# Design and Implementation of Small-scale Sensor Network Based on Raspberry Pi

Honglei Zheng[1](✉), Hong Guo[2], and Shimin Wu[1]

[1] College of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan, China
derekzheng125@163.com

[2] Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial System, Wuhan, China

**Abstract.** With the development of modern technology, embedded technology has continued to develop. In the deep cooperation between embedded and wireless network technologies, the technology of sensor network has been born. The software system based on traditional wireless network cannot collect data in the real world. In some production environments, the embedded sensor network consists of a large number of sensor nodes, integrating sensing, acquisition, processing, and transceiving functions into an autonomous network system to achieve dynamic and intelligent collaborative sensing of the physical world. Based on the existing wireless sensor network system architecture, this paper designs and implements a small-scale sensor network system based on Raspberry Pi. After the software and hardware test of the sensor network system, the test results show that the system can complete the information collection and transmission, and has stability and synchronization to meet project requirements and stability requirements.

**Keywords:** Embedded · Sensing network · WiringPi · Websocket

## 1 Introduction

In today's wave of technology that promotes Internet+, traditional industries are beginning to seek cooperation with emerging Internet technologies to achieve industrial internetization. The traditional Internet network design can't exchange data with the real world, so the function is limited in some production and living environments, and the embedded sensor network can access a variety of sensor devices, integrating information collection and information processing functions. Forming an independent and complete network system to achieve the perceived synergy of the physical world [1].

The current mainstream embedded transmission protocol considers the power consumption and performance of the central control board. Most of them use low-rate network programming protocols to reduce the power consumption during transmission. With the performance of embedded chips, the Raspberry Pi class the high-performance development board has become more cost-effective, which can support high-complexity high-speed transmission protocols, and the number of access devices is greatly improved.

Therefore, this paper mainly designs and implements a small-scale sensor network using the Raspberry Pi as the central control board. The Raspberry Pi is used to collect sensor information, and the information is sent to the server through the JSON transmission format after the information is collected and processed, and the instructions sent by the server are used to operate the hardware devices in the sensing system, thereby realizing stable data transmission and more. The type sensor accesses and automatically detects the reconnection after disconnection, which solves the problem that the device node goes to sleep after the network is accidentally disconnected. The sensor network has high stability and good synchronization, and has application prospect and value.

## 2 Related Technology

### 2.1 Sensor Network System

The basic sensor network system mainly includes a sensor node, a sink node, and a management node. The sensor data is transmitted hop by hop through the network, and is routed to the aggregation node through the multi-hop route in the self-organizing network, and the data is exchanged between the management nodes by using the Internet network.

(1) Sensor node. Mainly engaged in information collection and data processing tasks, processing power is general. At the same time, it receives other node data and cooperates with the task in the self-organizing network.
(2) Aggregation node. It is responsible for connecting the sensor network system to the external network. It is used as a gateway node for data exchange. At the same time, it accepts the management node command to monitor and control the sensor node. The aggregation node can also be an enhanced sensor node.
(3) Management node. The user of the sensor network system operates through the management node and accesses the hardware and software resources in the system. After the command is issued, the node is sent to other nodes through the node to manage the entire sensor network system [2].

Differentiated from the functional modules, the infrastructure of the sensor network uses the microprocessor minimum system as the core module, the sensing module is the functional module, including all access sensors in the sensor network, and the wireless communication module includes all network communication and data transmission. The processing module and the power module are responsible for powering the above three modules. The specific architecture is shown in Fig. 1.

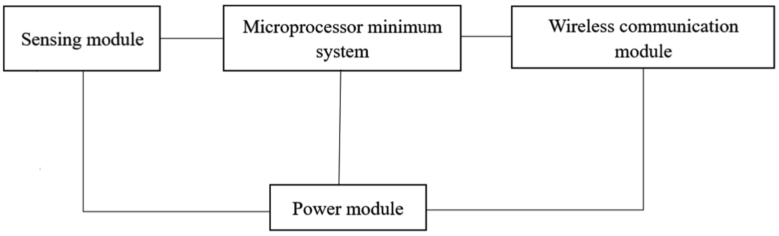


**Fig. 1.** Sensor network system structure

## 2.2 Websocket

Websocket is a full-duplex communication protocol first proposed by HTML5. The protocol is established on a single TCP connection [3], which makes the exchange of data between the client and the server easier. The protocol allows the server to take the initiative. Push data to the client. In the WebSocket API, the browser and the server only need to complete a handshake. The two can directly create a persistent connection and perform bidirectional data transmission. Through this fast data channel, the data between the server and the client. Transfer is more efficient [4]. The WebSocket protocol based on the HTTP 1.1 version can save server resources and bandwidth, and communicate in a stable and real-time manner (Fig. 2).

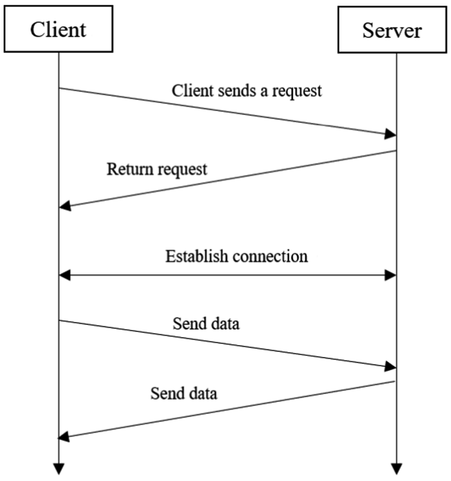


**Fig. 2.** Schematic diagram of the connection process between the server and the client

After the connection is completed, in the characteristics of the Websocket, the heartbeat mechanism is used to realize the long-lived keep-alive. The basic method based on TCP can be adopted, or a custom protocol can be adopted. The custom protocol has a smaller packet format and avoids the HTTP packet. Byte-level overhead improves transmission efficiency and saves bandwidth in real-world transmission applications.

## 3 Sensor Network Design

### 3.1 System Function Design

Nowadays, IoT devices are entering people's lives. The family Internet of Things makes people's lives more intelligent. People can use their mobile devices to control home appliances or get the data they need from their home sensor networks. The sensor network system studied in this thesis starts with the hardware part sensor network of the Internet of Things. It explores that one can send data to the server by the sensor, and the server can send instructions to the sensor node in the sensor network to the home. The system in which the device is controlled. With this system, the user can issue commands in the client of the mobile phone to operate the home device, and can select the device to view the data generated by the device at the set time point in real time. The overall architecture is shown in Figs. 3 and 4.

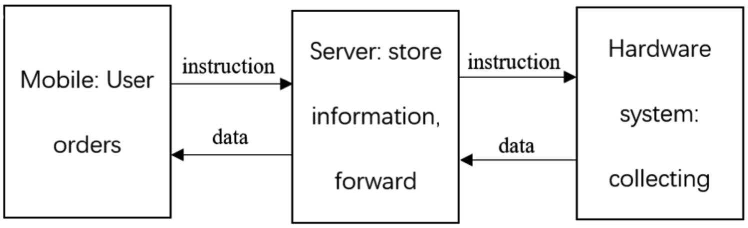

Mobile: User orders
instruction
data
Server: store information, forward
instruction
data
Hardware system: collecting

**Fig. 3.** Functional design of the Internet of Things system

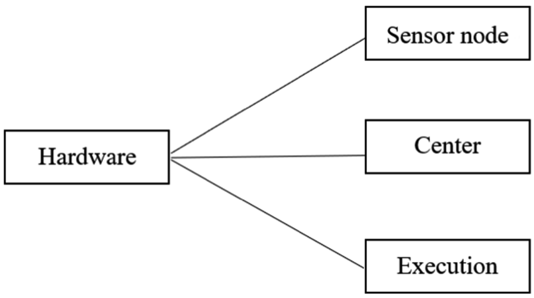


**Fig. 4.** Hardware system architecture

Cut into the sensor network part of this study, that is, the function design is: the sensor node reads the data and then reads it by the central control board, encapsulates it with the embedded JSON format, and uploads it to the data server by the central control board using the wireless network. After receiving the instruction information from the user, the central control board parses the received JSON information packet, thereby reading the operation state of the sensor node, and performing corresponding settings. The sensor network hardware system architecture.

### 3.2 System Architecture Design

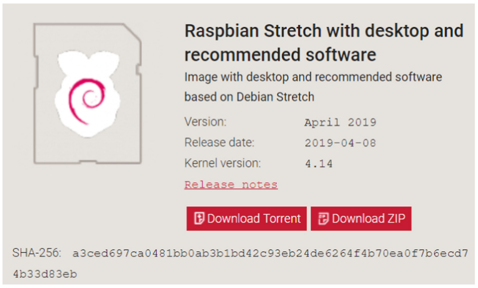
The system service model includes a Raspberry Pi central control panel and a three-part access sensor node. The Raspberry pie card is used to control the hardware. The temperature and humidity sensor DHT11 is the sensor, which is responsible for collecting the temperature and humidity of the environment. Information, and IRGree Gree air-conditioning infrared control module, IOTWebsocket home decoration lights for the execution module, when the central control board receives the user instructions forwarded by the server, performs control settings on the actuator, all data processing, data transmission, control execution programs are integrated In the Raspberry Pi card board. The Raspberry Pi card and the server system complete the two-way transmission of data and instructions.

## 4 Implementation of Small-scale Sensor Networks

The development environment of the small-scale sensor network based on the Raspberry Pi is based on the Linux Debian family Rasbian operating system. Due to the performance limitation of the Raspberry Pi, part of the compilation work is compiled by configuring the cross-compilation environment on the PC side. After compiling, the Raspberry is compiled. Run the execution file.

### 4.1 Rasbian System Burning

The Raspberry Pi development board itself comes with 1G RAM, and the ROM needs to be inserted into the SD card. That is, the Raspberry Pi development board in the initial state does not have an operating system, and it needs to be burned by itself. The system can choose to minimize the version. The minimized version itself does not have a graphical interface. It is more used in the server. The system studied in this paper is used as the central control board. The system program will output the visualization data, so the research system selects the full version. As shown in Fig. 4. After the system is burned, it can be powered on after power-on. When booting, it will initialize the system through the initialization configuration file of the boot disk, and enter the system desktop after the initialization is completed. After the time zone, the American pinyin keyboard, and the network card are set through the terminal interface, you can use the Raspberry Pi card to program (Fig, 5).



Raspbian Stretch with desktop and recommended software

Image with desktop and recommended software based on Debian Stretch

| | |
|---|---|
| Version: | April 2019 |
| Release date: | 2019-04-08 |
| Kernel version: | 4.14 |

Release notes

SHA-256: a3ced697ca0481bb0ab3b1bd42c93eb24de6264f4b70ea0f7b6ecd7 4b33d83eb

**Fig. 5.** Rasbian complete system

## 4.2 Cross Compilation Environment Configuration

The Raspberry Pi is a card-type single-chip microcomputer, and its performance is limited. Therefore, the compilation of some code is performed by the PC. The executable file compiled in another system environment by one system is called cross-compilation [5]. In the development process of the sensor network system studied in this paper, in order to save resources and time, the Raspberry Pi development environment is configured in the PC-side UBUNTU18.04 system, and the Raspberry Pi official cross-compilation tool chain is used for configuration.

## 4.3 Websocket Long Connection Implementation

In the sensor network system studied in this paper, the Websocket protocol is used for information transmission, and the custom class of the system is packaged on the basic websocket-client class to achieve stable long connection and complete bidirectional information transmission to the server.

### 4.3.1 Heartbeat Mechanism

In the provisions of the Websocket protocol, the heartbeat mechanism is used to keep the long connection. In the two forms provided by it, the second mode is selected: the connection layer is customized based on the application layer. The custom connection package adopts the JSON format. When the client establishes a Websocket connection, it sends a JSON package to apply for a connection. After the server receives the key value, the server matches the verification value. If the verification is passed, the server sends a "HeartBeating" to simulate the heartbeat packet. After that, the server sends a heartbeat packet to the Raspberry Pi every 3 s. The customized heartbeat function of the Raspberry Pi is started every 6 s after receiving the heartbeat packet sent by the server. If it is within 6 s again, After receiving the heartbeat packet of the server, it determines that the connection between the two parties is stable. If it is not received, it determines that the current connection has been disconnected, and needs to call the reconnect module to re-initiate the connection.

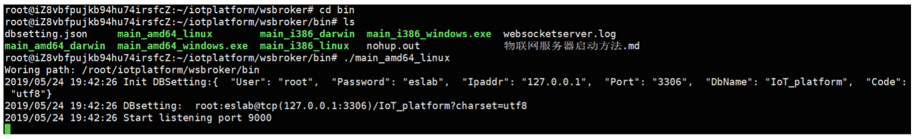
### 4.4 Sensor Node Resolution

The sensor node uses DHT11 for information collection and continuously sends temperature and humidity information to the server. This system was developed using the Python language to programmatically read data from sensors. The encoding process is performed according to the timing diagram, using a single-bus bidirectional serial communication protocol, each time the acquisition is initiated by the start signal, and then the DHT11 sends a response to the Raspberry Pi and begins transmitting 40-bit data frames, with the high order first. The data format is: 8 bit humidity integer data plus 8 bit humidity decimal data plus 8 bit temperature integer data plus 8 bit temperature decimal data plus 8 bit check digit, the temperature and humidity fractional part defaults to 0, that is, the data collected by the Raspberry Pi is Integer, the check digit is 4 bytes of data plus the result of the lower 8 bits of data as a checksum. If the final result does not match the checksum, there is a problem in the data processing. In this research system, all data for verification failure is not sent to the server, and the data is defined as "BrokenData" in the Raspberry Pi.

### 4.5 System Function Test

After the development is completed, the system studied in this paper has carried out the final functional test. The following is a detailed step and a physical test situation diagram:

(1) Start the server and listen for the specified Websocket port. The startup process is shown in Fig. 6.



```
root@iZ8vbfpujkb94hu74irsfcZ:~/iotplatform/wsbroker# cd bin
root@iZ8vbfpujkb94hu74irsfcZ:~/iotplatform/wsbroker/bin# ls
dbsetting.json     main_amd64_linux        main_i386_darwin  main_i386_windows.exe  websocketserver.log
main_amd64_darwin  main_amd64_windows.exe  main_i386_linux   nohup.out              物联网服务器启动方法.md
root@iZ8vbfpujkb94hu74irsfcZ:~/iotplatform/wsbroker/bin# ./main_amd64_linux
Woring path: /root/iotplatform/wsbroker/bin
2019/05/24 19:42:26 Init DBSetting:{  "User": "root",  "Password": "eslab",  "Ipaddr": "127.0.0.1",  "Port": "3306",  "DbName": "IoT_platform",  "Code":
 "utf8"}
2019/05/24 19:42:26 DBsetting:  root:eslab@tcp(127.0.0.1:3306)/IoT_platform?charset=utf8
2019/05/24 19:42:26 Start listening port 9000
```

**Fig. 6.** Server startup

(2) Power on the Raspberry Pi and connect peripherals such as the display via USB.
(3) Connect DHT11 temperature and humidity sensor, connect with DuPont line, connect DHT11 VCC pin to Raspberry Pi pin 1, GND to Raspberry Pi pin 6, realize 3.3 V power supply of DHT11 sensor, data The pin is connected to the Raspberry Pi pin 7. The above Raspberry Pi pin connections are all based on the WiringPi pin position. The connection status is shown in Fig. 7.

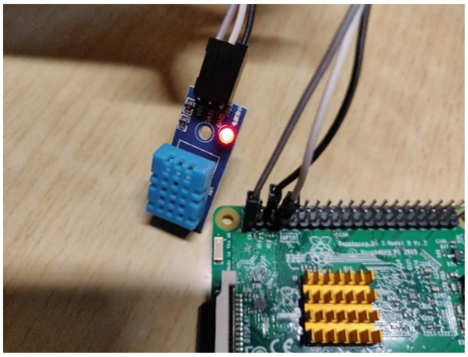
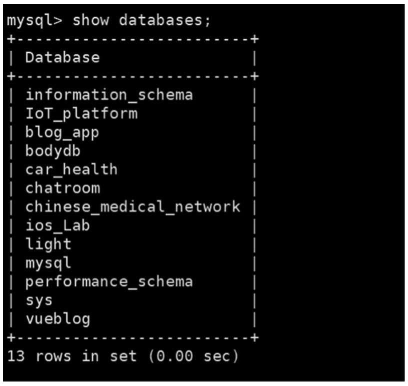


**Fig. 7.** DHT11 and Raspberry Pi connection

(4) Run the program, the server verifies that the device ID is successful, and sends the heartbeat packet to the Raspberry Pi. After receiving the information, the Raspberry Pi prints the successful connection information on the external display and continues to print the connection. Heartbeat information to show whether the current connection status is normal.

(5) The Raspberry Pi sensor network system maintains a stable and long connection between the server and the server. The server continuously sends a heartbeat packet to the Raspberry Pi. The Raspberry Pi receives the heartbeat packet from the server and detects the connection status. The heartbeat information is printed on the external display, allowing the user to check whether the current system is running normally and print heartbeat information [6].

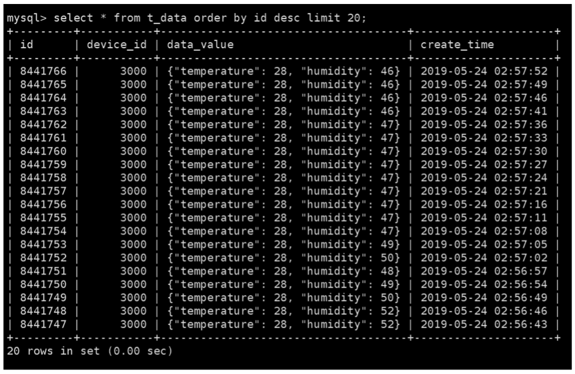
(6) After receiving the message sent by the sensor network, the server stores the data in the MySQL database. All databases of the system in the server are shown in Fig. 8.



```
mysql> show databases;
+--------------------------+
| Database                 |
+--------------------------+
| information_schema       |
| IoT_platform             |
| blog_app                 |
| bodydb                   |
| car_health               |
| chatroom                 |
| chinese_medical_network  |
| ios_Lab                  |
| light                    |
| mysql                    |
| performance_schema       |
| sys                      |
| vueblog                  |
+--------------------------+
13 rows in set (0.00 sec)
```

**Fig. 8.** Database design in the server

(7) After selecting the t_data data table, you can view the collection information of all sensing devices, including the device ID number, sensor data in JSON format, and data acquisition time. The specific data is shown in Fig. 9.



```
mysql> select * from t_data order by id desc limit 20;
+---------+-----------+-------------------------------------+---------------------+
| id      | device_id | data_value                          | create_time         |
+---------+-----------+-------------------------------------+---------------------+
| 8441766 |      3000 | {"temperature": 28, "humidity": 46} | 2019-05-24 02:57:52 |
| 8441765 |      3000 | {"temperature": 28, "humidity": 46} | 2019-05-24 02:57:49 |
| 8441764 |      3000 | {"temperature": 28, "humidity": 46} | 2019-05-24 02:57:46 |
| 8441763 |      3000 | {"temperature": 28, "humidity": 46} | 2019-05-24 02:57:41 |
| 8441762 |      3000 | {"temperature": 28, "humidity": 47} | 2019-05-24 02:57:36 |
| 8441761 |      3000 | {"temperature": 28, "humidity": 47} | 2019-05-24 02:57:33 |
| 8441760 |      3000 | {"temperature": 28, "humidity": 47} | 2019-05-24 02:57:30 |
| 8441759 |      3000 | {"temperature": 28, "humidity": 47} | 2019-05-24 02:57:27 |
| 8441758 |      3000 | {"temperature": 28, "humidity": 47} | 2019-05-24 02:57:24 |
| 8441757 |      3000 | {"temperature": 28, "humidity": 47} | 2019-05-24 02:57:21 |
| 8441756 |      3000 | {"temperature": 28, "humidity": 47} | 2019-05-24 02:57:16 |
| 8441755 |      3000 | {"temperature": 28, "humidity": 47} | 2019-05-24 02:57:11 |
| 8441754 |      3000 | {"temperature": 28, "humidity": 47} | 2019-05-24 02:57:08 |
| 8441753 |      3000 | {"temperature": 28, "humidity": 49} | 2019-05-24 02:57:05 |
| 8441752 |      3000 | {"temperature": 28, "humidity": 50} | 2019-05-24 02:57:02 |
| 8441751 |      3000 | {"temperature": 28, "humidity": 48} | 2019-05-24 02:56:57 |
| 8441750 |      3000 | {"temperature": 28, "humidity": 49} | 2019-05-24 02:56:54 |
| 8441749 |      3000 | {"temperature": 28, "humidity": 50} | 2019-05-24 02:56:49 |
| 8441748 |      3000 | {"temperature": 28, "humidity": 52} | 2019-05-24 02:56:46 |
| 8441747 |      3000 | {"temperature": 28, "humidity": 52} | 2019-05-24 02:56:43 |
+---------+-----------+-------------------------------------+---------------------+
20 rows in set (0.00 sec)
```

**Fig. 9.** Sensor data in the server database

### 4.6 Analysis of System Test Results

Based on the new network protocol technology, this paper proposes a design and implementation process of a small-scale sensor network system, and provides relevant theoretical basis as a support. On the basis of theory, through the analysis and combing of the application scenarios, the system has been designed for functional design, architecture design, and the overall operation process of the system. After the design is completed, actual development is carried out and basic robustness testing is performed. In summary, the sensor network system implements a full-duplex communication mode and guarantees a long-lived connection and disconnection reconnection mechanism to achieve the expected experimental goals.

## References

1. Luo, S.: The development of the Internet of Things industry and the trend of technological innovation. Inf. Commun. Technol. Cont. Unit **12**(04), 4–8 (2018)
2. Gao, F., Wen, H., Zhao, L., et al.: Design and optimization of a cross-layer routing protocol for multi-hop wireless sensor networks. In: International Conference on Sensor Network Security Technology & Privacy Communication System, pp. 16–87. IEEE (2013)
3. Hou, L., Zhao, S., Li, X., et al.: Design and implementation of application programming interface for Internet of Things cloud. Int. J. Netw. Manag. 57–58 (2016)
4. Khan, I., Belqasmi, F., Glitho, R., et al.: Wireless sensor network virtualization: a survey. IEEE Commun. Surv. Tutorials **18**(1), 553–576 (2017)
5. Kamoshida, Y.: Simplifying install-time auto-tuning for cross-compilation environments by program execution forwarding. In: IEEE International Conference on Parallel & Distributed Systems, pp. 15–97. IEEE (2013)
6. Park, S., Kang, J., Park, J., et al.: One-bodied humidity and temperature sensor having advanced linearity at low and high relative humidity range. Sens. Actuators B (Chem.) **76**(1–3), 322–326 (2001)