modules with block diagrams and your Verilog codes must be uploaded on mooKIT for verification.
- A bonus question is added to the problem statement. In case, your team finishes the design including this bonus, and you wish to earn more bonus points, please contact the instructor. More bonus design problems will be provided.

---

1 Design Specifications

You are tasked with designing a **backend** for a mixed-signal IC as shown in Fig. 1. Only the signals relevant to the backend design are shown here.

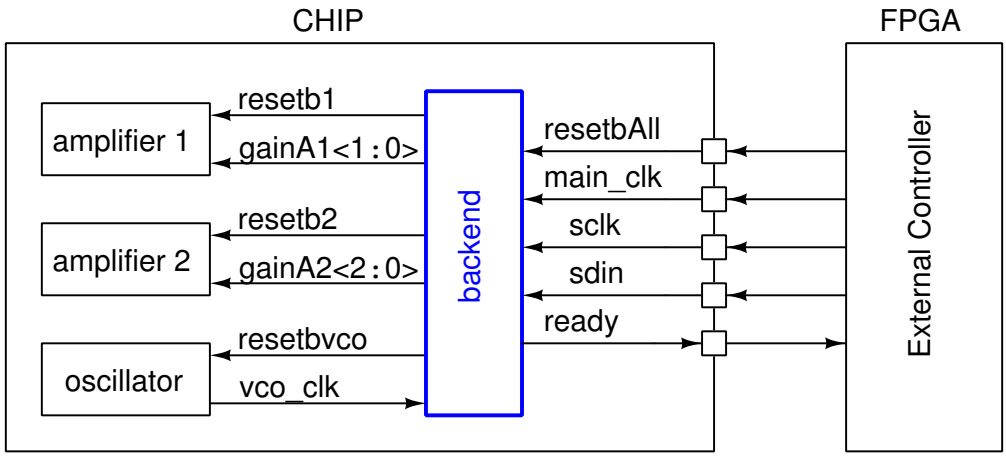

Figure 1: Overall block diagram.

The chip consists of two amplifiers and an oscillator. The backend has to ensure that these analog modules are initialized correctly and the chip is ready for use. The backend has to execute a start-up sequence for this, the details of which are given in Section 1.2. The chip will be interfaced with an external controller (most likely an FPGA) to control the start-up sequence.

1.1 Inputs and Outputs

The input and output pins of the backend are shown in Fig. 2, along with the nets that will be connected to these pins at the top-level.

Inputs

- **i_resetbALL :** Active low reset for the whole chip. When i_resetbALL = 0, the backend and other analog modules will be reset. When i_resetbALL goes from 0 to 1, startup sequence is initiated.
- **i_clk :** Main clock provided to the backend. Frequency: 200 MHz.

- **i_sclk :** Clock provided for serial communication with the backend. Used to synchronize the programming data coming in through the i_sdin pin. i_sclk remains 1 when not in use. i_sclk, toggles like a clock when data is being sent from FPGA to the chip through i_sdin.
- **i_sdin :** Serial data input pin. FPGA will send 1 bit data at a time to the chip, to configure the gains of the amplier 1 and the amplifier 2.
- **i_vco_clk** : Voltage controlled oscillator (VCO) output. This is a clock whose frequency is expected to be within 100 MHz to 250 MHz. i_vco_clk remains 0, when o_resetbvco is 0.

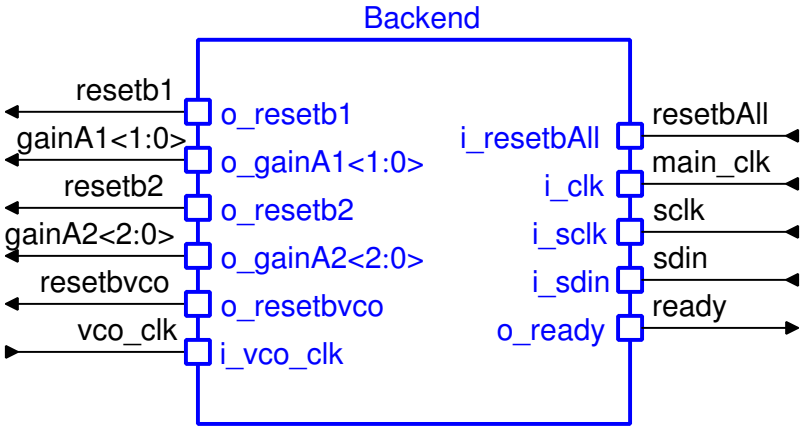

Figure 2: Input output configuration of the backend.

Outputs

- **o_ready :** When o_ready is 1, it indicates that the start-up sequence has been completed successfully and the amplifiers within the chip are ready to be used.
- **o_resetb1 :** Active low reset for amplifier 1.
- **o_gainA1<1:0> :** Two bit value that controls the gain of amplifier 1.
- **o_resetb2 :** Active low reset for amplifier 2.
- **o_gainA2<2:0> :** Three bit value that controls the gain of amplifier 2.
- **o_resetbvco :** Active low reset for the VCO.

1.2 Backend funcionality - start-up sequence

Please design the backend so that it executes the following start-up sequence.

1. When i_resetbAll = 0, all outputs of the backend should be pulled to 0. All internal registers should be either set or reset.
2. When i_resetbAll becomes 1, the start up sequence is initiated. The following instructions have to be executed in the given order.
3. Wait for the serial data from i_sdin. Read the data in. Details of this serial communication is explained in the Section 1.3.
4. Based on the serial data received, set the values of o_gainA1 and o_gainA2 (refer Section 1.3).
5. Wait for two clock cycles.
6. Set o_resetbvco=1.
7. Wait for 10 clock cycles.
8. Set o_resetb1=1 and o_resetb2=1.
9. Wait for 10 clock cycles.
10. Set o_ready=1.
11. No more actions required. Hold the values of all registers as is, till the next falling edge of i_resetbAll.

1.3 Serial data communication

To configure the gain of the amplifiers, overall 5-bit information is required − two bits for amplifier 1 and three bits for amplifier 2. Instead of dedicating five pins for these, we send the data serially, one bit at a time. This technique is often used to reduce the number of pins required on the chip.

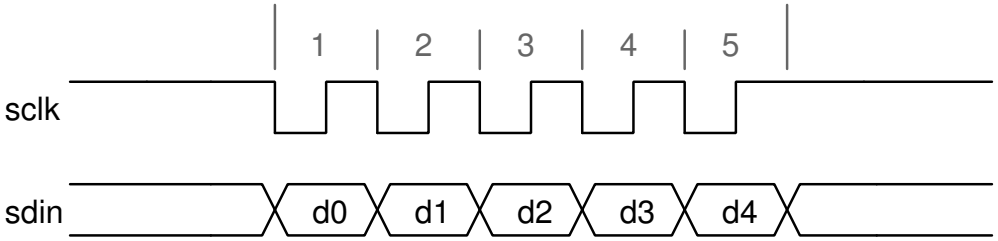

Figure 3: Serial data transmission using sclk and sdin lines.

The external controller will send the five programming bits after o_resetbAll is deasserted. The data transmission will be as shown in Fig. 3. The FPGA will sent the data at the falling edge of sclk. The backend has to receive these at the rising edge of sclk. The data corresponding to the first sclk cycle is d0, second sclk cycle is d1 and so on. You can use a shift register configuration to read the sdin.

d0 to d4 are related to the amplifier gains as follows.

- o_gainA1<1:0> : {d1,d0}
- o_gainA2<2:0>: {d4, d3, d2}

The sclk frequency is expected to be lower than 10 MHz. The exact frequency of sclk is not known at the time of design.

BONUS QUESTION

(Please attempt the bonus question after you have completed the rest of the project. The bonus marks will not add to the project marks. This will be tabulated separately and will be considered for awarding A grades. This will also be used to award the higher grade of the two grades, if a student is at the boundary between two grades.)*

BONUS QUESTION-I

The vco_clk is not available outside the chip and we need to estimate frequency of vco_clk. The frequency of the vco_clk is expected to be between 100 MHz to 250 MHz. Designers are tasked with estimating this frequency with a resonable precision. You can use the main_clk (200 MHz) for estimating the frequency. This operation has to be carried out immediately after the ready signal goes high. The final data from frequency estimation needs to be stored in an internal register with appropriate size.

You can do this by using two counters. Run the first counter (say counter_main) at the main_clk and the second counter (say, counter_vco) at vco_clock. Start both the counters together, and stop both counters at the same time - say when counter_main reaches 1000. The value of the counter_vco corresponding to this point can be used to estimate the frequency of the vco_clock. In this example, we stopped both the counters when counter_main reached 1000. Please use an appropriate value to estimate the frequency with better than 1 MHz precision.