FOREX Publication
Open Access | Rapid and quality publishing

International Journal of
Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

Enhancing Data Security through Hybrid Error Detection: Combining Cyclic Redundancy Check (CRC) and Checksum Techniques

Adham Hadi Saleh[1*] and Mohammed Sami Mohammed[2]

[1]*Department of Electronic Engineering, University of Diyala, 32001 Diyala, Iraq; adham.hadi@yahoo.com*
[2] *Department of Computer, University of Diyala, 32001 Diyala, Iraq; dr.mohammed.sami@uodiyala.edu.iq*

***Correspondence:** Adham Hadi Saleh; adham.hadi@yahoo.com

ABSTRACT- Error detection is a critical aspect of ensuring the accuracy of data transmission in communication systems. In this study, the performance of two error detection techniques has been investigated when combined to achieve a Bit Error Rate of $10^{-5}$for single and multiple error detection ability. The two techniques studied were Cyclic Redundancy Check and Checksum with a new combination process. This proposed method showed that when CRC and Checksum were combined, the overall error detection performance significantly improved compared to using either technique alone. Specifically, the combined technique was able to achieve a BER of 10^{-5} for 6 given examples with higher accuracy and lower false positive rates. These findings demonstrate the potential benefits of combining error detection techniques to enhance the reliability of data transmission systems. These combinations were demonstrate using both VHDL and Python to identify the unexpected behavior of system before its utilization. The combination process provides 72 bits only for memory usage with 1 millisecond to finish checking and detecting process. These steps calculations and waveform are simulated using python for verification process based on overall combination steps. In addition, this paper provided a novel method for polynomial generation depending on the IP addresses of trusted sites. This evaluation of CRC generator was unique and provide double steps of protection for users in same or different networks.

Keywords: Checksum, CRC, IP addresses, VHDL, Error Detection, and Error Correction.

ARTICLE INFORMATION

Author(s): Adham Hadi Saleh and Mohammed Sami Mohammed;
Received: 18/05/2024; **Accepted**: 16/07/2024; **Published**: 25/07/2024;
e-ISSN: 2347-470X;
Paper Id: IJEER 1805-16;
Citation: 10.37391/IJEER.120312
Webpage-link:
https://ijeer.forexjournal.co.in/archive/volume-12/ijeer-120312.html

Publisher's Note: FOREX Publication stays neutral with regard to Jurisdictional claims in Published maps and institutional affiliations.

1. INTRODUCTION

Error detection methods utilized for data transmission process involving polynomial generation and division in a mathematical procedure like Cyclic Redundancy Check (CRC). To demonstrate the integrity of sending and receiving data as a matching mechanism based on the reminder which obtained after division. CRC calculations start with a critical polynomial selection which the capability of CRC depends on these selections. CRC needs an additional bit of zeroes to equalize the degree of generated polynomial as a first step. Second step took a place as an XOR operation to generate the checksum of final CRC results. Code word would be created after the previous two steps to transmit these results to the next sides by repeating the same overall process based on the received bits. Transmitted data would be zero error for sender and receiver when the reminder equals zero. Otherwise, the reminder with any one's value would refer to error detection. Several types of communications protocols like Wi-Fi and Ethernet require techniques to provide accurate and reliable transmission process as explained in [1-3]. In the other hand, Checksum which is considered as another mechanism of error detection, but adding all that required sending bites to be sent after. In the receiving side, additional process will occur for all received bites as a similar sending process based on the sum comparison between both sides. These mechanisms are shown with a different data type as explained in [4-5]. Internet protocol (IP) needs to treat data as a 16 bits' sequence by adding these data according to one's compliment as shown in [6]. Some of author presented a modify check mechanism to use 32 bits instead of 16 bits as in Alder-32 through applying modular arithmetic to the obtained sum as more explained in [7]. Due to detect fail of Checksum and errors occurred in this mechanism, it needs a modification or a techniques combination like CRC combination. This combination gives CRC and Checksum when makes together a fast and efficient detection by improve Checksum faults detection with a CRC best privilege. In [8], authors explained more details about the minimization time of CRC ability based on its own XOR or operation that is considered as an impact privilege of CRC. The capability of CRC burst error multiple detection that happened through noisy bits as explained for DRAM multiple bit error detection in [9]. Through the presented works of authors and researchers, CRC utilized for variety communication services such in controlling the network system based on ID database as in [10-11]. Additional combination of techniques mixed with CRC to provide faster

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

and accurate detection such as forward Error Correction (FEC) or an optimized table such in [12-13]. Error secured in storage system or in communication channels cannot be detected based on CRC only due to incapability of random or non-detection error. Authors developed CRC mechanism to provide better performances with 5G as an application, which explained in [14]. The issue of CRC-inability to detect the specific position of error even if error detected by CRC, which it need needs to fix this issue through automation detection and recovery process as explained by author [15]. As mentioned before, storage systems need a massive data or high-speed network requirement that makes CRC unsuitable for such an application. CRC provide several privileges for short type of dataset as short packet due to size depending as explained for 5G and multiplies with more definition in [16-18]. In the other hand, Checksum needs a minimal time and power which provide a fast detection process with verification steps as well. These characteristics make Checksum suitable and applicable for real time usage as used by researchers in file transmission, encryption operation of cloud and for an automation discovery in [19-21]. Also, Checksum requires a minimum capacity of memory with less computational resources compared to other techniques which make it suitable in software or hardware applications. Authors presented a low-cost system with real online system checker as in [22]. Flipping of data bits, file transmission process is widely combined and applied with Checksum as introduced by authors in [23-25]. However, if two different set of data produce a similar value of Checksum results which lead to fall positive detection. Due to the Checksum issues, authors suggested embedded system within neural network as well as a parallel system presented in [26-27] to cope these properties.

2. PROPOSED METHODS FOR ERROR DETECTION COMBINATION

A robust detection system for error detection system can be produced through several type combinations like CRC Checksum either privately applied or comparing results. Authors can compare multiple techniques to demonstrate the result with the matching values of CRC and the Checksum as in [28] with 32 bits using FPGA. Another technique such as parity and Hamming code were mixed to enhance the error detection as in [29-31] for optical computers. FEC can be also applied with error detection as a combination process for several application as utilized for 5G or comparative study review and [32-33]. This combination process is wildly utilized especially for both CRC and Checksum to prove the integrity of communication data system and to investigate this combination, completely complexity as in [34-35]. Combination process of this work was a presented to provide a robust detection system with more investigation of this process. This combination can be explained and listed in more details for both CRC and Checksum as follow:

- Divided sending data it into equal size of blocks.
- Regardless of any resulting carries, the sum of these packets is evaluated.
- Complement of the results will be taken and added to the original data as a final step of the first method.
- Then polynomial generation of CRC techniques with binary representation, if the CRC length is not match, then zeros will be added to sending data.
- Obtaining the reminder of CRC division.
- Then both sending code and the data will be transmitted to other side to repeat the process for those validations.

This combination omitted the issues occurred in both CRC and the Checksum, which also reducing the undeleting error probability on the Checksum detection. Networking is not the only field that might need a novel and fast detection methods but also telecommunications and data storage fields as well. Fast approaches assist system to ensure data integrity and allow faster transmission when error occurred as explained in [36]. File transmission needs a swift and smooth operation search for traditional FTB or trivial FTB to verify file correctness or not. Memory controllers also employ CRC to compare data if corrupted or not such as particularly applied for RAM due to fault models as mentioned and [37]. In addition, during read and write operation for storage devices like hard drives, solid-state drives (SST) or even for optical disc requires detection and verification. Also, corrupted packets identified using CRC through wireless protocols such as a Bluetooth and Wi-Fi that trigger appropriate measures to make data retransmission if needed. Fast method is required particularly where data needs to be securely stored and verified search for applications need digital signatures or cryptographic process. Due to CRC simplicity and effectiveness that makes it widely utilized and employed in various domain where error detection is across crucial. In the same topics, Checksum detectors are also employed for different applications. CRC in TCP/IP packet detection and large file storage process as well. For example, in Redundant Array Independent Discs (RAID)S, Checksum hired for error detection across multiple disk drives. Checksum also employed in data backup and recovery systems to check and verify if data was successfully backed up and accurately restored if required. An important network field is IP checking process which is less presented for the approaches for different variances like IP-v4 which is utilized to identify and locate devices on network. Huge numbers of IP can be created reach to 4.3 billion addresses with 32 bits composed of four sets ranging from 0 to 255 such as explained more and [38-39]. Despite IP-v4 limited addresses number and the adoption process of other longer virgin such as IP-v6, IP-v4 still widely used and supported across the net [40-41]. This address is considered as legitimate without molecules activity involved when obtained from reputable resources like Regional Internet Registry (RIR) which is clearly explained in [42]. To prevent computers from excepting any communications or transmission through suspicious IP, checking these IP's can protect users from hijacking. Matching addresses between networks should be considered as a novel issue that ought to be solved to provide information of whether IP flagged for any suspicious activity. This investigation would be helpful to implement monitoring and logging mechanism for malicious activity tracking that associated with the received IP-v4 addresses as in [43-44].

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

2.1 Hybrid Error Detection: CRC and Checksum Combination

Mathematical operations that involved in CRC is polynomial generator which is considered as a CRC crucial component. CRC size determination is based on polynomial degree which represented in binary form. Data will be divided by this value and if reminder value obtained, it will be added to data before transmission. Register set have zero initialization values; these contents will be updated after each process at the time. The same process is done for both sides, sending on receiving sides as explained in [45 -47]. Common transmission errors that occur in Wireless Sensor Network (WSN), can be detected based on CRC due to its simplicity of effectiveness. WSN requirements specify CRC polynomial selection, which makes data frame include CRC in their building structures like other Internet layers. When, WSN node is wants to send data, CRC will be calculated according to selected polynomial that should provide a good balance between detection capabilities and computational efficiency. In the receive side, WSN may employee error recovering mechanism if and only if error was detected such as data retransmission required or requested again and explain and [48-50]. While in GSM networks, polynomial selection is a standardized and implemented within GSM frames that employed to the burst of data sent over the air interface. Frame retransmission may be requested by the receiving side when error occur such in WSN area. CRC polynomial is defined in GSM specification and might be very depending on GSM standardization such as GSM 1800. Practically, CRC and polynomial process implemented in GSM, module or by a chip available in basis stations or even in mobile devices, as in [51-52]. While for IOT, CRC polynomial selection will be based on the requirements of IOT application. The selection of polynomial is a critical and depends on some factors like detection capabilities, computational efficiency as in WSN and GSM. Similar process utilized in receiving and sending side with appropriate error handling mechanism and re-transmission if needed as explained for this field in [53-55]. Such as v4, does not employed error detection by itself. IP-v4 relies on higher layer, such as TCP to handle errors, transport layer implement Checksum within their models. While link-layer like Ethernet employed CRC, which built in Ethernet frames that includes also Mac addresses, type field and CRC field with CRC-32 polynomial as in [56-57]. *Figure 1* shows the unique and new method for creating and generating polynomial generator of CRC.

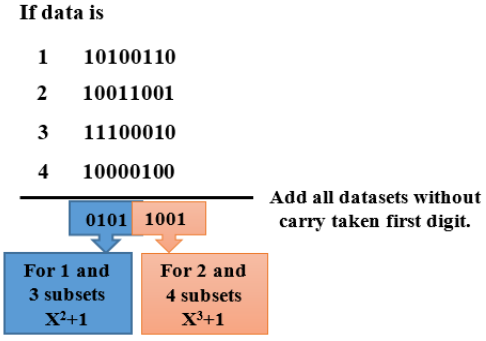


Figure 1. CRC Double Polynomial Generation methods for IP-v4

The steps of combination process for the proposed method are shown in *figure 2* and listed as follow:

- Selection process: IPV4.0 selection.
- Addition: Add 8 bits to each other without carry calculations.
- Division process: divided the first and the third of (8-bits of IPV4) by G1 and the second and the fourth of (8-bits of IPV4) by G2 to specify the reminder with original data addition. When data is less than G1 or G2 then data will stay as original data. In case of all zeroes results or the divider is less than divined, a new polynomial generator will create a new one until getting a reminder.
- Combination step: Applying Checksum for all the three cases to generate the sending and receiving complement number.
- Repeating: Repeating the same process for the receiver (by vice versa) with zero reminding results which mean the word is accepted for the first inner step.
- Finalization step: Back to CRC by divided the new data with its specific divider to get zero reminder which approve the second step of CRC technique.

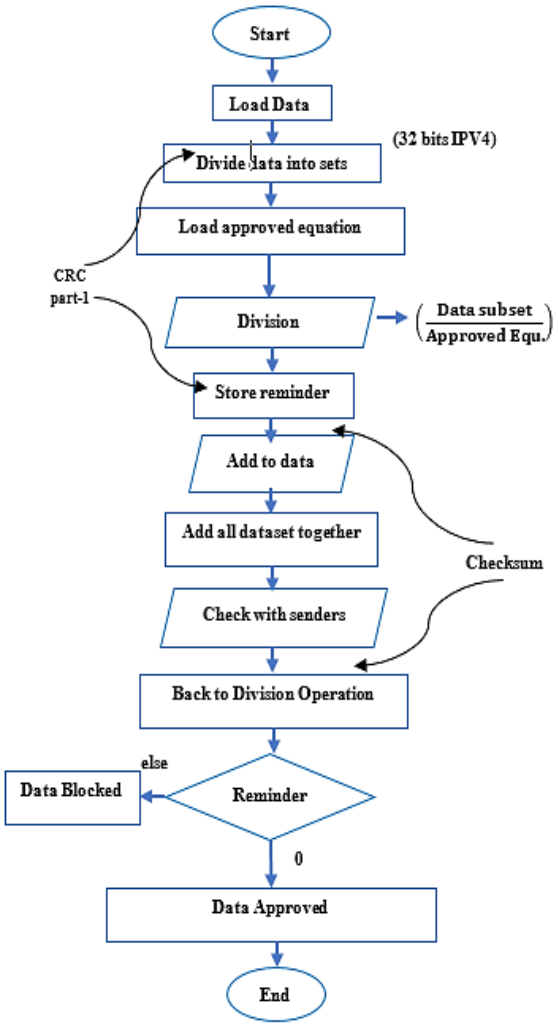


Figure 2. The proposed method combinations between CRC + Checksum

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

These steps were applied for an example which is shown in *figure 3*, six steps were applied and obtaining the same results at sending and receiving side as well.

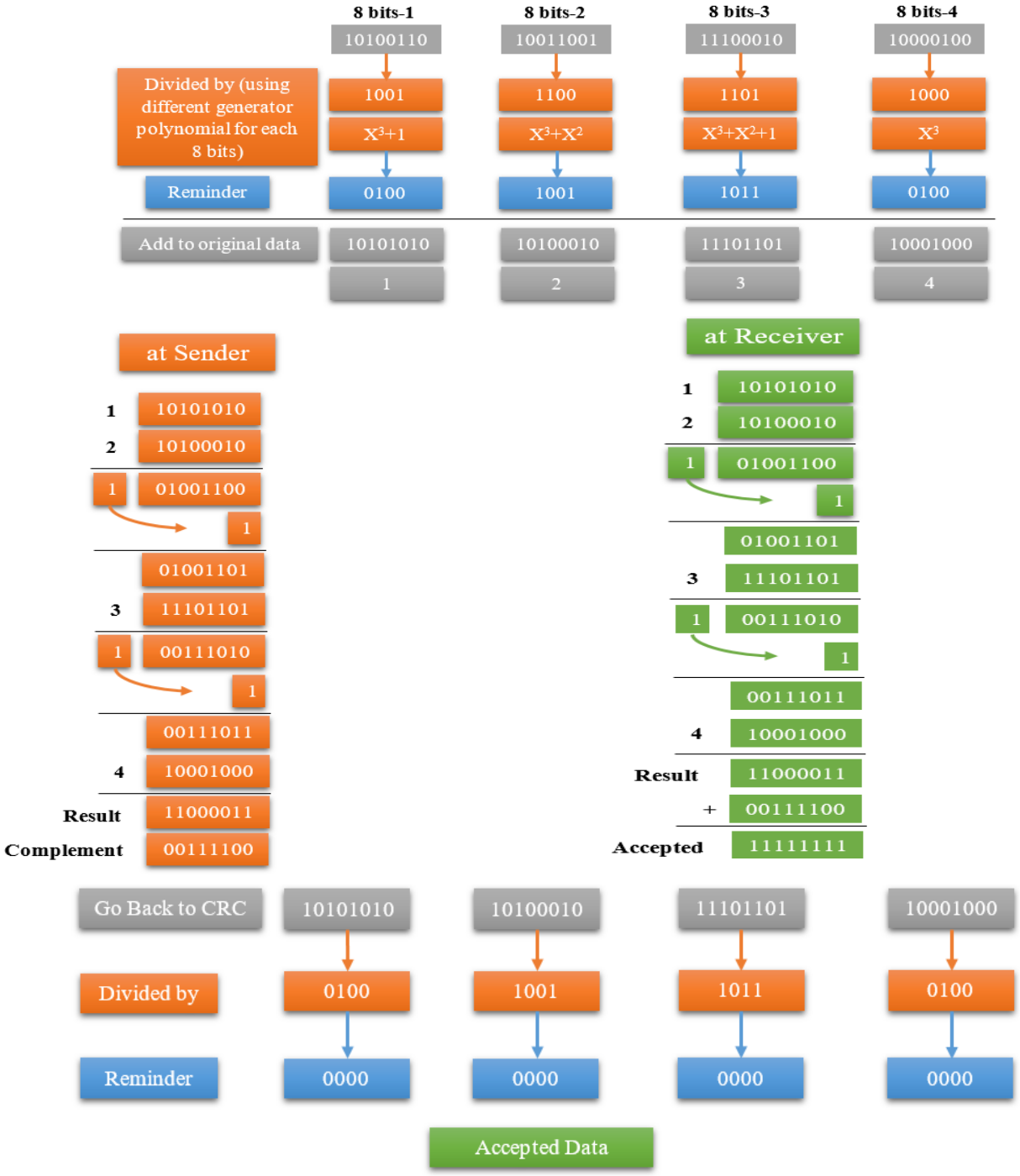


Figure 3. CRC+ Checksum proposed method applied for IP-v4 example.

2.2 Scenario of Combined Methods

In case of data integrity is critical aspect in addition to detection speed, Combination between CRC and checksum will be practically beneficial. The Advantages of the proposed method Could be listed based on researchers works as in in *figure 3*. Combination scenario Can enhance the overall error detection Capability Certain type of errors can be detected based on checksum techniques while CRC has the ability of detecting wider range of errors due to simple sum and division approach respectively. By this method data Corruption or transmission failures can be avoided by adding an additional Layer of fault tolerance when both CRC techniques are applied. Also, using both techniques can balance both performance, and Computational Cost, where checksum is light weigh with fast

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

computation and CRC is more intensive case. These means, the proposed method is suitable for certain applications where real time/low overhead is crucial. Combination approach provide system ability to mitigate the risk of undetected errors that might another slipping by a single method. This mean another privilege of these combination to increase reliability of data transmission in addition, different utilization techniques scenario provides different use Cases, which adapt to varying requirements without and need to depend on one technique that could be less effective for certain Cases. Several applications and cases where combination between these techniques can be successful better than a Single technique utilization. In Wireless Communication for example, data packets are prone to errors due to interferences which can be treated by using these Combined. This ability was due to CRC error detection occurred in contiguous bits and fast detection property of checksum. In storage system as well, when system need integrity characteristic to prevent losing data. This was due to CRC detection of complex error in Read/Write operation and due to fast checksum process before storage is finisher while in limited processing power or limited memory size that consider at constraints for embedded systems and IoT, Combination method can balance the cost with a fast detection. During initial transmission, CRC Can lead the overall process for detecting errors while Checksum provide quick verification of data integrity. In fields of durability cored long term integrity are paramount such and for backup system, these combination helps ensure that related data remains intact over extended period. Due to this combination, CRC protect against Sophisticated errors which Compromise data over time and through retrieval process, checksum will provide immediate checking.

2.3 Scenario with Specific Error Detection Methods

Some specific scenarios and case where combination of these techniques provide more robust system than using the technique alone based on these cases. In (TCP/IP) networking for example when large files transmitted over the internet by FTP as mentioned in [58]. CRC was applied to check the entire data packet, while Checksum was utilized for verifying smaller segment of the overall packet. In Storage System for example when multiple disks are applied to store or retrieve data redundantly for both methods as well like explained in [59]. CRC was applied in these fields to verify the integrity of data blocks across disks, while Checksum ensure the accurate calculations. Also, in Wireless Sensor Networks when a temperature sensor used to send data periodically about environment to a server by any type of network topology like mesh as presented in [60]. CRC was applied for verifying reading integrity before transmission and Checksum in turn utilized to verify sequence numbers of packets to ensure correctly received. Embedded systems as well when have controlling commands across a vehicle and internal network by sensors that send data to Engine Control Unit (ECU) as in [61-62]. CRC was applied to approve the integrity of large data frames through sensors readings or diagnosis information, while Checksum has provided a quick verification of smaller packet related to real time commands. In addition to all these fields, data achieving has also utilized with these techniques as well, for example when users have historical document achieving with multimedia content as in [63]. CRC was applied during initial process to verify entire files, while Checksum used for individual files or single section validation.

3. RESULTS

For the first step, identifying the IP version 4 to be applied which is most used for internet communications. The given IP's have 32 bits in length and divided into subsets of four groups as shown in *table 1*. The selection of these examples due to that most internet devices used these protocols or even near to these values.

For the second step, take two 8-bit binary numbers that you want to add together. For example, let us use 10110101 and 01101011. Start from the rightmost bit and add each pair of bits together. Continue this process for each pair of bits, if you reach the leftmost bit and still have a carry, exclude that in the final sum as shown in *table 2*. In this case and based on above example, 10110101 + 01101011 without carry calculations would result in 00011100.

Table 1: IP-v4 selection for the proposed method

IP V4.0 Address	Binary Conversion
192.168.1.20	11000000.10101000.00000001.00010100
8.8.8.8	00001000.00001000.00001000.00001000
172.16.0.0	10101100.00010000.00000000.00000000
192.168.0.0	11000000.10101000.00000000.00000000
192.168.5.255	11000000.10101000.00000101.11111111
10.12.2.85	00001010.00001100.00000010.01010101

Table 2: Addition Process related to the four sets regardless the result carry

8 bits addition	Polynomial generator
0111-1101	$G1=x^2+x+1$, $G2= x^3+x^2+1$.
0010-0000	G1=x, G2=0.
1011-1100	$G1=x^3+x+1$, $G2= x^3+x^2$.
0110-1000	$G1=x^2+x$, $G2= x^3$.
0110-1100	$G1=x^2+x$, $G2= x^3+x^2$.
0110-1101	$G1=x^2+x$, $G2= x^3+x^2+1$.

As a third step process, by dividing the first and the third numbers by G1, while second and fourth by G2. For example, considering the input would be a set of data: 5, 13, 20, and 185. Then divide the first and third numbers (5 and 20) by G1 and the second and fourth numbers (13 and 185) by G2. After that reminder calculations for each division process if available or not. This process shown in Table 3 for selective IP from *table 1 and 2.*

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

Table 3: Division process of Numbers by G1 and G2

Division Reminder	New Data (Added Reminder to original data)
0011-1100-0000-0111	11000011-10110100-00000000-00011011
0111-0100-0000-0000	10110011-00011100-00000000-00000000
0100-0000-0000-0111	00001110-00001100-00000010-01011100

The final sending calculation step was the Checksum process, involves calculating a checksum value for the data being transmitted to ensure the integrity of data during transmission as shown in *table 4*.

Table 4: Applying Checksum to the results of step 3

Checksum Addition Results	Complement
10010010	01101101
11001111	00110000
0110110	1001001

Repeating the same process for the receiver by submitting the number for validation and checks if it meets the criteria for acceptance. The system would accept it without the need for any reminders. This signifies that the word has been approved for the first inner step of the validation process. The receiver can proceed with the next steps in the validation process knowing that the number has been accepted without any further reminders. Back to CRC, dividing the new number with its specific divisor in the CRC process. The new contribution of this work is to generate a polynomial function depending on the data and provided more security and complexity to original data. In addition, applying hybrid techniques to detect error for IPV4 to give more secure sending and receiving data. Also, this process takes a short time for finding the results and check the wrong data at specified bits. The overall process was done also using python to demonstrate the low cost, low usage, and high speed of code execution. These steps are repeated as well in both sides and gaining the same results with a 100 of full iterations. The used memory was for each binary addition and for all other related calculations based on equation below.

Total memory used = Binary Numbers + Binary addition + Result + Other carries (if available)

Total memory used = 32 bits + 16 bits + 8 bits + 8 bits + 8 bits = 72 bits

While program speed was determined by operations complexity and data size, which involves in this case basic strings and simple mathematical operations. The size of input data is also small, for that reason the proposed method provides a fast running of about milliseconds or even less. Combined error detection methods were implemented using Python for CRC and Checksum. The results of the error detection process were then visualized in a pulses mode showing the different steps taken in the error detection process. Each step was applied to the data to detect any errors present in these steps based on the transmitted IP-v4. *Figure 4*, *5*, *6*, *7* and *figure 8*. showed the data before and after error detection steps, highlighting any discrepancies or errors that were successfully detected and corrected. This approach to error detection ensures a high level of accuracy in the transmission of data, which is be applied again for the receiving side.

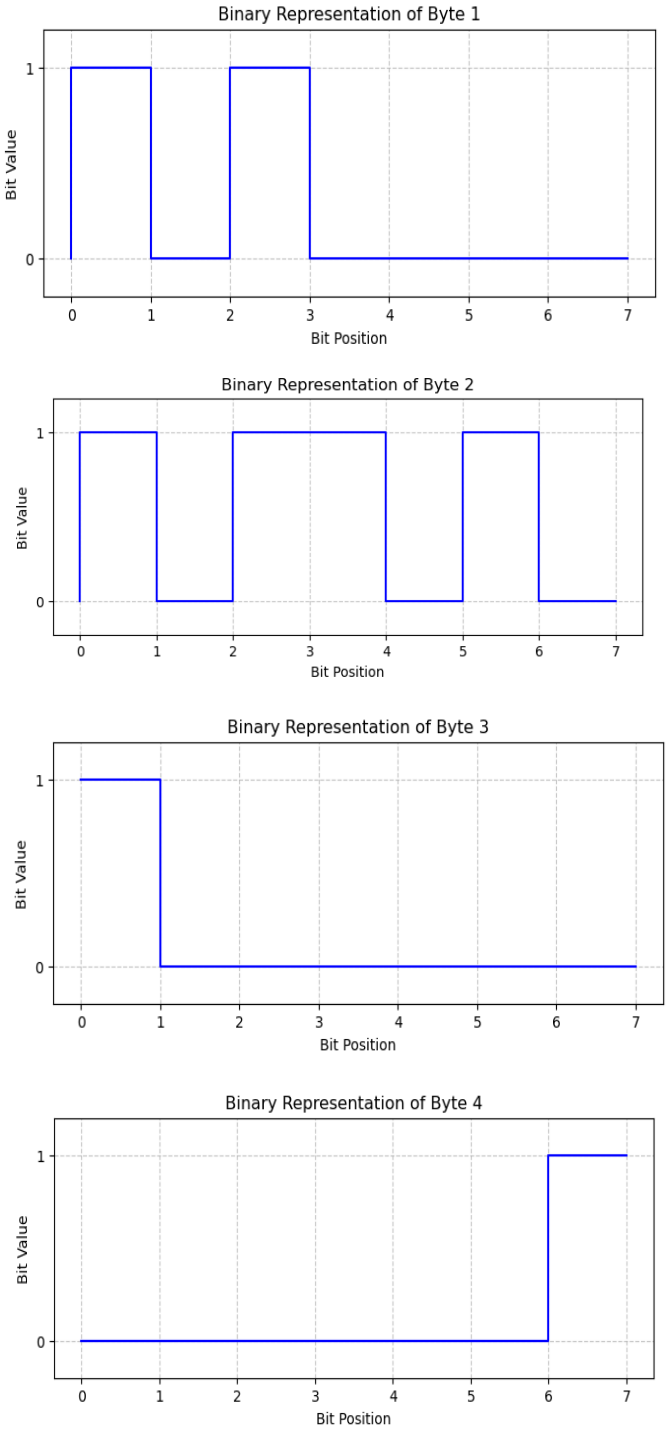


Figure 4. A 32 bits' conversion process presented as a pulse waves.

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

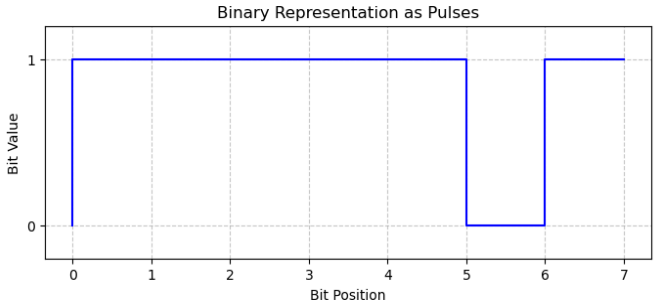


Figure 5. Addition process of all bytes without carry for G1 and G2 polynomial generation values

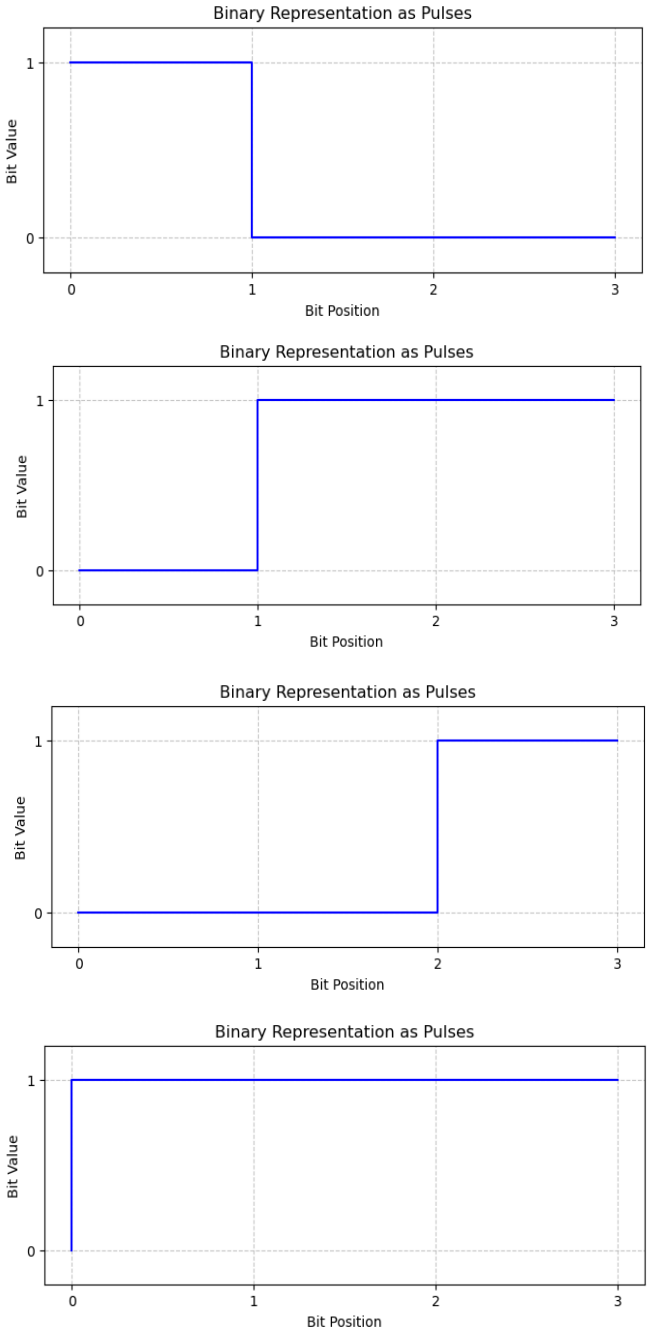


Figure 6. Division process (the first and the third by G1) and (the second and the fourth by G2)

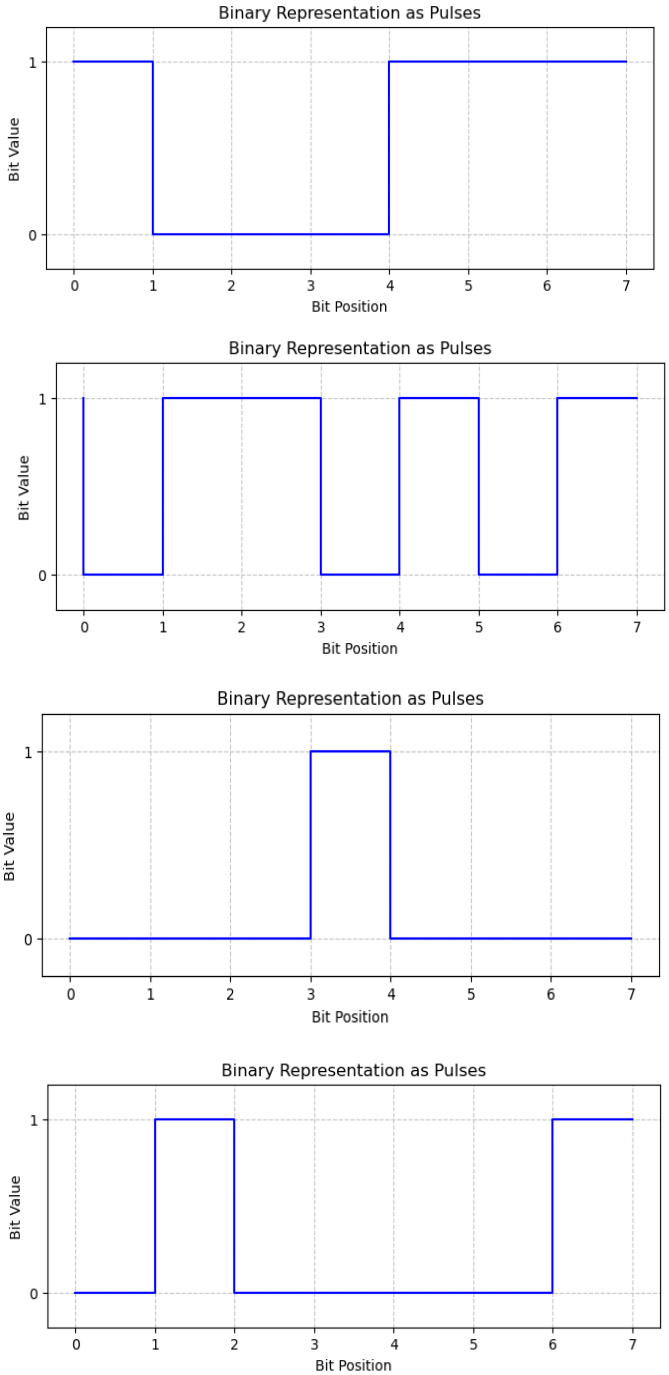


Figure 7. Add the reminder to the original data

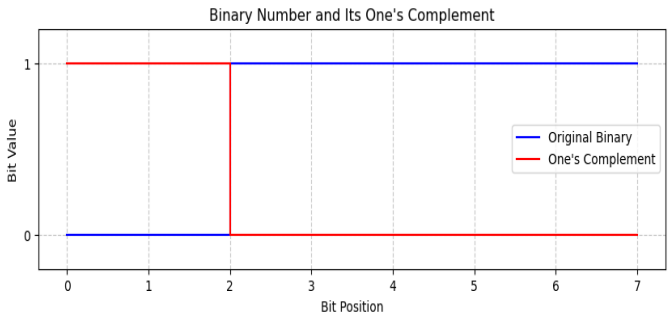


Figure 8. Add overall sets regarding carry based on Checksum with its 1st complement.

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

CRC + Checksum algorithms add an additional layer of security to the data transmission process, converting it to a robust system against errors. The new method suitable for use in applications where data integrity is critical such as in communication systems, network protocols and storage devices. The transmitter circuit design with VHDL language which contain the block diagram, internal structure, and the time simulation are shown in *figures 9* and *figure 10.* Where the transmitter pins description is shown in *table 5.*

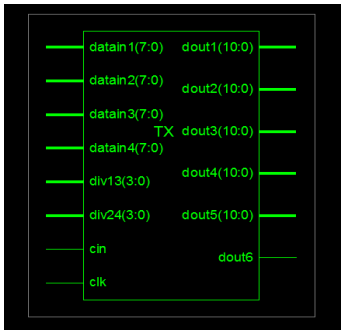


Figure 9. Transmitter circuit block diagram

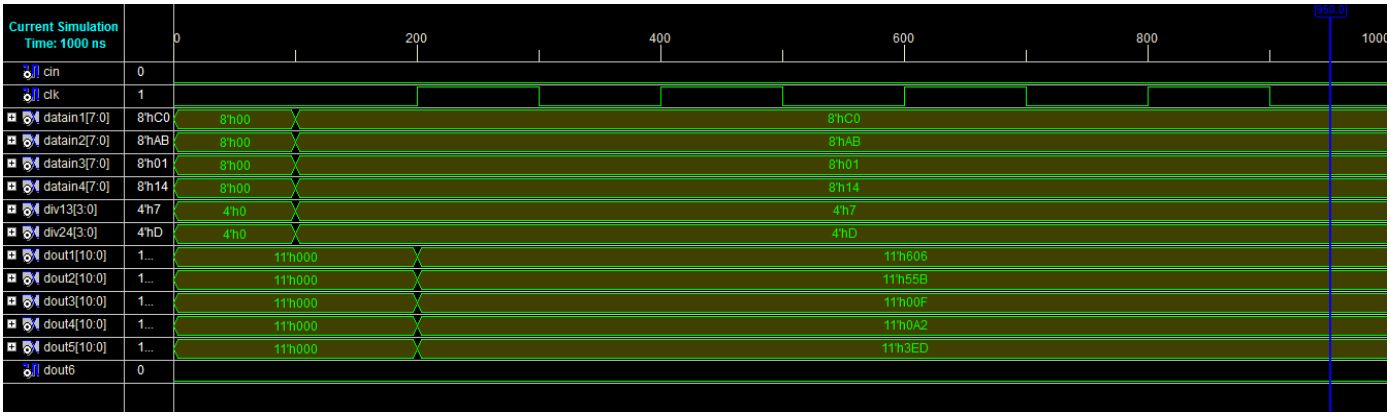


Figure 10. Time simulation of the transmitter circuit data

Table 5: The transmitter pin description

Transmitter Pins	Description	Receiver Pins	Description
datain1	1st Input block data stream with 8 bits	clk	Clock signal
Datain2	2nd Input block data stream with 8 bits	dout1	Output transmitted data 11 Bits 1st block data stream.
Datain3	3rd Input block data stream with 8 bits	dout2	Output transmitted data 11 Bits 2nd block data stream.
Datain4	4th Input block data stream with 8 bits	dout3	Output transmitted data 11 Bits 3rd block data stream.
DIVISOR13	CRC require input divisor for data packet 1 and 3.	dout4	Output transmitted data 11 Bits 4th block data stream.
DIVISOR24	CRC require input divisor for data 2 and 4 packet.	dout5 and dout6	Check sum statues with 12 Bits

On the other hand, the receiver circuit is design in VHDL language which contain the receiver bock circuit, the receiver internal circuit, and the time simulation of a receiver circuit with different states as shown in *figures 11* and *figure 12.* Where the receiver pins description is shown in *table 6.*

This system can have detected the errors for each byte separately and send feedback signal (RETRAINS) for each byte, so the system not required to resend all data only the error byte, that make system more utilization as compare with other techniques. As shown in *figures 13 to16.* The design summary of the transmitter and receiver circuit is shown in *table 7.*

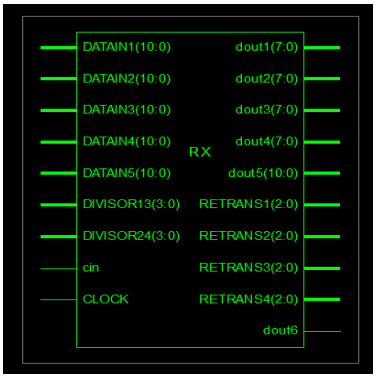


Figure 11. Receiver circuit block diagram

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

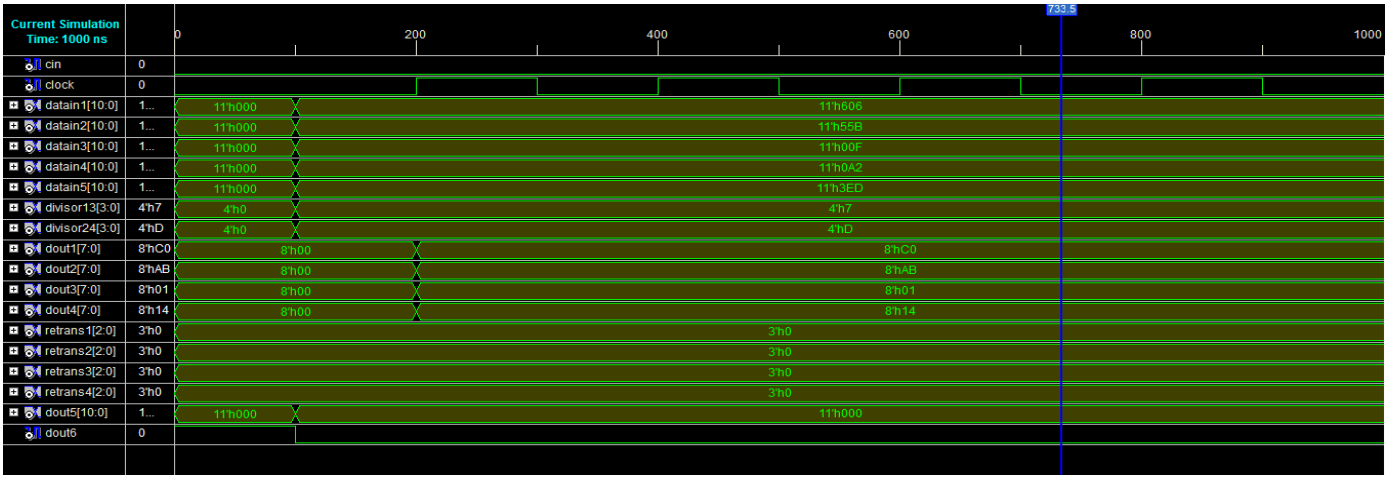


Figure 12. Time simulation of the receiver circuit with corrected received data

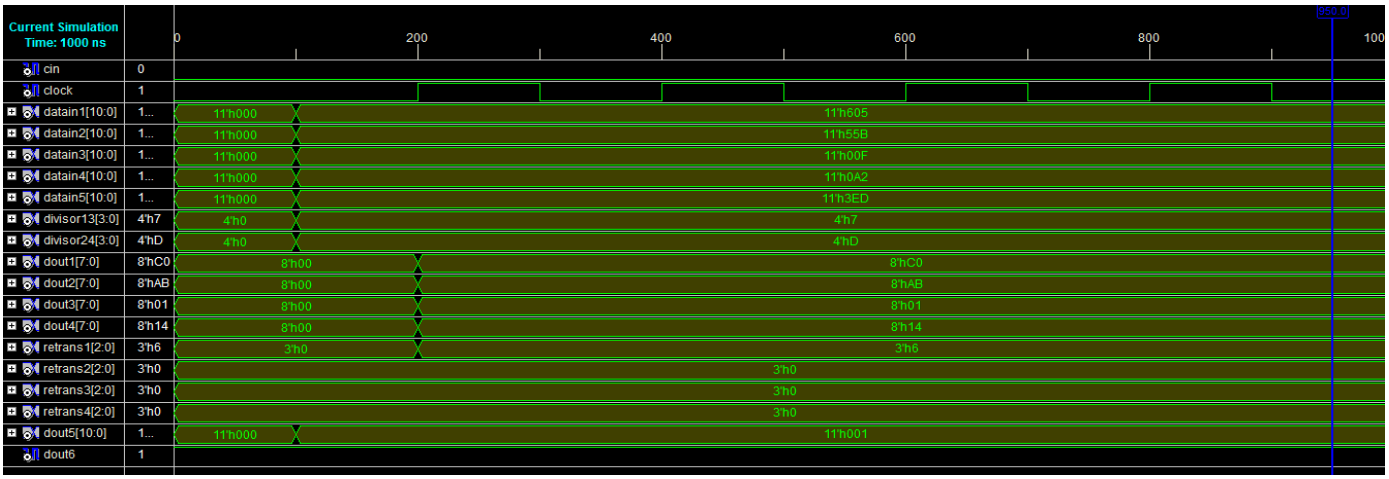


Figure 13. Time simulation of the corrupted received data at first byte (retrains1 is 6 and not 0)

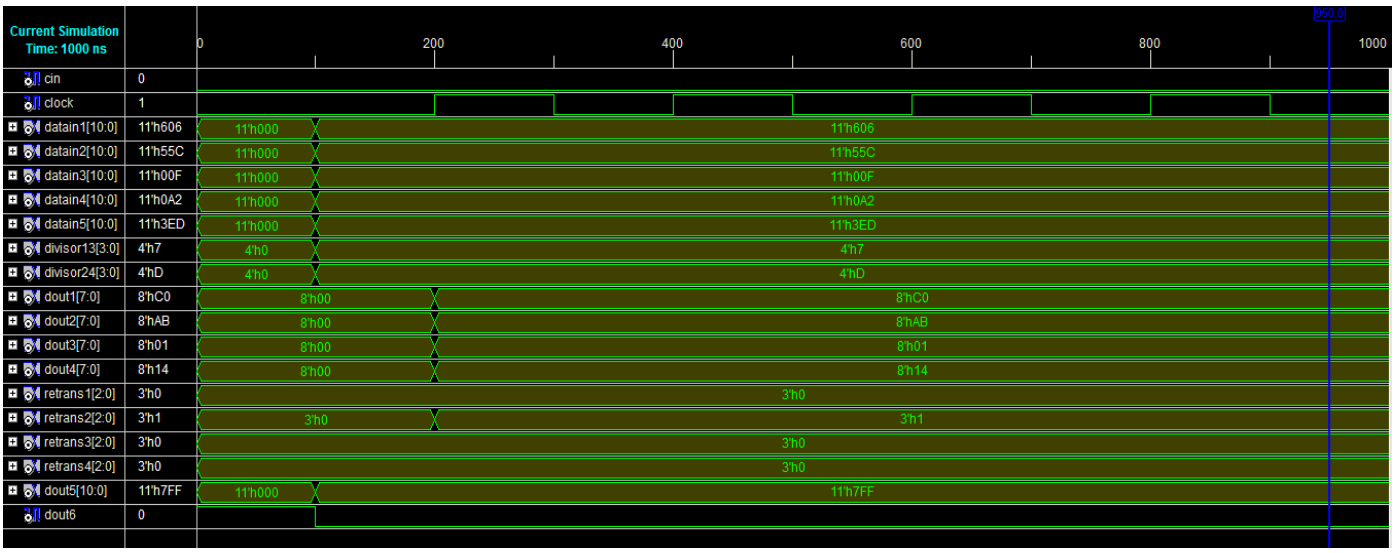


Figure 14. Time simulation of the corrupted received data at second byte (retrains2 is1and not 0)

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

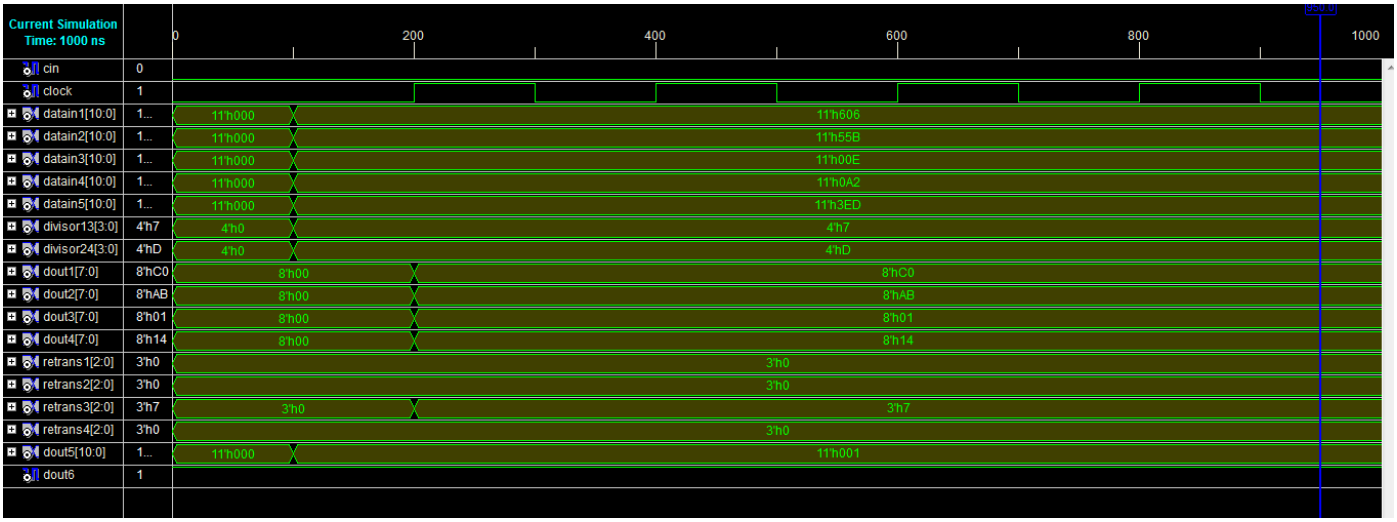


Figure 15. Time simulation of the corrupted received data at third byte (retrains3 is7and not 0)

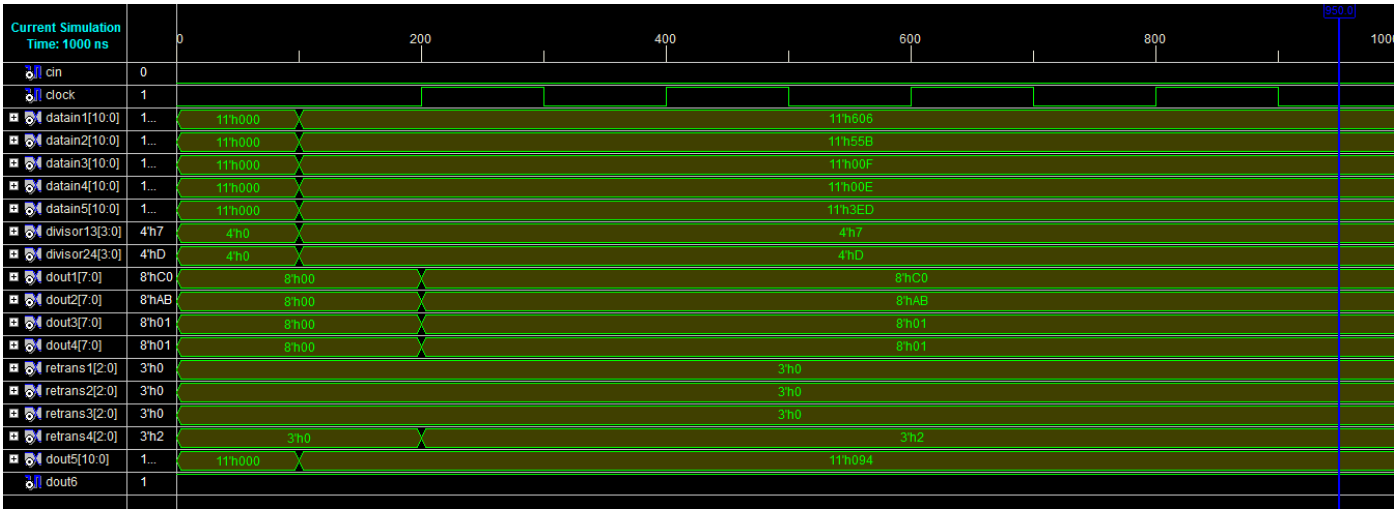


Figure 16. Time simulation of the corrupted received data at fourth byte (retrains4 is 2 and not 0)

Table 6: The receiver pins description

Receiver Pins	Description	Receiver Pins	Description
DATAIN1	1st Input block data stream with 11 bits	CLOCK	Clock signal
DATAIN2	2nd Input block data stream with 11 bits	dout1	Output data first byte
DATAIN3	3rd Input block data stream with 11 bits	dout2	Output data second byte
DATAIN4	4th Input block data stream with 11 bits	dout3	Output data third byte
DIVISOR13	CRC require input divisor for data packet 1 and 3.	dout4	Output data fourth byte
DIVISOR24	CRC require input divisor for data packet 2 and 4.	dout5 and dout6	Check sum statues if the output value 0 the data is correct else data has corrupted and need to retransmitted
RETRANS1	CRC output checking state for received data if the output is equal to value 0 then the first byte is correct else data has corrupted and need to retransmitted	RETRANS3	CRC output checking state for received data if the output is equal to value 0 then the third byte is correct else data has corrupted and need to retransmitted
RETRANS2	CRC output checking state for received data if the output is equal to value 0 then the second byte is correct else data has corrupted and need to retransmitted	RETRANS4	CRC output checking state for received data if the output is equal to value 0 then the fourth byte is correct else data has corrupted and need to retransmitted

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

Table 7: Design summary of the transmitter and receiver circuit

Logic Utilization	used		available		Utilization	
	Transmitter circuit	Receiver circuit	Transmitter circuit	Receiver circuit	Transmitter circuit	Receiver circuit
Number of slices LUTs	114	156	28800	28800	0%	0%
Number of fully used bit slices	0	0	114	156	0%	0%
Number of bonded IOBs	95	118	360	360	26%	32%

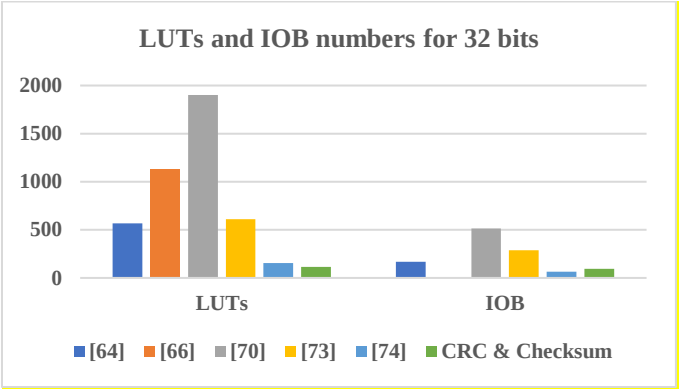
A comparison between selected references as mentioned with their listed number in *table 8* to demonstrate the effectiveness of proposed method when applying both techniques at the same time. In addition, *figures 17 and 18* have been summarized the comparison points for similar bit lengths as well as for different bit lengths. In *table 8* and based to [65] comparison notes, LUTs number has huge differences despite 24-bit length differences between both techniques. The results compared with [71] in spite LUTs and IOB was less than the proposed method but it had a 30-bit difference and close to it. Also, the proposed method was utilized about <1% for both LUTs and IOB numbers, while some references have huge ratio like [69]. The results recorded also huge differences as shown in *figure 17* with about 1008 and 1788 for references values in [66] and [70] respectively. These outcomes demonstrate that proposed technique for combination two detection methods is succeeded in minimizing LUTs and IOB numbers. *Table 8* also showed an impact differences with [7] of about 420, which is effect on memory and time parameters.

Table 8: Design summary Compared to related works for the same approaches based on selected references

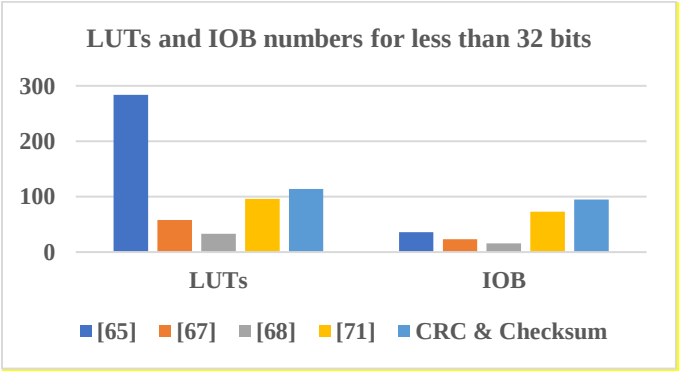
No.	Bits	LUTs	IOB
1- [64]	32 bits	568	168
2- [65]	8 bits	284	36
3- [66]	32 bits	1122	Not Specified
4- [67]	18 bits	58	23
5- [68]	4 bits	33	16
6- [69]	512*512 pixels	15.70%	15%
7- [69]	512*512 pixels	16.58%	21.11%
8- [70]	32 bits	1902	515
9- [71]	2 bits	96	73
10- [72]	64 bits	246	98
11- [73]	32 bits	610	288
12- [74]	32 bits	153	64
Proposed method	32 bits	114	95

The comparison was done with 11 references related to same network application as well as with different fields. It was also suggested for same and different bit-length. *Table 8* shows that LUTs and IOB numbers for the proposed technique obtained the best optimal value as also shown in *figure 16*. In [74], results were less than proposed technique in IOB numbers but with 39 LUTs differences for the proposed technique. Comparison was also done for 512 * 512 pixels (24 bits) as well, with impact ratio differences for 2D and 3D of about 14.7% and 15.58% respectively for LUTs numbers. While for the same reference's IOB numbers with huge difference of about 14% and 20.11 for 2D and 3D images despite 8 bit-length differences. LUT is essential blocks assist designer to generate logical circuit and make it easier in FPGA or other circuit design. While IOB is referred to the connection between external and internal FPGA circuits.

This work concluded as well that minimum number of LUTs impact on the complexity and increase circuit work performances as in FPGA. LUTs optimal number lead to fast operation and accurate use of circuit resources. While IOB numbers effect on timing, data integrity and reliability between internal and external connection units. For those reasons, well balanced and effective circuit design can be done by LUTs and IOB well optimized through hybrid techniques.



(a)



(b)

Figure 17. Comparison results (LUTs and IOB) between proposed method and selected references for (a) Similar bit lengths and (b) different bit length.

FOREX Publication
Open Access | Rapid and quality publishing
International Journal of Electrical and Electronics Research (IJEER)
Research Article | Volume 12, Issue 3 | Pages 813-826 | e-ISSN: 2347-470X

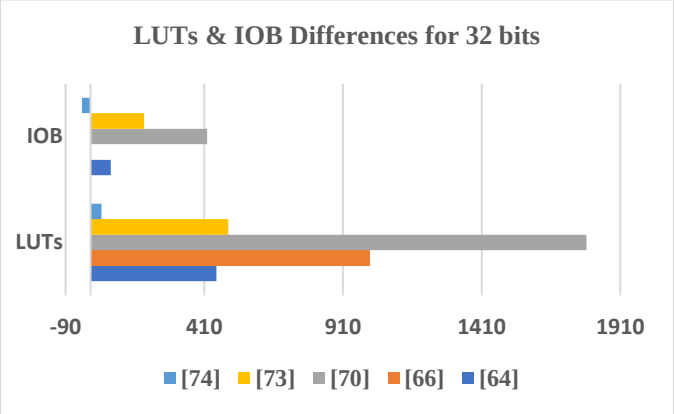


Figure 18. Comparison differences (LUTs and IOB) between proposed method and selected references for Similar bit lengths

4. CONCLUSION

To obtain more robust and reliable of detecting process in any electronic systems, combinations of two error detection methods was presented in this paper. CRC and Checksum provide more comprehensive detection mechanism which have the ability to catch different error types with a random bit place. Overall security also increased by implementing two steps of verifications which add additional layer of protection against data corruption or manipulation. The proposed model was applied for IP-v4 to provide security measurements for checking addresses that would prevent unauthorized access and identify theft. By verifying IP addresses for the sender, it ensures that the communication was connecting through a legitimate source. This proposed method also prevented malicious actors from occurring for another places and sites. The proposed method was also useful to protect sensitive data and maintain the overall security of networks or even for any electronic systems. In addition, programming error detection using codes such as Python language was useful for many reasons such as identifying the unexpected system behaviour for long period and before its utilization. Also, addressing issues that could occurred in the development process which solved by this step to save time and efforts. The proposed method was done using also VHDL plus Python to demonstrate its hardware and software high performances with 6 examples. All of steps are applied one by one with its results, outputs and calculations which shown in this paper. New combination process with new polynomial generation mechanism was presented that take just 72 bits of memory with milliseconds only or even less depending on hardware. Generating polynomial of CRC based on IP, specifically for version. 6 offers a significant security benefit with potential limitation as well. Creating CRC polynomial equation based on IPv6 provide high detection capability through segmenting the IP itself into different fields. Also, CRC Polynomial Can be designed to aim common errors pattern typically in IpV6 addresses. Based on IPv6 addressers, CRC Polynomial creation add complexity in protocol implementation which lead to additional resource requirements. In addition, if the selected polynomial does not adequately align with IpV6 may be provide a compression between Performances and utilization. It is also should be complemented with additional security metrics, like Supplementing addresses that could be replaced at any time.

REFERENCES

[1] Boussard, Vivien, Firouzeh Golaghazadeh, Stéphane Coulombe, François-Xavier Coudoux, and Patrick Corlay. "Robust H. 264 video decoding using CRC-based single error correction and non-desynchronizing bits validation." In 2020 IEEE International Conference on Image Processing (ICIP), pp. 1098-1102. IEEE, 2020.

[2] Bale, Ajay Sudhir, Karmanyaraj Singh Yadav, Mahboob Alam, Abhinav Shrivastava, Raj A. Varma, Rajdeep Singh Solanki, and Mamta B. Savadatti. "An Intelligent 64-bit parallel CRC for high-speed communication system applications." International Journal of Intelligent Systems and Applications in Engineering 11, no. 10s (2023): 543-551.

[3] Zhou, Zhixiong, and Rui Wang. "A Method of High-speed Parallel CRC Computation." In 2023 5th International Conference on Electronic Engineering and Informatics (EEI), pp. 298-303. IEEE, 2023.

[4] Linn, Kyi Sein, and Lin Min Ko. "Error Detection Based on Generator Polynomial." PhD diss., MERAL Portal.

[5] Rajeswari, R. Raja. "Analysis of Error Detection and Correction in Data Link Layer." International Journal of Innovative Research in Engineering & Management (IJIREM) (2021).

[6] Meylan, Alexandre, Mauro Cherubini, Bertil Chapuis, Mathias Humbert, Igor Bilogrevic, and Kévin Huguenin. "A study on the use of checksums for integrity verification of web downloads." ACM Transactions on Privacy and Security (TOPS) 24, no. 1 (2020): 1-36.

[7] Borchert, Christoph, Horst Schirmeier, and Olaf Spinczyk. "Compiler-Implemented Differential Checksums: Effective Detection and Correction of Transient and Permanent Memory Errors." In 2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 81-94. IEEE, 2023.

[8] Dodmane, Radhakrishna, Ganesh Aithal, and Surendra Shetty. "Construction of vector space and its application to facilitate bitwise XOR–Free operation to minimize the time complexity." Journal of King Saud University-Computer and Information Sciences 34, no. 10 (2022): 9836-9843.

[9] Yeleswarapu, Ravikiran. "Addressing multi-bit errors in DRAM/memory subsystem." PhD diss., Iowa State University, 2020.

[10] Boussard, Vivien, Stéphane Coulombe, François-Xavier Coudoux, and Patrick Corlay. "Table-free multiple bit-error correction using the CRC syndrome." IEEE Access 8 (2020): 102357-102372.

[11] Longari, Stefano, Matteo Penco, Michele Carminati, and Stefano Zanero. "Copycan: An error-handling protocol based intrusion detection system for controller area network." In Proceedings of the ACM Workshop on Cyber-Physical Systems Security & Privacy, pp. 39-50. 2019.

[12] An, Wei, Muriel Médard, and Ken R. Duffy. "CRC codes as error correction codes." In ICC 2021-IEEE International Conference on Communications, pp. 1-6. IEEE, 2021.

[13] Boussard, Vivien, Stéphane Coulombe, François-Xavier Coudoux, and Patrick Corlay. "CRC-based correction of multiple errors using an optimized lookup table." IEEE Access 10 (2022): 23931-23947.

[14] Egilmez, Zeynep B. Kaykac, Luping Xiang, Robert G. Maunder, and Lajos Hanzo. "The development, operation and performance of the 5G polar codes." IEEE Communications Surveys & Tutorials 22, no. 1 (2019): 96-122.

[15] Gini, Maria. "Automatic error detection and recovery." In Robot technology and applications, pp. 445-484. CRC Press, 2020.

[16] Zhang, Yi, Xiangyang Luo, Xiaodong Zhu, Zhenyu Li, and Adrian G. Bors. "Enhancing reliability and efficiency for real-time robust adaptive

steganography using cyclic redundancy check codes." Journal of Real-Time Image Processing 17 (2020): 115-123.

[17] Dilli, Ravilla. "Channel code rate matching design in cyclic redundancy check-aided polar coding for 5g nr uplink communication." Telecommunications and Radio Engineering 80, no. 4 (2021).

[18] Canto, Alvaro Cintas, Mehran Mozaffari-Kermani, and Reza Azarderakhsh. "Reliable CRC-based error detection constructions for finite field multipliers with applications in cryptography." IEEE Transactions on Very Large-Scale Integration (VLSI) Systems 29, no. 1 (2020): 232-236.

[19] Alhussen, Ahmed, and Engin Arslan. "Avoiding data loss and corruption for file transfers with Fast Integrity Verification." Journal of Parallel and Distributed Computing 152 (2021): 33-44.

[20] Kara, Mostefa, Abdelkader Laouid, Mohammad Hammoudeh, and Ahcène Bounceur. "One Digit Checksum for Data Integrity Verification of Cloud-executed Homomorphic Encryption Operations." Cryptology ePrint Archive (2023).

[21] Labell, Lauren, Jared Chandler, and Kathleen Fisher. "Automatic Discovery and Synthesis of Checksum Algorithms from Binary Data Samples." In Proceedings of the 15th Workshop on Programming Languages and Analysis for Security, pp. 25-34. 2020.

[22] Filippas, Dionysios, Nikolaos Margomenos, Nikolaos Mitianoudis, Chrysostomos Nicopoulos, and Giorgos Dimitrakopoulos. "Low-cost online convolution checksum checker." IEEE Transactions on Very Large-Scale Integration (VLSI) Systems 30, no. 2 (2021): 201-212.

[23] Charyyev, Batyr. "Protecting File Transfers Against Silent Data Corruption with Robust End-to-End Integrity Verification." PhD diss., University of Nevada, Reno, 2019.

[24] Charyyev, Batyr, Ahmed Alhussen, Hemanta Sapkota, Eric Pouyoul, Mehmet Hadi Gunes, and Engin Arslan. "Towards securing data transfers against silent data corruption." In 2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), pp. 262-271. IEEE, 9.

[25] Alhussen, Ahmed, and Engin Arslan. "Avoiding data loss and corruption for file transfers with Fast Integrity Verification." Journal of Parallel and Distributed Computing 152 (2021): 33-44.

[26] Javaheripi, Mojan, Jung-Woo Chang, and Farinaz Koushanfar. "AccHashtag: Accelerated Hashing for Detecting Fault-Injection Attacks on Embedded Neural Networks." ACM Journal on Emerging Technologies in Computing Systems 19, no. 1 (2022): 1-20.

[27] Siebert, Christian. "Highly Scalable Parallel Checksums." In 2021 IEEE 27th International Conference on Parallel and Distributed Systems (ICPADS), pp. 812-818. IEEE, 2021.

[28] Abdulnabi, Mohamed Abdulnabi. "High Speed Low Power Cyclic Redundancy Check-32 using FPGA." Journal of Engineering and Computer Science (JECS) 19, no. 2 (2019): 50-56.

[29] Subhasri, G., and N. Radha. "VLSI design of parity check code with hamming code for error detection and correction." In 2019 International Conference on Intelligent Computing and Control Systems (ICCS), pp. 15-20. IEEE, 2019.

[30] Hillier, Caleb, and Vipin Balyan. "Error detection and correction on-board nanosatellites using hamming codes." Journal of Electrical and Computer Engineering 2019 (2019).

[31] Song, Kai, Zhenxing Wang, Jinliang Zhu, and Liping Yan. "Research and application of error correction theory for ternary optical computer based on Hamming code." Optik 267 (2022): 169647.

[32] Akpu, Ni, Om Akpu, and Fc Anyadiegwu. "Comparative review of automatic repeat request and forward error correction method of error control coding in digital communication." Nigerian Journal of Scientific Research 18, no. 3 (2019): 209-213.

[33] Kanona, Rusul M., Zainab N. Al-Qudsy, and Shaymaa Azzam Alyawer. "Forward error correction in 5G heterogeneous network." Periodicals of Engineering and Natural Sciences 10, no. 2 (2022): 90-100.

[34] Turdiev, Odilzhan A., Vladimir A. Smagin, and Vladimir N. Kustov. "Investigation of the computational complexity of the formation of checksums for the Cyclic Redundancy Code algorithm depending on the width of the generating polynomial." Models and Methods for Researching Information Systems in Transport 2020 (MMRIST 2020) 1 (2020): 129-135.

[35] Bhukra, Sonia, and Ruchi Sharma. "The performance optimization of CRC network scanners based on the calculation of CRC checksums." In 2022 International Interdisciplinary Humanitarian Conference for Sustainability (IIHC), pp. 338-342. IEEE, 2022.

[36] Zakariyya, Rabiu Sale, Khalid Hossen Jewel, Akinwale O. Fadamiro, Oluwole John Famoriji, and Fujiang Lin. "An efficient polar coding scheme for uplink data transmission in narrowband internet of things systems." IEEE Access 8 (2020): 191472-191481.

[37] Chen, Mengfu, Chenguang Guo, Lei Chen, Wenjie Li, Fan Zhang, Xiaoxiang Hu, and Jiancheng Xu. "Research on EDAC Schemes for Memory in Space Applications." Electronics 10, no. 5 (2021): 533.

[38] Padmanabhan, Ramakrishna, John P. Rula, Philipp Richter, Stephen D. Strowes, and Alberto Dainotti. "DynamIPs: Analyzing address assignment practices in IPv4 and IPv6." In Proceedings of the 16th International Conference on emerging Networking EXperiments and Technologies, pp. 55-70. 2020.

[39] Moore, Samuel J., Chris D. Nugent, Shuai Zhang, and Ian Cleland. "IoT reliability: a review leading to 5 key research directions." CCF Transactions on Pervasive Computing and Interaction 2 (2020): 147-163.

[40] Hughes, Lawrence E. "The Depletion of the IPv4 Address Space." In Third Generation, Internet Revealed: Reinventing Computer Networks with IPv6, pp. 119-146. Berkeley, CA: Apress, 2022.

[41] Jadhav, Shraddha S., and Beena R. Ballal. "Review of IPv4 and IPv6 and various implementation methods of IPv6." Arabia 4, no. 783 (2022): 66.

[42] Huston, Geoff. "IP Addresses through 2022: This article is reproduced from “The ISP Column-A monthly column on things Internet” of Geoff Huston's Blog." International Journal of Advanced Network, Monitoring and Controls 8, no. 2: 27-45.

[43] Ashraf, Zeeshan, Adnan Sohail, Sohaib A. Latif, Abdul Hameed Pitafi, and Muhammad Yousaf Malik. "Challenges and Mitigation Strategies for Transition from IPv4 Network to Virtualized Next-Generation IPv6 Network." Int. Arab J. Inf. Technol. 20, no. 1 (2023): 78-91.

[44] Knossen, Silke, Joseph Hill, and Paola Grosso. "Hop recording and forwarding state logging: Two implementations for path tracking in p4." In 2019 IEEE/ACM Innovating the Network for Data-Intensive Science (INDIS), pp. 36-47. IEEE, 2019.

[45] Turdiev, Odilzhan A., Vladimir A. Smagin, and Vladimir N. Kustov. "Investigation of the computational complexity of the formation of checksums for the Cyclic Redundancy Code algorithm depending on the width of the generating polynomial." Models and Methods for Researching Information Systems in Transport 2020 (MMRIST 2020) 1 (2020): 129-135.

[46] Arifin, Md Mashrur, Md Tariq Hasan, Md Tarikul Islam, Md Almahmud Hasan, and Himadri Shekhar Mondal. "Design and implementation of high performance parallel crc architecture for advanced data communication." In 2019 4th International Conference on Electrical Information and Communication Technology (EICT), pp. 1-5. IEEE, 2019.

[47] Cintas-Canto, Alvaro, Mehran Mozaffari-Kermani, Reza Azarderakhsh, and Kris Gaj. "CRC-oriented error detection architectures of post-quantum cryptography niederreiter key generator on FPGA." In 2022 IEEE Nordic Circuits and Systems Conference (NorCAS), pp. 1-7. IEEE, 2022.

[48] Zhang, Quanwei, Dazhong Li, Yue Fei, Jiakang Zhang, Yu Chen, and Fei Tong. "RDCPF: a redundancy-based duty-cycling pipelined-forwarding MAC for linear sensor networks." Sensors 20, no. 19 (2020): 5608.

[49] Yagoub, Mudathir FS, Joel JPC Rodrigues, Othman O. Khalifa, Abuagla B. Mohammed, and Valery Korotaev. "Service redundancy and cluster-based routing protocols for wireless sensor and mobile ad hoc networks: A survey." International Journal of Communication Systems 33, no. 16 (2020): e4471.

[50] Kadel, Rajan, Krishna Paudel, Deepani B. Guruge, and Sharly J. Halder. "Opportunities and challenges for error control schemes for wireless sensor networks: A review." Electronics 9, no. 3 (2020): 504.

[51] Gupta, Megha. "Cyclic redundancy check based data authentication in opportunistic networks." In 2nd International Conference on Wireless Intelligent and Distributed Environment for Communication: WIDECOM 2019, pp. 17-26. Springer International Publishing, 2019.

[52] Dilli, Ravilla. "Channel code rate matching design in Cyclic Redundancy check-aided polar coding for 5g nr uplink communication." Telecommunications and Radio Engineering 80, no. 4 (2021).

[53] Bian, Hongxiu, Rongke Liu, Aryan Kaushik, Yingmeng Hu, and John S. Thompson. "Design of segmented CRC-aided spinal codes for IoT applications." IET Communications 14, no. 20 (2020): 3541-3548.

[54] Boussard, Vivien, Stéphane Coulombe, François-Xavier Coudoux, and Patrick Corlay. "CRC-based correction of multiple errors using an optimized lookup table." IEEE Access 10 (2022): 23931-23947.

[55] Bhuvana, B. P., and VS Kanchana Bhaaskaran. "Design and analysis of IPAL for ultra-low power CRC architecture for applications in IoT based systems." AEU-International Journal of Electronics and Communications 108 (2019): 127-140.

[56] Kishore, P., Bolli Abhinay Pal, Lolakapuri Nanda Kishore, and Chintamreddy Venkata Revathi. "Implementation of Table-Based Cyclic Redundancy Check (CRC-32) for Gigabit Ethernet Applications." In 2023 4th International Conference for Emerging Technology (INCET), pp. 1-4. IEEE, 2023.

[57] Borchert, Christoph, Horst Schirmeier, and Olaf Spinczyk. "Compiler-Implemented Differential Checksums: Effective Detection and Correction of Transient and Permanent Memory Errors." In 2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 81-94. IEEE, 2023.

[58] A. Tanenbaum and D. Wetherall, "Computer Networks, 5th ed., Pearson Education: United States of America, 2011., chapter 3, discusses error detection techniques including CRC and checksums", P.P 210-220.

[59] M. K. McKusick et al. "The Design and Implementation of the FreeBSD Operating System", 2nd ed., Addison-Wesley, United States of America, 2015. Chapter 8 covers storage subsystems and fault tolerance techniques, P.P 305-315.

[60] I. F. Akyildiz et al., "Wireless sensor networks: a survey, Computer Networks", vol. 38, No. 4, pp. 393-422, 2002. Science Direct.

[61] K. C. Barr and T. S. Barnes, "Embedded Systems: World Class Designs, Newnes: Oxford and Boston", 2008. Chapter 2 discusses embedded system architectures and reliability, P.P 49-60.

[62] M. Foster et al., "Embedded systems: a contemporary design tool", IEEE Potentials, vol. 21, No. 2, pp. 29-33, 2002.

[63] A. Berman and M. L. Dolan, "Digital Preservation for Libraries, Archives, and Museums", Rowman & Littlefield: UK, Lanham, Maryland, 2018. Chapter 5 discusses digital preservation strategies including error detection and correction techniques.

[64] A. Abdala, M., A. R. Kadhim, A.-K., & Sh.Rijab, "A FPGA Implementation of a Modified Cryptographic Method Based on Panama Module". I-Manager's Journal on Software Engineering,2008, 3(2), 49–57.

[65] Arasavalli, N. "Low Power Optimized Fault Tolerant Systolic Array Process Systolic Array Process Elements for System and Network-On-Chip Devices". ICRACE-23,2023, June, 1–6.

[66] I Indu, T. S. M. "Cyclic Redundancy Check Generation Using Multiple Lookup Table Algorithms." International Journal of Modern Engineering Research, 2(4),2012, 2445–2451.

[67] Implementation, F., & USB, O. F. "FPGA implementation of USB 2.0 receiver protocol". International Conference on Advanced Computing, Communication and Networks'11,2011, 758–762.

[68] Kale, S. D., & Zade, P. G. N. "Design of Baugh-wooley Multiplier using Verilog HDL". IOSR Journal of Engineering (IOSRJEN), 2015, 05(10), 25–29.

[69] Nagar, M., "Proposed architecture of LOW memory LUT for LUT base parallel CRC generation". International Journal of Advance Engineering and Research Development (IJAERD), 2020, 1(3), 1–7.

[70] Nazar, G. L. "Fine-grained error detection techniques for fast repair of FPGAs" (2013). (Issue July). http://hdl.handle.net/10183/77746

[71] Piltan, F., Gavahian, A., & Marhaban, M. H. "Novel Sliding Mode Controller for Robot Manipulator using FPGA". Journal of Advanced Science and Engineering Research,2011, 1, 1–22.

[72] Pisare, T. S., & Charniya, N. "A Design and implementation of hybrid GALOIS filed encoder & decoder". Journal of Computing Technologies (JCT), 2022, 3, 1–6.

[73] Saini, P. "FPGA Implementation of 2D and 3D Image Enhancement Chip in HDL Environment. International". Journal of Computer Applications,2013, 62(21), 24–31.

[74] Samaiya, S., & Jain, A A implementation of GA based FPGA ALU unit. International Journal of Engineering Sciences & Research Technology,2018, 7(7), 247–253.