Implementing OFDM Modulation for Wireless Communications

January 2008, version 1.0 Application Note 503

Introduction

This application note discusses various implementation schemes for orthogonal frequency division multiplexing (OFDM) modulation and demodulation. The focus of this application note is cyclic prefix (CP) insertion and removal.

Desired Knowledge

You should have a basic knowledge of OFDM systems before learning the operation of cyclic prefix insertion and removal.

The methods proposed in this application note use the Altera® fast Fourier transform (FFT) MegaCore® function. You should have a basic knowledge of the FFT MegaCore function and Avalon® Streaming Interface.

You can find the design example used in this application note at: www.altera.com/support/examples/vhdl/vhd-cyclic-prefix-insertion-ofdm.html

This application note covers the following topics:

- "OFDM Modulation Basics"
- "Implementing OFDM Modulation"
- "OFDM Modulation Example"
- "Conclusion"

OFDM Modulation Basics

OFDM is a technique used in modern broadband wireless communications systems. To mitigate the effect of dispersive channel distortion in high data rate OFDM systems, cyclic prefix is introduced to eliminate inter-symbol interference (ISI). It copies the end section of an IFFT packet to the beginning of an OFDM symbol. Typically, the length of the cyclic prefix must be longer than the length of the dispersive channel to completely remove ISI.

OFDM modulation in a transmitter includes inverse fast Fourier transform (IFFT) operation and cyclic prefix insertion. In an OFDM receiver, the cyclic prefix is removed before the packet data is sent to FFT for demodulation.

Next generation wireless systems feature highly dynamic configurations, where the cyclic prefix length changes according to the transmission mode, frame structure, and higher level protocol. For instance, the cyclic prefix configuration for 3GPP LTE changes within each slot, as shown in Table 1. The CP length is based on an OFDM symbol with 2048 time intervals. You can also observe similar variable cyclic prefix structures in WiMAX systems.

For more information about the cyclic prefix configuration in 3GPP LTE, refer to *3GPP Technical Specification 36.211 V1.2.0 (2007-06): Physical Channels and Modulation*.

For more information about cyclic prefix structures in WiMAX systems, refer to *IEEE Standard for Local and Metropolitan Area Networks, Part 16: Air Interface for Fixed Broadband Wireless Access Systems*.

Table 1. Cyclic Prefix Parameters for 3GPP LTE *Note (1)*

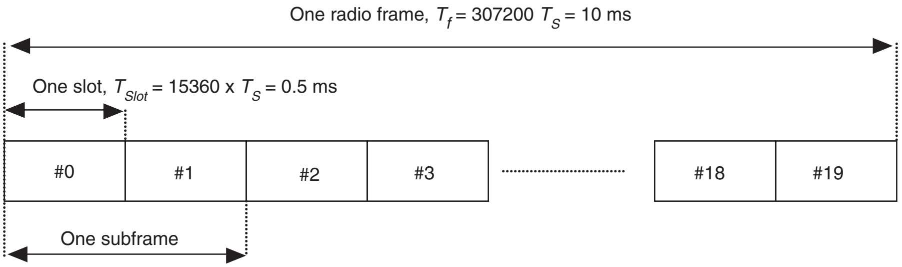
Configuration		Cyclic Prefix Length $N_{CP,\,l}$		Guard Interval
		Frame Structure Type 1	**Frame Structure Type 2**	
Normal cyclic prefix	Δf = 15 kHz	160 for l = 0 144 for l = 1, 2, ... 6	512 for l = 8 in slot 0 244 otherwise	0 for slot 0 288 otherwise
Extended cyclic prefix	Δf = 15 kHz	512 for l = 0, 1, ... 5	768 for l = 7 in slot 0 512 otherwise	0 for slot 0 256 otherwise
	Δf = 15 kHz	1024 for l = 0, 1, 2	1280 for l = 3 in slot 0 1024 otherwise	—

Note to Table 1:

(1) For more information, refer to *3GPP Technical Specification 36.211 V1.2.0 (2007-06): Physical Channels and Modulation*.

Figures 1 and 2 show the frame structures referenced in Table 1.

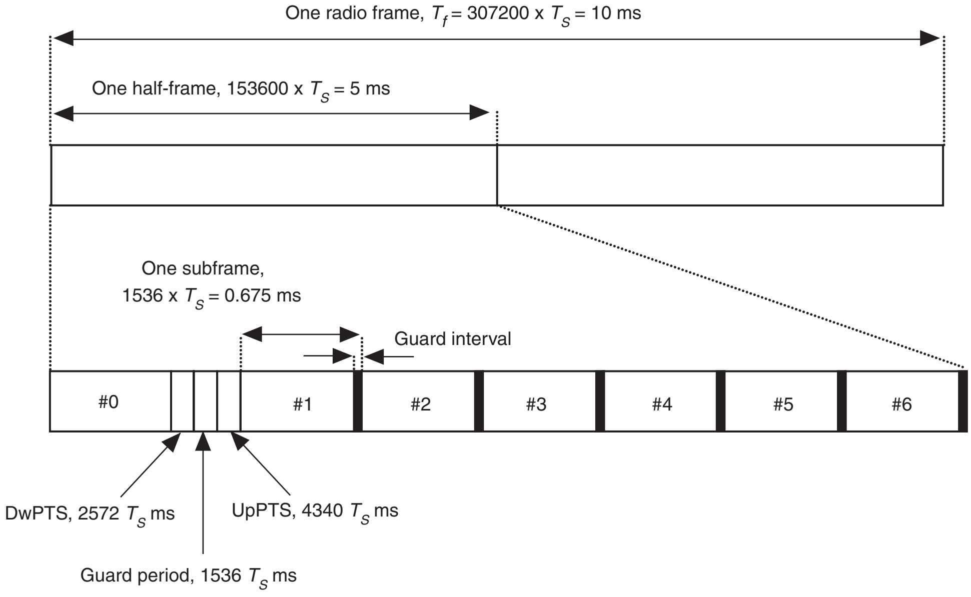
Figure 1. 3GPP LTE Frame Structure 1, Applicable to TDD and FDD Note (1)


One radio frame, $T_f = 307200\ T_S = 10$ ms
One slot, $T_{Slot} = 15360 \times T_S = 0.5$ ms
#0
#1
#2
#3
#18
#19
One subframe

Note to Figure 1:

(1) For more information, refer to *3GPP Technical Specification 36.211 V1.2.0 (2007-06): Physical Channels and Modulation*.

Figure 2. 3GPP LTE Frame Structure 2, Applicable to TDD Note (1)


One radio frame, $T_f = 307200 \times T_S = 10$ ms
One half-frame, $153600 \times T_S = 5$ ms
One subframe,
$1536 \times T_S = 0.675$ ms
Guard interval
#0
#1
#2
#3
#4
#5
#6
DwPTS, 2572 T_S ms
UpPTS, 4340 T_S ms
Guard period, 1536 T_S ms

Note to Figure 2:

(1) For more information, refer to *3GPP Technical Specification 36.211 V1.2.0 (2007-06): Physical Channels and Modulation*.

Implementing OFDM Modulation

The following sections discuss implementing cyclic prefix insertion and removal for OFDM modulation and demodulation.

FFT and Inverse FFT Operations

The most computationally intensive operation of OFDM modulation is IFFT, and similarly, the core of OFDM demodulation is FFT. High FFT throughput is essential in broadband systems, especially when FFT is shared between multiple data paths. In modern scalable wireless systems such as WiMAX and 3GPP LTE, run-time reconfigurability is also an integral part of system requirements. The Altera FFT MegaCore function in variable streaming mode targets specifically reconfigurable wireless communications and is a suitable candidate for designing OFDM systems.

In this application note, the FFT MegaCore function is configured in the variable streaming mode, which allows FFT size and direction change on a packet-by-packet basis. The FFT MegaCore function also takes advantage of the memory-efficient engine-only mode of the FFT core, which outputs bit-reversed symbols directly from the FFT butterfly engines. You can combine bit reversal with cyclic prefix insertion outside the FFT core. The overall OFDM modulation saves a single buffer.

For various cyclic prefix and bit-reversal schemes, refer to *Reduced Transceiver-Delay for OFDM Systems*.

In this application note, “single buffer” refers to the memory buffer of N_{MAX} complex data, where N_{MAX} is the largest FFT size the system supports. “Double buffer” refers to the memory buffer of $2N_{MAX}$ complex data.

FFT Module Reuse

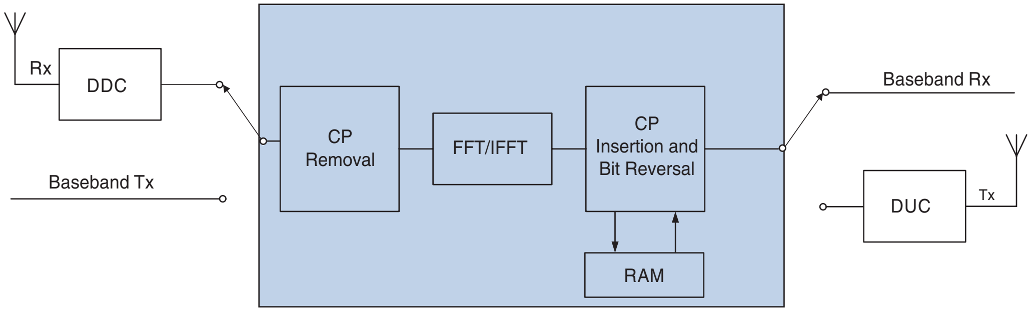
To reduce logic usage, the FFT module is often clocked much faster than the rest of the baseband modules and reused. You can share the FFT module by different sources; for instance, by multiple antennas, transmission and reception in time division duplex (TDD), and frequency division duplex (FDD) systems.

You can also share the FFT module with other functional modules, such as crest factor reduction or channel estimation. However, such reuse depends on the user-specific algorithms and may not be a general design consideration. This application note focuses on the most common wireless communications scenarios: MIMO techniques and TDD and FDD communications.

TDD Operations

In TDD base stations, transmission and reception occurs on non-overlapping time slots. You can easily share the FFT module between the transmitter and the receiver, with proper signal multiplexing. Figure 3 shows a typical single antenna TDD OFDM modulator.

Figure 3. Shared OFDM Modulation and Demodulation in a Single Antenna TDD System



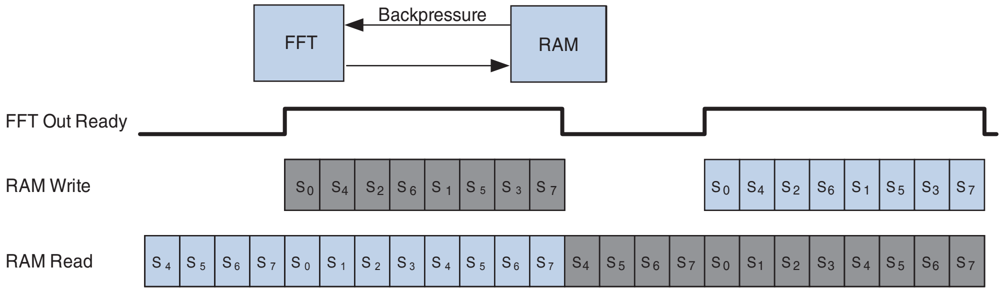
On the transmitting data path, base band data is fed directly to the IFFT module. To insert cyclic prefix and implement bit reversal after IFFT, you can use many different configurations.

For more information, refer to *Reduced Transceiver-Delay for OFDM Systems*.

Figure 4 shows a highly efficient implementation using the Altera Avalon Streaming Interface (Avalon-ST).

On the transmitter data path, IFFT output in bit reversed order is written into a single buffer sequentially, where natural ordered samples from the previous OFDM symbol are read out at the same time from the dual port RAM. When generating cyclic prefix, the FFT core is stalled via Avalon-ST backpressure. Cyclic-prefix-appended continuous OFDM symbols are then sent to the digital up converter (DUC) for transmission.

Figure 4. A Memory Efficient Implementation of Cyclic Prefix Insertion with Backpressure



On the receiving path, after digital down conversion (DDC), cyclic prefix is stripped off from the incoming OFDM symbols. The cyclic removal module in Figure 3 searches the correct start of an OFDM symbol and feeds the data to FFT for demodulation. The single buffer after the FFT module only serves as a bit reversal buffer on the receiving path and no backpressure is asserted.

To reuse the control units, the CP removal and CP insertion modules in Figure 3 are able to distinguish if the current data packet is for transmission or reception, and take action accordingly. In this memory efficient implementation, the FFT core operates at the symbol rate. A single buffer is sufficient for cyclic prefix insertion and bit reversal.

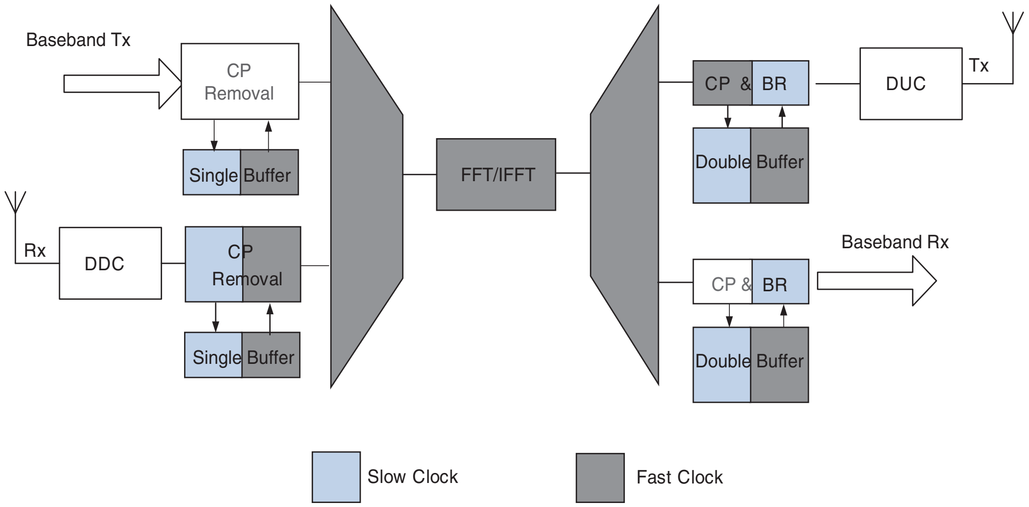
FDD Operations

In FDD operation, transmission and reception occur at the same time. FFT core sharing requires it to operate at no less than twice the baseband symbol rate. The transmission and reception data paths need to have their own set of data buffers.

Figure 5 shows a possible configuration of FFT reuse in an FDD system. The data path operations for transmission and reception are similar to a TDD system—the difference is that these operations occur concurrently. As a result, the pre-FFT data must be buffered and rate changed to higher clock frequency. A single buffer is sufficient for rate change, since the buffer write clock is always slower than or equal to the read clock. While the current packet of data is written at slow clock to the buffer, the previous packet's data is read out at a higher clock rate. When reading and writing into the same memory location, you need to configure the two-port RAM to output old memory contents.

After FFT processing, the high data rate is converted back to OFDM transmission rate via a two-port RAM. This post-FFT memory buffer also acts as a bit reversal buffer. Due to the high-rate-to-low-rate conversion, if you want streaming output, you will need a double buffer. That is, while one FFT packet is written into the buffer, the previous packet's data is read out from the other buffer.

Figure 5. Shared FFT Core for OFDM Modulation and Demodulation in a Single Antenna

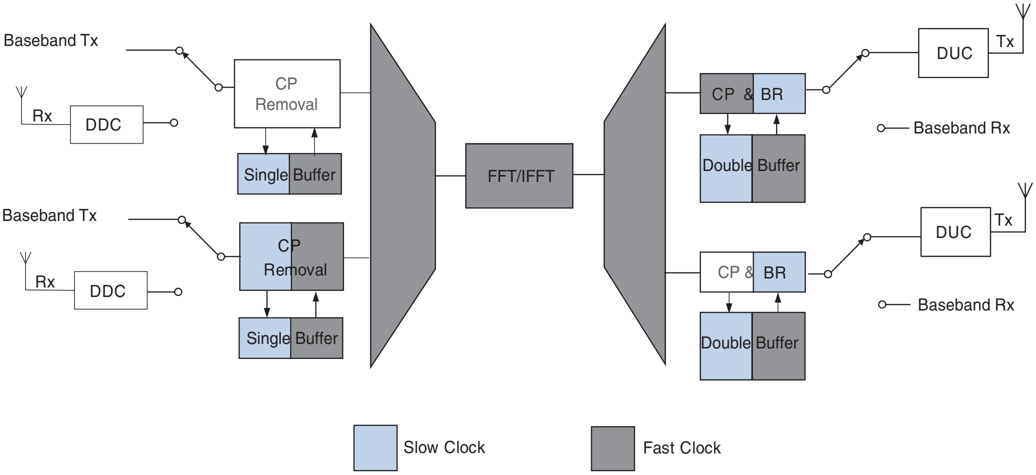


MIMO Configuration

Multiple antenna configuration is a mandatory requirement in modern wireless systems, including WLAN, WiMAX, and 3GPP LTE systems. A straightforward implementation of OFDM modulation in multiple-input multiple-output (MIMO) systems is to duplicate the data path, including the FFT core for every antenna. A more resource-friendly solution is to share the FFT core among antennas. FFT reuse for MIMO requires that the FFT core is clocked at least n times faster than the baseband data path, where n is the number of antennas. You can share the same FFT core across two dimensions, at the expense of pre-FFT data buffering, when combining MIMO and TDD and FDD.

For example, Figure 6 shows a possible configuration for a two-antenna MIMO base station transceiver operating in TDD mode. The FFT core is shared by multiple antennas and by transmission and reception. The control units for cyclic prefix insertion and removal must be able to operate for both transmission and reception. Due to the clock rate difference, you need a double buffer for each antenna post FFT processing.

Figure 6. Shared FFT Core for OFDM Modulation and Demodulation in a Two-Antenna MIMO TDD System



OFDM Modulation Example

This section presents an example that implements OFDM modulation and demodulation on an Altera Stratix III FPGA. FFT and memory buffers are implemented using Altera MegaCore functions. This example is intended for systems that exploit FFT reuse. That is, in this example, the FFT core is clocked much faster than the baseband signals so that it can be shared. The design targets reconfigurable OFDM systems with run-time changeable FFT size and cyclic prefix sizes. Pre-FFT data changes rate via a single buffer. The post-FFT bit reversed data changes rate and order via a double buffer. All control modules support TDD operation and are used for both transmission and reception. Therefore, you can easily extend this example to MIMO and/or TDD systems such as the one shown in Figure 6, or FDD systems such as the one shown in Figure 5.

Key Features

The key features of this design example include the following:

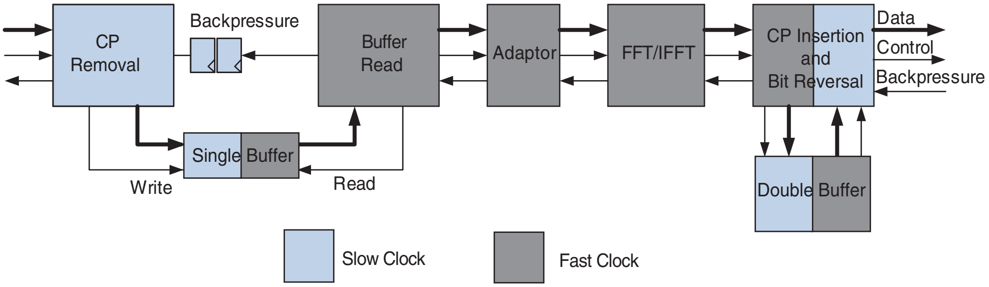
- Support for the most commonly used FFT sizes (128, 256, 512, 1024, and 2048) corresponding to 3GPP LTE channel bandwidths of 1.25, 2.5, 5, 10, and 20 MHz
- Support for fixed or run-time reconfigurable cyclic prefix size on a packet basis

- Support for fixed or run-time reconfigurable FFT size on a packet basis
- Support for TDD operation; you can reuse all control modules for both transmission and reception
- Support for different downlink and uplink FFT sizes in a TDD system with FFT reuse
- Support for arbitrary integer cyclic prefix size, as long as it is less than the maximum FFT size
- Supports Avalon Streaming protocol including backpressure
- Support for parameterization of data width, memory depth, FFT packet size width, and cyclic prefix port width
- Easy extension to MIMO and FDD with FFT reuse

Functional Description

This design example has two parts: OFDM modulation, which includes IFFT and cyclic prefix insertion with bit reversal, and OFDM demodulation, which includes cyclic prefix removal and data buffer for rate change. Figure 7 shows the top-level integration of the two parts. You can view this as an extension of the single antenna TDD system shown in Figure 3. The added pre-FFT buffer allows the design to easily extend to MIMO systems or FDD systems with FFT reuse.

Figure 7. Structure for OFDM Modulation and Demodulation Design Example



Post-FFT Processing

The cyclic prefix insertion for OFDM modulation includes four functional submodules:

- Double buffer implemented using two-clock dual port RAM
- Memory write with bit reversal
- Memory read with cyclic prefix insertion
- Clock synchronization

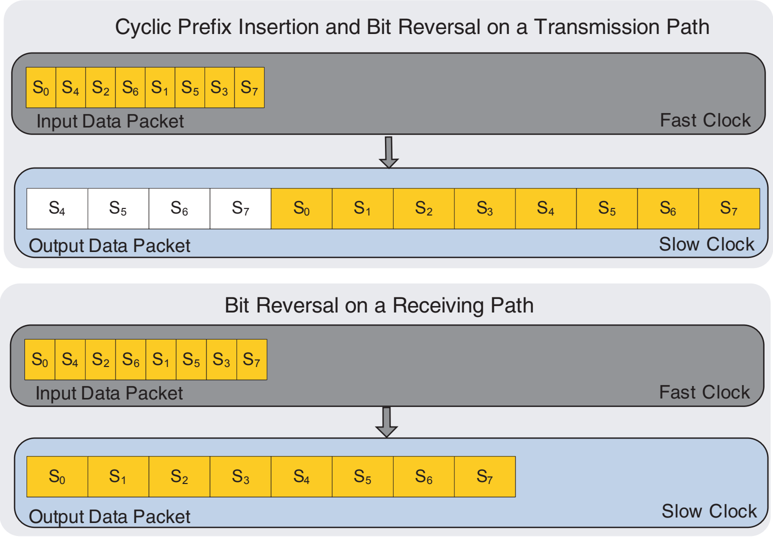
On the transmission data path, IFFT output data in bit reversed order are read in the cyclic prefix insertion module. A control unit analyzes the data address and writes to the corresponding memory location. After one complete IFFT packet is written, the last few samples corresponding to the cyclic prefix are read out in natural order. At the same time, symbols from the next IFFT packet are written to the other buffer if it is available. If both buffers have data for reading, a stall signal is sent to the IFFT core via the Avalon Streaming Interface backpressure.

The output data after OFDM modulation usually is continuous. Downstream modules, for example IF modem and antennas, should not assert backpressure.

On the receiving data path, post-FFT processing limits to bit reversal and rate change. FFT output data in bit reversed order is written to the proper memory address, as was done previously. Once a complete packet of data is written into the buffer, it is read out sequentially in natural order. Note that to avoid excessive backpressure, you need a double buffer because the read clock is generally slower than the write clock. Figure 8 shows the post-FFT data operation.

Control signals indicating buffer status cross two clock domains and are synchronized via synchronization logic.

Figure 8. Data Packet Before and After Cyclic Prefix Insertion Operation


Cyclic Prefix Insertion and Bit Reversal on a Transmission Path
S_0 S_4 S_2 S_6 S_1 S_5 S_3 S_7
Input Data Packet
Fast Clock
S_4 S_5 S_6 S_7 S_0 S_1 S_2 S_3 S_4 S_5 S_6 S_7
Output Data Packet
Slow Clock
Bit Reversal on a Receiving Path
S_0 S_4 S_2 S_6 S_1 S_5 S_3 S_7
Input Data Packet
Fast Clock
S_0 S_1 S_2 S_3 S_4 S_5 S_6 S_7
Output Data Packet
Slow Clock

Pre-FFT Processing

Pre-FFT processing includes four modules:

- Cyclic prefix removal or memory write
- Memory read or rate change
- Two-clock dual port RAM
- Avalon Streaming ready latency converter

As discussed earlier, if the FFT is clocked at a different rate for resource sharing, you need the pre-FFT data buffer. On the receiving data path, the cyclic prefix removal module counts when to start writing input data to the single buffer. After a complete packet of data is written to the single buffer, data is read out starting from address 0.

For fixed FFT size, because the read clock is no slower than the write clock, a single buffer is sufficient. For variable FFT size, a single buffer may not be enough. In an extreme case, suppose the fast clock and the slow clock are at the same rate. In this case, while new data is written into memory, data for the previous packet is read out from the same memory location. If the previous packet is a larger FFT/IFFT size, writing the current packet may be finished well before reading the previous packet is completed. As a result, to prevent overwriting data, you must stall the upstream modules until the reading of the larger packet is complete.

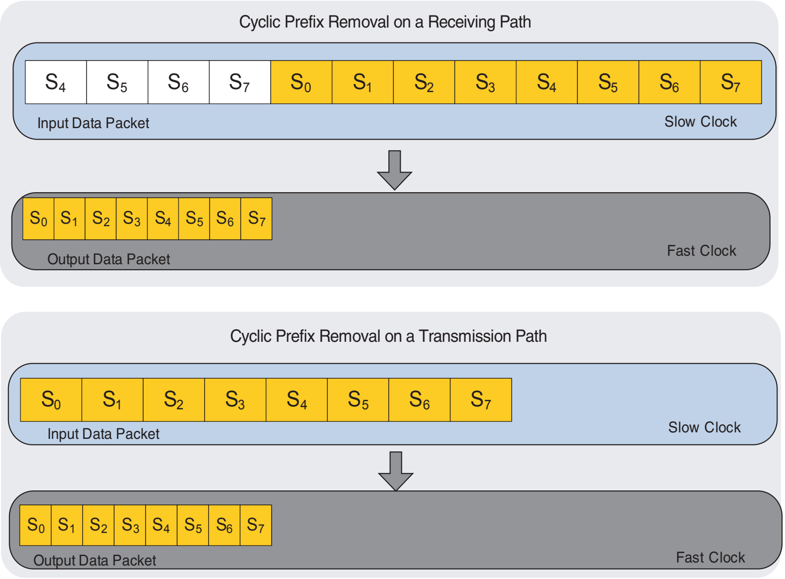
Fortunately, such backpressure is only asserted when the FFT size changes from a larger value to a smaller value. In practical systems, it is unlikely that the FFT size will change frequently. If the FFT size change only occurs on frame boundaries, the backpressure never asserts, due to the quiet period between frame symbols. Alternatively, to avoid using backpressure, you can clock the read signals and FFT core much faster so that reading a packet is complete before writing a packet is complete. The required clock rate depends on the worst case FFT size change ratio. In an extreme case, if the FFT size changes from 2048 to 128, the read clock has to be at least 16 times faster than the write clock.

On the transmission data path, the CP removal submodule simply writes input data sequentially to the single buffer. The same data is read out at a different rate. Figure 9 shows the pre-FFT data operation.

☞ The input data to the OFDM demodulator on the receiving data path is from antennas and IF modem and is usually continuous. Therefore, pre-FFT modules in the receiving mode should not assert backpressure to the upstream modules.

In TDD operations, you can potentially reuse the post-FFT two-port RAM for bit reversal in pre-FFT data buffering, since these operations do not overlap in time.

Figure 9. Data Packet Before and After a Cyclic Prefix Removal Operation


Cyclic Prefix Removal on a Receiving Path
S_4 S_5 S_6 S_7 S_0 S_1 S_2 S_3 S_4 S_5 S_6 S_7
Input Data Packet
Slow Clock
S_0 S_1 S_2 S_3 S_4 S_5 S_6 S_7
Output Data Packet
Fast Clock
Cyclic Prefix Removal on a Transmission Path
S_0 S_1 S_2 S_3 S_4 S_5 S_6 S_7
Input Data Packet
Slow Clock
S_0 S_1 S_2 S_3 S_4 S_5 S_6 S_7
Output Data Packet
Fast Clock

Clock Schemes

This design uses two clock domains, `clk_f` and `clk_s`. The FFT core operates at the fast clock `clk_f`. The two clock domains can be asynchronous if `clk_f` is faster than `clk_s`. Handshaking signals are inserted in the design to synchronize cross-domain control signals. Although in practical systems the frequency of `clk_f` is usually several integer multiples of `clk_s`, in this design, `clk_f` can be the same as the slow clock `clk_s`. In this case, the two clocks must be synchronous.

☞ If the two clocks are synchronous and are of the same rate, the clock-synchronization modules in the demodulator must be taken out due to the single buffer memory depth. In practical systems, this is not a concern, since there is no need to use the pre-FFT buffer if the FFT core is not reused.

Interface

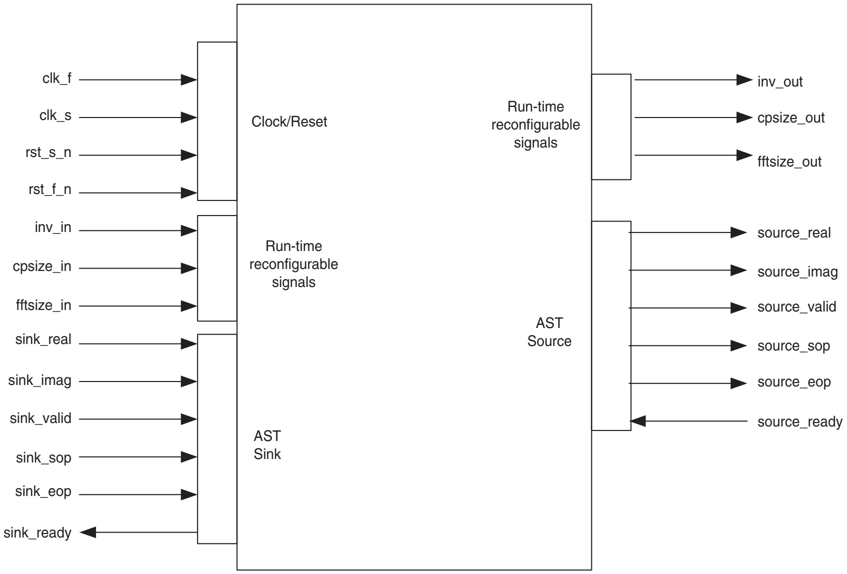
This design example uses the Avalon Streaming protocol for data transfer control. To support variable FFT size, cyclic prefix size, and run time change of FFT direction, you must buffer and align these run time signals with the output data start-of-packet (SOP) signal. Therefore, in every control module of OFDM modulation and demodulation, some logic is dedicated to signal alignment.

This example also supports backpressure via Avalon Stream signals `sink_ready` (an input-ready signal) and `source_ready` (an output-ready signal). The cyclic prefix insertion module has an input ready latency 0 and an output ready latency 4. The cyclic prefix removal module has an input ready latency 0. However, the subsequent buffer read submodule has an output ready latency 2, since the output data is from the memory. To connect to the FFT core, which has a ready latency 0, you need to insert a ready latency adaptor between the buffer read module and the FFT core. The ready latency adaptor accepts input data with a ready latency 2 and generates output data with a ready latency 0. The adaptor also supports backpressure and aligns the reconfigurable cyclic prefix size and FFT direction signals with the output packet data.

I/O Ports

The input and output data of the modulator and demodulator are in signed fixed-point format with configurable bit-width. The output data of the IFFT core in variable streaming mode has full resolution based on the maximum IFFT size. You can perform data truncation inside the cyclic prefix insertion module, if needed, and you can parameterize the bit width. Figure 10 shows the I/O ports of the OFDM modulation and demodulation modules.

Figure 10. I/O Ports of OFDM Modulation and Demodulation Modules



Getting Started

The following sections describe the procedure used to run this design example.

System Requirement

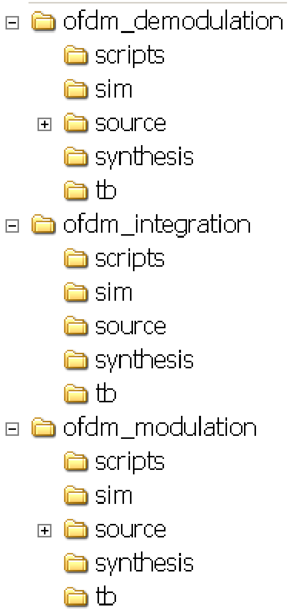
To run this design example, you need the following software tools:

- Windows 2000/XP
- Altera Quartus II software version 7.2 or later
- Altera FFT MegaCore function version 7.2 or later
- ModelSim® SE or PE

Folder Structure

Figure 11 shows the folder structure for this design example. Table 2 describes the contents of each folder.

Figure 11. Design Example Folder Structure



```
⊟ ofdm_demodulation
     scripts
     sim
  ⊞ source
     synthesis
     tb
⊟ ofdm_integration
     scripts
     sim
     source
     synthesis
     tb
⊟ ofdm_modulation
     scripts
     sim
  ⊞ source
     synthesis
     tb
```

Table 2. Design Example Folder Structure (Part 1 of 2)

Folder	Description
ofdm_demodulation	Contains pre-FFT processing units including CP removal and data buffer.
scripts	Contains scripts for ModelSim simulation.
sim	Contains simulation input and output files and the MATLAB script for checking RTL simulation.
source	Contains VHDL source files.
synthesis	Contains Quartus II compilation files.
tb	Contains testbench files.

Table 2. Design Example Folder Structure (Part 2 of 2)

Folder	Description
ofdm_integration	Contains designs for top-level integration of modulation and demodulation.
scripts	Contains tcl scripts for ModelSim simulation.
sim	Contains simulation output files and the MATLAB script for checking RTL simulation.
source	Contains VHDL source files.
synthesis	Contains Quartus II compilation files.
tb	Contains testbench files.
ofdm_modulation	Contains IFFT and post-IFFT designs including P insertion and bit reversal.
scripts	Contains tcl scripts for ModelSim simulation.
sim	Contains simulation input and output files and the MATLAB script for checking RTL simulation.
source	Contains VHDL source files.
synthesis	Contains Quartus II compilation files.
tb	Contains testbench files.

Testbench

The testbenches provided with the design example in the **tb** folder mimic a TDD 3GPP LTE system using Frame Structure 1, as shown in Figure 1. The FFT size changes on a slot basis, where a slot has seven OFDM symbols. The cyclic prefix size changes on a symbol basis. Downlink and uplink communications can use different FFT sizes. A random test of the backpressure signal `source_ready` is also included in the testbenches.

Simulation Scripts

You can use the provided ModelSim simulation script to automate the simulation process. Whether for OFDM modulation, demodulation, or system integration, browse to the **\sim** folder and locate the *<top level design>***_msim.tcl** script. Modify the design installation directories in the tcl script and run it from the **Execute Macro** from the ModelSim Tools menu.

To verify RTL simulation results, compare the RTL results with the MATLAB simulation results by running a routine provided in the **\sim** folder. The MATLAB script is provided for OFDM modulation, demodulation, and system integration. Use the FFT core MATLAB model to generate the FFT/IFFT output. To run the MATLAB simulation, browse to **\sim\check_rtl_<tx, rx, top>.m** and invoke it from the MATLAB command line.

☞ Note that by default, **check_rtl_*.m** uses full resolution for FFT/IFFT output. If the FFT/IFFT output data is truncated, you must modify the MATLAB script accordingly.

Compilation

To compile this design example in the Quartus II software, load the Quartus II project from the **\synthesis** folder. From the Quartus II Software Project menu, browse to the **Add/Remove Files in Project** and verify the path of all the design files and library locations are correct. You need to include all the required library files for FFT before you compile. There are two ways to include the necessary library files:

- Option 1: in the Quartus II software, add **\ofdm_modulation\source\fft72** to the user library. Then re-generate the FFT MegaCore based on the wrapper provided in **\source\fft72\**. Doing this ensures all the necessary FFT library files are included.

- Option 2: automate the library file addition by running the **setup.tcl** script provided in **\ofdm_integration\synthesis\**. In the Quartus II software, on the Tools menu, click **Tcl Scripts**. Select **setup.tcl** and run it. Alternatively, you can invoke **setup.tcl** from the tcl console.

Verify the device selection and click **Start Compilation**.

Performance

The top-level integration of OFDM modulation and demodulation in Stratix III devices are:

- Maximum FFT size: 2048
- ALU: 4776
- Logic registers: 7338
- Memory (bits): 407740 bits
- Memory (M9K): 58 M9K
- DSP Block (18 × 18): 40
- `clk_f` F_{MAX} (MHz): 250 MHz
- `clk_s` F_{MAX} (MHz): 283 MHz

Conclusion

This application note provides an example for implementing OFDM modulation and demodulation in reconfigurable wireless systems. FFT module reuse is the core of the architectures considered in this application note. The design example implements OFDM modulation and demodulation for 3GPP LTE, which supports reconfigurable FFT size and cyclic prefix size. The design example is also applicable to WiMAX, WLAN, and other OFDM-based modern communications systems.

References

1. *3GPP Technical Specification 36.211 V1.2.0 (2007-06): Physical Channels and Modulation*.

2. F. Kristensen, P. Neilson and A. Olssen, *Reduced Transceiver-Delay for OFDM Systems*, IEEE Vehicular Technology Conference, VTC, Spring 2004.

3. *IEEE Standard for Local and Metropolitan Area Networks, Part 16: Air Interface for Fixed Broadband Wireless Access Systems*, IEEE P802.16-REVd/D5-2004, May 2004.

Document Revision History

Table 3 shows the revision history for this application note.

Table 3. Document Revision History

Date and Document Version	Changes Made	Summary of Changes
January 2008, 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