

Article

Not peer-reviewed version

---

# Neural Network Technology for Cryptographic Protection of Data Transmission at UAV

---

[Ivan Tsmots](#) , [Vasyl Teslyuk](#) \* , [Andrzej Łukaszewicz](#) \* , Yuriy Lukashchuk , [Iryna Kazymyra](#) , [Andriy Holovatyy](#) , [Yurii Opotyak](#)

Posted Date: 14 April 2023

doi: 10.20944/preprints202304.0252.v2

Keywords: neural network technology; cryptographic protection; UAV; UAS, onboard system; encryption; decryption; tabular-algorithmic method; scalar product; real time



Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Article

# Neural Network Technology for Cryptographic Protection of Data Transmission at UAV

Ivan Tsmots <sup>1</sup>, Vasyl Teslyuk <sup>1,\*</sup>, Andrzej Łukaszewicz <sup>2,\*</sup>, Yurii Lukashchuk <sup>1</sup>, Iryna Kazymyra <sup>1</sup>, Andriy Holovatyy <sup>3</sup>, Yurii Opotyak <sup>1</sup>

<sup>1</sup> Department of Automated Control Systems, Lviv Polytechnic National University, 79013 Lviv, Ukraine; ivan.h.tsmots@lpnu.ua (I.T.), vasyi.m.teslyuk@lpnu.ua (V.T.), urijlukas@gmail.com (Y.L.), iryna.y.kazymyra@lpnu.ua (I.K.), yurii.v.opotyak@lpnu.ua (Y.O.)

<sup>2</sup> Department of Machine Design and Exploitation, Faculty of Mechanical Engineering, Bialystok University of Technology, Bialystok, Poland; a.lukaszewicz@pb.edu.pl (A.L.)

<sup>3</sup> Department of Computer-Aided Design Systems, Lviv Polytechnic National University, 79013 Lviv, Ukraine; andrii.i.holovatyy@lpnu.ua (A.H.)

\* Correspondence: vasyi.m.teslyuk@lpnu.ua (V.T.), a.lukaszewicz@pb.edu.pl (A.L.)

**Abstract:** The neural network technology for real-time cryptographic data protection with symmetric keys (masking codes, neural network architecture and weights matrix) for unmanned aerial vehicles (UAV) onboard communication systems has been developed. It provides hardware and software implementation with high technical and operational characteristics. The development of neural network technology for real-time cryptographic data protection was performed using an integrated approach based on theoretical foundations of neural network cryptographic data protection, new algorithms and structures of neural network data encryption and decryption, modern hardware components with programmable structure, and computer-automated design (CAD) of hardware and software. The development and implementation of the on-board neural network system for cryptographic data protection in real-time are based on the following principles: variable composition of equipment; modularity; conveyorization and spatial parallelism; hardware and software modification and making them suitable for data encryption and decryption. The tabular-algorithmic method for calculation of the scalar product has been improved that makes it possible to perform fast calculation of the scalar product of fixed-point and floating-point input data by bringing them to the largest common order of weights and forming tables of macro-partial products. Components of neural network cryptographic data encryption and decryption have been developed on the processor core supplemented by specialized scalar product calculation modules. The specialized hardware for neural network cryptographic data encryption was developed using VHDL for equipment programming in the Quartus II development environment ver. 13.1 and the appropriate libraries, and implemented on the basis of the FPGA EP3C16F484C6 Cyclone III family.

**Keywords:** neural network technology; cryptographic protection; UAV; UAS; onboard system; encryption; decryption; tabular-algorithmic method; scalar product; real time

## 1. Introduction

A key problem is to guarantee cryptographic security of data transmission in the management of UAV [1, 2], intelligent robots [3], microsatellites [4] and various mobile transport systems [5]. Due to the security vulnerabilities of UAVs, and illegal and malicious attacks against UAVs, especially against communication data and UAV control, solutions to prevent such attacks are needed, and one of them is to encrypt UAV's communication data [6, 7, 8]. Unmanned Aerial Vehicles (UAVs) must be energy-efficient, especially in data processing, because of limited battery capacity [9]. Solving this problem requires the development of neural network (NN) technology [10, 11, 12] for cryptographic data protection, which is focused on use in UAV onboard communication systems. When developing onboard cryptographic data protection systems, it is necessary to provide real-time mode, increase cryptographic resistance, and noise immunity and reduce power consumption, weight, size and cost

[13 - 23]. The usage of an auto-associative NN of direct propagation, which is trained on the basis of the principal components analysis, helps to conform such requirements. A specific feature of such neural networks is the ability of weight pre-calculation and to apply the tabular-algorithmic method for the implementation of neuro-like elements using the basis of elementary arithmetic operations. For NN cryptographic encryption and decryption of data, it is proposed to use symmetric keys, which include masking codes, NN architecture and a matrix of weights [24, 25].

Through the extensive use of a modern component base and the development of new VLSI methods, algorithms and structures, high technical and operational rates of on-board cryptographic data protection systems are achieved. Onboard systems for NN cryptographic protection of data must have variable hardware for rapid changes in NN architecture. The use of modern element base (microcontrollers, programmable logic integrated circuits FPGA) in the development of onboard and embedded systems makes it possible to reduce their weight, size and power consumption [26, 27], and in the development of onboard systems of NN cryptographic data protection provides a quick change of encryption and decryption keys.

NN cryptographic encryption and decryption of data in real-time is achieved through the application of parallel encryption and decryption of data, hardware implementation of neuro-like elements based on a multi-operand approach and macro-partial products tables.

Therefore, an urgent problem is to develop NN technology for cryptographic data protection, focused on implementing in on-board systems with high technical and operational characteristics. The objective of the work is to develop the onboard NN technology for real-time cryptographic data protection. In order to achieve this goal, the following tasks have to be solved:

- development of NN technology for cryptographic data protection;
- development of the structure of the system of NN cryptographic protection and real-time data transmission;
- development of components of onboard systems of NN cryptographic encryption-decryption of data;
- implementation of the specialized hardware components of NN cryptographic data encryption on FPGA.

This article is structured as follows. In the introduction, we have considered the problem relevancy and the main objectives of this research. Section 2 contains a brief review of the related works (the research context). The structure of NN technology for cryptographic data protection is described in Section 3 and the main stages of NN data encryption/decryption are considered here as well. Section 4 presents the structure of the system for NN cryptographic data protection and transmission (stationery and UAV onboard parts) developed using an integrated approach. The components of the onboard system for NN cryptographic data encryption and decryption are proposed in Section 5, the diagrams for the specialized hardware are given.

## 2. Related Works

The study of the main trends in the area of UAV onboard systems development for real-time cryptographic data protection shows that NN methods are increasingly used for performing data encryption and decryption in such systems [28-32]. These publications show that the implementation of NN methods of cryptographic data protection is performed generally by software. The critical drawback of software implementation of NN cryptographic data protection is the difficulty of providing real-time mode and the constraints imposed on on-board systems in terms of weight, size, power consumption and cost.

The possibilities of adapting the auto-associative NN with non-iterative learning for data protection tasks are considered in [28-32]. The peculiarity of the functioning of such a NN is the preliminary calculation of weights as a result of its training based upon the principal component analysis (PCA). In this case a system of eigenvectors is used, that correspond to the eigenvalues of the covariance matrix of input data [33]. To encrypt and decrypt data the auto-associative NN with pre-calculated weights is applied. In [34] it was shown that the masking codes, the architecture of the

NN and the matrix of weights are the basis to cryptographic encryption and decryption of data in neural networks.

Publications [35-37] are devoted to the hardware implementation of neural networks showing that they are based on neural elements. The feature of such neural elements is that the number of inputs and their bit length are determined by the NN architecture, which is one of the characteristics of the data encryption key. The main operation of the neuroelement is the calculation of the scalar product using pre-calculated weights.

In [37-39] the methods for calculating the scalar product using the basis of elementary arithmetic operations - addition and shift, are considered. The peculiarity of these methods is the formation of macro-partial products, their shift, and addition to the previously accumulated amount. Hardware implementation of such methods requires significant equipment costs. The implementation of a tabular-algorithmic method for calculating the scalar product, which is reduced to the operations of reading macro-partial products, addition and shift, requires fewer equipment costs and less computation time. The disadvantage of this method is that it is limited to fixed-point data format (for input data and weights).

Analyzing the works [31, 41] it can be noted that NN tools for cryptographic symmetric encryption and decryption of data [42] are implemented on the basis of microprocessors supplemented by hardware that implements time-consuming computational operations using FPGA [43]. High speed of NN tools for cryptographic encryption and decryption of data is achieved through parallelization, pipeline computing processes and hardware implementation of neural elements. The disadvantage of the existing NN tools for cryptographic data protection is the difficulty of changing the encryption and decryption key rapidly.

### **3. The development of NN technology for cryptographic data protection**

#### *3.1. Structure of NN technology of cryptographic data protection*

The development of NN technology for cryptographic protection of data transmission is focused on hardware and software implementation with high technical and operational characteristics. It is proposed to carry out such development on the basis of an integrated approach that includes:

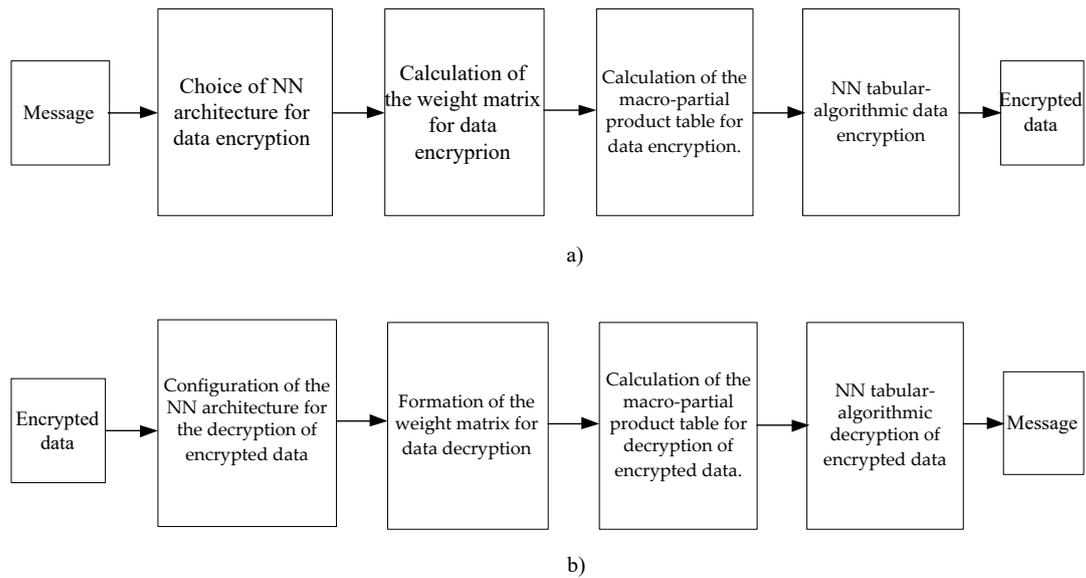
- research and development of theoretical foundations of neuro-like cryptographic data protection;

research and development of new algorithms and structures of neuro-like encryption and decryption of data focused on modern element base;

modern element base with the ability to program the structure;

means for automated design of software and hardware.

Figure 1 shows the developed structure of NN technology for cryptographic data protection, which is focused on hardware and software implementation and provides encryption with symmetric keys. When implementing the symmetric cryptosystem, the encryption key and the decryption key are the same or the decryption key is easily calculated from the encryption key. The tabular-algorithmic method makes it possible to implement high technical and operational characteristics of data encryption-decryption tools.



**Figure 1.** Structure of NN technology for cryptographic data protection: a) the process of data encryption; b) the process of decrypting data.

### 3.2. Main stages of NN encryption

The encryption uses a key consisting of  $N$  neurons in the NN, a weight matrix and masking operations. The main stages of message encryption are considered below.

*Choice of NN architecture.* The number of neuro elements  $N$ , the number of inputs  $k$  and the bit inputs  $m$  determine the architecture of the NN. The number of neural elements is defined according to the following formula:

$$N = \frac{n}{m}, \quad (1)$$

where  $n$  is the bit length of the message, and  $m$  – the bit length of the inputs.

The incoming messages which are encrypted can have different bit length ( $n$ ) and different inputs number ( $k$ ), which is equal to the number of neuroelements  $N$ . The architecture of the NN depends on the value of the bit length of the message  $n$  and the number of inputs  $k$ . Such configuration of the NN architecture is available to encrypt the  $n = 16$  bit message:  $m = 2, k = 8, N = 8$ ;  $m = 4, k = 4, N = 4$ ;  $m = 8, k = 2, N = 2$ , in case of  $n = 24$  they are:  $m = 2, k = 12, N = 12$ ;  $m = 3, k = 8, N = 8$ ;  $m = 4, k = 6, N = 6$ ;  $m = 6, k = 4, N = 4$ ;  $m = 8, k = 3, N = 3$ ;  $m = 12, k = 2, N = 2$ .

*Calculation of the weight matrix.* For data encryption-decryption we will use an auto-associative NN, which learns non-iteratively using the principal components analysis (PCA), which performs a linear transformation following the formula:

$$\bar{y} = W \cdot \bar{x}. \quad (2)$$

According to formula (2), the matrix  $W \in R^{n \times n}$  is used to convert the input vector  $\bar{x} \in R^n$  into the output vector  $\bar{y} \in R^n$ . The conversion is as follows. System of linearly independent vectors selects an orthonormal system of eigenvectors corresponding to the eigenvalues of the covariance matrix of the input data.

The input data is a set of  $N$  vectors  $\bar{x}_j, j = 1, \dots, N$ , with dimension  $n$ ,  $\bar{x}_j = (x_{j1}, x_{j2}, \dots, x_{jn})$ :

$$X = (\bar{x}_1, \bar{x}_2, \dots, \bar{x}_N)^t. \quad (3)$$

For  $N$  vectors the autocovariance matrix  $\bar{x}_j$  is:

$$R = X^t \cdot X, \quad (4)$$

where each of the elements is expressed by:

$$r_{jl} = \sum_{i=1}^N \bar{x}_{ji} \bar{x}_{il} = \sum_{i=1}^N (\bar{x}_{ji} - \mu_j)(\bar{x}_{il} - \mu_l), \quad (5)$$

where  $j, l = 1, 2, \dots, n$ , and  $\mu_j, \mu_l$  – mathematical expectations of vectors  $\bar{x}_j, \bar{x}_l$ .

The eigenvalues of  $R$  symmetric non-negative matrix are real and positive numbers. They are arranged in descending order  $\lambda_1 > \lambda_2 > \dots > \lambda_n$ . Similarly, the eigenvectors corresponding to  $\lambda_i$  are placed. Therefore, a linear transformation (2) is defined by the matrix  $W$ . Here  $\bar{y} = (y_1, y_2, \dots, y_n)$  is a vector of the PCA principal components corresponding to the input data vector  $\bar{x}$ . The number of principal components vectors  $N$  conforms with the number of input data vectors  $\bar{x}$  [29]. The matrix of weights used to encrypt the data is as follows:

$$\begin{bmatrix} W_{11} & W_{12} & \dots & W_{1k} \\ W_{21} & W_{22} & \dots & W_{2k} \\ \vdots & \vdots & \dots & \vdots \\ W_{N1} & W_{N2} & \dots & W_{Nk} \end{bmatrix}. \quad (6)$$

The basic operation of the NN used to encrypt data is the operation of calculating the scalar product. This operation should be implemented using the tabular-algorithmic method because the matrix of weights  $W_{js}$ , where  $j = 1, \dots, N, s = 1, \dots, k$ , is pre-calculated.

*Calculation of the table of macro-partial products for data encryption.* The specificity of the scalar product calculation operation used in data encryption is that the weights are pre-calculated (constants) and set in floating point format, and the input data  $X_j$  is in fixed point format with its fixing before the high digit of a number. The scalar product is calculated by means of the tabular-algorithmic method according to the formula:

$$Z = \sum_{j=1}^N W_j X_j = \sum_{i=1}^n 2^{-i} \sum_{j=1}^N W_j X_{ji} = \sum_{i=1}^n 2^{-i} \sum_{j=1}^N P_{ji} = \sum_{i=1}^n 2^{-i} P_{Mi}, \quad (7)$$

where  $N$  – number of products,  $X_j$  – input data,  $W_j$  –  $j$ -th weight coefficient,  $n$  – bit length of the input data,  $P_{ij}$  – partial product,  $P_{Mi}$  – macro-partial product, formed by adding  $N$  partial products  $P_{ij}$ , as follows:  $P_{Mi} = \sum_{j=1}^N P_{ji}$ .

Formation of the tables of macro-partial products for floating-point weights  $W_j = w_j 2^{m_{W_j}}$  (where  $w_j$  – mantissa of  $W_j$  weight coefficient,  $m_{W_j}$  – order of  $W_j$  weight coefficient) foresees the following operations to be performed:

- defining the largest common order of weights  $m_{W_{\max}}$ ;
- calculation of the order difference for each  $W_j$  weight coefficient:
 
$$\Delta m_{W_j} = m_{W_{\max}} - m_{W_j};$$
- shift the mantissa  $w_j$  to the right by a difference of orders  $\Delta m_{W_j}$ ;
- calculation of  $P_{Mi}$  macro-partial product for the case when  $x_{1i} = x_{2i} = x_{3i} = \dots = x_{Ni} = 1$ ;
- determining the number of overflow bits  $q$  in the  $P_{Mi}$  macro-partial product for the case when  $x_{1i} = x_{2i} = x_{3i} = \dots = x_{Ni} = 1$ ;
- obtaining scalable mantissas  $w_j^h$  by shifting them to the right by the number of overflow bits;
- adding to the largest common order of weight  $m_{W_{\max}}$  the number of overflow bits  $q$ , as per formula  $m_j = m_{W_{\max}} + q$ .

The table of macro-partial products is calculated by the formula:

$$P_{Mi} = \begin{cases} 0, & \text{if } x_{1i} = x_{2i} = x_{3i} = \dots = x_{Ni} = 0 \\ w_1^h, & \text{if } x_{1i} = 1, x_{2i} = x_{3i} = \dots = x_{Ni} = 0 \\ w_2^h, & \text{if } x_{1i} = 0, x_{2i} = 1, x_{3i} = \dots = x_{Ni} = 0 \\ w_1^h + w_2^h, & \text{if } x_{1i} = 1, x_{2i} = 1, x_{3i} = \dots = x_{Ni} = 0 \\ \vdots & \\ w_2^h + \dots + w_N^h, & \text{if } x_{1i} = 0, x_{2i} = x_{3i} = \dots = x_{Ni} = 1 \\ w_1^h + w_2^h + \dots + w_N^h, & \text{if } x_{1i} = x_{2i} = x_{3i} = \dots = x_{Ni} = 1 \end{cases}, \quad (8)$$

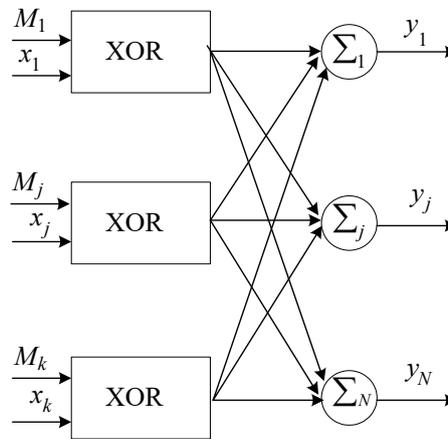
where  $x_{1i}, x_{2i}, x_{3i}, \dots, x_{Ni}$  – address inputs of the table,  $w_j^h$  – mantissa of  $W_j$  weight coefficient brought to the greatest common order.

The possible combinations number of  $P_{Mi}$  macro-partial products and the table size is as follows:

$$Q = 2^N. \quad (9)$$

Dividing all  $N$  products by parts  $N1$  and  $N2$  we can reduce the table size. For each of these parts separate tables of macro-partial products  $P_{N1Mi}$  and  $P_{N2Mi}$  are formed and stored in separate memory blocks or a single memory block. When using two memory blocks, parts of the macro-partial products  $P_{N1Mi}$  and  $P_{N2Mi}$  are read in one clock cycle, and in one memory block - in two clock cycles. The sum of two macro-partial products  $P_{N1Mi}$  and  $P_{N2Mi}$  gives us the macro-partial product  $P_{Mi}$ .

*NN tabular-algorithmic data encryption.* During the training of the NN the matrix of weights  $W$  is determined. Figure 2 shows the structure of auto-associative NN used for data encryption. Here  $M_j$  is the mask for the  $j$ -th input,  $x_j$  is the  $j$ -th input data, XOR is the masking operation using the exclusive OR elements.



**Figure 2.** The structure of the data encryption NN.

To perform the NN data encryption we multiply the  $W$  matrix by the input data vector  $\bar{x}$  according to the formula:

$$y_j = \begin{bmatrix} W_{11} & W_{12} & \dots & W_{1k} \\ W_{21} & W_{22} & \dots & W_{2k} \\ \vdots & \vdots & \dots & \vdots \\ W_{N1} & W_{N2} & \dots & W_{Nk} \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \end{bmatrix}. \quad (10)$$

The multiplication of the matrix of weights  $W$  by the vector of input data  $\bar{x}$  is reduced to performing  $N$  scalar product calculations:

$$y_j = \sum_{s=1}^k W_{js} x_s \quad (11)$$

where  $k$  – number of products,  $s = 1, 2, \dots, k$ ;  $j = 1, 2, \dots, N$ .

The calculation of scalar products will be achieved using the tabular-algorithmic method, where the weights  $W_{js}$  are set in floating-point format, and the input data  $x_s$  – in a fixed-point format with fixation before the highest digit. Tabular-algorithmic calculation of the mantissa of the scalar product is reduced to reading the macro-partial product  $P_{Mi}$  from the  $j$ -th table (memory) at the address corresponding to the  $i$ -th bit slice of  $N$  input data, and adding it to the before accumulated sums according to:

$$y_{Mji} = 2^{-1} y_{Mj(i-1)} + P_{Mji}, \quad (12)$$

where  $y_{Mj0} = 0$ ,  $i = 1, \dots, m$ ,  $m$  – bit length of the input data. The number of tables of macro-partial products corresponds to  $N$  – the number of rows of the matrix (10). The result of calculating of the scalar product  $y_j$  consists of the mantissa  $y_{Mj}$  and the order  $m_j$ .

The time required to compute the mantissa of the scalar product (SP) is determined by the formula:

$$t_{SP} = m(t_{table} + t_{reg} + t_{add}), \quad (13)$$

where  $t_{SP}$  is the time of calculation of the scalar product,  $t_{table}$  is the time of reading from the table (memory),  $t_{reg}$  is the time of reading (writing) from the register,  $t_{add}$  is the time of adding.

Data encryption can be performed either sequentially or in parallel, depending on the speed required. In the case of sequential encryption, the encryption time is the result of the formula:

$$t_{encrypt} = Nm(t_{table} + t_{reg} + t_{add}), \quad (14)$$

where  $t_{encrypt}$  – time required for encryption. The encryption time can be reduced by performing  $N$  operations of calculating the scalar product in parallel.

As a result of NN data encryption, we obtain  $N$  encrypted data in the form  $y_j = y_{Mj}2^{m_j}$ , where  $y_{Mj}$  – mantissa at the  $j$ -th output,  $m_j$  – order value at the  $j$ -th output. It is advisable to bring all encrypted data to the highest common order for transmission and such reduction to the greatest common order is performed in three stages:

- define the greatest order  $m_{encr}$ ;
- for each encrypted data  $y_j$  calculate the difference between the orders  $\Delta m_j = m_{encr} - m_j$ ;
- by performing shift of the mantissa  $y_{Mj}$  to the right by the difference of orders  $\Delta m_j$  we obtain mantissa of the encrypted data  $y_{Mj}^h$  reduced to the greatest common order.

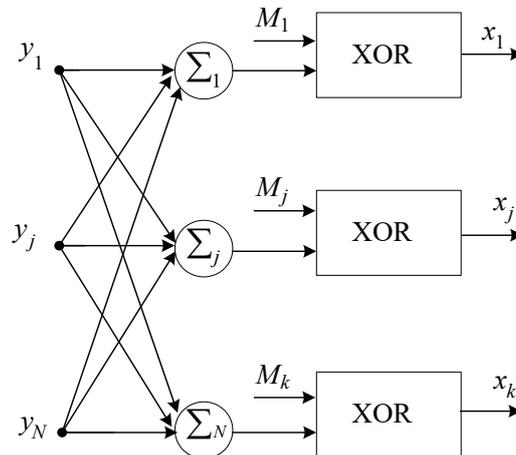
The mantissa of the encrypted data  $y_{Mj}^h$  reduced to the largest common order and the largest common order  $m_{encr}$  are sent for decryption.

### 3.3. The main stages of NN cryptographic data decryption

Now the encrypted data presented by mantissa  $y_{Mj}^h$ , reduced to the largest common order  $m_{encr}$  need to be decrypted. The encrypted data will be decrypted according to the following procedure.

*Configuration of the NN architecture for the decryption of encrypted data.* The architecture of the NN for the decryption of encrypted data, in terms of the number of neural elements, is the same as the architecture of the NN used for the encryption of data. In this NN, the number of inputs and the number of neurons corresponds to the number of the encrypted mantissa  $y_{Mj}^h$ . The NN architecture used to decrypt encrypted data is presented in Figure 3.

The bit rate of the inputs during decryption corresponds to the bit rate of the encrypted mantissa  $y_{Mj}^h$ . Its value determines the decryption time and to reduce it the lower bits of the mantissa may be discarded, because they will not affect the original message recovery.



**Figure 3.** The NN architecture for decryption of encrypted data.

*Formation of the weight matrix.* The matrix of weights for decrypting encrypted data is formed from a matrix of weights for encrypting input data by transposing it:

$$\begin{bmatrix} W_{11} & W_{12} & \dots & W_{1k} \\ W_{21} & W_{22} & \dots & W_{2k} \\ \vdots & \vdots & \dots & \vdots \\ W_{N1} & W_{N2} & \dots & W_{Nk} \end{bmatrix}^T = \begin{bmatrix} W_{11} & W_{21} & \dots & W_{N1} \\ W_{12} & W_{22} & \dots & W_{N2} \\ \vdots & \vdots & \dots & \vdots \\ W_{1k} & W_{2k} & \dots & W_{Nk} \end{bmatrix}. \quad (15)$$

The basic operation for encryption of input data and decryption of encrypted data is the calculation of the scalar product, which is implemented using a tabular-algorithmic method.

*Calculation of the table of macro-partial products for decryption of encrypted data.* A specific feature of the scalar product calculation operation used to decrypt encrypted data is that the weights are pre-calculated (constants) and set in floating-point format, while the encrypted data  $y_j$  are received in block-floating-point format. The calculation of the scalar product using the tabular-algorithmic method is performed by formula (7). Preparation and calculation of possible variants of macro-partial products is performed as in the previous case under the formula (8).

Amount of encrypted data determines the number of macro-partial products  $P_{Mi}$  and the size of the table. The largest common order  $m_{Pms}$  is computed for each table.

*NN tabular-algorithmic decryption of encrypted data.* The NN decryption is specified by multiplying the  $W$  matrix by the encrypted data vector  $\bar{y}$ :

$$x_s = \begin{bmatrix} W_{11} & W_{21} & \dots & W_{N1} \\ W_{12} & W_{22} & \dots & W_{N2} \\ \vdots & \vdots & \dots & \vdots \\ W_{1k} & W_{2k} & \dots & W_{Nk} \end{bmatrix} \times \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}. \quad (16)$$

The multiplication of the weights matrix  $W^T$  by the input data vector  $\bar{y}$  is reduced to performing  $N$  scalar product calculations:

$$x_s = \sum_{j=1}^N W_{sj} y_j \quad (17)$$

where  $N$  – number of products,  $s = 1, 2, \dots, k$ ;  $j = 1, 2, \dots, N$ .

Tabular-algorithmic calculation of the mantissa of the scalar product is reduced to reading the macro-partial product  $P_{Mi}$  from the table (memory) at the address corresponding to the  $i$ -th bit-slice of  $k$  input data, and adding it to the previously accumulated sums, according to the formula:

$$x_{Msi} = 2^{-1} y_{Ms(i-1)} + P_{Msi}, \quad (18)$$

where  $x_{s0} = 0$ ,  $i = 1, \dots, g$ ,  $g$  – bit rate of the encrypted data. The time necessary to calculate the scalar product mantissa is defined under the formula:

$$t_{SP} = g(t_{table} + t_{reg} + t_{add}), \quad (19)$$

where  $t_{SP}$  – time for scalar product calculation,  $t_{table}$  – time for reading from a table (memory),  $t_{reg}$  – time of reading (writing) from the register,  $t_{add}$  – time for adding. The result of the calculation of  $x_s$  scalar product consists of a mantissa  $x_{Ms}$  and order, which is equal to  $m_{decrs} = m_{Pms} + m_{encr}$ .

At the output of the NN (see Figure 3), we obtain  $k$  decrypted data in the following form  $x_s = x_{Ms} 2^{m_{decrs}}$ , where  $x_{Ms}$  is the mantissa at the  $s$ -th output,  $m_{decrs}$  is the value of the order at the  $s$ -th output. To obtain the input data, it is necessary to shift the  $s$ -th mantissa  $x_{Ms}$  by the value of the order  $m_{decrs}$ .

#### 4. The structure of the system for NN cryptographic data protection and transferring in real-time mode

The development of the structure of the system for NN cryptographic data protection and transmission in real-time will be carried out using an integrated approach, which contains:

research and development of theoretical foundations of NN cryptographic data encryption and decryption;  
 development of new tabular-algorithmic algorithms and structures for NN cryptographic data encryption and decryption;  
 modern element base, development environment and computer-aided design tools.

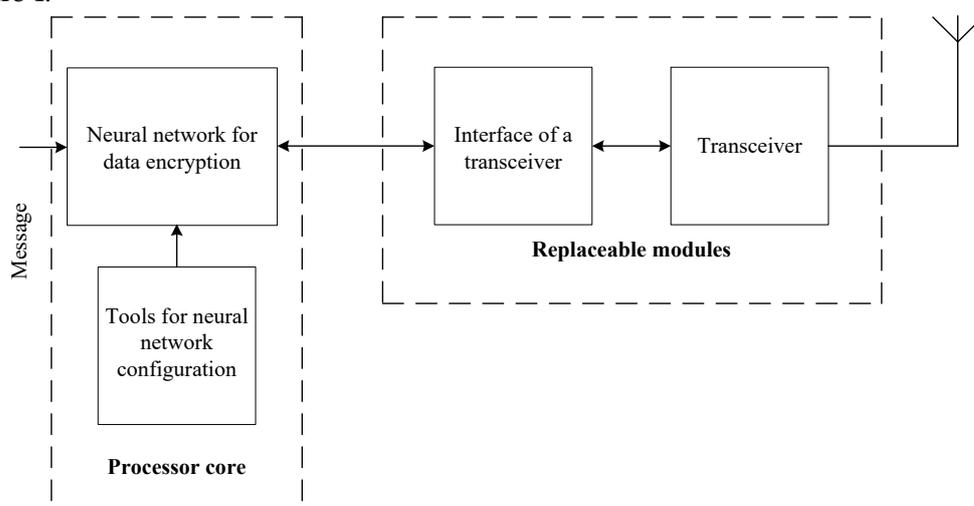
A system for NN cryptographic data protection in real-time was developed using the following principles:

changeable composition of the equipment, which foresees the presence of the processor core and replaceable modules, with which the core adapts to the requirements of a particular application;  
 modularity, which involves the development of system components in the form of functionally complete devices;  
 pipeline and spatial parallelism in data encryption and decryption;

- the openness of the software, which provides opportunities for development and improvement, maximising the use of standard drivers and software.;
- specialising and adapting hardware and software to the structure of tabular algorithms for encrypting and decrypting data.;

the programmability of hardware module architecture through the use of programmable logic integrated circuits.

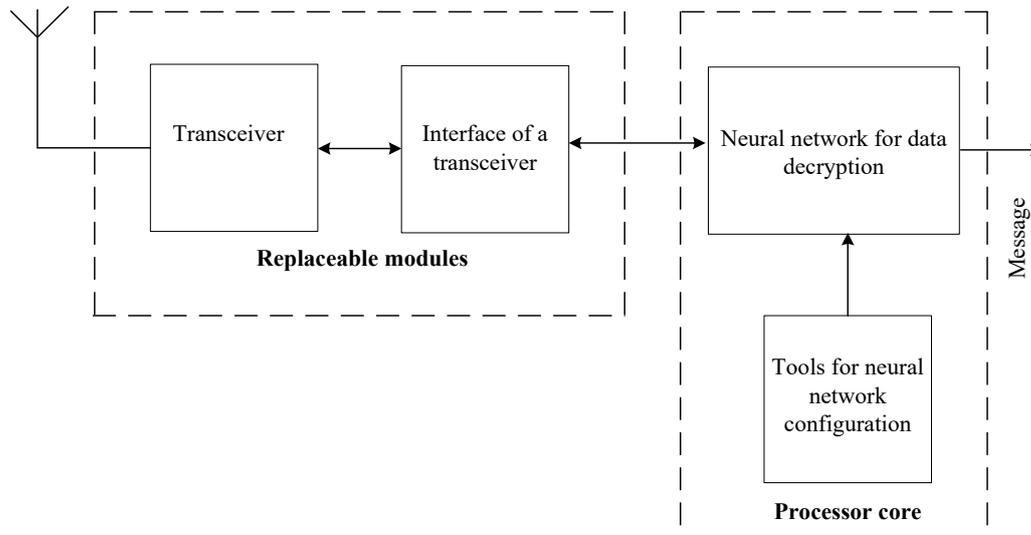
The system of NN cryptographic real-time data protection and transmission consists of a stationary part, which is a remote-control centre, and an UAV onboard part. The structure of the stationary part of the system of NN cryptographic data protection and transmission is shown in Figure 4.



**Figure 4.** Structure of the stationary part of the system of NN cryptographic data protection and transmission.

The processor core of the remote-control centre is implemented on the basis of a personal computer. The transceiver is used to transmit encrypted data; it communicates with the processor core through the interface based on a microcontroller.

The UAV onboard part of the system for NN cryptographic real-time data protection and transmission is implemented on the processor core, which is supplemented by dedicated hardware and software. The processor core of the UAV onboard part of the system is designed on a microcomputer. The structure of the onboard part of the system of NN cryptographic data protection and receiving is depicted in Figure 5.



**Figure 5.** Structure of the UAV onboard part of the system of NN cryptographic data protection and transmission.

The effective implementation of NN encryption-decryption and encoding-decoding algorithms in real time is achieved by combining universal and customized software and hardware. The use of modern elements (microcomputer, microcontroller, FPGA) in the development of the UAV onboard part ensures the accomplishment of the requirements for weight, dimensions and energy consumption.

The effectiveness of the system for NN cryptographic real-time data protection and transmission is directly associated with the choice of both hardware and software implementation.

## 5. Development of the components of the onboard system for NN cryptographic data encryption and decryption

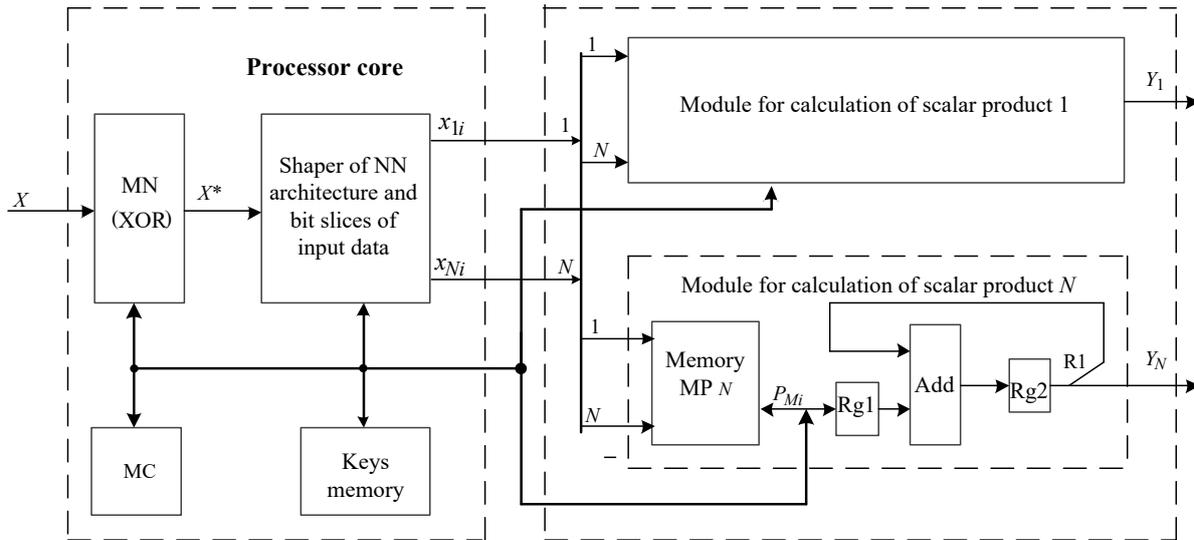
### 5.1. Development of the structure of the components for NN cryptographic data encryption and decryption.

In general, the problem of developing onboard systems for NN cryptographic encryption-decryption of data can be formulated as follows:

- to develop an algorithm for the onboard system of NN encryption-decryption of data and present it in the form of a specified flow graph;
- to design the structure of the onboard system for NN data encryption-decryption with the maximum efficiency of equipment use, taking into account all the limitations and providing real-time data processing;
- to determine the main characteristics of neural elements and carry out their synthesis;
- to choose exchange methods, determine the necessary connections and develop algorithms for exchange between system components;
- to determine the order of implementation in time of NN data encryption-decryption processes and develop algorithms for their management.

Components of the onboard system of NN cryptographic data encryption and decryption should provide the implementation of the selected NN, the ability to change masks, calculate matrices of weights  $W_j$  and tables of macro-partial products  $P_{Mi}$  for possible NN options. To effectively implement the components of the onboard system of NN cryptographic encryption-decryption of data, it is proposed to use hardware-software implementation of the algorithms based on a microcontroller supplemented by specialized hardware. The structure of the component of NN

cryptographic data encryption, which meets such requirements, is presented in Figure 6, where MC – microcontroller, MN – mask node, MP – macro-partial product, Rg – register, Add – adder.



**Figure 6.** Structure of the component of NN cryptographic encryption of data.

The developed component of NN cryptographic data encryption has a variable composition of equipment, which is based on the core of the system and a set of modules for calculating the scalar product. The system core is constant for all applications and consists of microcontroller MC, mask node MN, keys memory and module of the shaper of the NN architecture and bit slices of input data. The scalar product calculation modules implement the basic operation of the tabular-algorithmic method of scalar product calculation under the formula:

$$Z_i = 2^{-1}Z_{i-1} + P_{Mi}, \quad (20)$$

where  $Z_0 = 0$ .

The number of modules for calculating the scalar product depending on the required speed is determined by the following formula:

$$s = \frac{N}{2^v}, \quad (21)$$

where  $N$  is the number of neuro-like elements,  $v = 0, \dots, d$ ,  $d = \log_2 N$ . The system of NN cryptographic data encryption reaches its highest speed when the number of computational modules of the scalar product corresponds to the number of neural elements  $N$ . To ensure real-time data encryption, it is proposed to implement the scalar product calculation modules, mask node module (MN), and module of the shaper of NN architecture and bit slices of the input data in the form of specialized hardware.

The NN cryptographic data encryption component works as follows. Before encrypting the data, the MC configures the NN architecture (determines the number of neural elements  $N$ , the number of inputs  $k$  and their bit-size  $m$ ). For the selected NN architecture matrix of weights  $W_j$  and tables of  $P_{Mi}$  macro-partial products are calculated by MC, and then they are written in the memory of MP. In addition, the masks selected from the keys' memory are stored in the MN node. The message  $X$  to be encrypted comes to input of MN in fixed-point format, here it is masked. The masked message  $X^*$  from the output of MN comes to input of the module of the shaper of NN architecture and bit slices, where it is divided into  $N$  groups with  $m$  bit rate and bit slices are formed  $x_{1i}, \dots, x_{Ni}$ . It should be noted that forming of bit slices  $x_{1i}, \dots, x_{Ni}$  begins with lower bits. The formed bit slices  $x_{1i}, \dots, x_{Ni}$  are the addresses for reading macro-partial products  $P_{Mi}$  from the MP memory. The read macro-partial product  $P_{Mi}$  is written to the Rg1 register. The adder (Add) performs a summation of macro-partial products  $P_{Mi}$  as per formula (20). The number of cycles required to calculate the scalar product is

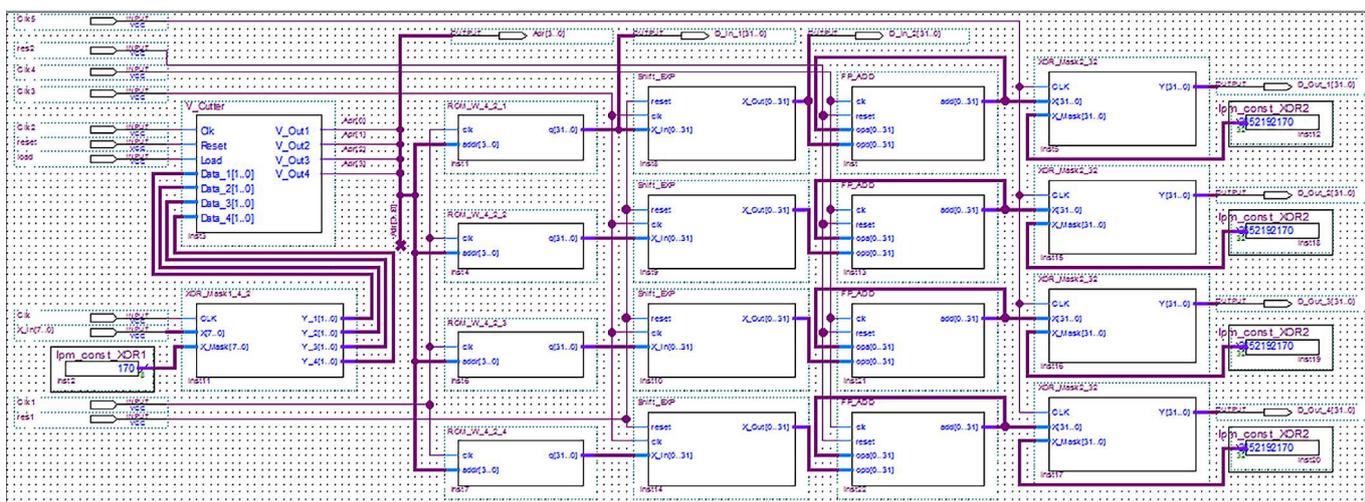
determined by the bit-size of input  $m$ . Control of the encryption process in the onboard system of NN cryptographic data encryption is performed by MC.

The structure of the component of NN cryptographic decryption of the encrypted data corresponds in general to the structure of the component of NN cryptographic encryption of data, presented in Figure 6.

### 5.2. Implementation of the specialized hardware components of NN cryptographic data encryption on FPGA.

The design of specialized on-board hardware systems for NN cryptographic data encryption was performed in the VHDL hardware programming language in the Quartus II ver. 13.1 development environment using its libraries. Nowadays, the hardware description languages such as VHDL, VHDL-AMS, Verilog, Verilog-AMS are widely used for creating behavioral descriptions and models of digital, analog and mixed-signal devices and systems [44, 45]. The Quartus II development environment supports the entire process of designing specialized hardware from user input to FPGA programming, debugging of both the chip itself and the tools as a whole.

A schematic diagram of the specialized hardware components of NN cryptographic data encryption is shown in Figure 7. The inputs of module XOR\_Mask1\_4\_2: X[7..0] – are the input data; Clk – input sync for input data download; X\_Mask[7..0] – 8-bit mask. At the output of this block  $N$  vectors with bit length  $m$  are formed. Synchronization is implemented on the leading edge of Clk pulses.



**Figure 7.** A circuit of the specialized hardware components of NN cryptographic data encryption.

Block V\_Cutter with  $N = 4$  input vectors of bit length  $m = 2$  consists of  $N$  registers of parallel-serial type and forms vertical bit slices. Input data: Data\_1 [n-1..0], ..., Data\_N [n-1..0] –  $N$  input vectors with bit length  $n$ ; Clk – pulses of synchronization of forming vertical bit slices; Reset – the signal of the initial reset in the "0" output of the registers R\_Par\_Ser; Load – the signal to allow data to be loaded into the R\_Par\_Ser registers. Outputs: V\_Out1, ..., V\_OutN – vertical bit slice. The formation of vertical sections begins with the lower bit.

The weights of the NN with  $N = 4$  inputs with a bit length of  $m = 2$  are stored in the FPGA ROM in the form of 4 tables. Each of them consists of 16 words with a bit length of 32 bits. Reading data from these tables is performed using blocks ROM\_W\_4\_2\_1, ..., ROM\_W\_4\_2\_4.

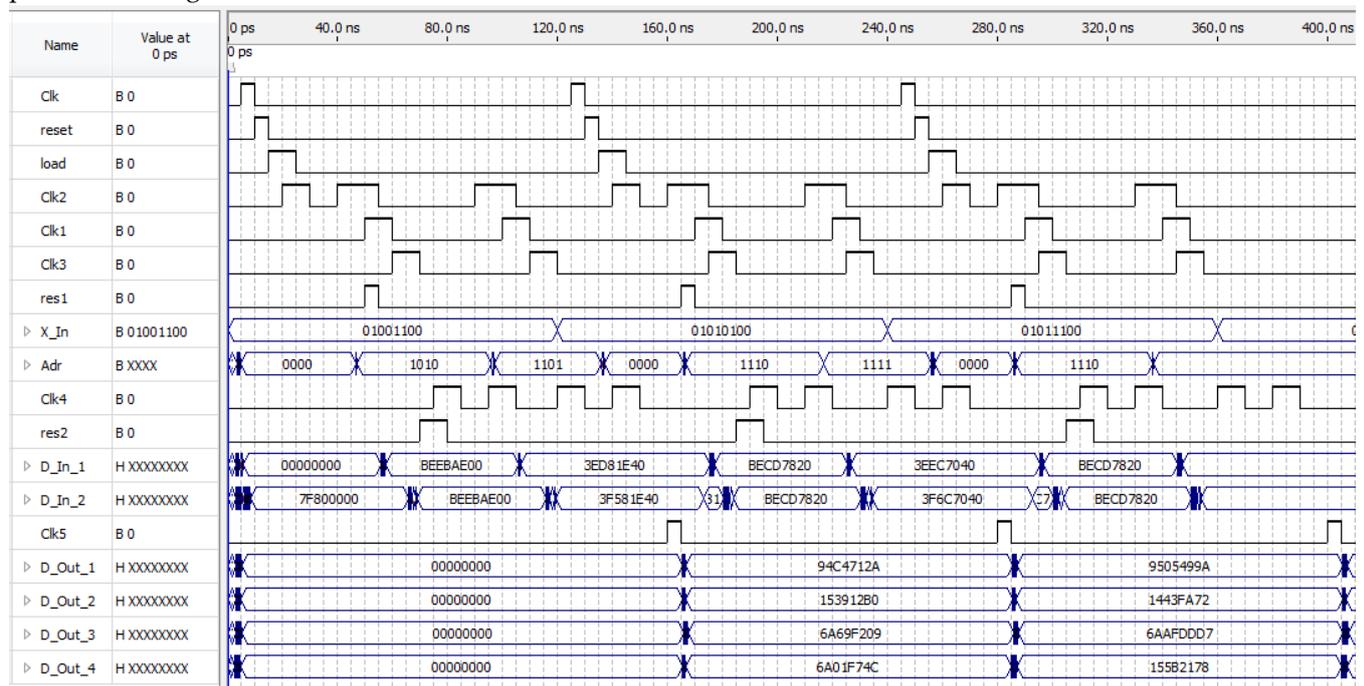
Inputs of these blocks: addr [3..0] – the address of the cell of the table from which the data will be read; clk – synchronization pulses for reading data from the table. Synchronization is implemented on the leading edge of the pulses clk. Output: q [31..0] – data read from the cell with the input address.

The data read from the tables is transmitted to the input blocks Shift\_EXP, which perform their multiplication by  $2^j$ , where  $j = 0, \dots, n - 1$ . Upon receipt of this block of data corresponding to the zero digit, the bit counter is reset. Synchronization of this block is carried out by means of clock pulses Clk. At the output X\_Out [0..31] we get the input data multiplied by  $2^j$ .

From the output of the Shift\_EXP blocks, the data are sent to one of the inputs of the adders FP\_ADD. The other input of the adders is connected to their output. Adder input signals: clk – synchronization pulses; reset – signal to reset the input data opa when implementing the adder with the battery; opa[0..31], opb[0..31] – terms. On the leading edge of the first pulse clk the adders are loaded into the adder, and on the leading edge of the second pulse the received sum is displayed. Adder output: the sum add[0..31].

From the output of the adders, FP\_ADD data is fed to the input of the block XOR\_Mask2\_32, which performs the overlay of the 32-bit mask. Inputs of the block XOR\_Mask2\_32: X [31..0] – encrypted output data; Clk – synchronization of input data download; X\_Mask [31..0] – 32-bit mask. Block output: vector Y [31..0]. Synchronization is implemented on the leading edge of Clk pulses. The encrypted data are obtained at the outputs D\_Out\_1, D\_Out\_2, D\_Out\_3, D\_Out\_4.

The timing diagram of the specialized hardware of NN cryptographic data encryption is presented in Figure 8.



**Figure 8.** The timing chart of the specialized hardware of NN cryptographic data encryption.

The implementation of the specialized hardware for NN cryptographic data encryption based on the FPGA EP3C16F484C6 Cyclone III family [46] requires 3053 logic elements and 745 registers. Approximately 160 nanoseconds are required to encrypt one input vector.

## 6. Conclusions

The NN technology for real-time cryptographic data protection with symmetric keys for UAV onboard communication systems has been presented in this work. In the proposed implementation the keys are defined by masking codes, NN architecture and weigh matrix. It makes possible the effective hardware-software implementation with good technical and operational characteristics.

The tabular-algorithmic scalar product calculation method has been improved. This method is suitable for fast scalar product calculation of input data in fixed-point or floating-point format. It does this by finding the largest common order of weights and building tables of macro partial products for them.

This paper describes the UAV onboard system for NN cryptographic data protection in real-time using an integrated approach based on the following principles: variable equipment composition; modularity; conveyorization and spatial parallelism; software openness; suitability for hardware implementation on FPGA.

Components of NN cryptographic data encryption/decryption have been designed on the basis of the processor core supplemented by the specialized scalar product calculation modules.

The specialized hardware for NN cryptographic data encryption was developed in the VHDL equipment programming language in the Quartus II environment and implemented using family Cyclone III FPGA EP3C16F484C6.

**Author Contributions:** Conceptualization, I.T. and V.T.; methodology, I.T., Y.O. and Y.L.; software, Y.L. and Y.O.; validation, Y.L., Y.O. and I.K.; formal analysis, I.T. and V.T.; investigation, A.L. and A.H.; resources, A.L. and I.K.; data curation, Y.L. and I.K.; writing—original draft preparation, I.T., I.K. and Y.O.; writing—review and editing, I.K., A.H. and A.L.; visualization, I.K. and Y.O.; supervision, V.T. and A.L.; project administration, I.T. and V.T. All authors have read and agreed to the published version of the manuscript.

**Funding:** Please add: “This research received no external funding” or “This research was funded by NAME OF FUNDER, grant number XXX” and “The APC was funded by XXX”. Check carefully that the details given are accurate and use the standard spelling of funding agency names at <https://search.crossref.org/funding>, any errors may affect your future funding.

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

## References

1. Han, B.; Qin, D.; Zheng, P.; Ma, L.; Teklu, M.B. Modeling and performance optimization of unmanned aerial vehicle channels in urban emergency management, *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 478. <https://doi.org/10.3390/ijgi10070478>
2. Śledź, S.; Ewertowski, M.W.; Piekarczyk, J. Applications of unmanned aerial vehicle (UAV) surveys and Structure from Motion photogrammetry in glacial and periglacial geomorphology. *Geomorphology* **2021**, *378*, 107620. <https://doi.org/10.1016/j.geomorph.2021.107620>
3. Zhang, C.; Zou, W.; Ma, L.; & Wang, Z. Biologically inspired jumping robots: A comprehensive review. *Robotics Auton. Syst.* **2020**, *124*, 103362. <https://doi.org/10.1016/j.robot.2019.103362>
4. Weng, Z.; Yang, Y.; Wang, X.; Wu, L.; Hua, S.; Zhang, H.; Meng, Z. Parentage analysis in giant grouper (*epinephelus lanceolatus*) using microsatellite and SNP markers from genotyping-by-sequencing data. *Genes* **2021**, *12*, 1042. <https://doi.org/10.3390/genes12071042>
5. Boreiko, O.; Teslyuk, V.; Zelinsky, A.; Berezsky, O. Development of models and means of the server part of the system for passenger traffic registration of public transport in the “smart” city. *EEJET* **2017**, *1* (85), 40–47. <https://doi.org/10.15587/1729-4061.2017.92831>
6. Kim, K.; and Kang, Y. Drone security module for UAV data encryption. In Proceedings 2020 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea (South), 2020, 1672-1674. <https://doi.org/10.1109/ICTC49870.2020.9289387>
7. Samanth, S.; K V, P.; & Balachandra, M. Security in Internet of Drones: A Comprehensive Review. *Cogent Engineering* **2022**, *9*(1), 2029080. <https://doi.org/10.1080/23311916.2022.2029080>
8. Kong, P.-Y. A survey of cyberattack countermeasures for unmanned aerial vehicles. *IEEE Access* **2021**, *9*, 148244-148263. <https://doi.org/10.1109/ACCESS.2021.3124996>
9. Shafique, A.; Mehmood, A.; Elhadeif, M.; Khan, KH. A lightweight noise-tolerant encryption scheme for secure communication: An unmanned aerial vehicle application. *PLOS ONE* **2022**, *17*(9), e0273661. <https://doi.org/10.1371/journal.pone.0273661>
10. Verma, A.; and Ranga, V. Security of RPL based 6LoWPAN Networks in the Internet of Things: A Review. *IEEE Sensors J.* **2020**, *20*, 11, 5666–5690. <https://doi.org/10.1109/ISEN.2020.2973677>
11. Morales-Molina, C.D.; Hernandez-Suarez, A.; Sanchez-Perez, G.; Toscano-Medina, L.K.; Perez-Meana, H.; Olivares-Mercado, J.; Portillo-Portillo, J.; Sanchez, V.; Garcia-Villalba, L.J. A Dense Neural Network Approach for Detecting Clone ID Attacks on the RPL Protocol of the IoT. *Sensors* **2021**, *21*, 3173. <https://doi.org/10.3390/s21093173>
12. Sohaib, O.; W. Hussain, M. Asif, M. Ahmad and M. Mazzara, A PLS-SEM neural network approach for understanding cryptocurrency adoption. *IEEE Access* **2020**, vol. 8, pp. 13138-13150, <https://doi.org/10.1109/ACCESS.2019.2960083>
13. Holovatyy A., Łukaszewicz A., Teslyuk V., Ripak N. Development of AC Voltage Stabilizer with Microcontroller-Based Control System In Proceedings of the 2022 IEEE 17th International Conference on Computer Sciences and Information Technologies (CSIT), 2022, Institute of Electrical and Electronics Engineers, 2022, 527-530. <https://doi.org/10.1109/CSIT56902.2022.10000461>

14. Grodzki W., Łukaszewicz A. Design and manufacture of unmanned aerial vehicles (UAV) wing structure using composite materials, *Materialwissenschaft und Werkstofftechnik*, 2015, 46 (3), 269-278. <https://doi.org/10.1002/mawe.201500351>
15. Łukaszewicz A., Szafran K., Jóźwik J. CAx techniques used in UAV design process, In Proceedings of the 2020 IEEE 7th International Workshop on Metrology for AeroSpace (MetroAeroSpace), 2020, 95-98. <https://doi.org/10.1109/MetroAeroSpace48742.2020.9160091>
16. Łukaszewicz A., Skorulski G., Szczebiot R. The main aspects of training in the field of computer aided techniques (CAx) in mechanical engineering. In Proceedings of the 17th International Scientific Conference on Engineering for Rural Development, May 23-25, 2018, Jelgava, Latvia, 2018, 865-870. <https://doi.org/10.22616/ERDev2018.17.N493>
17. Łukaszewicz A., Miatluk K. Reverse Engineering Approach for Object with Free-Form Surfaces Using Standard Surface-Solid Parametric CAD System, *Solid State Phenomena*, 2009, 147-149, 706-711. <https://doi.org/10.4028/www.scientific.net/SSP.147-149.706>
18. Miatluk K., Łukaszewicz A., Siemieniako F. Coordination method in design of forming operations of hierarchical solid objects, In Proceedings of the 2008 International Conference on Control, Automation and Systems, ICCAS 2008, pp. 2724–2727, 4694220. <https://doi.org/10.1109/ICCAS.2008.4694220>
19. Puchalski R., Giernacki W. UAV Fault Detection Methods, State-of-the-Art, *Drones*, 2022, 6, 330. <https://doi.org/10.3390/drones6110330>
20. Zietkiewicz J., Kozierski P., Giernacki W. Particle swarm optimisation in nonlinear model predictive control; comprehensive simulation study for two selected problems, *International Journal of Control*, 2021, 94(10), 2623-2639. <https://doi.org/10.1080/00207179.2020.1727957>
21. Kownacki C., Ambroziak L. Adaptation Mechanism of Asymmetrical Potential Field Improving Precision of Position Tracking in the Case of Nonholonomic UAVs, *Robotica*, 2019, 37(10), 1823-1834. <https://doi.org/10.1017/S0263574719000286>
22. Kownacki C., Ambroziak L., Ciężkowski M., Wolniakowski A., Romaniuk S., Bożko A., Ołdziej D. Precision Landing Tests of Tethered Multicopter and VTOL UAV on Moving Landing Pad on a Lake, *Sensors* 2023, 23, 2016. <https://doi.org/10.3390/s23042016>
23. Basri, E.I.; Sultan, M.T.H.; Basri, A.A.; Mustapha, F.; Ahmad, K.A. Consideration of Lamination Structural Analysis in a Multi-Layered Composite and Failure Analysis on Wing Design Application. *Materials* 2021, 14, 3705. <https://doi.org/10.3390/ma14133705>
24. Al-Haddad L.A., Jaber A.A., An Intelligent Fault Diagnosis Approach for Multicopter UAVs Based on Deep Neural Network of Multi-Resolution Transform Features, *Drones*, 2023, 7, 82. <https://doi.org/10.3390/drones7020082>
25. Yang J., Gu H., Hu C., Zhang X., Gui G., Gacanin H. Deep Complex-Valued Convolutional Neural Network for Drone Recognition Based on RF Fingerprinting, *Drones*, 2022, 6, 374. <https://doi.org/10.3390/drones6120374>
26. Holovatyy A., Teslyuk V., Lobur M., Sokolovskyy Y., Pobereyko S. Development of Background Radiation Monitoring System Based on Arduino Platform. *International Scientific and Technical Conference on Computer Science and Information Technologies*, 2018, pp. 121 – 124. <https://doi.org/10.1109/STC-CSIT.2018.8526696>
27. Holovatyy A., Teslyuk V., Lobur M., Szermer M., Maj C. Mask Layout Design of Single- and Double-Arm Electrothermal Microactuators. *Perspective Technologies and Methods In MEMS Design, MEMSTECH 2016 – Proceedings of 12th International Conference*, 2016, pp. 28 – 30. <https://doi.org/10.1109/MEMSTECH.2016.7507513>
28. Volna, E.; Kotyrba, M.; Kocian, V.; & Janosek, M. Cryptography Based On Neural Network. In Proceedings of the 26th European Conference on Modeling and Simulation (ECMS 2012), edited by: K. G. Troitzsch, M. Moehring, U. Lotzmann. European Council for Modeling and Simulation. Koblenz, Germany, May 29 – June 1, 2012, 386-391. <http://dx.doi.org/10.7148/2012-0386-0391>
29. Shihab, K. A backpropagation neural network for computer network security. *Journal of Computer Science* 2006, 2 (9), 710–715. <https://doi.org/10.3844/jcssp.2006.710.715>
30. Sagar, V.; Kumar, K. A symmetric key cryptographic algorithm using counter propagation network (CPN). In Proceedings of the 2014 ACM International Conference on Information and Communication Technology for Competitive Strategies, (ICTCS'14). Udaipur Rajasthan India, November 14-16, 2014. <https://doi.org/10.1145/2677855.2677906>
31. Arvandi, M.; Wu, S.; Sadeghian, A.; Melek, W.W.; Woungang, I. Symmetric cipher design using recurrent neural networks. In Proceedings of the IEEE International Joint Conference on Neural Networks, 2006, 2039–2046. <https://doi.org/10.1109/IJCNN.2006.246972>

32. Tsmots, I.; Tsymbal, Y.; Khavalko, V.; Skorokhoda, O.; Teslyuk, T. Neural-like means for data streams encryption and decryption in real time. In Proceedings of the 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP). Lviv, Ukraine, August 21-25, 2018, 438-443. <https://doi.org/10.1109/DSMP42010.2018>
33. Scholz, M.; Fraunholz, M.; Selbig, J. Nonlinear principal component analysis: neural network models and applications. In: Gorban, A.N., Kégl, B., Wunsch, D.C., Zinovyev, A.Y. (eds) *Principal Manifolds for Data Visualization and Dimension Reduction. Lecture Notes in Computational Science and Engineering*, 58, 2008, Springer, Berlin, Heidelberg. [https://doi.org/10.1007/978-3-540-73750-6\\_2](https://doi.org/10.1007/978-3-540-73750-6_2)
34. Rabyk, V.; Tsmots, I.; Lyubun, Z.; Skorokhoda, O. Method and Means of Symmetric Real-time Neural Network Data Encryption. In Proceedings of the 2020 IEEE 15th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT 2020), 2020, 1, 47-50. <https://doi.org/10.1109/CSIT49958.2020.9322006>
35. Chang, A.X.M.; Martini, B.; Culurciello, E. Recurrent Neural Networks Hardware Implementation on FPGA: arXiv preprint arXiv:1511.05552. 2015. <https://doi.org/10.48550/arXiv.1511.05552>
36. Nurvitadhi, E. et al. Can FPGAs beat GPUs in accelerating next-generation deep neural networks? In Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. Monterey, California, USA, February 22-24, 2017, 5-14. <https://doi.org/10.1145/3020078.3021740>
37. Misra, J.; Saha, I. Artificial neural networks in hardware: A survey of two decades of progress. *Neurocomputing* **2010**, 74 (1-3), 239-255. <http://dx.doi.org/10.1016/j.neucom.2010.03.021>
38. Guo, K. et al. From model to FPGA: Software-hardware co-design for efficient neural network acceleration. In Proceedings of the 2016 IEEE Hot Chips 28 Symposium (HCS). 2016, 1–27. <http://dx.doi.org/10.1109/HOTCHIPS.2016.7936208>
39. Ovtcharov, K. et al. Accelerating Deep Convolutional Neural Networks Using Specialized Hardware. Microsoft Research Whitepaper. 2016. Available online: <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/CNN20Whitepaper.pdf> (accessed 29 April 2022)
40. Wang, Y.; Xu, J.; Han, Y.; Li, H. and Li, X. DeepBurning: automatic generation of FPGA-based learning accelerators for the neural network family. In Proceedings of the 53rd Annual Design Automation Conference (DAC'16). Association for Computing Machinery, New York, NY, USA, Article 110, 1–6. <https://doi.org/10.1145/2897937.2898003>
41. Nurvitadhi, E.; Sheffield, D.; Sim, J.; Mishra, A.; Venkatesh, G., and Marr, D. Accelerating Binarized Neural Networks: Comparison of FPGA, CPU, GPU, and ASIC. In Proceedings 2016 International Conference on Field-Programmable Technology (FPT), 2016, 77-84. <http://dx.doi.org/10.1109/FPT.2016.7929192>
42. Yayik, A.; Kutlu, Y. Neural Network Based Cryptography. *Neural Network Worldc*, 2014, 24 (2), 177-192. <http://dx.doi.org/10.14311/nnw.2014.24.011>
43. Govindu, G.; Zhuo, L.; Choi, S.; and Prasanna, V. Analysis of high-performance floating-point arithmetic on FPGAs. In Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS 2004). 26-30 April, 2004, Santa Fe, New Mexico, USA, 149. <http://dx.doi.org/10.1109/IPDPS.2004.1303135>
44. Holovatyy A., Teslyuk V., Lobur M. VHDL-AMS model of delta-sigma modulator for A/D converter in MEMS interface circuit. Perspective Technologies and Methods In MEMS Design, MEMSTECH 2015 – Proceedings of the 11th International Conference, 2015, pp. 55-57. <https://doi.org/10.1109/CADSM.2015.7230865>
45. Holovatyy A., Lobur M., Teslyuk V. VHDL-AMS model of mechanical elements of MEMS tuning fork gyroscope for the schematic level of computer-aided design. Perspective Technologies and Methods In MEMS Design – Proceedings of the 4th International Conference of Young Scientists, MEMSTECH 2008, 2008, pp. 138 – 140. <https://doi.org/10.1109/MEMSTECH.2008.4558762>
46. Electronic components database. Available online: <https://www.digchip.com/datasheets/parts/datasheet/033/EP3C16F484C6.php> (accessed on 29 April 2022).

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