**Changelog for Laser-Lance’s AD57X4 Library**

Authored by Wesley Kuegler, NIFS Summer 2017 Intern at NASA’s Langley Research Center

**Introduction**

This document lists all of the significant changes I’ve made to Laser-Lance’s AD57X4 library, available at <https://www.laserlance.com/projects/arduino-dac-library-and-shield/>, in my time at NASA Langley. Many of these changes were made so that the library could operate multiple DACs in daisy-chain mode. Some of the changes were made because the original implementation was unclear, cluttered, or otherwise not optimal. One or two of the changes were made because the original implementation was incorrect and simply didn’t work.

It is **highly** recommended that any user of this library (or reader of this document, for that matter) have on hand a copy of the Analog Devices’ AD57X4 Data Sheet, which is available for free at <http://www.analog.com/en/products/digital-to-analog-converters/ad5754.html#product-overview>.

Most of my changes could probably be integrated with the original library to maintain Laser-Lance’s single-DAC operation while allowing for daisy-chain operation as well, but the scope of my internship did not give me enough time to do so. Many thanks to Laser-Lance for putting this library together initially – while I made some improvements, he provided the foundation onto which I could build.

**Major Changes to Operation**

* SendData() no longer latches the input shift register by setting SYNC high; instead, that functionality has been moved to a new function called SYNCdata(). For single DAC operation, simply call SYNCdata() after each call to SendData(). For daisy-chaining, use multiple calls to SendData() to load a command into each input shift register in the chain, then make a single call to SYNCdata() to latch the commands in all of the registers simultaneously
* Likewise, SetDAC() (now renamed pushDACvoltage()) must now be followed by a call to SYNCdata() once a set voltage command has been pushed out for every DAC’s input shift register
* LoadDACs() is no longer called automatically at the end of pushDACvoltage(); it must be called externally after a voltage set command has been pushed to and latched (using SYNCdata()) into each DAC’s input shift register
* Readback operations in ReadDACsRegister don’t work. I am not convinced that they worked in the original implementation, but I don’t have a single DAC setup to test it with. They definitely don’t work for daisy-chain operations

**In the Constructor**

* Renamed DACQTY argument to numChans (number of channels per DAC)
* Added numDACs argument (number of DACs in the daisy-chain)
* Renamed dacvolts to voltSwitch
* Renamed sspin to sync\_pin
* Removed (if sspin != SS)
* Added some initialization stuff to the constructor: setting pinMode() for the MOSI, CLR, and LDAC pins
* Added int labels for the CLR and LDAC pins (clr\_pin and ldac\_pin, respectively)
* Removed the call to PowerDACs(), moved to ConfigDACs() instead

**ConfigDacs()**

* Changed the config command, from “0x190000 | 0x1 | 0x4” to “0x19000E.” In the old command, the last 4 bits were “0101,” leaving the SDO line disabled, the TSD disabled, and the CLR select option on 0V (referring to bipolar operation). In the new command, they are “1110,” enabling TSD and SDO and switching the CLR select to negative full scale. The CLR select change is implementation-specific, not necessarily a fix
* Added a loop for the SendData(config) call to initialize every DAC in the chain
* Nested switch statement commented out, replaced with hard-coded loops that initialize 4 DAC channels to the +/-10V range, for as many DACs as are in the chain. This means that the voltage range is not selectable without altering the code/reworking the old switch statements. It will however work for any number of 4-channel DACs at the +/-10V range
* Added a call to PowerDACs() (after removing the one in the constructor)

**PowerDACs()**

* DACQTY is now numChans
* Added unsigned long variable to hold output
* Added loop that sends output for each of the DACs in the chain

**SetDAC() –** Renamed to **pushDACvoltage()**

* This function was renamed to more accurately reflect what it does, i.e. it pushes a valid voltage set command out along the chain, rather than comprehensively handling the set operation for one DAC
* Removed call to LoadDACs() at the end, because each DAC in the chain must have received a voltage set command via pushDACvoltage, and had it latched with SYNCdata(), before the load command can be sent
* Added a masking operation to the bitstream being pushed out to the daisy-chain. This mask converts the command from a twos complement to an offset binary representation

**LoadDACs()**

* Added a loop to send the load command to each DAC in the chain
* Added code to set LDAC low, then sync the data with SYNCdata(), then set LDAC high again

**ClearDACs()**

* Added a loop to send the clear command to each DAC in the chain
* Added a call to SYNCdata()

**ReadDACs() –** new function, **NOT WORKING**

* Because ReadDACsRegister() was only designed for a single DAC, the intention was that ReadDACs would use ReadDACsRegister() to read in the output from each DAC in the chain
* ReadDACs was intended to take an array of byte pointers, each of which would have 3 bytes of storage available (3 bytes = the 24 bits of the input shift register for each DAC) and store the readback data from one DAC in the chain
* Time constraints led to ReadDACs never being finished

**ReadDACsRegister() – NOT WORKING**

* Might not have worked originally. Time constraints kept readback functionality from being finished, so it definitely doesn’t work now
* Added a loop to SendData(data) for each DAC in the chain
* Added a SYNCdata() call to latch the read operation in each input shift register
* Commented out function call setting the MOSI pin low
* Nested the SPI.transfer(0x180000) loop and changed the indexing in an effort to store each received byte in the right location in the byte array array
* Removed function call setting the MOSI pin high
* Added a call to SYNCdata()
* Note: not enough testing was done to determine if the DACs are sending their data out over the SDO line correctly. That could be the case, meaning that the issue is in the nested loop for saving the information; it could also be the case that the process of sending the read command to each DAC in the chain is incorrect (though this seems extremely unlikely)

**SendData()**

* Removed function call setting the MOSI pin high

**SYNCdata() –** new function

* SYNCdata() only does one thing – it sets the MOSI pin high. This was moved out of the SendData() function and into its own, for reasons covered under “Major Changes to Operation”