

ADC_DMA_VDAC Example Project

1.0

Features

- Delta-Sigma ADC in single-ended mode
- DMA used to transfer ADC result to VDAC
- VDAC and LCD used to verify output

General Description

This example project is also a PSoC Creator starter design. It shows how you can use PSoC to transfer data from one peripheral (ADC) to another (VDAC), using DMA without any CPU intervention. This design has already configured the DMA for you, so you can adapt it to fit your needs.

Development Kit Configuration

The following configuration instructions provide a guideline to test this design. For simplicity, the instructions describe the stepwise process to be followed when testing this design with the PSoC Development Kit (CY8CKIT-001) board, but can be generalized for the PSoC 3 Development Kit (CY8CKIT-030) and PSoC 5 Development Kit (CY8CKIT-050) as well.

1. Set LCD power jumper J12 to ON position, position SW3 to the 5V position, and leave the rest of the board at default configuration.
2. Connect P0_1 to the voltage regulator VR on the board. Observe the VDAC output P0_2 using a multi-meter.
3. Connect a character LCD to P2[6:0].
4. Build the ADC_DMA_VDAC project and then program the hex file onto the PSoC device using the MiniProg3. After programming is complete, disconnect the MiniProg3.
5. Reset the PSoC device.

Project Configuration

The example project consists of DelSig ADC, DMA, VDAC, Status Register and Char LCD components. The top design schematic is shown in Figure 1. The Character LCD component is used for displaying the ADC output and the VDAC input register values. The DMA is used to transfer the converted digital value from the ADC component to the VDAC which then converts this digital value to an analog voltage.



The Character LCD and VDAC are configured in their default configuration. The ADC is configured in the default single ended mode with 8-bit Continuous conversion mode. Ensure that the input voltage to the ADC remains between 0-1.024V because the ADC is configured for this range. The ADC_DeISig component configuration window is shown below in Figure 2. The DMA is configured using the DMA wizard to transfer one byte from the ADC output register to the VDAC input register on the rising edge of each ADC EOC signal. The NRQ output of the DMA is connected to a 'sticky' Status Register which is polled to find out when the DMA transfer is complete.

The screenshot shows the 'Configure 'ADC_DeISig'' dialog box with the following settings:

- Name:** ADC_DeISig_1
- Config1** (selected tab)
- Comment:** Default Config
- Configuration name:** CFG1 (ADC_DeISig_1_CFG1)
- Modes:**
 - Conversion mode: 2 - Continuous
 - Resolution (bits): 8
 - Conversion rate (SPS): 10000 (Range: 8000 - 384000 SPS)
 - Actual conv. rate (SPS): 10000
 - Clock frequency (kHz): 160.000
- Input options - Single ended mode:**
 - Input range: Vssa to 1.024V (0.0 to Vref)
 - Buffer gain: 1
 - Buffer mode: Rail to Rail
- Reference:**
 - Reference: Internal 1.024 Volts
 - Vref (V): 1.024
- Alignment:**
 - ☒ Right (Coherency = LOW)
 - ☐ Left (16 bits (OVF Protected))
- Input range diagram:** A schematic showing the ADC range from Vssa to Vdda. The range is labeled 'ADC Range (Rail to Rail Mode)'. The voltage difference between Vdda and Vssa is indicated as ~250 mV, and the voltage difference between Vssa and the input range is indicated as 100 mV.
- Buttons:** Datasheet, OK, Apply, Cancel

Figure 2. ADC_DeISig Component Configuration

Project Description

In the main function all components are started. Also, the DMA is configured to transfer the ADC output to the VDAC data register. For the proper usage of the Character LCD and VDAC components, please refer to their corresponding component datasheets.

Once started, the ADC conversion is continuous and generates an EOC pulse after each conversion. In the 'forever' loop in main.c, the Status Register is polled to check whether the DMA transfer is complete. The Status Register is Sticky, so that we do not miss an ADC sample. If the DMA transfer is complete, the Status register is Cleared-on-Read, and we print

the ADC result and VDAC input data on the LCD. The VDAC then converts this input digital value to the corresponding analog voltage.

Expected Results

The ADC output register value and VDAC input register value displayed on the LCD should match. Additionally, the VDAC output voltage measured using a multi-meter should be equal to the analog voltage given to the input terminal of the ADC component. The input voltage to the ADC should be within the range (0 – 1.024V). Out of this range causes incorrect result.



Figure 3.Expected output on LCD

Related Material

Example Projects

- Filter_ADC_VDAC
- ADC_16Channel
- DelSig_I2CM
- DelSig_I2CS
- DelSig_SPIM
- SAR_SPIM_USB

Application Notes

- [AN52705 - PSoC® 3 and PSoC 5 - Getting Started with DMA](#)
- [AN61102 - PSoC® 3 and PSoC 5 - ADC Data Buffering Using DMA](#)
- [AN57821- PSoC® 3 and PSoC 5 Mixed Signal Circuit Board Layout Considerations](#)

Training

- [PSoC 3 Video - Create an ADC LCD Project](#)
- [PSoC 3 and PSoC 5 104: Introduction to Analog Peripherals](#)



Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

© Cypress Semiconductor Corporation, 2012. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges. PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

This Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and/or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.

