# ALGORITHMIZATION OF THE SOFTWARE TESTING SYSTEM BASED ON FINITE AUTOMATA

*M. M. Zozulya(1), O. Ja. Kravets(2)*, I. V. Atlasov(3), I. A. Aksenov(4), L. M. Bozhko(5), P. A. Rahman(6)*

[1]Military Training and Research Center of the Air Force "Professor N. E. Zhukovsky and Yu.A. Gagarin Air Force Academy", Voronezh; [2]Voronezh state technical university, Voronezh; [3]Moscow University of Ministry of Internal Affairs of Russian Federation named by V.Ja. Kikot, Moscow; [4]Vladimir State University named after Alexander and Nikolay Stoletovs, Vladimir; [5]Emperor Alexander I St. Petersburg State Transport University, Saint Petersburg; [6]Ufa State Petroleum Technological University
Russian Federation

[*] Corresponding Author, e-mail: csit@bk.ru

**Abstract:** Software testing is one of the key stages in the development of a digital device based on a microcontroller. The article discusses the features of the implementation of algorithms for the functioning of the microcontroller software testing system based on finite automata. An enlarged block diagram of a testing system at the software and hardware levels is presented, a model of such a system using the theory of finite automata is presented. A detailed finite automaton for the testing process model has been formed, its states have been determined and algorithms for operation are given.

**Key words:** software testing, digital device, finite automata, block diagram, testing process model.

## 1. INTRODUCTION

Software testing is one of the key stages in the development of a digital device based on a microcontroller (MC) [1]. If, until recently, software development was conducted in assembly language and required deep professional competencies from the developer, at present, in the process of developing microprocessor technologies on the one hand and special software on the other hand, software development for MC increasingly falls into the field of visibility of application software programmers. A number of technical limitations lead to the fact that the usual

approaches to the development and testing of software for a personal computer cannot be applied when creating programs for microcontrollers [2].

If in the field of development tools for microcontrollers of various manufacturers there have been trends towards their unification, then in the field of testing, the issue of creating a universal testing system for special software (SPO) is still acute.

## 2. THE PROBLEM FEATURES

In the article, we will consider a possible algorithm for the functioning of a microcontroller's SPO testing system without using an operational system.

At the same time, we detail the main data structures, the method of specifying the source data and obtaining output data. By the initial data, we will understand the parameters of the target microcontroller for which the SPO is being created and tested; the set of interfaces of this MC used by the software.

As output data, we mean a structured report on the results of the SPO test.

Features of storing and creating a database; algorithmic structures that collect source data; software development tools and its loading into the MC will be presented in general form.

## 3. STRUCTURE OF THE TESTING SYSTEM

The software part of the testing system consists of two blocks, six modules and three objects for storing heterogeneous information. The blocks are allocated by functional affiliation (Figure 1).

The block of work with the source data is represented by two modules that are physically executed in different memory areas: the primary information input module is a computer application (computer level), the initialization module is a series of low-level functions for reading the unique code of the target microcontroller and transferring it to the control microcontroller, which is a structural element of the system of testing the software at the hardware level (MC level).

The primary information input module collects information about the parameters of a particular microcontroller: clock frequency, memory volume, availability of interfaces, physical location of interface pins, manufacturer, architecture bit depth, etc. Data is entered by the operator. The operation of this unit precedes the physical prototyping of the tested node of the digital device. The data collected by this module is recorded in the object configuration storage, which is physically a database or a structured text file. The hardware and software structure of the testing system is shown in Figure 2.

The figure shows that the testing system at the hardware level [3], as well as at the software level, is located at two levels – the microcontroller level and the PC level.

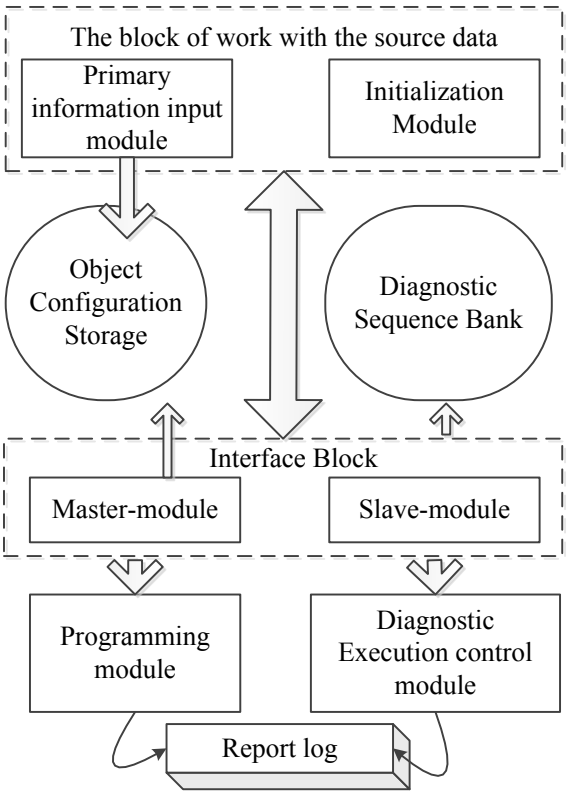


*Figure 1. Block diagram of the testing system*

At the level of a personal computer, the following program modules are located, shown in Figure 1: the primary information input module (its functionality is discussed above), the programming module (represented by a set of scripts in TCL, calling the standard means of writing the control program to the MC) and the Master module (described below).

The MC level contains the following software modules: the initialization module (discussed above), the diagnostic control module and the Slave module. This division is very conditional, because they are all part of one "application" working as part of the control microcontroller software.

## 4. FINITE-AUTOMATON MODEL OF THE TESTING SYSTEM

The program instructions for the microcontroller are executed sequentially, one after the other. However, for the correct implementation of the interaction of the MC with the PC, it is necessary to perform time-critical tasks in the first place, and secondary tasks in the second. At the operating system level, this is implemented by the task scheduler using, for example, the semaphore mechanism. Despite the fact that modern MC have a sufficient amount of program memory in their composition,

sufficient for using a real-time OS (for example, Free RT OS), abandoning it gives a number of advantages, the main of which is full control over the resources of the MC.

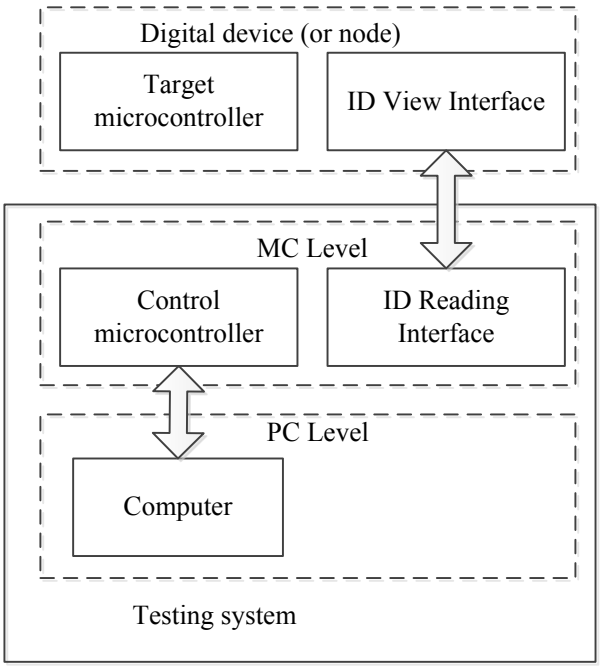


*Figure 2. The hardware and software structure of the testing system*

Another approach to the organization of the MC program is the use of a system state model (as, for example, implemented in [4]). Figure 3 shows the finite state machine of the SPO testing system. Each state characterizes the tasks performed at both levels of the testing system. The transition between the states [5] can be carried out both by external influence (operator actions or mutual sending of messages between levels), and by internal - for example, working off a timer, the end of data reception, etc. At least seven states can be distinguished (Figure 3):

1. **Waiting.** In this state, the software of both the MC and the PC is waiting for the operator's action. The initialization module represented by the readdata(void) function in the body of the overflow interrupt reads the code of the target MC. If the operator installs the target microcontroller into the corresponding hardware connector, the function automatically switches the system to state 4. The read-Command(void) function reads the message from the USART receiver-transmitter, which is hardware-coupled to the PC. If the "lookUp" message is received, the system switches to state 2.

2. **Data exchange.** In this state, messages are exchanged between the MC and the PC. If the initiator of the exchange is a PC, then the MC returns an OK response and the system switches to the state according to Table 1.

*Table 1. PC command system*

| **PC request** | **Transition to the state** |
|---|---|
| connectToMe | 3 |
| loadSoftware | 5 |
| assemblyDevice | 4 |
| failureRead | 6 |
| submitReport | 7 |

Data transfer to the PC is initiated by the MC in case of return from states 3-7 and is possible only if the *readCommand(void)* function receives the "OK" message. In this case, the data stored in the transmission stack is transmitted in the format defined for the corresponding state.

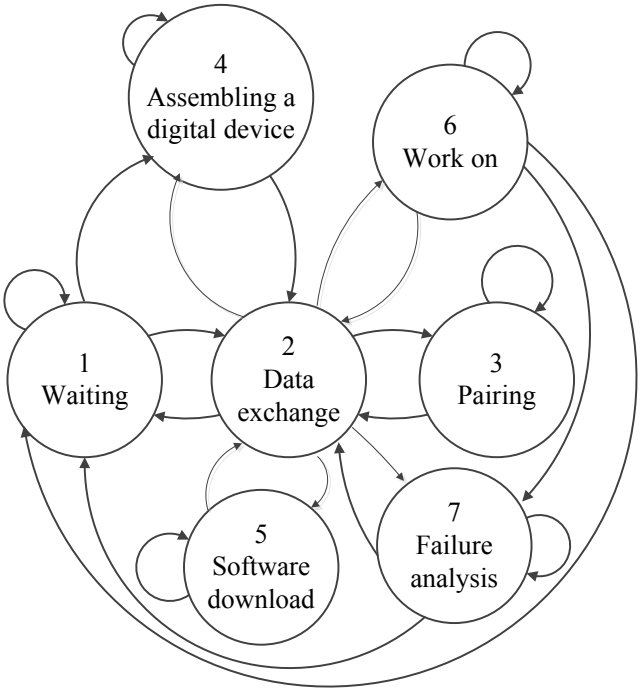


*Figure 3. Finite state automata of the testing system*

3. **Pairing.** In this state, the control microcontroller transmits to the PC the unique identifier of the target microcontroller MCU_ID, read in state 1. MCU_ID is an integer in the range 0.65537, which uniquely determines the parameters of the target microcontroller (set in the primary information input module).

4. **Assembling a digital device.** The state of the system in which the operator physically connects to the interface outputs of the target microcontroller peripheral devices forming the projected digital device (digital device node).

In this state (Figure 4), the MC waits for a message from the operator via the general-purpose I/O port (GPIO) about the connected interface and writes it to the array of I/O port states of the target MC *arrayOfStateGPIO*. The array is a matrix N*4, where N is the number of peripheral devices (N>0) connected to the target MC. The four columns of the matrix contain hexadecimal binary numbers – the states of ports A, B, C and D, where 0 in the discharge indicates the free (unconnected) state of the corresponding port output. At the same time, the PC records the fact of the transition to state 4 and waits for its end. At the end of the assembly of the designed digital device, the message "assembly OK" is sent to the control MC. After that, the MC writes the generated array to the transmission stack, and the system switches to state 2, and the PC sends a request to read the transmission stack of the control microcontroller.

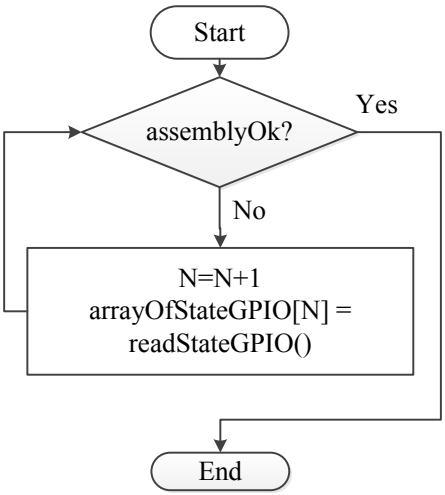


*Figure 4. Block diagram of the digital device assembly algorithm*

5. **Software download.** The transition to this state means that the PC is ready to transfer the tested SOFTWARE to the target MC. The transfer is carried out through the external software and hardware (programmers), which are launched using a set of scripts in the TCL language. In this state, the MC waits for the end of the PDF loading process, which is indicated by the message "load Software OK" transmitted by the PC. Receiving this message leads to setting the flag (in the memory of the control MC) of loading the PDF into the target MC.

6. **Work on.** The transition to this state leads to loading into the control MC an array of *arrayOfFailure* service messages transmitted from the PC. The specified array is formed at the stage of development of the SPO for the target MC.

The messages in the array are arranged in the order they are issued to the service port as the program under test is executed. Thus, the controlling MC, having switched to the current state and received an array of service messages, starts listening to the service port. In case of non-coincidence of the control and the received message, the new element of the array that caused the failure is recorded in the failures array.

7. **Failure analysis.** The transition to the current state initiates the transfer of the generated failures array to the PC, where its further analysis is carried out (Figure 5).

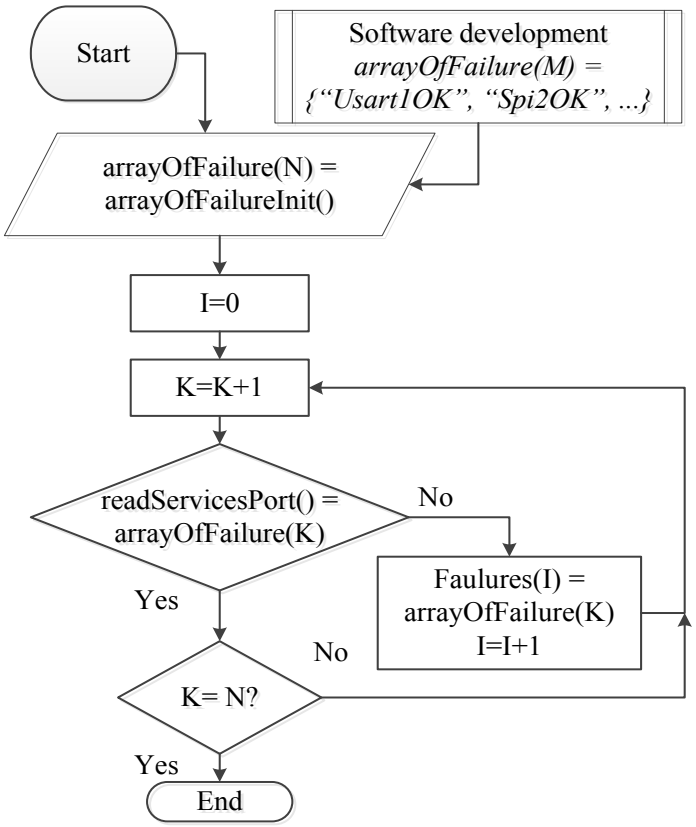


*Figure 5. Block diagram of the failure detection algorithm*

In accordance with the developed finite automaton [6] of the system states, the flowchart of the algorithm for the functioning of the SPO testing system can be presented in Figure 6.

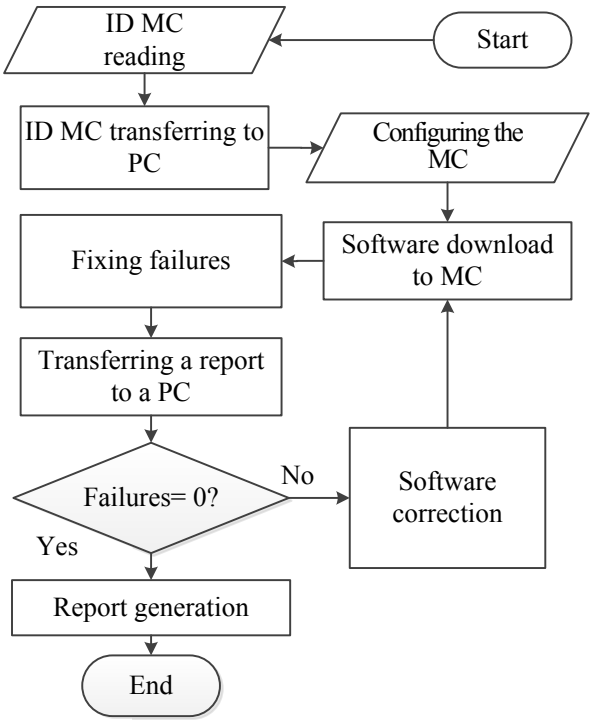


*Figure 6. Block diagram of the algorithm for the functioning of the SPO testing system*

## 5. FORMATION OF INITIAL DATA

Data on the target MC is collected in the primary information input module. General information about the target MC is entered (from the documentation for the microcontroller [7]): name, architecture, manufacturer; MC parameters: memory capacity, operating frequency, number of pins, etc. The physical outputs of the micro-controller are compared to the lines of a specific interface in accordance with the documentation on the MC. For example, for the Atmega8 MK, the SPI interface is located in accordance with Table 2.

*Table 2. Location of the Atmega8 MK SPI interface*

| Name of the line | Designation of the port and the serial number of the line |
|---|---|
| MOSI | B3 |
| MISO | B4 |
| SCK | B5 |
| SS | B2 |

The involved, i.e. assigned, port output corresponds to the required logical unit, and the unassigned one corresponds to zero. Thus, the state of port B for the Atmega8 MC SPI interface corresponds to the binary number 0x00111100, and the state of ports A, C, D is 0.

Information about the generated interface is entered in the table, the appearance of which is shown in the figure in the fifth area. Similarly, data is entered on all other interfaces of the target microcontroller. The specified information is then stored in the database and converted into a structured text file. A text file with the source data is loaded onto a memory card, which is connected to the control MC.

## 6. CONCLUSION

The programming industry has always been and remains very complex [8]. In the production of embedded software, problems such as errors in the code, deadlines, and the strong influence of the human factor increases by an order of magnitude. The increased responsibility of the developer to the customer, tight deadlines, and the inadmissibility of errors in the code make "embedded" programming a difficult task.

An enlarged block diagram of a testing system at the software and hardware levels is presented, a model of such a system using the theory of finite automata is presented. A detailed finite automaton for the testing process model has been formed, its states have been determined and algorithms for operation are given.

The use of the presented microcontroller software testing system can provide invaluable assistance to the developer.

## REFERENCES

[1] Neelesh J., Singh H., Hariom G. et al. Multilevel Test Method for Testing Microcontroller based ECG System. *International Journal of Computer Applications*, Vol. 46, No. 9, 2012, pp. 31-43.

[2] DeBenedictis E.P. Accelerated Architectures Create Programming Opportunities. *Computer*, Vol. 51, No. 6, 2018, pp. 82-85.

[3] Wang H., Sayadi H., Manor S.P.D. et al. Enabling Micro AI for Securing Edge Devices at Hardware Level. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, Vol. 99, No. 11, 2021, DOI: 10.1109/JETCAS.2021.3126816.

[4] Sarma M., Mall R. Synthesis of system state models. *ACM SIGPLAN Notices*, Vol. 42, No. 11, 2008, pp. 5-14.

[5] Priem F., Canters F. Modelling transitions in sealed surface cover fraction with Quantitative State Cellular Automata. *Landscape and Urban Planning*, Vol. 211, No. 1, 2021, pp. 104081, DOI: 10.1016/j.landurbplan.2021.104081.

[6] Zozulya M.M., Kravets O.Ja. Features of the architecture of the digital device diagnostics system. *Control Systems and Information Technologies*, vol. 86, No. 4, 2021, pp. 55-59.

[7] Cao Y., Zheng Y., Liu Y. et al. Design of programmable control electric current drive based on ATmega8. *Journal of Physics: Conference Series*, Vol. 1637, No. 1, 2020, pp. 012067, DOI: 10.1088/1742-6596/1637/1/012067.

[8] Ebert C. Embedded software. *Computer*, Vol. 45, No. 7, 2012, pp. 6-17.

### *Information about the authors:*

**Zozulya Mikhail Mikhailovich** - Researcher, Military Training and Research Center of the Air Force "Professor N. E. Zhukovsky and Yu.A. Gagarin Air Force Academy", areas of scientific research –simulation of complex objects

**Oleg Jakovlevich Kravets** – Dr. Sci (IT), professor of Voronezh state technical university, areas of scientific research – system analysis, optimization, simulation of complex objects

**Igor Viktorovich Atlasov** – Professor of Moscow University of Ministry of Internal Affairs of Russian Federation named by V.Ja. Kikot', areas of scientific research – system analysis, optimization, simulation of complex objects

**Ilia Antonovich Aksenov** – Associate professor of Vladimir State University named after Alexander and Nikolay Stoletovs, areas of scientific research – application of information technologies in economic systems

**Lesya Mikhailovna Bozhko** - Professor of Emperor Alexander I St. Petersburg State Transport University, areas of scientific research – innovation management, strategic management, organization management

**Pavel Azizurovich Rahman**, PhD, associate professor of Ufa State Petroleum Technological University, areas of scientific research - information technology, computer networks, reliability modeling and engineering

**Manuscript received on 25 January 2022**