

APPLICATION NOTE

EAGLE_16-125-VTG

AN2100



■ Date: December 7, 2020

■ Revision: 2.0

Review tracker

Revision	Date (MM/DD/YY)	Details of change	Writer
1.0	11/21/2016	Creation	BMA
2.0	09/10/2019	New graphic charter	ERO

Contents

1	Applicable documents	5
1.1	TECHWAY documents	5
2	Warning	6
3	Scope of document	6
4	Designing EAGLE_16-125 narrow band DDC's processing	7
4.1	DDC filtering specifications	7
4.1.1	First stage: CIC filter	7
4.1.1.1	CIC filter design	7
4.1.1.2	Decimation configuration	9
4.1.2	Second stage: cFIR filter	9
4.1.2.1	cFIR filter design	9
4.1.2.2	cFIR configuration	12
4.1.3	Third stage: pFIR filter	12
4.1.3.1	pFIR filter design	12
4.1.3.2	pFIR configuration	14
5	Complete DDC filter	15
6	DDC Filtering frequency response verification	16
6.1	EAGLE_16-125 configuration	16
6.1.1	Acquisition parameters	16
6.1.2	Filtering parameters	16
6.1.3	Network configuration	18
6.2	Processing results	18
7	APPENDIX	20
7.1	cFIR filter coefficient set coe file	20
7.2	pFIR filter coefficient set coe file	20
7.3	SNMP configuration script	21
7.4	MATLAB design script	23
8	Support information	28

Tables

Table 1: TECHWAY documents	5
Table 2: Central frequencies of the Processing Units' DDS	16
Table 3: Processing Units outputs' results	19

Figures

Figure 1: EAGLE_16-125 narrow band DDC	6
Figure 2: Filter specifications	7
Figure 3: Gain Bode diagram of CIC filter	8
Figure 4: Pass-band droop detail of the CIC filter	8
Figure 5: DDC decimation configuration from the WEB interface	9
Figure 6: SNMP command for DDC decimation setting	9
Figure 7: Gain Bode diagram of cFIR filter	10
Figure 8: CIC pass-band droop compensation	11
Figure 9: Frequency response of CIC & cFIR combination	11
Figure 10: cFIR's coefficient set loading through the WEB interface	12
Figure 11: SNMP commands for cFIR coefficient set loading	12
Figure 12: pFIR filter gain Bode diagram	13
Figure 13: pFIR pass-band ripple	13
Figure 14: SNMP commands example for pFIR coefficient file loading	14
Figure 15: DDC filter gain Bode diagram	15
Figure 16: DDC filter pass-band ripple	15
Figure 17: Acquisition parameters	16
Figure 18: Processing Unit (PU) number 0 configuration	17
Figure 19: Configuration summary of the five processing units	17
Figure 20: Network configuration	18
Figure 21: Complex signal modulus spectrums from processing units' outputs	18

1 APPLICABLE DOCUMENTS

1.1 TECHWAY documents

Table 1: TECHWAY documents

Ref.	Title	Origin	Document ID
[T1]	EAGLE_16-125 user manual	TECHWAY	UM2100
[T2]	EAGLE_16-125 narrow band processing user manual	TECHWAY	UM2140
[T3]	EAGLE_16-125 narrow band user manual of WEB interface	TECHWAY	UM2110
[T4]	EAGLE_16-125 narrow band user manual of SNMP agent	TECHWAY	UM2120
[T5]	EAGLE_16-125 user manual of TEA API	TECHWAY	UM2130
[T6]	Document list	TECHWAY	UM2199

2 WARNING

INFORMATION AND FIGURES SUBJECT TO CHANGE WITHOUT PRIOR NOTICE.

USER ASSUMES ENTIRE RISK FOR THE USE OF THE PRODUCT.

TECHWAY SHALL NOT BE HELD LIABLE FOR ANY SYSTEM DAMAGE, DATA LOSS OR OTHER DAMAGES RESULTING FROM THE USE OR MISUSE OF THE PRODUCT.

3 SCOPE OF DOCUMENT

EAGLE_16-125 allows users to tune its DDC to fit a wide range of applications.

Details on EAGLE_16-125's narrow band processing can be found in EAGLE_16-125's narrow band processing user manual [T1].

When configured with narrow band baseline **BL_EAGLE_VERTIGO_FWx_SWy**, EAGLE_16-125's DDC looks like in *Figure 1*.

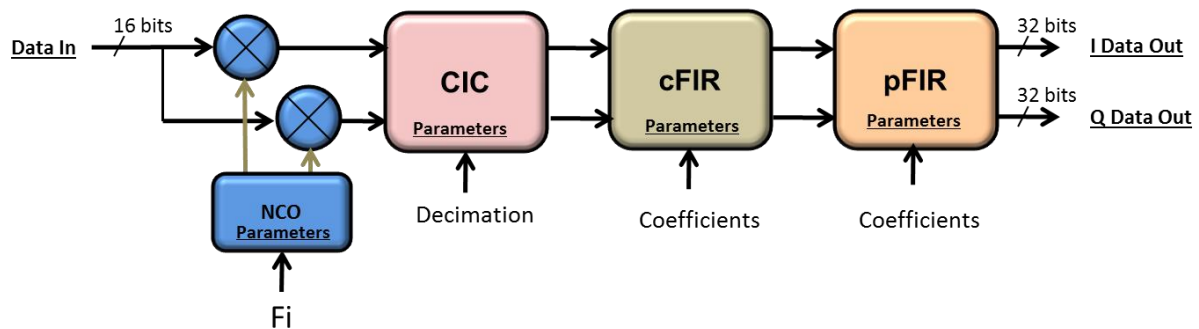


Figure 1: EAGLE_16-125 narrow band DDC

Here are the parameters which are customizable:

- DDS's central frequency
- CIC filter's decimation
- cFIR filter's frequency response
- pFIR filter's frequency response

This application note presents the filters' different roles and how to configure them to fit a specific need.

4 DESIGNING EAGLE_16-125 NARROW BAND DDC'S PROCESSING

4.1 DDC filtering specifications

Through this application note, a filter will be designed. This filter will have the following specifications:

- Sample frequency (f_s): 100 MHz
- Decimation: 420
- Pass-band ripple: 0.1 dB
- Cutoff frequency: 86 kHz
- Stop-band frequency: 120 kHz
- Stop-band attenuation: 100 dB

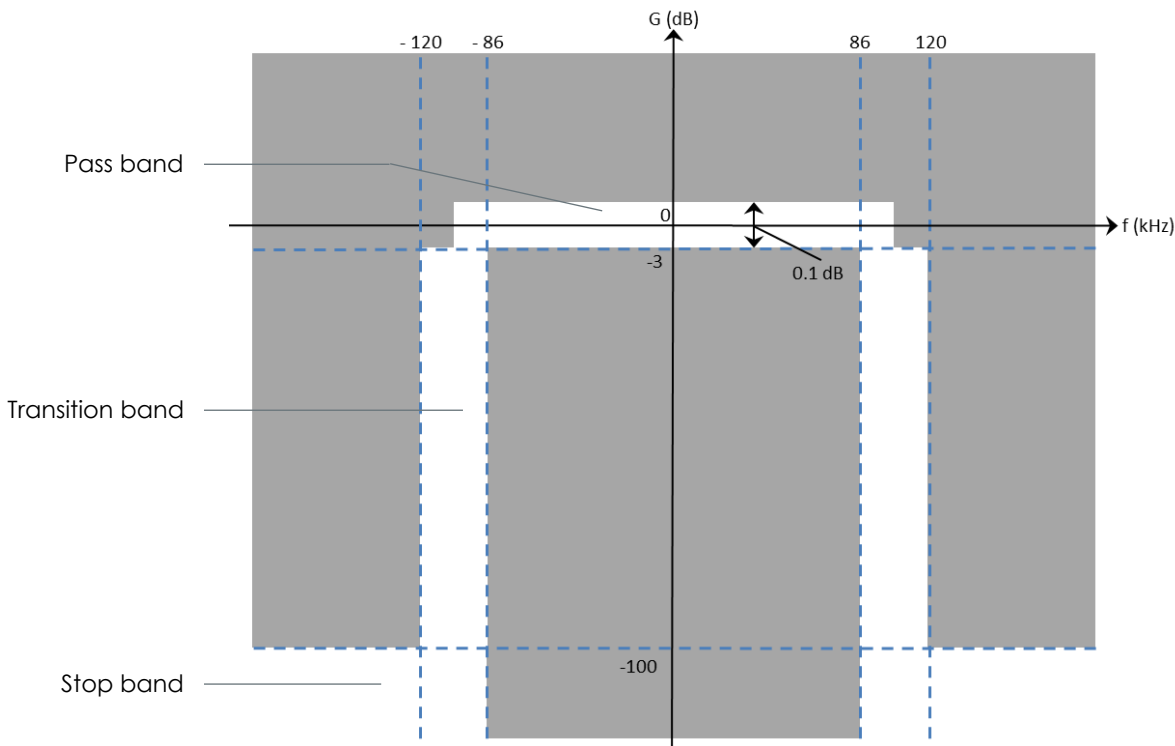


Figure 2: Filter specifications

4.1.1 First stage: CIC filter

4.1.1.1 CIC filter design

The CIC filter is the first part of the filtering stage of the DDC. Its role is to provide a low-pass-high-decimated frequency response.

The EAGLE_16-125 CIC filter has the following fixed parameters:

- Number of stages: 5
- Differential delay: 1

The only parameter that can be configured is the decimation, which can go from 8 to 256.

In our example, as decimation factor of 420 is required, and cFIR and pFIR filters both have a decimation of 2, then the CIC filter has to be configured with a decimation factor of 105.

With these parameters, the CIC filter has the following frequency response:

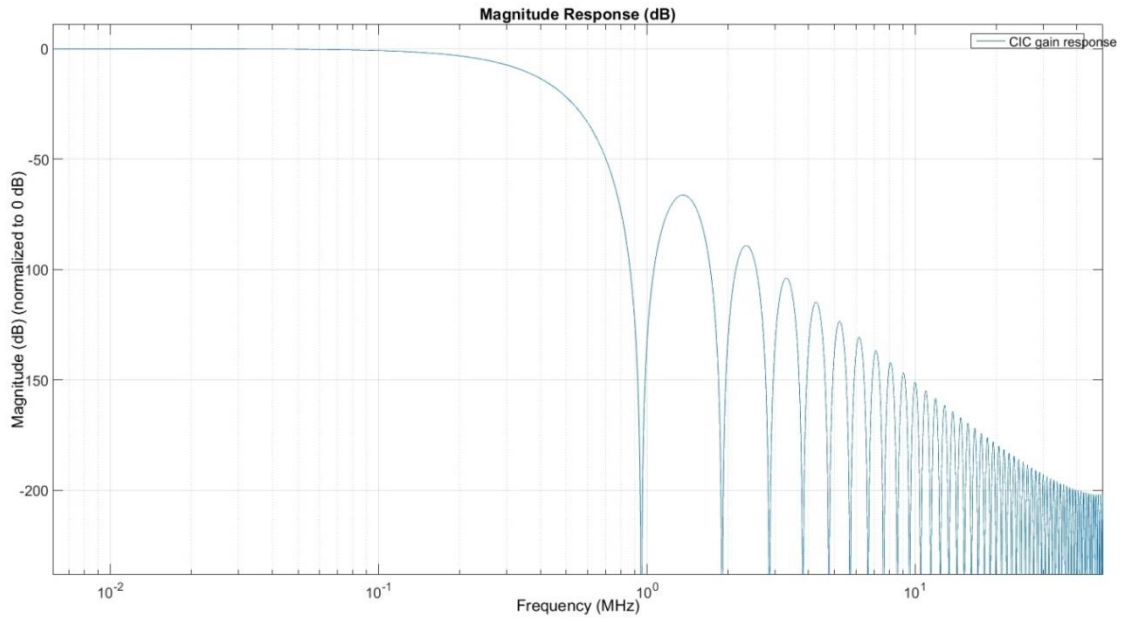


Figure 3: Gain Bode diagram of CIC filter

Although the CIC filter brings a high decimation and a low-pass frequency response, it has major drawbacks:

- First, the stop band ripple, which is above the -100 dB stop-band specification, will bring frequency aliases into the pass-band of the final filter if left as it is (Figure 3).
- The second drawback is the pass-band droop. As the CIC has not a flat pass-band, the droop will degrade the pass-band of the final filter (Figure 4).

To compensate these two points, the CIC filter is followed by a compensation filter: the cFIR filter.

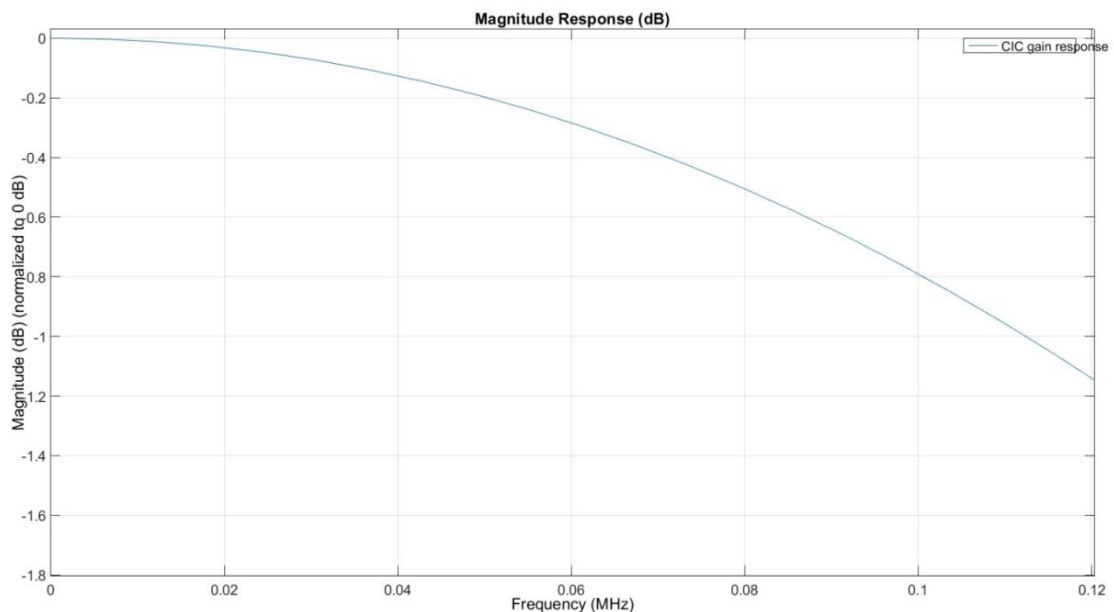


Figure 4: Pass-band droop detail of the CIC filter

4.1.1.2 Decimation configuration

User only has to configure the desired total decimation of the DDC and the EAGLE_16-125 will apply itself the correct decimation on the CIC filter. Configuration can be made either through the EAGLE_16-125 WEB interface (Figure 5) or via an SNMP command (Figure 6).

Channel linked	Central Frequency	Decimation	Peak Rate	Data rate
Channel 0	69.95 MHz	420	1.905 MB/s	0.08 MB/s

Input Channel selection

Channel 0 ?

Output Destination selection

Interface 0 ; IP 0: 10.1.1.4 ?
Configure network params. first

Filtering selection

Filter enabled ?

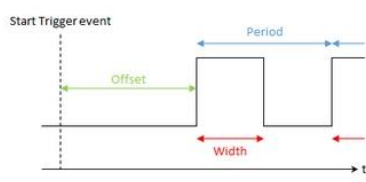
Input Windowing

Offset ? 0 samples

Width ? 4200000 samples

Period ? 4294967 samples

Set infinite period



DDS Central Frequency

69.95 MHz ?

Filtering DDC Decimation

420 ?

Figure 5: DDC decimation configuration from the WEB interface

```

echo "DDC decimation setting"
IP=192.168.0.253
pu_num=0
decimation=420
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.${pu_num}.7.0 i ${decimation}
    
```

Figure 6: SNMP command for DDC decimation setting

Note:

It is important to configure the decimation factor prior to configure windowing **width**. Decimation factor value is used by EAGLE_16-125 to calculate the real windowing width, which includes DDC filter's rise time, which depends on the decimation factor.

4.1.2 Second stage: cFIR filter

4.1.2.1 cFIR filter design

The cFIR filter's (for compensation Finite Impulse Response filter) main role is to compensate the pass-band droop of the CIC filter. It narrows the bandwidth and increases the overall attenuation within transition band and stop-band regions of the final filter as well.

It is implemented as a decimate-by-2 26-taps symmetric FIR filter. Transition band roll-off of the cFIR filter does not have to be really steep, as this task will be handled by the pFIR filter. Meanwhile, it is essential that the cFIR filter's stop band starts before half of the CIC filter's sampling frequency so as to avoid frequency aliasing.

In our case, the CIC filter's output sampling frequency is:

$$f_s/\text{CIC decimation factor} \Rightarrow 100\text{e}6/105 = 952\,380 \text{ Hz}$$

Then, cFIR filter's stop band shall start before half of that value: 476 190 Hz. A good practice is to choose quarter of CIC filter's output sampling frequency, so $(f_s/\text{CIC decimation factor})/4$.

In order to obtain a final pass-band as flat as possible, care must be taken to create a cFIR filter with a minimum pass-band ripple.

Here are the chosen parameters for this example cFIR:

- Cutoff frequency: $(f_s/105)/4 = 238 \text{ kHz}$
- Pass-band ripple: 0.001 dB
- Stop-band attenuation: 100 dB

With these parameters, the cFIR has the following frequency response:

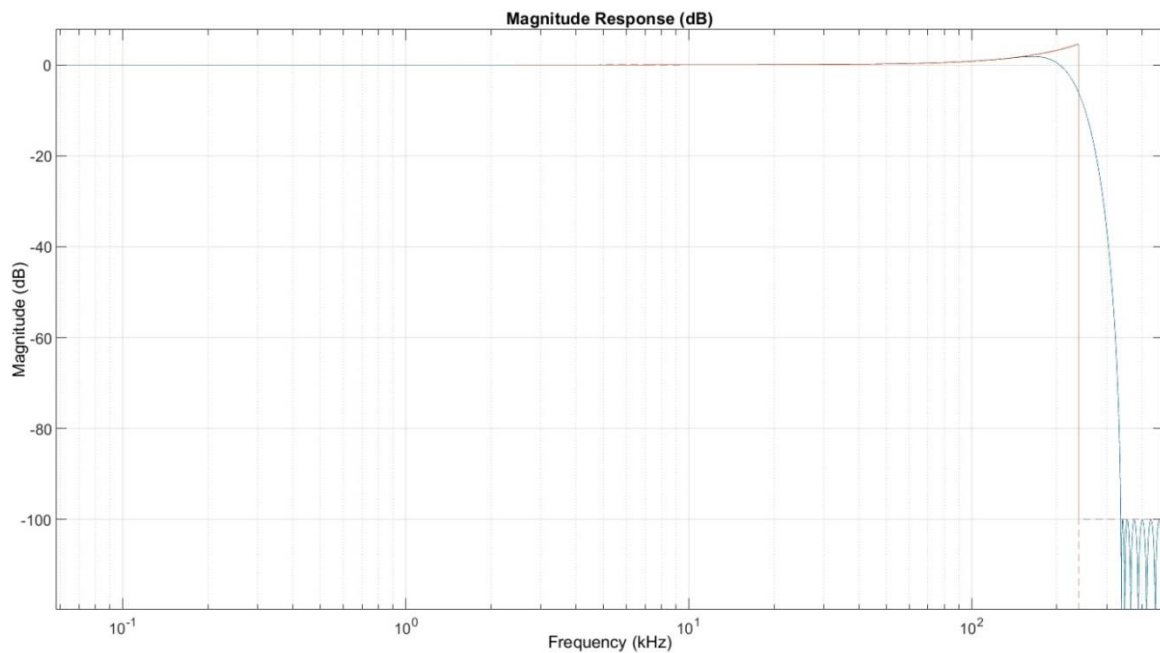


Figure 7: Gain Bode diagram of cFIR filter

The frequency response of the cFIR filter is determined by a set of coefficients which are used by filter's multipliers. Coefficient set must be provided by user. Some tools like Matlab© can help you to determine or to generate a set of coefficient by giving them the parameters determined above.

If we take a close look at the droop compensation (*Figure 8*), we can see that the pass-band is correctly flattened by the cFIR.

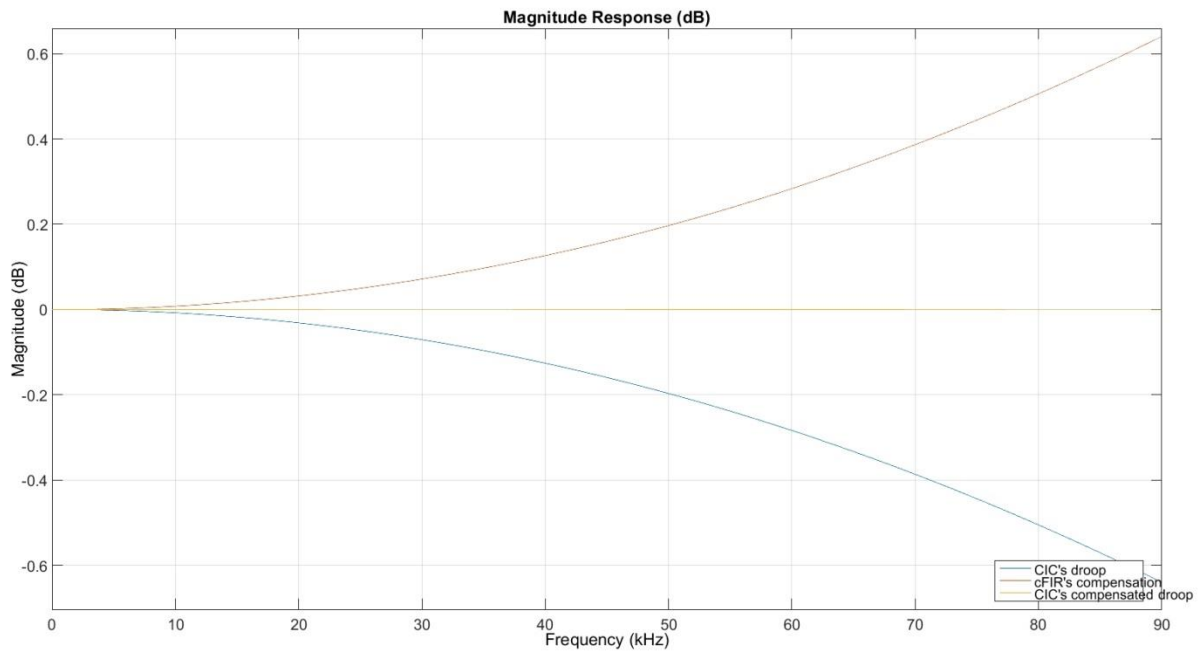


Figure 8: CIC pass-band droop compensation

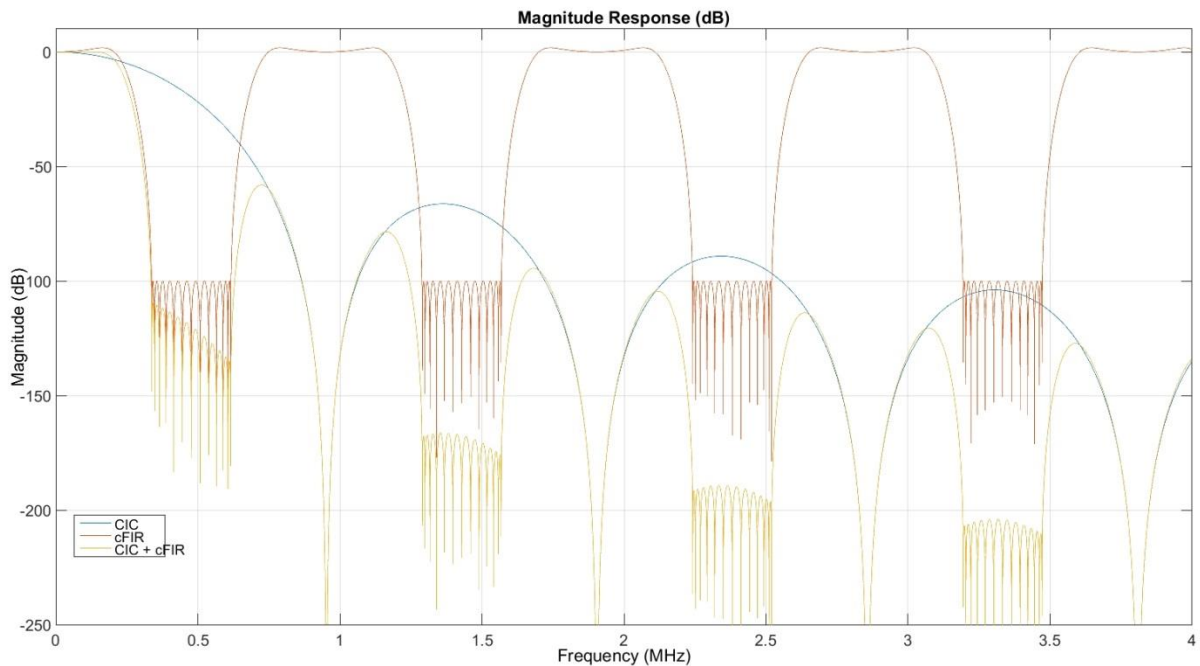


Figure 9: Frequency response of CIC & cFIR combination

Taking a look at the resulting frequency response of the combination of the CIC and cFIR filters (Figure 9), we can verify that the cFIR filter attenuates the CIC stop band ripple lobes, and narrows the pass-band of the final filter.

Still, the three remaining lobes above -100 dB will create aliases within final filter bandwidth. That point will have to be corrected by the next and final filtering stage: the pFIR filter.

4.1.2.2 cFIR configuration

Frequency response of cFIR filter must be adjusted by loading a custom coefficient set. This set can be loaded either through the WEB interface (Figure 10) or by an SNMP command (Figure 11). Coefficient sets have to be provided respecting Xilinx filter coefficient format with file extension ".coe" (see the APPENDIX section at the end of this document).

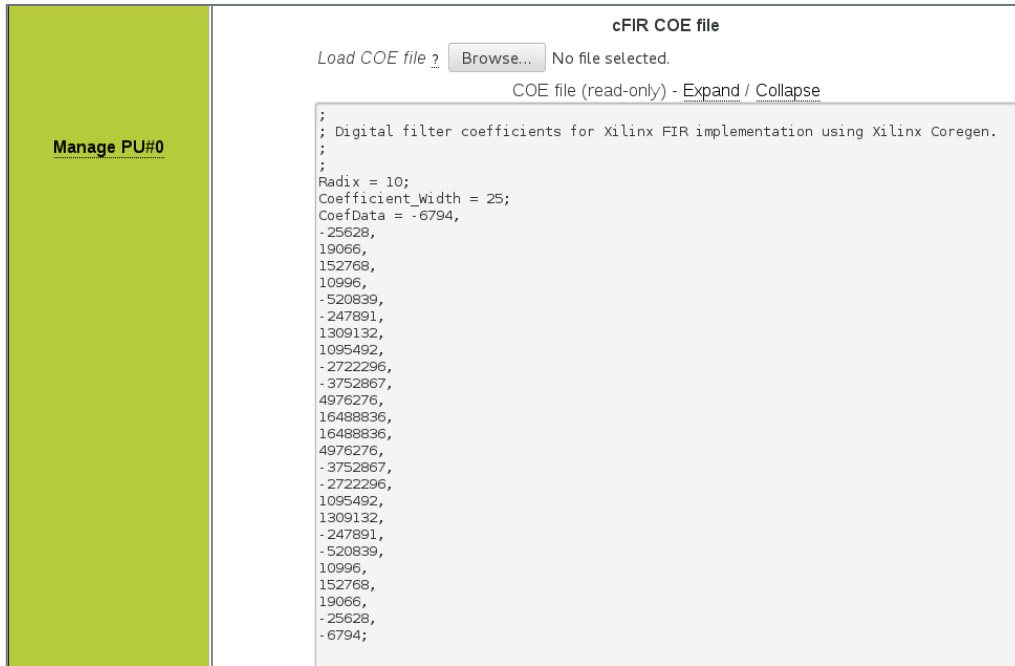


Figure 10: cFIR's coefficient set loading through the WEB interface

```

echo "cFIR coefficient loading"
IP=192.168.0.253
pu_num=0
coefile=$(cat ../EAGLE_fs_100_MHz_Decim_420_fc_86_kHz_cFIR.coe)
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.${pu_num}.8.0 s "$coefile"

```

Figure 11: SNMP commands for cFIR coefficient set loading

4.1.3 Third stage: pFIR filter

4.1.3.1 pFIR filter design

The role of the pFIR filter is to realize a filtering to obtain the requested pass-band cutoff and stop-band attenuation. It is implemented as a decimate-by-2 63-taps symmetric FIR filter, which allows implementation of filters with pretty steep transition band roll-off.

Here are the chosen parameters for this example pFIR:

- Cutoff frequency: 89.5 kHz
- Pass-band ripple: 0.1 dB
- Stop band attenuation: 100 dB

With these parameters, the pFIR has the following frequency response:

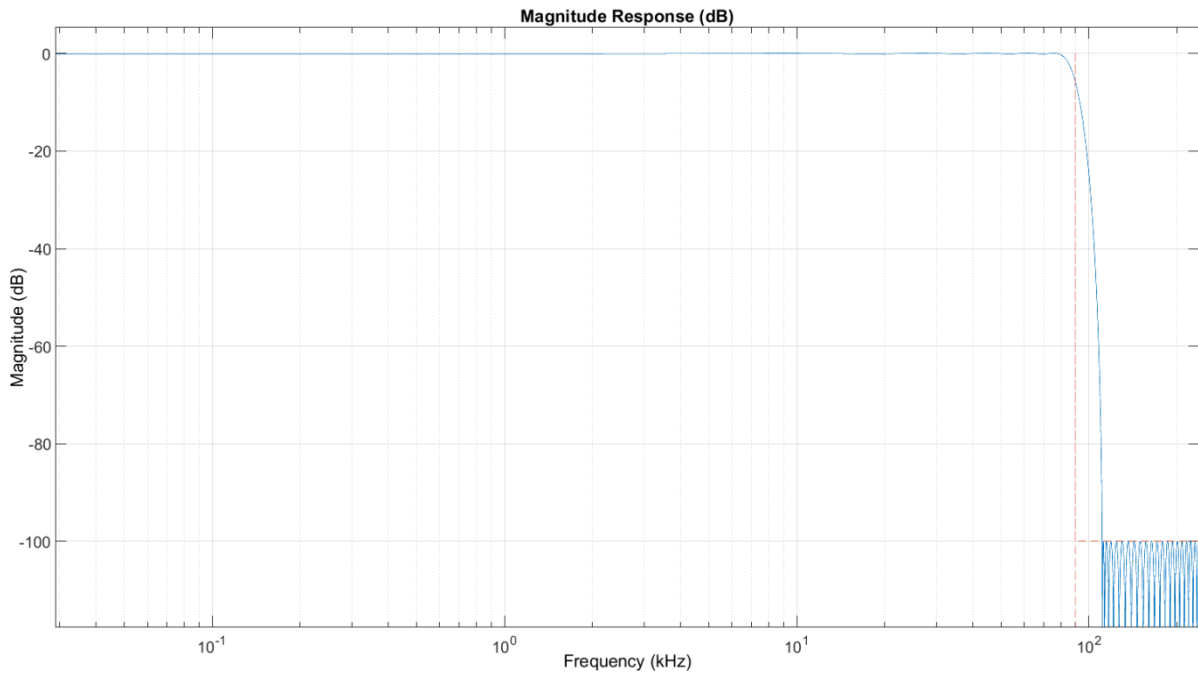


Figure 12: pFIR filter gain Bode diagram

The frequency response of the pFIR filter is determined by a set of coefficients which are used by filter's multipliers. Coefficient set must be provided by user. Some tools like Matlab® can help you to determine or to generate a set of coefficient by giving them the parameters determined above.

A closer look at the pass-band ripple of the pFIR (Figure 13) indicates, with the previous flat pass-band of CIC and cFIR combination, that the 0.1 dB ripple specification of the final filter will be fulfilled.

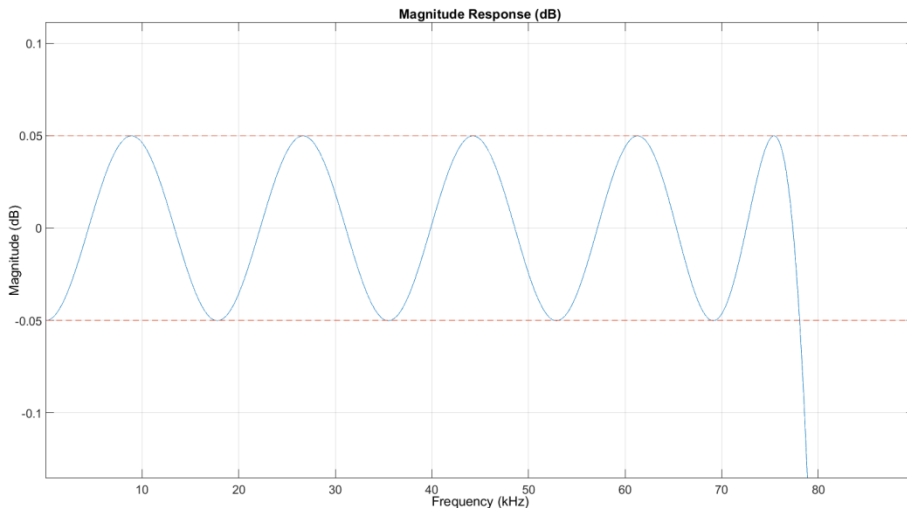


Figure 13: pFIR pass-band ripple

4.1.3.2 pFIR configuration

pFIR filter's frequency response can be adjusted by loading a custom coefficient set. This set can be loaded either through the WEB interface (in the exact same way than with the cFIR, see *Figure 10*) or by an SNMP command (*Figure 14*). Coefficient sets have to be provided respecting Xilinx filter coefficient format with file extension ".coe" (see the *APPENDIX* section at the end of this document).

```
echo "pFIR coefficient loading"  
IP=192.168.0.253  
pu_num=0  
coefile=$(cat ../EAGLE_fs_100_MHz_Decim_420_fc_86_kHz_pFIR.coe)  
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.${pu_num}.9.0 s "$coefile"
```

Figure 14: SNMP commands example for pFIR coefficient file loading

5 COMPLETE DDC FILTER

When combining CIC, cFIR & pFIR filters, the following gain bode diagram is obtained:

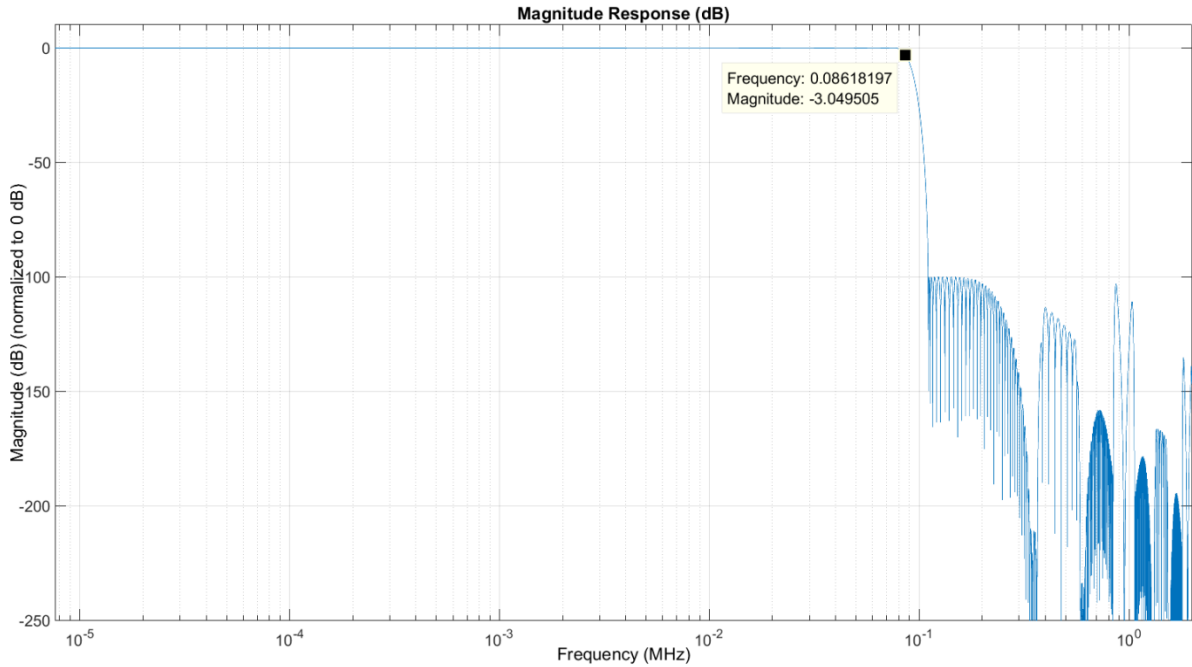


Figure 15: DDC filter gain Bode diagram

We can see that the resulting cutoff frequency is at 86 kHz, and that the stop-band attenuation is at -100 dB (Figure 15).

If we take a closer look at the pass-band (Figure 16), we can see that the 0.1 dB ripple requirement has been successfully met as well:

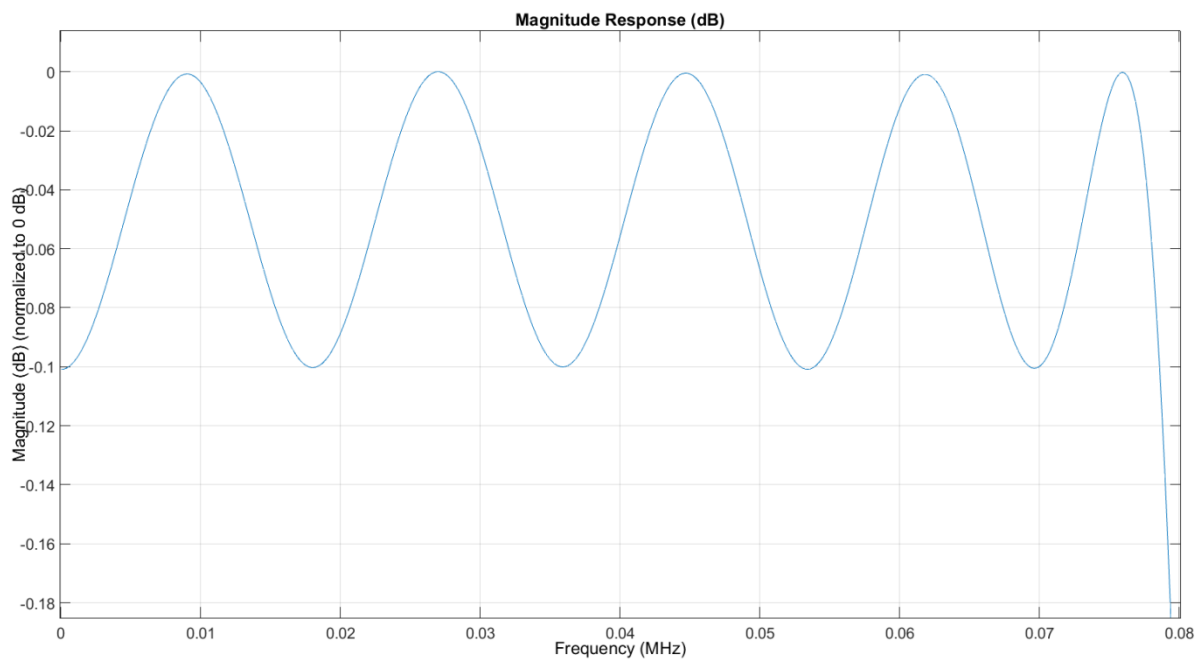


Figure 16: DDC filter pass-band ripple

6 DDC FILTERING FREQUENCY RESPONSE VERIFICATION

To verify the frequency response of the previously designed filter, the EAGLE_16-125 narrow band will be fed with a 70 MHz signal on analog input n°0. This input will be mapped to processing units 0 to 4 so as to each processing unit can treat a specific frequency range, each by using a different demodulation frequency. The sampling frequency will be 100 MHz.

We are going to see how to configure the EAGLE_16-125 to do this verification.

6.1 EAGLE_16-125 configuration

6.1.1 Acquisition parameters

Acquisition parameters consist in several configurations:

- sampling clock frequency
- trigger edge and sampling clock (internal or external)
- trigger electrical standard

Sampling Frequency ?	100000000 Hz	Current: 100.000849 MHz
Capture mode ?	Edge: capture at trigger RISING edge	
	Clock sampling source External clock	
Trigger electrical std. ?	LVPECL	

Figure 17: Acquisition parameters

Note:

It is important to configure the sampling clock frequency **prior** to configure the DDS frequencies (Figure 17). Sampling clock frequency is used by EAGLE_16-125 to calculate the configuration value which will be applied to the DDS.

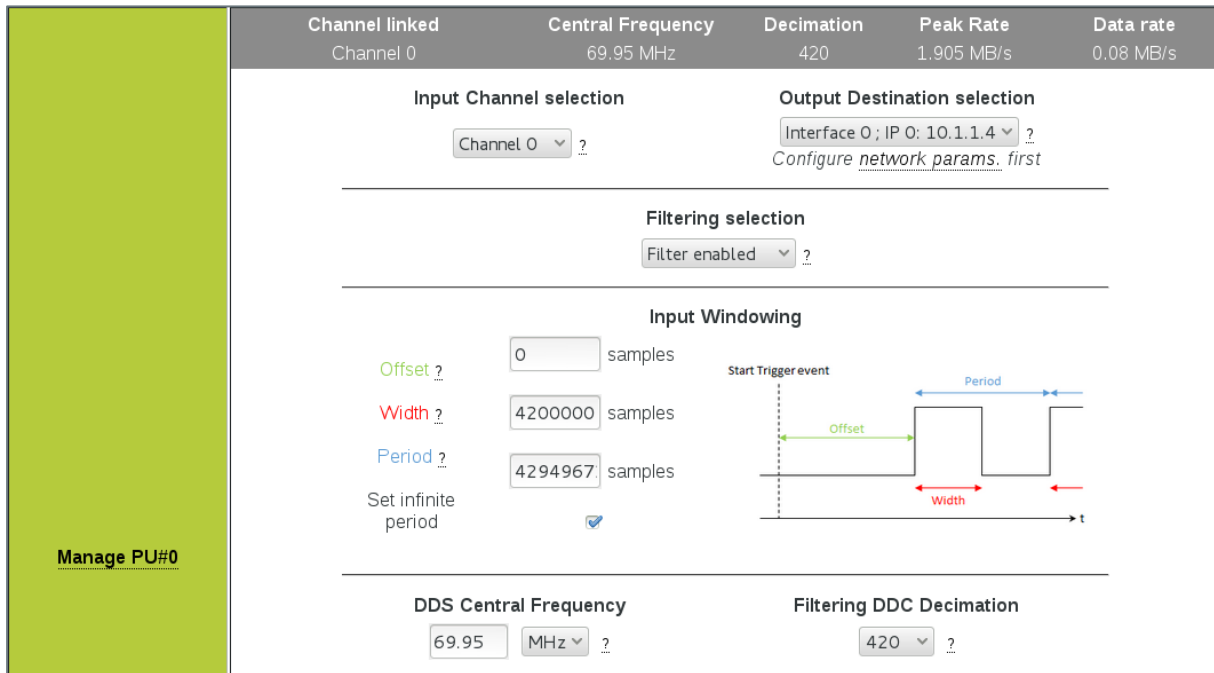
6.1.2 Filtering parameters

In order to verify the filter's frequency response in one recording session, the five processing units are used simultaneously. On the five processing units (PU [0...4]), central frequencies of the DDS are set to observe filter attenuation at different frequencies (Table 2).

Table 2: Central frequencies of the Processing Units' DDS

PU number	f_{DDS} (MHz)	Frequency obtained after demodulation (kHz)
0	69.95	50
1	69.914	86
2	69.9	100
3	69.885	115
4	69.8	200

In order to have a descent FFT, widths of the windowing on the five processing units are set to get 10 000 I/Q complex output samples. We do not use any offset, and the period is set to infinity (Figure 18).



Channel linked	Central Frequency	Decimation	Peak Rate	Data rate
Channel 0	69.95 MHz	420	1.905 MB/s	0.08 MB/s

Input Channel selection
Channel 0 ?

Output Destination selection
Interface 0 ; IP 0: 10.1.1.4 ?
Configure network params. first

Filtering selection
Filter enabled ?

Input Windowing

Offset ? 0 samples

Width ? 4200000 samples

Period ? 4294967 samples

Set infinite period

DDS Central Frequency
69.95 MHz ?

Filtering DDC Decimation
420 ?

Diagram: A graph showing a square wave pulse. The x-axis is labeled 't'. A vertical dashed line marks the 'Start Trigger event'. A green arrow labeled 'Offset' points from the trigger event to the start of the pulse. A red arrow labeled 'Width' points across the pulse. A blue double-headed arrow labeled 'Period' spans from the start of one pulse to the start of the next.

Figure 18: Processing Unit (PU) number 0 configuration

Once the five processing units configured, user can check the filtering configuration on the summary configuration page (Figure 19).

Processing Unit#	Summary configuration				
	Channel linked	Central Frequency	Decimation	Peak Rate	Data rate
Manage PU#0	Channel 0	69.95 MHz	420	1.905 MB/s	0.08 MB/s
Manage PU#1	Channel 0	69.914 MHz	420	1.905 MB/s	0.08 MB/s
Manage PU#2	Channel 0	69.9 MHz	420	1.905 MB/s	0.08 MB/s
Manage PU#3	Channel 0	69.885 MHz	420	1.905 MB/s	0.08 MB/s
Manage PU#4	Channel 0	69.8 MHz	420	1.905 MB/s	0.08 MB/s

Figure 19: Configuration summary of the five processing units

6.1.3 Network configuration

Do not forget to configure the network in order to correctly receive EAGLE_16-125 data.

IP and MAC Host Address							
IP Interface 0	IP	10	1	1	1		
	MAC	40 : D8 : 55 : 16 : B0 : 20					
Node#	IP/MAC Destination Addresses				Active	Delete	
#1	IP	10	1	1	4	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	MAC	00 : 00 : 00 : 00 : 00 : 00					
<i>Add new IP destination</i>							

Figure 20: Network configuration

6.2 Processing results

With these configurations applied, the outputs of the five processing are recorded, the complex I/Q modulus spectrums calculated and plotted. Here are the results:

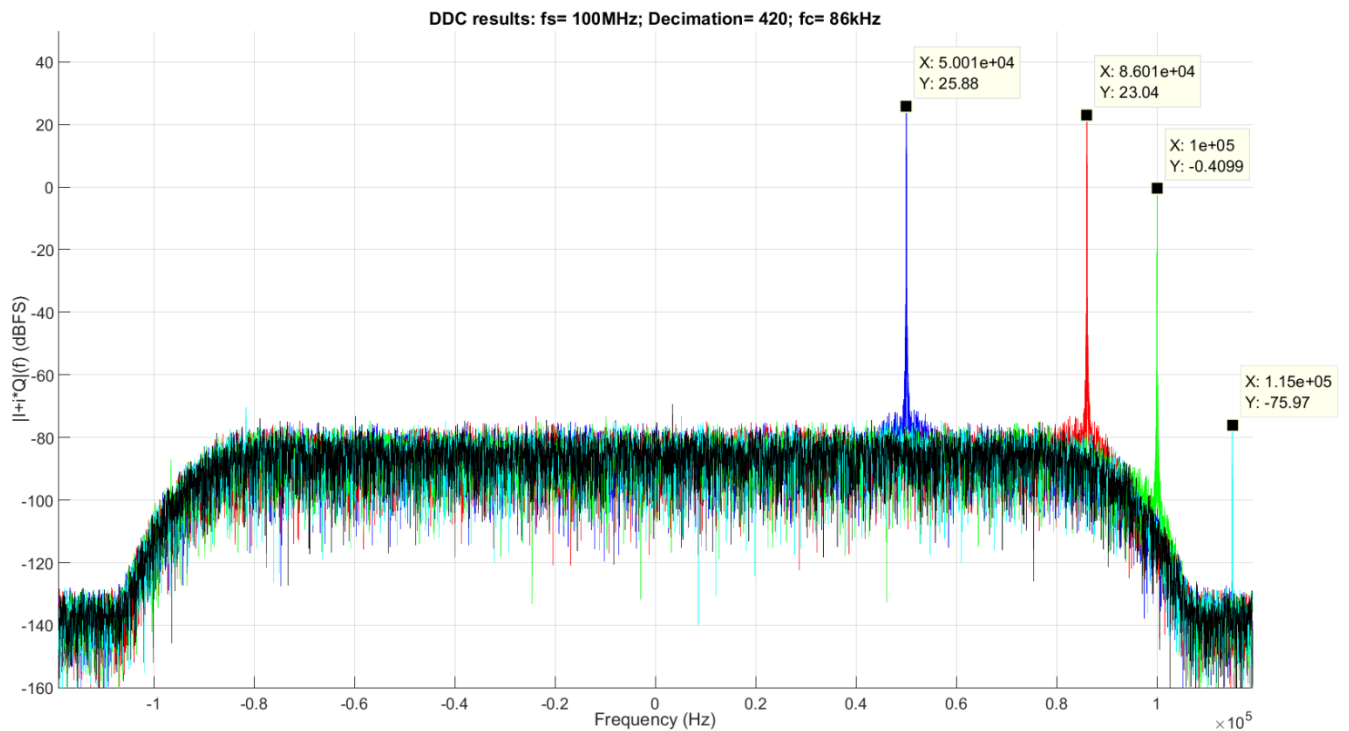


Figure 21: Complex signal modulus spectrums from processing units' outputs

Table 3 explains the Figure 21 results.

Table 3: Processing Units outputs' results

PU number	Plot color	Frequency (kHz)	Comments
0	Blue	50	Within filter's bandwidth
1	Red	86	Filter's cutoff frequency: -3dB from bandwidth gain
2	Green	100	Filter's roll-off
3	Cyan	115	Filter's stop-band -100 dB from bandwidth gain
4	Black	200	Completely attenuated: filter's attenuation goes stronger as frequency increases

The designed filter fits the specification's needs.

7 APPENDIX

7.1 cFIR filter coefficient set coe file

Here is the coe file used within this application note to configure the cFIR filter:

```
;  
; Digital filter coefficients for Xilinx FIR implementation using Xilinx Coregen.  
;  
;  
Radix = 10;  
Coefficient_Width = 25;  
CoefData = -6794,  
-25628,  
19066,  
152768,  
10996,  
-520839,  
-247891,  
1309132,  
1095492,  
-2722296,  
-3752867,  
4976276,  
16488836,  
16488836,  
4976276,  
-3752867,  
-2722296,  
1095492,  
1309132,  
-247891,  
-520839,  
10996,  
152768,  
19066,  
-25628,  
-6794;
```

7.2 pFIR filter coefficient set coe file

Here is the coe file used within this application note to configure the pFIR filter:

```
;  
; Digital filter coefficients for Xilinx FIR implementation using Xilinx Coregen.  
;  
;  
Radix = 10;  
Coefficient_Width = 25;  
CoefData = 1394,  
1382,  
-8899,  
-35117,  
-60783,  
-48806,  
15695,  
80723,  
59080,  
-59529,  
-147405,  
-59387,  
154922,
```

```
233604,
2217,
-325120,
-303784,
170905,
573806,
284514,
-533130,
-874463,
-46518,
1197142,
1169994,
-702598,
-2545628,
-1388284,
3682586,
9865269,
12653715,
9865269,
3682586,
-1388284,
-2545628,
-702598,
1169994,
1197142,
-46518,
-874463,
-533130,
284514,
573806,
170905,
-303784,
-325120,
2217,
233604,
154922,
-59387,
-147405,
-59529,
59080,
80723,
15695,
-48806,
-60783,
-35117,
-8899,
1382,
1394;
```

7.3 SNMP configuration script

```
#!/bin/sh

IP=${1:-192.168.0.150}

#System Configuration
echo "#####"
echo "  SYSTEM CONFIGURATION"
echo "#####"
echo "Rack ID"
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.2.1.0 i 1

#Network configuration
echo "#####"
echo "  NETWORK CONFIGURATION"
```

```

echo "#####"
echo "FPGA IP"
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.4.1.0 s "10.1.1.1"
echo "IP Destination 0"
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.4.2.1.0 s "10.1.1.4"

#Acquisition Configuration
echo "#####"
echo " ACQUISITION CONFIGURATION"
echo "#####"

echo "Sample frequency"
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.2.9.0 i 100000000
echo "Capture mode: Rising Edge trigger"
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.2.6.0 i 0
echo "Trigger Level: LVPECL"
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.2.7.0 i 1

#Processing configuration
echo "#####"
echo " PROCESSING CONFIGURATION"
echo "#####"

echo "DDS frequencies"
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.1.1.0 i 69950000 # 50kHz
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.2.1.0 i 69914000 # 86kHz
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.3.1.0 i 69900000 # 100kHz
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.4.1.0 i 69885000 # 115kHz
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.5.1.0 i 69800000 # 200kHz
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.6.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.7.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.8.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.9.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.10.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.11.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.12.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.13.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.14.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.15.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.16.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.17.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.18.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.19.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.20.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.21.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.22.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.23.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.24.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.25.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.26.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.27.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.28.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.29.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.30.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.31.1.0 i 15000000
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.32.1.0 i 15000000

echo "Channel/PU mapping"
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.1.6.0 i 0 # Analog input 0
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.2.6.0 i 0
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.3.6.0 i 0
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.4.6.0 i 0
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.5.6.0 i 0
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.6.6.0 i 16 # 16 means OFF
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.7.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.8.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.9.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.10.6.0 i 16

```

```
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.11.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.12.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.13.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.14.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.15.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.16.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.17.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.18.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.19.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.20.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.21.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.22.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.23.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.24.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.25.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.26.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.27.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.28.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.29.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.30.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.31.6.0 i 16
snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.32.6.0 i 16
```

```
#Filters configuration
echo "#####"
echo " FILTERS CONFIGURATION"
echo "#####"

for pu_num in $(seq 1 15) ; do
  echo "DDC decimation setting"
  decimation=420
  snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.${pu_num}.7.0 i ${decimation}

  echo "cFIR coefficient loading"
  coefile=$(cat ../COE/EAGLE_fs_100_MHz_Decim_420_fc_86_kHz_cFIR.coe)
  snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.${pu_num}.8.0 s "$coefile"

  echo "pFIR coefficient loading"
  coefile=$(cat ../COE/EAGLE_fs_100_MHz_Decim_420_fc_86_kHz_pFIR.coe)
  snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.${pu_num}.9.0 s "$coefile"
done;

# Windowing configuration
echo "#####"
echo " WINDOWING CONFIGURATION"
echo "#####"

for pu_num in $(seq 1 15) ; do
  ##### BYPASS PARAMETER MUST BE SET BEFORE INPUT WINDOW #####
  echo "ByPass"
  snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.${pu_num}.3.0 i 0
  echo "Input Window"
  snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.${pu_num}.4.0 s "0 4200000 4294967295"
  echo "Output Destination"
  snmpset -v 1 -r 0 -t 15 -c public ${IP} 1.3.6.1.4.1.35959.1.2.3.${pu_num}.5.0 s "0 0"
done;
```

7.4 MATLAB design script

```
clc; close all; clear all;

%% System parameters
fs = 100e6; % Sampling frequency
User_Decim = 420; % CIC decimation factor
```

```

User_pFIR_Fc      = 89.5e3;    % Desired filter cutoff frequency
User_pFIR_Ripple = 0.1;      % Bandpass ripple

%% Targeted .coe files paths

% cFIR's coe file write location
cFIR_coe_path = strcat('./coe/EAGLE_fs_',...
    num2str(fs/1e6),...
    '_MHz_Decim_',...
    num2str(User_Decim),...
    '_fc_',...
    num2str(User_pFIR_Fc/1e3),...
    '_kHz_cFIR.coe');

% pFIR's coe file write location
pFIR_coe_path = strcat('./coe/EAGLE_fs_',...
    num2str(fs/1e6),...
    '_MHz_Decim_',...
    num2str(User_Decim),...
    '_fc_',...
    num2str(User_pFIR_Fc/1e3),...
    '_kHz_pFIR.coe');

%% CIC filter design

% Parameters
M_CIC      = User_Decim/4; % Decimation factor
DD_CIC     = 1;           % Differential delay
Nsecs_CIC  = 5;           % Number of sections

% Design
hCIC = mfilt.cicdecim(M_CIC,DD_CIC,Nsecs_CIC);

% Frequency response with normalized magnitude
hfvt_cic = fvtool(hCIC, 'Fs', fs);
legend(hfvt_cic, 'CIC gain response');
set(hfvt_cic, 'Name', 'CIC Gain Bode diagram');
set(hfvt_cic, 'ShowReference', 'off');
set(hfvt_cic, 'NormalizeMagnitudetol', 'on');
set(hfvt_cic, 'FrequencyScale', 'Log');

%% CIC droop compensation FIR (cFIR) design

% Parameters
M_cFIR      = 2;           % Decimation factor
cFIR_fcutoff = (fs/M_CIC)/4; % Cutoff frequency @ 6dB (max.: (fs/M_CIC)/4 to avoid
aliasing)
cFIR_Order  = 25;         % Filter order
Apass       = 0.001;      % Pass band maximum ripple
Astop       = 100;        % Stop band attenuation

% Design
cFIR_Specs = fdesign.decimator(M_cFIR, 'ciccomp', DD_CIC, Nsecs_CIC,...
    'n,fc,ap,ast', cFIR_Order, cFIR_fcutoff,...
    Apass, Astop, fs/M_CIC);
hcFIR = design(cFIR_Specs, 'equiripple');

% Frequency response with normalized magnitude
hfvt_cfir = fvtool(hcFIR, 'Fs', fs/M_CIC);
set(hfvt_cfir, 'FrequencyScale', 'Log');
legend('cFIR gain response');

% Gain normalization for comparison with the cFIR filter
hNorm_Gain = dfilt.scalar(1/gain(hCIC));
hCIC_norm = cascade(hNorm_Gain,hCIC);

```



```

% Displaying CIC & cFIR combined response
hfv_cic_cfir = fvtool(hCIC_norm, hcFIR, cascade(hCIC_norm, hcFIR), 'Fs', [fs
fs/M_CIC fs]);
legend(hfv_cic_cfir, 'CIC', 'cFIR', 'CIC + cFIR');
set(hfv_cic_cfir, 'ShowReference', 'off')
set(hfv_cic_cfir, 'NumberofPoints', 65536)
set(hfv_cic_cfir, 'FrequencyRange', 'Specify freq. vector');
fvect = linspace(0,4e6, 65536);
set(hfv_cic_cfir, 'FrequencyVector', fvect);
axes = get(hfv_cic_cfir, 'CurrentAxes');
set(axes, 'YLim', [-250 10]);

% Displaying cFIR's compensation effect on CIC's droop
hfv_droop = fvtool(hCIC_norm, hcFIR, cascade(hCIC_norm, hcFIR), 'Fs', [fs fs/M_CIC
fs]);
legend(hfv_droop, 'CIC's droop', 'cFIR's compensation', 'CIC's compensated
droop');
set(hfv_droop, 'ShowReference', 'off')
set(hfv_droop, 'NumberofPoints', 65536)
set(hfv_droop, 'FrequencyRange', 'Specify freq. vector');
fvect = linspace(0, User_pFIR_Fc, 65536);
set(hfv_droop, 'FrequencyVector', fvect);

%% pFIR design

% Parameters
M_pFIR      = 2;           % Decimation factor
pFIR_Fc     = User_pFIR_Fc; % Cutoff frequency @ 6dB
pFIR_Order  = 60;         % Filter order
Apass      = User_pFIR_Ripple; % Pass band maximum ripple
Astop      = 100;         % Stop band attenuation

% Design
pFIR_Specs = fdesign.decimator(M_pFIR, 'Lowpass', 'n,fc,ap,ast', pFIR_Order,
pFIR_Fc, Apass, Astop, fs/(M_CIC*M_cFIR));
hpFIR = design(pFIR_Specs, 'equiripple');

% Frequency response with normalized magnitude
hfv_pfir = fvtool(hpFIR, 'Fs', fs/(M_CIC*M_cFIR));
% set(hfv_pfir, 'FrequencyScale', 'Log');
legend('pFIR gain response');

%% Bit-true modelisation of the complete filter with truncatures

% Successive filters' bit widths
BitCicIn      = 17; % CIC input width
BitFirIn      = 17; % cFIR input width
BitpFirIn     = 24; % pFIR input width
BitpFirTruncOut = 32; % pFIR output width

% CIC Bit-true modelisation with truncatures
% CIC filter design
set(hCIC, 'InputWordLength', BitCicIn, ...
'InputFracLength', 0, ...
'FilterInternals', 'FullPrecision');

BitCicOut = hCIC.OutputWordLength;

% Fixed-point CIC output truncature
hCICtrunc = dfilt.scalar(2^(BitFirIn-BitCicOut));
set(hCICtrunc, 'Arithmetic', 'fixed', ...
'InputWordLength', BitCicOut, ...
'InputFracLength', 0, ...
'OutputMode', 'SpecifyPrecision', ...

```

```

        'OutputWordLength', BitcFirIn,...
        'OutputFracLength', 0,...
        'RoundMode', 'round');

%% cFIR Bit-true modelisation

% cFIR parameters
BitcFirCoef = 25;

% cFIR filter design
set(hcFIR, 'Arithmetic', 'fixed',...
        'InputWordLength', BitcFirIn,...
        'InputFracLength', 0,...
        'CoeffWordLength', BitcFirCoef,...
        'FilterInternals', 'FullPrecision');

% Filter's coefficients written to .coe file
set2int(hcFIR, 25);
FirCoef2XilinxCoe(hcFIR.Numerator, 25, cFIR_coe_path);

% cFIR truncature
BitcFirOut    = hcFIR.OutputWordLength;
BitcFirFracOut = hcFIR.OutputFracLength;

hcFIRtrunc = dfilt.scalar(2^(BitpFirIn-(BitcFirOut-BitcFirFracOut)));
set(hcFIRtrunc, 'Arithmetic', 'fixed',...
        'InputWordLength', BitcFirOut,...
        'InputFracLength', BitcFirFracOut,...
        'OutputMode', 'SpecifyPrecision',...
        'OutputWordLength', BitpFirIn,...
        'OutputFracLength', 0,...
        'RoundMode', 'round');

%% pFIR Bit-true modelisation

% pFIR parameters
BitpFirCoef = 25;

% pFIR filter design
set(hpFIR, 'Arithmetic', 'fixed',...
        'InputWordLength', BitpFirIn,...
        'InputFracLength', 0,...
        'CoeffWordLength', BitpFirCoef,...
        'FilterInternals', 'FullPrecision');

% Filter's coefficients written to .coe file
set2int(hpFIR, 25);
FirCoef2XilinxCoe(hpFIR.Numerator, 25, pFIR_coe_path);

% pFIR truncature
BitpFirOut    = hpFIR.OutputWordLength;
BitpFirFracOut = hpFIR.OutputFracLength;

hpFIRtrunc = dfilt.scalar(2^(BitpFirTruncOut-(BitpFirOut-BitpFirFracOut)));
set(hpFIRtrunc, 'Arithmetic', 'fixed',...
        'InputWordLength', BitpFirOut,...
        'InputFracLength', hpFIR.OutputFracLength,...
        'OutputMode', 'SpecifyPrecision',...
        'OutputWordLength', BitpFirTruncOut,...
        'OutputFracLength', 0,...
        'RoundMode', 'round');

%% Fixed-point complete FM filter

% FM filter static gain calculation

```

```

% CIC gain
CIC_static_gain = 20*log10(gain(hCIC));
% CIC truncature gain
CIC_trunc_gain = 20*log10(2^(hCICtrunc.OutputWordLength-
hCICtrunc.OutputFracLength)/2^(hCIC.OutputWordLength-hCIC.OutputFracLength));

% cFIR gain
f = linspace(0,38e3,10);
cFIR_gain = 20*log10(abs(freqz(hcFIR, [0 2*pi*f/fs])));
cFIR_static_gain = cFIR_gain(1);

% cFIR truncature gain
cFIR_trunc_gain = 20*log10(2^(hcFIRtrunc.OutputWordLength-
hcFIRtrunc.OutputFracLength)/2^(hcFIR.OutputWordLength-hcFIR.OutputFracLength));

% pFIR gain
f = linspace(0,10e3,1e6);
pFIR_gain = 20*log10(abs(freqz(hpFIR, [0 2*pi*f/fs])));
pFIR_static_gain = mean(pFIR_gain(2:end));

% pFIR truncature gain
pFIR_trunc_gain = 20*log10(2^(hpFIRtrunc.OutputWordLength-
hpFIRtrunc.OutputFracLength)/2^(hpFIR.OutputWordLength-hpFIR.OutputFracLength));

% FM filter static gain calculation
FM_filter_static_gain = CIC_static_gain + CIC_trunc_gain +...
                        cFIR_static_gain + cFIR_trunc_gain +...
                        pFIR_static_gain + pFIR_trunc_gain;

hFM_fi_Filter = cascade(hCIC, hCICtrunc, hcFIR, hcFIRtrunc, hpFIR, hpFIRtrunc);
hfvt_fm_filter = fvtool(hFM_fi_Filter, 'Fs', fs);
set(hfvt_fm_filter, 'NumberofPoints', 262144)
set(hfvt_fm_filter, 'NormalizeMagnitudetol', 'on');
hfvt_fm_filter = fvtool(hFM_fi_Filter, 'Fs', fs);
legend(hfvt_fm_filter, 'Complete filter gain response');
set(hfvt_fm_filter, 'ShowReference', 'off')
set(hfvt_fm_filter, 'NumberofPoints', 262144)
set(hfvt_fm_filter, 'FrequencyRange', 'Specify freq. vector');
fvector = linspace(0, 2e6, 262144);
set(hfvt_fm_filter, 'FrequencyVector', fvector);
set(hfvt_fm_filter, 'NormalizeMagnitudetol', 'on');
axes = get(hfvt_fm_filter, 'CurrentAxes');
set(hfvt_fm_filter, 'FrequencyScale', 'Log');
set(axes, 'YLim', [-250 10]);

```

8 SUPPORT INFORMATION

Should you have any questions or support requests, please feel free to contact TECHWAY.

Website: www.techway.com
Email: support@techway.com
Phone: +33 (0)1 64 53 37 90

