Learning Objectives

- Use the Xilinx Embedded Development Kit (EDK) to develop an embedded system
- Design an embedded system with Analog Output
- Use the Serial Peripheral Interface (SPI) to interface with the PMOD DA2 peripheral digital to analog converter.
- Use of analog instrumentation measurements to verify system design requirements.

Introduction and Overview

The design of a function generator in this experiment once again involves a soft-core microprocessor-based design (MicroBlaze) which will be downloaded onto the programmable fabric of the Spartan 3 FPGA of the Nexys board. This design interfaces with the PMOD DA2 digital to analog peripheral and the four buttons of the Nexys board. In this experiment you will gain experience designing an embedded system with analog output. The MicroBlaze computer system is interfaced to an external Digital to Analog Converter (DAC) that will be used to drive the analog output. In this experiment, you will be required to implement a function generator that is capable of producing three different waveforms. These waveforms include a saw tooth waveform, a square wave with a variable duty cycle, and a sinusoidal waveform. Push buttons will be used to select the output waveform, set the frequency of the output waveform and set the duty cycle of the square wave.

Function Generator System Requirements

1. The function generator should be able to output three different waveforms as shown in figure 1.
   a. Saw tooth
   b. Square wave with a settable duty cycle
   c. Sinusoidal wave

One of the pushbuttons on the Nexys board will be used to set the current output waveform. At power up the saw tooth waveform should be output. Then the mode pushbutton will be used to change the output mode from saw tooth to square wave to sinusoidal and back to the saw tooth wave as shown in figure 2.
2. The frequency of all three waveforms must be variable. At power up the frequency of the waveform should be 100 Hz. One pushbutton will be used to adjust the output frequency by increasing the output by 100 Hz increments from 100 Hz to 1 kHz. At 1 kHz, the next assertion of the pushbutton will
return the frequency to 100 Hz, the initial state. You can use a ten state FSM counter to realize this frequency control.

3. The duty cycle of the square wave should be variable. At power up the duty cycle should be set to 50%. One pushbutton will be used to adjust the output duty cycle by decrementing the duty cycle by 10% steps by decreasing the duty cycle from 50% to 0%. At 0%, the next assertion of the pushbutton will change the duty cycle to 100%, and subsequent assertions will decrement by 10% as before. You can use an eleven state FSM counter to realize this duty cycle control.

4. Use the minimum code space (number of words in BRAM) and the minimum number of FPGA programmable logic resources (slices).

**Function Generator Specification**

Implement an embedded computer system on the Nexys board that can output the analog output requirements on the PMOD DA2 peripheral board using the pushbuttons to adjust the output mode, frequency and duty cycle as described in the system requirements section. The embedded computer system should be centered about a MicroBlaze soft-core processor and use the OPB Serial Peripheral Interface (opb_spi 1.00d) as the interface to the PMOD AD2. Configure the MicroBlaze processor, write the firmware, and download the hardware and firmware into the Spartan 3 FPGA on the Nexys board.

The function generator output should be adjustable using various pushbuttons on the Nexys development board. Setting the output waveform is referred to as the function generator output mode. The signal used to increase the frequency is freq_up while the signal used to decrease the duty cycle is DC_dn. The signals used to adjust the function generator output, and the associated I/O devices used to generate these signals, are listed in Table 1.

<table>
<thead>
<tr>
<th>Name</th>
<th>Function</th>
<th>DIO5 I/O Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mode</td>
<td>Sets the function generator output waveform</td>
<td>BTN0</td>
</tr>
<tr>
<td>Freq_up</td>
<td>Increments the frequency by 100 Hz</td>
<td>BTN2</td>
</tr>
<tr>
<td>DC_dn</td>
<td>Decrements the duty cycle by 10%</td>
<td>BTN3</td>
</tr>
</tbody>
</table>

**Table 1: Function Generator input signal names and the associated DIO5 I/O devices.**

**Function Generator System Architecture and Components**

The minimum hardware configuration for the function generator is comprised of a MicroBlaze soft-core processor, memory controllers, program memory, data memory, general purpose I/O peripherals, the OPB SPI interface, the PMOD AD2 DAC, and the Nexys board with its peripherals. You may decide to use a timer/counter peripheral (OPB Timer) with interrupts to generate the proper timing intervals needed to meet the system requirements. For this experiment each group will have to partition the digital system into the hardware and software architecture they believe will best meet the system requirements.

Again each group will have to determine the most efficient software implementation that meets the system requirements. A software algorithm will be needed to generate the output waveforms correctly. Two approaches that can be used to generate the output waveforms include a mathematical expression or a table based design. If you can identify a simple mathematical expression that can be used to generate the output waveform then this would probably be the most efficient way to implement the output waveform. However, you might also consider a table based approach that uses a minimum size RAM lookup table (integer array) to generate the output waveform.
PMOD AD2 digital to analog converter:

You should review the reference manual and the schematic for the PMOD AD2 on the Digilent website. This peripheral for the Nexys board contains two National Semiconductor DAC 121S101 chips, and you should review this specification, at least to ascertain the maximum frequency that it can operate; use Google and the part identification to find this specification.

OPB SPI serial peripheral interface

The OPB SPI interface has many capabilities. You only need to use the master mode in the N byte write function, which should considerably focus your review of the EDK provided .pdf documentation. You should not use the interrupt capability. The following modified steps given in the documentation should allow you to interface to the PMOD AD2:

SPI master device with or without FIFOs where the slave select vector is asserted manually via SPICR bit(24) assertion

This flow permits the transfer of N-bytes in a single toggling of the slave select vector (default mode). Follow these steps to successfully complete an SPI transaction:

1. Start from proper state including SPI bus arbitration.
2. Configure DGIE and IPIER registers as desired.
3. Configure target slave SPI device as required.
4. Write initial data to master SPIDTR register/FIFO. This assumes that the SPI master is disabled.
5. Insure SPISSR register has all ones.
6. Write configuration data to master SPI device SPICR as desired including setting bit(24) for manual asserting of SS vector and setting both enable bit and master transfer inhibit bit. This initializes SCK and MOSI but inhibits transfer.
7. Write to SPISSR to manually assert SS vector.
8. Write the above configuration data to master SPI device SPICR, but clear inhibit bit which starts transfer.
9. Wait for interrupt (typically IPISR bit(30)) or poll status for completion. Wait time depends on OPB to SPI clock ratio.
10. Set master transaction inhibit bit to service interrupt request. Write new data to master register/FIFOs and slave device then clear master transaction inhibit bit to continue N 8-bit character transfer. Note that an overrun of the SPIDRR register/FIFO can occur if the SPIDRR register/FIFOs are not read properly. Also note that SCK will have "stretched" idle levels between byte transfers (or groups of byte transfers if utilizing FIFOs) and that MOSI can transition at end of a byte transfer (or group of transfers) but will be stable at least one-half SCK period prior to sampling edge of SCK.
11. Repeat previous two steps until all data is transferred.
12. Write all ones to SPISSR or exit manual slave select assert mode to deassert SS vector while SCK and MOSI are in the idle state.
13. Disable devices as desired.

The following timing diagram (Figure 16) from this documentation also identifies the relevant signals. Review the bottom five signals for the functional use of the SPI to interface with the PMOD AD2; actually, only the three signals SCK, MOSI and SS are used for the writing the signal for your function generator output. Access the .pdf opb_spi documentation and use this information to start your review of the specification. The progress of the students will be monitored so that you should complete the interface within a week’s time. Then you can concentrate on the design of the function generator.
Figure 16: Data Write Cycle on SPI Bus with CPHA = 1 and CR(24) = 0 for 8-bit Character

A0 = Address of Transmit_Data Register
Dt = Transmitted Data
Dr = Received Data
SCK is shown for CPOL = '0'
** Not defined, but normally LSB of previously transmitted character
Project Demonstration and Lab Report Submission

When your project is working properly and your lab report is ready to submit, zip up the project and lab report into one file and submit it electronically for grading. Your lab report should also include the following design sections that describe the stages of the standard design methodology as they relate to this experiment:

1. System Requirements
2. Specification
3. Architecture
4. Components
5. System Integration

Your report should include flowcharts and written descriptions of the algorithms you used to implement your design. The flow chart and written description should readily communicate the approach you used to design the function generator. Flow charts are used to communicate software designs to engineers since other engineers should not have to read the code to determine how a system was designed. Your final write-up should also contain comments indicating that you understood the overall concept of the EDK tools and design methodology.

Your report should be clear and concise in its description of the function generator design. Remember, diagrams are a viable method to conveying important design information; consider using them in conjunction with written descriptions of your design.

You lab report should also include a Conclusions section. This section should not be a summary of the procedures or a description of what was done in the experiment. This section should include an analysis of the final system and may include topics such as: were the system requirements met? Is there any error in the final system and if so estimate it? Could you have met the system requirements with a better design or an alternate approach? If so how? What tradeoffs were made in the areas of cost, performance, accuracy, design time, power consumption or reliability (if any of these are appropriate)?

Grading

Successful completion of the system requirements and a well written lab report will earn a maximum grade of 9 out of 10 points. In order to achieve a grade above 9 on this lab, you need to demonstrate a complete understanding of the lab by adding additional features/functionality to it. Some suggested lab enhancement ideas are listed below. Higher complexity will earn a higher grade. Adding other features/functionality is permitted as long as they are approved by an instructor and documented in an “Added Features” section following the conclusion of your report. Any changes made to the original specification due to added features must be listed along with an explanation of why the enhancement was made.

Suggested added features:
- Add the ability to change amplitude and/or change the DC offset (0 to 100% by 10% increments)
- Add a resolution change option (higher quality sine/triangle wave)
- Add a triangle wave that increments from 0% to 100% over the first half of the period and decreases form 100% to 0% over the second half of the period. Add functionality to allow the user to set the maximum amplitude of the triangle wave.