EE214 Digital Circuits Laboratory
Wadhwani Electronics Laboratory
Electrical Engineering IIT Bombay

Project Date: October 10, 2024

Instructions:

1. **Team Composition:** Each group should consist of two members for the project.
2. **Project Structure:** The project is divided into three main tasks.
3. **Task Reports:** For each individual task, you are required to submit a report detailing the work completed and the distribution of tasks between group members, and the template for each week's task will be shared with you.
4. **Submission Timeline:** The submission of tasks will be in parts, and for the final submission, you will need to combine all task reports into a single comprehensive report.

SPI

Serial Peripheral Interface (SPI) is a communication interface that allows microcontrollers to communicate with peripheral devices over short distances. SPI is a synchronous, full-duplex protocol that uses a master-slave configuration to transmit and receive data simultaneously.
Read **introduction-to-spi-interface.pdf** *Reference- Piyu Dhaker, piyu.dhaker@analog.com*

ADC

An analog-to-digital converter (also known as an ADC or an A/D converter) is an electronic circuit that measures a real-world signal (such as temperature, pressure, acceleration, and speed) and converts it to a digital representation of the signal. We are using MCP3008 ADC.
Refer datasheet **MCP3008.pdf**

DAC

A Digital-to-Analog Converter (DAC) is a device used in digital signal processing systems to convert a binary number input into a corresponding analog voltage output. It is responsible for converting processed digital signals back into their analog form. We are using MCP4921 DAC.
Refer datasheet **DAC.pdf**

Problem Statement

Task 1: VHDL Code Development for SPI Master and SPI Slave (10 Marks)

In this part, you are required to design and implement both an SPI Master and an SPI Slave module in VHDL. The SPI Master will handle the communication protocol and send data to the SPI Slave, while the Slave will simultaneously transmit its own data to the Master. Communication between the Master and Slave will be simulated in Quartus to verify correct data transmission and reception.

The SPI protocol must be correctly implemented, including signals such as MOSI (Master Out Slave In), MISO (Master In Slave Out), SCLK (Serial Clock), and $\overline{CS}$ (Chip Select low), with the CS signal implemented as $\overline{CS}$, indicating an active-low configuration. The SPI clock frequency should be set at 10 MHz to ensure proper synchronization between the Master and Slave.

The communication sequence will begin with the Master pulling the $\overline{CS}$ signal low to initiate the transmission. Data will then be sent simultaneously by both the Master and Slave:

- The Master will transmit the data value 5 over the MOSI line.
- The Slave will transmit the data value 7 over the MISO line.

Both data transmissions will be synchronized with the SCLK signal. The Master will generate the clock pulses and ensure proper synchronization throughout the communication. Once the data transmission is complete, the Master will terminate the communication by pulling the $\overline{CS}$ signal high.

Choose Mode 0 for the clock configuration as specified in the SPI_VHDL PDF to ensure proper operation. You do not need to instantiate all the signals provided in the SPI_VHDL PDF; instead, include only the necessary signals required for the implementation.

A top-level entity should be created to instantiate both the SPI Master and SPI Slave modules. The data received by the Master from the Slave will be output from the top-level entity, and the communication can be observed by capturing the relevant signals. A testbench will be used to apply clock signals to the Master and verify that both the Slave and Master correctly receive and transmit data.
Read **SPI_VHDL.pdf** *Reference- Mitu raj, iammituraj@gmail.com*

Steps:

- Design and implement both the SPI Master and SPI Slave modules in VHDL.
- Handle communication at appropriate clock frequencies (10 MHz) and ensure proper synchronization between the Master and Slave using Mode 0 as specified.
- Demonstrate that the SPI Master and Slave modules can successfully communicate by transferring data over MOSI and MISO, synchronized with the SCLK and $\overline{CS}$ signals.
- The Master module will read and send the data stored in its register (e.g., data = '5') to the Slave module, while the Slave will simultaneously transmit its own data (e.g., data = '7') to the Master.
- Create a top-level entity to instantiate both the Master and Slave modules and verify the communication via a testbench, capturing relevant signals.

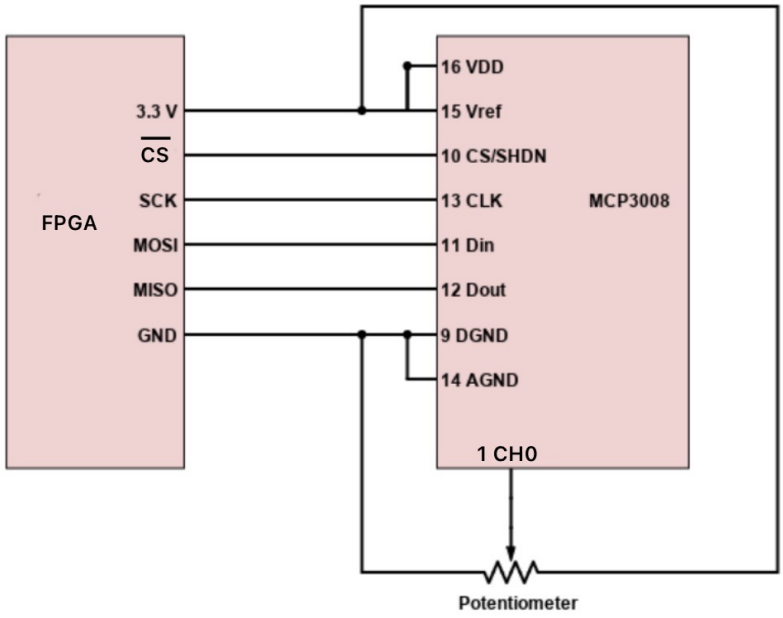
Task 2: System Synthesis, ADC Interface, and LCD Display (20 Marks)

This part of the project involves interfacing the FPGA with external components, including an ADC, DAC, and an LCD. In this task, you are required to use only the master module which you have previously implemented in VHDL. Additionally, you may need to modify the master VHDL code to suit the requirements of this task. Specifically, since the slave in this case is the ADC, you will need to send additional configuration bits from the MOSI line to properly configure the ADC for communication. Refer to (Section 5.0) and the given waveform from the **MCP5008 datasheet** for details on the necessary configuration bits that must be transmitted.

Part 2(a): ADC to LCD via SPI (10 Marks)

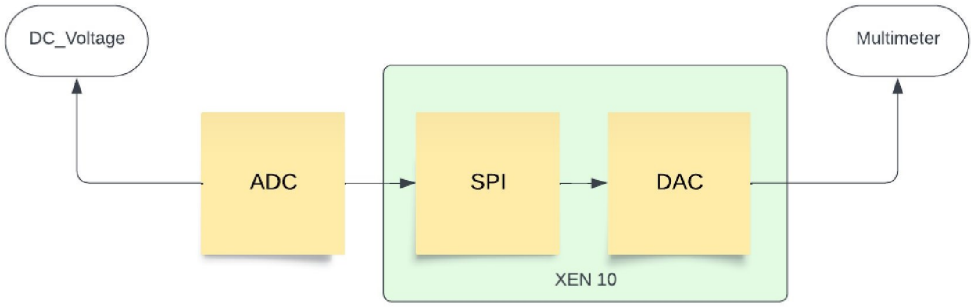
- You will interface the ADC with a reference voltage (Vref) of 3.3V present on the board while adjusting the analog DC input voltage to 2V using a potentiometer.
- Ensure to scale down the voltage to 2V before providing it to the ADC and verify the voltage using a multimeter.
- Using SPI, you will send the 10 bit- digital data from the ADC to the FPGA, where it will be stored in a register (e.g., reg_a).
- First, you need to display the result on the LEDs by using a 10-bit output signal to check the functioning of your SPI VHDL.
- After displaying in LEDs, use LCD to display the same value stored in the register(reg_a).
- You need to provide a done output signal to any of the LED to show that your 10-bit digital value has been stored in the register.

- After storing the value, display the value stored in the reg_a on the LCD by pressing the reset button(which is pressed and then released), giving a 50Mhz clock signal from the board, and using the LCD_VHDL file provided. Merge both VHDL(SPI_master and LCD_VHDL) and the register(reg_a) value should be given to the (10-bit)input of LCD_VHDL internally.
- Also, make sure that you use a separate reset signal for your SPI_master and a separate reset is used for the LCD. The reset of LCD should be provided by the push_button.(Pressed(value=1), released(value=0)).
- So, now you have to display the 10-bit digital value on the LCD.
- Both the VHDL files for the LCD configuration and the SPI module should be integrated into a single top-level module.



Part 2(b): SPI to DAC and multimeter (10 Marks)

- In this task, you will retrieve the digital value stored in the register(reg_a).
- The stored data will be sent to a DAC via SPI with a reference voltage (Vref) of 3.3V, which will convert it back into an analog signal.
- The analog output from the DAC should be visualized using a Multimeter to verify that the signal corresponds to the original analog DC input voltage.



Bonus Task 3: Regeneration of Sine Wave using ADC - DAC (10 Marks)

In this task, a sine wave will be provided as an analog input to the system, and the students are required to perform the following steps similar to Task 2(b) but with an emphasis on handling continuous waveforms.

- The ADC will sample the sine wave, which varies between 0 and 2 volts, and convert each sample into a 10-bit digital value.
- Since the SPI can only transfer one bit at a time, it will take 10 clock cycles of the SPI to transfer each 10-bit sample value from the ADC to the FPGA.
- To transfer a new 10-bit sample value, the $\overline{CS}$ signal must first be pulled high and then pulled low again before initiating another transfer from the ADC.
- After receiving the complete 10-bit digital value for each sample, it will be stored in a register on the FPGA.
- The stored 10-bit value will then be transmitted to the DAC via SPI, which will convert it back into an analog signal.
- Finally, the analog output from the DAC should be displayed on the Digital Storage Oscilloscope (DSO), and the students must ensure that the sine wave visualized on the DSO matches the original sine wave provided as input.