J Supercomput (2018) 74:4451–4469
https://doi.org/10.1007/s11227-017-2010-1

High-performance fieldbus application-specific integrated circuit design for industrial smart sensor networks

Ching-Han Chen[1] · **Ming-Yi Lin**[1] ·
Xing-Chen Guo[1]

Published online: 20 March 2017

Abstract Smart sensor networks (SSNs) play a crucial role in an industrial internet of things (IIoT). SSNs are also a key enabling technology for future IIoT. The main challenge in designing these networks is coping with the reliability and real-time communication limitations that characterize the network architecture. In this manuscript, first, a master/slave SSN architecture based on the high-performance Modbus communication protocol was proposed. An SSN with this architecture consists of a master controller and distributed slave sensors. Next, a smart sensor management engine that enables the control of sensor node addition, insertion, removal, automatic configuring, error processing in, and ID management of the sensor network was designed for the SSN master controller. Finally, two high-performance Modbus application-specific integrated circuits were developed for the SSN master and SSN slave. The proposed SSN solution improves the SSN performance 100 times compared with the conventional software solution. Moreover, the smart sensor management engine embedded in the master facilitates superior re-configurability, maintainability, and reliability of the SSN. Experimental results show our high-performance Modbus ASIC can satisfy the requirements of industrial and smart sensor IoT.

Keywords Fieldbus · Modbus · Application-specific integrated circuit (ASIC) · Smart sensor network

✉ Ming-Yi Lin
pk4881@gmail.com

Ching-Han Chen
pierre.miat@gmail.com

Xing-Chen Guo
gxchen007@gmail.com

1 Department of Computer Science and Information Engineering, National Central University, Taoyuan City, Taiwan

1 Introduction

In future years, the industrial internet of things (IIoT) is expected to bridge diverse technologies to enable industrial manufacturing by connecting smart objects together in support of intelligent big data analysis and optimal decision making. The current revolution in smart sensors, reliability T2T (Thing to Thing) communication technologies, and real-time big data collection can be seen as the first phase of the IIoT.

The smart sensors have all the ingredients to enable IIoT. Central to the functionality and utility of the IIoT are smart sensors embedded in diverse automation equipment. Such smart sensors are capable of detecting signals in a specific physical quantity, communicating the physical signal data via a gateway to the cloud server, and receiving the control command from the gateway or communicating with other smart things.

The basic premise is to have smart sensor collaborating directly without human intervention. For SSN (smart sensor network) system installed in unattended, remote, and harsh environments, a reliable and stable smart sensor management gateway is required between the end device and the cloud server for enabling intelligent big data analysis and intelligent decision making.

IIoT embraces smart things that interoperate and communicate with one another. T2T communication of IIoT is the advanced version of M2M communication, where each object connects with another object, without human involvement. As the number of interconnected objects grows, the development of efficient T2T communication protocols becomes increasingly critical to support IIoT operations.

As more smart objects are connected to the IIoT, huge data are collected from them in order to perform computations, make intelligent decisions, and control. Real-time big data collection refers to these large big data sets that need to be computed, collected, stored, queried, analyzed, and generally managed in order to deliver on the promise of the IIoT. Further compounding the technical challenges of big data is the fact that industrial IoT systems must deal with not only the big data collection from smart objects, but also reliable real-time requirement is needed.

Sensor network systems have been considered promising for monitoring and measurement applications in both industrial and environmental fields [1,2] where real-time and reliable communication is a fundamental requirement. Examples of such applications include industrial manufacturing, environmental monitoring, emergency response, critical infrastructure protection, and medical care [3–5]. An industrial manufacturing network is a system of interconnected equipment used to monitor and control physical equipment in industries such as mining [6]. The network involves an automatic continuous process for optimizing control processes and monitoring systems and differs considerably from traditional industrial networks because of specific requirements for its operation. The industrial manufacturing network is expected to monitor the performance of a dynamic process to ensure that the process is within limits imposed. The sensor in an environmental monitoring system can emit certain forms of physical signals that reveal its identity. Sensors are required to sample physical environments periodically according to specifications (e.g., detecting a drop in the voltage or a rise in ambient temperature), and sensor outputs are the only objective source of information for decision making. Ensuring reliable real-time message exchange is a major problem [7] such as battlefield monitoring and security surveillance, since message exchange requires the delivery of emergency response messages by ensuring the prompt establishment of the point-to-point link in real-time applications. In the absence of such prompt link establishment, the emergency response information is useless or even harmful because the sensor data do not reflect the current physical status, resulting in misleading information being conveyed.

In a real-time network, the communication protocol paradigm, sensor hardware configuration, and management mechanism play a crucial role in most real-time applications. The fieldbus communication protocols (Profibus, Modbus, HART, LonWorks, and CAN), also known as industrial network protocols [8], enable reliable communication. Numerous studies on the capability of fieldbus networks to cope with real-time requirements have been presented, such as the studies of Zhang [6] and Tovar and Vasques [9] on CAN; the study of Li et al. [10] and Salt et al. [11] on Profibus; and the study of Ferrari et al. [12] and Ferrari et al. [13] on HART. The advantages of a fieldbus include low cost, easy implementation, and distributed control; moreover, it lacks a sophisticated bus line and transmission on the serial port is asynchronous. Nevertheless, most fieldbus standards are not uniform, resulting in difficulties in connecting the bottom components such as sensors and actuators in industrial control networks [14].

Recently, Wang and Li [15] designed an online train condition monitoring network based on an industrial fieldbus. Gaitan et al. [16] developed an IoT (Internet of Things) architecture in a real-time industrial environment by using the Modbus protocol. Despite considerable research having been conducted in the field of sensor networks, there are several problems that have not been addressed, such as the difficulty in adopting the functionality of smart sensors in the industrial automation domain. The smart sensor has the features of self-calibration, self-compensation, self-validation, and self-diagnosis, and faulty sensor replacement must be considered as an integral part of the design of a monitoring or control network system [17].

Tian [18] demonstrated the viability of the concept of a fieldbus-based intelligent sensor and developed an approach that was applied to distributed measurement systems adopting the LonWorks fieldbus for achieving communication. The author described the sensor functionalities self-calibration, self-compensation, self-validation, data fusion, and comprehensive communication. The advantages of a smart sensor include low maintenance costs, minimal risk of using uncalibrated or faulty sensors, high instrument reliability, and consequently low equipment inactivity [19]. The LonWorks fieldbus [20,21] is used for the automation of various functions in the process of intelligent building, and its advantages include high scalability for numerous network topologies, easy deployment, and easy maintenance. A limitation of the LonWorks fieldbus is that it can be implemented only by using the "Neuron chip" platform hardware and the LonTalk protocol. The protocol has not been published for this implementation technique and is not appropriate for the main real-time requirement of mass data processing capability.

The objective of this manuscript was to develop an innovative smart sensor network (SSN) architecture capable of enabling remote control, online monitoring, and online measurements for enhancing the compatibility and real-time performance and reducing the power consumption of sensor nodes.

The concept of the system proposed in this manuscript is based on the fieldbus architecture characteristics desirable in a network to be used for smart sensors operating in a continuous process. A master/slave network was chosen for the fieldbus architecture because the application layer of the fieldbus protocol can be used to actualize numerous functions of smart sensor networks, such as signal extraction, communication control, node addition, node insertion, node removal, fault detection, and automatic configuring, to meet the definition. Furthermore, the smart sensors designed in the current study met the definition of intelligent sensor; in other words, they had the following functionalities: nonlinear calibration, self-compensation, self-inspection, self-validation, and self-diagnosis. The activities of this manuscript can be summarized as follows:

- The faulty sensor problem was considered for examining reliability requirements in sensor networks mostly operating in the normal mode. Fault detection verification was performed, and a multiple-error-correcting code was used to resolve the problem.
- Because the real-time applications require an emergency response, it was verified whether the smart sensor management engine (SSME) guarantees real-time message exchange for data acquisition. The sampling rate of the SSME was set to be high to achieve high real-time performance.
- The performance of the proposed network architecture was evaluated through extensive implementations.
- Sensor nodes and an SSN gateway were realized using hardware. They satisfied the smart sensor definition and provided effective and utility-oriented remote management mechanisms.

The rest of this paper is organized as follows. In Sect. 2, a fieldbus for smart sensor networks is discussed. Subsequently, the development of an application-specific integrated circuit (ASIC) design for the SSN is presented in Sect. 3, and the operation of the function modules is explained. Experimental results are presented in Sect. 4, and finally, Sect. 5 concludes the paper.

2 Fieldbus for smart sensor networks

Among all the fieldbus communication protocols, the Modbus protocol was selected because it is widely used by business establishments, supports multiple electrical interfaces and transmission media, and facilitates communication among various devices through different network types.

Compared with other fieldbus communication protocols, Modbus has advantages such as early emergence, ease of use, openness, high price–performance ratio, and high reliability. Modbus expedites industrial network deployment [22], supports numerous types of electrical interfaces and transmission media, and facilitates communications between different devices through various network types. Thus, Modbus has been widely used by business establishments.

Each device can use the Modbus protocol to start remote manipulation. The Modbus protocol based on serial connection and the Ethernet TCP/IP network enables mutual correspondence. Because the Modbus protocol is adopted, the master control module

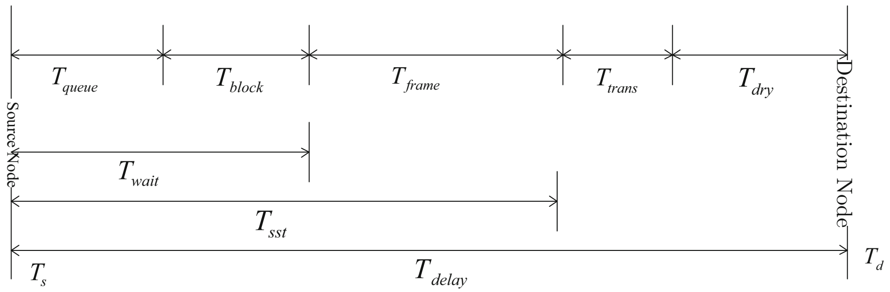

Fig. 1 Transmission delay model

realizes the slave end possessing three function codes: (0x11, Report Slave ID) (0x17, Read/Write Multiple Registers) (0x42, Change Slave ID).

The Modbus transmission mode was the high-speed, high-transmission-efficiency remote terminal unit (RTU) transmission mode that is required in all devices. In addition, the function codes and transmission baud rates were adjustable.

2.1 Transmission delay model

To achieve high real-time performance, numerous studies [23] have defined the performance evaluation index. Figure 1 illustrates a scenario of the transmission delay model; this scenario was considered in [23]. The time between the start of message transmission from source node T_{s} and the time of message reception at destination node T_{d} is defined as the transmission delay T_{delay}. The transmission delay comprises three components: total source node transmission delay T_{sst}, channel transmission delay T_{trans}, and total destination node reception delay T_{dry}.

Furthermore, the total source node transmission delay can be divided into wait time T_{wait} and net message transmission delay T_{frame}; T_{wait} is the sum of queue time T_{queue} and block time T_{block}.

$$T_{\mathrm{delay}} = T_{\mathrm{d}} - T_{\mathrm{s}} = T_{\mathrm{sst}} + T_{\mathrm{trans}} + T_{\mathrm{dry}} = T_{\mathrm{queue}} + T_{\mathrm{block}} + T_{\mathrm{frame}} + T_{\mathrm{trans}} + T_{\mathrm{dry}} \tag{1}$$

In a complete communication process, the total delay in a complete cycle of data communication among n interconnected nodes on the bus is $T_{\mathrm{delay}}^{\mathrm{sum}}$, where N_{node} is the total number of sensor nodes and $M(i)$ represents message of node i.

$$T_{\mathrm{delay}}^{\mathrm{sum}} = \sum_{i \in N_{\mathrm{node}}} \sum_{j=1}^{M(i)} T_{\mathrm{delay}}^{(i,j)} \tag{2}$$

2.2 Transmission delay model

According to [17], performance evaluation indices of fieldbus performance can be divided into bus communication capability, transmission efficiency, real time, and stability indices. Protocol efficiency describes the efficiency of a protocol in transmitting effective data. The ratio of the effective data digits in transmission data package N_{eff} to the total length of the data package N_{sum} is termed the expression of protocol efficiency P_{inf}.

$$P_{\text{inf}} = \frac{N_{\text{eff}}}{N_{\text{sum}}} \tag{3}$$

The throughput Q_{thr} represents the communication capability of the network in transmitting effective data, and it is expressed as the ratio of the effective data volume transmitted between nodes, N_{d}, to the total transmission delay required $T_{\text{delay}}^{\text{sum}}$; its unit of measurement is bits per second (bps).

$$Q_{\text{thr}} = \frac{N_{\text{d}}}{T_{\text{delay}}^{\text{sum}}} \tag{4}$$

The network efficiency P_{eff} is the ratio of T_{frame} in a complete communication process to $T_{\text{delay}}^{\text{sum}}$.

$$P_{\text{eff}} = \frac{\sum i \in N_{\text{node}} \sum_{j=1}^{M(i)} T_{\text{frame}}^{(i,j)}}{T_{\text{delay}}^{\text{sum}}} \tag{5}$$

The network utility rate P_{util} reflects the utility efficiency of the network bandwidth, and it is calculated as the ratio of $T_{\text{delay}}^{\text{sum}}$ to the total operating time T_{sum}.

$$P_{\text{util}} = \frac{T_{\text{delay}}^{\text{sum}}}{T_{\text{sum}}} \tag{6}$$

The data package loss rate P_{loss} reflects the stability of the communication system, and it is expressed using the ratio of the amount of data packages lost during the communication process N_{loss} to the overall amount of data packages (including resent packages) N_{sum}.

$$P_{\text{loss}} = \frac{N_{\text{loss}}}{N_{\text{sum}}} \tag{7}$$

3 Application-specific integrated circuit design for smart sensor networks

This section presents the design of a novel SSN communication architecture that shows superior management performance. This SSN system consists of smart sensor nodes and an SSN gateway, both of which were realized using hardware; furthermore, the system involves a client monitoring program.

In the system, Modbus hardware modules were used as the network communication infrastructure. In addition, the cyclic redundancy checks (CRCs), automatic repeat

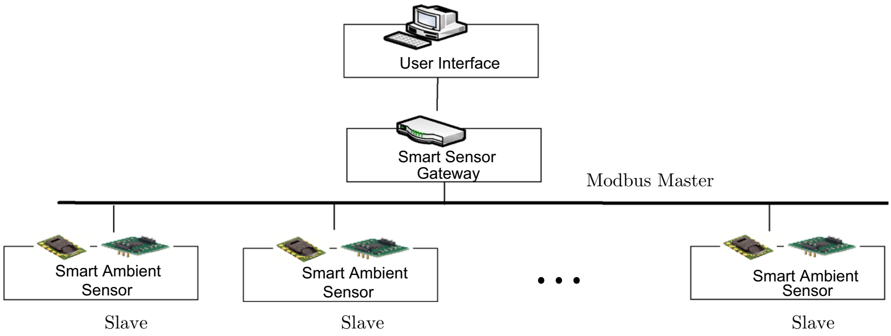

Fig. 2 Smart sensor network architecture

request, and Hamming code error correction mechanisms were adopted to improve the reliability of real-time performance communication. The function modules in the architecture were integrated into the design of a SSME, which provides an Internet access port to a sensor. Users can rapidly access or manage sensors from the host computer or through remote control. The designed SSME is a chip and consists of the main function modules. Each module was described using VHDL language.

The SSN architecture comprises two layers as shown in Fig. 2. The core is the smart sensor management gateway that controls and manages the bottom-layer smart sensors, provides the upper-layer slave network data report, and manipulates interfaces. The sensor gateway is connected to multiple smart sensors in the bottom layer through the Modbus function hardware modules.

Smart sensor nodes based on the Modbus master/slave architecture were adopted for the bottom layer of the architecture proposed in this manuscript, to measure the temperature and humidity. The use of a combination of sensor nodes and microprocessors is the simplest method of fabricating smart sensors (termed nonintegrated smart sensors). These sensors involve the integration of conventional sensors, signal conditioning circuits, and microprocessors that include bus interfaces. The sensor controllers in the sensor nodes were integrated with I^2C hardware communication modules, which enabled sensor controllers to communicate directly with most sensors available in the market, thereby increasing the compatibility between and reducing the power consumption of sensor nodes.

The upper layer of the architecture is a smart sensor gateway. The use of a communications protocol with high reliability and instantaneity in combination with sensor interface control and service frameworks in the architecture enables functions such as the addition, insertion, and removal of smart sensors, as well as fault detection and automatic configuration.

A personal computer program was used to develop a user graphical interface that helped the user perform tasks such as code configuration, parameter setting, data reading, and adding, inserting, and removing smart sensors.

In the sensor network, the industrial automation communication protocol standard Modbus was used as the standard of communications between sensor nodes and gateways. The network architecture can expand to include any number of sensors, and the sensor nodes are interconnected with a gateway through the universal asynchronous

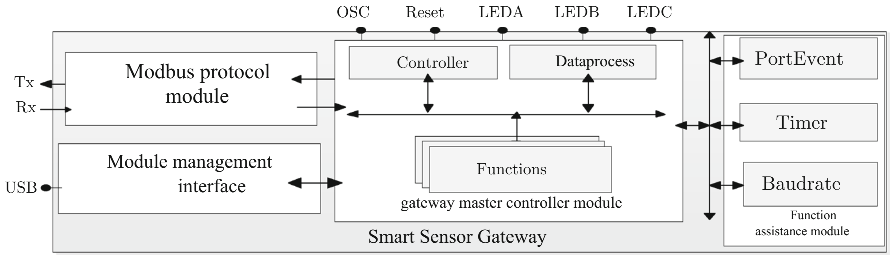

Fig. 3 Smart sensor management gateway architecture

receiver/transmitter (UART) protocol of cable or wireless media. In the UART, data error correction functions are integrated to increase the accuracy of communications. The Modbus application-layer protocol can help actualize the definition of smart sensor networks.

3.1 Smart sensor management gateway

The hierarchical architecture of the smart sensor management gateway (Fig. 3) integrates the **Modbus protocol module, function assistance module, gateway master controller module,** and **module management interface**, and it mainly controls communication with other modules and packaging processing tasks. A master control module integrates functions, such as signal extraction, mode control, node addition, node insertion, node removal, fault detection, error report generation, and automatic ID configuration, thereby satisfying the definition of SSNs.

Modbus protocol module comprises three submodules, namely a UART module and the submodules responsible for Modbus packaging and transmission. The communication module detects bus signals, receives bus data, corrects errors in Hamming codes, performs packaging according to Modbus protocol standards, checks and corrects errors, and performs cyclic redundancy checks (CRCs) on the data to be transmitted after receiving the transmission command from the master control program to generate a complete frame and determine error-correcting codes.

Modbus protocol module is responsible for transmitting and receiving signals. The UART module consists of a parallel-to-serial converter, a serial-to-parallel converter, and Hamming code submodules.

The hierarchical structure of a frame transmission module is shown in Fig. 4a, and the block diagram is depicted in Fig. 4c. The top-layer transmission module is responsible for controlling the communication among six submodules in the second layer. After the frame transmission module receives transmission signals from the master control module, the CRC code for the data to be transmitted is determined, and the checked codes are inserted in the data tail to form a complete frame. Subsequently, the *Transmit_Finite-State Machine* (FSM) is used to sequentially transmit bytes in the frame from the parallel-to-serial converter modules to the bus and to store the current state signals. The **Baudrate** and **PortEvent** submodules are **function assistance modules**. The frame transmission and reception modules are made to share the CRC module to conserve hardware resources.

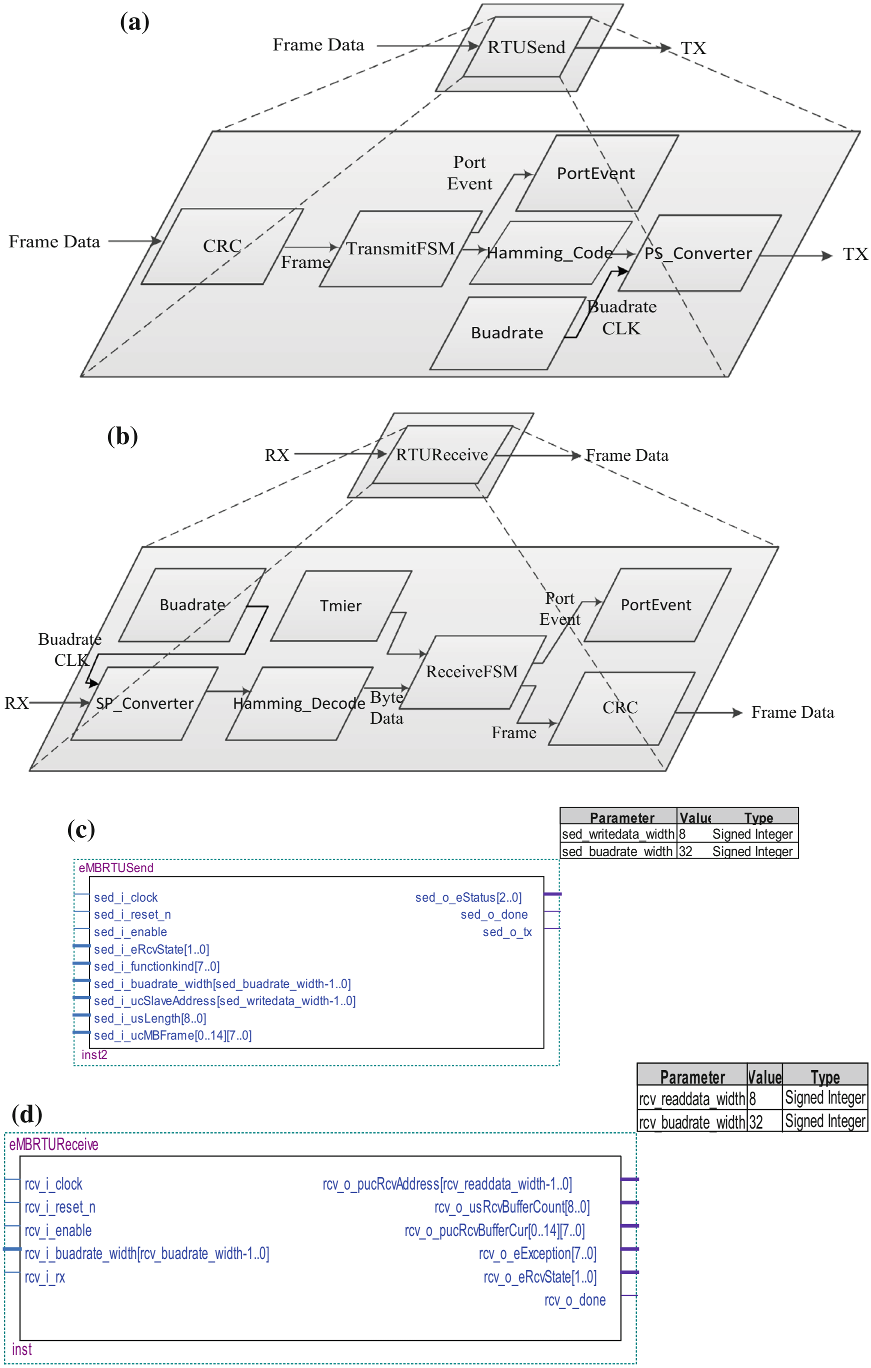

Fig. 4 Block diagram of the frame communication module: **a** hierarchical structure of frame transmission module, **b** hierarchical structure of frame reception module, **c** block diagram of frame transmission module, **d** block diagram of frame reception module

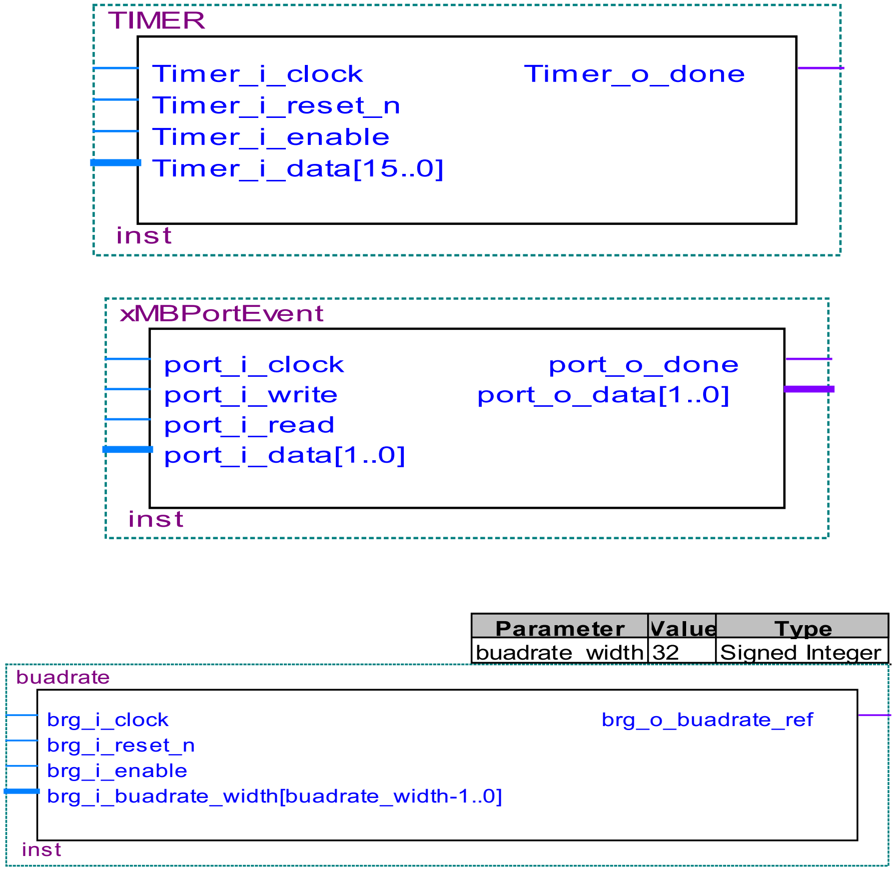

Fig. 5 Block diagram of function assistance module

The frame reception module structure is shown in Fig. 4b, and the block diagram is depicted in Fig. 4d. The reception module in the top layer is responsible for controlling the communications between seven modules in the second layer. The frame reception module receives control signals from the master control module. The data on the bus are collected through a serial-to-parallel converter submodule. Subsequently, the control module **TransmitFSM** collects the received bytes into a complete frame and sends a command to the CRC module to check for errors and correct them. If the received bytes are correct, the data are output and the current state signals are stored.

The function assistance module assists the master control module in completing specific tasks. The **Timer** submodule enables the communication modules to judge the ending signals of the Modbus frame and activate the master control module for error processing. The **PortEvent** submodule reports the control state signals of the master control module to facilitate the task of controlling the master control module and other modules. The **Baudrate** submodule is used to generate specific pulse signals for other modules. The block diagram is depicted in Fig. 5.

The **Timer** submodule provides regular triggering signals. The completion of **Timer** is determined through consistently accumulated sensed values and comparison between input parameters. The external module can set the time through the control port settings. The **PortEvent** submodule records the master control module state signals in smart sensor nodes; there are four states: preparation, frame reception, frame processing, and frame transmission. Other modules can access state signals through the control port. The **Baudrate** submodule can set the **Baudrate** through the baud rate parameter input port. The **Baudrate** submodule parameters are input to determine the accumulated counts and to output the corresponding pulse signals.

The gateway master controller module is the core of the smart sensor management gateway, and it mainly controls the communication with other modules, including the **Functions** submodule that can flexibly add and remove function codes, **controller** submodules is responsible for state transfer control, **Dataprocess** submodules responsible for data processing management and packaging processing tasks. The gateway master control module is responsible for the management and control of sensor networks as well as communication with slaves. The master control module also performs functions such as signal extraction, mode control, node addition, node insertion, node removal, fault detection, error report generation, and automatic ID configuration, thereby satisfying the definition of SSNs.

In the system, Modbus hardware modules were used as the network communication infrastructure. In addition, the cyclic redundancy check (CRC), automatic repeat request, and Hamming code error correction mechanisms were adopted to improve the reliability of real-time performance communication. The function modules in the architecture were integrated into the design of a SSME, which provides an Internet access port to a sensor.

3.2 Smart sensor node architecture

As depicted in Fig. 6, the hierarchical architecture of the smart sensor node is integrated with **gateway communication module**, **sensor device module**, **function assistance module**, and **smart sensor controller module**. The smart sensor controller module is responsible for controlling communication with other modules and packaging processing tasks, and they integrate the self-test and self-diagnosis functions. Thus, the smart sensor control modules satisfy the definition of smart sensors.

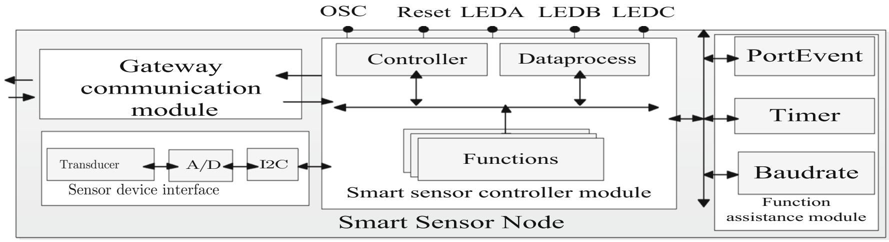

Fig. 6 Hierarchical architecture of the smart sensor node

The sensor device interface submodule comprises transducer, analog-to-digital converter (A/D) and the inter-integrated circuit (I^2C), and it is responsible for the management and sensor communication, correcting received sensor data, compensating data, and identifying error data. The $\mathbf{I^2C}$ is responsible for accessing sensor data. The **Baudrate** submodule of the **function assistance module** provides clock rate signals. The **Dataprocess** submodule processes the received sensor data, including information on nonlinear correction, self-compensation, and data diagnosis.

The hierarchical modularity design and the function module of SSN system based on reconfigurable FPGA platform can flexibly add and remove function codes that be applied for cost and fabrication time reduction. Moreover, the SSN with our proposed SSME and the gateway is made by fully hardware electric circuit without the extra CPU to avoid using the piling up software on top of controller. It not only can reduce the system power loss efficiency, but also can reduce the system computational complexity, simplified to facilitate module expansion, and modification for the future. This work attempted to use hardware methods to complete the SSN system to achieve the goal of communication reliability, minimum power consumption, and energy conservation.

4 Experiment

The system hardware architecture of the SSN consisted of **three** major modules (Fig. 7). The prototype of the SSN system constructed in this manuscript included a smart sensor node, an SSN gateway, and a PC user interface. Both the smart sensor node and SSN gateway were implemented and tested on a real field-programmable gate array (FPGA) platform. In this project, the prototypes of the SSN gateway and SSN sensor node are implemented with the resource utilization displayed in Table 1.

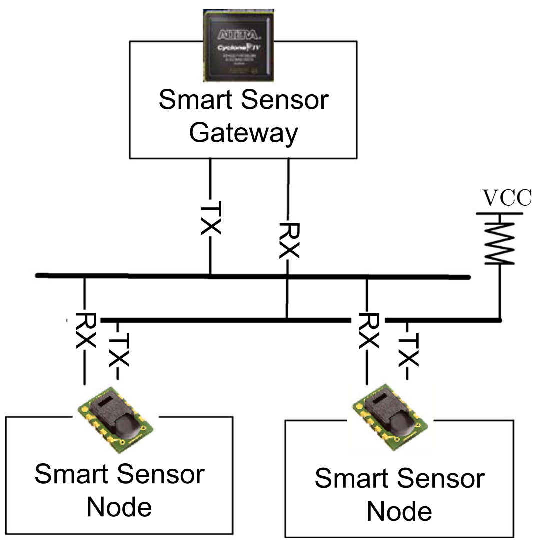

Fig. 7 Hardware architecture of the smart sensor network system

Table 1 Resource utilization of implemented for smart sensor networks

	SSN sensor node	SSN gateway
Device	Cyclone III EP3C25F256	Cyclone IV EP4CE115F29C7
System clock (Max)	143.43 MHz	149.66 MHz
Embedded memory	594 K Bits	3888 K Bits
Total logic element	2942/24,624 (12%)	3239/114,480 (3%)
Total register	2188	2417
Total pins	10/157 (6%)	12/529 (2%)
Total PLLs	1/4 (25%)	1/4 (25%)

The number in brackets indicates the device utilization

4.1 Smart sensor management master

To verify the network functions, six experiments were designed according to the predefined network functions proposed in the smart sensor management gateway.

1. Smart sensor addition and removal: After adding new nodes to the network, select any known node or added node codes at the test ID of code reset function zones.
2. Smart sensor code configuration: Select the corresponding codes at the target ID; next, update the ID in the ID reset function zone. Click code change and the results are shown. The target ID and update ID code options can be used for updating the ID according to the network state.
3. Smart sensor signal extraction: Signal extraction experiments were conducted over single-sensor nodes. Select the required target nodes in the drop-down menu of the test ID in the code reset function zone. Click on the access button and the access results are displayed.
4. Smart sensor network mode control: Smart sensor networks have three operating modes. In the general mode, the system continually accesses all sensor node data. In the power saving mode, the sensor gateway accesses all sensor node data regularly. In the timer mode, time settings of individual timing data of each sensor node are accessed. Select the timer mode at the mode settings; the corresponding access time of all node codes on the network can be set. The SSN refreshes specific sensor node data according to the set time.
5. Smart sensor fault detection: The smart sensor nodes provide 14 types of error report signals, which are shown in Table 2. The SSN system can monitor the real-time sensor node states. When the sensor on the sensor nodes is removed, error signals occur: (0x14, MB_EX_SENSOR_REEOR).

4.2 Performance evaluation

For validation purposes, SSNs as well as the Modbus fieldbus were implemented in an SSN consisting of an FPGA processor platform and Modbus hardware modules. The

Table 2 Smart sensor function codes

Function name	Code	Explanation	Note
MB_EX_NONE	0x00	No error	M/S
MB_EX_ILLEGAL_FUNCTION	0x01	Illegal function code	S
MB_EX_ILLEGAL_DATA_ADDRESS	0x02	Illegal memory address	S
MB_EX_ILLEGAL_DATA_VALUE	0x03	Illegal data	S
MB_EX_MEMORY_PARITY_ERROR	0x08	Memory error	M
MB_EX_COMMUNICATION_FAILURE	0x0C	Communication failure	M
MB_EX_NOT_CONNECT_DEVICE	0x0D	Lost device	M
MB_EX_ID_NO_INCONSISTENCY	0x0E	ID inconsistency	M
MB_EX_FUNCTION_CODE_ERROR	0x0F	Code inconsistency	M
MB_EX_FORMAT_TOO_LONG	0x10	Format too long	M/S
MB_EX_FORMAT_TOO_SHORT	0x11	Format too short	M/S
MB_EX_RESPONSE_TIMEOUT	0x12	Response time-out	M
MB_EX_CRC_ABNORMAL	0x13	CRC code error	M/S
MB_EX_SENSOR_REEOR	0x14	Sensor node error	S

M master, *S* slave

Table 3 Test table for continuous processing of single-sensor node

Testing times	Receive	Execution time (s)	Response time (μs)
10,000	10,000	1	100
50,000	50,000	3	60
100,000	100,000	4.7	47
500,000	500,000	23.2	46.4
1,000,000	1,000,000	46.4	46.4

performance criteria were the packet loss rate, execution time, retransmission rate, success rate, and finally the response time.

In a performance experiment, function tests, including the sensor power-on self-test and bus state self-test, were conducted. The overall communication performance of the sensor node was tested after confirming individual debugging of the self-diagnosed error report signals of the sensors by using serial port debugging software. Function code 23 (0x17, Read/Write Multiple Registers) of a single-sensor node was used, and tests were conducted for different numbers of consecutive transmissions under the following conditions: frame length, 15; response frame length, 9; maximum baud rate, 7.06 Mbps; number of retransmissions, 1; time-out response time, 100 μs; and frame block time, 10 μs. Individual tests for different numbers of transmissions were conducted ten times, and the mean of the execution time and the response time was determined.

The results are shown in Table 3. For different numbers of transmissions, the success rates are 100%, with a retransmission rate of 0.

Table 4 Test table for various baud rates of single-sensor nodes

Baud rates (bps)	Receive	Execution time (s)	Success rate (%)	Response time (μs)
115,200	100,000	311	100	3110
0.92M	100,000	112	100	1120
2.73M	100,000	12	100	120
6.00M	100,000	7.6	100	76
7.06M	100,000	6.1	100	61

Table 5 Test table for continuous processing of multisensor nodes

Testing times	Receive	Execution time (s)	Success rate (%)	Response time (μs)
10,000	10,000	1	100	100
50,000	50,000	3.7	100	74
100,000	100,000	7.4	100	74
500,000	500,000	36.6	100	73.2
1,000,000	1,000,000	73.8	100	73.8

Table 6 Test table of various baud rate of multisensor nodes

Baud rates (bps)	Receive	Execution time (s)	Success rate (%)	Response time (μs)
115,200	100,000	335	100	3350
0.92M	100,000	122.6	100	1226
2.73M	100,000	15.4	100	154
6.00M	100,000	8.2	100	82
7.06M	100,000	7.4	100	74

The results of testing different baud rates for 100,000 consecutive transmissions are shown in Table 4. The mean response time for the maximum baud rate is 61 μs, with a retransmission rate of 0.

When two smart sensors were used at a maximum baud rate of 7.06 Mbps, the success rates were 100%, with a retransmission rate of 0, for different numbers of consecutive transmissions (Table 5). Thus, the system satisfied the sensor network user requirements, showing the practical values of the parameters.

Different baud rates were tested for 100,000 consecutive transmissions, and the success rates were 100%, with a retransmission rate of 0 (Table 6). Because the system needs additional initialization when testing times from 10,000 to 50,000, system initialization phase accounts for a large proportion of total response time, affecting the response time of the overall system in performance experiment. The mean response time for the maximum baud rate was 74 μs, far exceeding the industrial real-time system response time requirement. Different baud rates can satisfy the requirements of different devices.

In summary, the four experimental setups in Table 3, 4, 5, and 6 of this study test two performance indices of the industrial SSN system. The first index is the reliability. For different numbers of transmissions from 50,000 to 1,000,000 testing times, the baud rate varies from 115,200 bps to 7.06 Mbps for SSN system. Experimental results show that the success rates were 100%, with a retransmission rate of 0.

The second index is the real time. After obtaining the maximum system baud rate speeds up to 7.06 Mbps, the consecutive transmissions of single-sensor node response time is only 61 μs, and multisensor nodes response time is 74 μs, respectively. With data transfer success rate achieves 100% and no data losing phenomenon.

These two indices show our high-performance Modbus ASIC can satisfy the requirements of industrial manufacturing, such as control precision, sampling speed, equipment scalability. And provided effective and efficient analog input (AI), analog output (AO), digital input (DI), and digital output (DO) function modules for diverse automation equipment, such as programmable logic controller (PLC), human machine interface (HMI), and supervisory control and data acquisition (SCADA).

4.3 Network performance analysis

A performance analysis of the smart sensor network was performed using the Modbus network performance evaluation indices proposed in Section II B. The Modbus protocol was used as the SSN communication protocol. The Modbus protocol has characteristics such as openness and interoperability, which are required for a fieldbus. However, the sensor network performance, real-time operation, and reliability in industrial applications require further exploration; the system performance indicator analysis is as follows:

- The system adopts a smart sensor gateway N_0 and two smart sensor nodes N_1 and N_2.
- Modbus RTU, baud rate 7.06 Mbps.
- The access message with sensor data function code 23 (0x17) was used as the polling message.
- The polling message comprises 15 bytes, and the response message comprises 9 bytes.
- The testing time was 1,000,000, the data packet loss rate was 0, and the system execution time was 73.8 s.

A complete communication process involves polling by using the gateway to two nodes and the reception of response messages returned from the nodes. In the Modbus protocol, at any time, a message frame is being transmitted or prepared to be transmitted. Old messages and message conflicts do not exist; thus, T_{queue} and T_{block} are 0. Because the implementation system has short communication distances, T_{trans} can be omitted. The target point T_{dry} represents the neighboring frame interval time of 1000 μs. Values obtained from (1)–(6) are shown in Table 7.

A further analysis shows that when the message length increases, the protocol efficiency, throughput and network efficiency increase. The increase and decrease in the number of sensor nodes influence only the total transmission delay of networks

Table 7 Network performance analysis

Symbol	Explanation	Value
$T_{\text{delay(req)}}$	Transmission delay time for request	41.9 μs
$T_{\text{delay(Response)}}$	Transmission delay time for response	29.1 μs
$T_{\text{frame(Req)}}$	Message sending net delay time for request	31.9 μs
$T_{\text{frame(Response)}}$	Message sending net delay time for response	19.1 μs
$T_{\text{delay}}^{\text{sum}}$	Total delay time of data communication	142 μs
$T_{\text{frame}}^{\text{sum}}$	Total delay time of data communication for T_{frame}	102 μs
P_{inf}	Protocol efficiency	40%
Q_{thr}	Throughput	2.028 Mbit/s
P_{eff}	Network efficiency	71.83%
P_{util}	Network utility	48.1%

Table 8 Network performance analysis

Parameter	ASIC	Software
Maximum baud rate	7.06 Mbps	1 Mbps
Throughput	2.028 Mbit/s	0.037 Mbit/s
Mean response time	73.8 μs	7500 μs

and have no effect on the protocol efficiency, throughput, and network parameters because the communications are processed between gateways and each sensor node.

We have also compared with existing work to validate the hardware system performance improvement, and this manuscript compared Modbus communication software system that is executed using command codes on MCU (ARM 920T) [24].

Two nodes were considered, and function code 23 (0x17) was used as the test target. The mean response time and throughput were measured when the system was executed. The comparison results are shown in Table 8. For maximum baud rates, the system performance in terms of the mean response time and throughput substantially improved after hardware implementation.

1. Real-time analysis: In high-performance synchronous motion control, the required real-time response time was lower than 100 ms for server motion control with 100 nodes. For a baud rate of 7.06 Mbps, the mean system response time was 73.8 μs, considerably lower than the 1 ms requirement; thus, the SSN meets the industrial real-time requirement.
2. Reliability analysis: On the network topology structure, according to the Modbus communication protocol, the proposed sensor network is a master/slave model. At any arbitrary time, only one master node exists on the network. For networks with N nodes, $N - 1$ channels exist. The network logic topology structure is shown in Fig. 6. Because all nodes are connected in parallel on the bus, when a problem occurs on the bus, all subsequent nodes on the bus encounter problems. However, single-sensor node errors do not affect other nodes. On the communication serial link layer, a single-byte Hamming code error correction and a mechanism com-

bining data frame CRC and automatic repeat request were adopted as innovative methods to increase the system reliability. The experimental results showed that the communication reliability of the proposed system meets the required standards.

Modbus is master/slave protocol provides a real open, flexible, and standard communication technology. It's widely used in many industrial monitoring control processes and advanced manufacturing infrastructure environment, which is not dependent on the specific type of network, due to its simplicity and high performance. Moreover, the baud rate, Modbus function modules can be easily adjusted and reconfigured, raising system flexibility and scalability.

Most of the traditional industrial automation applications, the response time requirements of digital input (DI)/digital output (DO) is from 10 to 20 ms when transmitting 1-bit, especially in high-performance synchronous motion control, require real-time response time of less than 1ms. In this work, the mean response time for the maximum baud rate of 7.06 Mbps was 74 μs, far exceeding the industrial real-time system response time requirement. Therefore, the Modbus is in line with real time and reliability of system requirements.

Moreover, we present high computational performance to illustrate how the Modbus protocols presented in the manuscript fit reliability real-time big data collection to deliver desired industrial IoT services.

5 Conclusions

Aim at the reliable, real-time big data collection via smart sensors the IIoT applications, in this paper, a fieldbus communication architecture for improving the reliability of real-time communication in SSNs is proposed, and an ASIC chip is designed according to the specifications of the proposed architecture. The proposed solution is based on SSME that involves the use of a Modbus function hardware module combined with a smart sensor management gateway and a smart sensor node. The Modbus function hardware module was designed using a novel design methodology, and it is capable of stand-alone operation and does not require a traditional CPU. The long execution time and high energy consumption of traditional CPUs are no longer acceptable in industrial applications.

The measurement results showed that the SSME of the proposed solution can increase the throughput from 0.037 to 2.028 Mbps and reduce the mean response time from 7500 to 73.8 μs, unlike software based on an ARM MCU. The obtained results attest to the efficiency of the proposed approach compared with other circuits designed for similar purposes.

Thus, the proposed sensor network system not only meets the definition of SSNs but also satisfies the reliable real-time big data collection requirement of industrial IoT.

References

1. Quang PTA, Kim D-S (2014) Throughput-aware routing for industrial sensor networks: application to ISA100. 11a. IEEE Trans Industr Inform 10:351–363

2. Rawat P, Deep Singh K, Chaouchi H, Bonnin J-M (2013) Wireless sensor networks: a survey on recent developments and potential synergies. J Supercomput. doi:10.1007/s11227-013-1021-9
3. Chen Z, Xia F, Huang T, Bu F, Wang H (2011) A localization method for the internet of things. J Supercomput. doi:10.1007/s11227-011-0693-2
4. Ding ZM, Xu JJ et al (2012) SeaCloudDM: a database cluster framework for managing and querying massive heterogeneous sensor sampling data. J Supercomput. doi:10.1007/s11227-012-0762-1
5. Yu J, Bang H-C et al (2015) Adaptive internet of things and web of things convergence platform for internet of reality services. J Supercomput. doi:10.1007/s11227-015-1489-6
6. Zhang T (2014) Monitor and control node in coal production safety based on LonWorks Fieldbus. In: Li X (ed) Progress in mine safety science and engineering II. CRC Press, Boca Raton, pp 309–312
7. Guo ZW, Chen PP, Zhang H, Jiang MX, Li CR (2012) IMA: an integrated monitoring architecture with sensor networks. IEEE Trans Instrum Meas 61(5):1287–1295
8. Saranli U, Avci A, Öztürk M (2011) A modular real-time fieldbus architecture for mobile robotic platforms. IEEE Trans Instrum Meas 60(3):916–927
9. Tovar E, Vasques F (1999) Real-time fieldbus communications using Profibus networks. IEEE Trans Ind Electron 46(6):1241–1251
10. Li Q, Jiang J, Rankin DJ (2011) Evaluation of delays induced by profibus PA networks. IEEE Trans Instrum Meas 60(8):2910–2917
11. Salt J, Casanova V, Cuenca A, Piza R (2014) Multirate control with incomplete information over Profibus-DP network. Int J Syst Sci 45(7):1589–1605
12. Ferrari P, Flammini A, Marioli D, Rinaldi S, Sisinni E (2010) On the implementation and performance assessment of a wirelessHART distributed packet analyzer. IEEE Trans Instrum Meas 59:1342–1352
13. Ferrari P, Flammini A, Rizzi M, Sisinni E (2013) Improving simulation of wireless networked control systems based on WirelessHART. Comput Stand Interfaces 35(6):605–615
14. Flammini A, Ferrari P, Sisinni E, Marioli D, Taroni A (2002) Sensor interfaces: from field-bus to ethernet and internet. Sens Actuators A Phys 101(1/2):194–202
15. Wang Y, Li B (2014) Design and implementation of a new power transducer of switch machine. Mechatron Autom Control Syst 237:811–816
16. Gaitan VG, Gaitan NC, Ungurean I (2014) A flexible acquisition cycle for incompletely defined fieldbus protocols. ISA Trans 53(3):776–786
17. Reyes J, Marley V, Ricardo T (2014) Fault detection and measurements correction for multiple sensors using a modified autoassociative neural network. Neural Comput Appl 24(7–8):1929–1941
18. Tian GY (2001) Design and implementation of distributed measurement systems using fieldbus-based intelligent sensors. IEEE Trans Instrum Meas 50(5):1197–1202
19. Bruckner D, Picus C, Velik R, Herzner W, Zucker G (2012) Hierarchical semantic processing architecture for smart sensors in surveillance networks. IEEE Trans Industr Inform 8(2):291–301
20. Canovas S, Cugnasca C (2010) Implementation of a control loop experiment in a network-based control system with LonWorks technology and IP networks. IEEE Trans Ind Electron 57(11):3857–3867
21. Silva RM, Arakaki J, Junqueira F, Santos Filho DJ, Miyagi PE (2012) Modeling of active holonic control systems for intelligent buildings. Autom Constr 25:20–33
22. Lee DS, Liu YH, Ling CR (2012) A wireless sensor enabled by wireless power. Sensors 12:16116–16143
23. Pan Y (2007) Research of Modbus protocol and design of the experiment system. Master's thesis, Dept. of Electrical Engineering and Automation, Harbin Institute of Technology
24. Ma L, Xia F, Peng Z (2008) Integrated design and implementation of embedded control systems with scilab. Sensors 8:5501–5515