LatticeCORE

Peak Cancellation Crest Factor Reduction IP Core User's Guide

May 2014
IPUG109_1.1

Table of Contents

Chapter 1. Introduction 4
- Quick Facts 4
- Features 5
- Release Information 5
- Device Support 5

Chapter 2. Functional Description 6
- Functional overview 6
- Block Diagram 8
- Mathematical Model 9
- Filter Selection 10
- CFR Performance 12
- Primary I/O 14
- Interface Descriptions 18
  - Data Input/Output 18
  - Control Signals and Timing 18
  - Recommended CFR Coefficients Generation 19
  - DSP Utilization Estimation 20

Chapter 3. Parameter Settings 21
- Architecture Tab 23
- Coefficients Tab 24
- Others Tab 25

Chapter 4. IP Core Generation 27
- Licensing the IP Core 27
- IP Core Generation in IPexpress 27
  - Getting Started 27
  - Configuring CFR Core in IPexpress 28
  - IPexpress-Created Files and Top Level Directory Structure 30
  - Instantiating the Core 31
  - Running Functional Simulation 31
  - Synthesizing and Implementing the Core in a Top-Level Design 32
  - Hardware Evaluation 32
  - Enabling Hardware Evaluation in Diamond 32
  - Updating/Regenerating the IP Core 32
- IP Core Generation in Clarity Designer 33
  - Getting Started 33
  - IP Core Implementation 38
  - Hardware Evaluation 39
  - Enabling Hardware Evaluation in Diamond 39
  - Regenerating an IP Core in Clarity Designer Tool 39
  - Recreating an IP Core in Clarity Designer Tool 40

Chapter 5. Support Resources 41
- Lattice Technical Support 41
  - E-mail Support 41
  - Local Support 41
  - Internet 41
  - IEEE 41
- References 41
  - LatticeECP3 41
  - ECP5 41

©2014 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal. All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

References 41
Revision History 41
Appendix A. Resource Utilization 42
LatticeECP3 Devices 42
Ordering Part Number 42
ECP5 Devices 42
Ordering Part Number 42

Introduction

Crest Factor Reduction (CFR) selectively reduces the peak-to-average ratio (PAR) of wideband digital signals, such as those used in third-generation (3G) code division multiple access (CDMA) or long term evolution (LTE) wireless applications. A power amplifier is used in the final stage of transmitting signal in cellar base stations and is performance limited by high PAR. Amplifiers work best in the linear range where input signals stay within a bounded range. Large peaks in the input signal drive the amplifier into the non-linear region, which can cause adjacent channel leakage and low power efficiency.

The Lattice Peak Cancellation Crest Factor Reduction (PC-CFR) IP core is flexible and has an efficient implementation supporting the LatticeECP3 FPGA device families. The highly configurable design takes advantage of the embedded DSP blocks available in Lattice FPGAs.

Quick Facts

Table 1-1. PC-CFR IP Core Quick Facts

		PC-CFR Core Configuration		
		1 Antenna 1 Stage 8 Clip engines	**1 Antenna 2 Stages 10 Clip engines**	**1 Antenna 3 Stages 14 Clip engines**
Core Requirements	FPGA Families Supported	LatticeECP3™, ECP5™		
	Minimum Device Required	LFE3-35EA, LFE5UM-25F		
Resource Utilization	Width of data	18		
	Width of coefficient	18		
	Targeted Device	LFE3-70EA-9FN672C		
	Registers	2750	4321	6338
	LUTs	2643	4493	6599
	sysMEM EBRs	5	7	10
	MULT18X18	18	24	34
Resource Utilization	Targeted Device	LFE5UM-85F-8MG756C		
	Registers	2988	4672	6800
	LUTs	2704	4537	6658
	sysMEM EBRs	5	7	10
	MULT18X18	10	14	20
Design Tool Support	Lattice Implementation	Lattice Diamond® 3.2		
	Synthesis	Synopsys® Synplify™ Pro I-2013.09L		
	Simulation	Mentor Graphics® ModelSim™ SE Plus 6.3f		
		Aldec® Active-HDL™ 9.3 Lattice Edition		

Features

- The Lattice PC-CFR IP supports one to four antennas.
- The core can be configured for clock-to-sample ratios of 1, 2 or 4.
- Provides 1, 2 or 3 sequential detection and cancellation stages to remove peaks.
- Cancellation pulses may be real or complex.
- Supports up to 4 coefficient sets for dynamic switching.
- Dynamically programmable clipping amplitude threshold and peak detector window width.
- Number of clip engines for each filter stage can be independently configured.
- The peak cancellation pulse length is configurable.
- IP provides parallel or TDM (Time-Division Multiplexing) quadrature (I & Q) format for input/output mode.
- Configurable data width.
- Configurable coefficient width.
- Option for output gain.
- Supports DSP high-speed mode for ECP5 devices

Release Information

Peak Cancellation Crest Factor Reduction IP core version 2.0

Last updated May 2014

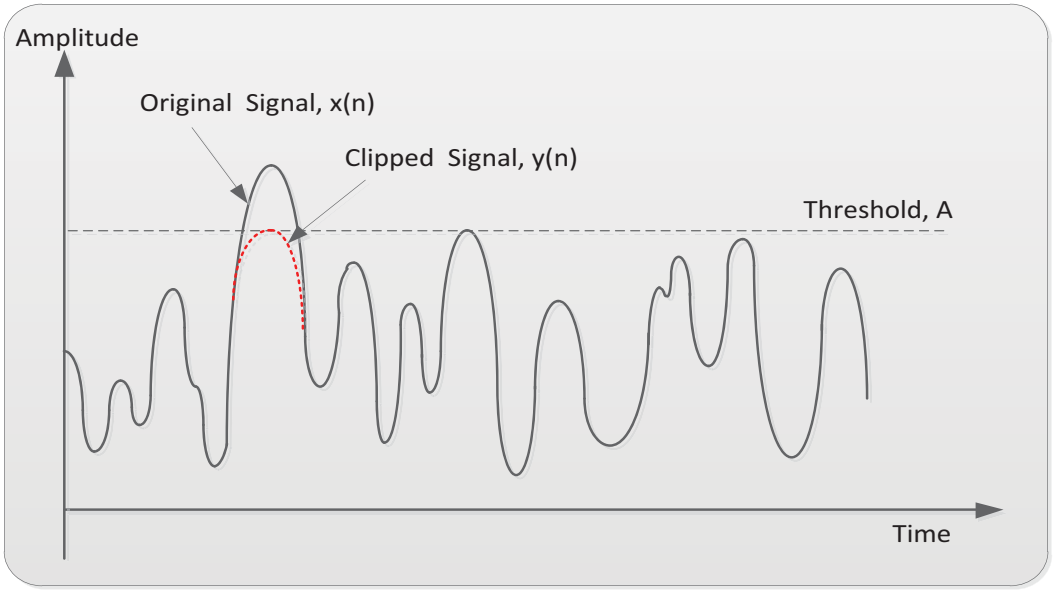
Device Support

LatticeECP3, ECP5

Functional overview

The crest factor reduction block processes and reduces the Peak to Average Power of the signal while minimizing the adjacent and alternate channel power leakage. Figure 2-1 shows output clipping with CFR.

Figure 2-1. Output clipping with CFR



Typical techniques for crest factor reduction are grouped into 2 sections:

- Signal Agnostic (or protocol independent)
- Signal Dependent (or protocol specific).

Although signal dependent techniques have some advantages in the performance that can be reached, they use and take advantage of the baseband modulation and thus cannot be used in a multi-mode remote radio head.

Of the signal agnostic techniques, the most common are:

- Peak Cancellation Crest Factor Reduction (PC-CFR)
- Peak Windowing Crest Factor Reduction (PW-CFR)

The simple clipping of the peaks greater than a threshold A is achieved by saturating the signal at A. The saturation non-linearity is given by the following equation.

$$x'(n) = \begin{cases} A & x(n) > A \\ x(n) & |x(n)| \le A \\ -A & x(n) < -A \end{cases}$$

The above equation can be written in the following multiplicative form:

$$x'(n) = c(n)x(n)$$

Where *c(n)* is given by

$$c(n) = \begin{cases} 1, & |x(n)| \le A \\ \dfrac{A}{|x(n)|}, & |x(n)| > A \end{cases}$$

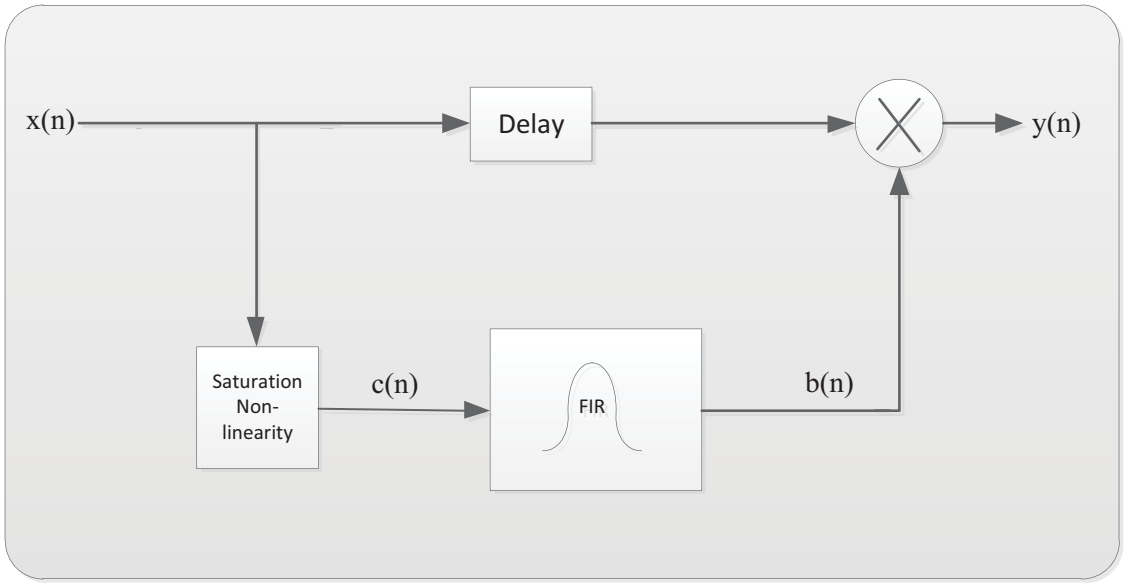
This simple clipping causes sharp corners in the clipped signal leading to high out-of-band errors. To reduce the distortion due to clipping, a window function is used in-place of the saturation non-linearity.

The output after convolving with the windowing function w(n) in time domain is given by the following equation.

$$b(n) = 1 - \sum_{k=-\infty}^{\infty} [1 - c(k)]w(n-k)$$

This convolution can be implemented using a suitable FIR filter. The block diagram of peak windowing method is presented in Figure 2-2.

Figure 2-2. Block Diagram of the Windowing Method



Peak Cancellation CFR is a type of Peak Windowing CFR that uses fewer multiplier resources. Since multiplier resources in a Remote Radio Head are the typical limiting factor, PC-CFR is preferred over most other implementations. PC-CFR also has shown to outperform PW-CFR.

The main driver in CFR is the reduction of the PAR of the signal. This is accomplished by identifying the peaks of the signal, and modifying them to reduce their values. This modification of the signal has direct effect on the system performance, so two measures are used to determine if the modification is correct:

- EVM: Error vector magnitude
- ACLR: Adjacent Channel Leakage Ratio

Block Diagram

An overview of a PC-CFR module can be seen in Figure 2-3. The CORDIC unit is used to calculate the magnitude and phase of the high PAR input signal. Then the peak detect block identifies the signal sample at which the magnitude exceeds the CFR threshold. The clip engine manger is to allocate and assign clip engines to peaks.

Figure 2-4 shows a single clip engine component and Figure 2-5 shows the block diagram of the clip engine manager. Based on the users' requirements, the CFR is flexible to set a finite number of clip engine to reduce the crest of the input signal. This means that, not all the peaks are guaranteed to be cancelled in one pass, so users can select multiple stages.

As the cancellation pulse coefficients are generated offline and stored into RAM, CFR can support a wide variety of carrier configurations and bandwidth using the same hardware. The Lattice PC-CFR is flexible and can support different air interfaces.

Figure 2-3. PC-CFR Stages with N Clip Engines

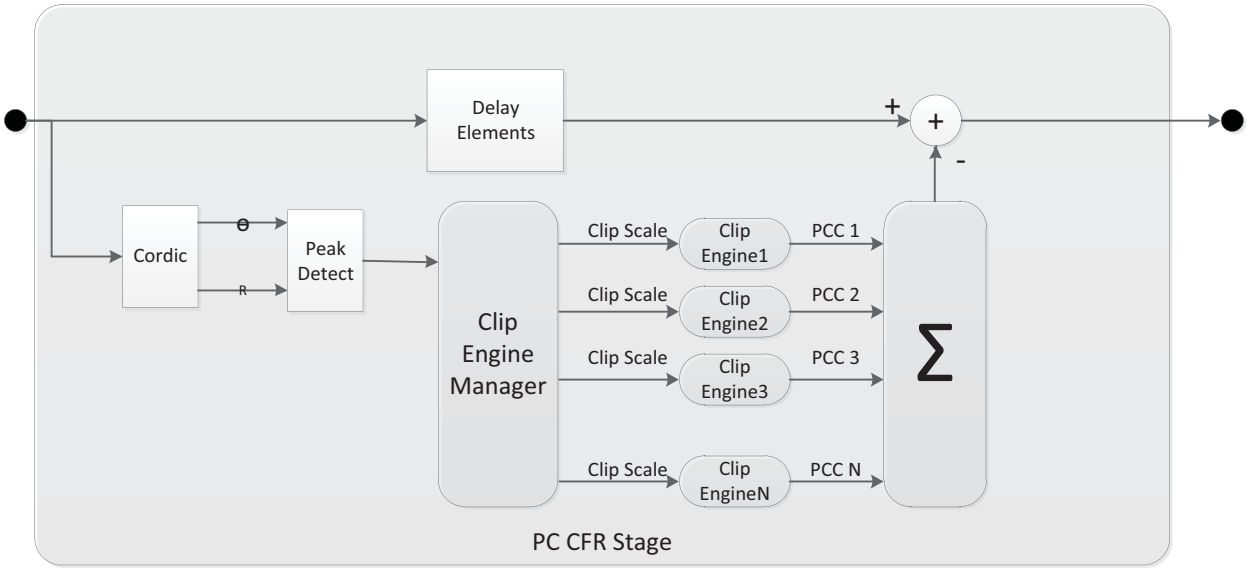


Figure 2-4. Clip Engine

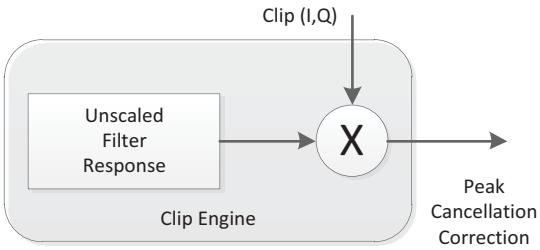
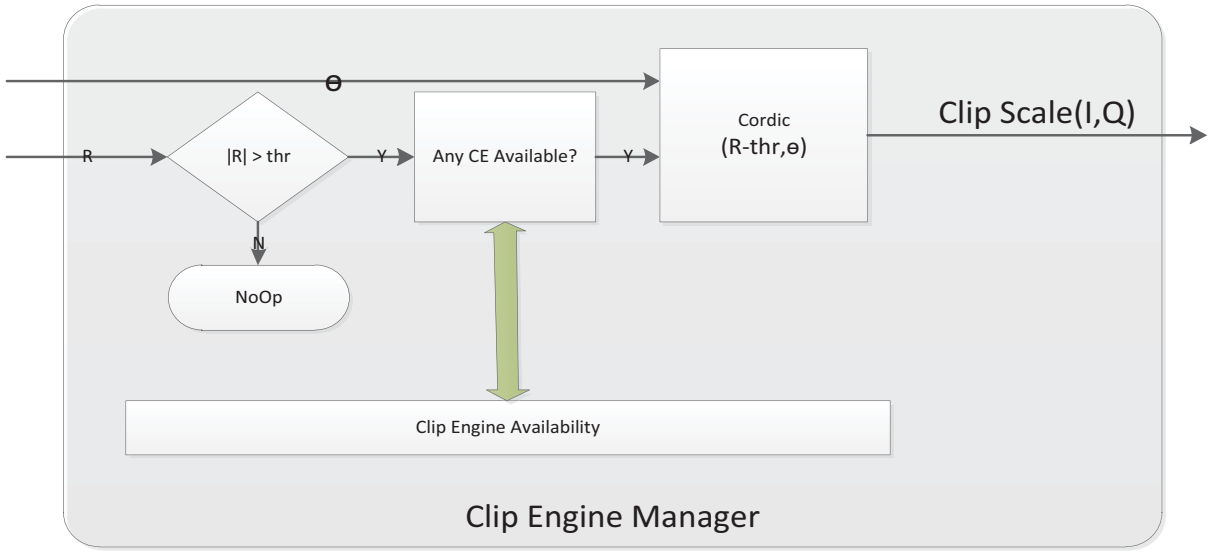


Figure 2-5. Clip Engine Manager



Mathematical Model

Defining the input to a stage of crest factor at sample n to be:

$$s(n) = r(n)e^{j2\pi\theta(n)}$$

$s(n)$	is a complex signal
$r(n)$	is the magnitude of the signal $s(n)$, and a real number
$\theta(n)$	is the angle of the signal $s(n)$

The simplest model of peak window crest factor reduction (PW-CFR) simply selects the data that is larger than the threshold:

$$e(n) = \begin{cases} (r(n) - \gamma)e^{j2\pi\theta(n)} & r(n) > \gamma \\ 0 & otherwise \end{cases}$$

The PW-CFR simply filters the error or correction with a filter response of length (K+1).

$$H(l) = \{h_0, h_1, \ldots, h_K\}.$$

The signal that carries the correction to reduce the peaks is now:

$$c(n) = e(n) \otimes h(n)$$

PW-CFR then simply adds the correction *c(n)* to a delayed version of the original signal *s(n)*. The amount of delay needed is the group delay of the filter *H(n)*. This delay is K/2 samples.

So the final signal after the stage in PW-CFR becomes

$$s_{OUT}(n) = s_{IN}(n - K/2) - c(n)$$

This implementation of Crest Factor reduction uses a filter operation to create *c(n)*. If the number of taps (K+1) is large and the FPGA clock rate is same as the data sample rate, then 2*(K+1) multipliers would be needed for the filtering for the I&Q components if the filter is a real filter.

Looking at the signal *e(n)*, it is apparent that it is zero for majority of the time. Therefore, direct implementation of the filter results in the use of unnecessary multipliers.

The Peak Cancellation approach selects each instance of the signal *e(n)* that exceeds the threshold and multiplies that single instance by the filter coefficients. This implementation lets the user select the number of clips that the signal can simultaneously perform. This parameter is chosen in the GUI and is called number of clip engines. Once a peak is selected, the clip engine stays occupied for (K+1) samples, to generate the response of the single instance by the filter H.

In the limit, if the number of clip engines equals the number of taps in the filter H both PC-CFR and PW-CFR are mathematically identical. Due to the nature of the distribution of the signal *e(n)*, the number of active clips at any one time for a filter of size 255 typically does not exceed 8.

When the signal presented to CFR is an upsampled signal, contiguous samples can exceed the threshold at the same time. This is because these samples have been created by filtering a peak through an upsampled filter. To account for this the PC-CFR has a parameter called skip that can ignore certain peaks. The mathematical model is such that a peak is only chosen for cancellation if it is the largest peak in the range [1,skip]. Setting the skip parameter to 1 allows for cancellation of all the clips.

Also one must note that since a corrected signal *c(n)* is added to the original signal there exists a non-zero probability that some samples will add coherently and thus new peaks may be created. This can be reduced by staging multiple PC-CFR stages back to back. The CFR core allows the user to create a core with one, two or three CFR stages, and allowing the user to configure the skip, threshold and filter independently for each stage.

Filter Selection

The CFR core allows the user to program the filter coefficients of each filter stage independently. This can be done prior to the generation of the core or can be selected to change at run-time. The user will be able to create a CFR core with either real or complex filter coefficients. If real filter coefficients are used the implementation is smaller.

Some restrictions are added to the filter types depending on whether the filter is real or complex. If the filter is real, the core requires the coefficients to be symmetric and the filter length to be odd. The reason for the symmetry requirement is to ensure that the filter has linear phase. The odd filter length is required to properly match the delay between the signals. *i.e.*

$$s_{OUT}(n) = s_{IN}(n - K/2) - c(n)$$

The filter H(n) should be chosen to exceed the spectral mask requirements from the specific standard for which the CFR core is deployed.

The choice of real or complex filter depends on the type of signal that is presented to the CFR block. If a single carrier centered at 0Hz is presented then the filter can be real.

$$H(l) = \left\{h_0, h_1, \ldots., h\left(\frac{K}{2}\right), \ldots, h_{K-1}, h_K\right\}$$

$$h_M = h_{K-M} \quad M = \{0,1, \ldots, \frac{K}{2}\}$$

When the signal that makes up the data presented to the CFR block is a composite of carriers not centered around 0Hz the filter must be complex.

Assume that there are N carriers, the carrier baseband offset of carrier P is f_{c_P}, the gain (linear) of carrier P is g_{C_p}, and the CFR sampling rate is F_s.

To create the filter for the composite signal one must mix each individual carrier filter.

$$H_P(l) = \left\{h_0, h_1, \ldots., h\left(\frac{K}{2}\right), \ldots, h_{K-1}, h_K\right\}$$

$$h_M = h_{K-M} \quad M = \{0,1, \ldots, \frac{K}{2}\}$$

One must create for each of the Bandwidths and standard types, a basic CFR filter of the same length (K+1).

Create a sample sequence of length (K+1) from –K/2 to +K/2:

$$sample(l) = \left\{-\frac{K}{2}, -\left(\frac{K}{2}-1\right), \ldots, -2, -1, 0, 1, 2, \ldots, \frac{K}{2}-1, \frac{K}{2}\right\}$$

$$l = \{0,1, \ldots, n\}$$

$$sample(l) = -s(K-l) \quad l = \{0,1, \ldots, \frac{K}{2}\}$$

After creating a filter per carrier and mixing,

$$mixer\ F_{c_P}(l) = g_{C_P} * H_P(l) * \exp^{-j*2\pi*\frac{f_{c_P}}{F_s}*sample(l)}$$

One can see that the mixer F_{c_P} is conjugate symmetric.

$$mixF_{carrier}(l) = mixF^*_{carrier}(K-l)$$

$$l = \{0,1, \ldots, \frac{K}{2}\}$$

The final CFR filter is then created by adding all of the mixed filters. Since the mixed filters are all conjugate symmetric, the composite CFR filter is also conjugate symmetric.

The I & Q components of the filter are:

$$\begin{bmatrix} i_0 & q_0 \\ i_1 & q_1 \\ \vdots & \vdots \\ i_{\frac{K}{2}} & q_{\frac{K}{2}} \\ \vdots & \vdots \\ i_{n-1} & q_{n-1} \\ i_n & q_n \end{bmatrix} = \begin{bmatrix} i_0 & q_0 \\ i_1 & q_1 \\ \vdots & \vdots \\ i_{\frac{K}{2}} & 0 \\ \vdots & \vdots \\ i_1 & -q_1 \\ i_0 & -q_0 \end{bmatrix}$$

When the filter is chosen to be complex then the filter is assumed to be of odd length and conjugate symmetric. This is done to save resources on the filter storage per clip engine.

CFR Performance

Typically the performance of a CFR algorithm is determined by examining the complementary cumulative distribution function (CCDF) curve. The CCDF of the transmit output power is the probability that the signal power is greater than a given PAR. At a given probability level (usually 10^{-4}), the PAR of an OFDM symbol that has been compressed by a CFR algorithm can be compared with the OFDM symbol that has not been compressed. The modulation accuracies of the WCDMA and LTE signals are measured by error vector magnitude (EVM). EVM is a measure for the difference between the theoretical waveform and modified version of the measured waveform. CCDF, EVM and other measures like ACLR are all inter-related. As reducing PAR leads to increasing EVM, the designer has to balance the PAR reduction to suit his needs.

The CCDF and EVM performance plots for the PC-CFR IP are given below. Figure 2-6 and Figure 2-8 show the performance plots for one carrier (20MHz bandwidth) LTE applications with an Fs of 122.88 Msps.

Figure 2-6. CCDF at 7dB PAR Target with 2 Stages for One Carrier 20MHz LTE

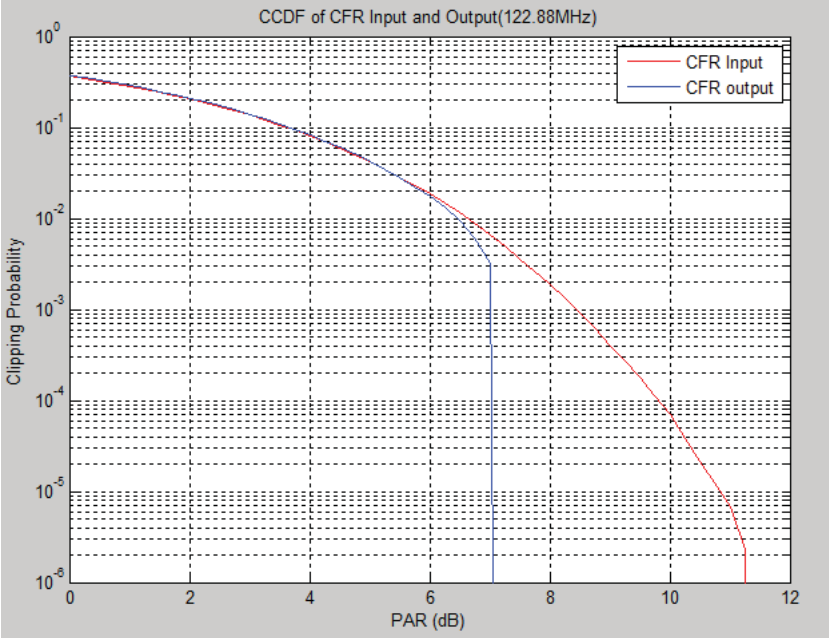


Figure 2-7. Signal Amplitudes of CFR Input and Output for One Carrier 20MHz LTE

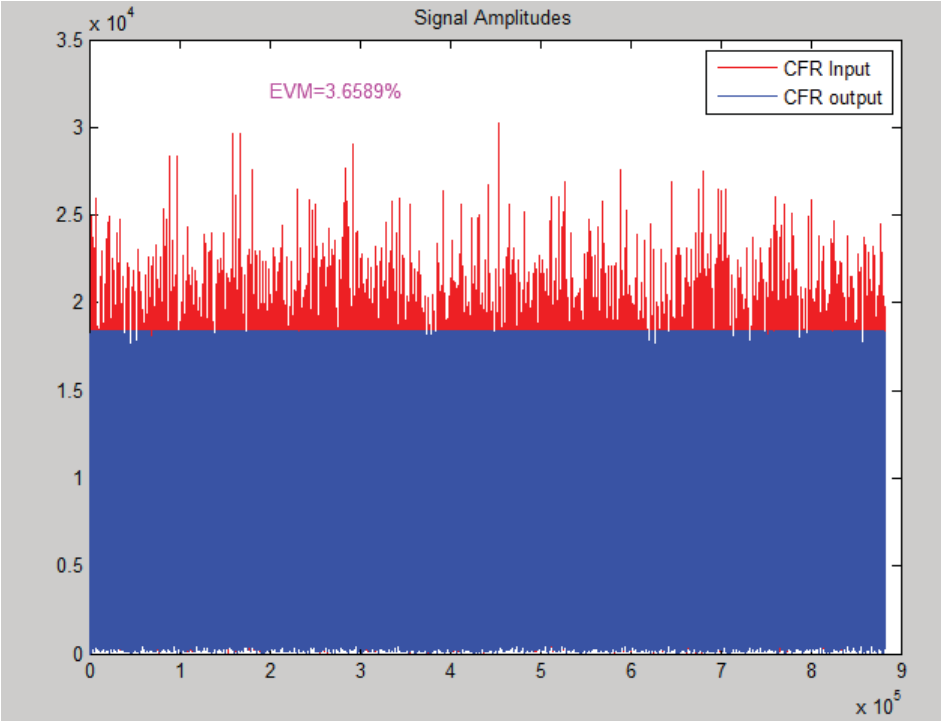
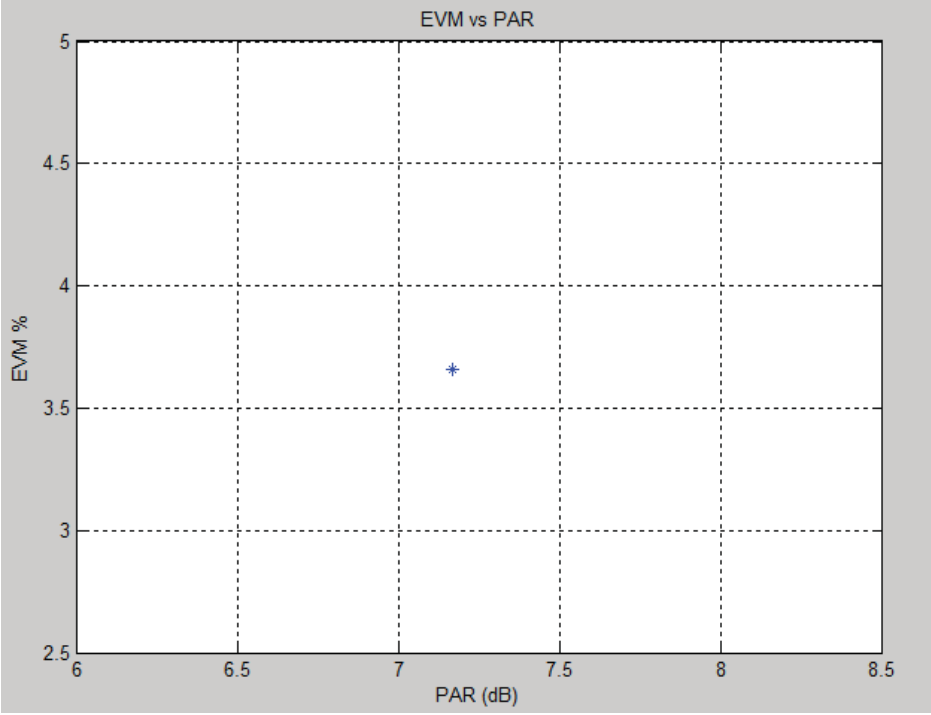


Figure 2-8. EVM vs PAR for One Carrier 20MHz LTE



Primary I/O

Figure 2-9 shows the PC-CFR core top diagram and Table 2-1 gives a description of the input-output ports.

Figure 2-9. CFR Top-level Block Diagram

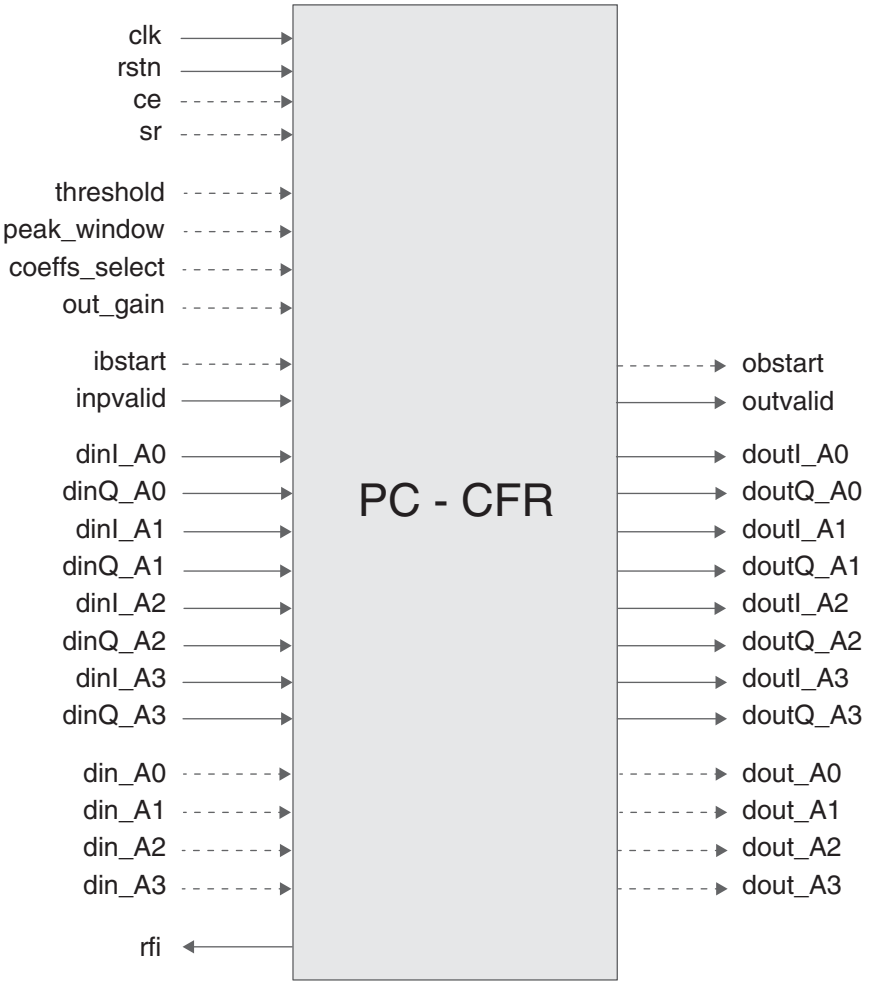


Table 2-1. Lattice CFR I/O Port Description

Port	Bits	I/O	Description
All configurations			
clk	1	I	System clock (reference clock for input and output data).
rstn	1	I	System wide asynchronous active-low reset signal.
rfi	1	O	Ready for New Input. When asserted the core can receive new valid input data.
dinI_A0	11-18	I	Slave interface for the first antenna I component of data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_parallel”)

Table 2-1. Lattice CFR I/O Port Description

Port	Bits	I/O	Description
dinQ_A0	11-18	I	Slave interface for the first antenna Q component of data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_parallel”)
dinI_A1	11-18	I	Slave interface for the second antenna I component of data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_parallel” and number of CFR antennas is greater than 1)
dinQ_A1	11-18	I	Slave interface for the second antenna Q component of data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_parallel” and number of CFR antennas is greater than 1)
dinI_A2	11-18	I	Slave interface for the third antenna I component of data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_parallel” and number of CFR antennas is greater than 2)
dinQ_A2	11-18	I	Slave interface for the third antenna Q component of data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_parallel” and number of CFR antennas is greater than 2)
dinI_A3	11-18	I	Slave interface for the fourth antenna I component of data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_parallel” and number of CFR antennas is “4”)
dinQ_A3	11-18	I	Slave interface for the fourth antenna Q component of data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_parallel” and number of CFR antennas is “4”)
din_A0	11-18	I	Slave interface for the first antenna data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_tdm”)
din_A1	11-18	I	Slave interface for the second antenna data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_tdm” and number of CFR antennas is greater than 1)
din_A2	11-18	I	Slave interface for the third antenna data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_tdm” and number of CFR antennas is greater than 2)

Table 2-1. Lattice CFR I/O Port Description

Port	Bits	I/O	Description
din_A3	11-18	I	Slave interface for the fourth antenna data input to the core. Port width equal to DINWIDTH parameter specified at configuration. (available only when input_mode is “iq_tdm” and number of CFR antennas is “4”)
inpvalid	1	I	Active high. When asserted, the core reads the data on the dinI /dinQ or din ports
ibstart	1	O	Active high. when asserted, the core reads I component from din, otherwise reads Q component. (available only when input_mode is “iq_tdm”)
doutI_A0	11-18	O	Master interface for the first antenna I component of data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_parallel”)
doutQ_A0	11-18	O	Master interface for the first antenna Q component of data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_parallel”)
doutI_A1	11-18	O	Master interface for the second antenna I component of data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_parallel” and number of CFR antennas is greater than 1)
doutQ_A1	11-18	O	Master interface for the second antenna Q component of data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_parallel” and number of CFR antennas is greater than 1)
doutI_A2	11-18	O	Master interface for the third antenna I component of data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_parallel” and number of CFR antennas is greater than 2)
doutQ_A2	11-18	O	Master interface for the third antenna Q component of data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_parallel” and number of CFR antennas is greater than 2)
doutI_A3	11-18	O	Master interface for the fourth antenna I component of data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_parallel” and number of CFR antennas is “4”)

Table 2-1. Lattice CFR I/O Port Description

Port	Bits	I/O	Description
doutQ_A3	11-18	O	Master interface for the fourth antenna Q component of data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_parallel” and number of CFR antennas is “4”)
outvalid	1	O	Active high. When asserted, indicates that the current output data on the doutI and doutQ ports is valid.
dout_A0	11-18	O	Master interface for the first antenna data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_tdm”)
dout_A1	11-18	O	Master interface for the 2nd antenna data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_tdm” and number of CFR antennas is greater than 1)
dout_A2	11-18	O	Master interface for the third antenna data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_tdm” and number of CFR antennas is greater than 2)
dout_A3	11-18	O	Master interface for the 4th antenna data output from the core. Port width equal to DOUTWIDTH parameter specified at configuration. (available only when output_mode is “iq_tdm” and number of CFR antennas is “4”)
obstart	1	O	Active high. When asserted, the core reads I component from din, otherwise reads Q component. (available only when output_mode is “iq_tdm”.)
Optional I/Os			
ce	1	I	Clock Enable. While this is de-asserted, the core will ignore all other synchronous inputs and maintain its current state.
sr	1	I	Synchronous Reset. Asserted for at least one clock period duration in order to re-initialize the core state.
threshold	11-18	I	Specify the clipping amplitude threshold of CFR core. Port width equal to DINWIDTH parameter specified at configuration.
peak_window	5	I	Specify the peak detector window width. Its range is 3-31.
coeffs_select	2	I	Specify the coefficient set: 2’b00 selects coefficient set 0; 2’b01 selects coefficient set 1; 2’b10 selects coefficient set 2; 2’b11 selects coefficient set 3;
out_gain	16	I	Specify the scalar gain to be applied to the output data of the last stage. It is a binary fractional number (16.13). Its value is (-4, 4) in floating point. 16’d8192 means 1.0 in floating point.

Interface Descriptions

Data Input/Output

The Lattice CFR core uses a simple handshake to pass input data into the core. The core asserts its *rfi* output when it is ready to receive data. When the driving module has data to give the core, it drives the core's inpvalid port to a "1" synchronously with the rising edge of the *clk* signal, providing the input data on port *dinI_A0*, *dinQ_A0*… (or *din_A0*… when TDM input mode). The *ibstart* input should be driven to a"1" during the clock cycle when the valid input data of I channel is incoming.

Correspondingly, *outvalid* is active when valid output data is available on (*doutI_A0*, *doutQ_A0*… (or *dout_A0*… when TDM output mode)*dout*, and *obstart* marks the I channel of the output valid data.

Control Signals and Timing

Figure 2-10 shows the CFR interface timing when FOLD_RATE is 1. in this configuration, "rfi" will always be high.The core can accept input I/Q data pairs and output I/Q data pairs every clock cycle and I/Q data are processed in parallel. Dynamic control ports "threshold", "peak_window", "coeffs_select" and "out_gain" can be set to specified values if they are enabled. And those parameter values are closely related to input data property.

Figure 2-10. CFR Timing Diagram for FOLD_RATE 1

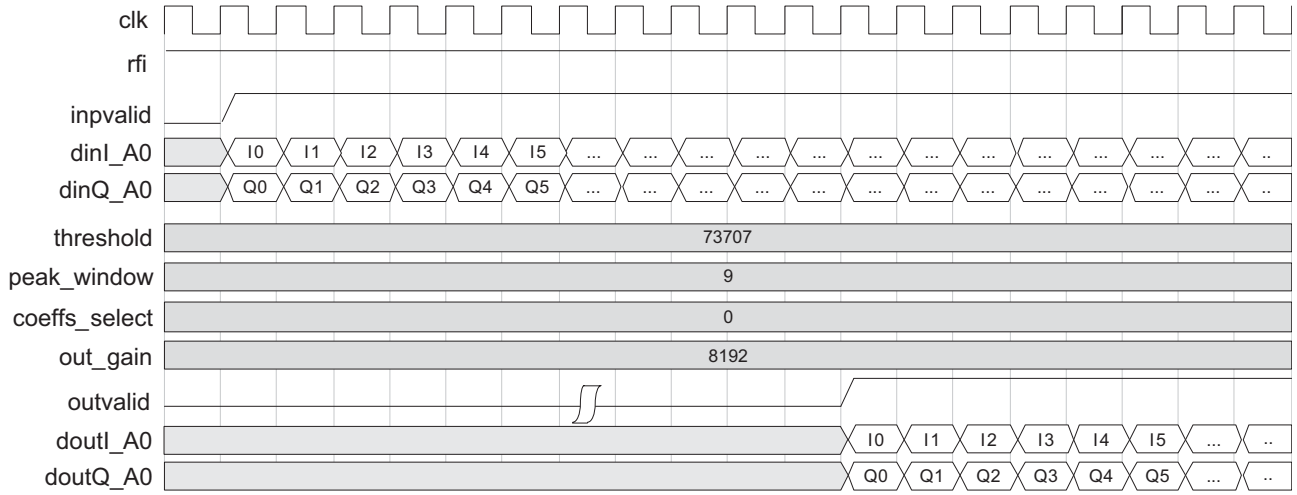


Figure 2-11 shows the interface timing when FOLD_RATE is 2 and input/output mode are IQ parallel. In this configuration, the core can accept input I/Q data pairs and output I/Q data pairs every two clock cycle.

Figure 2-11. CFR Timing Diagram for FOLD_RATE 2 with IQ Parallel

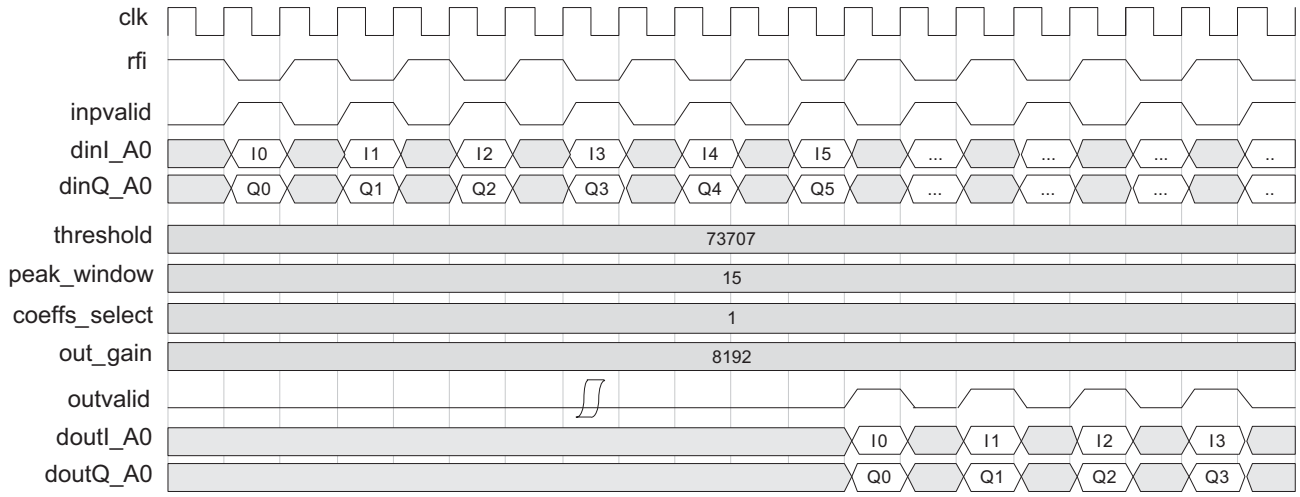
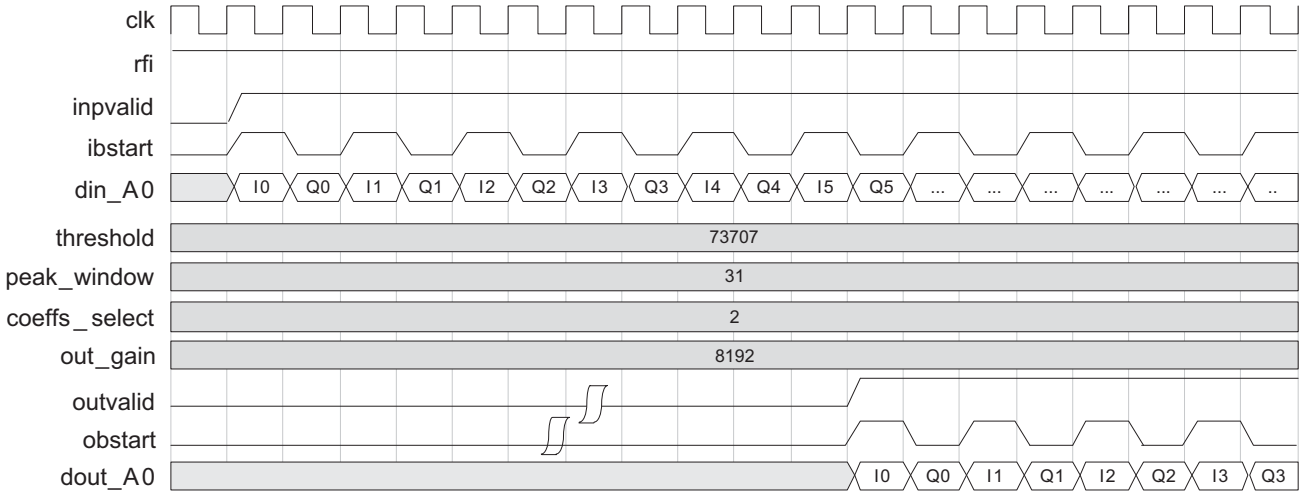


Figure 2-12 shows the interface timing when FOLD_RATE is 2 and input/output mode are IQ TDM. In this configuration, “rfi” will always be high. The core can accept input I/Q data and output I/Q data every clock cycle. I/Q data are interleaved in the CFR input/output data port. “ibstart” and “obstart” indicates the I channel data

Figure 2-12. CFR Timing Diagram for FOLD_RATE 2 with IQ TDM



Recommended CFR Coefficients Generation

We recommend CFR coefficients generated by Matlab fucntion “firls” with the input data property. Here is a example of CFR coefficients generation for 10MHz bandwidth data stream.

```
% Calculate the coefficients using the FIRLS function.
BW    = 10;       % Signal bandwidth
Fs    = 122.88;   % Sampling Rate
N     = 255;      % Order
Fpass = 4.5;      % Passband Frequency
Fstop = 5;        % Stopband Frequency
Wpass = 1;        % Passband Weight
Wstop = 50;       % Stopband Weight
Fls  = firls(N-1, [0 Fpass Fstop Fs/2] /(Fs/2), [1 1 0 0], [Wpass Wstop]);
```

Lattice CFR core supports odd symmetric coefficients only. And the coefficients file can only be in floating point format with one coefficient in one line. Such as:

```
-0.000009318083267740
 0.000099706080556368
 0.000240879880800352
 0.000411396748259621
 0.000606203071643045
 0.000818068719931488
 0.001037801420774560
 …
```

DSP Utilization Estimation

When coefficients type is “real”, the each stage of each antenna of CFR core will cost

```
((2 + 2*Number of clip engines)/FOLD_RATE)
```

multipliers (MULT18x18).

When coefficients type is “complex”, the each stage of each antenna of CFR core will cost

```
((2 + 4*Number of clip engines)/FOLD_RATE)
```

multipliers (MULT18x18).

For ECP5 devices, if DSP high-speed mode is enabled, When coefficients type is “real”, the each stage of each antenna of CFR core will cost

```
((2 + Number of clip engines)/FOLD_RATE)
```

multipliers (MULT18x18).

When coefficients type is “complex”, the each stage of each antenna of CFR core will cost

```
((2 + 2*Number of clip engines)/FOLD_RATE)
```

multipliers (MULT18x18).

And if output gain is enabled, extra 2 multipliers are needed for the CFR core.

Parameter Settings

This section describes how to generate the Lattice CFR IP core using the Diamond IPexpress™ tool. Refer to "IP Core Generation" for a description of how to generate the IP.

The CFR configuration GUI is accessed via the IPexpress tool and provides an interface for setting the desired parameters and invoking the IP core generator. Since the values of some parameters affect the size of the resultant core, the maximum value for these parameters may be limited by the size of the target device. Table 3-1 provides a list of user-configurable parameters for the CFR IP core.

Table 3-1. Lattice CFR IP Core Parameters

Parameter	Dependency and Description	Range	Default
N_STAGES	Independent Specify number of CFR Stages	1, 2, 3	1
N_ANTENNAS	The number of CFR antennas	1, 2, 3, 4	1
OUT_GAIN	Scalar gain to be applied to the output data of the last stage	true, false	false

N_CE	Dependent on FOLD_RATE and COETYPE
	Specify number of Clip engines
	(In order to maximum resource usage, only a subset of value can be selected from GUI)
		COETYPE = "real" {
		Fold_rate = 1
		{2, 4, 6, 8, 10, 12, 14, 16}
		Fold_rate = 2
		{4, 8, 12, 16}
		Fold_rate = 4
		{8, 16} for "real" only
		}
		COETYPE = "complex" {
		Fold_rate = 1
		{ 2,3,4,5,6,7,8}
		Fold_rate = 2
		{4,6,8}
		Fold_rate = 4
		{8,16}
		}
			4

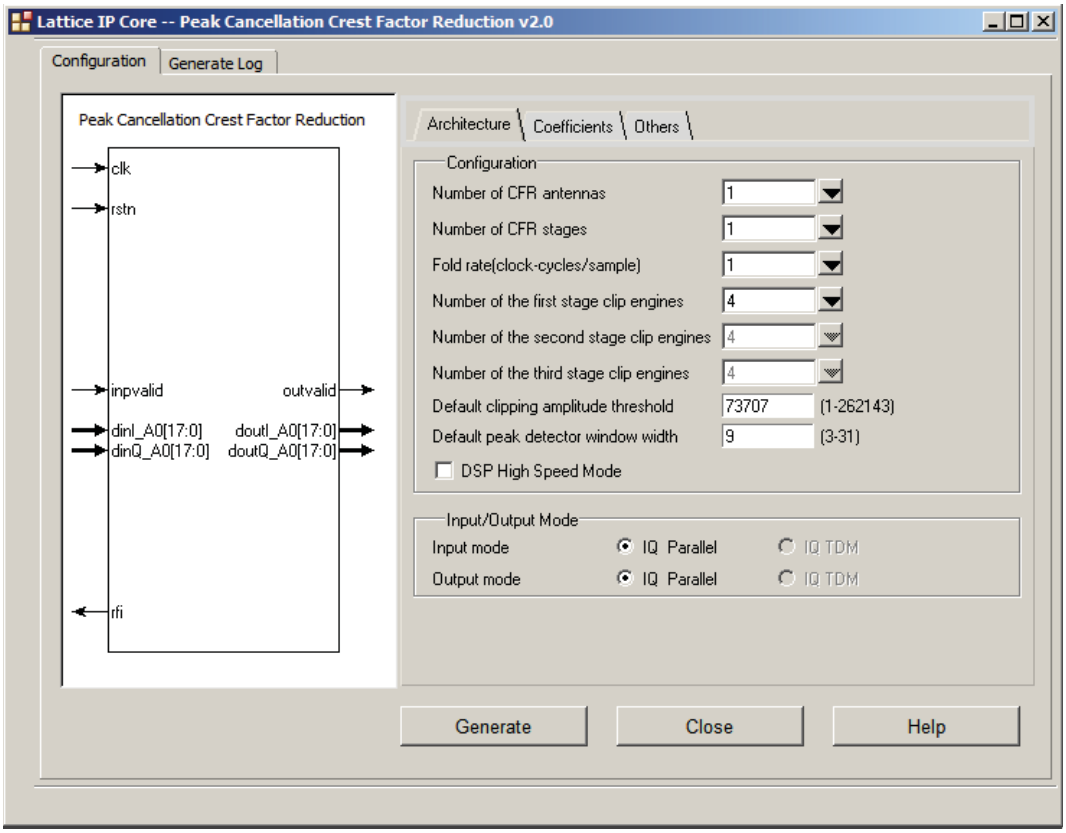
DYNAMIC_TH	Enable dynamic clipping amplitude threshold	true, false	false
DEFAULT_TH	Specify the default clipping amplitude threshold value	(1-262143)	73707
DYNAMIC_SK	Enable dynamic peak detector window width	true, false	false
DEFAULT_SK	Specify the default peak detector window width	(3, 31)	9
HIGH_SPEED	DSP high-speed mode	true, false	false
FOLD_RATE	Independent Specify fold rate of the core	1, 2, 4	1
DINWIDTH	Independent Specify data input width	11-18	18
CWIDTH	Independent Specify coefficient data width	11-18	18
DOUTWIDTH	Independent Specify output data width	11-18	18

Parameter	Dependency and Description	Range	Default
INPUT_MODE	Dependent on FOLD_RATE Specify input data mode	FOLD_RATE = 1 {iq_parallel} FOLD_RATE =2/4 {iq_parallel, iq_tdm}	iq_parallel
OUTPUT_MODE	Dependent on FOLD_RATE Specify output data mode	FOLD_RATE = 1 {iq_parallel} FOLD_RATE =2/4 {iq_parallel, iq_tdm}	iq_parallel
MAX_FILTER_LENGTH	Specify peak cancellation pulse length	{127, 255, 511}	255
COE_SETS	Specify the number of coefficient sets	(1-4)	1
COETYPE	Independent Specify coefficient type	{real, complex}	real
COEFILE_0	Specify the coefficients file for coefficients memory bank 0		none
COEFILE_1	Specify the coefficients file for coefficients memory bank 1		none
COEFILE_2	Specify the coefficients file for coefficients memory bank 2		none
COEFILE_3	Specify the coefficients file for coefficients memory bank 3		none

Architecture Tab

The Architecture tab provides settings for PC-CFR configuration and Input/Output mode options.

Figure 3-1. Architecture Tab



Number of CFR antennas
This setting specifies the number of CFR antennas. all the antennas are processing in parallel.

Number of CFR stages
Typically two stages are needed for a 2 to 3 dB PAR reduction in common applications. But for some tough cases, three stages may be required. For some signals requiring lower PAR reduction, such as in WiMAX systems, one CFR stage is sufficient.

Fold rate (clock-cycles/samples)
This setting specifies the data-rate in clocks per sample where the core will expect a new sample every n clock cycles based on the value specified. This enables the core to save DSP resources.

Number of clip engine for each stage
This setting specifies the number of clip engines for each stage independently. If large number of clip engines is selected, the CFR core can clip more peaks resulting in larger PAR reduction. Correspondingly, more DSP resources will be used.

Default clipping amplitude threshold
This setting specifies the default value of clipping amplitude threshold of CFR core. The threshold value is closely related to the input data property and target PAR reduction. It can be dynamically updated through the input control port to achieve a expected PAR.

Default peak detector window width
This setting specifies the default value of peak detector window width of CFR core. The peak detector window width value is closely related to the input data property. Normally it is fixed value, if necessary, it can be dynamically updated through the input control port.

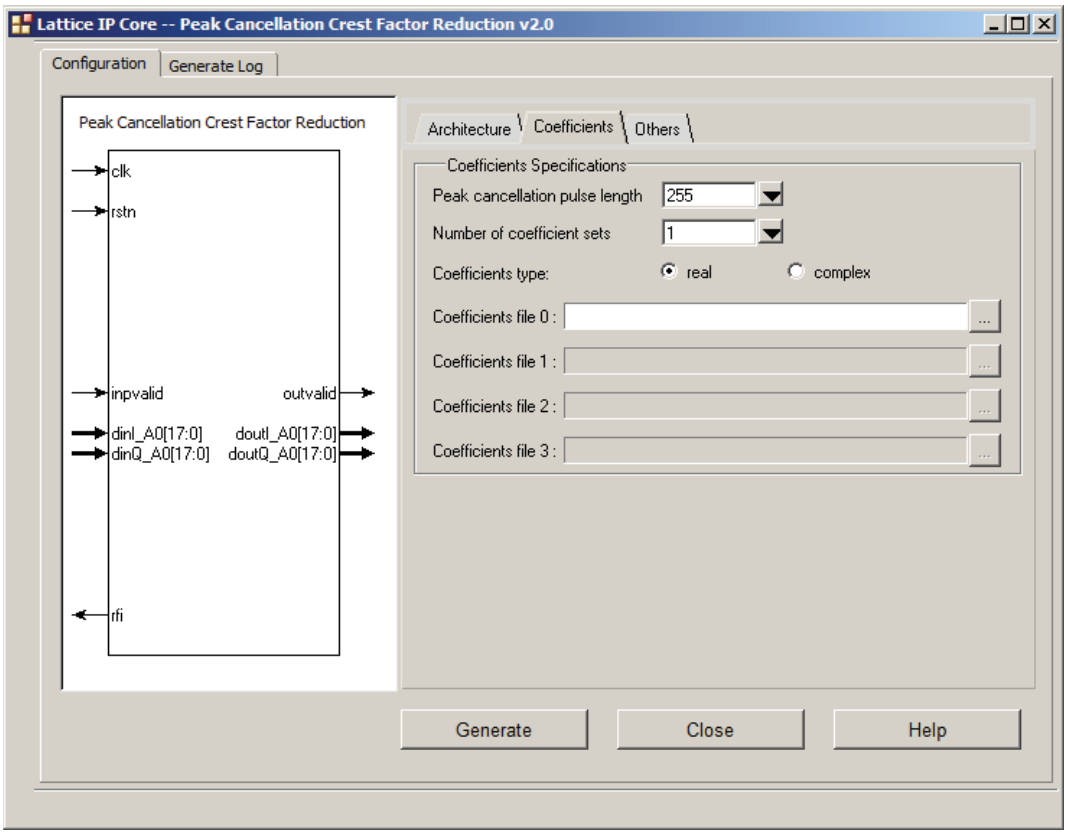
DSP High Speed Mode
The DSP High Speed Mode checkbox determines whether DSP high speed mode structure to save DSP resource, only for ECP5 devices.

Input Mode
This setting specifies input data mode. IQ TDM is available only for FOLD_RATE being 2 or 4.

Output Mode
This setting specifies output data mode. IQ TDM is available only for FOLD_RATE being 2 or 4.

Coefficients Tab

Figure 3-2. Coefficient Tab



Peak Cancellation Pulse Length
This setting specifies the number of peak cancellation pulse coefficients. The coefficients must be odd symmetric.

Number of coefficient sets
This setting specifies the number of coefficient sets for dynamic selecting. The maximum value is 4.

Coefficients file 0
This setting specifies the coefficient file for internal coefficient memory bank 0.

Coefficients file 1

This setting specifies the coefficient file for internal coefficient memory bank 1. This setting is only available when the number of coefficient sets is greater than 1.

Coefficients file 2
This setting specifies the coefficient file for internal coefficient memory bank 2. This setting is only available when the number of coefficient sets is greater than 2.

Coefficients file 3
This setting specifies the coefficient file for internal coefficient memory bank 3. This setting is only available when the number of coefficient sets is 4.

When the number of coefficient sets is greater than 1, the input control port *coeffs_select* is available for the CFR core.

When coeffs_select is set to 0, coefficient memory bank 0 (coefficient file 0) is used for peak cancellation.

When coeffs_select is set to 1, coefficient memory bank 1 (coefficient file 1) is used for peak cancellation.

When coeffs_select is set to 2, coefficient memory bank 2 (coefficient file 2) is used for peak cancellation.

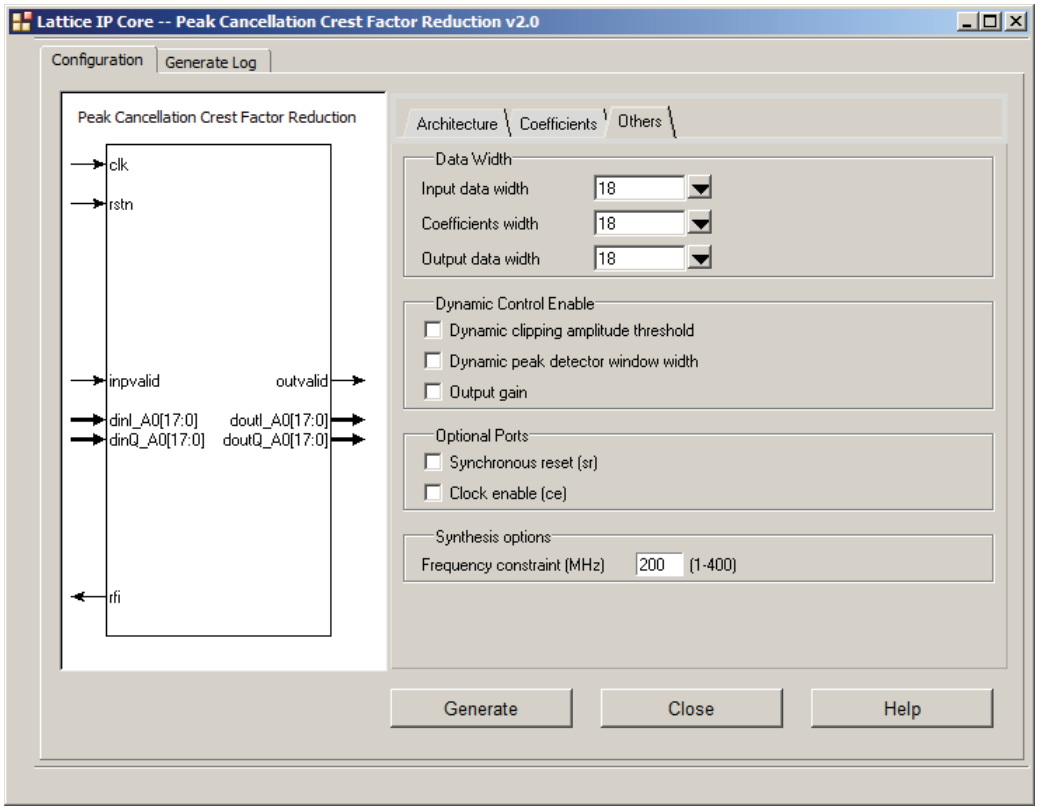
When coeffs_select is set to 3, coefficient memory bank 3 (coefficient file 3) is used for peak cancellation.

If no coefficients file is set, a random value is used for the CFR core evaluation only.

Others Tab

The Others tab provides settings for input/output data and coefficient widths, dynamic parameter control and output gain. The others tab of the GUI dialog box is shown in Figure 3-3 and some of the parameters are explained below.

Figure 3-3. Others Tab



Input data width

This setting sets the bit width of the incoming data values and may vary between 11 and 18, inclusive.

Coefficients width

This setting sets the bit width of the coefficients and may vary between 11 and 18, inclusive.

Output data width

This setting sets the output data bit width and may vary between 11 and 18, inclusive.

Dynamic clipping amplitude threshold

This checkbox determines whether CFR updates clipping amplitude threshold dynamically.

Dynamic peak detector window width

This checkbox determines whether CFR updates peak detector window width dynamically.

Output gain

This checkbox determines whether the scalar gain defined by the input port *out_gain* is applied to the output data after the last stage. when dynamic output gain is enabled, a 16-bit input port *out_gain* is available on the CFR core interface.

Synchronous reset

This checkbox determines whether the core has a synchronous reset port.

Clock enable

This checkbox determines whether the core has a clock enable port.

Frequency constraint (MHz)

This edit box determines the clock frequency in MHz for core generation synthesis. This is only available in core generation process.

IP Core Generation

This section provides information on how to generate the Lattice CFR IP core using the Diamond IPexpress tool and Clarity Designer tool, and how to include the core in a top-level design.

Licensing the IP Core

An IP core- and device-specific license is required to enable full, unrestricted use of the Lattice CFR IP core in a complete, top-level design. Instructions on how to obtain licenses for Lattice IP cores are given at www.latticesemi.com/Products/DesignSoftwareAndIP.aspx.

Users may download and generate the Lattice CFR IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The CFR IP core supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core which operate in hardware for a limited time (approximately four hours) without requiring an IP license. See Hardware Evaluation for further details. However, a license is required to enable timing simulation, to open the design in Diamond and to generate bitstreams that do not include the hardware evaluation timeout limitation.

IP Core Generation in IPexpress

Getting Started

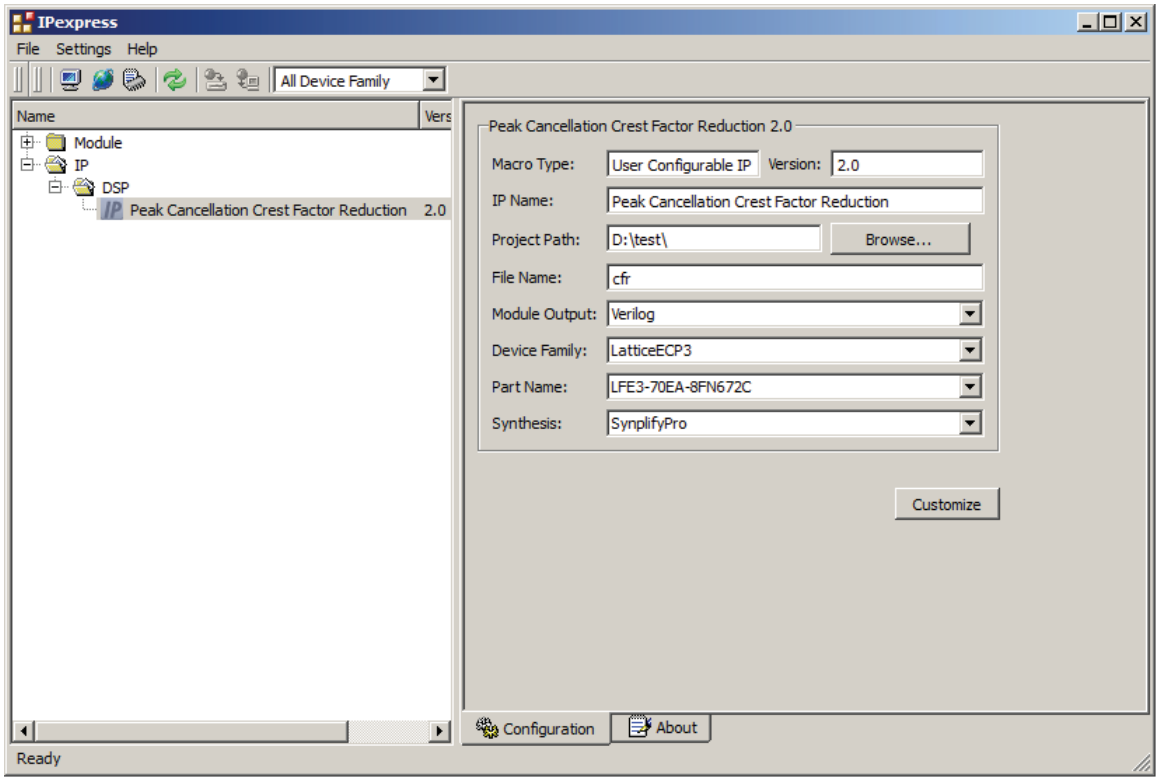
The CFR IP core is available for download from the Lattice IP Server using the IPexpress tool. The IP files are automatically installed using ispUPDATE technology in any user-specified directory. After the IP core has been installed, the IP core will be available in the IPexpress GUI dialog box shown in Figure 4-1. To generate a specific IP core configuration, the user specifies:

- **Project Path** – Path to the directory where the generated IP files will be located.
- **File Name** – "username" designation given to the generated IP core and corresponding folders and files.
- **(Diamond) Module Output** – Verilog or VHDL.
- **Device Family** – Device family to which IP is to be targeted. Only families that support the particular IP core are listed.
- **Part Name** – Specific targeted part within the selected device family.

Configuring CFR Core in IPexpress

The CFR configuration GUI is accessed via the Diamond IPexpress tool, and provides an interface for setting the desired parameters and invoking the IP core generator. The start-up IPexpress page allows the user to select the IP to be generated, project directory, user-designated module name, design entry type and target device. The File Name is used as the username in the core generation. The CFR IP core is found under **IP > DSP** as shown below.

Figure 4-1. IPexpress Dialog Box

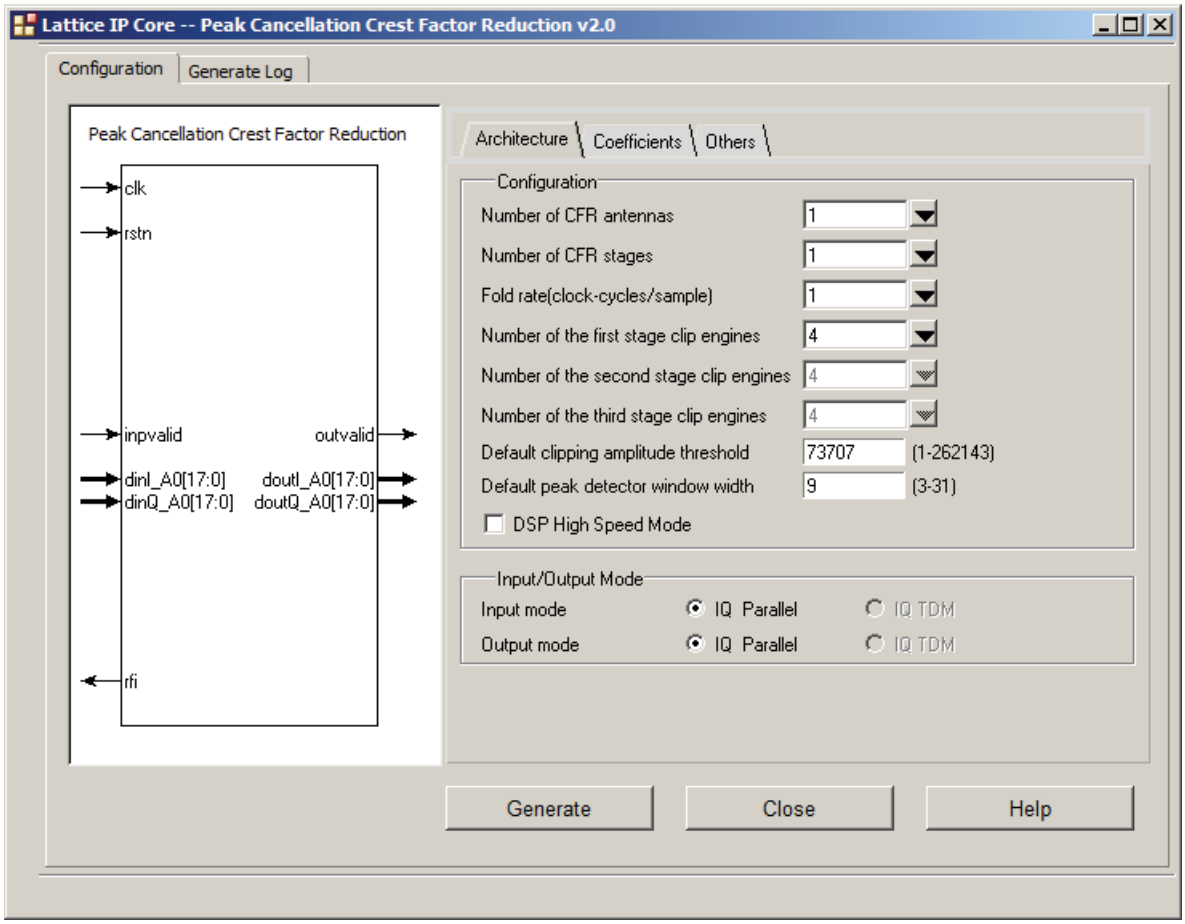


Important Note: File Name cannot be "CFR_core," as this name has been used in the internal design of the core.

Note that if the IPexpress tool is called from within an existing project, Project Path, Module Output, Device Family and Part Name default to the specified project parameters. Refer to the IPexpress tool online help for further information.

To create a custom configuration, the user clicks the **Customize** button in the IPexpress tool dialog box to display the CFR IP core Configuration GUI, as shown in Figure 4-2. From this dialog box, the user can select the IP parameter options specific to their application. Refer to Parameter Settings for more information on CFR IP core parameter settings.

Figure 4-2. Configuration GUI



IPexpress-Created Files and Top Level Directory Structure

When the user clicks the **Generate** button in the IP Configuration dialog box, the IP core and supporting files are generated in the specified Project Path directory. The directory structure of the generated files is shown in Figure 4-3. This example shows the directory structure generated with the CFR for a LatticeECP3 device.

Figure 4-3. CFR IP Core Directory Structure

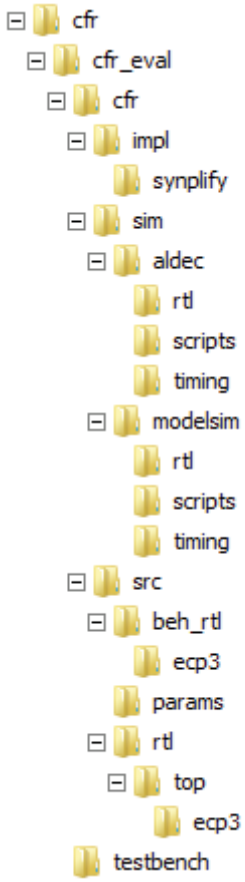


Table 4-1 provides a list of key files and directories created by the IPexpress tool and how they are used. The IPexpress tool creates several files that are used throughout the design cycle. The names of most of the created files are customized to the user's module name specified in the IPexpress tool.

Table 4-1. File List

File	Description
<username>.lpc	This file contains the IPexpress tool options used to recreate or modify the core in the IPexpress tool.
<username>.ipx	The IPX file holds references to all of the elements of an IP or module after it is generated from the IPexpress tool. The file is used to bring in the appropriate files during the design implementation and analysis. It is also used to re-load parameter settings into the IP/module generation GUI when an IP/module is being re-generated.
<username>.ngo	This file provides the synthesized IP core.
<username>_bb.v	This file provides the synthesis black box for the user's synthesis.
<username>_inst.v	This file provides an instance template for CFR IP core.
<username>_beh.v	This file provides the front-end simulation library for CFR IP core.

Table 4-2 provides a list of key additional files providing IP core generation status information and command line generation capability are generated in the user's project directory.

Table 4-2. Additional Files

File	Description
<username>_generate.tcl	This file is created when the GUI **Generate** button is pushed. This file may be run from command line.
<username>_generate.log	This is the synthesis and map log file.
<username>_gen.log	This is the IPexpress IP generation log file.

Instantiating the Core

The generated CFR IP core package includes black-box (<username>_bb.v) and instance (<username>_inst.v) templates that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file that can be used as an instantiation template for the IP core is provided in \<project_dir>\frame_buffer_eval\<username>\src\rtl\top. Users may also use this top-level reference as the starting template for the top-level for their complete design.

Running Functional Simulation

Simulation support for the CFR IP core is provided for Aldec Active-HDL (Verilog and VHDL) simulator, Mentor Graphics ModelSim simulator. The functional simulation includes a configuration- specific behavioral model of the CFR IP core. The test bench sources stimulus to the core, and monitors output from the core. The generated IP core package includes the configuration-specific behavior model (<username>_beh.v) for functional simulation in the “Project Path” root directory. The simulation scripts supporting ModelSim evaluation simulation is provided in \<project_dir>\CFR_eval\<username>\sim\modelsim\scripts. The simulation script supporting Aldec evaluation simulation is provided in \<project_dir>\CFR_eval\<username>\sim\aldec\scripts. Both Modelsim and Aldec simulation is supported via test bench files provided in \<project_dir>\CFR_eval\testbench. Models required for simulation are provided in the corresponding \models folder. Users may run the Aldec evaluation simulation by doing the following:

1. Open Active-HDL.
2. Under the **Tools** tab, select **Execute Macro**.
3. Browse to folder **\<project_dir>\CFR_eval\<username>\sim\aldec\scripts** and execute one of the “do” scripts shown.

Users may run the ModelSim evaluation simulation by doing the following:

1. Open ModelSim.
2. Under the File tab, select **Change Directory** and choose the folder **<project_dir>\frame_buffer_eval\<username>\sim\modelsim\scripts**.
3. Under the **Tools** tab, select **Execute Macro** and execute **..\scripts\<username>_rtl_se.do**.

*Note: When the simulation is complete, a pop-up window will appear asking “Are you sure you want to finish?” Choose **No** to analyze the results. Choosing **Yes** closes ModelSim.*

Synthesizing and Implementing the Core in a Top-Level Design

Synthesis support for the CFR IP core is provided for Synopsys Synplify. The CFR IP core itself is synthesized and is provided in NGO format when the core is generated in IPexpress. Users may synthesize the core in their own top-level design by instantiating the core in their top level as described previously and then synthesizing the entire design with Synplify.

The top-level file <username>_eval_top.v provided in **\<project_dir>\CFR_eval\<username>\src\top** supports the ability to implement CFR core in isolation. Push-button implementation of this top-level design with Synplify is supported via the project files **<username>_eval.ldf(Diamond)** located in **\<project_dir>\CFR_eval\<username>\impl\synplify**.

To use this project file in Diamond:

1. Choose **File > Open > Project**.
2. Browse to **\<project_dir>\CFR_eval\<username>\impl\synplify** in the Open Project dialog box.
3. Select and open **<username>.ldf**. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the **Process** tab in the left-hand GUI window.
5. Implement the complete design via the standard Diamond GUI flow.

Hardware Evaluation

The CFR IP core supports the Lattice IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase of an IP license. It may also be used to evaluate the core in hardware in user-defined designs.

Enabling Hardware Evaluation in Diamond

Choose **Project > Active Strategy > Translate Design Settings**. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default.

Updating/Regenerating the IP Core

By regenerating an IP core with the IPexpress tool, you can modify any of its settings including device type, design entry method, and any of the options specific to the IP core. Regenerating can be done to modify an existing IP core or to create a new but similar one.

To regenerate an IP core:

1. In IPexpress, click the **Regenerate** button.
2. In the Regenerate view of IPexpress, choose the IPX source file of the module or IP you wish to regenerate.
3. IPexpress shows the current settings for the module or IP core in the Source box. Make your new settings in the **Target** box.
4. If you want to generate a new set of files in a new location, set the new location in the **IPX Target File** box. The base of the file name will be the base of all the new file names. The IPX Target File must end with an .ipx extension.
5. Click **Regenerate**. The module's dialog box opens showing the current option settings.
6. In the dialog box, choose the desired options. To get information about the options, click **Help**. Also, check the **About** tab in IPexpress for links to technical notes and user's guides. The IP core may come with additional

information. As the options change, the schematic diagram of the module changes to show the I/O and the device resources the module will need.

7. To import the module into your project, if it's not already there, select Import **IPX to Diamond Project** (not available in stand-alone mode).
8. Click **Generate**.
9. Check the **Generate Log** tab to check for warnings and error messages.
10. Click **Close**.

The IPexpress package file (.ipx) supported by Diamond holds references to all of the elements of the generated IP core required to support simulation, synthesis and implementation. The IP core may be included in a user's design by importing the .ipx file to the associated Diamond project. To change the option settings of a module or IP that is already in a design project, double-click the module's .ipx file in the File List view. This opens IPexpress and the module's dialog box showing the current option settings. Then go to step 6 above.

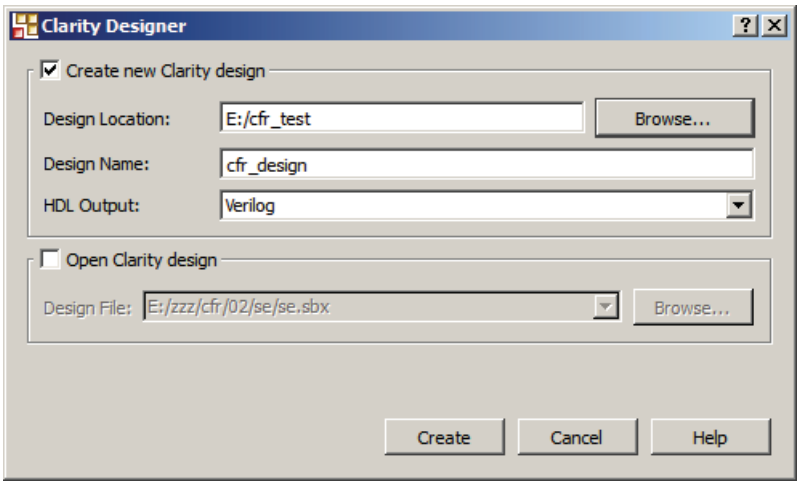
IP Core Generation in Clarity Designer

Getting Started

The first step in generating an IP Core in Clarity Designer is to start a project in Diamond software with the ECP5 device. Click the **Clarity Designer** button to open the Clarity Designer tool.

- **Create new Clarity design** – Choose to create a new Clarity Design project directory in which the IP core will be generated.
- **Design Location** – Clarity Design project directory Path.
- **Design Name** – Clarity Design project name.
- **HDL Output** – Hardware Description Language Output Format (Verilog or VHDL).
- **Open Clarity design** – Open an existing Clarity Design project.
- **Design File** – Name of existing Clarity Design project file with .sbx extension.

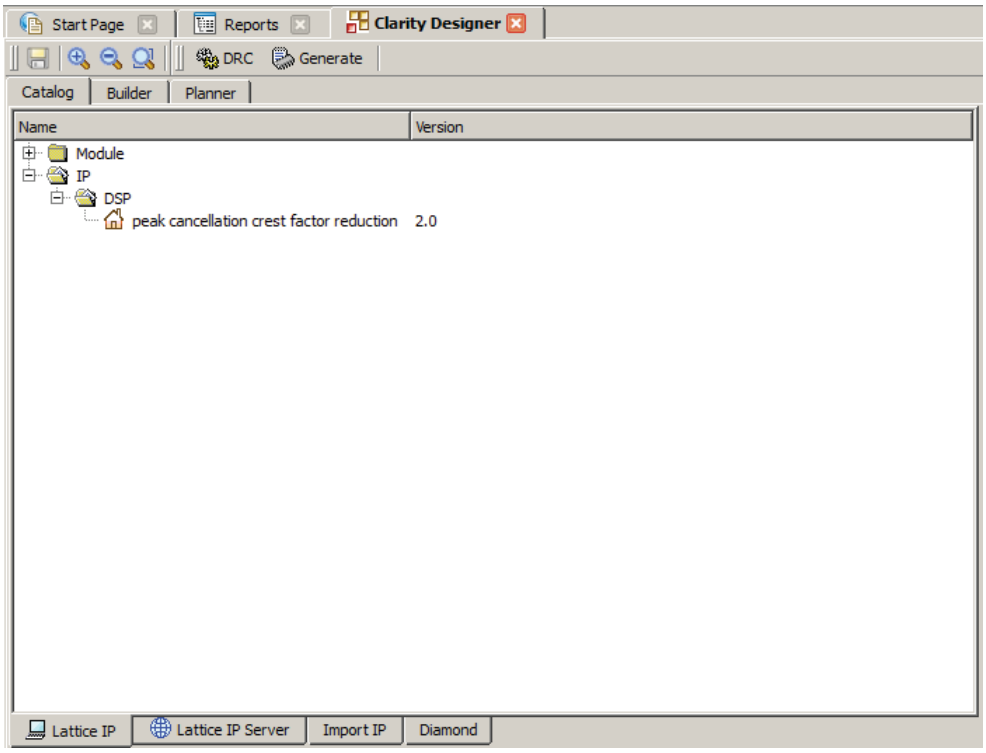
Figure 4-4. Clarity Designer Tool Dialog Box



As shown in Figure 4-4, you can create a new design or open an existed one. Specify the Design Location, Design Name and HDL Output format. Click **Create** to open the Clarity Designer main GUI window.

The CFR IP core is available for download from the Lattice IP Server using the Clarity Designer tool. The IP files are automatically installed using ispUPDATE technology in any customer-specified directory. After the IP core has been installed, the IP core becomes available in the Clarity Designer tool as shown in Figure 4-5.

Figure 4-5. Clarity Designer IP Window

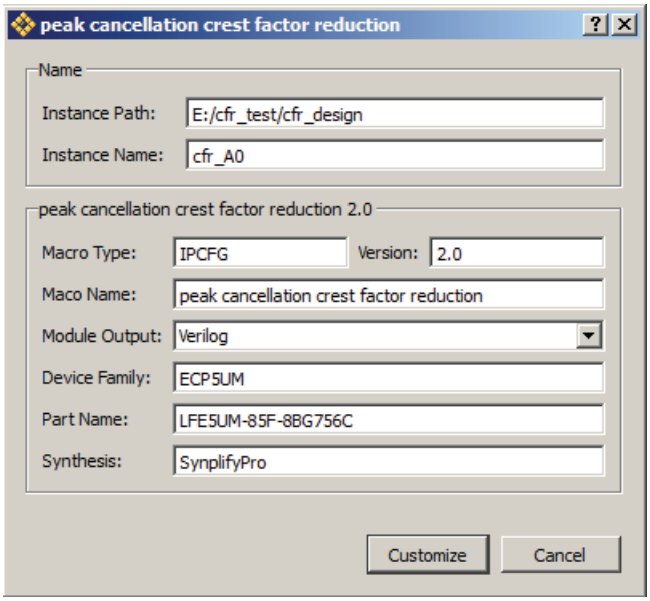


You can generate IP core configuration by double-clicking the IP name in the Catalog tab.

In the dialog box shown Figure 4-6, specify the following:

- **Instance Path** – Path to the directory where the generated IP files will be located.
- **Instance Name** – “username” designation given to the generated IP core and corresponding folders and files.
- **Module Output** – Verilog or VHDL.
- **Module Output** – Verilog HDL or VHDL.
- **Device Family** – Device family to which IP is to be targeted.

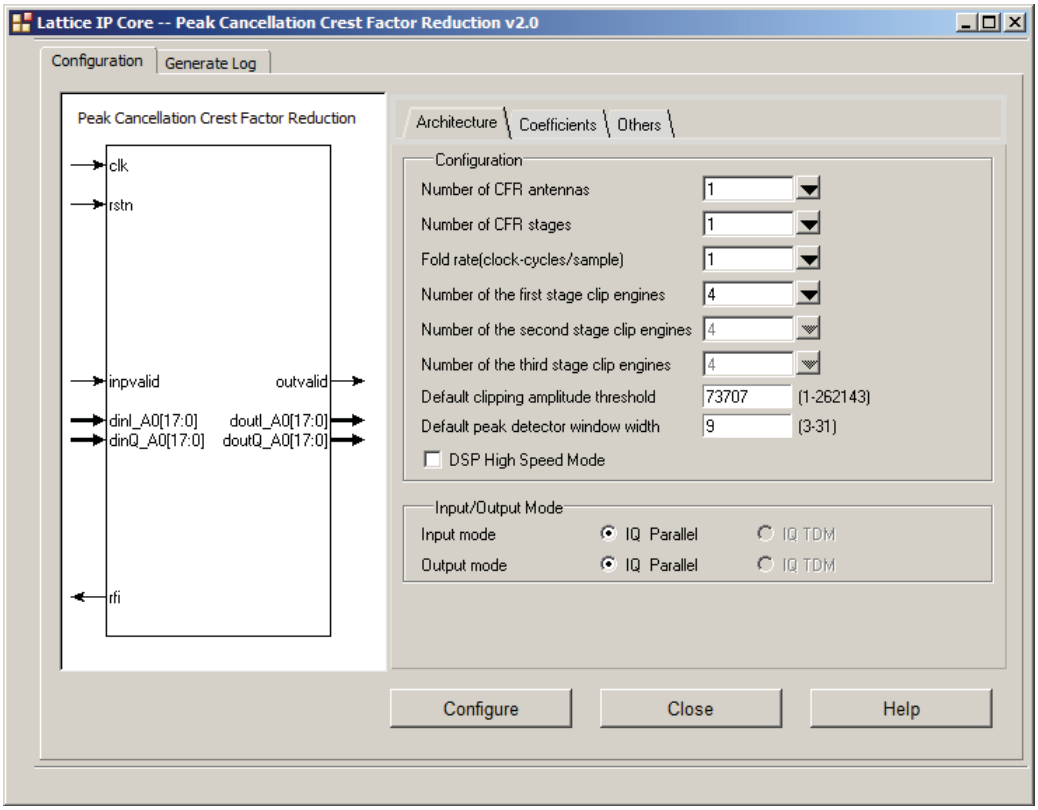
Figure 4-6. IP Generation Dialog Box



Note that if the Clarity Designer tool is called from within an existing project, Design Location, Device Family and Part Name default to the specified project parameters. Refer to the Clarity Designer tool online help for further information.

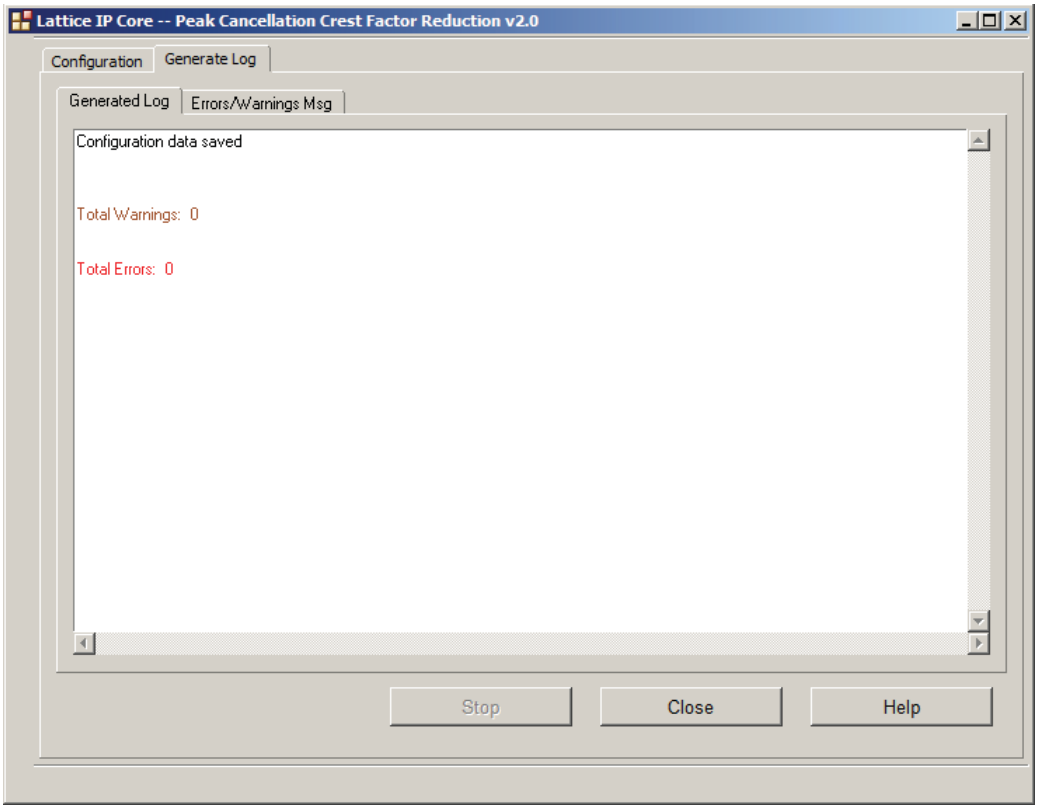
To create a custom configuration, click the **Customize** button in the IP generation dialog box to display the IP core Configuration GUI, as shown in Figure 4-7. From this dialog box, you can select the IP parameter options specific to their application. Refer to Parameter Settings for more information on the IP core parameter settings.

Figure 4-7. IP Configuration GUI in Clarity Designer



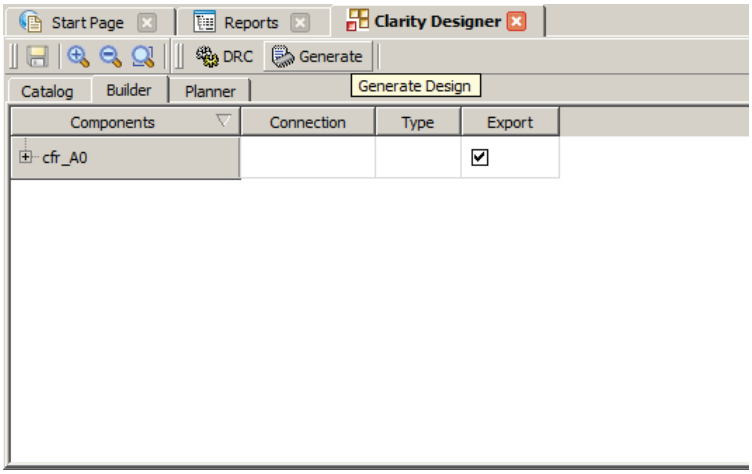
After setting the parameters, click **Configure**. The Generate Log tab, shown in Figure 4-8, displays logs, errors and warnings.

Figure 4-8. Clarity Designer Generate Log Tab



Click the **Close** button to generate the IP case and supporting files. The *cfr_A0* IP case is displayed in the Clarity Designer Builder tab as shown in Figure 4-9.

Figure 4-9. Clarity Designer Builder Tab



IP Core Implementation

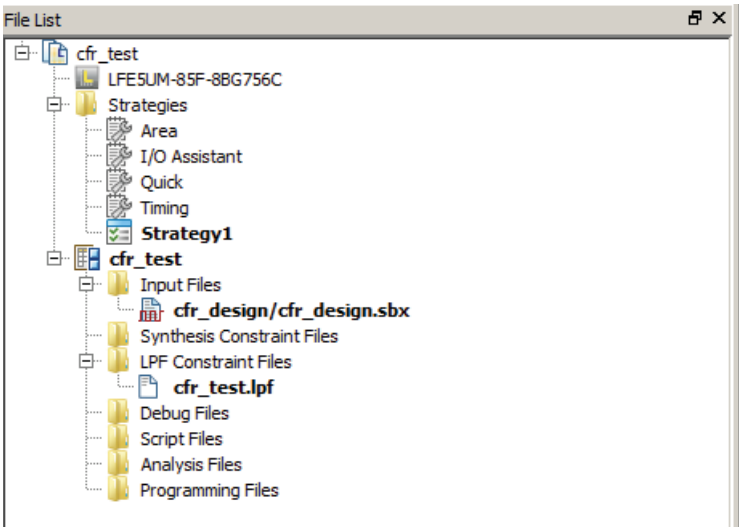
You may generate other IP cases into Clarity Designer case cfr_design, such as *cfr_A1* for example.

Click the **Generate** button in the Clarity Designer to generate the Clarity Designer file (.sbx) and the Clarity Designer case cfr_design top file. All the related IP cases are instantiated in the Clarity Designer case top file. You can use the Clarity Designer case top module to implement using CFR IP in the design.

Clarity Designer (.sbx) files can be used in your design projects. A key difference between IPexpress generated files and Clarity Designer generated files is that the latter may contain not only single block but multiple modules or IP blocks and may represent a subsystem.

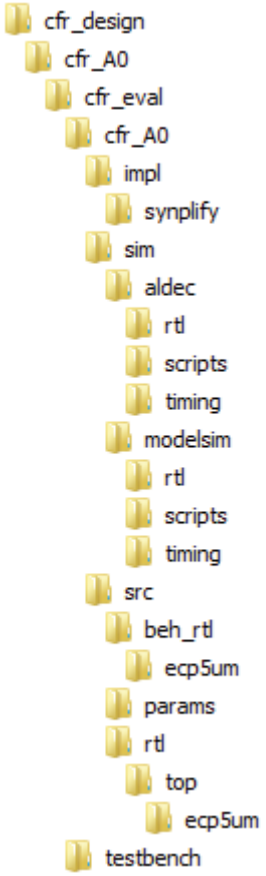
After the Generate step is completed, the ".sbx" file is automatically added to current Diamond Project Input Files list as shown in Figure 4-10.

Figure 4-10. Project File List in Diamond



The IP core and supporting files are generated in the specified Project Path directory. An example of the directory structure of the generated files is shown in Figure 4-11.

Figure 4-11. ECP5 CFR IP Core Directory Structure



The list of key files and directories created by the IP and how they are used, please refer Table 4-1 and Table 4-2.

For information on functional simulation, please refer to Running Functional Simulation.

For information on synthesizing and implementing the core, please refer to Synthesizing and Implementing the Core in a Top-Level Design.

Hardware Evaluation

The CFR IP core supports the Lattice IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase of an IP license. It may also be used to evaluate the core in hardware in user-defined designs.

Enabling Hardware Evaluation in Diamond

Choose **Project > Active Strategy > Translate Design Settings**. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default.

Regenerating an IP Core in Clarity Designer Tool

To regenerate an IP core in Clarity Designer:

1. In the Clarity Designer Builder window, right-click on the existing IP instance and choose **Config**.
2. In the dialog box, choose the desired option settings.
3. Click **Configure**.

Recreating an IP Core in Clarity Designer Tool

To recreate an IP core in Clarity Designer:

1. In the Clarity Designer Catalog window, click the **Import IP** tab.
2. In the Import IP tab, choose the existing IPX/LPC source file of the module or IP to regenerate.
3. Specify the instance name in Target Instance. Note that this instance name should not be the same as any of the existing IP instances in the current Clarity Design project.
4. Click **Import**. The module's dialog box opens showing the option settings.
5. In the dialog box, choose the desired options.
6. Click **Configure**.

Chapter 7:

Support Resources

This chapter contains information about Lattice Technical Support, additional references, and document revision history.

Lattice Technical Support

There are a number of ways to receive technical support.

E-mail Support

techsupport@latticesemi.com

Local Support

Contact your nearest Lattice sales office.

Internet

www.latticesemi.com

IEEE

IEEE offers publications and technology standards on its web site at http://www.ieee.org.

References

Complete details on the CPRI IP core low latency configuration are given in:

- IPUG74, CPRI IP Core Low Latency Variation Design Considerations User's Guide

LatticeECP3

- HB1009, LatticeECP3 Family Handbook

ECP5

- HB1012, ECP5 Family Handbook

References

- John G.Proakis and Dimitris K.Manolakis, **Digital Signal Processing (4th Edition)**, Prentice Hall, April 2006.
- Olli Vaananen, **Digital Modulators with CFR Techniques**, Ph.D. Thesis, Helsinki University of Technology, 2006.
- HB1009, LatticeECP3 Family Handbook.

Revision History

Date	Document Version	IP Core Version	Change Summary
June 2013	1.0	1.1	Initial release.
May 2013	1.1	2.0	Interface update and resource optimization.
			Added support for DSP high-speed mode.
			Added support for ECP5 FPGA family and Clarity Designer tool.
			Updated Technical Support information.

Resource Utilization

This appendix gives resource utilization information for Lattice FPGAs using the CFR IP core.

IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the Diamond design tools. Details regarding the usage of IPexpress can be found in the IPexpress and Diamond help systems. For more information on the Diamond design tools, visit the Lattice web site at www.latticesemi.com/software.

LatticeECP3 Devices

Table A-1. Performance and Resource Utilization[1]

User-Configurable Mode	Data Width	Coefficient Width	High-speed Mode	Regis-ters	LUT4s	Slices	EBR	MULT18X 18s	fMAX (MHz)
1Antenna, 1Stage, 8Clip engines	18	18	false	2750	2643	2029	5	18	259
1Antenna, 2Stages, 10Clip engines	18	18	false	4321	4493	3313	7	24	263
1Antenna, 3Stages, 14Clip engines	18	18	false	6338	6599	4870	10	34	266

1. Performance and utilization data are generated targeting a LFE3-70EA-9FN672C device using Lattice Diamond 3.2 and Synplify Pro I-2013.09L software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP3 family.

Ordering Part Number

The Ordering Part Number (OPN) for the CFR IP core targeting LatticeECP3 devices is CFR-E3-U1.

ECP5 Devices

Table A-2. Performance and Resource Utilization[1]

User-Configurable Mode	Data Width	Coefficient Width	High-speed Mode	Regis-ters	LUT4s	Slices	EBR	MULT18X 18s	fMAX (MHz)
1Antenna, 1Stage, 8Clip engines	18	18	true	2988	2704	2193	5	10	223
1Antenna, 2Stages, 10Clip engines	18	18	true	4672	4537	3535	7	14	223
1Antenna, 3Stages, 14Clip engines	18	18	true	6800	6658	5166	10	20	223

1. Performance and utilization data are generated targeting a LFE5UM-85F-8MG756C device using Lattice Diamond 3.2 and Synplify Pro I-2013.09L software. Performance may vary when using a different software version or targeting a different device density or speed grade within the ECP5 family.

Ordering Part Number

The Ordering Part Number (OPN) for the CFR IP core targeting ECP5 devices is CFR-E5-U1.