ANALOG DEVICES

Mini Tutorial
MT-234

One Technology Way • P.O. Box 9106 • Norwood, MA 02062-9106, U.S.A. • Tel: 781.329.4700 • Fax: 781.461.3113 • www.analog.com

Using the ADA2200 as a Time Domain Filter

by Gustavo Castro,
Analog Devices, Inc.

IN THIS MINI TUTORIAL

This mini tutorial briefly introduces the operation of the ADA2200 as a programmable filter.

INTRODUCTION

The ADA2200 is a very flexible device that can be used as a programmable filter by disabling the synchronous demodulator function. Different filter configurations, such as low-pass, band-pass, high-pass, and band-stop filters can be implemented on the programmable filter block shown in Figure 1 by writing to 23 different registers via the SPI port or with an EEPROM during power up/reset. In addition, the frequency location of the corners or center frequencies can be adjusted by varying the clock rate fed into CLKIN or by changing the value of the clock divider.

The filter is composed of two sections: the decimation filter and the programmable filter. The decimation filter has a fixed transfer function, and it helps to prevent aliasing into the programmable block. The programmable filter is an infinite impulse response (IIR) filter that samples the output of the decimation filter at ⅛ the sampling rate of the input, and updates its output at the same rate. The SYNCO terminal generates a synchronization pulse every time the output changes, and can be used to reconstruct the filtered signal synchronously at other discrete-time devices such as ADCs.

Because the filters in the ADA2200 are determined by a well-defined set of capacitor arrays, the transfer function characteristics are repeatable, the filter location can be manipulated with the input clock, and its operation requires very low power consumption.

The goal of this tutorial is to briefly introduce the operation of the ADA2200 as a programmable filter, and to provide the user with the register contents to define the filters shown in Table 1.

Table 1.

Filter Type	Order, Q	Corner/Center Frequency (Hz)[1]	Pass-Band Gain (dB)
Band-Pass (BP1)	2nd, Q = 8.4	$f_{SI}/32$	0
Band-Pass (BP2)	2nd, Q = 4.3	$f_{SI}/32$	0
Low-Pass (LP1)	4th	$f_{SI}/40$	0
Low-Pass (LP2)	4th	$f_{SI}/64$	0
Notch	1st	$f_{SI}/32$	0

[1] For example, if $f_{SI} = f_{CLK}$ = 500 kHz, the corner frequency is $f_{SI}/32$ = 15.625 kHz.

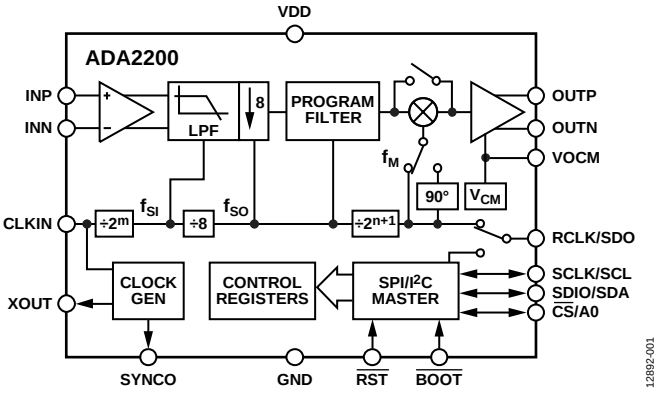

Figure 1. ADA2200 Simplified Block Diagram

DECIMATION FILTER

The decimation filter helps prevent aliasing from the input and simplifies the input antialiasing requirements. As shown in Figure 2, without the decimation filter, all images above the Nyquist frequency ($f_{SO}/2$) alias into the pass band. For example, the output for an input signal of frequency, f, would be identical to that of an input signal of frequency, $f + f_{SO}$, and it is impossible to distinguish between the two by simply looking at the output.

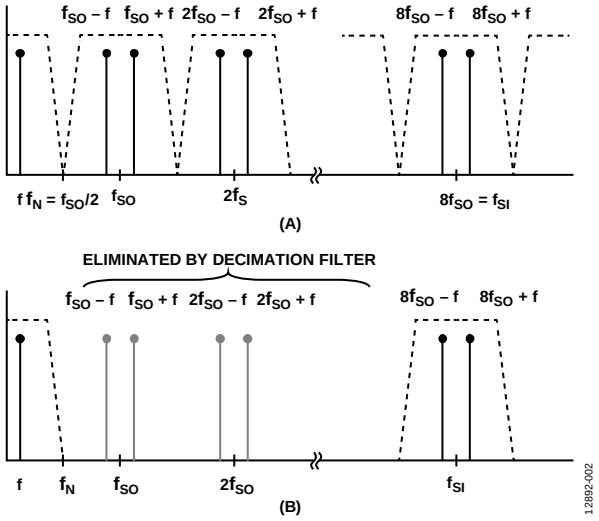

Figure 2. Aliasing Rejection with Decimation Filter

The decimation filter removes all the images above $f_{SO}/2$ and below $f_{SI} - f_{SO}/2$. If large signals are expected to fall in this higher frequency band, it may be necessary to add an antialiasing filter before the input of the ADA2200. If the preceding stages are limited in frequency content, a simple RC network can be used to add additional aliasing rejection.

The transfer function for the decimation filter is shown in Figure 3. As it can be observed, significant attenuation is introduced for signal frequencies above $f_{SO}/4$ (half of Nyquist), and it introduces a linear phase shift. This roll-off and phase shift need to be considered in the design of programmable filters.

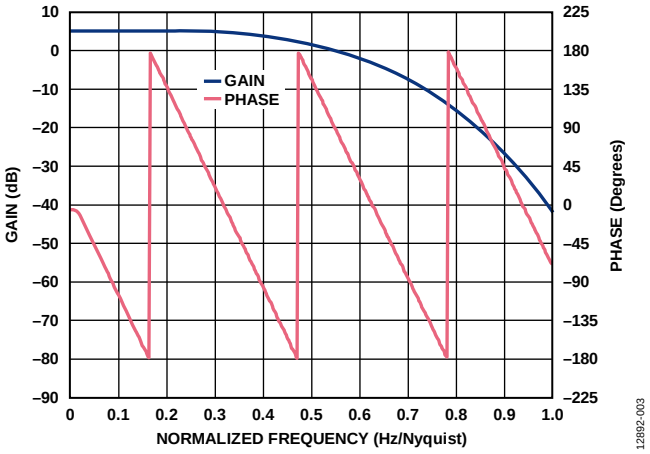

Figure 3. Transfer Function of the Decimation Filter

Disabling the Decimation Filter

It is possible to disable the decimation filter; however, that does not eliminate the decimation action. When the decimation filter is disabled, the input sampling occurs at the same rate f_{SI}, and the programmable filter block continues to sample the output of the decimator block at ⅛ of the input rate. Therefore, the maximum operation frequency of the IIR continues to be $f_{SI}/8$.

What is accomplished by disabling the decimator is eliminating the decimator roll-off and the phase shift introduced by it. This may be desired if the additional antialiasing is not required.

Disable the decimator by writing a Logic 1 to Register 0x027 Bit 6.

IIR PROGRAMMABLE FILTER

An IIR filter is a recursive filter. Recursion is analogous to feedback in a continuous time filter. For this reason, it is possible to define filters with only a few coefficients, but it is also possible to create unstable filters. By default, the IIR block is configured as a band-pass filter with a center frequency at $f_{SO}/8$ ($f_{SI}/64$). Figure 4 shows the transfer function for the default filter, including the effect of the decimation filter.

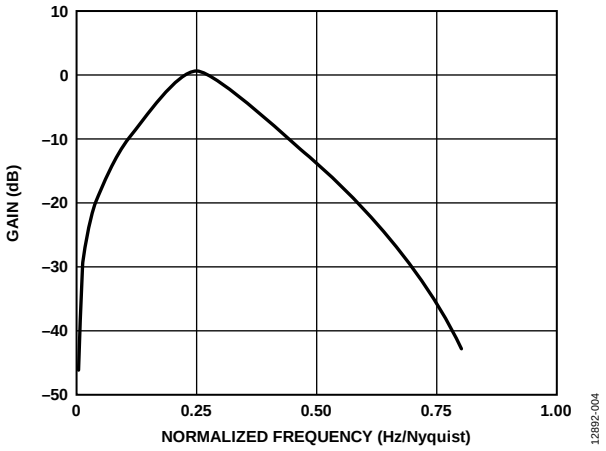

Figure 4. Transfer Function of the Default Band-Pass Filter

To modify the IIR filter characteristics, it is necessary to modify the default filter definition, which is stored in the internal memory between Register 0x0011 through Register 0x0027. Note that while the content of these registers define the filter characteristics, they do not represent the actual filter coefficients, but the internal configuration of the ADA2200 that generates the coefficients that represent the desired filter. For detailed instructions on how to access these registers and change their contents, refer to the Programming IIR Filters section.

FREQUENCY SCALING

Because the ADA2200 is a sampled analog technology device, its frequency characteristics directly depend on the clock frequency. Therefore, parameters such as corner frequency, the center of the pass band, or the location of the Nyquist frequency depend on the frequency of the clock used to drive the device. A natural consequence of this fundamental property is that it is possible to scale these filters by changing the clock frequency or using the on-chip clock dividers. This allows the user to perform frequency sweeps, lock the filter to a source for coherent filtering, or change the corner frequency without having to reprogram the device.

For example, the corner frequency for a 4th-order low-pass filter with a corner frequency at $f_C = 0.4\ f_N$ can be calculated by

$$f_C = 0.4 \times 0.5\, f_{SO} = 0.4 \times 0.5 \times 0.125\, f_{SI} = 0.025 \times 2^{-m}\, f_{CLK}$$

where:

m is the divider of the clock frequency and can take values of 1, 2, or 8.

f_{CLK} is the frequency of the clock signal applied to CLKIN. If f_{CLK} = 500 kHz and m = 1, the corner will be located at 12.5 kHz, f_{SO} = 62.5kHz and f_N = 31.25 kHz. With all the parameters remaining the same, reducing the clock frequency to 100 kHz would move the corner frequency down to 2.5 kHz.

Note that band-pass or band-stop filters will remain with a constant Q. This means that if the frequency doubles, the width of the pass band will double as well. In the case of the notch, the stop band doubles. This effect is shown in Figure 5.

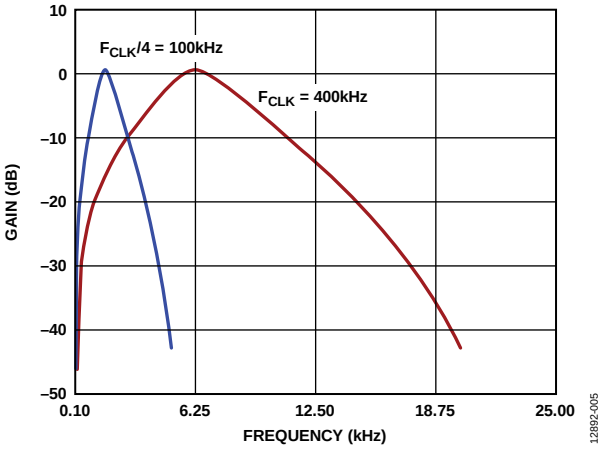

Figure 5. Effect of Scaling the Clock Frequency on a Band-Pass Filter

An additional advantage that comes with reducing the clock frequency is that it lowers the power consumption, thus the system scales according to the bandwidth of interest.

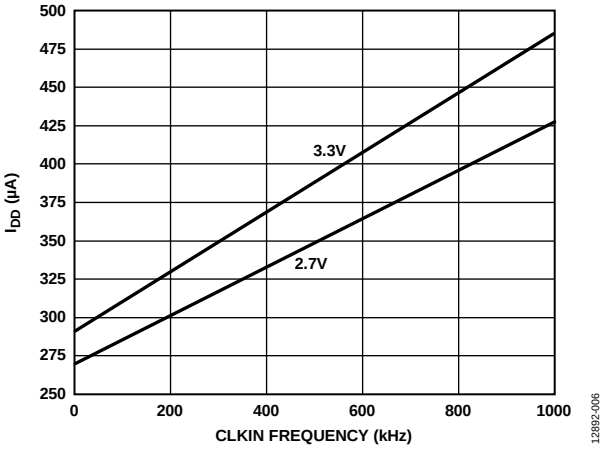

Figure 6. Typical Current Draw vs. CLKIN Frequency at V_{DD} = 2.7 V and 3.3 V

COHERENT FILTERING AND THE REFERENCE CLOCK

Certain applications, especially those involving pass-band filters or notches, require that the signal of interest (or the signal being rejected) fall precisely at the center frequency of the filter. Any mismatch between the signal and the center frequency (due to design tolerances, time and temperature drift, and uncertainty on the interference source), can cause undesired attenuation and phase errors in band-pass applications. In the case of a notch filter (or band stop), these mismatches limit the effectiveness of the interference rejection, making it very difficult to completely eliminate undesired signals. This is shown in Figure 7 for two different notch filters. A narrow notch is desired to avoid affecting adjacent frequencies, but its effectiveness is much less than that of the wider notch if the undesired signal does not match the center.

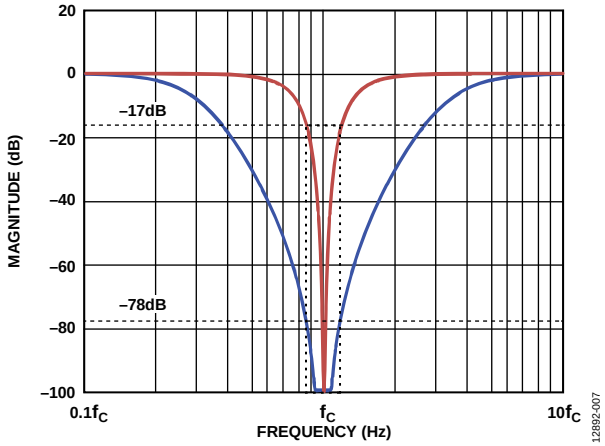

Figure 7. Mismatch Between the Notch Frequency and the Unwanted Signal Limits Band Stop Effectiveness

In cases where the signal of interest is known, it can be used to generate a clock signal to drive the ADA2200. As an alternative, the reference clock can be used to generate a time base for the signal of interest, synchronization, or to be the actual signal required as an excitation source, for example. The synchronization between the ADA2200 and the signal in question guarantees that it will always fall at the center frequency of the filter.

RECONSTRUCTION OF THE OUTPUT SIGNAL

The output voltage present across the OUTP and OUTN terminals of ADA2200 is equivalent to the voltage across a parallel capacitor combination, and therefore, this value remains constant until the next update cycle. For this reason, the output signal is formed by a sequence of discrete-time steps, as shown in Figure 8.

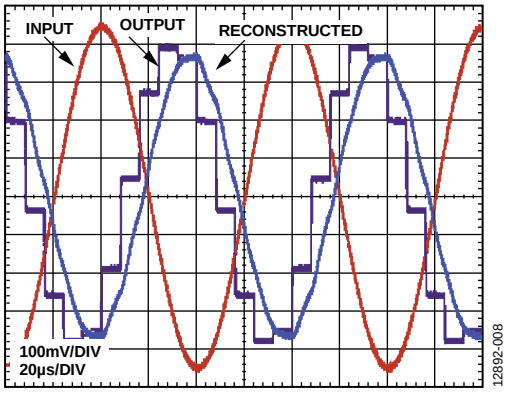

Figure 8. Input, Output, and Reconstructed Waveforms

In a continuous time system, the step sequence generates images at frequencies multiples of the output update rate. This effect can be observed in Figure 8, where the vertical lines are a consequence of the finite bandwidth of the output (the pin driver is a continuous-time device). The high-frequency image can be visualized by tracing a sine wave across the output steps and then subtracting it. The resulting saw-tooth waveform is a multiple of the fundamental frequency, and it is a typical and undesired artifact. To reduce undesired images in continuous time systems, a reconstruction filter is required at the output, as shown in Figure 9.

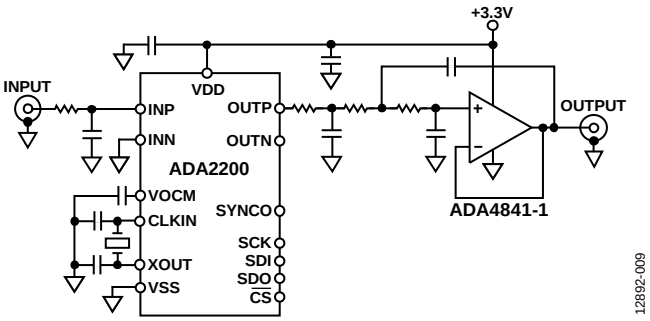

Figure 9. Adding a Second-Order Reconstruction Filter

When interfacing to discrete time systems (such as SAR ADCs), a reconstruction filter is not required if the sampler of the following device is synchronized with the output. For example, if a SAR ADC takes only one sample for every output update of the ADA2200, the signal does not need to be reconstructed and all the high-frequency images will disappear; the ADC will not know they ever existed.

The ADA2200 generates an output pulse through the SYNCO terminal, which can be used by a microprocessor or directly by an ADC to initiate an analog-to-digital conversion of the ADA2200 output. The SYNCO signal ensures that the ADC sampling occurs at an optimal time during the ADA2200 output sample window by generating a pulse with a frequency equal to the output update rate, f_{SO}. It is possible to configure the pulse polarity, and to adjust 1 of 16 different timing delays, as shown in Figure 10. The timing delays are spaced at ½ f_{SI} clock cycle intervals and span the full output sample window. Choosing the proper delay relative to the output update ensures a properly settled signal and avoids feed-through errors in the ADC. The configuration settings for the SYNCO pulse such as polarity and delay are contained in Register 0x0029.

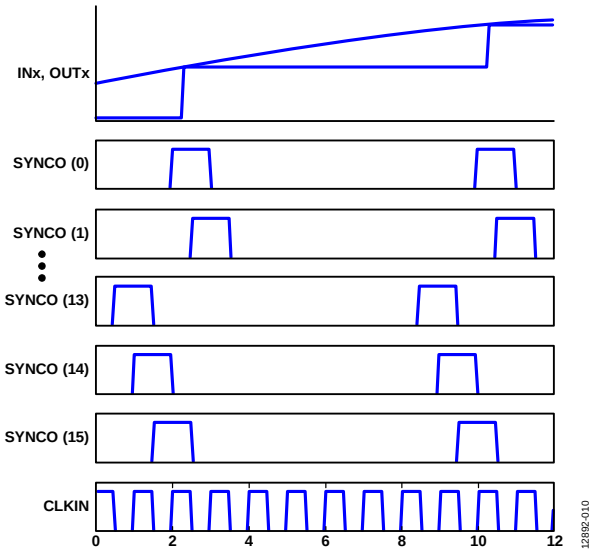

Figure 10. SYNCO Output Timing Relative to OUTP/OUTN, INP/INN, and CLKIN

FILTERING SIGNALS ABOVE NYQUIST

The maximum recommended input sampling frequency for the ADA2200 is 1 MHz (equivalent to the clock frequency when the clock divider is set to 1), which would theoretically limit its bandwidth to $f_{SO}/16$ or 62.5 kHz. Because the input signal bandwidth of the ADA2200 is 4 MHz, the ADA2200 will accept input signals up to this frequency. However, any of such signals will be down converted at the output, limited by $f_{SO}/2$ (the Nyquist frequency of the output update rate). In other words, for frequencies above Nyquist, the ADA2200 acts as a filter and a downconversion mixer by taking advantage of the sampling nature of the input.

For example, let us assume the ADA2200 samples the input at $f_{SI} = f_{CLK} = 500$ kHz, and the filter is configured for a center frequency at 7.8 kHz and pass band from 6 kHz to 10 kHz. If the decimator filter is enabled, a filter with 4 kHz bandwidth will appear around the center frequencies such as 492.2 kHz, 507.8 kHz, 992.2 kHz, 1.0078 MHz, 1.4922 MHz. Therefore, if a signal with a frequency of 3.992 MHz appears at the input of the ADA2200, it will appear as a 8 kHz signal at the output. The equivalent Q of the filter image with a center frequency at 3.9922 MHz for this example will be almost 1000.

It is important to note that in all these cases, there is always an adjacent filter image below and above any multiple of f_{SI}, therefore the designer must consider possible aliasing. In addition, the higher the frequency, the higher the Q that the antialiasing filter requires.

PROGRAMMING IIR FILTERS

To program the filter via the SPI port, initiate a write cycle to transfer the new filter definition to the ADA2200 registers from Address 0x0011 to Address 0x0027. For a list of filter options and their corresponding register values, see Table 2. After loading the filter values, write 0x03 to Register 0x0010 to configure the ADA2200 with the new filter. For more information for interfacing via the SPI port, refer to the ADA2200 data sheet.

To program the ADA2200 via an external EEPROM during boot or after a reset operation, it is necessary to preload the EEPROM with the memory configuration between Address 0x0011 to Address 0x002B, which includes the filter definition plus additional configuration registers. For the default memory contents and predefined filter definitions, refer to Table 2. For detailed instructions on how to program the ADA2200, refer to the Device Configuration section in the ADA2200 data sheet.

The filter measurements in this tutorial were performed with the ADA2200-EVALZ, by rewriting the contents of the on-board EEPROM with an EEPROM programmer. The EEPROM contents were the same as in Table 2 for each configuration, starting at Offset 0x00. Loading the programmed EEPROM on X1 and setting the EEPROM_BOOT switch automatically loads the new filter after applying power to the board.

The test setup is shown in Figure 11. The board can get power from any active USB port. All the filters are relative to the 400 kHz on-board oscillator. The AD8476-EVALZ and the EVAL-INAMP-82RZ with the AD8429 coupled the single-ended terminals on the network analyzer to the ADA2200. For additional details and circuit schematics on the ADA2200-EVALZ board, refer to the ADA2200-EVALZ user guide.

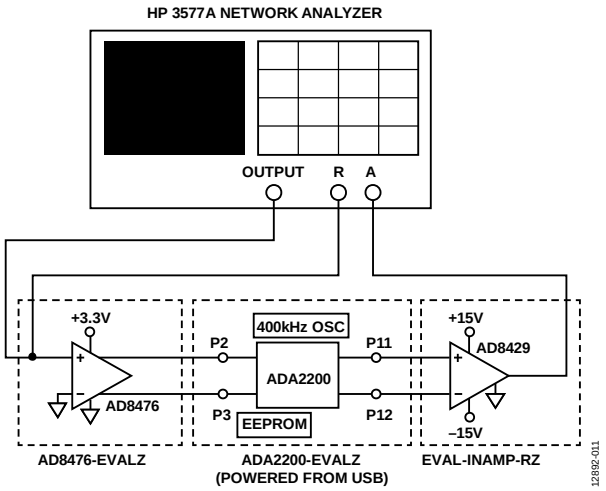

Figure 11. Test Setup Using ADA2200-EVALZ

The recommended register contents to configure the ADA2200 as a time domain filter are shown in Table 2.

Table 2. Recommended Register Contents for Different Filters

ADA2200 Address	Register Name	BP1	BP2	LP1	LP2	Notch	Default	All-Pass	External EEPROM Address[1]
0x0011		0xC0	0xC0	0x52	0x51	0xC0	0xC0	0x00	0x000
0x0012		0x0F	0x0F	0xAE	0x80	0x4F	0x0F	0xA0	0x001
0x0013		0x36	0xFA	0x52	0x40	0x84	0x1D	0xC0	0x002
0x0014		0xD1	0xD5	0xA6	0x80	0x9B	0xD7	0x0F	0x003
0x0015		0xC0	0xC0	0x52	0x51	0xC0	0xC0	0xC0	0x004
0x0016		0x0F	0x0F	0xAE	0x80	0x0F	0x0F	0x0F	0x005
0x0017		0x07	0x15	0x74	0x56	0xC0	0xC0	0xC0	0x006
0x0018		0x80	0x92	0x81	0x10	0x0F	0x0F	0x0F	0x007
0x0019		0x07	0x15	0x74	0x56	0x84	0x1D	0xC0	0x008
0x001A		0x80	0x92	0x81	0x10	0x9B	0x97	0x0F	0x009
0x001B	Filter	0x00	0x00	0x4E	0xC8	0x36	0x7E	0xC0	0x00A
0x001C	configuration	0x20	0x20	0x9D	0xA0	0x14	0x88	0x0F	0x00B
0x001D		0xC0	0xC0	0x22	0x97	0xC0	0xC0	0xC0	0x00C
0x001E		0x4F	0x4F	0x53	0xD9	0x0F	0x0F	0x0F	0x00D
0x001F		0xAA	0xB2	0x4F	0xED	0xC0	0xC0	0xC0	0x00E
0x0020		0xAA	0x2F	0x80	0x12	0x0F	0x0F	0x0F	0x00F
0x0021		0xC0	0xC0	0xC0	0xC0	0xC0	0xC0	0xC0	0x010
0x0022		0x0F	0x0F	0x0F	0x0F	0x0F	0x0F	0x0F	0x011
0x0023		0xC0	0xC0	0xF1	0x00	0xC0	0x00	0xC0	0x012
0x0024		0x4F	0x4F	0xDE	0xE0	0x4F	0xE0	0x0F	0x013
0x0025		0x23	0x23	0x23	0x23	0x23	0x23	0x23	0x014
0x0026		0x02	0x02	0x02	0x02	0x02	0x02	0x02	0x015
0x0027		0x02	0x02	0x12	0x06	0x07	0x24	0x00	0x016
0x0028	Analog pin configuration	0x00	0x00	0x00	0x00	0x00	0x00	0x00	0x017
0x0029	Sync control	0x0D	0x0D	0x0D	0x0D	0x0D	0x2D	0x2D	0x018
0x002A	Demod control	0x08	0x08	0x08	0x08	0x08	0x08	0x08	0x019
0x002B	Clock configuration	0x02	0x02	0x02	0x02	0x02	0x02	0x02	0x01A

[1] The EEPROM is loaded on the X1 position in ADA2200-EVALZ.

The following plots show the transfer function characteristics for the filters on Table 2.

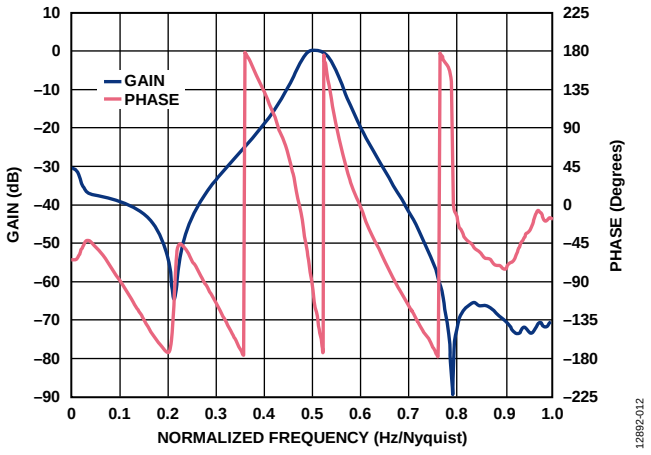

Figure 12. Transfer Function of the BP1 Filter

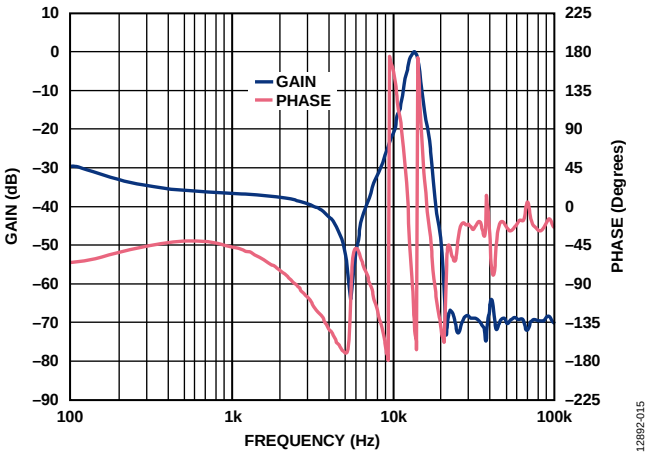

Figure 15. Transfer Function of the BP1 Filter, Logarithmic Sweep, $f_{CLKIN} = f_{SI} = 400$ kHz

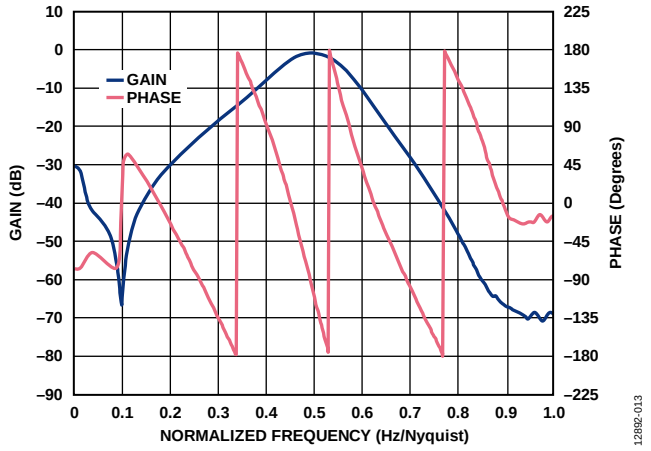

Figure 13. Transfer Function of the BP2 Filter

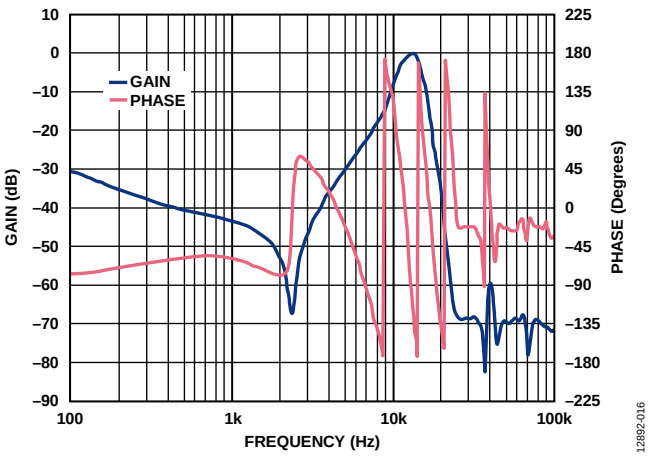

Figure 16. Transfer Function of the BP2 Filter, Logarithmic Sweep, $f_{CLKIN} = f_{SI} = 400$ kHz

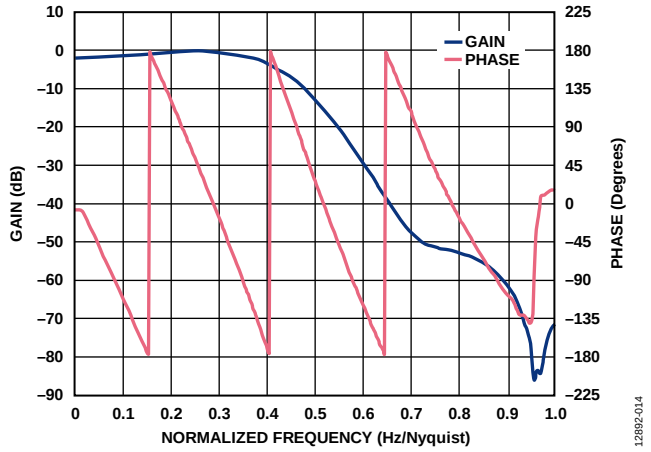

Figure 14. Transfer Function of the LP1 Filter

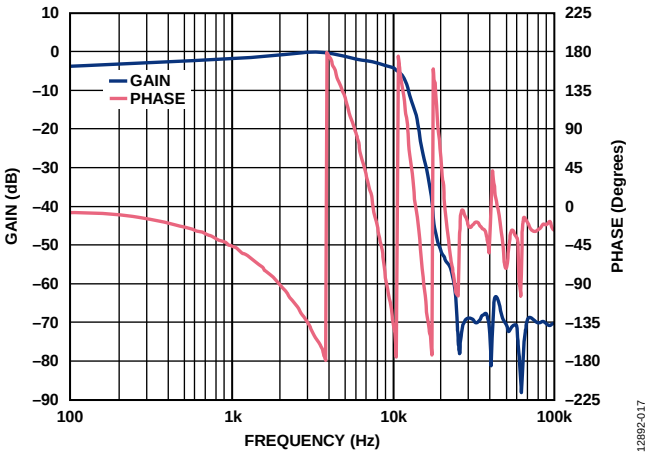

Figure 17. Transfer Function of the LP1 Filter, Logarithmic Sweep, $f_{CLKIN} = f_{SI} = 400$ kHz

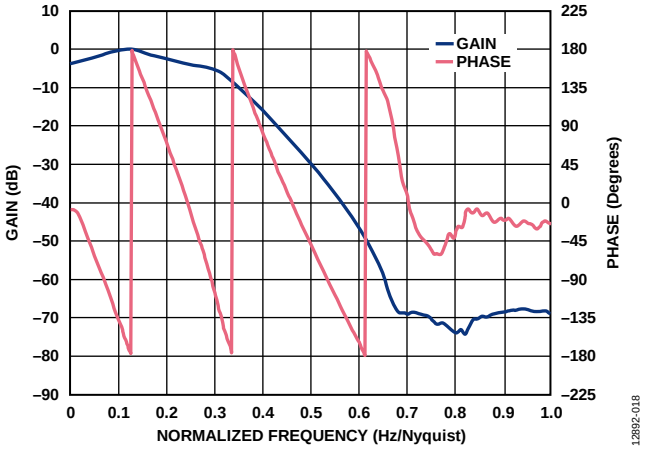

Figure 18. Transfer Function of the LP2 Filter

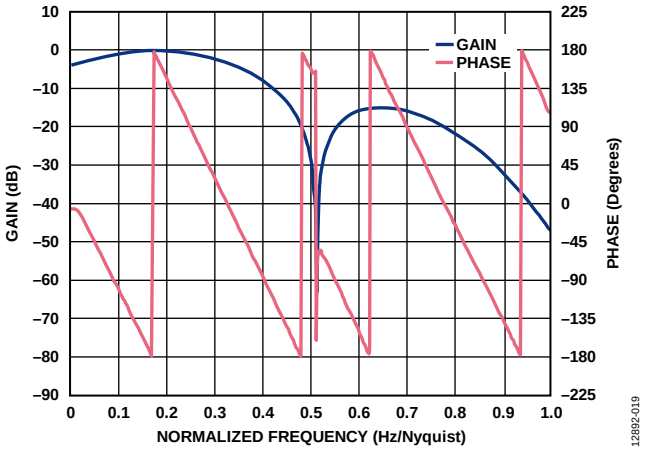

Figure 19. Transfer Function of the Notch Filter

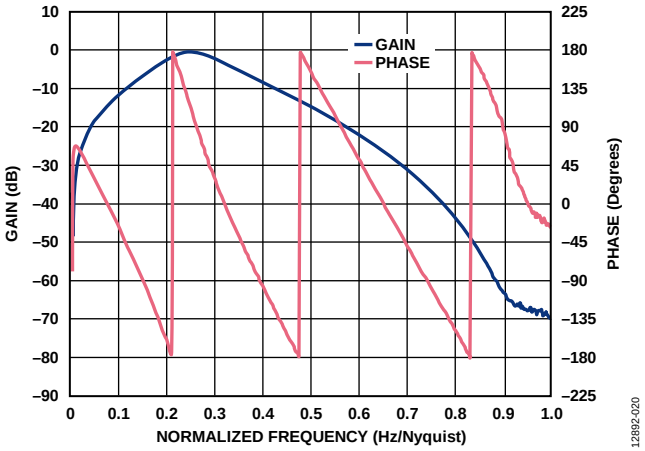

Figure 20. Transfer Function of the Default Filter

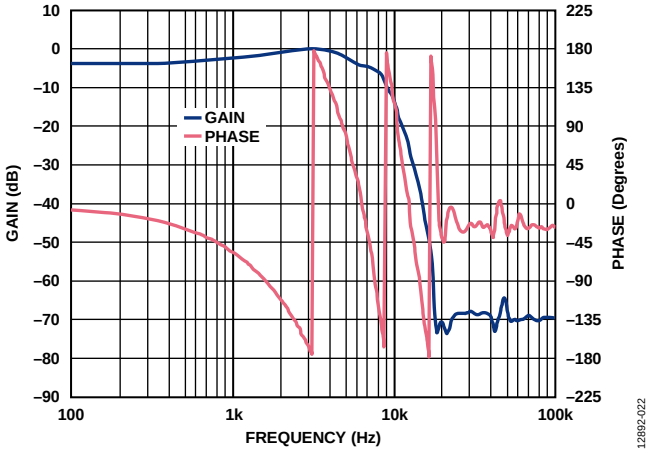

Figure 21. Transfer Function of the LP2 Filter, Logarithmic Sweep, $f_{CLKIN} = f_{SI} = 400$ kHz

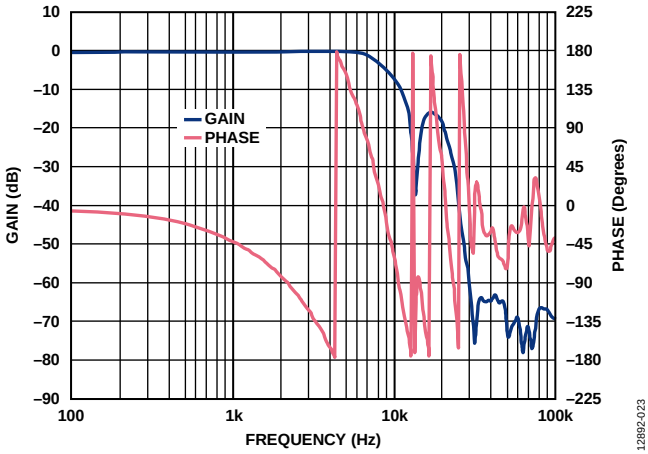

Figure 22. Transfer Function of the Notch Filter, Logarithmic Sweep, $f_{CLKIN} = f_{SI} = 400$ kHz

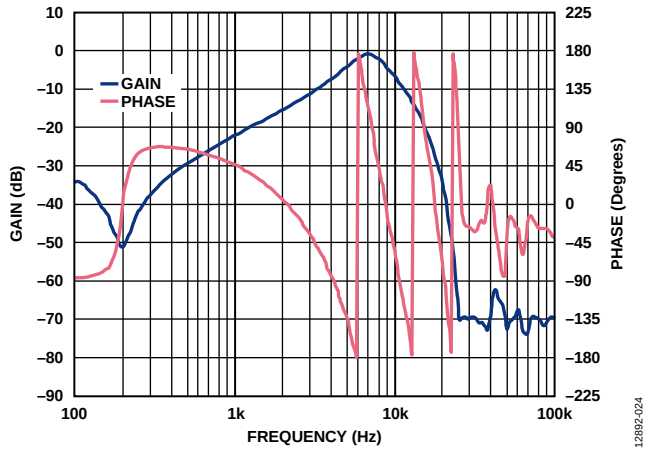

Figure 23. Transfer Function of the Default Filter, Logarithmic Sweep, $f_{CLKIN} = f_{SI} = 400$ kHz

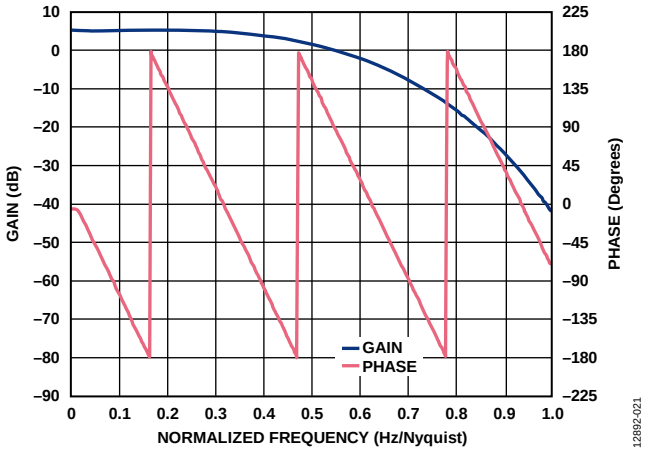

Figure 24. Transfer Function of the All-Pass Filter, Decimator Enabled

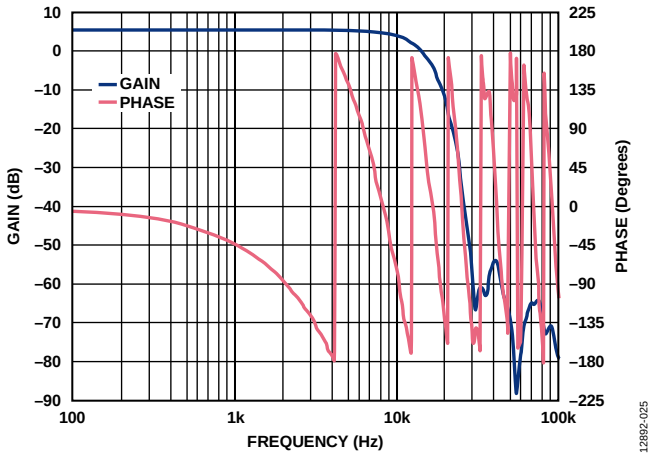

Figure 25. Transfer Function of the All-Pass Filter, Decimator Enabled, Logarithmic Sweep, $f_{CLKIN} = f_{SI} = 400$ kHz

REVISION HISTORY

12/14—Revision 0: Initial Version

MT12892-0-12/14(0)