


Article

# Low Power QC-LDPC Decoder Based on Token Ring Architecture

Mateusz Kuc \* , Wojciech Sułek  and Dariusz Kania 

Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology,  
ul. Akademicka 16, 44-100 Gliwice, Poland; wojciech.sulek@polsl.pl (W.S.); dariusz.kania@polsl.pl (D.K.)

\* Correspondence: Mateusz.Kuc@polsl.pl

Received: 5 November 2020; Accepted: 26 November 2020; Published: 30 November 2020



**Abstract:** The article presents an implementation of a low power Quasi-Cyclic Low-Density Parity-Check (QC-LDPC) decoder in a Field Programmable Gate Array (FPGA) device. The proposed solution is oriented to a reduction in dynamic energy consumption. The key research concepts present an effective technology mapping of a QC-LDPC decoder to an LUT-based FPGA with many limitations. The proposed decoder architecture uses a distributed control system and a Token Ring processing scheme. This idea helps limit the clock skew problem and is oriented to clock gating, a well-established concept for power optimization. Then the clock gating of the decoder building blocks allows for a significant reduction in energy consumption without deterioration in other parameters of the decoder, particularly its error correction performance. We also provide experimental results for decoder implementations with different QC-LDPC codes, indicating important characteristics of the code parity check matrix, for which an energy-saving QC-LDPC decoder with the proposed architecture can be designed. The experiments are based on implementations in the Intel Cyclone V FPGA device. Finally, the presented architecture is compared with the other solutions from the literature.

**Keywords:** LDPC; QC-LDPC; FPGA; Min-Sum; distributed control system; token ring; partially-parallel decoder

---

## 1. Introduction

With the advancements in information technology, there is a constantly growing volume of data processing, for example, for medical, financial, social communication, and entertainment purposes. The increasing quality of multimedia content, like photos, videos and audio recordings has led to the need to process but also transmit digital information in a faster and more efficient way. High-speed and errorless data transmission systems are an important part of the information technology development. However, in order to make, for example, a radio communication system errorless, special mechanisms called channel coding or Error Correction Coding (ECC) are needed. ECC makes it possible to correct a certain portion of transmission errors without the need for data retransmission. Such mechanisms are based on algorithms with relatively high computational complexity, and thus the elements responsible for data correction in high-speed communication technology are usually complex and energy-consuming.

The energy-efficiency problems are significant in any Very Large Scale Integration (VLSI) circuit design, where an increase in temperature can lead to serious damage [1]. However, in the case of mobile communication devices powered by batteries, the efficient use of power is of special importance. By reducing power consumption, the operating time of a mobile device is extended [2,3]. Let us also

mention the resulting environmental benefits. By limiting electricity consumption of designed devices, we reduce the negative impact on the environment, as well as reduce the cost of operating the device itself.

There exists a relationship between the seemingly independent issues of reliable data transmission and the electricity consumption in data transmission systems. It results from the relationship between the power of the transmitted signal and the expected level of reliability (error rate), as well as from the fact that there are possibilities to limit the energy consumption of hardware systems implementing the advanced correction mechanisms. A properly designed system should ensure error-free data transfer, with the lowest possible consumption of electricity. Most of the published research works in the field of ECC focus on finding fast coding-decoding solutions. However, few works take into account the need to search also for energy-saving solutions.

With the development of communication technology, many methods of ECC have been developed, with different levels of computational complexity on the one hand and error correction effectiveness (performance) on the other hand. One of the basic parameters characterizing the ECC coding system is the Bit Error Rate (BER), which quantifies the number of erroneous bits observed in the receiver to the total number of transmitted bits. Similarly, a Frame Error Rate (FER) indicates the relative number of observed erroneous data frames (blocks) in a coding scheme, for schemes that partition data into blocks (which is the case in a broad range of block coding methods).

Low-Density Parity-Check (LDPC) codes are one of the best block ECC methods known today. They were first presented by R. G. Gallager [4], but the computing technology available at that time did not allow for their practical use. In 1999, LDPC codes were recalled by D. J. C. MacKay [5]. The scientific community began to look for methods to simplify the algorithm for implementation in integrated circuits. The proposed simplifications enabled the use of LDPC codes in the telecommunication industry, for example, (DVB-S2) [6,7], Ethernet 10GBase-T [8,9], 802.16e (WiMax) [10,11], 802.11ad (WiGig) [12,13], 802.11n/ac/ax (WiFi) [14,15], GSFC-STD-9100 (NASA) [16] and recently 5G mobile [17,18].

Along with the growing popularity of LDPC codes, the development of optimized LDPC codes and their decoding methods with improved performance is observed. Due to the high computational complexity of such effective solutions, the power consumption seems to be an important, but often neglected issue. Designing energy-efficient LDPC decoders calls for providing a system solution with power saving potential at several design stages. It seems that at each project level, specific techniques to reduce energy consumption can be used [19].

Probably great benefits can be obtained by minimizing expected energy consumption at the system level. For example, we have found that it is possible to design the architecture with the possibility of involving elements responsible for power gating [20–22] of individual elements, or the clock gating [23–25]. Also, while reviewing methods to reduce energy consumption, the method of local voltage reduction was considered. Reducing the supply voltage allows to significantly reduce energy consumption by adjusting the supply voltage of individual system elements at the expense of deteriorating dynamic parameters. Power gating enables great energy consumption reduction according to [19]. Clock gating can be used in FPGAs, unlike power gating, which is not applicable there.

Clock gating allows for the greatest reduction in dynamic energy consumption. However, there are other methods of reducing energy consumption that are worth mentioning. These methods were not used in the presented article due to the insufficient energy-saving in relation to additional elements increasing the hardware demand of the decoder. The first interesting method is a preliminary calculation of the state of outputs (precomputation logic), which consists of adding a circuit, the purpose of which is to determine the state of the outputs one clock signal period earlier [26–28]. The second interesting method is determining the kernel [29–31], consisting of determining the dominant states of the sequential system from a subset of all states, and then implementing a separate, smaller element performing operations only for dominant states systems. The third interesting method is the implementation of the system in the

Globally Asynchronous Locally Synchronous (GALS) architecture [32–34], involving decomposition of a locally synchronous system through which data are exchanged via an asynchronous bus, and the clock frequency of each module is individually adapted to the needs of this module.

The aim of the paper is to implement an energy-saving LDPC decoder characterized by non-deteriorated dynamic parameters compared to a decoder without mechanisms limiting energy consumption. For this reason, and taking into account the limitations resulting from the use of FPGA, the article proposes an LDPC decoder architecture based on Token Ring oriented clock gating. The second chapter presents a theoretical introduction describing the most important issues concerning the energy-saving QC-LDPC decoder. The third chapter presents the implementation of the QC-LDPC decoder based on Token Ring architecture with clock gating, which was then tested and the results are presented in the fourth chapter. The article ends with chapter five, which summarizes the obtained results and then indicates further work.

Therefore, the following list of contributions within the framework of the article can be formulated:

1. General idea of technology mapping of the proposed QC-LDPC decoder in LUT-based FPGA oriented to clock gating (Section 2).
2. Development, implementation and testing of a new QC-LDPC decoder architecture based on Token Ring (Sections 2 and 3).
3. Proposal and design of original architecture low power QC-LDPC decoder oriented to a reduction in dynamic power into LUT-based FPGA (Section 3).
4. Performing experiments and presenting results confirming the effectiveness of the proposed solutions (Section 4).

## 2. LDPC Codes and Their Decoding—Preliminaries

LDPC codes are an ECC scheme belonging to the class of linear block coding methods. The encoding process of an  $(N, K)$  code of rate  $R = K/N$  adds  $M = N - K$  parity check elements to the vector  $\mathbf{u} = \{u_1, u_2, \dots, u_K\}$ , which is an information vector to be ECC protected. As a result, the code vector (codeword)  $\mathbf{c} = \{c_1, c_2, \dots, c_N\}$  is obtained. Vectors  $\mathbf{u}$  and  $\mathbf{c}$  are over Galois field  $\text{GF}(2)$  for the binary codes considered in this paper.

A binary  $(N, K)$  LDPC code is specified by a low density parity check matrix  $\mathbf{H}_{M \times N}$  with  $\text{GF}(2)$  elements. In the decoder, a vector  $\mathbf{X}$  of length  $N$  is approved to be a correct codeword if and only if it satisfies the parity check equation  $\mathbf{H}\mathbf{X}^T = \mathbf{0}_{M \times 1}$  over  $\text{GF}(2)$  field arithmetic. Otherwise, a specific error correction decoding algorithm can be used. The LDPC systems employ iterative belief propagation decoding algorithms [5].

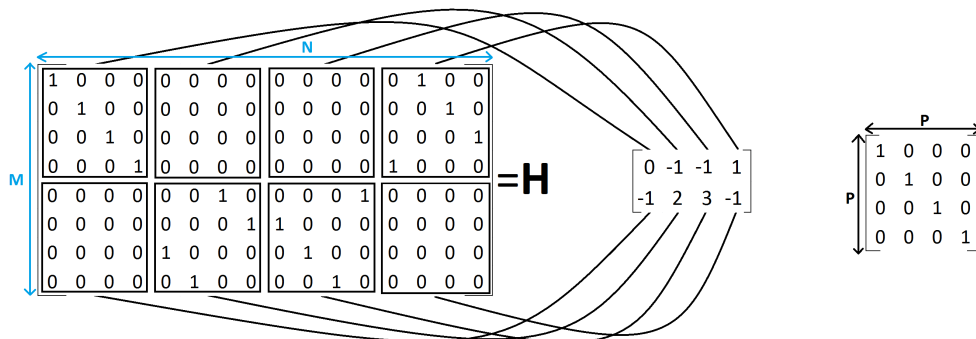
A parity check matrix that contains the same number of “1” values in every row and the same in every column is called a regular matrix. A matrix that does not satisfy the above condition is called an irregular matrix and a corresponding code is called irregular code. Irregular codes, with properly selected weight distributions of ones in columns of  $\mathbf{H}$  are known to outperform their regular counterparts.

### 2.1. QC-LDPC Codes

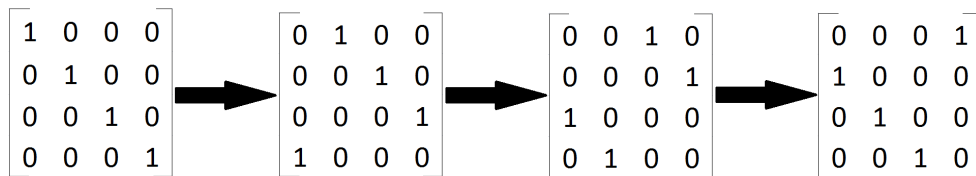
The structure of the parity check matrix may be subject to additional constraints according to design requirements, in particular to the codec implementation method. For example, in order to facilitate the hardware implementation of LDPC decoders, the matrix structured into Quasi-Cyclic (QC) form, as shown in the example in Figure 1, is commonly used. In this illustration, the matrix  $\mathbf{H}$  of size  $8 \times 16$  is divided into square submatrices of size  $4 \times 4$ , where each submatrix is either a zero matrix or a matrix formed by cyclically shifting the columns of the identity matrix (CPM—Circulant Permutation Matrix [35]). In general, the parity check matrix of a QC-LDPC code is partitioned with square submatrices  $\mathbf{P}$ , where each

is either a zero matrix of size  $P \times P$  or a CPM of size  $P \times P$ . QC-LDPC codes are particularly attractive to systems implemented in hardware, because the block-circulant parity check matrix structure enables fast and efficient message routing in a decoder implementation.

The parity check matrix  $\mathbf{H}$  of QC-LDPC code can be specified by a table (matrix) of cyclic shifts in CPMs (an example is shown in Figure 1), in which the nonnegative integers (from 0 to  $P$ ) represent the number of cyclic shift positions in CPM, while the values “-1” indicate null submatrices. Such a representation of the parity check matrix  $\mathbf{H}$  is convenient because it allows  $\mathbf{H}$  to be stored in a compressed form in the memory of the decoder. All possible shifts in the CPMs with  $P = 4$  are shown in Figure 2.



**Figure 1.** An example of a check matrix  $\mathbf{H}$  of the Quasi-Cyclic Low-Density Parity-Check (QC-LDPC) code and simplified notation in the form of an offset matrix.



**Figure 2.** Shifts of the submatrices matrix  $P$  by 0, 1, 2 and 3 positions.

### 2.2. LDPC Decoding Algorithm

The classical LDPC decoding algorithms belong to the class of iterative belief propagation (BP) algorithms, as recalled by MacKay [5]. In BP, beliefs are propagated between nodes in a Tanner graph [36] representation of an LDPC code. This graph is a bipartite graph with so-called control nodes representing rows of  $\mathbf{H}$ , bit nodes representing columns of  $\mathbf{H}$  and edges representing nonzero positions in  $\mathbf{H}$ . The commonly used simplified method for determining the beliefs (messages propagated between nodes in the algorithm) is known as the Min-Sum algorithm [37–39]. The decoding process is an estimation process of the transmitted code vector  $\mathbf{x} = [x_1, x_2, \dots, x_N]$  from input data (received “soft” values  $\mathbf{y} = [y_1, y_2, \dots, y_N]$ ). The following denotations are used in the algorithm presentation:

- $:=$  symbol means “becomes”,
- $\mathbf{c} = [c_1, c_2, \dots, c_N]$ —code vector,
- $\mathbf{y} = [y_1, y_2, \dots, y_N]$ —received vector,
- $Q_{nm}$ —belief LLR value from the  $n$ -th bit vertex to the  $m$ -th Tanner graph control vertex,
- $L(x_n|y_n)$ —LLR a priori probabilities for the  $n$ -th bit,
- $(1, N)$ —set of integers between 1 and  $N$ ,
- $\mathcal{N}(m)$ —a set of column indexes in the parity check matrix  $\mathbf{H}$  containing one in the  $m$ -th row,

$\mathcal{M}(n)$ —a set of row indexes in the parity check matrix  $\mathbf{H}$  containing one in the  $n$ -th column,  
 $R_{mn}$ —message from the  $m$ -th control vertex to the  $n$ -th bit vertex of Tanner graph,  
 $X_n$ —the  $n$ -th value of the decoded vector.

The Min-Sum decoding algorithm can be presented in consecutive steps as follows:

1. *Initialization of the decoder based on “soft” decisions:* Based on the vector of data  $\mathbf{y}$  received from the demodulator, the probability values are determined, which can be written as  $P(c_n = 0|y_n)$  and  $P(c_n = 1|y_n)$  for  $n = 1, \dots, N$ , which are the probabilities that the  $c_n$  bit was originally transmitted if the  $y_n$  value has been received. From values  $P(c_n = 0|y_n)$  and  $P(c_n = 1|y_n)$ , Log-Likelihood Ratios (LLR) are determined, which are the actual algorithm input. For every  $m \in \mathcal{M}(n)$  and  $n \in (1, N)$ . Initialization of  $Q_{nm}$  beliefs (messages) can be expressed as:

$$Q_{nm} := L(c_n|y_n) = \ln \left( \frac{P(c_n = 0|y_n)}{P(c_n = 1|y_n)} \right) \quad (1)$$

2. *Control nodes processing:* Calculation of  $R_{mn}$ , messages from  $m$ th control node to  $n$ th bit node. For every  $n \in \mathcal{N}(m)$  and  $m \in (1, M)$ :

$$R_{mn} := \left[ \prod_{k \in \mathcal{N}(m) \setminus n} \text{sgn}(Q_{km}) \right] \min_{k \in \mathcal{N}(m) \setminus n} |Q_{km}| \quad (2)$$

3. *Bit nodes processing:* Recalculation of  $Q_{nm}$ , messages from  $n$ th bit node to  $m$ th control node. For every  $m \in \mathcal{M}(n)$  and  $n \in (1, N)$ :

$$Q_{nm} := L(c_n|y_n) + \sum_{k \in \mathcal{M}(n) \setminus m} R_{kn} \quad (3)$$

4. *Total beliefs update:* Calculation of the current beliefs about all bits in  $\mathbf{c}$ . For every  $n \in (1, N)$ :

$$Q_n := L(c_n|y_n) + \sum_{k \in \mathcal{M}(n)} R_{kn} \quad (4)$$

5. *Hard decisions update:* Making trial hard decisions. For every  $n \in (1, N)$ :

$$X_n := \begin{cases} 1 & \text{gdy } Q_n < 0 \\ 0 & \text{gdy } Q_n \geq 0 \end{cases} \quad (5)$$

6. *Verification of control equations:*

$$\mathbf{H}\mathbf{X}^T = 0 \quad (6)$$

7. *Another iteration:* If the parity check Equation (6) is satisfied or the maximum number of iterations is reached, the computations are terminated. Otherwise, the next iteration starts from point (2).

The hardware implementation of the QC-LDPC decoder is closely related to the content of the parity check matrix  $\mathbf{H}$ . Considering the data processing parallelism in an implementation, the decoder architecture can be serial, partially-parallel or parallel. Partially-parallel decoders are the best choice for most applications as being a compromise between decoding throughput and hardware resources overhead. The parity check matrix of QC-LDPC code fits perfectly into the concept of partially-parallel decoders by structuring the  $\mathbf{H}$  matrix into  $\mathbf{P}$  submatrices that have only single “1” value in each column and each row. This allows the individual decoding steps to be parallelized and messages efficiently

propagated with the use of memories, with single memory word grouping messages corresponding to the single submatrix of  $\mathbf{H}$ .

Figure 3 presents the concept of partially-parallel decoder architecture. We have noticed that this architecture of the QC-LDPC decoder is consistent with the idea of a Token Ring network, one of the methods of creating a Local Area Network (LAN) developed by IBM [40]. In the concept of “token passing”, a token is awarded in a given moment to this network element that is about to start processing and transmitting data. We have applied this concept in our partially-parallel QC-LDPC decoder implementation. Here the token is passed between the control and bit nodes computing units, which require obtaining the result of the previous iteration, according to the presented MS decoding algorithm, to perform the next iteration of computation. The proposed token passing scheme for the partially-parallel QC-LDPC decoder architecture is shown in Figure 3.

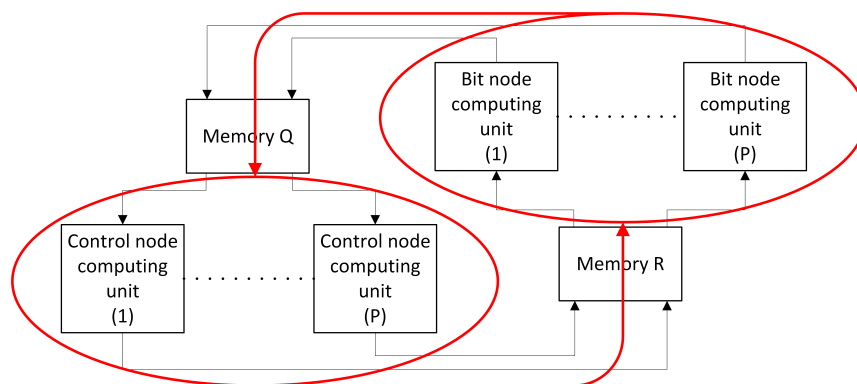


Figure 3. General idea of token passing in the designed partially-parallel QC-LDPC decoder architecture.

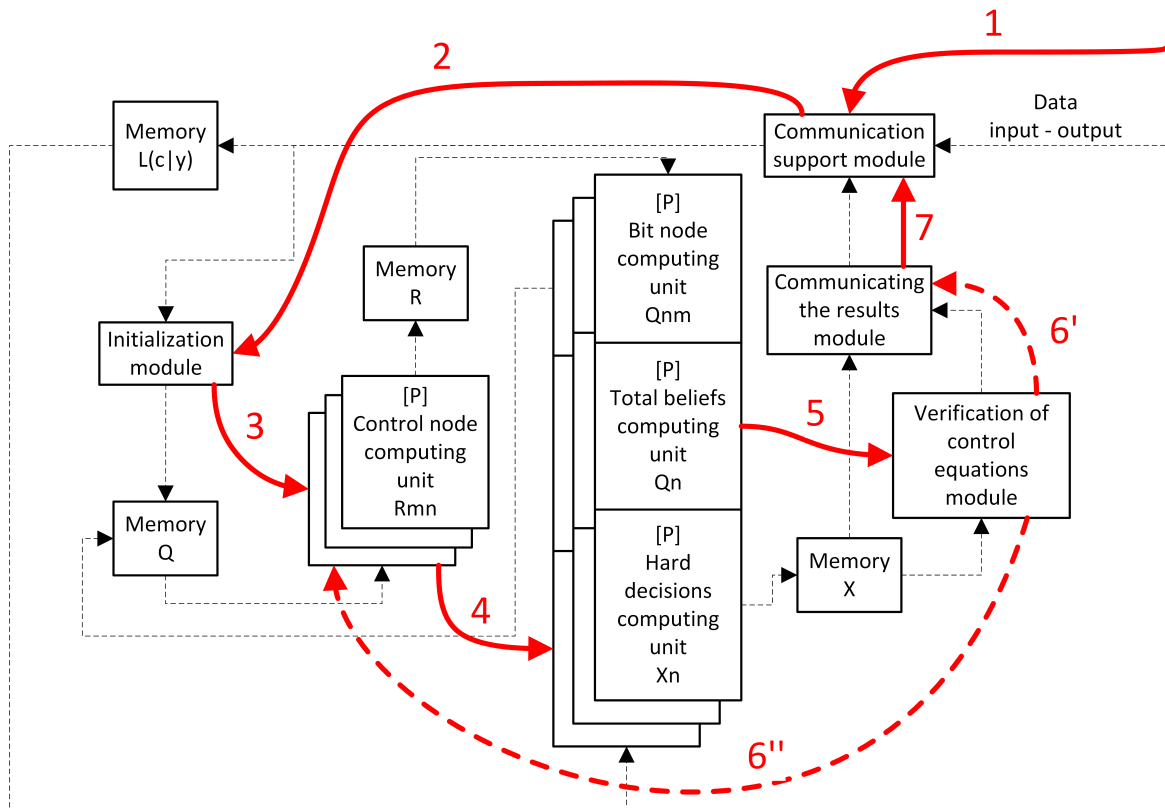
### 3. Implementation of the QC-LDPC Decoder Based on the Token Ring Architecture with Clock Gating

It has been assumed that in the designed implementation of a decoder with an architecture based on Token Ring, the set of implementable QC-LDPC codes should not be limited. The proposed general concept of the decoder in the form of a block diagram is shown in Figure 4. The flexibility of the proposed architecture results from the fact that it is easy to design a decoder for various parameters of the parity check matrix  $\mathbf{H}$ . Such configurable parameters of the  $\mathbf{H}$  matrix are: regular or irregular matrices, with different block lengths ( $N$ ), code rate ( $R$ ) and the size of the submatrix,  $P$ .

The decoder comprises five separate memory blocks,  $P$  control node computing units,  $P$  bit node computing units, a control equation verification module and modules responsible for initialization and communication with the environment. In the developed decoder structure, memories constitute a specific interface between the computing module. The message passing between the nodes is done via memory  $\mathbf{Q}$  (messages  $Q_{nm}$  in (2)–(3)) and memory  $\mathbf{R}$  (messages  $R_{mn}$  in (2)–(3)).

The communication support module receives the data to be decoded in the form of the LLR and saves it in the proper order in the memory  $L(c|y)$ . The received data is also passed to the decoder initialization module as in (1), which fills the memory  $\mathbf{Q}$ . Starting the initialization while the data transmission is in progress can slightly reduce the time in which the QC-LDPC decoder completes the decoding process. After finishing initialization, the process of calculating the control node messages begins. The computing unit of the control nodes takes the input data (messages) from the memory  $\mathbf{Q}$ , performs the computations according to the algorithm step (2) and stores results in the memory  $\mathbf{R}$ . The computing unit core consists of  $P$ -computing units responsible for parallel computations corresponding to output messages of  $P$  nodes.





**Figure 4.** The block diagram of the designed QC-LDPC decoder and the idea of transferring the Token (with the order marked) between the elements of the decoder.

In the next processing time slot, the messages of bit nodes (3), total beliefs (4) and hard decisions (5) are computed. Due to an obvious similarity in expressions (3) and (4), we found that a single complex decoder element can jointly realize these computations. The current hard decisions as in (5) are equivalent to the sign bits of the result of computations in (4). Therefore, computations according to (5) can also be integrated into this complex computing block, which we will call the bit node computing unit. The whole computation core consists of  $P$  such complex bit node computing units. Every bit node computation retrieves data (messages) from memories  $\mathbf{R}$  and  $L(c|y)$ , determines the bit node messages and saves results in memory  $\mathbf{Q}$  as well as current hard decisions in memory  $\mathbf{X}$ .

After the bit node processing step is completed, verification of the control equations module for current hard decisions is performed according to expression (6). This is done in the verification of the control equations module by checking each of the control equations defined by rows of  $\mathbf{H}$  in (6). When all controls are satisfied for current hard decisions, the final decoding result is sent to the decoder output via the communication of the result module. If the control equations are not satisfied, the next iteration in the decoder starts. The communication of the results module contains a counter that is incremented with each iteration. If the counter reaches a predetermined maximum value, the decoding is aborted and the decoder signals an erroneous decoding result.

Figure 4 shows the idea of a token passing between elements of the QC-LDPC decoder based on the Token Ring architecture. This idea is directly related to the operation of the distributed control system where each decoder element is responsible for the correct receipt and transfer of the token (token passing)

at the time of starting and finishing its work. The presented QC-LDPC decoder based on Token Ring architecture is an innovative design approach in relation to other known solutions from the literature.

The token is passed sequentially up to the point of verification of control equations module, where—depending on the result of the verification—one of the two possible token transfers,  $6'$  or  $6''$ , can be made. The token transfer according to  $6''$  will occur when the current hard decisions do not pass the verification, and the next decoding iteration is performed, with the token returning to the control nodes computing units. The global control system is partitioned into six separate controller modules in this proposed solution, constituting a distributed control system. In the proposed solution, at each stage of the computation, only one control unit is active, which gives a potential for reduction of dynamic energy consumption.

The distributed control system used in the proposed decoder architecture is the result of the search for a solution that is well suited to LUT-based FPGA resources. Typically, the control system is implemented in the form of an integrated control block from which control signals are distributed to individual executive blocks (decoder elements). This type of solution implemented in FPGA introduces a number of problems related to delays. In the proposed QC-LDPC architecture, the integral control system would create problems due to clock-skew [41]. The result of the search supported by numerous experiments is a distributed implementation of the control system, in which the control unit responsible for blocking the clock signal is integrated with the executive module. This proposed scheme is modeled on the solutions used in GALS systems. Practical experiments have shown that it works well in systems where the clock gating technique is used.

When presenting the concept of implementing a decoder with a distributed control system (Token Ring architecture), it is worth paying attention to the role of memories, which constitute a specific interface between the QC-LDPC decoder modules. The memory blocks in Figure 4 do not have a separate control module, because the control elements are located in the modules that communicate with them. The memory blocks can, therefore, be associated with interface elements, which are the place of storage of data used by individual decoder modules.

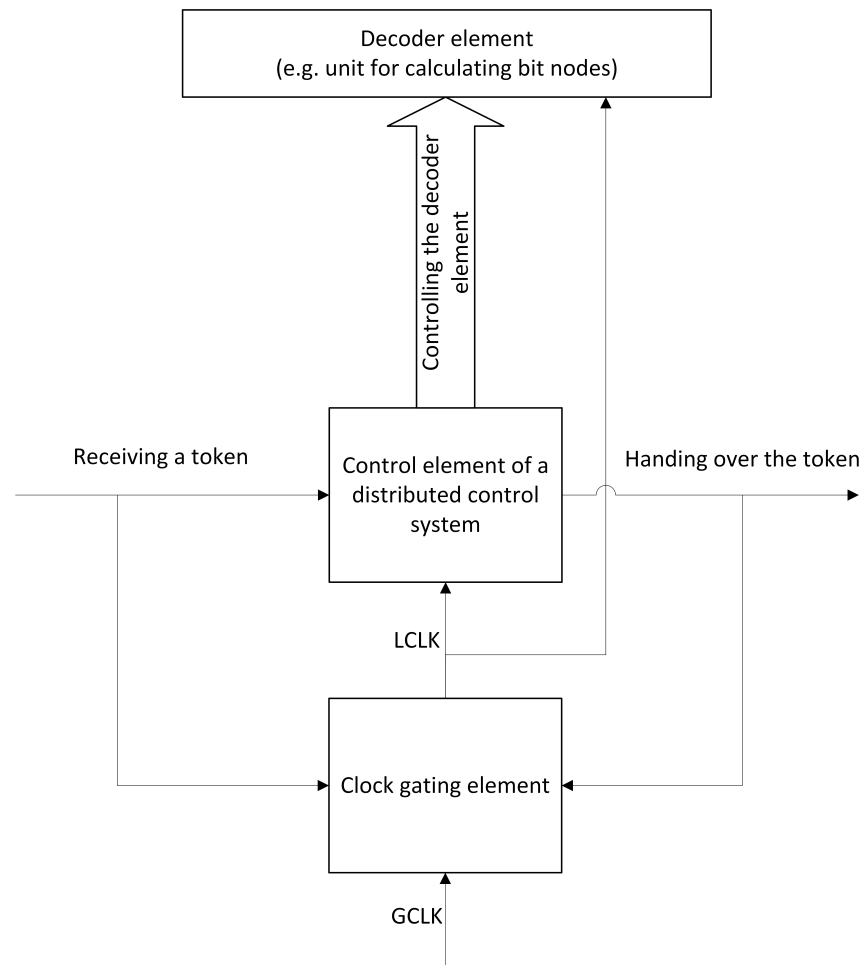
The Token Ring architecture with a distributed control system forms a basis for implementing clock gating of decoder modules for a significant dynamic power reduction. Activation of a selected control unit with a token may be done together with the computing elements controlled by this control unit. In this method, when computations are completed the module and the token is passed, the clock signal distributed in the respective computing unit is blocked (deactivated). Figure 5 illustrates the clock gating processing in a single control unit of the proposed distributed system. A similar circuit is found in every element of the decoder, to which a token can be passed.

Immediately after receiving the token, the clock gating element shown in Figure 5 activates the local clock signal (LCLK) by unblocking the gating circuit of the global clock signal (GCLK). The LCLK is passed to the element of the distributed control system, which starts to control the decoder element (computing module and memory). At the same time, the local clock signal LCLK is delivered to this computing module. After local computations are finished, the control system passes the token to the next control system element and blocks the LCLK. This processing scheme allows for significant reduction of the energy consumption due to activating the clock signal delivered to computation logic only in the needed time slot.

Table 1 presents the analysis of the operating time of individual decoder elements and the number of Adaptive Logic Module (ALM) blocks [37,42] utilized by the designed QC-LDPC implementation. These results have been obtained for a single-iteration-decoding of a QC-LDPC regular matrix with dimensions  $512 \times 1024$  ( $M \times N$ ),  $P = 64$  and  $Z = 3072$ , where  $Z$  is the number of nonzero elements of the parity check matrix  $\mathbf{H}$ . During normal operation, in the first iteration the decoder performs initialization once and in the last iteration it communicates the results module. For all other iterations, both decoder



elements are not active. The communication support module was not taken into account during the analysis because it is an element necessary for the proper operation of the decoder but it strictly depends on the design assumptions and can be implemented in any form (serial, parallel or partially-parallel), so it should not be included in Table 1. In Table 1, it is worth noting that for the proposed architecture, a single decoder element works no more than 34% of the total time and no more than 50% of ALM blocks have an active clock signal at any time.



**Figure 5.** The block diagram of clock gating and activation of a distributed control element using a token in an architecture based on a Token Ring.

**Table 1.** Analysis of the operation time of individual elements and the number of active Adaptive Logic Module (ALM) blocks of the QC-LDPC decoder based on the Token Ring architecture with the following QC matrix parameters:  $512 \times 1024$  ( $M \times N$ ),  $P = 64$  and  $Z = 3072$ .

Name of the Element	Time [%]	ALM [%]
Initialization module	11.2676	10.4995
Control node computing unit	26.7606	36.1635
Bit node computing unit	33.8028	49.5803
Verification of control equations module	16.9014	3.3714
Communicating the results module	11.2676	0.3853

#### 4. Experimental Results

The proposed architecture of the QC-LDPC decoder based on Token Ring with a distributed control system and clock gating has been implemented in the Altera (Intel) Cyclone V FPGA devices, for a number of different parity check matrices  $\mathbf{H}$  of QC-LDPC codes. The obtained implementation and power loss testing results are presented in Table 2. The presented results concern decoders with 4-bit data quantization of decoder messages and the Min-Sum computation algorithm. Table 2 contains characterizations of the parity check matrices  $\mathbf{H}$ , where  $I$  refers to an irregular matrix and the  $R$  refers to a regular matrix. Two variants were implemented and analyzed for each  $\mathbf{H}$ : “Without Token Ring”, which does not include clock gating and token forwarding, and a “With Token Ring” variant, which includes a distributed control system, an architecture based on a Token Ring with token passing and clock gating. It is worth noting a significant reduction of the  $P_d$  (dynamic energy consumption) for the architecture based on a Token Ring (on average by 40%), with a simultaneous slight increase in the demand for ALM units (on average by 9%). It is worth noting also that the architecture based on a Token Ring does not increase the demand for memory bits (Mem).

Besides the reduction of power in the Token Ring solution, several other observations can be made from the results in Table 2. Since an important decision in the LDPC coding system design is the choice between regular and irregular codes, it is valuable to compare the energy consumption for these two cases. We have observed a clear difference in energy consumption for regular and irregular decoders in the proposed implementation. In Table 3, we emphasize results for two selected matrices, regular and irregular, which are direct counterparts; that is, they have the same size and the overall number of nonzero elements ( $Z = 3072$ ). In this table, we also include  $L_c$ —the number of GCLK ticks needed to perform one complete decoding iteration and  $E_d$ —the average dynamic energy per iteration. As can be seen, a QC-LDPC decoder of a regular code requires slightly more clock cycles  $L_c$  than an irregular counterpart. This is due to the possibility of faster execution of computations through their parallelization, resulting from a different construction of both matrices  $\mathbf{H}$  in the form of QC: irregular column weights allow for faster computations for a portion of rows or columns. Due to the different values of  $L_c$ , it is improper to compare the two decoders on the basis of  $P_d$ ; and it is valuable to determine the average dynamic energy  $E_d$ . Comparing  $E_d$  reveals that the decoder based on an irregular matrix is characterized by a much higher energy consumption (more by 168%) compared to the decoder based on the regular matrix. Table 3 also shows an increased hardware demand for ALM units (44% more) in irregular codes decoder. This shows a tradeoff between coding performance (favorable for irregular codes) and energy consumption (favorable for regular codes).

We have also determined the power loss in particular decoded elements. The results are shown in Table 4. They allow for a comparison of  $P_d$  in regular and irregular decoders. Significant differences can be seen, in particular for the control node computing units, bit node computing units and the clock tree being the power loss of the clock tree designed in the QC-LDPC decoder (for each FPGA device). In relation to the above, it can be concluded that especially regular matrices decoded in a Token Ring architecture can be perceived as an energy-saving QC-LDPC decoding framework.

The dynamic power loss  $P_d$  and the hardware utilization (ALM and Mem) are also dependent on the size  $P$  of the submatrix. Table 5 presents a comparison of the decoders for the same matrix  $\mathbf{H}$  size but different values of  $P$ . The submatrix size  $P$  corresponds to the number of processing units (see Figure 4), and at the same time it affects the number of clock cycles  $L_c$ . As a result, the decoder throughput, but also power loss, is strongly affected by  $P$ . On the other hand, the average energy consumption  $E_d$  is supposed to be independent of  $P$ . However, we still observed  $E_d$  dependent to some extent on  $P$ , which reveals the energy efficiency of the proposed implementation is dependent on  $P$ .

**Table 2.** Comparison of implementations of QC-LDPC decoders with and without architecture based on a Token Ring.

Parity Check Matrix H				Without “Token Ring”			With “Token Ring”			Differences	
Type	$M \times N$	$P$	$Z$	$P_d$ [mW]	ALM	Mem	$P_d$ [mW]	ALM	Mem	$P_d$ [%]	ALM [%]
I	336 × 672	42	2184	201.95	7568	24,424	119.92	7928	24,424	−40.619	4.7569
I	512 × 1024	64	3072	289.63	11,336	34,064	178.24	11,868	34,064	−38.4594	4.693
I	512 × 1024	64	3200	301.8	12,906	35,264	196.55	13,425	35,264	−34.8741	4.0214
I	512 × 1024	64	3264	312.1	13,531	35,864	201.04	14,065	35,864	−35.5847	3.9465
I	1152 × 2304	96	7296	445.67	19,227	79,528	261.08	20,006	79,528	−41.4185	4.0516
R	256 × 512	64	1536	122.37	6997	16,912	64	8169	16,912	−47.8615	16.75
R	384 × 768	32	2304	120.17	3724	26,304	67.8	3994	26,304	−43.5799	7.2503
R	416 × 832	32	2496	122.37	3716	28,496	72.43	3978	28,496	−40.8107	7.0506
R	448 × 896	32	2688	118.68	3723	30,688	73.8	3988	30,688	−37.816	7.1179
R	480 × 960	32	2880	118.22	3732	32,880	74.04	3991	32,880	−37.371	6.94
R	512 × 1024	8	3072	22.95	887	41,984	15.88	1092	41,984	−30.8061	23.1116
R	512 × 1024	64	3072	114.34	7029	33,920	66.41	8199	33,920	−41.9188	16.6453
R	544 × 1088	32	3264	116.7	3641	37,264	68.52	3972	37,264	−41.2853	9.0909
R	608 × 1216	32	3648	116.71	3641	41,648	69.12	3972	41,648	−40.7763	9.0909
R	576 × 1152	4	3456	40.99	598	57,600	28.41	712	57,600	−30.6904	19.0635
R	576 × 1152	8	3456	44.84	979	47,232	27.64	1119	47,232	−38.3586	14.3003
R	576 × 1152	18	3456	91.05	2183	41,856	44.37	2359	41,856	−51.2685	8.0623
R	576 × 1152	24	3456	101.43	2837	40,608	55.59	3054	40,608	−45.1937	7.6489
R	576 × 1152	32	3456	116.7	3641	39,456	75	3987	39,456	−35.7326	9.5029
R	576 × 1152	48	3456	178.97	5580	38,736	99.71	6085	38,736	−44.2868	9.0502
R	576 × 1152	64	3456	230.15	7883	38,160	124.91	8210	38,160	−45.7267	4.1482
R	576 × 1152	96	3456	343.69	11,822	37,800	181.45	12,165	37,800	−47.2053	2.9014

**Table 3.** Comparison of energy efficiency of decoders based on regular and irregular matrices for  $M \times N = 512 \times 1024$ ,  $P = 64$ ,  $Z = 3072$  and  $f = 400$  MHz.

Type	$L_c$	$P_d$ [mW]	$E_d$ [nJ]	ALM	Mem
I	416	178.24	185.36	11,868	34,064
R	440	66.41	73.05	8199	33,920

**Table 4.** Comparison of energy losses of elements of regular and irregular decoders.

Decoder Element	$P_d$ [mW]	
	Irregular	Regular
Communication support module	0.19	0.48
Memory $L(c y)$	12.77	5.21
Initialization module	6.03	2.03
Memory $Q$	0.27	0.42
Control node computing unit	31.64	17.99
Memory $R$	0.13	0.17
Bit node computing unit	47.46	12.84
Memory $X$	4.5	2.62
Verification of control equations module	3.76	1.1
Communicating the results module	0.05	0.05
Clock tree	71.44	23.5

**Table 5.** Analysis of the impact of  $P$  on energy consumption and hardware demand ( $f = 400$  MHz).

$P$	Without “Token Ring”				With “Token Ring”		
	$L_c$	ALM	$P_d$ [mW]	$E_d$ [nJ]	ALM	$P_d$ [mW]	$E_d$ [nJ]
4	7920	598	40.99	811.602	712	28.41	562.518
8	3960	979	44.84	443.916	1119	27.64	273.636
18	1760	2183	91.05	400.62	2359	44.37	195.228
24	1320	2837	101.43	334.719	3054	55.59	183.447
32	990	3641	116.7	288.8325	3987	75	185.625
48	660	5580	178.97	295.3005	6085	99.71	164.5215
64	495	7883	230.15	284.8106	8210	124.91	154.5761
96	330	11,822	343.69	283.5443	12,165	181.45	149.6963

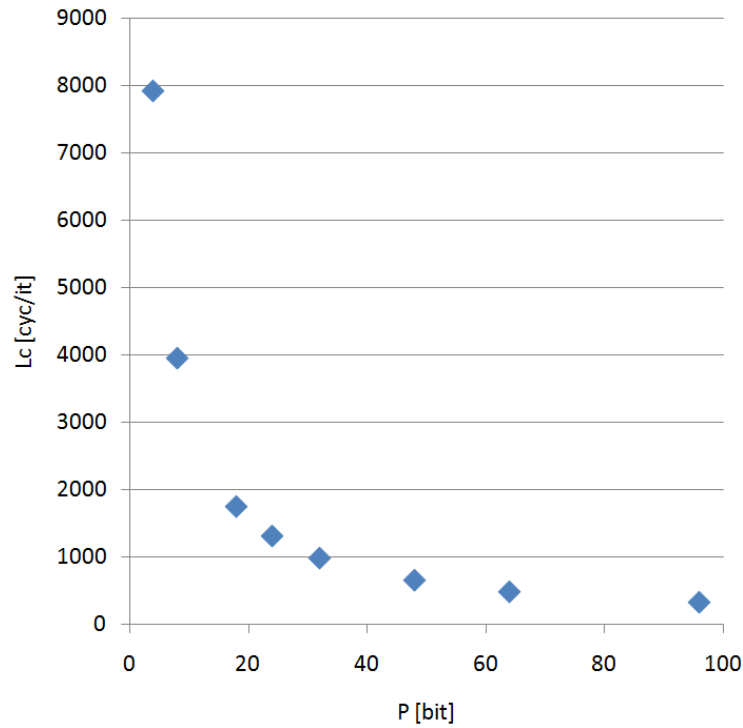
Results presenting the mentioned dependencies are shown in Table 5 as well as Figures 6–8, where “[cyc/it]” denotes the number of ticks of a global clock needed to perform one decoding iteration. Figure 6 shows the dependence of  $L_c$  on the value of  $P$ . Figure 7 shows the dependence of the hardware demand for ALM units on the value of  $P$ . One can see an approximately linear increase in hardware requirements with increasing values of  $P$ . Figure 8 shows the dynamic energy consumption  $E_d$  depends on the value of  $P$ . It can be seen that despite the increase in  $P_d$  with increasing  $P$ , the  $E_d$  significantly decreases with  $P$ . This means that a semi-parallel decoder with a high degree of parallelization of computations will be characterized by a greater energy-efficiency in comparison to a decoder with a lower degree of parallelization. In view of the above, it can be seen that the correct selection of  $P$  for the design requirements is an important system design choice in terms of hardware requirements and dynamic energy losses.

The presented architecture should finally be compared with the results from the literature. This is very complicated due to the enormous number of possible parity check matrices  $\mathbf{H}$  as well as the decoder design configurations. Even standards (like IEEE802.11ad, WiGig) are characterized by a large number of utilized matrices  $\mathbf{H}$  upon which the QC-LDPC decoder can be designed. The most popular architectures in the literature are parallel architectures, which also pose a significant difficulty when trying to make a comparison, although in engineering practice the highest decoding speed is not always required. Nevertheless, an attempt was made to compare the proposed Token Ring-based architecture for the WiGig standard code. This is presented in Table 6, where NMS means Normalized Min-Sum decoder algorithm, which is a slightly modified decoding computation method in control nodes of the presented Min-Sum algorithm [37].

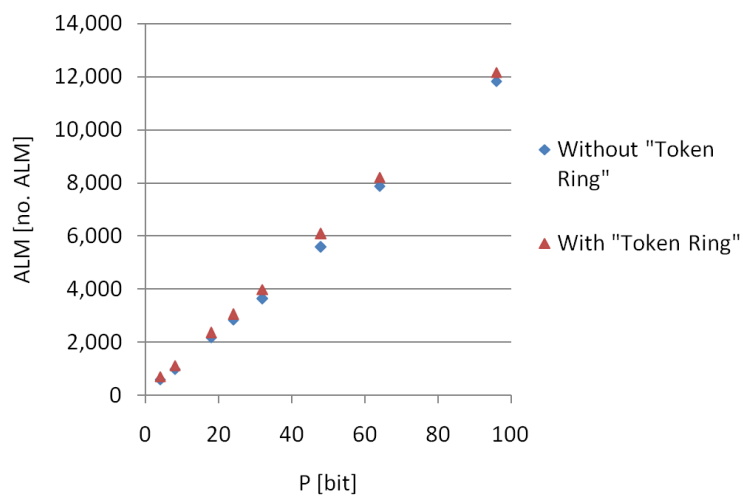
**Table 6.** Comparison of IEEE802.11ad standard decoders.

Work	This Work	[43]	[44]	[45]	[46]
Voltage (V)	1.1	0.9	1.1	1	1.1
Core area mm <sup>2</sup>	—	1.99	0.575	1.56	0.8
Total memory (bits)	24,424	160,727	7875	—	—
Clock frequency (MHz)	400	202	400	197	220
Block length (bits)	672	672	672	672	672
Code rate	1/2	1/2	1/2	1/2, 5/8, 3/4, 7/8	1/2, 5/8, 3/4, 13/16
Throughput (Gb/s)	0.611	6.78	9.25	5.79	6.16
Measured power (mW)	119.92	289	272.9	361	203
Decoding algorithm	MS	MS	MS	NMS	MS
Message quantization (bits)	4	5	4	6	5

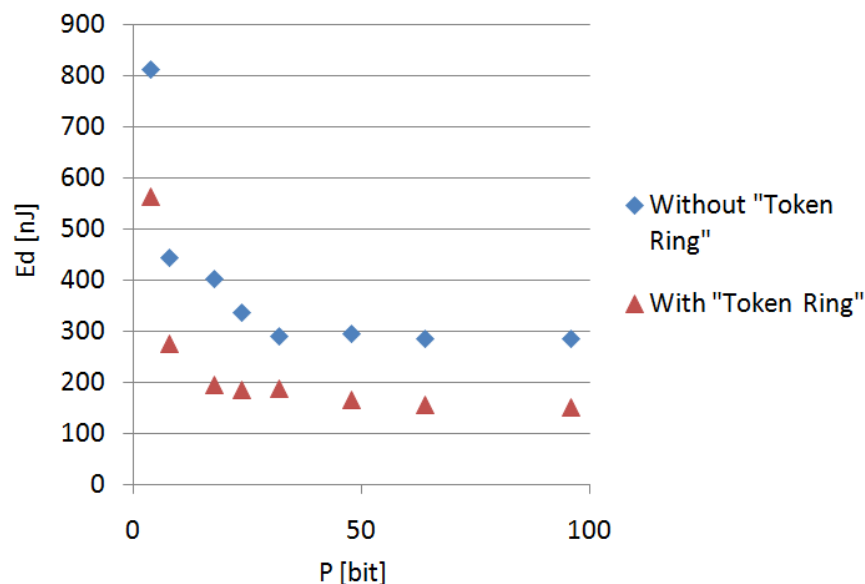
The design proposed in this article will be recommended in the systems when the maximum possible throughput is not critical, while maintaining a reduced  $P_d$  value, which has a direct impact on the temperature generated in the integrated circuit.



**Figure 6.** Influence of the  $P$  value on the  $L_c$  value for a QC-LDPC decoder with an architecture based on Token Ring.



**Figure 7.** Influence of the  $P$  value on the ALM value for a QC-LDPC decoder with an architecture based on Token Ring.



**Figure 8.** Influence of the  $P$  value on the  $E_d$  value for a QC-LDPC decoder with an architecture based on Token Ring.

## 5. Conclusions

This article presents an innovative idea of the QC-LDPC decoder design using the Token Ring architecture. The proposed architecture is oriented to reduce energy consumption. A significant reduction in dynamic energy consumption has been demonstrated with the decoder throughput remaining unchanged. Moreover, the analysis of the implementation results of decoders based on regular and irregular matrices  $\mathbf{H}$ , as well as matrices with different submatrix sizes, allowed us to indicate an energy-saving configuration. Such a configuration is the regular code decoder with a sufficiently large submatrix size  $P$ . It is likely that similar conclusions can be drawn for other QC-LDPC decoder designs, but we cannot claim that for sure.

The architecture based on a Token Ring can undoubtedly be used in the implementations of QC-LDPC decoders as well as in many other modern digital systems. In the proposed solution, which is not optimized in terms of throughput, significantly lower losses of dynamic power were achieved combined with lower throughput. This type of implementation can be applied wherever the throughput is not a critical parameter of the designed device.

The obtained results of the experiments indicate the general principles of selecting the  $\mathbf{H}$  matrix form, which has a positive effect on the power consumption of the designed decoder. It cannot be denied that it is difficult to judge whether the obtained results are universal in nature, or only apply to a specific family of FPGAs. It seems that they should be universal in nature, but confirming this would require experiments for several families of FPGAs. A comparison of the designed decoder with alternative solutions taken from the literature shows that the proposed solution has high implementation potential. It seems that it is also worth using them in ASICs, which do not have a number of the limitations characteristic of programmable systems.

## 6. Future Work

The results of the work presented in the article encourage further research. We plan to perform similar experiments first for several other FPGA families, both from Intel and Xilinx. We hope that they will allow us to confirm the thesis about the universal nature of the proposed solutions and their suitability for a wide range of programmable systems.



The second area of research will be the search for a way to use the reconfiguration of the programmable structure in order to optimize the parameters of the low-power LDPC decoder. The modern FPGA system can not only be statically reconfigured but also reconfigured during its operation, called dynamic reconfiguration. Such a decoder implementation, in which the circuit structure would be adapted to the currently decoded information, would make it possible to adapt the hardware solution to the specificity of the decoded information, providing application-appropriate correction parameters. Multi-context systems, i.e., those that have the ability to adapt their hardware structure to the currently performed task, are the object of research in various areas of application of programmable systems. It cannot be denied that the implementation of the multi-context LDPC code decoder, probably currently possible, is a real challenge. This challenge is behind our planned scientific and research work that we intend to conduct in the near future.

**Author Contributions:** Conceptualization, M.K., W.S. and D.K.; methodology, M.K., W.S. and D.K.; software, M.K., W.S. and D.K.; validation, M.K., W.S. and D.K.; formal analysis, M.K., W.S. and D.K.; investigation, M.K., W.S. and D.K.; resources, M.K., W.S. and D.K.; data curation, M.K., W.S. and D.K.; writing—original draft preparation, M.K., W.S. and D.K.; writing—review and editing, M.K., W.S. and D.K.; visualization, M.K., W.S. and D.K.; supervision, M.K., W.S. and D.K.; project administration, M.K., W.S. and D.K.; funding acquisition, M.K., W.S. and D.K. All authors have read and agreed to the published version of the manuscript.

**Funding:** The work reported in the paper is partially supported by the European Social Funds; project no POWR.03.02.00-00-I007/17-00 “CyPhiS—the program of modern PhD studies in the field of Cyber-Physical Systems” and financed with university funds (BKM).

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

## Abbreviations

The following abbreviations are used in this manuscript:

ALM	Adaptive Logic Module
BER	Bit Error Rate
BP	Belief Propagation
CPM	Circulant Permutation Matrix
FER	Frame Error Rate
FPGA	Field Programmable Gate Array
GALS	Globally Asynchronous Locally Synchronous
GCLK	Global Clock Signal
LAN	Local Area Network
LCLK	Local Clock Signal
LDPC	Low-Density Parity-Check
LLR	Log-Likelihood Ratio
MS	Min-Sum
NMS	Normalized Min-Sum
QC-LDPC	Quasi-Cyclic Low-Density Parity-Check
VLSI	Very Large Scale Integration

The following notations are used in this manuscript:

$:=$	symbol means “becomes”
$K$	number of information bits
$M$	number of redundant code bits
$N$	the sum of the number of information bits and redundant code bits (block length)
$R$	code rate
$\mathbf{H}$	low density parity check matrix
$X$	decoded vector

$\mathbf{P}$	square submatrices
$P$	the size of the $\mathbf{P}$ submatrices
$\mathbf{c} = [c_1, c_2, \dots, c_N]$	code vector
$\mathbf{u} = \{u_1, u_2, \dots, u_K\}$	information vector
$\mathbf{y} = [y_1, y_2, \dots, y_N]$	received vector
$Q_{nm}$	belief LLR value from the $n$ -th bit vertex to the $m$ -th Tanner graph control vertex
$L(x_n y_n)$	LLR a priori probabilities for the $n$ -th bit
$(1, N)$	is a set of integers between 1 and $N$
$\mathcal{N}(m)$	a set of column indexes in the parity check matrix $\mathbf{H}$ containing one in the $m$ -th row
$\mathcal{M}(n)$	a set of row indexes in the parity check matrix $\mathbf{H}$ containing one in the $n$ -th column
$R_{mn}$	message from the $m$ -th control vertex to the $n$ -th bit vertex of Tanner graph
$X_n$	the $n$ -th value of the decoded vector

## References

- Kaczer, B.; Degraeve, R.; Pangon, N.; Groeseneken, G. The influence of elevated temperature on degradation and lifetime prediction of thin silicon-dioxide films. *IEEE Trans. Electron Devices* **2000**, *47*, 1514–1521. [[CrossRef](#)]
- Ridhawi, I.A.; Kotb, Y.; Aloqaily, M.; Jararweh, Y.; Baker, T. A Profitable and Energy-Efficient Cooperative Fog Solution for IoT Services. *IEEE Trans. Ind. Inform.* **2020**, *16*, 3578–3586. [[CrossRef](#)]
- Raffin, E.; Nogues, E.; Hamidouche, W.; Tomperi, S.; Pelcat, M.; Menard, D. Low power HEVC software decoder for mobile devices. *J. Real-Time Image Process.* **2016**, *12*, 495–507. [[CrossRef](#)]
- Gallager, R.G. *Low-Density Parity-Check Codes*; MIT Press: Cambridge, MA, USA, 1963.
- MacKay, D.J.C. Good error-correcting codes based on very sparse matrices. *IEEE Trans. Inf. Theory* **1999**, *45*, 399–431. [[CrossRef](#)]
- Kim, S.; Park, C.; Hwang, S. A novel partially parallel architecture for high-throughput LDPC Decoder for DVB-S2. *IEEE Trans. Consum. Electron.* **2010**, *56*, 820–825. [[CrossRef](#)]
- Papaharalabos, S.; Papaleo, M.; Mathiopoulos, P.T.; Neri, M.; Vanelli-Coralli, A.; Corazza, G.E. DVB-S2 LDPC Decoding Using Robust Check Node Update Approximations. *IEEE Trans. Broadcast.* **2008**, *54*, 120–126. [[CrossRef](#)]
- Zhang, Z.; Anantharam, V.; Wainwright, M.J.; Nikolic, B. An Efficient 10GBASE-T Ethernet LDPC Decoder Design With Low Error Floors. *IEEE J. Solid-State Circuits* **2010**, *45*, 843–855. [[CrossRef](#)]
- Cohen, A.E.; Parhi, K.K. A Low-Complexity Hybrid LDPC Code Encoder for IEEE 802.3an (10GBase-T) Ethernet. *IEEE Trans. Signal Process.* **2009**, *57*, 4085–4094. [[CrossRef](#)]
- Liu, C.; Yen, S.; Chen, C.; Chang, H.; Lee, C.; Hsu, Y.S.; Jou, S.J. An LDPC Decoder Chip Based on Self-Routing Network for IEEE 802.16e Applications. *IEEE J. Solid-State Circuits* **2008**, *43*, 684–694. [[CrossRef](#)]
- Masera, G.; Quaglio, F.; Vacca, F. Implementation of a Flexible LDPC Decoder. *IEEE Trans. Circuits Syst. II Express Briefs* **2007**, *54*, 542–546. [[CrossRef](#)]
- Li, M.; Lee, Y.; Huang, Y.; der Perre, L.V. Area and energy efficient 802.11ad LDPC decoding processor. *Electron. Lett.* **2015**, *51*, 339–341. [[CrossRef](#)]
- Pisek, E.; Rajan, D.; Cleveland, J.R. Trellis-Based QC-LDPC Convolutional Codes Enabling Low Power Decoders. *IEEE Trans. Commun.* **2015**, *63*, 1939–1951. [[CrossRef](#)]
- Jin, J.; Tsui, C. An Energy Efficient Layered Decoding Architecture for LDPC Decoder. *IEEE Trans. Very Large Scale Integr. Syst.* **2010**, *18*, 1185–1195. [[CrossRef](#)]
- Karimi, M.; Banihashemi, A.H. Counting Short Cycles of Quasi Cyclic Protograph LDPC Codes. *IEEE Commun. Lett.* **2012**, *16*, 400–403. [[CrossRef](#)]
- Alcorn, G.; Scott, S.; Figueroa, O.; Watkins, M. *Low Density Parity Check Code for Rate 7/8*; NASA: Washington, DC, USA, 2017.
- Zhang, Y.; Peng, K.; Wang, X.; Song, J. Performance Analysis and Code Optimization of IDMA With 5G New Radio LDPC Code. *IEEE Commun. Lett.* **2018**, *22*, 1552–1555. [[CrossRef](#)]
- Li, H.; Bai, B.; Mu, X.; Zhang, J.; Xu, H. Algebra-Assisted Construction of Quasi-Cyclic LDPC Codes for 5G New Radio. *IEEE Access* **2018**, *6*, 50229–50244. [[CrossRef](#)]

19. Altera. *Reducing Power Consumption and Increasing Bandwidth on 28-nm FPGAs*; Altera Corporation: San Jose, CA, USA, 2012.
20. Ashenafi, E.; Chowdhury, M.H. A New Power Gating Circuit Design Approach Using Double-Gate FDSOI. *IEEE Trans. Circuits Syst. II Express Briefs* **2018**, *65*, 1074–1078. [[CrossRef](#)]
21. Li, L.; Choi, K.; Nan, H. Activity-Driven Fine-Grained Clock Gating and Run Time Power Gating Integration. *IEEE Trans. Very Large Scale Integr. Syst.* **2013**, *21*, 1540–1544. [[CrossRef](#)]
22. Greenberg, S.; Rabinowicz, J.; Tschanski, R.; Paperno, E. Selective State Retention Power Gating Based on Gate-Level Analysis. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2014**, *61*, 1095–1104. [[CrossRef](#)]
23. Mahmoodi, H.; Tirumalashetty, V.; Cooke, M.; Roy, K. Ultra Low-Power Clocking Scheme Using Energy Recovery and Clock Gating. *IEEE Trans. Very Large Scale Integr. Syst.* **2009**, *17*, 33–44. [[CrossRef](#)]
24. Wimer, S.; Koren, I. The Optimal Fan-Out of Clock Network for Power Minimization by Adaptive Gating. *IEEE Trans. Very Large Scale Integr. Syst.* **2012**, *20*, 1772–1780. [[CrossRef](#)]
25. Bezati, E.; Casale-Brunet, S.; Mattavelli, M.; Janneck, J.W. Clock-Gating of Streaming Applications for Energy Efficient Implementations on FPGAs. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2017**, *36*, 699–703. [[CrossRef](#)]
26. Yang, J.-L.; Wong, A.K.K. Designing of precomputational-based low-power Viterbi decoder. In Proceedings of the IEEE 6th Circuits and Systems Symposium on Emerging Technologies: Frontiers of Mobile and Wireless Communication (IEEE Cat. No. 04EX710), Shanghai, China 31 May–2 June 2004; Volume 2, pp. 603–606. [[CrossRef](#)]
27. Ruan, S.; Wu, C.; Hsieh, J. Low Power Design of Precomputation-Based Content-Addressable Memory. *IEEE Trans. Very Large Scale Integr. Syst.* **2008**, *16*, 331–335. [[CrossRef](#)]
28. Lin, C.-S.; Chang, J.-C.; Liu, B.-D. A low-power precomputation-based fully parallel content-addressable memory. *IEEE J. Solid-State Circuits* **2003**, *38*, 654–662. [[CrossRef](#)]
29. Hachtel, G.D.; Macii, E.; Pardo, A.; Somenzi, F. Markovian analysis of large finite state machines. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **1996**, *15*, 1479–1493. [[CrossRef](#)]
30. Benini, L.; Micheli, G.D.; Liyo, A.; Macii, E.; Odasso, G.; Poncino, M. Synthesis of power-managed sequential components based on computational kernel extraction. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2001**, *20*, 1118–1131. [[CrossRef](#)]
31. Sudnitson, A. Computational kernel extraction for synthesis of power-managed sequential components. In Proceedings of the 9th International Conference on Electronics, Circuits and Systems, Dubrovnik, Croatia, 15–18 September 2002; Volume 2, pp. 749–752. [[CrossRef](#)]
32. Chong, K.; Chang, K.; Gwee, B.; Chang, J.S. Synchronous-Logic and Globally-Asynchronous-Locally-Synchronous (GALS) Acoustic Digital Signal Processors. *IEEE J. Solid-State Circuits* **2012**, *47*, 769–780. [[CrossRef](#)]
33. Beigne, E.; Vivet, P.; Thonnart, Y.; Christmann, J.; Clermidy, F. Asynchronous Circuit Designs for the Internet of Everything: A Methodology for Ultralow-Power Circuits with GALS Architecture. *IEEE Solid-State Circuits Mag.* **2016**, *8*, 39–47. [[CrossRef](#)]
34. Bertozzi, D.; Miorandi, G.; Ghiribaldi, A.; Bursleson, W.; Sadowski, G.; Bhardwaj, K.; Jiang, W.; Nowick, S.M. Cost-Effective and Flexible Asynchronous Interconnect Technology for GALS Systems. *IEEE Micro* **2020**. [[CrossRef](#)]
35. Kim, S.; No, J.S.; Chung, H.; Shin, D.J. Girth analysis of Tanner’s (3, 5) QC LDPC codes. *Int. Symp. Inf. Theory* **2005**, 1632–1636. [[CrossRef](#)]
36. Tanner, R. A recursive approach to low complexity codes. *IEEE Trans. Inf. Theory* **1981**, *27*, 533–547. [[CrossRef](#)]
37. Kuc, M.; Sułek, W.; Kania, D. FPGA-Oriented LDPC Decoder for Cyber-Physical Systems. *Mathematics* **2020**, *8*, 723. [[CrossRef](#)]
38. 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](#)]
39. Zhang, J.; Fossorier, M.; Gu, D.; Zhang, J. Two-dimensional correction for Min-Sum decoding of irregular LDPC codes. *IEEE Commun. Lett.* **2006**, *10*, 180–182. [[CrossRef](#)]

40. *IEEE Standard for Information Technology—Telecommunications and Information Exchange between Systems—Local and Metropolitan Area Networks—Part 5: Token Ring Access Method and Physical Layer Specifications 100 Mbit/s Dedicated Token Ring Operation*; IEEE: Piscataway, NJ, USA, 2000; pp. 1–288. [[CrossRef](#)]
41. Zhu, K.; Wong, D.F. Clock skew minimization during FPGA placement. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **1997**, *16*, 376–385. [[CrossRef](#)]
42. Intel. *Cyclone V Device Overview, CV-51001*; Intel: Santa Clara, CA, USA, 2018.
43. Milicevic, M.; Gulak, P.G. A Multi-Gb/s Frame-Interleaved LDPC Decoder With Path-Unrolled Message Passing in 28-nm CMOS. *IEEE Trans. Very Large Scale Integr. Syst.* **2018**, *26*, 1908–1921. [[CrossRef](#)]
44. Ajaz, S.; Lee, H. Multi-Gb/s multi-mode LDPC decoder architecture for IEEE 802.11ad standard. In Proceedings of the 2014 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), Ishigaki, Japan, 17–20 November 2014; pp. 153–156. [[CrossRef](#)]
45. Yen, S.; Hung, S.; Chen, C.; Chang, H.; Jou, S.; Lee, C. A 5.79-Gb/s Energy-Efficient Multirate LDPC Codec Chip for IEEE 802.15.3c Applications. *IEEE J. Solid-State Circuits* **2012**, *47*, 2246–2257. [[CrossRef](#)]
46. Motozuka, H.; Yosoku, N.; Sakamoto, T.; Tsukizawa, T.; Shirakata, N.; Takinami, K. A 6.16 Gb/s 4.7 pJ/bit/iteration LDPC decoder for IEEE 802.11ad standard in 40 nm LP-CMOS. In Proceedings of the 2015 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Orlando, FL, USA, 14–16 December 2015; pp. 1289–1292. [[CrossRef](#)]

**Publisher’s Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2020 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 (<http://creativecommons.org/licenses/by/4.0/>).