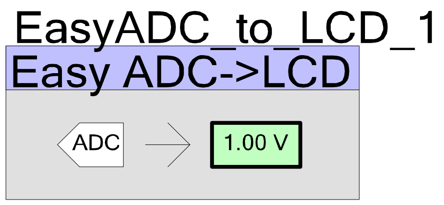
EasyADC to LCD

1.00

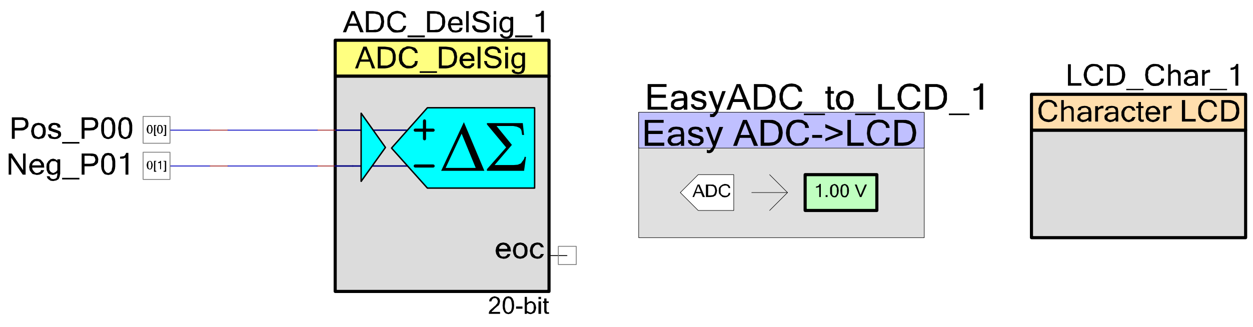
Features



- Easy setup to do quick measurements with the ADC
- Displays ADC reading in volts
- The number of decimal places displayed is adjustable
- Optional software filter stabilizes readings
- Optional custom transfer function for converting readings

General Description

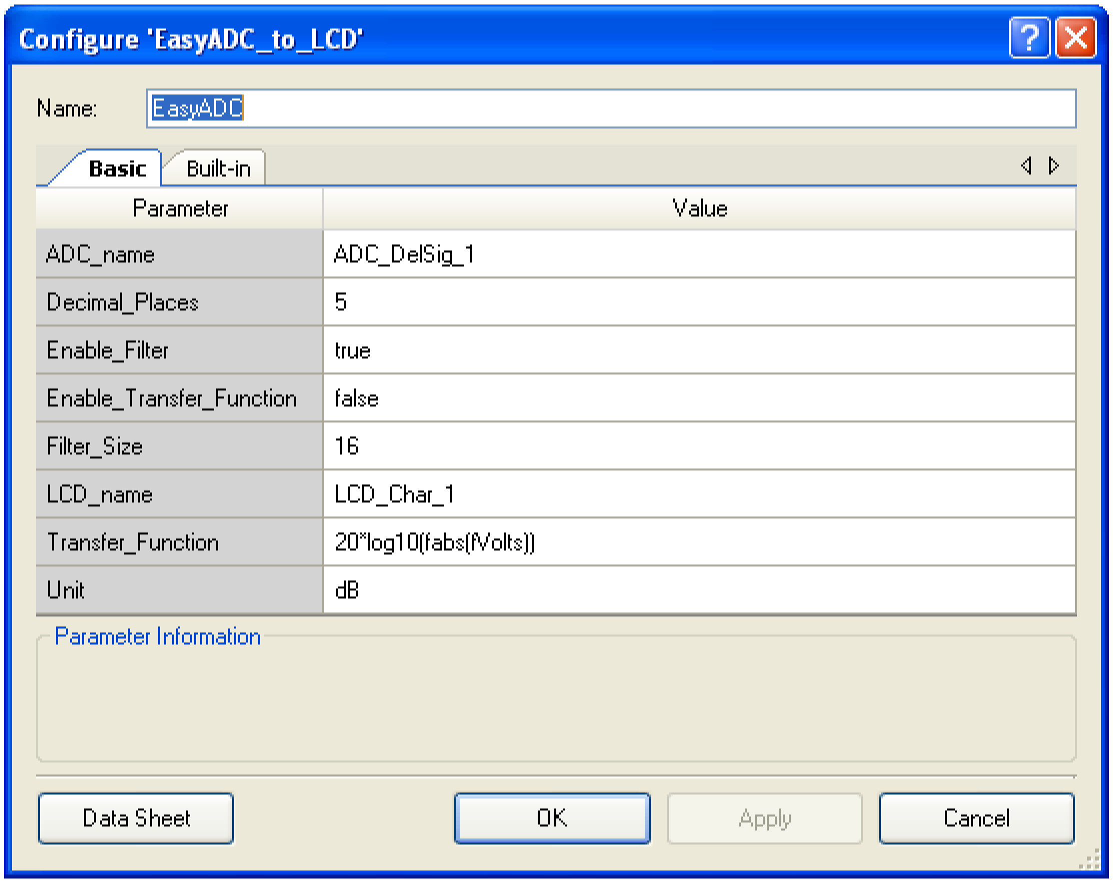
The EasyADC to LCD component takes samples from the ADC, converts them into volts using the ADC APIs and displays them on a character LCD module. Using this component makes taking readings with the ADC and displaying them on the LCD very simple. The component also has an optional software filter which performs a running average on up to 256 samples to stabilize readings. If the ADC reading is the output of sensor, you can also enable a custom transfer function which will convert the ADC voltage output into a user defined value. This value will be displayed on the second line of the LCD with a user defined unit label.



Parameters and Setup

Drag an EasyADC to LCD component onto your design and double-click it to open the Configure dialog.

Figure 1 Configure Opamp Dialog



Parameter	Value
ADC_name	ADC_DelSig_1
Decimal_Places	5
Enable_Filter	true
Enable_Transfer_Function	false
Filter_Size	16
LCD_name	LCD_Char_1
Transfer_Function	20*log10(fabs(fVolts))
Unit	dB

The EasyADC to LCD has the following parameters:

ADC_name

This parameter is the instance name of the ADC you wish to take samples from. This can be the DelSig ADC or the SAR ADCs. This needs to be the instance name of an ADC component placed in your schematic. Make sure you enter the instance name exactly as it appears in the "name" field of the ADC you wish to use.

Decimal_Place

This parameter sets the number of decimal places that will be displayed on the LCD.

Note This parameter also affects the number of decimal places displayed for the custom transfer function.

Enable_Filter

This parameter enables the software filter. When enabled each sample taken from the ADC will be placed in a running average. The results will be averaged together and the final averaged value will be displayed on the LCD.

Note Each call to the EasyADC_to_LCD_Display() API only takes one sample. To fill up the filter, you must call the _Display() function at least as many times as the size of your filter to ensure that you get a valid reading.

Enable_Transfer_Function

This parameter enables the user defined transfer function. When enabled, the resulting ADC voltage is passed through a user defined transfer function. This enables the ADC reading to be converted into a more meaningful value if the ADC is sampling a sensor output. The result of this conversion will be displayed on the second line of the LCD below the ADC voltage. The result can be appended with a user defined unit to ease readability.

Note The transfer function operates on the final voltage which will be filtered if the filter is enabled.

Filter_Size

This parameter sets the number of samples in the running average filter.

Note Each sample stored in the filter is a signed 32 bit value. Large filter sizes can use significant amounts of RAM.

Note Each call to the EasyADC_to_LCD_Display() API only takes one sample. To fill up the filter, you must call the _Display() function at least as many times as the size of your filter to ensure that you get a valid reading.

LCD_name

This parameter is the instance name of the LCD you wish to display the information on. This needs to be the instance name of a character LCD component placed in your schematic. Make sure you enter the instance name exactly as it appears in the "name" field of the ADC you wish to use.

Transfer_Function

This parameter is a string that will be used as C code to convert the ADC voltage into a user defined value. When the Enable_Transfer_Function parameter is set to true, this parameter defines what the transfer function will be. Math.h is included when the transfer function is enabled, so any C expression that uses math.h functionality is valid. The (optionally) filtered voltage is stored in a variable named 'fVolts'. Use this variable when defining your transfer function. There is no need to add a semicolon at the end of the string. Here are some simple examples:

Transfer_Function entry	Explanation
fVolts+1	Adds an offset to the ADC voltage
20*log10(fabs(fVolts))	Converts the ADC voltage into dB
127.5*fVolts+sin(fVolts/6.28)	Example of using constants and other math

Note The transfer function operates on the final voltage which will be filtered if the filter is enabled.

Unit

This parameter is the unit that will be displayed after the custom transfer function value. It is a string that will be placed at the end of the custom result.

Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. The following table lists and describes the interface to each function. The subsequent sections cover each function in more detail.

By default, PSoC Creator assigns the instance name "EasyADC_to_LCD_1" to the first instance of a component in a given design. You can rename it to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol.

Function	Description
void EasyADC_to_LCD_1_Start(void)	Turns on the LCD and the ADC
void EasyADC_to_LCD_1_Display(void)	Takes a sample from the ADC, (optionally) filters it, converts the reading to volts and displays the voltage on the first line of the ADC. If the transfer function is enabled it also converts the voltage and displays the result on the second line of the ADC.

void EasyADC_to_LCD_1_Start (void)

Description:	Turns on the LCD and the ADC
Parameters:	None
Return Value:	None
Side Effects:	None

void EasyADC_to_LCD_1_Display(void)

Description: Starts an ADC conversion, waits for the conversion to complete and grabs the result using the 32 bit read so all resolutions of the ADC are supported then stops the ADC conversion. The result is then added to the running average if the filter is enabled and the running average is computed. This result is then passed to the CountsTo_Volts function of the ADC to get the ADC voltage as a floating point number. This number is then displayed on the first line of the LCD. If the transfer function is enabled, the voltage is converted using the transfer function and the result is displayed on the second line of the LCD.

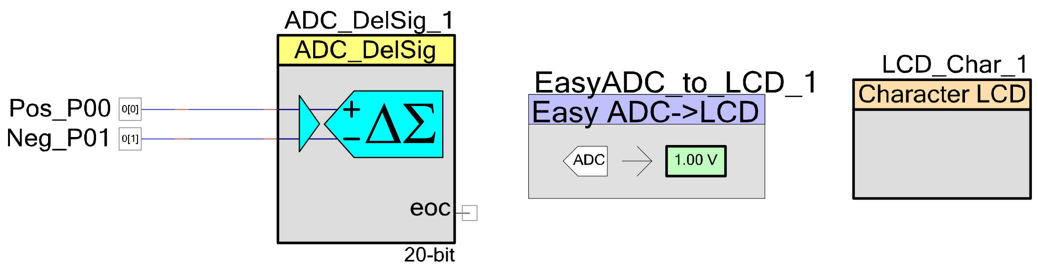
Parameters: None

Return Value: None

Side Effects: None

Sample Firmware Source Code

The following example is all that is required to display data on the LCD. No need to call ADC start or LCD start. Everything is handled by the component.



```
void main()
{
    /* Place your initialization/startup code here (e.g. MyInst_Start()) */

      EasyADC_to_LCD_1_Start();

    /* CYGlobalIntEnable; */ /* Uncomment this line to enable global interrupts. */
    for(;;)
    {
            EasyADC_to_LCD_1_Display();
    }
}
```