International Journal of Mechanical and Electrical Engineering
ISSN: 3005-9615 (Print), ISSN: 3005-7132 (Online) | Volume 7, Number 1, Year 2025
DOI: https://doi.org/10.62051/ijmee.v7n1.01
Journal homepage: https://wepub.org/index.php/IJMEE/index

Design of a Human Health Monitoring System Based on STM32

Zihang Li

BEIJING ACADEMY, Beijing, 100018, China

ABSTRACT

This paper presents a multifunctional health monitoring system based on the STM32F103C8T6 microcontroller, capable of real-time tracking of multiple vital signs, including body temperature, heart rate, blood oxygen saturation, blood pressure, and body posture. Equipped with a Wi-Fi module, the system enables real-time remote monitoring of the user's health status and triggers an alarm upon detection of abnormal health indicators. By integrating multiple sensors in a coordinated manner, the system provides reliable health monitoring and enhances personal safety.

KEYWORDS

Health Monitoring System; STM32; Sensor.

1. INTRODUCTION

With the accelerating pace of work and increasing life stress, people are paying more attention to their health conditions. There is a growing demand for convenient and efficient methods to monitor human physiological status. Body temperature, peripheral oxygen saturation, blood pressure, and heart rate are critical physiological indicators, often serving as essential references in clinical settings for disease diagnosis and treatment evaluation [1]. Therefore, this paper presents a multifunctional health monitoring system based on the STM32F103C8T6(STM32)minimum system microcontroller, capable of real-time monitoring and assessment of multiple vital health metrics. The DS18B20 temperature sensor is employed for accurate body temperature measurement, while the MAX30100 heart rate and peripheral oxygen saturation provides real-time monitoring of heart rate and peripheral oxygen saturation, offering crucial insights into cardiovascular health. The MPU6050 triaxial gyroscope detects falls, and a pressure sensor is used to simulate blood pressure measurement. The system features an organic light-emitting diode(OLED) display to intuitively present health data to users, enabling them to check their health status at any time. Additionally, a Wi-Fi module is integrated for real-time remote health monitoring. The system triggers an alarm when abnormal health indices are detected. Through the collaborative operation of these sensors, the system accurately monitors users' health conditions, providing robust safeguards for their well-being.

2. SYSTEM DESIGN AND FUNCTIONAL ANALYSIS

This paper designs a health monitoring system based on the STM32minimum system microcontroller. This paper adopts a modular design, collects key physiological parameters through sensors, and uses STM32 for data processing and health monitoring. The system includes the STM32 minimum system microcontroller, OLED display module, sensor module, key module and alarm module. As shown in Figure 1.

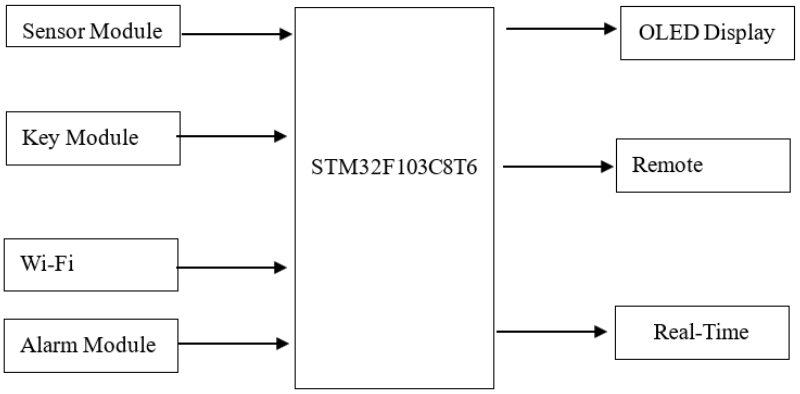

Figure 1. Framework diagram of the health monitoring system

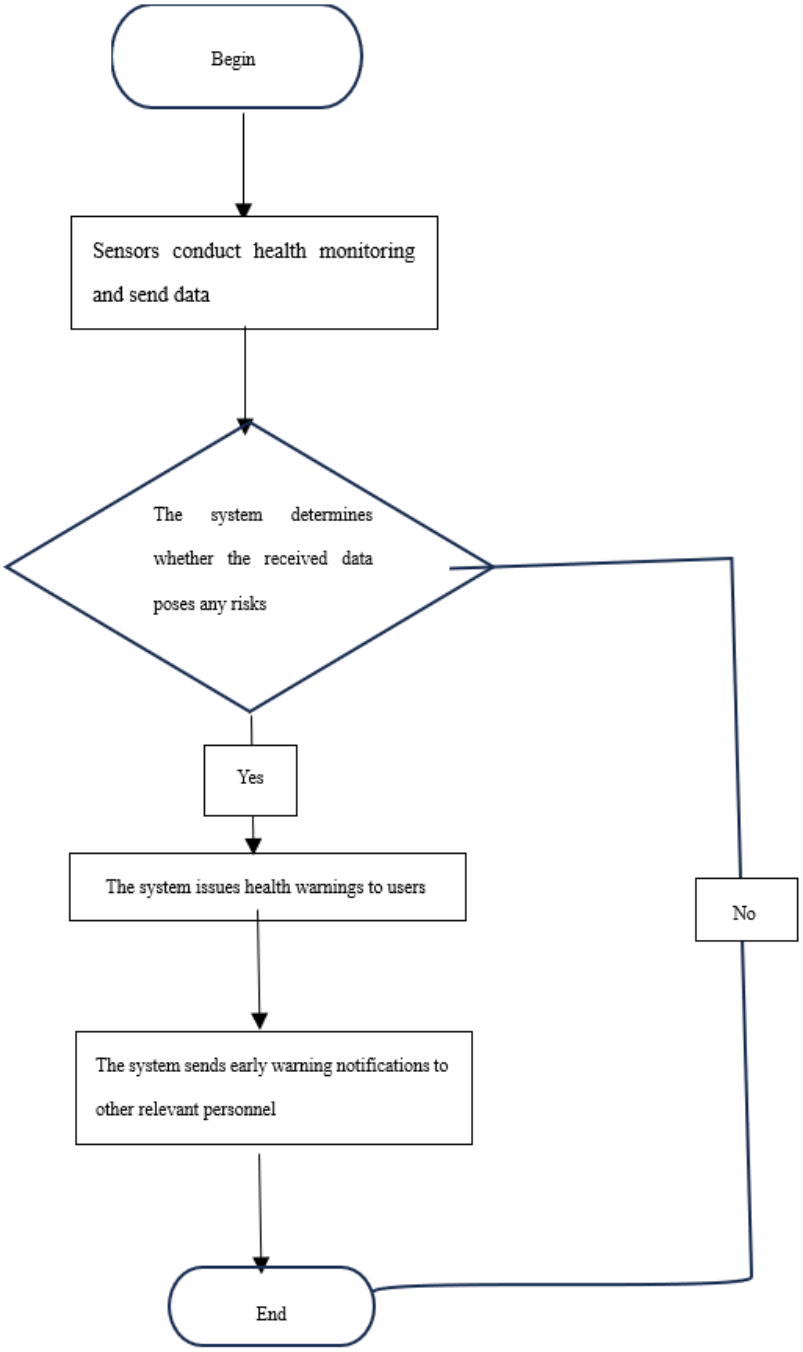

Figure 2. Workflow diagram of the alarm module

Through the coordinated operation of multiple modules, this system enables health monitoring, remote monitoring, real-time data visualization, data parsing, and alert functionalities.

The DS18B20 temperature sensor is employed for accurate body temperature measurement, while the MAX30100 heart rate and peripheral oxygen saturation provides real-time monitoring of heart rate and peripheral oxygen saturation, offering crucial insights into cardiovascular health. The MPU6050 triaxial gyroscope detects falls, and a pressure sensor is used to simulate blood pressure measurement.

The remote monitoring function is achieved by integrating a wireless communication module, enabling real-time collection, processing, and remote transmission of physiological parameters. This allows healthcare professionals or family members to remotely track the individual's health status.

The real-time alarm function refers to the system using preset thresholds for normal heart rate, peripheral oxygen saturation, body temperature, and blood pressure as evaluation criteria. Once a user's monitored data exceeds these thresholds, the system triggers an alarm, as illustrated in Figure 2.

3. HARDWARE CIRCUIT DESIGN AND IMPLEMENTATION

3.1. The STM32 Minimum System Microcontroller

3.1.1. Introduction to STM32

The STM32 is a 32-bit ARM Cortex-M3 microcontroller that integrates essential circuits and components required for microcontroller operation, including a clock circuit, reset circuit, decoupling circuit, and I/O interfaces. It features 64KB to 128KB of high-speed flash memory and 20KB of SRAM, with 37 GPIO pins (including special-function I/O ports). The operating voltage ranges from 2V to 3.6V[2]. Characterized by low power consumption and robust peripheral capabilities, it supports multiple low-power modes, as shown in Figure 3.

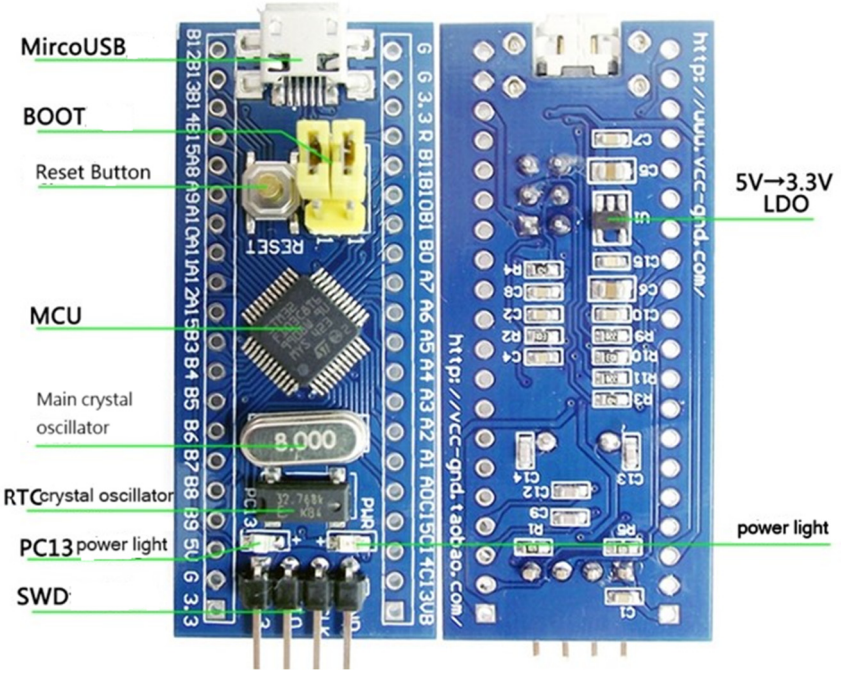

Figure 3. STM32F103C8T6

3.1.2. Reset Circuit

The reset circuit consists of a reset pin, a resistor, and a capacitor[3]. It initializes internal registers, memory devices, and other critical components to their predefined states according to design requirements, ensuring system reliability and stability[4]. This design employs a system reset mechanism, whose circuit configuration is shown in Figure 4. This mechanism serves the following two functions:

(1) During microcontroller startup, the capacitor C5 is in its charging phase. This holds the NRST pin at a low logic level, thereby triggering a system reset. Once C5 is fully charged, it effectively disconnects (from the reset circuit), allowing the NRST pin to be pulled back to a high logic level, which terminates the reset process.

(2) If the button is pressed, the NRST terminal is connected to ground (GND). This pulls the NRST terminal down to a low logic level, thus initiating a system reset operation.

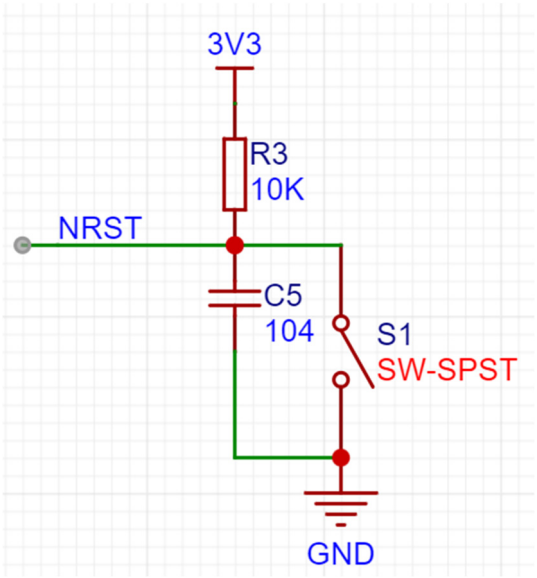

Figure 4. Schematic diagram of the reset circuit

3.1.3. Clock Circuit

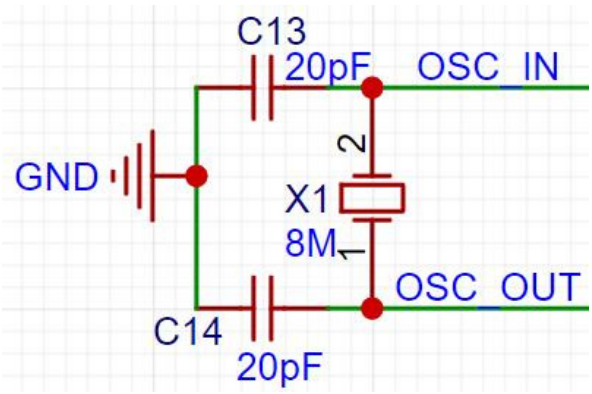

Figure 5. Main crystal oscillator circuit

The crystal oscillator in the clock circuit is made of quartz crystal. When the crystal is subjected to mechanical stress, the natural charge distribution within it is altered, generating positive charge on one end and negative charge on the other[5]. The crystal oscillator provides a precise clock signal for the entire system. This circuit employs 8 MHz and 32.768 KHz crystals as core components, whose oscillation performance directly affects the XTAL1 and XTAL2 interfaces[6]. These interfaces

receive and output the oscillation signals, respectively. Through precise configuration, two 20 pF capacitors are connected in parallel, equivalent to a 40 pF capacitor network, ensuring the stability of the crystal oscillator. This enables the circuit to effectively generate accurate, consistent clock signals, guaranteeing smooth system operation and efficient execution capability in any application environment (as shown in Figures 5 and 6).

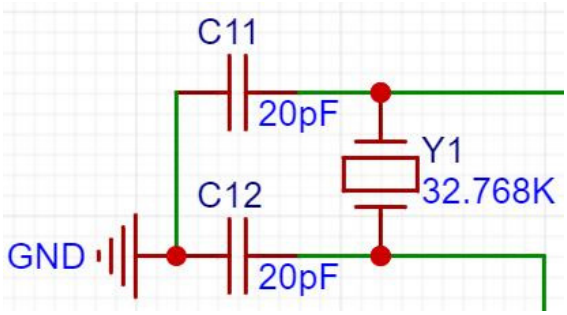

Figure 6. RTC crystal oscillator circuit

3.1.4. Power Conversion Circuit

The STM32minimum system microcontroller requires multiple supply voltages, including core logic voltage (VDD) and I/O port voltage (VDDA). Therefore, a power conversion circuit is essential[7]. This circuit incorporates a voltage regulator that precisely converts a 5V input voltage to a 3.3V output voltage suitable for the STM32[8]. It includes capacitive filters to reduce ripple and noise, as well as overload and overvoltage protection circuits to prevent damage to the STM32 from short circuits or abnormal voltage conditions. This ensures reliable operation and delivers the required performance and reliability of the STM32, as illustrated in Figure 7.

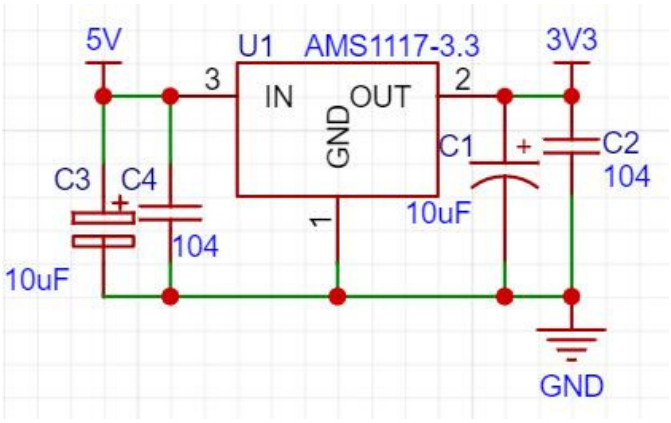

Figure 7. Power conversion circuit

3.1.5. Decoupling Circuit

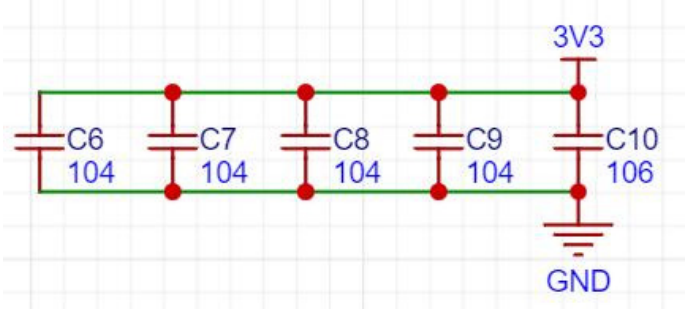

Figure 8. Decoupling circuit

Decoupling capacitors are added to ensure a stable power supply for the STM32 by reducing noise induced by load fluctuations, as illustrated in Figure 8.

3.2. Key Circuit

In this system, the button is solely responsible for emergency calls; therefore, a discrete button configuration is adopted. A discrete button refers to a simple switch circuit directly connected to an I/O port, where each individual button requires a dedicated I/O pin, and these pins operate independently without interference[9]. The board features one discrete button connected to PB9. For the button input, a low-level trigger mechanism is employed: the I/O line remains high when the button is not pressed, and pulling it to low level activates the input. The button circuit design is illustrated in Figure 9.

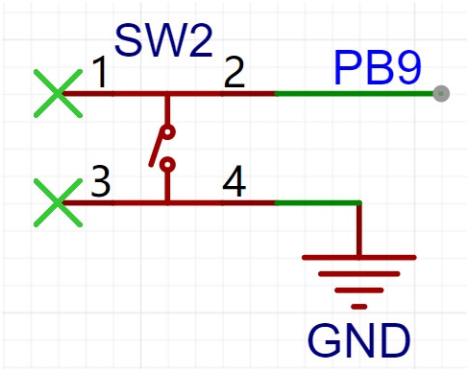

Figure 9. Key circuit

3.3. Active Buzzer

An active buzzer is an electronic sound-producing component with an integrated oscillation circuit, which emits a fixed-frequency sound when powered[10]. In this design, the buzzer is triggered by a low logic level. When no signal is present, its control pin remains at a high logic level. The circuit design is shown in Figure 10.

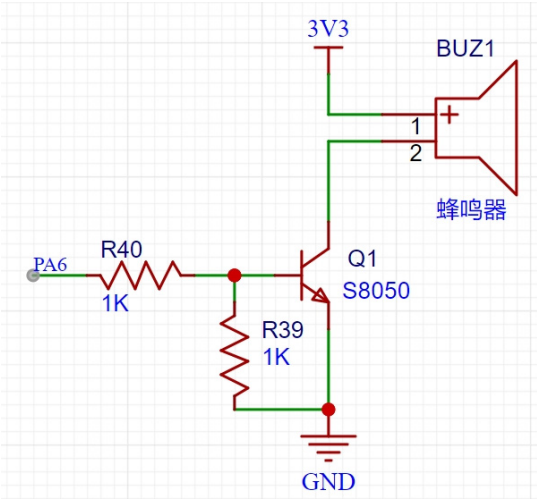

Figure 10. Active buzzer

3.4. OLED Display Circuit

OLED is an organic light-emitting device driven by electric current[11]. This design utilizes a 4-pin 0.96-inch OLED liquid crystal display module. Its operating voltage ranges from 3.3V to 5V, with a display area of 128 × 64 dot matrix. Each pixel can emit light independently, enabling the display of

text, ASCII codes, patterns, and more. The interface circuit is simple and user-friendly. As shown in Figure 11.

Figure 11. OLED Liquid Crystal Display Module

The schematic diagram of the OLED circuit is shown in Figure 12. The GND pin is connected to the GND of the Microprogrammed Control Unit[MCU], while the VCC pin can be connected to either 3.3V or 5V for power supply. SCL and SDA are critical pins on the IIC communication bus and must be connected to the MCU's IIC communication interface. The SDA interface circuit is bidirectional, capable of both transmitting data to the bus and receiving data from it. Its pin is connected to PB10 of the MCU for sending and receiving data signals. The serial clock line SCL is also bidirectional. It serves as the clock pin on the OLED display for transmitting clock signals and is connected to PB11 of the MCU[12].

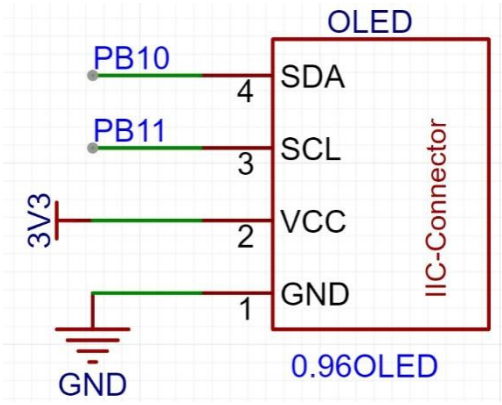

Figure 12. OLED Display Circuit

3.5. Heart Rate and Peripheral Oxygen Saturation

Figure 13. Heart rate and peripheral oxygen saturation

MAX30100 is a sensor that integrates pulse oximetry and heart rate monitoring functions. It utilizes advanced photoelectric detection technology, incorporating built-in red and infrared LEDs along with a highly sensitive photodetector[13]. By detecting the absorption of light at different wavelengths through human tissue, it accurately measures heart rate and blood oxygen saturation. This sensor features a compact size and low power consumption. Additionally, it is equipped with an IIC interface, facilitating communication with microcontrollers and enabling the transmission of collected data to other devices for further processing and analysis. As shown in Figure 13.

The implementation circuit of the MAX30100 heart rate and peripheral oxygen saturation is primarily based on the principle of photoplethysmography (PPG). The circuit consists of a light-emitting section and a light-sensing section. The light-emitting section integrates red and infrared LEDs. When activated, these LEDs emit light at two different wavelengths toward human tissue. The light-sensing section comprises a highly sensitive photodetector, which captures the light either reflected from or transmitted through the tissue.As the heart beats, changes in blood volume within the tissue alter the absorption of light at different wavelengths, causing fluctuations in the intensity of light received by the detector[14]. The sensor's internal signal processing circuit converts these variations in light intensity into electrical signals, which are then amplified, filtered, and processed to remove noise interference.The processed data is transmitted via the I^2C interface to the microcontroller, which analyzes it using specific algorithms to calculate heart rate and blood oxygen saturation (SpO_2). The circuit diagram is shown in Figure 14. The SCL pin is connected to PA5 of the MCU to provide the clock signal, while the SDA pin is connected to PA4 of the MCU for data transmission.

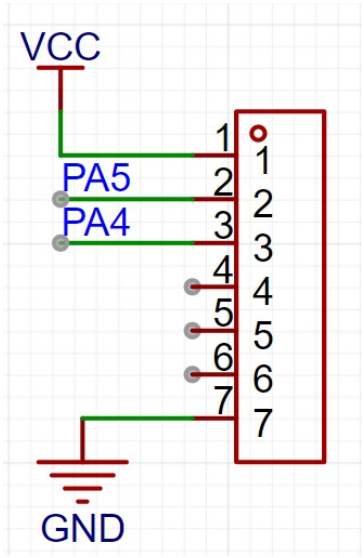

Figure 14. Heart rate and peripheral oxygen saturation Module Circuit

3.6. Temperature Sensor

Figure 15. DS18B20

The DS18B20 is an integrated temperature and humidity detector that incorporates an NTC thermistor for temperature sensing, combined with a high-performance 8-bit microcontroller[15]. This design ensures the sensor features rapid response speed, excellent anti-interference capability, and a convenient 3-pin single-row pin package for easy connection. As shown in Figure 15.

The core component of the temperature sensing module is the temperature sensing unit, which is responsible for monitoring ambient temperature variations. The data port of the DS18B20 employs an open-drain configuration. Without an external pull-up resistor, its output can only achieve a low-level state or high-impedance state, and cannot generate a high-level signal. Therefore, an external pull-up resistor must be added to ensure proper high-level output[16]. In this circuit, a 10kΩ resistor is used.Pin 3 (DATA) of the DS18B20 is connected to PA7 of the microcontroller. This DATA port serves as a communication bridge between the microprocessor and the DS18B20, facilitating synchronous serial bidirectional communication following the single-wire bus data protocol. The circuit schematic is shown in Figure 16.

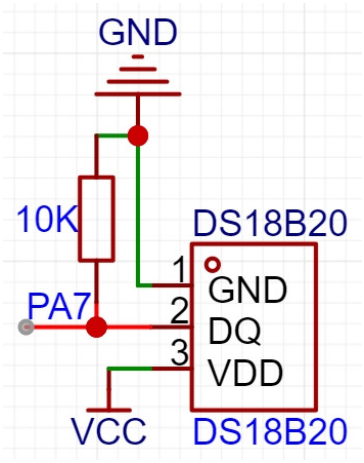

Figure 16. DS18B20 Circuit

3.7. Pressure Sensor

The XGZP6847A is a high-performance pressure sensor characterized by high sensitivity, high accuracy, and excellent linearity. It features low power consumption, making it suitable for battery-powered devices. The sensor provides stable output signals, effectively resists external interference, and is easy to integrate, as shown in Figure 17.

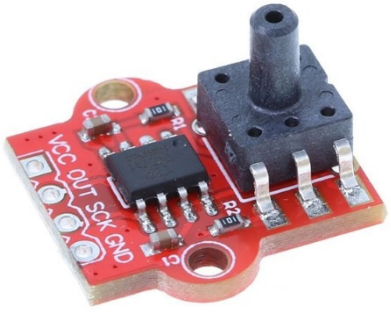

Figure 17. Pressure Sensor

The core of the XGZP6847A pressure sensor is a piezoresistive pressure-sensitive element. When external pressure acts on the sensor's pressure port, the diaphragm of the sensing element undergoes slight deformation. Since varistor is integrated on the diaphragm, this deformation causes a change in the resistance of these elements. The internal circuitry of the sensor converts this resistance variation into a corresponding voltage signal.This voltage signal is then amplified, calibrated, and subjected to temperature compensation[17]. Temperature compensation is applied to eliminate the

influence of ambient temperature variations on the measurement results, ensuring high accuracy. Finally, the processed and stabilized voltage signal is output through the sensor's output pin, where it is connected to subsequent signal acquisition and processing circuits, such as a microcontroller, for further analysis and application—enabling precise pressure measurement.The circuit schematic is shown in Figure 18. The VCC pin is connected to an external voltage supply of 3.3V to 3.5V, the out pin is connected to PB8 of the MCU for signal output, and the SCK pin is connected to PB7 of the MCU to provide the clock signal.

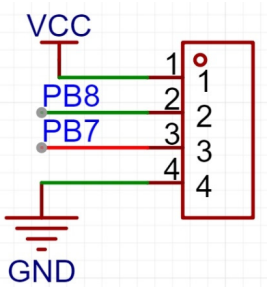

Figure 18. Pressure Sensor Circuit

3.8. The MPU6050 Triaxial Gyroscope

The MPU6050 integrates a three-axis gyroscope and a three-axis accelerometer into a single chip, offering significant advantages of compact size and low power consumption. The three-axis gyroscope accurately measures angular velocity around three distinct axes, while the three-axis accelerometer detects linear acceleration along these three axes. Additionally, the MPU6050 features an integrated Digital Motion Processor (DMP) hardware acceleration engine, which processes complex motion attitude calculations, greatly reducing the computational load on the external microcontroller. It utilizes an IIC interface for communication, facilitating easy connection with various microcontrollers[18], as shown in Figure 19.

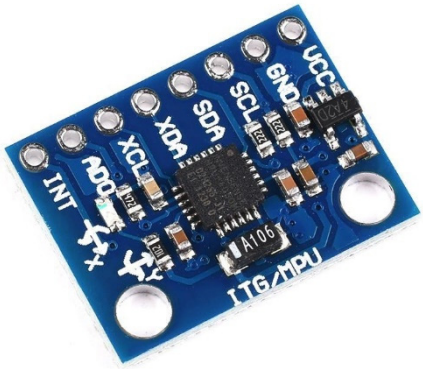

Figure 19. MPU6050 triaxial gyroscope

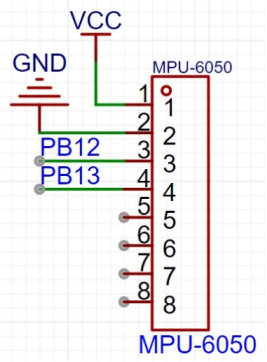

Figure 20. MPU6050 Schematic Diagram

The circuit schematic of the MPU6050 is shown in Figure 20. The clock pin (SCL) of the sensor is connected to PB12 of the MCU, and the data pin (SDA) is connected to PB13 of the MCU. PB12 provides the clock signal, while PB13 transmits the data signal.

3.9. Wi-Fi Module

The ESP8266 is a low-cost, high-performance Wi-Fi module that highly integrates a TCP/IP protocol stack, providing convenient WiFi connectivity for other devices[19]. With its compact size and low power consumption, this module is well-suited for applications with strict space and energy requirements. It can function as an access point (AP), allowing other devices to connect and share a network, or as a station (STA) to join existing Wi-Fi networks.

The ESP8266 offers a variety of interfaces, such as GPIO and UART, facilitating easy connection and communication with various microcontrollers, sensors, and other peripherals. It also supports multiple development frameworks, including Arduino and MicroPython[20], as shown in Figure 21.

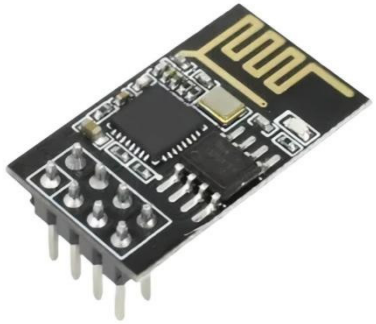

Figure 21. WiFi

The implementation circuit of the ESP8266 Wi-Fi module is primarily designed around its highly integrated system-on-chip (SoC), which incorporates a radio frequency (RF) transceiver, baseband processor, and TCP/IP protocol stack. The schematic diagram of the circuit is shown in Figure 22.

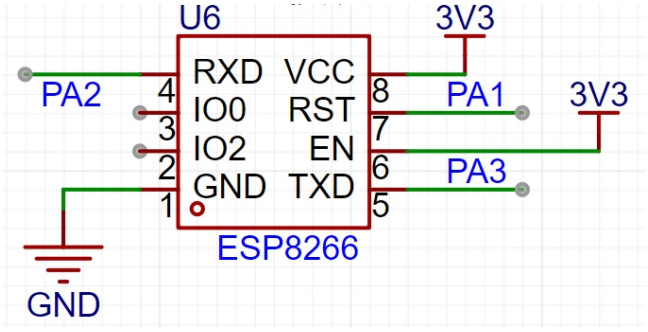

Figure 22. Circuit diagram of the Wi-Fi module

4. SYSTEM SOFTWARE DESIGN AND IMPLEMENTATION

4.1. Introduction to Keil MDK-ARM

Keil MDK-ARM (Keil5) is used for development in this paper. Keil MDK is developed by ARM. Keil5 is a complete software development suite whose core is the µVision IDE, integrated with a series of tightly collaborative components. µVision provides an all-in-one environment for project management, source code editing, compilation, building, debugging, and simulation. It supports modern editor features such as multi-window, code folding, syntax highlighting, and intelligent sensing, greatly improving coding efficiency[21]. Keil5 can generate machine code that achieves an optimal balance between code size and execution speed, offering excellent code efficiency and

performance, robust debugging and diagnostic capabilities, and strong compatibility. Its workflow is as follows:

(1) Acquire and configure the Keil Environment. Firstly, the Keil Integrated Development Environment (IDE) installation file suitable for the device needs to be located and downloaded from the official Keil website. After initiating the download, follow the step-by-step guide to perform a thorough system compatibility check and a customized installation setup.

(2) Create a project.Launch the installed Keil uVision program. Click on the "Project" menu option and select the "New Project" function. This will open an interface where the user must specify the project's save location and assign a name to the project for future management and identification.

(3) Configure parameters.Within the project settings, select the appropriate microcontroller unit (MCU) model (e.g., STM32F103C8T6, used in this design) to ensure the compiler and linker can correctly identify the target hardware.

(4) Add files.Add the required source code files (e.g., .c, .cpp) to the project. Create a new text file and name it main.c.

(5) Write code.Write the code within the main.c. file in Keil. This includes initialization routines, functional functions, and the main application logic.

(6) Compile the project.Compile the project by clicking the "Build" button on the toolbar. If errors occur during compilation, utilize operations like setting breakpoints and single-stepping through the code to debug the program and correct the erroneous code.

(7) Download code to the microcontroller: Click "Download" to flash the successfully compiled program into the microcontroller's memory.

4.2. Define Variables

First, define various states, sensor data, and flag bits required for system operation through code.

4.2.1. Status Flag Bits (Flags)

Boolean variables are used to record certain states of the system, typically with 1 indicating "abnormal" and 0 indicating "normal".

```
u8 flag=0;                //General Flag
u8 str[20];
_Bool send_flag = 0;     // TX Flag
u8 mpu_flag = 0;          //Fall Detection Master Flag[1: Fall, 0: Normal]
_Bool mpu_1_flag = 0;    //Attitude Anomaly Flag. Determined by the attitude estimation
                          algorithm. This flag is set to 1 if the angle (e.g., body tilt) exceeds the
                          safe threshold[1: Fall, 0: Normal]
_Bool mpu_2_flag = 0;     //Acceleration Anomaly Flag. Determined by the SVM magnitude.
                            This flag is set to 1 if the acceleration magnitude (impact force)
                            exceeds the safety threshold.
u8 t=0,i=10;
u8 T;
u8 BJ=0;                  //Alarm Flag
```

4.2.2. Data Computation and Communication Buffer

```
char *subString;
char *subStringNext;
```

```
signed short HeartRate_val=0;
u8 SPO2_val = 0;
float pitch,roll,yaw;                    //Euler Angles
short aacx,aacy,aacz;                    //Raw Accelerometer Data
char tp[255];
u8 ii=0;
u16 s=0;
u16 JJ=0;                               //Emergency Button Flag
```

4.2.3. Threshold Setting

```
int sxx_value=140;                  // Blood Pressure Threshold
int sxy_value=100;                  // Blood Oxygen Threshold
int sxL_value=140;                  //Heart Rate Threshold
int st_value=37;                    //Temperature Threshold
```

4.3. Initialization Functions and Hardware Devices

The main function wakes up, checks, and configures all hardware devices, then connects to the network[22].

```
int main(void)
{
Initialization Function:
      LED_Init();                       // Indicator Light Initialization
       Usart1_Init(9600);                 //UART1 Initialization
       Usart2_Init(115200);              // UART2 Function Initialization, Baud Rate 115200
       TIM4_Init(300,7200);               //Timer Initialization
                                        Timing Period: 300 * 7200 * 1000 / 72,000,000 = 30 ms
       WiFi_ResetIO_Init();               //Wi-Fi Reset IO Pin Initialization
       MQTT_Buff_Init();                  //Initialize the buffers for receiving, transmitting,
                                        and command data, along with all status parameters
       AliIoT_Parameter_Init();             //Initialize the connection parameters for the Alibaba
                                            Cloud IoT Platform MQTT server
       DS18B20_Init();                   //Temperature Sensor Initialization
       OLED_Init();                       //OLED Module Initialization
       IIC_Init();                           //Initialize IIC
       TIM1_Init();                          //Init TIM1 for 1ms interrupt
       SPO2_Init();                          //Blood Oxygen Sensor Initialization
       MPU_Init();                            //MPU6050 Sensor Initialization
       delay_ms(1000);                   //Wait for Initialization to Stabilize
       while(mpu_dmp_init())             //DMP Library Initialization
       {
            delay_ms(200);
       }
//     show_interface();                      //Display Home Page
//     mpu_dmp_init();
       wifi_IN();                          Wi-Fi Connection Function
```

```
Init_HX711pin();                Pressure Sensor Initialization
```

4.4. Infinite Loop

The program will cyclically perform sensor data acquisition, processing, display, judgment, communication, and listening [23].

4.4.1. Data Acquisition Scheduling

To reduce system power consumption, sensors with different sampling rate requirements are scheduled in a time-division manner. Controlled by a loop counter t, the acquisition of temperature, simulated blood pressure values, and motion attitude data is executed once every 10 loop cycles (approximately several hundred milliseconds, depending on the execution time of the loop body). Among these, the motion attitude data is processed by the built-in Digital Motion Processor of the MPU6050 to calculate Euler angles and the acceleration vector magnitude (SVM), providing input for fall detection.

4.4.2. Fall Detection Algorithm

Based on the fusion of posture and acceleration data for judgment. The algorithm is defined as follows: when the absolute value of any human body posture euler angle (pitch, roll, yaw) exceeds 40 degrees, it is determined as a posture anomaly. This criterion, combined with the acceleration vector magnitude (SVM), forms the core logic of fall detection. Once triggered, the fall flag mpu_flag is set.

4.4.3. Health Threshold Monitoring and Alert

The system employs an independent long-cycle counter s (200 cycles) to perform comprehensive health status evaluation. This mechanism continuously compares the collected heart rate, blood oxygen, systolic blood pressure (simulated value), and body temperature data with user-preset safety thresholds. Any parameter exceeding the limit will trigger an alert.Upon alert triggering, the system executes a two-level response:

Local response:Immediately activates LED indicator flashing and buzzer sounding to provide an intuitive on-site warning.

Status encoding: Records the specific type of exceeded parameter in the alert code BJ. This code will be transmitted to the cloud along with the data reporting process, facilitating precise problem identification by the backend system.

4.4.4. Automatic Alert Clearance

When all monitored physiological parameters return to normal levels, the system will automatically deactivate the audible and visual alarms and reset the alert status. This closed-loop control mechanism prevents user disturbance caused by persistent false alarms.

Specific code is as follows：

```
while(1)
    {
     if(flag==1){
            POupdate();                 //Acquire Heart Rate Data
//====Data Acquisition====
                if((t%10)==0)
                {
```

```
            if(K1==0)JJ=1;
            else       JJ=0;
                    T=DS18B20_Get_Temp()/10;                //Acquire Body Temperature
                    show_temp(T);                           //Display Temperature on Screen
            Get_Maopi();
            show_XY(Weight_Maopi/100000);               //Display Blood Pressure on Screen
//                      show_XY(sxx_value);
//                      show_temp(st_value);
//                      show_max30100(sxL_value,sxy_value);
                        if(mpu_dmp_get_data(&pitch,&roll,&yaw)==0)
                        {
                        MPU_Get_Accelerometer(&aacx,&aacy,&aacz);    //Acquire   Accelerometer
Data
SVM = sqrt(pow(aacx,2)+   pow(aacy,2) + pow(aacz,2));
                                //Analysis of Abnormal Judgment for X, Y, and Z Angles
                                if( fabs(pitch)>40 || fabs(roll)>40 || fabs(yaw)>40 )
//A fall is determined to occur when the absolute value of the tilt angle exceeds 40° and the SVM
exceeds the set threshold.
                                                mpu_1_flag = 1;
                                else
                                     mpu_1_flag = 0;


//Comprehensive anomaly detection based on Euler angles and SVM abnormality judgment
                                if( mpu_1_flag )mpu_flag = 1;
                                else mpu_flag = 0;


//                              show_mpu(mpu_flag);
                        }
                        t=0;
                }
                t++;
        s++;
        if(s==200)
        {
//          LED2=~LED2;
```

```
if((Weight_Maopi/100000)>sxx_value || HeartRate_val>sxL_value || SPO2_val>sxy_value || T>st_value)// Set parameter upper threshold: blood pressure at 140, heart rate at 140, blood oxygen at 140, temperature at 37.2 {
  LED2=~LED2;
BEEP=~BEEP;
  if(Weight_Maopi/100000>sxx_value)BJ=1;
  if(SPO2_val>sxy_value)  BJ=2;
if(HeartRate_val>sxL_value)  BJ=3;
if(T>st_value)  BJ=4;
}
if((Weight_Maopi/100000)<=sxx_value && HeartRate_val<=sxL_value && SPO2_val<=sxy_value && T<=st_value)
{
  BJ=BEEP=LED2=0;
```

4.5. Connect to Wi-Fi for Wireless Communication

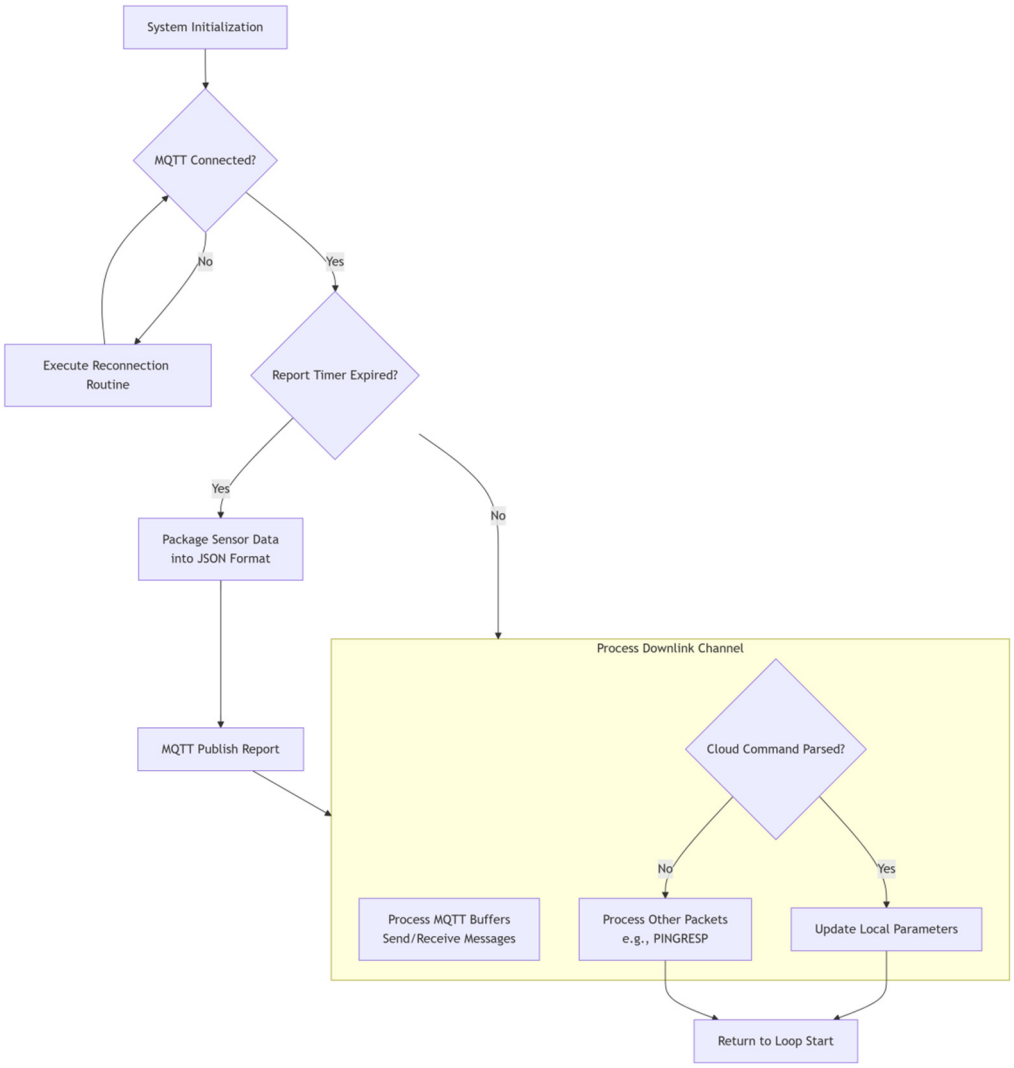

Figure 23. Workflow Diagram

To enable remote monitoring and data visualization of the device, this system utilizes the MQTT protocol to connect to the Alibaba Cloud IoT Platform, ensuring reliable data uplink and real-time command downlink.

4.5.1. Communication Process Overview

After the device completes initialization upon power-on, it first establishes an MQTT connection with the Alibaba Cloud Platform and subscribes to relevant topics. It then enters the main loop. The main communication loop forms a closed-loop process encompassing data reporting, packet processing, command parsing, and disconnection reconnection. The workflow is as Figure 23.

4.5.2. Data Uplink

The system periodically packages collected physiological parameters and status information into JSON messages according to the Alibaba Cloud IoT Platform's Thing Specification Language (TSL) standard format. Key reported data includes heart rate (XL),peripheral oxygen saturation (XY), body temperature (T), fall detection alarm flag (ZT), GPS positioning data (JD, WD), and various threshold settings. After encapsulation is completed, the data is published to the property reporting topic (/thing/event/property/post) by calling the MQTT_PublishQs0() function, thereby completing the reporting of data to the cloud.

4.5.3. Command Downlink

One of the core functions of this system is its support for remote dynamic configuration. Cloud applications or users can issue JSON commands containing new threshold values. The device firmware processes these commands through the following procedure:1)Reception and extraction: The JSON-formatted command string is extracted from the MQTT push packet.2)Keyword retrieval: The strstr() function is used to search for key parameter names within the string (e.g., "SXX" represents the systolic blood pressure threshold).3)Parameter update: After locating the value position, a custom simple_atoi() function converts the string-formatted numerical value into an integer and updates the corresponding global variable (e.g., sxx_value).This implementation enables remote modification of alarm thresholds for heart rate, blood pressure,peripheral oxygen saturation ,and body temperature, significantly enhancing the system's flexibility and usability.

4.5.4. Connection Maintenance and Reliability Assurance

To ensure the stability of the communication link, the system implements the following mechanisms: 1) Heartbeat keep-alive: Regularly sends PING requests to the cloud platform. If no response (PINGRESP) is received, an accelerated retry mechanism will be triggered. 2) Automatic reconnection: Continuously monitors the connection status (Connect_flag). Once a disconnection is detected, the system immediately clears the communication buffers and automatically executes a complete reconnection procedure until the connection to the cloud platform is restored, thereby ensuring the device's long-term online capability.

5. CONCLUSION

This paper presents the design of a multifunctional health monitoring system based on the STM32 microcontroller. The system enables real-time tracking and remote supervision of key health metrics, including body temperature, heart rate, peripheral oxygen saturation, fall detection, and simulated blood pressure. A modular architecture is adopted, incorporating sensors such as the DS18B20, MAX30100, and MPU6050 to achieve high-precision data acquisition, while an OLED display offers an intuitive human-machine interface. Furthermore, the integration of a Wi-Fi module allows real-time data transmission to cloud platforms or mobile devices, facilitating remote monitoring by healthcare providers or family members. The system also provides prompt alerts in case of abnormalities, significantly enhancing the intelligence of health monitoring. With advantages such as low cost, low power consumption, high integration, and user-friendly operation, the system is well-

suited for a wide range of applications including home healthcare, elderly care, chronic disease management, and fitness monitoring. Future work should focus on extending system functionality, optimizing power efficiency for wearable applications, and developing cloud-based personal health management platforms. Incorporating big data analytics could offer users personalized health insights and long-term medical record management, while further improving the user interaction experience.

REFERENCES

[1] Zhang Dahui, Sun Yumiao, Fu Jinglei, et al. Embedded Services: Solving the Essential Questions of an Aging Society. Health News, 2024-02-19(6).

[2] Xu Jiangdiyu, Yu Anjun, Zhu Cong, et al. A Remote Real-Time Human Health Monitoring System Based on ZigBee. Information Technology, 2017(3): 57-59+64.

[3] Wu Fengbo, Zhou Yunru, Zhang Huike. Design and Implementation of an Intelligent Health Monitoring Terminal. Modern Electronics Technique, 2017, 40(14): 64-67+71.

[4] Fan Yu, Wang Zhong, Yang Qi. An Elderly Health Monitoring System Based on Android. Information Technology and Informatization, 2018, No.221(08): 75-78+82.

[5] Tang Yinsheng, Su Te, Guo Lin. Design of an Elderly Health Monitoring System Under Cloud Computing. Techniques of Automation and Applications, 2020, 39(02): 57.

[6] Luo Qian. Design of a Home Health Monitoring System for the Elderly Based on NB-IoT. Chengdu: Xihua University, 2020 [Accessed: 2024-2-22].

[7] Gao Minghua, Xu Lijin, Ke Chengcheng, et al. Design of a Portable Human Health Monitoring System Based on Android. Modern Electronics Technique, 2017(12): 86-89.

[8] Chen Qiang, Wu Jiajia. Research and Design of a High-Concurrency IoT Server Based on Netty. Electronic Technology & Software Engineering, 2018, 000(007): 34-35.

[9] Chen Huan, Fan Qiyuan, Pang Quanhai. Design and Implementation of a Multi-Physiological Information Acquisition and Monitoring System Based on Wi-Fi Protocol. Scientific Chinese, 2015, (33): 70.

[10] Han Lu. Design and Preliminary Implementation of a Chronic Disease Health Monitoring Platform.Guangzhou: Southern Medical University, 2013.

[11] Tan Jinhua, Lü Jianchao. Design of a Blood Pressure Monitor Based on STC89C52 Microcontroller. Techniques of Automation and Applications, 2011(8): 30-33.

[12] Hu Guangshu. Modern Signal Processing Tutorial. Beijing: Tsinghua University Press, 2004.

[13] Zeng Jincheng. Development of an Intelligent Medical Information Transmission System Based on Android Terminal and Multi-Network Integration. Nanjing University of Posts and Telecommunications, 2013.

[14] Wang Jia. Design of a Multi-Point Infrared Temperature Measurement System Based on MLX90615 and STM32. Modern Electronics Technique, 2013(7): 146-147.

[15] Jiang Changzhen. Signal Analysis and Processing. Tianjin: Tianjin University Press, 2000.

[16] Huang Weimin. Design Scheme of an Instant Messaging System Client on the Android Platform. Modern Electronics Technique, 2011, 34(16): 141-142.

[17] Huang Haibo, Zhang Miao. Design and Implementation of a Wi-Fi Data Transceiver Module Based on STM32. 2012, 3(10): 14-17.

[18] Chen Guangyi. Real-Time Human Health Indicator Monitoring System. Harbin University of Science and Technology, 2017.

[19] Li Guangwu. Design of a Human Physiological Signal Acquisition System Based on USB 2.0 Protocol. Harbin Engineering University, 2004.

[20] Ge Luyao. Design and Implementation of a Mobile Medical System Based on a Cloud Platform. Shandong Normal University, 2015.

[21] REZA S A, AZIZUL M H, ZUBAYER M M, et al. Research and Development of an IoT-based Remote Asthma Patient Monitoring System. Journal of Healthcare Engineering, 2021, 2021:2192913-2192913.

[22] Melillo P, Orrico A, Scala P, et al. Cloud-based Smart Health Monitoring System for Automatic Cardiovascular and Fall Risk Assessment in Hypertensive Patients. Journal of Medical Systems, 2015, 39: 1-7.

[23] GAO P, HOU Y H, WEI L, et al. Design Implementation and of ECG Monitoring and Management System Based on Smartphone Platform. Chinese Medical Equipment Journal, 2017, 38(7):6-11.