Telecommun Syst
DOI 10.1007/s11235-015-0098-2

RDTP: reliable data transport protocol in wireless sensor networks

Ali Barati[1] · Ali Movaghar[2] · Masoud Sabaei[3]

Abstract The paper deals with a new reliable data transport protocol for wireless sensor networks, referred as RDTP. One of the most prominent challenges in wireless sensor networks is reliable transport of data from sensor nodes to sink node. For designing protocols for such networks hardware, some constraining factors associated with energy and processing power must be taken into consideration. There are two generic methods, ARQ and FEC to achieve reliable data transport. Here, a reliable data transport protocol for wireless sensor networks is proposed, in which an efficient moduli set in redundant residue number system has been employed. The modulus set is exploited as a means for adding redundancy to transmitted data. Error controlling in proposed method is performed in a hop by hop manner. The simulation results indicate that the proposed method shows significant decreases in the energy consumption, compared to similar methods. The results also show that this leads to a rise in packet delivery ratio, with a simultaneous reduction in end to end delay.

Keywords Wireless sensor networks · Redundant residue number system · Reliability · Data transport

1 Introduction

The wireless sensor networks are composed of several small nodes (with limited resources) that communicate with sink node, using wireless links [1–3]. These nodes are installed in the network by aim to collecting data about their surrounding environment. This data is then transmitted to the sink node. One of the major characteristics of wireless sensor networks is their ability to interacting with their surrounding environment, a process which is not viable for human intervention. These networks might have a whole variety of applications, including military, medicine, industrial automation, unknown environments identification, object tracking and others like these [4,5]. The sensor nodes in these networks, by their virtue, sense the environmental data and transfer them to the sink node. Some factors such as environmental noises may generate error in transmitted data. Thus the packet reliability becomes a prominent challenge in WSNs. There are two methods to achieve packet reliability; ARQ and FEC [6,7]. These mechanisms are able to operate either in end-to-end or hop-by-hop manner. In automatic repeat request methods (ARQ), an acknowledgement message (ACK) is forwarded to the transmitter when the data is received by the receiver node. The transmitter then sends the data once more if it does not receive acknowledgement message, before the corresponding timeout. Such methods do not yield acceptable performance in noisy environments, because the data is distorted again in second transmission. As another method, a redundancy is used in forward error correction codes, as a means of reassuring reliability. Their error correction and error detection capability depends on the amount of redundancy and coding approach. This paper presents a Reliable Data Transport Protocol for WSNs. In the proposed scheme, error controlling is performed in a hop-by-hop manner, in which error detecting and correcting is

✉ Ali Barati
abarati@iaud.ac.ir

[1] Department of Computer Engineering and Information Technology, Qazvin Branch, Islamic Azad University, Qazvin, Iran

[2] Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

[3] Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran

Published online: 22 October 2015

possible in each hop. The method proposed here is categorized as an FEC method and might be implemented on either tree-like or cluster based network topologies. The rest of the paper has been structured as follow: Sect. 2 explains the research methodology. The proposed method, its function in mentioned topologies and some examples will be presented in Sect. 3. These are followed by results evaluation and then the conclusion, during Sects. 4 and 5, respectively.

2 Related work

As mentioned in previous section, error control methods are generally divided into two groups; FEC and ARQ.

Retransmissions-based reliability

Retransmission-based methods are in the forms, hop-by-hop or end-to-end. These are based on acknowledgement mechanism. ACK and NACK control messages might be mentioned as validation mechanisms. If data is taken by receiver, it sends ACK message; whereas, a NACK message is sent to transmitter under the circumstances where the data is corrupted or distorted.

Forward error correction (FEC)

The FEC based mechanisms, by its virtue, adds redundancy to transmitted packets. The original packets might be recovered if the sufficient number of correct packets is received by destination node. It is usually easy to manage feedback-free protocols, except for encoding and decoding overheads for receiver and transmitter. Nevertheless, FEC utilizes redundancy to obtain data reliability. By the more redundancy, the FEC gets the more capacity for error detection and correction. The FEC codes, suitable for channel conditions and reliability level include Convolution, BCH and Reed Solomon codes [8,9]; the important cases of which are described as follows:

2.1 Sensor transmission control protocol (STCP)

Iyer et al. [10] have introduced STCP as an end-to-end reliable transport protocol. STCP provides different levels of reliability that are proportional to the type of application. Their proposed method generates two levels of reliability for event-driven and continuous data-flow applications. In the former case, ACK packets are employed to ensure correct receiving of packets at destination; while, in the latter, the NACK packets are utilized for the same purpose. The sensor nodes transmit a session initiation packet to the base station, prior to packet transmission. This packet consists of parameters like the number of flows, transmission rate, and the data flow type. Upon the reception of this packet, it is stored in the base station, and based on it, the necessary adjustments to the timer and other parameters are performed. This makes it possible for the base station to send an ACK message to establish the connection. Subsequent to the ACK message, the sensor node transmits data while base station sends ACK or NACK messages, with respect to type of data flow.

2.2 Distributed transport for sensor networks (DTSN)

Marchi et al. [11] have proposed a reliable transport protocol known as DTSN. It ensures two levels of reliability, namely, full and differentiated. The former is used under the circumstances where all packets are required to be received correctly; otherwise, the latter would be exploited. To support full reliability, DTSN utilizes Selective Repeat ARQ and ACK and NACK control packets. The DTSN protocol uses FEC mechanism to support differentiated reliability, according to which, the session is a source/destination communication, defined by the following tuple and identified by session ID:

$$< \text{source address, destination address, application identifier, session number} >$$

The intermediate nodes store the packets in their buffers, and then they are sent to be retransmitted, if needed. When a huge volume of data is transmitted, data buffering requires a large amount of memory that is bond to pose serious challenges.

2.3 Pump slowly fetch quickly (PSFQ)

Wan et al. [12] proposes a reliable downstream mechanism that functions in such as manner to pump slowly and fetch rapidly. It is a hop-by-hop method. The source node sends data slowly; meanwhile, the nodes with lost packets have to fetch them very fast. This method is designed for applications requiring reliability of full data delivery type. PSFQ utilizes three operations fetch, pump and report [13]. The sink node slowly transmits code segments to the sensor nodes which immediately trigger fetching. The nodes stop forwarding new packets as far as there are lost packets. Meanwhile, the nodes send NACK message to their neighbors, so that they retransmit the lost packets again. The computation overhead is high in this method as there are so many timers to maintain the paths. These timers are crucial for starting pump and fetch operations.

In PSFQ, all nodes must save all received packets to provide a successful reprogramming for the sensor nodes. However, in wireless sensor networks, the nodes have limited memory units in which storing all received packets wont be permitted. The propagation in fetching and pumping operations leads to congestion which in turn causes to increase the energy consumption.

2.4 A reliable erasure-coding based data transfer scheme (RDTS)

Srouji et al. [14] introduced RDTS that employs hop-by-hop ensure coding and partial coding. The probability of successful data input in each hop is assumed to be $P_{success}.P_i$ denotes $P_{success}$ in ith hop located between nodes N_i and N_{i+1}. Thus, the probability of successful delivery through n hop is calculated as follows:

$$p_{path} = \prod_{i=1}^{n} p_i \tag{1}$$

The relation between the number of transmitted and received packets are derived from undergoing equation

$$N_{received} = p_{success} \times N_{sent} \tag{2}$$

In order to recover the original data using erasure coding, the number of received fragments, m', should be larger or equal to original fragments, m; i.e. $m' \geq m$. Hence, the number of transmitted data fragments must be larger or equal to $\frac{m}{p_{success}}$. The number of transmitted data fragments equals to decoding data fragments, $m + k$; therefore:

$$m + k \geq \frac{m}{p_{success}} \tag{3}$$

The original data can't be recovered even if one fragment is lost. For the sake of more reliability, δ redundant fragments are transmitted; thus,

$$m + k = \frac{m}{p_{success}} + \delta(p_{success}, N_{hops}) \tag{4}$$

where δ is dependent on $P_{success}$ and N_{hops}. In RDTS, the number of redundant fragments for i^{th} hop is determined with respect to P_i using hop-by-hop coding. The total number of fragments transferred between the nodes is obtained as follows:

$$S_{RDTS} = \frac{m}{p_1} + \frac{m}{p_2} + \cdots + \frac{m}{p_n} = \sum_{i=1}^{n} \frac{m}{p_i} \tag{5}$$

In partial coding, if an intermediate node performs normal erasure coding, it is bond to behave as follows. First, it decodes the received fragments to obtain original data fragments. Afterwards, it codes recovered data to achieve sufficient fragments for the next hop. This process is called ”Full Coding”.

2.5 Reliable multi-segment transport (RMST)

Stann et al. [15] proposed a reliable multi-segment transport (RMST) protocol which is a transport layer protocol. It is NACK based and also one of the first transport layer protocols, designed for WSNs. RMST has the capabilities needed for a transport layer protocol, including reliable transport and multiplexing/demultiplexing. The proposed RMST has two modes:

- Non-caching mode: a very simple mode in which only the source and destination determine the reliability. After detecting the lost packets by the sink node, it transmits a NACK packet to the source. This mode is advantageous as it needs less process.
- Caching mode: in this mode, intermediate nodes store transferred packets so as to reduce overhead of end-to-end retransmission. In RMST, each packet is identified via a unique number. Thus, appearing a gap among the sequence of received numbers indicates error occurrence. In the case of error occurrence, the destination node transmits a NACK packet to ensure packet retransmission.

2.6 Achieving effective reliability for downstream communication (GARUDA)

In [16], a reliable downstream data delivery has been proposed by Park et al. In this scheme, the sink node transmits some control codes to other nodes, for reprogramming purpose. To retransmit the lost packets, GARUDA determines some nodes as the cores. This protocol employs A-map to avoid repeated transmission of NACK messages. A-map is a bit map generated based on packets which exist in the core node. It reveals that which packets might be transmitted to the requesting node, via the core node. The reliability in GARUDA is supported by retransmission of packets from the core node. The GARUDA is not an appropriate choice for congestion control systems, and it only guarantees the reliability of first packet transmission.

2.7 Coded packets over lossy links: a redundancy-based mechanism

A redundancy based approach is presented by Wu et al. [17] to obtain high reliability and low end-to-end delay. In the case of an unreliable link, this protocol uses network coding to add redundancy in the packet. The redundancy level is determined with respect to application conditions and the link loss ratio. Before transmitting the packet, the sensor node checks whether the redundancy must be used. If the path is unreliable, the sensor node uses network coding to improve packet reception ratio. Re-decoding/encoding operation in intermediate nodes is optional. The intermediate node may transmit

Table 1 Summary of existing reliable transport protocols for WSNs

Protocols	Reliability level	Loss recovery	Reliability direction	Retransmission-based/ redundancy-based
STCP [10]	Packet/event	End-to-end	Upstream	Retransmission-based
DTSN [11]	Packet	End-to-end	Upstream	Retransmission-based and Redundancy-based
PSFQ [12]	Packet	Hop-by-hop	Downstream	Retransmission-based
RDTS [13]	Packet	Hop-by-hop	Upstream	Redundancy-based
RMST [15]	Packet	Hop-by-hop	Upstream	Retransmission-based
GARUDA [16]	Packet	End-to-end	Downstream	Retransmission-based
Wu et al. [17]	Packet	Hop-by-hop	Upstream	Redundancy-based

the packet without any further process (re-coding/encoding), by increasing the redundancy level, or by decreasing it. The protocol utilizes the redundant transmission in case of a link loss ratio higher than specified threshold. Data is encoded in application layer. Furthermore, the destination node is able to decode the received packets based on data vector. The source node examines link loss ratio to determine the redundancy. If it (the final value used by the source node) is higher than predefined threshold, the source node determines the redundancy level by considering the application layer constraints. However, if the link loss ratio is less than or equal to the threshold value, encoding is not performed and the redundancy would not be added. If the protocol is not implemented in intermediate nodes, the packets are forwarded without any process.

In this section several reliable data transport protocols in wireless sensor networks were reviewed. A comparison between the mentioned methods will be shown in Table 1.

3 Reliable data transport protocol (RDTP)

This paper explains the RDTP with hop-by-hop error controlling. The proposed protocol is redundancy-based, is categorized as FEC class and provides error correction and detection. Here, the redundant residue number system (RRNS) is applied to generate redundancy. The residue number system concept and redundant residue number system are investigated. Then, a novel moduli set is introduced which is also compared to its counterparts. In the proposed method, the redundancy is generated using defined moduli set. The notations used in this paper are listed in Table 2.

3.1 Residue number system

The residue number system (RNS) is a non-weighted system in which the numbers are denoted by the residue of their division by a specific moduli set [18–20]. In this number system, the moduli set is defined as a set of values $\{P_1, P_2, \ldots, P_n\}$ such that $\gcd(P_i, P_j) = 1$ (for $i \neq j$). In this system each number X is demonstrated as $\{x_1, x_2, \ldots, x_n\}$ in which each x_i is calculated as follows.

Table 2 Notations and definition

Notation	Definition
P_i	ith modulus of a moduli set
X	Initial number
x_i	Residue of the division of X by ith modulus
Y	Recovered initial number
y_i	Residue of ith modulus received at the destination
M	Dynamic range
h	Number of the main moduli
r	Number of the redundant moduli
$\lvert P_i^{-1}\rvert_{p_i}$	Multiplicative inverse of P_i modulo P_j
$V_i, 1 \le i \le n$	Mixed-radix coefficients
$Residue_{ij}$	ith residue from jth node

$$x_i = Xmodp_i = |X|_{p_i} \qquad 1 \le i \le n. \tag{6}$$

Given that the moduli set is $\{P_1, P_2, P_3\} = \{3, 5, 7\}$ and initial number is 29 $(X = 29)$, residues will be calculated as follows using residue number system:

$$x_i = x\text{mod}p_i = |X|_{p_i}, 1 \le i \le 3 \rightarrow \left.\begin{array}{r} x_1 = 29\text{mod}3 = 2 \\ x_2 = 29\text{mod}5 = 4 \\ x_3 = 29\text{mod}7 = 1 \end{array}\right\}$$
$$\rightarrow (x_1, x_2, x_3) = (2, 4, 1)$$

RNS is employed in computation applications which require real time processing, such as digital filters [21], RSA cryptography algorithm [22] and digital communications [23].

If some redundant moduli are added to RNS, the Redundant Residue Number System (RRNS) is resulted. RRNS is represented by a moduli set similar to $\{P_1, P_2, \ldots, P_h, P_{h+1}, \ldots, P_{h+r}\}$ where for each $i = 2, 3, \ldots, h + r$, $P_i > P_{i-1}$.

If all possible two-moduli combinations are coprime, the dynamic range equals to [24,25]:

$$\left[0, \prod_{i=1}^{h+r} P_i\right) \tag{7}$$

In a redundant residue number system with $h+r$ moduli, the number X ($\alpha \leq X < \alpha + M$) is represented by $h+r$ residues in the form of $\{X_1, X_2, \ldots, X_h, X_{h+1}, \ldots, X_{h+r}\}$ [26–28]. Where 3 main moduli are $\{3, 5, 7\}$ and 2 redundant moduli are $\{11, 13\}$, RRNS will be:

$$\{P_1, P_2, , P_h, P_{h+1}, , P_{h+r}\} = \{3, 5, 7, 11, 13\}$$

The dynamic range equals to:

$$\left[0, \prod_{i=1}^{h+r} P_i\right) = \left[0, \prod_{i=1}^{3+2} P_i\right) = [0, 3*5*7*11*13)$$
$$\text{if} \quad X = 29 \quad \text{then} \rightarrow (x_1, x_2, , x_h, x_{h+1}, , x_{h+r}) = (2, 4, 1, 7, 3)$$

In this paper a new moduli set including 3 main moduli and 2 redundant moduli is presented. As these moduli are going to be applied to wireless sensor networks, they should be selected with respect to limitations of such networks. The proposed moduli set is $\{2^{2n+2}, 2^{n+1}-1, 2^n-1, 2^{2n+2}+1, 2^{2n+3}-1\}$ where $\{2^{2n+2}, 2^{n+1}-1, 2^n-1\}$ are main moduli and $\{2^{2n+2}+1, 2^{2n+3}-1\}$ are redundant ones. Designing an efficient reverse converter is an essential challenge for the moduli set. A proper reverse converter should have low volume of required hardware, while imposing a small delay. Increasing the size of main moduli facilitates the dealing with much larger numbers, while makes their representation easier. On the other hand, a rise in the number of redundant moduli increases error detection and correction capability.

3.2 Designing and implementing the proposed residue to binary converter

There are two general approaches for designing a reverse converter; Chinese remainder theorem (CRT) and mixed radix conversion (MRC). Among these two methods one which is able to calculate multiplicative inverses with less complexity would be selected. A simpler calculation of multiplicative inverses leads to less design cost. The MRC is utilized according to the aforementioned criteria. The following equations must be satisfied to design the reverse converter [29–31].

$$X = V_n \prod_{i=1}^{n} P_i + \cdots + V_3 P_2 P_1 + V_2 P_1 + V_1 \tag{8}$$

Also, the coefficients are calculated as follows.

$$V_1 = x_1 \tag{9}$$
$$V_2 = |(x_2 - x_1)|P_1^{-1}|_{p_2}|_{p_2} \tag{10}$$
$$V_3 = |((x_3 - x_1)|P_1^{-1}|_{p_3} - V_2)|P_2^{-1}|_{p_3}|_{p_3} \tag{11}$$

To obtain V_n, the undergoing equation is used.

$$V_n = (((x_n - V_1)|P_1^{-1}|_{p_n} - V_2)|P_2^{-1}|_{p_n} - \cdots - V_{n-1}) |P_{n-1}^{-1}|P_n|_{p_n} \tag{12}$$

where $|P_i^{-1}|_{p_j}$ denotes the multiplicative inverse of P_i modulo P_j. In mathematics, a multiplicative inverse for a number X, is a number that when multiplied by X yields to the multiplicative identity, 1. The modular multiplicative inverse of a modulo, m, can be derived from the extended Euclidean algorithm.

According to Eqs. (8)–(11), the proposed reverse converter can be designed for the new 3-moduli set $\{2^{2n+2}, 2^{n+1}-1, 2^n-1\}$ as follows. Consider the 3-moduli set $\{P_1, P_2, P_3\} = \{2^{2n+2}, 2^{n+1}-1, 2^n-1\}$ with three corresponding residues (x_1, x_2, x_3). In order to design a residue for the binary converter, at first, we need to obtain the multiplicative inverse values and substitute these values with the modulus set in the conversion algorithm formulas. The resultant equations should then be simplified by using the arithmetic properties. The final conversion stage includes the implementation of these simplified equations using hardware components, such as full adders and logic gates. Based on Euclid's theorem:

$$gcd(a, b) = gcd(b, a \mod b), a > b \tag{13}$$

Hence,

$$gcd(2^{2n+2}, 2^{n+1}-1) = gcd(2^{n+1}-1, 1) = 1 \tag{14}$$
$$gcd(2^{2n+2}, 2^n-1) = gcd(2^n-1, 1) = 1 \tag{15}$$
$$gcd(2^{n+1}-1, 2^n-1) = gcd(2^n-1, 1) = 1 \tag{16}$$

The following propositions are used to obtain the closed form expressions, which will be the means to compute the multiplicative inverses, based on the MRC algorithm.

Proposition 1 *The multiplicative inverse of* (2^{2n+2}) *modulo* $(2^{n+1}-1)$ *is* $k1 = 1$

Proof One has

$$|2^{2n+2}|_{2^{n+1}-1} = 1 \tag{17}$$

□

Proposition 2 *The multiplicative inverse of* (2^{2n+2}) *modulo* (2^n-1) *is* $k_2 = 2^{n-2}$.

Proof One has

$$|2^{n-2} \times 2^{2n+2}|_{2^n-1} = |2^{3n}|_{2^n-1} = 2^{n-2} \tag{18}$$

□

Proposition 3 *The multiplicative inverse of* $(2^{n+1}-1)$ *modulo* (2^n-1) *is* $k_3 = 1$.

Proof One has

$$|2^{n+1}-1|_{2^n-1} = 1 \tag{19}$$

Therefore, by assuming the values $< k_1 = 1, k_2 = 2^{n-2}, k_3 = 1, P_1 = (2^{2n+2}), P_2 = (2^{n+1}-1), P_3 = (2^n-1) >$ in (8–11), we have:

$$\begin{aligned} X &= x_1 + P_1(V_2 + V_3P_2) = x_1 \\ &\quad + (2^{2n+2})(V_2 + (2^{n+1}-1)V_3) \end{aligned} \tag{20}$$

$$V_1 = x_1 \tag{21}$$

$$V_2 = |(x_2 - x_1)|P_1^{-1}|_{p_2}|_{p_2} = |(x_2 - x_1)|_{2^{n+1}-1} \tag{22}$$

$$\begin{aligned} V_3 &= |((x_3 - x_1)|P_1^{-1}|_{p_3} - V_2)|P_2^{-1}|_{p_3}|_{p_3} \\ &= |2^{n-2} \times (x_3 - x_1) + (-V_2)|_{2^n-1} \end{aligned} \tag{23}$$

□

According to following two properties, the Eqs. (20–23) can be simplified to decrease the hardware complexity:

Property 1 *The residue of a negative residue number* $(-v)$ *in modulo* $((2^{n-1}))$ *is the one's complement of* v, *where* $0 \leq v < (2^{n-1})$ [32].

Property 2 *The multiplication of a residue number* v *by* 2^p *in modulo* $((2^{n-1}))$ *is carried out by* P *bit circular left shift, where* P *is a natural number.*

In order to design an efficient reverse converter, the expressions (20) and (22)–(23) can be rewritten as follows:

$$\begin{aligned} V_2 &= |(x_2 - x_1)|P_1^{-1}|_{p_2}|_{p_2} = |(x_2 - x_1)|_{2^{n+1}-1} \\ &= |x_2|_{2^{n+1}-1} + |-x_1|_{2^{n+1}-1} = V_{21} + V_{22} \end{aligned} \tag{24}$$

where

$$V_{21} = |x_2|_{2^{n+1}-1} = x_{2,n}x_{2,n-1} \cdots x_{2,0} \tag{25}$$

$$\begin{aligned} V_{22} &= |-x_1|_{2^{n+1}-1} = -\{x_{1,n}, x_{1,n-1} \cdots x_{1,0}\} \\ &\quad -\{x_{1,2n+1}, x_{1,2n} \ldots x_{1,n+1}\} \end{aligned} \tag{26}$$

Now, in order to implement V_3 on the basis of (23), we have

$$\begin{aligned} V_3 &= |((x_3 - x_1)|P_1^{-1}|_{p_3} - V_2)|P_2^{-1}|_{p_3}|_{p_3} \\ &= |2^{n-2} \times (x_3 - x_1) + (-V_2)|_{2^n-1} \\ &= |2^{n-2} \times x_3|_{2^n-1} + |-2^{n-2} \times x_1|_{2^n-1} + |-V_2)|_{2^n-1} \\ &= V_{31} + V_{32} + V_{33} \end{aligned} \tag{27}$$

Table 3 Characterization of each part of the proposed reverse converter

Parts	FA	NOT	OR/XNOR	Delay
OPU(1)	–	$(2n+2)$	–	t_{Not}
CSA(1)	$(n+1)$	–	–	t_{FA}
CPA(1)	$(n+1)$	–	–	$(2n+2)t_{FA}$
CSA(2)	n	–	–	t_{FA}
CSA(3)	2	–	$(n-2)$	t_{FA}
OPU(2)	–	$(n+1)$	–	t_{Not}
CSA(4)	n	–	–	t_{FA}
CSA(5)	1	–	$(n-1)$	t_{FA}
CPA(2)	n	–	–	$(2n)t_{FA}$
OPU(3)	–	n	–	t_{Not}
R-CPA	n	–	$(n+1)$	$(2n+1)t_{FA}$

where

$$V_{31} = |2^{n-2} \times x_3|_{2^n-1} = x_{3,1}x_{3,0}x_{3,n-1} \ldots x_{3,2} \tag{28}$$

$$\begin{aligned} V_{32} &= |-2^{n-2} \times x_1|_{2^n-1} \\ &= -\{x_{1,1}x_{1,0}x_{1,n-1}, \ldots, x_{1,2}\} \\ &\quad -\{x_{1,n+1}x_{1,n}x_{1,2n-1}, \ldots, x_{1,n+2}\} \\ &\quad -\{x_{1,2n+1}x_{1,2n}0, \ldots, 0\} \end{aligned} \tag{29}$$

$$\begin{aligned} V_{33} &= |-V_2|_{2^n-1} = -\{V_{2,n-1}, \ldots, V_{2,1}, V_{2,0}\} \\ &\quad -\{0, \ldots, 0, V_{2,n}\} \end{aligned} \tag{30}$$

Finally, to obtain X based on (20), we have

$$\begin{aligned} X &= x_1 + P_1(V_2 + V_3P_2) \\ &= x_1 + (2^{2n+2})(V_2 + (2^{n+1}-1)V_3) \\ &= x_1 + (2^{2n+2})C \end{aligned} \tag{31}$$

$$C = V_2 + (2^{n+1}-1)V_3 \tag{32}$$

$$\begin{aligned} C &= \{(V_{3,n-1}, \ldots, V_{3,0})(V_{2,n})(\bar{V}_{3,n-1}, \ldots, \bar{V}_{3,0})\} \\ &\quad + \{(1, 1, \ldots, 1)(V_{2,n-1}, \ldots, V_{2,0})\} \end{aligned} \tag{33}$$

$$X = x_1 + (2^{2n+2})C \tag{34}$$

The area and delay specifications for the proposed reverse converter are shown in Table 3. From this Table, we have

$$\begin{aligned} \text{Total area} &= (6n+5)A_{FA} + (4n+3)A_{NOT} \\ &\quad + (3n-2)A_{OR} + (3n-2)A_{XNOR} \end{aligned} \tag{35}$$

$$\text{Total delay} = (6n+8)t_{FA} + 3t_{NOT} \tag{36}$$

In the proposed moduli set, different values can be selected for n. If n equals 3:

$$\begin{aligned} &\{2^{2n+2}, 2^{n+1}-1, 2^n-1, 2^{2n+2}+1, 2^{2n+3}-1\} \\ &\to \text{if}(n=3)\text{then}\,\{P_1, P_2, P_3, P_4, P_5\} \\ &= \{256, 15, 7, 257, 511\} \end{aligned} \tag{37}$$

where initial number is $22817 (X = 22817)$,

$$(x_1, x_2, x_3, x_4, x_5) = (33, 2, 4, 201, 333)$$

Sender transmits $(33, 2, 4, 201, 333)$ instead of X, and the value of X will be calculated as follows at the destination:

In order to design a residue to binary converter, we first need to obtain the multiplicative inverse values and substitute these values with the modulus set in the conversion algorithm formulas The multiplicative inverse of $(2^{2n+2} = 2^{2*3+2} = 256)$ modulo $(2^{n+1} - 1 = 2^{3+1} - 1 = 15)$ is $k_1 = 1$.

$$|2^{2n+2}|_{2^{n+1}-1} = |256|_{15} = 1$$

The multiplicative inverse of $(2^{2n+2} = 2^{2*3+2} = 256)$ modulo $(2^n - 1 = 2^3 - 1 = 7)$ is $k_2 = 2^{n-2} = 2^{3-2} = 2$.

$$|2^{n-2} \times 2^{2n+2}|_{2^n-1} = |2^{3n}|_{2^n-1} = |2^{3*3}|_{2^3-1} = |512|_7 = 1$$

The multiplicative inverse of $(2^{n+1} - 1 = 2^{3+1} - 1 = 15)$ modulo $(2^n - 1 = 2^3 - 1 = 7)$ is $k_3 = 1$.

$$|2^{n+1} - 1|_{2^n-1} = |2^{3+1} - 1|_{2^3-1} = |15|_7 = 1$$

Now, values of mixed-radix coefficients are calculated:

$$\begin{aligned} V_1 &= x_1 \rightarrow V_1 = 33 = (100001)_2 \\ V_2 &= |(x_2 - x_1)|P_1^{-1}|_{P_2}|_{P_2} \\ &= |x_2 - x_1|_{2^{n+1}-1} = |x_2|_{2^{n+1}-1} + |-x_1|_{2^{n+1}-1} \\ &= V_{21} + V_{22} \end{aligned}$$

where

$$\begin{aligned} V_{21} &= |x_2|_{2^{n+1}-1} = x_{2,n}x_{2,n-1}\ldots x_{2,0} \rightarrow V_{21} \\ &= |2|_{15} = (0010) \\ V_{22} &= |-x_1|_{2^{n+1}-1} = -\{x_{1,n}x_{1,n-1}\ldots x_{1,0}\} \\ &\quad - \{x_{1,2n+1}x_{1,2n}\ldots x_{1,n+1}\} \\ &\rightarrow V_{22} = -(\{0001\} + \{0010\}) = -\{0011\} = 1100 \\ V_2 &= V_{21} + V_{22} = 0010 + 1100 = (1110)_2 = 14 \\ V_3 &= |((x_3 - x_1)|P_1^{-1}|_{P_3} - V_2)|P_2^{-1}|_{P_3}|_{P_3} \\ &= |2^{n-2} \times (x_3 - x_1) - V_2|_{2^n-1} \\ &= |2^{n-2} \times x_3|_{2^n-1} + |-2^{n-2} \times x_1|_{2^n-1} + | \\ &\quad - V_2|_{2^n-1} = V_{31} + V_{32} + V_{33} \end{aligned}$$

where

$$\begin{aligned} V_{31} &= |2^{n-2} \times x_3|_{2^n-1} = (x_{3,1}x_{3,0})(x_{3,n-1}\ldots x_{3,2}) \\ &= (x_{3,1}x_{3,0})(x_{3,2}) = 001 \\ V_{32} &= |-2^{n-2} \times x_1|_{2^n-1} \\ V_{32} &= -\{x_{1,1}x_{1,0}x_{1,n-1},\ldots,x_{1,2}\} \\ &\quad - \{x_{1,n+1}x_{1,n}x_{1,2n-1},\ldots,x_{1,n+2}\} \\ &\quad - \{x_{1,2n+1}x_{1,2n}0,\ldots,0\} \\ V_{32} &= -2(\{001\} + \{100\}) \\ &= -(\{010\} + \{010\}) = -\{011\} = 100 \\ V_{33} &= |-V_2|_{2^n-1} = -\{V_{2,n-1},\ldots,V_{2,1},V_{2,0}\} \\ &\quad - \{0,\ldots,0,V_{2,n}\} \\ &= -(\{110\} + \{001\}) = -\{111\} = 000 \\ V_3 &= V_{31} + V_{32} + V_{33} \\ &= \{001\} + \{100\} + \{000\} = (101)_2 = 5 \end{aligned}$$

Finally, to obtain initial number in destination, we have

$$\begin{aligned} X &= x_1 + P_1(V_(2) + V_3P_2) \\ &= 33 + 256(14 + 5 \times 15) = 22817 \end{aligned}$$

As shown, initial number obtained at the destination equals the number sent from the origin. For the moduli set $\{2^{2n+2}, 2^{n+1} - 1, 2^n - 1\}$, a residue to binary converter was proposed. For hardware implementing this converter, FAs, Carry Propagation Adder (CPA) with EAC and Carry Save Adder (CSA) with end around carry (EAC) [33,34] are exploited. The hardware implementation of proposed converter is illustrated in Fig. 1.

The performance of proposed reverse converter for new moduli set $\{2^{2n+2}, 2^{n+1} - 1, 2^n - 1\}$ is compared with the reverse converters with a lower dynamic range and in terms of hardware requirement and speed of operations. This comparison is shown in Table 4.

Characterization of Each Part of the Proposed Reverse Converter: The hardware architecture of the proposed reverse converter for the 3-moduli set $\{2^{2n+2}, 2^{n+1} - 1, 2^n - 1\}$ is shown in Fig. 1. The implementation is based on (25, 28, 33, 35). As shown in Eq. (25), to implement V_2, it is required to add up Eqs. (26) and (27) each having $(n + 1)$ bits. Thus, a $(n + 1)$-bit CPA with EAC and a $(n + 1)$-bit CSA with EAC are required. (CSA1 in the second row and CPA 1 in the third row of Table 3).

Regardless of the number of bits, CSAs constantly have a delay of t_{FA}. Also, the delay of k bits CPA with EAC is $2kt_{FA}$. Thus, the delay of a $(n + 1)$-bit CPA1 equals $(2n + 2)t_{FA}$. As shown in Eqs. (25), (26) and (27) are added up (each one has (n+1) bits) so that a (n+1)-bit CPA with EAC is required (CPA 1 in the second row of Table 3). Also in Eq. (16), $(2n + 1)$ NOT gates are required (OPU1 in the first row of Table 3).

Also, $(2n + 2)$ NOT gates are required (OPU1 in the first row of Table 3). Due to the fact that these NOT gates operate parallel (regardless of their numbers), the delay is always t_{NOT}. For hardware implementation of V_3 in Eq. (28), it is required to add up Eqs. (29), (30) and (31). To add up these equations, we need to use some

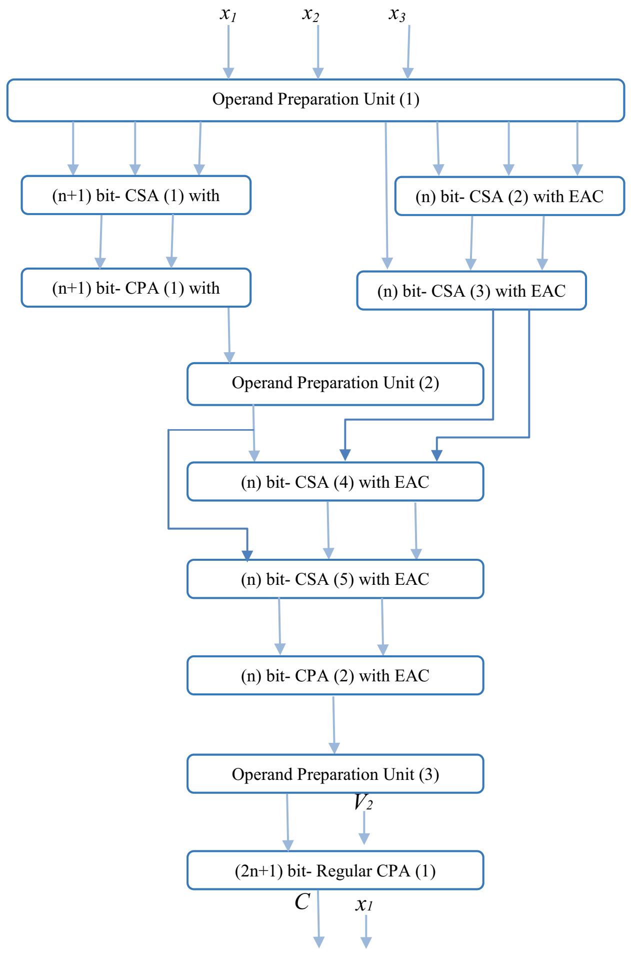

x_1
x_2
x_3
Operand Preparation Unit (1)
(n+1) bit- CSA (1) with
(n) bit- CSA (2) with EAC
(n+1) bit- CPA (1) with
(n) bit- CSA (3) with EAC
Operand Preparation Unit (2)
(n) bit- CSA (4) with EAC
(n) bit- CSA (5) with EAC
(n) bit- CPA (2) with EAC
Operand Preparation Unit (3)
V_2
(2n+1) bit- Regular CPA (1)
C
x_1

Fig. 1 Residue to binary converter for moduli set $\{2^{2n+2}, 2^{n+1} - 1, 2^n - 1\}$

Table 4 Area and delay comparison

Reverse converter	DR	Area (A_{FA})	Delay (t_{FA})
[35]-1	$4n + 1$	$9n + 5$	$11.5n + 6$
[35]-2	$4n + 1$	$8n + 4$	$9n + 6$
[35]-3	$4n + 1$	$n^2 + 12n + 12$	$16n + 22$
[35]-4	$4n + 1$	$9n + 10$	$11n + 14$
[18]	$4n + 1$	$10n + 6$	$7.5n + 11$
[36]	$4n$	$26n + 8$	$7n + 8$
[37]-1	$4n + 1$	$n^2/2 + 3.5n$	$11n + 8$
[37]-2	$4n + 1$	$n^2 + 10n + 3$	$9n + 6$
[38]-CICE	$4n$	$2.5n^2 + 25.5n + 12$	$18n + 23$
[38]-CIHS	$4n$	$2.5n^2 + 37.5n + 28$	$12n + 15$
[38]-C2CE	$4n$	$20n + 17$	$13n + 22$
[38]-C2HS	$4n$	$42n + 61$	$7n$
[38]-C3CE	$4n$	$23n + 11$	$16n + 14$
Proposed	$4n + 3$	$6n + 5$	$6n + 8$

CSAs with EAC. The number of required CSAs with EAC depends on the number of inputs. Each CSA with EAC includes 3 inputs and 2 outputs. Since there are 6 n-bit inputs, 4 n-bit CSAs with EAC are used (CSA2-CSA 3-CSA4-CSA 5); and finally one n-bit CPA with EAC is employed too. In Eqs. (30) and (31), for $(n - 1)$ bits and $(n - 2)$ bits with constant value of 1, it is possible to employ OR/XNOR instead of full adder gates to decrease costs. The number of negative bits should equal that of NOT gates. OPU 2 prepares the required operands for Eq. (31).

To implement Eq. (34), it is required to use a regular CPA. Regarding to $(n+1)$ value1,(n+1) OR/XNOR gates are used instead of $(n+1)$ full adder. Since n NOT gates are required, OPU 3is used. These parts have been shown in the last 2 rows of Table 3. To compute $(2^{2n+2})C$ using Eq. (35), $(2n + 2)$ bits of 0 should be placed to the right of C and then be added to x_1. Since x_1 contains $(2n + 2)$ bits, the value of x_1 is placed to the right of C. Thus, only a concatenation is performed. To calculate all hardware area, it is required to add up hardware cost of implementation of V_2, V_3, and C.

To evaluate the cost of reverse converters, only the cost of used Full Adder at reverse converters are compared since their area is greater than that of other parts of the reverse converter. To compare delays, only the delay of Full Adders is considered since the delay of not gates is less than Full Adders. The cost of the proposed reverse converter and that of other reverse converters (reverse convertors represented in [18,35–38]) are compared in Table 4. This comparison revealed that the proposed reverse converter had smaller area (number of Full Adders). Therefore, energy consumption and hardware volume is reduced significantly. Also, the delay of the proposed reverse converter was seen to be comparatively less.

3.3 RDTP design

This section explains how RDTP operates. As mentioned earlier, RDTP can be implemented on either tree-like network topologies or cluster based network topologies.

In the rest of this paper, the environment for implementing RDTP is assumed to be a tree-like network topology. After sensing data, each node calculates the residues of sensed values with respect to moduli set $\{2^{2n+2}, 2^{n+1} - 1, 2^n - 1, 2^{2n+2} + 1, 2^{2n+3} - 1\}$. Then, each node sends residues, smaller than initial numbers, to its parent. To illustrate this, suppose that a node receives value X from environment (let $X = 5142$). Instead of sending X to the parent, the node will send residues $(x_1, x_2, x_3, x_4, x_5)$ which are computed using the following equation.

$$x_i = X \bmod P_i = |X|_{p_i}, 1 \leq i \leq 5 \tag{38}$$

As explained in the example represented in Sect. 3.2, given that

$$\{P_1, P_2, P_3, P_4, P_5\} = \{256, 15, 7, 257, 511\}$$

$$\left.\begin{array}{r} x_1 = 5142 \mod 256 = 22 \\ x_2 = 5142 \mod 15 = 12 \\ x_3 = 5142 \mod 7 = 4 \\ x_4 = 5142 \mod 257 = 2 \\ x_5 = 5142 \mod 511 = 32 \end{array}\right. \rightarrow (x_1, x_2, x_3, x_4, x_5) = (22, 12, 4, 2, 32)$$

The MRC algorithm facilitates the RNS to decimal conversion for the numbers. Since the proposed moduli set is composed of three main moduli, the original number might be recovered if three residues are available. After receiving residues, the parent node recovers initial number using MRC algorithm and three residues (it is assumed to be Y). Subsequently, the below values can be calculated:

$$y_4 = Y \bmod P_4, \quad y_5 = Y \bmod P_5 \tag{39}$$

If $y_4 = x_4$ and $y_5 = x_5$, the node concludes that no error has occurred; otherwise, the error correction subroutine is called. The total number of ways, by which three moduli could be selected among 5 moduli, is $10(\binom{5}{3} = \frac{5!}{3!2!} = 10)$. In the error correction subroutine, the parent node derives original number for all ten states. During data transmission, some residues may be distorted due to errors; thus all 10 numbers may not be correct and they are not equal. As a result, the numbers which are beyond the dynamic range will be eliminated (this range is calculated by multiplication of main moduli $\prod_{i=1}^{3} p_i$). The rest of numbers are voted and the number with the most frequency is selected as the final number. The same process is separately conducted for the data received from each child node. Then, data is aggregated and the result is derived. The resulted number is divided by moduli set and the residues are transmitted to the parent node. The procedure continues until the data reaches to the sink node.

This paper, has considered the employment of managing inventory control [39] in wireless sensor networks. This employment is based on counting operation; thus in aggregation, the sum operator is applied. In RNS, some operations such as addition, multiplication and subtractions might be rapidly performed in parallel. Therefore, the aggregation is performed at rapid speed and in turn, reduces end-to-end delay.

The RDTP algorithm in the source node (algorithm 1), forwarder node (algorithm 2) and the sink node (algorithm 3) are demonstrated in the following.

Algorithm 1: Actions at the source node

```
1: Sensing data from the environment
2: Calculating the residues resulted from dividing sensed
   data by moduli set
3: Sending residues to the parent node
```

Algorithm 2: Actions at the forwarder node

```
1: Receiving residues from children
2: Calculating original data from received residues using
   MRC algorithm
3: If error is detected then call Error Correction subroutine:
    a: calculating all numbers resulted from received
       residues using MRC algorithm
    b: among calculated numbers, omitting those larger than
       dynamic range.
    c: among remained numbers, selecting the most frequent
       one as the original number.
4: Dividing the number derived in step 3 by moduli set and
   calculating residues.
5: If the data is received from $n$ nodes, calculating $|\sum_{j=1}^{n} Residue_{ij}|_{p_i}$
   for all moduli ($i = 1, 2, , 5$). ($Residue_{ij}$
   in this equation is $i^{th}$ resiude from $j^{th}$ node)
6: Sending value obtained in step 4, to the parent node.
```

Algorithm 3: Actions at sink node

```
1: Receiving residues from children
2: Calculating original data from received residues using
   MRC algorithm
3: If error is detected then call Error Correction subroutine:
    a: calculating all numbers resulted from received
       residues using MRC algorithm
    b: among calculated numbers omitting those larger than
       dynamic range.
    c: among remained numbers, selecting the most frequent
       one as the original number.
```

4 Performance evaluation

In this section, the performance of RDTP is assessed using NS-2 simulation tool [40]. Moreover, RDTS [14], PSFQ [12] and the method proposed by Wu et al. [17] are compared to RDTP. PSFQ is an ARQ based method and two other methods are FEC. For our simulation, a 150 m × 150 m area is considered, where 200 nodes are randomly distributed. The simulation time is considered to be 100 seconds, while antenna model is omni-directional. The initial energy dedicated in the nodes, is 0.25J and the interface queue type is assumed to be Droptail. The mentioned methods are compared considering total energy consumption, end-to-end delay and packet delivery ratio. The simulations are repeated

Table 5 Simulation parameters

Parameters	Value
Area of sensor field	150 × 150m^2
Antenna model	Omnidirectional
Radio range of a sensor node	75 m
Number of sensor nodes	200
Number of sink nodes	1
Initial energy	0.25 J
Interface queue type	Droptail
Interface queue (IFQ) Length	50 packets
Energy model	Battery
Data rate	2.5 Kb/s
Amount of the sender nodes	100

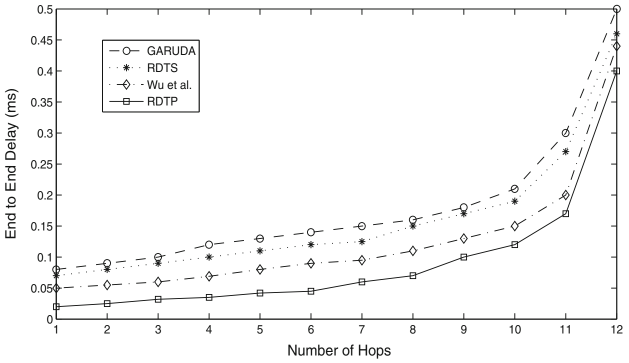
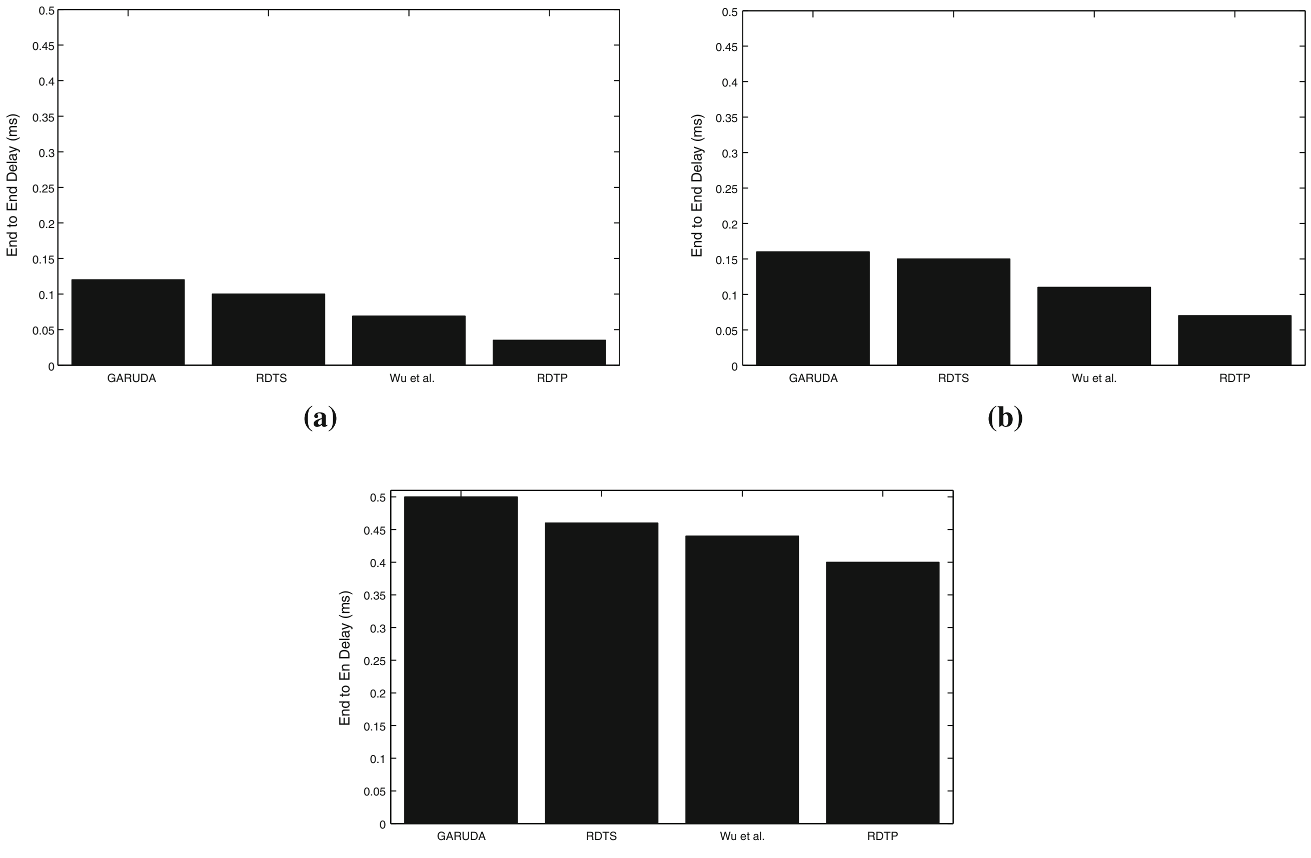


Fig. 3 Comparing the end to end delay of RDTP as a function of hops count

several times and the average results are reported. Table 5 presents the simulation parameters.

To begin, RDTP is compared to previous three methods, considering end-to-end delay. In Fig. 2a–c, the number of hops are considered to be 4, 8 and 12, respectively. Figure 3 shows a comparison between RDTP and other methods, from the end-to-end delay viewpoint. Evidently, our proposed method provides a much less end-to-end delay, compared to other mentioned methods.

Figure 4 depicts a comparison between energy consumption of these methods. It can be seen that RDTP is superior in this aspect, as well. By changing the number of hops, the packet delivery ratio has been examined. As illustrated in Fig. 5, in all circumstances, RDTP is superior to its counterparts from the viewpoint of packet delivery ratio.



(a)

(b)

(c)

Fig. 2 Comparing the End to End delay for RDTP as a function of hops count; **a** 4 hops, **b** 8 hops, **c** 12 hops

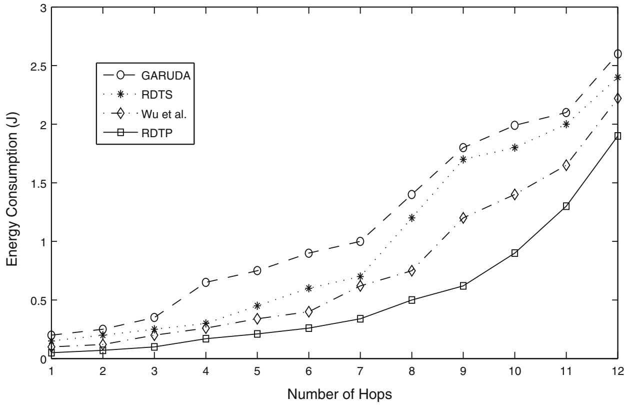


Fig. 4 Comparing the energy consumption of RDTP as a function of hops count

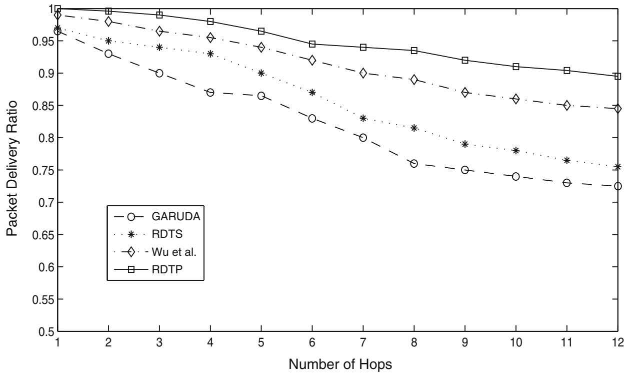


Fig. 5 Comparing the packet delivery ratio of RDTP as a function of hops count

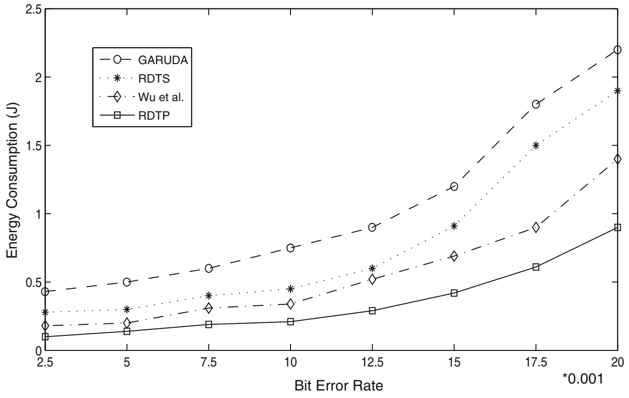


Fig. 6 Comparing the energy consumption of RDTP as a function of bit error rate

Changing number of hops, packet delivery ratio was examined. As illustrated in Fig. 5, in all circumstances RDTP is superior to its counterparts from packet delivery ratio perspective.

Then, different values are considered for bit error rate and the proposed method is evaluated with varying values of bit error rate. Figure 6 shows this comparison.

The simulation results reveal that end-to-end delay is significantly reduced by RDTP. Besides, the proposed method (RDTP) causes a magnificent decrease of energy consumption, which leads to a rise in packet delivery ratio.

5 Conclusion

In this paper, a reliable data transport protocol (RDTP) for Wireless Sensor Networks has been proposed. It might be implemented on either tree-like or cluster based network topologies. The proposed scheme is categorized as Forward Error Correction codes and it employs hop-by-hop error control. To guarantee the error correction and detection, RRNS and an efficient moduli set of $\{2^{2n+2}, 2^{n+1}-1, 2^n-1, 2^{2n+2}+1, 2^{2n+3}-1\}$ was employed. This moduli set has lower volume of required hardware. Moreover, it imposes less delay to the system. The performance of RDTP was evaluated using NS-2 simulation tool. It was also compared to PSFQ, RDTS and the scheme proposed by Wu et al. [17]. The simulation results demonstrate that the proposed method has lower end-to-end delay, while it has higher packet delivery ratio. Moreover, the simulation results illustrated that RDTP has lower energy consumption and its packet delivery ratio is acceptable.

References

1. Mahmood, M. A., Seah, W. K., & Welch, I. (2015). Reliability in wireless sensor networks: A survey and challenges ahead. *Computer Networks*, *79*, 166–187.
2. Fortino, G., Bal, M., Li, W., & Shen, W. (2015). Collaborative wireless sensor networks: Architectures, algorithms and applications. *Information Fusion*, *22*, 1–2.
3. Hammoudeh, M., & Newman, R. (2015). Adaptive routing in wireless sensor networks: QoS optimisation for enhanced application performance. *Information Fusion*, *22*, 3–15.
4. Mitra, S., & Roy, A. (2015). Communication void free routing protocol in wireless sensor network. *Wireless Personal Communications*, *82*(4), 2567–2581.
5. Rault, T., Bouabdallah, A., & Challal, Y. (2014). Energy efficiency in wireless sensor networks: A top-down survey. *Computer Networks*, *67*, 104–122.
6. Somasundara, A., Kansal, A., Jea, D. D., Estrin, D., & Srivastava, M. B. (2006). Controllably mobile infrastructure for low energy embedded networks. *IEEE Transactions on Mobile Computing*, *5*(8), 958–973.
7. Shah, K., Di Francesco, M., Anastasi, G., & Kumar, M. (2011). A framework for resource-aware data accumulation in sparse wireless sensor networks. *Computer Communications*, *34*(17), 2094–2103.
8. Vuran, M. C., & Akyildiz, I. F. (2009). Error control in wireless sensor networks: a cross layer analy-sis. *IEEE/ACM Transactions on Networking*, *17*(4), 1186–1199.
9. Xie, P., Zhou, Z., Peng, Z., Cui, J. H., & Shi, Z. (2010). SDRT: A reliable data transport protocol for underwater sensor networks. *Ad Hoc Networks*, *8*(7), 708–722.
10. Iyer, Y. G., Gandham, S., & Venkatesan, S. (2005, October). STCP: A generic transport layer protocol for wireless sensor networks.

In *Proceedings of the 14th international conference on computer communications and networks* (pp. 449-454), ICCCN 2005. New York: IEEE Press.

11. Marchi, B., Grilo, A., & Nunes, M. (2007, July). DTSN: Distributed transport for sensor networks. In 1*2th IEEE symposium on computers and communications* (pp. 165–172), ISCC 2007.
12. Wan, C. Y., Campbell, A. T., & Krishnamurthy, L. (2005). Pump-slowly, fetch-quickly (PSFQ): A reliable transport protocol for sensor networks. *IEEE Journal on Selected Areas in Communications*, *23*(4), 862–872.
13. Mahmood, M. A., & Seah, W. K. (2012). *Reliability in wireless sensor networks: Survey and challenges ahead*. Wellington: School of Engineering and Computer Science, Victoria University of Wellington.
14. Srouji, M. S., Wang, Z., & Henkel, J. (2011, December). RDTS: A reliable erasure-coding based data transfer scheme for wireless sensor networks. In 2011 *IEEE 17th international conference on parallel and distributed systems (ICPADS)* (pp. 481–488).
15. Stann, F., & Heidemann, J. (2003, May). RMST: Reliable data transport in sensor networks. In *Proceedings of the first IEEE international workshop onsensor network protocols and applications* (pp. 102–112). IEEE.
16. Park, S. J., Sivakumar, R., Akyildiz, I. F., & Vedantham, R. (2008). GARUDA: Achieving effective reliability for downstream communication in wireless sensor networks. *IEEE Transactions on Mobile Computing*, *7*(2), 214–230.
17. Wu, C., Ji, Y., Xu, J., Ohzahata, S., & Kato, T. (2014). Coded packets over lossy links: A redundancy-based mechanism for reliable and fast data collection in sensor networks. *Computer Networks*, *70*, 179–191.
18. Hosseinzadeh, M., Molahosseini, A. S., & Navi, K. (2008). An improved reverse converter for the moduli set $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$. *IEICE Electronics Express*, *5*(17), 672–677.
19. Garner, H. L. (1959). The residue number system. *IRE Transactions on Electronic Computers*, *2*, 140–147.
20. Navi, K., Molahosseini, A. S., & Esmaeildoust, M. (2011). How to teach residue number system to computer scientists and engineers. *IEEE Transactions on Education*, *54*(1), 156–163.
21. Conway, R., & Nelson, J. (2004). Improved RNS FIR filter architectures. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *51*(1), 26–28.
22. Bajard, J. C., & Imbert, L. (2004). A full implementation RSA in RNS. *IEEE Transactions on Computers*, *53*(6), 769–774.
23. Ramrez, J., Garca, A., Meyer-Baese, U., & Lloris, A. (2002). Fast RNS FPL-based communications receiver design and implementation. In *Field-programmable logic and applications: Reconfigurable computing is going mainstream* (pp. 472–481). Berlin: Springer Heidelberg.
24. Taylor, F. J. (1984). Residue arithmetic: A tutorial with examples. *Computer*, *17*(5), 50–62.
25. Sengupta, A., & Natarajan, B. (2014). Redundant residue number system based space-time block codes. *Physical Communication*, *12*, 1–15.
26. Barsi, F., & Maestrini, P. (1973). Error correcting properties of redundant residue number systems. *IEEE Transactions on Computers*, *100*(3), 307–315.
27. Kinoshita, E., & Lee, K. J. (1997). A residue arithmetic extension for reliable scientific computation. *IEEE Transactions on Computers*, *46*(2), 129–138.
28. Haron, N. Z., & Hamdioui, S. (2011). Redundant residue number system code for fault-tolerant hybrid memories. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, *7*(1), 4.
29. Matutino, P. M., Chaves, R., & Sousa, L. (2014). An efficient scalable RNS architecture for large dynamic ranges. *Journal of Signal Processing Systems*, *77*(1–2), 191–205.
30. Modiri, S., Movaghar, A., & Barati, A. (2012). Study of error control capability for the new moduli set $\{2^{2n+1} + 2^n - 1, 2^{2n+1} - 1, 2^n - 1, 2^{2n}, 2^{2n+1} - 1\}$. *Journal of Advanced Computer Science & Technology*, *1*(4), 176–186.
31. Modiri, S., Movaghar, A., & Barati, A. (2012). An efficient reverse converter for the new modulus set $\{2^{2n+2}1, 2^{2n+1}1, 2^n\}$. *International Journal of Advanced Research in Computer Science and Software Engineering*, *2*(8), 447–452.
32. Piestrak, S. J. (1995). A high speed realization of a residue to binary converter. *IEEE Transactions on circuits and systems. II, Analogue and digital Signal Processing*, *42*(10), 661–663.
33. Molahosseini, A. S., Navi, K., Dadkhah, C., Kavehei, O., & Timarchi, S. (2010). Efficient reverse converter designs for the new 4-moduli sets and based on new CRTs. *IEEE Transactions on Circuits and Systems I: Regular Papers*, *57*(4), 823–835.
34. Wang, W., Swamy, M. N. S., Ahmad, M. O., & Wang, Y. (2000). A high-speed residue-to-binary converter for three-moduli $(2^k, 2^{k-1}, 2^{k-1} - 1)$ RNS and a scheme for its VLSI implementation. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, *47*(12), 1576–1581.
35. Mohan, P. A., & Premkumar, A. B. (2007). RNS-to-binary converters for two four-moduli sets $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ and $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} + 1\}$. *IEEE Transactions on Circuits and Systems I: Regular Papers*, *54*(6), 1245–1254.
36. Sheu, M. H., Lin, S. H., Chen, C., & Yang, S. W. (2004). An efficient VLSI design for a residue to binary converter for general balance moduli $\{2^n - 3, 2^n - 1, 2^n + 1, 2^n + 3\}$. *IEEE Transactions on Circuits and Systems II: Express Briefs*, *51*(3), 152–155.
37. Cao, B., Srikanthan, T., & Chang, C. H. (2005). Efficient reverse converters for four-moduli sets $\{2^n - 1, 2^n, 2^n + 1, 2^{n+1} - 1\}$ and $\{2^n - 1, 2^n, 2^n + 1, 2^{n-1} - 1\}$. *IEE Proceedings-Computers and Digital Techniques*, *152*(5), 687–696.
38. Mohan, P. A. (2008). New reverse converters for the moduli set $\{2^n - 3, 2^n - 1, 2^n + 1, 2^n + 3\}$. *AEU-International Journal of Electronics and Communications*, *62*(9), 643–658.
39. Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). A survey on sensor networks. *IEEE Communications magazine*, *40*(8), 102–114.
40. Network simulator (ns-2). http://www.isi.edu/nsnam/ns/.

Ali Barati received the B.Sc. degree in Computer Hardware Engineering and M.Sc. degree in Computer Software Engineering in 2001 and 2004 respectively. He is currently working toward the Ph.D. degree in Computer Software Engineering at Department of Computer Engineering and Information Technology, Qazvin Branch, Islamic Azad University, Qazvin, Iran. Currently he is faculty of Department of Computer Engineering, Dezful Branch, Islamic Azad University, Dezful, Iran. Between 2007 and 2011 he was appointed as the Head of Department of Computer Science at the University. His research interests include mobile Ad-Hoc networks, wireless sensor networks, high speed networks and fault tolerant systems.

Ali Movaghar is a Professor in the Department of Computer Engineering at Sharif University of Technology in Tehran, Iran and has been on the Sharif faculty since 1993. He received his B.S. degree in Electrical Engineering from the University of Tehran in 1977, and M.S. and Ph.D. degrees in Computer, Information, and Control Engineering from the University of Michigan, Ann Arbor, in 1979 and 1985, respectively. He visited the Institut National de Recherche en Informatique et en Automatique in Paris, France and the Department of Electrical Engineering and Computer Science at the University of California, Irvine in 1984 and 2011, respectively, worked at AT&T Information Systems in Naperville, IL in 1985–1986, and taught at the University of Michigan, Ann Arbor in 1987–1989. His research interests include performance/dependability modeling and formal verification of wireless networks and distributed real-time systems. He is a senior member of the IEEE and the ACM.

Masoud Sabaei received his B.Sc. degree from Esfahan University of Technology, Esfahan, Iran, and his M.Sc. and Ph.D. form Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran, all in the field of Computer Engineering in 1992, 1995 and 2000, respectively. Dr. Sabaei has been professor of Computer Engineering Department, Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran since 2002. His research interests are wireless networks, software defined networks, and telecommunication network management.