

Yet Another PSoC-Based Oscilloscope

Powered by Design4Fun

Author: Andrea Giacosi

Associated Project: Yes

Associated Part Family: CY8C26443 and CY8C27443

PSoC Designer Version: 4.1

Associated Application Notes:

Abstract

This Application Note describes how to realize a “just for fun” oscilloscope with a PSoC™ and some additional components. The project has two parts: the PSoC hardware and firmware and a Microsoft.Net front-end application. The oscilloscope hardware is connected to the host PC through an RS232 interface to exchange data and commands. The RS232 communication protocol is specified in a separate document to let readers who want to design their own hardware/firmware or their own front-end application to do so.

Introduction

Figure 1 shows a block diagram of the system structure. A resistor network is used to feed the PSoC with a differential signal in the desired range. The differential signal is driven to an instrumental amplifier, which amplifies the input signal according to the value specified by the oscilloscope firmware. The amplifier is built using two PSoC analog continuous time (CT) blocks (AMP_CH1 NON_INV, INV).

The low part of the resistor network is connected to the output of two 6-bit voltage output digital-to-analog converters (ADC), built using two analog switched capacitor (SC) blocks (VOFF_INV, VOFF_NINV). The two voltages are selected by the firmware to shift the input voltages of the instrumental amplifier into the desired range. With a 1:10 resistor ratio and a usable input range less than 4V (when the power supply is 5V DC), the maximum differential input signal is +/- 25V. The common mode that can be “compensated” by the offset generator has the same magnitude.

The firmware reads samples from the ADC and sends data to the host PC through the UART. A MAX232 driver is used to translate from TTL to RS232.

The firmware also implements some of the commands described in:

“OscilloscopeCommunicationDescription.pdf.”

Note that this document covers the reserved time base commands for further extensions using the faster ADC.

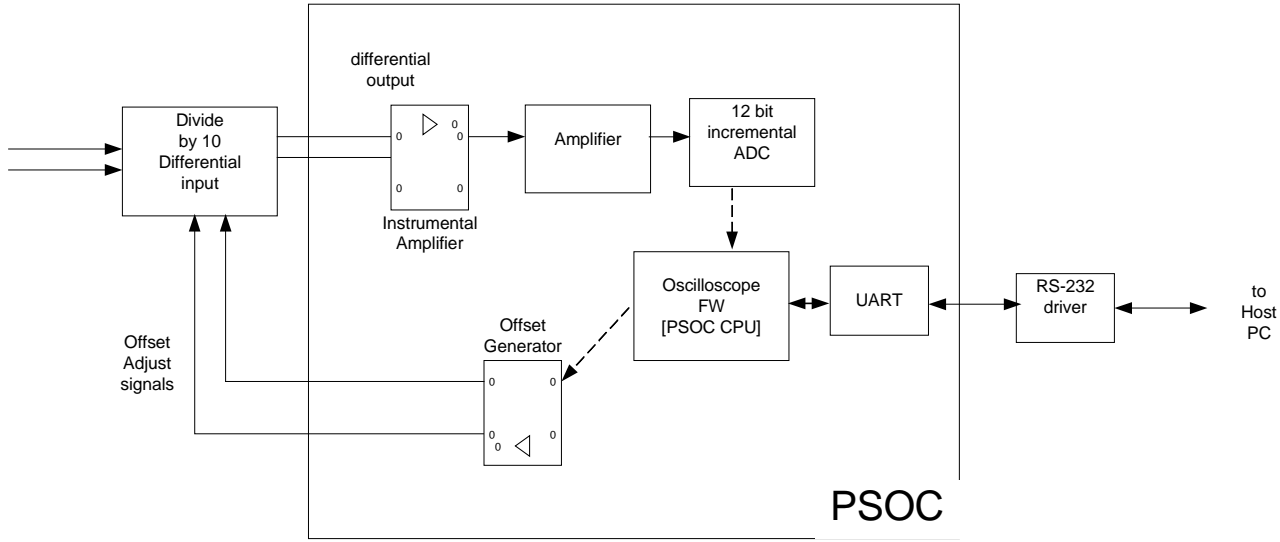


Figure 1. System Block Diagram

PSoc Project

The implementation of the PSoc blocks described is straightforward. Standard library blocks are used. A 5V power supply voltage is selected. The CPU frequency is set to 12 MHz. 24V1 and 24V2 divisors are 8 and 10, respectively, to achieve 3 MHz and 300 kHz frequencies. A UART is configured as 38400 bps 8N1.

In this project the 12-bit incremental ADC is used and the sample frequency was set to 180 sps. Users can employ a faster ADC, for example, the 8-bit sigma-delta ADC, if a higher sample rate is required. The UART speed can be increased to 115200 bps without any problems.

Placement of the different blocks is shown in Figure 2.

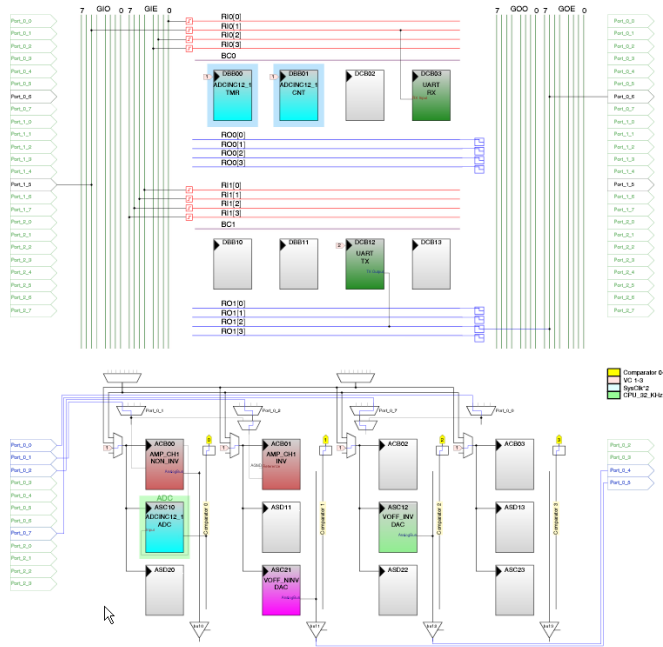


Figure 2. PSoc Block Placement

Pin names, sources and types are shown in Figure 3.

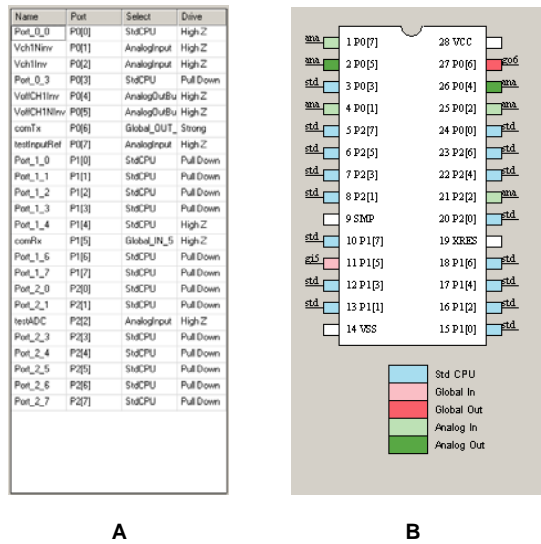


Figure 3. Pin Names and Sources (A) and Types (B)

Circuit Realization

A schematic for the project is shown in the Appendix.

Figure 4 shows the breadboard built to support the code-debugging phase. The PSoC ICE-4000 debugger is used instead of the final chip in order to speed-up code debugging. The ICE is also used to power the system, which eliminates the power section from the breadboard.

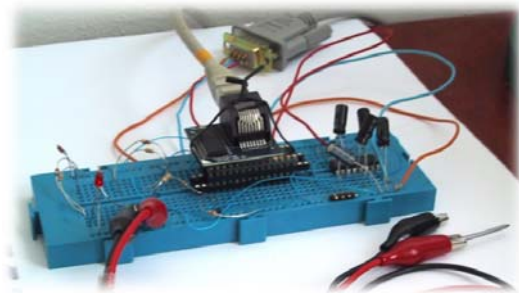


Figure 4. Breadboard

Figure 5 shows the breadboard with the basic functional blocks highlighted.

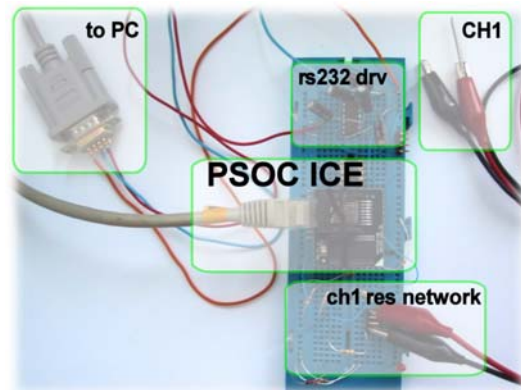


Figure 5. Breadboard Functional Blocks

Figure 6 shows a board working with a 6V DC supply voltage.

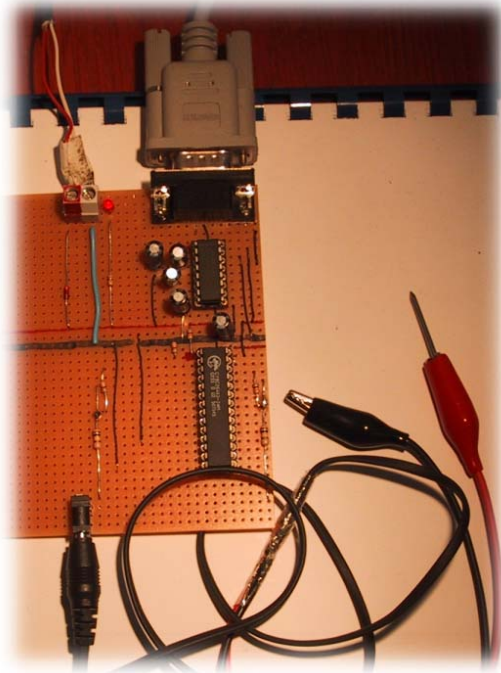


Figure 6. 6V-Powered Example Board

Firmware Overview

The firmware architecture is very simple. After all the devices are initialized, the system enters in the main loop where commands are received from the UART and processed as soon as they are received.

Global variables are used to keep the configuration persistent for the oscilloscope required by the host system.

When a continuous sample mode is selected, as soon as a new sample is available, it is sent to the host through the UART. If the trigger is ON, data is transmitted en mass to the host only after 128 data bits have been sampled. Data are saved in the *sample's* array.

All commands are a single byte and are decoded by a simple switch. Offset and gain commands directly change the gain of the instrumental amplifier and the output of the DAC.

The trigger mode, selected by the host system, is saved in a bit-field variable, which is checked with each loop of the main loop where the sampling is performed.

Client Application

A simple .Net Windows application, developed using C#, is included with this Application Note. To access RS232, the application uses a VB.NET class written by Corrado Cavalli and freely available for download on the Internet.

The ScopeEngine class implements the IScopeEngine interface used by the main class, Scope, to retrieve data to be viewed and to send commands to the PSoC board. Commands for the Scope class are shown in Figure 7.

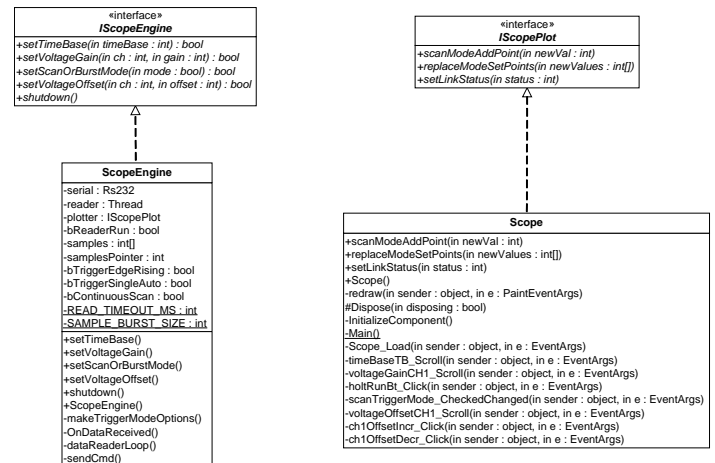


Figure 7. Scope and ScopeEngine commands.

Figure 8 shows a screenshot of the front-end application.

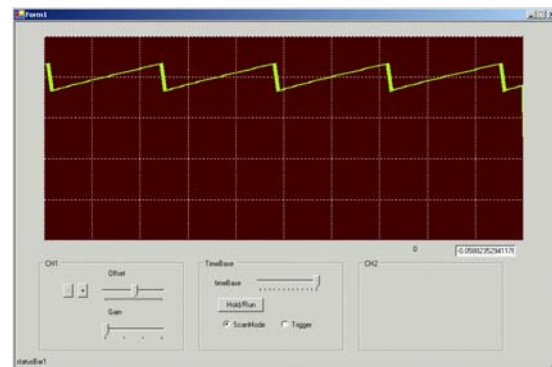


Figure 8. Front-End Application

Future Improvements

- The second channel will be implemented.
- All the missing commands will be added to the firmware, especially the basic trigger functionalities.
- A voltmeter form will be created.
- A "calibration" XML file will be added to the client side to calibrate the actual speed and gain of the channel.

Appendix

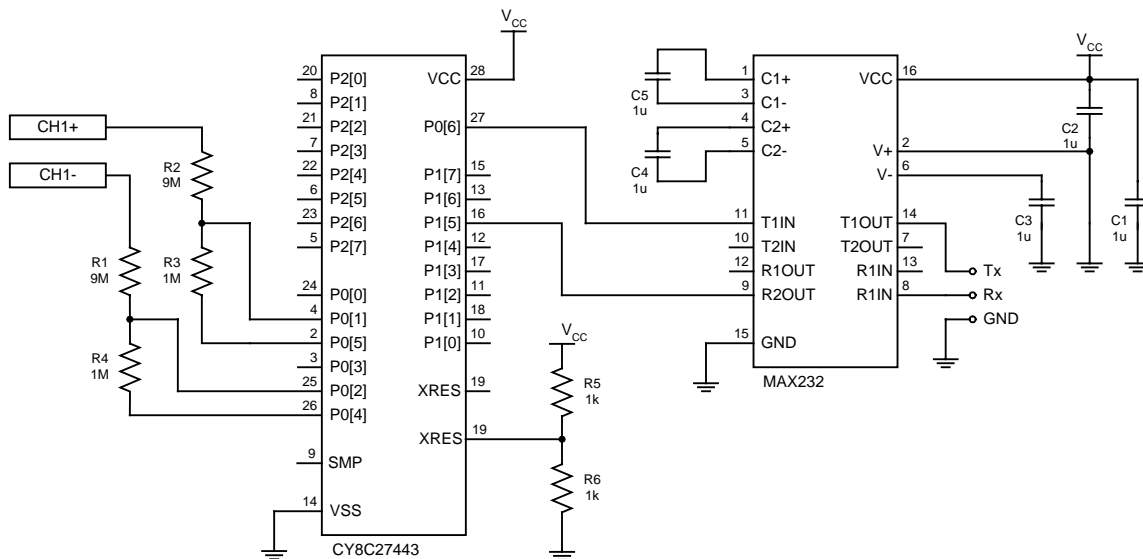


Figure 9. Device Schematic

About the Author

Name: Andrea Giacosi

Title: Electronic Engineer

Background: Real-Time Software

Contact: giacosi@libero.it

Cypress MicroSystems, Inc.
2700 162nd Street SW, Building D
Lynnwood, WA 98037
Phone: 800.669.0557
Fax: 425.787.4641

<http://www.cypress.com/> / <http://www.cypress.com/support/mysupport.cfm>

Copyright © 2004 Cypress MicroSystems, Inc. All rights reserved.

PSoC™, Programmable System-on-Chip™, and PSoC Designer™ are trademarks of Cypress MicroSystems, Inc.

All other trademarks or registered trademarks referenced herein are the property of their respective owners.

The information contained herein is subject to change without notice. Made in the U.S.A.