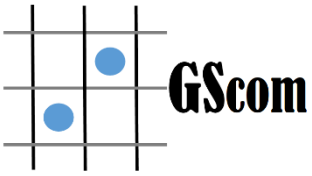


Broadband Wireless 1024-QAM Modem

System Document

Broadband Wireless 1024-QAM Modem

Table of Contents

© GScom, Corp.

1. Introduction

We developed a broadband wireless modem to achieve Giga bit range high speed data communication for both microwave and mm-wave radio systems. The broadband wireless modem is a single carrier QAM modem for frequency efficient modulation and can be used in broadband wireless communication systems for fixed wireless backhaul communication, cellular access networking, or small cell backhaul communication. The broadband wireless modem supports modulation levels up to 1024-QAM and is a very robust to phase noise, developed for point-to-point or point-to-multi point microwave and mm-wave radio systems. The broadband wireless modem achieves the high speed data communication capability through advanced technology development in the modulation, coding, signal processing, and system operation areas. From this point on this document will refer to the broadband wireless modem as the "small cell backhaul modem" for convenience.

2. System Description

This section provides a system overview and functional description of each system's operational mode.

2.1System Overview

The broadband wireless modem (or small cell backhaul modem) is a base band modem and can support any radio frequency communication system. The modem supports very high speed data communication capability with very low latency to meet the requirements for next generation wireless communication systems. The modem provides the following system features:

- Various channel bandwidths of 28/56/112 MHz
- Various modulation schemes: 4QAM/16QAM/64QAM/256QAM/1024QAM
- Gray code mapper/de-mapper with very low latency LLR decoding
- Joint estimation/correction of carrier recovery and equalizer
- Architecture-aware Parallel LDPC code
    - Variable code lengths (2160/4320/8640/17280) with mother code rates of 1/2 & 3/4

The broadband wireless modem (small cell backhaul modem) performs four operational modes to get into the data communication from the initial operation of acquisition:

- Carrier frequency acquisition mode.
- Symbol synchronization mode.
- Frame synchronization mode.
- Data communication mode.

The broadband wireless modem shows excellent performance such as:

- Excellent performance of phase noise mitigation by combining DFE and Carrier Recovery.
- Excellent coding gain from LDPC.
- Real time processing of the de-mapping and its performance.
- Very reliable Structured System Operation Architecture.

2.1.1 High Level System Block Diagram

© GScom, Corp.

A variety of data transport types can be used as the interface to the small cell backhaul modem (or broadband wireless modem) such as E1/T1 ports, Ethernet port, and General Purpose Interface (GPI) port.

Figure 1 shows the high level system block diagram of the small cell backhaul modem showing the input and output ports of the communication system.

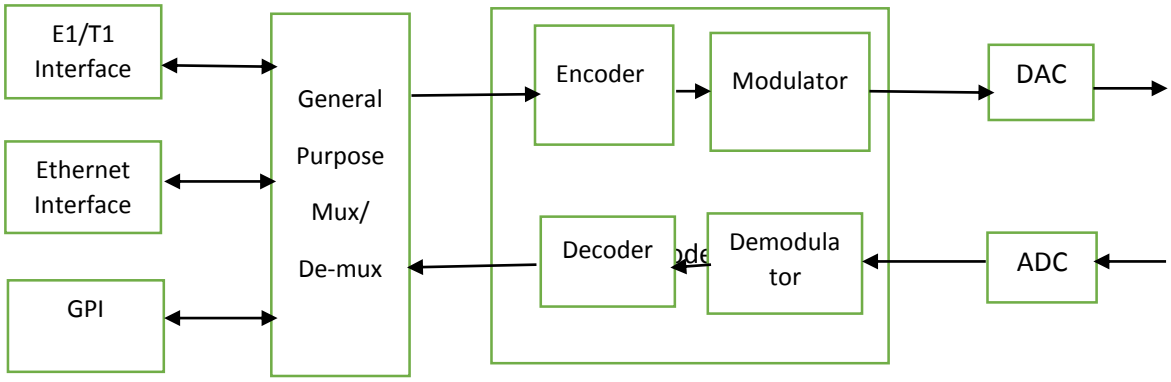


Figure 1: High Level Block Diagram of Wireless Communication System

For the transmit direction (TX), the user data is obtained from the physical interface and transferred to the general purpose multiplexer (GPM). The GPM organizes data from multiple sources and builds data frames to be transmitted and transfers the data frames to the encoder of the modem. The modem encodes and modulates the data, and then sends the modulated digitally formatted data to the digital-to-analog converter (DAC) which converts the data into an analog signal for radio frequency modulation and then sends the analog data to the transmit antenna.

For the receive direction (RX), the analog-to-digital converter (ADC) receives the analog signal, converts it into a digital signal and then sends it to the modem. The modem demodulates and decodes the digital data, and then sends it to GPM. The GPM receives the digitally formatted data from the modem and then distributes the data to the corresponding addressed user data interface.

© GScom, Corp.

2.1.2 High Level System Operation

The broadband wireless modem (or small cell backhaul modem) performs four operational steps to set up the data communication link following initialization of the communication system- coarse carrier acquisition, symbol synchronization, frame synchronization and data communication.

2.1.2.1 Coarse Carrier Acquisition and AGC Operation Mode

Oscillators in two separate communication systems (transmitter and receiver sides of each system) may have more than a few tens of kHz of frequency offset between them at the initial stage of communication. The coarse carrier acquisition process achieves the frequency offset reduction within a few KHz using feed-forward carrier estimation/correction techniques. In the carrier acquisition mode the automatic gain control (AGC) process keeps the ADC input signal at a certain level to avoid signal clipping. The coarse carrier acquisition process employs the carrier acquisition frame and then takes the cross correlation of two consecutive and repeated preamble symbol blocks of the carrier acquisition frame, and then obtains the offset frequency estimate by taking the arc tangent of the averaged correlation value.

The coarse carrier acquisition uses the preamble symbols in the carrier frequency acquisition frame. The carrier frequency acquisition frame size is 4096 symbols. The carrier frequency acquisition frame consists of 4096 symbols of repeated (0, 2) QPSK symbols as can be seen in Figure 2.

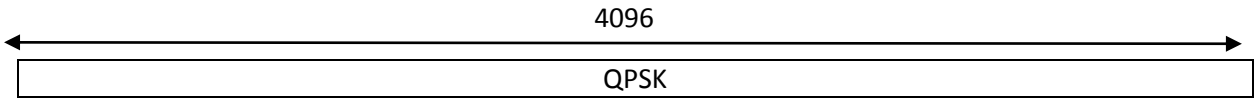


Figure2. Carrier frequency acquisition frame configuration

2.1.2.2 Symbol Synchronization and AGC Operation Mode

Symbol synchronization uses the preamble symbols in the acquisition frame and calculates the timing error using the following delayed locked loop technique:

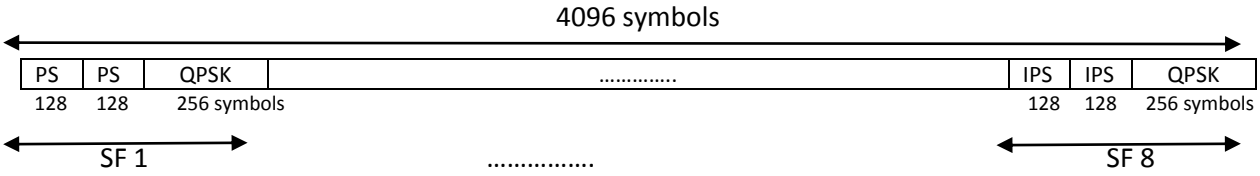
$$\text{Timing Error} = (s(n+2) - s(n)) * s(n+1)$$

Where, s(n) is a complex symbol signal and n is the symbol number of the timing sequence. The type II second-order loop filter uses the timing error estimate to track the symbol timing. The phase of the symbol timing drives the ADC output timing. Note that the symbol synchronization block also works together with the AGC which keeps the input signal level constant with a second-order loop filter.

© GScom, Corp.

The symbol synchronization mode uses the acquisition frame. The acquisition frame size is 4096 symbols. The Acquisition Frame consists of eight sub frames where each sub frame consists of identical 256 preamble symbols and 256 symbols of repeated (0, 2) QPSK symbols from sub-frame 1 to sub-frame 7. The last sub-frame (sub-frame 8) has a sign inverted 256 symbols of preamble, instead of the same 256 preamble symbols as those in the previous sub-frames. The 256 preamble symbols consist of two identical blocks of 128 preamble symbols. The block of 128 preamble symbols are obtained from the 256 Gold Sequence Generator after taking the QPSK modulation of the 256 Gold sequence. The 256 Gold Sequence Generator block consists of two 8-bit shift registers with several XOR gates between those two registers.

Figure 3 illustrates the Acquisition frame configuration.



Note: PS: Preamble symbols, IPS: inverted signed preamble symbols, SF 1: sub-frame 1, SF8: sub-frame 8

Figure3. Acquisition frame configuration

2.1.2.3 Frame Acquisition and Carrier Frequency Acquisition Mode

The frame acquisition and carrier frequency acquisition process also uses the acquisition frame shown in Figure 3 in the symbol synchronization section. The acquisition frame consists of eight sub-frames. Each sub-frame, from sub-frame 1 to sub-frame 7 in the acquisition frame consists of identical 256 preamble symbols. The last sub-frame (sub-frame 8) has a sign inverted 256 preamble symbols, instead of the same 256 preamble symbols as those in the previous sub-frames. The 256 preamble symbols consist of 128 repeated (0, 2) QPSK symbols. The 256 preamble symbols consist of two identical blocks of 128 preamble symbols. The 128 preamble symbols are obtained from the 256 Gold Sequence Generator and then taking the QPSK modulation on the 256 Gold sequence data.

The preamble symbols in each of the first seven sub-frames of the acquisition frame are used to identify the timing information of the sub-frame. The preamble symbols in the last sub-frame (sub frame 8) in the acquisition frame are used to identify the timing information of the acquisition frame. Fine carrier frequency acquisition can be obtained by applying a feed-forward carrier frequency estimation/correction technique to the 256 symbols of the repeated (0, 2) QPSK symbols in each sub-frame in the acquisition frame.

Continuous fine carrier tracking is performed using the phase locked loop (PLL) in part of analog baseband modulation process through a voltage controlled oscillator (VCO).

© GScom, Corp.

2.1.2.4 Data Communication Mode

After obtaining the symbol timing, frame timing and carrier frequency acquisition, data communication takes place using the Data Frame. The Data Frame is 4744 symbols long and consists:

128 preamble symbols used for the training equalizer

128 pilot symbols used for the training equalizer and the channel estimation

16 physical layer control (PLC) symbols used for communication with the upper layer

16 automatic coding and modulation (ACM) symbols used for communication with the upper layer

2160 Data symbols

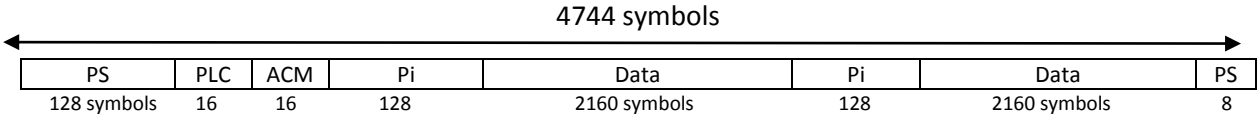
128 pilot symbols used for the training equalizer and the channel estimation

2160 Data symbols

8 preamble symbols used to take care of the delay in the equalizer

The preamble, pilot, PLC, and ACM symbols are all QPSK modulated symbols for reliable control communication. The data symbols are QAM symbols for high speed data communication. Note that the data communication block also works together with the phase recovery for phase noise mitigation and the AGC which keeps the input signal level constant using a second-order loop filter.

Figure 4 illustrates the data frame configuration.



Note:

PS: Preamble symbols for frame tracking

PLC: Physical layer control for communication with upper layer control software

ACM: Adaptive control and modulation parameter for communication with upper layer control software

Pi: pilot signal used for channel condition estimation and carrier frequency tracking

Data: Coded symbol data for data communication

Figure 4. Data Frame configuration

© GScom, Corp.

3. Functional Description

3.1 High Level System and Function Block Description

As can be seen in Figure 5, the high level modem system consists of both a transmitter (Tx) and a receiver (Rx) section. The encoder and modulator in the transmitter receives data from the general purpose multiplexer (GPM) and creates the data frame for transmission through the DAC and analog radio frequency devices to the antenna. The encoder adds low density parity check (LDPC) parity bits to the input data bits to create LDPC encoded data formatted for forward error correction (FEC). The LDPC encoded data bits go to a Mapper to map those bits into symbols according to the selected modulation scheme. The symbols are then modulated into real signals (I) and imaginary signals (Q). The digital format of the airframe is converted to analog signals by a digital to analog converter (DAC) and is then sent for transmission.

At the receiver, the analog to digital converter (ADC) receives the analog signal, converts the received analog signal to a digital signal and then sends it to the demodulator block. The digital signals are processed by the demodulator, sent to the De-mapper for converting the demodulated symbols to bits according to the selected modulation scheme, and then sent to the decoder for error correction and for decoding.

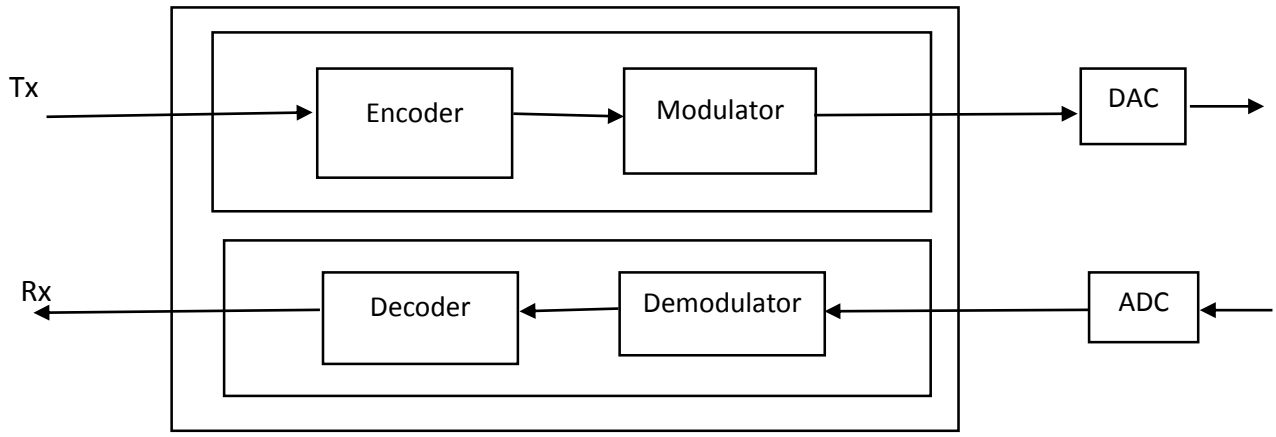


Figure 5, High level modem system

© GScom, Corp.

3.2 Encoder Block Description

The encoder block is consists of the following function blocks:

- Low Density Parity Check (LDPC) Encoder
- Mapper

3.2.1 LDPC Encoder Description

Low-Density-Parity-Check (LDPC)is the one of the most powerful linear block codes in terms of performance (coding gain) and its decoding algorithm is inherently parallel which is attractive for high-speed applications.

The Turbo-Decoding-Message-Passing (TDMP) algorithm is one of the LDPC decoding algorithms. The TDMP processes the rows of the parity check matrix (H) sequentially which results in a lower processing throughput compared to the LDPC-like schedule. The TDMP algorithm can be parallelized by embedding some structure into the parity check matrix (H) that allows the parallel processing of multiple rows without communicating messages between the nodes corresponding to these rows. The resulting matrix (H) is composed of several bands of non-overlapping rows, referred to as a Parallel Turbo-Decoding-Message-Passing (P-TDMP) algorithm. The P-TDMP algorithm was chosen for decoding the Architecture Aware Low Density Parity Check (AA-LDPC) codes due to its faster convergence speed and corresponding throughput advantage over the standard LDPC and the standard TDMP codes. The modem employs a reduced complexity message computation mechanism (Soft-Input-Soft-Output (SISO) decoder), which is free of lookup tables and features a programmable network for message interleaving based on the code structure.

To design finite length LDPC codes that can be efficiently implemented on the target multi-processor architecture for high speed communication, we utilize a design process that combines the optimal degree distribution for asymptotic performance, characteristics of structured sub-matrices, finite length code optimization criteria, architectural constraints such as the number of processors, and the width of the SIMD unit. The design process is performed in two levels- the block matrix level which constructs the H_b matrix, and the sub-matrix level which assigns the shift values to the sub-matrices in the H_b block matrix.

We use AA-LDPC code with the parity check matrix which has a D x B array of S x S sub-matrices. The (n,k) LDPC code has k information bits, n coded bits, and (n-k) parity bits with a code rate of r=k/n. The parity check matrix H is of dimension (n-k) x n, which defines a set of equations as follows:

$$\mathrm{H} \,.\, v^T = 0 \qquad (1)$$

Denote H = $[H_1\ H_2]$, where H_1 and H_2 have dimensions (n-k) x k and (n-k)x(n-k), respectively.

Note that H_1 and H_2 can also be represented as an m x b array of S x S sub matrices and m x m of S x S sub-matrices, respectively, where m = (n-k)/S and b = k/S. Denote the code word $\mathrm{v} = [\mathrm{s\ p}]$, where s is the k information bits and p is the n-k parity bits. Using the decomposed check matrix H and the decomposed code word v in the above check matrix equation, we have

© GScom, Corp.

$$H_1.s^T + H_2.p^T = 0 \pmod 2 \qquad (2)$$

$$p^T = H_2^{-1}H_1.s^T \pmod 2 \qquad (3)$$

The Quasi Cyclic Architecture Aware (QC AA) LDPC code provides efficient encoding and decoding with excellent performance. For the parity check matrix in the QC AA-LDPC code, H_2 has a simple deterministic structure where the encoding can be performed recursively using the structure of the QC AA-LDPC code as shown in below:

The deterministic structure of H_2 enables the encoding procedure to be recursively done. The first column of the sub-matrix in h = [$h_0, h_1, \dots, h_{m-1}$]^T satisfies $\sum_{i=0}^{c-1} h_i = I_{sxs}$ (mod2). The other column in H_2 has two identity sub-matrices.

As discussed before, H_1 consists of an m x b array of S x S sub-matrices, which are either zero or shifted identity matrices. Given a block of information bits s, if we decompose the s information bits into a 1 x b array of 1 x S sub-matrices, and also decompose the p parity bits into a 1 x m array of 1 x S sub-matrices, we have the following recursive parity check vector equation using the decomposed (H, v) in equation (2) above.

$p_0^T = \sum_{i=0}^{m-1} H_1^{(row\ i)}.s^T$ (mod2)

$p_1^T = H_1^{(row\ 0)}.s^T + h_0.p_0^T$ (mod2)

$p_2^T = H_1^{(row\ 1)}.s^T + h_1.p_0^T + p_1^T$ (mod2)

……

$p_{m-1}^T = H_1^{(row\ m-2)}.s^T + h_{m-2}.p_0^T + p_{m-2}^T$ (mod2)

Please refer to the white paper attached on the home page of the GSCom website for a more detailed description of the LDPC design.

© GScom, Corp.

3.2.2 Mapper Description

The Mapper module takes the stream of coded bits, divides them into groups of *m* bits, maps each group of *m* bits into one of M (=2^m) signal constellations, and generates the stream of corresponding symbols. The Mapper is programmable to support different constellations and different modulations up to 1024QAM. The Mapper supports five standard Gray mapped constellations- QPSK, 16QAM, 64QAM, 256QAM and 1024QAM.

The Gray mapping for mapping the coded binary bits into symbols can be done using the following recursive formula:

gray (k) = $2^{n-1} + \text{gray}\ (2^n - 1 - k)$, $2^{n-1} \leq \text{k} < 2^n$

with gray (0) = 0, gray (1) = 1, as initial values.

The Mapper module takes the stream of coded bits, divides them into groups of *m* bits, and maps each group of *m* bits into one of the M (=2^m) signal constellations using the Gray code mapping rule as follows:

$$s_{k,m-1}s_{k,m-2}s_{k,m-3}\ldots\ldots s_{k,0} \xrightarrow{\text{map}} I_k, Q_k$$

In the above equation, $s_{k,i}$ (i = 0,1,2, …., m-1) represents the i^{th} bit in the k^{th} symbol to be coded by the Gray code mapping, and I_k and Q_k represents the in-phase and quadrature component of the k^{th} symbol, respectively.

Please refer to the white paper attached on the home page of the GSCom website for a more detailed description of the mapper design.

© GScom, Corp.

3.3 Modulator Functional Block Description

The modulator in the transmitter receives a stream of symbol data from the Mapper. The symbols are then modulated into real signals (I) and imaginary signals (Q). The digital format of the airframe is converted to analog signals by a digital-to-analog converter (DAC) and is then sent for transmission.

Figure 6 shows the functional block diagram of the modulator with adjacent functional blocks (which are in green boxes).

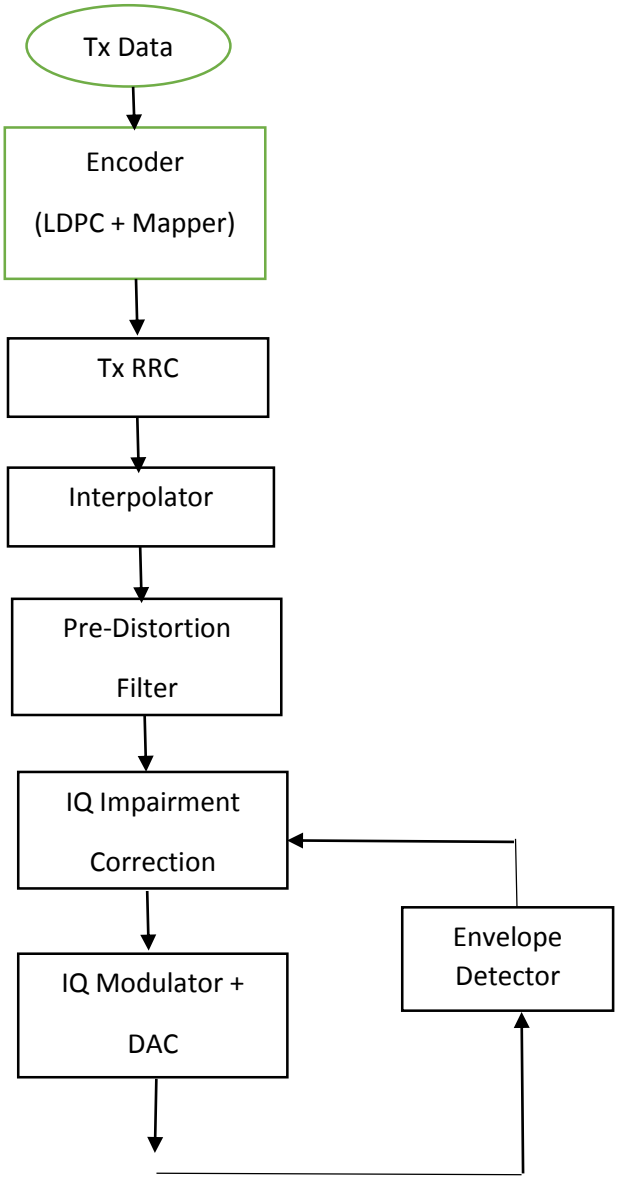


Figure 6: Modulator Functional Block Diagram

© GScom, Corp.

3.3.1. Tx RRC

The Transmitter's squared Root Raised Cosine (Tx RRC) filter is an interpolating 64-tap symmetric Finite Impulse Response (FIR) filter. It is used to reduce the inter-channel interference by shaping the modulated signal spectrum. Inputs to the filter are symbols provided at the baud rate. Outputs are sampled at twice the symbol rate.

3.3.2. Interpolator

The interpolator is a 12-tap Poly-Phase interpolator filter which provides the interpolation for a different time offset, from zero to half of a symbol interval (with 1/2048 symbol accuracy). The interpolator module adjusts the rate between the sampling clock of the DAC (digital-to-analog converter) and twice the symbol clock.

3.3.3. Pre-Distortion Filter

The pre-distortion filter is designed to compensate for the power amplifier's non-linearity. Successful pre-distortion reduces the required amplifier back-off, and thus increases the transmit power and amplifier efficiency. The module implements a 5^{th} order complex polynomial to compensate for 3^{rd} and 5^{th} order inter-modulation. The user has to update the coefficients of the polynomial according to the transmission power.

3.3.4. Tx IQ Impairment Correction

The IQ correction module corrects the IQ imbalance of the transmitter IQ modulator. The algorithm used in this module uses the feedback returning from the envelope detector. The algorithm is an adaptive algorithm which estimates the IQ gain, phase imbalances and DC offset, and then adaptively corrects them.

3.4 Demodulator Functional Block Description

The demodulator in the receiver receives a stream of symbol data from the analog-to-digital converter (ADC). The demodulator performs both frequency and timing (symbol and frame) synchronization processes while also performing the automatic gain control (AGC) process on the received symbols. After obtaining the synchronization, the demodulator performs an Equalization process to mitigate the channel effect on the received symbols. Then the output of the Equalizer is sent to the De-mapper (or Slicer) and then sent to the LDPC decoder for data correction.

© GScom, Corp.

The Demodulator is consists of the following sub-blocks:

- Rx Impairment Correction
- Rough Frequency Synchronization
- Symbol Synchronization
- Frame Synchronization
- Equalization
- Carrier Synchronization

Figure 7 shows the functional block diagram of the demodulator with adjacent functional blocks (which are in green boxes).

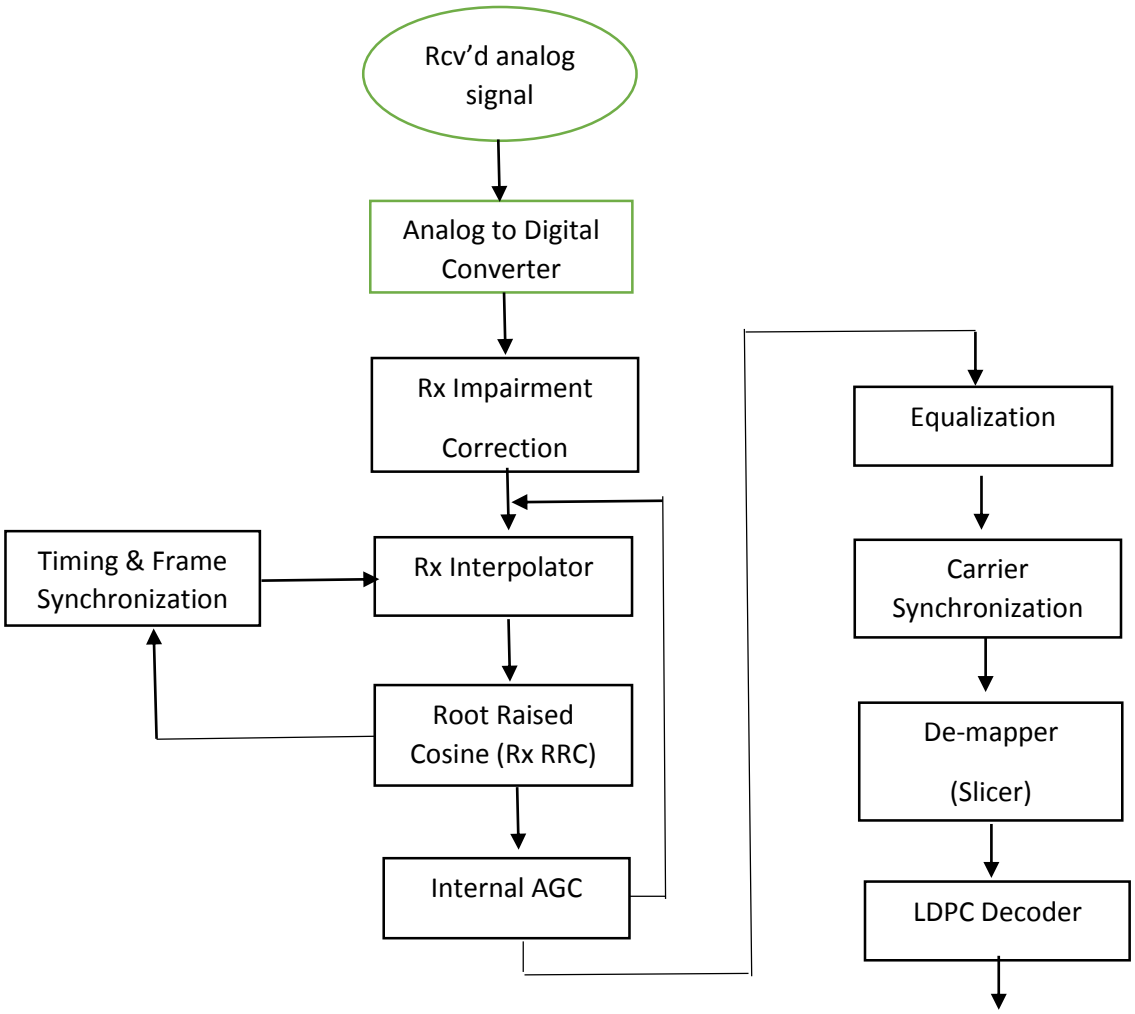


Figure 7: Demodulator Functional Block Diagram

© GScom, Corp.

3.4.1. Rx Impairment Correction

The Rx Impairment correction module corrects the gain, phase and DC offset IQ imbalances, caused by the receiver IQ demodulation. The algorithm is an adaptive algorithm which estimates the IQ gain imbalance, phase imbalance and DC offset, and adaptively corrects them.

3.4.2. Rx Interpolator

There are two functionalities to be performed in the Rx Interpolator. The first is the sampling rate reduction and the second is the timing synchronization between the ADC sampling time and the symbol timing. The Rx Interpolator reduces the data rate from the ADC sampling rate to twice the symbol rate. The Rx Interpolator is a 12-tap Poly-Phase interpolator filter which provides the interpolation for a different time offset, from zero to half of a symbol interval (with 1/2048 symbol accuracy). The interpolation filter is controlled by the timing loop from the Symbol Synchronization module.

3.4.2. Rx RRC

The Receiver squared Root Raised Cosine (Rx RRC) filter is an interpolating 64-tap symmetric Finite Impulse Response (FIR) filter. It is used to reduce the inter-channel interference by shaping the modulated signal spectrum. Inputs and outputs of the filter are sampled at twice the symbol rate.

3.4.3. Timing and Frame Synchronization

There are two synchronization function modules used at the output of the Rx RRC- The first is the Symbol Timing Synchronization module and the second is the Frame Timing Synchronization module. Note that we need to achieve the symbol timing synchronization first and then the frame timing synchronization for correct system operation.

3.4.3.1. Symbol Timing Synchronization

Symbol timing synchronization is performed to synchronize the symbol timing of the receiver with the transmitted symbol timing. The synchronizer uses the eye pattern of the received data, where there exists one half of a symbol period between the eye opening and the eye transition timing of the data.

© GScom, Corp.

3.4.3.2. Frame Synchronization

Frame Synchronization is performed to synchronize the frame timing of the receiver with the transmitted frame timing. This function block identifies the start of the airframe. The frame synchronization is performed after the symbol synchronization.

The frame synchronization module uses an Acquisition Frame whose frame format is known between the transmitter and receiver. The Acquisition Frame is 4096 symbol size frame and consists of eight sub-frames as can be seen in Figure 8. Each sub-frame consists of 256 preamble symbols and 256 QPSK symbols. The 256 preamble symbols in the sub-frame repeat a block of 128 preamble symbols twice. The 128 preamble symbols are obtained by generating a 256 Gold Sequence (GS) and then mapping the 256 GS into 128 QPSK constellation signals. The generator for the 256 Gold Sequence block consists of two 8-bit shift registers and several XOR gates between those two registers.

Sub-frames 1 through 7 contain identical symbols. The symbols in Sub-frame 8 however differ from the symbols of the preceding 7 Sub-frames. The 256 preamble symbols in Sub-frame 8 have their sign-bits inverted compared to the 256 preamble symbols for the Sub-frames 1 through 7 of the Acquisition Frame. The inverted sign is used to indicate that the sub-frame is the last sub-frame in the Acquisition Frame.

The Frame Synchronization module uses a preamble correlator to search for the block of 256 preamble symbols and locks onto it. Once the last sub-frame is detected, a state machine changes the frame timing counter to synchronize the receiver frame timing with the transmitted frame timing of the transmitter.

Figure 8 shows the Acquisition Frame configuration

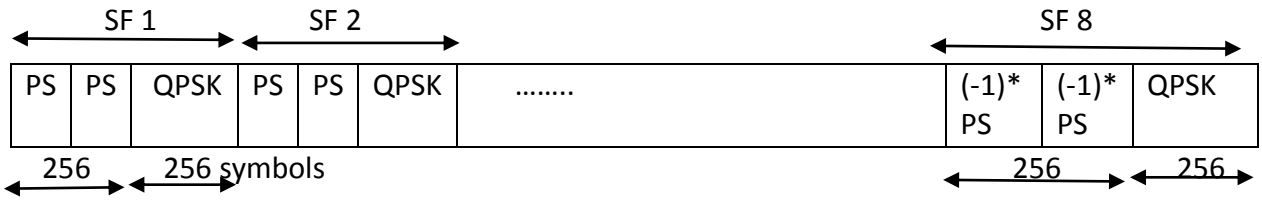


Figure 8: Acquisition Frame Format

© GScom, Corp.

3.4.5. Internal Automatic Gain Control (AGC)

The AGC is used to accommodate a wide dynamic range on the input signal and to provide a constant signal level for the communication system to ensure that the communication system operates reliably and effectively. Usually, the receiver is required to accommodate a dynamic range of over 100 dB for the signal in backhaul RF systems.

In general, the AGC needs to keep the output of the variable gain amplifier (VGA) constant by keeping the sum of the input power, low noise amplifier (LNA) gain, and the VGA input. In this approach, we need to ensure that the numerical control signal matches the physical signal level consistently.

In backhaul or point-to-point (PTP) systems, a fast AGC should not impair the function of the receiver's algorithms. Unless the AGC is capable of perfectly tracking the fast fading or 100 dB/sec fast block rate of the signal, we may encounter headroom problems in the AGC and the signal could be clipped by the A/D converter. The fast AGC will however impair the soft information collected for decoding. In this case, we may need to use weighting with the received signal strength information (RSSI) to update the AGC on a frame-by-frame basis.

The AGC power control loop consists of the antenna, the LNA, the VGA, and the AGC loop. In general, the output signal from the AGC is used to control the VGA input signal from the LNA to keep the VGA output signal constant. Figure 9 models the general AGC loop diagram representing the LNA and VGA as summation nodes.

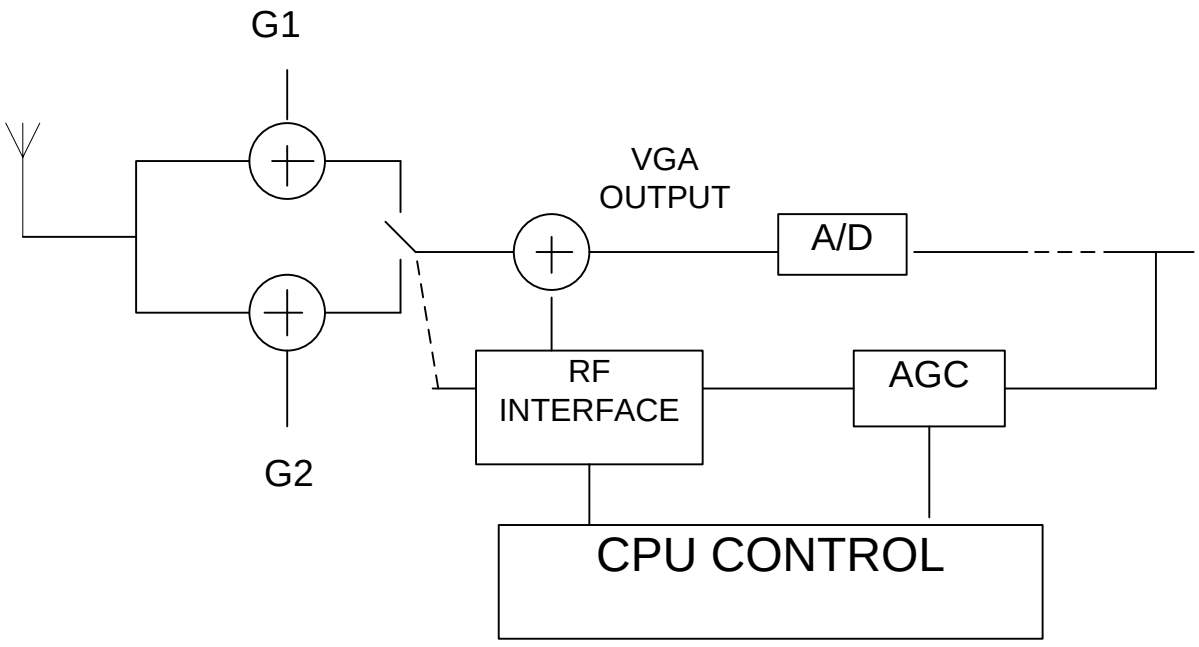


Figure 9. AGC Loop System Diagram

From the perspective of power level, we can summarize the signal flow as follows:

© GScom, Corp.

P_{IN} + G_{LNA} + G_{VGA} = VGA OUT = Constant.

Where, P_{IN} = Input signal power from antenna, G_{LNA} = LNA gain, G_{VGA} = VGA gain.

In other words, as the power of the input signal changes, the gain of the VGA should change accordingly. If the power of the input signal to the VGA is increasing, the gain of the VGA should be decreased to keep its output constant. We use the AGC loop to control the gain of the VGA, G_{VGA}.

The AGC control loop consists of the signal energy estimation block, the energy comparison block with the AGC reference (or target), and the RF interface block. The signal energy estimation block estimates the input signal energy, where the energy estimate is then used to compare with the AGC reference signal and the resulting difference is used to estimate the input signal power of P_{IN} (which is the received signal strength indicator (RSSI) or AGC loop output power estimate). The RF interface block transforms the output of the power estimation block into the VGA control input signal, VGA_{IN}. The VGA_{IN} is used to control the VGA gain, G_{VGA}, and to keep the output of the VGA constant.

Please refer to the white paper attached on the home page of the GSCom website for a more detailed description of the AGC parameter design.

3.4.6. Equalization

Equalization is used to mitigate the channel effect on the received signal for clear communication. The Equalizer estimates the communication channel distortion (such as amplitude distortion, phase distortion, fading, and channel interference, etc.) and mitigates the channel effect.

In micro-wave or millimeter wave communication systems which use high carrier frequency, the equalizer combined with the carrier recovery is one of the most powerful solutions to fight against phase noise through the latency reduction between the phase noise estimation and correction.

We use a fractional spaced decision feedback equalizer (DFE) to mitigate the channel effect and inter-symbol interference. The DFE consists of two sections in the equalizer. The first is the Feed Forward Filter (FFF) and the second is the Feedback Filter (FBF). The FFF consists of either a 24-tap or 32-tap T/2 spaced finite impulse response (FIR) filter. The FBF consists of either a 4-tap or 5-tap T spaced FIR filter. In general, the number of taps used is determined by the modulation order (or bit rate) that is used.

The phase correction is applied at the input of the FFF and at the input of decision device (DD) in the DFE using the phase error estimate which is obtained from a digital phase locked loop (DPLL). The SNR estimate is obtained by applying a signal and noise energy measurement algorithm at the output of the DD in the DFE.

For the operation of the equalizer during the acquisition mode, the preamble symbols located in the acquisition frame are used as a training sequence for the equalizer. This precedes the data mode operation (Pease refer to acquisition frame format in section 3.4.3.2).

© GScom, Corp.

For the operation of the equalizer during the data mode, the preamble symbols and pilot symbols located in the data frame are used for channel estimation and mitigation (Pease refer to the data frame format in figure 4).

For updating the filter coefficients in both the FFF and FBF a least mean square (LMS) algorithm is used. The step size (μ) used for the filter coefficient update in the LMS algorithm is programmable.

Please refer to the white paper attached on the home page of the GSCom website for a more detailed description of the equalizer system design.

3.4.7. Carrier Synchronization

The carrier synchronization loop tracks the phase error at the input of the decision device (DD) in the decision feedback equalizer (DFE). The carrier synchronization loop is a 2nd order type II digital phase locked loop (DPLL) operating at the symbol rate. The phase error estimate is obtained from the phase difference between the input and output of the decision device and is decision directed. Figure 10 is the functional block diagram of the carrier recovery loop (CRL) combined with the part of the DFE that is designed for joint detection, estimation, and compensation of the phase noise. The CRL tracks the phase error at the input of the DD in the DFE.

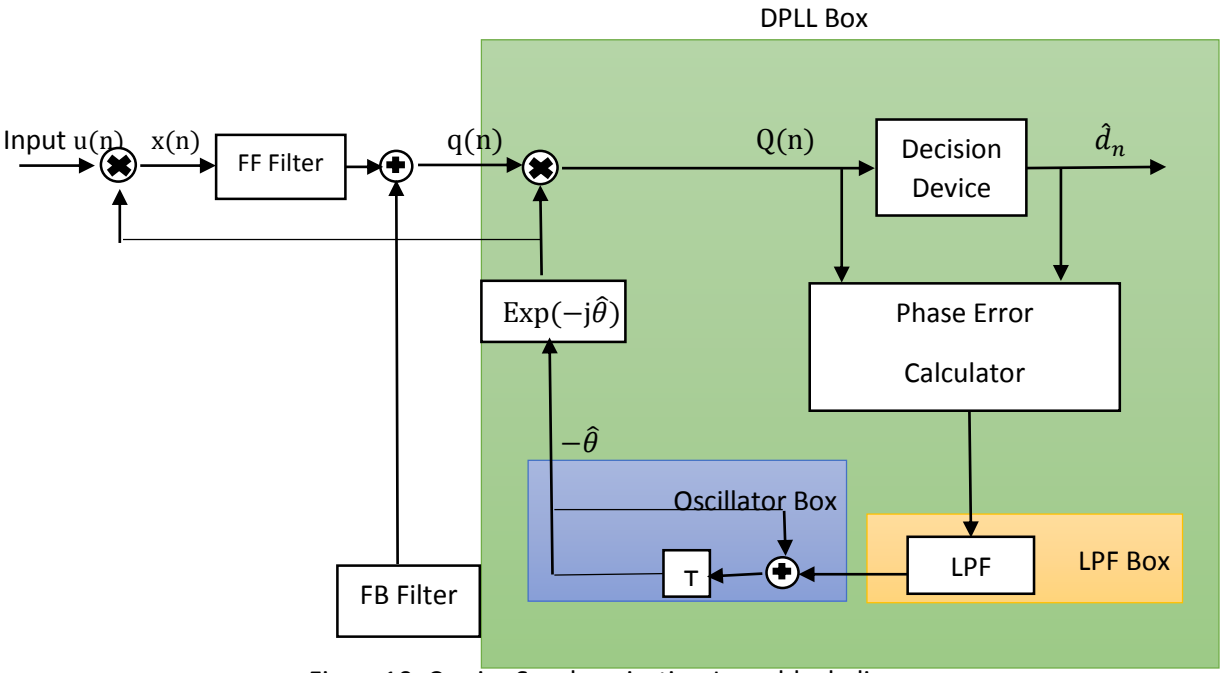


Figure10. Carrier Synchronization Loop block diagram

© GScom, Corp.

The low-pass-filter (LPF) in the Digital Phase Locked Loop (DPLL) in figure10 has two signal paths. One is the proportional path and the other is the integration path. The proportional path gain K_p and the integration path gain K_i in the LPF determine the bandwidth of the carrier synchronization loop filter.

The carrier synchronization loop supports a range of loop bandwidths designed to maximize the phase noise tracking capability and minimize the effect of additive white Gaussian noise (AWGN). The loop bandwidth requires optimization based on the expected signal to noise ratio (SNR) and the expected amount of the phase noise.

Please refer to the white paper attached on the home page of the GSCom website for a more detailed description of the carrier synchronization system design.

3.5 Decoder Block Description

The Rx decoder receives demodulated symbols from the demodulator, converts the received symbols into a bit stream by the de-mapping process, and then performs the error correction on the received bit streams of data. The Rx Decoder consists of a De-Mapper (or Slicer) and the LDPC Decoder.

3.5.1. De-Mapper Description

The Receiver receives the transmitted signal through an analog-to-digital converter (ADC) and demodulates the received symbols. The De-Mapper receives the demodulated symbols from the demodulator and converts the received symbols into the soft decision values of the corresponding bit stream using a soft decision decoding process. The De-Mappers output signals are the soft decision value of the bit stream of the corresponding symbols. As it is well known, an LDPC provides its best performance with the soft decision value as its input.

We can represent the demodulated receive signal, R_k,, as follows:

$$R_k = X_k + jY_k = g_k(I_k + jQ_k) + (n_k^I + jn_k^Q)$$

Where, X_k and Y_k are the in-phase and quadrature components of the modulated signal, g_k is the transmission path gain of the signal, and n_k^I and n_k^Q are independently distributed zero mean white Gaussian noise.

The estimation of bit, $S_{k,i}$ (i = 0, 1, 2 … m), in symbol S_k is obtained by taking the log likelihood ratio (LLR) of the conditional probability of the bit being correct, as follows:

$$\Lambda(S_{k,i}) = \log \frac{\Pr\{S_{k,i}=0 \mid X_k, Y_k\}}{\Pr\{S_{k,i}=1 \mid X_k, Y_k\}}, \quad i = 0, 1, 2, \ldots, m-1$$

The above LLR equation can be approximated by a dual minimum metric approximation as follows:

© GScom, Corp.

[Equation 4]

$$\Lambda(S_{k,i}) = \log \frac{\sum_{z_i} \exp\{ -1/\sigma_n^2 [R_k - z_k (S_{k,i}=0)]^2 \}}{\sum_{z_i} \exp\{ -1/\sigma_n^2 [R_k - z_k (S_{k,i}=1)]^2 \}} \quad .$$

$$\approx \{ \min [R_k - z_k(S_{k,i} = 1)]^2 - \min [R_k - z_k(S_{k,i} = 0)]^2 \} \quad .$$

$$= (2n_{k,i} - 1) \{ [R_k - z_k(S_{k,i} = \mathrm{n}_{k,i})]^2 - \min [R_k - z_k(S_{k,i} = \bar{n}_{k,i})]^2 \}$$

Where, $z_k(S_{k,i}$=0) and $z_k(S_{k,i}$=1) are the values of I_k+jQ_k when $S_{k,i}$=0 and $S_{k,i}$=1, respectively. In the above equation $\mathrm{n}_{k,i}$ represents the value of the i^{th} bit of the symbol, which is the closest to the received symbol, R_k. The $\bar{n}_{k,i}$ represents the negative value of $\mathrm{n}_{k,i}$.

As can be seen from the above equation, the LLR can be calculated by finding the values of $z_k(S_{k,i} = 1)$ and $z_k(S_{k,i} = 0)$ which minimize the values of $[R_k - z_k(S_{k,i} = 1)]^2$ and $[R_k - z_k(S_{k,i} = 0)]^2$, respectively. The values of $z_k(S_{k,i} = 1)$ and $z_k(S_{k,i} = 0)$ which minimize the values of $[R_k - z_k(S_{k,i} = 1)]^2$ and $[R_k - z_k(S_{k,i} = 0)]^2$ can be determined by the range of the values of the in-phase and quadrature component of the received symbol, R_k.

The first term in the third line of the [Equation 4] can be represented using the in-phase and quadrature component of received symbol (or signal) as follows:

[Equation 5]

$$[R_k - z_k(S_{k,i} = \mathrm{n}_{k,i})]^2 = (X_k - U_k)^2 + (Y_k - V_k)^2$$

Where, U_k and V_k are the in-phase and quadrature components of the received symbol on a signal constellation.

The second term in the third line of the [Equation 4] can be represented using the in-phase and quadrature component of received symbol (or signal) as follows:

[Equation 6]

$$\min [R_k - z_k(S_{k,i} = \bar{n}_{k,i})]^2 = (X_k - U_{k,i})^2 + (Y_k - V_{k,i})^2$$

Where, $U_{k,i}$ and $V_{k,i}$ are the in-phase and quadrature components of the received symbol on the signal constellation which minimizes $[R_k - z_k(S_{k,i} = \bar{n}_{k,i})]^2$.

Inserting [Equation 5] and [Equation 6] into [Equation 4], the LLR can be obtained as follows:

[Equation 7]

$$\Lambda(\mathrm{S_{k,i}}) = (2\mathrm{n_{k,i}}\text{-}1) [\{ (\mathrm{X_k}\text{-}\mathrm{U_k})^2 + (\mathrm{Y_k}\text{-}\mathrm{V_k})^2 \} - \{ (\mathrm{X_k}\text{-}\mathrm{U_{k,i}})^2 + (\mathrm{Y_k}\text{-}\mathrm{V_{k,i}})^2 \}] \quad .$$

$$= (2\mathrm{n_{k,i}}\text{-}1) [(\mathrm{U_k}+\mathrm{U_{k,i}} - 2\mathrm{X_k}) (\mathrm{U_k}\text{-} \mathrm{U_{k,i}}) + (\mathrm{V_k}+\mathrm{V_{k,i}} - 2\mathrm{Y_k}) (\mathrm{V_k}\text{-} \mathrm{V_{k,i}})]$$

We can use [Equation 7] to derive the soft decision values of the LLR of any M-ary QAM for the input into the decoder. As we support from 4QAM up to 1024QAM in our modem, we need to derive 5 sets of soft decision values for those LLRs derived for the 4QAM, 16QAM, 64QAM, 256QAM and 1024QAM modulation rates.

© GScom, Corp.

As can be seen in [Equation 7], we need to find the values of U_k, $U_{k,i}$, V_k, and $V_{k,i}$ for the corresponding received symbol $S_{k,i}$ (= X_k + jY_k) as a function of the bit position of i (= 0, 1, ….., *m-1*) to calculate the LLR of the received symbol $\Lambda(S_{k,i})$.

Please refer to the white paper on the De-Mapper attached on the home page of the GSCom website for a more detailed description of the derivation of the soft decision values of the LLR for the various QAM levels.

3.5.2. LDPC Decoder Description

The classic LDPC decoding algorithm is a two-phase message passing (TPMP) algorithm or the so called belief propagation (BP) algorithm. The decoding procedure revolves around the two-phase transmission of extrinsic information between check nodes and bit nodes on the Tanner Graph. The message passing decoding algorithm utilizes the bit-to-bit dependence required in check-sum equations of the LDPC code to adjust the probability of each bit until obtaining a valid code-word.

The Turbo Decoding Message Passing (TDMP) algorithm outperforms the standard two-phase decoding algorithm with its faster convergence speed of roughly a factor of two, in terms of the number of decoding iterations required, and with its memory savings of more than 50%. With the TDMP scheme both variable and check messages collapse into a single type of message, and compared to the TPMP algorithm the TDMP algorithm requires lower complexity and is therefore more suitable for hardware implementation.

The Quasi Cyclic LDPC (QC-LDPC), used in our system, is an Architecture Aware LDPC (AA-LDPC). The ones in its each block row (or every s row) in the QC-LDPC are not overlapped. As a result, decoding can be processed for s rows simultaneously. The TDMP algorithm can be modified to a parallel version referred to as P-TDMP using the architecture of the AA-LDPC. The P-TDMP algorithm provides an improvement in decoding throughput over the ordinary TDMP algorithm and is attractive for high speed applications.

In the LDPC we used, the parity matrix H is decomposed into SxS binary sub-matrices. These sub-matrices satisfy the following two conditions- each column has at most a single 1, and each row has at most a single 1. Note that zero columns, zero rows and null matrices are allowed under this definition.

By decomposing the parity check matrix H of an LDPC code in such a way as to restrict the column positions of the ones, the LDPC decoding problem is transformed into a turbo-decoding problem where messages flow in tandem only between the adjacent super-codes as opposed to potentially all the sub-codes on the parity check matrix. The inter-leavers are factored into smaller inter-leavers that are more practical to implement.

© GScom, Corp.

3.5.2.1. LDPC Soft Decoding Algorithm

We review the LDPC decoding algorithm using a classical two phase message passing (TPMP) approach, called iterative layered belief propagation approach, using the following parity check matrix as an example:

$$\begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

In the above parity check matrix, we have four check nodes (m = 4) and eight variable nodes (j=8). Let $\mathrm{L}(q_{mj})$ denote the variable node log likelihood ratio (LLR) message sent from variable node j to the check node m, then:

$$\mathrm{L}(q_{mj}) = \mathrm{L}(q_j) - R_{mj}\ ,$$

$$R_{mj} = \prod_{j' \epsilon N(m)\backslash\{j\}} sign(\ L(q_{mj'})\ )\ \Psi\left[\sum_{j' \epsilon N(m)\backslash\{j\}} \Psi(\ \ L(q_{mj'})\ ) \right]$$

$$\mathrm{L}(q_j) = \mathrm{L}(q_{mj}) + R_{mj},$$

$$\Psi(\mathrm{x}) = -\log\ [\ \tanh(\tfrac{|x|}{2})\ ]$$

Where, R_{mj} is the check node LLR message sent from the check node m to the variable node j, and $\mathrm{L}(q_j)$ (j=1,2,…..,N) represents the a posteriori probability ratio (APP) for all variables. The APP messages are initialized with the channel reliability values of the coded bits. N(m) is the set of all variable nodes connected to the check node m.

The LDPC is an iterative decoding scheme which performs the above three parameter estimations and updates in each iteration step to improve the reliability of the bit estimation using LLR information and parity check equations. If all parity check equations are satisfied or the pre-determined maximum number of iteration is reached, then the decoding stops.

3.5.2.2. Turbo Decoding Message Passing (TDMP) Algorithm

The previous decoding algorithm operates using two types of messages and requires the saving all intermediate messages between both rounds at every iteration. Moreover, newly computed messages in a round of computations do not participate in further message computations until the decoding iteration is over. If updated messages are used directly within an iteration to compute new messages, then refined estimates will speed up the convergence behavior of the algorithm. Further, new check messages become directly variable messages within the iteration, hence, both variable and check messages collapse into a single type of message leading to a significant savings in the memory storage required.

The TDMP algorithm utilizes the most recently updated message directly within the iteration to compute a new message, which speeds up the convergence speed in terms of total iterations required. Furthermore, the two phase message computation in the TPMP algorithm is substituted by a single

© GScom, Corp.

computation, and the memory savings is significant. As a result, the TDMP algorithm outperforms the TPMP algorithm in terms of throughput (20%-50%), coding gain, and hardware efficiency.

The TDMP algorithm can be described with the help of the parity check matrix in the above figure in Section 3.5.2.1 which has four parity check equations corresponding to a code of length 8. The algorithm is based on decoding the rows (parity check equations) of the parity check matrix sequentially. Extrinsic messages generated from decoding earlier rows are used as prior messages to decode subsequent rows. To each row i in H, we associate the vector $\underline{\lambda}^i = [\lambda_1^i, \ldots\ldots, \lambda_{c_i}^i]$ of extrinsic messages corresponding to the nonzero entries in that row. The number of nonzero c_i in a row is called the weight. Let I_i denote the set of indexes of ones in row I and $\underline{\gamma}(I_i)$ represent the posterior messages of row i. For example, in the parity check matrix H above in the figure shown in Section 3.5.2.1, the weight of row 2 is c_2=4, while its index set I_2={1,4,6,7}, and its extrinsic message vector $\underline{\lambda}^2$=[$\lambda_1^2, \lambda_2^2, \lambda_3^2, \lambda_4^2$]. Where λ_1^2 corresponds to bit 1 and λ_2^2 corresponds to bit 4, etc. The extrinsic messages $\underline{\lambda}^i$ are associated to the extrinsic estimates about the bits from all parity check equations that these bits participate. Let $\underline{\gamma}$= $[\gamma_1, \ldots\ldots, \gamma_N]$ denote posteriori messages that stores the sum of all messages generated by the rows in which each bit participates. For example, N=8, $\gamma_1=\lambda_1^2 + \lambda_1^3$, $\gamma_2=\lambda_1^1 + \lambda_2^3$,…,$\gamma_8=\lambda_4^1 + \lambda_4^3$. The posterior messages of row i are indexed as $\underline{\gamma}(I_i)$ and the hard decisions are determined by slicing the $\underline{\gamma}$ vector.

Decoding the *i* th parity check row involves the following four steps which constitute a decoding sub-iteration:

1) Read: $\underline{\lambda}^i$ and $\underline{\gamma}(I_i)$ are read for row i from memory.
2) Subtract: $\underline{\lambda}^i$ are subtracted from $\underline{\gamma}(I_i)$ to generate prior messages $\underline{\rho}$ = $[\rho_1, \ldots\ldots, \rho_{c_i}]$.
3) Decode: Decode row i using a SISO algorithm with $\underline{\rho}$ as input and $\underline{\Lambda}^i$ = $[\Lambda_1^i, \ldots\ldots, \Lambda_{c_i}^i]$ as output.
4) Write back: Replace the original extrinsic message $\underline{\lambda}^i$ with $\underline{\Lambda}^i$ and update $\underline{\gamma}(I_i)$ by $\underline{\gamma}(I_i)$ = $\underline{\rho}$ + $\underline{\lambda}^i$.

A decoding iteration comprises multiple sub-iterations corresponding to the rows of H. A round of sub-iterations over all rows of H constitutes a decoding iteration. The whole decoding procedure will be terminated when it satisfies all the parity check equations in the matrix or the number of iterations has reached a predefined number. The TDMP algorithm requires storage memory for

$$M_{TDMP} = \sum_{i=1}^{M} c_i \; + \; N$$

messages, assuming H has M rows, N columns, and row weights c_i.

A formal description of the TDMP algorithm is given using the SISO algorithm which is a reduced-complexity message computation algorithm. The SISO algorithm will be explained in detail in the next section. The inputs to the TDMP algorithm are a sparse parity check matrix $H_{M\,X\,N}$ representing a repeated accumulated (RA) code, LDPC code (where N=n+k), input channel observations $\underline{\delta}$ = $[\delta_{u_1}, \ldots\ldots, \delta_{u_k}; \delta_{y_1}, \ldots\ldots, \delta_{y_n}]$, and maximum number of decoding iterations T. The outputs are hard decision estimates of the information bits $\underline{u} = [u_1, u_2, \ldots\ldots, u_k]$.

© GScom, Corp.

3.5.2.2.1. SISO algorithm for message decoding in TDMP algorithm

The SISO (soft input soft output) algorithm is adopted for message decoding in the TDMP algorithm as can be seen in the decoding step in its sub-iteration procedure. Unlike other schemes of

$$\Lambda_j^i = \Psi^{-1}\left(\sum_{l\epsilon l\backslash j}\Psi(\rho_l)\right), \qquad i = 1,\ldots.,M;\ \ j = 1,\ldots.,c_i$$

the SISO decoder need not use a lookup table, and can be implemented using simple logic gates. Moreover, the SISO circuit can be implemented using the "max-quartet" function $\mathrm{Q}(\mathrm{x},\mathrm{y})$ as can be seen in Figure 11 resulting in memory savings, an improvement in efficiency for the hardware implementation, and it provides a more accurate approximation than any other available at this time.

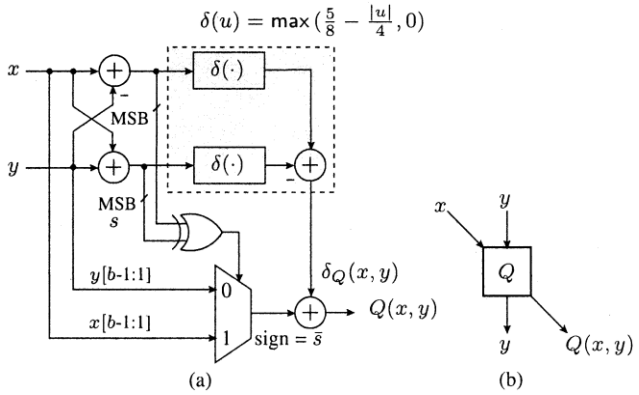


Figure 11. Max-quartet function $\mathrm{Q}(\mathrm{x},\mathrm{y})$: (a) Logic circuit, and (b) Symbol

3.5.2.3. Parallel Turbo Decoding Message Passing (P-TDMP) Algorithm

The TDMP algorithm processes the rows of H sequentially which results in a lower processing throughput compared to the LDPC-like schedule. The TDMP algorithm can be parallelized by embedding some structure into the parity check matrix, H, that allows parallel processing of multiple rows without communicating messages between the nodes corresponding to these rows. The resulting H is composed of several bands of non-overlapping rows. We refer to the LDPC code with such a structure as an architecture-aware LDPC (AA-LDPC) code, and the parallel TDMP as a P-TDMP algorithm.

The P-TDMP algorithm is based on having the ones in every s number of rows of sub-matrices as non-overlapped, so the decoding procedure can be handled in parallel using s SISO decoders. The s SISO decoders constitute a decoder group working simultaneously. Decoder i processes row i in each sub-matrix and maintains the extrinsic messages $\lambda^{i,j}$ in local memory, while posteriori messages are passed to the decoders s messages at a time from a global memory. The decoding procedure runs over the rows of H in c iterations and processes s rows simultaneously for every iteration which improves the de-coding throughput by a factor of s as compared to the TDMP algorithm. The AA-TDMP reduces the required number of multiplexers/de-multiplexers by an order of n (O(n)), and the required number of control overhead by an order of O(log n).

© GScom, Corp.

The parallel Turbo-decoding message-passing (P-TDMP) algorithm is chosen for the AA-LDPC codes providing a faster convergence speed and hence a throughput advantage over the standard TDMP codes. We employ a reduced complexity message computation mechanism (SISO algorithm) and the features of a programmable network for message interleaving, based on the code structure.

To design finite length LDPC codes that can be efficiently implemented on the target multi-processor architecture for high speed communication, we utilize a design process that combines the optimal degree distribution for asymptotic performance, characteristics of structured sub-matrices, finite length code optimization criteria, architectural constraints such as the number of processors, and the width of the SIMD unit. The design process is performed at two levels. First is the block matrix level design which constructs the H_b matrix, and second is the sub-matrix level design which assigns shift values to the sub-matrices.

The S rows of ones in each row of sub-matrices in an AA_LDPC H are non-overlapping, and hence can be processed in parallel using S SISO decoders. Decoder s processes row s in each row of sub-matrices, for a total of D rows, and maintains the extrinsic messages denoted by $\underline{\lambda}^{d,s}$, $\mathrm{d} = 1,2,\ldots.,\mathrm{D}$, in a local memory. Posterior messages are stored in a global memory of size N and passed in parallel (S messages at a time) to the decoders using a network that implements the factored edge permute process. The innermost parallel loop runs over the rows of H in D iterations processing S rows during each iteration period. This results in a factor of S improvement in decoding throughput over the TDMP algorithm.

© GScom, Corp.

3.5.2.3.1. Programmable P-TDMP Decoder Architecture

Figure 12 shows one architecture option for a high level programmable decoder implementing the P-TDMP algorithm. We assume that the corresponding parity check matrix is for an AA-LDPC code and has the following parameters:

1) S is the size of the sub-matrix partitions (assuming permutation matrices).

2) B = n/S denotes the number of sub-matrices per block row.

3) D = m/S denotes the number of sub-matrices per block column.

4) r is the maximum number of permutation matrices per block row.

5) $c_1, \ldots\ldots, c_B$, corresponds to the number of permutation matrices in the block columns.

The architecture includes B memory modules for storing the messages, S MPUs (message processing units) that operate in parallel, and read/write networks for transporting the messages between the memory and the MPUs. The parameter S acts as a scaling parameter.

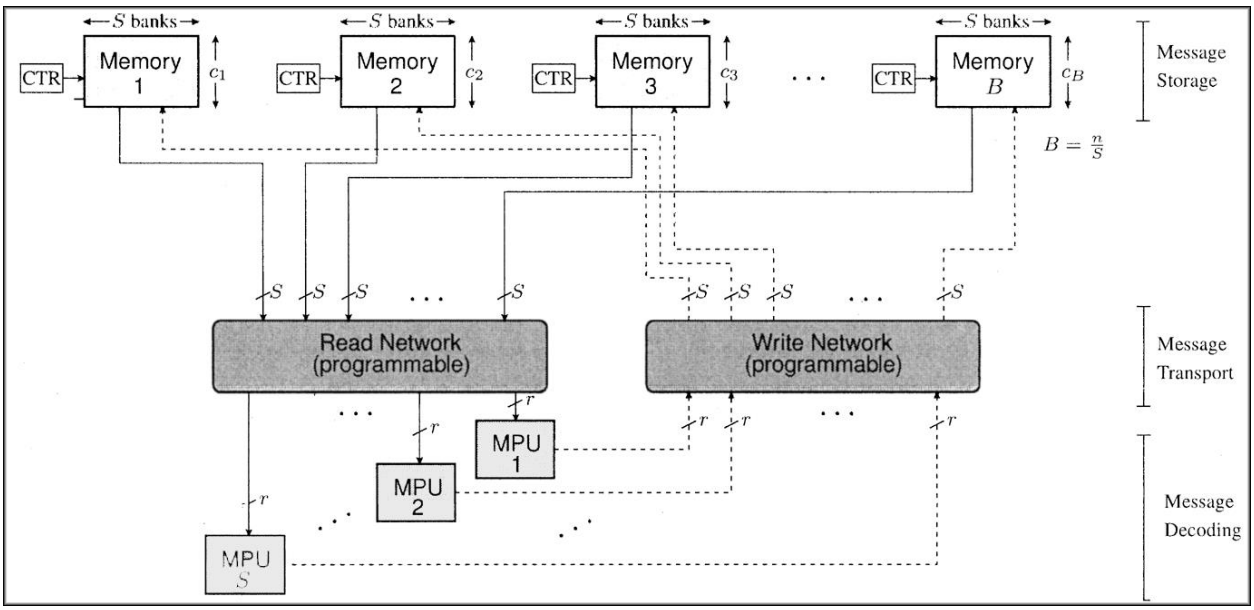


Figure 12. Decoder architecture implementing the P-TDMP algorithm

The decoder completes a single decoding iteration by performing a round of D updates across the super-codes. An update corresponding to a single super-code C^j constitutes a sub-iteration which involves the following four steps:

1) The network performs c read operations from memory, where c is the maximum node degree of C^j. It then forwards r messages to each of the S MPUs.
2) The MPUs update the messages in parallel using the SISO algorithm.

© GScom, Corp.

3) The updated messages are routed by the write network to the appropriate memory modules scheduled to receive message related to C^j.
4) The messages are written in the designated memory modules and the address counters are updated.

Please refer to the white paper attached on the home page of the GSCom website for a more detailed description of the LDPC system design.

© GScom, Corp.

4. Performance Test Result

We carried out the system performance simulation of the broadband wireless modem on 4QAM, 16QAM, 64QAM, 256QAM and 1024QAM modulation QAM levels inserting varying SNR levels of additive white Gaussian noise, along with phase noise masking levels obtained from one of tier-1 backhaul equipment vendors.

4.1 Performance improvement due to LDPC & Phase noise mitigation

Two LDPC code-rates (0.5 and 0.75) were used for the performance testing of the broadband wireless modem while applying varying SNR noise levels. The following tables show the performance improvement of the modem due to the LDPC codec architecture chosen for our broadband wireless modem design.

- **LDPC (0.5 CR) Coding gain with reference to 1e-6 BER**

	4QAM	16QAM	64QAM	256QAM	1024QAM
AWGN	10 dB	8 dB	8 dB	8 dB	8 dB
PN+AWGN	8 dB	5 dB	1 dB	0 dB	0 dB
PN+AWGN w/DPLL	8 dB	6 dB	8 dB	6 dB	8dB

- **LDPC (0.75 CR) Coding gain with reference to 1e-6 BER**

	4QAM	16QAM	64QAM	256QAM	1024QAM
AWGN	8 dB	6 dB	5 dB	6 dB	6 dB
PN+AWGN	6 dB	5 dB	1 dB	0 dB	0 dB
PN+AWGN w/DPLL	6 dB	6 dB	5 dB	6 dB	4 dB

- CR = Code Rate
- BER = Bit Error Rate

© GScom, Corp.

- AWGN = Additive white Gaussian noise
- PN = Phase noise
- DPLL = digital phase lock-loop (phase noise mitigation technique)

4.2 Performance Test Summary

The broadband wireless modem provides very reliable functionality while being very robust to any kind of noise such as AWGN, phase noise, etc.

The following is a summary of the improvements and advantages of our broadband wireless backhaul modem verified during the performance testing of the modem:

- Excellent coding gain from the LDPC.
- Two LDPC iterations are typically enough in most of the practical operational communication decoding environments.
- Excellent performance of the phase noise mitigation from the DFE combined with the Carrier Recovery in terms of the signal quality recovery and the latency.
- One digital PLL (DPLL) was used to take care of the phase noise (PN). In practical communication systems an analog PLL, in addition to the DPLL, is used which gives a few more dB of performance improvement.
- Real time processing of the de-mapping and its performance.
- Very reliable Structured System Operation Architecture.

© GScom, Corp.