Lab 3: NCOs, SPI, and DACs

Concepts and Background

Generating non-trivial mathematical functions is very useful part of signal processing systems. Sinusoids are common, but relatively complicated to implement numerically. Since sinusoids are very not-linear, some algorithm is required to generate the successive values of the output. The simplest method is to pre-calculate the values of the sinusoid using a normal computer, and then store them inside a read-only memory such that the address is the input to the function and the data at that address is the output. More complicated algorithms can accurately calculate the trigonometric function without using as much memory to hold the complete function's look-up table. In the end, the device is a numerically controlled, sinusoidal oscillator. The implementation details fortunately don't matter for this course because the Altera NCO Intellectual Property (IP) does everything in a convenient and modularized component.

Interfacing to various peripheral devices over just a few wires makes some designs much simpler than peripheral devices that require many wires. Early data connections were all parallel. Since these computing systems were 4- or 8-bit, the number of connections was generally manageable. However, wider data buses and higher clock rates made for higher data rates, but more complicated circuit board and chip interface design. To make the layout easier and the pin-count lower, devices use serial interfaces. At low and medium data rates, serial interfaces usually have up to four signals: a reference clock, a chip select, a master-to-slave data, and a slave-to-master data. These serial peripheral interface (SPI) signals are conventionally named SCLK, CS, MOSI, and MISO, respectively. The details of operation (or presence) of each of these signals are defined by the datasheet of each device. Fortunately, despite the lack of an industry standard for using these signals, devices are designed for interoperability. The usual process of operation uses the CS to enable the serial communication and one bit is passed in each direction via MOSI and MISO for each SCLK cycle.

The usual method of implementing serial/parallel conversion is the use of shift registers. These are a sequence of registers that automatically shift the values in one or both directions. For SPI, both the master and the slave devices shift data through their registers. In a single direction, the sender loads parallel data into its shift register. Then, on the enable of the chip select, the registers shift synchronous serial bits to the other enabled register. Finally, at the disable of chip select, the data transfer should be completed and the parallel data has moved to the receiver's registers.

Communicating from the land of digital signals to useful analog signals usually requires a dedicated digital-to-analog converter. These devices vary greatly in speed (number of updates per second) and resolution (data bit width). The internal operation is some variety of special mixed-signal electronics followed by an analog amplifier or buffer to create a useable analog signal. The DAC on the expansion board is a medium speed, medium resolution component.

Implementation Requirements

This lab is an implementation of a sinusoidal-output, variable-frequency, and variable-amplitude signal generator.

The input stage of this design has only the user input components. To control the output signal parameters, switch 0 should change which variable the encoder changes and the motor encoder should change the stored frequency and amplitude up and down without overflow. Button 0 should reset the output settings to 1kHz and full amplitude. Switch 1 should enable/disable the SPI output from DAC.

The signal source for the design is an Altera NCO IP block. Implement frequency control to set the sine wave frequency between 20Hz to 20kHz. The amplitude control should be from 0% to 100% of the full-scale data path amplitude. The output data path must match the maximum size of data sent to the DAC. The sample rate must be high enough to reasonably produce the 20kHz signal and must satisfy the timing requirements for communication with the DAC. The sample rate of the NCO and the DAC must match exactly such that each sample generated by the NCO is sent to the DAC with neither dropped nor duplicated samples.

The signal is output using the SPI DAC. Implement parallel-to-serial communication to DAC using the Altera LPM_SHIFTREG IP block and additional control logic. The output must be correctly converted from 2's complement to unsigned binary such that the mid-scale output of the DAC is the zero value of the signed sine wave.

Demonstration Goals

- To establish that the group has read and properly understands the specification, the group must demonstrate that all listed features and requirements are implemented correctly. Please prepare your demonstration appropriately.
- Sinusoidal analog output from DAC using the full output range (nominal 0-5V)
- Adjustable frequency with correct minimum/maximum range
- Adjustable amplitude with no roll-over between 0% and 100%
- Reset button correctly changes settings to 1kHz, 100% amplitude sine wave

Code Organization Requirements

- All flip-flops use the 50MHz system clock
- Modularization of design: Verilog modules, one per file, to implement each unit of functionality in your design
- Organized, *separate* files and directories for Verilog modules and configured IP blocks
- Top-level module contains only wires, buffers/inverters, and module instances
- Assignment of *meaningful names* to the GPIO interface signals from the expansion board
- Correctly formatted module headers and organized code
- Project submitted on BlackBoard, meeting the submission policy requirements

Design Recommendations/Hints

- The design needs a constant frequency signal to control when the NCO generates a sample and when the DAC controller sends out that sample. This signal, known as the 'sample clock,' should enable the NCO and then trigger the DAC, but *must not* be used as a flip-flop clock.
- Use synchronous LDAC mode: hold the LDAC_n pin low and use a command that update's the DAC's registers. This will make the analog value change after each time the data is transmitted.
- Consider using a state machine to control the shift register enable/load/shift operation and use a counter to count off the 32 clock cycles of shifting.
- If the DAC receives incorrect commands due to implementation problems or accidental communication interruption during board reprogramming, it may enter an unknown state. Turn the FPGA board power off to reset the DAC to ensure that it starts from a known state.

- Use the DAC output terminal block to make the analog signals accessible to an oscilloscope probe.

Design Questions

1. What is the minimum required DAC sample rate to achieve the frequency output requirements? Hint: look up the Nyquist Sampling Theorem.
2. At the DAC's maximum serial clock frequency, what is the maximum single-channel sample rate using the write-and-update command while still meeting the delay-after-update requirement? Reference which parameters in the datasheet are needed to determine this specification. Hint: determine the data period first.
3. What sample rate does your design use? How does this affect the appearance of the DAC output at higher frequencies?
4. What are the GPIO signals used to connect to the pins of the DAC interface? Use the expansion board schematic and the GPIO documentation page on Blackboard.
5. For your mono-channel output, which DAC command does your design use?
6. How do you control the NCO such that each sample is sent to the DAC correctly?
7. Find the equation in the NCO datasheet that shows how to calculate sine frequency versus phase increment. Since your sample rate is the effective clock rate of the NCO, what range of phase increments do you need to meet the output frequency specification?

Report Requirements

- Complete answers to the above design questions.
- Discuss how your design controls the LPM_SHIFTREG IP block.
- Include a block diagram of the connections between the modules and their connections to external signals through the top-level.
- Include oscilloscope captures (total 5x) of the analog sinusoidal signal output at minimum/maximum frequencies at 100% amplitude and at 100%, ~50%, and ~5% amplitude at 1kHz. Use the 'measure' feature to determine the peak-to-peak voltage and approximate frequency.
- Include a SignalTap capture of all inputs/outputs to the DAC module demonstrating how the module receives data, is triggered to send data, and sends that data to the DAC chip. Explain what events occur when in the captured signals.
- Give a brief summary of how the group members collaborated to complete this lab.
- Organize the report with sequential answers to questions and separate paragraphs for each discussion item and *each* module. Format the report such that it is easy to find the required information.

Grading Distribution

- Demonstration 30%
- Code Organization 10%
- Report Diagrams 10%
- Oscilloscope Captures 5%
- SignalTap Capture and Explanation 10%
- Report Answers and Discussion 35%