electronics

MDPI

Article

A Data Compression Hardware Accelerator Enabling Long-Term Biosignal Monitoring Based on Ultra-Low Power IoT Platforms

Christos P. Antonopoulos * and Nikolaos S. Voros

Computer & Informatics Engineering Department, Technological Educational Institute of Western Greece, 30020 Antirio, Greece; voros@teiwest.gr

* Correspondence: cantonopoulos@teiwest.gr; Tel.: +30-693-610-6468

Received: 31 May 2017; Accepted: 27 July 2017; Published: 31 July 2017

Abstract: For highly demanding scenarios such as continuous bio-signal monitoring, transmitting excessive volumes of data wirelessly comprises one of the most critical challenges. This is due to the resource limitations posed by typical hardware and communication technologies. Driven by such shortcomings, this paper aims at addressing the respective deficiencies. The main axes of this work include (a) data compression, and (b) the presentation of a complete, efficient and practical hardware accelerator design able to be integrated in any Internet of Things (IoT) platform for addressing critical challenges of data compression. On one hand, the developed algorithm is presented and evaluated on software, exhibiting significant benefits compared to respective competition. On the other hand, the algorithm is fully implemented on hardware providing a further proof of concept regarding the implementation feasibility with respect to state-of-the art hardware design approaches. Finally, system-level performance benefits, regarding data transmission delay and energy saving, are highlighted, taking into consideration the characteristics of prominent IoT platforms. Concluding, this paper presents a holistic approach based on data compression that is able to drastically enhance an IoT platform's performance and tackle efficiently a notorious challenge of highly demanding IoT applications such as real-time bio-signal monitoring.

Keywords: IoT Wireless Sensor Network platforms; data compression; hardware accelerator; Wireless Sensor Networks; embedded systems; complete solution; experimental evaluation; hardware design; ultra-low power

1. Introduction

Internet of Things (IoT) short-range, ultra-low power communication technologies comprise one of the most rapidly evolving research areas attracting significant interest both from academia and the industry [1]. Consequently, respective communication protocols as well as platforms have emerged as prominent infrastructure upon which future Cyber-Physical Systems (CPS) can be based. A major factor for this growth can be attributed to the novel communication paradigm, through which relative communication approaches have been introduced enabling flexible communication, distributed operation and rapid deployment without the need for any pre-existing infrastructure, mobility support, low-power functionality and many more critical features. Additionally, IoT and CPS have significantly benefited from the advancements in hardware and Very Large Scale Integration (VLSI) design leading to very low cost, complexity, size and, most importantly, power consumption embedded systems able to be used as a suitable hardware infrastructure [2–4]. Furthermore, another aspect related to hardware design, concerns the anticipated advantages yielded by the hardware accelerators. The latter, comprise highly specialized hardware components integrated on general-purpose IoT platforms, dedicated to perform highly complex operations in the most efficient way. Such approach offers multifaceted benefits since the design of nodes can exhibit significantly higher degree of dynamicity and modularity, tailored to specific applications. At the same time, demanding operations are performed with maximum efficiency, while the main processing unit of the nodes is not overwhelmed.

A consequence of the aforementioned benefits is that the relative application domain has significantly expanded from simple data processing and transmission scenarios to increasingly demanding ones. Respective examples entail significant processing workload (e.g., medical applications), the support of complex communication standards (e.g., IPv6 routing protocol support enabling true IoT deployment), and the cooperative functionality among nodes in order to achieve a common goal etc. Of course, aspects such as time-constrained communication and power aware operation impose stringent requirements. In order to meet the respective requirements, researchers must surpass the challenge of the extremely limited resource availability encountered in all nowadays Wireless Sensor Networks (WSN) platforms. The respective limitation ranges from low processing power offered by typical low-end micro-controller units, to limited communication bandwidth offered by protocols like IEEE 802.15.4 or Bluetooth, as well as scarce availability in terms of memory (few tens of Kbytes) and energy. Concerning the latter limitation (i.e., energy availability), emphasis must be put on the fact that the wireless radio interface comprises the main power consumption factor, which highlights the respective deficiencies. Therefore, a common critical objective in various pragmatic approaches, aiming towards lifetime extension, is the deactivation of the relative hardware when respective functionality is not required for extended periods of time [5,6].

In this paper, the authors tackle the aforementioned deficiencies stemming from scarce bandwidth and energy availability, which in conjunction with demanding applications comprise a notorious challenge. As it will be analyzed in the following section, real-time analysis of biomedical modalities requires excessive volume of data transmission, forming two challenging side effects. On one hand, it is easily quantifiable that in many cases the bandwidth offered cannot handle the required data rate. On the other hand, even in cases where the channel capacity is adequate for data creation rates, the respective scenarios lead to continuous operation of highly power consuming components, and especially the wireless radio interface. Respective cases contradict typical WSN platform paradigm, which effectively dictates the transition of power-hungry hardware to low-power states for substantial periods of time.

Driven by such conditions, this paper proposes a holistic solution that allows mitigating the respective side effects based on a highly efficient and resource conservative data compression hardware accelerator. Specifically, the critical contribution of this work is the design, development and system level evaluation of the potential benefits of the proposed hardware component taking into consideration realistic, demanding biomedical datasets as well as real performance characteristics of state-of-the-art IoT platform. As it will be presented, the developed compression algorithm and its hardware design is able to compress the targeted modalities on-the-fly. Additionally, it is able to yield a considerable compression rate in all cases, thus enhancing the transmission capabilities of an IoT platform. Furthermore, the proposed hardware accelerator leads to drastic power conservation due to the data volume reduction, assuming that IoT nodes are able to deactivate radio interface when not transmitting. To the best of the authors' knowledge, this is the first attempt to offer a practical, efficient and feasible complete IoT compression component going beyond proposing an isolated software compression approach. On the contrary, the proposed solution has been implemented in hardware and evaluated taking into account realistic wireless transmission delays and power consumption statistics of prominent IoT wireless communication interfaces.

The rest of the paper is structured as follows: In Section 2 critical information is provided highlighting the application scenarios, relative literature and the data characteristics of the specific problem. Section 3 presents the design and evaluation of the proposed compression algorithm yielding the required characteristics so as to optimally address the targeted problem. Section 4 comprises the cornerstone of this work and presents the actual hardware module design, implementation and evaluation. Section 5 offers a critical system level performance evaluation of the proposed hardware accelerator assuming that it has been integrated in prominent real IoT platforms and two different wireless communication interfaces. Finally, Section 6 offers a summarizing discussion on performance evaluation and module's implementation capabilities while Section 7 highlights the main points of this work and offers significant future extensions based on the presented work.

2. Background Information

2.1. Relative WSN Compression Approaches

Based on the previous analysis, an elicitation concerning adequate compression algorithms for epilepsy monitoring and utilization in WSN networks is conducted. The elicitation process followed a multifaceted approach considering (a) the nature of the application scenario the authors focused on, which is electrocardiogram/electroencephalogram (ECG/EEG) physiological signal monitoring, (b) the representation of the digitized data as time-series and (c) the performance aspects taking into consideration the utilization in on-the-fly WSN scenarios. With respect to the first characteristic, the criticality and the accuracy required for the monitoring and study of epileptic people dictates zero tolerance to datum corruption due to the compression process. Consequently, only lossless compression algorithms suitable for wireless sensors have been considered [7,8] as opposed to lossy approaches. Regarding the second parameter, we focused on algorithms well known for their effectiveness on time-series datasets [7–11] taking into consideration that both Electroencephalography (EEG) and Electrocardiography (ECG) result into time-series datasets. Finally, emphasizing on the utilization of such algorithms in the context of WSNs, our elicitation process focuses on low complexity in order to offer viable solutions for typical WSN nodes. In addition, it aims at minimizing the delay overhead and operating in a time constrained manner [10]. Taking into account a wide range of adequate compression approaches, the algorithms Lossless Entropy Compression (LEC) [8] and Adaptive Lossless Entropy Compression (ALEC) [10] were finally selected as the starting point for the hardware implementation. LEC is a low complexity lossless entropy compression algorithm resulting in a very small code footprint that requires very low computational power [12]. Its operation is based on utilizing a very small dictionary and exhibits impressive on-the-fly compression capabilities. Consequently, LEC is quite attractive for WSN deployments. The main steps of the algorithm include:

- Calculation of differential signal.
- Computing of the difference d_i between the binary representations r_i and $r_i - 1$ of the current and previous measurements respectively, encoding is applied upon d_i resulting in the corresponding bit sequence bs_i.
- The sequence bs_i is then concatenated to the bit sequence stream generated so far.

The main processing complexity is attributed to the encoding phase of the compression, which aims at transforming d_i to bs_i bit sequences. During this process, firstly the number n_i of bits needed to encode the value of d_i is computed. Secondly, the first part of bs_i, indicated as s_i, is generated by using the table that contains the dictionary adopted by the entropy compressor. In that respect, in our initial evaluation JPEG algorithm has been adopted because the coefficients used in the JPEG have similar statistical characteristics to the measurements acquired by the sensing unit. In our implementation, the table has been extended so as to cover the necessary 16-bit resolution of the Analog to Digital Convertor (ADC) in the sensors. Thirdly, the second part of bs_i, indicated as a_i is the n_i low-order bits of d_i is calculated [8].

Focusing on the implementation features of LEC, it can be implemented by maintaining in memory only the column s_i of the aforementioned table. Overall, it can be easily extracted that LEC avoids any computationally intensive operation, which is highly appreciated in scarce resource WSN platforms. As a result, it exhibits very low execution delay facilitating real-time operation. However, basing its operation solely on a static lookup table, it does not support dynamic configuration capabilities with respect to the characteristics of a specific signal. The latter, is a significant drawback that negatively affects LEC's compression rate capabilities [12].

ALEC, on the other hand, is based on an adaptive lossless entropy compression approach also requiring low computational power. However, it uses three small dictionaries, the sizes of which are determined by the resolution of the analog-to-digital converter (ADC). Adaptive compression schemes allow the compression to dynamically adjust to the data source. The data sequences to be compressed are partitioned into blocks and for each block the optimal compression scheme is applied. Effectively, the algorithm is similar to LEC algorithm, while the main difference is that ALEC algorithm uses three Huffman coding tables instead of the one table used for the DC coefficients in JPEG algorithm. Specifically, ALEC uses adaptively two Huffman tables and three Huffman tables, respectively. It is noted that compared to LEC, ALEC exhibits an increased number of the lookup tables it employs. As a result, its compression-rate efficiency is also increased. However, each data block is passed through two lookup tables, which eventually will result in an increase of the algorithm's processing delay [12].

2.2. Application Scenarios and Data Characteristics

A critical objective of this effort concerns the performance enhancement offered by an efficient compression module to the state-of-the-art WSN platforms targeting highly demanding applications. Epilepsy monitoring based on IoT platform, represents the main targeted application scenario of this paper. From an engineering point of view, the respective application scenarios are based on acquiring excessive amounts of data (digitized physiological measurements), for extended periods of time, which must be either stored locally or transmitted to an aggregation point. The most demanding case, comprising the main objective of this paper, concerns the real-time monitoring of an epileptic person. This represents a critical requirement since epileptic events can occur unexpectedly and unpredictably, while the triggering event is highly personalized to the specific person. For that reason, periodic-based monitoring leads to myopic and unreliable results and conclusions. However, in both cases highly accurate signal monitoring is also required, which represents also a critical requirement towards effective compression.

WSNs comprise a rapidly evolving research area proposing a new communication paradigm making them appealing for such cases. However, in realistic scenarios the respective platforms suffer from critical resource limitations especially in areas such as energy availability, communication bandwidth and processing power that, at the same time, are interdependent resource consuming factors. More specifically, the operation of state-of-the-art WSN platforms is based on small batteries, typically offering energy capacity from 450 mAh (e.g., Shimmer platform) up to 3300 mAh (e.g., two AA batteries). Consequently, all components must offer low-power characteristics resulting in processing units with low processing capabilities (provided by 16-bit-based Micro-Controller Units) and limited available memory (in the area of 10 Kbyte RAM). Additionally, the data transfer is based on low-power wireless interfaces such as IEEE 802.15.4 and Bluetooth offering bandwidth capabilities at physical layer from 250 Kbps up to a few Mbps.

Taking into consideration the aforementioned application scenarios, the two modalities of paramount importance in epileptic seizure study, resulting into significant amount of accumulated data, are EEG and ECG measurements. Typical acquisition devices produce samples represented as 16-bit numbers. Furthermore, a wired EEG setup is usually comprised of 64 sensors with sampling frequency up to 2.5 kHz, while ECG typically requires 4 sensors with adequate sampling frequency of a few hundreds of Hertz. Given that the main application scenario targeted is the real-time monitoring of epileptic persons, even with some rough calculations, it can be derived that a setup of 64 EEG sensors requires higher wireless bandwidth (not considering packet headers and control data) than the one prominent communication technologies can provide. Specifically, assuming typical wireless technologies such as IEEE 802.15.4 and Bluetooth, such scenarios pose overwhelming burden to WSN platforms typically offering extremely limited resources [9,13–18].

Such workload scenarios demand that power-hungry components (such as the wireless interfaces or/and the processing unit) are continuously operating at their maximum power consumption operational state. Such behavior, however, contradicts to the main objective of typical WSNs of residing such components to low-power states for extended periods of time enabling drastic extension of the network lifetime and thus meet demanding requirements of many hours or days of monitoring.

Therefore, in such cases the reduction of the amount of data (i.e., by compressing them) can yield significant and multifaceted benefits to the system performance. On one hand, if we assume that a specific amount of data is compressed by $x\%$, the respective data volume reduction can be correlated to analogous reduction of radio utilization resulting in significant energy conservation. On the other hand, the $x\%$ compression percentage can also be envisioned as analogous reduction of bandwidth requirement and thus effectively increasing the limited wireless channel utilization while reducing the resulting data transfer delay.

3. Compression Algorithm Design

In this section, the proposed, and implemented, lossless entropy compression algorithm is presented, focusing on low computational power. Its main characteristic is the exploitation of the observed data's frequencies in the time series in order to offer an efficient dynamically adaptive Huffman table for the encoding process. The main novelty is based on extending the approaches derived from previously presented LEC and ALEC algorithms (i.e., low complexity, low code footprint, low compression delay). In this way, the proposed algorithm retains the very good processing delay performance characteristic. However, at the same time it significantly enhances achieved compression rate by increasing adaptability to the data characteristics. Consequently, compression rate exhibited by the proposed extension is highly competitive compared to respective solutions published in relative literature, while it also offers optimum compression rate vs compression processing delay trade-off. It is noted that a detailed analysis of the characteristics of this compression approach and evaluation at software level based on Matlab implementations can be found in [12].

3.1. Rationale of the Proposed Algorithm

LEC algorithm compresses data in its entropy encoder with the use of a fixed table leading to a very fast execution performance. This table is an extension of the table used in the JPEG algorithm to reach the size necessary for the resolution of the ADC in use. Additionally, it is based on the fact that the closer the absolute of a value is to zero more frequently it is observed in the differential signal.

However, it has been noticed that this frequency distribution may be valid for a file or a stream of data, however it is not always accurate considering fractions of the file or the data stream. Based on this observation ALEC algorithm uses a small amount of fixed Huffman Codes tables that can be alternatively used to produce smaller code for a packet of data. Furthermore, the specific table is not optimal for the specific data under test at each particular experiment.

Therefore, in the proposed scheme (as depicted in Figure 1) a novel approach is introduced, where the Huffman Codes lookup tables used are continuously adjusted according to the characteristics of the specific data used. Also the degree that the tables are adjusted is also configurable, offering fine tuning capabilities and enhancing the added value of the respective novel approach.

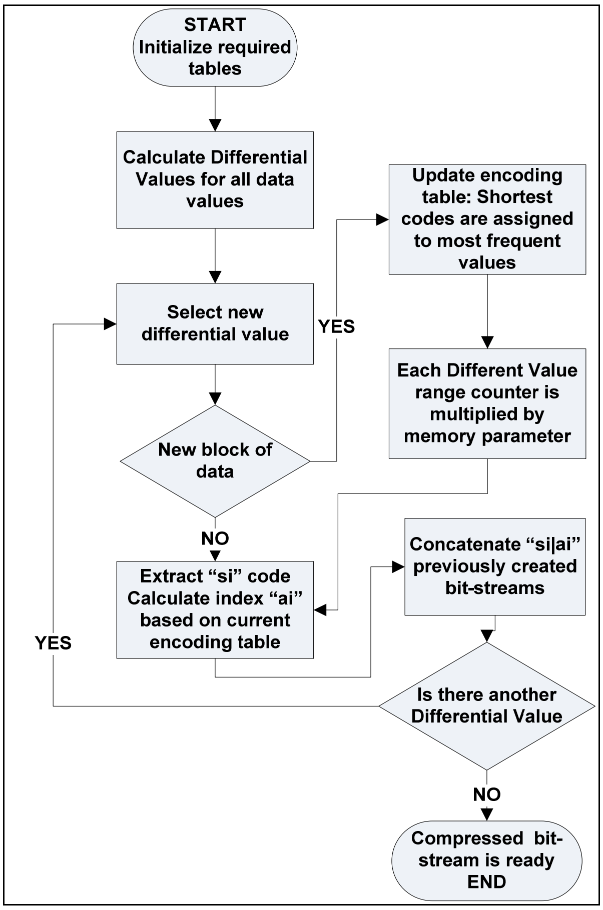
START
Initialize required tables
Calculate Differential Values for all data values
Select new differential value
New block of data
YES
Update encoding table: Shortest codes are assigned to most frequent values
Each Different Value range counter is multiplied by memory parameter
NO
Extract “si” code Calculate index “ai” based on current encoding table
Concatenate “si|ai” previously created bit-streams
Is there another Differential Value
YES
NO
Compressed bit-stream is ready END

Figure 1. Real-time huffman scheme flow chart.

3.2. Utilization of Data Statistical Knowledge

Usually statistical knowledge is available when time-series data are measured and transmitted through wireless sensors. A method can be used based on earlier observations, but since data are changing over time this knowledge can be of questionable value as far as the compression effectiveness is concerned.

Therefore, in the proposed scheme, the previously observed frequency values are exploited to effectively update Huffman Code tables for the values to follow. Initially, the differential signal is produced, as most likely it has values that can be more effectively compressed. Following the initial phase, the differential signal is separated in fixed size packets. In the first packet, since there is no statistical knowledge of the data, the method is using the table from the LEC algorithm. However, in each following packet the statistical knowledge from previous data is used to create on the fly an adaptive Huffman Code table. The alphabet of numbers is divided into groups, the sizes of which increase exponentially. Each new differential signal value observed, leads to the appropriate group’s frequency increment. When the processing of a data packet ends, the frequencies of each group are used to extract the possibilities of a value that belongs to that group. The blocks are sorted in descending order by their possibilities and a binary tree is created.

After the formation of the Huffman Code table for the following packet, the current frequency table is element wise multiplied with a factor varying between 1 and 0. Therefore, as this parameter is approaching 0 the degree by which history (i.e., frequencies observed in previous packets) is taken into account in the next Huffman Code table diminishes. Therefore, if “0” is selected only the frequencies of the cur-rent packet are used; if 1 is selected the frequencies of every previous packet are equally used in the encoding of the next packets [12].

3.3. Performance Analysis of the Proposed Algorithm

In this section, a brief comparative analysis is presented of the proposed compression algorithm (Real Time Huffman) against LEC and ALEC approaches highlighting critical advantages offered. Considering the former comparison, all three algorithms are implemented in Matlab environment and are being evaluated with respect to the compression rate achieved and compression delay assuming execution on the same personal computer. The evaluation is based upon real EEG and ECG signals, extracted either from respective open data bases or from real measurements undertaken in the context of the specific evaluation effort, which comprise the most challenging modalities in epilepsy monitoring. A brief description of the sources of the datasets as well as the execution environment is as follows.

3.3.1. PhysioNet Database

PhysioNet [19] was established in 1999 as the outreach component of the Research Resource for Complex Physiologic Signals cooperative project. From this database signals were extracted and used as evaluation testbeds of the implemented algorithms from the following two specific subcategories:

1. *Apnea-ECG Database*

This database has been assembled for the PhysioNet/Computers in Cardiology Challenge 2000 [20]. From this database the ecgA04apnea and ecgB05apnea used in the evaluation process have been acquired.

2. *CHB-MIT Scalp EEG Database*

This database [21], collected at the Children's Hospital Bos-ton Massachusetts Institute of Technology (MIT), consists of EEG recordings from pediatric subjects with intractable seizures [22].

3.3.2. University of Patras, EEG and ECG Signals

The dataset is provided by the Neurophysiology Unit, Laboratory of Physiology School of Medicine, University of Patras (UoP) [23]. EEG was recorded using 58 EEG Ag-AgCl electrodes according to the extended international 10–20 system.

3.3.3. EkgMove ECG Signals

EkgMove [24] is a psycho-physiological measurement sys-tem for research applications. From the various measurement capabilities of the sensor, the ECG signal of a single subject has been used.

As a result, from the respective evaluation, Figure 2 presents the achieved compression rate of the three different compression algorithms over the dataset of eight different signals is presented. As depicted, the proposed RT-Huffman algorithm manages to offer the highest compression in all cases. Specifically, compared to LEC, RT-Huffman offers an increased compression rate varying from 1.5 up to 4.3% while the same variation with respect to ALEC reaches up to 3.5%.

Table 1 presents the measurements concerning the processing delay of compression algorithms under evaluation. In this case, due to drastic processing delay differences of the different biosignals, in order to extract more objective conclusions Table 1 depicts the absolute delay for the algorithm offering the lowest measurement and the percentage deviation for the other two algorithms.

As extracted from Table 1, LEC yields the lowest processing demands upon the processing unit. However, what is even more important is that in all cases the proposed RT-Huffman proves to be the second less resource demanding solution clearly outperforming ALEC and exhibiting a steady and relative small overhead compared to LEC thus able to meet time constrained performance demands.

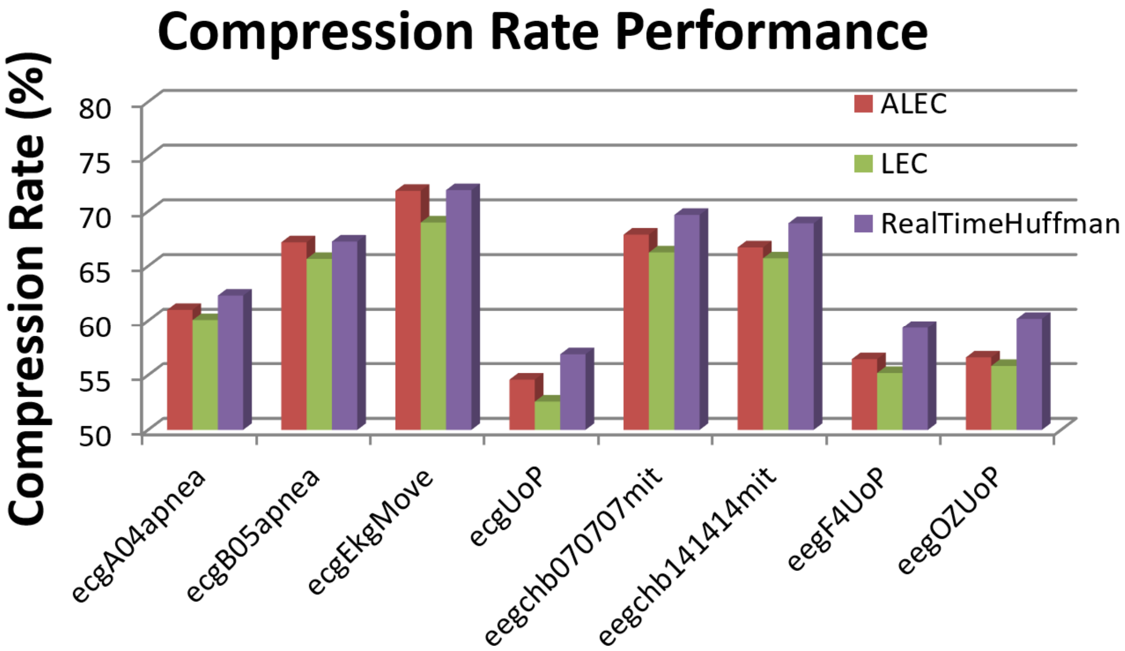

Figure 2. Compression rate performance evaluation.

Table 1. Processing delay performance evaluation.

Data Set	RT-Huffman	LEC (s)	ALEC
ecgA04apnea	+18%	7.5	+30%
ecgB05apnea	+28%	6.7	+35%
ecgEkgMove	+26%	16.8	+35%
ecgUoP	+11%	201	+131%
eegchb070707mit	+14%	18.1	+34%
eegchb141414mit	+23%	17.8	+38%
eegF4UoP	+18%	191.9	+144%
eegOZUoP	+10%	198.17	+124%

4. Hardware Implementation Design

The proposed algorithm was implemented at hardware level and a respective system level performance evaluation was carried out. The compression module processes input samples on the fly with a latency of 4 clock cycles. As depicted in Figure 3, a small 16 × 16 bit *Input Buffer* stores the incoming 16-bit samples and propagates them to the compression module when instructed by the *Local Controller*. On system power-up the samples are propagated through the differential datapath comprised by a *Subtractor* and an *Absolute Calculation Unit*. The absolute value of all samples is used to update a metric table with statistical information and is also used to produce the compressed output. This is the initialization phase of the system.

When a number of samples equal to the defined block size has been collected, the controller enters in calculations' phase and pauses further sample propagation to the rest of the system. The *Huffman Micro-Processor Unit* calculates and produces the Huffman (S) table, based on the populated metric table, which will be applied on the next block of incoming data. The custom microprocessor functions with encoded operations are designed so as to optimize this phase. The core of the Huffman algorithm is implemented by performing parallel memory accesses on the 33 × 9 bits *Tableful Parallel Memory* and by un-rolling all nested while and for loops to serial operations. A 512 × 36 *Instruction Memory* drives the micro-processor to execute all Real Time Huffman algorithm calculations which leads to a substantial processing latency. The worst-case processing latency due to the iterative nature of the algorithm is calculated to 4175 clock cycles per block of 500 samples. Once the Huffman (S) table has been calculated, the controller resumes propagation of samples through the differential data-path and S is applied to the next block of incoming samples in order to produce the compressed output. When a number of samples equal to the defined block size has been collected, the calculation phase is activated again and so on.

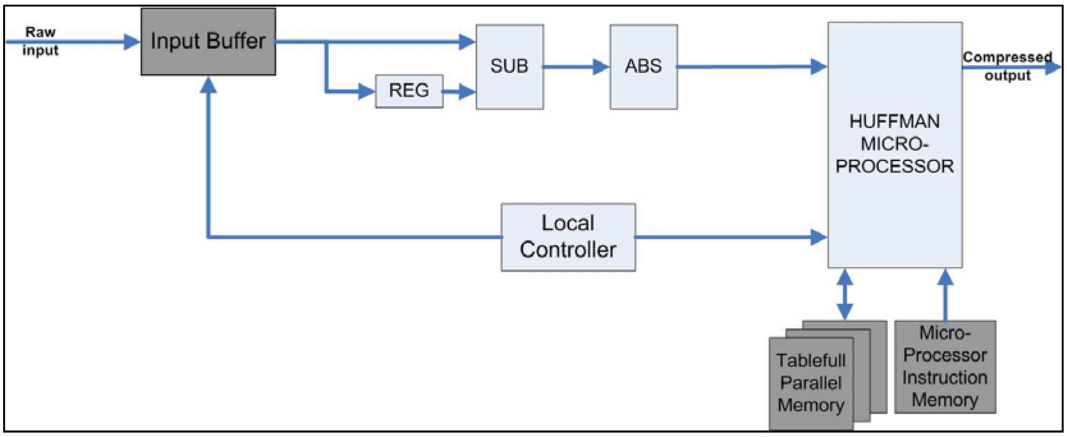

Figure 3. Compression module block diagram.

In Figure 4, the block data flow with respect to compression of the input data block and the calculations of the S table is depicted. It is noted (indicated with the immediate receipt of the data of the 4th data block following the 3rd data block) that if the intermediate interval between the last datum of one block and the first datum of the subsequent is smaller than the time required to perform the S table calculations, the Ready For Data (RFD) signal will go low; then excessive data can be stored in an input buffer until the S table calculations are concluded, and then the compression process resumes.

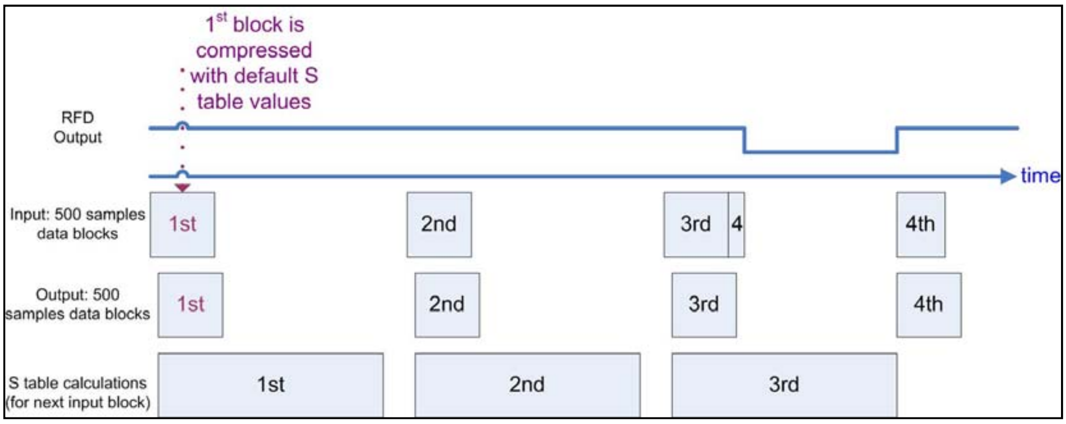
1st block is compressed with default S table values
RFD Output
time
Input: 500 samples data blocks
1st
2nd
3rd
4
4th
Output: 500 samples data blocks
1st
2nd
3rd
4th
S table calculations (for next input block)
1st
2nd
3rd

Figure 4. Compression module block data flow.

4.1. I/Os Timing Protocols

As seen in Figure 5, the mechanism has a 4 CPU cycle delay between the time an uncompressed datum is processed and the time the corresponding compressed equivalent is ready at the output. Therefore, and based on the lookup table at that point, datum 0×FFD8 corresponds to 0×39A requiring 10 bits according to the lookup table, datum 0×000 corresponds to 0×3A5 also corresponding to 10 bits and so on.

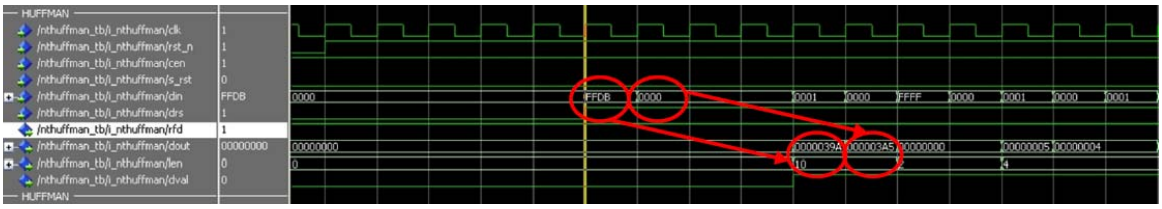
HUFFMAN
/nthuffman_tb/i_nthuffman/clk
/nthuffman_tb/i_nthuffman/rst_n
/nthuffman_tb/i_nthuffman/cen
/nthuffman_tb/i_nthuffman/s_rst
/nthuffman_tb/i_nthuffman/din
/nthuffman_tb/i_nthuffman/drs
/nthuffman_tb/i_nthuffman/rfd
/nthuffman_tb/i_nthuffman/dout
/nthuffman_tb/i_nthuffman/len
/nthuffman_tb/i_nthuffman/dval
FFDB
00000000
FFDB
0000
0001
FFFF
0000039A
000003A5
00000005
00000004
10

Figure 5. Compression module Input/Output block timing. Start of block.

The same timing process is followed until the end of the block as indicated in Figure 6, where the cycled datum is the 500th (i.e., last) datum of the block corresponding to the corresponding cycled datum of data out pin.

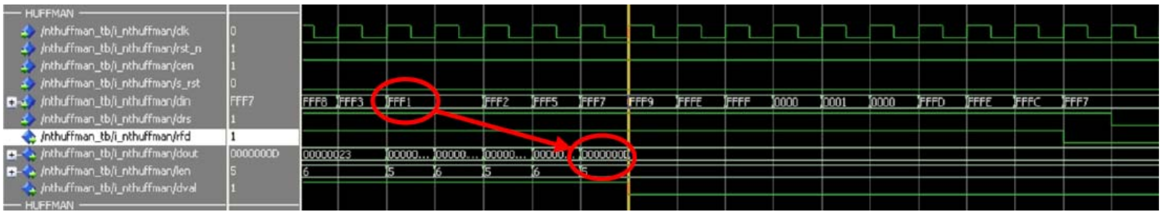
HUFFMAN
/nthuffman_tb/i_nthuffman/clk
/nthuffman_tb/i_nthuffman/rst_n
/nthuffman_tb/i_nthuffman/cen
/nthuffman_tb/i_nthuffman/s_rst
/nthuffman_tb/i_nthuffman/din
/nthuffman_tb/i_nthuffman/drs
/nthuffman_tb/i_nthuffman/rfd
/nthuffman_tb/i_nthuffman/dout
/nthuffman_tb/i_nthuffman/len
/nthuffman_tb/i_nthuffman/dval
FFF7
0000000D
FFF8
FFF3
FFF1
FFF2
FFF5
FFF9
FFFE
FFFF
FFFD
FFFC
00000023
0000000D

Figure 6. Compression module Input/Output block timing. End of block.

4.2. Implementation Results

In Table 2, the processing performance characteristics are presented regarding the processing delay of one block of data and the corresponding throughput considering a relative low frequency clock, adequate for embedded systems.

Table 2. Compression module processing delay and throughput rate (Assuming × MHz operating frequency).

Processing Delay (In Clock Cycles)	Throughput Rate (Worst Case)
4175 (Worst Case)	106.8 Mbps (62.5 MHz Clk)

Calculation of the throughput rate:
A data block of 500, 16 bit samples will be processed at worst case every:
4 cc (data-path latency) + 500 cc (input samples) + 4175 c (processing delay) = 4679 cc.

$$\text{Throughput rate (bps)} = (16 \times 500 \text{ bits})/(4679 \times \text{Tclk}), \tag{1}$$

For 62.5 MHz CLK, we get worst case processing latency estimation at 106.8 Mbps, which is equal to 13.375 "500 × 16 bits samples" data-blocks per second. Based on such throughput capacity the following comparison with software-based respective implementation can be made.

Considering 10-minute EEG signals acquired from the MIT data base [22] the corresponding file is 307.200 bytes and the required compression delay exhibited by the software-based algorithm was measured to be approximately 20 s considering an i7 dual core pc. Based on the previous calculation, the respective delay of the hardware accelerator is anticipated to be around 20 ms, thus offering a drastic decrease regarding delay as well as resource consumption. Furthermore, such evaluation indicates that the presented hardware accelerator comprises a highly efficient solution when considering multiple channels acquired concurrently or/and low frequency processing units being available by the sensors.

Additionally, the implementation presented is resource conservative as indicated in Table 3, depicting the degree of FPGA resources required by the specific implementation. Such measurements effectively indicate that more than one hardware accelerator could coexist in the same FPGA board offering increased flexibility.

Table 3. Compression module resources (Spartan3E xc3s100 – 4ft256 Xilinx FPGA Technology).

Configurable Logic Block (CLB) Slices	Flip Flops (FFs)	Block RAMs	I/Os
1676/7680 − 21%	1148/15,360 − 7%	7/12 − 29%	60/173 − 34%

4.3. Power Dissipation

For 1 Kbps requirement, 1 kHz clock will be sufficient. However due to XPower Analyzer limitations [23] we can get a power estimate for frequencies as low as 1 MHz (for worst case 100% toggling rates). Specifically, the total power usage of an FPGA device can be broken down to total static power and total dynamic power. Static power is associated with DC current while dynamic power is associated with AC current. Static power is power consumed while there is no circuit activity. Dynamic power is power consumed while there is inputs/circuit activity (Table 4).

Table 4. Compression module power dissipation (Spartan3E Xilinx FPGA Technology).

Clock Frequency (MHz)	Total Power (mW)	Dynamic Power (mW)	Static Power (mW)
1	114.27	15.53	98.74
62.5	198.52	98.45	100.07

5. System-Level Performance Evaluation

The main objective of this section is to analyze both qualitatively and quantitatively the benefits that the integration of the proposed compression module can offer to a real WSN platform, by taking into account both the performance and the energy conservation.

The degree by which the bandwidth required to transmit a specific amount of data is reduced, due to utilization of compression, depends solely on the algorithm and not on the integration efficiency. However, the system-level transmission delay and the energy consumption metrics depend heavily on the hardware design of the compression modules, as well as on the integration of the component to a specific WSN platform. To evaluate both compression delay and compression power consumption, we assume a signal sample of X Bytes that can be compressed by $Y\%$ and adopt the following rationale.

As far as the transmission delay is concerned, the two equations that enable us to calculate the required time interval to send the respective information with and without the compression module are the following ones.

Without the compression module integrated:

$$Tx_{no\_compress} = Tx_{XBytes}, \tag{2}$$

where Tx_{XBytes}: Represents the time interval required to successfully transmit X Bytes and it is measured using a specific platform and specific WSN communication technology.

With the compression module integrated:

$$Tx_{compress} = Tc_{XBytes} + Tx_{x\times y\%Bytes} + Tdc_{XBytes}, \tag{3}$$

where Tc_{XBytes} is the time interval required to compress X Bytes that can be accurately calculated for a specific processing clock frequency of the implementation presented in Section 4. $Tx_{x\times y\%Bytes}$ is the time required to successfully transmit the compressed amount of data which can be accurately measured using a specific platform and a specific WSN communication technology. Tdc_{XBytes} is the time interval required to decompress the compressed data back to X Bytes which, without loss of accuracy, can be considered equal to Tc_{XBytes} since the compression and decompression algorithms are symmetrical in the proposed solution.

Moving on to energy consumption the respective equations used are as follows.

Energy consumption without integrated compression module:

$$Ex_{no\_compress} = P_{tx\_radio} \times Tx_{XBytes} = V_{tx\_radio} \times I_{tx\_radio} \times Tx_{XBytes}, \tag{4}$$

where $Ex_{no\_compress}$ is the energy consumed for sending successfully X Bytes of data. P_{tx_radio} is the transmission power consumption of a specific radio transceiver. V_{tx_radio} is the voltage supply required of specific radio transceiver (provided by respective datasheets). I_{tx_radio} is the current draw during transmission of a specific radio transceiver (provided by respective datasheets). Tx_{XBytes} is the time required to successfully transmit X bytes; it is measured using a specific platform and a specific communication technology.

Energy consumption with integrated compression module:

$$\begin{aligned} Ex_{compress} = Ec_{XBytes} + E_{tx_x \times y\%Bytes} = (P_{compr} \times Tc_{XBytes}) + (P_{tx_{radio}} \times Tx_{X \times y\%Bytes}) = \\ (P_{compr} \times Tc_{XBytes}) + (V_{tx\_radio} \times I_{tx\_radio} \times Tx_{X \times y\%Bytes}), \end{aligned} \tag{5}$$

where $Ex_{compress}$ is the energy required to compress X Bytes and successfully transmit the compressed data. Ec_{XBytes} is the energy required for compressing X Bytes. $E_{tx_x \times y\%Bytes}$ is the energy required to successfully transmit the compressed data. P_{compr} is the power consumption (static + dynamic) of the compression module, when active. Tc_{XBytes} is time interval required to compress X Bytes, which can be accurately calculated for a specific processing clock frequency. V_{tx_radio} is the voltage supply required for a specific radio transceiver (provided by respective datasheets). I_{tx_radio} is the current draw during the transmission of specific radio transceiver (provided by respective datasheets). $Tx_{X \times y\%Bytes}$ is the time required to successfully transmit $X \times Y\%$ Bytes; it is measured using a specific platform and a specific WSN communication technology.

It is noted that energy consumption is not considered for the decompression phase, under the assumption that the receiver (i.e., the home gateway) is always plugged in the main power supply. Furthermore, two assumptions are made so as to extract the results presented in following section. First, it is assumed that when the compression module is not used it is shut down and does not consume energy; second, it is assumed that when the radio does not transmit it is tuned off and thus there is no energy consumption.

5.1. Performance Evaluation Setup

In order for the analysis approach presented in the previous section to lead to realistic and useful results, it is important to present the main hardware components (including the different configuration options considered) utilized as well as the actual data, comprised by samples taken from real EEG/ECG signals, which are two of the most demanding modalities in epilepsy monitoring (which represents our main application scenario).

The most important component influencing our evaluation is the implemented compression hardware module. As analyzed in Section 4, the hardware module throughput (thus the delay required to compress a specific amount of data) depends on the clock frequency based on Equation (1).

The power consumption depends also on the frequency clock. Therefore, we considered four different clock frequencies; then, we measured the power consumption and based on Equation (5) the following table has been extracted. Table 5 presents the two main performance metrics of the compression hardware implementation proposed.

Table 5. Power consumption and throughput performance of compression hardware component.

Clock Frequency (MHz)	Total Power (mW)	Dynamic Power (mW)	Static Power (mW)	Throughput (Mbps)
1	114.27	15.53	98.74	17.09
20	142.38	43.20	99.18	34.19
40	170.07	70.45	99.62	68.39
62.5	198.52	98.45	100.07	106.86

Table 5 depicts also a tradeoff that, as will be proven, comprises a critical factor with respect to the overall performance evaluation. As depicted, as the clock frequency increases, both power consumption and throughput also increase. Consequently, when compressing a specific amount of data increasing the clock frequency implies that, on one hand, the component needs more energy while active but, on the other hand, it performs its operation much faster, and consequently the time period required to be active is drastically reduced. A secondary observation concerns the fact that the main factor of power consumption increase is the dynamic power and not the static power (its influence is rather negligible).

5.1.1. Short-Range Wireless Communication Radio Characteristics

The second component influencing the system-level performance is the specific off-the-shelf radio transceivers used to transfer data. In the context of this evaluation, we considered the Shimmer2R platform [25] comprising a prominent commercial solution offering highly efficient Bluetooth Class 2.1 as well as IEEE 802.15.4 interfaces.

Bluetooth Interface Characteristics:

The specific platform uses the RN-42 Bluetooth radio chip exhibiting the following voltage and current demands [26].

Voltage supply = 3.3 V

Typical Current draw during transmission = 45 mA

Additionally, the respective node is programmed using TinyOS operating system so additional delay could be added as expected to be the case in any off-the-shelf WSN sensor.

IEEE 802.15.4 Interface Characteristics:

The specific platform uses the CC2420 IEEE 802.15.4 radio chip exhibiting the following voltage/current demands [14].

Voltage supply = 3.3 V

Typical Current draw during transmission (max) = 17.4 mA

Additionally, the respective node is programmed using TinyOS operating system so additional delay could be added as expected to be the case in any off-the-shelf WSN sensor.

5.1.2. Dataset Compression Characteristics

The actual signals comprising the evaluation dataset comprise also a significant parameter influencing the performance of hardware accelerator. It is assumed that all datasets are parts of signals of equal time duration and specifically of 10min interval. Specifically, the signals considered are derived from the extended dataset presented in Section 3.3 and the characteristics relevant to the specific evaluation are as follows:

1. ECG_UoP

An ECG signal with sampling rate 2.5 KHz, 16 bits per sample and 10 min duration leads to an amount of data equal to 3.000.000 Bytes; the implemented compression algorithm is able to compress them to 1.292.952 Bytes, thus yielding 56% compression rate. Shimmer Platform using BT technology is able to transfer compressed and uncompressed data in 60 and 25 s respectively. Shimmer Platform using IEEE 802.15.4 technology is able to transfer compressed and uncompressed data in 488 and 211 s respectively considering a configuration of 108 byte per packet and a packet transmitter every 2 ms.

2. ECG_EkgMove

An ECG signal acquired using the EkgMove sensor with sampling rate of 256 Hz, 16 bits per sample and 10 min duration leads to an amount of data equal to 307.200 Bytes; the implemented compression algorithm is able to compress to 86.536 Bytes yielding 72% compression rate. Shimmer Platform using BT technology is able to transfer compressed and uncompressed data in 6 and 2 s respectively. Shimmer Platform using IEEE 802.15.4 technology is able to transfer compressed and uncompressed data in 50 and 15 s respectively.

5.2. Evaluation Results

In this section, we present the most interesting quantitative results of our evaluation. In order to extract the respective observations more easily, Figure 7 presents the results from Tables 6 and 7 in comparison to the power demands of the compression module.

Table 6. Energy and delay reduction for ECG_UoP using shimmer bluetooth nodes.

Clock	Transmission Delay (s)			Energy Consumption (mJ)		
MHz	Without Compression	With Compression	% Reduction	Without Compression	With Compression	% Reduction
1	60	53.074	11.5	8910	5316.5	40.3
20	60	26.4	55.99	8910	3812.4	57.2
40	60	25.7	57.1	8910	3772.18	57.8
62.5	60	25.44	57.5	8910	3757.1	57.6

Table 7. Energy and delay reduction for ECG_UoP using shimmer 802.15.4 nodes.

Clock	Transmission Delay (s)			Energy Consumption (mJ)		
MHz	Without Compression	With Compression	% Reduction	Without Compression	With Compression	% Reduction
1	488	239	51	28,020.96	13,719.63	51
20	488	212	56	28,020.96	12,215.55	56
40	488	211	57	28,020.96	12,175.3	57
62.5	488	211	57	28,020.96	12,160.21	57

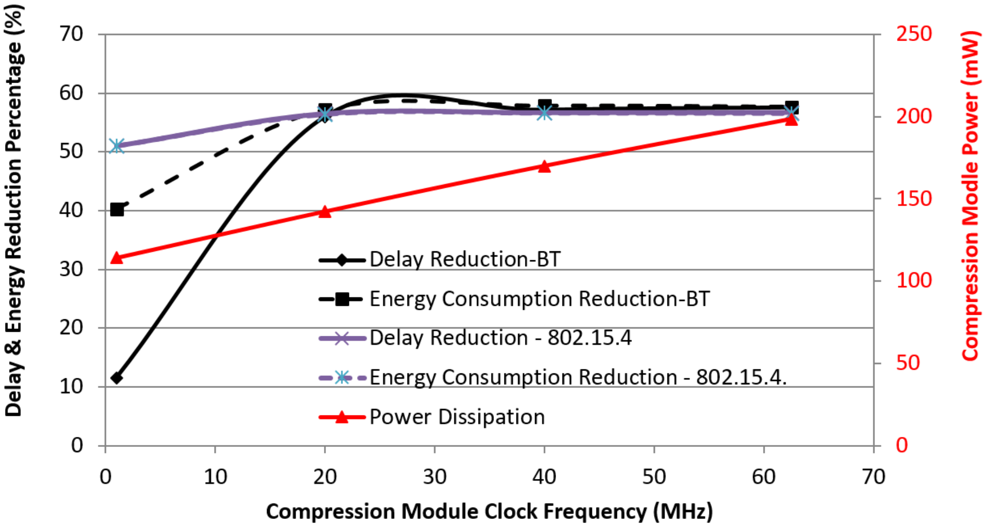

Figure 7. Compression module performance effect wrt ECG_UoP/shimmer setup.

5.2.1. ECG_UoP

Based on Equations (2)–(5) and the individual components' characteristics as presented in previous section, the analytical results are presented in Tables 6 and 7.

As it can be easily extracted, significant benefits can be offered by integrating the proposed compression module into a WSN node. What is interesting to note is that when increasing the frequency clock, although it leads to increased power for the compression module, it also leads to performance enhancements in both metrics considered i.e., end-to-end delay and power consumption. This is attributed to the considerably faster operation of the module, which decreases drastically the time period required by the module to be active. However, this performance enhancement also seems to have an upper limit, since increasing the frequency beyond 20 MHz it leads to negligible performance enhancement. Thus, the respective increase to power requirements beyond 20 MHz clock frequency is effectively wasted since it does not lead to further performance enhancement.

Finally, it is important to emphasize the fact that a reduction of almost 60% can be calculated both for the delay and the energy dissipation using the proposed hardware compression module.

Focusing on IEEE 802.15.4-based measurements, what is especially important in this case is that a drastic enhancement is apparent even with the lowest clock frequency considered (i.e., only 1 MHz). This is attributed to the much longer delay required to transmit data compared to the BT case. Consequently, in this case, it proves significantly beneficial to compress data before wireless transmission. This is indicated both delay and energy wise, since the compression phase, due to its hardware implementation, leads to drastically low delay operation, which in turn minimizes any end-to-end delay and energy overhead. Furthermore, the same pattern is identified in the graphs indicating 20 MHz as the optimum frequency selection, yielding considerable performance enhancement, and limited power consumption overhead. Overall, this maximum performance enhancement is a bit lower compared to BT case but still reaching up to 57%.

5.2.2. ECG_EkgMove

Based on Equations (2)–(5) the analytical results are as follows described in Tables 8 and 9. In order to extract the respective observations more easily, in Figure 8 the results from Tables 8 and 9 are presented in comparison to the power demands of the compression module.

Table 8. Energy and delay reduction for ECG_EkgMove using shimmer BT nodes.

Clock	Transmission Delay (s)			Energy Consumption (mJ)		
MHz	Without Compression	With Compression	% Reduction	Without Compression	With Compression	% Reduction
1	6	4.87	19	891	461.25	48
20	6	2.14	64	891	307.23	65
40	6	2.07	66	891	303.11	66
62.5	6	2.04	65	891	301.56	66

Table 9. Energy and delay reduction for ECG_EkgMove using shimmer 802.15.4 nodes.

Clock	Transmission Delay (s)			Energy Consumption (mJ)		
MHz	Without Compression	With Compression	% Reduction	Without Compression	With Compression	% Reduction
1	50	17.9	64	2871	1025.55	64
20	50	15.1	70	2871	871.53	70
40	50	15.1	70	2871	867.41	70
62.5	50	15.1	70	2871	865.87	70

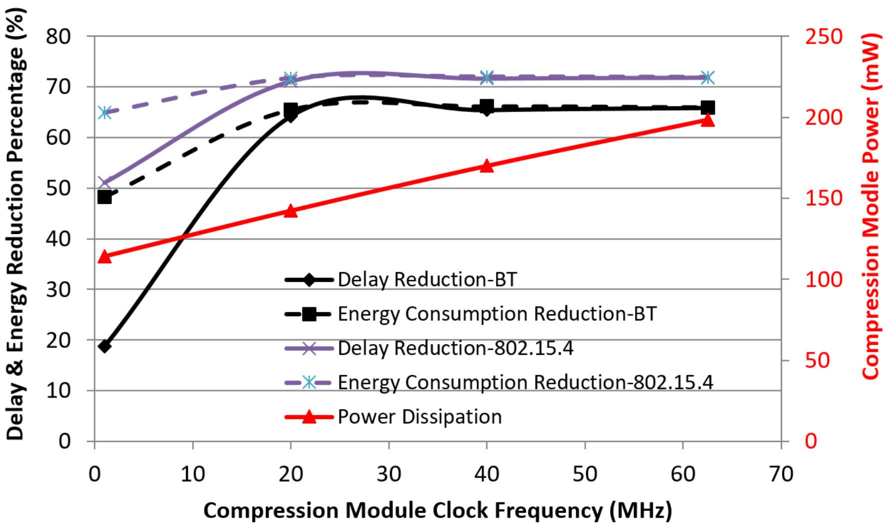

Figure 8. Compression module performance effect wrt ECG_EkgMove/shimmer BT setup.

Figure 8 reveals behavioral patterns similar to the ones in previous cases, strengthening the indications regarding the performance enhancement of the compression module. Focusing on BT-based graph, an interesting point is the fact that in comparison to the respective measurements considering ECG UoP signal, in this case a steadily higher performance enhancement (~8%) can be extracted. This enhancement increase is attributed to the lower sampling frequency of the EkgMove sensor leading to fewer samples for same time interval, thus leading significantly faster compression execution from the proposed compression module. This is actually the part that effectively saves network resources either in the delay or in energy consumption domain. Also, in this case the type of signal has a critical role; this is due to the fact that the more compression prone the signal is the higher the performance enhancement will be, which in this case is close to 70% both for delay and energy reduction.

Focusing on 802.15.4-based graphs, once again the exhibited delay and energy consumption decrease are both quite emphatic. Both reach up to 70%, which can be achieved even with a moderate FPGA clock frequency of 20 MHz (62.5 MHz being the highest supported by the FPGA development board considered for the implementation).

Another interesting observation concerns the fact that even considering the significantly lower power consumption of IEEE 802.15.4 transceivers (compared to Bluetooth counterpart), the integration of a compression accelerator module would still lead to a significant performance boost and consequent enhance the efficiency of any decision support system.

6. Discussion

This paper explores the possibility to exploit data compression, in order to effectively tackle the wireless data transmission requirements of Epilepsy monitoring by typical WSN platforms. Adequate and objective study of Epilepsy requires continuously acquisition of respective modalities at high rates, thus creating excessive volumes of data. Following a multifaceted research and development approach, in this paper, firstly the problem is addressed algorithmically by the comparison of software-based compression algorithms targeting at optimum trade-off between compression rate and compression delay. In that respect, as analytically presented in [12] the proposed algorithm highly competitive performance. Secondly, going a step further towards practical solutions, the proposed algorithm is implemented at hardware level in the context of a typical FPGA platform. The implemented hardware module is presented in detail in this paper, while performance wise it reduced the compression delay by a factor of 1/1000. At the same time, dynamic power consumption ranges between 15 and 98 mW representing a viable solution for contemporary WSN platforms. The latter is further emphasized in the third part of this work, where the anticipated effect of integrating such a component with real prominent WSN platforms is evaluated.

In this aspect two well-known wireless interfaces are considered, characterized by difference features and capabilities. Furthermore, the effect of running the proposed module at different clock frequencies is also taken into consideration. In all cases the benefits of using the proposed solution are quiet apparent. Specifically, considering Bluetooth wireless transmission, end-to-end delay reduction ranges between 11% and 65%, while energy is respective reduction ranges between 40% and 66%. Performance enhancement is even higher when considering IEEE 802.15.4 wireless communication since both transmission delay and energy consumption are reduced by from 5% to 70%.

Without a doubt, such performance and resource conservation enhancements offer convincing arguments that hardware-based data compression comprises a viable solution towards network/node lifetime increase and wireless communication bandwidth optimum management.

7. Conclusions

In order for IoT platforms to comprise a reliable and efficient technological infrastructure for nowadays as well as future CPS applications, novel hardware, real-time approaches are required in order to enhance the platforms' capabilities while minimizing resource wastage. This paper addresses a notorious challenge of wireless interfaces' excessive utilization considering highly demanding application scenarios such as bio-signal monitoring. To tackle respective shortcomings of nowadays platforms, the two main pillars of this paper are (a) the design and implementation of an efficient, real-time compression algorithm, and (b) the design and implementation of a highly efficient and resource conservative hardware accelerator for the proposed algorithm. The idea behind this approach is to offer on-the-fly hardware compression to any IoT platform, thus overloading the main processing unit, which is usually of limited processing capabilities. Consequently, this paper offers a holistic solution to the problem of wirelessly transmitting excessive volumes of data, tackling all aspects from the design and software algorithmic performance, to the design of the actual hardware accelerator and finally the overall system performance. With respect to the latter, taking into consideration the power consumption of a real prominent IoT platform and respective delay transmission capabilities, the proposed hardware compression accelerator can yield an approximate 70% system wide delay and energy consumption reduction. Furthermore, the proposed module comprises a highly feasible and practical solution since it captures just 21% of the Spartan3 FPGA's configurable logic block slices and the maximum performance gains are achieved at just 1/3 of the maximum frequency clock tested. Such characteristics advocate the integration of the proposed module to nowadays WSN platforms.

Acknowledgments: This study is part of the collaborative project armor which is partially funded by the European commission under the seventh framework programme (FP7/2007–2013) with grant agreement: 287720 and through the horizon 2020 programme (H2020/PHC-19-2014) under the research and innovation action "radio-robots in assisted living environments: unobtrusive, efficient, reliable and modular solutions for independent ageing".

Author Contributions: Christos P. Antonopoulos conceived and designed the experiments concerning the proposed compression algorithm. Nikolaos S. Voros conceived and implemented the hardware implementation of the compression algorithm's module. Christos P. Antonopoulos conceived, designed and carried out the integrated experiments, Both Christos P. Antonopoulos and Nikolaos S. Voros contributed on data analysis. Both Christos P. Antonopoulos and Nikolaos S. Voros wrote the paper.

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

References

1. Perera, C.; Liu, C.H.; Jayawardena, S. The emerging internet of things marketplace from an industrial perspective: A survey. *IEEE Trans. Emerg. Top. Comput.* **2015**, *3*, 585–598. [CrossRef]
2. Zivkovic, M.; Popovic, R.; Tanaskovic, T. The Survey of Commercially Available Sensor Platforms from Energy Efficiency Aspect. In Proceedings of the 20th Telecommunications Forum (TELFOR), Belgrade, Serbia, 20–22 November 2012; pp. 1528–1531.
3. Xu, T.; Wendt, J.B.; Potkonjak, M. Security of IoT Systems: Design Challenges and Opportunities. In Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design, San Jose, CA, USA, 3–6 November 2014; IEEE Press: Piscataway, NJ, USA, 2014. [CrossRef]

4. Tonneau, A.S.; Mitton, N.; Vandaele, J. A Survey on (Mobile) Wireless Sensor Network Experimentation Testbeds. In Proceedings of the 2014 IEEE International Conference on Distributed Computing in Sensor Systems, Marina Del Rey, CA, USA, 26–28 May 2014; pp. 263–268. [CrossRef]
5. Cotuk, H.; Tavli, B.; Bicakci, K.; Akgun, M.B. The Impact of Bandwidth Constraints on the Energy Consumption of Wireless Sensor Networks. In Proceedings of the 2014 IEEE Wireless Communications and Networking Conference, Istanbul, Turkey, 6–9 April 2014; pp. 2787–2792. [CrossRef]
6. Abdel-Aal, M.O.; Shaaban, A.A.; Ramadan, R.A.; Abdel-Meguid, M.Z. Energy Saving and Reliable Data Reduction Techniques for Single and Multi-Modal WSNs. In Proceedings of the 2012 International Conference on Engineering and Technology (ICET), Cairo, Egypt, 10–11 October 2012; pp. 1–8. [CrossRef]
7. Salomon, D. *Data Compression: The Complete Reference*, 4th ed.; Springer Science and Business Media: Berlin, Germany, 2004.
8. Marcelloni, F.; Vecchio, M. An efficient lossless compression algorithm for tiny nodes of monitoring wireless sensor networks. *Comput. J.* **2009**, *52*, 969–987. [CrossRef]
9. Low, K.S.; Win, W.N.N.; Er, M.J. Wireless Sensor Networks for Industrial Environments. In Proceedings of the International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, Vienna, Austria, 28–30 November 2005; pp. 271–276.
10. Kolo, J.G.; Shanmugam, S.A.; Lim, D.W.G.; Ang, L.M.; Seng, K. P. An adaptive lossless data compression scheme for wireless sensor networks. *J. Sens.* **2012**. [CrossRef]
11. Takezawa, T.; Asakura, K.; Watanabe, T. Lossless compression of time-series data based on increasing average of neighboring signals. *Electron. Commun. Jpn.* **2010**, *93*, 47–56. [CrossRef]
12. Antonopoulos, C.P.; Voros, N.S. Resource efficient data compression algorithms for demanding, WSN based biomedical applications. *Elsevier J. Biomed. Inform.* **2016**, *59*, 1–14. [CrossRef] [PubMed]
13. Antonopoulos, C.; Prayati, A.; Stoyanova, T.; Koulamas, C.; Papadopoulos, G. Experimental Evaluation of a WSN Platform Power Consumption. In Proceedings of the 2009 IEEE International Symposium on Parallel & Distributed Processing, Rome, Italy, 23–29 May 2009; pp. 1–8.
14. Polastre, J.; Szewczyk, R.; Sharp, C.; Culler, D. The Mote Revolution: Low Power Wireless Sensor Network Devices. In Proceedings of the Hot Chips, Stanford, CA, USA, 22–24 August 2004.
15. Ultra Low Power IEEE 802.15.4 Compliant Wireless Sensor Module. Available online: http://www.eecs.harvard.edu/~konrad/References/TinyOSDocs/telos-reva-datasheet-r.pdf (accessed on 30 July 2017).
16. Jurcik, P.; Koubaa, A.; Severino, R.; Alves, M.; Tovar, E. Dimensioning and worst-case analysis of cluster-tree sensor networks. *ACM Trans. Sens. Netw. (TOSN)* **2010**, *7*, 14. [CrossRef]
17. Dietrich, I.; Dressler, F. On the lifetime of wireless sensor networks. *ACM Trans. Sens. Netw. (TOSN)* **2009**, *5*, 5. [CrossRef]
18. Mottola, L.; Picco, G.P.; Ceriotti, M.; Guna, S.; Murphy, A.L. Not all wireless sensor networks are created equal: A comparative study on tunnels. *ACM Trans. Sens. Netw. (TOSN)* **2010**, *7*, 15. [CrossRef]
19. Goldberger, A.L.; Amaral, L.A.N.; Glass, L.; Hausdorff, J.M.; Ivanov, P.C.; Mark, R.G.; Mietus, J.E.; Moody, G.B.; Peng, C.K.; Stanley, H.E. PhysioBank, PhysioToolkit, and PhysioNet: Components of a New Research Resource for Complex Physiologic Signals. *Circulation* **2000**, *101*, e215–e220. [CrossRef] [PubMed]
20. Penzel, T.; Moody, G.B.; Mark, R.G.; Goldberger, A.L.; Peter, J.H. The Apnea-ECG Database. In Proceedings of the Computers in Cardiology 2000, Cambridge, MA, USA, 24–27 September 2000; pp. 255–258.
21. Shoeb, A.H. Application of Machine Learning to Epileptic Seizure Onset Detection and Treatment. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 10 September 2009.
22. PhysioNet. Available online: http://www.physionet.org/ (accessed on 30 July 2017).
23. Xpower Analyzer. Available online: http://www.xilinx.com/support/documentation/sw_manuals/xilinx11/xpa_c_overview.htm (accessed on 30 July 2017).
24. Movisens GmbH. Available online: https://www.movisens.com/en/products/ (accessed on 30 July 2017).
25. Shimmer WSN Platform. Available online: http://www.shimmer-research.com/ (accessed on 30 July 2017).
26. Roving Networks Bluetooth™ Product User Manual. Available online: http://www.mct.net/download/rn/rn-41_42.pdf (accessed on 30 July 2017).