УДК 517.9:621.325.5:621.382.049.77 https://doi.org/10.15407/pp2021.02.063

M. Kosovets, L. Tovstenko

SPECIFIC FEATURES OF THE USE OF ARTIFICIAL INTELLIGENCE IN THE DEVELOPMENT OF THE ARCHITECTURE OF INTELLIGENT FAULT-TOLERANT RADAR SYSTEMS

The problem of developing the architecture of modern cognitive radar systems in the form of a set of heterogeneous neuromultimicroprocessor modules using artificial intelligence technologies, taking into account the requirements for the purpose, the influence of external and internal factors, is considered. The concept of a resource in general and an abstract reliability resource in particular and its role in the design of a neuromultimicroprocessor with fault tolerance properties are introduced. The change in the ratio of performance and reliability of a neural network is shown, which is rebuilt in the process of solving a problem in real time with a lack of reliability resources at the system level by means of the operating system which dynamically changing the architectural appearance of the system with structural redundancy, using fault-tolerant technologies and dependable computations.

Keywords: neuromultimicroprocessor, probability of trouble-free operation, initialization, resource, interface, modularity, supervisor, multiprogramming, reconfiguration system, access method.

Introduction

The growth of the use of radar technologies in various sectors of the economy: medicine, military equipment, security, agriculture, geology, IoT and others became possible due to the miniaturization of the element base, the development of artificial intelligence technologies, cloud computing [1,2]. Cognitiveness of modern radars plays a key role, and there is no alternative to this path [3]. The main requirements for radars are to minimize the negative impact on human health, the surrounding electronic devices, their invisibility, to obtain information about environmental pollution, 3D images of the location scene, information about the health of the environment, the presence of viruses and bacteria [4-9]. These requirements complicate both the transmitting part of the radar and the receiving part, which is associated with the problem of extracting signals from the noise. The development of technology explains the failed attempts to build ground penetrating radar (GRP), mine detection radars (Mine Radar), Portable Smart through Wall 3D Imager Radar (PSTW), marine radars with a low probability of interception of the near and far zone ("Low Probability of Intercept" (LPI)) and ways overcoming them. This is the intellectualization of radars: the use of neural networks and their deep learning [10-20].

The deployment of radar systems has a multi-level system with continuous coverage and close connection between the levels. The primary link forms a network of radars, telecommunications environment, the second link - the deployment of military, and security radars [21-24], the third link - IoT radars, radars built into gadgets located in cars, pollution checkpoints, and virus detection and others. The construction of radars of different links has significant differences. The focus of the first link is on a reliable neural network and its connections to cloud computing and providing fast access. The second link is characterized by the use of a neural network to organize a neural computer and the use of neural network computing through deep learning. The third link is the most widely used, characterized by small size, often placed in gadgets, which teach artificial neural network, in contrast to the first two links. In first two links the neural network is designed and reproduced in hard ware by large teams of developers, creating and providing resources to the third link. Collection, processing, storage and reproduction of

information are possible from other sensors and information artificially entered for processing and storage.

Existing architectural models are not able to adequately display applied radar information. Unfortunately, artificial intelligence technologies do not yet have sufficient development, developed neural network components, experience of deep learning of the neural network, developments in cognitive algorithms, issues of achieving fault tolerance, providing dependability of computations in real time. Therefore, today we use a compromise option that combines signal processing and in-depth learning, neural networks with multiprocessing [25-28]. This will help to use all the developments so far, to make new developments that will be relevant in full intellectualization.

The use of neural network systems for processing radar information has many advantages. It is possible to gradually build up the neural network and train the modified network. Also, in the neural network, we lay the possibilities of increasing fault tolerance by reconfiguring the system in the process of solving the problem. Fault tolerance of a radar system with minimal time redundancy and deadlock termination is especially important in real-time systems [29-33]. An analysis of the problems associated with the processing of radar information shows that many tasks of processing radar information are solved using a neural network, the architecture of which does not correspond to the class of tasks. Thus, the tasks of processing multidimensional fields are solved using built-in micro-computers with a streaming accelerator, the architecture of which reflects the problems of organizing computations, ranked by dependability levels.

We pay special attention to the infrastructure of neural network design tools. Unfortunately, there are no neural network design tools on the market, except for debugging tools for individual components. The manufacture of a neural network on a crystal is spreading. Architectural models reflect the functionality of hardware and software components using high-level abstraction in the form of data streams and abstractly represent implementation technology over time. Architectural models contain arbitration schemes, can be parameterized and typed. Thus, the configuration parameters of the architectural model make it possible to determine the relationship between the implementation of functions by hardware and software. System-level design using architectural modeling simplifies the design specification, makes a smooth transition from functional requirements to formal requirements, since it separates the problems of developing functional requirements and design specifications, since there are usually no means to quantify specifications [34-38].

The problem of reliability of real -time radar intelligent systems

The problem of reliability plays an important role in the design of systems in networks for collecting, processing and transmitting information. By reliability we mean the probability that the system will perform a given function in a given period of time under specified environmental conditions. The analysis of the systems has shown that reliability at a basic level is a fundamental parameter. Since the systems are distributed in space, failure leads to the collapse of the entire system. Solution paths are changing all the time, especially now when neural networks are simultaneously used to ensure reliability and to solve an applied problem. On the one hand, they simplify the solution of the problem of reliability, survivability of systems, and on the other hand, they complicate the hardware and software. The article discusses design methods for fault-tolerant neuromultimicroprocessor real-time systems. These methods include software and circuit methods for detecting failures, which must ensure the regular operation of each processor module, data exchange between subsystems and the reliability of information before using it.

The problem of reliability is always considered at the design stage. Traditionally, failure prevention has been achieved in various ways, for example: by creating ultra-reliable multiprocessor components; improving maintenance through the development of effective troubleshooting methods; improving the procedure for controlling the technological process of manufacturing, testing and certification of finished products; implementation of hardware redundancy; the creation of technology for designing systems that have the properties

of resistance to failures in conditions when defects inevitably exist and manifest themselves in the form of failures and random failures. By fault tolerance, we mean such a property of the architecture of a digital system that allows a logical machine to continue working even when a variety of component failures occur in a real system that is its carrier [39]. The main task of the fault tolerance solution is to restore the computational process from the point of failure. To do this, it is necessary to detect and isolate the failure. To restore operability, knowledge of the state vector at the current time is required. Recovery techniques depend on the ability to isolate a detected failure in the system at the lowest possible level of system abstraction. The recovery mechanism proposed in this article not only ensures modularity and simplicity of the system, but also enables quick recovery and accurate prediction of the task completion time.

Initially, fault-tolerant technologies were developed for on-board electronic equipment, for which a number of parameters are critical: weight, dimensions, power consumption, system unification, time and money spent on designing a new system, the complexity of modernization procedures, and high reliability requirements. A typical system based on a real-time multiprocessor, insensitive to failures, should at least recognize 98% of all possible errors and identify at least 95%. To do this, we use built-in control systems, tests of acceptability and meaningfulness, and the implementation of reliability procedures.

After an error is detected, the faulty components are localized and excluded from the computational process. The system is reconfigured, the task is redistributed between free processors, which are initialized and included in the computational process from the point of failure. A restore point is defined by an application program that stores information up to the restore point. If an error is detected in the subsystem, recovery is possible through restart, which is impractical, since the computational process starts from the initial value, and if the computational processes are interconnected, then it becomes difficult to isolate the refusal from affecting other parallel processes. Therefore, when developing programs, parallel processes must be carefully structured so, that restore points in interacting processes are mutually consistent.

A replay of one process can propagate to other processes and events, which is called replay propagation. Sometimes there is an avalanche of repetitions. In such cases, the process goes back a few steps. In this case, redundancy in the recovery process and loss of productivity are inevitable. We have considered the issues related to the resource management of a fault-tolerant real-time neuromultimicroprocessor, now we will try to cover in more details the issue of the impact of resource management on the fault-tolerant system.

In the design of radar systems, solutions are still being sought to improve reliability through redundancy but their architecture is outdated. When building special-purpose systems, especially airborne, radar, telecommunications, there is no alternative to fault-tolerant architectures. The challenge of building fault-tolerant systems lies in the complete revision of ingrained design principles and ideology. The value of reliability is calculated and laid down at the system level and depends not only on hardware and software resources, but to a greater extent on their interaction, resource management. As a result, reliability acts as an abstract resource of the system and varies depending on the task being performed.

Fault tolerance is provided by hardware, software or hardware-software redundancy. Failure of an individual processor module manifests itself in a limited loop. Fault tolerance during operation is determined by error detection, reconfiguration of system components and restoration of error-free operation of the neuromultimicroprocessor.

The greatest recovery efficiency in case of failures of a neuromultimicro-processor element is achieved at the hardware level. To restore processor operation means to restore the correct state of the processors. The software implementation of the control process based on breakpoints is ineffective and is determined by the application problem and the requirements for the reliability of the computing system. When a malfunction is detected, diagnostic tests isolate the malfunction and rule out the defective processor element.

A neural network for solving loosely coupled processes is based on the principles

of functional separation and has a structure consisting of multi-microprocessor sets, a communication network and a system module. Each processor element is able to independently solve the assigned tasks using the internal structure and exchanging messages over the communication network. The scenario for diagnostics, recovery and degradation is embedded in the system module. With this architecture, the multiprocessor has its own local and system resources available to all processor elements.

The base module has a bus architecture neuromultimicroprocessor and includes serial, parallel and local bus exchange, private resources and resources that are shared by processor subsystems. Such architecture of the basic module allows processor expansion with homogeneous and heterogeneous modules, having previously agreed on the bus exchange protocol. It is important to allocate system resources related to share multi-microprocessor resources and individual local resources. This allows you to organize the reliability, integration, performance, efficiency of the base module at the level of system resources. Local resources provide the functioning of individual consumer tasks: exchange with cloud resources, provision of high-performance computing, and receiving data from multidimensional information sensors. Local resources are isolated from errors that appear in other parts of the system, which increases the reliability of the entire system.

We introduce a message space for the synchronous exchange of information in blocks at maximum speed without using the processor resource. Basic modules are identified by the central service module, initializing all modules in the system by geographic principle. With the geographic distribution of the base modules, information is exchanged over a radio channel. The coding of the exchange channel, protection, exchange rate depends on the application. We use the local serial bus in the base module to control and diagnose system resources.

The capabilities of the multi-microprocessor are flexibly rebuilt. The function of the interface module is responsible for the transmission and reception of expected and unexpected messages, access to the components of the system module: I / O registers and system memory. Dividing messages into expected and unexpected will optimize the transmission of short and long messages. The system module assigns arbitration, generates a system reset, provides data protection in the event of a power failure, and resumes starting when it occurs.

The work of a neuromultimicro-processor begins with initialization, transferring system components from an undefined state to a known one. The initialization process includes resetting, initializing individual modules, initializing the entire system, and booting. Initially, the processor boards have the same priority, during initialization, you can assign the modules by priority and assign the master, that is, assign the highest priority.

If the task is divided into N processes, then, if available, we assign N processor elements. Saving the state of a task involves saving the state of these processor elements. The device for storing the state of the task represents the stack memory, that is, the current state is the last write to the memory and is taken out first.

Suppose that the task is distributed among N processors (i = 1, 2 ... n). Saving the state of the task means saving the state of the processors. Repeating the process is equivalent to restoring the states of the corresponding processors. Due to the ambiguity of the interaction of processes and the asynchronous nature of saving states, restarting other processes or multi-step recovery may be required. To solve the problem of reconfiguration, network management and other system issues, a system monitor and a switch controller are included in the neuromultimicroprocessor. The controller analyzes replays and multi-step recovery. Performance Monitor receives a command to execute processes and allocates processors and system memory for it. Physically, the system monitor is located in the system unit or a separate processor is allocated and is endowed with the functions of a monitor and a communication network controller.

When an error occurs, System Monitor indicates that the operation has been restarted, and if it repeats, all processor modules are suspended. It detects the failed processors and resumes solving the problem using the processors in which there were no errors. In the

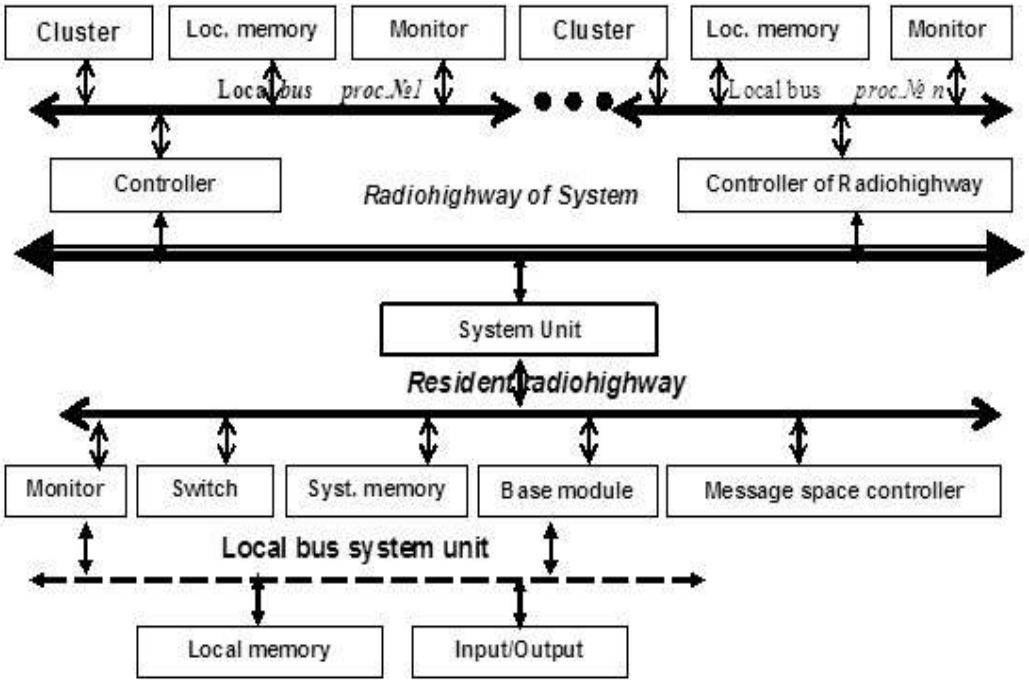

Fig.1. The architecture of an intelligent m-ensemble fault-tolerant neuromultimicroprocessor with a high-speed radio communication channel and a built-in recovery mechanism

presence of free processors, the task is redistributed taking into account them; in the absence of an adequate replacement, processes are distributed taking into account the available ones. When the number of processors decreases below the critical value, the system degrades, that is, processes are redistributed among the available processors, in which case the system performance decreases without deterioration in quality.

Let the failure of one processor module entail restarting a part of the computational process in all processor modules interacting with the data. We will assume that the restart process P_i entails a restart process P_i, that is, there is a propagation of restarts. Let's us denote the n-interval P_i as $T_i(n)$, and the initial time when P_i it retained its value as $t_i(n)$. Then, if a computation error occurs at the beginning of the time interval, and at the moment $t_i(n)$ the state is saved, interaction between processes is possible. In this case, the first process is restarted, and the interacting processes are suspended until the first is restored.

Consider a probabilistic model for estimating the restart propagation of a part of the computational process, with multi-step recovery. If an error occurs in calculations at a point in time t_s during the interval $T_i(k)$, a check is made for the possibility of recovery in one step. This will enable us to evaluate the effectiveness of this built-in failover mechanism. We assume that the processor module has $(N+1)ms$ cells for storing valid flags, and the task is distributed among M processor modules.

Let's calculate the range of multi-step recovery. It can be argued on the basis of the above stated that there was at least one call from module i to module j during the state saving interval. Formally, you can write it like this:

$$f_{ij} = f_{ji} = g_{ij} + g_{ji} - g_{ij}g_{ji}$$

where f_{ijn}: is the average probability of the propagation of restarts from the processor module i to the processor module j due to n-step recovery in the module i;

$$g_{ij} = \frac{1}{N_1}\sum_{k=1}^{N_1}(1-e^{u_{ijk}T_{ss}})$$

represents the average probability of access from module i to module j in one interval.

N_t: The total number of states of the processor unit saved until the completion of the task, provided that there were no failures.

$$N_t = \lfloor T_{ef}/(T_{ss} - T_{su}) \rfloor.$$

where: $T_i(k)$: Duration of the k-го interval of the i processor module. Let us assume that the signal of persistence and self-preservation are equal in time. That is $T_{ss} = T_i(k) = \text{const}$.

T_{ef}: The total running time of the task, provided that there are no errors. The uptime is not included, for example, test sequences, recovery unit generation, etc.

T_{su}: Recovery unit generation time.

Because the total number of calls between modules i is j equal to

$$a_{ij}\left[T_{ef}/\sum_{m=1}^{N}a_{im}e_{im}\right] \text{ and}$$

$$\sum_{k=1}^{N}u_{ijk}(T_{ss}-T_{su}),$$

where: u_{ijk} The average intensity of the processor module i to the module j during k - ro the module interval i.

Suppose that the messages obey Poisson's law of distribution of the probability of a sequence of appeals. Given that the number of states of the processor module is large, it u_{ijk} can be considered constant during the k - ro interval, we obtain the following relation:

The maximum value of the intensity of memory u_{ijk} accesses must be less than or equal to the inverse value e_j, that is:

$$\frac{1}{e_{ij}} \geq (u_{ijk})_{\max} \geq u_{ijk} \geq 0$$

where E : Matrix $[e_{ij}]i, j = 1,2,...,M$, and $e_{i,j}$ represents the average execution time of calls from module i to module j.

Function f_{ij} – a monotonically increasing function from a bounded concave function of an argument g_{ij} has the maximum value when $u_{ij1}=u_{ij2}=\ldots=u_{ijNt}$ and the minimum value $\left\lfloor f_{ij}^{N}\right\rfloor$, when there are h intervals.

$$h=\frac{e_{ij}T_{ef}a_{ij}}{\left[(T_{ss}-T_{su})\sum_{m=1}^{M}a_{im}e_{im}\right]}$$

where $u_{ijk}=1/e_{ij}$, when (N_t-h-1) intervals and u_{ijk} =0 when one interval,

$$u_{ijk}=\left\{\frac{T_{ef}a_{ij}}{(T_{ss}-T_{su})\sum_{m=1}^{M}a_{im}e_{im}}\right\}-\frac{h}{e_{ij}}$$

The value f_{ij} can be considered as the probability of a direct connection between nodes i and j. To determine the probability of restarting r_{ij}: in the processor module j, we use calculations from the theory of network reliability:

$$r_{ij}=\bigcup_{q}(D_{ij,q})$$

where $D_{ij,q}$ represent the probability of what is q - way out of the node i to node j and $\bigcup$ - probabilistic operation unification. Let us introduce an additional proposition that the occurrence of a fault in the static sense is uniformly distributed over the entire set of modules. Then the range of one-step recovery is determined as follows:

$$C(1)=(1/M)\sum_{i=1}^{M}\prod_{j=1}^{M}\left[1-r_{ij}\left(1-\sum_{k=1}^{M}b_{jk}\right)\right]$$

Calculations show that it is enough to have a small number of cells for registering states to achieve a satisfactory recovery result through restarting the task processes. Reducing access to resources leads to an increase in the memory value of the valid flags.

Architecture of a fault-tolerant neuromultimicrocomputer with a high-speed radio exchange channel for processing multidimensional radar signals

The optimal choice of architecture is ensured by its maximum approximation to the class of problems to be solved. Digital radar signal processing refers to the processing of signals that can be represented as a sequence of multidimensional arrays of numbers, such as sampling signals continuously varying over time from multiple sensors. Since field processing tasks are distinguished by a large amount of information that needs to be processed in real time, it is advisable to develop neural network ensembles in the form of a set of functional modules aimed at solving them.

Ensembles perform computations on data and interact with other ensembles to generate computations on distributed data. A multimicroprocessor ensemble organization defines an adaptive organization and distribution of functions for control, computations, data transfer and restructuring of a neural network in the process of a reliable solution of a given

task and ensuring the necessary fault tolerance. Information exchange between ensembles is provided by a wireless communication channel. We introduce the central processing unit as a system unit for testing, diagnosing and ensuring the initial startup of the neuromicrocomputer.

When building systems with an inter-ensemble exchange radio channel, we will use the system backbone for high-performance systems with advanced functionality associated with the presence of additional address spaces - message space and interconnection space. Using the architecture of a neural network with a radio channel and removing system bus operations from the central processor makes its processor independent.

The traditional way of communicating and transferring has been to use a shared memory space when using two or a maximum of three ensembles on the backbone, but it is not efficient for a larger number. As the number of neural network components grows, the time required to access the data increases unacceptably. More efficient is the exchange of data through the message space (see Fig. 1). Just as mail decouples sender and receiver, so processors are decoupled from the task of passing messages. To transfer a message, the sending ensembles prepare the message in a local buffer and indicate to the communication processor the transfer address. If the recipient is ready to accept the message, he gives his consent to the transfer of data. The coprocessors then perform the actual transfer and inform their processing units that the transfer is complete. This is the ability to implement a standard network protocol for data transmission and work within one neurosystem to various operating systems.

A neurocomputer can use a virtual interrupt scheme: an interrupt is carried out not by physical interrupt signals, but by transmitting a special interrupt message. A virtual interrupt is a message that contains the interrupt source address, destination, and qualified information.

The interconnection space allows for easy system configuration, simplifies unit testing and system reconfiguration. When a faulty module is found, the latter can be programmatically removed from the structure of the neuromultimicroprocessor and replaced with another one in hot standby. Diagnostic software is located in each module. Each module can perform a built-in self-test. This operation can be carried out over the network from a remote terminal.

The transmission of information in the message space is transmitted continuously and the channel is not blocked. Thus, real-time mode is provided, information "freezing" during the exchange of ensembles modules is excluded. Messages are transmitted in quanta - data packets by means of a communication processor through the message space used to identify, configure and test the board. Each of the neurocomputer processors runs under its own operating system. To solve the problem of testing, initialization and initial loading of the system, neurocomputer architecture with a system bus has been developed, the physical transmission medium of which is a radio channel. The order of starting, testing and loading the system has been determined. Thus, openness, flexibility, and deep systemic development allow using the radio channel as a system bus for building highly reliable fault-tolerant neural network systems of high performance.

Neural network architecture is optimal for solving loosely coupled problems with natural parallelization. The tightly coupled central service module uses the radio channel at the bus-resident level, significantly increasing the bus bandwidth and ensemble performance as a whole and implements the following functions: system initialization at power-on, power supply control and switching to a backup source, timeout control. It can isolate failed modules, allowing other modules to continue to function normally. The functions of the central service module can be taken over by any other module in the event of its failure, without impairing the reliability of the neurocomputer.

The architecture of the real-time neurocomputer ensemble belongs to the systems of the MIMD type [40] with distributed memory and consists of a plurality of processors that autonomously execute various instructions on different data, i.e. are asynchronous systems with decentralized control.

The Central Processing Unit (CPU) architecture provides parallelism at the level of individual instructions, the level of loops and

iterations, the level of subroutines, the level of job steps, and the level of independent jobs and programs. Independent processor nodes provide parallelism at the level of individual instructions, but the efficiently of parallelization at the levels of loops and subroutines will already depend on the speed of the communication structure connecting the processor nodes. As for the levels of steps of a task and independent tasks and programs, they are usually associated with the multitasking mode of the system and when mapping tasks to individual processors or processor ensembles should not impose special requirements on the speed of data exchange between processors.

In a split-job system, supervisory functions are performed by each processor in accordance with its own needs and the requirements of the programs executed by that processor. Since the supervisor modules are executed by multiple processors, we provide for their re-entry or load copies of them into each processor. The number of conflicts associated with locking system tables is small, since each processor can have its own set. At the same time, the number of common control tables will not be large.

Systems with separate execution of tasks in each processor impose certain limiting requirements on the type of initial information, because systems work efficiently only when the tasks solved by individual processors of the system are well balanced, that is, they use the equipment approximately equally effectively. From the point of view of reliability, all processors in the system are a bottleneck, because failure of any processor means the loss of its program and violation of all program exchanges in which this processor participates. Restoring the system to work requires long-term external intervention. Extending the system without changing programs is impossible.

Systems with symmetric, or homogeneous, processing in all processors are most fully implemented when using a set of functionally homogeneous processor units. Each of the processors can equally effectively perform supervisory functions that "flow" from one processor to another and perform those supervisory functions that are inextricably linked to the problem being solved, and those functions that are necessary for a new task, in the case when the current one is interrupted or completed completely. However, any processor can perform all or most of the system-wide functions. Due to the fact that the processors are homogeneous and can be used in the same way, any task during its execution can be processed by different processor units of the system. We use different sets of processors for its successful implementation. System-wide control is continuously redistributed between processors: at a time, only one processor can be the control one; a certain priority can be set for the processors, firstly, to resolve conflicts and, secondly, to rank control functions.

The neurocomputer does not impose strict requirements on the nature of the input information. When processor modules fail, performance gradually decreases (system degradation). The expansion of the system is possible without any functional limitations.

The peculiarity of the architecture of the neurocomputer lies in the combination of ensembles by means of a high-speed system highway with a physical transmission medium over a radio channel. A fast and efficient messaging service is implemented using a specialized operating system with a distributed kernel. The program being executed is represented as a set of simultaneously running processes that exchange data and synchronize their work by sending messages [41]. In other words, the program is viewed as a network of processes. The network consists of logical nodes, each of which contains a subset of processes that, from the point of view of the programmer, should run together in one physical node. The radio channel is divided between network subscribers in time, providing multiple accesses to the channel. There are no conflicts through the use of a code modulation radio channel. You can neglect the transit time of the signal through the radio channel. The concept of a network category as a short-range or long-range network loses its meaning. As the main characteristics when assessing the quality, we use the average message delay and the channel capacity (or the channel utilization factor, which is defined as the part of the channel capacity attributable to conflict-free transmission). A common way to match network traffic to incoming user requests is through flow control procedures.

Approbation of the technology for processing radar information was carried out in the SPE "Quantor" laboratory using an ensemble in the form of a multi-microprocessor for collecting radar information from multidimensional sensors in the THz range (75-115 GHz). Spatial configuration is provided by base stations in the 40 GHz range. The research was carried out for the study of hidden prohibited items, the study of the structure of the coatings of special devices, receiving through wall 3D-Image.

The possibility of 3D-radar calibration is studied in the exploring of material properties to the example of Plexiglas, depending on the distance between the sample and the antenna using an absorber. The results of preliminary studies indicate the possibility of measuring the thickness of the material.

In the implementation of 3D scanning small objects is used FMCW radar at operating frequency 100 GHz and bandwidth about 40 GHz of terahertz frequency range, Fig. 2.

We have developed algorithms and have obtained the required accuracy - less than 3 mm. In reality, we can accurately assess the environment model to take it into account in processing. For it we previously will try to calibrate the radar.

On the calibration, a small metal plate and several measurement cycles for averaging the noise were used. It is shown that the accuracy of measurements is influenced by the width of the radiation pattern, the number of measurement cycles at one point, the accuracy of positioning and moving the head during the measurements, and the time interval between the calibrations.

As a result of the measurement cycle, a frequency dependence of the attenuation in the microwave channel $D(f) = U_{ref}(f)/U_{inc}(f)$ was obtained.

Unknown parameters of the dielectric structure are determined by procedure of global minimization of discrepancy between the measured attenuation in channel $D(f)$ and one calculated theoretically $D_{th}(f, \mathrm{p})$

$$F(\mathbf{p}) = \sum_{f} \left| D(f) - D_{th}(f, \mathbf{p}) \right|^2$$

Here $D_{th}(f, \mathrm{p})$ is defined according to the formula

$$D_{th} = \left| k_0 + k_1 \frac{V - V_c}{(1 - k_3 V)(1 - k_3 V_c) - k_2 V V_c} \right|^2$$

and $k_0(f), .., k_3(f)$ are complex coefficients, which are determined experimentally using reference samples and describe properties of the microwave channel; f is the frequency of sounding waves; $V_C(f)$ is the complex reflection coefficient (CRC) of the reference arm 3; $V(f, p)$ is a theoretically calculated CRC of the dielectric structure, which depends on a vector of the structure parameters p (thickness of layers and electrical parameters of materials).

We consider that in free space extends a plane electromagnetic wave and normally in-

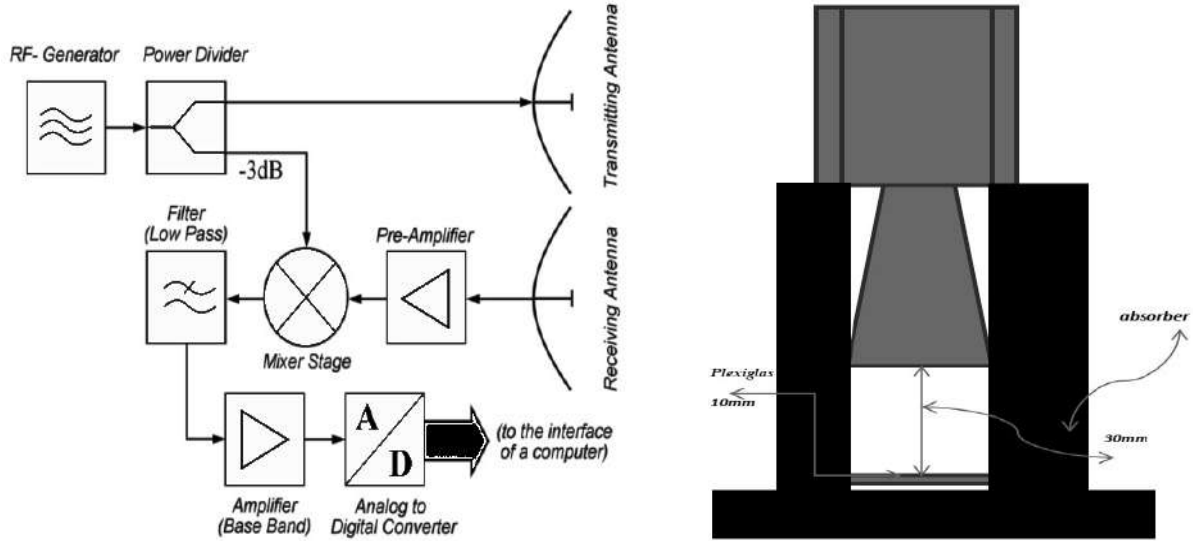

Fig.2. System for testing 3D FMCW Terahertz Radar

cident on the infinite (M-1)-layer medium with flat boundaries. The CRC V (f, p) is related of the CRC of the structure in free space V_S (f, p) through the scattering matrix of the antenna **S**, which is determined experimentally:

$$V = S_{11} + \frac{S_{21}V_S}{1 - S_{22}V_S}$$

The CRC of the structure in free space depends on the thickness and electrophysical parameters of structure layers:

$V_S = V_S(f, h1 \ldots hm_{-1}, \varepsilon 1 \ldots \varepsilon_m, \mathrm{tg}\delta_1 \ldots \mathrm{tg}\delta m)$,

where h_m, ε_m, $\mathrm{tg}\delta_m$ is thickness, permittivity and loss tangent of m-th layer. The CRC of the plane wave from dielectric plane-layered medium $V_S(f,p)$ is determined by the known formulas:

$$V_S = \frac{W_0 - Y_1}{W_0 + Y_1}$$

We see an Average Basic Function (BF) of 40 response signal from 6x6mm metal at 40 different distance — estimation of Non Removable response from constructive elements (horn and others), and Average BF of 40 response signal from Absorber Only without metal plane. [2] We can see a small difference between Absorber Only Average BF and Calibration (by metal) Average BF (Fig.3).

Step x 104 Average BF and BF Series after 0 compensation. SPC "Quantor", Ukraine

We have developed algorithms and have obtained the required accuracy - less than 3 mm. But in reality, we cannot accurately assess the environment model to take it into account in processing. For it we will test the radar system, having previously calibrated it.

Conclusions

This article discusses the architecture of intelligent fault-tolerant radar systems based on a neurocomputer. Fault tolerance is provided by varying the ratio of performance and reliability with a shortage of reliability resources of a real-time neural network. We briefly got acquainted with resource management, special attention was paid to the impact of fault tolerance on the reliability resource and the ability to manage it when solving an applied problem. The discussion covered the issues of building a hypothetical model of a real-time multiprocessor with a resource of performance and reliability, as well as their relationship. The prototype of the original neurocomputer operating system was the real-time operating system "RMX-86". The architecture of this real-time multiprocessor system is determined by the applied task of processing radar information. The elements of fault tolerance and survivability were introduced into the system, initial and diagnostic test support during the execution of an applied task, control of the system backbone, and majorization.

When forming an article on neural network systems for processing radar infor-

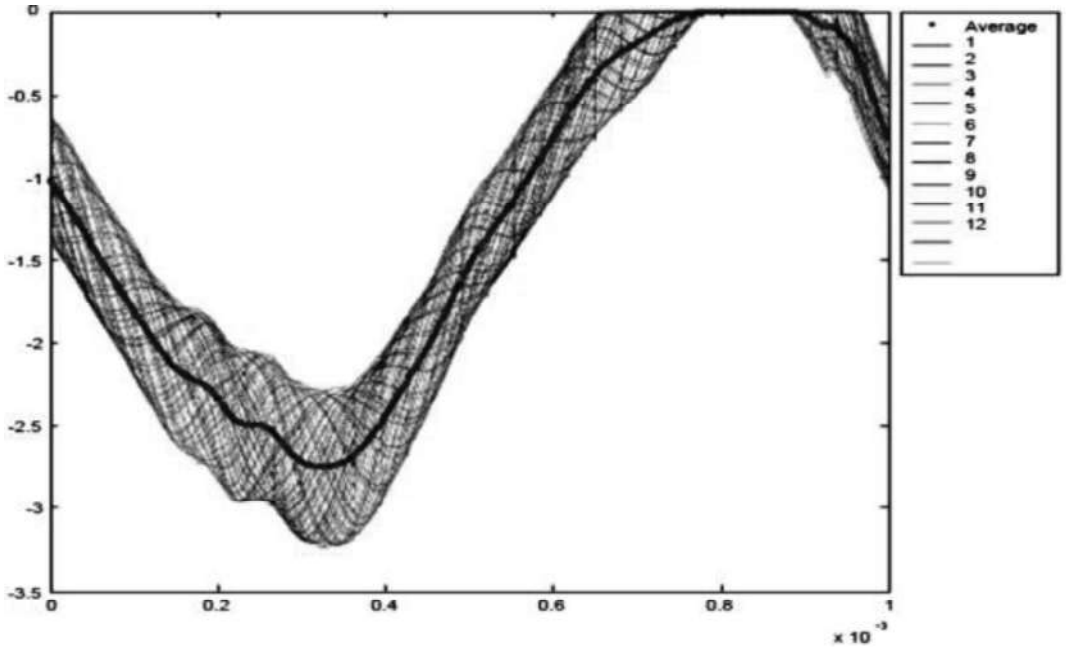

Fig.3 Comparing and Average BF and Series of BF

mation, the number one task was to highlight the principle of organizing the computational process. Unfortunately, the volume of the article did not allow highlighting the issues about the basis of the organization of macro-pipeline calculations of a fault-tolerant neurocomputer, the description of abstract system resources and the management of the reliability resource of a fault-tolerant real-time neurocomputer, the organization of the computational process for solving an applied problem of multidimensional radar information. Also, the issues of building a modeling complex for solving radar problems on neural network structures with deep learning are not covered. In the future, special attention will be paid to the tools for debugging a neurocomputer, since the debugging tools of a multiprocessor are difficult to design, and debugging a neurocomputer requires the design of original tools that are not supplied in a finished form by manufacturers of conventional processor components.

Collection, processing, technical implementation features, testing, diagnostics, calibration of systems based on neural networks will be covered later. They do not affect the general understanding of the organization of computations using neural networks, although the difference is significant in comparison with conventional calculators. In our last works, the elements of construction of THz radars, LPI of marine radars, construction of trained neural radar networks are touched upon. [42] Modeling of processes in a neural network showed the nonlinear nature of the system's behavior from the influence of external and internal disturbances of various powers, information in a neural network is often heuristic. The reliability of computations is ensured by the fault tolerance of neural networks and depends on the reliability resource.

Biomimetic methods are gaining popularity in the construction of radar systems and one of its important characteristics. That is, the external environment affects the operation of the radar system and, in turn, the radar system generates a sounding signal, takes into account the indicators of the environment. Most clearly it sees when using radar in an IoT environment. The second important biomimetic indicator is "reticular function". And if early cognition was realized through adaptability, albeit to an incomplete extent, then the reticular function was realized through the fault tolerance of multiprocessors, but in practice it was not implemented at all, due to the high cost of design. For the first time, we applied reliability improvement by fault-tolerant methods in control systems, collection, processing and display of information on board Ukrainian aircraft AN of the Antonov Design Bureau, where the author was the Chief Designer of the fault-tolerant multiprocessor system. When using neural network technologies, this function is modified and becomes more understandable, providing "homeostasis" of the neural network system for collecting and processing multidimensional information. An example of the use of cognition and a complete disregard for reticularity was demonstrated by the latest publications on research on radar technologies within the framework of NATO, where issues related to the fight against failures, solutions to the problem of fault tolerance, survivability, and dependability of computations were practically ignored. Neural network architecture assumes the solution of both problems as interconnected.

References

1. P. Barros, G.I. Parisi, C. Weber, S. Wermter, Emotion-modulated attention improves expression recognition: a deep learning model, Neurocomputing 253 (2017) 104–114. Learning Multimodal Data
2. Applications & Challenges of Deep Learning in the field of bioinformatics International Journal of Computer Science and Information Security (IJCSIS), Vol. 15, No. 7, July 2017
3. 2 [10] X. Chen et al., "Multi-view 3D object detection network for autonomous driving," in Proc. CVPR, 2017, pp. 6526–6534.
4. S. Ren et al., "Object detection networks on convolutional feature maps," IEEE Trans. Pattern Anal. Mach. Intel., vol. 39, no. 7, pp. 1476–1481, Jul. 2017.
5. Steen, K., Therkildsen, O., Green, O., Karstoft, H.: Detection of bird nests during mechanical weeding by incremental background

modelling and visual saliency. Sensors 15(3), 5096–5111 (2015)
6. Wu, X., Yuan, P., Peng, Q., Ngo, C., He, J.: Detection of bird nests in overhead catenaries system images for high-speed rail. Pattern Recogn. 51, 242–254 (2016)
7. Wang, Christopher Rasmussen (B), and Chunbo Song. Fast, Deep Detection and Tracking of Birds and Nests Qiaosong Department of Computer and Information Sciences, University of Delaware, Newark, DE, USA cer@cis.udel.edu Springer International Publishing AG 2016 G. Bebis et al. (Eds.): ISVC 2016, Part I, LNCS 10072, pp. 146–155, 2016. DOI: 10.1007/978-3-319-50835-1 14
8. Author J K. Classification of human cancer diseases by gene expression profiles. App Soft Comput. 2017; 124:134.
9. Bard E, Hu W. Identification of a 12 gene signature for lung cancer prognosis through machine learning. J of cancer. 2011; 148-156.
10. R. J. Cintra et al., "Low-complexity approximate convolutional neural networks," IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 12, pp. 5981–5992, 2018.
11. D. Tomè et al., "Deep convolutional neural networks for pedestrian detection," Signal Process., Image Commun., vol. 47, pp. 482–489, Sep. 2016.
12. J. Ngiam et al., "Multimodal deep learning," in Proc. ICML, 2011, pp. 689–696.
13. Z. Luo et al., "Non-local deep features for salient object detection," in Proc. CVPR, 2017, pp. 6593–6601.
14. Seonwoo Min, Byunghan Lee, SungrohYoon, Deep Learning in Bioinformatics‖ Briefings in Bioinformatics, Briefings in Bioinformatics, 2017; 18(5), , 851–869
15. J. Zhang, W. Li, P.O. Ogunbona, P. Wang, C. Tang, Rgb-d-based action recognition datasets: a survey, Pattern Recogn. 60 (2016) 86–105, doi: 10.1016/j.patcog. 2016.05.019.
16. Haohan Wang, Bhiksha Raj, and Eric P. Xing. On the origin of deep learning. CoRR, abs/1702.07800, 2017a.
17. Joel Moniz and Christopher J. Pal. Convolutional residual memory networks. CoRR, abs/1606.05262, 2016.
18. Chen CL, Mahjoubfar A, Tai L-C et al. Deep Learning in Label-free Cell Classification. Scientific reports 2016.
19. S. Sabour, N. Frosst, G. E. Hinton, Dynamic routing between capsules, 1945 in: NIPS 2017, 2017.
20. A. Diba, V. Sharma, A. Pazandeh, H. Pirsiavash, L. V. Gool, Weakly 1237 supervised cascaded convolutional networks, in: 2017 IEEE Conference 1238 on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 5131-1239 5139. doi:10.1109/CVPR.2017.545.
21. Z. Bylinskii, T. Judd, A. Borji, L. Itti, F. Durand, A. Oliva, A. Torralba, MIT Saliency Benchmark, 2015, (http://saliency.mit.edu).
22. Edgar Xi, Selina Bing, and Yang Jin. Capsule network performance on complex data. CoRR, abs/1712.03480v1, 2017. URL https://arxiv.org/abs/1712.03480v1.
23. Chunxiao Jiang, Haijun Zhang, Yong Ren, Zhu Han, Kwang-Cheng Chen, and Lajos Hanzo. Machine learning paradigms for nextgeneration wireless networks. IEEE Wireless Communications, 24(2):98–105, 2017.
24. Xenofon Foukas, Georgios Patounas, Ahmed Elmokashfi, and Mahesh K Marina. Network slicing in 5G: Survey and challenges. IEEE Communications Magazine, 55(5):94–100, 2017.
25. U. Iqbal, A. Milan, J. Gall, Posetrack: joint multi-person pose estimation and tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pat- tern Recognition (CVPR), 2017.
26. D. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, Proceedings of the 3rd International Conference on Learning Representations, 2015.
27. J. Zhang, W. Li, P.O. Ogunbona, P. Wang, C. Tang, RGB-based action recognition datasets: a survey, Pattern Recogn. 60 (2016) 86–105, doi: 10.1016/j.patcog. 2016.05.019.
28. Ivan Garvanov, Lyubka Doukovska, Vladimir Kyovtorov, Christo Kabakchiev. COMPARATIVE ANALYSIS OF CFAR STRUCTURES FOR GPS SIGNALS IN CONDITIONS OF INTENSIVE URBAN PULSE INTERFERENCE Institute of Information Technologies Bulgarian Academy of Sciences.
29. Z. Cao et al., "Realtime multi-person 2D pose estimation using part affinity fields," in Proc. CVPR, 2017, pp. 1302–1310.
30. Bewley A, Ge Z, Ott L, Ramos F, Upcroft B. Simple online and realtime tracking. In: 2016

IEEE International Conference on Image Processing (ICIP). IEEE; 2016. pp. 3464-3468
31. Wojke N, Bewley A, Paulus D. Simple online and realtime tracking with a deep association metric. In: 2017
32. IEEE International Conference on Image Processing (ICIP). IEEE; 2017. pp. 3645-3649
33. Sercan ¨Omer Arik, Mike Chrzanowski, Adam Coates, Greg Diamos, Andrew Gibiansky, Yongguo Kang, Xian Li, John Miller, Jonathan Raiman, Shubho Sengupta, and Mohammad Shoeybi. Deep voice: Real-time neural text-to-speech. CoRR, abs/1702.07825, 2017.
34. S. H. Khan et al., "Cost-sensitive learning of deep feature representations from imbalanced data," IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 8, pp. 3573–3587, Aug. 2018.
35. Khan G, Ghani MU, Siddiqi A, Seo S, Baik SW, Mehmood I, et al. Egocentric visual scene description based on human-object interaction and deep spatial relations among objects. Multimedia Tools and Applications. Vol. 77. Springer; 2018. pp. 1-22
36. Thenmozhi K, Shanthi S. Optimized Data Retrieval in Big Data Environment using PPFC Approach AJRSSH. 2017; 683-690.
37. C. Feng, M. Cui, B.-M. Hodge, J. Zhang, A data-driven multi-model 1914 methodology with deep feature selection for short-term wind forecasting, 1915 Applied Energy 190 (2017) 1245{1257.
38. S. Bouktif, A. Fiaz, A. Ouni, M. Serhani, Optimal deep learning lstm 1855 model for electric load forecasting using feature selection and genetic al1 856 algorithm: Comparison with machine learning approaches, Energies 11 (7) 1857 (2018) 1636.
39. Avizhenis A. Fault tolerance is a property that ensures the continuous performance of digital systems. / IEEE - 1978. – Vol.66 – №10 - p.5-15.
40. Flynn M.J., Some computer organizations and their effectiveness // IEEE Trans. Comput. - 1972. - Vol. C-21, No. 9. - pp. 948-960.
41. Palagin A.V. About computers with virtual architecture// CSM. –1999.-№3.-p.33-43.
42. *Kosovets M., Tovstenko L.* The practical aspect of using the artificial intellectual technology for building a multidimensional function CFAR for smart-handled LPI Radar. Magazine "Programming Problems". ISSN 1727-4907. 2020. №2-3. Special issue. Proceedings of the thirteenth international scientific-practical conference on PROGRAMMING. UkrPROG'2020. September 16-17, 2020 Kyiv, Ukraine. Pages 202 - 212. http://www.pp.isofts.kiev.ua.

Received: 20.05.2021

About the authors:

Mykola Kosovets
Leading Constructor,
Number of scientific publications in Ukrainian publications -53
Number of scientific publications in foreign publications -14
Index Hirsha -2
https://orcid.org/0000-0001- 8443-7805
Scopus Author ID: 5644007500

Lilia Tovstenko
Leading Software Engineer
Number of scientific publications in Ukrainian publications -21
Number of scientific publications in foreign lands -6
Index Hirsha -2
https://orcid.org/0000-0002-3348-6065
Scopus Author ID: 56439972800

Affiliations:

SPE "Quantor"
03057, c. Kyiv-57, str. E.Potye, 8-A
Ph.: (380)66-2554143
E-mail: quantor.nik@gmail.com

Institute of Cybernetics of Glushkov
National Academy of the National Academy Sciences Ukraine
03187, Kyiv-187, Academician Glushkov Avenue, 40.
Ph.: (380)67-7774010
E-mail: 115lili@incyb.kiev.ua