ComBlock

COM-1812SOFT CCSDS LDPC AR4JA codes encoder/decoder VHDL source code overview / IP core

Overview

The COM-1812SOFT is a LDPC code error correction encoder/decoder compliant with CCSDS and IRIG specifications. It is written in generic portable VHDL.

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

Key features and performance:

- Includes encoding, decoding, frame synchronization and data randomization.
- Compliant with the AR4JA codes specified in CCSDS 131.0-B-3, Blue Book, section 7.4, 9, 10.
  Compliant with IRIG standard 106-15 Appendix R.
- User-selected configuration:
  - Information block lengths k:
    1024, 4096 bits (selected dynamically at runtime or prior to synthesis to control device utilization)
  - Code rates 1/2, 2/3, and 4/5 (at runtime)
- Typical Bit Error Rate / Frame Error Rate for rate 1/2 k=4096:
  BER = 10^{-6} FER = 1 10^{-5} @ E_b/N_o = 1.35 dB
- Throughput:
  Encoding: > 100 Mbits/s
  Decoding: [0.19-0.61]* FPGA clock frequency @ 10^{-5} BER
- Provided with IP core:
  - VHDL source code
  - Matlab .m file for simulating the encoding and decoding algorithms, for generating stimulus files for VHDL simulation and for end-to-end BER/FER performance analysis at various signal-to-noise ratios
  - VHDL testbench

- See COM-1811SOFT for CCSDS LDPC C2 (rate ~7/8) codec.

Portable VHDL code

The code is written in generic standard VHDL and is thus portable to a variety of FPGAs. The code was developed and tested on a Xilinx 7-series FPGA but is expected to work similarly on other targets. No manufacturer-specific primitive is used.

Configuration

Synthesis-time configuration parameters

The following constants are user-defined in the decoder component generic section prior to synthesis. These parameters generally define the size of the decoder embodiment.

Synthesis-time configuration parameters	
Encoder & Decoder **CCSDS_LDPC_ENC_B, CCSDS_LDPC_DEC_B**	
Circulant matrices enabled **CM_ENABLED**	Instantiate the resources necessary to implement various frame lengths k and coding rates. k=1024 bit 0 = rate 1/2 bit 1 = rate 2/3 bit 2 = rate 4/5 k=4096 bit 3 = rate 1/2 bit 4 = rate 2/3 bit 5 = rate 4/5 k=16384 (reserved) bit 6 = rate 1/2 bit 7 = rate 2/3 bit 8 = rate 4/5
Decoder **CCSDS_LDPC_DEC_B**	
Number of soft-quantized bits at the decoder input **IN_NBITS**	Typical value: 4. A minor performance improvement can be achieved with 5-bits.
Decoder maximum number of iterations **N_ITER_MAX**.	The higher the number of iterations, the better the error correction performance. Not much improvement above 50. The decoder stops the iterative process as soon as all parity checks are verified, or when it reaches **N_ITER_MAX** iterations, whichever happens first.
Randomizer / De-randomizer **CCSDS_RANDOMIZER** **CCSDS_DERANDOMIZER**	
IO_BIT_ORDER	always '1' = LSb first for compatibility with encoder / decoder.
Number of frames between sync markers **NFRAMES**	generally 1
SYNC_WORD	0: 32-bit x"1ACFFC1D", MSb first 1: 64-bit x"034776C7272895B0" , MSb first
NBITS	Soft-decision input precision. Typically 4 or 5 bits.

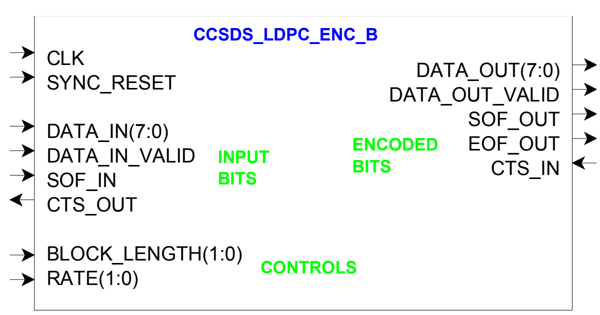
I/Os

General

CLK: input
The synchronous clock. The user must provide a global clock (use BUFG). The CLK timing period must be constrained in the .xdc file associated with the project.

SYNC_RESET: input
Synchronous reset. The reset MUST be exercised at least once to initialize the internal variables. It must be exercised whenever a control parameter is changed.

Encoder



BLOCK_LENGTH(1:0):
Information block length k:
0 = 1024 bits
1 = 4096 bits
2 = 16384 bits (reserved).
Enacted at SYNC_RESET.

RATE(1:0): coding rate:
0 = 1/2
1 = 2/3
2 = 4/5
Enacted at SYNC_RESET.

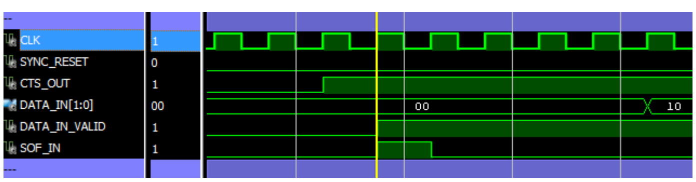
DATA_IN(7:0): Input data is read one Byte at a time. Bits are packed LSb first. Always a full Byte, no partial Byte allowed.

DATA_IN_VALID: input.
1 CLK-wide pulse indicating that DATA_IN is valid.

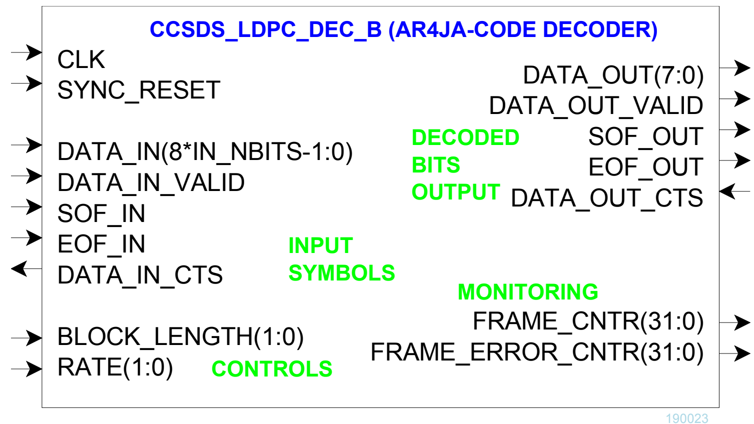
SOF_IN: input Start Of Frame. 1 CLK-wide pulse. The SOF is aligned with **DATA_IN_VALID**. Note that there is no need for an end of frame as the input frame size is determined by the **BLOCK_LENGTH** selection. Input bits in excess are discarded.

CTS_OUT: output.
Clear-**T**o-**S**end flow control. '1' indicates that the encoder is ready to accept another input byte. The encoder stops requesting input data when the input elastic buffer is 3/4 full.

The encoder outputs mirror its inputs: **DATA_OUT(7:0)**, **DATA_OUT_VALID**, **SOF_OUT**, **EOF_OUT**, **CTS_IN**.



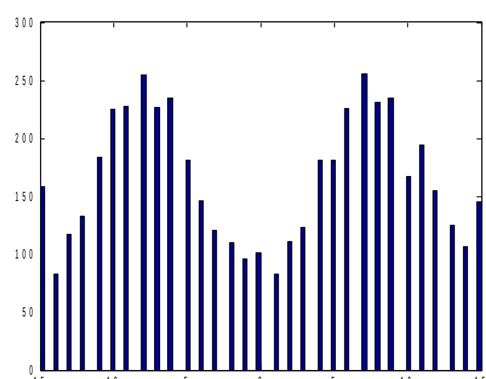
Decoder



DATA_IN(8*IN_NBITS-1:0): eight soft-quantized input symbols. The precision (**IN_NBITS**) is selectable at the time of synthesis. Typical values are 4- or 5-bit soft-quantization. The soft-quantized input symbols are expected to be symmetrical around zero, for example ranging from -7 to +7 or -15 to +15
although this rule is enforced within.

Convention: throughout the code, a positive symbol represents a '1', negative a '0'. The eight symbols are packed LSb first.

Usage: it is expected that the demodulator preceding this decoder will normalize the demodulated samples prior to soft-quantization by using an AGC loop. The AGC target level is important in maximizing the decoder BER performance. The optimum level is such that the mean absolute value of the amplitude is at mid-range. For example in the case of **IN_NBITS**=5 (range -15 to + 15), the input samples should be scaled so that the mean absolute value is near 7.5.



Example of input sample distribution
(IN_NBITS = 5, Eb/No = 3.0 dB, rate 4/5)

DATA_IN_VALID: input.
1 CLK-wide pulse indicating that **DATA_IN** is valid.

SOF_IN / EOF_IN: inputs Start Of Frame and End Of Frame. 1 CLK-wide pulses.
A aligned with **DATA_IN_VALID**.
Each frame consists of **RATE** *1024/4096/16384 symbols, entered 8 at a time.

DATA_IN_CTS: output **C**lear-**T**o-**S**end flow control. '1' indicates that the decoder is ready to accept another group of 8 parallel input symbols.

The decoder outputs mirror its inputs:
DATA_OUT(7:0), **DATA_OUT_VALID**, **SOF_OUT**, **EOF_OUT, DATA_OUT_CTS**.
Output data **DATA_OUT** is sent one Byte at a time. Bits are packed LSb first.

Performance

Encoder information throughput

The maximum encoder (information) input rate depends on the information frame length (k), the encoding Rate and the processing clock frequency.

k \ Rate	1/2	2/3	4/5
1024 bits	1617 clocks / frame 101.3 Mbits/s @ 160 MHz	1889 clocks / frame 86.7 Mbits/s @ 160 MHz	2529 clocks / frame 64.7 Mbits/s @ 160 MHz
4096 bits	5457 clocks / frame 120.0 Mbits/s @ 160 MHz	5537 clocks / frame 118.3 Mbits/s @ 160 MHz	6081 clocks / frame 107.7 Mbits/s @ 160 MHz

Decoder iteration time

Number of clocks per decoding iteration

Decoder configuration	clocks
k=1024, rate 1/2	385
k=1024, rate 2/3	257
k=1024, rate 4/5	385
k=4096, rate 1/2	1537
k=4096, rate 2/3	1025
k=4096, rate 4/5	769

In this implementation, the minimum number of iterations is two.

Decoder average number of iterations vs Eb/No

The average number of iterations affects the overall decoder throughput. It is a function of k, rate, and the threshold operating Eb/No.

Decoder configuration	Threshold Eb/No (dB)	BER	Average number of decoding iterations
k=1024, rate 1/2	1.9	10^{-5}	10.3
k=1024, rate 2/3	2.8	10^{-5}	6.1
k=1024, rate 4/5	3.7	10^{-5}	4.6
k=4096, rate 1/2	1.3	10^{-5}	13.2
k=4096, rate 2/3	2.1	10^{-5}	13.7
k=4096, rate 4/5	3.1	10^{-5}	7.8

Decoder throughput

The decoder operates in two phase: Input/output phase and iterative decoding phase.

During the I/O phase, encoded soft-quantized inputs and decoded binary outputs can be concurrent. Given the 8-symbol parallel input, the minimum duration is k/8*rate clocks:

The minimum decoding phase spans two iterations.
The maximum decoding phase typically spans 50 iterations.

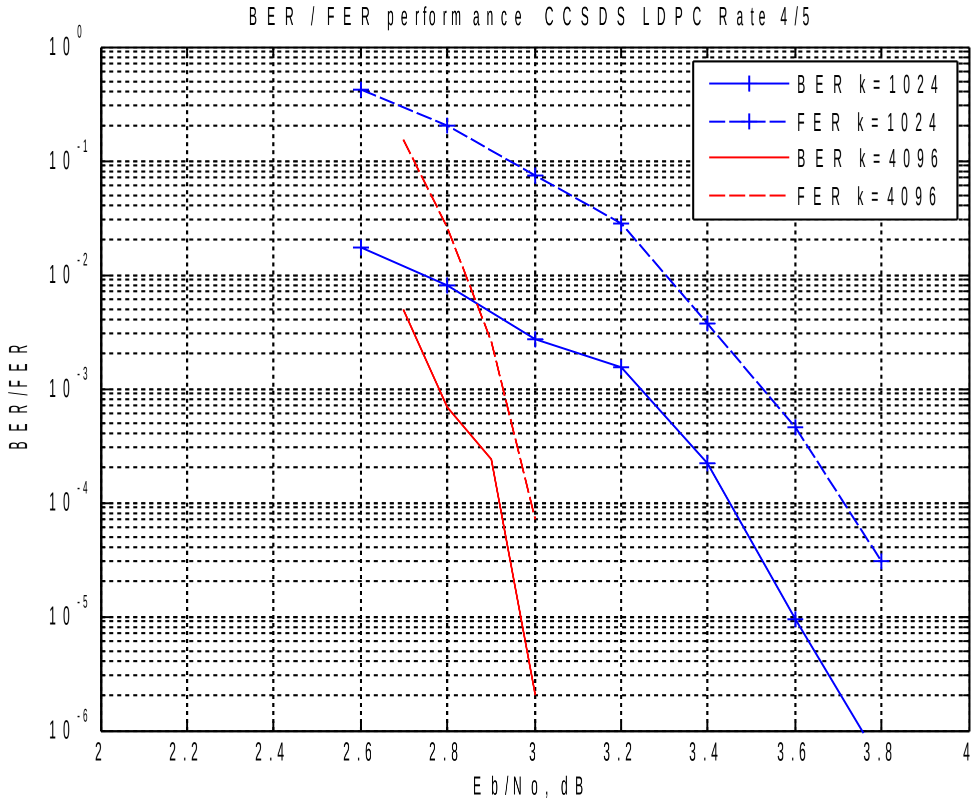
Throughput examples (at Eb/No threshold for 10^{-5} BER):

FPGA speed	Configuration	Min number of clocks	Average number of clocks (for 10^{-5} BER)	Average decoded throughput
Artix7-100T-1 80 MHz	k = 1024, rate = 1/2	1024*2/8+385*2 = 1026 clks	1024*2/8+385*10.3 = 4222 clks	19.4 Mbits/s
Artix7-100T-1 80 MHz	k = 1024, rate = 2/3	1024*3/16+257*2 = 706 clks	1024*3/16+257*6.1 = 1760 clks	46.5 Mbits/s
Artix7-100T-1 80 MHz	k = 1024, rate = 4/5	1024*5/32+385*2 = 930 clks	1024*5/32+385*4.6 = 1931 clks	42.4 Mbits/s
Artix7-100T-1 80 MHz	k = 4096, rate = 1/2	4096*2/8+1537*2 = 4098 clks	4096*2/8+1537*13.2 = 21312 clks	15.4 Mbits/s
Artix7-100T-1 80 MHz	k = 4096, rate = 2/3	4096*3/16+1025*2 = 2818 clks	4096*3/16+1025*13.7 = 14811 clks	22.12 Mbits/s
Artix7-100T-1 80 MHz	k = 4096, rate = 4/5	4096*5/32+769*2 = 2178 clks	4096*5/32+769*7.8 = 6639 clks	49.3 Mbits/s

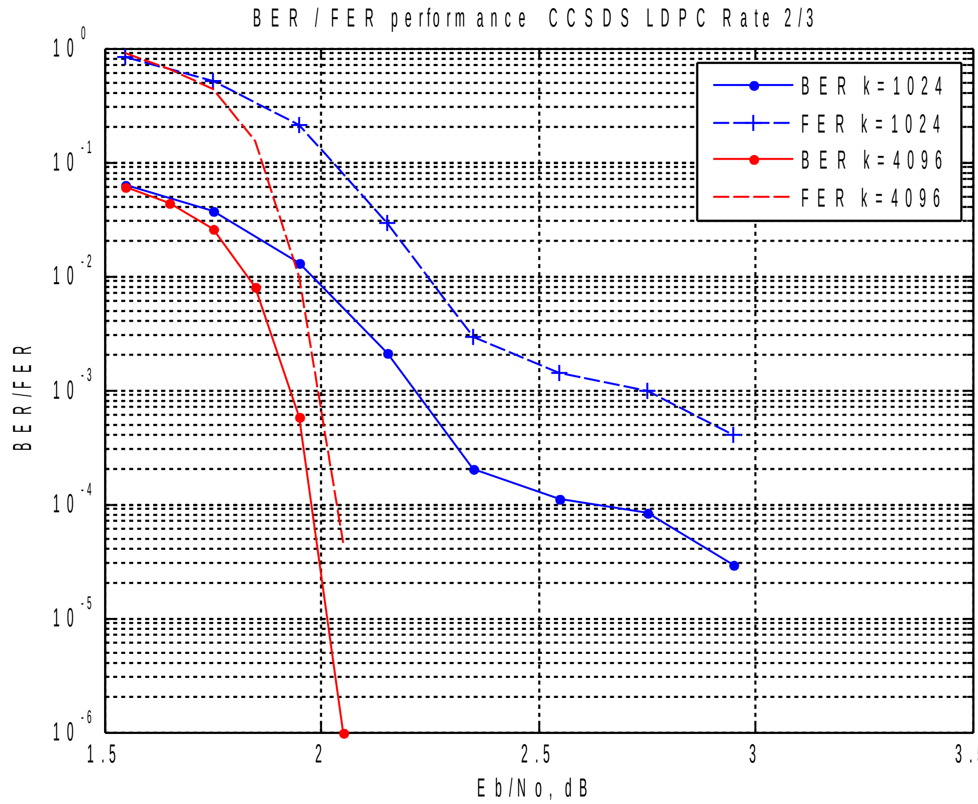
BER/ FER performance

The decoded errors are somewhat bursty in nature, with many error-free decoded frames followed by an occasional erroneous frame with multiple bit errors. Therefore, we also express the decoder performance in terms of frame error rate (FER).

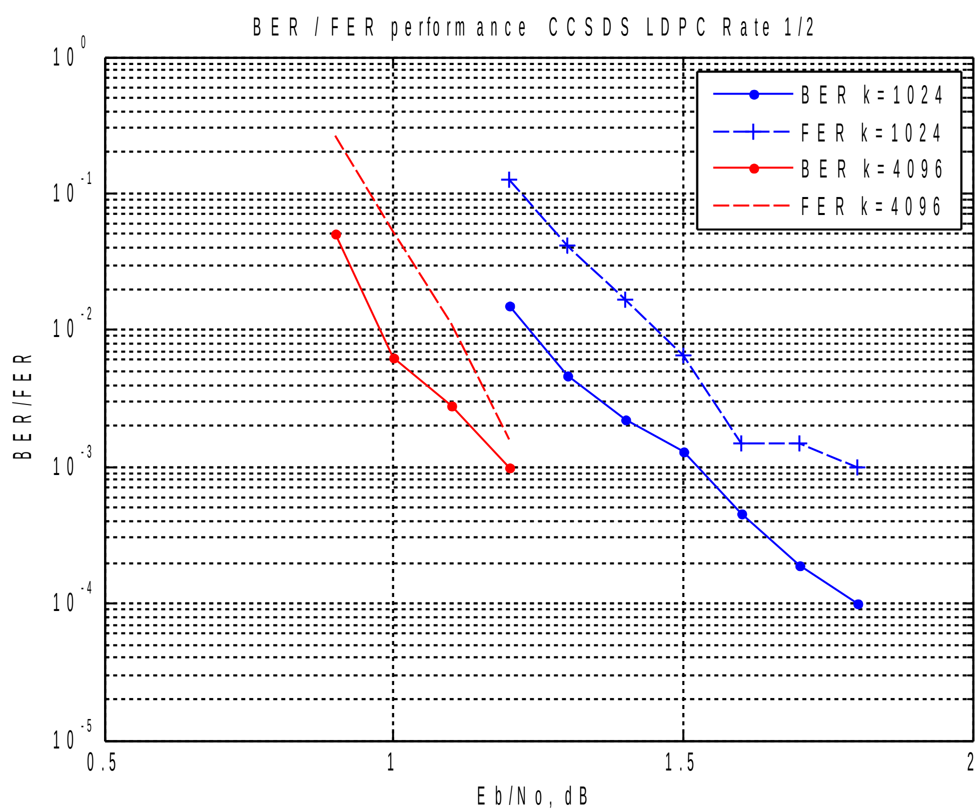
Test conditions: rate 4/5, k=1024 and k=4096, 50 iterations, 5-bit soft-quantization



Test conditions: rate 2/3, k=1024 and k=4096, 50 iterations, 5-bit soft-quantization, LQNBITS=9



Test conditions: rate 1/2, k=1024 and k=4096, 50 iterations, 5-bit soft-quantization



Computation precision

The computation precision (LQNBITS in the code) affects the BER. We selected LQNBITS = 8 bits as a good tradeoff between performance and device utilization. The next precision (LQNBITS = 9) improves the BER performance by approximately 0.07 dB.

Software Licensing

The COM-1812SOFT 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 bit stream 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,.pkg packages, .xdc constraint files (Xilinx) One component per file.
/sim	VHDL test benches
/matlab	Matlab .m file for simulating the encoding and decoding algorithms, for generating stimulus files for VHDL simulation and for end-to-end BER performance analysis at various signal to noise ratios
/bin	.bit configuration files (for use with ComBlock COM-1800 FPGA development platform)

Project files:

Xilinx ISE 14 project file: com-1812.xise
Xilinx Vivado v2017.4 project file: project_2.xpr

VHDL development environment

The VHDL software was developed using the following development environment:

(a) Xilinx ISE 14.7 for synthesis, place and route
(b) Xilinx Vivado 2017.4 for synthesis, place and route and VHDL simulation

The entire project fits easily within a Xilinx Artix7-100T. Therefore, the ISE project can be processed using the free Xilinx WebPack tools.

Reference documents

[1] CCSDS "Recommended Standard for TM Synchronization and Channel Coding",

CCSDS 131.0-B-3, Blue Book, September 2017.

Applicable sections:
Section 7.4: Low-density parity-check code family with rates 1/2, 2/3 and 4/5 (AR4JA code)

Section 9: Frame Synchronization

Section 10: Pseudo-Randomizer

[2] Telemetry Standards, IRIG Standard 106-15 (Part 1), Appendix R, July 2015

[3] "Implementing the NASA Deep Space LDPC Codes for Defense Applications", Zhao, Long, 2013.

[4] 'Efficient Implementations of the Sum-Product Algorithm for Decoding LDPC Codes",

Xiao-Yu Hu, Evangelos Eleftheriou, Dieter-Michael Arnold, and Ajay Dholakia, 2001

Device Utilization Summary

AR4JA LDPC encoder k=1024, all 3 rates		% of Xilinx Artix7-100T
Registers	688	0.5%
LUTs	1348	2.1%
Block RAM/FIFO 36Kb	2.5	1.9%
DSP48	0	0%
GCLKs	1	3.1%
AR4JA LDPC encoder k=4096, all 3 rates		% of Xilinx Artix7-100T
Registers	2482	2.0%
LUTs	5199	8.2%
Block RAM/FIFO 36Kb	8.5	6.3%
DSP48	0	0%
GCLKs	1	3.1%

Randomizer + sync marker		% of Xilinx Artix7-100T
Registers	47	<0.1%
LUTs	59	<0.1%
Block RAM/FIFO 36Kb	0	0%
DSP48	0	0%
GCLKs	1	3.1%

Sync marker detection + de-randomizer		% of Xilinx Artix7-100T
Registers	926	0.8%
LUTs	1110	1.8%
Block RAM/FIFO 36Kb	0	0%
DSP48	0	0%
GCLKs	1	3.1%

AR4JA LDPC decoder Smallest case k=1024, rate 4/5, IN_NBITS=4		% of Xilinx Artix7-100T
Registers	11736	9.3%
LUTs	22497	35.5%
Block RAM/FIFO 36Kb	16	11.9%
DSP48	0	0%
GCLKs	1	3.1%
AR4JA LDPC decoder k=1024, all 3 rates IN_NBITS=5		% of Xilinx Artix7-100T
Registers	11167	8.8%
LUTs	21631	34.1%
Block RAM/FIFO 36Kb	48	35.6%
DSP48	0	0%
GCLKs	1	3.1%
AR4JA LDPC decoder k=1024,4096 all 3 rates IN_NBITS =5		% of Xilinx Artix7-100T
Registers	11302	8.9%
LUTs	21840	34.2%
Block RAM/FIFO 36Kb	49	36.3%
DSP48	0	0%
GCLKs	1	3.1%

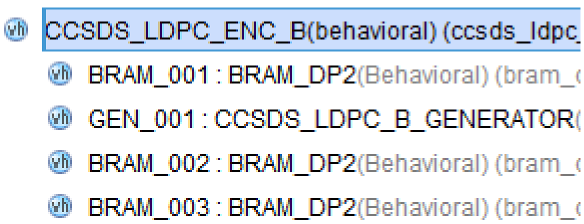
Clock and decoding speed

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

Device family	**Encoder**	**Decoder**
Xilinx Artix 7 -1 (lowest) speed grade k_{max} = 1024 bits	132 MHz	84.5 MHz
Xilinx Artix 7 -1 (lowest) speed grade k_{max} = 4096 bits	166 MHz	83.8 MHz
Xilinx Kintex 7 -2 k_{max} = 4096 bits		

VHDL components overview

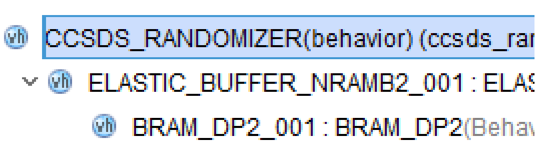
Encoder top level



The *CCSDS_LDPC_ENC_B* component buffers the input Byte stream and computes the parity bits for each input frame. The concatenated information bits and parity bits are sent to the output. Both inputs and outputs are 8-bit parallel.

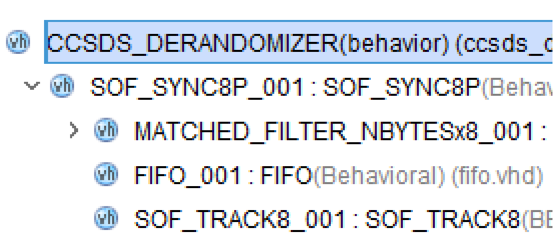
The *CCSDS_LDPC_B_GENERATOR* component retrieves the stored first rows of the circulant matrices for the LDPC AR4JA code. There are 8 columns and 8/16/32 rows of circulant matrices of size m*m, where m =32 to 2048 bits, depending on k and rate. The matrices are read row by row from the upper left (1) to right (8) and top to bottom. It takes 1,2 or 4 clock cycles to read each circulant matrix top row.

BRAM_DP2.vhd is a generic dual-port memory, used as input and output elastic buffers. Memory is inferred for code portability (no primitive is used).



The *CCSDS_RANDOMIZER.vhd* component performs bit stream pseudo-randomization and sync marker insertion as per sections 9 and 10 of the specifications [1].

Decoder top level

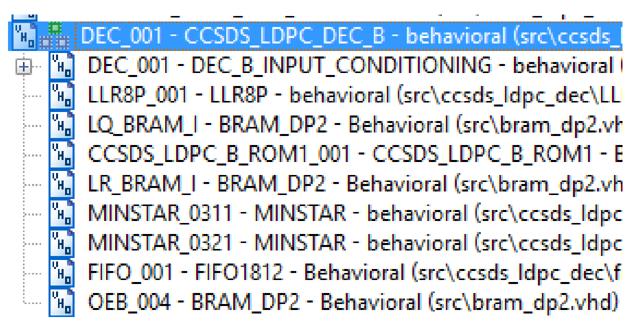


The *CCSDS_DERANDOMIZER.vhd* component detects and removes the periodic sync markers, reconstructs the start of frame and end of frame pulses and descrambles the received soft-quantized bit stream. It complies with sections 9 and 10 of the specifications [1].

The *SOF_SYNC8P.vhd* component detects, confirms and removes the periodic sync markers. It includes a fly-wheel mechanism to reconstruct the frame structure in the event of high bit errors. It also reports and corrects the input symbols bit to Byte packing alignment. Finally, it monitors the bit error rate within the received sync markers. I/Os are 8-symbols in parallel.

MATCHED_FILTER_NBYTESx8.vhd: a 64-bit matched filter operating on 8-parallel 1-bit hard-quantized input symbols. The matched filter detects a match 'on-the-fly' on all 8 possible bits/byte alignments. It also report inverted sequences. Default detection threshold is 10 mismatches out of 64 (15.6% BER). The threshold can be adjusted through the DETECT_THRESHOLD generic parameter.

SOF_TRACK8.vhd: Confirmation circuit for the frame synchronization. It generates a reliable SOF_LOCK_DETECT status based on the detection of the periodic sync marker at the expected time.



CCSDS_LDPC_DEC_B.vhd performs the iterative error correction decoding. The decoding stops when all parity checks are verified or when the number of decoding iterations reaches the maximum N_ITER_MAX, whichever occurs first. Each decoding iteration takes between 257 and 1537 clocks depending on (k,rate). Eight input symbols are entered in parallel to maximize throughput.

DEC_B_INPUT_CONDITIONING.vhd appends M zero symbols (punctured at transmission). Zero means 'could be a bit 0 or bit 1 with equal probability.

LLR8P.vhd computes the LLR for each soft-decision input sample. The LLR is $2*y_i/\sigma^2$ where y_i are the soft-decoded input samples and σ^2 the noise variance. Although the component can scale the samples as a function of the SNR, a fixed SNR is set in the code as a tradeoff between computation precision and algorithm accuracy.

CCSDS_LDPC_B_ROM1.vhd is a generic dual-port ROM customized for reading v-node addresses, in c-node sequence. In effect, scanning the non-zero elements of the parity check matrix H horizontally from top to bottom.

MINSTAR2.vhd computes the minstar* function as described in [3] and [4].

MINSTAR.vhd computes the minstar* function as described in [3] while using fewer resources than *MINSTAR2.vhd.*

FIFO1812.vhd delays the Lqij (L4) by 3/6/10/16 + 3 CLK, to align with the Lrji (LR2x_D)

INFILE2SIM.vhd reads an input file. This component is used by the testbench to read a 5-bit soft-quantized encoded bit stream generated by the `ccsds_ldpc_b.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.

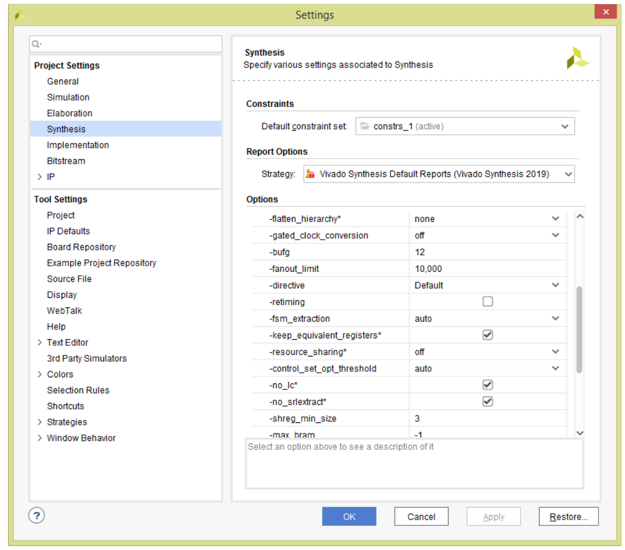
VHDL simulation

The two main bit-accurate VHDL simulation avenues are:

- *COM1802.vhd*, an end-to-end simulation testbench encompassing encoder, decoder, randomization, de-randomization, sync marker insertion, sync marker detection, PRBS-11 test sequence generation and BER tester. Set the SIMULATION generic parameter to '1' prior to starting the VHDL simulation.

- *TB_CCSDS_LDPC_DEC_B.vhd* is the decoder testbench. Its input consists of soft-quantized noisy samples generated by the supplied Matlab program `ccsds_ldpc_b.m`.

Xilinx Vivado: Synthesis settings
(* denotes changes from the default settings)



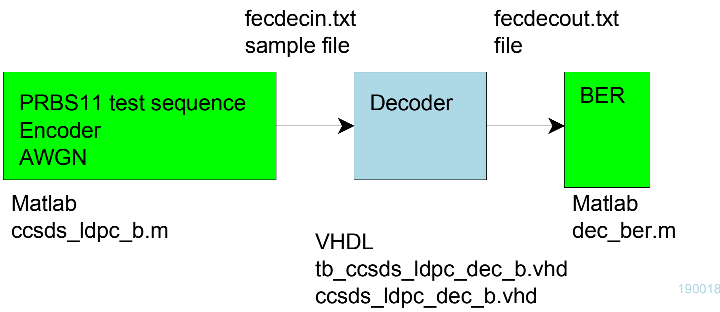
Matlab simulation

The `ccsds_ldpc_b.m` program

- generates a stimulus file `fecdecin.txt` for use as input to the decoder VHDL simulation. The file includes a frame of pseudo-random (PRBS11) data bits, LDPC encoding, Additive White Gaussian Noise and 4- or 5-bit soft-quantization.
- Performs end-to-end BER performance analysis of the LDPC-codec over a noisy (AWGN) channel.

The `ccsds_ldpc_b_vhdlalgo.m` program simulates a decoding algorithm representative of the actual VHDL implementation, instead of a generic decoding algorithm.

The `dec_ber.m` program reads a file of decoded data `fecdeccout.txt`
generated by VHDL simulation and compare it with the original PRBS-11 test sequence. It counts the number of bit errors.



When moving the project folder location, be sure to change accordingly the FILENAME file paths in *tb_ccsds_ldpc_dec_b.vhd* INFILE2SIM and SIM2OUTFILE components generic section.

The following .m programs were used during the design:

`ccsds_ldpc_b_H.m` generates the three parity check matrices $H_{1/2}$, $H_{2/3}$ and $H_{4/5}$ for the CCSDS AR4JA family of LDPC codes with rates 1/2, 2/3 and 4/5 respectively, as per [1], section 7.4. This program uses the `ccsds_ldpc_b_PIk.m` for generating the permutation submatrices.

Two methods are used to construct the generator matrix G for the CCSDS AR4JA family of LDPC codes:
`ccsds_ldpc_b_G1.m` is tabled-based (see [2]). It is fast but limited to a few use cases.
`ccsds_ldpc_b_G2.m` uses the matrix inversion method described in [1] section 7.4.3. Computation is slow but includes all use cases and must run only once. The resulting generator matrices G_rate_k.mat are saved as files in the /matlab folder for subsequent use.

The `ccsds_ldpc_b_rom1.m` program is a design utility to generate look-up tables for the *ccsds_ldpc_b_rom1.vhd* component. At each clock of the parity check phase, the look-up table returns an LQi RAM address according to the H parity check matrix non-zero elements (equivalent to scanning the H matrix from left (v-node 0) to right, then top (c-node 0) to bottom.

Implementation Overview

AR4JA code decoder

A received frame consists of k/rate soft-quantized symbols with 4 or 5 bits each.

Zeros are appended to the received frame to make room for the punctured bits removed at encoding. The punctured bits will be reconstructed and refined at each decoding iteration.

The decoder output frame comprises k=1024 or 4096 bits.

The decoder uses two groups of block RAMs to store the LQi and Lrji respectively

Computations are performed with the following precision:
LQi: 8-bit signed fixed-point 5.3
Lrji: 6 bit signed fixed point 4.2
LLR input: 4 or 5-bit

Messages between v-nodes and c-nodes are computed in parallel by groups of 64 to maximize throughput. It takes 3,6,10 or 18 clocks to compute the v-nodes to c-nodes messages depending on the number of non-zero elements in the parity check matrix H row.

Lr memory organization

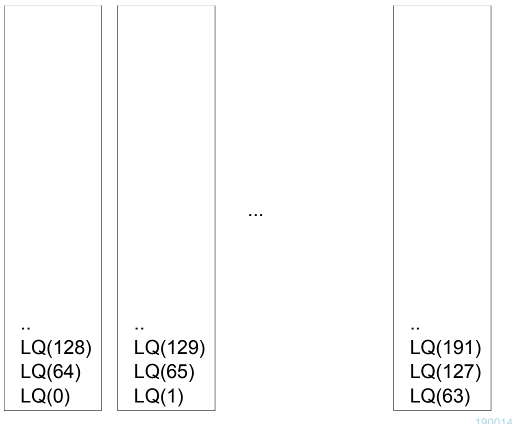
All Lrji are saved in dual-port block RAM with signed fixed-point format 4.2.
11 BRAMs are used, with 6 Lrji per 36-bit word, thus allowing access to 64 Lrji per clock.
The dual-port A-side is reserved exclusively for writing while the B-side is for reading only.

The total memory size requirement for storing the Lrji is:

```
--// Lrji MEMORY ---------------------------------
-- The Lr(j->i) messages are stored in 11 BRAMs.
-- 3*1*512 + 6*2*512 = 7680 Lrji when k=1024, rate 1/2
-- 3*1*2048 + 6*2*2048  = 30720 Lrji when k=4096, rate 1/2
-- 3*1*8192 + 6*2*8192  = 122880 Lrji when k=16384, rate 1/2
-- 3*1*256 + 10*2*256   = 5888 Lrji when k=1024, rate 2/3
-- 3*1*1024 + 10*2*1024 = 23552 Lrji when k=4096, rate 2/3
-- 3*1*4096 + 10*2*4096 = 94208 Lrji when k=16384, rate 2/3
-- 3*1*128 + 18*2*128   = 4992 Lrji when k=1024, rate 4/5
-- 3*1*512 + 18*2*512   = 19968 Lrji when k=4096, rate 4/5
-- 3*1*2048 + 18*2*2048 = 79872 Lrji when k=16384, rate 4/5
```

LQ memory organization

The LQi are stored in 64 BRAMs with 8-bit wide data path. The LQ memory organization is illustrated below:



64 dual-port BRAMs

Thus, 64 LQi can be read and written in one clock cycle. Each BRAM stores 40/160/640 LQi with a precision of 8-bits, for k=1024/4096/16384 bits respectively.

Lr_{ji} memory organization

The Lrji variables are stored in 11 BRAMs. They are written and read at a rate of 64 Lrji per clock. (packed 6*6-bit wide in each BRAM).

Matlab-VHDL

Although the Matlab program `ccsds_ldpc_b.m` and the VHDL component ccsds_ldpc_dec_b.vhd implement the same fundamental algorithm, the VHDL code can be difficult to understand. To help follow the algorithmic steps, a list of the key variable names in VHDL and their corresponding names in the Matlab program is shown below:

Algorithm	Matlab variable	VHDL variable
LLR	Lc	LLR1 8 sample-word, fixed-point format 5.3
LQ_i	LQ	LQ3 64-sample word, updated before saving to LQ memory. LQ1 64-sample word after reading from LQ memory. LQ2 re-arranged LQ_i for c-nodes computation. 64 c-nodes in parallel. All are signed fixed-point format 5.3
Lr_{ji}	Lr	LR2x 64-samples before saving to Lr_{ji} memory. LR1x 64 samples signed after reading from Lr_{ji} memory. Both are signed fixed point format 5.3 during computation but saved with truncated 6-bit precision in block RAM memory.
Lq_{ij}		L4 or L7(after 3/6/10/16 + 3 CLK delay) signed fixed-point format 5.3

A key algorithmic difference between the reference Matlab simulation and the VHDL code is the LQi / Lrji update rate. The Matlab algorithm updates the LQi and Lrji in-block once every iteration, while the VHDL code does a progressive update every 64 check nodes. Consequently, the VHDL code converges faster and requires fewer iterations

(except in the error-free case where the minimum number of iterations is 2)

Acronyms

Acronym	Definition
AWGN	Additive White Gaussian Noise
CCSDS	Consultative Committee For Space Data Systems
BRAM	Dual-port Block RAM
CCSDS	Consultative Committee For Space Data Systems
LDPC	Low-Density Parity-Check
LLR	Log-Likelihood Ratio
LSb	Least Significant bit
MSb	Most Significant bit
PRBS-11	Pseudo-Random Binary Sequence, 2047-bit period

ComBlock Ordering Information

COM-1812SOFT CCSDS LDPC AR4JA codes encoder/decoder. VHDL source code / IP core

Contact Information

MSS • 845-N Quince Orchard Boulevard • Gaithersburg, Maryland 20878-1676 • U.S.A.

Telephone: (240) 631-1111
Facsimile: (240) 631-1676
E-mail: info@comblock.com