
ADC and Noise Suppression

Features

- Noise Suppression Using ADC Computation modes: Basic, Accumulate, Average, Burst Average, LPF
- Auto-Conversion Trigger
- Signal & Noise Generator Board to Generate Signal With and Without Noise
- Plotting Graph of ADC Samples in *Data Visualizer*
- Graph with Noisy Signal and Filtered Signal for Both AC and DC Signals
- Noise Level Comparison with 5V and 3.3V ADC Reference Voltage

Introduction

Author: Rupali Honrao, Microchip Technology Inc.

This application note will explain how to use the noise reduction features available in the Analog-to-Digital Converter with Computation (ADC²) in Microchip's 8-bit PIC[®] microcontrollers (MCU). This ADC has built-in computational features that provide post-processing functions such as oversampling, averaging, and low-pass filtering. Built-in registers help to eliminate the software overhead while processing of oversampling, averaging, and low-pass filtering.

Using the Signal & Noise Generator board, this application note will demonstrate noise suppression by the ADC² in the following modes: Basic, Accumulate, Average, Burst Average, and Low Pass Filter (LPF). Pros and cons of each mode are also listed. The signal with noise and filtered signal will be shown in the *Data Visualizer* graph. A test setup for generating noisy signals is suggested and directions on how to illustrate the results in the Data Visualizer, are provided.

Example code for replicating the results described in this application note is available from:

- MPLAB Xpress code example:
 - Noise Suppression and ADCC Computation modes using PIC18F45Q10: <https://mplabxpress.microchip.com/mplabcloud/example/details/798>

Table of Contents

Features.....	1
Introduction.....	1
1. Noise Suppression Theory.....	3
2. ADC ² Computation Modes.....	5
3. Add Noise to Signal	7
4. Demonstrate Noise Suppression.....	8
4.1. Hardware Prerequisites.....	8
4.2. Software Prerequisites.....	8
4.3. Hardware Setup.....	9
4.4. Source Code Overview.....	11
4.5. Results With Graph.....	13
5. Noise Level Comparison With ADC Reference 5V/3.3V.....	34
5.1. Graph With ADC Reference 3.3V.....	34
5.2. Graph With ADC Reference 5V.....	35
6. Appendix A: Plotting Graph in <i>Data Visualizer</i>	38
7. Revision History.....	41
The Microchip Website.....	42
Product Change Notification Service.....	42
Customer Support.....	42
Microchip Devices Code Protection Feature.....	42
Legal Notice.....	43
Trademarks.....	43
Quality Management System.....	44
Worldwide Sales and Service.....	45

1. Noise Suppression Theory

Many MCU applications involve measuring analog signals. In the ideal case of a completely noiseless signal, achieving a high-quality digital representation of the signal is simple: Single ADC conversions triggered with a fixed time interval is sufficient. In reality, most analog signals are affected by noise and the modern Microchip ADCs provide functionality that can be used to increase the signal-to-noise ratio.

Noise can be defined as undesirable electrical signals, which interfere with an original (or desired) signal. Noise reduction, the recovery of the original signal from the noise-corrupted one, is a very common goal in the design of signal processing systems.

Every sample from an ADC can be a combination of signal (S) and noise (N). Noise suppression is a process of reducing noise with minimal impact on the desired signal. One way this can be achieved is by computing the average of many samples of a noisy signal, which will reduce the fluctuations and leave the desired signal visible.

Figure 1-1 illustrates a noisy signal. Single conversions spaced in time results in a noisy digital representation of the signal. A potential solution could be to filter the acquired samples in software, but this would require additional CPU resources. A better option would be to use the ADC Computation modes supported by the ADC.

Figure 1-2 illustrates how a single ADC conversion trigger results in a burst of multiple successive ADC conversions.

Each conversion will be accumulated in hardware and the average value of the accumulated samples can be calculated by dividing the accumulated result by burst size in the Accumulation mode whereas in Average mode, Burst Average mode, and Low Pass Filter (LPF) mode the filtered result is available in the built-in Filter register. Because the sampled noise has a zero mean, the averaged result will be close to the actual signal value.

Here, the ADC Computation modes can be used for averaging by configuring the ADC to accumulate m samples automatically.

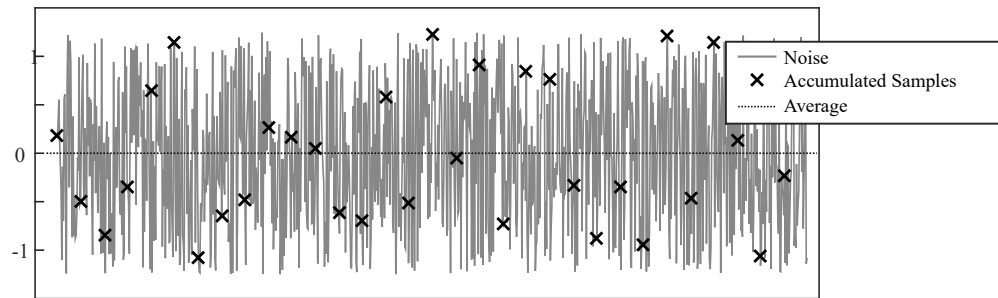
In Burst Average mode, the ADC sampling rate is affected by the number of samples accumulated. The total sampling time for m samples is the multiplication of sampling time for one sample and m , the number of samples taken.

Figure 1-1 illustrates a noisy signal: DC signal mixed with random noise.

Figure 1-1. DC Signal Mixed With Noise



If the signal is zoomed in, consider the ADC samples that are taken as indicated in Figure 1-2. Here, it illustrates how 32 ADC samples are accumulated in a short time window.

Figure 1-2. Averaging Zero Mean Noise

Each individual sample differs from zero by a random value, but with equal probability of being either positive or negative. The accumulated noise samples will approach zero and the noise is successfully suppressed. If this noise is imposed on a non-zero signal, the accumulated value will approach a scaled version of this signal's average.

As oversampling is done with multiple samples, the average result of all the sampled values will be approximately equal to the original DC signal. That means it results in zero mean noise.

By increasing the burst size (accumulating more samples), it helps to flatten out more peak signals and results in more suppression in noise.

2. ADC² Computation Modes

The ADC² module can be operated in one of five Computation modes listed below:

- **Basic:** In this mode, the ADC conversion can be manually triggered by the core or auto-triggered by other peripherals and external sources. The accumulator logic is not active in Basic mode, which means that the Accumulator (ADACC) and Count (ADCNT) register values are not used throughout the operation, and the Result register (ADRES) will hold the sampled ADC result. Threshold error comparison, double sampling, Continuous mode, and all CVD (Capacitive Voltage Divider) features are still available, but no feature involving the digital filter or average features are used.
- **Accumulate:** In this mode, with each trigger, the ADC conversion result is added to the accumulator and ADCNT increments indicate the number of samples accumulated. The ADCNT value saturates at 255 samples and does not rollover to zero. If the ADACC register overflows, the Accumulator Overflow bit (ADAOV) of the Status register (ADSTAT) will be set. ADCNT and ADACC registers need to be cleared by software once ADCNT reaches the desired number of samples to get the correct average of accumulated ADC samples. The accumulated value can be right-shifted up to six times (by the CPU) by configuring the value of the ADC Accumulate Calculation Right Shift Select bits (ADCRS) of the ADCON2 register. This means that the accumulated value is effectively divided by 2^{ADCRS} after each conversion. The result of the shifted accumulated value is stored in the ADFLTR register. The code below is an example of how to calculate the average value of ADC samples using the Accumulate mode if the number of samples to be accumulated is 32:

```
if(ADC_conversion_result_ready)
{
    if(ADCNT>=32)
    {
        read_average_result = ADCC_GetFilterValue(); //ADCRS needs to be configured to 5 in
        ADC_init
        ADCC_ClearAccumulator(); // will clear ADACC and ADCNT
    }
}
```

As the filtered value is available after each ADC conversion, and its value is accumulated value divided by 32 (right shifted by five positions). Because of that, the average value at the 32, 64, 96... sample will be the correct expected average value. At this point the ADACC register value is the sum of 32 samples, so after dividing it by 32, an average value can be obtained (division which means right shifting is done by CPU automatically so you need to read the ADFLTR value only). All the other ADFLTR values can be ignored. After the 32nd sample the accumulator needs to be cleared which will clear the value of the ADACC register and ADCNT.

- **Average:** This mode is like the Accumulate mode, where the ADACC accumulates the data sample and the ADCNT increments with each sample, except that the number of samples being accumulated is up to the value configured in the ADC Repeat Setting Register (ADRPT). When the ADCNT is equal to ADRPT, the value stored in the ADC Filter register (ADFLTR) is the average value of the input signal ($\text{ADFLTR} = \text{ADACC} / 2^{\text{ADCRS}}$). When the ADCNT exceeds the value of ADRPT, the ADCNT and ADACC registers reset automatically by the CPU to accumulate the data samples again. If the Threshold Interrupt Mode Select bits (ADTMD[2:0] = 7 in ADCON3 register) are configured to *Interrupt regardless of threshold test result* then the ADTIF flag will be set after accumulating samples configured in ADRPT.
- **Burst Average:** This mode works like the Average mode, except that instead of software re-enabling the conversion, the CPU continuously retriggers the ADC conversion until the ADCNT value is equal to the ADRPT value. In other words, with single ADC conversion trigger, all the data samples are accumulated up to the configured ADRPT and when the ADCNT matches the set ADPRT value, the

average value of the input signal is attained by reading the ADFLTR value as $ADFLTR = ADACC / 2^{ADCCRS}$. At the trigger, the accumulator ADACC and ADCNT are automatically cleared by the CPU.

- **Low-Pass Filter (LPF):** This mode is similar to the Average mode but instead of a simple average, it performs a low-pass filter operation on all the samples, passes signals with frequencies below its cutoff, and attenuates frequencies above its cutoff. The ADCRS bits determine the cut-off frequency of the low-pass filter.

For more details on LPF cutoff frequency calculation and other computation modes, refer to the technical briefs:

- TB3146: <http://www.microchip.com/wwwAppNotes/AppNotes.aspx?appnote=en584645>
- TB3194: <http://www.microchip.com/wwwAppNotes/AppNotes.aspx?appnote=en606039>

3. Add Noise to Signal

To verify the ADC² features, artificial noise can be generated with various methods and then this noise can be added to the input signal. Here the Signal & Noise Generator will be used to generate noise and verify noise suppression. This board is designed to generate an analog signal with or without noise. For more information on the Signal & Noise Generator, refer to the user guide: <https://www.microchip.com/DevelopmentTools/ProductDetails/PartNO/DM080100>

4. Demonstrate Noise Suppression

The noise suppression is demonstrated by using example source code and plotting a graph of ADC samples in the Data Visualizer.

A signal with noise is given as an input signal to the ADC. It will be sampled and the ADC result value is sent over EUSART to the serial terminal of the *Data Visualizer*, and a graph of the ADC samples is plotted in the Data Visualizer. Different graphs with different computation modes such as Basic mode, Accumulate mode, Average mode, Burst Average mode, and Low Pass Filter mode are plotted. From these graphs, it can be observed how the ADC result count range gets reduced when noise is suppressed in the configured computation mode. A detailed explanation is provided in the further sections.

4.1 Hardware Prerequisites

Hardware used in this application:

- Curiosity High Pin Count (HPC) Development Board (DM164136)
 - <http://www.microchip.com/developmenttools/ProductDetails/PartNo/DM164136>
- PIC18F45Q10 40-lead microcontroller (PDIP package to fit Curiosity HPC)
 - <https://www.microchip.com/wwwproducts/en/PIC18F45Q10>
- Power debugger (ATPOWERDEBUGGER)
 - <http://www.microchip.com/developmenttools/ProductDetails/PartNo/ATPOWERDEBUGGER>
- Signal & Noise Generator (DM080100)
 - <https://www.microchipdirect.com/product/search/all/DM080100>
- Oscilloscope
- Two micro-USB cables
- Six male-to-male wires

Note:

- The example application is using the PIC18F45Q10 device but the application can also be created using any device from the Q10 family
 - <https://www.microchip.com/promo/pic18f-q10-product-family>
- The HW setup describes using the Signal & Noise Generator board but any signal with or without noise can also be used

4.2 Software Prerequisites

Software used in this application:

- MPLAB™ X IDE v.5.10
 - <http://www.microchip.com/mplab/mplab-x-ide>
- MPLAB XC8 C Compiler v.2.00
 - <http://www.microchip.com/mplab/compilers>
- MCC PIC10/PIC12/PIC16/PIC18 library v1.75
 - <https://www.microchip.com/mplab/mplab-code-configurator>
- Data Visualizer v.2.18.663

- <https://www.microchip.com/mplab/avr-support/data-visualizer>
- Noise Suppression and ADCC Computation Modes using PIC18F45Q10.X source code
 - <https://mplabxpress.microchip.com/mplabcloud/example/details/798>
- Data Streamer configuration file which is included in source code:
noise_suppression_data_streamer.txt

4.3 Hardware Setup

- The Curiosity High Pin Count (HPC) Development Board (DM164136) is used as the test platform. Switch S1 and S2 are used to select the ADC² mode, and LEDs D2, D3, D4, D5 are used to display which mode is active.
- A signal from the Signal & Noise Generator board is connected to an analog input (RA1) of the HPC.
- The Power Debugger includes a CDC virtual COM port interface, which is used to transmit ADC²'s conversion results to a PC over the EUSART to the Data Visualizer. The EUSART TX (pin RC6) of the HPC is connected to CDC 'TX ←' pin of Power Debugger.
- HPC uses USB power supply, configured to 3.3V VCC by a jumper.

HW Setup: Curiosity HPC with Power Debugger and Signal & Noise Generator Board

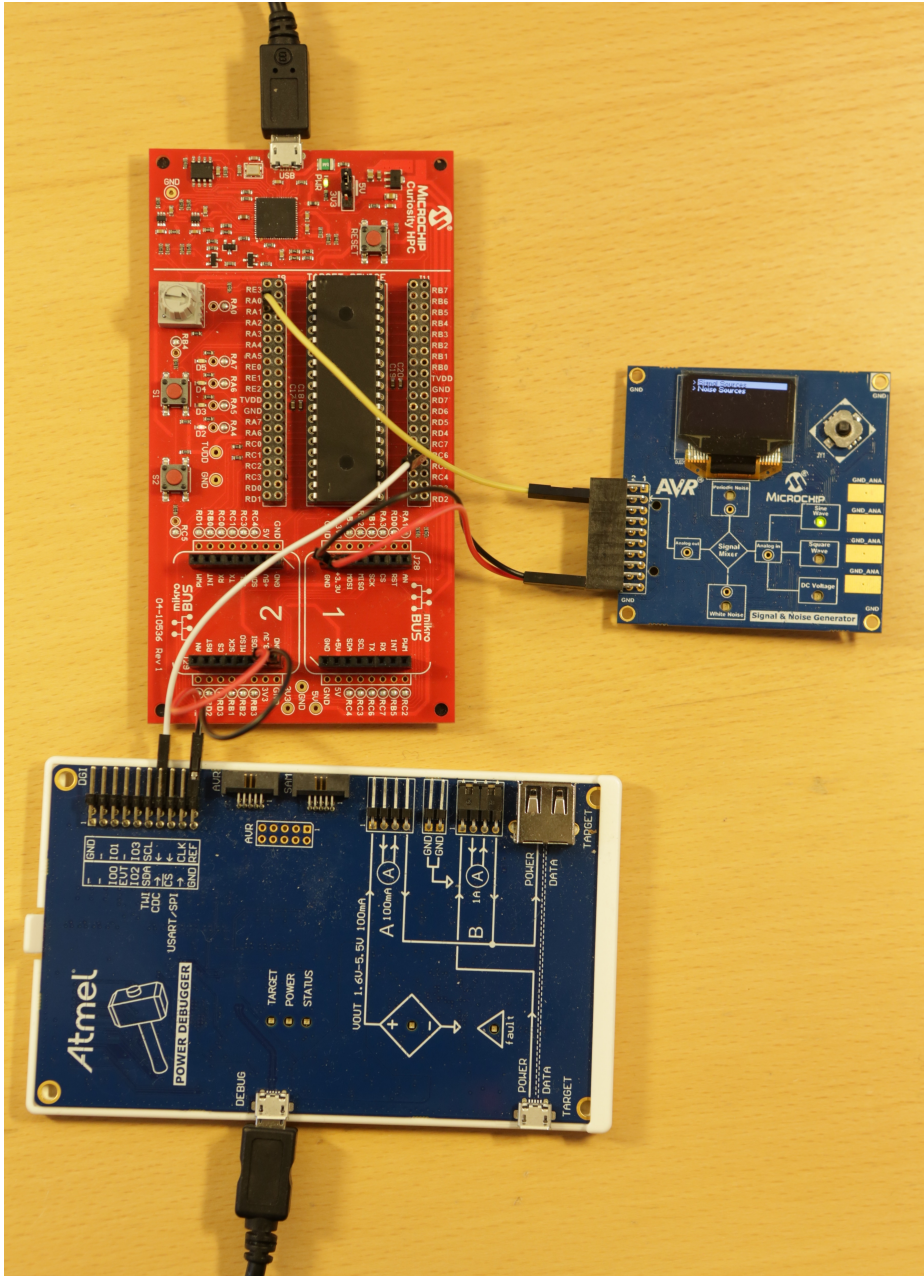


Table 4-1. HW Connection HPC- Signal & Noise Generator

HPC	Signal & Noise Generator	
Pin	Pin Number	Name
RA1	3	ADC(+)
GND	19	GND
3V3	20	VCC

Table 4-2. HW Connection HPC- Power Debugger

HPC	Power Debugger (DGI port)	
Pin	Pin Number	Name
RC6	14	TX ←
GND	19	GND
3V3	20	VCC

4.4 Source Code Overview

Source Code Overview Using the PIC MCU Q10

- CPU clock: 32 MHz
- Peripherals used:
 - ADC²
 - ADC input channel is AIN1: pin RA1
 - ADC reference voltage: V_{DD}
 - ADC clock: 1 MHz ($F_{osc}/32$)
 - ADC Conversion time for single sample: 11.5 μ s
 - EUSART
 - TX pin RC6
 - Baud rate: 500000, ADC result is sent to the serial terminal
 - TMR0:
 - Timer 0 is configured to have 625 μ s (1.6 kHz sampling frequency) trigger period, which the ADC in auto conversion trigger uses.
 - GPIO
 - pin RB4: Switch S1, to increase ADCC mode
 - Pin RC5: Switch S2, to decrease ADCC mode
 - pin RA4: LED D2
 - pin RA5: LED D3
 - pin RA6: LED D4
 - pin RA7: LED D5
- ADC² functions used in example application apart from MCC driver generated

Table 4-3. ADC² Functions Apart From MCC Driver Generated

Function Name	File	Purpose
<code>void ADCC_ComputationModeSetup(uint8_t)</code>	<code>main.c</code>	Calls the mode specific ADC ² initialization function
<code>void ADCC_InitializeBasicMode(void)</code>	<code>adcc.c</code>	Initializes ADC ² to Basic mode
<code>void ADCC_InitializeAccumulateMode(void)</code>	<code>adcc.c</code>	Initializes ADC ² to Accumulate mode
<code>void ADCC_InitializeAverageMode(void)</code>	<code>adcc.c</code>	Initializes ADC ² to Average mode

.....continued		
Function Name	File	Purpose
void ADCC_InitializeBurstAverageMode(void)	adcc.c	Initializes ADC ² to Burst Average mode
void ADCC_InitializeLPFMode(void)	adcc.c	Initializes ADC ² to LPF mode
void ADCC_SetChannel(adcc_channel_t)	adcc.c	Configures the ADC ² channel
void ADCC_ProcessResult(void)	main.c	Reads and process the ADC ² result
void DataStreamerTxPkt(pkt_t *)	data_streamer.c	Initializes the data streamer packet with (SOF) Start of Frame, ADC ² mode, (EOF)End of Frame
void DataStreamerInit(pkt_t *)	data_streamer.c	Transmits data streamer packet through EUSART
void CheckButtonPress(void)	main.c	Reads the status of push button S1 and S2

Table 4-4. Computation Mode: Associated Number and LED

Computation Mode	Computation Mode Number in Data Visualizer	LED Illuminated
Basic	1	All LEDs: D2,D3,D4,D5
Accumulate	2	D2
Average	3	D3
Burst Average	4	D4
LPF	5	D5

The ADC² is configured in the *adcc.c* in the project folder. Below are the mode specific functions defined in *adcc.c* which are used in this application.

```
void ADCC_InitializeBasicMode(void);
void ADCC_InitializeAccumulateMode(void);
void ADCC_InitializeAverageMode(void);
void ADCC_InitializeBurstAverageMode(void);
void ADCC_InitializeLPFMode(void);
```

Note: In *adcc.c* the *ADCC_Initialize()* function is a driver generated function and called in *SYSTEM_Initialize()* as part of standard driver initialization.

Upon power-up, the default mode is the Basic mode, which calls the initialization routine *ADCC_InitializeBasicMode()* configured for the Basic mode. The computation mode number will be configured to 1 to represent it in the *Data Visualizer* and all LEDs are turned on. When S1 push-button is pressed, the computation mode will be increased and it will initialize the ADC with the selected mode. Different LEDs are turned on depending on selected computation mode. The Computation mode and Associated Computation mode number and LED illumination are shown in [Table 4-4](#).

The interrupt, regardless of threshold test results, is configured in ADCON3 registers (ADTMD[2:0] = 7). When ADC result is ready, the ADTIF flag will be set and the ADC result and selected computation mode number will be sent to the Data Visualizer over EUSART.

The ADTIF flag is set on every ADC conversion for the Basic and Accumulate modes. For the Average, Burst Average, and LPF modes it will be set after accumulating configured n number of samples. In this example source code, it will be set after accumulating 32 samples (as ADCRS is configured to 5).

4.5 Results With Graph

The results will be shown by plotting different graphs in the *Data Visualizer* by configuring advanced ADC features. Here, different graphs with different ADC² computational modes are plotted. With the availability of both ADC results and ADC filtered result registers, it is possible to observe both noisy and filtered signals in the same graph.

A separate graph indicating the selected ADC computation mode will also be plotted.

From these graphs, it can be observed how the ADC result count range gets reduced when noise is suppressed with configured ADC features.

Note:

- Complete the HW setup shown in the [4.3 Hardware Setup](#) topic
- Program the device PIC Q10 with example source code
- Any signal with or without noise can be connected to pin RA1 on HPC board if the Signal & Noise Generator board is not available
- Refer [6. Appendix A: Plotting Graph in Data Visualizer](#) for more details on how to plot graphs in *Data Visualizer*

4.5.1 Graphs: Basic Mode

4.5.1.1 DC Signal With No Noise

Task: The graph is plotted in *Data Visualizer* using a DC signal without noise.

- ADC² computation mode: Basic
- Input signal: DC ~1V with no artificial noise added

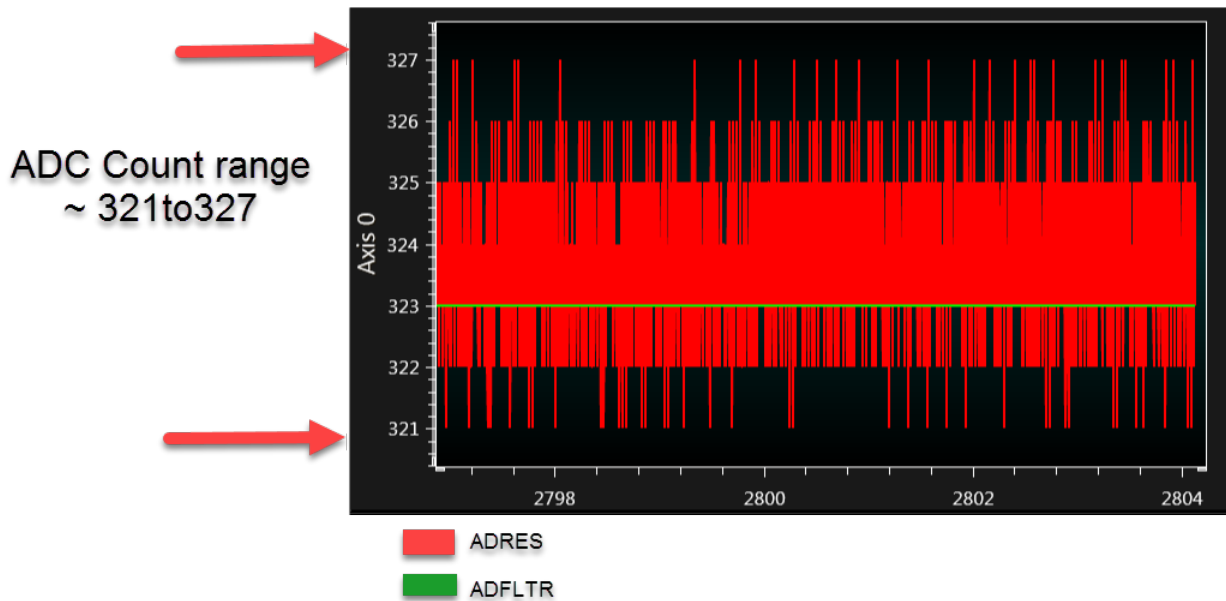
Test Setup:

- Configure VDD on the HPC board to 3.3V
- Configure the Signal & Noise Generator board to generate a DC signal of ~1V
- Verify the input signal using an oscilloscope
- Verify that LEDs D2, D3, D4, and D5 are illuminated

The input signal is connected to the ADC input pin RA1 and a graph in *Data Visualizer* is plotted as illustrated in [Figure 4-1](#). A red colored signal is the ADC input signal (ADRES values), a green colored signal is a filtered signal using a simple averaging technique.

Note: Though any artificial noise hasn't been added to the input signal, some system noise is present and showed in the graph as the ADC count value of the input signal is varying from 321 to 327. The result may vary from one setup to another.

Figure 4-1. DC Signal With No Artificial Noise Added

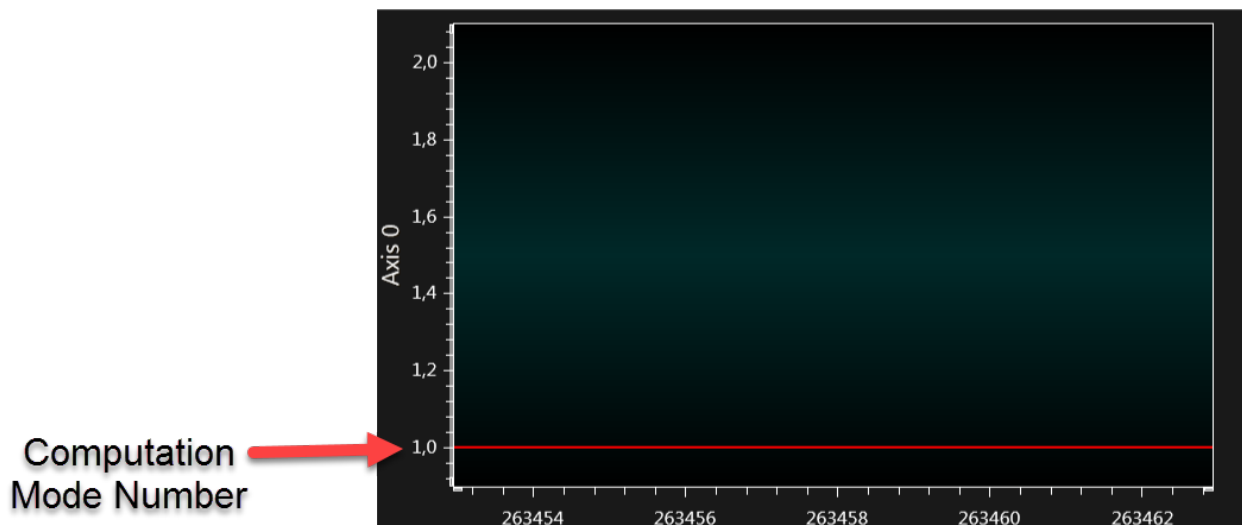


From the graph, the ADC filtered signal shows ADC count 323. In Basic mode, the filtered result is attained using software averaging in code with 32 samples.

Note: Test setup measures DC 1.06V here. The ADC reference voltage is $V_{DD}=3.3V$ and the ADC resolution is 10-bit. Ideally measured ADC count should be $(1023 \times 1.06)/3.3 = 328$.

ADC² computation mode number is 1 for the Basic mode as shown in the *Data Visualizer* graph illustrated in [Figure 4-2](#).

Figure 4-2. Computation Mode Number



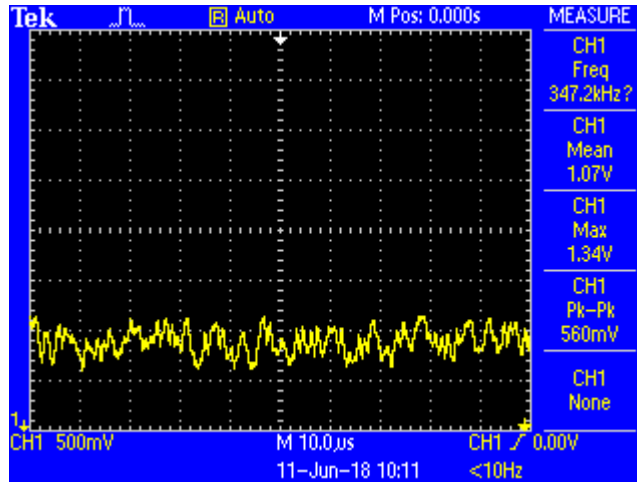
4.5.1.2 DC Signal With Random Noise

Task: The graph is plotted in *Data Visualizer* using a DC signal with random noise and noise suppression is observed.

- ADC² computation mode: Basic
- Input signal: DC ~1V + Random noise 0.5V peak-to-peak

Test setup:

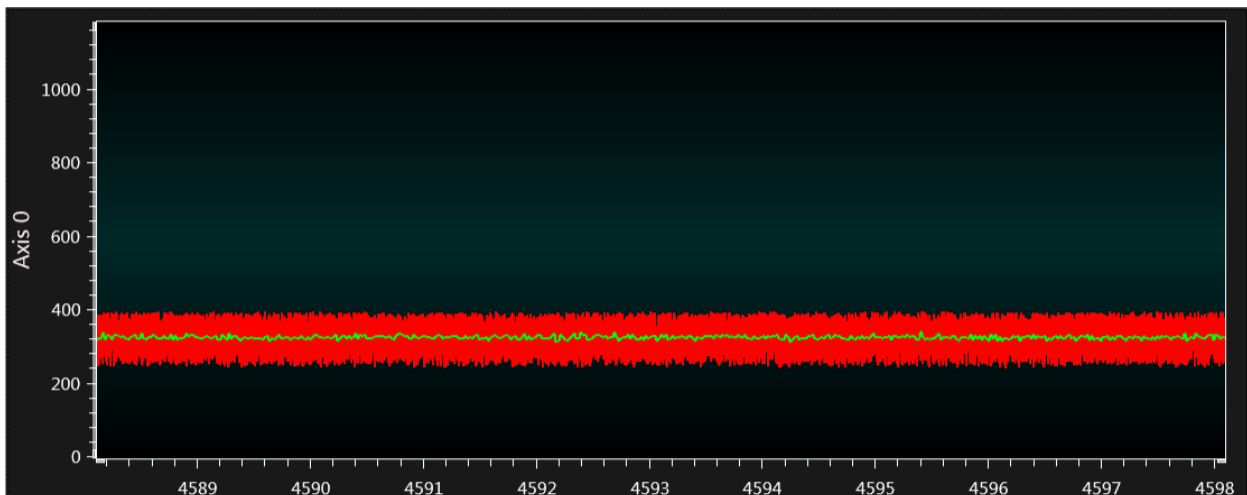
- Verify that LEDs D2, D3, D4, and D5 are illuminated
- Configure the Signal & Noise Generator board to generate a DC signal of ~1V and random noise with 0.5V peak-to-peak
- Verify the input signal using an oscilloscope. The expected result is as shown below.

Figure 4-3. Signal With Random Noise Oscilloscope Capture

Note: Figure 4-3 shows a DC offset random noise signal. The DC offset is approximately 1V, and the noise peak-to-peak magnitude is close to 500 mV.

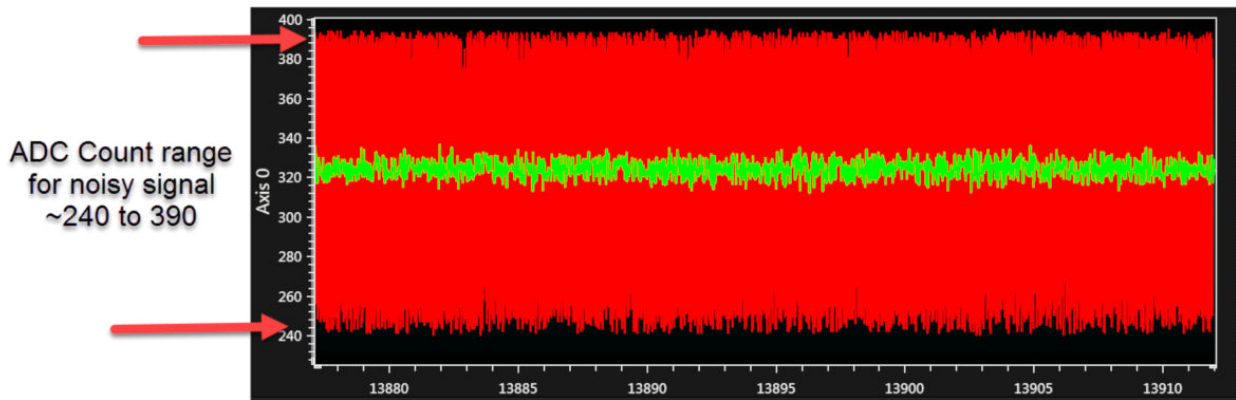
The *Data Visualizer* graph is as shown in Figure 4-4. A red colored signal is noisy input signal (ADRES value), a green colored signal is the filtered signal attained using software averaging in code with 32 samples.

From the graph, it can be seen that the noise has been suppressed using a simple averaging technique. In the example source code, the averaging has been done with 32 samples.

Figure 4-4. DC Signal With Random Noise

If the signal is zoomed, the image is as shown in Figure 4-5.

Figure 4-5. Zoomed DC Signal With Random Noise

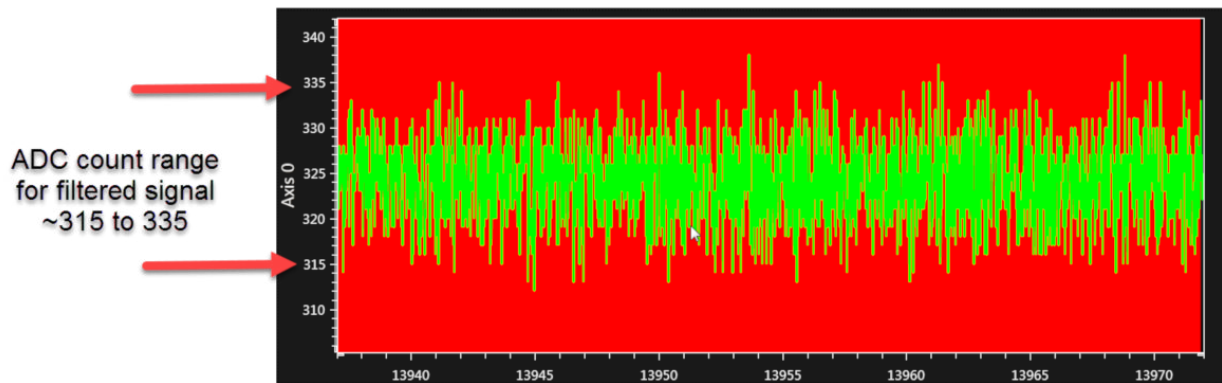


From the graph, the ADC count for a noisy signal can be seen varying from ~ 240 to 390 because of random noise. That means the ADC count is varying ± 75 counts ($390 \text{ to } 240 \rightarrow 315 \pm 75$ counts).

Note: The graphed signal is a noisy DC signal. The actual signal levels can vary between boards, but the result is expected to be similar to the illustrated result. In this illustration, the result is in the range of 240 to 390 ADC counts, and the noise level can be estimated to 150 ADC counts peak-to-peak.

If the signal is zoomed further, the image is as shown in [Figure 4-6](#).

Figure 4-6. Further Zoomed DC Signal With Random Noise



ADC count for a filtered signal can be seen varying from ~ 315 to 335. That means the ADC count is varying ± 10 counts ($335 \text{ to } 315 \rightarrow 325 \pm 10$ counts).

Pros: Using this mode, the number of samples to be accumulated for averaging is not limited whereas it is limited to 64 for other ADC² modes.

Note: Ideally, with 10-bit ADC and 16-bit variable for sample accumulation, the overflow will occur after accumulating 64 samples ($65535/1023 = 64$).

Cons: The software overhead of accumulating and averaging needs to be handled in the code.

4.5.2 Graphs: Accumulate Mode

4.5.2.1 DC Signal With Random Noise

Task: The graph is plotted in *Data Visualizer* using a DC signal with random noise and noise suppression is observed.

- ADC² computation mode: Accumulate mode
- Input signal: DC ~1V + Random noise 0.5V peak-to-peak

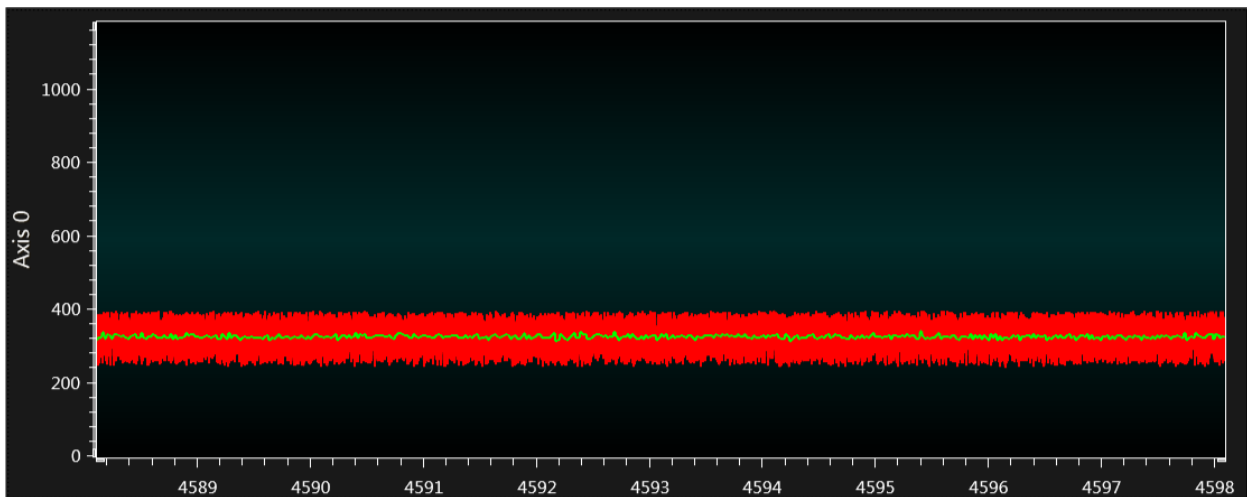
Test setup:

- Press push button S1 on the HPC board
- Verify the Computation mode number in *Data Visualizer* graph is 2
- Verify LED D2 is illuminated
- Configure Signal & Noise Generator board to generate a DC signal of ~1V and random noise with 0.5V peak-to-peak
- Verify the input signal using an oscilloscope

The *Data Visualizer* graph is as shown in [Figure 4-7](#). The red colored signal is a noisy input signal (ADRES value), the green colored signal is a filtered signal (ADFLTR value).

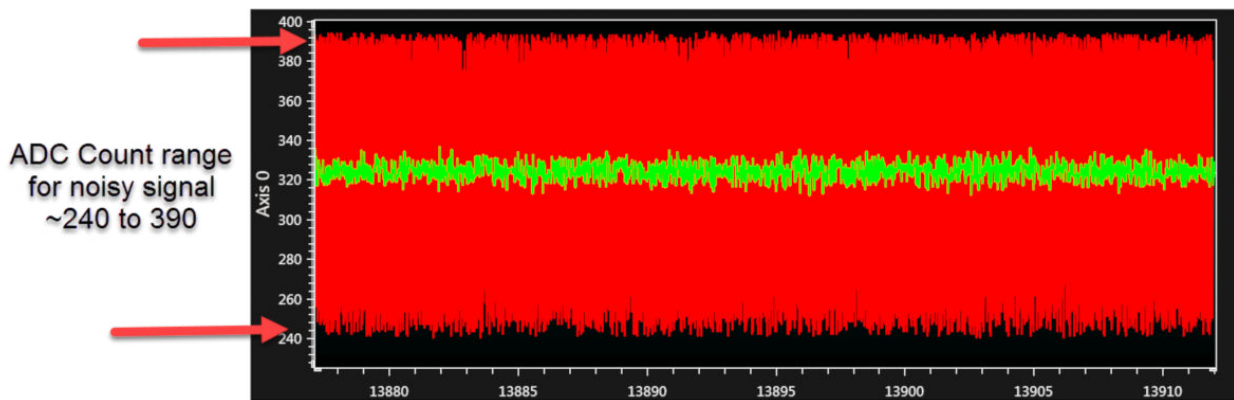
From the graph, it can be seen that the noise has been suppressed using the ADC² Accumulate mode. In the example source code the ADCRS bits are configured to 5. This means averaging is attained with 32 samples. The graph looks similar to the graph for the Basic mode. The advantage here is sample accumulation, and averaging handling is not needed in the code as the ADACC and ADFLTR registers are available.

Figure 4-7. DC Signal With Random Noise



If the signal is zoomed, the image is as shown in [Figure 4-8](#).

Figure 4-8. Zoomed DC Signal With Random Noise



From the graph, the ADC count for a noisy signal (red colored signal) can be seen varying from 240 to 390 because of random noise. That means the ADC count is varying ± 75 counts (390 to 240 \rightarrow 315 ± 75 counts).

The ADC count for a filtered signal (green colored signal) can be seen varying from 315 to 335. That means the ADC count is varying ± 10 counts (335 to 315 \rightarrow 325 ± 10 counts).

Pros: Using this mode, the ADACC register gives accumulated values of up to 64 samples and the ADFLTR register gives an average value of all the accumulated samples so, software overhead can be avoided.

Cons: Determining the number of sample accumulation and clearing of ADACC registers needs to be handled in code.

4.5.3 Graphs: Average Mode

4.5.3.1 AC Signal With No Noise

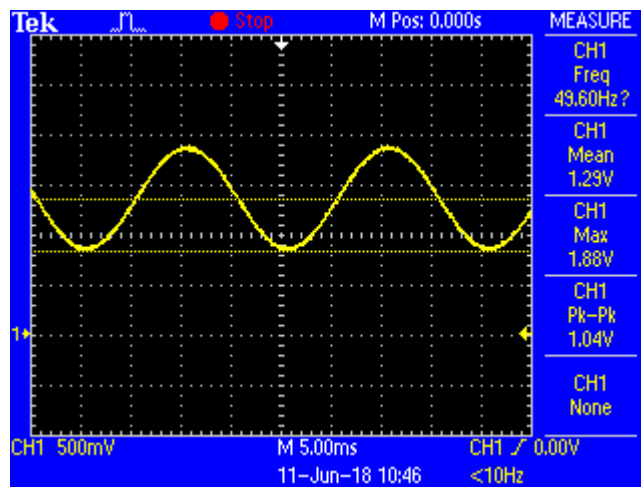
Task: The graph is plotted in *Data Visualizer* using an AC signal with no noise and an average value of a given input signal over a full cycle is observed.

- ADC² computation mode: Average mode
- Input signal: AC 1V peak-to-peak, frequency 50 Hz and offset at 1.25V

Test setup:

- Press push button S1 on the HPC board
- Verify the computation mode number in the *Data Visualizer* graph is 3
- Verify that LED D3 is illuminated
- Configure the Signal & Noise Generator to generate a sine wave of amplitude $\sim 1V$, frequency 50 Hz and offset at 1.25V
- Disable the random noise
- Verify the input signal using an oscilloscope. The expected result is as shown in [Figure 4-9](#)

Figure 4-9. AC Signal Oscilloscope Capture



The *Data Visualizer* graph is as shown in [Figure 4-10](#). The red colored signal is an AC input signal (ADRES value), the green colored signal is a filtered signal (ADFLTR value).

The average voltage of a periodic waveform whose two halves are exactly similar, either sinusoidal or non-sinusoidal, will be the mean of V_{MAX} and V_{MIN} over one complete cycle.

Sine waves are symmetrical about time axis, thus average is zero: positive area cancels negative area.

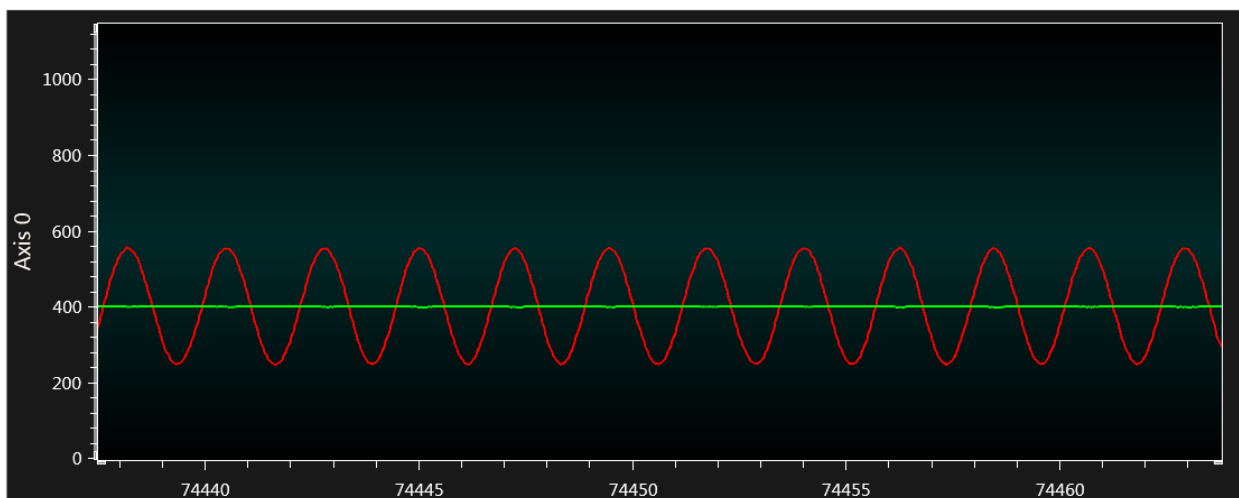
From the [Figure 4-10](#) it can be seen that the average result of the AC signal is approaching the offset value. In the example source code the averaging has been done with 32 samples as ADCRS bits

configured to 5 and ADC auto conversion trigger is at every 625 μ s. The advantage here is that the sample accumulation and averaging handling is not needed in code as ADACC and ADFLTR registers are available.

Average value calculation of an AC signal depends upon the number of equidistant samples accumulated in a cycle and the sampling frequency and frequency of the input signal.

To calculate the correct average value of 50 Hz (20 ms) signal over the full cycle with 32 equidistant samples, the sampling frequency should be, $20 \text{ ms}/32 = 625 \mu\text{s}$ (1.6 kHz).

Figure 4-10. 50 Hz AC Signal



As the input sine signal has an offset of 1.29V, the average value of the sine signal is the offset with ADC count 400 ($1023 \times 1.29/3.3 = 399$).

Pros: Using this mode, the ADACC register gives accumulated values of up to 64 samples and the ADFLTR register gives an average value of all the accumulated samples so, software overhead can be avoided.

Cons: The number of samples to be accumulated is limited to 64.

4.5.3.2 AC Signal With Random Noise

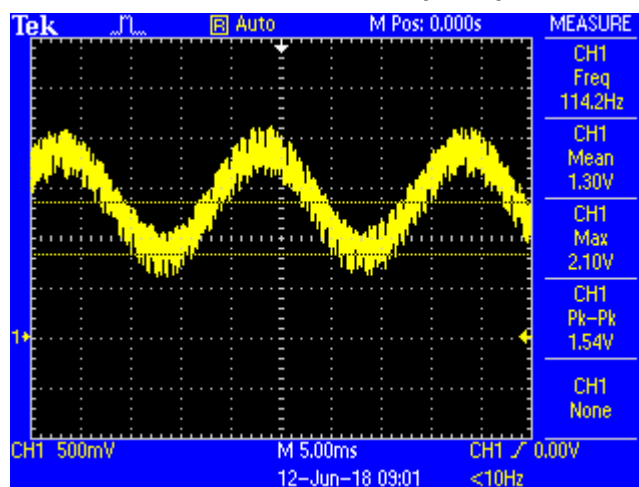
Task: The graph will be plotted in *Data Visualizer* using an AC signal with random noise and an average value of the given input signal over a full cycle is observed.

- ADC² computation mode: Average mode
- Input signal: AC 1V peak-to-peak, frequency 50 Hz and offset at 1.25V + random noise 0.5V peak-to-peak

Test setup:

- Verify the computation mode number in the *Data Visualizer* graph is 3
- Verify that LED D3 is illuminated
- Configure Signal & Noise Generator to generate an AC signal of amplitude ~1V, frequency 50 Hz, and offset at 1.25V
- Enable random noise with 0.5V peak-to-peak
- Verify the input signal using an oscilloscope. The expected result is as shown in [Figure 4-11](#).

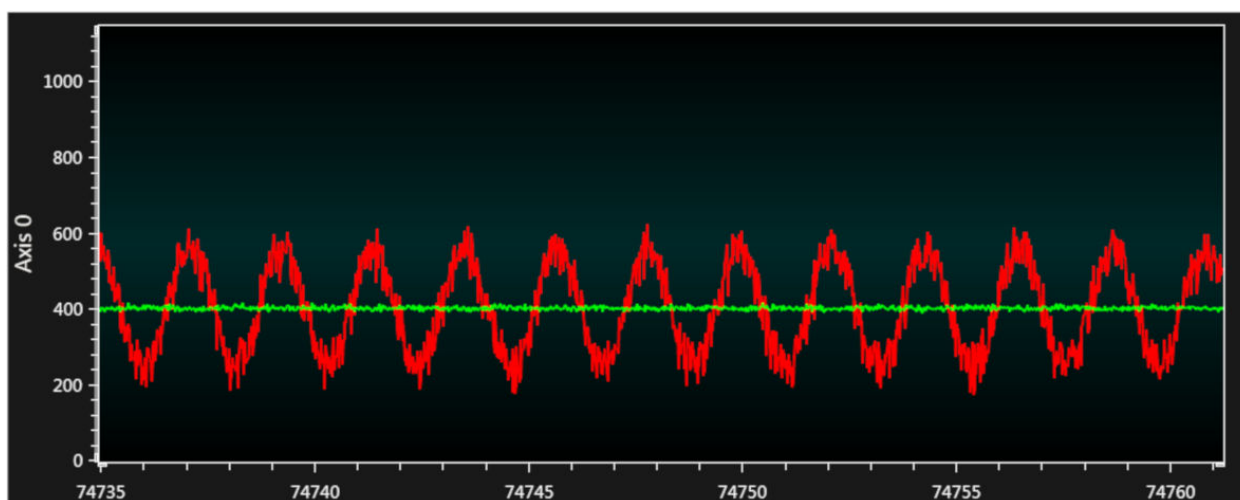
Figure 4-11. AC Signal With Random Noise Oscilloscope Capture



The *Data Visualizer* graph is as shown in Figure 4-12. The red colored signal is an AC input signal (ADRES value), the green colored signal is a filtered signal (ADFLTR value).

From the graph, it can be seen that the average result of a noisy AC signal is approaching to the offset value.

Figure 4-12. 50 Hz AC Signal



It can be seen that the average value of the noisy sine signal (the green colored signal) is ~390 to 410.

Note: It can be verified by adjusting the graph's Y-axis to get a clear view of the graphed signal.

- Click the graph and press and hold the Ctrl key, and scroll the mouse wheel

The average value of an AC signal without noise is ~400 from Figure 4-10. By comparing figures Figure 4-12 and Figure 4-10, it can be seen that an average value of a noisy AC signal is close to the average value of an AC signal without noise and using the ADC² Average mode makes it easier to calculate the average value of a noisy AC signal.

4.5.3.3 DC Signal With Random Noise

Task: The graph plotted in *Data Visualizer* is using a DC signal with random noise and noise suppression is observed.

- ADC² computation mode: Average mode

- Input signal: DC ~1V + random noise 0.5V peak-to-peak

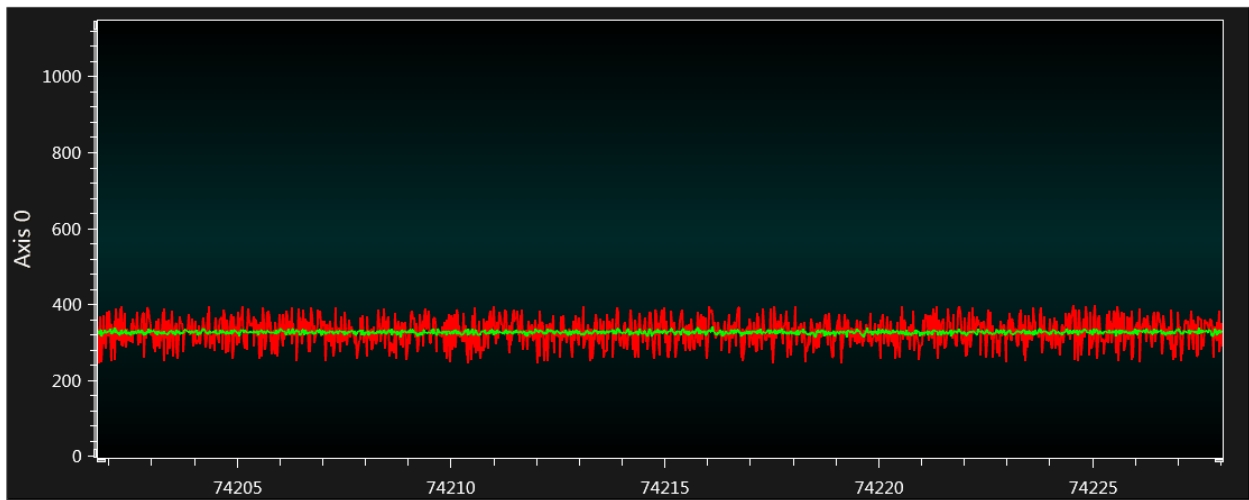
Test setup:

- Verify the computation mode number in the *Data Visualizer* graph is 3
- Verify that LED D3 is illuminated
- Configure the Signal & Noise Generator board to generate a DC signal of ~1V and random noise with 0.5V peak-to-peak
- Verify the input signal using an oscilloscope

The *Data Visualizer* graph is as shown in [Figure 4-13](#). The red colored signal is a noisy input signal (ADRES value), the green colored signal is a filtered signal (ADFLTR value).

From the graph, it can be seen that the noise has been suppressed using the ADC² Average mode. In the example source code the ADCRS bits are configured to 5 resulting the averaging with 32 samples. This graph looks similar to the Basic mode ([Figure 4-4](#)) and Accumulate mode ([Figure 4-7](#)) graphs. The advantage here is that the sample accumulation and averaging handling are not needed in code as the ADACC and ADFLTR registers are available.

Figure 4-13. DC Signal With Random Noise



4.5.4 Graphs: Burst Average Mode

4.5.4.1 AC Signal With Random Noise

Task: The graph is plotted in *Data Visualizer* using an AC signal with random noise and noise suppression in AC signal is observed.

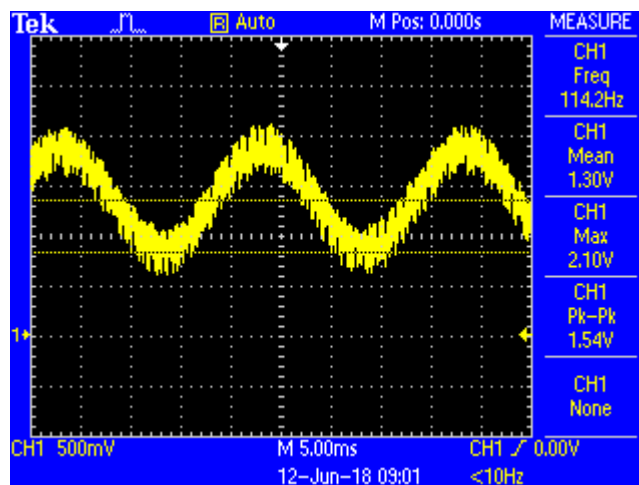
- ADC² computation mode: Burst Average mode
- Input signal: AC 1V peak-to-peak, frequency 50 Hz, and offset at 1.25V + random noise 0.5V peak-to-peak

Test setup:

- Press push button S1 on the HPC board
- Verify the computation mode number in the *Data Visualizer* graph is 4
- Verify that LED D4 is illuminated
- Configure Signal & Noise Generator to generate an AC signal of amplitude ~1V, frequency 50 Hz, and offset at 1.25V
- Enable random noise with 0.5V peak-to-peak

- Verify the input signal using an oscilloscope. The expected result is as shown in [Figure 4-14](#)

Figure 4-14. AC Signal With Random Noise Oscilloscope Capture



The *Data Visualizer* graph is as shown in [Figure 4-15](#). The red colored signal is an AC input signal (ADRES value), the green colored signal is filtered signal (ADFLTR value).

From the graph, it can be seen that the noise has been removed from the input noisy signal and relatively clean AC signal is illustrated in [Figure 4-15](#).

In this mode, with single ADC conversion trigger, all the data samples are accumulated up to the configured number of samples and after accumulating all samples, the average value of the accumulated sample is attained by reading the ADFLTR value which results in noise suppression in given input AC signal.

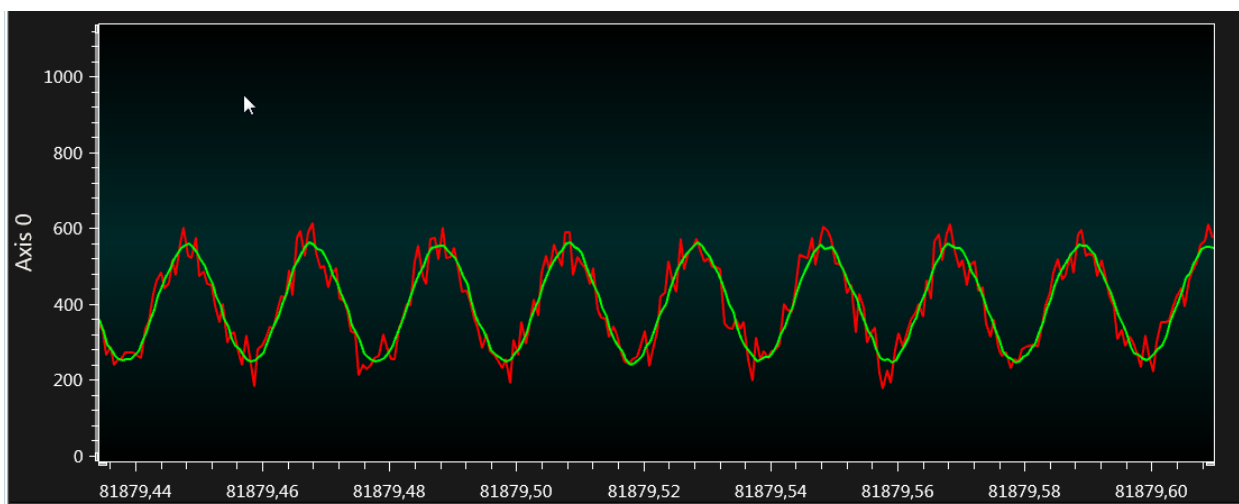
In the application, ADC Conversion Time for a single sample is configured to 11.5 μ s. So with a burst of 32 samples, total conversion time is $11.5 \times 32 = 368 \mu$ s. It means the maximum sampling rate can be configured to 2.7 kHz ($1/368 \mu$ s) with 32 samples using this mode.

With the burst of 64 samples, the maximum sampling rate can be achieved is 1.3 kHz ($1/736 \mu$ s).

Burst average mode helps in flattening out the noise from the AC signal and the clean AC signal is displayed.

Note: In the example source code the sampling rate is configured to 1.6 kHz ($1/625 \mu$ s).

Figure 4-15. AC Signal, 50 Hz



Pros: Noise suppression from an AC signal can be achieved using the *Burst Average* mode.

Cons:

- Sample accumulation is limited to 64 samples
- ADC sampling rate is affected by the number of samples accumulated in a single burst. Total conversion time for m samples is the multiplication of conversion time for one sample and m , the number of samples.

4.5.4.2 DC Signal With Random Noise

Task: The graph is plotted in *Data Visualizer* using a DC signal with random noise and noise suppression is observed.

- ADC² computation mode: Burst Average mode
- Input signal: DC ~1V + random noise 0.5V peak-to-peak

Test setup:

- Verify the computation mode number in the *Data Visualizer* graph is shown as 4 and that LED D4 is illuminated
- Configure Signal & Noise Generator to generate a DC signal of ~1V and random noise with 0.5V peak-to-peak
- Verify the input signal using an oscilloscope

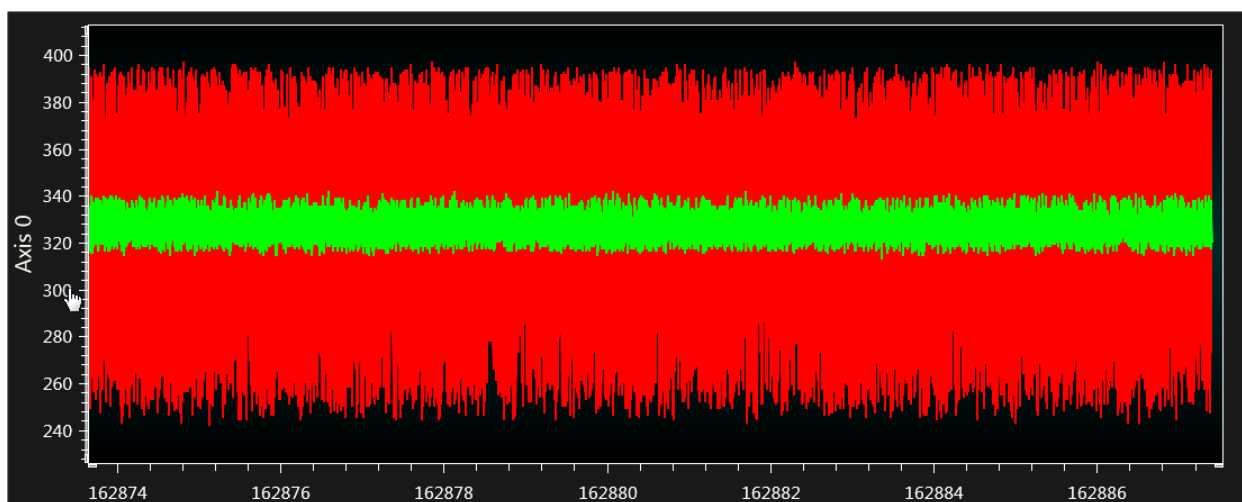
The *Data Visualizer* graph is as shown in [Figure 4-16](#). The red colored signal is a noisy input signal (ADRES value), the green colored signal is the filtered signal (ADFLTR value).

From the graph, it can be seen that the noise has been suppressed using ADC² Burst Average mode. In example source code averaging has been done with 32 samples as ADCRS bits configured to 5. The graph looks almost similar to the graph for Basic mode, Accumulate mode, and the Average mode where the ADC count of filtered signal (green colored signal) can be seen as ~ 315 to 340.

In Burst Average mode, the ADC count may vary more randomly than in other modes. Here the samples are accumulated in a burst. This means that once the conversion is triggered ADC samples are continuously read until 32 samples are collected. Because of this the noise filtering may not be as effective as other modes.

Use case: This mode can be useful in application where the input signal is relatively noise free and ADC resolution needs to be increased by oversampling.

Figure 4-16. DC Signal With Random Noise



Pros: Using this mode, the ADACC register gives accumulated values of up to 64 samples and the ADFLTR register gives an average value of all the accumulated samples in a single burst. Software overhead can be avoided.

Cons:

- The number of samples accumulated is limited to 64
- Noise reduction is comparatively less effective than the Basic, Accumulate, and Average mode
- The ADC sampling rate is affected by the number of samples accumulated in a single burst. Total conversion time for m samples is the multiplication of conversion time for one sample and m , the number of samples. In this example, the ADC Conversion Time is configured to 11.5 μs . So with a burst of 32 samples, total conversion time is $11.5 \times 32 = 368 \mu\text{s}$

4.5.5 Graphs: LPF Mode

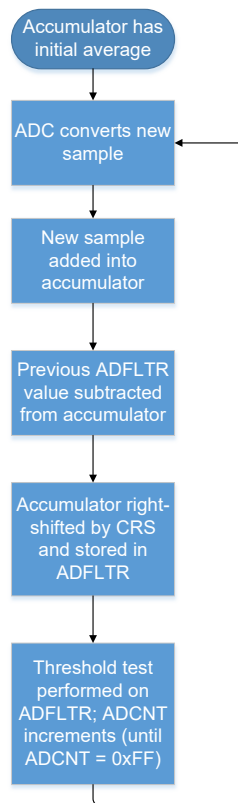
4.5.5.1 Low-Pass Filter Operation

The LPF can be considered as having two main processes that work in succession: an initial averaging process followed by a continuous filtering operation.

The initial averaging process begins by accumulating samples until the ADC Count register (ADCNT) is equal to the ADRPT register. During this initial accumulation process, each new sample is added into the accumulator. After each sample is added, the accumulator right-shifts (divides) its current value by the value of the ADC Accumulated Calculation Right Shift Selection (CRS<2:0>) bits of the ADCON2 register. The new right-shifted value appears in the ADFLTR registers. When $\text{ADCNT} = \text{ADRPT}$, a threshold comparison test is performed on the ADFLTR value. During this initial averaging process, the ADRPT value acts as an RC time constant, allowing the computed average to reach a steady state before performing a threshold comparison. This prevents threshold tests on each sample until after an average has been established, which helps to reduce 'false alarm' threshold violations due to random variations of a single sample.

Once the initial averaging process completes, the module moves into continuous filtering operation. The figure below explains what happens during continuous filtering operation.

Figure 4-17. Continuous Filtering Operation Flowchart



Equation 4-1 explains the ADFLTR calculation in mathematical terms. It is important to note that the accumulator is not cleared after the initial averaging process, or after any subsequent conversion, but instead continues to accumulate samples until software disables the module. During continuous filtering operation, ADRPT is ignored, ADCNT continues to count until ADCNT = 0xFF (ADCNT is ignored after reaching 0xFF), and the CRS value continues to act as the accumulator divider.

Equation 4-1. ADFLTR Calculation in LPF Mode

$$ADFLTR = \frac{ACC_{NEW}}{2^{ADCRS}}$$

Where:

$$ACC_{NEW} = (ACC_{PREV} + ADRES) - \frac{ACC_{PREV}}{2^{ADCRS}}$$

ACC_{PREV} – Previous accumulator result

ADRES – Current conversion result

4.5.5.2 DC Signal With Random Noise

Task: The graph is plotted in *Data Visualizer* using a DC signal with random noise and noise suppression is observed.

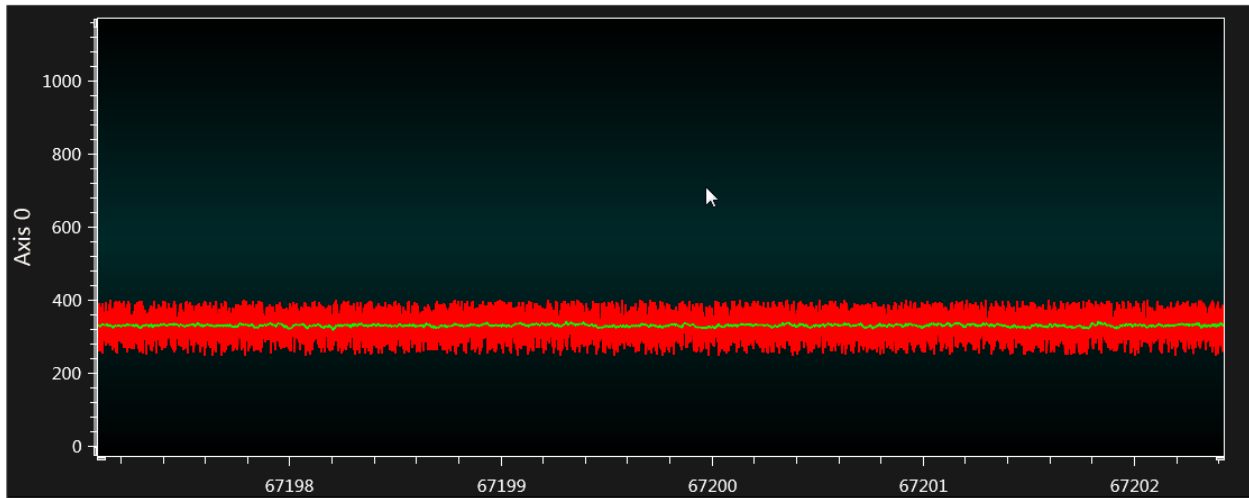
- ADC² computation mode: LPF mode
- Input signal: DC ~1V + random noise 0.5V peak-to-peak

Test setup:

- Press push button S1 on the HPC board and verify the computation mode number in the *Data Visualizer* graph is shown as 5 and that LED D5 is illuminated
- Configure Signal & Noise Generator to generate a DC signal of ~1V and random noise with 0.5V peak-to-peak
- Verify the input signal using an oscilloscope

The *Data Visualizer* graph is as shown in [Figure 4-18](#). The red colored signal is the noisy input signal (ADRES value), the green colored signal is the filtered signal (ADFLTR value).

Figure 4-18. DC Signal With Random Noise



From the [Figure 4-18](#), it can be seen that the noise has been suppressed using the ADC² *LPF* mode. In the example source code the ADCRS bits are configured to 5. If zoomed in, the graph looks almost similar to the graph for *Basic mode*, *Accumulate mode*, and the *Average mode* where the ADC count of the filtered signal can be seen as ~ 315 to 340.

In *LPF* mode, a new ADC result is available in ADFLTR with every new sample, while in *Average mode* a new filtered value is available every 2^{ADCRS} sample.

The ADCRS can be configured from 1 to 6 for *LPF* mode. When the signal is noisy, the ADCRS value determines how drastically the filtered output changes. When the ADCRS value is higher (e.g. '5' or '6'), sudden changes in the input signal have less of an influence on the output signal. Conversely, when the ADCRS value is lower (e.g. '1' or '2'), sudden changes in the input signal are also observed in the output signal. It means when ADCRS = 1, noise has a significant impact on the output signal. When ADCRS = 6, noise has much less of an impact on the output signal.

Pros: The *LPF* mode suppresses noise from a DC signal by taking a continuous running average of samples without software overhead.

Cons: The ADCRS bits act as an RC time constant and it can be configured from 1 to 6. As the ADCRS value increases, the time it takes for the filtered output to achieve a steady state increases, but the effects of any deviations from the overall average have less of an impact on the filtered output.

4.5.5.3 AC Signal

Task: The graph will be plotted in *Data Visualizer* using an AC signal without noise and the impact of the *LPF* mode on a given input AC signal is observed. In other words, when the frequency of a given input AC is reached to roll-off frequency, the attenuation in an input signal is observed.

- ADC² computation mode: *LPF* mode
- Input signal: AC 1V peak-to-peak, frequency 1 Hz, and offset at 1.25V

Test setup:

- Verify the computation mode number in the *Data Visualizer* graph is shown as 5 and that LED D5 is illuminated
- Configure the Signal & Noise Generator to generate an AC signal of amplitude ~1V, frequency 1 Hz, and offset at 1.25V
- Disable the random noise if enabled earlier
- Verify the input signal using an oscilloscope. The expected result is as shown below

The LPF removes unwanted high frequency signals from an AC input signal. The LPF hardware operates the same way as when it filters DC signals; each new sample is added to the accumulator, averaged, and divided by 2^{ADCRS} to get the filtered output. The difference between AC and DC signal is how the ADCRS value is used. When filtering AC signals, the ADCRS value determines the -3 dB roll-off frequency of the single-pole filter.

Table 4-5 shows the -3 dB points in terms of radians per second for each valid ADCRS value.

Note: The table with -3 dB points for each ADCRS value is available in the data sheet in the ADC² chapter of any device containing the ADC² module.

4.5.5.3.1 ADCRS Effects on -3 dB Roll-Off Frequency

In Low-Pass Filter mode, the ADCRS value also determines the -3 dB roll-off frequency of the single-pole filter. The table below shows the radian values at the -3 dB roll-off frequency based on ADCRS values.

Table 4-5. Radian Values at -3 dB Roll-Off

ADCRS	RPT	Radians @ -3 dB Roll-Off
1	2	0.72
2	4	0.284
3	8	0.134
4	16	0.065
5	32	0.032
6	64	0.016

The radian values listed in the table above are defined by the ADC²'s hardware. These values are used to calculate the -3 dB roll-off point in terms of frequency. The following equation can be used to determine the -3 dB point.

Equation 4-2. Equation -3 dB Roll-Off Frequency

$$\text{Frequency@-3dBpoint} = \frac{\text{Radians@-3dB}}{2\pi T}$$

Where:

Radians @ -3 dB = the value from the Table 4-5 based on the ADCRS value.

T = total sampling time.

The total sampling time is the measured time between samples.

The total sampling time includes the ADC acquisition time, the conversion time, interrupt time, and the time required for post-processing of the ADC result such as serial transmission etc. The number of instructions contained in the ADC routine also influences the total sampling time.

In this example, after acquiring the ADC sample, the ADC result is transmitted over EUSART. Baud rate is kept sufficiently high so the sampling rate is not affected by serial transmission.

The [Table 4-6](#) shows the different roll-off frequencies (*Calculated Frequency @ -3 dB Point (Hz)*) which is calculated by using, [Equation 4-2](#) where the sampling time $T = 625 \mu s$, as the ADC auto conversion trigger is configured to $625 \mu s$ in example source code.

For example: when $ADCRS = 5$, Radians @ -3 dB Cut-Off = 0.032, $T = 625 \mu s$.

$$Frequency@-3dBpoint = \frac{0.032}{2\pi \times 625} = 8.15 Hz$$

Once the given input AC signal is reached to ~8 Hz, the attenuation in the input AC signal is observed.

Table 4-6. Effects of Sampling Time on Roll-Off Frequency

ADCRS	Radians @ -3 dB Cut-Off	Measured Sampling Time [μs]	Calculated Frequency @ -3 dB Point [Hz]
1	0.72	625.0	183.34
2	0.284	625.0	72.35
3	0.134	625.0	34.14
4	0.065	625.0	16.56
5	0.032	625.0	8.15
6	0.016	625.0	4.07

A graph illustration using $ADCRS = 5$ will be demonstrated in the next section.

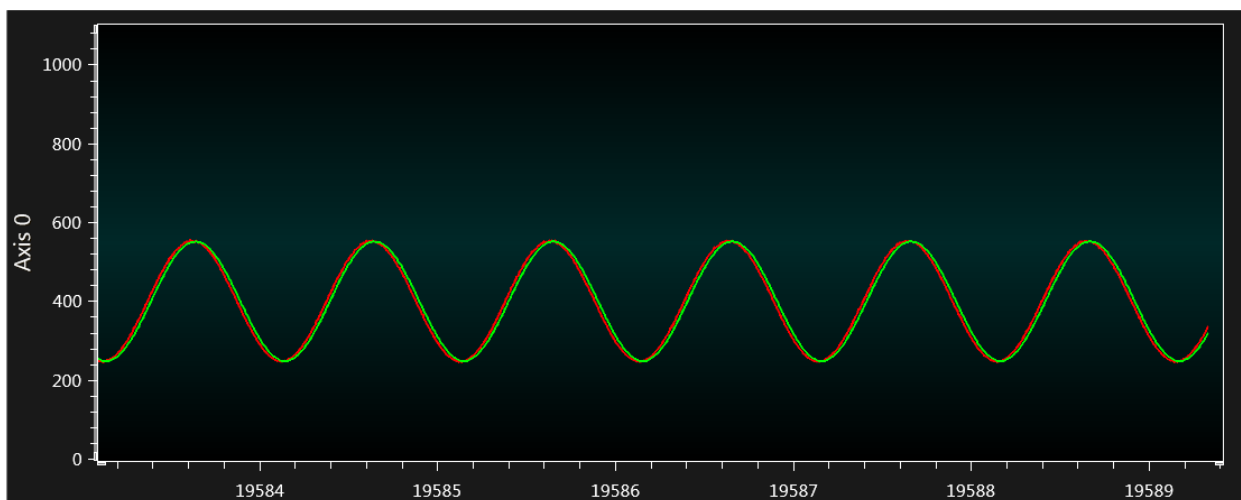
4.5.5.3.2 Graph Illustration

Sine Signal, 1 Hz:

In the example source code, $ADCRS$ is configured to 5. Referring to [Table 4-6](#), the -3 dB roll-off point is 8 Hz with $ADCRS = 5$.

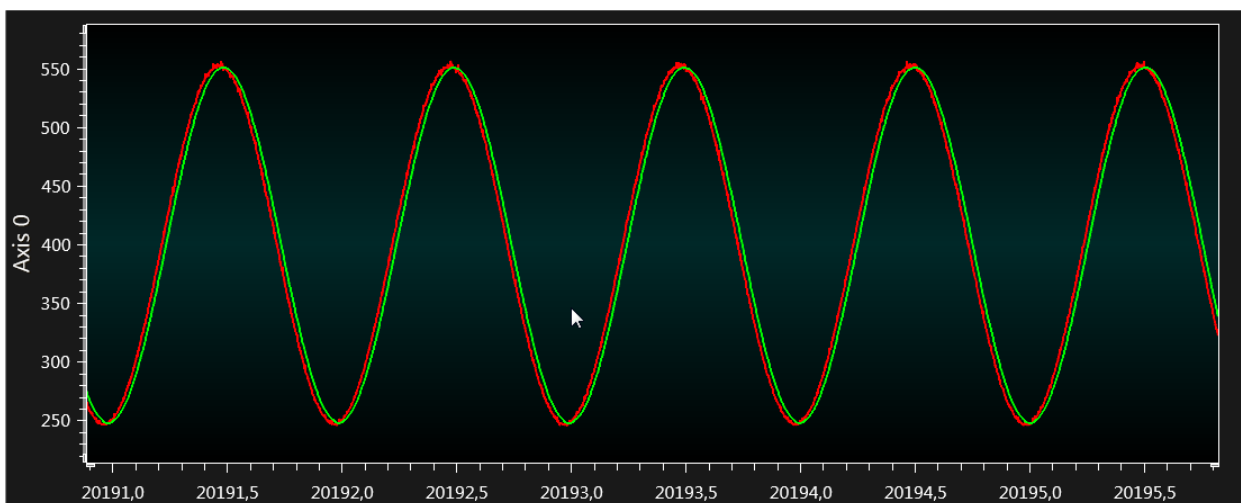
The ADC input signal has a 1 Hz frequency which is below the -3 dB roll-off point at 8 Hz. As 1 Hz is safely within the low pass-band of the filter, filtered signal and original signal are equal. It is illustrated in the [Figure 4-19](#) graph, the red colored signal is the ADC input signal ($ADRES$ values), the green colored signal is the filtered signal ($ADFLTR$ values).

Figure 4-19. AC Signal, 1 Hz



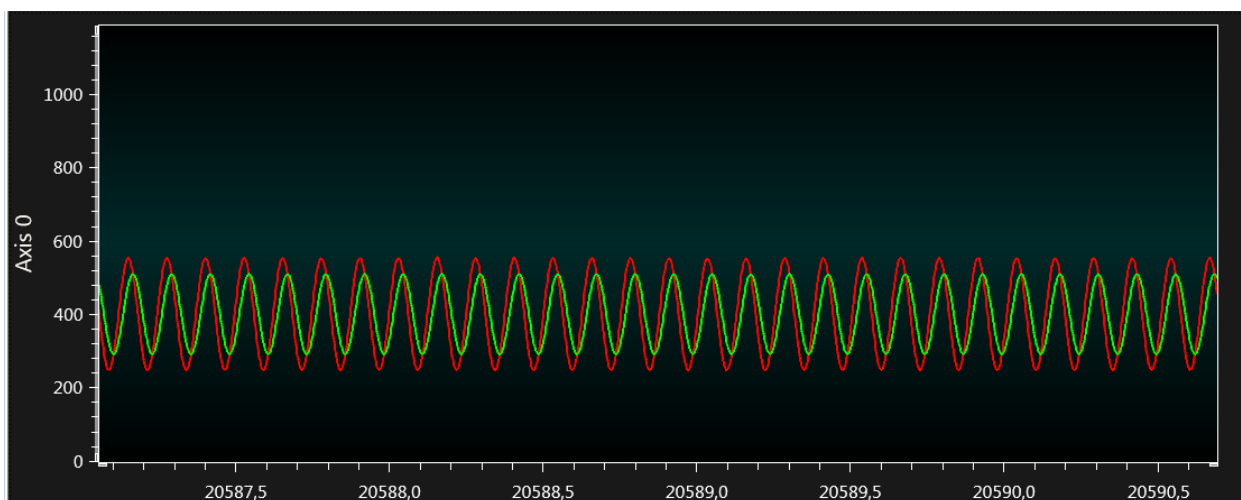
If the signal is zoomed in by selecting 'Automatically fit Y' checkbox available below graph, it can be seen that the ADC count is varying from 560 to 250. So the difference between ADC counts is 310, which is the ADC count value for configured input signal 1V peak-to-peak ($310 \times 3.3 / 1023 = 1$).

Figure 4-20. Zoomed In Signal, AC 1 Hz



Sine Signal, 8 Hz:

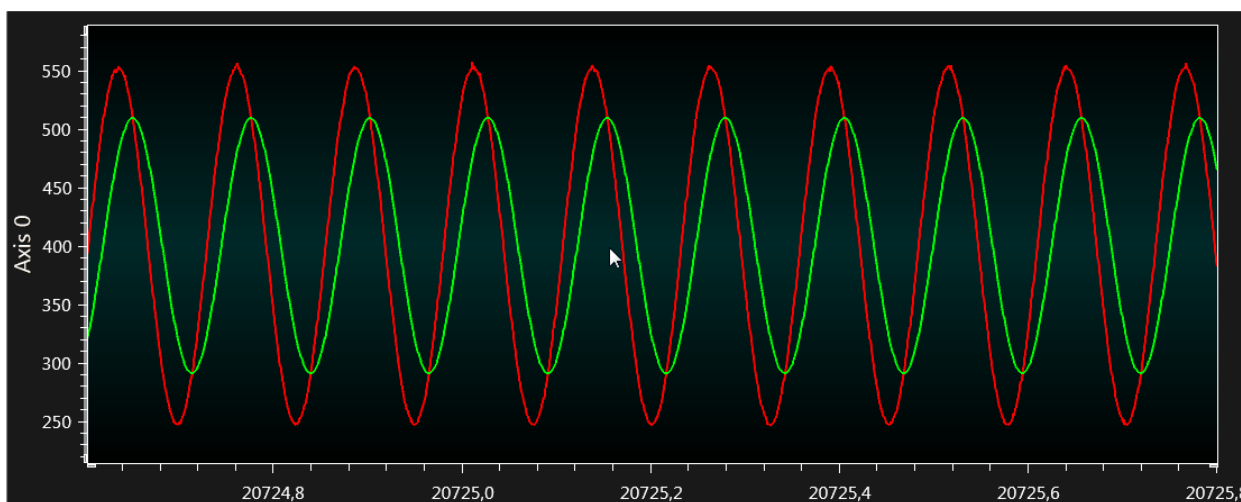
Now, configure the Signal & Noise Generator to generate an 8 Hz signal and adjust the scale on the graph and observe the graph.

Figure 4-21. AC Signal, 8 Hz

When the sine-wave's frequency is increased and reaches 8 Hz, a reduction in peak-to-peak voltage takes place as the filter actively reduces the magnitude of the signal, as observed in [Figure 4-21](#).

At the -3 dB frequency, the peak-to-peak amplitude should be roughly 70.7% of the original input signal.

If the signal is zoomed in the image is as shown in [Figure 4-22](#).

Figure 4-22. Zoomed In Signal AC 8 Hz

From the graph, the ADC count for the filtered signal (green colored signal) is ~ 290 to 510 (difference in ADC count values is 220) which gives the peak-to-peak voltage at 8 Hz, 0.709 V ($220 \times 3.3 / 1023 = 0.709$).

If the original 1V peak-to-peak is multiplied by 0.707, a peak-to-peak voltage is 0.707V.

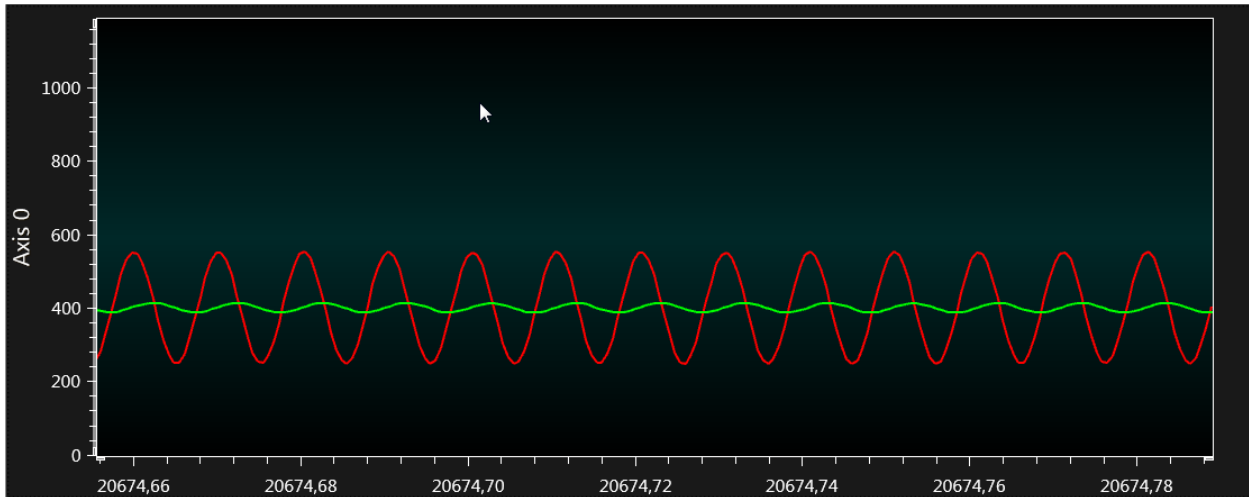
From the graph, the filtered signal shows that it is 70.7% of the original input signal.

If the frequency is further increased, further attenuation in an input signal is seen.

Sine Signal, 100 Hz:

Configure the Signal & Noise Generator to generate a 100 Hz signal and adjust the scale on the graph and observe the graph.

Figure 4-23. AC Signal, 100 Hz



As the frequency continues to increase, the peak-to-peak range will shrink, as observed in [Figure 4-23](#).

4.5.5.3.3 LPF Mode: AC Signal With Random Noise

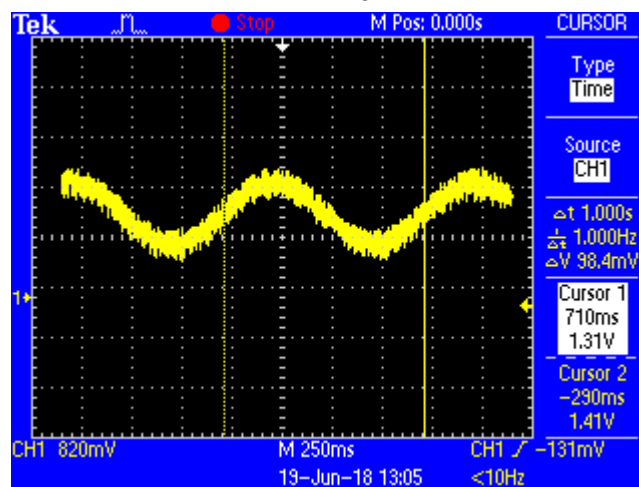
Task: The graph will be plotted in *Data Visualizer* using an AC signal with random noise and the effect of LPF in noise suppression is observed

- ADC² computation mode: LPF mode
- Input signal: AC 1V peak-to-peak, frequency 1 Hz, and offset at 1.25V

Test setup:

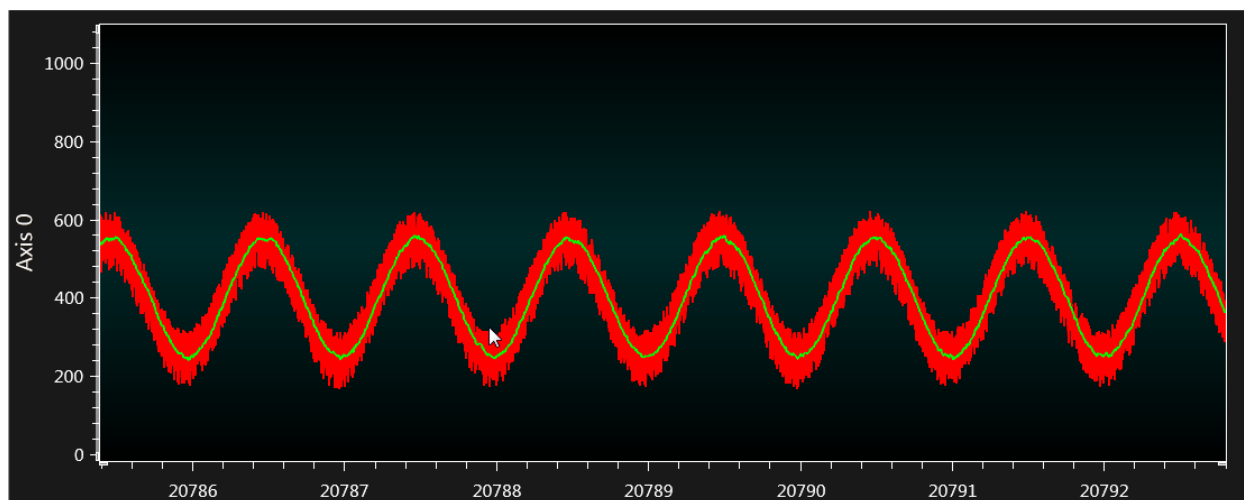
- Verify the Computation mode number in the *Data Visualizer* graph is shown as 5 and that LED5 is illuminated
- Configure Signal & Noise Generator to generate an AC signal of amplitude ~1V, frequency 1 Hz, and offset at 1.25V
- Enable the random noise with 0.5V peak-to-peak
- Verify the input signal using an oscilloscope. The expected result is as shown in [Figure 4-24](#).

Figure 4-24. AC 1 Hz + Random Noise Oscilloscope Capture



The *Data Visualizer* graph is as shown in [Figure 4-25](#). The red colored signal is the noisy AC input signal (ADRES value), the green colored signal is the filtered signal (ADFLTR value).

Figure 4-25. AC Signal, 1 Hz With Random Noise



The 1 Hz signal is below the 8 Hz roll-off frequency with configured ADCRS value 5 (refer to [Table 4-6](#)). From the graph in [Figure 4-25](#), it can be seen that when random noise is added to the 1 Hz signal, the noise suppression is achieved using the *LPF* mode and a relatively clean signal is observed.

Pros: The LPF removes unwanted high frequency signals from an AC input signal.

4.5.6 Graph Illustration: All Modes Difference

Noise suppression result on DC signal with random noise in all modes is illustrated in one graph in [Figure 4-26](#).

Figure 4-26. Graph Illustration: All Modes Difference

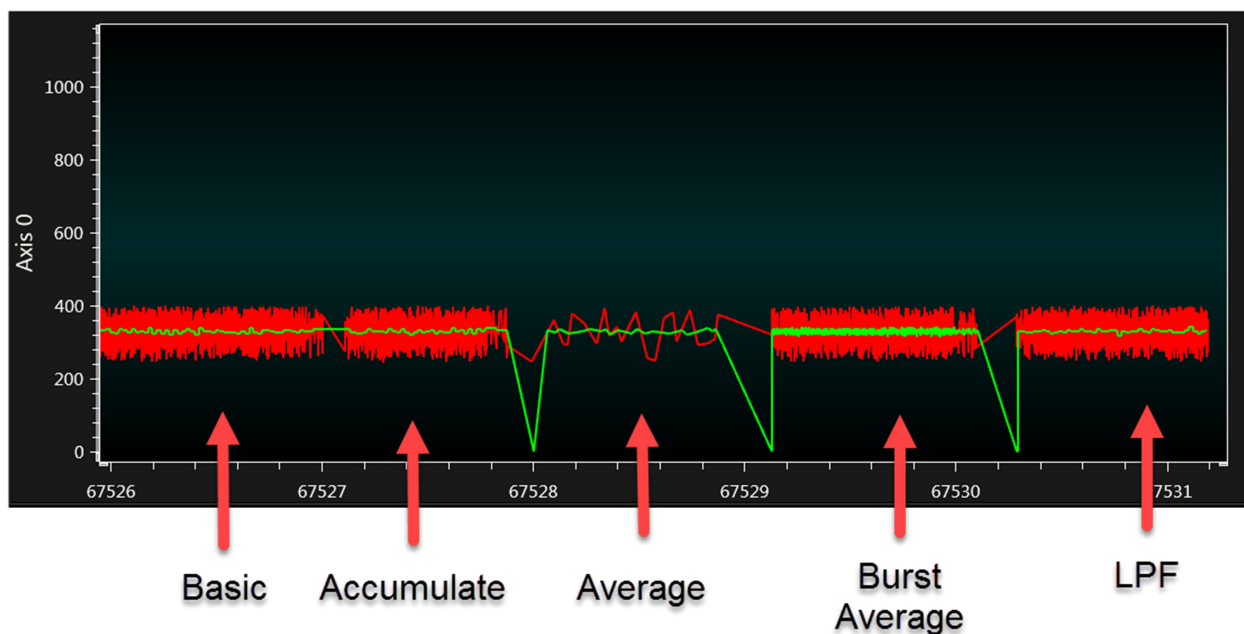


Table 4-7. Difference

	Basic	Accumulate	Average	Burst Average	LPF
Pros	The number of samples to be accumulated for averaging is not limited whereas it is limited to 64 for other ADC ² modes.	ADACC register gives accumulated values of up to 64 samples and ADFLTR register gives an average value of all the accumulated samples so, software overhead can be avoided.	No software overhead for sample accumulation and averaging as ADACC and ADFLTR registers are available.	Noise suppression from an AC signal can be achieved. No software overhead while averaging.	Removes unwanted high frequency signals from an AC input signal.
Cons	Software overhead of accumulating and averaging needs to be handled in code.	The number of samples accumulated check and clearing of ADACC registers need to be handed in code.	The number of samples to be accumulated is limited to 64.	Noise reduction comparatively less effective. ADC sampling rate is affected by the number of samples accumulated in a single burst.	The ADCRS bits act as an RC time constant in DC signals. With increased ADCRS, the time it takes for the filtered output to achieve a steady state increases.

5. Noise Level Comparison With ADC Reference 5V/3.3V

Here, the *Data Visualizer* graph of a DC noisy signal is plotted with both 3.3V and 5V ADC reference and the noise level is compared with a given input DC signal $VDD/2$.

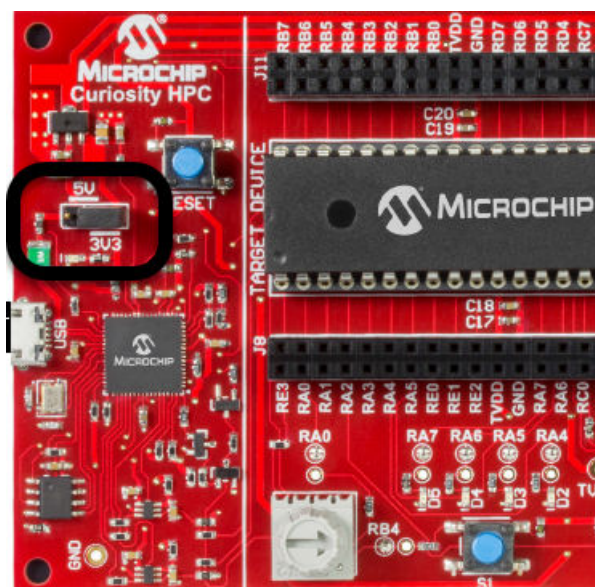
As ADC is configured with reference VDD, by changing the jumper on HPC board the ADC reference voltage can be changed.

Input signal from Signal & Noise Generator board: DC $VDD/2$ + random noise 0.5V peak-to-peak.

The ADC² computation mode is selected as the Average mode.

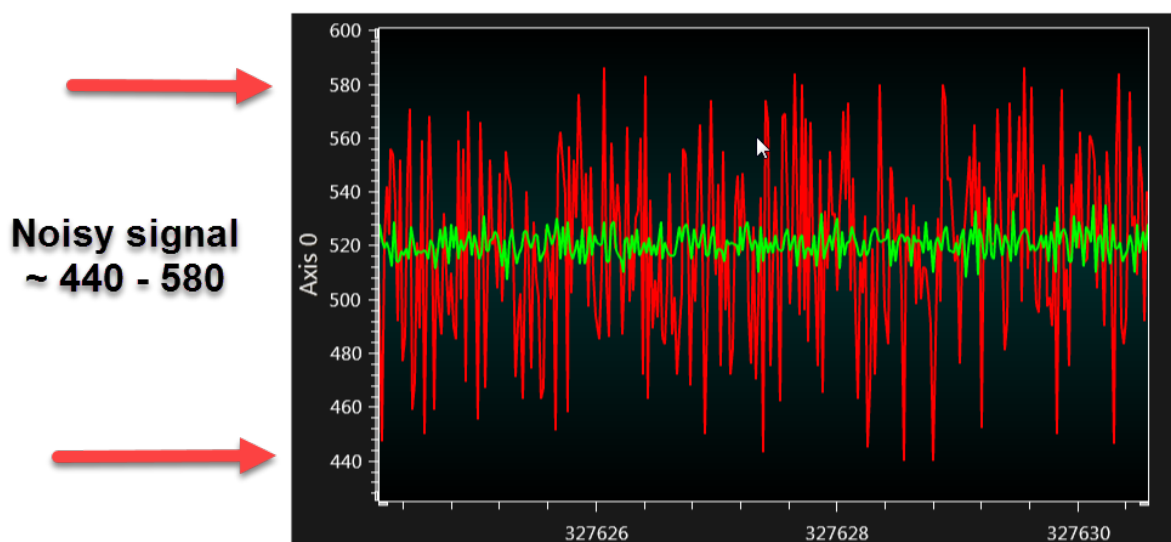
5.1 Graph With ADC Reference 3.3V

Figure 5-1. VDD Selection 3V3



The *Data Visualizer* graph with $VDD = 3.3V$ and ADC input signal $VDD/2 = 1.65V$ is as shown in [Figure 5-2](#). The red colored signal is the noisy input signal (ADRES value), the green colored signal is the filtered signal (ADFLTR value).

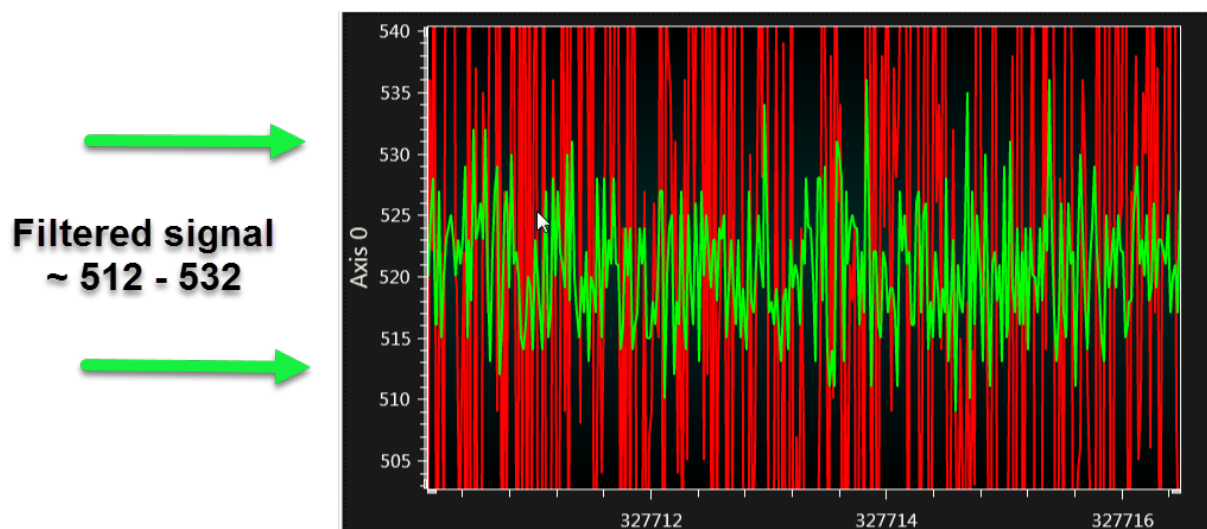
Figure 5-2. Graph With DC 1.65V and ADC Reference 3.3V



From the graph, it can be seen that the ADC count value for a non-filtered signal (noisy input signal) is ~ 440 - 580.

If the signal is zoomed in, it is illustrated in [Figure 5-3](#).

Figure 5-3. Zoomed In Graph With DC 1.65V and ADC Reference 3.3V



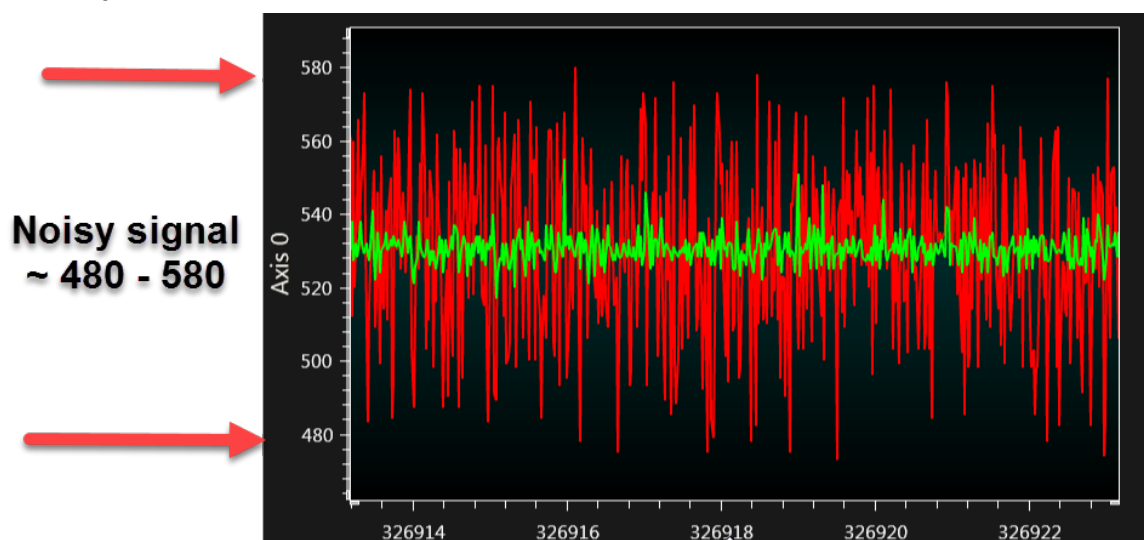
It can be seen from [Figure 5-3](#) that the ADC count value for the filtered signal is ~ 512 - 532.

5.2 Graph With ADC Reference 5V

Note: The jumper position on HPC needs to be selected to configure VDD = 5V and the input signal from Signal & Noise Generator is 2.5V

The *Data Visualizer* graph with VDD = 5V and ADC input signal VDD/2= 2.5V is as shown in [Figure 5-4](#). The red colored signal is the noisy input signal (ADRES value), the green colored signal is the filtered signal (ADFLTR value).

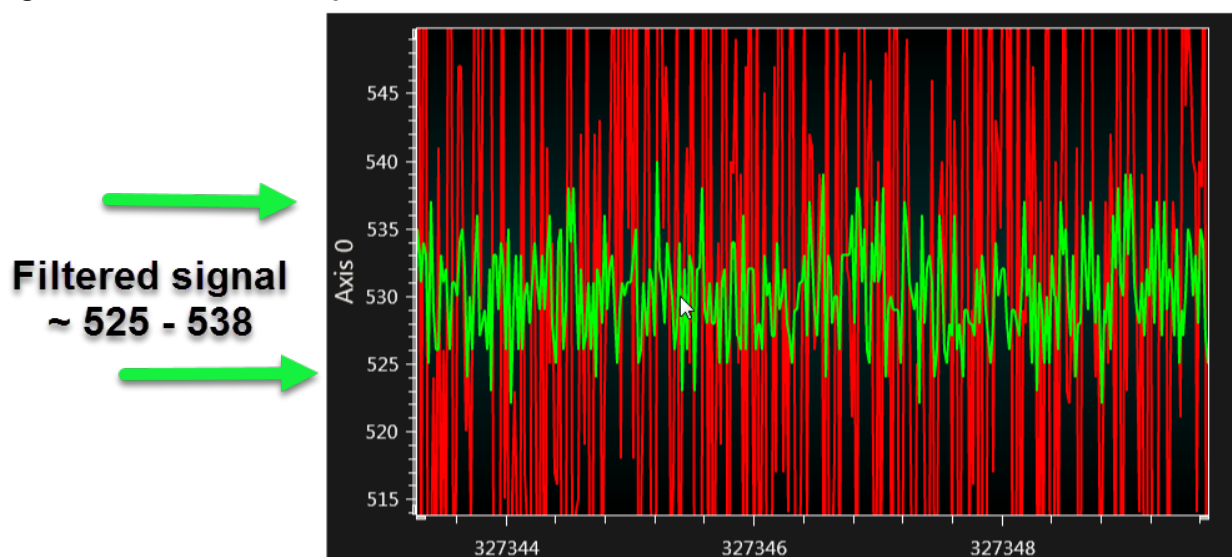
Figure 5-4. Graph With DC 2.5V and ADC Reference 5V



From the graph, it can be seen that the ADC count value for a non-filtered signal (noisy input signal) is ~ 480 - 580.

A zoomed in signal is illustrated in [Figure 5-5](#).

Figure 5-5. Zoomed In Graph With DC 2.5V and ADC Reference 5V



It can be seen from [Figure 5-5](#) that the ADC count value for a filtered signal is ~525 - 538.

A noise level comparison table is as shown in [Table 5-1](#).

Table 5-1. Noise Level Comparison 3.3V and 5V With Approximate ADC Count Values

	ADC Reference 3.3V	ADC Reference 5V
ADC count NON filtered	440 - 580	480 - 580
Noise count NON Filtered	140	100
ADC count Filtered	512 - 532	525 - 538
Noise Count Filtered	20	13

A signal with 5V ADC reference shows less noise count than 3.3V ADC reference.

6. Appendix A: Plotting Graph in *Data Visualizer*

Note: For detailed information on *Data Visualizer*, refer to the [Data Visualizer User's Guide](#).

In the example source code, the ADC result value is sent through EUSART to the serial terminal of the *Data Visualizer* and this serial terminal data is fed as input to plot the graph.

The data streamer protocol is used to send the ADC result to the serial terminal.

How to use data streamer protocol to send 16-bit value:

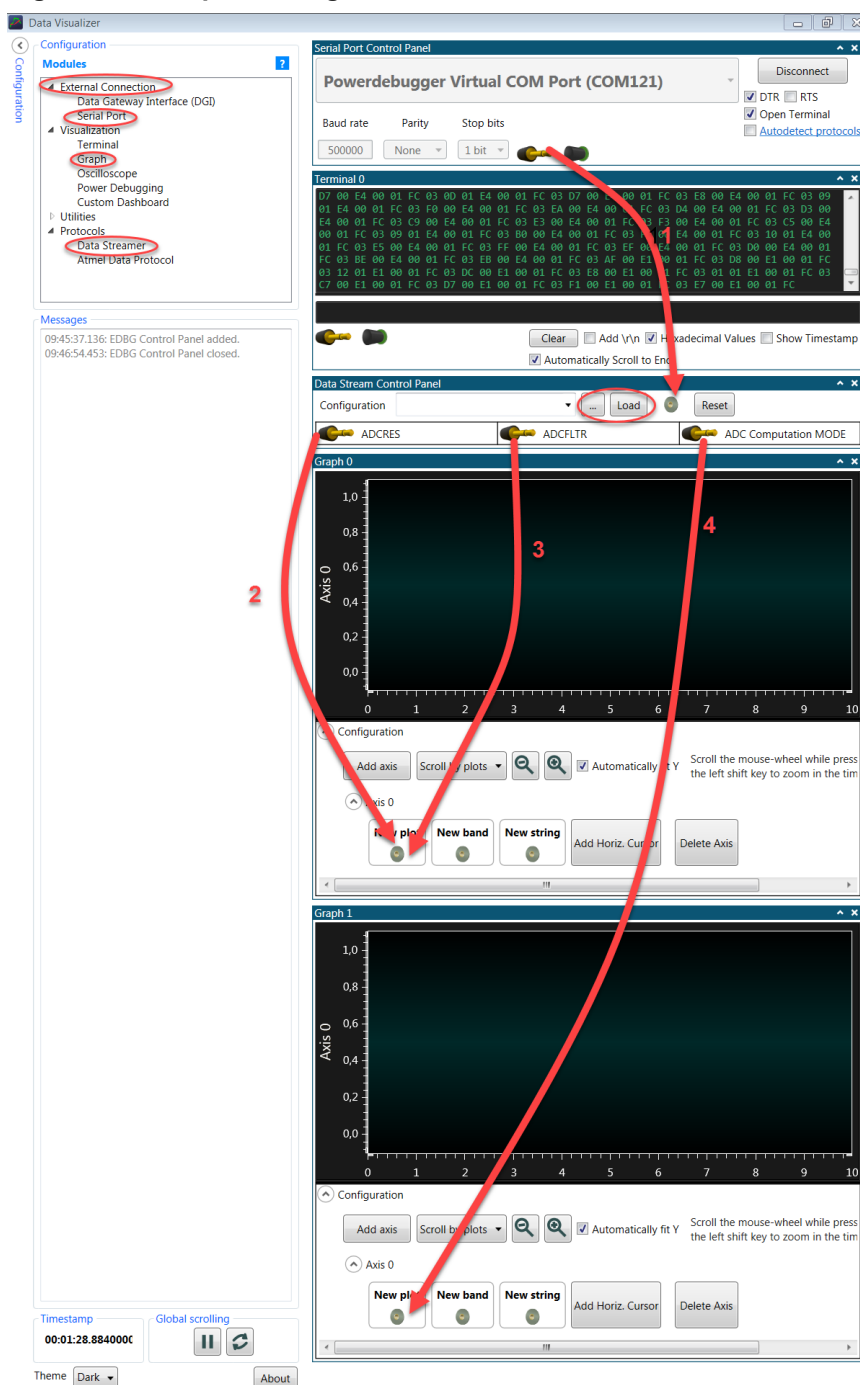
The ADC has been configured for 10 bits and this 10-bit ADC result needs to be sent to an 8-bit EUSART. As one ADC result value will be sent as two bytes, a data streamer protocol has been used to send the ADC result to a serial terminal as below, so that one 16-bit value will be used to plot the graph.

```
EUSART_write(0x03); //START
EUSART_write(adc_data_LSB);
EUSART_write(adc_data_MSB);
EUSART_write(0xFC); //END
```

Data Visualizer Configuration:

- Open *Data Visualizer*
- Open *Configuration* → *External Connection* → *Serial Port*, in *Data Visualizer*
- Select the EDBG Virtual COM port, Baud rate: 500000 and then select *Connect*
- Open *Configuration* → *Protocols* → *Data Streamer*
- In *Data Stream Control Panel*, under *Configuration*, browse to the configuration file and then select *Load*
Note: In this case, the configuration file is `noise_suppression_data_streamer.txt` and can be found in the example source code project folder.
Note: For more details on *Data Streamer*, refer to [Data Visualizer User's Guide](#).
- Open *Configuration* → *Visualization* → *Graph*
Note: Open two windows of *Graph*, one for 'ADC results' and one for 'ADC Computation mode number'.
- Drag the connections as shown with red arrows in [Figure 6-1](#) to plot the graph

Figure 6-1. Graph Plotting



To adjust the Y-axis in the graph, follow the points below:

- Under *Configuration* in *Graph*, deselect *Automatically Fit Y*
- Click somewhere inside the plot area
- Scroll the mouse-wheel while pressing or holding the *Ctrl* key

To adjust the X-axis in the graph, follow the points below:

- Click somewhere inside the plot area

- Scroll the mouse-wheel while pressing or holding the *Shift* key

Note: For more details on Data Visualizer → Graph, refer to the [Data Visualizer User's Guide](#).

7. Revision History

Doc. Rev.	Date	Comments
A	06/2019	Initial document release

The Microchip Website

Microchip provides online support via our website at <http://www.microchip.com/>. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to <http://www.microchip.com/pcn> and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the web site at: <http://www.microchip.com/support>

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE. Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AnyRate, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, chipKIT, chipKIT logo, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PackTime, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TempTrackr, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

APT, ClockWorks, The Embedded Control Solutions Company, EtherSynch, FlashTec, Hyper Speed Control, HyperLight Load, IntelliMOS, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet-Wire, SmartFusion, SyncWorld, Temux, TimeCesium, TimeHub, TimePictra, TimeProvider, Vite, WinPath, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BlueSky, BodyCom, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, INICnet, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

The Adaptec logo, Frequency on Demand, Silicon Storage Technology, and Symmcom are registered trademarks of Microchip Technology Inc. in other countries.

GestlC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.

© 2019, Microchip Technology Incorporated, Printed in the U.S.A., All Rights Reserved.

ISBN: 978-1-5224-3557-0

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit <http://www.microchip.com/quality>.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: http://www.microchip.com	Australia - Sydney Tel: 61-2-9868-6733 China - Beijing Tel: 86-10-8569-7000 China - Chengdu Tel: 86-28-8665-5511 China - Chongqing Tel: 86-23-8980-9588 China - Dongguan Tel: 86-769-8702-9880 China - Guangzhou Tel: 86-20-8755-8029 China - Hangzhou Tel: 86-571-8792-8115 China - Hong Kong SAR Tel: 852-2943-5100 China - Nanjing Tel: 86-25-8473-2460 China - Qingdao Tel: 86-532-8502-7355 China - Shanghai Tel: 86-21-3326-8000 China - Shenyang Tel: 86-24-2334-2829 China - Shenzhen Tel: 86-755-8864-2200 China - Suzhou Tel: 86-186-6233-1526 China - Wuhan Tel: 86-27-5980-5300 China - Xian Tel: 86-29-8833-7252 China - Xiamen Tel: 86-592-2388138 China - Zhuhai Tel: 86-756-3210040	India - Bangalore Tel: 91-80-3090-4444 India - New Delhi Tel: 91-11-4160-8631 India - Pune Tel: 91-20-4121-0141 Japan - Osaka Tel: 81-6-6152-7160 Japan - Tokyo Tel: 81-3-6880-3770 Korea - Daegu Tel: 82-53-744-4301 Korea - Seoul Tel: 82-2-554-7200 Malaysia - Kuala Lumpur Tel: 60-3-7651-7906 Malaysia - Penang Tel: 60-4-227-8870 Philippines - Manila Tel: 63-2-634-9065 Singapore Tel: 65-6334-8870 Taiwan - Hsin Chu Tel: 886-3-577-8366 Taiwan - Kaohsiung Tel: 886-7-213-7830 Taiwan - Taipei Tel: 886-2-2508-8600 Thailand - Bangkok Tel: 66-2-694-1351 Vietnam - Ho Chi Minh Tel: 84-28-5448-2100	Austria - Wels Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 Denmark - Copenhagen Tel: 45-4450-2828 Fax: 45-4485-2829 Finland - Espoo Tel: 358-9-4520-820 France - Paris Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 Germany - Garching Tel: 49-8931-9700 Germany - Haan Tel: 49-2129-3766400 Germany - Heilbronn Tel: 49-7131-72400 Germany - Karlsruhe Tel: 49-721-625370 Germany - Munich Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 Germany - Rosenheim Tel: 49-8031-354-560 Israel - Ra'anana Tel: 972-9-744-7705 Italy - Milan Tel: 39-0331-742611 Fax: 39-0331-466781 Italy - Padova Tel: 39-049-7625286 Netherlands - Drunen Tel: 31-416-690399 Fax: 31-416-690340 Norway - Trondheim Tel: 47-72884388 Poland - Warsaw Tel: 48-22-3325737 Romania - Bucharest Tel: 40-21-407-87-50 Spain - Madrid Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 Sweden - Gothenberg Tel: 46-31-704-60-40 Sweden - Stockholm Tel: 46-8-5090-4654 UK - Wokingham Tel: 44-118-921-5800 Fax: 44-118-921-5820