

Article

Energy-Efficient Raptor-like LDPC Coding Scheme Design and Implementation for IoT Communication Systems

Jakub Hyla ¹ and Wojciech Sułek ^{2,*} ¹ TKH Technology Poland Sp. z o.o., 64-100 Leszno, Poland; j.hyla@tkhtechnology.com² Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology, 44-100 Gliwice, Poland

* Correspondence: wojciech.sulek@polsl.pl

Abstract: The large number of inexpensive and energy-efficient terminals in IoT systems is one of the emerging elements of the recent landscape of information and communication technologies. IoT nodes are usually embedded systems with limited processing power devices and strict requirements on energy consumption. In this paper, we consider the design and implementation of a part of the IoT communication uplink stack, namely the error correction coding scheme, for energy-efficient operation. We examine how an efficient rate-adaptive coding scheme, namely the Raptor-like (RL) quasi-cyclic (QC) subclass of low-density parity check (LDPC) codes, can be applied. We present an encoding algorithm designed for an embedded CPU; a respective QC-RL-LDPC decoding scheme, based on typical LDPC iterative decoding; and a combined design procedure for construction of the QC-RL-LDPC parity check matrices. Next, we conduct experiments to explore the time intervals for implemented encoding and the goodput of the coding system with incremental redundancy. We provide the statistical results by combining the measured energy consumption of the encoder and the simulated radio transmitter energy consumption. We demonstrate the comparison of normalized energy consumption statistics with the fixed-rate LDPC coding case. As conclusion, we note that under an unknown channel corruption level, the short block QC-RL-LDPC code implemented in the CPU can achieve higher energy efficiency per information bit compared to a fixed-rate LDPC code. Future work will include an extension of the work to nonbinary QC-RL-LDPC coding design.

Keywords: error correction; ECC; low-density parity check codes; LDPC; QC-LDPC; IoT systems



Citation: Hyla, J.; Sułek, W.

Energy-Efficient Raptor-like LDPC Coding Scheme Design and Implementation for IoT

Communication Systems. *Energies*

2023, 16, 4697. <https://doi.org/10.3390/en16124697>

Academic Editors: Daniele D. Giusto and Matteo Anedda

Received: 30 April 2023

Revised: 7 June 2023

Accepted: 9 June 2023

Published: 14 June 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The Internet of Things, or IoT, is typically perceived as a system of a large number of interrelated computing devices, often constituting a kind of cyberphysical system. It is often made up of elements (IoT nodes) that are inexpensive terminals, with an important feature of optimized energy efficiency of operation [1]. The general concept of the IoT includes many possibilities for designing the hardware and software of the system. There is no single unified IoT architecture, and there is no common definition of communication protocols for IoT parts. While developers use existing technologies to build the IoT, research groups are still working to adapt protocols and improve communication quality and energy efficiency [1,2].

The IoT nodes that make up the bottom level of the system are typically battery-powered devices based on embedded low-power processors (microcontrollers) with limited processing capabilities. When the system requires a wireless communication protocol, the energy efficiency of such embedded devices could be highly dependent on the energy efficiency of the data communication protocol. Since the messages sent from/to constrained IoT nodes are typically relatively short, the important factor is also the activity time of the CPU and radio modem, in relation to the inactivity time (when the device is in sleep mode). Due to the limited resources of the processing units of the IoT nodes,

efficient communication protocols are desired to enable communication with reasonable energy efficiency.

The effectiveness and reliability of communication depend heavily on the error control coding (ECC) mechanism used. Numerous effective coding schemes, such as LDPC coding [3], turbo coding [4] and polar coding [5], have been developed in recent years. One problem with the implementation of these schemes is that they require computationally demanding iterative algorithms for decoding. These coding schemes are typically utilized in complex communication protocols such as WiFi, WiMAX and 5G, with high-performance computing hardware required for implementation.

In the case of a time-varying communication channel, additional improvement in the uplink transmission efficiency can be obtained if a properly designed rate-adaptive or rateless system is employed, because of the two limitations of the traditional fixed-rate coding, inherently designed for a specific channel noise level (or SNR, signal-to-noise ratio). First, if the SNR is significantly higher than the level that the fixed-rate code has been designed for, then the rate-adaptive code can correct error with higher rate, which means a lower number of coded bits must be encoded and delivered. Second, if, on the other hand, the SNR is significantly lower than expected, then the fixed-rate code fails to provide sufficient redundancy to combat channel corruptions, while the rate-adaptive code can adapt and provide such additional redundancy.

Rateless codes are a class of ECC that can generate an unlimited number of coded bits given a finite number of information bits. It is the core of Hybrid ARQ (HARQ) protocols, which allows for efficient transmission of signals over a channel that changes over time. The system's transmitter first sends the high-rate codeword of the necessary size over the channel. If the receiver cannot decode the received noisy codeword, the transmitter then sends repeated or additional parity symbols to the receiver to allow the receiver to attempt another round of decoding, but now with a higher redundancy level. The process continues until the decoder successfully decodes the received codewords. HARQ can be classified into two main types: type-I with chase-combining (CC) and type-II with incremental redundancy (IR). The type-II IR provides superior throughput performance to CC-HARQ, due to the retransmission of only incremental parity bits, when required [6,7]; therefore, we consider IR HARQ in this research.

The important and well-investigated case of rateless codes is Raptor coding [8]. A Raptor code is a concatenation of a high-rate LDPC outer code and a rateless Luby transform (LT) inner code. Raptor codes are shown to achieve the capacity of the binary erasure channel (BEC), while research to design them for the additive white Gaussian noise (AWGN) channel is already quite advanced [9–11]. While LT codes and Raptor codes share some similarities with LDPC codes, more closely related to LDPC are protograph-based Raptor-Like (PBRL) LDPC codes [12,13]. Unlike the Raptor codes, in the PBRL scheme, a high-rate LDPC codeword is transmitted directly as the initial block, and additional parity symbols are sent only if necessary. PBRL-LDPC codes are well suited for low-complexity encoder implementation, because the well-known LDPC and QC-LDPC efficient encoding schemes can be adapted [14,15].

The general rate-adaptive communication system model that we consider in our research is presented in Figure 1. We investigate a system based on a low-complexity embedded device that acts as an IoT node and can function in a variety of noisy environments, regardless of the communication channel. The communication scheme automatically adapts to the channel; hence, the system can work properly in a wide range of noisy environments. The developed system is classified as a type II hybrid automatic repeat request (HARQ) [6], which involves an initial transmission of a length- K block, followed by transmissions of shorter length- P chunks. The receiver combines these chunks in a decoding scheme, which signalizes the successful decoding by sending positive acknowledge (ACK), after receiving as many chunks as are needed for correct decoding. The coding scheme that we employ is PBRL-LDPC, constrained to quasi-cyclic (QC) submatrices, which we refer to as QC Raptor-like (RL) coding (QC-RL-LDPC).

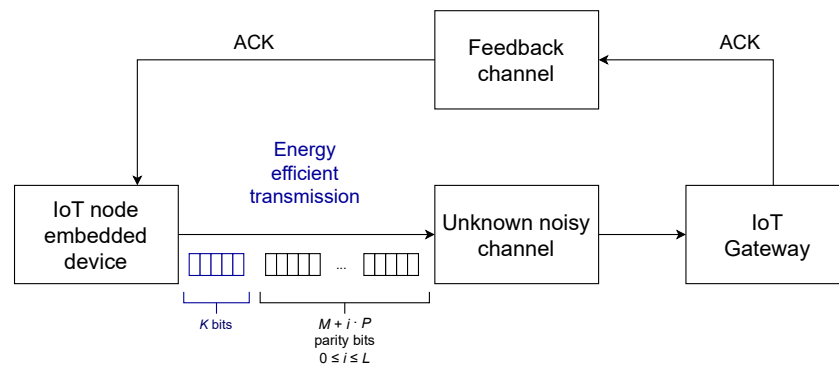


Figure 1. Considered model of the IoT uplink communication with incremental redundancy ECC scheme.

Since the gateway in Figure 1 is assumed to possess significantly more computational resources, it is possible to consider an ECC scheme with high decoding complexity for data blocks sent in an uplink direction. Therefore, we claim RL-LDPC coding to be a good choice due to its great correction performance, which can potentially also reduce the transmission power needed, and in consequence, enhance the transmitter energy efficiency. However, in the proposed system, the QC-RL-LDPC encoder must still be implemented with an embedded CPU in an IoT node.

In this context, the research presented in this paper is initiated. Although there have been many research efforts in the implementation of LDPC and Raptor-like encoding and decoding, most of them involve high-throughput FPGA or ASIC hardware implementations, while CPU codec investigations are missing. However, we show that with some effort, the encoding side can be designed in a way that also enables implementation in low-power processors with limited processing capabilities. Moreover, for the system designed according to the outline in Figure 1, the energy efficiency of the IoT node operation is very important. We note that the energy efficiency of the node radio transmitter can be enhanced by using a modern and well-performing ECC, such as the LDPC coding scheme, and additional enhancement in the case of unknown channel corruption can be achieved by a rate-adaptive scheme devoted from LDPC.

Therefore, this paper concerns the design of the entire system, including the construction of flexible QC-RL-LDPC codes and the implementation of the encoding scheme in the IoT node of Figure 1, as well as the experimental investigation of the uplink transmission system performance and the energy efficiency, especially by comparing the incremental redundancy system with the standard fixed-rate coding. This comparison is made in terms of energy efficiency of message transmissions, which is dependent on both the encoding energy consumption and the transmitter energy consumption, which are experimentally assessed in this research.

Our focus in energy efficiency assessment is on the IoT node (transmitter) procedures that have been formulated and implemented in a microcontroller CPU device. Other details of the communication protocol are abstracted as a typical digital modulation scheme and noisy communication channels; however, for complete system description, we also provide the decoding algorithm, based on modified message-passing flow. The detailed particular contributions of this paper can be listed as follows.

1. With regard to implementation, we show how the QC-RL-LDPC encoding can be efficiently implemented in a CPU device, by decomposing the encoding into simple cyclic shifts and GF(2) addition operations, in direct correspondence to the CPU program instructions.
2. We verify the designed CPU implementation of the encoder, with several QC-RL-LDPC matrix representations, measuring the encoding time and providing measurement results.

3. We provide a concise procedure for a QC-RL-LDPC binary code design, which is based on the PEG construction of base graph [16] of the HR code; an IR part optimization with reciprocal channel approximation, as in [13]; and then combined lifting of protographs with a proprietary computer search algorithm.
4. For the receiver side of the system, we provide an effective decoding scheme, based on LDPC belief propagation (BP) decoding, with incremental redundancy.
5. We perform a number of experiments of the entire system operation, providing the results in the form of simulated transmission goodput, which is defined as the ratio of the number of information bits transmitted to the total number of modulation symbols utilized in the IR, required for successful decoding. Several different designed QC-RL-LDPC codes with different code lengths are experimentally tested. Moreover, we also simulate rate-adaptive coding based on state-of-the-art 5G codes [17,18] and compare them with our designed codes.
6. Taking into account the timing results and system simulation goodput results, we assess the energy consumption of the encoder plus radio transmitter in a typical IoT wireless transmission scenario, and provide the results in the form of energy consumption normalized per transmitted information bit, which allows for an energy efficiency comparison with a fixed-rate coding system, as well as with a 5G coding system. The experimental results show the energy efficiency gain that can be achieved in comparison to fixed-rate coding.

2. Research Methods

The QC-RL-LDPC encoder is implemented in low-power microcontroller unit from the STM32 family, as will be presented in Sections 5 and 8.1. All the remaining parts of the communication model considered, presented in Figure 1, are implemented as software models developed in the Matlab environment. The research methodology is outlined in Figure 2.

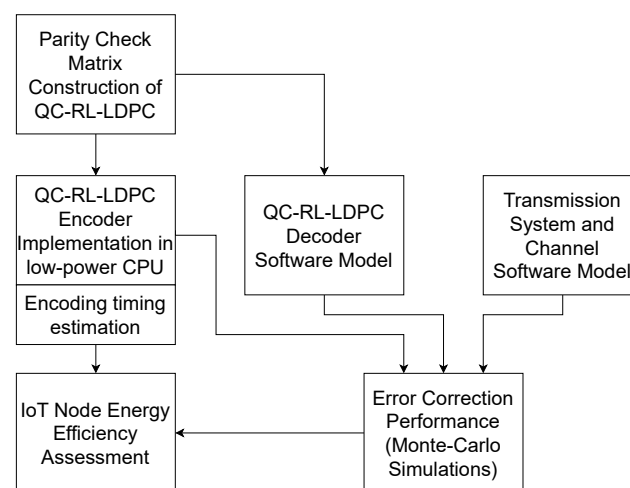


Figure 2. Outline of research methods.

The parity check matrix construction is a flexible algorithm that enables construction of QC-RL-LDPC code with any code length, base rate, submatrix size and chunk size, as is presented in Section 7. The developed decoding algorithm is presented in Section 6, and in general, it can be used in software or hardware implementations; however, for this research, the software model created in Matlab is used. Similarly, the transmission system with QAM/QPSK modulation and communication channel are modeled with software.

The models are then used in Monte-Carlo simulations that provide, as a result, the estimated error correction performance of the system, in the function of channel corruption level represented by signal-to-noise ratio (SNR). The simulation results are provided in Section 8.3. The second part of the experimental design is the encoding timing estimation, which is conducted by injection into the encoder source code, a time measurement unit based on one of the hardware timers in the STM32 microcontroller. The timer is executed for as long as the encoding and the measurement are read by an interface with PC computer and then converted into time units. The results of these experiments are presented in Section 8.1.

Finally, both the system performance results and the encoding timing results are used to assess energy consumption of the coding and transmission module in an IoT node, and the results are provided in Section 8.4.

3. LDPC Coding—Preliminaries

The low-density parity check coding was initially proposed by Robert Gallager in [19], and later recalled by MacKay [3,20], and demonstrated that irregular LDPC codes can asymptotically approach the Shannon limit [21] of lossless transmission. In the last decade, LDPC codes have become very popular as a powerful error correction coding method with the possibility of algorithm parallelization that enables the development of hardware codecs with optimized algorithms and very high throughput [22–25], typically for high-speed industrial communication standards [26,27].

The particular LDPC code is defined by its parity check matrix (PCM) $\mathbf{H}$, which defines the specific parity checks involving the transmitted message bits. The important feature of PCM, which is its sparseness, allows LDPC codes to be iteratively decoded using low-complexity message passing algorithms [3]. Moreover, low-complexity encoding algorithms have also been well recognized [14]. Recent research in the field also involves nonbinary coding algorithms, as well as code design and implementation [28–30].

3.1. LDPC Codes

The binary parity check matrix $\mathbf{H}$ of a binary (N, K) LDPC code has a size $M \times N$, where N is the length of the code vector (encoded message), $K < N$ is the length of an information vector, and $M = N - K$ represents the number of parity bits in the code vector. The measure of coding redundancy is the code rate, defined as $R = K/N$. In the systematic encoding process, M parity bits are added to the information vector $\mathbf{u} = \{u_1, u_2, \dots, u_K\}$, forming the code vector $\mathbf{c} = \{c_1, c_2, \dots, c_N\}$. All vectors are binary, and it is convenient to consider them to be defined over the Galois finite field $\text{GF}(2)$.

In the decoder, the data received $\mathbf{y}$, which is a vector of length N , are corrupted by channel imperfections. If—and only if—the hard decisions made from $\mathbf{y}$ meet the parity check equation, it will be recognized as a correct, (error-free) vector $\hat{\mathbf{c}}$. The parity check equation is expressed as $\mathbf{H}\hat{\mathbf{c}}^T = \mathbf{0}_{M \times 1}$, where the matrix operation is in $\text{GF}(2)$ arithmetic. If the equation is not satisfied, the iterative decoding algorithm is started that can correct errors, with soft decisions as the algorithm input, which are usually in the form of log-likelihood ratio (LLR) vector:

$$\text{LLR}(\mathbf{c}|\mathbf{y}) = \left[\log \frac{P(c_1 = 0|\mathbf{y})}{P(c_1 = 1|\mathbf{y})}, \log \frac{P(c_2 = 0|\mathbf{y})}{P(c_2 = 1|\mathbf{y})}, \dots, \log \frac{P(c_N = 0|\mathbf{y})}{P(c_N = 1|\mathbf{y})} \right]. \quad (1)$$

Alternatively to matrix representation, LDPC codes can also be represented by the associated bipartite Tanner graph [3,31]. In the Tanner graph, the variable nodes (VN) representing codeword bits are associated with columns of $\mathbf{H}$ and the check nodes (CN) representing parity checks are associated with rows of $\mathbf{H}$. The nonzero elements in $\mathbf{H}$ correspond to edges in the code graph. This means that the graph nodes are divided into two distinct sets (VNs and CNs), and that the edges only connect two different types of nodes. For an (N, K) LDPC code with block length N , the graph consists of N variable nodes and $M = N - K$ check nodes.

The most popular LDPC decoding algorithms are all algorithms involving belief propagation (BP), including sum product (SPA) algorithms and several simplified versions operating on LLR beliefs, such as min-sum algorithms (MSA), offset min-sum algorithms (OMSA), or normalized min-sum algorithms (NMSA) [32–34].

Since the message length in our assumed IoT system is assumed to be relatively short, the block lengths of codes that are considered and investigated in this article are short-to-medium. We investigate such codes constructed with a developed procedure that will be briefly presented in one of the next sections.

3.2. Quasi-Cyclic LDPC Codes

Many communication systems use the quasi-cyclic (QC) subclass of LDPC codes [15]. These codes have properties that make them suitable for efficient encoder and decoder designs, as well as for industrial implementations [17,24,35,36]. The PCM of QC-LDPC codes has a distinct structure composed of circulant permutation matrices (CPMs):

$$\mathbf{H} = \begin{bmatrix} \mathbf{P}^{s_{1,1}} & \mathbf{P}^{s_{1,2}} & \dots & \mathbf{P}^{s_{1,C}} \\ \mathbf{P}^{s_{2,1}} & \mathbf{P}^{s_{2,2}} & \dots & \mathbf{P}^{s_{2,C}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P}^{s_{R,1}} & \mathbf{P}^{s_{R,2}} & \dots & \mathbf{P}^{s_{R,C}} \end{bmatrix}, \quad (2)$$

where $\mathbf{P}^{s_{r,c}}$ is either a $P \times P$ all-zero submatrix (by convention, it is indicated by $s_{r,c} = \infty$) or an $s_{r,c}$ -CPM of size $P \times P$, that is, a matrix obtained by cyclically shifting every row of an identity matrix $\mathbf{I}_{P \times P}$ by $s_{r,c}$ positions to the right [37]. The particular code can be unambiguously defined by its size and the values of $s_{r,c} \in \{0, 1, \dots, P-1, \infty\}$, for macrorows $r = 1, \dots, R$, and macrocolumns $c = 1, \dots, C$.

The design of QC-LDPC codes can be treated as an important special case of the protograph-based LDPC code construction process [38]. In essence, this construction process is based on the copying and edge permuting of a smaller graph (protograph, base graph) to achieve a larger Tanner graph of a QC-LDPC code.

3.3. Raptor-like LDPC Codes

In contrast to block coding with a fixed code rate, rateless coding is an error correction scheme that, given a fixed information vector, generates a (potentially unconstrained) series of redundant bits [39], making use of IR decoding on the receiver side. However, constructing good rateless codes with the possibility of efficient encoding and decoding requires some effort. One of the recently achieved solutions is protograph-based Raptor-like (PBRL) LDPC coding [12,13].

The PBRL-LDPC code can be considered to be a concatenation of a high-rate code (HRC), which is an LDPC code with a parity check matrix $\mathbf{H}_{\text{HR}}$ of size $M \times N$ and an incremental redundancy code (IRC) characterized by a low-density generator matrix (LDGM) $\mathbf{G} = [\mathbf{I}, \mathbf{H}_{\text{IR}}^T]$ [13]. In this paper, we employ the QC subclass of PBRL-LDPC codes, which will be called QC-RL-LDPC codes.

The matrices defining the QC-RL-LDPC code $\mathbf{H}_{\text{HR}}$ and $\mathbf{H}_{\text{IR}}$ both have a structure consisting of CPMs, as in (2). The size of matrix $\mathbf{H}_{\text{IR}}$ is $LP \times N$, which means that it has L macrorows of $P \times P$ circulants, giving a total of LP parity checks in the incremental redundancy scheme. The Tanner graph of such a QC-RL-LDPC code can be illustrated as presented in Figure 3, where the blocks denoted by $\mathbf{H}_{\text{HR}}$ and $\mathbf{H}_{\text{IR}}$ are edge permutators according to the matrices' structures.

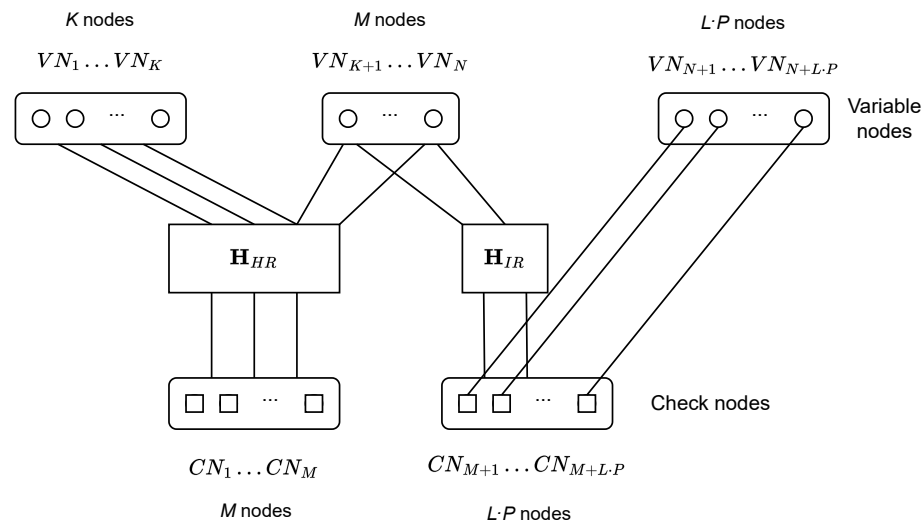


Figure 3. Tanner graph of a QC-RL-LDPC code.

4. The Developed System

This coding system is specifically created for a low-complexity IoT embedded device that communicates using a radio transmission protocol, with an energy-efficient ECC. The IoT node must meet certain criteria, such as being low-complexity and embedded-MCU-based, requiring a wireless communication system for various conditions and prioritizing energy efficiency of the radio transmitter during the design process. With these considerations in mind, it is suggested that a modern incremental-redundancy-based transmission could be advantageous.

The model of the proposed communication system employing incremental redundancy is presented in Figure 4. On the transmitter side, the HR (high-rate) and IR (incremental redundancy) parts of coded message are distinguished. The HR parity bits vector $\mathbf{p}_{HR}$ is an element of the codeword $\mathbf{c}$ that is initially transmitted, with a chosen transmission method. In the model presented in Figure 4, we assume that QAM modulation over a wireless communication channel is used; however, other transmission techniques and mediums can also be supported. After the initial transmission of $\mathbf{c}$, the transmitter generates and subsequently transmits IR parity bits in vectors $\mathbf{p}_{IR,j}$, $j = 1, \dots, L$ as long as it receives an acknowledgment (ACK) of successful decoding. The IR vectors are of length P , and will also be called IR chunks.

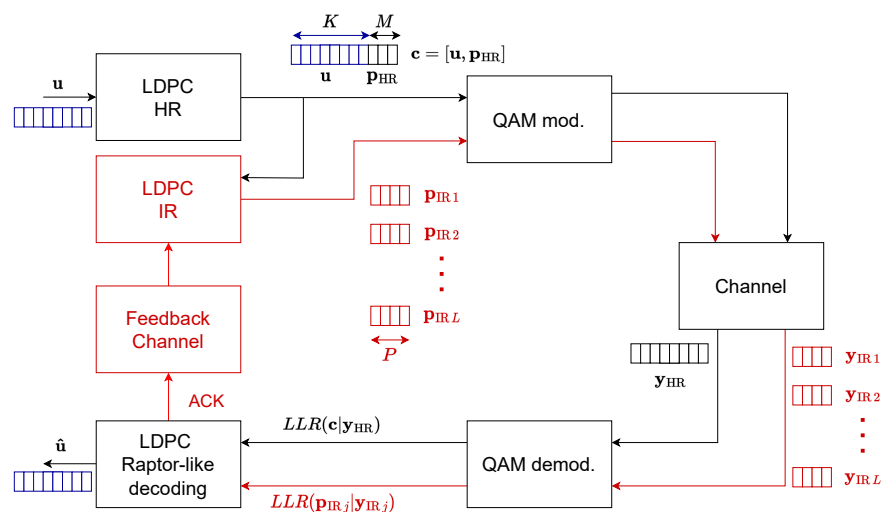


Figure 4. Communication system model with the developed QC-RL-LDPC coding.

The developed system is a kind of adaptive-rate scheme, in which the coding rate of the transmission system is variable in the range $R_{\max} \geq R \geq R_{\min}$, with the highest rate corresponding to the HR code being as in (3), and the lowest rate (4) reflecting the situation where the maximum number of chunks is required for correct transmission. If decoding is still unsuccessful after sending L IR chunks, the message can be retransmitted, either immediately or scheduled for later retransmission.

$$R_{\max} = \frac{K}{N}, \quad N = K + M \quad (3)$$

$$R_{\min} = \frac{K}{K + M + L \cdot P} \quad (4)$$

We performed an analysis of the energy efficiency of a developed transmitter embedded in an IoT device, with microcontroller (MCU) hardware. The receiver is assumed to be implemented with any device (CPU/GPU/hardware) with significantly greater computational power. It is assumed that either:

- The computational resources of the receiver (gateway) are far greater than the resources of the transmitter (IoT node), to the extent that IR decoding is performed in a fraction of the encoding time, as well as the feedback channel latency can be neglected; or
- The CPU that implements encoding in the transmitter, after encoding a $\mathbf{p}_{IR_j}$ chunk enters a sleep state for a specified time needed to receive a feedback message, in which state the energy consumption is negligible.

If one of these points is satisfied, the ACK can be assumed to be received virtually immediately after transmitting only as many chunks as are needed for correct decoding. This assumption is important for the energy efficiency calculations we made.

In the following sections, we provide the details of the whole designed system: the encoder design for a CPU; the decoding algorithm that was developed based on LDPC iterative decoding; and the code construction algorithm that can be used to design QC-RL-LDPC codes of arbitrary lengths and submatrix sizes. We provide experimental results in the form of encoding and decoding timings, and then energy consumption. We show that, in the case of unknown channel SNR, the implemented scheme can achieve a greater transmission energy efficiency than a fixed-rate coding system.

We also present the developed decoding scheme, based on the iterative LDPC decoding algorithm, that can be utilized in an incremental redundancy QC-RL-LDPC system. While the receiver (gateway) hardware is undefined but usually possesses a great computing power, a QC-RL-LDPC decoder in a CPU has also been developed for possible use in the reverse (gateway to IoT node) direction. We also provide the results of the decoder implementation for the same CPU as on the encoder side.

5. QC-RL-LDPC Encoding

The encoding of QC-RL-LDPC codes, as shown in Figure 4, always involves the HR part and then as many chunks in the IR part as are required to efficiently decode the message in the receiver.

The encoding of the HR part, that is, determining $\mathbf{p}_{HR}$, can be performed with one of the known efficient LDPC encoding algorithms, utilizing the QC feature. In [36], we presented a CPU-based QC-LDPC encoder implementation, which can also be used as an HR part encoder implementation in the QC-RL-LDPC system presented in this article. If this implementation is used, for the sake of code construction, it is important to note that the parity check matrix $\mathbf{H}_{HR}$, in addition to being in QC form, should follow the ALT (almost lower-triangular) form, as in the Richardson–Urbanke encoding [14], that is,

$$\mathbf{H}_{HR} = \begin{bmatrix} \mathbf{A} & \mathbf{B} & \mathbf{T} \\ \mathbf{C} & \mathbf{D} & \mathbf{E} \end{bmatrix} \quad (5)$$

where $\mathbf{T}$ is a lower-triangular matrix and all submatrices are CPMs. Cyclic changes in CPMs should preferably be selected so that $\Phi = \mathbf{E}\mathbf{T}^{-1}\mathbf{B} + \mathbf{D}$ is an identity matrix, $\Phi = \mathbf{I}_{g \times g}$, which can be achieved by fulfilling a condition originally proposed in [37]. As a result of the encoding of the HR information vector $\mathbf{u}$, the code vector $\mathbf{c} = [\mathbf{u}, \mathbf{p}_{\text{HR}}]$ is obtained. The Details of the encoding algorithm for QC-LDPC and its microcontroller implementation can be found in [14,36,37].

The IR encoding is based on the $\mathbf{H}_{\text{IR}}$ parity check matrix, which can be expressed as follows:

$$\mathbf{H}_{\text{IR}} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \\ \vdots \\ \mathbf{H}_L \end{bmatrix}, \quad \mathbf{H}_l = [\mathbf{P}^{s_{R+l,1}}, \mathbf{P}^{s_{R+l,2}}, \dots, \mathbf{P}^{s_{R+l,C}}], \quad l = 1, 2, \dots, L \quad (6)$$

where each macrorow $\mathbf{H}_l$ defines the parity checks (7) in GF(2) arithmetic for the l th chunk of incremental redundancy, $\mathbf{p}_{\text{IR}_l}$.

$$\mathbf{H}_l \cdot \mathbf{c}^T + \mathbf{p}_{\text{IR}_l} = \mathbf{0} \quad (7)$$

The consecutive chunks for $l = 1, 2, \dots, L$ can be computed as in the following expression:

$$\mathbf{p}_{\text{IR}_l} = \mathbf{H}_l \cdot \mathbf{c}^T = \mathbf{P}^{s_{R+l,1}} \mathbf{c}_1^T + \mathbf{P}^{s_{R+l,2}} \mathbf{c}_2^T + \dots + \mathbf{P}^{s_{R+l,C}} \mathbf{c}_C^T \quad (8)$$

Every chunk calculation as in (8) requires elementary operations of the type $\mathbf{P}^{s_{R+l,j}} \mathbf{c}_j^T$, as well as additions over GF(2), where every $\mathbf{P}^{s_{R+l,j}}$ is either:

- a zero matrix, in which case the product $\mathbf{P}^{s_{R+l,j}} \mathbf{c}_j^T$ is $\mathbf{0}$;
- or a circular shift of an identity matrix by $s_{R+l,j}$ positions, in which case it can be easily verified that the product $\mathbf{P}^{s_{R+l,j}} \mathbf{c}_j^T$ is a circular shift of $\mathbf{c}_j^T$ by $s_{R+l,j}$ positions.

Therefore, computing (8) for the QC case requires elementary operations of circular shifts of binary length- P subvectors in $\mathbf{c}^T$, as well as modulo-2 additions of length- P subvectors. Implementation in the microcontroller can be developed in C language, in which modulo-2 additions are realized with an “^” operator, and the circular shifts can be realized by using a bitwise shift operation and a bitwise OR operation, as in the following example:

```
output = (input << offset) | (input >> (size - offset));
```

where `offset` is the cyclic shift value and `size` is equivalent to the submatrix size P . Using this instruction for the circular shift, the input value can be represented in `uint32_t` type (if $P \leq 32$), while the `size` and `offset` can be represented in any unsigned type, for example, `uint8_t`. The output value is in the same format as the input.

6. Raptor-like Decoding of QC-RL-LDPC Codes

The decoding can be based on the belief propagation (BP) scheme with messages (beliefs) calculated as in the LDPC decoding with an SPA algorithm, or one of its LLR variants. In accordance with the proposal in [13], the overall concept is to decode the HRC and IRC parts together. The joint parity check matrix of both parts, corresponding to the whole Tanner graph of the QC-RL-LDPC code (as in Figure 3), is given as

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{\text{HR}} & \mathbf{0} \\ \mathbf{H}_{\text{IR}} & \mathbf{I} \end{bmatrix} \quad (9)$$

where $\mathbf{I}$ is an identity matrix. BP generally proceeds according to the graph in Figure 3. The decoding algorithm calculates the LLR messages for every edge of the graph in every iteration. However, it can be noted that it is inefficient to initialize the decoding process with a priori beliefs every time the decoder receives a new IR chunk. Therefore, we have developed the decoding procedure, in which the iterative process is suspended if the correct

vector is not found after some number of iterations and then resumed after receiving a new chunk with LLR-input IR information. Other than IR, beliefs (messages exchanged between nodes) remain as obtained in the previous iteration. The decoding algorithm is presented in Figure 5, where the processing sequence as well as the data flow (in blue lines) are shown. The algorithm can be summarized as follows.

1. Soft input decisions of the HR part, $LLR(c|y)$, are delivered first to the decoder, which proceeds with a number of I_{HR} iterations, computing variable node messages, denoted as Q_{nm} , and check node messages, denoted as R_{mn} , for $n = 1, \dots, N$ and $m = 1, \dots, M$.
2. In every iteration, based on the total beliefs for the HR codeword symbols Q_n , the error syndrome s is checked, and if a correct codeword is obtained, the decoding ends with success.
3. After I_{HR} iterations, the decoder waits for the next IR chunk. The chunk counter is denoted as j , while the iteration counter is i in Figure 5. After every subsequent chunk reception, the decoding is resumed for I_{IR} iterations, but the initial check-to-variable messages R_{nm} are taken from the previously suspended decoding part.

The computations of beliefs R_{mn} , Q_{nm} , for $n = 1, \dots, N$ and $m = 1, \dots, M$, can be done with one of the well-known BP message calculation methods, such as min-sum or normalized min-sum [34]. In our experiments, we used a computer model of the corrected min-sum algorithm [34]. An important choice of the number of iteration limits I_{HR} and I_{IR} can be made experimentally. The experimental results of the decoding performance will be provided in Section 8.

For a highly erroneous initial HR code block, the information calculated during initial decoding iterations is not enough to correct errors, but all iteratively enhanced beliefs are saved. Based on it, the next iterations are started, with additional soft information received. The point is that there is no need to calculate the beliefs from the beginning after every chunk transmission.

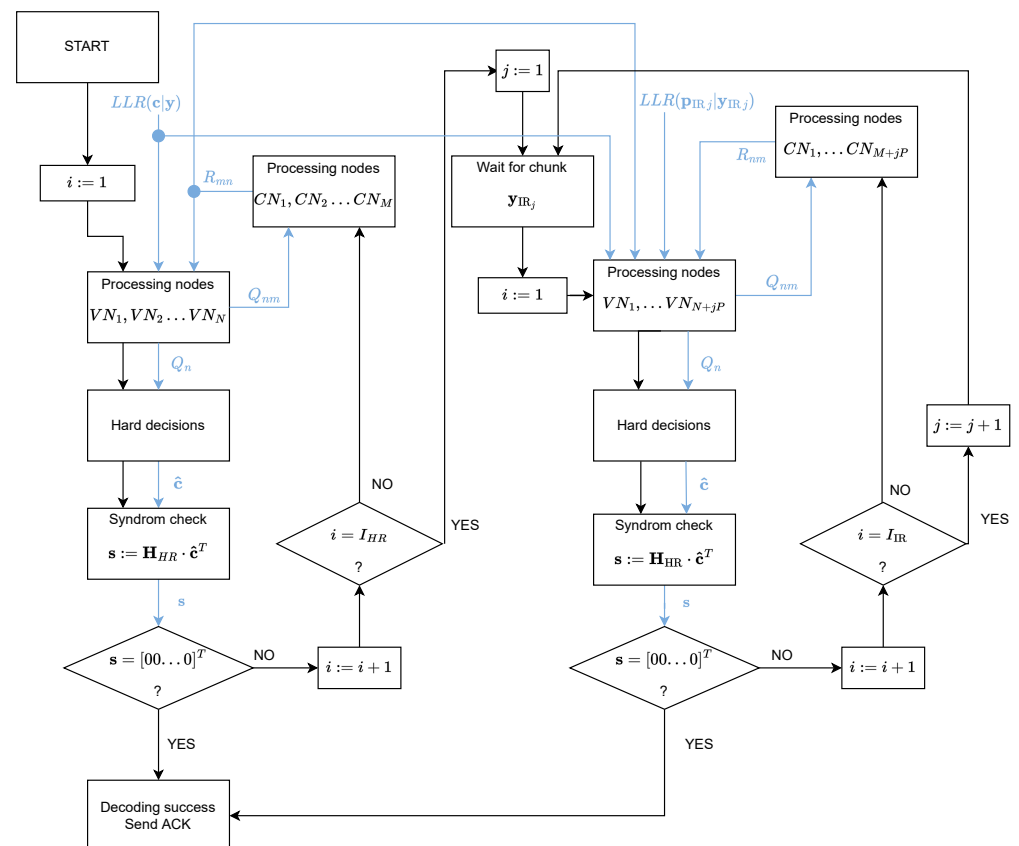


Figure 5. The developed algorithm for Raptor-like decoding.

7. QC-RL-LDPC Codes Construction for the Proposed System

Numerous methods for the construction of QC-LDPC have already been proposed, and a majority of finite code designs are centered around creating a Tanner graph structure with advantageous features, for example, in [40–44]. However, in this investigation, in order to design the entire QC-RL-LDPC coding scheme, we integrated the base graph construction by progressive edge growth (PEG, [40]) with protograph optimization for PBRL codes using density evolution [13] and proprietary algorithms to determine CPMs [36]. This allowed us to create a single, comprehensive method that gives both $\mathbf{H}_{\text{HR}}$ and $\mathbf{H}_{\text{IR}}$ as a result.

The proposed approach to the construction of the QC-RL-LDPC can be summarized as in the following steps:

1. The HRC base graph, that is, a graph corresponding to the $R \times C$ base matrix $\mathbf{W}_{\text{HR}}$ indicating nonzero submatrices in $\mathbf{H}_{\text{HR}}$ in (2), is constructed with an improved PEG algorithm [16]. By row and column permutations, $\mathbf{W}_{\text{HR}}$ is converted to ALT (almost lower-triangular) form.
2. The rows of the IR code base matrix $\mathbf{W}_{\text{IR}}$ are determined subsequently with a greedy optimization that uses density evolution with the reciprocal channel approximation, the method described in [13].
3. The parity check matrix of the quasi-cyclic HR code $\mathbf{H}_{\text{HR}}$ and the IR part $\mathbf{H}_{\text{IR}}$ are initialized with CPMs with zero shifts in the positions indicated by ones in $\mathbf{W}_{\text{HR}}$ and $\mathbf{W}_{\text{IR}}$, respectively.
4. The cyclical shift values $s_{r,c}$ in $\mathbf{H}_{\text{HR}}$, together with the matrix $\mathbf{H}_{\text{IR}}$, at positions corresponding to the values of 1 in $\mathbf{W}_{\text{HR}}$ and $\mathbf{W}_{\text{IR}}$ are subsequently modified in the entire structure $\mathbf{H}$ (9) simultaneously, with an algorithmic search method similar to the one proposed in [36,45] for LDPC codes. The algorithm seeks a Tanner graph with minimized existence of small cycles with low external connectivity, measured by the extrinsic message degree (EMD) feature.

The crucial characteristic of the designed algorithm is that after determining base matrices (protographs) for both parts $\mathbf{H}_{\text{HR}}$ and $\mathbf{H}_{\text{IR}}$, with appropriate methods previously known for rate-fixed LDPC codes, the whole graph of the QC-RL-LDPC coding scheme is optimized with a joint algorithm.

8. System Implementation Experimental Studies

Our research has been validated through experimental studies that examine various aspects of the developed QC-RL-LDPC system. These include encoding time; memory utilization of the encoder implemented with a microcontroller; investigation of decoding time and decoding parameter selection; error correction performance; and finally, normalized energy consumption estimation of the developed QC-RL-LDPC system. Due to the specifics of the design objective, which is an energy-efficient IoT uplink transmission scheme design, we investigated relatively short codes of length within 100 . . . 1000. We provide the microcontroller implementation results and a performance simulation for the channel with an unknown corruption level, represented by a wide range of SNRs (signal-to-noise ratios).

Several different codes ($\mathbf{H}_1, \dots, \mathbf{H}_5$) with diversified block lengths, chunk sizes and minimum rates, as listed in Table 1, were designed and used for this experimental study. As an example, Figure 6 shows a whole parity check matrix $\mathbf{H}$ of the code $\mathbf{H}_1$.

The parts of the designed communication system were implemented specifically with an STM32L476 microcontroller, for which we present the implementation results, including a case study of the power consumption and the energy efficiency calculations.

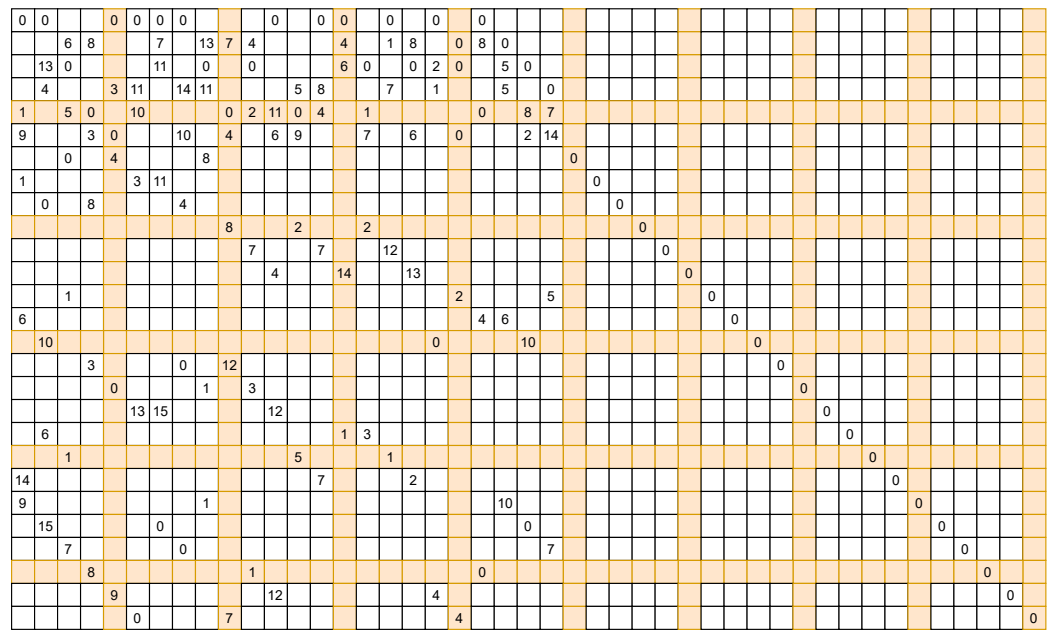


Figure 6. Parity check matrix of a QC-RL-LDPC code **H1**, both HR and IR parts, with submatrix size $P = 16$. The cyclic shift values are provided for nonzero submatrices.

Table 1. Parameters of the QC-RL-LDPC codes used for numerical experiments.

Matrix	K	$N_{\min}$	$N_{\max}$	$R_{\max}$	$R_{\min}$	P
H1	288	384	720	0.75	0.4	16
H2	256	384	896	0.667	0.286	32
H3	144	208	464	0.692	0.310	16
H4	192	240	600	0.8	0.320	8
H5	384	480	1200	0.8	0.320	16

8.1. Encoder Implementation

The encoder was designed with a C language implementation of the encoding algorithm presented in Section 5. Parts of the parity check matrix $\mathbf{H}_{\text{HR}}$ and $\mathbf{H}_{\text{IR}}$ are stored in the microcontroller flash memory in the form of positions (indexes) of nonzero submatrices together with corresponding circular shifts, as illustrated by an example in Figure 6.

We measured the encoding time in the QC-RL-LDPC system with the developed encoder, which was compiled and implemented in the STM32 low-power microcontroller unit. The results are presented in Figure 7. Note that the first part of the encoding (the HR part, shown as chunk-0 in Figure 7) takes a significantly longer time period than the subsequent encoding iterations of the following chunks (the IR part). We also implemented encoders for selected 5G codes, with Base-Graph-1 (BG1, see [17]) and submatrix sizes of 6 and 12, giving codes with information block lengths of 132 and 264, which are similar to our designed codes **H3**, **H2** and **H5**, respectively. The presented time measurements were then used to assess the energy consumption of the encoder, which is presented in Section 8.4.

The QC-RL-LDPC encoder stores the parity check matrices in memory using static allocation. Its memory utilization is comparable to the efficient QC-LDPC encoder previously presented by us in [36]. The maximum memory requirement for QC-RL codes is related to the maximum code size, with all chunks. The utilized microcontroller is able to allocate the needed memory and process the encoding.

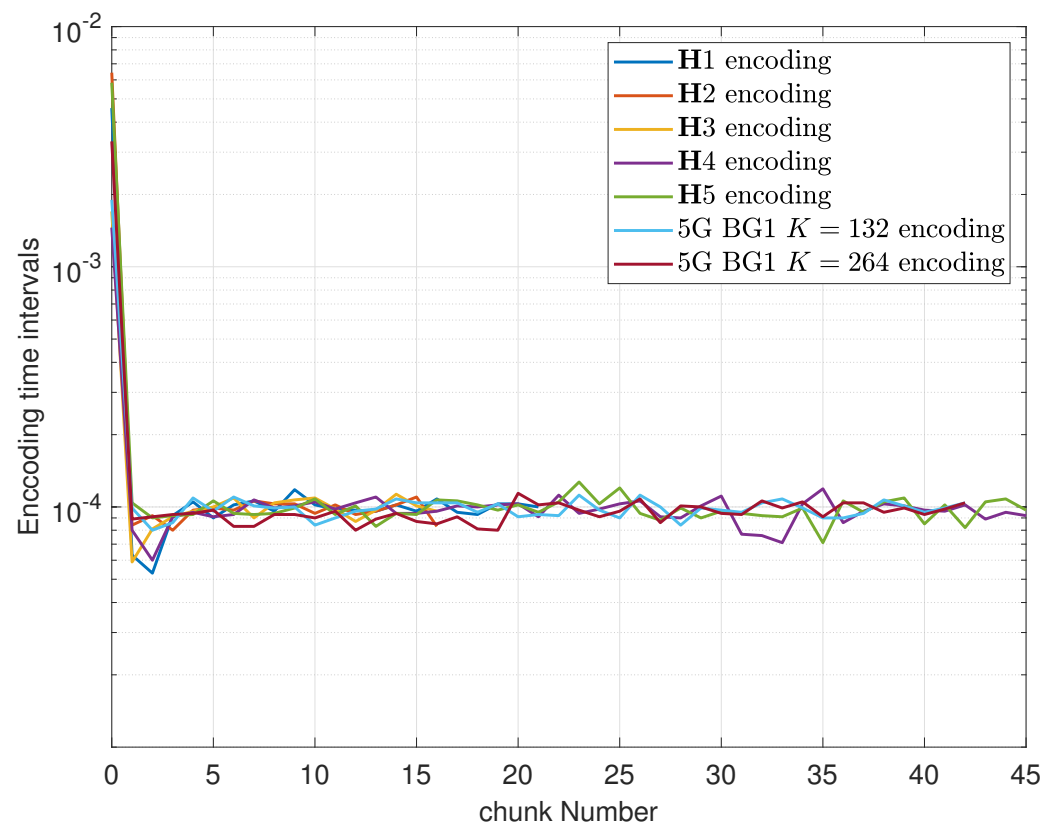


Figure 7. Observed encoding time intervals for initial HR block (chunk No. 0) and IR chunks $1, \dots, L$.

8.2. Decoder Implementation

The developed Raptor-like decoding algorithm, presented in Section 6, can be implemented on any computing platform utilized in the system gateway (Figure 1), including CPU, GPU, ASIC or FPGA. For the experimental investigation in this paper, Matlab software implementation on a personal computer was used. However, additionally, we developed the C language implementation for the STM32 microcontroller, the same as that used for the encoder. In Figure 8, we present the decoding time measurements of this implementation.

The microcontroller with the developed decoding algorithm is able to decode information portions efficiently, with lower computing time requirements if the received signal is of high quality (with small number of errors) and higher computing time consumption if the received signal is of low quality. In case of an erroneous received signal, the algorithm adapts and continues decoding until the correct result is achieved. During decoding, subsequent parity checks are used from $\mathbf{H}_{IR}$. With every additional chunk, the total (cumulative) decoding time increases, as shown in Figure 8. The figure provides averaged measured cumulative decoding time versus the number of chunks required for successful decoding. This means that the first (leftmost) point in every curve presents the time of decoding just the core HR part (with 0 additional chunks), and then the subsequent points present the total decoding times if the number of required chunks increases.

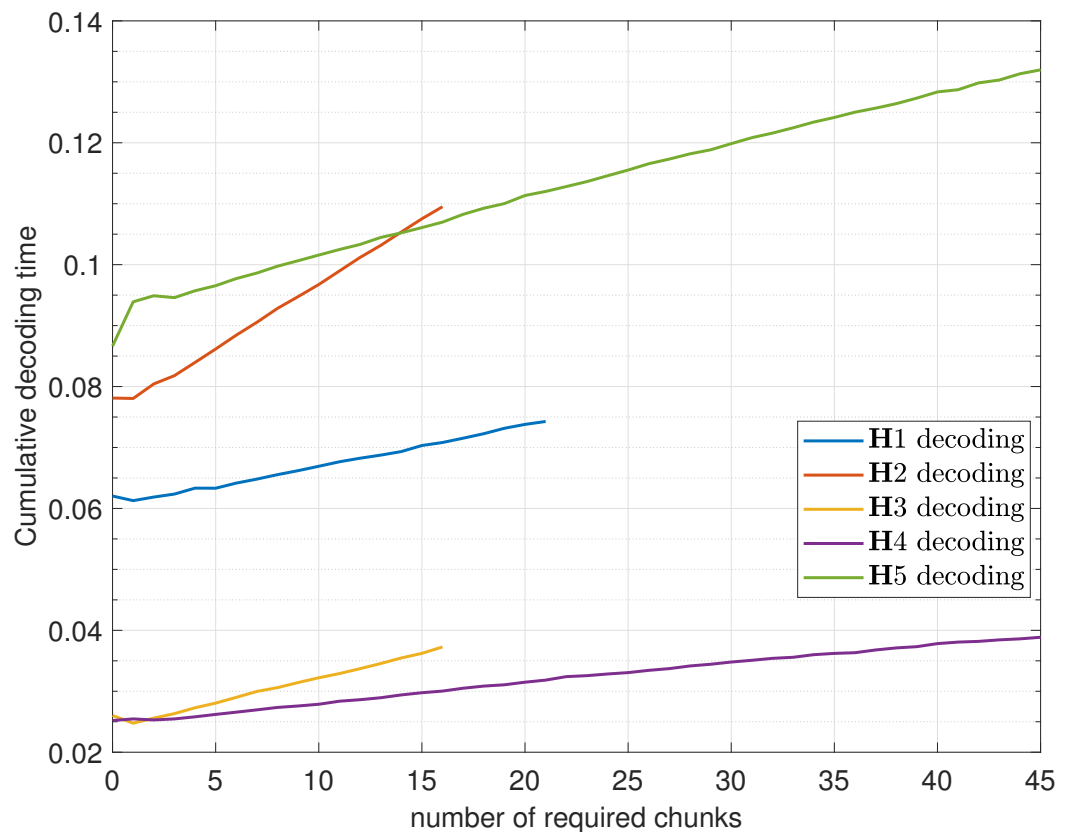


Figure 8. Mean of observed cumulative decoding time.

8.3. System Simulation Results

The experimental verification of error decoding performance that can be achieved with the proposed QC-RL-LDPC implementation was carried out by simulating a system as in Figure 4. In the results presented in this section, we assume a QPSK modulation (two bits per symbol) and an additive white Gaussian noise (AWGN) channel, with a broad range of simulated signal-to-noise ratios (SNRs). Correction performance is evaluated with Monte Carlo simulations, with the number of messages such that at least 100 of them are in error after HR part decoding, for every SNR.

In Figure 9, we provide simulation results in the form of goodput performance, which is defined as the ratio of the number of information bits correctly delivered to the total number of QPSK symbols transmitted, required for successful decoding [11]. Obviously, the higher the SNR, the lower the number of average transmitted chunks needed, which means the goodput is higher. The upper limit of goodput for QPSK is $2R_{\max}$, since if only the initial HR coded vector is transmitted, then $2R_{\max}$ information bits are successfully delivered per every QPSK symbol. The chunk size is assumed to be equal to P , because the encoding is performed in blocks of P bits.

We obtained the parity check matrix **H1** by lifting the protograph taken from [13] (Equations (11) and (12) therein), with submatrix size $P = 16$. Since the minimum achievable rate of this code is relatively high, $R_{\min} = 0.4$, the correction performance of the code **H1** in the low-SNR regime can be improved by proposing codes with more chunks (or rows in the IR part). To address this, we constructed matrix **H2** using the procedure outlined in Section 7, with an increased number of chunks and lower $R_{\min}$. This code can successfully correct errors with SNR well below 0dB, which is shown in Figure 9. Some other matrices that we also designed are **H3**, which can be used for shorter messages, with information vector length $K = 144$ but slightly underperforming **H2**; as well as **H4** and **H5**, with an increased initial rate $R_{\max}$, which provides a greater maximum goodput of 1.6.

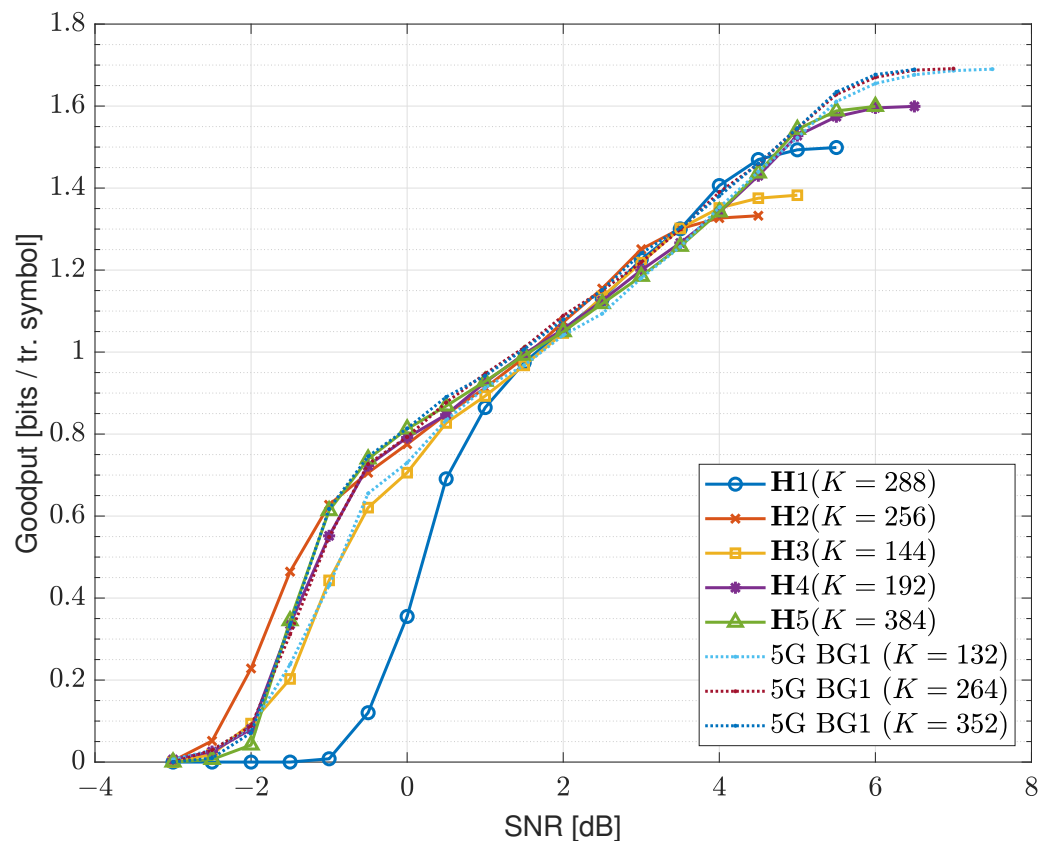


Figure 9. Goodput of the designed codes obtained by Monte Carlo simulation with AWGN channel and QPSK modulation.

As a point of comparison to other methods, we conducted simulations of a rate-adaptive scheme that utilizes 5G codes. Specifically, we examined BG1 with information block lengths of 132, 264 and 352. It can be observed in Figure 9 that the performance is similar to our designed codes H3, H2 and H5, respectively. This proves that the code construction part of this research provides codes that are not worse than state-of-the-art industrial LDPC codes. However, the advantage of the proprietary QC-RL-LDPC design is that the code parameters of HR code rate, information block length and submatrix size are flexible and can be freely chosen, while the 5G code family offers only two HR code configurations (BG1/BG2) and a discrete set of submatrix size/block length pairs.

The designed QC-RL-LDPC codes successfully achieve the objective of the rate self-adaptability processing. As shown in Figure 10, the number of chunks required for error-free reception increases as the SNR decreases. The code should be selected for a specific use case, and the algorithm developed can design codes for diversified needs of information block length K , rates $R_{\max}$, $R_{\min}$ and chunk sizes P .

Another decision that needs to be made in the system design is choosing the iteration limits for decoding, I_{HR} and I_{IR} (as defined in Section 6). This selection involves a trade-off between computational burden and decoding performance. However, as illustrated in Figure 11, which shows results for the same set of received input vectors, decoded with different parameters I_{HR} and I_{IR} , the choice of these parameters does not have a very significant impact on the performance. For the system simulation in this paper, we used $I_{\text{HR}} = 20$ and $I_{\text{IR}} = 7$.

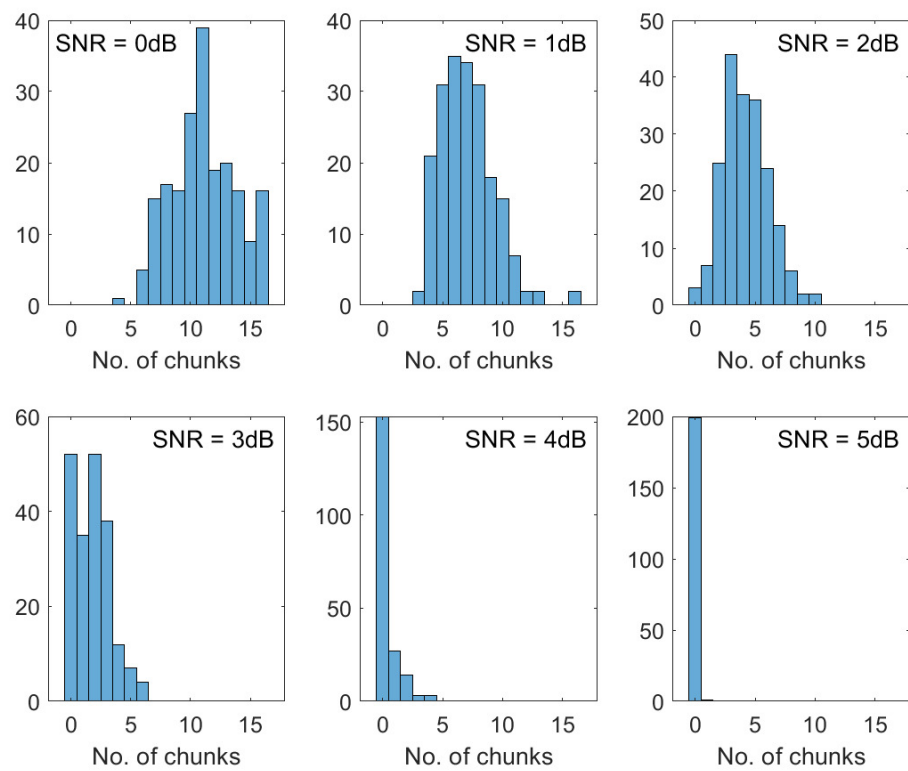


Figure 10. Number of chunks transmitted for different SNR levels for code H3 in simulation of transmission of 200 messages.

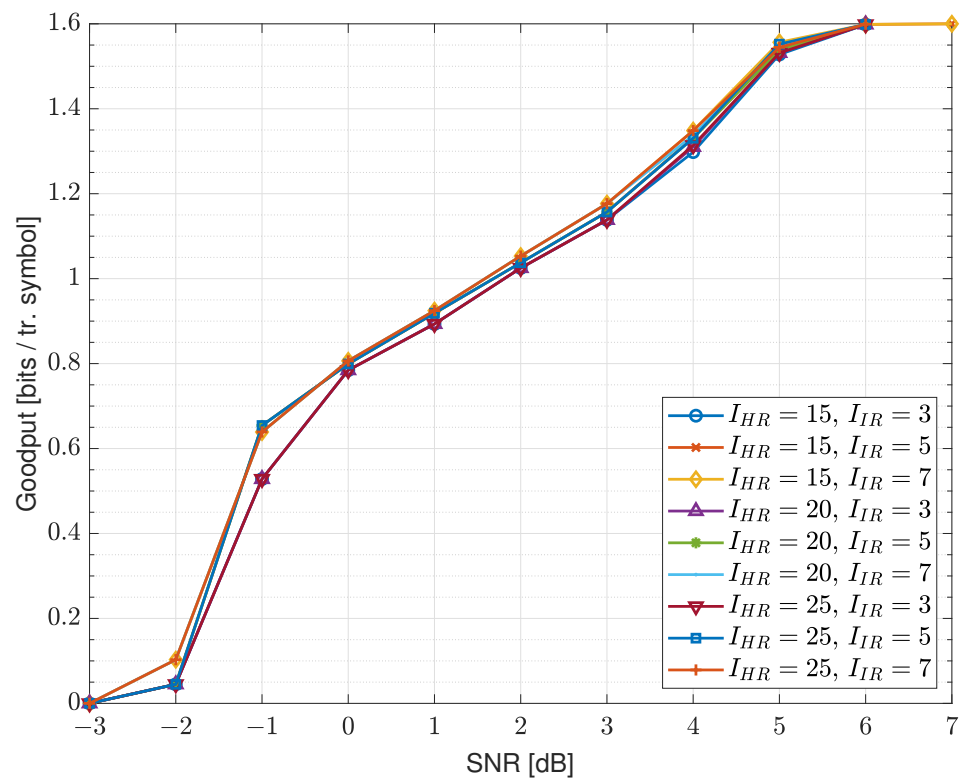


Figure 11. Goodput of the code H5 with different decoding iteration limits.

8.4. Energy Consumption Case Study

Based on the experimentally determined transmission performance (goodput) and encoding efficiency (time required for encoding) presented in previous subsections, we aimed to compare the designed system with the fixed-rate coded transmission system, as well as a system employing selected 5G coding configurations, in terms of the IoT transmitter energy efficiency. Since the calculations presented in this section are based on several assumptions, they should be treated as assessments of the energy consumption that can be expected in a real implementation. However, the results illustrate well the gains that can be achieved in the rate-adaptive system.

We assess the portion of the transmitter energy consumption that depends on the utilized coding system, that is, the sum of consumption by the encoding performed in a microcontroller plus consumption by the transmitter module. In the calculations, we have made the following assumptions:

1. The encoder is implemented in the STM32L476 microcontroller, with a supply voltage of 3.3 V and an active mode current draw of 10.7 mA.
2. The radio signal power is 25 mW (equal to the uplink limit in LoRa radio), the RF (radio frequency) module power efficiency is 50% and the transmission rate is 12,000 QPSK symbols per second (also comparable to LoRa limits).
3. The RF transmitter is active only during initial message bits and chunk bits transmissions.

By adding up the radio transmit energy and the encoding energy, factoring in the average number of chunks for the given SNR, we can determine the overall transmitter energy consumption. In the QC-RL-LDPC system, the energy consumption of the transmitter is dependent on the SNR, because with the change in the number of chunks needed, the total encoding time and the radio signal transmission time are also changing. This is illustrated by the evaluation results in Figures 12 and 13 for our designed codes of information block lengths of $K = 144$ and $K = 256$, respectively, as well as codes taken from 5G standard [17] with BG1 and similar block lengths of $K = 132$ and $K = 264$, respectively. The energy consumption provided is normalized and presented as one energy per one correctly decoded information bit.

In contrast, the energy consumption of the fixed-rate encoding is independent of the SNR. In Figures 12 and 13 we also include the energy consumption of the LDPC coding system, calculated with the same assumptions, for the same information block lengths $K = 144$ and $K = 256$, respectively, and the coding rate $R = 0.5$. It can be observed that:

- In the high-SNR regime (higher than about 2 dB), the normalized energy consumption is lower for the QC-RL-LDPC coding than for LDPC fixed-rate coding.
- In the regime just above 0 dB, the LDPC fixed-rate coding is more energy-efficient than QC-RL-LDPC coding. This is the regime for which an LDPC rate-0.5 code would be used under conditions that it was designed for; therefore, its advantage in this regime is as expected.
- For SNRs lower than about 1 dB for $K = 144$ and lower than about 0 dB for $K = 256$, the fixed-rate LDPC code can no longer correct errors with reasonable BER, while QC-RL-LDPC consumes more energy, but can still be used for transmission and correctly decoded.
- The developed transmission system, with the same IR redundancy scheme, but utilizing the state-of-the-art 5G codes, has very similar energy efficiency to the developed QC-RL-LDPC codes in the case of $K = 256$ (264) block length (Figure 13), and a little worse energy efficiency in the case of $K = 144$ (132) bits (Figure 12).

The justification for this slight difference between the designed QC-RL-LDPC codes and 5G codes is that the code H3 has the submatrix size $P = 16$, while the 5G code has $P = 6$, which means that more smaller chunks need to be encoded for a given parity vector size; the efficient QC encoding in a CPU can compute a chunk with similar speed, for different chunk sizes. This result emphasizes the flexibility of the submatrix size selection that is given by the proprietary QC-RL-LDPC design framework.

It can be inferred that the QC-RL-LDPC system exhibits higher energy efficiency compared to fixed-rate LDPC in high-SNR scenarios. This is due to its reduced transmission rate and encoding time. Additionally, it surpasses LDPC in low-SNR scenarios (below approximately 0 dB) since it can still effectively correct errors while utilizing more parity bit chunks. It should be especially desired to use rate-adaptive coding in situations where the channel SNR is not known in advance. It then improves not only the error correction possibilities, but also the energy efficiency.

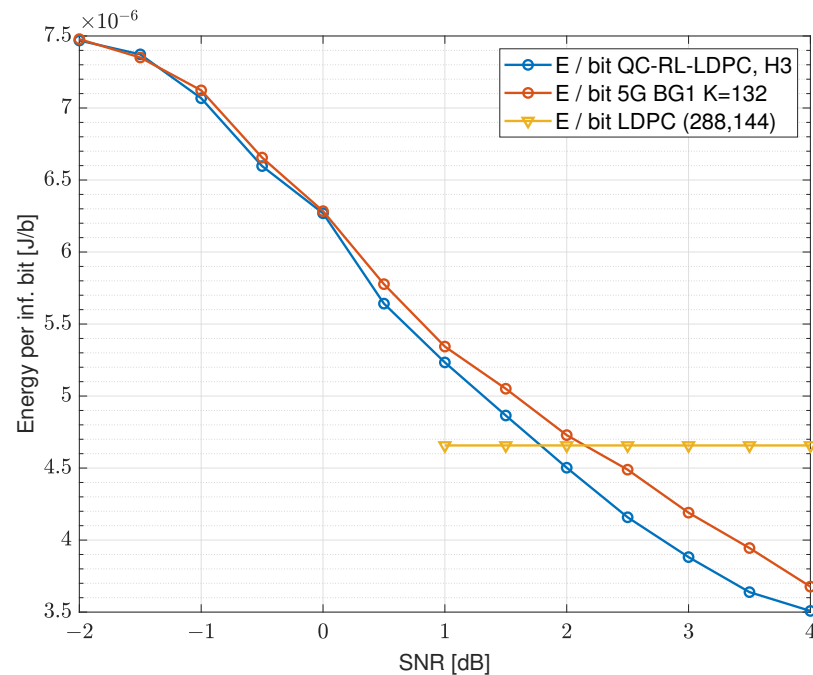


Figure 12. Transmitter energy consumption per information bit—codes of information block length 144b.

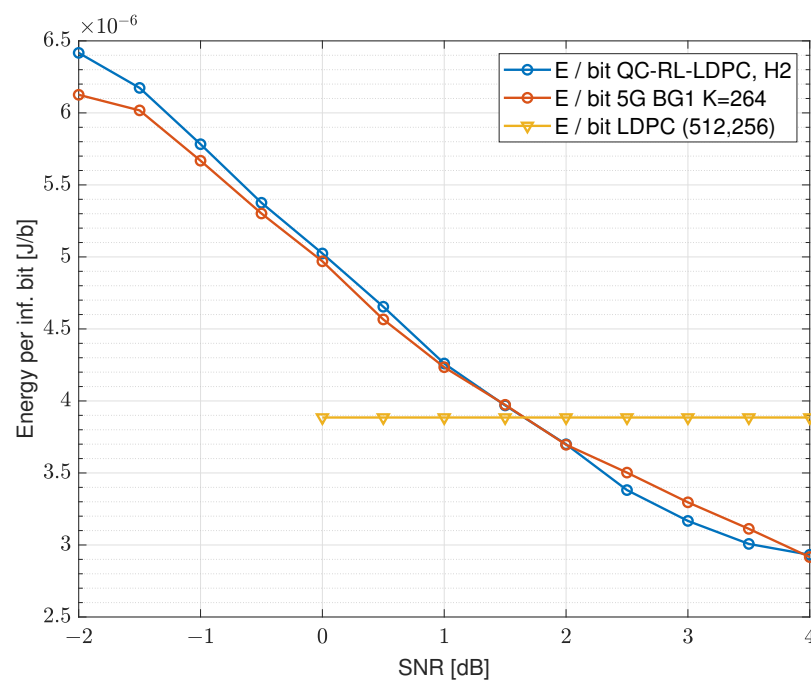


Figure 13. Transmitter energy consumption per information bit—codes of information block length 256b.

9. Concluding Remarks

This paper discusses a complete design procedure and provides implementation results for a QC-RL-LDPC coding system, specifically tailored for IoT uplink transmission. Our focus is on creating an energy-efficient encoding procedure for IoT nodes built with microcontroller devices, without strict throughput requirements, but where the energy efficiency is an important factor. To achieve this, we implemented a short-block-length Raptor-like LDPC scheme that is efficiently encodable. We showed that LDPC coding, commonly used in high-throughput communication systems and advanced protocols, is also applicable to such computationally constrained devices. Our experimental results showed that Raptor-like QC-RL-LDPC coding can be beneficial in terms of energy efficiency, in comparison with fixed-rate LDPC coding, and not worse than state-of-the-art 5G coding. Then, the developed design procedure, including QC-RL-LDPC code design, encoder implementation and the proposed decoding scheme for Raptor-like decoding, can be used in a rate-adaptive uplink transmission, with flexible parameter selection: block length, code rate and submatrix size.

The entire proposed system was implemented and validated for several designed binary QC-RL-LDPC parity check matrices, as well as compared with a similar design based on 5G codes. The comparison shows that the designed codes have at least similar efficiency to the 5G codes, while in some configurations, the flexible submatrix size choice can provide a slight energy efficiency improvement, even over 5G codes. The coding system can still evolve, and future work will be concentrated on the very promising extension of the proposed concepts to the nonbinary (NB) case, that is, the development of NB QC-RL-LDPC codes and the respective codec implementations.

Author Contributions: Conceptualization, J.H. and W.S.; Methodology, W.S.; Software, J.H.; Validation, J.H.; Investigation, J.H.; Supervision, W.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was co-financed by the Ministry of Education and Science of Poland under grant No. DWD/3/7/2019 as well as research subsidy for Silesian University of Technology, No. 02/160/BK_23/0009.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Al-Obaidi, K.M.; Hossain, M.; Alduais, N.A.M.; Al-Duais, H.S.; Omrany, H.; Ghaffarianhoseini, A. A Review of Using IoT for Energy Efficient Buildings and Cities: A Built Environment Perspective. *Energies* **2022**, *15*, 5991. [\[CrossRef\]](#)
2. Shammur, E.A.; Zahary, A.T. The Internet of Things (IoT): A survey of techniques, operating systems, and trends. *Libr. Hi Tech* **2020**, *38*, 5–66. [\[CrossRef\]](#)
3. MacKay, D.J.C. Good Error-Correcting Codes Based on Very Sparse Matrices. *IEEE Trans. Inf. Theory* **1999**, *45*, 399–431. [\[CrossRef\]](#)
4. Berrou, C.; Glavieux, A.; Thitimajshima, P. Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes. In Proceedings of the ICC '93—IEEE International Conference on Communications, Geneva, Switzerland, 23–26 May 1993; pp. 1064–1070.
5. Arikan, E. Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels. *IEEE Trans. Inf. Theory* **2009**, *55*, 3051–3073. [\[CrossRef\]](#)
6. Lin, S.; Costello, D.J., Jr. *Error Control Coding: Fundamentals and Applications*, 2nd ed.; Prentice-Hall, Inc.: Upper Saddle River, NJ, USA, 2004.
7. Le, H.D.; Mai, V.V.; Nguyen, C.T.; Pham, A.T. Throughput Analysis of Incremental Redundancy Hybrid ARQ for FSO-Based Satellite Systems. In Proceedings of the 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall), Honolulu, HI, USA, 22–25 September 2019; pp. 1–5.
8. Shokrollahi, A. Raptor codes. *IEEE Trans. Inf. Theory* **2006**, *52*, 2551–2567. [\[CrossRef\]](#)
9. Jayasooriya, S.; Shirvanimoghaddam, M.; Ong, L.; Johnson, S.J. Analysis and Design of Raptor Codes Using a Multi-Edge Framework. *IEEE Trans. Commun.* **2017**, *65*, 5123–5136. [\[CrossRef\]](#)
10. Jayasooriya, S.; Shirvanimoghaddam, M.; Ong, L.; Johnson, S.J. Raptor Codes for Higher-Order Modulation Using a Multi-Edge Framework. *IEEE Wirel. Commun. Lett.* **2018**, *7*, 110–113. [\[CrossRef\]](#)
11. Yu, W.; Narayanan, K.; Cheng, J.; Wu, J. Raptor Codes with Descending Order Degrees for AWGN Channels. *IEEE Commun. Lett.* **2020**, *24*, 29–33. [\[CrossRef\]](#)
12. Park, S.I.; Wu, Y.; Kim, H.M.; Hur, N.; Kim, J. Raptor-Like Rate Compatible LDPC Codes and Their Puncturing Performance for the Cloud Transmission System. *IEEE Trans. Broadcast.* **2014**, *60*, 239–245. [\[CrossRef\]](#)

13. Chen, T.Y.; Vakilinia, K.; Divsalar, D.; Wesel, R.D. Protograph-Based Raptor-Like LDPC Codes. *IEEE Trans. Commun.* **2015**, *63*, 1522–1532. [CrossRef]
14. Richardson, T.J.; Urbanke, R.L. Efficient Encoding of Low-Density Parity-Check Codes. *IEEE Trans. Inf. Theory* **2001**, *47*, 638–656. [CrossRef]
15. Fossorier, M.P.C. Quasi-Cyclic Low-Density Parity-Check Codes from Circulant Permutation Matrices. *IEEE Trans. Inf. Theory* **2004**, *50*, 1788–1793. [CrossRef]
16. Xiao, H.; Banihashemi, A.H. Improved Progressive-Edge-Growth (PEG) Construction of Irregular LDPC Codes. *IEEE Commun. Lett.* **2004**, *8*, 715–717. [CrossRef]
17. 3GPP Technical Specification TS 38.212 Version 16.2.0 Release 16. 5G; NR; Multiplexing and Channel Coding, 2020-07. Available online: https://www.etsi.org/deliver/etsi_ts/138200_138299/138212/16.02.00_60/ts_138212v160200p.pdf (accessed on 7 June 2023).
18. Bae, J.H.; Abotabl, A.; Lin, H.P.; Song, K.B.; Lee, J. An overview of channel coding for 5G NR cellular communications. *APSIPA Trans. Signal Inf. Process.* **2019**, *8*, e17. [CrossRef]
19. Gallager, R. Low-density parity-check codes. *IRE Trans. Inf. Theory* **1962**, *8*, 21–28. [CrossRef]
20. MacKay, D.J.C.; Neal, R.M. Near Shannon Limit Performance of Low Density Parity Check Codes. *Electron. Lett.* **1996**, *32*, 1645–1646. [CrossRef]
21. Richardson, T.J.; Shokrollahi, M.A.; Urbanke, R.L. Design of Capacity-Approaching Irregular Low-Density Parity-Check Codes. *IEEE Trans. Inf. Theory* **2001**, *47*, 619–637. [CrossRef]
22. Stark, M.; Lewandowsky, J.; Bauch, G. Information-Bottleneck Decoding of High-Rate Irregular LDPC Codes for Optical Communication Using Message Alignment. *Appl. Sci.* **2018**, *10*, 1884. [CrossRef]
23. Nguyen, T.T.B.; Tan, T.N.; Lee, H. Low-Complexity High-Throughput QC-LDPC Decoder for 5G New Radio Wireless Communication. *Electronics* **2021**, *10*, 516. [CrossRef]
24. Hailes, P.; Xu, L.; Maunder, R.G.; Al-Hashimi, B.M.; Hanzo, L. A Survey of FPGA-Based LDPC Decoders. *IEEE Commun. Surv. Tutorials* **2016**, *18*, 1098–1122. [CrossRef]
25. Li, M.; Derudder, V.; Bertrand, K.; Desset, C.; Bourdoux, A. High-Speed LDPC Decoders Towards 1 Tb/s. *IEEE Trans. Circuits Syst. I* **2021**, *68*, 2224–2233. [CrossRef]
26. Wu, H.; Wang, H. A High Throughput Implementation of QC-LDPC Codes for 5G NR. *IEEE Access* **2019**, *7*, 185373–185384. [CrossRef]
27. Likhobabin, E.A.; Ovinnikov, A.A.; Goriushkin, R.S.; Nikishkin, P.B.; Khokhryakov, E.I. High Throughput FPGA Implementation of LDPC Decoder Architecture for DVB-S2X Standard. In Proceedings of the 2022 International Conference on Information, Control, and Communication Technologies (ICCT), Astrakhan, Russia, 3–7 October 2022.
28. Zhao, Z.; Jiao, X.; Mu, J.; He, Y.C. Randomly Penalized Symbol Flipping Decoding of Non-Binary LDPC Codes. *IEEE Trans. Veh. Technol.* **2021**, *70*, 639–654. [CrossRef]
29. Bocharova, I.E.; Kudryashov, B.D.; Ovsyannikov, E.P.; Skachek, V.; Uustalu, T. Design and Analysis of NB QC-LDPC Codes Over Small Alphabets. *IEEE Trans. Commun.* **2022**, *70*, 2964–2976. [CrossRef]
30. Ferraz, O.; Subramaniam, S.; Chinthala, R.; Andrade, J.; Cavallaro, J.R.; Nandy, S.K.; Silva, V.; Zhang, X.; Purnaprajna, M.; Falcao, G. A Survey on High-Throughput Non-Binary LDPC Decoders: ASIC, FPGA, and GPU Architectures. *IEEE Commun. Surv. Tutorials* **2022**, *24*, 524–556. [CrossRef]
31. Tanner, R.M. A Recursive Approach to Low Complexity Codes. *IEEE Trans. Inf. Theory* **1981**, *27*, 533–547. [CrossRef]
32. Chen, J.; Dholakia, A.; Eleftheriou, E.; Fossorier, M.; Hu, X.Y. Reduced-complexity decoding of LDPC codes. *IEEE Trans. Commun.* **2005**, *53*, 1288–1299. [CrossRef]
33. Chen, J.; Fossorier, M. Density evolution for two improved BP-Based decoding algorithms of LDPC codes. *IEEE Commun. Lett.* **2002**, *6*, 208–210. [CrossRef]
34. Zhao, J.; Zarkeshvari, F.; Banihashemi, A.H. On Implementation of Min-Sum Algorithm and Its Modifications for Decoding Low-Density Parity-Check (LDPC) Codes. *IEEE Trans. Commun.* **2005**, *53*, 549–554. [CrossRef]
35. Bocharova, I.E.; Kudryashov, B.; Johannesson, R. Searching for Binary and Nonbinary Block and Convolutional LDPC Codes. *IEEE Trans. Inf. Theory* **2016**, *62*, 163–183. [CrossRef]
36. Hyla, J.; Sulek, W.; Izydorczyk, W.; Dziczkowski, L.; Filipowski, W. Efficient LDPC Encoder Design for IoT-Type Devices. *Appl. Sci.* **2022**, *12*, 2558. [CrossRef]
37. Myung, S.; Yang, K.; Kim, J. Quasi-Cyclic LDPC Codes for Fast Encoding. *IEEE Trans. Inf. Theory* **2005**, *51*, 2894–2901. [CrossRef]
38. Fang, Y.; Bi, G.; Guan, Y.L.; Lau, F.C.M. A Survey on Protograph LDPC Codes and Their Applications. *IEEE Commun. Surv. Tutorials* **2015**, *17*, 1989–2016. [CrossRef]
39. Erez, U.; Trott, M.D.; Wornell, G.W. Rateless Coding for Gaussian Channels. *IEEE Trans. Inf. Theory* **2012**, *58*, 530–547. [CrossRef]
40. Hu, X.Y.; Eleftheriou, E.; Arnold, D.M. Regular and Irregular Progressive Edge-Growth Tanner Graphs. *IEEE Trans. Inf. Theory* **2005**, *51*, 386–398. [CrossRef]
41. Wang, Y.; Draper, S.C.; Yedidia, J.S. Hierarchical and High-Girth QC LDPC Codes. *IEEE Trans. Inf. Theory* **2013**, *59*, 4553–4583. [CrossRef]
42. Tasdighi, A.; Banihashemi, A.H.; Sadeghi, M.R. Efficient Search of Girth-Optimal QC-LDPC Codes. *IEEE Trans. Inf. Theory* **2016**, *62*, 1552–1564. [CrossRef]
43. Sulek, W. Quasi-Regular QC-LDPC Codes Derived From Cycle Codes. *IEEE Commun. Lett.* **2016**, *20*, 1705–1708. [CrossRef]

44. Kim, I.; Song, H.Y. Some New Constructions of Girth-8 QC-LDPC Codes for Future GNSS. *IEEE Commun. Lett.* **2021**, *25*, 3780–3784. [[CrossRef](#)]
45. Sulek, W. Protograph Based Low-Density Parity-Check Codes Design with Mixed Integer Linear Programming. *IEEE Access* **2019**, *7*, 1424–1438. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.