RW-WLAN Modem Functional Specification

Functional Specifications

RW-WLAN-Modem-FS/0.1

Version 0.1

2019-01-04

RivieraWaves
http://www.rivierawaves.com/

Revision History

Version	Date	Revision Description	Author
0.1		Initial Revision	

TBD	Date entered	Description	Status
TBD-1			
TBD-2			
TBD-3			
TBD-4			
TBD-5			
TBD-6			
TBD-7			
TBD-8			
TBD-9			
TBD-10			
TBD-11			

Table 1-1: Items to be determined in the future versions of this document

Table of Contents

Revision History **2**

Table of Contents **3**

List of figures **5**

List of tables **6**

1 Overview **7**

1.1 Document overview 7
1.2 Product overview 7
1.2.1 What is the RW-WLAN nX Modem? 7
1.2.2 Supported configurations 7
1.2.3 Details of the 1x1:1-40 configuration 8
1.2.4 Radio Interface Unit Reference Design 9
1.3 SIFS Latency budget 9

2 Modem operating modes overview **10**

2.1 Operating modes 10
2.2 Bandwidth modes 10
2.3 BCC and LDPC 12

3 Hardware architecture overview **13**

3.1 System overview 13
3.2 Control 14
3.3 OFDM modem core overview 14
3.3.1 OFDM data path 14
3.3.2 FFT/IFFT 15
3.3.3 Tx data path 15
3.3.4 Rx data path 15
3.4 Legacy DSSS/CCK modulation support 16
3.5 MAC-PHY interface 16
3.6 Beamforming support 17
3.6.1 Beamformee support HW accelerator 17
3.7 AHB Slave interface 17
3.8 Clocking scheme 19
3.8.1 Overview 19
3.8.2 Rx Bit Domain clock domain 20
3.8.3 Phy clock domain 20
3.8.4 DSSS-CCK Modem domain 21
3.8.5 Clock gating 21
3.9 Reset scheme 22
3.10 Memories 22
3.10.1 Overview 22
3.10.2 FFT/IFFT/H Memory access and organization 23
3.10.3 BD/FD Memory size and sharing 25
3.11 Diagnostic features 26

4 OFDM Modem Transmit Path **27**

4.1 Block diagram 27
4.2 TX vector decoder 28
4.3 Tx Bit Domain 29
4.3.1 Phy padding 29
4.3.2 Scrambler 29
4.3.3 Bit parser 29

4.3.4 BCC 30
4.3.5 LDPC 30
4.3.6 Stream parser 30
4.3.7 Bit domain control fsm 30
4.4 Tx Frequency Domain 30
4.4.1 Interleaver 30
4.4.2 Preamble generator 30
4.4.3 Pilot generator 30
4.4.4 Frequency mapping 31
4.4.5 Constellation mapping 31
4.4.6 STBC 31
4.4.7 Frequency duplication 31
4.4.8 CSD 31
4.4.9 Spatial mapping 31
4.4.10 Frequency domain control 32
4.5 IFFT processing 32
4.6 Tx Time Domain 32
4.6.1 Guard interval and Windowing 32

5 OFDM Modem Receiver 33

5.1 Rx Time Domain 35
5.2 Rx Frequency Domain 37
5.2.1 Configuration 37
5.2.2 Blocks description 38
5.3 Rx Bit Domain 38
5.3.1 Bit domain deinterleaver 39
5.3.2 Stream mux 39
5.3.3 Depuncturer 39
5.3.4 Field splitter 40
5.3.5 Viterbi 40
5.3.6 LDPC decoder 40
5.3.7 Bit mux 40
5.3.8 Bit domain control sub-system 40
5.3.8.1 LSIG/HTSIG/VHTSIG/HESIGA parser 41
5.3.8.2 HESIGB parser 41
5.3.8.3 Data parser 41
5.3.8.4 RX BD FSM 41
5.3.8.5 RX BD UCPU 42
5.4 Receiver sequencing 42
5.4.1 Frame format handling diagram 42
5.4.2 Receiver State Machine 43
5.5 Receiver timings 45
5.5.1 Overall latency 45

6 Common 46

6.1 FFT and IFFT 46
6.2 UCPU 46

7 Design recommendations 48

7.1 Configurable Implementation 48

References 50

List of figures

Figure 1-1 : RW-WLAN Modem overview 7
Figure 2-1 : RW-WLAN Modem 20 MHz Operating Mode 10
Figure 2-2 : RW-WLAN Modem 20/40 MHz Operating Mode 11
Figure 2-3 : RW-WLAN Modem 20/40/80 MHz Operating Mode 11
Figure 3-1 : RW-WLAN Modem overview 13
Figure 3-2 : Use of memory buffers in the RW-WLAN Modem OFDM data flow 14
Figure 3-3 : Memory connections of the Beamformee Support Module 17
Figure 3-4: H memory connections 24
Figure 3-5: H memory RX data organization for 2 RX paths and 4 streams 25
Figure 5-1: Rx top level data path with Nrx = 2, Nss = 2, 20/40/80MHz bandwidth mode 33
Figure 5-2: Rx top level data path with Nrx = 1, Nss = 1, 20/40MHz mode 34
Figure 5-3: Rx top level data path with Nrx = 1, Nss = 1, 20/40/80MHz mode 34
Figure 5-4: Time boundary estimate sequence 36
Figure 5-5: Rx modem core state-machine in 20MHz configuration 43

List of tables

Table 1-1: Items to be determined in the future versions of this document 2
Table 1-1: SIFS time split 9
Table 2-1: 2x2:2-80 Modem Operating Modes 10
Table 2-2: ADC and DAC frequency for Bandwidth Operating Modes 12
Table 3-1: Modem clock domains 19
Table 3-2: Clock requirements vs configuration for the OFDM Rx Bit Domain 20
Table 3-3: Clock requirements vs configuration for the MAC-PHY Interface Domain 21
Table 5-1 : FD configuration parameters vs frame field under reception 37
Table 5-2: Modem RX latency for BCC frames 45
Table 5-3: Modem RX latency for LDPC frames 45
Table 7-1: Configuration options 49

1 Overview

1.1 Document overview

This document provides the functional description of the RW-WLAN- 802.11ax modem. The main goal is to provide an overview of the top-level architecture as well as the main sub-blocks.

1.2 Product overview

1.2.1 What is the RW-WLAN nX Modem?

As illustrated in Figure 1-1, associated with ADC, DACs, a WLAN radio and a Radio Interface Unit (RIU), the RW-WLAN-Modem implements the PHY level of the 802.11ax standard.

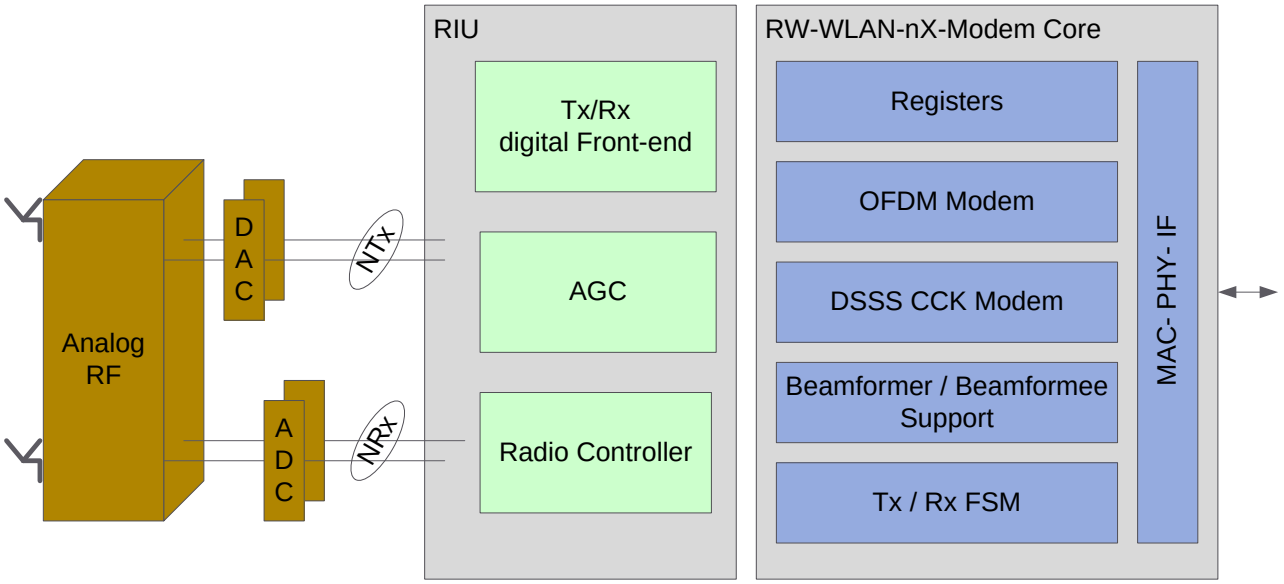


Figure 1-1 : RW-WLAN Modem overview

1.2.2 Supported configurations

The modem is designed in such a way that different configurations can be chosen at implementation, thanks to the use of parameters and defines in the RTL code. The configuration parameters are fully described in [7]. Main parameters are:

- Ntx, the number of transmit paths is from 1 to 2
- Nrx, the number of receive paths is 1 to 2
- Nss, the number of spatial streams is 1 or 2 (Note that Nrx >= Nss and Ntx >= Nss)
- BW, the largest channel bandwidth is 20MHz, 40MHz or 80MHz.

Each modem configuration is named Ntx x Nrx : Nss - BW. For example, 1x1:1-20 is a modem having 1 path in transmission, 1 path in reception, supporting only one spatial stream and 20MHz bandwidth packets.

The Modem supports all the mandatory features. As such, the 802.11ax when configured in a 20MHz or 40MHz configuration supports all the previously defined frame format (eg 802.11a/b/g/n). When configured with an 80MHz channel bandwidth it also support 802.11ac frame format.

Please note that some mandatory features on 802.11ax (for instance LDPC, MU-MIMO) were optional on 802.11n/ac. Whenever those features are backward compatible they are available in 802.11n/ac mode.

For instance, LDPC can be used in 802.11n and 802.11ac.

Same for MU-MIMO (DL) and beamforming which can be used in 802.11ac.

Please note that the 802.11n beamforming feature is not backward compatible, hence not supported.

- If the Modem configuration includes support of 11ac 80 MHz frames, another set of parameters allows adding support of MU-MIMO.

In the 802.11nac standard, MU-MIMO is a downlink feature, used only from AP to STA. Therefore, the Modem, as a STA, only needs to be able to receive frames from an AP using MU-MIMO. An AP needs channel information before sending a MU-MIMO frame to an STA, so the pre-requisite to support Rx MU-MIMO is to be able to act as a beamformee for up to 4 streams. All this requires the following specific HW support:

- Channel estimate able to process up to 4 streams
- Beamformee 4x2 capability: SVD module to generate the channel information report
- Pre-processing block for interference cancellation added before the equalizer in the Rx path
- No support of Tx of MU-MIMO frames

Note that this configuration also allows the Modem, if needed, to act as a beamformee with 2x2 capability, without MU-MIMO support.

As an AP, the Modem can act as a MU-MIMO beamformer device. This requires the following specific HW support:

- Beamformer Support and Multi User Tx PreProcessing blocks to decompress one or two beamforming reports and write the resulting Q steering matrix in the H memory
- TX path modifications to use the steering matrix content
- Tx Path for second user

1.2.3 Details of the 1x1:1-40 configuration

The RW-WLAN Modem 1x1:1-40 supports features listed below

- 1 Tx paths
- 1 Rx paths
- 20MHz, 40MHz
- 1 spatial streams
- DSSS/CCK rate
- OFDM Legacy rate
- OFDMA (SU / MU) UL/DL
- MCS 0 to 7 for 802.11n and MCS 0 to 9 for 802.11ac/ax
- GI 0.4/0.8 (11n/11ac) 0.8/1.6/3.2(11ax)
- 11ac/11ax Beamforming procedure (no support of beamforming in 11n mode):
- MU-MIMO DL as a STA (11ac/11ax)
- Custom synchronization and offset compensation algorithms
- LDPC coding scheme for Forward Error Correction

1.2.4 Radio Interface Unit Reference Design

The RIU is not part of the Modem, because it may have to be adapted to the RF/Analog components to get a complete system compliant with the standard requirements. As its operating modes are closely linked to the Modem's, some details on its implementation are given in this document. It also shares the Modem's AHB slave interface. It includes:

- A digital front-end, which mainly performs filtering operations to adapt the signal sampling rate between the ADC/DAC and the modem core. The sampling rate at the modem core - RIU interface is equal to the frame bandwidth in Rx, and twice the frame bandwidth in Tx.

  The digital front-end also includes digital modulators which allow to center 20MHz BW frames from/into the lower or upper channel, using+/- 20 MHz and +/- 10 MHz shifting.

- The ADC and DAC sampling rate requirement is a 2x factor (ie ADC/DAC @40MSPS for a 20MHz modem, and a @80MSPS for a 40MHz mode)

- An AGC / CCA, responsible for detecting incoming frames and set the RF transceiver gain. It is implemented using hardware accelerators for signal power computation, cross-correlations, and a programmable micro-sequencer for the state machine. This way, it is possible to tune it during the validation phase, and to easily adapt it to the targeted RF transceiver.

- A Radio Controller, which design and RF interface depend on the targeted RF transceiver. It transforms the commands sent by the AGC/CCA FSM into RF transceiver specific control signals, and adds required delays between them.

1.3 SIFS Latency budget

The modem follows IEEE latency budget recommendation for the 16 µs SIFS, which is split into:

- up to **12 µs** reserved for the RX chain,

- **1 to 3 µs** depending on the MAC core clock frequency for the MAC turnaround time, to switch the TX chain to answer after receiving a frame (as for Acknowledge packet for example). Note that the actual response data is not required until 16 µs after at the earliest, since the modem will start by sending the preambles,

- up to **3 µs** for the TX chain.-

Actual SIFS values in the Modem are shown in Table 1-1.

Maximum supported configuration	40MHz BW + LDPC	40MHz BW, BCC only	20 MHz BW + LDPC	20 MHz BW, BCC only	Remarks
MAC RX-TX turnaround	3.0	3.0	3.0	3.0	Assumption worst case
Modem TX delay [STF]	2.2	2.2	1.3	1.3	
Modem RX delay	10.2	6.7	9.6	8.9	See 5.5
SIFS Time (RX-TX worst case)	**14.9**	**11.4**	**15.1**	**14.4**	**T< 16 us : OK**

Table 1-1: SIFS time split

2 Modem operating modes overview

2.1 Operating modes

The Modem can be limited by registers to a sub-set of operating modes, e.g. to save power while connected to a BSS supporting only a lower mode. When the Modem is limited to a given operating mode, it will abort the reception of frames outside of the scope of this mode, and must not be programmed to transmit such frames. Details of the Modem programming can be found in [3].

Care must be taken to program the Modem only into the operating modes its configuration and system integration supports. For instance, the LDPC operation requires higher clock frequencies, or the higher bandwidth modes can be used only if the Modem configuration supports the highest of the bandwidths of the operating mode.

Operating modes for the 1x1:1-40 Modem configuration are listed in Table 2-1. Note that when the Modem's bandwidth mode is changed, the RIU configuration and the ADCs and DACs clock frequency must be changed accordingly to ensure correct Modem operation. The other clock frequencies are suggested frequencies equal or higher to the minimum requirements for each domain. Refer to chapter 3.8 for details on the clock domains.

Modem Config.	Supported Frame Type	Maximum number of Spatial Streams	Supported coding scheme	Required ADC/DAC frequency (MHz)	Required PhyClk frequency (MHz)	Suggested BDRxClk frequency (MHz)
0	20 MHz	1	BCC and LDPC	40	120	240
1	40 MHz	1	BCC and LDPC	80	120	240

Table 2-1: 2x2:2-80 Modem Operating Modes

More details on the bandwidth modes and associated ADC/DAC clock frequencies are given in chapter 2.2. More details on BCC and LDPC operation are given in chapter 2.3.

2.2 Bandwidth modes

Depending on the maximum bandwidth supported by its configuration, the modem can operate in the following bandwidth modes:

- 20MHz mode: the modem is able to transmit and receive only 20MHz frames, as illustrated in Figure 2-1. In this mode the ADC and DAC must be running at 40MHz and the RF carrier frequency is in the middle of the 20MHz channel.

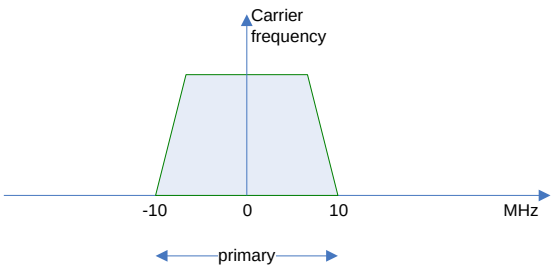


Figure 2-1 : RW-WLAN Modem 20 MHz Operating Mode

- 20/40MHz mode: the modem is able to transmit and receive 40MHz frames, and 20MHz frames in the 20MHz primary channel. This is illustrated in Figure 2-2. In this mode the ADC and DAC must be running at 80MHz and the RF carrier frequency is in the middle of the 40MHz channel. The 20MHz primary channel can be in the upper or in the lower part (as illustrated) of the 40MHz band.

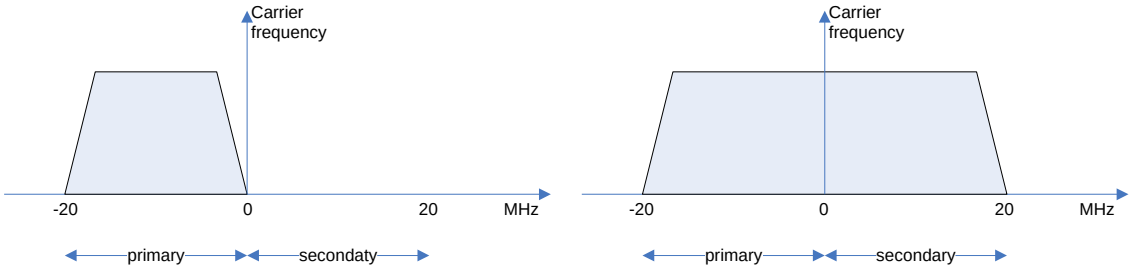


Figure 2-2 : RW-WLAN Modem 20/40 MHz Operating Mode

- 20/40/80MHz mode: the modem is able to transmit and receive 80MHz frames, 40MHz frames in the 40MHz primary channel and 20MHz frames in the 20MHz primary channel. This is illustrated in Figure 2-3. In this mode the ADC and DAC must be running at 160MHz and the RF carrier frequency is in the middle of the 80MHz channel. The 40MHz primary channel can be in the upper or in the lower part of the 80MHz band.

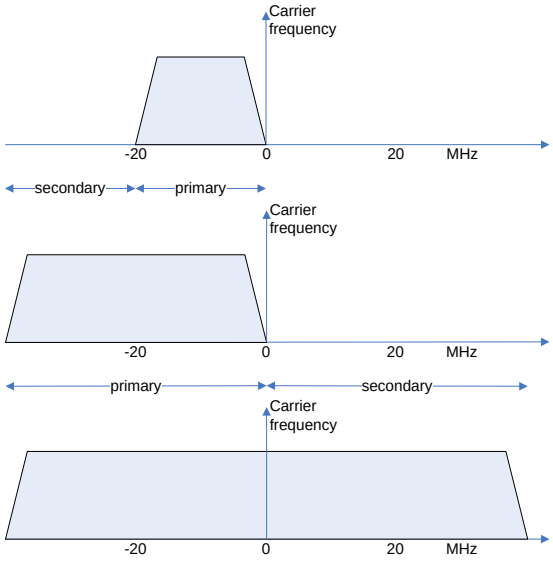


Figure 2-3 : RW-WLAN Modem 20/40/80 MHz Operating Mode

For each mode, the sampling rate of the ADC and DAC and the RF carrier frequency must be adapted. The Table 2-2 gives the ADC/DAC sampling rate as a function of the bandwidth mode.

Modem's HW Channel bandwidth configuration	Programmed Operating Mode	Supported Frames	ADCs frequency	DACs frequency
20MHz	20MHz	20MHz	40MHz	80MHz
40MHz	20MHz	20MHz	40MHz	160MHz

	20/40MHz	40MHz and 20MHz in primary	80MHz	160MHz
80MHz	20MHz	20MHz	40MHz	80MHz
	20/40MHz	40MHz and 20MHz in primary	80MHz	160MHz
	20/40/80MHz	80MHz, 40MHz in primary and 20MHz in primary	160MHz	320MHz

Table 2-2: ADC and DAC frequency for Bandwidth Operating Modes

2.3 BCC and LDPC

The 802.11n/ac/ax standard includes Forward Error Correction (FEC). Two coding schemes can be used. Support of Binary Convolution Code (BCC) is mandatory, and support of Low-Density Parity Check (LDPC) is mandatory for 40 MHz ax. The Modem can be configured to support LDPC or not (see [7]).

BCC operation is performed with a BCC encoder and a Viterbi Decoder. LDPC operation requires two additional modules, an LDPC encoder and an LPDC decoder.

The Viterbi and LDPC decoders drive the required clock frequency of the Modem Rx Bit Domain. LDPC is a block code, needing data buffering. This requires additional memories (see 3.4) and higher clock speeds (see 3.8).

3 Hardware architecture overview

3.1 System overview

As illustrated in the Figure 3-1, associated with ADC, DACs, a WLAN radio and a radio controller, the RW-WLAN-Modem implements the PHY level of the 802.11nacax standard.

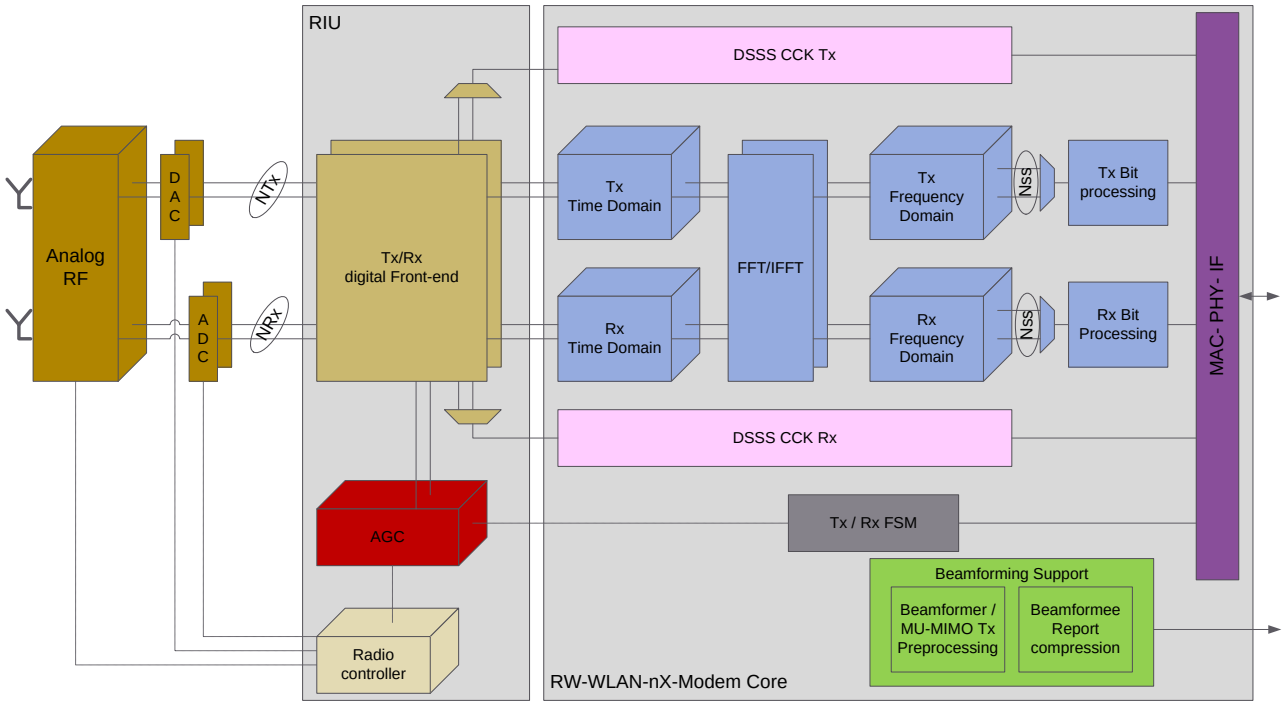


Figure 3-1 : RW-WLAN Modem overview

Note that this figure presents the modem in a 2x2:2 configuration, and shows how the Modem's configuration impacts its internal data path:

- some functions (Front-End , FFT…) are duplicated as a function of the number of paths
- others (3-dimension boxes) are doing multi-antenna processing.

The RIU is not part of the Modem, because it may have to be adapted to the RF/Analog components to get a complete system compliant with the standard requirements. Some details on the RIU reference design for the RW-WLAN-Modem are given in 1.2.4.

The modem core is independent of the rest of the PHY level components. It includes:

- A Main State Machine, described in 3.2.
- An OFDM/OFDMA modem (blue boxes) supporting the 802.11g, 802.11a, 802.11n, 802.11ac and 802.11ax standards. More details on this part are given in chapter 3.3.
- A DSSS/CCK modem (pink boxes) to ensure backward compatibility with the 802.11 and 802.11b standard, see 3.4.
- A MAC-PHY interface (purple box), handling all data and control signals going between the MAC and PHY parts of the system, see 3.5.

- If included in the Modem's configuration, Beamforming Support HW accelerators (green boxes). See 3.6 for more details on this part. The Beamformee part is actually included inside the Rx Frequency Domain.

3.2 Control

All the modem modules are controlled by a main TX/RX state machine, which controls the shared FFT data path multiplexing and the activity of the RX or TX parts for OFDM and DSSS/CCK modulations.WLAN

Some modules include a local state machine, which provides sequencing at sample or symbol level. The interface between main TX/RX FSM and local modules FSMs is based on a level signal mechanism, so that the main state machine timings are not dependent from relative block frequency. This scheme makes the main FSM design independent from any blocks timing and relative latencies. The only requirement is that all modulation/demodulation operations have to be completed before the symbol boundary.

3.3 OFDM modem core overview

3.3.1 OFDM data path

As shown in Figure 3-1, the Tx and Rx paths of the OFDM modem core are split into three main sections, the time domain (TD), the frequency domain (FD), and the bit-processing domain (BD).

Figure 3-2 illustrates how the Modem uses memory buffers to handle its internal data flow. For more details on BD/FD buffer structure and memory sharing, see 3.10.3.

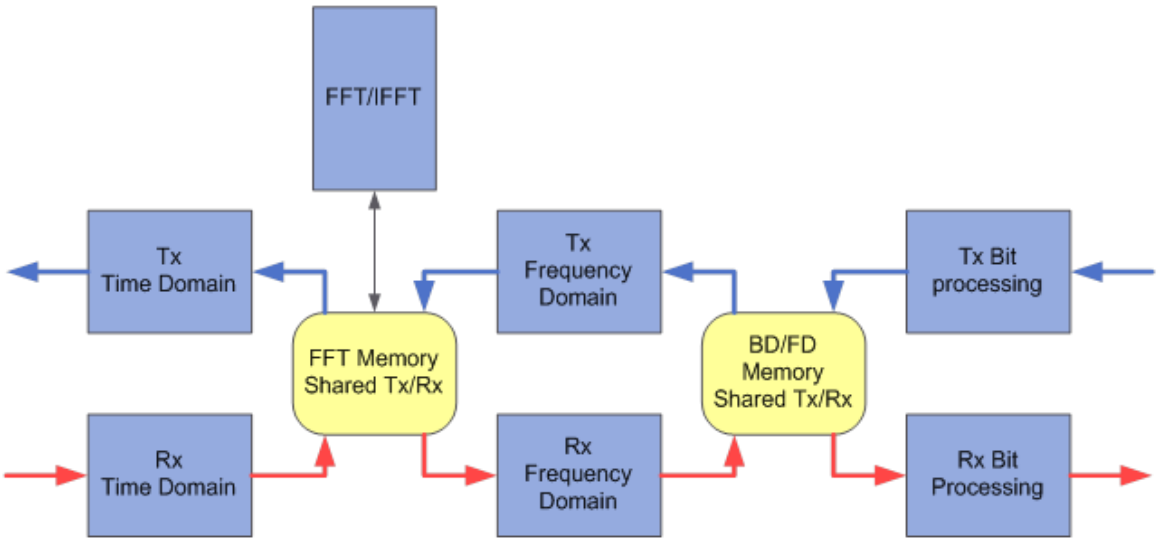


Figure 3-2 : Use of memory buffers in the RW-WLAN Modem OFDM data flow

The TD and the FD are separated by the IFFT in transmission and the FFT in reception. During reception, the TD writes data in one FFT buffer and after FFT processing; the FD reads data from an FFT buffer. During transmission, data is written to the FFT memory by the FD and read by the TD.

The FD and the BD are separated by the BD/FD memories. During reception, the FD writes data in the BD/FD memories, and the BD reads data from it. During transmission, the data is written by the BD and read by the FD. If BCC is used, interleaving and deinterleaving is partly done while writing and reading the data. If LDPC is used, interleaving is not needed, so the memories are only used for data buffering.

This flexible architecture allows the following:

- Each of the TD, FD and BD can belong to a separate clock domain if needed. Data transfer from one domain to the other is easily done using two-ports memories, read in one clock domain, and written in another one.
- Each of the TD, FD and BD works independently of one another, as soon as they have available data to process in their input buffers and as long as there is space available in their output buffers.
- Flexibility for LDPC processing: in reception, the average BD processing for one symbol can take longer than one symbol time. In this case, the chain of buffers is filled with incoming data, which is processed later, within the RX latency limits.

3.3.2 FFT/IFFT

The FFT and IFFT are the key element of an OFDM modem. There are as many IFFT as transmit paths and as many FFT as receive paths. 802.11n/ac (and preamble part of 802.11ax) 20MHz frames receptions involve 64-points FFT, 40MHz frames require 128-points FFT and 80MHz frames require 256-points FFT. Whereas the 802.11n/ax (and preamble part of 802.11ax) transmissions involves a 512-points IFFT for a 20MHz frame, a 1024-points IFFT for a 40Mhz frame, and a 2048-points FFT for a 80Mhz frame. The Modem uses a common FFT/IFFT block.

The FFT/ IFFT block I/O multiplexing and operating mode (FFT or IFFT) is controlled by the Main Tx/Rx State Machine.

- In TX mode, frequency domain samples coming from the frequency domain are transformed into time domain samples sent to the front end. A guard interval is added by using a specific FFT memory addressing.
- In RX mode, time domain samples coming from the frequency offset compensation blocks are transformed into frequency domain samples and sent to the equalizer.

3.3.3 Tx data path

The OFDM modem core TX path includes:

- In the bit processing part
  - the data scrambler,
  - one interleaver per spatial stream,
  - preambles generators
  - Puncturing and encoding for BCC
  - If included in the Modem's configuration, LDPC encoder (2 encoders in case of MU-MIMO TX support)
- In the frequency domain
  - the QAM modulation,
  - the frequency mapping & duplication, responsible for assigning the data and the pilots into iFFT bins, depending on the mode: 20/40/80 MHz, legacy, HT, HT-duplicate, VHT...,
  - the spatial expansion block, transforming the spatial streams into TX streams.

More details on the OFDM Tx data-path, along with timing information, are given in chapter4.

3.3.4 Rx data path

The OFDM modem core Rx path includes:

- In the time domain:
  - A DC offset estimate, tracking and compensation
  - A TD frequency offset estimation and compensation on the preambles

  - A synchronization (Time Boundary Estimation) block responsible for finding the symbol boundaries. This block works also in conjunction with the FD offset estimation block, which can adjust the boundaries by +/- 1 sample for each symbol to compensate for the integer part of the sampling clock offset.

- In the frequency domain:
  - A channel estimator, which estimates the NRx * Nss channel matrix H, for each data and pilot subcarriers. This estimation uses first the L-LTF in legacy mode, then the data HT_LTFs and extension HT_LTFs in all HT modes or VHT_LTFs in VHT modes or HE_LTFs in HE modes. This block also performs the channel estimation and smoothing (frequency response filtering),
  - An FD offset estimator, which computes from the rotation of pilot subcarriers the residual frequency offset after TD compensation and the sampling clock offset,
  - An FD offset compensator, which applies the phase ramp computed above,
  - An equalizer working with coefficients computed once per packet on the preamble, and proceeding to equalization and softbits generation and compression on the data subcarrier of each symbol.
- In the Bit processing chain (decoding):
  - Deinterleavers, one per spatial streams
  - Depuncturer and stream multiplexer for BCC decoding, which merges the softbits coming from each spatial stream into a single one, and replace the punctured bits by zero valued softbits
  - The Viterbi decoder with a radix-4 implementation (2 decoders in case of 2x2-80MHz support), which allows decoding two bits per cycle
  - If included in the Modem's configuration, an LDPC decoder, which manages the shortening, puncturing and repetitions schemes on the LDPC blocks of data, and decodes them using an efficient iterative algorithm. The number of iterations is adapted for each block based on the decode difficulty and the average time available over the complete packet in 802.11n/ac, and fixed iteration in 802.11ax.

More details on the OFDM Rx data-path, along with timing information, are given in chapter5.

3.4 Legacy DSSS/CCK modulation support

The RW-WLAN nX Modem supports the legacy 802.11b frames. The reception of such frames reuses the legacy RW-WLAN DSSS-CCK modem.

In the RIU, the samples stream is routed to the DSSS-CCK modem after the half-band filter, which generates the 40 MS/s stream. A re-sampling filter converts the incoming 40 MS/s stream into 44 MS/s stream, to be compliant with the rates expected by the DSSS-CCK modem.

In transmit mode, the frame length and rate are extracted from the TX Vector to activate the DSSS-CCK modem instead of the OFDM modem.

3.5 MAC-PHY interface

The MAC-PHY interface is fully described in [2]. It has two specific data paths:

- **TX path:** this is used to carry the data to be transmitted and the TX VECTOR following the MAC-PHY Interface definition (see [2]), and also some specific commands issued by the MAC HW during the frames reception. The path uses an asynchronous FIFO, all the flow management is handled by the MAC side of the interface.
- **RX path:** this path is used to carry the received data and the RX Vector. The flow control is also handled by the MAC side of the interface.

The TX Vector decoding FSM keeps track of the number of parameters received, and is responsible to determine the actual starting byte of the PSDU. The TXVector length is always the same, so its transfer delay can be used as a constant to manage the RF transceiver and PA power up sequence.

3.6 Beamforming support

3.6.1 Beamformee support HW accelerator

This block for Beamformee support is also called SVD for Singular Value Decomposition, which is the name of the operation needed to compress information into a beamforming report. It is illustrated in Figure 3-3. When the Modem receives an NDP frame, it fills the H memory with channel information. The Beamformee Support module reads the H memory and generates the compressed beamforming report upon the receipt of the NDP frame. The report is stored in the dedicated Beamformee memory. It is read read upon MAC request for beamforming protocol exchange as described in [6].

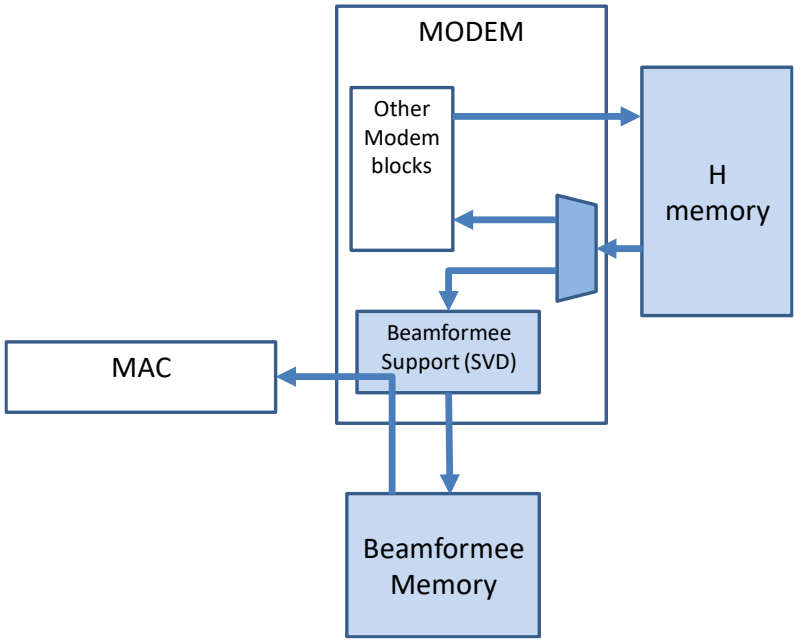


Figure 3-3 : Memory connections of the Beamformee Support Module

3.7 AHB Slave interface

The modem implements a slave AHB interface, which provides access to the following modules:

- configuration and status registers
- LDPC configuration memory

The register map is split in 6 zones. A detailed register list is provided in [3]. Some zones are reserved for modules accessed independently of the modem: CRM registers, AGC memory used by the RF control unit for AGC RAM loading, and RIU (Radar, AGC…) configuration registers

The AHB interface clock is synchronous with the platform (MAC, host CPU…) but asynchronous with the rest of the modem. This ensures that the AHB slave interface can be easily integrated into any platform, while allowing burst access to the access memory. This is mainly intended to reduce the transfer time.

The AHB slave interface implements a bridge towards the Modem registers blocks:

- **Configuration registers**, which are written by the CPU through the AHB interface, hold all programmable parameters existing in the DSP algorithm implemented in both the RX and TX chain. These parameters are supposed to be programmed at startup, and remain static during the normal operations. Hence, no synchronization to the modem clock domain is required. Configuration registers can be read back through the AHB interface.
- **Status registers,** holding variables providing status information about relevant sections of the modem that evolve at a rate slow enough to be captured by the CPU (frequency offset, STO/CPE). This is mainly be used for debug purposes. These status registers are placed in the modem clock domain, since they are updated by the hardware itself.

The configuration and status registers block RTL description is automatically derived from its XLS description, as described in [4].

3.8 Clocking scheme

3.8.1 Overview

Different parts of the Modem have different constraints regarding the minimum clock frequency needed for their correct operation. In order to easily support various configurations, the Modem has been divided into several asynchronous clock domains, listed in Table 3-1 and detailed in the following chapters.

Clock Domain	Included blocks	Required Frequency
Phy	Main TX/RX FSM and Modem's logic not included in the other domains	120 MHz
Rx Bit Domain	Rx Bit Domain	240 MHz
AHB	Registers and memory access modules	
DSSS/CCK Modem	DSSS/CCK Modem	44 MHz

Table 3-1: Modem clock domains

When possible, clock frequencies are set to a multiple of 30 or 40 MHz, so that they can be easily be derived from a common source clock (e.g. 240 MHz clock source for 40MHz ax support).

3.8.2 Rx Bit Domain clock domain

This domain contains the OFDM Core Rx bit domain (see 3.3.4).

Its target clock frequency is driven by the LDPC and Viterbi decoders. It must support the maximum data rate of a given Modem configuration, as defined by the standard.

When the Modem configuration includes the LDPC, the complete Rx Bit domain, including Viterbi and LDPC decoders, runs at 240 MHz. The LDPC decoder clock frequency has a direct link to the block performances, as it decides of the maximal number of iterations that the decoder can perform during an OFDM symbol. For good performance in a 2 spatial streams, 80 MHz bandwidth ac configuration, the LDPC decoder clock frequency must be set to 320 MHz. For good performance in a 1 spatial streams, 40 MHz bandwidth ax configuration, the LDPC decoder clock frequency must be set to 240 MHz. This frequency also allows the decoder to quickly load and unload the data, which is processed by blocks and not as a stream.

As explained below, this frequency is also sufficient for correct Viterbi operation.

When the LDPC is not included in the Modem configuration, the clock frequency is driven by the Viterbi clock speed requirement. The Viterbi decoder outputs two bits per clock cycle. Depending on the MCS, the RX path can use 1 or 2 Viterbi decoders in parallel. The corresponding Rx Path frequency is shown in Table 3-2.

Supported frames	Supported spatial-streams	Maximum throughput per Viterbi decoder (Mb/s)	Minimum Viterbi Frequency (MHz)
11n 20 MHz	1	72.2	37
11n 20 MHz	2	144.4	73
11n/ax 20/40 MHz	1	150	75
11n 20/40 MHz	2	300	150
11ac 20 MHz	1	86.7	44
11ac 20 MHz	2	173.3	87
11ac 40 MHz	1	200	100
11ac 40 MHz	2	400	200
11ac 20/40/80 MHz	1	433.3	217
11ac 20/40/80 MHz	2	585 (MCS6 with Nes=1)	293

Table 3-2: Clock requirements vs configuration for the OFDM Rx Bit Domain

3.8.3 Phy clock domain

The Phy clock domain contains the Main TX/RX FSM and the Modem's logic not included in the other domains.

Its frequency is 120 MHz

This relatively low speed has been selected to keep a high confidence on quickly achieving timing closure at synthesis stage, and to be independent from any high performance technology. For example, it would allow using only high Vt cells, thus saving leakage power. It also allows having a common RTL code both for ASIC and FPGA targets.

For MU MIMO RX or beamformee operation, a 60 MHz SVD clock is needed. This clock is fully synchronous and phase-aligned with the 120 MHz PHY clock, and belongs to the Phy clock domain.

This domain also contains the MAC-PHY interface block (MPIF), responsible for all the communication between the PHY and the MAC.

Its clock frequency must support the maximum achievable modulation throughput on the 8bit interface. The resulting frequencies for BCC encoding are shown in Table 3-3.

When the Modem configuration includes the LDPC, to manage the burst throughput of the decoder, the MAC-PHY interface runs at 160 MHz in case of 2 supported spatial streams as shown in Table 3-1.

Supported frames	Supported spatial-streams	Maximum throughput (Mb/s)	Maximum MPIF throughput (MB/s)	Minimum MPIF Domain clock (MHz)
20 MHz	1	86.7	13	30 (120 with LDPC for clock ratio)
20 MHz	2	173.3	25	30 (120 with LDPC for clock ratio)
20/40 MHz	1	200	29	60 (120 with LDPC for clock ratio)
20/40 MHz	2	400	57	60 (120 with LDPC for clock ratio)
20/40/80 MHz	1	433.3	61	60 (120 with LDPC for clock ratio)
20/40/80 MHz	2	866.7	122	120 (160 with LDPC for clock ratio)

Table 3-3: Clock requirements vs configuration for the MAC-PHY Interface Domain

3.8.4 DSSS-CCK Modem domain

The target clock frequency of the **DSSS/CCK modem** is 44 MHz. It is possible to derived this clock from a 240 MHz clock, using a sequence of 5/6/5/6/5/6/5/6 cycles out of 60. The duty cycle is not required to be 50 %.

3.8.5 Clock gating

The Modem provides several clock enables and clock inputs for each clock domain. They can be used for module level clock gating, implemented in an external Clock and Reset Manager (CRM). This allows to save power consumption in idle mode, and also when receiving and transmitting data at lower rates. To further reduce the consumption, one can use finer granularity automated clock gating insertion inside modules during synthesis.

The clocks enable commands are provided by the TX and RX sequencer FSMs, which are fed by a non-gated clock to avoid deadlocks. One exception is the register and AHB slave blocks controlled directly by the AHB clock. For debug and evaluation purposes, this automated clock gating mechanism can be overridden and all clocks can be forced on/off using a specific register.

For more details, please refer to the RW-WLAN nX Modem – Integration Guidelines [7]

3.9 Reset scheme

Each of the Modem's clock domains has an associated asynchronous reset input.

In addition, the modem is able to request assertion of the asynchronous reset to the CRM for some part of the logic, after an error during the demodulation in order to avoid any memory effect in the signal processing blocks.

In addition, reset request can be asserted by the SW to the CRM through AHB slave interface (via configuration registers) in case of potential recovery needed by the system.

3.10 Memories

3.10.1 Overview

To reduce the total silicon area of the modem, the blocks requiring intermediate data storage use embedded synchronous SRAM (single or 2- port) instead of DFFs arrays. For each module, to simplify the BIST controllers design, care has been taken to group data in a single memory using a wide data bus instead of several smaller memories.

In order to ease the integration and conform to the IP design guidelines, all technology-dependent memories macros used in the modem are implemented at the top level of the design, so that they can be easily replaced. This approach leads to a more complex interconnect, since all the data and address buses have to cross the whole hierarchy, from the module to the memory itself. In addition, extra care must be taken at synthesis to avoid any negative impact on the data access timings.

When LDPC is included, the memories must support Read before Write behavior.

The Modem needs the following memories:

- BD/FD memory: this 2-port SRAM is used for
    - BCC interleaving and deinterleaving
    - Bit Domain – Frequency Domain data buffering for clock domain crossing and LDPC burst regulation
    - STBC symbol buffering
    - HT_SIG versus L_SIG and L_DATA versus VHT_SIG_A2 discrimination
- FFT memory: is a 2-port SRAM used
    - In Tx:
        - IFFT processing (2 full symbol, 2x oversampled)
    - In Rx:
        - FFT processing (2 full symbol, 1x oversampled)
        - H memory (up to 4 LTF) – used for channel estimation/equalization/time domain sample storage
        - Sample FIFO (used during preamble processing only)
        - G Memory used for MIMO Equalization in 2x2-2 configuration
- STBC Memory: single port SRAM used for STBC Equalization
- LDPC Encoder memories: 2 instances of single port SRAM with 9 write-maskable sections, used in a ping-pong fashion to load, encode and unload data
- LDPC Decoder memories: several 2-port SRAM for input buffering, and Check Metrics, Variable Metrics, and Hard Decisions storage

- LDPC Configuration memory: single port RAM containing LDPC parameters depending on the Modem clock frequencies configuration. Its content must be updated if the Rx Bit domain or the Mac-Phy interface frequencies change. This memory is written through the Modem's AHB interface. At each decoding start, the LDPC decoder requests one access to obtain the configuration value corresponding to the packet parameters. The access is done one the AHB clock domain, and the read value resynchronized to the Rx Bit domain.
- Beamformee Support memory: this single-port SRAM is loaded with the compressed beamforming reports by the Beamformee Support module; the reports are sent to the MAC to manage beamforming protocol

Please refer to [7] for size of each memory and for each configuration. The following chapter describes memory organization when the memory is shared between several modules. When the memory is accessed by only one module, refer to the module's documentation.

3.10.2 FFT/IFFT/H Memory access and organization

The H memory is accessed by several modules inside of the Modem. The data format, accessing modules and priority scheme is described here.

The H memory can be read by:

1. the SVD beamformee module, which retrieves channel information from a previously received NDP packet to prepare a Beamforming Report. The MAC ensures that during this time, the Modem is not in Rx Mode and does not transmit a beamformed or MU-MIMO packet
2. the Time Domain Frequency Offset module (TDFO), which works on the currently received packet short preamble and long preamble before FFT processing
3. the Channel Estimation and Smoothing module, which works on the currently received packet long preamble after FFT processing
4. the Equalizer and the Interference Cancellation pre-processing (MU-MIMO only) , which starts its processing after the end of the Channel Estimation
5. the FFT/IFFT module

Note that the H memory content is also used by the FD offset estimation module for WLS matrix computation, but that the corresponding read accesses are driven by the Equalizer.

The H memory can be written, by order of priority:

1. the Time Domain Frequency Offset module (TDFO), as above
2. the Channel Estimation and Smoothing module, as above
3. the Equalizer, as above
4. the FFT/IFFT module

This is illustrated in Figure 3-4.

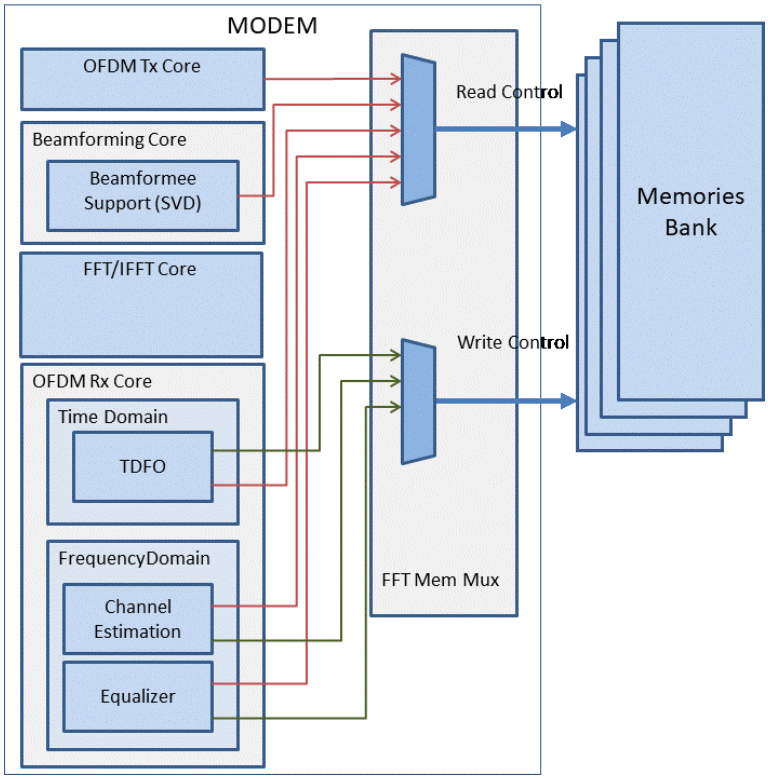


Figure 3-4: H memory connections

The H memory data mapping depends on the Modem's configuration.

- The 2nd complement of the H memory address always corresponds to the subcarrier's index.

When used for reception,

- The H memory width is divided into several sections, each containing the data for one Rx path. This is driven by the Nrx parameter (see 1.2.2)
- Each of these Rx Path sections is divided into Nr streams:
  - In 802.11ac/ax MU-MIMO configuration, Nr = Nsts,tot = 4 to support MU-MIMO beamforming report computation
  - In SU-MIMO configuration, the number of data spatial streams is 1 or 2, defined by the Nss parameter (see 1.2.2).
- Each stream is represented by a signed complex value on 13 bits for the real part and 13 bits for the imaginary part.

The H memory data format for reception is illustrated in Figure 3-5.

Warning! Note that on the memory data bus, the RX path with the highest number is on the MSB side, but that inside each RX path, the stream with the highest number is on the LSB side. Inside each stream, the imaginary part of the data is on the MSB side.

MSB LSB

	Rx path 2								Rx path 1								
	Stream 1		Stream 2		Stream 3		Stream 4		Stream 1		Stream 2		Stream 3		Stream 4		
@0	im	re	im	re	im	re	im	re	im	re	im	re	im	re	im	re	SC_0

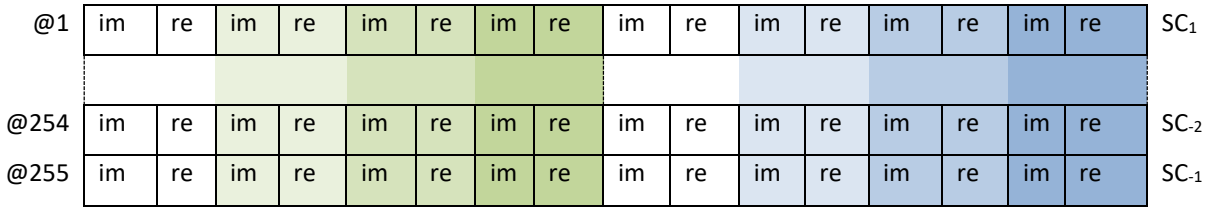


Figure 3-5: H memory RX data organization for 2 RX paths and 4 streams

3.10.3 BD/FD Memory size and sharing

The BD/FD memories are shared between the Modem Tx and Rx paths. The constraints for these two paths differ.

This configuration's constraints on the RX side are the following:

- The system must be able to process one sub-carrier per clock cycle up to 1024QAM, so the buffer width is set to 10 soft bits (50 bits)
- The system must be able to access two different memory blocks in parallel, if processing of two spatial streams is supported
- For each spatial stream, the memory block must contain two symbols, so that the Bit Domain can read and process one symbol while the Frequency Domain writes the next one. One spatial stream can contain up to 4680 softbits, so the memory block depth must be 1024

This configuration's constraints on the TX side are the following:

- The BCC interleaving process requires a data width of 52 bits per buffer.
- The system must be able to access two different memory blocks in parallel for processing of two spatial streams

Putting together these constraints, foe the 1x1:40 Mhz configuration, the memory is organized as 1 block of width 52bits and depth 1024 lines

Note that for STBC, the system must keep in memory two consecutive symbols. As STBC is only supported with one spatial stream, this is done storing the second symbol in the buffer otherwise used for the second spatial stream.

When LDPC is used, the depth of the BD/FD buffer in transmission is fully used to buffer several symbols, coming as a burst out of the LDPC block processing.

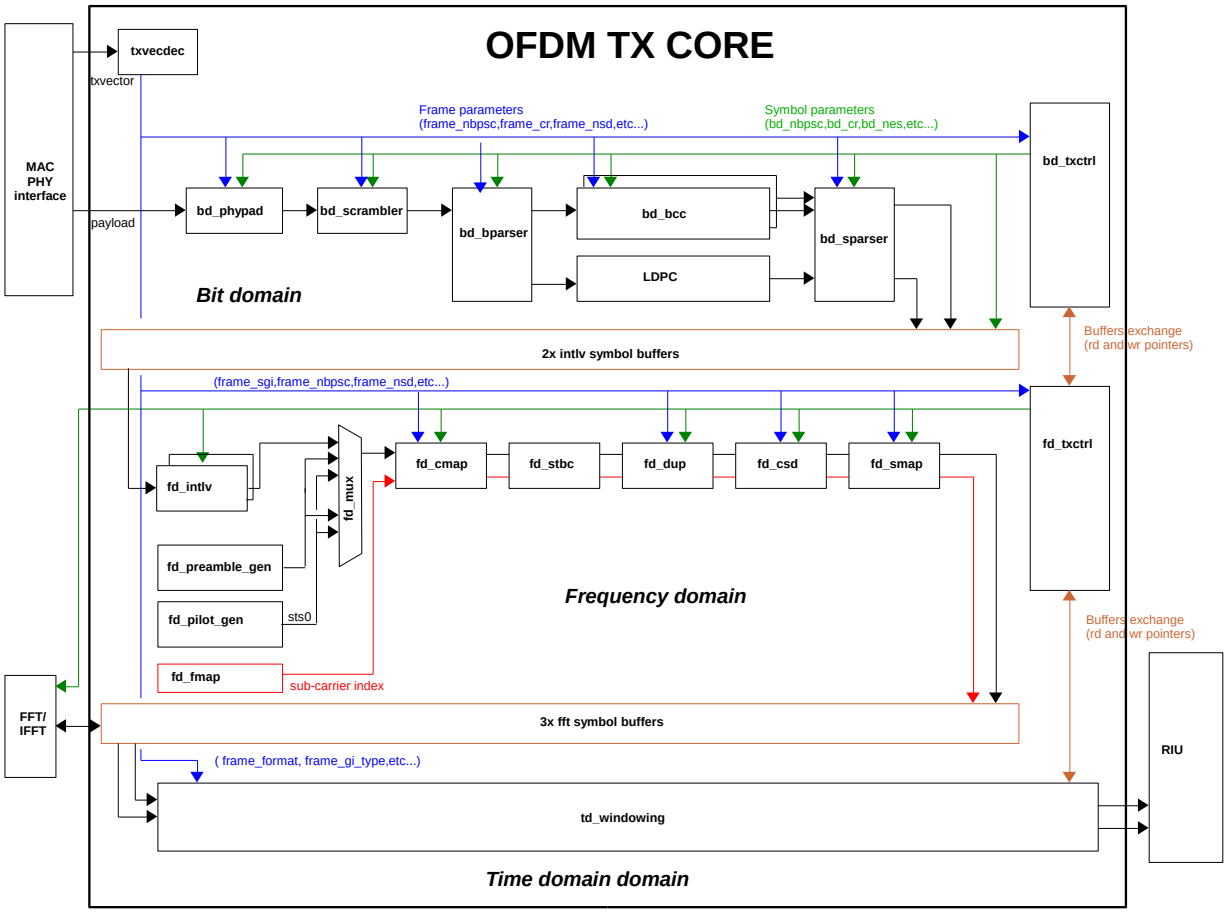
3.11 Diagnostic features

For each module, dedicated registers provide status information such as incoming signal evaluation results (power, offset estimations ...), current thresholds settings and current control FSM state.

The modem also features diagnostic ports. This can be used to record I/Q samples along the processing chain working at its normal rate with a logic analyzer. The list of the observable nodes is detailed in [3].

4 OFDM Modem Transmit Path

4.1 Block diagram



The TX data path is based on a three stages symbol pipeline architecture:

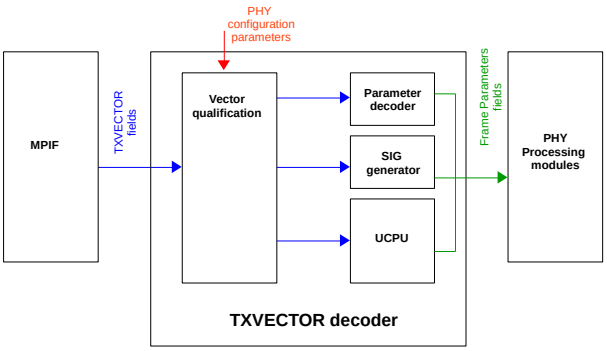
1. The bit domain. It generates the preamble SIG symbols from the TXVECTOR fields and encodes the data payload serialized by the MAC side into data symbols.
2. The frequency domain. It generates preamble training sequence symbols, data symbols pilots and splits the data symbol encoded bits from the bit domain into symbol sub-carriers.
3. The time domain, which renders the symbol time samples by adding the guard interval and applying the time windowing.

The symbol stages are delimited by memories that behave as fifos in between the different stages.

The pipeline gets all the required parameters from the txvector decoder.

The frequency domain also contains the IFFT processing which is shared with the RX core.

4.2 TX vector decoder



The txvecdec module decodes the TXVECTOR fields from the mac/phy side upon a TX request, computes the necessary frame parameters and dispatches them towards all modem sub-modules.

This function is accomplished in several steps:

- **TX vector field qualification**: the module checks the consistency of the TXVECTOR against the IEEE 802.11 standard and the modem enabled capabilities. In the case of a detected inconsistency or unsupported feature, the transmit request is then rejected and a phy error is signaled to the main FSM.
- **Frame parameters decoding:** Because many TX vector fields are synthetic, they needs to be translated to a lower abstraction level, for instance, MCS translated to coding rate and number of bit per sub-carrier. This is performed by classical look-up tables.
- **SIG generators:** The generators format the SIG symbols payload from the TXVECTOR fields and frame parameters.
- **UCPU:** Some frame parameters such as the number of data symbols (nsym) are solved by algorithms involving mathematical operations like divide, multiply, rounding, or flow control like test and branch. These algorithms are resolved by a hardwired microcode embedded into a dedicated micro-controller. The computation of the parameters lasts few micro-seconds and takes place during the modulation of the preamble training sequence (16 us).

Because some frame parameters are immediately available while some others require a delay for computation, the parameters are displayed into three different groups as shown below:

Parameter name	Description
framep1_update	Indicates the availability of parameters prefix by framep1
framep1_format	Indicates PPDU format (NONHT,NONHTDUP,HTMM,HTGF,VHT,HESU,HEMU,HETB,HEERSU)
framep1_ch_bandwidth	Indicates PPDU bandwidth (20,40,80 or 160 M)
framep1_beamformed	Indicates if frame has to be beamformed
framep1_trigbase	Indicates HE triggering method (TRIG,TRS)
framep1_nsd	Indicates number of sub-carrier per symbol for non HE frame
framep1_rulen	Indicates the RU length (26,52,106,242,484,996) for HE TB
framep1_ruindex	Indicates the RU index (1-36) for HE TB
framep1_hesubchan20	Indicates the 20M sub-band to be modulated for the pre-HE preamble part in HE TB
framep1_gi_type	Indicates the guard interval (400,800,1600,3200 ns)
framep1_heltf_type	Indicates the HE LTF type (1x,2x,4x)
framep1_nheltf	Indicates the number of HELTF symbol in the HE TB preamble
framep1_nbpsc	Indicates the number of bit per sub-carrier
framep1_dcm	Indicates the DCM modulation (on, off)
framep1_smmindex	Indicates the spatial expansion method or beamforming steering matrix
framep1_fec	Indicates the coding type (BCC, LDCP)
framep1_stbc	Indicates the STBC modulation (on, off)
framep1_cr	Indicates the coding rate (1/2, 2/3, 3/4, 5/6)

framep1_primary80	Indicates the segment containing the primary for HETB 160M
framep1_initseq	Indicates the scrambler initial sequence
framep1_beam_change	Indicates the if the pre-HE preamble is beamformed
framep1_trigger_responding	Indicates the frame is a response to a trigger frame
framep1_htdup,	Indicates the frame is HT duplicate (MCS32)
framep2_update	Indicates the availability of parameters prefix by framep2
framep2_nsym,	Indicates the number of data symbols
framep2_tpe	Indicates the duration of the packet extension (4, 8 ,12 , 16 us)
framep2_service	Indicates the value of the service field (VHTSIGB CRC only)
framep2_ndbps,	Indicates the number of data bits per symbol
framep2_ncbps	Indicates the number of coded bits per symbol
framep2_npad	Indicates the number of padding bits to finalize the last symbol
framep2_psdulen	Indicates the length in bytes of the PSDU
framep2_ncw	Indicates the number of LDPC codewords
framep2_lldpc	Indicates the length of LDPC blocks
framep2_nshrtq	Indicates the LDPC number of shortened bits (quotient)
framep2_nshrtr	Indicates the LDPC number of shortened bits (reminder)
framep2_npuncq	Indicates the LDPC number of punctured bits (quotient)
framep2_npuncr	Indicates the LDPC number of punctured bits (reminder)
framep2_nrepq	Indicates the LDPC number of repeated bits (quotient)
framep2_nrepr	Indicates the LDPC number of repeated bits (reminder)
framep3_update,	Indicates the availability of parameters prefix by framep3
framep3_lsig,	Indicates the payload of the LSIG symbol
framep3_htsig,	Indicates the payload of the HTSIG1,HTSIG2 symbols
framep3_hesiga	Indicates the payload of the HESIGA1, HESIGA2 symbols

4.3 Tx Bit Domain

The bit domain symbol stage is built from sub-blocks that forms a pipeline of bits.

4.3.1 Phy padding

The phy padding is in charge of the SIG fields and DATA field generation.

It performs the following steps:

1. Generates the SIG symbols from parameters computed by the TXVECTOR decoder module.
2. Prepends the service field to the MPDU/AMPDU provided by the MAC HW.
3. Appends the MAC byte padding to the MPDU/AMPDU in order to complete the PSDU length.
4. Appends the PHY bit padding to the PSDU in order to complete the PPDU length (NSS*NSYM*NBPSC). In HE context, it does the pre-FEC padding, post-fec padding is then handled by the stream parser module.

4.3.2 Scrambler

The scrambler performs the spectral whitening of the DATA field.

The initial sequence is provided by the TXVECTOR decoder.

4.3.3 Bit parser

The bit parser multiplexes the data bits between the LDPC or BCC encoder, or sequentially dispatches the bits to the different BCC encoders in HT, VHT where NES>=2.

4.3.4 BCC

The BCC module performs the convolutional encoding and the puncturing for the forward error coding.

4.3.5 LDPC

The LDPC module performs the low-density check parity encoding on the data field.

4.3.6 Stream parser

The stream parser is in charge of the spatial streams generation. According to the number of bits per sub-carrier and the number of spatial stream, it builds the sub-carrier components for each spatial stream with the bits provided by the encoders.

The generated sub-carrier bits are then stored into the BD/FD memories. The sub-carriers are stored following a frequency interleaving scheme.

In a HE context, the module also generates and appends the post-FEC padding bits to complete the last symbol. The post-fec padding bits are scrambled in the same fashion than the data field to prevent sample peaks.

4.3.7 Bit domain control fsm

The bit domain fsm does the sequencing of the bit domain sub-modules in order to build and encode the data field of the requested frame.

The bit domain fsm communicates with the frequency fsm by pushing SIG and DATA symbols into the BD/FD memories, and exchanging read and write pointers about the filling state of the BD/FD memories.

4.4 Tx Frequency Domain

4.4.1 Interleaver

The interleaving total function is shared between the stream parser and the interleaver.

There is one interleaver module per spatial stream.

The module fetches the symbol data sub-carrier bits from the BD/FD memories and feeds them to the constellation mapper.

In the case of a BCC symbol, it applies the remaining interleaving permutations. For LDPC symbols, interleaving operations are bypassed, and tone mapping is handled in the stream parser.

The module handles the optional STBC operation, then only one spatial is supported. STBC modulation requires two interleavers, each dedicated to the processing of odd or even data symbols.

4.4.2 Preamble generator

The module generates the short and long training symbol sequences (STF and LTF) for all supported frame formats and for all space time streams.

It is built from look up tables that contain the QBPSK and BPSK sub-carriers encoding and their associated frequencies.

It also performs the P and A matrix processing for preamble having multiple LTFs.

4.4.3 Pilot generator

The module generates the pilot sub-carriers of the data symbols for all space-time streams.

It handles the pattern rotation, scrambling for pilot modulation and also generates the associated pilot frequency.

4.4.4 Frequency mapping

The frequency mapping module works in parallel with the interleavers. It assigns to each sub-carrier displayed by the interleaver an index that corresponds to the sub-carrier frequency.

4.4.5 Constellation mapping

The module translates the sub-carrier code displayed by the interleaver into a complex number representing the phase and amplitude sub-carrier modulation.

The module supports pi/4-BPSK for STF symbols, BPSK, QBPSK, QPSK, QAM16, QAM64, QAM256 and QAM1024 for SIG and DATA symbols.

Depending of the frame parameters and modem configuration, the module also performs the amplitude normalization in order to get a constant output power.

The scaling and modulation values are implemented by look-up tables. The components of the mapped sub-carrier are displayed in the sign and absolute value to avoid storing of negative values.

4.4.6 STBC

The module performs the optional space-time block coding.

It generates 2 space time streams by combining a pair of consecutive odd and even symbols from a single spatial stream.

4.4.7 Frequency duplication

The module is in charge of several tasks:

- Frequency duplication, which is consists of duplicating preamble or data subcarriers to the desired 20MHz sub-band. This occurs for NONHT preamble symbols of HT, VHT or HE frames with bandwidth larger than 20MHz.
- The gamma rotation (+1,+j or -1) of sub-carriers according theirs indexes.
- The symbol centering to primary channel by applying an offset to the sub-carrier index prior to IFFT processing.

4.4.8 CSD

The module applies a cyclic shift with a variable delay of 50 ns to 400 ns, in steps of 50 ns, according to the IEEE definition (CSD). In frequency domain, this delay translates into a phase shift between samples.

The 1st spatial stream doesn't have any cyclic shift applied, so a pipeline delay is added to compensate for the latency introduced by the CSD modules on the other paths.

4.4.9 Spatial mapping

The module performs:

- The spatial expansion which consists of transforming the NSTS incoming space-time streams into NTX outgoing transmit streams. The linear transformation is implemented by predefined hardwired tables within the module and selected by the smmindex field of the TXVECTOR.
- The beamforming which is similar to spatial expansion except the linear transformation is defined by a Q matrix which results of beamforming calibration.

The generated transmit sub-carriers are pushed by the module into the FFT memories for IFFT processing.

4.4.10 Frequency domain control

The frequency domain fsm does the sequencing of the frequency domain sub-modules in order to modulate the symbols of the requested frame.

The frequency domain fsm communicates with the bit domain fsm by pulling SIG and DATA symbols from the BD/FD memories, and exchanging read and write pointers about the filling state of the BD/FD memories.

After that all the symbol sub-carriers are pushed into the FFT memories, the fsm triggers the FFT module for the invert FFT processing.

Once the IFFT processing is done, the fsm informs the time domain that a new symbol is ready by exchanging read and write pointers about the filling state of the FFT memories.

4.5 IFFT processing

The IFFT processing is performed by the FFT module that is shared with the RX core.

In order to ease the sub-sequent frontend, RF filtering and HE trigger base pre-compensations, the IFFT performs an x2 up-sampling on the time domain samples.

4.6 Tx Time Domain

The time domain purpose is to process and to deliver the time domain samples to the filters (RIU).

4.6.1 Guard interval and Windowing

The module reads the time domain samples from the FFT memories and constructs the symbols with the correct duration by prepending the guard interval with a cyclic repetition of symbol samples.

The module also performs the symbol windowing by averaging adjacent symbols boundaries.

The time domain communicates with the frequency domain by exchanging symbol buffers, attributes and pointers.

The samples that are played by the time domain can be flow controlled by the RIU with a classical valid/ready handshaking.

5 OFDM Modem Receiver

Figure 5-1 gives an overview of the architecture of the receive part of the modem in a configuration with 2 receive paths (Nrx = 2) and 2 spatial streams (Nss = 2) and supporting 20/40/80MHz frame. This picture allows identifying blocks which are duplicated as a function of the number of antennas and blocks (3-dimension boxes) which are doing multi-antenna processing.

The receiver architecture of the modem is split into 4 groups identified by different colors:

- Time domain (green)
- Frequency domain (blue)
- Bit Domain (purple)
- MAC-PHY interface (grey)
- DSSS/CCK modem (pink)

The receiver also includes RX state machines staying always on.

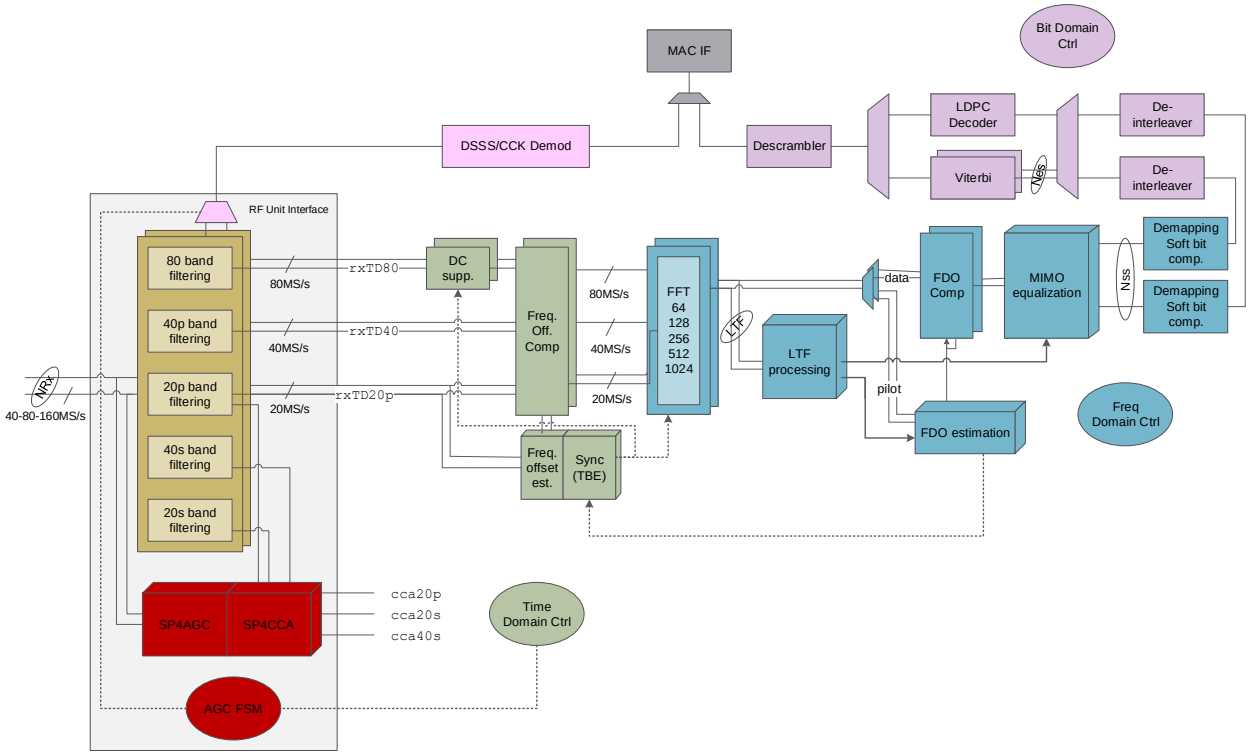


Figure 5-1: Rx top level data path with Nrx = 2, Nss = 2, 20/40/80MHz bandwidth mode

Note that LDPC decoder stands beside the Viterbi decoder.

Figure 5-2 gives an overview of the architecture in a single antenna configuration, supporting 20/40MHz mode without LDPC. This picture highlights the front-end and time-domain blocks required to support the 20/40MHz mode only and the several block reduction compared to Figure 5-1.

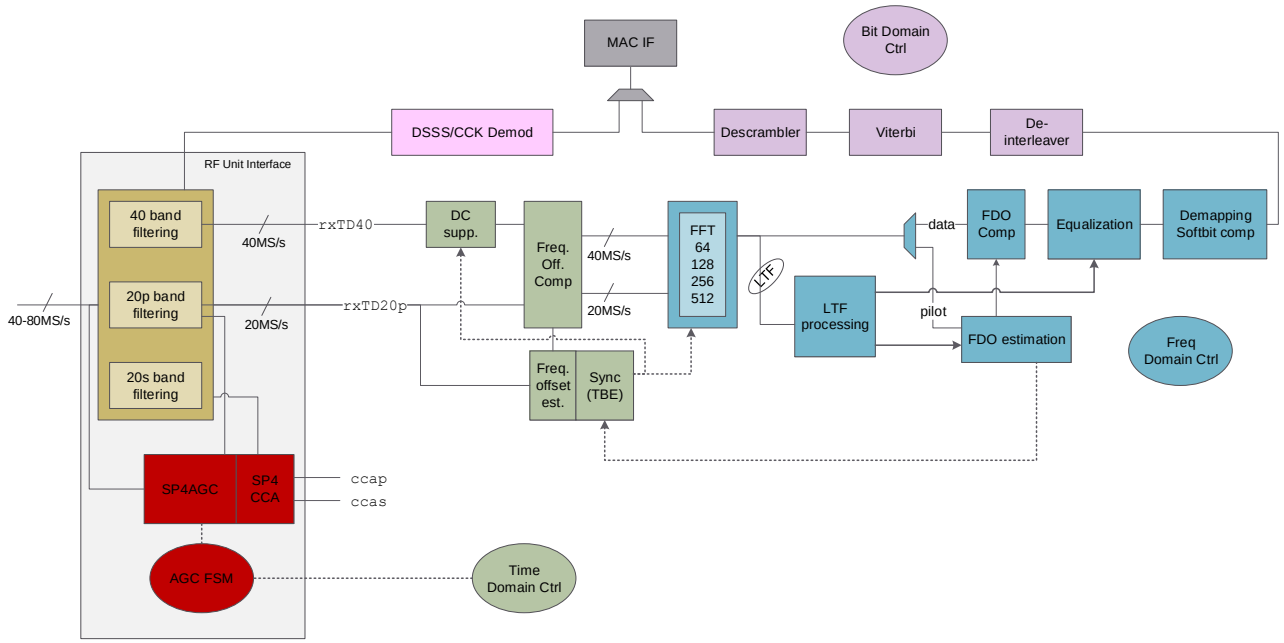


Figure 5-2: Rx top level data path with Nrx = 1, Nss = 1, 20/40MHz mode

Figure 5-3 gives an overview of the architecture in a single antenna configuration, supporting 20/40/80MHz mode with LDPC decoder. This picture highlights the front-end and FFT additional blocks required to support the 20/40/80MHz mode compared to Figure 5-2.

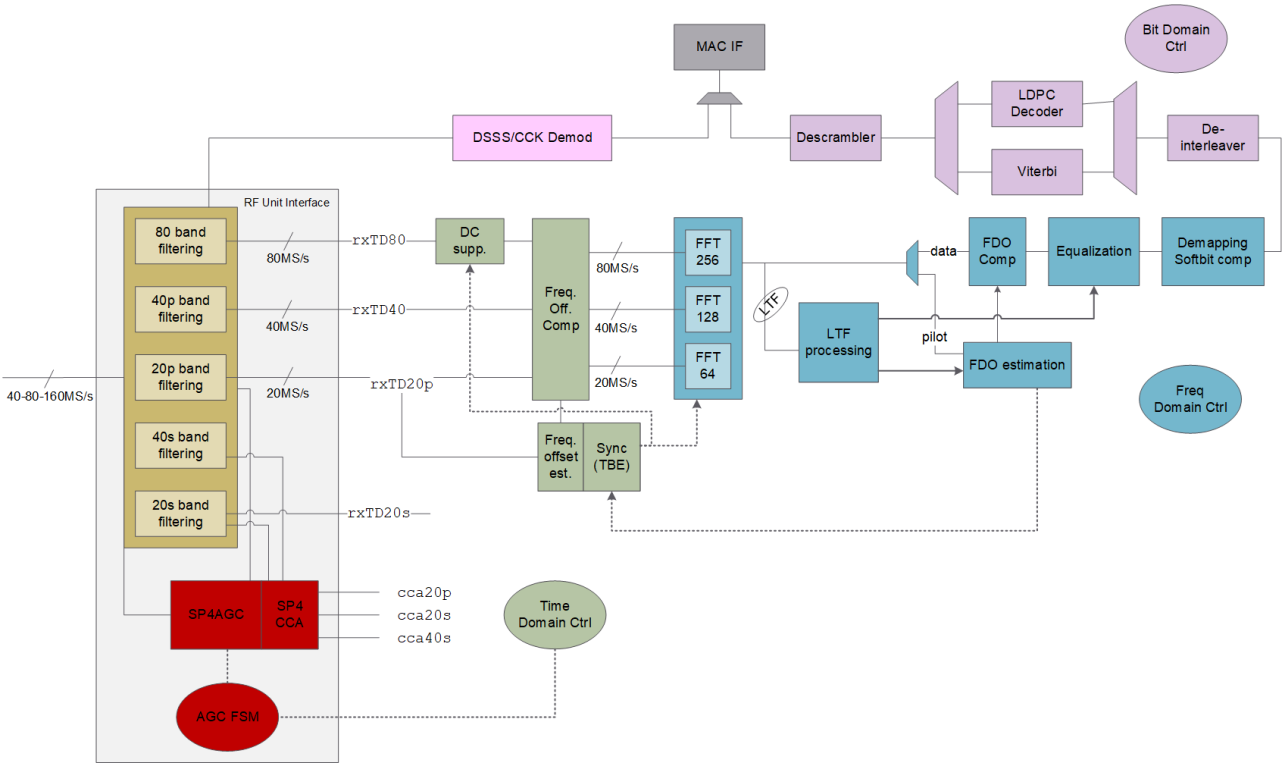


Figure 5-3: Rx top level data path with Nrx = 1, Nss = 1, 20/40/80MHz mode

5.1 Rx Time Domain

The time domain ensures the following functionalities:

- DC offset estimation and compensation.

The DC offset block estimates and compensates the constant signal of null-frequency introduced by zero IF radios. This estimation/compensation is performed only on samples which have not been frequency shift in the front-end, i.e. rxTD20 in 20MHz mode, rxTD40 in 20/40MHz mode and rxTD80 in 20/40/80MHz mode (cf Figure 5-3). Moreover, since DC offsets of each path are uncorrelated, the block is duplicated and performs a separate estimation / compensation on each path.

- coarse and fine frequency offset estimation and compensation.

The frequency offset estimations are achieved during the STF and LTF fields using the specific characteristics of these fields. For sake of simplicity, whatever the bandwidth mode and the frame bandwidth, the frequency offset estimations are performed on the 20MHz primary band.

The frequency offset is jointly estimated on all RX paths. First, a coarse frequency offset estimate is obtained by doing a 0.8µs autocorrelation on the STFs. Then, fine frequency offset estimation is done based on the 3.2µs autocorrelation of the two consecutive LTFs. This estimation is available 0.8µs before the end of the LTF.

The same frequency offset correction is applied on all the paths using a separate CORDIC rotator. In 20/40MHz mode, the frequency offset is compensated on the 20MHz and the 40MHs paths. The frequency offset estimate obtained from 20MHz samples must be divided by 2 before being applied to the 40MHz path. In 20/40/80MHz mode, the frequency offset is compensated on the 20MHz, the 40MHz and the 80MHs paths. The frequency offset estimate obtained from 20MHz samples must be divided by 4 before being applied to the 80MHz path.

- timing boundary estimation (for time synchronization).

The time boundary estimation block estimates the synchronization on the symbol boundaries by cross-correlating the received 20MHz signal with a reference sequence. The coarse frequency offset estimator provides the starting point of the time boundary estimation indicating the end of the STFs, identified by looking for a fall in the STF autocorrelation plateau.

As for the frequency offset, the timing boundary estimate is always performed from the 20MHz (primary) preamble samples. The block provides a symbol start indication, which is used as an indication on when to start filling the FFT memory. The best synchronization estimate, *TbeCount,* is given with respect to a sample counter, which starts in the same time as the TBE itself. This value is available after the middle of the 1st LTF, that is, about 1.6µs. In 20/40MHz mode, the best synchronization index estimated from the 20MHz samples must be adjusted to the 40MHz path latency. In 20/40/80MHz mode, the best synchronization index estimated from the 20MHz samples must be adjusted to the 80MHz path latency also.

The transfer into the FFT memory (and fine frequency offset compensation) starts 0.8 us before the end of the LTF and should finish less than 1.6 us (0.8 us + 0.8 us: GI of the next symbol). The full sequence is described in the Figure 5-4.

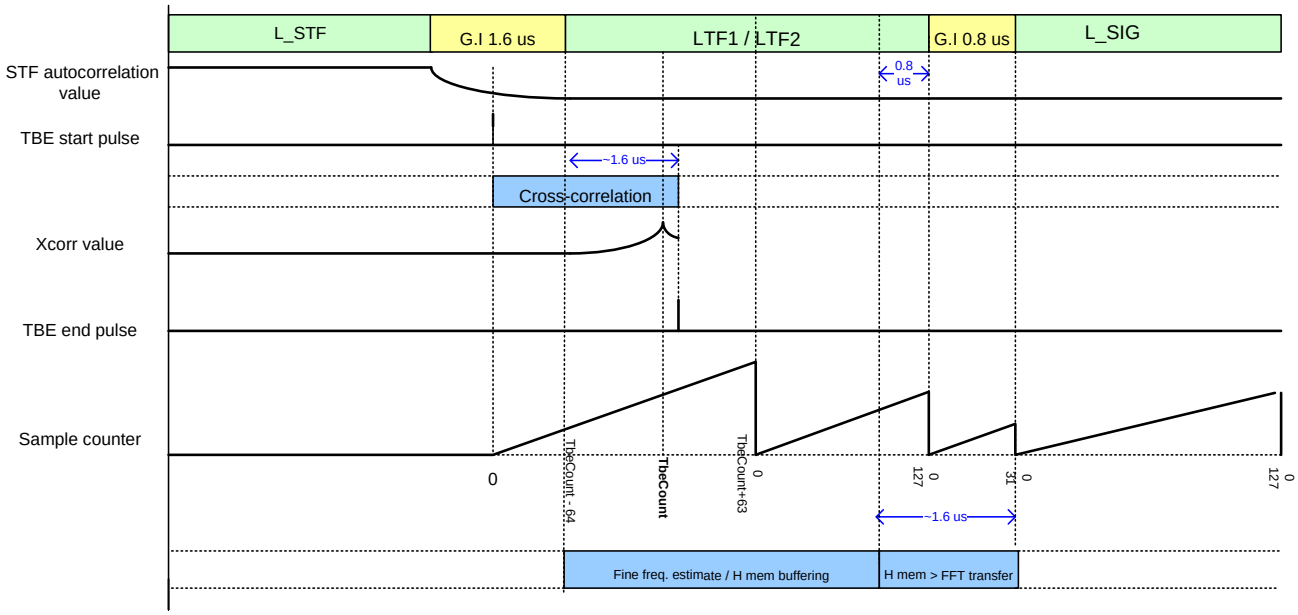


Figure 5-4: Time boundary estimate sequence

5.2 Rx Frequency Domain

5.2.1 Configuration

Depending on the frame bandwidth and on the field under demodulation, the frequency domain blocks must process a different number of data and pilot subcarriers in the considered symbol and a different number of spatial streams.

Thus, the main configuration parameters of the frequency domain blocks are:

- The RX Frequency Domain Operating Mode (FD-OpMd), controlled by the RX FSM through the register *receptionMode*. The possible formats are LM-20, LM-40, HT-20, HT-40, VHT-80, HE-20, HE-40 and HE-80.
- The number of space time stream RxNsts to be decoded.

Table 5-1 gives the value of the parameter as a function of the frame field under reception. As explained in the previous section, the frame bandwidth information comes in general from the TD frame bandwidth identification block.

Frame type		Field under reception	FD-OpMd	RxNsc	RxNsts
Format	**Bandwidth**				
NON_HT	20	L-LTF	20M-HT	56	1
		L-SIG	20M-LM	52	1
		Data	20M-LM	52	1
HT-MM	20 / 40	L-LTF	20M-HT	56	1
		L-SIG	20M-LM	52	1
		HT-SIG	20M-LM	52	1
		HT-LTF	HTSIG-CBW-MM	56/114	1 or 2 as defined in HT-SIG
		HT-Data	HTSIG-CBW-MM	56/114	1 or 2 as defined in HT-SIG
VHT	20 / 40 / 80	L-LTF	20M-HT	56/114	1
		L-SIG	20-LM	52	1
		VHT-SIG	20-LM	52	1
		VHT-LTF	VHTSIG-CBW-VHT	56/114/242	1 or 2 as defined in VHT-SIG
		VHT-Data	VHTSIG-CBW-VHT	56/114/242	1 or 2 as defined in VHT-SIG
HE	20 / 40 / 80	L-LTF	20M-HT	56/114	1
		L-SIG	20-LM	52	1
		HE-SIG	20/40-LM	56/114	1
		HE-LTF	HESIG-CBW-HE	56/114/242/484	1 or 2 as defined in HE-SIG
		HE-Data	HESIG-CBW-HE	26/52/106/242/484	1 or 2 as defined in HE-SIG

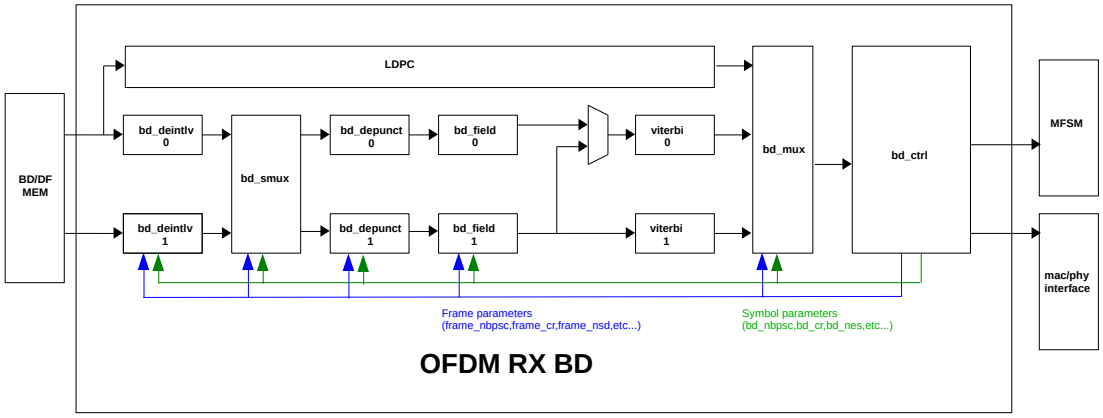
Table 5-1 : FD configuration parameters vs frame field under reception

5.2.2 Blocks description

The Frequency domain group is composed of the following blocks:

- **LTF processing**:
    - **Channel estimator:** This block retrieves the channel estimate matrix H for each spatial stream, by accumulating all the HE/VHT/HT_LTFn preambles transmitted sequentially. It computes the channel estimate for all subcarriers, using the D-HTLFs, and also the extension spatial streams, using the E_HTLTs. These last estimates are not used for data demodulation, but only to sound the entire channel dimensions. It also performs the initial channel estimation based on legacy preambles (L_LTF), which is the only one available to receive 802.11a/g OFDM frames, mixed mode HT-SIG and VHT-SIG-A and HE-SIG. This block uses a 2-port SRAM (2P-SRAM) to store the H coefficients. The two ports allow read-modify-write operations in a single cycle at different locations (write pointer lags read pointer due to operator's latencies). Therefore, the HTLFs samples received are accumulated sequentially. This accumulation performs the multiplication by the $[P_{HTLF}]^{-1}$ matrix. In addition, this block can perform a frequency–domain channel smoothing "on the fly" on the estimated H values.
    - **G matrix computation**: This block is responsible for computing the G matrix (Nsts x Nsts) from the H matrix, as required by the MMSE equalizer in case of MIMO configuration only. In SISO configuration this block is not instantiated. This computation happens once the H matrix is available. If channel smoothing is indicated in the HE/VHT/HT_SIG field, the G matrix is re-computed while H values get through the smoothing block. The G matrix computation block shares its internal multipliers with the MMSE equalizer, since it is only used during channel estimation and not during data symbols.
- **Equalization, demapping and soft-bit compression**: This block receives the mapped I/Q samples for each subcarrier, and generates the corresponding demapped softbits. When LDPC is used, the tone demapping function needed for data symbols is done in this block. The block uses the H, G matrices previously stored and det[G] (G matrix's determinant). To save memory space det[G] is computed "on the fly" for each subcarrier from G coefficients. The equalizer provides also a discriminator, which differentiates the SIGs (L_SIG/HT-SIG/VHT-SIG/HE-SIG) by detecting a BPSK/QPBSK mapping. This is used by the FSM to decide which kind of frame is received (legacy, HT, VHT or HE).
- **FD offset estimation:** This block performs two tasks:
    - The weighted least square matrix (WLS) computation performed during the 1st data symbol: this matrix depends from G and H matrix for pilot subcarriers, so the computation starts as soon as H and G are available in their respective memories.
    - Then, for each symbol, it computes the actual STO (sampling time offset) and CPE (common phase error) using the received pilots (after equalization). These estimates are used in two places:
        - They are sent to the FD offset compensation CORDICs (see below),
        - If the phase offset overflow, it also generates a sample skip command to the time domain synchronization. During the following guard interval, the time domain FSM will accordingly skip an additional sample before the symbol start, or skip a sample less.
- **FD offset compensation:** This block includes a CORDIC rotator per receive path. It applies the phase ramp correction as defined by the STO and CPE estimated previously. The compensation is performed only on the data subcarriers before the equalizer.

5.3 Rx Bit Domain



5.3.1 Bit domain deinterleaver

The BCC de-interleaving processing is split in between the frequency and bit domains. The bit domain de-interleaver reads from the BD/FD memories a chunk of 8 consecutive soft-bits per clock cycle.

The module also fetches the soft-bits for LDPC operations, interleaving processing is then bypassed.

There is one de-interleaver module per spatial stream. The first de-interleaver handles SIG and DATA soft-bits from the first spatial stream. The second de-interleaver handles the softbits of the second spatial stream.

For STBC operations, the even symbols are handled by the first interleaver, the odd symbols by the second de-interleaver.

For HE MU operations, the first interleaver provides two dedicated address generators that allow independent accesses to the different channel content soft-bits for PPDU bandwidth above 20M.

5.3.2 Stream mux

The stream mux purpose is to reconstruct the encoded data fields from the symbols composed by multiple spatial streams.

It provides two input ports for spatial streams and two output ports for It supports many configurations:

- 1 spatial stream (NSS=1) to 1 encoded stream (NES=1) for NONHT,HT,VHT and HE.
- 2 spatial streams (NSS=2) to 1 encoded stream (NES=1) for HT, VHT and HE.
- 2 spatial streams (NSS=2) to 2 encoded streams (NES=2), VHT 80M MCS>7.
- Support 1 or 2 channel content for HE MU SIGB data fields.
- STBC

It provides two input ports for spatial streams and two output ports for encoded streams.

For HE MU SIGB decoding, the two output ports correspond to the two SIGB channel content data fields.

For STBC decoding, the two input ports respectively correspond to the even and odd symbol of the spatial stream.

5.3.3 Depuncturer

The depuncturer inserts blank soft-bits into the encoded data field according the coding rate before the Viterbi decoding.

There is one depuncturer per encoded stream or per HE SIGB channel content.

5.3.4 Field splitter

The field splitter module purpose is to extract the individual encoded fields from the de-punctured encoded stream.

An individual field is a sequence of BCC encoded soft-bits terminated by a tail sequence of 6 soft-bits.

These individual fields are the LSIG, HERLSIG, HTSIG, VHTSIGA, VHTSIGB, HESIGA, DATA, HESIGB common and user block fields. After the Viterbi decoding, these fields are processed by their respective parser modules.

For HE MU, there is one field splitter module per channel content.

5.3.5 Viterbi

The Viterbi modules decodes a de-punctured encoded field.

After capturing the last tail soft-bit, the Viterbi trace-back is flushed by the corresponding field splitter module. The Viterbi consumes 4 soft-bits per clock cycles and produce 2 decoded bits per clock cycle.

For HE SIGB channel content decoding, the outputs of the two field splitter is multiplexed and processed sequentially so a single Viterbi decoder is required to handled two channel content.

5.3.6 LDPC decoder

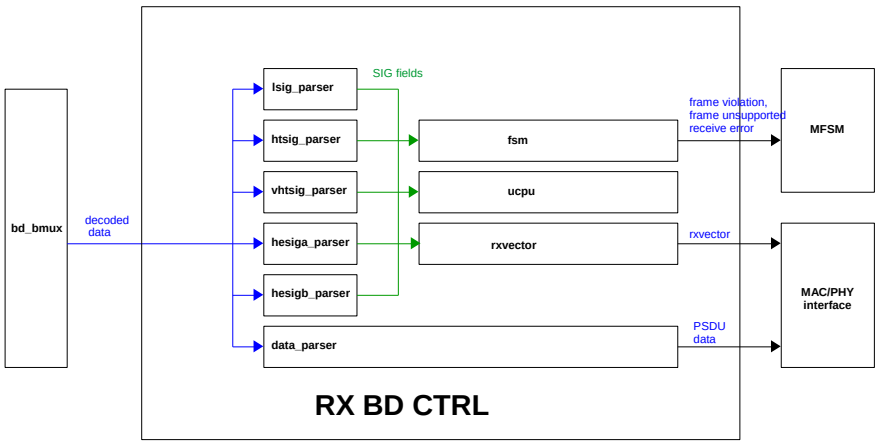
The LDPC module is used to decode the data part of LDPC encoded frames. It works on blocks of data which boundaries do not necessarily coincide with OFDM symbol boundaries, and outputs data bytes towards the bit mux module.

5.3.7 Bit mux

The bit mux is used to re-construct the data bytes from the pair bits delivered by the Viterbi decoders.

- For BCC NES=1, a data byte is reconstructed from 4 pairs of bits pushed by the viterbi decoder.
- For BCC NES=2, a data byte is reconstructed from 2 pairs of bits pushed by the 2 viterbi decoders. The bit combination does the revert operation defined by the standard as the bit parsing.
- For LDPC, the byte is straight forward produced by the LDPC decoder.

5.3.8 Bit domain control sub-system



The bit domain control is a sub-system that contains many functions:

- A dedicated parser for each kind of field (LSIG, HTSIG, SIGB user block field, etc…)
- A FSM that controls all bit domain sub-modules, does the frame detection and frame symbol sequencing.

- A dedicated parameter calculator that computes parameter from complex equations (NSYM, PSDULEN, HE pre-FEC padding factor, etc…)
- A RXVECTOR parameter generation for MAC communications
- A frame parameters decoder that describes data symbol characteristic for bit domain sub-modules.

The bit domain control also communicates with the main fsm to signal valid, unsupported or invalid frame.

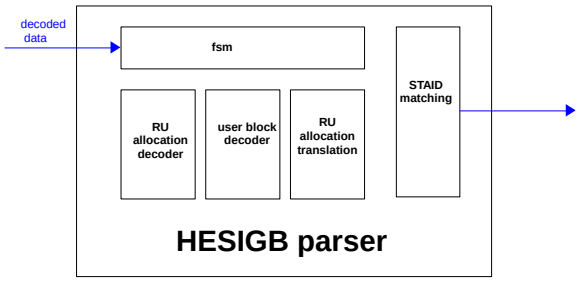
5.3.8.1 LSIG/HTSIG/VHTSIG/HESIGA parser

These parsers extract all the xxSIG sub-fields such as rate, length, bandwidth, fec, aggregation, gi_type, etc and parity and decode the corresponding frame parameters such as coding rate and the number of bit per sub-carrier.

For HE frame detection, the LSIG parser also parses the HERLSIG symbols and compares it with the previously captured LSIG field.

The parsers also checks against error (parity) or inconsistencies (invalid length).

5.3.8.2 HESIGB parser



The HESIGB parser is dedicated to the analysis of the common and user block fields of the channel contents. It can alternatively switch to any channel content to advance in the HE SIGB decoding.

It provides

- A FSM that takes over the bit domain FSM for the control of the data field splitter modules.
- A RU allocation decoder that retrieves from a RU allocation code all the associated informations such the 20M sub-band splitting in RU, the number of user per RU, the length of the different RU, etc…
- A user block decoder that extracts the user information such STAID, MCS, NSTS, FEC, etc…
- A RU allocation translation that associates the user field position with the corresponding RU position in the different 20M band.

All user block fields are decoded, the corresponding STAID field is compared against a targeted STAID. If a STAID match occurs, all the user parameters such RU index, RU length and others parameters are captured and hold for the DATA field decoding.

5.3.8.3 Data parser

The data parser is used to extract the service field and to feed the MAC/PHY data interface from the decoded bytes.

5.3.8.4 RX BD FSM

The bit domain could be split into two sub-domains: the symbol and field sub-domains. Where the sub-domain boundary is delimited by the stream parser.

The field sub-domain is mainly controlled by the parsers modules while the symbol sub-domain is control by the RX BD FSM.

The FSM detects the format of the incoming frame and then decode the symbols sequence by dispatching control signals to all bit domain modules.

The FSM also communicates:

- With the frequency domain by exchanging symbol buffers and pointers about the state of the buffers.
- With the Main FSM about the detection of a valid frame, an unsupported frame of a violated frame.

5.3.8.5 RX BD UCPU

The module instanciates a UCPU modules and implements an hardwire firmware to compute all required parameters such PSDULEN , FIELD len , number of symbol, LDPC parameters, etc…

5.4 Receiver sequencing

5.4.1 Frame format handling diagram

The Figure 5-5 illustrates the frame type decision (arrows) and the processing done on each symbol.

The top of the figure provides the different frames format that the modem can receive.

The rest of the figure represents, the processing done on the various part of the modem (Frequency Domain, Bit Domain) and the decision tree.

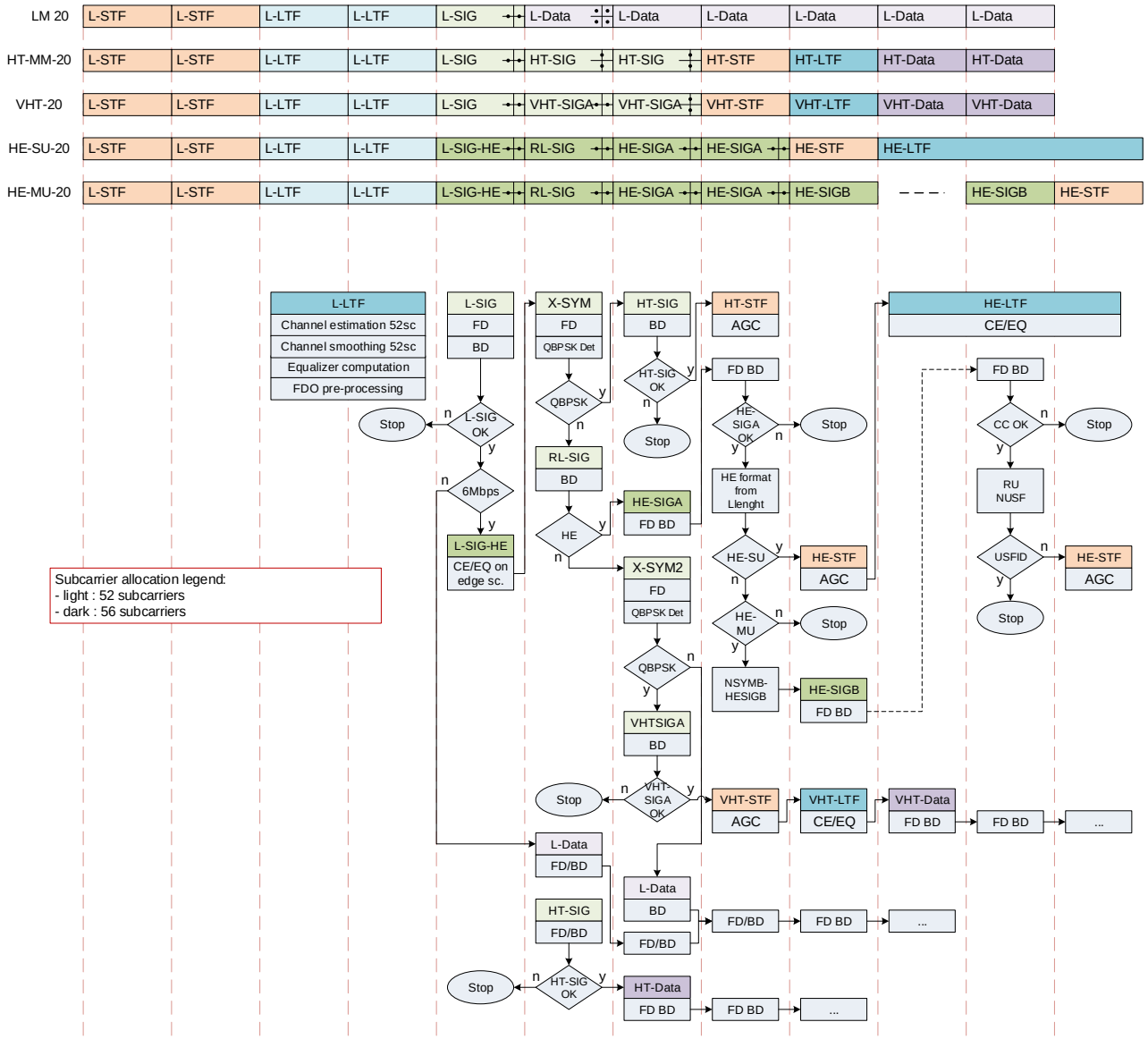


Figure 5-5: Rx modem core state-machine in 20MHz configuration

5.4.2 Receiver State Machine

The receiver state machine is enabled by the global RX/TX state machine. It is split into 3 FSMs, each in their proper hierarchy and clock domain in case of the Bit Domain:

- **Time domain FSM**: This FSM controls all the time domain related operations. It is triggered by the AGC FSM. It controls the initial frequency offset and DC offset estimation, and the synchronization process. It is responsible for getting the symbol boundary information from the TBE block, as illustrated on Figure 5-4, section 5.1. It maintains the sample timing and triggers a new FFT computation each time a full symbol has been received.
- **Frequency domain FSM**: This state machine controls the packet type decision and the demodulation process. It implements the decision tree described in section 5.4.1. It is linked to the time domain FSM through the FFT memory controller.

- **Bit Domain FSM**: This state machine takes care of all decoding operations (Deinterleavers, Viterbi decoder, LDPC). It controls the signal fields reception (L_SIG, HT_SIG, VHT_SIG or HE_SIG), and the data payload reception itself. At the end of each field, it flushes the Viterbi decoder. It works on a symbol basis, knowing when a new symbol is available by comparing the Bit Domain and Frequency Domain pointers to the BD/FD Memory (see 3.10.3)

5.5 Receiver timings

5.5.1 Overall latency

The overall modem latency is summarized in the Table 5-2, assuming a long G.I (symbol duration: 4 us).

Module	Latency (us) 20 MHz BW 20MHz frame	Latency (us) 40 MHz BW 40MHz frame
RX frontend (filters)	0.4	0.3
Frequency offset correction	0.2	0.1
FFT (64/128/256/512)	1.8	2
H matrix smoothing / G matrix computation (1)	2	1.5
WLS matrix computation	1.1	0.6
Equalization	1.5	1.3
Viterbi filling from de-interleaver	1.7	1.7
Viterbi flush (120 bits TB, 2 bits/cycle, 80/160/240 MHz IF)	0.8	0.3
Total	**8.9**	**6.7**

Table 5-2: Modem RX latency for BCC frames

Please refer to 1.3 for the SIFS split.

The overall modem latency for LDPC frames is summarized in the Table 5-3, assuming a long G.I (symbol duration: 4 µs).

Module	Latency (us) 20 MHz BW 60 MHz Phy clock	Latency (us) 40 MHz BW 120 MHz Phy clock
RX frontend (filters)	0.4	0.3
Frequency offset correction	0.2	0.1
FFT (64/128/256/512)	1.8	2
H matrix smoothing / G matrix computation (1)	0	0
WLS matrix computation	1.1	0.6
Equalization	1.5	1.3
LDPC	7	7
Total	**11.4**	**10.2**

Table 5-3: Modem RX latency for LDPC frames

6 Common

6.1 FFT and IFFT

The FFT and IFFT processing is performed by the same module and is shared between the RX and TX cores.

The FFT/IFFT computation is based on a form of the Cooley-Tuckey algorithm by iterating coefficients through a butterfly pipeline. The coefficients are encoded and computed in a proprietary complex floating point format (13 bits mantissa, 4 bits exponent) that minimizes the rounding noise.

Depending of the area/performance/modem configuration trade-off, 2, 4 or 8 butterflies are instantiated for computation parallelism. For instance, the 2 butterfly architecture is used on HE-20MHz, while 4 butterfly is on HE-40MHz.

6.2 UCPU

Some parameters require for modulation/demodulation are not explicitly defined by the TXVECTOR or RXVECTOR, so they need to be computed from quite complex math equations with operations such multiply, divide, modulo, min, max, rounding etc… as shown in few the examples below:

$$N_{\mathrm{SYM\_init},\,u} = \begin{cases} \left\lceil \dfrac{8 \cdot \mathrm{APEP\_LENGTH}_u + N_{service} + N_{tail} \cdot N_{ES,u}}{N_{DBPS,\,u}} \right\rceil & \text{when user } u \text{ uses BCC} \\ \left\lceil \dfrac{8 \cdot \mathrm{APEP\_LENGTH}_u + N_{service}}{N_{DBPS,\,u}} \right\rceil & \text{when user } u \text{ uses LDPC} \end{cases} \quad (22\text{-}64)$$

Some others require condition testing and branching as follow:

d) Compute the number of bits to be punctured, N_{punc}, from the codewords after encoding, as shown in Equation (19-38).

$$N_{punc} = \max(0, (N_{CW} \times L_{LDPC}) - N_{avbits} - N_{shrt}) \quad (19\text{-}38)$$

If $\left((N_{punc} > 0.1 \times N_{CW} \times L_{LDPC} \times (1-R)) \text{ AND} \left(N_{shrt} < 1.2 \times N_{punc} \times \frac{R}{1-R}\right)\right)$ is true OR if $(N_{punc} > 0.3 \times N_{CW} \times L_{LDPC} \times (1-R))$ is true, increment N_{avbits} and recompute N_{punc} by the following two equations once:

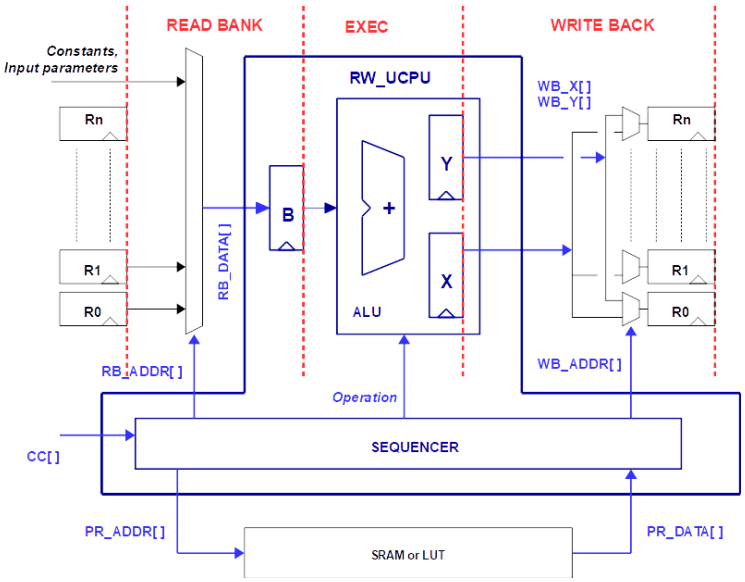
$$N_{avbits} = N_{avbits} + N_{CBPS} \times m_{STBC} \quad (19\text{-}39)$$

$$N_{punc} = \max(0, (N_{CW} \times L_{LDPC}) - N_{avbits} - N_{shrt}) \quad (19\text{-}40)$$

These equations and algorithms are solved by a micro-programmed approach.

The module is a small 3-stages pipelined microcontroller with:

- A LUT that implements the micro-coded firmware, around 300 opcodes of 17 bits are sufficient to solve all the parameters required by the 802.11 standards.
- An ALU that is built around 32 bits adder that does add, subtract, and iteratively multiply and divide operations.
- A sequencer that fetches, executes the micro-code, and does conditional branching.
- A register file that provides the read operands and stores the ALU result. It contains intermediate registers and parameters.


READ BANK
EXEC
WRITE BACK
Constants,
Input parameters
RW_UCPU
WB_X[ ]
WB_Y[ ]
Rn
R1
R0
RB_DATA[ ]
B
+
Y
X
ALU
Rn
R1
R0
RB_ADDR[ ]
Operation
WB_ADDR[ ]
CC[ ]
SEQUENCER
PR_ADDR[ ]
SRAM or LUT
PR_DATA[ ]

7 Design recommendations

7.1 Configurable Implementation

As described in 1.2.2, the modem IP RTL code is configurable with parameters to implement different modem configurations.

These parameters are used in the blocks RTL code to define their internal bus size, and the numbers of identical sub-blocks that needs to be instantiated. To allow this configurability, operators are shared amongst subcarrier, or shared between two phases of the algorithms. For example:

- Multipliers used in the G matrix computation in the MIMO equalizer are re-used to perform the actual equalization since both don't append at the same time.
- Since the modem is half duplex: FFT and front-ends blocks, interleaver/de-intleaver memories are shared between the TX and RX stream.
- The H memory is shared between the time domain frequency offset estimation, the channel estimation and the Tx of beamformed packets.

The Viterbi decoder is also an exception. For Nss=2, the Viterbi decoder needs to provide a 300 Mbps/s throughput in 802.11n configuration. Hence the choice of a high performance Radix-4, register exchange architecture. For Nss=1, the Viterbi decoder has only to sustain 150 Mbps/s, so a memory based, Radix-2 implementation is enough. Currently, only Radix-4 is implemented in the RTL code.

The following Table 7-1 summarizes the configuration options.

Block	Parameters	Nb Instances
Front-ends	BW	Max (Ntx , Nrx)
DC I/Q estimation	-	Nrx
Freq Offset estimation	-	Nrx
FFT	BW (memory size)	Max (Ntx, Nrx)
FD offset compensation	-	Nrx
Equalizer	BW Nsts , Nrx	1
De-interleavers	BW (memory size)	Max (Ntx , Nrx)
FD offset estimation	BW Nsts , Nrx, Nss	1
Viterbi	Traceback length	2 to support 80MHz, Nss=2, MCS 7 to 9, else 1
Interleavers	BW (memory size)	Max (Ntx , Nrx)

Table 7-1: Configuration options

References

[1] IEEE P802.11™- Wireless LAN Medium Access Control (MAC), and Physical Layer (PHY) specifications

[2] MAC/PHY interface Functional Specification, RW-WLAN-nX-MAC-PHY-IF-FS, RivieraWaves (Documents/Specifications/HW/MAC-PHY_IF/FunctionalSpecs)

[3] RW-WLAN-nX IP Modem User Manual, RW-WLAN-nX-Modem_UM, RivieraWaves (Documents/UserManuals/Modem)

[4] XLS registers description format - RTL generation scripts description, RivieraWaves

[5] RW-WLAN-nX Karst Radio Controller Functional Specification, RW-WLAN-nX-RC-KARST-DD, RivieraWaves (Documents/Specifications/HW/Modem/DesignSpecs/RIU)

[6] IEEE P802.11ac™-2013 Wireless LAN Medium Access Control (MAC), and Physical Layer (PHY) specifications Amendment 4: Enhancements for Very High Throughput for Operation in Bands below 6GHz

[7] RW-WLAN-nX-Modem Integration Guidelines, RW-WLAN-nX-Modem-IG, RivieraWaves (Documents/Specifications/HW/Modem/IntegrationGuide)