Crest Factor Reduction for OFDMA Systems

November 2007, ver. 1.0 Application Note 475

Introduction

Crest factor reduction (CFR) is a technique for reducing the peak-to-average ratio (PAR) of an orthogonal frequency division multiplexing (OFDM) waveform. An OFDM signal is made up in the frequency domain as a set of orthogonal carriers that are each modulated by a constellation symbol. The main disadvantage of OFDM modulation is that the time domain representation approximates a Gaussian distribution and therefore exhibits large envelope variations. Power amplifiers usually only have a limited linear region, making it necessary to back off the amplifier so that the transmit signal never drives the amplifier into the non-linear region, which causes spectral regrowth and inefficient amplifier power.

By reducing the PAR of the input signal, you can achieve greater power efficiency from the amplifier or use a less expensive power amplifier with a more limited linear region. The CFR module reduces the PAR of the signal and ultimately reduces the overall basestation cost or power requirements.

CFR is ideal for implementation on Altera® FPGA families such as Stratix® III devices, where the capability for supporting high sampling frequencies, high throughput and low latency are essential. This document demonstrates a block-based constraint-driven design entry point that makes it easy for system engineers to develop optimal hardware that is scalable for a variety of operating scenarios.

The CFR reference design includes the following key features:

- Support for OFDM based systems, including Worldwide Interoperability for Microwave Access (WiMAX) and 3GPP Long Term Evolution (LTE)
- Support for additional functionality including orthogonal frequency division multiple access (OFDMA) symbol modulation, digital up conversion (4× interpolation) and cyclic prefix insertion
- Programmable mask for any environment
- Programmable error vector magnitude (EVM) budget
- Support for 10 and 20 MHz channel bandwidth, and is easy to modify for other channel bandwidths
- Up to 5dB PAR improvement

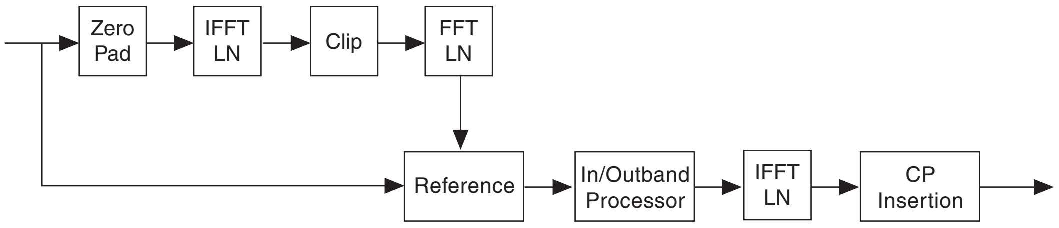
CFR Algorithm

There are many different techniques to achieve CFR. This document implements a technique that is well suited to wireless systems that use OFDM- or OFDMA-based modulation such as WiMAX and 3G LTE. The algorithm used is based on a modified version of the algorithm in *Constrained Clipping for Crest Factor Reduction in Multiple-user OFDM* [1]. The algorithm offers the following key advantages:

- Needs no receiver side modifications
- Guarantees to never violate the spectral mask
- Always meets the error vector magnitude specification
- Allows good PAR reductions

Figure 1 shows a block diagram of the necessary processing.

Figure 1. CFR Algorithm Block Diagram

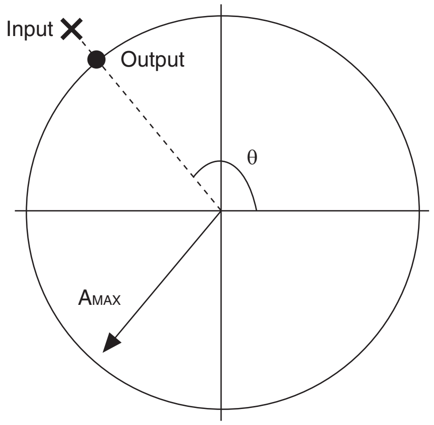


The system accepts data at the input as OFDM(A) symbols (of length N carriers) in the frequency domain. The first process is to upconvert the data by a factor $L = 4$ using perfect frequency domain interpolation (zero padding). The technique involves performing CFR at a higher sampling frequency because there is peak growth associated with upconversion. In addition, a higher sampling frequency spreads the non-linear distortions introduced by subsequent blocks across a larger bandwidth.

An inverse Fourier transform of length L times N is performed to generate a time domain representation of the OFDM(A) symbol. Next, a clipping operation constrains the envelope of the time domain signal to within the specified bounds. Constraint is achieved by calculating the magnitude of the complex samples. Where the samples exceed the threshold A_{MAX}, the magnitude of the samples is clipped to equal A_{MAX} while maintaining the original sample phase. This process is known as polar clipping. Polar clipping minimizes spectral regrowth better than the simpler cartesian clipping method.

Figure 2 shows the operation of the polar clipping block. All samples bounded by the unit circle scaled by A_{MAX} pass through the block unchanged, and those outside of the unit circle are scaled back onto the circle. Optimal operation of the CFR module requires that the value of A_{MAX} is set accordingly.

Figure 2. Polar Clipping in the Complex Plane



After clipping, the PAR of the signal is reduced, making it possible to transmit the new signal. However, the polar clipping results in distortion (and perhaps unrecoverable errors) in the constellation symbols. In addition, the out of band spectral components can exceed the spectral mask. Correcting distorted constellation symbols and constraining the out of band spectral energy to the spectral mask requires further processing.

To perform this additional processing, the time domain representation is converted back into the frequency domain using a forward Fourier transform. Each sample is analyzed. If the sample is associated with a constellation symbol (that is, an inband sample), the sample is compared with the perfect reference sample and corrected if necessary. If the sample is in the outband region, its magnitude is constrained to the spectral mask.

The purpose of the inband processor is to ensure that the overall EVM does not exceed a specified limit. The EVM is defined as the square root of the mean error power divided by the square of the maximum constellation magnitude (S_{MAX}).

(1)
$$EVM = \sqrt{\frac{\left(\frac{1}{N}\right)\sum_{k=0}^{N-1}|E_k|^2}{(S_{MAX})^2}}$$

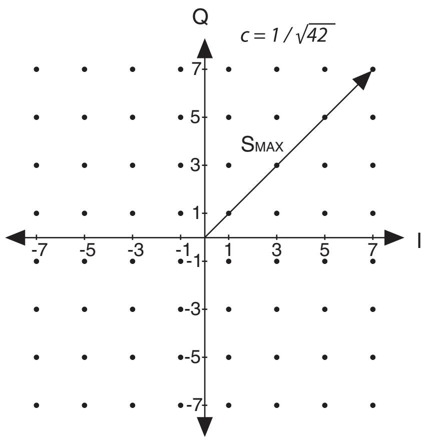
Table 1 shows S_{MAX} values for WiMAX and LTE.

Table 1. S_{MAX} Values for WiMAX and LTE

Highest Order Modulation Scheme	S_{MAX}
QPSK	1
16QAM	$\sqrt{\frac{18}{10}}$
64QAM	$\sqrt{\frac{98}{42}}$

Figure 3 shows how the S_{MAX} value is calculated for 64QAM.

Figure 3. S_{MAX} Calculation Example



(2)
$$S_{MAX} = \sqrt{\left(\frac{7}{\sqrt{42}}\right)^2 + \left(\frac{7}{\sqrt{42}}\right)^2} = \sqrt{\frac{98}{42}}$$

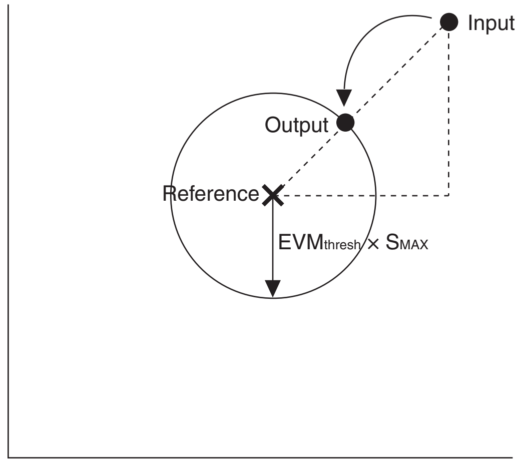
Despite the original algorithm [1] suggesting an optimal method for achieving the target output EVM, the hardware resources and latency required to implement such a large sorting network is not feasible. Instead, you can use a reduced complexity technique that requires minimal resources. Due to the statistical distribution of the errors introduced by the polar clipping block, the output PAR of the reduced complexity technique is almost identical to the output PAR of the original algorithm.

If the calculated error (E_k) power for a sample does not exceed the specified EVM threshold, simply output the clipped sample. If the error power for a sample is greater than or equal to the square of the product of the specified EVM threshold and S_{MAX}, output the reference signal plus a small error signal. This error signal has magnitude EVM threshold multiplied by S_{MAX} and a phase that is equal to the phase of the original error, namely:

$$EVM_{threshold} \times S_{MAX} \times e^{j\angle E_k} \quad (3)$$

Figure 4 shows an example where the magnitude of the error between the input sample and the reference sample is greater than the specified threshold and the resulting output sample.

Figure 4. Inband Processor Example



Finally, the corrected frequency domain representation of the symbol is converted back into the time domain for transmission. A cyclic prefix is applied to the stream before calling the digital predistortion and digital up conversion blocks.

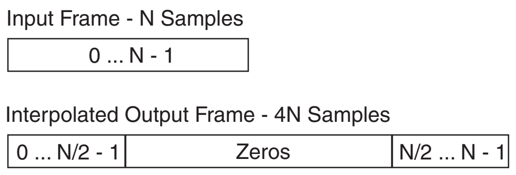
Hardware Architecture

This section summarizes the key processing blocks in the CFR design.

Zeropad

The zeropad block performs frequency domain interpolation. Figure 5 shows the interpolation is an operation that fills the input FFT frame with zeros.

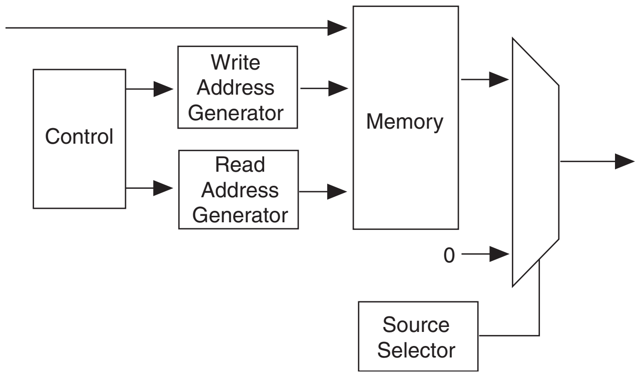
Figure 5. Frequency Domain Interpolation



In hardware, zero padding is implemented using a memory of size $N/2$. Data presented to the block is written straight to memory. All control logic in this block is controlled by the valid signal. The write address generator is a modulo $N/2$ counter. For the first $N/2$ cycles, a read address generator is also configured as a modulo $N/2$ counter. Data samples are read straight out and passed on to the next block.

After $N/2$ cycles, zeros are passed to the output of the zero pad block while the incoming data continues to write to the memory. A sequence generator determines when sufficient zeros have been passed to the output and configures a multiplexer to redirect samples from the memory to the output. Figure 6 summarizes these actions.

Figure 6. Zeropad Block Diagram



Fast Fourier Transform

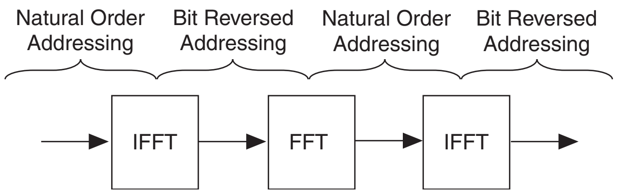
The fast Fourier transform (FFT) blocks are based on the variable streaming architecture [2] shared by the Altera FFT MegaCore function. The variable streaming architecture uses a radix-2^2 single delay feedback architecture, which is a fully pipelined architecture. The radix-2^2 algorithm has the same multiplicative complexity of a fully pipelined radix-4 architecture, however the butterfly unit retains a radix-2 architecture.

For details on the Altera FFT MegaCore function, refer to the *FFT MegaCore Function User Guide*.

With CFR, restricting the natural bitgrowth allows for reduction in the resource requirements. Mathematical analysis of the architecture shows that the worst case bitgrowth for the algorithm is 2.5 bits per stage. For example, for a 4K FFT with 6 stages, a 16 bit input results in 31 bits at the output. Because the distribution of the input data and resulting output data is known, the algorithm can determine the actual worst case bitgrowth for OFDMA stimulus. When the internal bitgrowth has reached this ceiling, you can restrict any further word length extension through the arithmetic elements. The result is a savings in overall logic and increased frequency of operation.

The second advantage of using variable streaming architecture is that you can implement blocks that accept data in either natural or bit reversed order. At the output of each FFT, the ordering scheme is the opposite of the ordering scheme associated with the input. The disadvantages of performing bit reversal at the input/output of the FFT are the 2*N* complex words of memory and the latency penalty of *N* clock cycles. But with CFR, you can cascade multiple FFTs to avoid performing bit reversal, assuming the other algorithms are carefully designed to be compatible. Figure 7 shows this approach, which saves significant memory and reduces the overall latency.

Figure 7. Natural and Bit Reversed Addressing



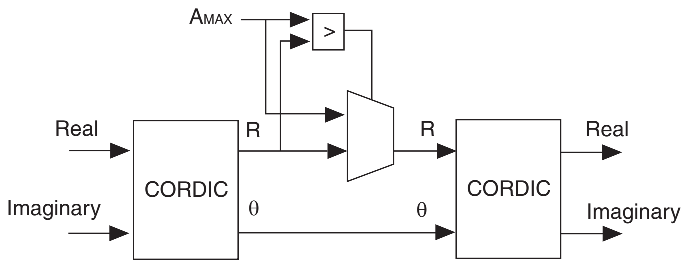
The third advantage is that the size of the twiddle memory look up tables is reduced by using symmetry and redundancy in the reference signal. However, there is a trade off between memory and logic when the size of the twiddle table is small; sometimes it is more efficient to store all values rather than use redundancy because of the quantization of the memory in the device.

Polar Clipping

The polar clipping block is achieved very easily in hardware, as shown by Figure 8. This block takes advantage of the coordinate rotation digital computer (CORDIC) algorithm [3][4]. CORDIC is an iterative algorithm that performs complex and trigonometric operations. For example, in the CFR design the CORDIC algorithm converts cartesian coordinates to polar coordinates and vice versa. The CORDIC algorithm works well in FPGA implementations because it can be fully unrolled.

A CORDIC block is used in vectoring mode to convert the cartesian input to a magnitude / phase representation. The magnitude is compared with and constrained to the A_{MAX} threshold. The modified magnitude value is passed on with the original phase to a second CORDIC block that is configured in rotation mode to convert the polar representation of the number back into cartesian coordinates.

Figure 8. Polar Clipping Hardware Architecture



Unfortunately, the CORDIC algorithm introduces a scaling factor to the magnitude and real/imaginary outputs. The diagram does not show the constant coefficient multipliers that remove this effect.

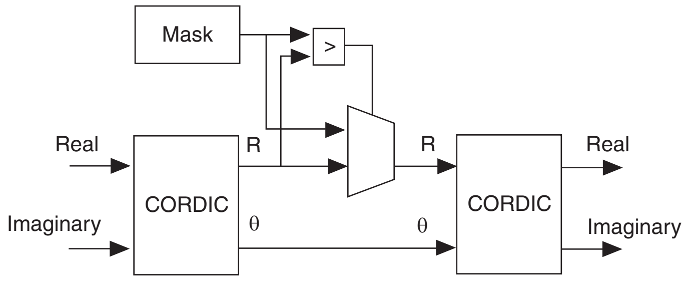
Inband and Outband Processors

The inband and outband processors require similar hardware resources. Because each sample is classified as an inband sample or an outband sample, the processing implementation uses shared hardware resources and an appropriate network of multiplexers to achieve the processing in the most efficient way possible.

Outband Processor

The outband processor is similar to the polar clipping block. Each frequency sample in the out of band region is compared with the magnitude of the spectral mask at that point. If the sample exceeds the magnitude, the sample is reduced down to the threshold. Hence, the only difference between the outband processor and the polar clipping block is that the magnitude threshold is variable and implemented as a memory that contains the spectral mask magnitude for each point. Figure 9 shows the outband processor architecture.

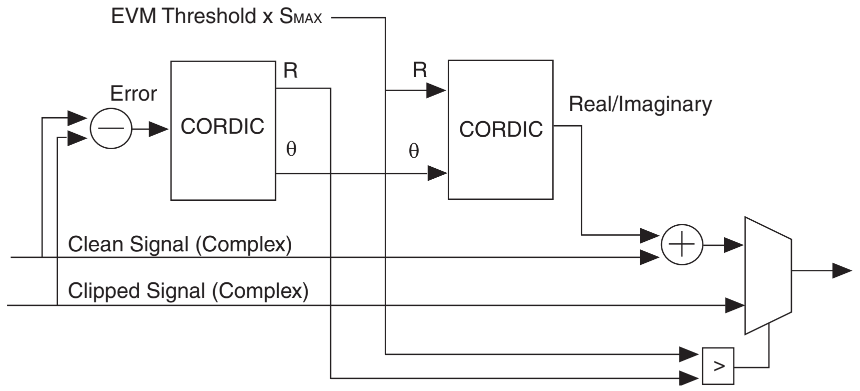
Figure 9. Outband Processor Architecture



Inband Processor

The inband processor calculates the magnitude of the error between the clipped signal and the clean reference signal. If this error is greater than the EVM threshold times S_{MAX}, the clean signal plus the maximum tolerable error magnitude is passed to the output. Otherwise, the signal is passed on unchanged. Figure 10 shows the inband processor architecture.

Figure 10. Inband Processor Architecture



Reference Block

The reference block is designed to synchronize the clean and clipped signals so that the combined inband/outband processor works properly. The latency associated with the clipped signal path passed to the inband/outband processor is $2 \times L \times N$ clocks plus additional pipeline delays. As a result, you need a FIFO that can store the clean reference signal. In addition, this block generates the spectral mask that is used by the inband/outband processor. Because the spectral mask is symmetrical about $L \times N / 2$, only half of the mask is stored and is automatically reflected and repeated by the control logic.

Further Latency Reduction Techniques

To further reduce latency, increase the level of parallelism in the CFR design. While rearchitecting the CFR design to process in parallel requires more resources, the scalability, high density, and DSP and memory capability of FPGAs make rearchitecting viable.

In most circumstances, parallelization of an algorithm requires that the input samples are available in a parallel fashion. Usually you cannot speed up the production of data from the upstream source, but in the case of the CFR module, zero padding accelerates the production of the data in the zeropad block. If you use the dual port memories in Altera devices, you can output samples from the zero padding block at a rate of two complex samples per clock cycle. If the entire signal chain can support the processing of two samples per clock cycle, the total number of clock cycles to perform the processing is halved, which in turn reduces the latency by the same factor. A small latency penalty is incurred in the zero padding block while waiting for $N/2$ samples to become available in the buffer. This penalty is the minimum time required before you can fully utilize the read ports of the memory for the configuration where the oversampling factor $L = 4$.

The polar clipping and in/outband processor blocks require datapath duplication to process two samples per clock. The only modifications required of note are to ensure that the spectral mask lookup table provides two samples on each clock cycle.

You can modify the FFT blocks to process two samples per clock cycle. The latency of the current FFT cores is $N - 1$ clock cycles plus pipelining. You can decompose the required transform into two smaller transforms of length $N/2$ that are combined by using the decimate in time (DIT) and decimate in frequency (DIF) techniques. These modifications result in a total FFT processing latency of approximately $N/2$ + pipelining for an N point transform.

For example, for the FFTs that have natural order addressing inputs, use the decimate in time algorithm — pass the even samples to one core, and the odd samples to another core. Two transforms are performed on both the even and odd samples, before a final recombination stage is used, as summarized by the following decimation in time FFT simplification:

(4)
$$X(k) = \sum_{n=0}^{N-1} x[n]e^{\frac{-j2\pi kn}{N}} = \sum_{n=0}^{N-1} x[n]W^n$$

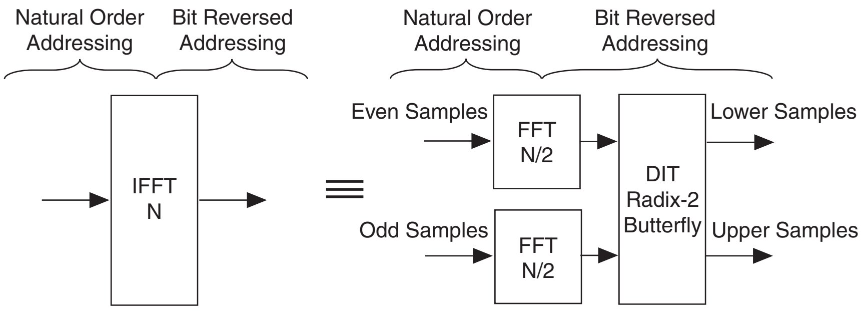
(5)
$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x[2n]W^{2n} + \sum_{n=0}^{\frac{N}{2}-1} x[2n+1]W^{2n+1}$$

(6)
$$X(k) = \sum_{n=0}^{\frac{N}{2}-1} x[2n]W^{2n} + W\sum_{n=0}^{\frac{N}{2}-1} x[2n+1]W^{2n}$$

(7)
$$X(k) = EvenFFT + \left(e^{\frac{-j2\pi k}{N}} \times OddFFT\right)$$

Figure 11 shows the parallel FFT architecture for natural order inputs.

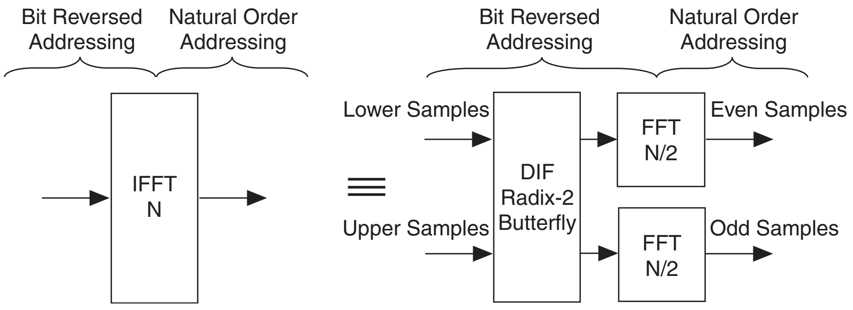
Figure 11. Parallel FFT Architecture for Natural Order Inputs



The output addressing is still bit reversed. One output conveys samples in the range 0 ... $(N/2) - 1$ (the lower samples) and the other output conveys the samples in the range $N/2$ ... $N - 1$ (the upper samples). The subsequent processing blocks must consider this bit-reversed addressing.

Figure 12 shows a similar decomposition for the bit reversed input case. In bit reversed input case, the decimate in frequency technique is more appropriate to allow compatibility between cascaded FFTs.

Figure 12. Parallel FFT Architecture for Bit Reversed Inputs



For more information on decimate in time and frequency techniques for FFT processing, refer to any DSP textbook [5].

Finally, at the end of the processing chain, the guard interval insertion block needs to merge the two streams together and pass them off to the correct antenna. This block requires an element of buffering and is the only block that needs a complete redesign.

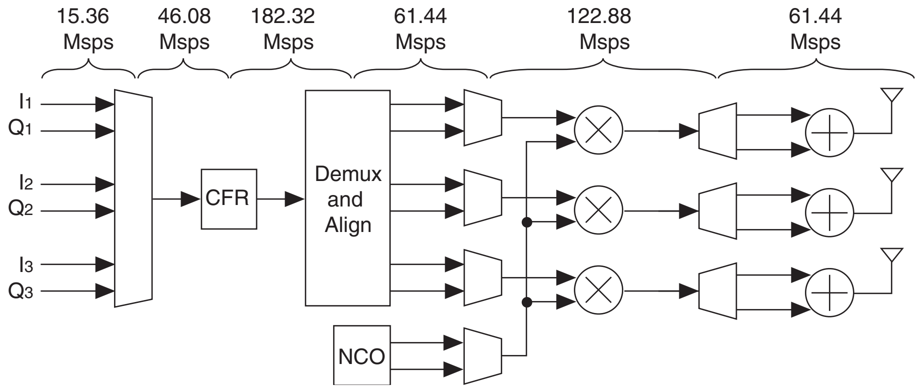
System Integration

The processing requirements of existing and emerging wireless technology tend towards an increase in the number of antennas. As a result, the hardware platform must have a very high throughput processing capability to perform complex functions such as CFR. The CFR reference design demonstrates the advantage of using an FPGA platform over a DSP processor. Using an FPGA reduces processing latency while maximizing throughput.

Figure 13 shows an example of a CFR module integrated into a 10 MHz LTE wireless system. The CFR module processes three baseband channels where each channel has a sampling rate of 15.36 Msps. The baseband channels are multiplexed together such that each frequency domain OFDM symbol is presented to the CFR module in a sequential fashion.

At the output of the CFR, the symbols are in the time domain and have been interpolated by a factor of four. The CFR uses an intermediate frequency of 61.44 Msps, so requires no further upconversion. If an IF sampling frequency of 122.88 Msps is required, just one single stage of interpolation by two filters are required. As a result of the CFR module processing multiple antennas in a time-multiplexed fashion, the signals need to demultiplexed so they can be mixed with the appropriate carrier frequency. This demultiplexing requires external buffering to align the three symbols that are all associated with the same time instant.

Figure 13. Example System Integration of CFR in 10MHz LTE System



The number of antennas supported by the design depends on the baseband sampling frequency (f_{sbb}) and the clock frequency (f_{clk}). Use the following formula to calculate the number of antennas (where $L = 4$):

$$Antennas = \frac{f_{clk}}{f_{sbb} \times L} \tag{8}$$

Based on this formula, Table 2 shows the number of antennas supported for WiMAX and Table 3 shows the number of antennas supported for LTE.

Table 2. Number of Antennas Supported for WiMAX

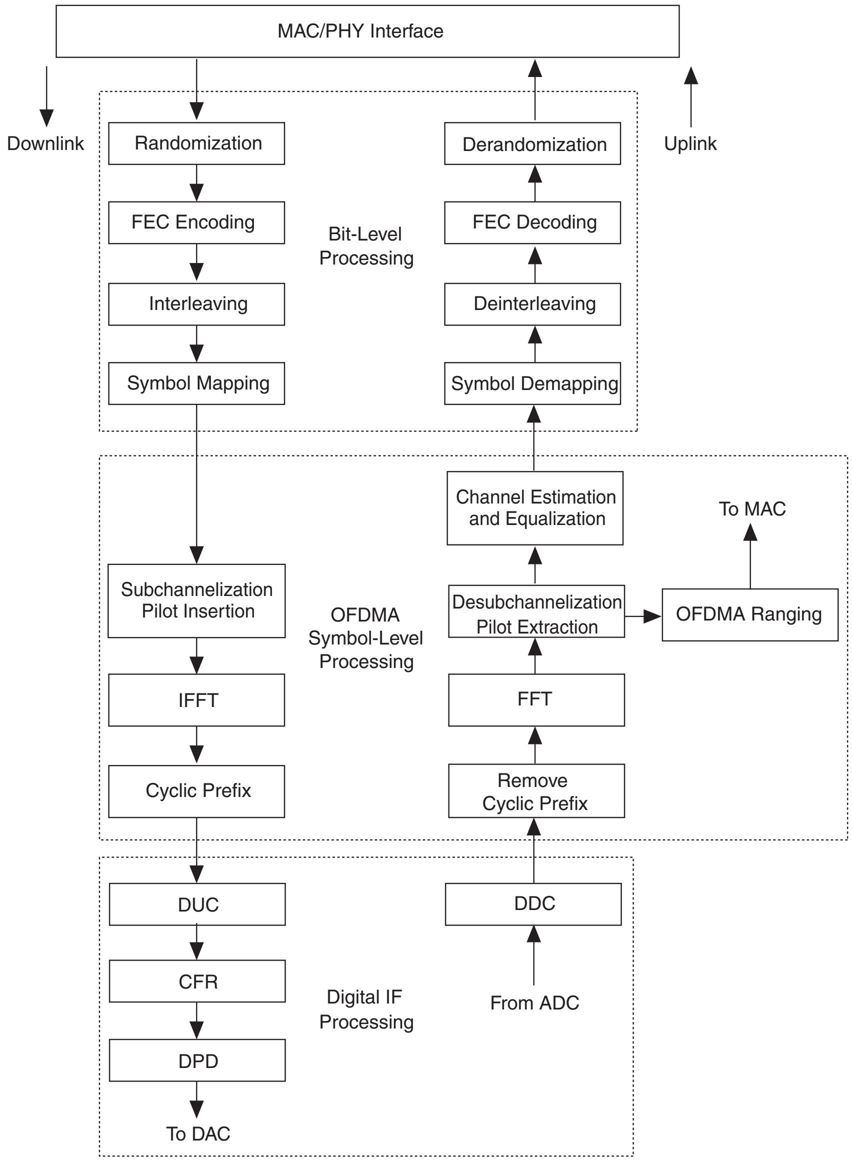
WiMAX (FFT Size)	f_{clk} = 182.784 MHz	f_{clk} = 274.176 MHz
128 (f_{sbb}= 1.428 Msps)	32	48
512 (f_{sbb}= 5.712 Msps)	8	12
1024 (f_{sbb}= 11.424 Msps)	4	6
2048 (f_{sbb}= 22.848 Msps)	2	3

Table 3. Number of Antennas Supported for LTE

LTE (FFT Size)	f_{clk} = 122.88 MHz	f_{clk} = 245.76 MHz
128 (f_{sbb}= 1.92 Msps)	16	32
512 (f_{sbb}= 7.68 Msps)	4	8
1024 (f_{sbb}= 15.36 Msps)	2	4
2048 (f_{sbb}= 30.72 Msps)	1	2

Figure 14 shows a typical physical layer OFDMA system. Often designers decide to architect a system in such a way that the baseband and digital IF processing exist on different cards or devices. Usually the CFR module is implemented on the digital IF card, but because the CFR module includes functionality associated with the baseband inverse FFT (IFFT), cyclic prefix insertion and digital up converter, you can place the CFR module on either device.

Figure 14. Physical Layer OFDMA System



The following list outlines some design considerations useful when partitioning the system:

- Because the input to the CFR module is in the frequency domain, it is ideal to connect it directly to the subchannelization block. If the IFFT/cyclic prefix insertion cannot be bypassed or removed, remove the cyclic prefix and use an additional FFT to regenerate the frequency domain waveform.
- If the CFR is located on the baseband card, the interconnect between the baseband and IF processing needs to run at four times the baseband sampling rate.
- It is not appropriate to decimate the output of the CFR to reduce the sampling rate for transmission between the two devices. Much of the PAR reduction achieved by the CFR algorithm is lost as a result of decimation and re-interpolation in the digital up converter.

Getting Started

System Requirements

The CFR reference design requires the following software:

- Windows XP SP2
- MATLAB version R2006B
- MATLAB Signal Processing Toolbox/Blockset
- MATLAB Fixed Point Toolbox/Blockset
- Quartus® II version 7.2

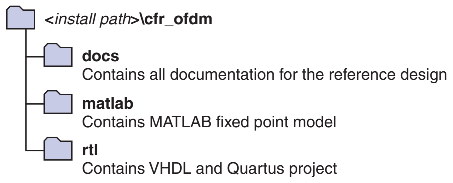
Installing the CFR Reference Design

To install the CFR reference design, run **an475-v7.2.exe** and follow the installation instructions. The CFR reference design installs by default in the directory **c:\altera\reference_designs\cfr_ofdm**. You can change the default directory during the installation.

For a copy of the reference design, please contact your Altera sales representative.

Figure 15 shows the directory structure after installation.

Figure 15. Reference Design Directory Structure


<install path>\cfr_ofdm
docs
Contains all documentation for the reference design
matlab
Contains MATLAB fixed point model
rtl
Contains VHDL and Quartus project

To use the MATLAB fixed point model, set the paths so that MATLAB can find all of the components by performing the following steps:

1. Change the MATLAB current directory to *<installation directory>***cfr_ofdm\matlab\utility**.

2. Run the **cfr_setpaths.m** script.

MATLAB Fixed Point Model

Altera provides a fixed point MATLAB model which you can use to understand the CFR algorithm and to verify algorithm performance. In addition, this model acts a baseline to verify the hardware implementation against and enables you to explore fixed point parameters. You can also use the fixed point model as a dashboard with which to configure the hardware implementation.

The following code sample demonstrates how to call the algorithm:

```
[output, Results] = cfr(input, Config)
```

Table 4 and Table 5 show the CFR module input and output variables.

Table 4. CFR Module Input Variables

Name	Type	Description
`input`	variable	Column vector containing one frequency domain OFDMA symbol of N samples
`Config`	structure	Structure containing parameters to override the default CFR configuration

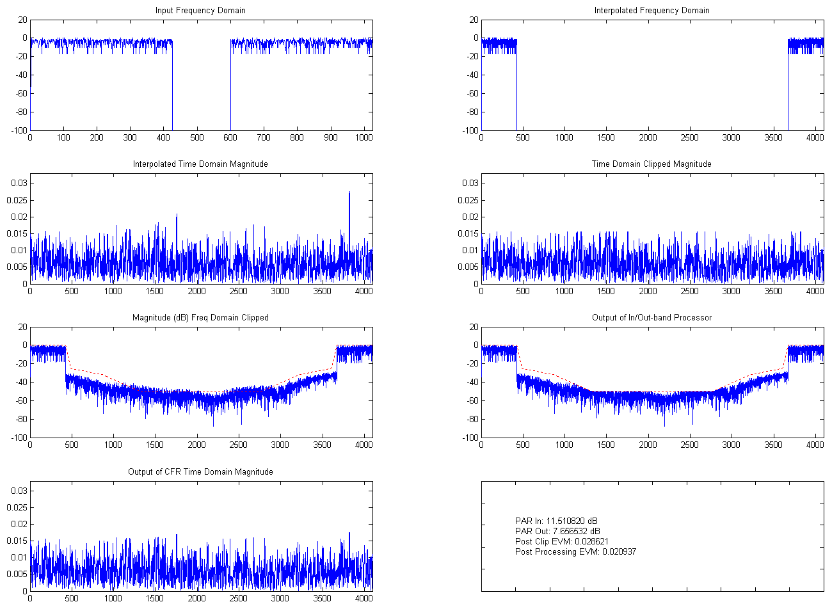
Table 5. CFR Module Output Variables

Name	Type	Description
`output`	variable	Column vector containing one oversampled time domain OFDMA symbol of $L \times N$ samples
`Results`	structure	Structure containing parameters to override the default CFR configuration

For an example of some test data and an example of the CFR module running, open the script **cfr_example.m** and step through the code. You can type `help` *`<function name>`* for command line help.

Figure 16 shows an example of CFR module output.

Figure 16. Example Output From CFR



```
PAR In: 11.510820 dB
PAR Out: 7.656532 dB
Post Clip EVM: 0.028621
Post Processing EVM: 0.020937
```

Parameter Override

After validating the arguments, the function populates the configuration structure for each parameter not defined at the input. To see the structure’s fields and default values, run **cfr_configuration** without any input arguments or type `help cfr_configuration` at the command line.

Table 6 describes the simulation control parameters.

Table 6. CFR Module Simulation Control Parameters

Name	Description
`Config.debug_mode`	When set to zero, disables extra messages output to the console. When set to one, enables extra messages output to the console.
`Config.plot_figures`	When set to zero, disables figure plotting capability for each input symbol. When set to one, enables figure plotting capability for each input symbol.
`Config.sample_order` *(1)*	When set to zero, the frequency domain input vector is in natural (0 to $N - 1$) order. When set to one, the frequency domain input vector is in DC centered ($-N/2$ to $N/2 - 1$) order.
`Config.sim_id`	String that identifies a simulation run

Note to Table 6:

(1) Internally, the algorithm operates in natural order. Any figures plotted adopt natural ordering regardless of the ordering at the input.

Table 7 describes the algorithmic parameters for design space exploration.

Table 7. CFR Module Algorithmic Parameters

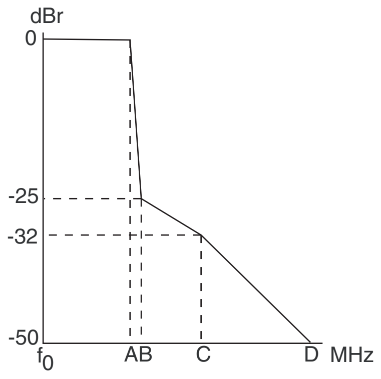
Name	Description
`Config.N_FFT`	The FFT size associated with the basestation deployment. For WiMAX, this is typically 128, 512, 1024 or 2048.
`Config.oversample_factor`	The value of *L*. Altera recommends always using a value of four.
`Config.evm_threshold`	The per symbol EVM that the output symbol must not exceed. For WiMAX 64QAM, the specification gives a total basestation budget of 3% EVM. Typically, the CFR is allowed between 25% and 75% of this total budget, so this value needs to be in the range 0.25 × 0.03 to 0.75 × 0.03.
`Config.Amax`	The magnitude at which the resulting signal is clipped, after conversion to the time domain. For constellation symbols that are normalized according to the WiMAX/LTE specifications, a sensible value for this parameter is in the range 1.0/(`Config.N_FFT` × `Config.oversample_factor`)$^{1/2}$ to 1.4/(`Config.N_FFT` × `Config.oversample_factor`)$^{1/2}$.
`Config.Smax`	The maximum magnitude of the highest order modulation scheme. For more information, see Figure 3.
`Config.spectral_mask_dB`	A vector of magnitude values that define the y vertices of the spectral mask. Note that the inband region is defined by the region where the spectral mask is equal to 0dB.
`Config.spectral_mask_f`	A vector of frequency values that define the x-axis vertices of the spectral mask. The two spectral mask vectors need to be the same length and the frequency vector should be in ascending order. The configuration script interpolates the two vectors to determine the allowed radiation at each frequency bin.
`Config.bandwidth`	The bandwidth associated with all N_FFT frequency bins, used by the spectral mask generation process to normalize the result.
`Config.guard_interval`	The guard interval associated with the OFDM system. Set to zero to disable, or a fraction such as 1/32, 1/16, 1/8 or 1/4.

For 1K WiMAX, you can define the spectral mask as follows:

```
Config.spectral_mask_dB = [0 0 -25 -32 -50];
Config.spectral_mask_f = [0 4.75 5.45 9.75 14.75];
Config.bandwidth = 11.424;
```

Figure 17 shows the spectral mask graphically.

Figure 17. Transmit Spectral Mask (10MHz Channelization)



	A	B	C	D
Frequency (MHz)	4.75	5.45	9.75	14.75

When determining the configuration of the CFR, consider the following points:

- The A_{MAX} parameter determines how aggressively the time domain signal magnitude is clipped. High values of A_{MAX} result in significantly reduced output PAR after the clipping block, but because this introduces much higher levels of distortion in the frequency domain, the subsequent corrections in the frequency domain result in significant peak regrowth.
- As the spectral mask is made more aggressive (that is, as the required stopband attenuation increases) PAR reduction capability is reduced, because distortion introduced in the frequency domain as a result of the time domain clipping is removed, contributing to the peak regrowth.
- Careful consideration of the operating mode is important. For instance, if 64QAM is not required, you can improve the PAR reduction capability significantly by allocating CFR a higher EVM budget, because the EVM requirements are significantly reduced for 16QAM.

Fixed Point Quantization

Normally, the CFR module returns the time domain output symbol. You optionally can use the `Config.rtl_testbench_capture` parameter to direct the CFR module to instead return a structure containing all intermediate signals and configuration details for the hardware. Table 7 describes the settings for the capture parameter.

Table 8. CFR Module RTL Capture Parameter

Name	Description
`Config.rtl_testbench_capture`	When set to zero, disables testbench capture feature. When set to one, enables testbench capture feature.

Use the following parameters to configure the fixed point parameters of the data path. Each block has both a fixed point and floating point implementation. The sink mode parameter determines the implementation used. At the sink of the block, the incoming data is cast to the specified data type. If fixed point is used, natural bitgrowth might occur within the block. At the source of the block, the data is recast back to the output data type.

For the following source and sink block interfaces, zero indicates floating point mode and one indicates fixed point mode:

- `Config.cfr_sink_mode`
- `Config.ifft1_sink_mode`
- `Config.ifft1_source_mode`
- `Config.polarclip_sink_mode`
- `Config.polarclip_source_mode`
- `Config.fft_sink_mode`
- `Config.fft_source_mode`
- `Config.inoutproc_sink_mode`
- `Config.inoutproc_source_mode`
- `Config.ifft2_sink_mode`
- `Config.ifft2_source_mode`
- `Config.cfr_source_mode`

If an interface is configured as fixed point, the type representation must be defined as a MATLAB fixed point `numerictype(signed, bitwidth, fractional width)`. For example, `numerictype(1, 16, 14)` represents a Q2.14 signed fractional number. The following source and sink types can be set to numerictype:

- `Config.cfr_sink_type`
- `Config.ifft1_sink_type`

- `Config.ifft1_source_type`
- `Config.polarclip_sink_type`
- `Config.polarclip_source_type`
- `Config.fft_sink_type`
- `Config.fft_source_type`
- `Config.inoutproc_sink_type`
- `Config.inoutproc_source_type`
- `Config.ifft2_sink_type`
- `Config.ifft2_source_type`
- `Config.cfr_source_type`

☞ If using fixed point types, generally all frequency domain signals have the same type, and all time domain signals have the same type. Utilization of fixed point types significantly reduces the simulation speed. The reference design utilizes signed fixed point format. The time domain signals have the format Q0.16 and the frequency domain signals have the format Q2.14.

Performance Measurement

Table 9 shows the main parameters that determine CFR algorithm performance.

Table 9. Main CFR Parameters

Name	Description
A_{MAX}	The lower the number for AMAX, the more aggressive the clipping. As the intensity of the clipping increases, the greater the outband spectral regrowth and distortion introduced to the constellation points.
EVM threshold	If a high EVM budget is assigned to the CFR algorithm, less of the distorted constellation points need correcting after clipping. This minimizes the peak regrowth associated with the correction applied by the inband processor and in turn increases the PAR reduction capability of the algorithm.
Spectral mask	An aggressive spectral mask results in a high level of correction required in the out of band region, resulting in greater peak regrowth.

Generally, the EVM threshold and spectral mask are related to the specification of the system. You can determine the optimal value for A_{MAX} for a given operating mode and data dynamic range with Monte Carlo simulation.

Typically the performance of a CFR algorithm is determined by examining a 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}), you can compare the PAR of an OFDM symbol that has been compressed by a CFR algorithm with the OFDM symbol that has not been compressed.

Figure 18 shows an example CCDF curve for the WiMAX 1K mode where the EVM threshold is equal to 75% of the EVM budget specified in the specification. The black curve shows that the PAR of the input OFDM signal exceeds 12.3dB for only one out of ten thousand symbols. The other curves on the graph show the output PAR for different values of A_{MAX}. At a probability of 10^{-4}, the output PAR is approximately 3.7dB less than the OFDM case for the optimal value of A_{MAX} (for this case).

Figure 18. Example CCDF Performance Curve

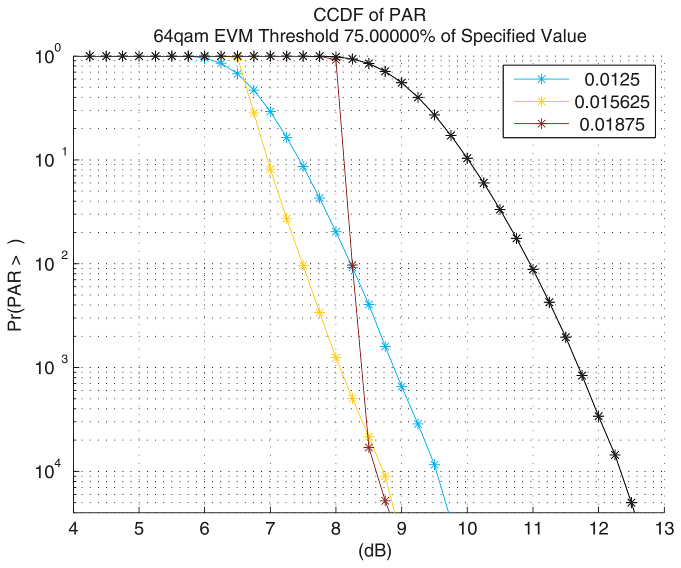


Figure 18 was generated by simulating the CFR model with a stream of WiMAX data. To be confident about each point plotted, there are at least one hundred observations. As a result, the points at 10^{-4}, simulate 10^6 different symbols.

There is an optional bypass configuration mode where the configuration structure is not fully populated when fields are missing. In this case, the `Config` structure must be fully defined. This is particularly useful for Monte Carlo simulation because the parameters remain the same for many different symbols. Typically the configuration structure is created by simply running `cfr_configuration`, and then calling the `cfr` function using the same arguments as above, but with an additional third argument equal to one.

The following code is an example Monte Carlo simulation:

```
% Prepare Configuration Structure
Config = cfr_configuration;

% Override some Parameters
Config.N_FFT = 1024;
Config.evm_threshold = 0.03 * 0.25;
Config.plot_figures = 0;

for Config.Amax =
[0.8:0.1:1.6]./sqrt(Config.N_FFT*Config.oversample_factor)
    for i=1:1e6 % number of symbols
        % Get an OFDMA symbol somehow, and store it in the variable symbol
        [output, Results] = cfr(symbol, Config, 1);
        % Do some processing on the results
        % Store results for later so we can generate a CCDF
    end
end
```

RTL Simulation

You can simulate the CFR module using Modelsim. The reference design provides a script to compile all of the VHDL files, and run a sample testbench to demonstrate the functionality. To invoke the simulation, start Modelsim and change the current directory to <*installation directory*>**\rtl\cfr_**<*x*>**k_tb**, where *x* is the design you wish to simulate. From the Tools menu, select **Execute Macro,** and then select the **Control.do** script.

Synthesis Results

The <*installation directory*>**\rtl** directory contains a Quartus II project you can use to synthesize the design. Table 10 shows the resources and speeds generated for a 10 MHz bandwidth design using the push button flow. The design was synthesized using Quartus II 7.2 software targeting the Stratix III EP3SE50F780C3 device.

Table 10. 10 MHz Bandwidth Design

Design Target (MHz)	Combinational ALUTs	Logic Registers	MATLAB Bits	Memory Bits	M-ALUT	M9K	M144K	18x18
280	16,637	28,561	18,382	976,269	972	133	0	114

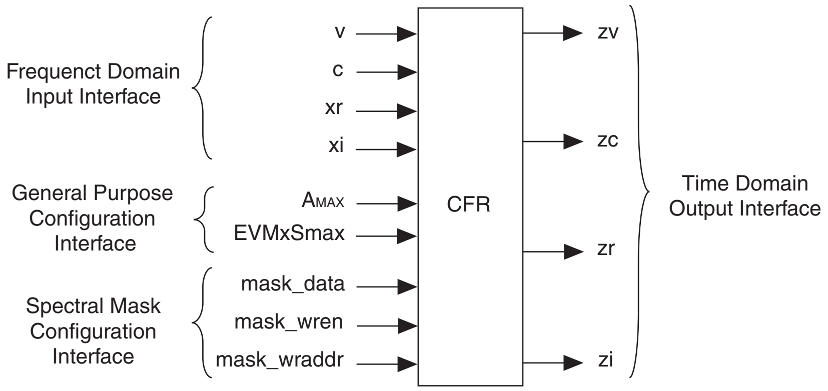
Interface Specification

The CFR module has the following interfaces:

- Frequency domain input interface
- Time domain output interface
- General purpose configuration interface
- Spectral mask configuration interface

Figure 19 shows the CFR top level module port specification.

Figure 19. CFR Top Level Port Specification


Frequenct Domain Input Interface
v
c
xr
xi
General Purpose Configuration Interface
AMAX
EVMxSmax
Spectral Mask Configuration Interface
mask_data
mask_wren
mask_wraddr
CFR
zv
zc
zr
zi
Time Domain Output Interface

Input and Output Interfaces

The following list describes the CFR module input and output signals:

- Frequency domain signals have a signed fractional representation of Q2.14.
- Time domain signals have a signed fractional representation of Q0.16.
- The `v` input and `zv` output signals represent a valid signal which signifies the integrity of the data at that interface.
- The `c` input and `zc` output signals are a symbol identifier. Typically this signal identifies the antenna associated with the data symbol.
- The `xr` input signal is for the real samples in the frequency domain. The `zr` output signal is for oversampled real samples in the time domain. These signals have 16 bit precision by default.
- The `xi` input signal is for the imaginary samples in the frequency domain. The `zi` output signal is for oversampled imaginary samples in the time domain. These signals also have 16 bit precision by default.

General Purpose Configuration Interface

A general purpose configuration interface parameterizes the design. General purpose configuration interface signals should be tied to constants. There are two signals:

- A_{MAX} is a time domain parameter that determines how aggressively the CFR clips.
- EVMxSmax is a frequency domain parameter that determines the overall error at the output. The value of EVMxSmax equals the product of the desired EVM threshold and the constellation S_{MAX} value.

Spectral Mask Configuration Interface

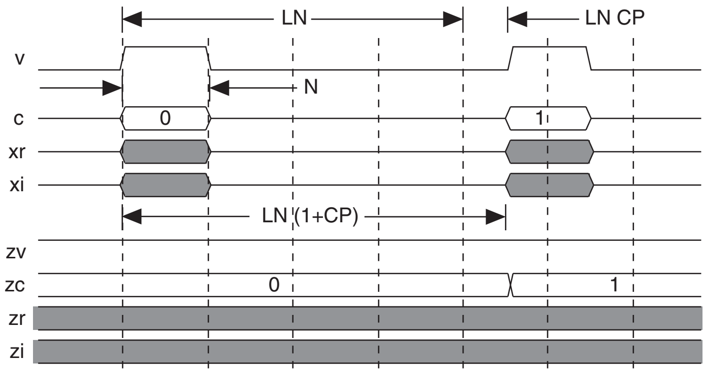
By default, the CFR module ensures the frequency representation of the output signal does not violate the WiMAX spectral mask. You can use any spectral mask by updating the internal lookup table. The internal lookup table contains `N_FFT` × $L/2$ samples that refer to the allowed signal radiation at each frequency domain bin. Because the spectral mask is symmetrical, only half of the samples are stored. The lookup table is organized in natural order, that is, address 0 stores sample 0 and address `N_FFT` × $L/2 - 1$ stores sample `N_FFT` × L/2 - 1. The lookup table contains the linear (not decibels) magnitude of the spectral mask multiplied by the S_{MAX} constant. To modify the table, assert `mask_wren` and increment the `mask_wraddr` counter while presenting values to `mask_data`.

Data Throughput

At the sink interface, you must present an entire FFT frame of N samples in N clock cycles. The frame represents frequency domain data ordered naturally, that is, 0 to $N - 1$. Given this input frame size, the output consists of $L \times N \times (1 + CP)$ samples where $L = 4$ and CP is a fraction that represents the guard interval size. As a result, only one frame of N samples is presented to the sink of the block every $L \times N \times (1 + CP)$ cycles. If this rule is violated, corruption of the data occurs.

To achieve maximum throughput (that is, to fully utilize the output bus) and maximum efficiency, observe the timing diagram shown in Figure 20.

Figure 20. Achieving Maximum Throughput



This timing diagram does not represent the latency associated with the processing that is applied to the output signals.

If data is presented at the sink at intervals greater than $L \times N(1 + CP)$, the source interface is not fully utilized.

Finally, this system does not self flush its pipeline. The hardware relies on subsequent data frames to continue processing the data that already exists in the pipeline. As a result, once the final frame has been presented, present an additional dummy frame of zeros to fully flush the pipeline.

Conclusion

A number of OFDM based wireless applications including WiMAX and LTE require CFR solutions to reduce the PAR of the output waveform. This application note and reference design demonstrate how to efficiently implement high throughput DSP design using an Altera FPGA.

References

1. C. Zhao, R. J. Baxley, G. T. Zhou, D. Boppana and J. S. Kenney, *Constrained Clipping for Crest Factor Reduction in Multiple-user OFDM*, in Proc. IEEE Radio and Wireless Symposium, pp. 341-344, Jan. 2007.

2. S. He and M. Torkelson, *A New Approach to Pipeline FFT Processor*, The 10th International Parallel Processing Symposium (IPPS), pp. 766- 770, 1996.

3. R. Andraka, *A Survey of CORDIC Algorithms for FPGA Based Computers*, Proceedings of the 1998.

4. ACM/SIGDA sixth international symposium on FPGAs, pp. 191-200, February 1998.

5. R. Lyons, *Understanding Digital Signal Processing*, Prentice Hall, 2001.

Document Revision History

Table 11 shows the revision history for this application note.

Table 11. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
November 2007 v1.0	Initial release.	—

101 Innovation Drive
San Jose, CA 95134
www.altera.com
Technical Support:
www.altera.com/support/
Literature Services:
literature@altera.com

I.S. EN ISO 9001