State University of New York at Stony Brook
Department of Electrical and Computer Engineering
ESE 218 Digital Systems Design

Lab 12. Direct Digital Synthesis

1. Objectives

Illustration of Direct Digital Synthesis (DDS) concept by generation of a saw-wave with Digital-to-Analog Converter (DAC). Design of the system comprising of the datapath and controller units. Implementation of the system with Field Programmable Gate Array (FPGA) Ice40HX1K on IceStick board and DAC MCP4801.

2. Description

The analog signal is approximated with an 8-bit DAC MCP4801 from Microchip Tech, Inc. The DAC communicates with the controller (Master) through a 3-wire link compatible with Serial Peripheral Interface (SPI). The DAC operates as SPI Slave. Thus, you have to design only SPI Master which generates chip select CS', Serial clock SCK, serial data-in SDI are shown in Figure 1.

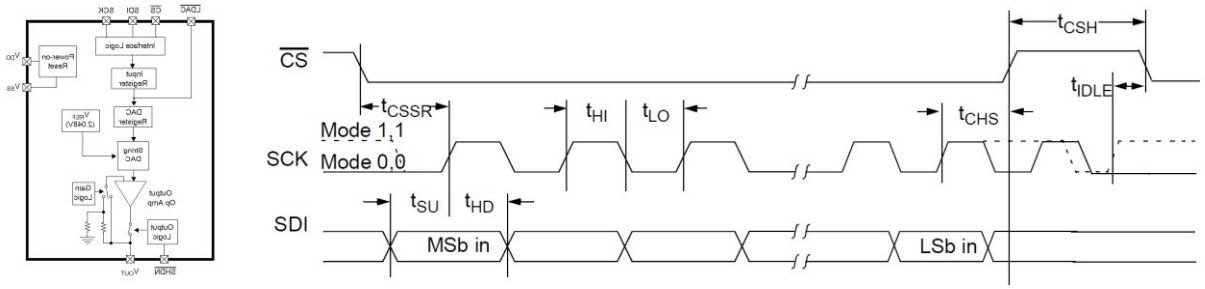

Figure 1. The DAC block diagram and the SPI compatible waveforms

For this implementation Mode 0, 0 is suggested. In this mode SCK stays at 0 before the transfer and data are latched in the DAC at positive clock edges (0 to 1 transitions). SCK =0 , CS' = 1 for at least t_idle, the data transfer is initiated by pulling down CS'. The data have to be stable in the vicinity of **positive** edges of SCK with setup t_su and hold t_hd times. The data format is shown in Figure 2. The data represent 16 bits: 4 configuration bits (we will use 0111) followed by an 8-bit magnitude from 0 to 255., the rest is ignored. The DAC converts the 8-bit magnitude to output voltage Vout in the range from 0 to 2.048 V with GA' =1.

REGISTER 5-3: WRITE COMMAND REGISTER FOR MCP4801 (8-BIT DAC)

W-x	W-x	W-x	W-0	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x	W-x
0	—	$\overline{GA}$	$\overline{SHDN}$	D7	D6	D5	D4	D3	D2	D1	D0	x	x	x	x
bit 15															bit 0

Where:

bit 15 (1) 0 = Write to DAC register
1 = Ignore this command
bit 14 — Don't Care
bit 13 **$\overline{GA}$:** Output Gain Selection bit
1 = 1x ($V_{OUT} = V_{REF} * D/4096$)
0 = 2x ($V_{OUT} = 2 * V_{REF} * D/4096$), where internal V_{REF} = 2.048V.
bit 12 **$\overline{SHDN}$:** Output Shutdown Control bit
1 = Active mode operation. V_{OUT} is available.
0 = Shutdown the device. Analog output is not available. V_{OUT} pin is connected to 500 kΩ (typical).
bit 11-0 **D11:D0:** DAC Input Data bits. Bit x is ignored.

Figure 2. 16-bit data format for transfer to the DAC

The SPI Master consists of datapath and controller units. The datapath is implemented with 2 counters and a shift register. One can use two identical 8-bit up counters with enable and clear: Magnitude counter Mag and Serial clock counter SC. The parallel-in serial –out (PISO) 12-bit shift register R converts the configuration bits and Magnitude bits to serial data SDI (Figure 3).

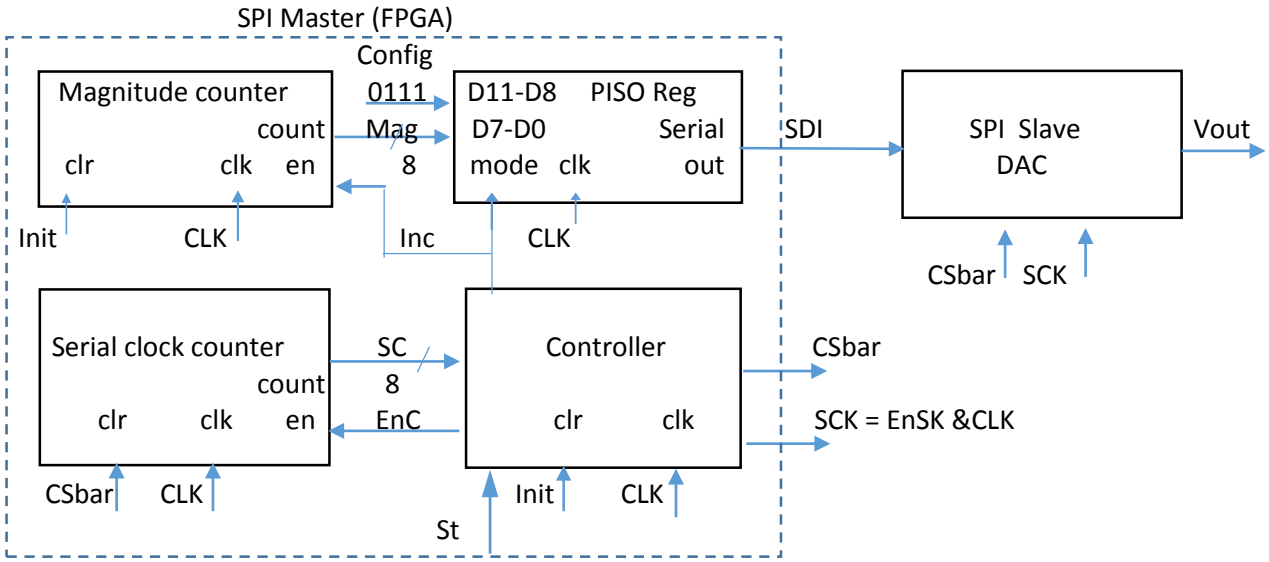

Figure 3. The block diagram of the system

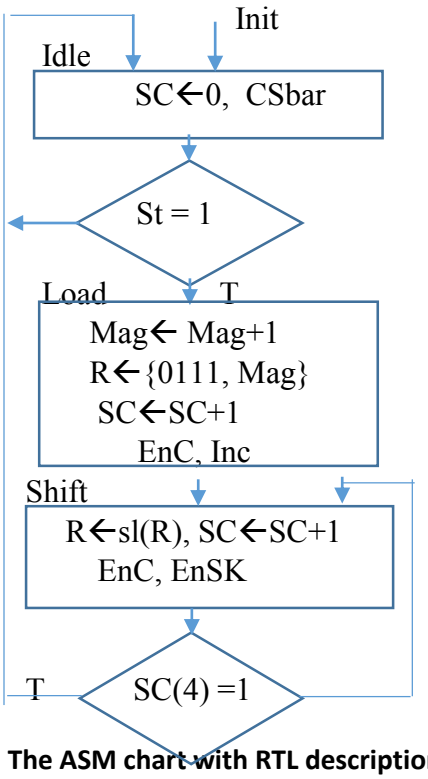

Figure 4. The ASM chart with RTL description

The counters have active-high synchronous inputs Clear (clr), Enable (en) taken at **positive** clock edges and 8-bit output (count). The PISO register has 12 data inputs loaded synchronously at **negative** clock edge with mode =1, otherwise it shifts left with a fill-in of 0 to LSB from serial-in. The ASM chart is shown in Figure 4. The system inputs are St for start, Init (active high). The outputs are as follows: EnC to enable counter SC, EnSK to enable serial clock SCK, Inc to enable Mag increment and data load to register R. Outputs SCK and CSbar support serial communication with the DAC. SCK is formed from clock and EnSK. The counter runs for 17 periods. SCK runs for 16 periods (in the 1st clock period SCK is zero). The flip-flops and counters are cleared upon initialization. With St = 1 the processes of incrementing Mag, loading R and shifting it out to the DAC over 16 clock periods repeat as long as St =1. The 3-state FSM for the controller can be implemented with 2 D-flip-flops and logic gates. One can use Verilog models for the counters and the register, gates from CAD library for flip-flop excitation and output forming logic. SC(4) is the bit of the counter to decode 16.

3. Prelab

a) **Design the datapath and the controller in Active-HDL.**
Obtain descriptions for the counter and register modules in Verilog. The model names should begin with your initials. Obtain the top level of the system with schematic entry. Simulate the system: obtain the waveforms for all signals. Submit the prelab report to the instructor by midnight before the lab session.

b) Synthesis of the bitmap for FPGA programming

Start Icecube2, double click Project, start New Project, right click on Add synthesis files, add *.v files.
Select Synplify Pro as synthesis tool, run Synplify Pro Synthesis: right click on Run Synplify Pro, select options, then Run (when done, **close SynplifyPro**).
Double click Run P&R. **Assign pins to your signals** with Pin constraints editor (the 4th icon from left) with the guidelines below.
Important: assign clock to pin 21. Use PMOD connector for communication with the DAC.
Figure 5 shows the pinouts of the PMOD connector on the FPGA board and the DAC (top views).

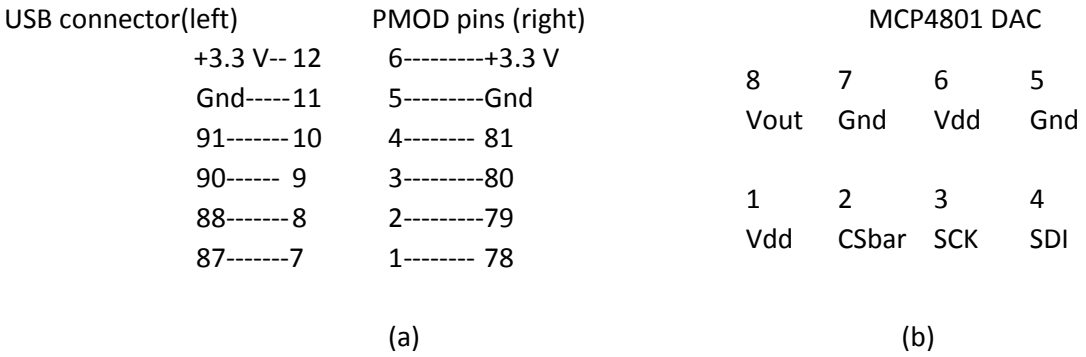

Figure 5. PMOD connector and FPGA pin assignment and pinout of the DAC

Assign CSbar, SCK and SDI in that order to the PMOD pin numbers corresponding to the last different digits of your ID number excluding 5, 6 already connected to Vdd and Ground. Example: for ID 109123456 delete 5, 6 and repeated digits. Use pin 2 (FPGA pin 79) for CSbar, pin 3 (FPGA pin 80) for SCK and pin 4 (FPGA pin 81) for SDI.
One can use 5 LED on the FPGA board for monitoring St and CS-bar signals, the FPGA pins assigned to LED are 99, 98, 97, 96 (all red), 95 (green).
IMPORTANT: click "locked " on the left side
Ctrl-S to save. Select Yes when asked to add files to the project. Double click Run P&R again.
Verify that the pin assignment was not changed.

3. Experiment

1. Connect the FPGA board with the DAC. Connect Vout to the analog input of the MSO19 or oscilloscope (make sure ground is connected). Obtain St, Init with the Pattern Generator. Connect Logic analyzer probes to CSbar, SCK and SDI signals. Power up the DAC from Vdd= 3.3V at the PMOD connector (pins 6 or 12). Make sure the ground of the DAC is connected to the ground of the PMOD (pins 5 or 11).
2. Connect the USB extension cord to your PC. Program the FPGA board with the Diamond programmer. Do not touch the FPGA board and USB cable during programming.
3. Obtain waveforms for St, Init, Clk, CSbar, SCK and SDI signals. Determine the frequency and amplitude of the saw-wave at the Vout pin of the DAC.

Save waveforms for the report. Take a photo of the setup with wires showing connections of the FPGA board and the DAC.

Sketch the diagram with the assignment of pin numbers of PMOS connector to the input and output signals. LDACbar is grounded, SHDNbar is connected to VDD.

Example of Vout waveform

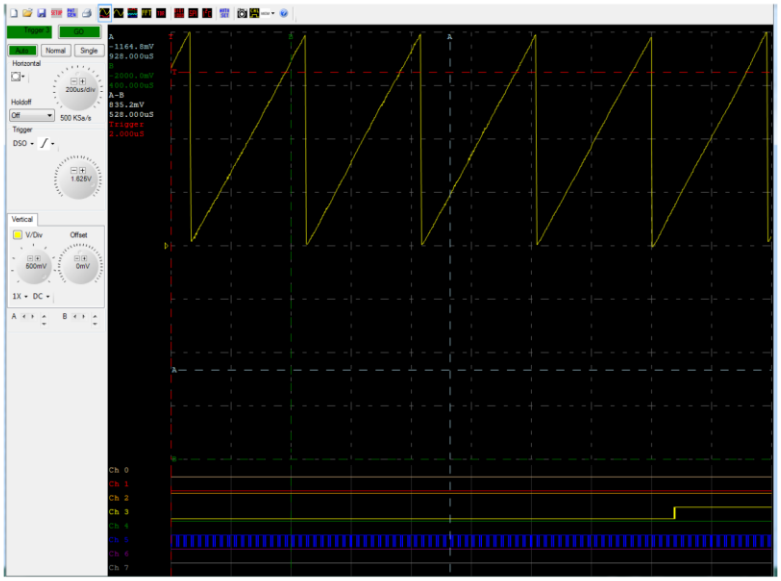

Figure 6. Example of the output waveform.

4. Report

The final report has to include the problem statement, approaches, Verilog models, circuit schematics obtained in Active-HDL, state table, state diagram for the control unit, simulated and measured waveforms, discussion of the waveforms, photo of the setup and a brief summary of results.