COM-1510SOFT
Block mode convolutional FEC codec
VHDL source code overview / IP core

Overview

The COM-1510SOFT is a convolutional FEC codec, including encoder and Viterbi block decoder.

This codec operates in block mode, whereby a finite length frame is encoded and decoded. The error correction configuration (K, rate, polynomials, puncturing) can be changed dynamically on a frame-by-frame basis.

The code can be configured for either high speed (1 encoded bit per clock period) or small footprint.

When configured in parallel mode, the maximum throughput is typically in the range 100 – 250 Mbits/s depending on the FPGA technology.

The entire **VHDL source code** is deliverable.

Key features and performance:

- Flexible dynamic (i.e. at runtime) user-selected configuration:
  - Constraint length K=5,6,7,9
  - Number of parity bits 2 to 5
  - G_x generator polynomial from a preset list
  - Puncturing pattern from a preset list
- Configuration prior to VHL synthesis control the maximum occupancy:
  - Hard (1-bit) or soft(4 to 8-bit) decision decoding
  - Maximum constraint length K_max
  - Maximum number of parity bits
  - Traceback length
- Support for erasures (code puncturing)
- Reduced decoder latency: K*25 bits
- Provided with IP core:
  - VHDL source code
  - Matlab .m file to generate stimulus files for VHDL simulation at various signal to noise ratios
  - VHDL testbench with PRBS11 sequence generator and bit error rate measurement.

Target Hardware

The code is written in generic standard VHDL so as to be ported to a variety of FPGAs. It was compiled and simulated using Xilinx ISE 14 and Xilinx Vivado v2014.2 tools.

MSS • 845-N Quince Orchard Boulevard • Gaithersburg, Maryland 20878-1676 • U.S.A.
Telephone: (240) 631-1111 Facsimile: (240) 631-1676 www.ComBlock.com
 Issued 7/22/2016

Configuration

Synthesis-time configuration parameters

The following constants are user-defined in the generic section of the encoder and decoder components prior to synthesis. These parameters generally define the size of the decoder embodiement.

Parameters	Configuration
Decoder-only	
Maximum frame payload size **FRAME_SIZE_MAX_LOG2**	Log2(maximum frame payload size in bits), rounded up.
Parallel vs Sequential decoding **N_PAR_LOG2**	Parallel vs sequential decoding. Tradeoff size versus speed. Log2(number of parallel ACS circuits), between 0 (fully sequential) and (**K_MAX-1**) (fully parallel)
Maximum constraint length **K_MAX**	Valid values are 5,6,7,9
Maximum number of parity bits **N_PARITY_BITS_MAX**	Valid range 2 to 5
Number of Soft-decision decoding bits **NBSD**	Valid range 4 to 8 bits
Hard/Soft-decision decoding **HARD_SOFTN_DECISION**	'1' for 1-bit width input '0' for NBSD-bit width input
Traceback depth **TB_DEPTH**	Typically 6* **K_MAX** for non-punctured, 12* **K_MAX** for punctured codes. Valid range 30 to 120
Input Bit Error Rate measurement window **BER_WINDOW_LENGTH**	Maximum value 2^{24} -1

Run-time configuration parameters

The user can set and modify the following controls at run-time through the top level component interface:

Parameters	Configuration
Encoder	
Code selection	Select one of the preset codes **CODE_SEL** dynamically on a frame-by-frame basis. The code selection is enacted at the input start of frame (SOF_IN)
Enable tail-biting	0 (disable) / 1 (enable) **CONTROL**(0)
Extend input frame with (K-1) zeros	0 (disable) / 1 (enable) **CONTROL**(1)
Decoder	
Code selection	Select one of the preset codes **CODE_SEL** dynamically on a frame-by-frame basis. The code selection is enacted at the input start of frame (SOF_IN)
Enable tail-biting	0 (disable) / 1 (enable) **TAIL_BITING**
Frame size	Decoded frame size **FRAME_OUT_SIZE** expressed as number of decoded bits.

Limitations

1. The code does not support GMR-1 3G specification [1] for repetition.
2. The frame payload maximum size is set by **FRAME_SIZE_MAX_LOG2**. For example 12 for a maximum payload size of 4095 bits.
3. When tail-biting and puncturing are both enabled, the frame size must be an integer multiple of the puncturing period (for example 3 for code 1, 5 for code 2).
4. When tail-biting is enabled, the product 3* **TB_DEPTH*** **N_PARITY_BITS_MAX** must be less than the frame payload maximum size.

Codes

A number of convolutional codes compatible with GMR-1 3G standard [1] are preset in the VHDL source code. Codes are defined by their generator polynomials Gx, constraint length K and puncturing patterns.

0 marks an erasure during puncturing.

GMR-1 3G [1]	
Code number	**Configuration**
0	K=5 Rate ½ convolutional code No puncturing
1	K=5 Rate ½ convolutional code

	Rate ¾ after puncturing
2	K=5 Rate ½ convolutional code Rate 5/8 after puncturing
10	K=5 Rate 1/3 convolutional code No puncturing
20	K=5 Rate 1/4 convolutional code No puncturing
30	K=5 Rate 1/5 convolutional code No puncturing
60	K=6 Rate 1/4 convolutional code No puncturing
64	K=7 Rate 1/2 convolutional code No puncturing
96	K=9 Rate 1/2 convolutional code No puncturing
160	K=9 Rate 1/4 convolutional code No puncturing
192	K=9 Rate 1/3 convolutional code No puncturing

Custom codes

Adding or removing codes from the list of preset codes is quite simple. Each code is defined by:

a) generator polynomials. For example $G_0(x) = 1 + x + x^2 + x^4$ is represented by the vector G0 <= "000010111";

b) constraint length K

c) number of parity bits (number of generator polynomials) N_PARITY_BITS

d) N_PUNCTURING_PHASES: number of phases (horizontal X axis) in the puncturing matrix.

These parameters are defined in the CODE_SEL_001 processes in both encoder and decoder. Note that the generator polynomial definition in the decoder is a flipped version as the input bit enters the register through the MSb, unlike the encoder.

In addition to the above parameters, one must also define:

e) the puncturing pattern for each code in the function *f_puncturing_pattern()*

f) the punctured encoding period in the function *f_punctured_encoding_period()*

See the commented source code for details.

Sequential vs Parallel decoding

It is possible to trade-off speed versus size by instantiating either a parallel decoder (all 2^{K-1} states computed in parallel) or a sequential decoder. The ratio of speed and size is roughly 2^{K-1}.

Hard/Soft-decision decoding

It is possible to trade-off implementation complexity versus decoding performance by controlling the precision of the encoded input samples. For example, it may be advisable to use hard-decision (1-bit) decoding to keep a K=9 Rate ¼ decoder to a practical size.

For small constraint length (K=5), higher-performance 4-bit soft-decision is recommended as the implementation size is fairly small.

Monitoring

Bit Error Rate Measurement

The decoder estimates the bit error rate on the encoded bit stream by comparing the actual received bit stream with an estimate of the transmitted bit stream. This estimate is generated by re-encoding the nearly error-free decoded bit stream.

The algorithm is based on the proposition that the decoded bit stream is nearly error-free. If the decoded bit stream were error-free, then the re-encoded bit stream would be the actual transmitted encoded bit stream before bit errors occur in the transmission channel.

The bit error rate is computed over a window of **BER_WINDOW_LENGTH** bits.

Encoder component interface

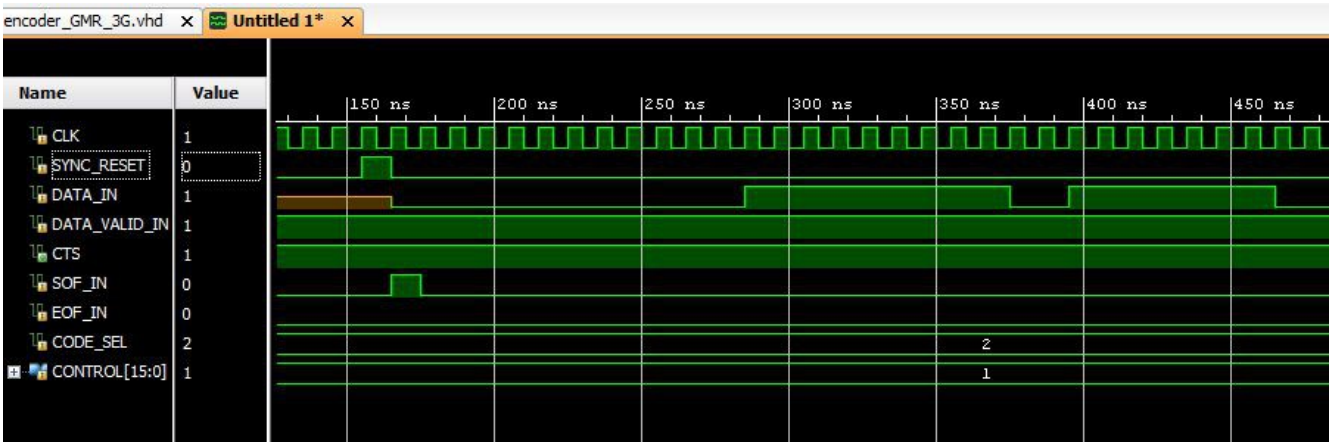
```
--GLOBAL CLOCKS, RESET
CLK : in std_logic;    -- master clock for this FPGA, synchronous
SYNC_RESET: in std_logic;    -- synchronous reset

--// Input samples
DATA_IN: in std_logic;
        -- input bit. Read at rising edge of CLK when DATA_VALID_IN = '1';
DATA_VALID_IN: in std_logic;
        -- one CLK-wide pulse
CTS: out std_logic;
        -- Clear to send. Always check CTS = '1' before sending a new input bit
        -- used for flow control.
SOF_IN: in std_logic;
EOF_IN: in std_logic;
        -- 1 CLK-wide pulses indicating start and end of frames (block mode)
        -- Aligned with DATA_VALID_IN.

--// CONFIGURATION
-- The configuration parameters below can be changed dynamically at run-time.
-- They are latched in at the start of frame SOF_IN = '1'.
CODE_SEL: in integer range 0 to 255;
        -- see GMR-1 sections 4.4 and 4.5 for details
        -- 0 = rate 1/2 convolutional code (K = 5) no puncturing 4.4.1.1, 4.4.5
        -- 1 = rate 1/2 convolutional code (K = 5) P(2;3) puncturing rate 3/4
        -- 2 = rate 1/2 convolutional code (K = 5) P(2;5) puncturing rate 5/8
        -- etc
        -- 10 = rate 1/3 convolutional code (K = 5) no puncturing
        -- 20 = rate 1/4 convolutional code (K = 5) no puncturing
        -- 30 = rate 1/5 convolutional code (K = 5) no puncturing
        -- 60 = rate 1/4 convolutional code (K = 6) no puncturing
        -- 64 = rate 1/2 convolutional code (K = 7) no puncturing
        -- 96 = rate 1/2 convolutional code (K = 9) no puncturing
        -- 160 = Rate ¼ Constraint length 9 Convolutional Encoder no puncturing
        -- 192 = Rate 1/3 Constraint length 9 Convolutional Encoder no puncturing
        -- MAXIMUM N_PARITY_BITS IS 5
CONTROL: in std_logic_vector(15 downto 0);
        -- bit 0: tail biting. initialize the encoder K-1 bits in the tail
        -- bit 1: extend input with K-1 zeros

--// Encoded output samples
DATA_OUT: out std_logic;
SAMPLE_CLK_OUT: out std_logic;
SOF_OUT: out std_logic;
        -- one CLK-wide pulse
EOF_OUT: out std_logic;
        -- because of puncturing, this end-of-frame pulse may or may not be aligned with the last
        -- encoded bit in a frame
SAMPLE_CLK_OUT_REQ: in std_logic;
        -- one CLK-wide pulse requesting another sample from the module upstream
        -- used for flow control.

--// Monitoring, test points
TP: out std_logic_vector(4 downto 0)
```

Decoder component interface

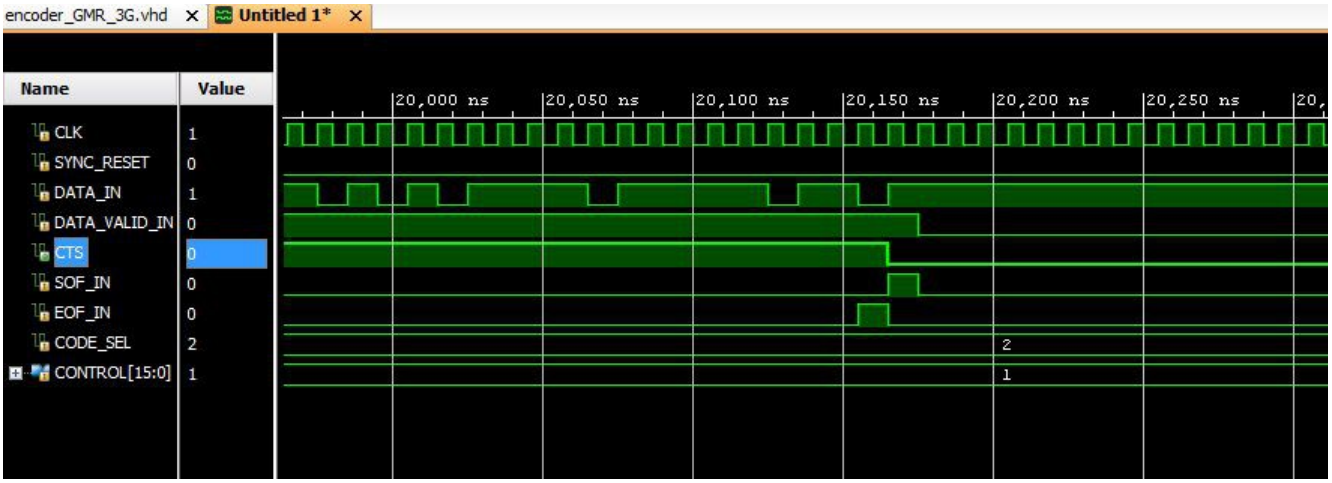
```
-- GLOBAL CLOCKS, RESET
CLK: in std_logic;
        -- synchronous clock. Must be a global clock constrained in the project constraint file.
SYNC_RESET: in std_logic;               -- synchronous reset. active high.
        -- MANDATORY to initialize internal variables

-- SOFT-DECISION INPUT BITS
-- All input samples are soft-quantized with NBSD bits
-- format example for 4-bit input samples:offset binary (0000 for strong '0', 1111 for strong '1')
-- When using hard-decision decoding, input samples should be either all zeros or all ones.
DATAIN: in std_logic_vector((NBSD-1) downto 0);
DATAIN_VALID: in std_logic;
DATAIN_READY: out std_logic;
        -- flow control bit. Always check DATAIN_READY is '1' before sending more input samples.
        -- In most cases, DATAIN_READY will only go low at the end of the frame to ensure a minimum separation between frames
        -- (the decoder need to add a small tail at the end of each frame)
SOF_IN: in std_logic;
        -- 1 CLK-wide pulse indicating start of frames (block mode)
        -- Aligned with the first DATA_VALID_IN.
EOF_IN: in std_logic;
        -- 1 CLK-wide pulse indicating end of frame.
        -- may or may not be aligned with the last DATA_VALID_IN

--// CONFIGURATION
-- The configuration parameters below can be changed dynamically at run-time.
-- They are latched in at the start of frame SOF_IN = '1'.
CODE_SEL: in integer range 0 to 255;
        -- see GMR-1 sections 4.4 and 4.5 for details
        -- 0 = rate 1/2 convolutional code (K = 5) no puncturing
        -- 1 = rate 1/2 convolutional code (K = 5) P(2;3) puncturing rate 3/4
        -- 2 = rate 1/2 convolutional code (K = 5) P(2;5) puncturing rate 5/8
        -- etc
        -- 10 = rate 1/3 convolutional code (K = 5) no puncturing
        -- 20 = rate 1/4 convolutional code (K = 5) no puncturing
        -- 30 = rate 1/5 convolutional code (K = 5) no puncturing
        -- 60 = rate 1/4 convolutional code (K = 6) no puncturing
        -- 64 = rate 1/2 convolutional code (K = 7) no puncturing
        -- 96 = rate 1/2 convolutional code (K = 9) no puncturing
        -- 160 = Rate ¼ Constraint length 9 Convolutional Encoder no puncturing
        -- 192 = Rate 1/3 Constraint length 9 Convolutional Encoder no puncturing
        -- MAXIMUM N_PARITY_BITS IS 5
TAIL_BITING : in std_logic;
        -- tail biting. encoder initialized the encoder K-1 tail bits
        -- LIMIT: 2*TB_DEPTH*N_PARITY_BITS_MAX <= 1024
FRAME_OUT_SIZE: in std_logic_vector(11 downto 0);
        -- expected output frame size, number of decoded bits

-- DECODER OUTPUTS
-- 1-bit serial
DATAOUT1b: out std_logic;
DATAOUT1b_VALID: out std_logic;
SOF_OUT: out std_logic;
EOF_OUT: out std_logic;
        -- 1-bit serial

--// MONITORING (when BER_MEASUREMENT_EN = '1')
-- BER_WINDOW_LENGTH is defined within VA_GMR1_3G.vhd
BER: out std_logic_vector(23 downto 0);
        -- encoded stream bit error rate
BER_VALID: out std_logic;
        -- read BER at rising edge of CLK when BER_VALID = '1'
BIT_ERROR: out std_logic
        -- 1 CLK-wide pulse for each detected bit error
        -- Helpful in understanding the bit error statistics (with an oscilloscope): bursty? or fairly uniformly distributed?
```

I/Os

Encoder input

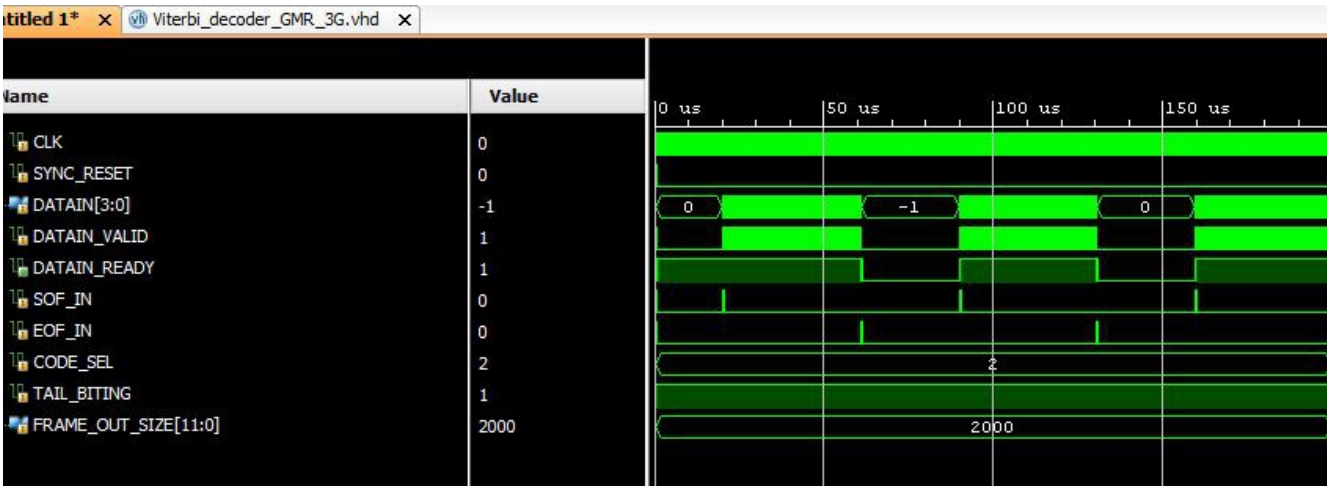


1. All input signals are synchronous with the CLK reference clock
2. SOF_IN (Start Of Frame) is aligned with the first input bit
3. The user should always check the “Clear-To-Send” flag before sending additional data bits to the encoder.



Because the data source latency in responding to the CTS clear-to-send signal from the encoder, the start of the next frame is sent to the encoder. The encoder will store these extra bits until it is ready to encode the complete next frame.

Decoder input



Input encoded frames must be separated as indicated by the DATAIN_READY flag from the decoder. It is NOT acceptable to send the start of the next encoded frame before the current frame is fully decoded.

Software Licensing

The COM-1510SOFT is supplied under the following key licensing terms:

1. A nonexclusive, nontransferable license to use the VHDL source code internally, and
2. An unlimited, royalty-free, nonexclusive transferable license to make and use products incorporating the licensed materials, solely in bitstream format, on a worldwide basis.

The complete VHDL/IP Software License Agreement can be downloaded from http://www.comblock.com/download/softwarelicense.pdf

Configuration Management

The current software revision is 1.

Directory	Contents
/doc	Specifications, user manual, implementation documents
/src	.vhd source code. One component per file.
/sim	Test benches, Matlab .m files to generate encoded input files with various Eb/N0 and to compute expected BER.
/bin	.ngc, .bit, .mcs configuration files

Key files:

Xilinx ISE project file: com-1510_ISE14.xise
Xilinx Vivado project file: /project1/project_1.xpr

VHDL development environment

The VHDL software was developed using the following development environment for VHDL synthesis and VHDL simulation.

(a) Xilinx ISE 14
(b) Xilinx Vivado 2014.2

The size is compatible with free Xilinx WebPack tools.

Device Utilization Summary

The implementation size depends essentially on three key user-defined parameters in the generic section of the decoder, namely:

- Parallel vs Sequential decoding, as defined by **P_SN**
- Hard/Soft-decision decoding as defined by **HARD_SOFTN_DECISION** and **NBSD**
- Maximum constraint length **K_MAX** (the actual constraint length **K** is dynamically configurable at run-time but cannot exceed the hardware capabilities defined by **K_MAX**)
- Maximum number of parity bits **N_PARITY_BITS_MAX**

(the actual number of parity bits **N_PARITY_BITS** is dynamically configurable at run-time but cannot exceed the hardware capabilities defined by **N_PARITY_BITS_MAX**)

- Traceback depth **TB_DEPTH**

Device: Xilinx Spartan-6 -2, **P_SN** = '0' parallel decoding, **K_MAX** = 7, **NBSD** = 4-bit soft decision, **_PARITY_BITS_MAX** = 2, **TB_DEPTH** = 84

	Used 1 decoder	% of Spartan-6 LX45
Registers	2354	4%
LUTs	4083	14%
Block RAM/FIFO	4	3%
GCLKs	1	6%

Maximum frequency: 147.9 MHz

Device: Xilinx Spartan-6 -2, **P_SN** = '0' parallel decoding, **K_MAX** = 9, **HARD_SOFTN_DECISION** = '1', **N_PARITY_BITS_MAX** = 4, **TB_DEPTH** = 120

	Used 1 decoder	% of Spartan-6 LX45
Registers	7318	13%
LUTs	16376	60%
Block RAM/FIFO	10	8%
GCLKs	1	6%

Maximum frequency: 134.1 MHz

Device: Xilinx Kintex-7 -1, , **P_SN** = '0' parallel decoding, **K_MAX** = 9, **HARD_SOFTN_DECISION** = '1', **N_PARITY_BITS_MAX** = 4, **TB_DEPTH** = 120

	Used 1 decoder	% of XC7K70T-1
Registers	7410	9%
LUTs	16023	39%
Block RAM/FIFO	5.5	4%
BUFG	1	3%

Device: Xilinx Kintex-7 -1, , **P_SN** = '1' serial decoding, **K_MAX** = 9, **NBSD** = 4-bit soft decision, **N_PARITY_BITS_MAX** = 4, **TB_DEPTH** = 120

	Used 1 decoder	% of XC7K70T-1
Registers	1148	1%
LUTs	1319	3%
Block RAM/FIFO	12	8%
BUFG	1	3%

Maximum frequency: 252.2 MHz

Clock and decoding speed

The entire design uses a single global clock CLK. Typical maximum clock frequencies for various FPGA families are listed below:

Device family	f_{CLK}
Xilinx Kintex 7 -2	250 - 300 MHz
Xilinx Spartan-6 –2	130 - 150 MHz

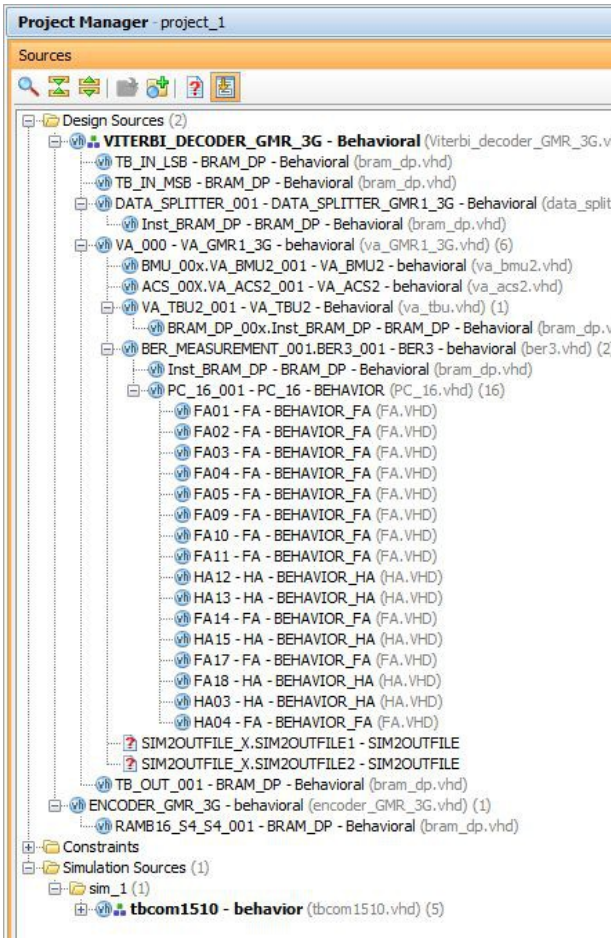
The maximum decoded bit rate is f_{CLK} / **N_PARITY_BITS**

No Xilinx-specific code

The VHDL source code is written in generic VHDL. No Xilinx CORE is used. No Xilinx primitive need to be used. Dual-port RAM blocks are inferred.

VHDL components overview

Top level



ENCODER_GMR_3G.vhd is the convolutional encoder. It supports tail-biting and zero-tail insertion mechanisms. The data source sends a complete frame, as delineated by the SOF_IN and EOF_IN flags. Once a complete input frame is received, the encoder will generate a complete encoded output frame. Thus, the encoding latency is one input frame duration.

VITERBI_DECODER_GMR_3G.vhd is the decoder top component in this hierarchical design. It includes state machines to control tail-biting and insertion of zero tail when applicable.

DATA_SPLITTER_GMR1_3G.vhd demultiplexes each received bit into N_PARITY_BITS bins (one for each code/generator polynomial). It also generates an erasure bit for each punctured parity bit.

VA_GMR1_3G.vhd is the heart of the Viterbi algorith which includes three key processes:

The branch metric unit (*VA_BMU2.vhd*), the Add-Compare-Select (*VA_ACS2.vhd*) and the traceback unit (*VA_TBU2.vhd*).

VA_BMU2.vhd computes the local distance between received soft-quantized bits and the hypothesis being tested. Received bits marked with the erasures flag are ignored in the distance computation.

VA_TBU2.vhd segments the stream of add-compare-select outputs into overlapping blocks of length 2* **TB_DEPTH**., long enough for the Viterbi algorithm to converge. The process is repeated every **TB_DEPTH** bits. The **TB_DEPTH** decoded bits are then read in the reverse order.

BER3.vhd synchronizes with the received bit stream and counts the number of bit error when a PRBS-11 sequence is being transmitted.

INFILE2SIM.vhd reads an input file. This component is used by the testbench to read a soft-quantized or hard-quantized encoded bit stream generated by the *viterbi_dec_gen_input.m* Matlab program for various Eb/No cases.

SIM2OUTFILE.vhd writes three 12-bit data variables to a tab delimited file which can be subsequently read by Matlab (load command) for plotting or analysis.

Test environment

Two VHDL testbenches are included in /sim directories.

tb_viterbi_decoder_GMR_3G.vhd reads a text file of encoded frames, feeds the soft-quantized samples to *VITERBI_DECODER_GMR_3G.vhd* for decoding and sends the decoded bits to the *BER2.vhd* bit error rate tester. A Matlab .m program generates stimulus files for various convolutional codes and additive white Gaussian noise levels. See *viterbi_dec_input_gen_GMR_3G.m* for details.

tbcom1510.vhd is a testbench consisting of a back-to-back PRBS-11 pseudo-random sequence

generator, convolutional encoder, Viterbi decoder and bit error rate tester. No stimulus file is needed.

Reference documents

[1] Geo-Mobile radio interface specifications (Release 3) ETSI TS 101 376-5-3 v3.1.1. 2009-7

GMR-1 3G 45.003

ComBlock Ordering Information

COM-1510SOFT BLOCK MODE CONVOLUTIONAL CODEC, VHDL SOURCE CODE / IP CORE

ECCN: 5E001.b.4

MSS • 845-N Quince Orchard Boulevard•
Gaithersburg, Maryland 20878-1676 • U.S.A.
Telephone: (240) 631-1111
Facsimile: (240) 631-1676
E-mail: sales@comblock.com