Sigma-Delta ADC

Revision 1: fixed number of digits in block diagram.

Introduction

In this lab you will build a digital voltmeter using a sigma-delta ($\Sigma\Delta$) ADC. The digital value corresponding to the voltage will be shown on an LED display. You will test your design by comparing the results shown on the LED display to DMM measurements of the same voltage.

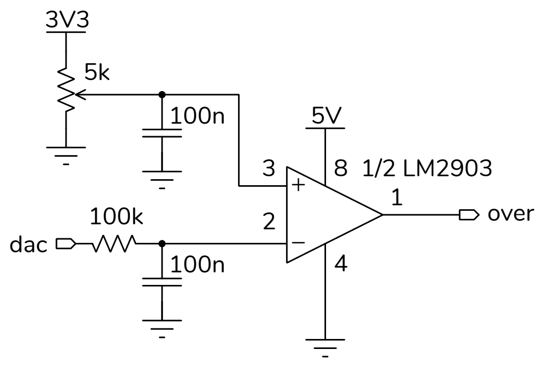
Your design will contain the following modules:

- a top-level module to implement the sigma-delta ADC and instantiate the remaining modules,
- a BCD (binary coded decimal) counter module (provided),
- a clock divider module (provided), and
- a multiplexed LED display module (provided).

You must design the logic for the ADC, integrate it with the supplied code, and demonstrate your design. Block diagrams and descriptions of the ADC are given below.

Analog Circuit

The analog portion of the ADC consists of an RC low-pass filter ($R = 100\,\mathrm{k\Omega}$, $C = 100\,\mathrm{nF}$, $RC = 10\,\mathrm{ms}$), an LM2903 dual analog comparator, and a $5\,\mathrm{k\Omega}$ variable resistor that you can adjust to set the analog input voltage to the ADC:



The LM2903 comparator output is open-collector so a pull-up resistor must be configured on the corresponding CPLD input (named **`over`**).

Digital Circuit

When the voltage on comparator pin 3 is higher than the voltage on pin 2, **`over`** becomes high. This causes the CPLD to set its **`dac`** output high. This charges the capacitor on pin 2 and causes this voltage to increase. When this voltage reaches the voltage on pin 3, **`over`** goes low and the CPLD sets **`dac`** low as well. This causes the capacitor to discharge until the voltage on pin 2 drops below the voltage on pin 3.

This feedback cycle runs continuously so that the filtered **`dac`** voltage on pin 2 continuously tracks the analog input voltage on pin 3.

The CPLD measures and displays the duty cycle of the **`dac`** output by counting the number of times **`dac`** is high over a fixed number of clock cycles. The ratio of these two numbers is the duty cycle.

We can compute the voltage by multiplying the duty cycle by the high logic voltage (approximately 3.3 V). This is the average voltage of the **`dac`** signal and thus also the average voltage of the analog input voltage.

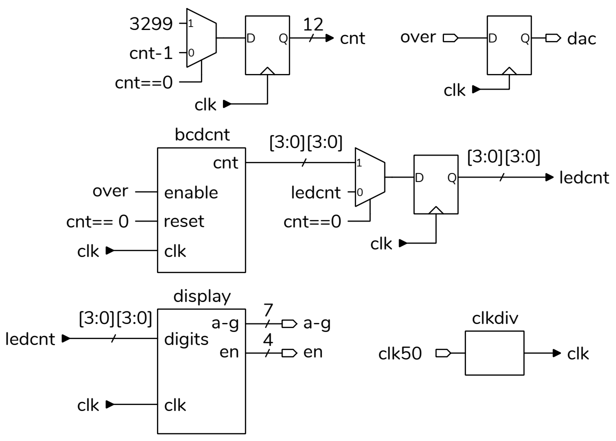
We can avoid multiplication by making the fixed number of clock cycles equal to the voltage in millivolts. The **`dac`** count over this interval is then equal to the input voltage in millivolts. For example, if the **`dac`** output was high for 1000 of 3300 clock cycles and the high-level logic level was 3.3 V then the voltage would be $1000/3300 \times 3.3 = 1\,\mathrm{V}$.

Two counters are used. One counts 3300 clock cycles (from 3299 to 0). The other counts the number of clock cycles during this time for which the 1-bit **`dac`** output was high.

A BCD counter is used to count **`dac`** pulses instead of a binary counter. This allows each digit of the count value to be displayed on the LED display without having to convert from binary to decimal. The supplied BCD counter has reset and enable inputs. A register holds the displayed BCD counter value and is loaded at the end of the counting period.

A 10 kHz clock will result in approximately three measurements per second which is sufficient for this DMM demonstration. A clock divider is used to generate this clock, **clk**, which also drives the LED digit multiplexer.

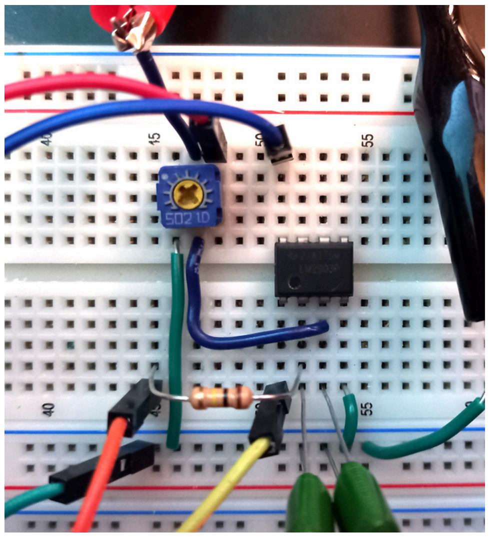
The block diagram below shows the various components:



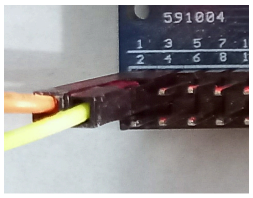
The **bcdcnt**, **display** and **clkdiv** modules are supplied in the **lab8modules.sv** file on the course website. You will need to instantiate these and implement the **cnt** binary counter, the **dac** output flip-flop, the **ledcnt** register and the associated combinational logic.

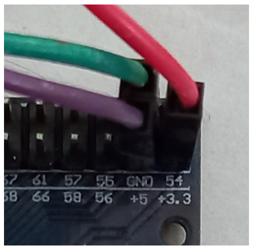
Procedure

Build the circuit shown in the schematic above on your breadboard. For example:



In the sample **.qsf** file **dac** is on pin 1 and **over** is on pin 3, both on connector P1. Connections to +5 V, +3.3 V, and ground are available on the connector P3 on the top right of the board:





Add the code required to implement the ADC described above to the **lab8.sv** file, compile your design, and program the CPLD.

Connect a 4-digit, 7-segment LED display to the **a-g** and **en** outputs as in previous labs.

Connect the CPLD and power it on. Adjust the variable resistor as you measure the DC voltage with a DMM at pin 3 of the comparator. You should be able to adjust the voltage from 0 to 3.3 V.

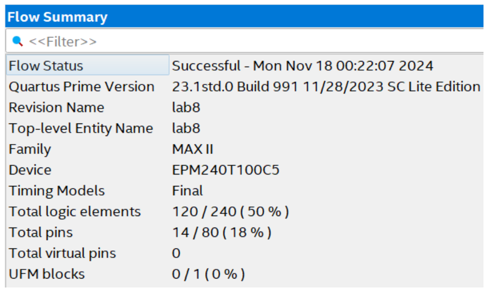
The LED display should show the measured voltage in millivolts.

The CPLD output voltage will not be exactly 3.3 V. Compensate for this by changing the count period to remove the offset at the maximum input voltage (increase/decrease the count period to increase/decrease the voltage on the LED display).

Report

Submit a report to the appropriate assignment folder, in PDF format, that includes the following:

- A listing of your Verilog code. Follow the course coding guidelines.
- A compilation report similar to:



Flow Summary

<<Filter>>

Flow Status	Successful - Mon Nov 18 00:22:07 2024
Quartus Prime Version	23.1std.0 Build 991 11/28/2023 SC Lite Edition
Revision Name	lab8
Top-level Entity Name	lab8
Family	MAX II
Device	EPM240T100C5
Timing Models	Final
Total logic elements	120 / 240 (50 %)
Total pins	14 / 80 (18 %)
Total virtual pins	0
UFM blocks	0 / 1 (0 %)

- The schematic generated from Tools > RTL Netlist such as that in Figure 1.

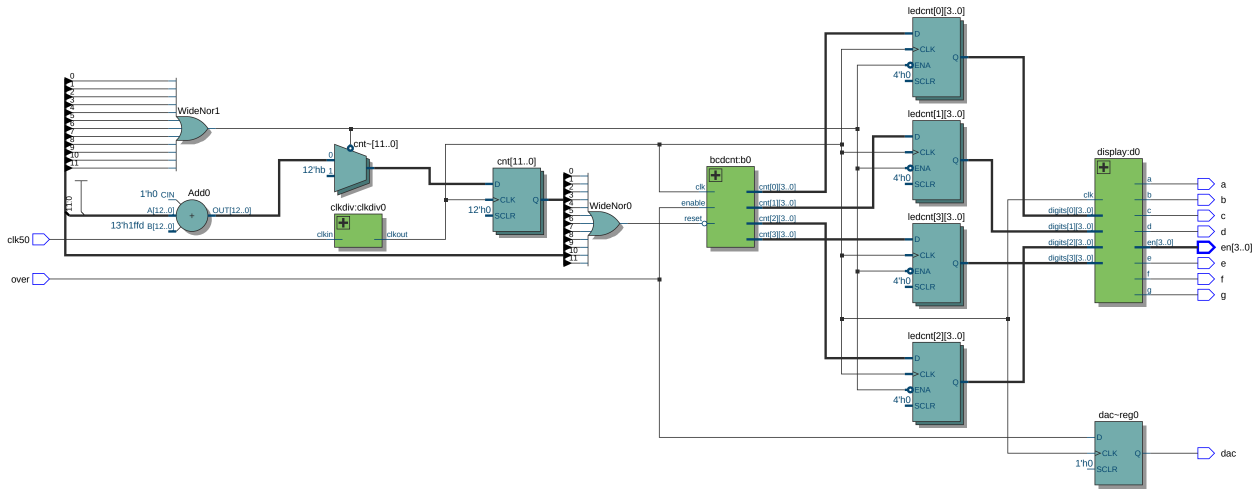


Figure 1: Synthesis results generated by Quartus.

If you do not demo your circuit in the lab, submit a video to the appropriate assignment folder showing both the LED and DMM results simultaneously as you adjust the input voltage from 3.3 V to 0. An example video is available on the course website. Ensure both displays are oriented right-side-up.

Hints

Measuring the comparator inputs and outputs with a 'scope or DMM may help you narrow down any problems.