Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

PSoC® 1 - DelSigADC and UART

CE58024

Project Name: Example_DelSigADC_UART
Programming Language: C
Associated Part Families: CY8C24x23, CY8C27x43, CY8C29x66 CY8C24x94, CY8C21x34
Software Version: PSoC Designer™ 5.4
Related Hardware: CY3210 PSoCEval1 Board
Author: Richa Dham

Project Objective

This project demonstrates the DelSigADC with second order modulator and the UART.

Overview

In this project the input analog signal applied to the DelSig ADC with second order modulator is converted to 8-bit digital data. From this 8-bit value, the input voltage is calculated. The PSoC multiplies the input voltage value with the scaling factor given by the user through the HyperTerminal. The result of multiplication is sent to the HyperTerminal over serial port along with the input voltage value and the scaling factor given by the user.

User Module List and Placement

The following table lists the user modules used in this project and the hardware resources occupied by each user module.

User Module	Placement
UART	DCB02 (TX Configuration) DCB03 (RX Configuration)
PGA	ACB00
DelSig	ASC10(Modulator 1) ASD20(Modulator 2) DBB00(PWM)

User Module Parameter Settings

The following table shows the user module parameter settings for the UART user module.

UART		
Parameter	**Value**	**Comments**
Clock	VC3	Input is 153.8 KHz clock. (that is, 8 times the required baud rate(19200 Hz). VC3=Sys.Clk / 156.
RX Input	Row_0_Input_1	Routed from pin P2[5] through GlobalInEven_5.
TX Output	Row_0_Output_3	Routed to pin P2[7] through GlobalOutEven_7.
TX InterruptMode	TXRegEmpty	Not used.
Clock Sync	Sync to Sys.Clock	Clock is synchronized with the SysClk, as VC3 is a derivative of

UART		
Parameter	**Value**	**Comments**
		SysClk
RxCmdBuffer	Enable	Enable the command buffer, so that the command received is stored in a RAM buffer.
RxBufferSize	32	Bytes - Length of buffer is 32 characters (including parameters).
Command Terminator	13	Carriage return (13) is the command terminator.
Param_Delimiter	32	Not used in this project.
IgnoreCharsBelow	32	Ignore control characters which have ASCII value below 32.
Enable Backspace	Disable	Not used.
Rx Output	None	Not used
Rx Clock Out	None	Not used
Tx Clock Out	None	Not used
InvertRX Input	Normal	Do not invert Rx Input.

Notes

- The clock to the TX8 user module should be eight times the desired baud rate
- The parameters Command Terminator, ParamDelimeter, and IgoneCharsBelow are used by the high level UART API to process the commands received. When the UART receives the character set as Command Terminator, the API reports through a flag that a valid command has been received. The API ignores any characters below the value set in IgonreCharsBelow value and does not store them in the command buffer.

PGA		
Parameter	**Value**	**Comments**
Gain	1.000	PGA acts as a buffer between DelSigADC input and P0 [1].
Input	AnalogColumn_InputMUX_0	Output of AnalogColumn_InputMUX_0 is connected to the input of PGA.
Reference	VSS	Reference to the PGA is set to VSS.
AnalogBus	Disable	Output of PGA is not connected to Analog output buffer.

DelSig		
Parameter	**Value**	**Comments**
DataFormat	Unsigned	All the 8 bits are used to represent magnitude.
DataClock	VC1	Column clock for the DelSig module is VC1(2 MHz)
Clock Phase	Normal	Phi 1→Charge acquire phase. Phi 2→Charge Transfer phase.
PosInput	ACB00	Output of unity gain buffer (PGA) is connected to the positive input of DelSigADC.
NegInput	ASD20	Not used in this project.
NegInputGain	Disconnected	Signal connected to the negative input of DelSigADC is not considered in the analog to digital conversion; that is, differential input is not used.
PWM Output	None	PWM output is not used in this project.
Pulse Width	10	This parameter determines the duty cycle of PWM.

Notes

The column clock to the analog column where the ADC is placed should be same as the DataClock parameter. When the input to the ADC comes from a CT block or a direct port pin, the Clock Phase may be set to either Normal or Swapped. But if the input to the ADC comes from another SC Block, then the clock phase should be set to Swapped.

Global Resources

Important Global Resources		
Parameter	**Value**	**Comments**
CPU_Clock	SysClk/2	Sets the CPU frequency to 12 MHz
VC1=SysClk / N	12	SysClk is divided by 12 to get 2 MHz column clock for DelSig module.
VC3 Source	SysClk/1	Set System Clock as the source for VC3
VC3 Divider	156	Divide 24 MHz system clock by 156, so that the effective baud rate is 19.2 kbps.
Analog Power	SC On/Ref High	Set the maximum operating power of analog blocks to Ref High.
Ref Mux	(Vdd/2)+/-(Vdd/2)	Ref High = 5 V Ref Low = 0 V AGND = 2.5 V

Note

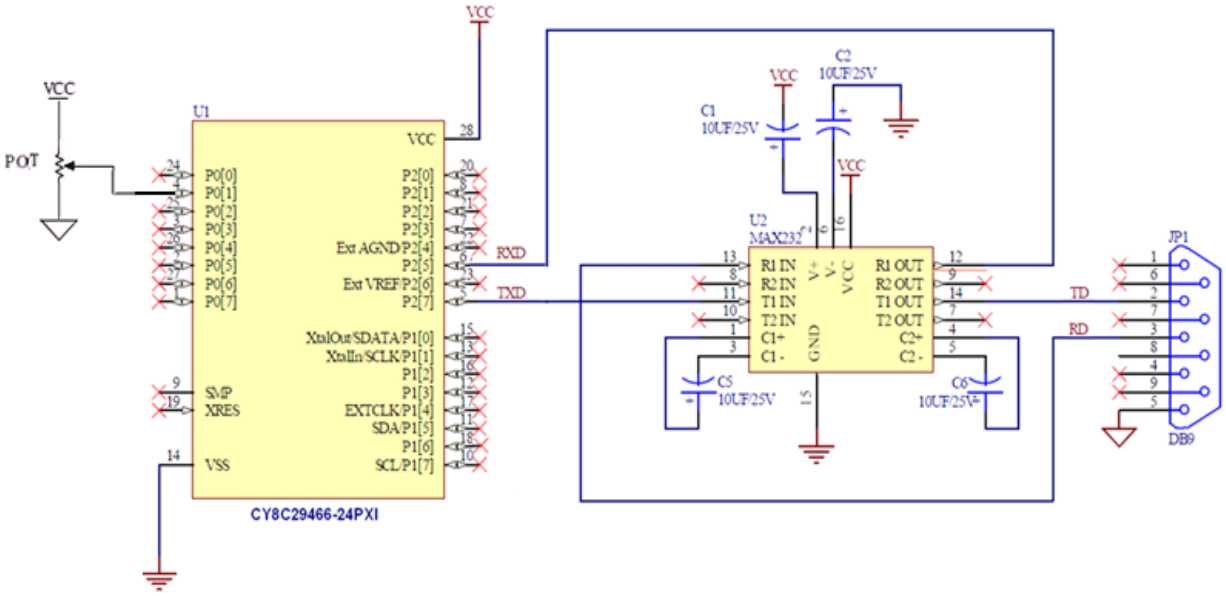
All other global resources are left at their default, as they are not specific to this project.

Pin Configuration

Pin Out			
Pin	**Select**	**Drive**	**Direction**
Port 0_1	Analog Input	High Z Analog	Input
Port 2_5	GlobalInEven_5	High Z	Input
Port 2_7	GlobalOutEven_7	Strong	Output

Hardware Connections

Following is the schematic of the project.



MAX232, an RS232 transceiver, is used to translate the +/-10V RS232 signals to transistor-transistor logic (TTL) level signals of the PSoC.

JP1 is a 9-pin Female serial port connector which is used to connect the project with a PC.

The project can be tested using CY3210 – PSoC Eval1 board. This board has an RS232 transceiver and a Serial port connector. To test the project using the CY3210 board, make the following connections:

- Connect P25 of J7 to RX of J13
- Connect P27 of J7 to TX of J13
- Connect P0[1] of J6 to VR of J5

Operation

Upon program execution all hardware settings from the device configuration are loaded into the device and *main.c* is executed.

The following operations are performed in *main.c*.

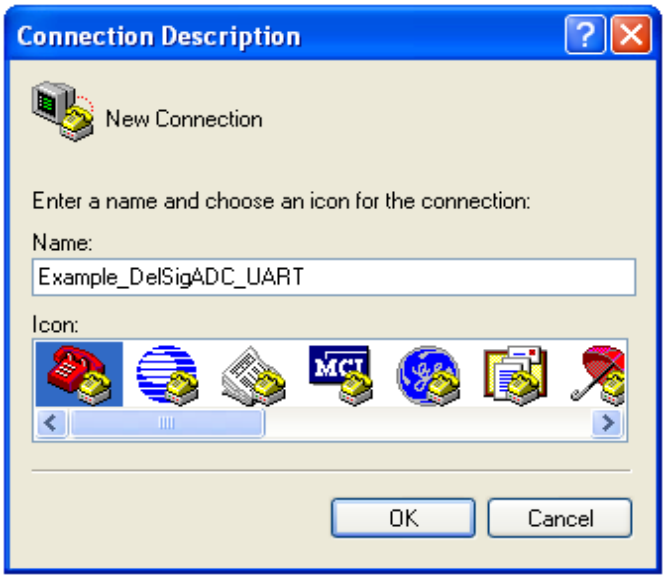
1. Enable receiver buffer of the UART module using UART_CmdReset() API.
2. Enable global interrupts.
3. Enable UART receiver interrupts.
4. Start UART module with no parity.
5. Start PGA in high power mode.
6. Start DelSigADC in high power mode.
7. Enable Timer and integrator blocks of DelSigADC using DelSig_StartAD() API.
8. Clear the screen in the HyperTerminal window by sending new page character over the serial port.
9. Send welcome strings over the serial port.

10. Wait for the reception of command terminator. If command terminator is received go to step 11. If not, repeat step 10.
11. Get the received data (scale factor) from command buffer using UART_szGetParam() API. Convert it into float using atof() library function and store it in the float variable scale factor.
12. Inside an infinite loop do the following:
    a) Wait till ADC data is available.
    b) Read the ADC data and clear the status flag.
    c) Multiply DelSig output data by step size of analog to digital conversion. The result of this multiplication gives the present value of the input signal in float form. (The step size is calculated by the user and the calculated value is defined as macro using *#define* directive at the top of *main.c*. Step size = (RefHigh-RefLow)/(2^Resolution)). In this project, step size = (5-0)/(2^8)= 0.01953125).
    d) Multiply the present input voltage value (calculated from step 14) by the scaling factor (obtained from step 11). The result of multiplication gives the scaled value of the present input signal.
    e) Send the constant string " Input signal (in volts) = " over serial port to HyperTerminal.
    f) Convert the actual input voltage value to ASCII and send the same over serial port to HyperTerminal.
    g) Send the constant string “scale factor =” over serial port to HyperTerminal.
    h) Convert the scaling factor to ASCII and send the same over serial port to HyperTerminal.
    i) Send the constant string “Scaled input signal (in volts) =” over serial port to HyperTerminal.
    j) Convert the scaled input signal to ASCII and send the same over serial port to HyperTerminal.

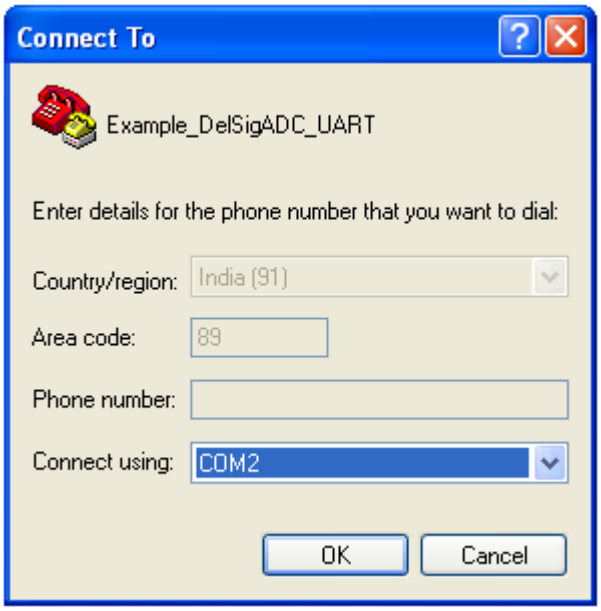
Testing the Project

To test the project, HyperTerminal (or any other terminal program) may be used. Use the following steps to configure HyperTerminal in Windows.

1. Connect the CY3210 board to the PC serial port using a Serial port cable.
2. Start HyperTerminal using **Start** > **All Programs** > **Accessories** > **Communication** > **HyperTerminal**.
3. Enter a Name for the connection, such as “Example_DelSigContinuous_UART”, and select **OK**.

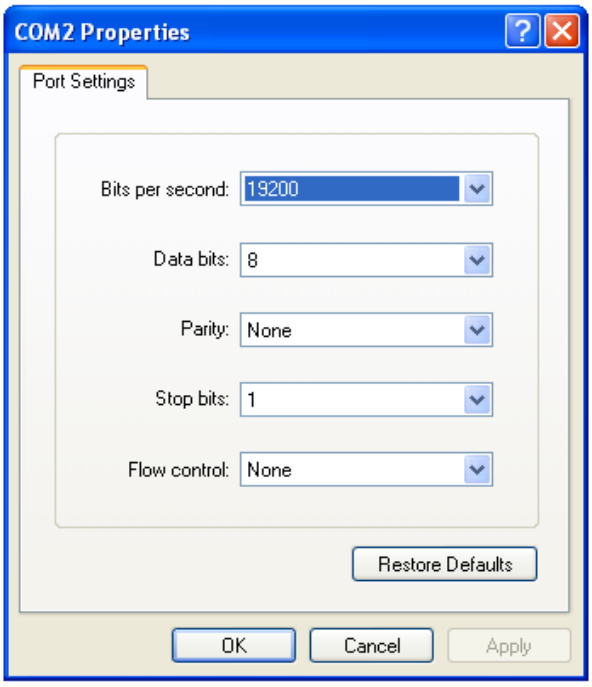


4. In the **Connect To** option, select the desired serial port (for example, COM2) from the **Connect using** list and click **OK**.



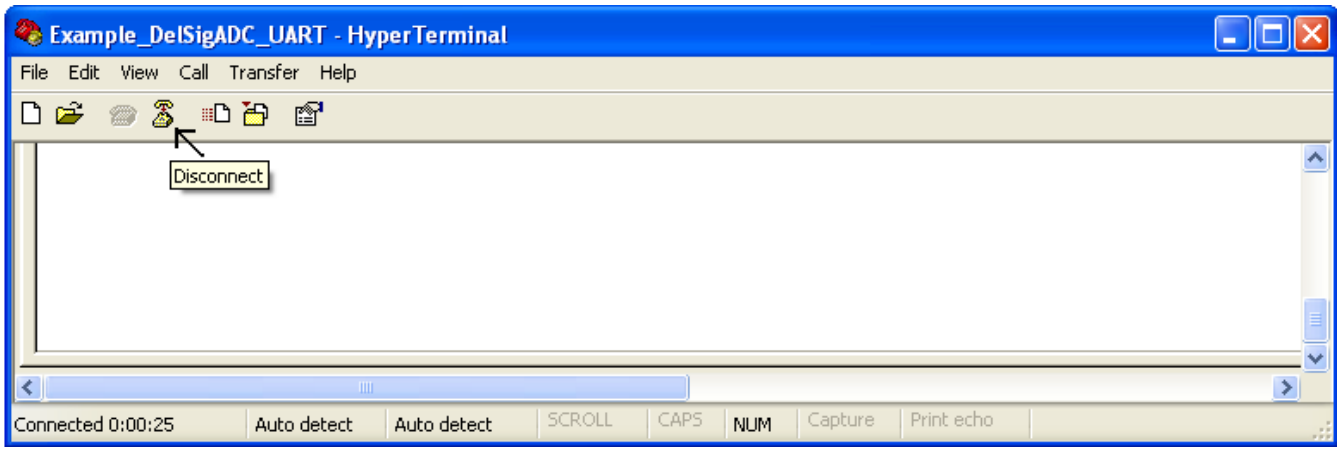
5. In the COM2 properties, configure the following parameters:

   - Bits per second = 19200
   - Data bits = 8
   - Parity = None
   - Stop Bits = 1
   - Flow Control = None
   - Click on **Apply** and then click **OK**.

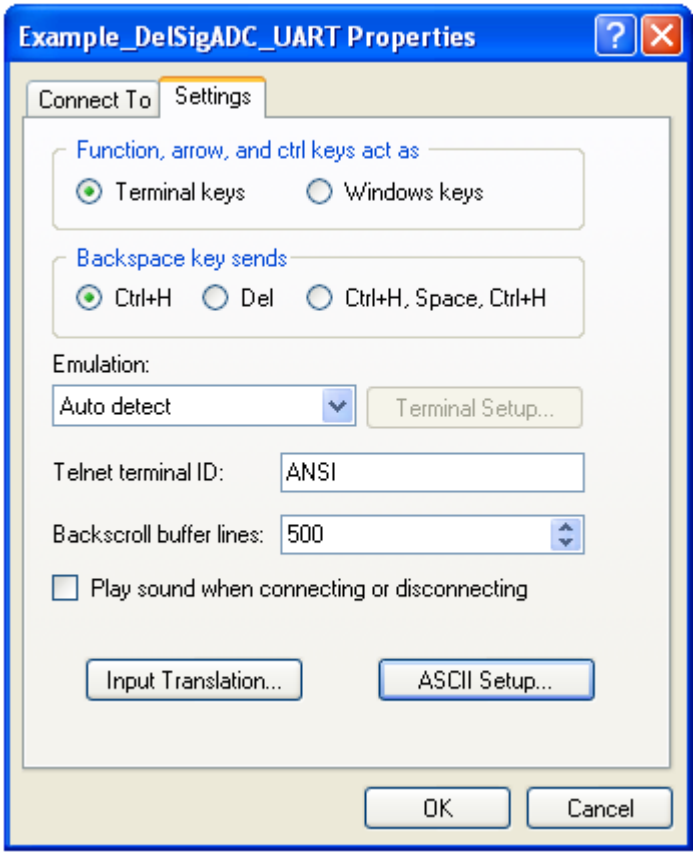


6. At this point, HyperTerminal connects to COM2. We must do some more configurations before the project is tested.

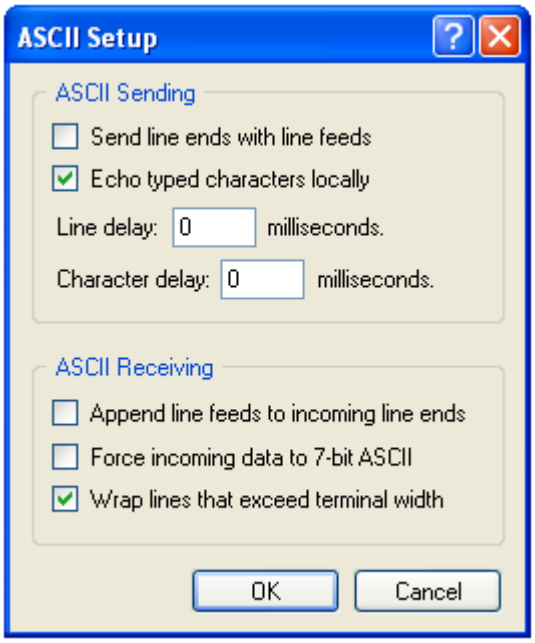
7. Click on the **Disconnect** icon



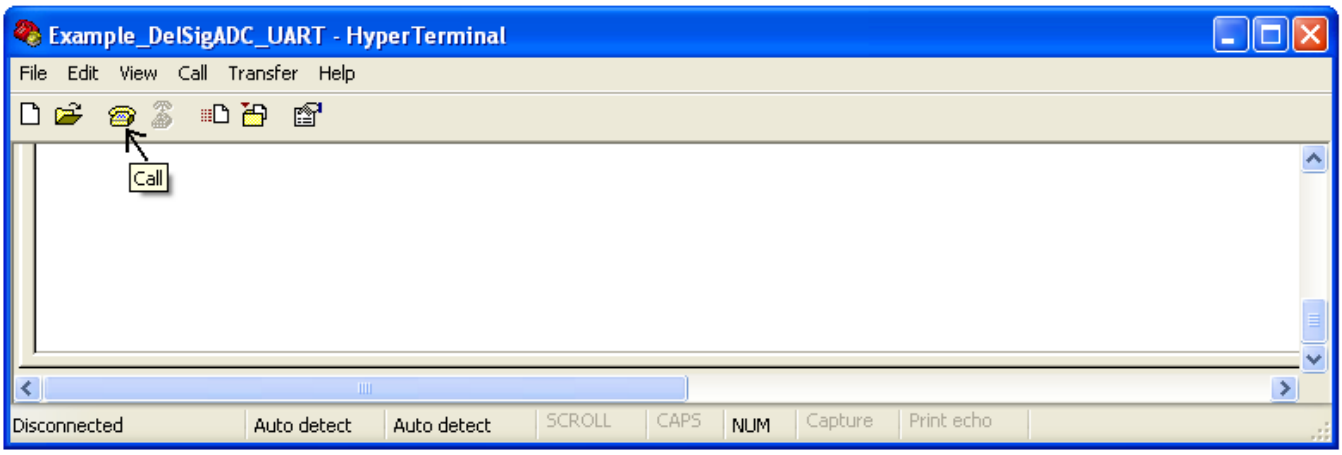
8. Click on **File** > **Properties** > **Settings** > **ASCII Setup**



9. Enable **Echo typed characters locally** and Click **OK**.

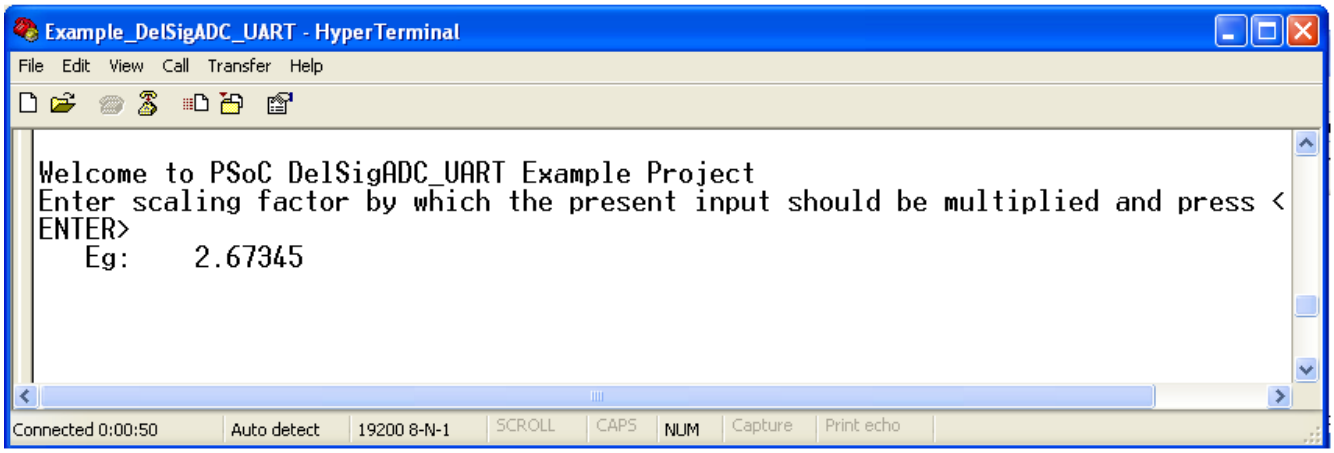


10. Click on **Call**.

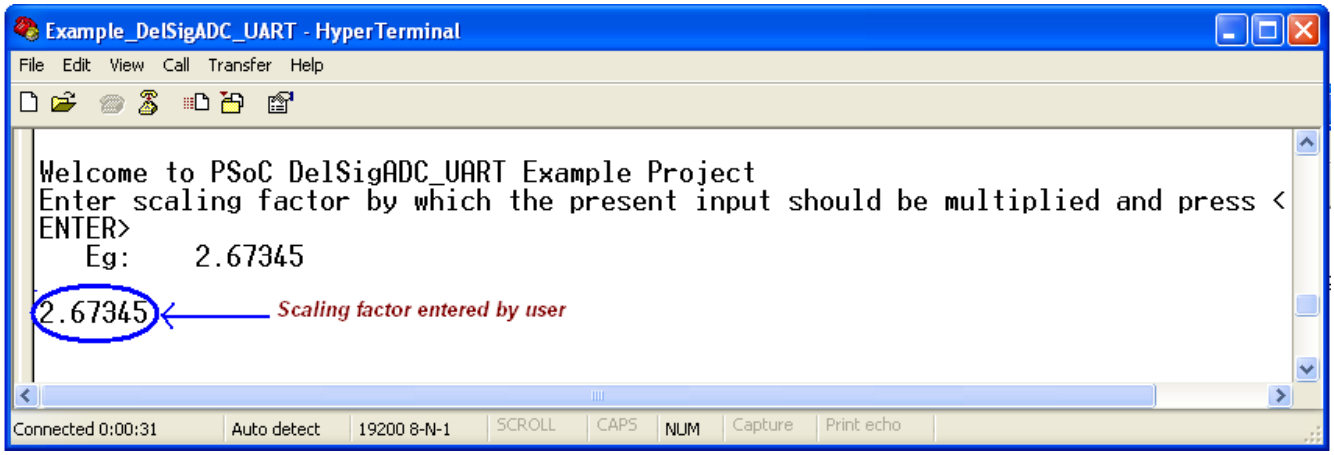


Now, the HyperTerminal is ready to be used with the Example_DelSigADC_UART project.

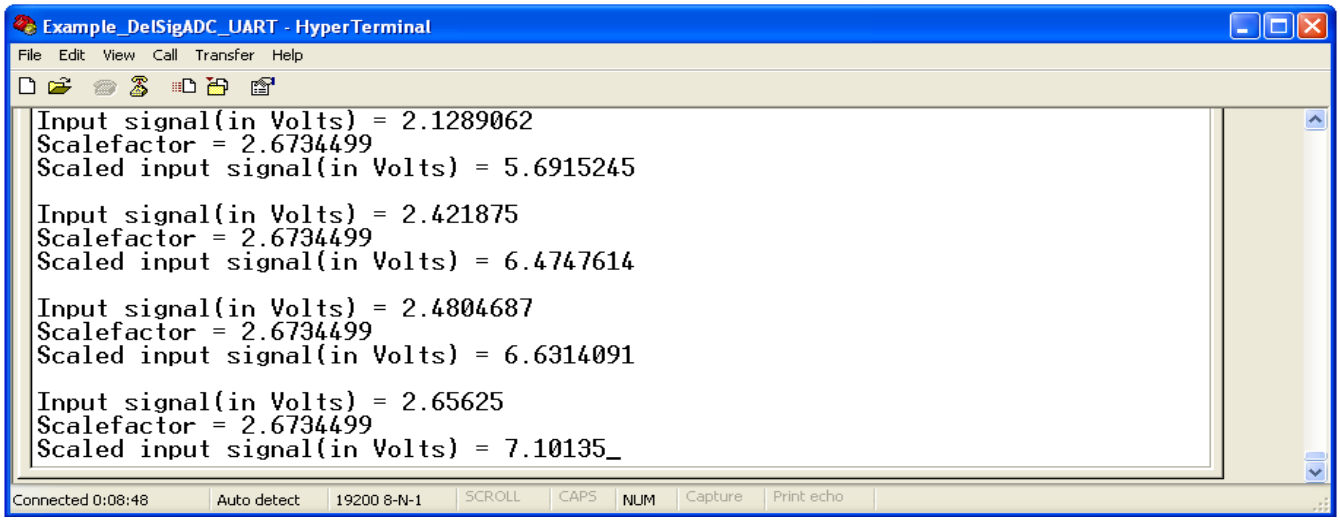
11. After setting the HyperTerminal as explained in the previous step, reset the PSoC. The welcome strings are displayed on the HyperTerminal as shown in the following figure.



12. Enter the scaling factor as shown in the following figure.



13. Now click **Enter**. The HyperTerminal displays the input signal value, scaling the factor value and scaled input signal value as shown in the following figure. Vary the input voltage and observe the values changing in HyperTerminal window.



Document History

Document Title: PSoC® 1 - DelSigADC and UART - CE58024

Document Number: 001-58024

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	2819642	DIMA	12/08/2009	New Example Project
*A	3100356	DIMA	12/06/2010	Updated to PD5.1
*B	3259452	DIMA	05/24/2011	Document title updated. Added CExxxxx in header. Changed Operation of the code example and updated screen shots. Updated software version.
*C	3722585	RICA	08/27/2012	Removed PD Example Project file attached with the document and including only the document.
*D	4225398	MSUR	12/19/2013	Updated Software version

PSoC is a registered trademark of Cypress Semiconductor Corp. PSoC Designer is a trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408-943-2600
Fax: 408-943-4730
http://www.cypress.com/

© Cypress Semiconductor Corporation, 2009-2013. 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.

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.