Improvement of the electronic circuit for the breathing simulator

Nurzhan Duzbayev[1,†], Zhandaulet Musilimov[1,†], Lyudmila Kozina[1,*,†] and Manat Tuyenbayev[1,†]

[1]*International Information Technology University, 34/1 Manas St., Almaty, 050040, Kazakhstan*

Abstract

Due to the recent COVID-19 pandemic, a large number of people are currently still struggling with the effects of the virus on the body. The human respiratory system has been particularly negatively affected. In a number of methods used for the rehabilitation of such patients, doctors recommend breathing exercises. The use of IoT technologies in medicine improves the efficiency of processes and allows achieving better results. The article provides an overview of successful examples of IoMT (Internet of Medical Things) projects using biofeedback, including for working with the respiratory system. The article is devoted to the transformation of the scheme for the project "Development of a software and hardware complex for monitoring and correcting respiratory functions based on multimodule technologies", the purpose of which is to develop a modern breathing simulator, and describes in detail the changes made. As a result, an updated project scheme is presented.

Keywords

IoMT, breathing simulator, biofeedback, Arduino Nano, sensors, PCA9685, CCS811, MAX30102

1. Introduction

Even though the COVID-19 pandemic has officially ended, many people are still suffering from the consequences of the virus. The impact of the virus on various body systems is discussed in this study [1]. One of the body systems most affected by the virus is the human respiratory system. For example, this research [2] notes that the overall prevalence of post-COVID breathlessness is 26% and more. In addition, the deteriorating environmental situation that many large cities are currently facing also negatively affects the condition of the human respiratory system and aggravates the course of associated diseases. This aspect is discussed in detail in this study [3]. Taking into account all of the above, it is obvious that the medical community is faced with the task of restoring the health of such patients. In particular, breathing exercises demonstrate effectiveness in improving the condition of patients [4].

Various information technologies are becoming increasingly used in medicine. For example, the use of IoT technologies to solve medical problems has even received a special term "IoMT" (Internet of Medical Things). The potential of this direction is huge, which is proved by the interest of the scientific community expressed in the number and quality of scientific publications on this topic [5]. Different studies reflect positive examples of IoMT projects [6, 7]. A significant advantage of using IoT technologies is the opportunity to receive biofeedback. The obtained data and their subsequent analysis further expand the possibilities and positive effect of the technologies implementation. For example, a number of studies have proposed solutions for detecting diseases based on data obtained from monitoring the functioning of the respiratory system, such as observations of coughing,

DTESI 2024: 9th International Conference on Digital Technologies in Education, Science and Industry, October 16–17, 2024, Almaty, Kazakhstan

[1*] Corresponding author.

[†] These authors contributed equally.

n.duzbayev@iitu.edu.kz (N. Duzbayev); zhandauletmusilimovaltin@gmail.com (Zh. Musilimov); l.kozina@iitu.edu.kz (L. Kozina); manattuyenbayev@gmail.com (M. Tuyenbayev)

0000-0002-7989-9463 (N. Duzbayev); 0009-0006-8864-0089 (Zh. Musilimov); 0000-0002-0333-3404 (L. Kozina)

breathing, sneezing, speech behavior, and others, and the subsequent application of ML algorithms [8, 9, 10, 11], as well as for advanced monitoring of patients' conditions [12].

Within the framework of the project "Development of a software and hardware complex for monitoring and correcting respiratory functions based on multimodule technologies" a breathing simulator is being developed, including real-time monitoring of the patient's body parameters and wireless transmission of the received data to the doctor's device, which is designed to provide full control over the training process and thus improve the final results. Previously, the article [13] covered the development of the electronic circuit for this project, however during the work on the project the circuit was significantly improved, which in turn allows improving the rehabilitation process. This article provides a detailed description of the changes made during the improvement process and an explanation of the reasons for the need for these changes. As a result, the new and improved scheme is presented.

2. Breathing simulator scheme improvement

An important aspect of any device is its improvement. This can involve addressing various factors such as design, internal structure, and enhancement of existing functions. As current article will discuss the reasons and methods for improving the circuit design of a breathing device, to achieve this, a block diagram (Figure 1), which illustrates the connection system between the elements of the breathing device's circuit, should be examined.

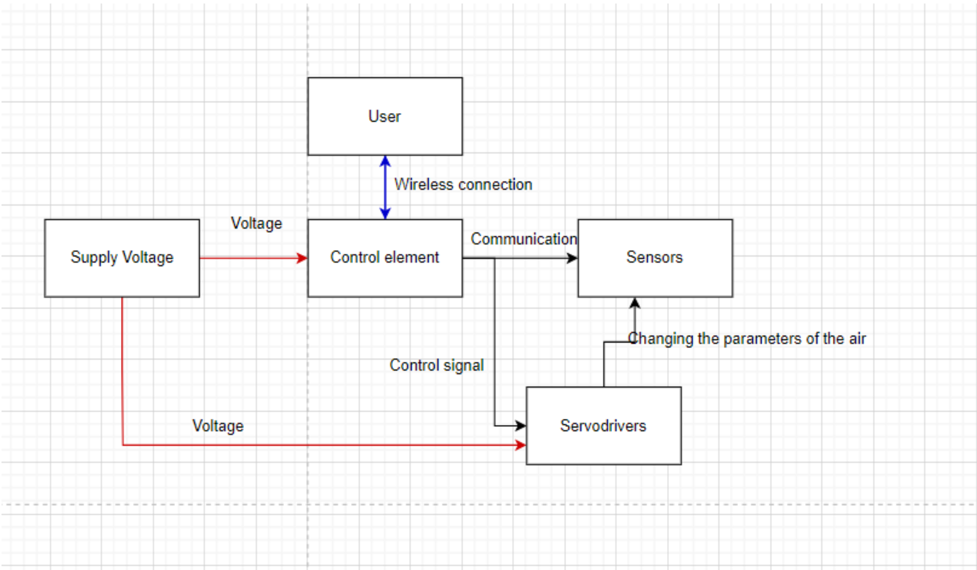


Figure 1: Breathing simulator working diagram.

The control element manages the sensors through specific communication protocols, sends control signals to set the movement and positions of servo drivers, and using wireless communication, sends and receives commands from the user. To begin the analysis, the operating algorithm of this system should be considered (Figure 2).

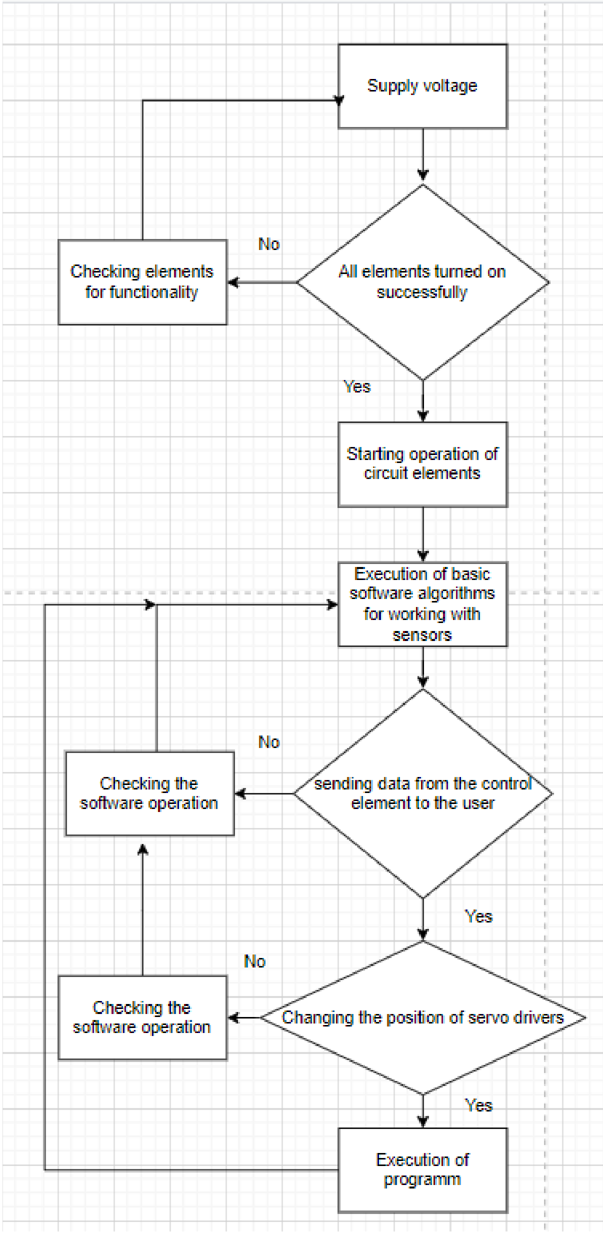


Figure 2: Algorithm block diagram.

When power is supplied, it's necessary to check if all components are functioning. For example, they may start to light up. If not, it's important to inspect the components themselves, as well as their connections. If they are working, the main tasks of the circuit can proceed. When data is transmitted, it may happen that the data isn't being sent. In that case, the software should be checked; if the data is being sent, the user might send a command to change the position of the servo driver. If all blocks are working correctly, the program should execute successfully. Otherwise, both the hardware and software should be inspected.

Now it is necessary to break down how these functions are implemented in the first circuit, review the test results, analyze them, and improve them if possible. Initially the block diagram of the first version of the project should be examined (Figure 3). It includes several functional blocks that can be improved by testing them and analyzing their role and tasks.

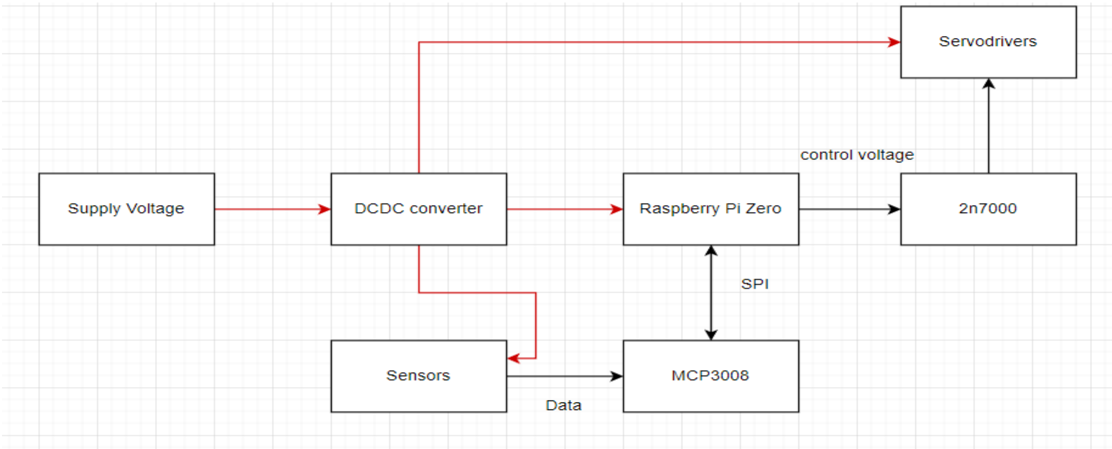


Figure 3: Block diagram of the elements of the first circuit.

2.1. Arduino NANO

In the block diagram in the Figure 3, the main control element is the Raspberry Pi Zero. This is a single-board computer based on the Linux system that can perform large calculations. Raspberry Pi Zero has a large number of inputs and outputs, which makes it a fix for complex projects. It has huge capabilities, an external processor, several ports such as USB, miniHDMI. Huge resources, computing power, as well as powerful support for the Linux system, were used little during testing. Having analyzed the role of Raspberry Pi Zero, it was concluded that the task of controlling element includes:

1. Receiving and processing information from sensors.
2. Sending information to the software developed for this study.
3. Receiving commands from the servo drive control application that the doctor will send.

It turns out that for the complete operation of the control unit, it's needed to work in real time and work with the periphery. All calculations will be done in software. Also, when working with Raspberry Pi Zero, it turned out that the energy consumption does not allow long-term operation on batteries for mobile devices, which is a breathing simulator. Microcontrollers are great for solving these problems. There are several types of microcontrollers such as STM32, ESP32, ESP8266, Arduino. Since it was decided to make this unit much simpler, STM32 is not suitable, due to the complexity of programming on it. ESP32 is known for having built-in Wi-Fi and Bluetooth in the boards, but for research there is no need for a network connection, and the huge energy consumption also interferes. The ESP8266 is the same. Arduino remains, among which there is an Arduino Nano microcontroller, which provides everything necessary for this study.

Arduino Nano (Figure 4) is a microcontroller based on ATmega328, which was developed for automation tasks. It works in the Arduino IDE programming environment. There are many libraries for working with various sensors, released by the developers of these sensors themselves.

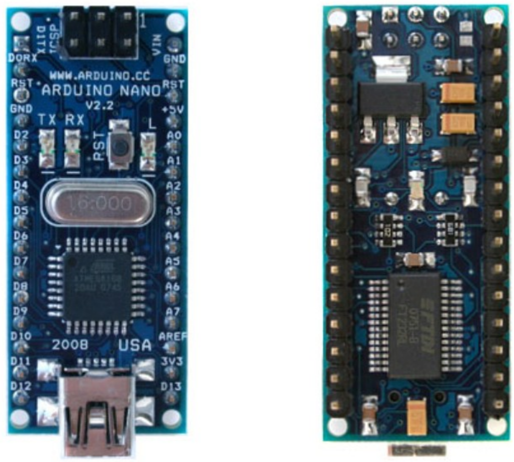


Figure 4: Arduino Nano [14].

Table 1 provides a comparative analysis of the characteristics of Raspberry Pi Zero and Arduino Nano.

Table 1
Comparison of Arduino Nano and Raspberry Pi Zero

Specification	Arduino Nano	Raspberry Pi Zero
Programming language	The programming language is based on C++	All programming languages compatible with Linux
CPU	ATmega328 (16 MHz, 8-bit)	Broadcom BCM2835 (1 GHz, 32-bit)
Memory	32 КБ Flash	32 Гб+
GPIO pins	14 digital, 6 analog	40 digital
Power consumption, operating voltage	19 mA at 5V	140 mA at 5V
Communication ports	UART, SPI, I2C	UART, SPI, I2C, USB, HDMI
Usage	For working with sensors and peripheral control	Full computer
Cost	2000 tenge	15 000 tenge
Areas of application	Automation	Projects that require a single board computer

Comparing these two elements, the following conclusion was reached. Arduino Nano is more suitable for solving simple tasks and tasks that require real-time operation, such as working with sensors. The Raspberry Pi Zero, in turn, is much more powerful than the Arduino Nano, but its enormous power will be superfluous in this study. Arduino Nano has a lower class than Rasbperry Pi Zero, but this Arduino Nano microcontroller was chosen for the reasons listed below:

1. Dimensions of the microcontroller allow the control part to be made much smaller than in the previous version.
2. Development on this microcontroller will be facilitated by the presence of a large number of libraries and examples for various tasks. The speed of program execution will be much higher, due to the properties of the C/C++ programming language.
3. The risk of an error is reduced due to frequent switching on because the microcontroller will execute the code with which it will be flashed. For this reason, the control unit will not be busy performing any other tasks.

4. The role of the control unit is reduced so that the computing part remains in the software. The only tasks of the control unit remain receiving data, sending data, receiving commands from the user.

The control element works with servo drivers. However, there are 8 of them, and some solutions are needed to control them. In the first version, such a solution was 2n7000 transistors due to their simplicity of operation. They work in such a way that when a signal is sent to the control electrode, the energy passes through the transistor and goes to the servo driver. And here several problems appear at once:

1. Inaccurate positioning of servo drivers due to the use of a transistor.
2. Also, the control signal was generated by the control unit, which took up resources.
3. To control 8 servo drives, many such transistors were needed.

After analyzing these results, it was decided to use a PCA9685 port expander instead of transistors.

2.2. PCA9685

The PCA9685 is a PWM controller for controlling up to 16 independent outputs with I2C interface. It is presented in Figure 5 [15].

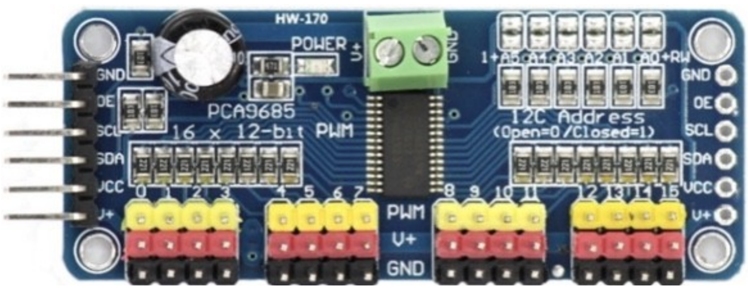


Figure 5: PCA9685.

Table 2 shows a comparative analysis of the characteristics of PCA9685 and 2n7000.

Table 2
Comparison of PCA9685 and 2n7000

Specification	PCA9685	2n7000
Numbers of channels	16	1
Control Type	PWM	PWM or on/off
Control signal	Built-in PWM generator	External PWM signal required
Connection to Arduino Nano	Through I2C, two pins are engaged	Each servo has its own pin
Load on Arduino Nano	Minimal, because the PCA9685 itself generates PWM	External PWM generation source
Load power	Depends on the actuator	Current up to 200mA and 60V
Protection	Not required	Additional diodes and resistors required
Cost	3000 tenge	40 tenge

PCA9685 is more suitable due to more precise selection of the transistor position and minimal load for Arduino NANO. In turn, 2n7000 is much cheaper, but requires more control from Arduino Nano to control.

2.3. Sensors

When building a breathing simulator system, the cornerstone is the sensors. To do this, it’s required to choose which sensors exactly will be, and also develop general software for the system to work with all the sensors. After that, the sensors must be adjusted.

Each sensor will monitor a specific parameter that is important for monitoring the patient's breathing. Consideration must begin with the air parameters, and the sensor that monitored it in the first version of the circuit.

2.3.1. Air quality monitoring

Air quality monitoring will include monitoring of CO2, volatile organic compounds in the air. In the first version of the scheme, the digital sensor CCS811 was chosen (Figure 6). It works via the I2C interface, which allows it to be connected to the PCA9685 and work on two wires. This sensor has not been changed, as it copes with its task perfectly. In addition, it is small.

Figure 6: CCS811 digital sensor [16]

2.3.2. Monitoring of temperature, humidity, and pressure

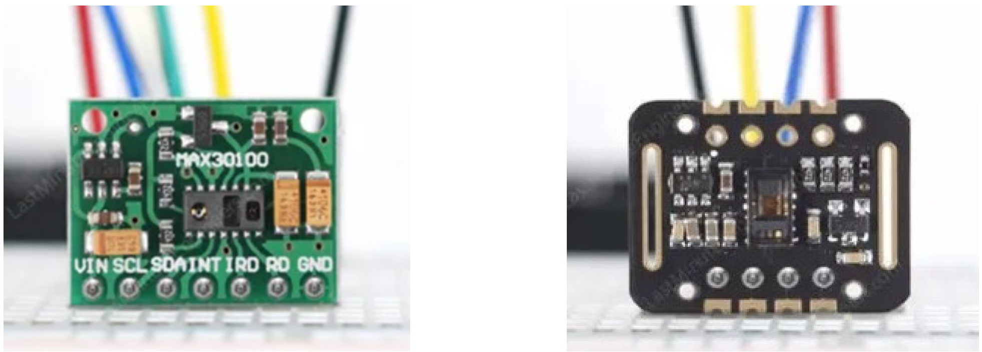
The HTU21 sensor [17] (Figure 7a) was selected for monitoring temperature and humidity, and the BME280 (Figure 7b) for pressure. These sensors coped with their task, but their functions can be performed by the BMP280 sensor [18] (Figure 7c), which, although more expensive, will take up less space, but will reduce the volume of the data queue coming to the Arduino Nano. Also, an important factor in replacing these sensors is the MAX30100 sensor.

a b c

Figure 7: Sensors HTU21 (a), BMP280 (b), BME208 (c).

2.3.3. Monitoring of heart rate and saturation

Saturation and heart rate in the first version of the circuit were monitored by MAX30100. When working with it, an error or "bug" appeared, in which the saturation value could be displayed incorrectly. The MAX30100 sensor (Figure 8a) hung up because of the FIFO cyclic buffer, which did not allow the other sensors to work normally. When changing the number of sensors, the error disappeared. But when improving, this sensor was replaced with a more improved version MAX30102 (Figure 8b). A comparative characteristic is represented in Table 3.



a b

Figure 8: Sensors MAX30100 and MAX30102 [19].

Table 3
Comparative characteristic of MAX30100 and MAX30102

Comparison parameter	MAX30100	MAX30102
ADC Resolution	16 bit	18 bit
FIFO	16 bit	32 bit
LED Pulse width	Wider	Narrower, lower consumption

2.4. Threshold signal

One of the problems identified during testing was that when critical values of saturation and heart rate were reached, there were no notifications about this. To solve this problem, a piezo element ("buzzer") was added (Figure 9), which, when critical values were reached, would receive a signal to emit a sound [20].

Figure 9: Piezo element.

2.5. Power handling

The next change was made to the power supply. To power the first version of the circuit, it was necessary to work with a DCDC converter and several batteries. The circuit requires 5V power supply for most elements, except for several sensors that operate on 3.3V, and the power supply must be stable for several hours of operation. For this, a 20,000 mAh Power Bank was selected, which allows for stable 5V power supply. For sensors operating on 3.3V power supply, an ASMS 1117-3.3 stabilizer was added, which has sufficient data for working with sensors and the required dimensions.

3. Results

Thus, the project scheme was transformed. The comparison results of testing Raspberry Pi Zero and Arduino Nano are presented in Table 4.

Table 4
Comparison of testing Raspberry Pi Zero and Arduino Nano

Comparison parameter	Raspberry Pi Zero	Arduino Nano	Result
Power-on time	At least 2 minutes	2 seconds	Arduino Nano turns on faster
Programming language	Python	C/C++	Arduino NANO executes the program faster
PCA9685	Excellent work with this element	Excellent work with this element	Both devices work great with PCA9685
Sensors	Excellent work with sensors	Excellent work with sensors	Both devices work great with sensors
Power consumption	30mA	~100mA	Arduino Nano works with less power consumption
Multitasking	Resources allow to easily work with a large number of devices	Absent, it's required to build a multitasking system from scratch	If the research requires a large number of different elements, then Rasbperry Pi Zero is more suitable
Complexity	High entry threshold	Low entry threshold	Arduino Nano has a lower entry threshold
Real-time operation	High power consumption and slow program execution	Low power consumption and fast program execution	Arduino Nano is more suitable for real-time tasks

Figure 10 shows the block diagram of the improved circuit. Analyzing the operation algorithm:

1. Power is supplied to the board, Arduino Nano microcontroller, Bluetooth module, sensors and PCA9685 start working.
2. Data from the sensors is received by the microcontroller and sent via Bluetooth to the software.
3. The doctor uses the software to send commands to change the position of the servos.
4. Upon receipt, the microcontroller sends commands to the PCA9685 to change the position of the servos.

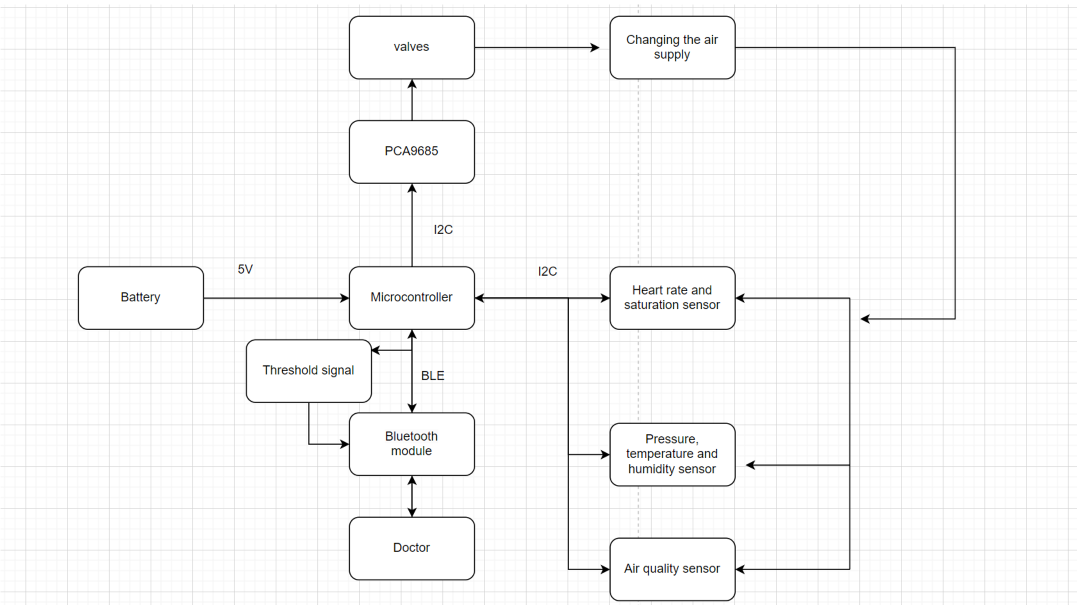


Figure 10: Block diagram of the improved circuit.

4. Conclusion

This article describes in detail the changes made to the scheme of the project "Development of a software and hardware complex for monitoring and correcting respiratory functions based on multimodule technologies" to improve it and explains the reasons for them. Each block of the circuit was analyzed, individual elements were tested. The main changes were made to the control units and servo drives. The control unit was the most important for improvement because its characteristics are important for autonomous operation. Although the Raspberry Pi Zero has great resources, but compared to the Arduino Nano it is less appropriate for this study. In the future, when changing the task, for example, communication with the network, it is possible to integrate ESP32 or ESP8266, as very well-proven microcontrollers. Also, global changes affected the control of the servo drives themselves, which simplifies the task for the microcontroller, only specifying the address in I2C for stable operation of the servo drives. The introduced changes contribute to the performance of the circuit. By making it modular, the circuit became easier to repair and test new software algorithms.

Acknowledgements

This research is funded by the Science Committee of the Ministry of Science and Higher Education of the Republic of Kazakhstan (Grant No. AP19680049).

Declaration on Generative AI

The authors have not employed any Generative AI tools.

References

[1] Zhou, Chen. "The Impact of COVID-19 on Different Human Systems and Related Research Progress." Highlights in Science, Engineering and Technology 91 (2024): 145-150.

[2] Zheng, Bang, et al. "Prevalence, risk factors and treatments for post-COVID-19 breathlessness: a systematic review and meta-analysis." European Respiratory Review 31.166 (2022).

[3] Tariq, Hasan, et al. "State-of-the-Art Low-Cost Air Quality Sensors, Assemblies, Calibration and Evaluation for Respiration-Associated Diseases: A Systematic Review." Atmosphere 15.4 (2024): 471.

[4] Tarigan, Amira P., et al. "Effectiveness of upper arm and breathing exercises to improve inflammatory markers in severe COVID-19 patients." Narra J 4.1 (2024).
[5] Dwivedi, Ruby, Divya Mehrotra, and Shaleen Chandra. "Potential of Internet of Medical Things (IoMT) applications in building a smart healthcare system: A systematic review." Journal of oral biology and craniofacial research 12.2 (2022): 302-318.
[6] Panda, Subhashree, et al. "An IoT-driven COVID and Smart Health Check Monitoring System." The Open Biomedical Engineering Journal 18.1 (2024).
[7] Dong, Shuqin, et al. "Remote Respiratory Variables Tracking With Biomedical Radar-Based IoT System During Sleep." IEEE Internet of Things Journal (2024).
[8] Belkacem, Abdelkader Nasreddine, et al. "End-to-end AI-based point-of-care diagnosis system for classifying respiratory illnesses and early detection of COVID-19: A theoretical framework." Frontiers in Medicine 8 (2021): 585578.
[9] Bagad, Piyush, et al. "Cough against covid: Evidence of covid-19 signature in cough sounds." arXiv preprint arXiv:2009.08790(2020).
[10] Ritwik, Kotra Venkata Sai, Shareef Babu Kalluri, and Deepu Vijayasenan. "COVID-19 patient detection from telephone quality speech data." arXiv preprint arXiv:2011.04299 (2020).
[11] Fan, Dou, et al. "A Contactless Breathing Pattern Recognition System Using Deep Learning and WiFi Signal." IEEE Internet of Things Journal (2024).
[12] Rajarajan, S., et al. "IoT-Enabled Respiratory Pattern Monitoring in Critical Care: A Real-Time Recurrent Neural Network Approach." 2024 10th International Conference on Communication and Signal Processing (ICCSP). IEEE, 2024.
[13] Duzbayev, Nurzhan T., et al. "Development of the Electronic Circuit and Printed Circuit Board for the Breathing Simulator." DTESI (workshops, short papers). 2023.
[14] Arduino Nano, URL: https://arduino.ru/Hardware/ArduinoBoardNano.
[15] PCA9685, URL: https://robotchip.ru/obzor-pca9685.
[16] CCS811 digital sensor, URL: https://robotchip.ru/obzor-datchika-kachestva-vozdukha-ccs811.
[17] HTU 21 GY21 sensor, URL: https://3d-diy.ru/wiki/arduino-datchiki/datchik-temperatury-i-vlazhnosti-gy-21.
[18] BMP280 sensor, URL: https://3d-diy.ru/wiki/arduino-datchiki/sensor-bmp280.
[19] MAX30100, URL: https://lastminuteengineers.com/max30100-pulse-oximeter-heart-rate-sensor-arduino-tutorial.
[20] Arduino Master, URL: https://arduinomaster.ru/uroki-arduino/pishhalka-pezodinamik-arduino.