IPCTEK

5G NR HIGH RATE LDPC EXAMPLE DESIGN

Example design user guide

UG006

Version 1.0

May 21, 2022

IPCTEK

Table of Contents

IPCTEK

Introduction

The purpose of this example design is to demonstrate the performance of the IPCTEK's 5G NR High Rate LDPC Encoder and Decoder FPGA IP cores. The IP cores are wrapped in an AXI4-Lite wrapper for easy integration in systems with an AXI4 bus (Microblaze, Zynq etc.). Simple C drivers used to parameterize the cores are also provided.

Principle characteristics of the LDPC Decoder are given in the table below.

Parameter	Value
Message passing method	Layered decoding
Decoding algorithm	Normalized Min-Sum
Normalized factor	0.75
Parallelism degree	384
Channel LLR quantization bits	5
Check node quantization bits	6
Variable node (APP) quantization bits	8
Input LLR data bus	384 LLR values per clock
Output decoded bits data bus	384 bits per clock

Table 1 - IP core's parameters

Example design source code download

In order to compile the example design we need to download the Vivado Hdl firmware, the SDK bare-metal application and the Qt test bench source code. All the design source code and an evaluation netlist can be requested by sending an e-mail to contact@ipctek.net

- **Vivado Hdl firmware**: this repository contains necessary source code and scripts to generate the FPGA firmware Vivado project.
- **SDK bare-metal application project**: the example design is running on a Microblaze-based subsystem. The source code is in **drivers/** folder.
- **Qt-based graphical interface (optional)**: The example design can be evaluated with serial UART port to pass commands and receive responses. However, for the sake of result representation and ease of use, an open-source Qt-based graphical interface is given in **Qt/** folder. Via this interface, the BER/FER simulation can be launched and results can be displayed and analyzed with the IPCTEK's graphical scope.

Firmware generation

In this section we detail the step-by-step process in order to generate the FPGA firmware used in the example design.

Firmware synoptic

The FPGA firmware synoptic is illustrated in the figure below. In the design, we implement a 5G NR High Rate LDPC encoder and a decoder which will be running on one of the Xilinx evaluation boards.

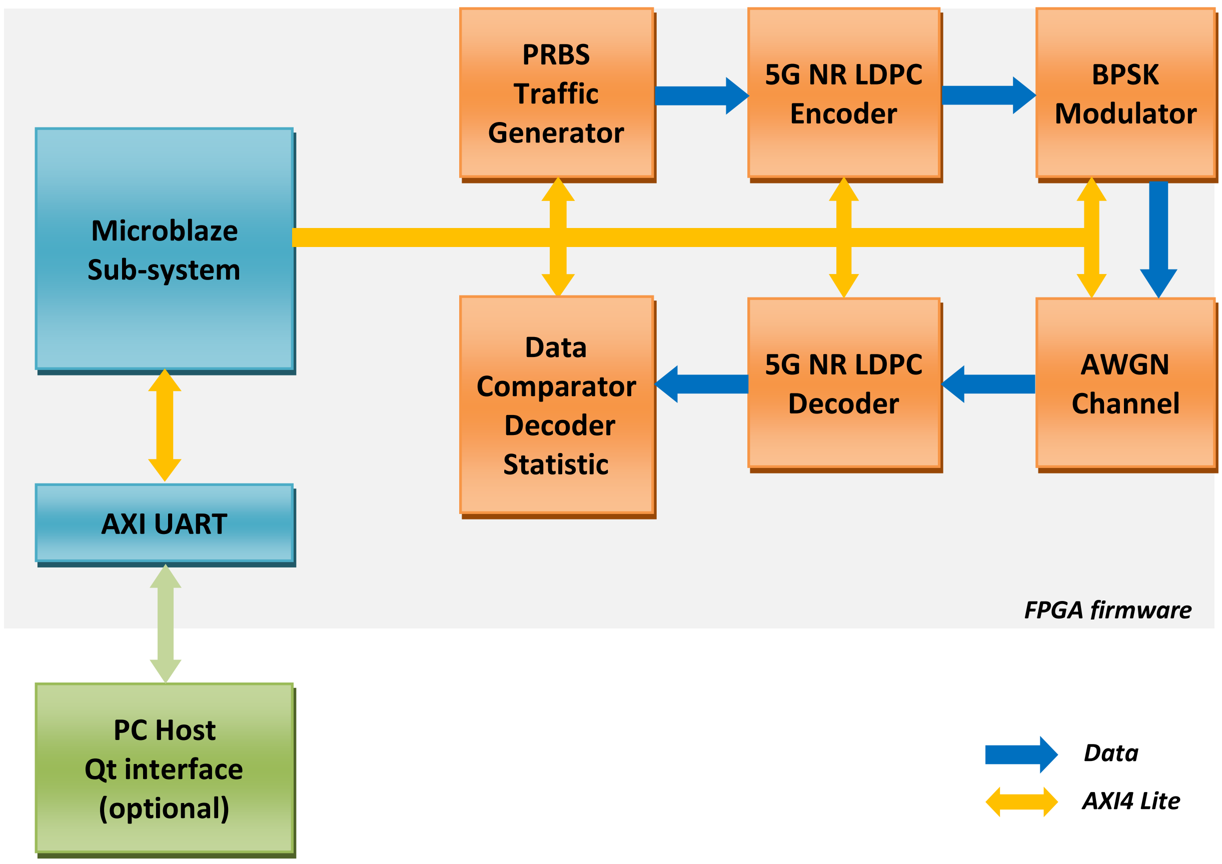


Figure 1 - FPGA firmware synoptic

The FPGA firmware includes following main components.

- **Traffic Generator**: this component can be considered as the transmitter in the simulation chain. By means of a feedback shift register, it transmits pseudo-random data (information bits) to the LDPC encoder.
- **LDPC Encoder**: this component encodes the information bits then sends the encoded bits to a BPSK modulator.

- **BPSK Modulator**: this component converts the transmitted bits into BPSK symbols, which are represented by a fixed point number Q(16, 11) (5 integer bits and 11 fractional bits).
- **AWGN Channel**: the BPSK symbols are then passed to an AWGN channel where a Gaussian noise is added. This module also calculates the log-likelihood ratio value of the noisy symbols before passing them to the LDPC Decoder.
- **LDPC Decoder:** the module carries out iterative decoding process. Decoding parameters such as the early stopping option and the maximum number of iterations can be programmed via the AXI4-Lite interface before launching the simulation. The decoded bits are then passed to the Statistic Module in order to calculate the BER/FER performance.
- **LDPC Statistic:** this component has the same PRBS generator as the Traffic Generator module. By comparing the Tx and the Rx data, the BER/FER values can be calculated.

Firmware generation steps

Library generation

Before generating the example project, different necessary IPs need to be compiled.

--- *Tip* --

In Windows, open the cmd console then use subst command to create a virtual Disk pointing to the Vivado Hdl root folder in order to avoid Windows's maximum path length limitation error. For example, to create a virtual disk named T, tap the following in the cmd console.

```
subst T: <path_to_the_root_folder>
```

Open Vivado, cd into the /library folder, use the Vivado tcl console

```
cd T:/library
```

Run the script buildLib.tcl to compile the library, use the Vivado tcl console

```
source ./buildLib.tcl
```

When the library compilation is finished, we proceed to generate the example project.

Project generation

cd into the project folder /projects/ ldpc_5g_nr_over_boadName (for example ldpc_5g_nr_over_kcu105), use the Vivado tcl console

```
cd T:/projects/ldpc_5g_nr_over_kcu105
```

Run the script system_project.tcl to generate the project, use the Vivado tcl console

```
source ./system_project.tcl
```

After the script is finished loading the board design, run the synthesis then the implementation and generate the bitstream. Use “Performance Explore” for the Implementation strategy for easing the timing closure. These processes take about one hour to finish depending on the host PC.

Export the bitstream then launch the SDK. We proceed to create an SDK application project.

SDK application project

Create an application project based on the hardware that we have just exported from the Vivado project.

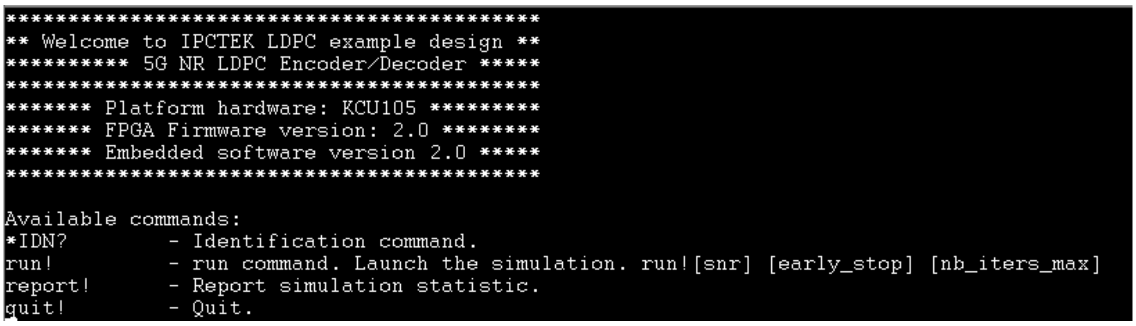
The SDK application source code for the example design is located at /drivers/ldpc_5g_nr/ldpc_5g_nr_over_kcu105/src folder. Import this folder into your SDK project.

--- Note--
Reconfigure the Linker Script to increase the stack size and the heap size to 8 KB for a stable application.

UART firmware test menu

If the macro IHM is not defined in the main.c file, we are in the console mode. Program the board, plug the USB/UART cable into the evaluation board, open a console application (e.g. Tera Term) then run the application. The UART baud rate is 115200 Hz. Remember to check the "***local echo***" option in Tera Term in order to see user's input commands.

Below is the screen after launching the SDK application.



```
*******************************************
** Welcome to IPCTEK LDPC example design **
********** 5G NR LDPC Encoder/Decoder *****
*******************************************
******* Platform hardware: KCU105 *********
******* FPGA Firmware version: 2.0 ********
******* Embedded software version 2.0 *****
*******************************************

Available commands:
*IDN?       - Identification command.
run!        - run command. Launch the simulation. run![snr] [early_stop] [nb_iters_max]
report!     - Report simulation statistic.
quit!       - Quit.
```

Figure 2 - Firmware UART console

Available commands for test are:

- *IDN? command: use this command to ask for the system identification.
- run! command: use this command to launch the simulation for one Eb/N0 value.
- report! command: use this command to ask for the statistic information of the ongoing simulation.
- quit! command: use this command to quit the application.

--- Tip ---
Don't forget the question mark or the exclamation mark at the end of each command. The run! command has several arguments. And there is no space between the first argument and the exclamation mark.
Do not define the macro IHM in the main.c file in the console mode.

Test examples

Run the simulation

In the UART console, use the run! command to launch the simulation. The command syntax is as follow

```
run![Eb/N0] [Early_stop] [Nb_iters]
```

For example, we will launch a simulation where Eb/N0 is equal to 4.5 dB, early stopping is ON and the maximum number of iterations is equal to 10. In this case we input the following command

```
run!4.5 1 10
```

If we want to launch another simulation with the early stopping disabled, we use

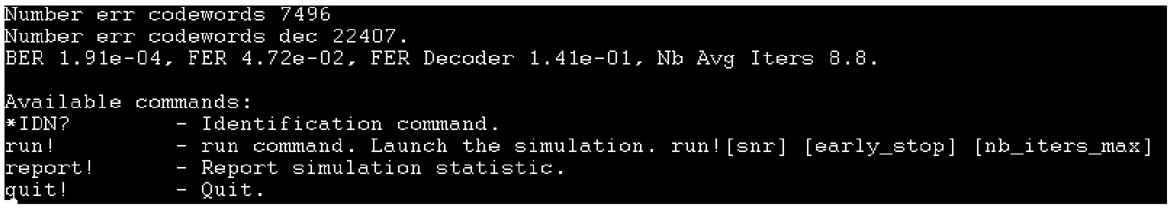
```
run!4.5 0 10
```

Read statistic information

Use the report! command to show the simulation statistic report. Input the following command in the console

```
report!
```

The system response is illustrated in the figure below



```
Number err codewords 7496
Number err codewords dec 22407.
BER 1.91e-04, FER 4.72e-02, FER Decoder 1.41e-01, Nb Avg Iters 8.8.

Available commands:
*IDN?         - Identification command.
run!          - run command. Launch the simulation. run![snr] [early_stop] [nb_iters_max]
report!       - Report simulation statistic.
quit!         - Quit.
```

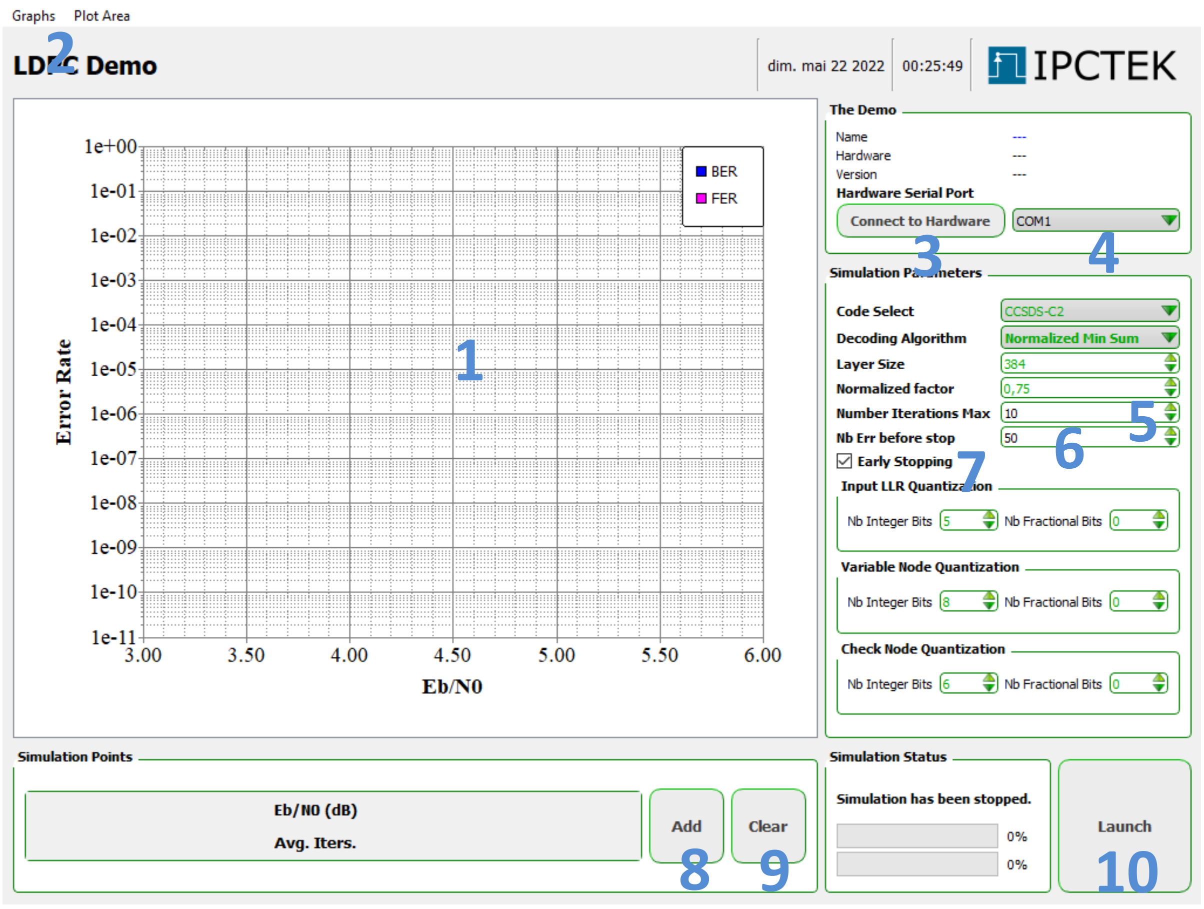
Figure 3 – Simulation statistic report. UART screen

Qt graphical interface

The Qt graphical interface use the same UART commands as described in the above section. However, the interface has a scope to display the BER/FER performance and the statistical information for better result analysis.

---*Note* ---
In the SDK project, do not forget to define the IHM macro (#define IHM) in the main.c file when used with the Qt interface.

The Qt interface is shown in the figure below.



Name ---
Hardware ---
Version ---
Hardware Serial Port: COM1

Code Select: CCSDS-C2
Decoding Algorithm: Normalized Min Sum
Layer Size: 384
Normalized factor: 0,75
Number Iterations Max: 10
Nb Err before stop: 50
☑ Early Stopping

Input LLR Quantization: Nb Integer Bits 5, Nb Fractional Bits 0
Variable Node Quantization: Nb Integer Bits 8, Nb Fractional Bits 0
Check Node Quantization: Nb Integer Bits 6, Nb Fractional Bits 0

Eb/N0 (dB)
Avg. Iters.

Simulation has been stopped.
0%
0%

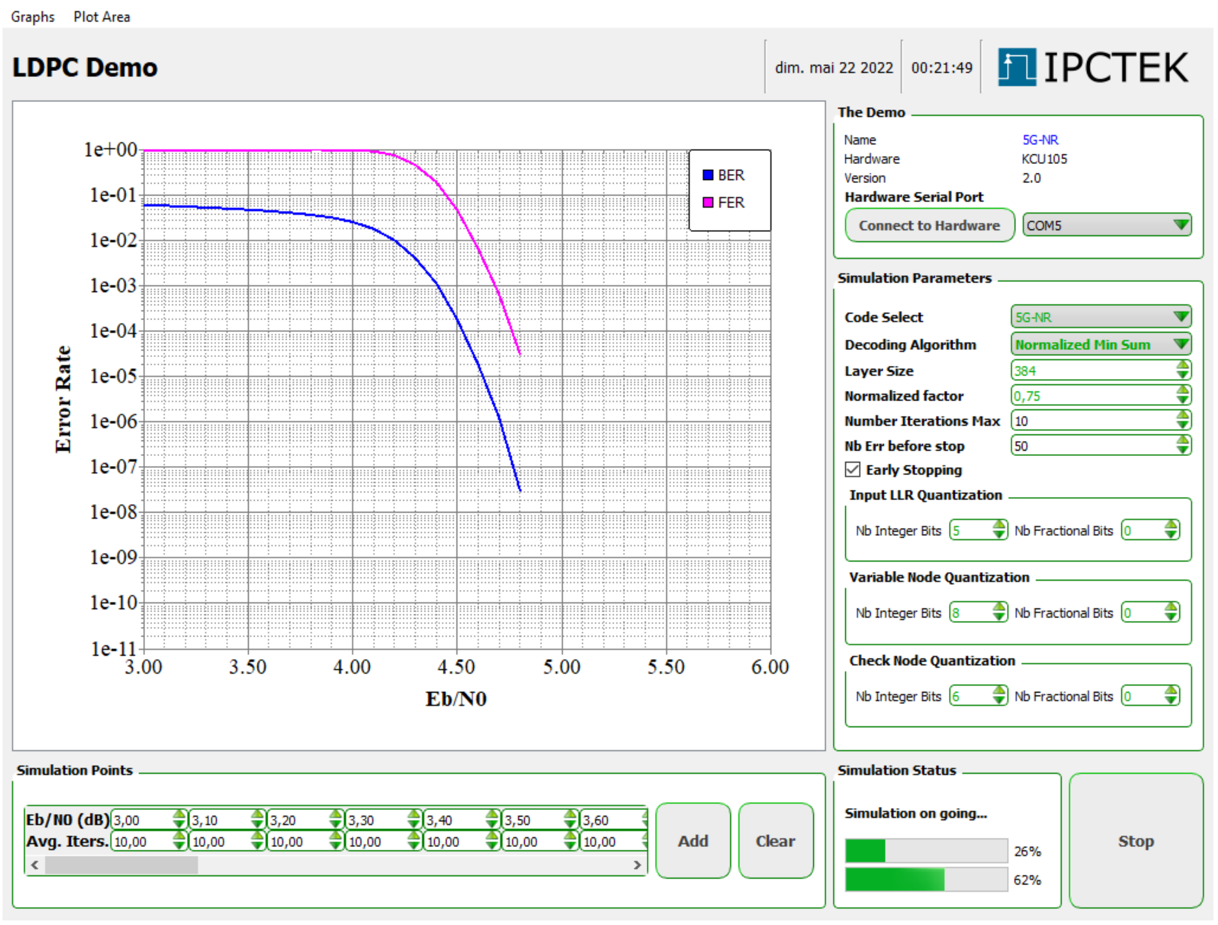
Figure 4 - Qt graphical interface

1. Plot area where FER and BER curves are displayed.
2. Plot area menu, used for data manipulation.
3. Connect to the FPGA firmware (Microblaze system via UART)
4. COM Ports select, remember to select the right port.
5. Maximum number of iterations parameter.
6. Maximum number of wrong codewords before passing to the next simulation point.

7. Early stopping option.
8. Add a new simulation point. Change the Eb/N0 value if necessary.
9. Clear all the simulation points.
10. Simulation Start/Stop button.

All parameters displayed in green color are read only. These are presented for reminding the key parameters in the simulation system. Use the hardware combo box to select the correct COM port then click on the **Connect to Hardware** button to connect to the Microblaze system. Upon successful connection, the identification command will be automatically sent to identify the simulation system.

Use the **Add** button to add simulation points then change the Eb/N0 values if necessary. Input the simulation parameters such as the maximum number of iterations and the early stopping option then click on the **Launch** button to begin the simulation. The users should find the simulation running as illustrated in Figure 5.



The Demo

Name	5G-NR
Hardware	KCU105
Version	2.0
Hardware Serial Port	COM5

Simulation Parameters

Code Select	5G-NR
Decoding Algorithm	Normalized Min Sum
Layer Size	384
Normalized factor	0,75
Number Iterations Max	10
Nb Err before stop	50
Early Stopping	✓

	Nb Integer Bits	Nb Fractional Bits
Input LLR Quantization	5	0
Variable Node Quantization	8	0
Check Node Quantization	6	0

Simulation Points

Eb/N0 (dB)	3,00	3,10	3,20	3,30	3,40	3,50	3,60
Avg. Iters.	10,00	10,00	10,00	10,00	10,00	10,00	10,00

Simulation Status

Simulation on going...

26%

62%

Figure 5 - Qt graphical interface. Simulation results.

Use the mouse wheel to zoom in and out on the plot area. Click-drag-release method can also be used to zoom into an area. Use the **Graphs** and **Plot Area** menu in order to save a curve's data into a csv file for use on other software. A csv data file, which is saved earlier, can also be loaded into the plot area for results comparison. The plot area can also be saved into a png file for documentation, printing or presentation etc.

End Of Document.