Using the Dual D/A 12

Introduction

The Dual D/A 12 is an analog output board that provides steady DC analog output voltages instead of the PWM provided by a native Arduino analog outputs. This eliminates the need for filters and a lot of concerns about noise. These instructions are oriented toward an Arduino board user. The Dual D/A 12 can be used with many other boards (Beagle Bone, Raspberry Pi etc.), in fact any processor with an SPI interface or 4 digital I/O lines (using bit-bang software).

Dual in the name means two independent analog output channels. Each channel can be set to 5 or 10 volts full scale and has 12 bit (4096 count) resolution.

Connections

The SPI interface definition shows the following signals:

- MOSI – Master Out Slave In
- MISO – Master In Slave Out
- SCK – Serial Clock
- SS – Slave Select

Of these we only use MOSI and SCK. These signals are part of the Arduino processor SPI definition and every board will have specific pins that must be used. See pin table below for examples. Note that all four SPI pins are specified in the table although only two are used. When an Arduino is setup for SPI, it claims all four pins and they cannot be used for other functions (such as the handshake pins). In addition, two signals are required to handshake with the D/A chip. These are Load (LD) and Chip Select (CS). These signals can be any unused digital outputs. Make sure to define them using the Dual D/A interface library (see software library below). We must also have power and ground of course. In the Dual D/A case we have two power supplies, 5volts and 12 volts. The 5 volt supply will usually be connected to the Arduino. The 12 volt supply must be from another source. Make sure both supplies have ground connections to the Dual D/A 12 board.

The Dual D/A board has two ends, a control end (J1) and an output end (J2). The control end hooks to the Arduino board and the output end hooks to the 12v power supply and whatever is being controlled.

Two of the control pins J1-1 (LD) and J1-4 (CS) can be connected to any digital I/O pins not otherwise used. Which pins will then be specified to the software (see Library). The other two control pins J1-2 (SD or MOSI) and J1-3 (SCK) must be connected to specific SPI interface pins for each Arduino board. A table of examples is given below:

SPI Pin Examples

D/A Pin	Pin Name	Uno	Nano	Pro-Mini	Mega2560
J1-2	SD (MOSI)	D11	D11	D11	D51
J1-3	SCK	D13	D13	D13	D52
N/A	MISO	D12	D12	D12	D50
N/A	SS	D10	D10	D10	D53

Software Library

The Arduino software library for the Dual D/A 12 is contained in a zip file named ARDDALib.zip. Unzip this file in a folder named ARDDA in your Arduino library folder. In your sketch, add the lines

#include "Ardda.h" and **#include <SPI.h>** above the setup section

Then in the setup section use the following commands to configure your Dual D/A 12 board

Ardda.Setup(l,c,a,b); where:

l is an int that is the digital pin number hooked to J1-1 (LD)
c is an int that is the digital pin number hooked to J1-4 (CS)
a is an int that is the full scale value of channel A (OTA, J2-1) must be 5 or 10
b is an int that is the full scale value of channel B (OTB, J2-3) must be 5 or 10

That completes the setup of the D/A. You must also include:

SPI.begin();

To command output voltages in your run time code, use the following commands:

Ardda.OutA(val); where val is a float number between 0 and the specified full scale for channel A

Ardda.OutB(val); where val is float number between 0 and the specified full scale for channel B

Software Example

Ardda.Setup(2,3,5,10); will setup the board using digital pin 2 as LD and digital pin 3 as CS with channel A full scale of 5 volts and channel B full scale of 10 volts.

Ardda.OutA(3.25); will set channel A to 3.25 volts

Ardda.OutB(6.50); will set channel B to 6.50 volts

Voltages greater than a channel's full scale will be set to full scale. Negative voltages will be set to zero.

Modifications

It is possible for the Dual D/A 12 to provide higher output voltages. This requires changing the output stage gain resistors and changing the 12 volt supply to a value at least 2 volts higher than the highest desired full scale output. The "12 volt" supply should never be higher than 30 volts. The output stage gain is set by resistors as follows: OTA gain = 1+(R2/R1) OTB gain = 1+(R4/R3). As an example let's say we want an output full scale of 20volts on OTA. First, we must have a higher supply voltage, for this case we will use the common 24volts. The output of the D/A chip is 2.048 volts full scale when set to 5 volts full scale and 4.096 volts full scale when set to 10 volts full scale. Therefore we need an op amp gain of (20/4.096) = 4.883. Choosing to keep that value of R2 as is, we need R1 to be R1 = 1/[(4.883 – 1)/14.7K] = 3.786K. The closest 1% value is 3.74K giving a gain of 4.93 and a full scale of 4.096 * 4.93 = 20.195 volts. Since R1 is now lower in value we could instead add a parallel resistor in the spare slot next to R1 and not have to remove R1. Recalling that parallel resistors = Ra*Rb/Ra+Rb, this would require adding 6.021K with a nearest 1% value of 6.04K. This will make R1=3.794 with a gain of 4.875 and a full scale of 19.968volts. Both of these values are within 1% of the desired value and about all we can expect with 1% resistors. Just remember to adjust your requested values to account for the gain change (in this case divide by 2) or change the library code.

It is also possible to operate the Dual D/A 12 on 5 volts only. That is, the "12 volt" supply is instead 5 volts. This will restrict the output voltages to 5 volts full scale and will require a change of op amp (U2) to realize even that. The issue is that the LM358A is a single supply op amp, not a rail to rail output op amp. It will have a maximum output of about 3.5 volts when operated from 5 volts. A possible substitute is a MicroChip MCP6002-E/P. If you make this substitution (or any other) be sure to check the op amp maximum supply voltage. For the MCP6002 it is 6 volts so when using this op amp you can only operate at 5 volts.

Limitations and Tricks

As with any electronic device tradeoffs are made in the design to meet the design objectives. In this case the objectives include reasonably fast setting, low cost, high voltage out and reasonable accuracy. One of the tradeoffs is the op amp choice as explained above.

Another tradeoff is the SPI interface. The advantage of SPI is high speed. The D/A chip will work up to 20MHz and a common Arduino board will run SPI up to 8MHz (or bits/second) meaning a set command takes about 2uS. The disadvantage is SPI was intended to be a short run interface (within a PC board actually) so you may have trouble with long wires. I have had no trouble with 6" lengths but many feet of wire will probably cause trouble. If you have a need for long wires, try twisting the ground connection and SCK connections together. Make sure the grounds are connected on both ends. As a last resort, try putting a 100ohm resistor in series with the SCK line at the Arduino board end. And use the set ClockDivider(SPI_CLOCK_DIV32) command to slow down the SPI interface.

The outputs will not swing quite to 0 volts, particularly if you are sinking current. The "zero" output will be about 20mV with a 10Kohm load but will be perhaps 0.8 volts when sinking 1mA. These results are

typical of single supply systems although a rail to rail output op amp (see above) may do somewhat better.

Any op amp will likely be unstable (oscillate) when driving capacitive loads. If you are trying to drive a capacitive load and have trouble, place 50 to 100 ohms in series with the output or you might try putting a small capacitor (100pF to 1000pF) across the feedback resistor (R2,R4).

Accuracy and Calibration

The accuracy is +/- 2% unless you calibrate the board. The parts are stable to within about +/- 0.2% over time and temperatures of 0 to 40 degrees C. A two point (gain and offset) calibration will achieve the +/-0.2% capability. A two point calibration is accomplished as follows: Command a voltage near ground (say 1 volt) and another near the full scale (say 4 or 9 volts). Measure the actual output with an accurate meter (0.05% or better) and perform the following calculations.

Vmh = voltage measured high, Vml = voltage measured low

Vch = voltage commanded high, Vcl = voltage commanded low

gain= (Vmh – Vml)/(Vch – Vcl) Where gain is the gain calibration factor

Use Vml - gain*(Vcl) = offset to calculate the offset, where offset may be positive or negative

After getting gain and offset, multiply your desired value by the calculated gain and add the calculated offset to get the calibrated command value to be used in the Ardda.OutA or Ardda.OutB commands. The calibration will be different for each channel.

If you purchased an assembled board from Microface Technology, the calibration factors for the standard gains will be provided.

