DACSPI2 Library Reference Manual

Revision: January 12, 2012

1300 NE Henley Court, Suite 3
Pullman, WA 99163
(509) 334 6306 Voice | (509) 334 6300 Fax

Introduction

The Digilent PmodDA2 contains two simultaneous D/A conversion channels. Digilent provides an Arduino driver library for this device that is able to access only one converter channel.
This document provides an overview of the operation of this driver library and describes the functions that define its programming interface.

Overview

The PmodDA2 device contains two separate channels having each a National Semiconductor DAC121S101, 12-bit D/A converter. Due to the physical pins layout of the SPI hardware interfaces, this library implements access only to the converter channel IC1 (corresponding to 1 pin of the J2 connector of PmodDA2 - VOUTA).
For more information about the hardware interface of the PmodDA2, refer to the PmodDA2 Reference Manual available for download from the Digilent web site (www.digilentinc.com).

The converter is a serial device that is accessed (written) via an SPI interface. The library allows the use of any of the two SPI hardware interfaces.

The values written to the converter are on 12 bits (from 0 to 0x0FFF) which is considered "*IntegerValue*". The library also deals with "*PhysicalValue*", which is a floating point value that corresponds to the "*IntegerValue"* and to a reference value that defines the reference voltage (associated to the maximum integer value).

PhysicalValue = IntegerValue * (ReferenceValue / (2^12 – 1))

The physical value corresponds to the value that will be reconstructed (and can be measured) at the output of the D/A converter.
By default, the reference value corresponds to reference voltage of 3.3V.

Library Operation

Library Interface

The header file DACSPI2.h defines the interfaces to the DACSPI2 driver. The library is accessed via the methods and constants defined for the DACSPI2 object class. In order to use this library, the user has to instantiate one library object.

SPI Initialization

In order to communicate with the PmodDA1, the SPI interface has to be initialized. Before calling any other library functions, the begin() function must be called. This function initializes the SPI port used by the library.

DA Converter Functions

In order to read values from the AD converter, the library provides two types of functions: WriteIntegerValue() and WritePhysicalValue(). Read Overview section for a description of these values.

DACSPI2 Library Functions

Errors

This list shows the possible error messages returned by the DACSPI2 library functions:

Table 1. List of errors

Value	Name	Description
0	DACSPI2_ERR_SUCCESS	The action completed successfully
1	DACSPI2_ERR_VAL_OUT_OF_RANGE	The value is out of range

void begin(uint8_t bAccessType)

Parameters:

- uint8_t bAccessType – the SPI interface where the PmodDA2 is connected. It can be one of the parameters from the following list:

Table 2. List of values for bAccessType parameter of begin function

Value	Name	Connector on Cerebot MX4CK board
0	PAR_ACCESS_SPI0	JB
1	PAR_ACCESS_SPI1	J1

This function initializes the selected SPI interface, setting the SPI frequency to 1 MHz.

uint8_t WriteIntegerValue(uint16_t wIntegerValue)

Parameters:

- uint16_t wIntegerValue - the 12 bits value to be written to DA converter

Return value:

- DACSPI2_ERR_SUCCESS - The action completed successfully
- DACSPI2_ERR_VAL_OUT_OF_RANGE - The value is not within the 0 - 0xFFF range

If the value is inside the allowed range (0 - 0xFFF), this function writes the 12 bits value to the DA converter, by writing 16 bits to SPI and returns the DACSPI2_ERR_SUCCESS status message. If the value is outside the allowed range, the function does nothing and returns the DACSPI2_ERR_VAL_OUT_OF_RANGE message.

uint8_t WritePhysicalValue(float dPhysicalValue, float dReference)

Parameters

- float dPhysicalValue - the physical value that must be reconstructed at the output of the DA converter
- float dReference - the value corresponding to the maximum converter value (reference voltage). If this parameter is not provided, it has a default value of 3.3.

Return value:

- DACSPI2_ERR_SUCCESS - The action completed successfully
- DACSPI2_ERR_VAL_OUT_OF_RANGE - The physical value is not within the accepted range.

The function computes the integer value corresponding to the physical value by considering the reference value as the one corresponding to the maximum integer value (0xFFF).
If the integer value is within the accepted range (0 - 0xFFF), this function writes the 12- bit value to the DA converter, by writing 16 bits to SPI, and returns the DACSPI2_ERR_SUCCESS message.
If the integer value is outside the allowed range, the function does nothing and returns the DACSPI2_ERR_VAL_OUT_OF_RANGE message.
If the dReference function argument is missing, 3.3 value is used as reference value.

Read Overview section for a description of Integer and Physical Values.

Library usage

This section of the document describes the way the library is used:

- The PmodDA2 should be plugged into one of the SPI connectors.

Table 3. List of possible connections for PmodAD1

Connection	Connector on Cerebot MX4CK board
SPI0	JB
SPI1	J1

- The analog value is generated on pin 1 of the J2 connector of PmodDA2, corresponding to VOUTA (converter IC1).
- The library files have to be placed in the same location with the .pde file in a folder named Libraries and a subfolder having the same name as the library, preferably at: C:\Users\Name\Documents\Arduino. If the MPIDE is open, it has to be closed and reopened and under the Sketch menu ->Import library -> Contributed - the name of the library should be found.

  For example, a good way to use this library is:
  - C:\Users\Name\Documents\Arduino\Libraries\DACSPI2 contains library files DACSPI2.h, DACSPI2.cpp and keywords.txt
  - C:\Users\Name\Documents\Arduino\ DACSPI2 contains the demo sketch DACSPI2.pde
- In the sketch, include the DACSPI2 library header file

  ```
  #include <DACSPI2.h>
  ```
- In the sketch, include the DSPI library header file. It is needed in order to access the SPI functionality.

  ```
  #include <DSPI.h>
  ```
- In the sketch, instantiate one library object called, for example, myDACSPI2

  ```
  DACSPI2 myDACSPI2;
  ```

- In the sketch, use library functions by calls such as:
  `myDACSPI2.WritePhysicalValue(dValue);`

Simple Demo

In order to run this demo, place the library and sketch and connect the wire corresponding to the analog value as explained in the Library usage section.
This demo performs the following operations:

- In the setup() function:
    - Initializes the DACSPI2 library.
- In the loop() function:
    - Repeatedly increases and decreases the physical value to be reconstructed by the DA converter.