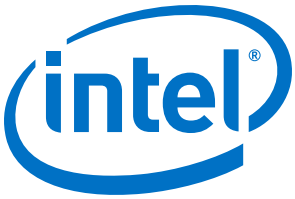


LDPC IP Core User Guide

Updated for Intel® Quartus® Prime Design Suite: **17.1**

Subscribe
Send Feedback

UG-01156 | 2017.11.06
Latest document on the web: **PDF** | **HTML**

Contents

1 About the LDPC IP Core

Low-density parity-check (LDPC) codes are linear error correcting codes that allow you to transmit messages over noisy channels. The Altera® LDPC IP core implements LDPC codes in your design.

Related Links

- LDPC IP Core Document Archive on page 25
  Provides a list of user guides for previous versions of the LDPC IP core.
- Introduction to Intel FPGA IP Cores
  Provides general information about all Intel FPGA IP cores, including parameterizing, generating, upgrading, and simulating IP cores.
- Creating Version-Independent IP and Qsys Simulation Scripts
  Create simulation scripts that do not require manual updates for software or IP version upgrades.
- Project Management Best Practices
  Guidelines for efficient management and portability of your project and IP files.

1.1 LDPC IP Core Features

The LDPC IP Core targets these standards:

- DOCSIS 3.1
- — Decoder only
  - — On-the fly switching between code
- WiMedia 1.5
  - — Encoder and decoder
  - — Optional on-the fly switching between code
  - — Supports short and long frame
- DVB-S2
  - — Encoder only
- NASA GSFC-STD-9100
  - — Encoder and decoder
  - — Optional low resource architecture
  - — MSA or layered MSA decoding

All decoders have:

- MATLAB models
- Double-buffered architecture to reduce latency and boost throughput
- Early stopping criterion
- Parameters for:
  - Input parallelism
  - Decoding parallelism
  - LLR width
  - Attenuation factor

1.2 LDPC IP Device Family Support

Intel offers the following device support levels for Intel FPGA IP cores:

- Advance support—the IP core is available for simulation and compilation for this device family. FPGA programming file (`.pof`) support is not available for Quartus Prime Pro Stratix 10 Edition Beta software and as such IP timing closure cannot be guaranteed. Timing models include initial engineering estimates of delays based on early post-layout information. The timing models are subject to change as silicon testing improves the correlation between the actual silicon and the timing models. You can use this IP core for system architecture and resource utilization studies, simulation, pinout, system latency assessments, basic timing assessments (pipeline budgeting), and I/O transfer strategy (data-path width, burst depth, I/O standards tradeoffs).
- Preliminary support—Intel verifies the IP core with preliminary timing models for this device family. The IP core meets all functional requirements, but might still be undergoing timing analysis for the device family. You can use it in production designs with caution.
- Final support—Intel verifies the IP core with final timing models for this device family. The IP core meets all functional and timing requirements for the device family. You can use it in production designs.

Table 1. DSP IP Core Device Family Support

Device Family	Support
Arria® II GX	Final
Arria II GZ	Final
Arria V	Final
Intel® Arria 10	Final
Cyclone® IV	Final
Cyclone V	Final
Intel Cyclone 10 GX	Final
Intel MAX® 10 FPGA	Final
Stratix® IV GT	Final
Stratix IV GX/E	Final

continued...

Device Family	Support
Stratix V	Final
Intel Stratix 10	Advance
Other device families	No support

1.3 LDPC IP Core Release Information

You need the release information to licence the IP core.

Table 2. Release Information

Item	Description
Version	17.1
Release Date	November 2017
Ordering Code	IP-LDPC (IPR-LPDC)

Intel verifies that the current version of the Quartus Prime software compiles the previous version of each IP core. Intel does not verify that the Quartus Prime software compiles IP core versions older than the previous version. The *Intel FPGA IP Release Notes* lists any exceptions.

Related Links

- Intel FPGA IP Release Notes
- Errata for LDPC IP core in the Knowledge Base

1.4 DSP IP Core Verification

Before releasing a version of an IP core, Intel runs comprehensive regression tests to verify its quality and correctness. Intel generates custom variations of the IP core to exercise the various parameter options and thoroughly simulates the resulting simulation models with the results verified against master simulation models.

1.5 LDPC IP Core Performance

Typical expected performance for a LDPC IP Core using the Quartus Prime software with Arria 10 (10AX115R2F40I1SG and 10AX115R4F40I3SG) devices.

The performance tables use the following parameters:

- soft is **Number of soft bits of the decoder variables**
- par is **Parallelism**
- lps is **Number of LLRs per input symbol**
- full is **Full-streaming architecture**
- lay is **Layered decoding**
- short is **Shorten codeword**
- rate is **Coding rate**

Decoder

Table 3. Decoder Performance for DOCSIS Standard

soft	par	lps	Device Speed	ALMs	M20K	f_{MAX}
6	4	4	1	20,256	118	349
			3	20,261	-	306
	12	12	1	70,395	82	250
			3	70,455	-	228
7	4	4	1	22,828	123	336
			3	22,861	-	288
	12	12	1	79,544	96	253
			3	79,747	-	220
8	4	-	1	25,195	127	332
		-	3	25,206	-	267
	12	-	1	88,390	109	248
		-	3	88,066	-	209

Table 4. Decoder Performance for NASA Standard

soft	par	full	lay	speed	ALMs	M20K	f_{MAX}
6	4	0	0	1	47,704	44	351
				3	47,723	-	298
			1	1	41,896	26	354
				3	41,935	-	312
		1	0	1	59,688	44	292
				3	59,730	-	263
			1	1	54,217	26	307
				3	54,191	-	266
	12	0	0	1	83,371	115	274
				3	83,404	-	225
			1	1	68,013	58	291
				3	68,061	-	244
		1	0	1	90,664	115	237
				3	90,778	-	206
			1	1	75,666	58	271
				3	75,565	-	212
	20	0	1	1	90,117	96	279
				3	90,082	-	243
		1		1	96,575	-	217

continued...

soft	par	full	lay	speed	ALMs	M20K	f_{MAX}
				3	96,703	-	195
7	4	0	0	1	54,812	55	324
				3	54,769	-	277
			1	1	47,746	33	342
				3	47,821	-	310
		1	0	1	66,513	55	271
				3	66,484	-	253
			1	1	60,191	33	310
				3	60,119	-	255
	12	0	0	1	96,921	134	252
				3	96,934	-	218
			1	1	77,783	67	289
				3	77,955	-	243
		1	0	1	103,427	134	208
				3	103,649	-	197
			1	1	85,297	67	252
				3	85,182		204
	20	0	1	1	103,554	112	259
				3	103,655	-	218
		1		1	109,648	-	219
				3	109,908	-	181

Table 5. Decoder Performance for WiMedia

Normal codeword and device speed grade = 3. WiMedia designs require no M20Ks

soft	par	rate	ALM	fMAX
7	2	0.89	29,576	277
		0.75	28,923	293
		0.625	29,234	276
		0.5	26,092	300
	4	0.89	45,902	253
		0.75	44,789	276
		0.625	45,736	258
		0.5	40,133	275
	8	0.89	84,145	224
		0.75	81,750	244
		0.625	84,221	224
		0.5	70,975	246

continued...

soft	par	rate	ALM	fMAX
8	2	0.89	33,179	257
		0.75	32,656	278
		0.625	33,100	259
		0.5	29,221	288
	4	0.89	51,424	255
		0.75	50,914	246
		0.625	51,921	254
		0.5	44,543	269
	8	0.89	94,644	224
		0.75	93,259	227
		0.625	94,539	221
		0.5	81,038	242

Encoder

Table 6. Encoder Performance for NASA Standard

Requires no M20Ks.

BPS	speed	ALMs	f_{MAX}
1	1	2,173	574
	3	2,175	552
4	1	5,083	512
	3	5,082	453
8	1	10,424	381
	3	10,411	344
20	1	24,525	323
	3	24,501	275

Table 7. Encoder Performance for WiMedia Standard

Normal codeword; variable coding rate.

BPS	speed	ALMs	M20Ks	fMAX
1	1	768	19	569
	3	635		496
5	1	1,854		563
	3	1,842		503
10	1	2,919		560
	3	2,441		499
15	1	4,062		522
	3	4,054		469

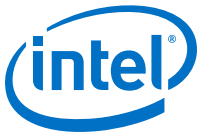


Table 8. Encoder Performance for DVB Standard

short	rate	BPS	speed	ALMs	M20K	f_{MAX}
0	1/4	1	1	12,540	1	561
			3	12,516		478
		10	1	13,245		458
			3	13,246		400
		20	1	14,184		421
			3	14,185		350
	8/9	1	1	3,869		538
			3	3,840		497
		10	1	3,869		538
			3	3,840		497
		20	1	6,774		501
			3	6,784		409
1	1/4	1	1	3,352		571
			3	3,353		527
		10	1	3,612		511
			3	3,611		493
		20	1	3,935		558
			3	3,933		476
	8/9	1	1	570		582
			3	569		570
		10	1	990		568
			3	975		524
		20	1	1,788		486
			3	1,793		490

2 Getting Started with the LDPC IP Core

2.1 Installing and Licensing Intel FPGA IP Cores

The Intel Quartus Prime software installation includes the Intel FPGA IP library. This library provides many useful IP cores for your production use without the need for an additional license. Some Intel FPGA IP cores require purchase of a separate license for production use. The Intel FPGA IP Evaluation Mode allows you to evaluate these licensed Intel FPGA IP cores in simulation and hardware, before deciding to purchase a full production IP core license. You only need to purchase a full production license for licensed Intel IP cores after you complete hardware testing and are ready to use the IP in production.

The Intel Quartus Prime software installs IP cores in the following locations by default:

Figure 1. IP Core Installation Path

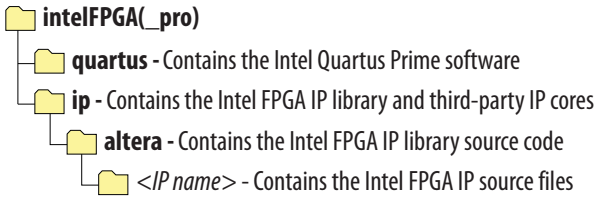


Table 9. IP Core Installation Locations

Location	Software	Platform
<drive>:\intelFPGA_pro\quartus\ip\altera	Intel Quartus Prime Pro Edition	Windows*
<drive>:\intelFPGA\quartus\ip\altera	Intel Quartus Prime Standard Edition	Windows
<home directory>:/intelFPGA_pro/quartus/ip/altera	Intel Quartus Prime Pro Edition	Linux*
<home directory>:/intelFPGA/quartus/ip/altera	Intel Quartus Prime Standard Edition	Linux

2.1.1 Intel FPGA IP Evaluation Mode

The free Intel FPGA IP Evaluation Mode allows you to evaluate licensed Intel FPGA IP cores in simulation and hardware before purchase. Intel FPGA IP Evaluation Mode supports the following evaluations without additional license:

- Simulate the behavior of a licensed Intel FPGA IP core in your system.
- Verify the functionality, size, and speed of the IP core quickly and easily.
- Generate time-limited device programming files for designs that include IP cores.
- Program a device with your IP core and verify your design in hardware.

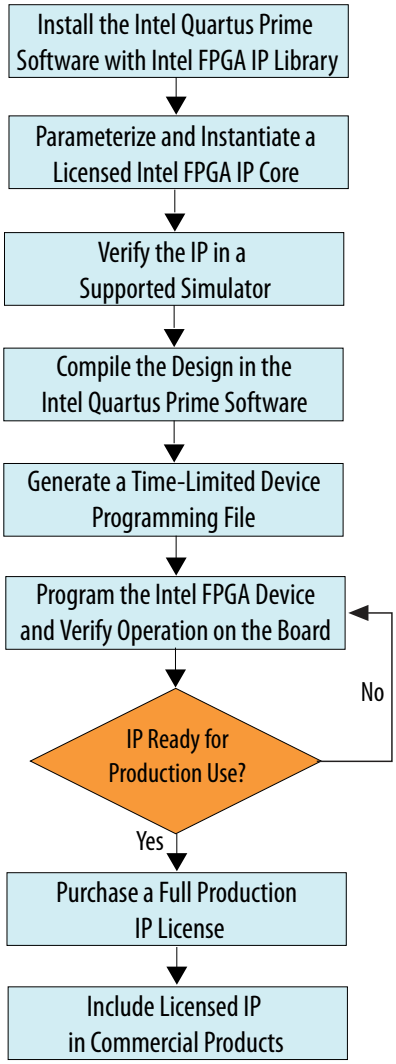
Intel FPGA IP Evaluation Mode supports the following operation modes:

- Tethered—Allows running the design containing the licensed Intel FPGA IP indefinitely with a connection between your board and the host computer. Tethered mode requires a serial joint test action group (JTAG) cable connected between the JTAG port on your board and the host computer, which is running the Intel Quartus Prime Programmer for the duration of the hardware evaluation period. The Programmer only requires a minimum installation of the Intel Quartus Prime software, and requires no Intel Quartus Prime license. The host computer controls the evaluation time by sending a periodic signal to the device via the JTAG port. If all licensed IP cores in the design support tethered mode, the evaluation time runs until any IP core evaluation expires. If all of the IP cores support unlimited evaluation time, the device does not time-out.
- Untethered—Allows running the design containing the licensed IP for a limited time. The IP core reverts to untethered mode if the device disconnects from the host computer running the Intel Quartus Prime software. The IP core also reverts to untethered mode if any other licensed IP core in the design does not support tethered mode.

When the evaluation time expires for any licensed Intel FPGA IP in the design, the design stops functioning. All IP cores that use the Intel FPGA IP Evaluation Mode time out simultaneously when any IP core in the design times out. When the evaluation time expires, you must reprogram the FPGA device before continuing hardware verification. To extend use of the IP core for production, purchase a full production license for the IP core.

You must purchase the license and generate a full production license key before you can generate an unrestricted device programming file. During Intel FPGA IP Evaluation Mode, the Compiler only generates a time-limited device programming file (<*project name*>`_time_limited.sof`) that expires at the time limit.

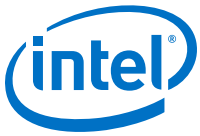
Figure 2. Intel FPGA IP Evaluation Mode Flow



Note: Refer to each IP core's user guide for parameterization steps and implementation details.

Intel licenses IP cores on a per-seat, perpetual basis. The license fee includes first-year maintenance and support. You must renew the maintenance contract to receive updates, bug fixes, and technical support beyond the first year. You must purchase a full production license for Intel FPGA IP cores that require a production license, before generating programming files that you may use for an unlimited time. During Intel FPGA IP Evaluation Mode, the Compiler only generates a time-limited device programming file (`<project name>_time_limited.sof`) that expires at the time limit. To obtain your production license keys, visit the Self-Service Licensing Center or contact your local Intel FPGA representative.

The Intel FPGA Software License Agreements govern the installation and use of licensed IP cores, the Intel Quartus Prime design software, and all unlicensed IP cores.



Related Links

- Intel Quartus Prime Licensing Site
- Intel FPGA Software Installation and Licensing

2.1.2 LDPC IP Core Intel FPGA IP Evaluation Mode Timeout Behavior

All IP cores in a device time out simultaneously when the most restrictive evaluation time is reached. If a design has more than one IP core, the time-out behavior of the other IP cores may mask the time-out behavior of a specific IP core .

For IP cores, the untethered time-out is 1 hour; the tethered time-out value is indefinite. Your design stops working after the hardware evaluation time expires. The Quartus Prime software uses Intel FPGA IP Evaluation Mode Files (`.ocp`) in your project directory to identify your use of the Intel FPGA IP Evaluation Mode evaluation program. After you activate the feature, do not delete these files..

When the evaluation time expires, for LDPC IP core encoders `out_data` goes low and `rst` goes high; for decoders `cw_out_data` goes low, `rst` goes high .

Related Links

AN 320: OpenCore Plus Evaluation of Megafunctions

2.2 IP Catalog and Parameter Editor

The IP Catalog displays the IP cores available for your project. Use the following features of the IP Catalog to locate and customize an IP core:

- Filter IP Catalog to **Show IP for active device family** or **Show IP for all device families**. If you have no project open, select the **Device Family** in IP Catalog.
- Type in the Search field to locate any full or partial IP core name in IP Catalog.
- Right-click an IP core name in IP Catalog to display details about supported devices, to open the IP core's installation folder, and for links to IP documentation.
- Click **Search for Partner IP** to access partner IP information on the web.

The parameter editor prompts you to specify an IP variation name, optional ports, and output file generation options. The parameter editor generates a top-level Intel Quartus Prime IP file (`.ip`) for an IP variation in Intel Quartus Prime Pro Edition projects.

The parameter editor generates a top-level Quartus IP file (`.qip`) for an IP variation in Intel Quartus Prime Standard Edition projects. These files represent the IP variation in the project, and store parameterization information.

Figure 3. IP Parameter Editor (Intel Quartus Prime Pro Edition)

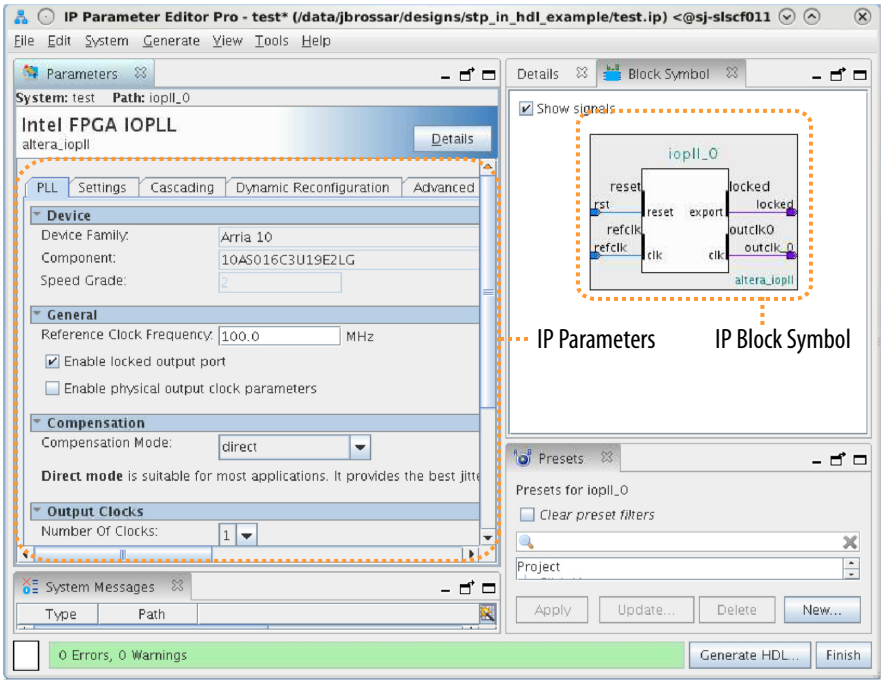
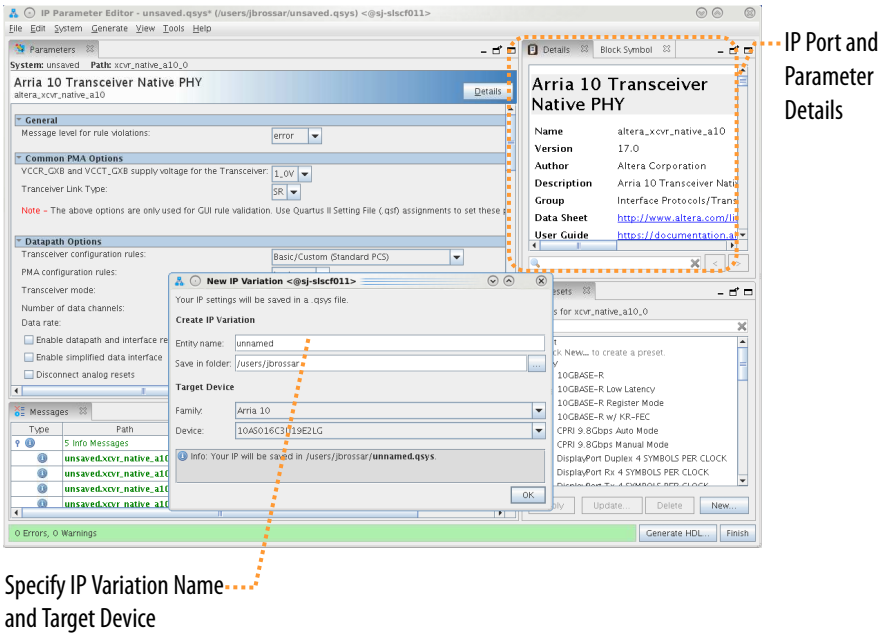


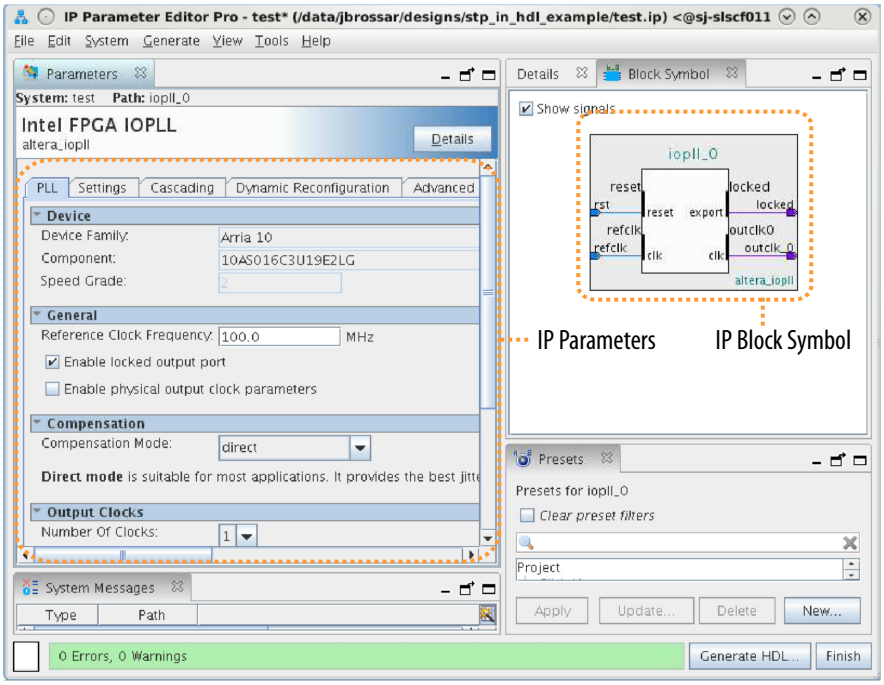
Figure 4. IP Parameter Editor (Intel Quartus Prime Standard Edition)



2.3 Generating IP Cores (Intel Quartus Prime Pro Edition)

Quickly configure Intel FPGA IP cores in the Intel Quartus Prime parameter editor. Double-click any component in the IP Catalog to launch the parameter editor. The parameter editor allows you to define a custom variation of the IP core. The parameter editor generates the IP variation synthesis and optional simulation files, and adds the `.ip` file representing the variation to your project automatically.

Figure 5. IP Parameter Editor (Intel Quartus Prime Pro Edition)



Follow these steps to locate, instantiate, and customize an IP core in the parameter editor:

1. Create or open an Intel Quartus Prime project (`.qpf`) to contain the instantiated IP variation.
2. In the IP Catalog (**Tools ➤ IP Catalog**), locate and double-click the name of the IP core to customize. To locate a specific component, type some or all of the component's name in the IP Catalog search box. The New IP Variation window appears.
3. Specify a top-level name for your custom IP variation. Do not include spaces in IP variation names or paths. The parameter editor saves the IP variation settings in a file named <*your_ip*>`.ip`. Click **OK**. The parameter editor appears.
4. Set the parameter values in the parameter editor and view the block diagram for the component. The **Parameterization Messages** tab at the bottom displays any errors in IP parameters:
   - Optionally, select preset parameter values if provided for your IP core. Presets specify initial parameter values for specific applications.
   - Specify parameters defining the IP core functionality, port configurations, and device-specific features.
   - Specify options for processing the IP core files in other EDA tools.

Note: Refer to your IP core user guide for information about specific IP core parameters.

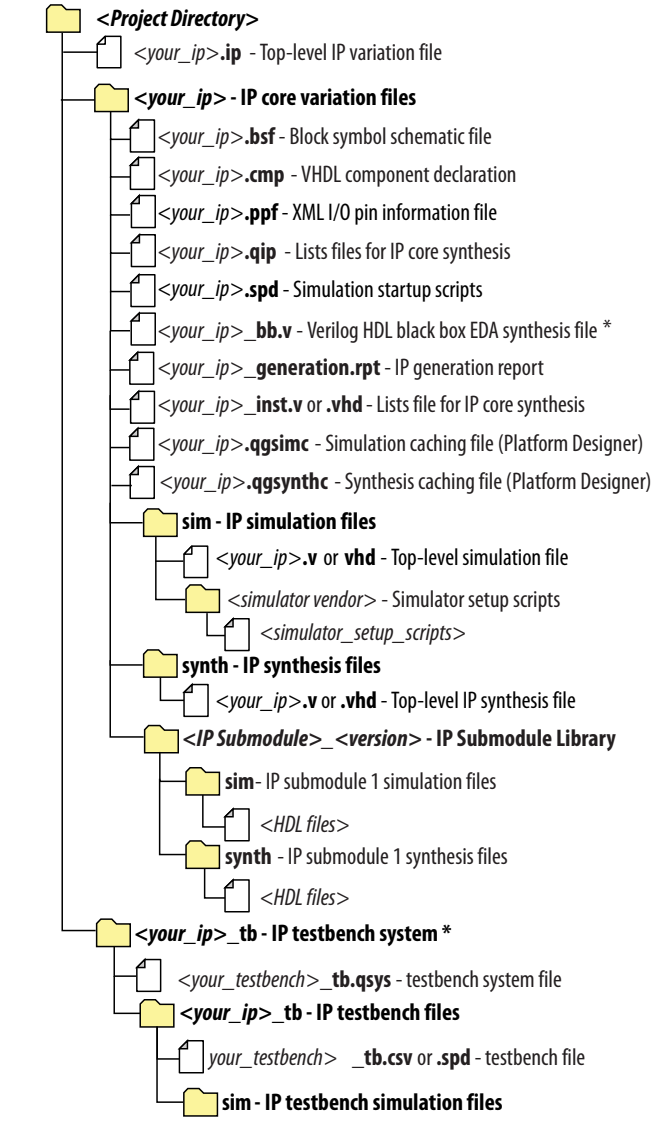
5. Click **Generate HDL**. The **Generation** dialog box appears.
6. Specify output file generation options, and then click **Generate**. The synthesis and simulation files generate according to your specifications.
7. To generate a simulation testbench, click **Generate ➤ Generate Testbench System**. Specify testbench generation options, and then click **Generate**.
8. To generate an HDL instantiation template that you can copy and paste into your text editor, click **Generate ➤ Show Instantiation Template**.
9. Click **Finish**. Click **Yes** if prompted to add files representing the IP variation to your project.
10. After generating and instantiating your IP variation, make appropriate pin assignments to connect ports.

    Note: Some IP cores generate different HDL implementations according to the IP core parameters. The underlying RTL of these IP cores contains a unique hash code that prevents module name collisions between different variations of the IP core. This unique code remains consistent, given the same IP settings and software version during IP generation. This unique code can change if you edit the IP core's parameters or upgrade the IP core version. To avoid dependency on these unique codes in your simulation environment, refer to *Generating a Combined Simulator Setup Script*.

2.3.1 IP Core Generation Output (Intel Quartus Prime Pro Edition)

The Intel Quartus Prime software generates the following output file structure for individual IP cores that are not part of a Platform Designer system.

Figure 6. Individual IP Core Generation Output (Intel Quartus Prime Pro Edition)



* If supported and enabled for your IP core variation.

Table 10. Output Files of Intel FPGA IP Generation

File Name	Description
<your_ip>.ip	Top-level IP variation file that contains the parameterization of an IP core in your project. If the IP variation is part of a Platform Designer system, the parameter editor also generates a .qsys file.
<your_ip>.cmp	The VHDL Component Declaration (.cmp) file is a text file that contains local generic and port definitions that you use in VHDL design files.
<your_ip>_generation.rpt	IP or Platform Designer generation log file. Displays a summary of the messages during IP generation.

continued...

File Name	Description
<*your_ip*>.qgsimc (Platform Designer systems only)	Simulation caching file that compares the .qsys and .ip files with the current parameterization of the Platform Designer system and IP core. This comparison determines if Platform Designer can skip regeneration of the HDL.
<*your_ip*>.qgsynth (Platform Designer systems only)	Synthesis caching file that compares the .qsys and .ip files with the current parameterization of the Platform Designer system and IP core. This comparison determines if Platform Designer can skip regeneration of the HDL.
<*your_ip*>.qip	Contains all information to integrate and compile the IP component.
<*your_ip*>.csv	Contains information about the upgrade status of the IP component.
<*your_ip*>.bsf	A symbol representation of the IP variation for use in Block Diagram Files (.bdf).
<*your_ip*>.spd	Input file that ip-make-simscript requires to generate simulation scripts. The .spd file contains a list of files you generate for simulation, along with information about memories that you initialize.
<*your_ip*>.ppf	The Pin Planner File (.ppf) stores the port and node assignments for IP components you create for use with the Pin Planner.
<*your_ip*>_bb.v	Use the Verilog blackbox (_bb.v) file as an empty module declaration for use as a blackbox.
<*your_ip*>_inst.v or _inst.vhd	HDL example instantiation template. Copy and paste the contents of this file into your HDL file to instantiate the IP variation.
<*your_ip*>.regmap	If the IP contains register information, the Intel Quartus Prime software generates the .regmap file. The .regmap file describes the register map information of master and slave interfaces. This file complements the .sopcinfo file by providing more detailed register information about the system. This file enables register display views and user customizable statistics in System Console.
<*your_ip*>.svd	Allows HPS System Debug tools to view the register maps of peripherals that connect to HPS within a Platform Designer system. During synthesis, the Intel Quartus Prime software stores the .svd files for slave interface visible to the System Console masters in the .sof file in the debug session. System Console reads this section, which Platform Designer queries for register map information. For system slaves, Platform Designer accesses the registers by name.
<*your_ip*>.v <*your_ip*>.vhd	HDL files that instantiate each submodule or child IP core for synthesis or simulation.
mentor/	Contains a msim_setup.tcl script to set up and run a ModelSim simulation.
aldec/	Contains a Riviera*-PRO script rivierapro_setup.tcl to setup and run a simulation.
/synopsys/vcs /synopsys/vcsmx	Contains a shell script vcs_setup.sh to set up and run a VCS* simulation. Contains a shell script vcsmx_setup.sh and synopsys_sim.setup file to set up and run a VCS MX* simulation.
/cadence	Contains a shell script ncsim_setup.sh and other setup files to set up and run an NCSIM simulation.
/submodules	Contains HDL files for the IP core submodule.
<*IP submodule*>/	Platform Designer generates /synth and /sim sub-directories for each IP submodule directory that Platform Designer generates.

2.4 Simulating Intel FPGA IP Cores

The Intel Quartus Prime software supports IP core RTL simulation in specific EDA simulators. IP generation creates simulation files, including the functional simulation model, any testbench (or example design), and vendor-specific simulator setup scripts for each IP core. Use the functional simulation model and any testbench or example design for simulation. IP generation output may also include scripts to compile and run any testbench. The scripts list all models or libraries you require to simulate your IP core.

The Intel Quartus Prime software provides integration with many simulators and supports multiple simulation flows, including your own scripted and custom simulation flows. Whichever flow you choose, IP core simulation involves the following steps:

1. Generate simulation model, testbench (or example design), and simulator setup script files.
2. Set up your simulator environment and any simulation scripts.
3. Compile simulation model libraries.
4. Run your simulator.

3 LDPC IP Core Specifications

This topic describes the architecture, interfaces, parameters, and signals.

3.1 LDPC IP Core Interfaces

The LDPC Avalon-ST interface supports backpressure, which is a flow control mechanism, where a sink can indicate to a source to stop sending data.

The number of symbols per beat is fixed to 1.

3.1.1 Avalon-ST Interfaces in DSP IP Cores

Avalon-ST interfaces define a standard, flexible, and modular protocol for data transfers from a source interface to a sink interface.

The input interface is an Avalon-ST sink and the output interface is an Avalon-ST source. The Avalon-ST interface supports packet transfers with packets interleaved across multiple channels.

Avalon-ST interface signals can describe traditional streaming interfaces supporting a single stream of data without knowledge of channels or packet boundaries. Such interfaces typically contain data, ready, and valid signals. Avalon-ST interfaces can also support more complex protocols for burst and packet transfers with packets interleaved across multiple channels. The Avalon-ST interface inherently synchronizes multichannel designs, which allows you to achieve efficient, time-multiplexed implementations without having to implement complex control logic.

Avalon-ST interfaces support backpressure, which is a flow control mechanism where a sink can signal to a source to stop sending data. The sink typically uses backpressure to stop the flow of data when its FIFO buffers are full or when it has congestion on its output.

Related Links

Avalon Interface Specifications

3.1.2 Clock and Reset Interfaces

The clock and reset interfaces drive or receive the clock and reset signal to synchronize the Avalon-ST interfaces and provide reset connectivity.

3.1.3 LDPC IP Core Signals

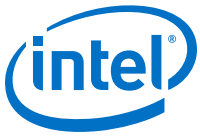


Table 11. Avalon-ST Input and Output Interface Signals

Name	Avalon-ST Type	Direction	Description
in_data[]	data	Input	Data input for each codeword. Valid only when you assert the in_valid signal. In Qsys systems, this Avalon-ST compliant data bus includes all the Avalon-ST input data signals.
in_endofpacket	eop	Input	End of packet (codeword) signal.
in_number	-	Input	Variable rate DOCSIS only. in_number is active at SOP. It is a 3-bit signal • 001 for large frame (rate=0.89) • 010 for medium frame (rate=0.85) • 100 for short frame (rate=0.75)
in_rate	-	Input	Variable rate WiMedia only. in_rate is active at SOP. It is a 4-bit width signal: • 0001 corresponds to code rate 0.8 • 0010 corresponds to code rate 0.75 • 0100 corresponds to code rate 0.625 • 1000 corresponds to code rate 0.5
in_ready	ready	Output	Data transfer ready signal to indicate that the sink is ready to accept data. The sink interface drives the in_ready signal to control the flow of data across the interface. The sink interface captures the data interface signals on the current clk rising edge.
in_startofpacket	sop	Input	Start of packet (codeword) signal.
in_valid	valid	Input	Data valid signal to indicate the validity of the data signals. When you assert the in_valid signal, the Avalon-ST data interface signals are valid. When you deassert the in_valid signal, the Avalon-ST data interface signals are invalid and must be disregarded. You can assert the in_valid signal whenever data is available. However, the sink only captures the data from the source when the IP core asserts the in_ready signal.
out_data	data	Output	The out_data signal contains decoded output when the IP core asserts the out_valid signal. The corrected symbols are in the same order that they are entered.
out_endofpacket	eop	Output	End of packet (codeword) signal. This signal indicates the packet boundaries on the in_data[] bus. When the IP core drives this signal high, it indicates that the end of packet is present on the in_data[] bus. The IP core asserts this signal on the last transfer of every packet.
out_startofpacket	sop	Output	Start of packet (codeword) signal. This signal indicates the codeword boundaries on the in_data[] bus. When the IP core drives this signal high, it indicates that the start of packet is present on the in_data[] bus. The IP core asserts this signal on the first transfer of every codeword.
out_ready	ready	Input	Data transfer ready signal to indicate that the downstream module is ready to accept data. The source provides new data (if available) when you assert the out_ready signal and stops providing new data when you deassert the out_ready signal. If the source is unable to provide new data, it deasserts out_valid for one or more clock cycles until it is prepared to drive valid data interface signals.
out_valid	valid	Output	Data valid signal. The IP core asserts the out_valid signal high, whenever there is a valid output on out_data ; the IP core deasserts the signal when there is no valid output on out_data .

3.1.3.1 LDPC IP Core Backpressure

The LDPC IP core allows you to use backpressure.

When `out_ready` goes low, the FIFO buffer stores the data produced by the encoder. If the FIFO buffer is almost full, it sends a signal that sets `in_ready` to low and stops the flow of input data from the source. At that point if `out_ready` goes high once again, the FIFO buffer outputs the stored data until it is almost empty. Then it sets `in_ready` to high to produce more data.

3.2 LDPC IP Core Parameters

Table 12. General Parameters

Parameter	Value	Description
Standard	DVB-S2 NASA GSFC-STD-9100 WiMedia 1.5 DOCSIS 3.1 WiFi 802.11n	Type of LDPC code.
LDPC module	Encoder or decoder	NASA GSFC-STD-9100, WiMedia 1.5, DOCSIS 3.1, and Wifi 802.11n only.
Codeword length	Encoder DVB-S2: 16200, 64800 Decoder: • WiMedia 1.5: 1200, 1320 • NASA: 8160, 8176	Number of bits per codeword.
Coding rate	DVB:S2: 1/4 1/3 2/5 1/2 3/5 2/3 3/4 4/5 5/6 8/9 DOCSIS 3.1: 75% 85% 89% NASA GSFC-STD-9100: 7/8 WiMedia 1.5: 0.5, 0.625, 0.75, 0.8	-
Number of bits per input symbol	1 to 30	Encoder only.
Number of LLRs per input symbol	2 to 40 (NASA, DOCSIS, and WiMedia)	The number of LLRs you feed in parallel to the decoder
Number of softbits per input LLR	2 to 16	Decoder only. The width of the input LLR. Assume that the number of bits per output symbol is equal the number of LLR per input symbol

Table 13. Decoder Options

Parameter	Value	Description
Number of decoding iterations	1 to 100	The maximum number of decoding iterations. A decoding iteration corresponds to the decoding of all the rows of the parity-check matrix. The decoder stops decoding once it finds a valid codeword.
Parallelism	1 to 100	The number of rows processed in parallel during the decoding.
Width of the decoder variables	4 or 16	The width of the input LLR. The decoder can represent the LLR it processes by a much larger number of softbits than the input LLRs.

continued...

Parameter	Value	Description
MSA attenuation factor	1, 0.875, 0.75, 0.625, 0.5, 0.375, 0.25	The design applies a constant attenuation factor to the output of the min-sum algorithm (MSA) processing to ease convergence. This factor is a sum of inverse of power of 2.
Full-streaming architecture	No or yes	With full streaming architecture, the decoder can accept data while decoding if the level of parallelization is sufficient. NASA only.
Layered decoding	No or yes	NASA GSFC-STD-9100 decoder only
Output parity check bits	No or yes	Decoder provides output parity check bits or information bits only.

4 Document Revision History

Table 14. LDPC IP Core User Guide Revision History

Date	Version	Changes
2017.11.06	17.1	• Added support for Intel Cyclone 10 GX devices • Added support for MATLAB models • Updated parameters
2016.05.02	16.0	Changed features to support NASA encoder
2015.11.01	15.1	• Changed device support • Added performance and resource utilization data • Added `in_rate` and `in_number` signals • Changed parameter options
December 2014	2014.12.01	Initial release.

A LDPC IP Core Document Archive

If an IP core version is not listed, the user guide for the previous IP core version applies.

IP Core Version	User Guide
16.0	LDPC IP Core User Guide
15.1	LDPC IP Core User Guide
14.1	LDPC IP Core User Guide