Hindawi
Complexity
Volume 2023, Article ID 9876318, 1 page
https://doi.org/10.1155/2023/9876318

Retraction

Retracted: Optimization and Simulation of Controller Area Network Communication Model Based on Industrial Internet of Things Platform

Complexity

Received 19 December 2023; Accepted 19 December 2023; Published 20 December 2023

Copyright © 2023 Complexity. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This article has been retracted by Hindawi following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of one or more of the following indicators of systematic manipulation of the publication process:

(1) Discrepancies in scope
(2) Discrepancies in the description of the research reported
(3) Discrepancies between the availability of data and the research described
(4) Inappropriate citations
(5) Incoherent, meaningless and/or irrelevant content included in the article
(6) Manipulated or compromised peer review

The presence of these indicators undermines our confidence in the integrity of the article's content and we cannot, therefore, vouch for its reliability. Please note that this notice is intended solely to alert readers that the content of this article is unreliable. We have not investigated whether authors were aware of or involved in the systematic manipulation of the publication process.

Wiley and Hindawi regrets that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our own Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

References

[1] H. Lin and L. Du, "Optimization and Simulation of Controller Area Network Communication Model Based on Industrial Internet of Things Platform," *Complexity*, vol. 2020, Article ID 8864722, 11 pages, 2020.

Hindawi
Complexity
Volume 2020, Article ID 8864722, 11 pages
https://doi.org/10.1155/2020/8864722

Research Article

Optimization and Simulation of Controller Area Network Communication Model Based on Industrial Internet of Things Platform

Haifeng Lin[1] **and Lin Du**[2]

[1]College of Information Science and Technology, Nanjing Forestry University, Nanjing 210037, China
[2]School of Information Science and Engineering, Qilu Normal University, Jinan 250200, China

Correspondence should be addressed to Lin Du; dul1028@163.com

Received 19 August 2020; Revised 27 September 2020; Accepted 30 September 2020; Published 26 October 2020

Academic Editor: Zhihan Lv

Copyright © 2020 Haifeng Lin and Lin Du. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The Internet of Things (IoT) is an information carrier based on network communication, and it can make all general physical objects which can be independently addressed from an interconnected network. Controller Area Network (CAN) is a kind of serial communication network which can effectively support distributed control or real-time control. This paper proposes the design of an IoT-oriented universal CAN bus. In order to build a complete CAN communication simulation model, message sending and message receiving models are constructed, respectively. The simulation experiment proves that the IoT-oriented network communication model designed in this paper not only has low power consumption, but it can also solve the front-end compatibility problems caused by different communication protocols.

RETRACTED

1. Introduction

Internet of Things technology has been listed as one of the emerging industries with strategic significance in the world and has developed rapidly. IoT is an object-oriented Internet. Its three major technologies, sensor technology, communication technology, and embedded system technology, correspond to its three attributes of perception, transmission, and control [1]. Among these three technologies, embedded system technology is the foundation, sensor technology is the function expansion, and the communication technology is the core connection technology of IoT [2]. The communication technology of IoT includes four types: wire transmission, wireless transmission, traditional Internet, and mobile air network. The IoT system does not need every device to be connected to the Internet; it can have one device as the control core, to which the other devices are connected through wireless communication technology and this device is connected to the Internet for communication [3]. This method can lower the application cost of IoT. CAN bus is a serial communication network that can effectively support distributed control and real-time control and it has been widely applied in the field of automatic control with its high performance and reliability. The CAN protocol has multimaster control and it also has the functions of error detection, error notification, and error recovery. All units can detect error and the unit which detects the error will notify all other units immediately [4]. Besides, CAN bus has many connection nodes, and it can be connected to the bus with multiple units. Theoretically, there is no limit for the number of connectable units. Besides, it has a fast communication speed and a long communication distance. The fastest speed is 1 Mbps (with the distance less than 40 m) and the longest distance is 10 km (with the speed slower than 5 kbps) [5]. In field bus, CAN bus has obtained extensive support of computer chip makers, who have released microcontroller unit (MCU) chip with direct CAN interface one after another. With the development of CAN bus technology, it has already become one of field buses that has been widely applied in many fields, including agriculture, robot, computer numerically controlled machine tool, medical apparatus and instruments, vehicles, vessels, and national defense [6].

The special contributions of this paper include the following:

(i) It has firstly elaborated the significance and background of this paper, analyzed the research status at home and abroad of IoT and CAN bus, and expounded the related theoretical foundation.

(ii) It has introduced the technological architecture and logical architecture of industrial equipment IoT platform, analyzed CAN real-time communication technology, and studied the network communication, socket mechanism, and breakpoint resume with reliable time sequence.

(iii) It has proposed an IoT-oriented universal CAN bus and explored some problems to which attention should be paid in application. It has also proposed the design method for CAN bus interface circuit for its highly integrated communication controller.

(iv) It has effectively simulated and constructed CAN communication, constructed the complete model structure, and conducted simulation and operation to check its effect. It has proven that the CAN communication design scheme of this paper is effective.

The remainder of this paper is organized as follows. Section 2 discusses related works, and IoT-oriented CAN communication design is outlined in Section 3. Experimental procedure is presented in Section 4. Section 5 shows the experimental simulation results, and Section 6 concludes the paper with a summary and proposed directions for future research.

2. Related Work

With the development of IoT, communication technology, and control technology, conventional industrial control field has been experiencing an unprecedented reform and the networked industrial control has expanded the development space for industrial control field and brought new development opportunities [7]. In a wide range of industrial fields, CAN bus can be taken as the field device-level communication bus and compared with other buses, which has high reliability and cost performance. This will be a main direction for CAN technology open for the development and applications of IoT [8]. CAN is a multimaster serial communication bus and according to its basic design specifications, it has a fast bit rate and high anti-electromagnetic interference, and it can detect any error. Firstly, the CAN controller works in a variety of modes, and each node in the network can adopt bit-by-bit arbitration of lossless structure and send data to the bus contention according to the bus access priority and message identifier. Besides, CAN protocol has replaced station address coding with communication data coding, enabling different nodes to receive the same data at the same time [9]. These characteristics have led to strong real-time data communication among network nodes formed by CAN bus and the easiness to constitute a redundant structure and improved the reliability and flexibility of the system. The data communication of CAN bus has outstanding reliability, timeliness, and flexibility. Due to its excellent performance and unique design, CAN bus has attracted more and more attention. Most automobile manufacturers have adopted CAN bus to realize the data communication between internal control system and various test and execution institutions. Meanwhile, because of the characteristics of CAN bus, its application is no longer restricted to auto industry, but it also develops towards IoT and other fields. CAN has become an international standard and it has already been considered as one of the most promising field buses [10, 11].

In as early as 1998, Massachusetts Institute of Technology in the United States had come up with the idea of IoT at that time, i.e., EPC system, and one year later, the concept of IoT had been deemed as the logistics network based on Radio Frequency Identification (RFID) [12]. IoT has not had a formal definition: IoT is the network that makes all objects which have and can implement independent functions achieve interconnection, until 2005 when the International Telecommunication Union (ITU) released "ITU Internet Report 2005: Internet of Things" [13]. CAN bus is one of the field buses which have been applied the most in the world. It was first launched by Bosch, a German company, and used in the data connection protocols between internal measurement and execution units. The application of CAN bus ranges from high-speed networks to low-cost multiline network and it is widely applied in the data communication between various detection and execution agencies in the control system [14]. CAN bus can mount many communication nodes, the signals between which are transmitted via the bus so as to realize between-node communication. The use of RS-485 can only constitute a master-slave structural system and the communication can only be conducted by means of master station polling. Worse still, the system has poor timeliness and reliability [15].

In theory, as long as the bus has enough load, the number of nodes is not limited, and the repeater can improve the load. CAN bus can effectively reduce the number of wires and it has flexible communication, excellent timeliness, and high reliability. However, the communication quality of CAN is related to many factors, mainly including the distance between communication nodes, the type and impedance of communication lines, crystal oscillator error of various communication nodes, and the deviation caused by external environment [16]. In practical engineering field, the preset CAN sample point may not be able to guarantee the data receiving and dispatching quality of communication nodes and there may be frame loss and receiving and dispatching abnormalities in data which affect the normal operation of product functions due to the impact of the above factors. To solve the problems in existing technology, this paper provides an IoT-oriented scheme used in CAN bus message management and the hardware design method

RETRACTED

RETRACTED

of CAN bus interface circuit of highly integrated communication controller so as to guarantee the safety of network communication [17].

3. IoT-Oriented CAN Communication Control Design

3.1. Key Issues in Interface Circuit Design. Any node in CAN can take an initiative and send a message to other nodes in the network irrespective of master and slave at any moment. It adopts nondestructive bus arbitration technology. When two nodes send a message to the network at the same time, the node with low priority voluntarily stops data transmission while the one with high priority continues to transmit the data insusceptibly. In addition, it has the functions of point-to-point, one point to multipoints, and global broadcast data transmission and reception. Every frame of message has CRC check and other error detection measures with an extremely low error rate and high reliability. When a serious mistake occurs in message transmission, the node can automatically cut its connection with the bus so that other operations in the bus are not affected [18].

3.1.1. Optoelectronic Isolation Circuit. Although optoelectronic isolation circuit can strengthen the antijamming capability of the system, it also increases the propagation delay time of CAN bus effective loop signal and decrease the communication speed or distance. CAN transceiver itself has instantaneous anti-interference, radio frequency interference (RFI), and the ability to achieve thermal protection; additionally, its current limit circuit has also provided further protection to the bus. Therefore, in case of short field transmission distance and little electromagnetic interference, optoelectronic isolation may not be adopted to make the system reach the maximum communication speed or distance and simplify the interface circuit. If optoelectronic isolation is needed in the field environment, high-speed optoelectronic isolation device should be selected to reduce the propagation delay time of effective loop signal of CAN bus. The propagation delay time is short and it is approximate to the level of TTL circuit propagation delay time [5].

3.1.2. Power Isolation. The power V_{dd} and V_{cc} used in the two sides of optoelectronic isolation device must be completely isolated. Otherwise, optoelectronic isolation will not play its due role. Power isolation can be achieved through low-power DC/DC power isolation module.

3.1.3. Bus Impedance Matching. The terminal of CAN bus must be connected with *n* resistors, which have a significant role in bus impedance matching; otherwise, it will greatly reduce the reliability and anti-interference performance of bus data communication and it may even make it impossible to carry out communication. In order to improve the antijamming capability of interface circuit, the following measures can also be taken into consideration: connect *n* parallel capacitors between CANH and CANL of the transceiver and the earth to filter the high-frequency interference in the bus and prevent electromagnetic radiation [19].

3.2. CAN Bus Synchronization Mechanism. CAN communication is an asynchronous serial communication. Its synchronization method is not achieved by providing extra clock signals but by realizing read-write synchronization through bit timing, fixed message frame structure, and hard synchronization and resynchronization operations. The condition for communication between various nodes in CAN bus network is that they have the same normal bit rate (NBR), i.e., the bits are transmitted every second. Its reciprocal is called normal bit time t_{bit} which is composed of four nonoverlapping time periods: reciprocal period, transmission period, phase buffer period 1, and phase buffer period 2. The normal bit time can be defined as follows:

$$t_{\text{bit}} = \frac{1}{\text{NBR}} = t_{\text{SyncSeg}} + t_{\text{propSeg}} + t_{\text{PS1}} + t_{\text{PS2}}. \quad (1)$$

Every period of time in normal bit time is made up of multiple time quanta TQ, the length of which depends on the frequency of oscillator f_{OSG}. The relation between them is as follows:

$$\text{TQ} = \frac{2\text{BRP}}{f_{\text{OSC}}}. \quad (2)$$

In which, BRP is the Baud rate prescale coefficient and for the device that adopts the same clock frequency of the master oscillator in CAN bus network, the normal bit rate NBR can be adjusted to the same numerical value only by setting BRP and the TQ in every time period [20, 21].

When IoT conducts wireless communication and if the transport layer has a high packet loss speed-up rate or the data is blocked, the channel protection mechanism starts automatically. Assume that $S(v_j, v_{\text{DAP}i})$ is the link communication of transmit data and v_0, $v_{\text{DAP}i}$ represents the data source and destination aggregation node. In other words, when the system accesses any aggregation point, stop the transmission if at the time of t_0 and the forward node v_i in the congestion point will find the homologous data continuously accumulated in the buffer zone and meanwhile conduct self-inspection and abandon. The backward node will not delete the link in the transport layer. So, there is no homologous data in the buffer zone. Therefore, the node can make a judgment according to whether the connection channel is congested or not so as to start the multiagent dynamic additional channel protection mechanism [22, 23].

With regards to protection mechanism, the node goes along the transmission channel in this technology. During the traverse, the proxy includes two parameters in composite measurement: the vacancy and surplus bandwidth rate of the receiving node. In the composite measurement, the corresponding equation to calculate the queue occupancy rate of the receiving node is as follows:

$$\mu_k(t) = 1 - \frac{q_k - \int_{t_{0-}}^{t_0} v_K^1(t)\mathrm{d}t + \int_{t_0}^{t_{0+}} v_K^x(t)\mathrm{d}t}{B_k} + \zeta. \quad (3)$$

In which, μ_k is the receiving node, v_k is the vacancy of queue, q_k is the current queue length of node v_k, and $v_k^1(t)$ and $v_k^r(t)$ represent the velocities for communication buffer zone of the node at t phase to send and receive grouping, respectively. ξ is the buffer zone of the queue and it is used to control the transmission of "flow" over message and $\xi \in [0.01, 0.08]$. B_k represents the size of the communication buffer zone of node v_k [24].

The composite measurement of proxy travel restoration is defined as follows:

$$M(e_{hk}) = a\mu(t) + b\gamma(e_{hk}) + 0.5. \quad (4)$$

In which, α and β represent weight coefficient and $a + b = 1$. It controls the transmission of real-time message through low a/b rate as data flow will generate low jitter during the transmission. So, the rate shall increase properly.

There are two main types of CAN physical layer. The CAN communication network in Figure 1 is a high-speed, short-distance "closed-loop network" that complies with the ISO11898 standard. Its maximum bus length is 40 m, and its communication speed is up to 1 Mbps. Each end requires a 120 Ω resistor [25].

Figure 2 is a low-speed, long-distance "open-loop network" that follows the ISO11519-2 standard. Its maximum transmission distance is 1 km, and its maximum communication rate is 125 kbps. The two buses are independent and do not form a closed loop. Each bus is required. There is a 2.2 kΩ resistor in series on each [26].

4. Experimental Procedure

4.1. Create the Message Transmission Model. This section is about creating the part which transmits CAN message in the model. It sets and connects with each module by using the relevant modules in Simulink library.

4.1.1. Create the Model. Start Simulink in MATLAB software platform, select "Blank Model" → "Create Model," and complete the creation of a new blank model window.

4.1.2. Add Related Modules. Click the button of "Library Browser" in the toolbar and open Simulink module library. In the left side is the available module library. Select the related modules from "Vehicle Network Toolbox" → "CAN Communication" and use them in the transmission of CAN communication in this instance.

Move the modules of "CAN Configuration," "CAN Pack," and "CAN Transmit" to the editing interface, respectively, and create one instance for each of them. The related module descriptions are shown in Table 1.

Select different types of modules as data source. In this paper, constant is used as the data source to verify the CAN communication model.

4.1.3. Connect Modules. The above modules are connected through wires: the output port of Constant is connected with the input port of CAN Pack, the output port of which is connected to the input port of Transmit and CAN Configuration is not connected with any other modules. In this module configuration, CAN channel is used for communication.

4.1.4. Set Module Parameters. Double click CAN configuration to configure its parameters. The specific parameter description is shown in Table 2.

Double click CAN Pack to configure its parameters, the specific descriptions of which are shown in Table 3.

Double click CAN Transmit to configure its parameters, the specific description of which can be found in Table 4.

The model structure after various modules are connected and configured in the transmission part is shown in Figure 3.

4.2. Create the Model to Receive Message. This section is used to create the part which receives CAN message in the model. It sets and connects various modules by using related modules in Simulink library.

4.2.1. Add Related Modules. Select related modules from "Vehicle Network Toolbox" → "CAN Communication" and use them to receive CAN communications in this instance. Select the module of "Simulink" → "Sinks" → "Scope" to display the data received.

Move the modules: "CAN Configuration," "CAN Receive," "CAN Unpack," and "Scope" to the editing interface, respectively, and create one instance for each. The related module descriptions are shown in Table 5.

4.2.2. Connect Modules. Create CAN message pack subsystem with the module of "Function-Call Subsystem" in "Ports & Subsystems" module library and move CAN Unpack to this subsystem.

In CAN Receive, CAN Msg (output port) is connected with In1 (input port) of CAN Unpack subsystem [27]. In CAN Receive, *f*() (output port) is connected with function() (input port) of CAN Unpack subsystem and in CAN Unpack subsystem, CAN Msg(output port) is connected with the input port of scope [28]. CAN Configuration1 is not connected with any other modules and it configures CAN channel for communication [29, 30].

4.2.3. Set Module Parameters. Double click CAN Configuration1 to configure its parameters and their descriptions are indicated in Table 6.

Double click CAN Receive to configure its parameters. The related descriptions of these parameters are demonstrated in Table 7.

Double click CAN Unpack to configure its parameters. Refer to CAN Pack for the related parameters' description.

The model structure of the reception part after various modules are connected and configured is shown in Figure 4.

RETRACTED

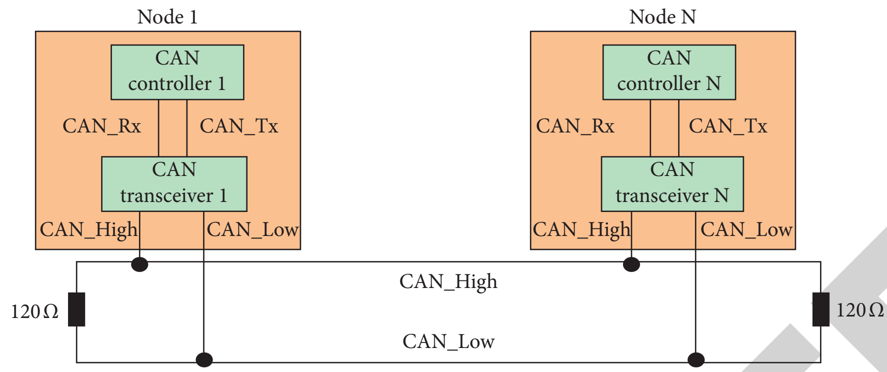


Figure 1: CAN closed-loop bus communication network.

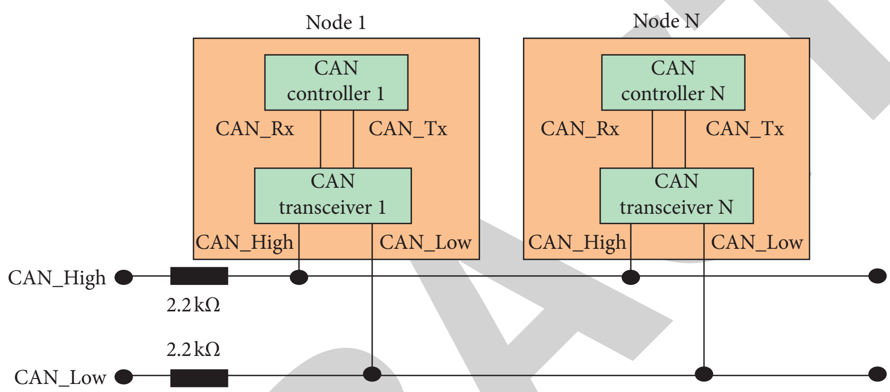


Figure 2: CAN open-loop bus communication network.

Table 1: Description of modules in CAN communication transmission.

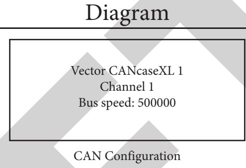
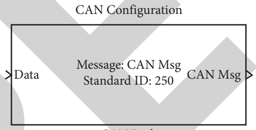
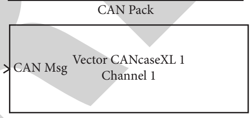
Module	Diagram	Description
CAN Configuration		It is used to configure the parameters of CAN device which is used in transmitting and receiving message and it uses one CAN Configuration module to configure every device that sends and receives message in the model.
CAN Pack		It is used to upload signal data at a designated interval to a message during the simulation. The number of module inputs depends on the dynamic adjustment of the number of specified signals. It has an output port.
CAN Transmit		It is used to send and transmit data to the virtual CAN channel, and it can transmit a single message or message array. There is no output port in this module.

Table 2: Description of parameters of CAN Configuration.

Parameter	Value	Description
Device	MathWorks Virtual 1 (Channel 1)	Select from list CAN device and its necessary channel. The device is used to transmit messages.
Bus speed	500,000	Set the bus speed attribute for the device selected above with bits per second as the unit.
Acknowledge mode	Normal	Check whether the designated channel is in normal mode (normal) or silent mode (silent). If normal, the channel can normally both receive and transmit message while if silent, the channel can only receive message.

RETRACTED

Table 3: Description of parameters of CAN pack.

Parameter	Value	Description
Data is input as	Raw data	It selects the input signal. The raw data input the data in the form of uint8 vector array; manually specified signals can assign the definition of data signal; and CANdb specified signals can assign the CAN database file which includes the definitions of message and signal.
Name	CAN Msg	The name of CAN message can be assigned with the default value of CAN Msg.
Identifier type	Standard (11-bit identifier)	It assigns whether the identifier of CAN message is standard or extended.
Identifier	500	It assigns the ID of CAN message. For standard identifier, the figure must be any positive integer from 0 to 2047; and for extended identifier, it must be any positive integer from 0 to 536870911.
Length (bytes)	8	It assigns the length of CAN message, the scope of which ranges from 0 to 8 bytes.

Table 4: Description of parameters of CAN transmit.

Parameter	Value	Description
Device	MathWorks Virtual 1 (Channel 1)	It selects the CAN device and channel which are used to transmit CAN message to the network.

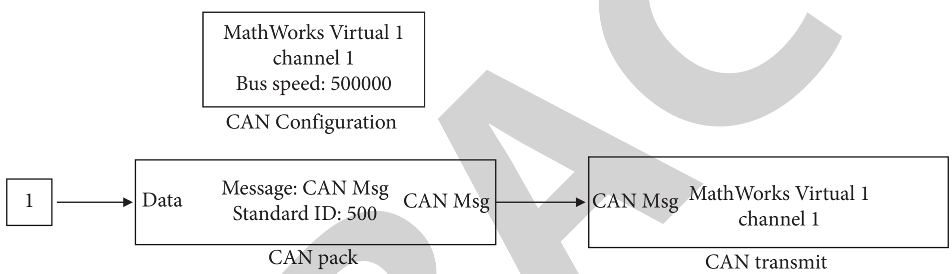


Figure 3: Transmission part model structure.

Table 5: Description of modules in the reception part of CAN communication.

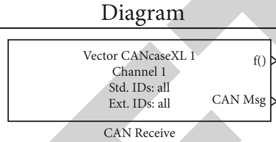
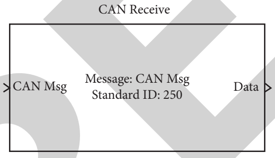
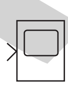
Module	Diagram	Description
CAN Receive		It is used to receive messages from CAN and transmit them to Simulink model. In this module, there are two output ports: *f*() is a trigger to function-call subsystem and CAN Msg includes the CAN messages received during the specific timestep.
CAN Unpack		It is used to unpack CAN message by using designated output parameters in every timestep into signal data and the data is output as a single signal.
Scope		It is used to display time-domain signal.

Table 6: Description of parameters of CAN Configuration1.

Parameter	Value	Description
Device	MathWorks Virtual 1 (Channel 2)	Select from list CAN device and the necessary channel and use this device to receive messages.

4.3. Operate the Model. Select in the Toolbar the button of "Run" and click it to simulate. When operating simulation, CAN Transmit will obtain message from CAN pack. Then it will transmit the message through virtual channel 1 and the CAN Receive in virtual channel 2 will receive this message and transmit it to CAN Unpack subsystem to unpack the message. Finally, double click scope to check the transmission result.

RETRACTED

Table 7: Description of parameters of CAN Receive.

Parameter	Value	Description
Device	MathWorks Virtual 1 (Channel 2)	It is used to select CAN device and channel to receive CAN message from this device.
Sample time	0.01	It is used to assign the sample time of modules during the simulation and it is defined as the operating frequency of CAN receive during this time.
Number of messages received at each timestep	All	It is used to select the number of messages the module receives at each designated timestep. All is to transmit all messages in the buffer zone to model and 1 is to transmit one message to the model.

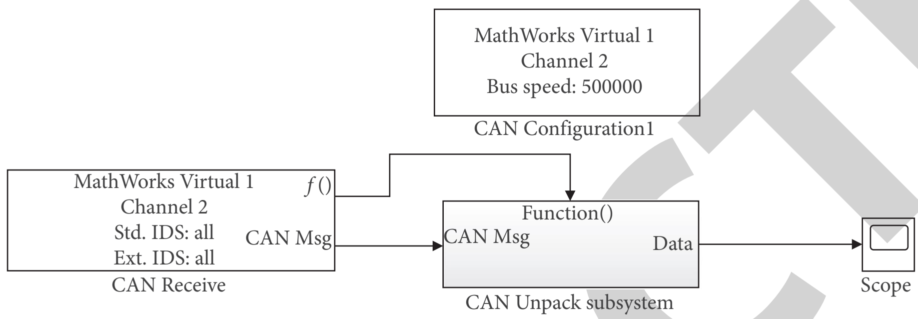


Figure 4: Receiving part model structure.

5. Simulation Experiment and Results Analysis

After the above links, the model structure which includes such links as transmission, reception, and operation are basically built. In this section, different types of modules will be selected as data sources so as to verify the CAN communication Simulink model.

Use "Constant" as the data source and add a Constant to the model from Simulink → Sources. Constant generates real-number or plural constant-value signal, and it is used to provide the input of constant signal. Among them, the parameter of "Constant value" is set as [1 2 3 4 5 6 7 8]. The final output result of CAN communication is shown in Figure 5.

Use "Sine Wave" as the data source and add a Sine Wave to the model from Simulink → Sources library. Sine Wave takes simulation time as the time source to generate sine wave and outputs sinusoid wave. The equation of wave function is

$$y = \text{amplitude} \times \sin(\text{frequency} \times \text{time} + \text{phase}) + \text{bias}. \tag{5}$$

In this instance, the value of each parameter is set as follows: amplitude is 1, bias is 200, frequency is 1, and phase is 0. The final output result of CAN communication is shown in Figure 6.

Use "Repeating Sequence" as the data source and add a Repeating Sequence to the model from Simulink → Sources library. Repeating Sequence is used to generate periodic signal of any shape and it can output periodic scalar signal with wave shape designated by parameters: time values and output values. Among them, Time values assign the output time vector and Output values specify the corresponding signal amplitude vector to output time. In this instance, the values of various parameters are set as follows: Time values within [0 5], and Output values within [0 100]; in other words, starting from simulation, repeat one sawtooth waveform with the maximum amplitude of 100 every 5 seconds. The final output result of CAN communication is indicated in Figure 7.

Use "Step" as the data source and add a Step Module to the model from Simulink → Sources library. Step is used to generate step function and it can provide 2 steps which can define level during the designated time. If the simulation time is less than the parameter value of Step time, the parameter value of Initial value will be the output and if it is bigger than or equal to Step time, the output is the parameter vale of Final value. In this instance, various parameters are set as follows: step time is 15, initial value is 5, and final value is 100. The final output result of CAN communication is shown in Figure 8.

Use "Pulse Generator" as the data source and add a module of Pulse Generator to the model from Pulse Generator library. Pulse Generator is used to generate square-wave pulse at a fixed interval and its waveform parameters, Amplitude, Pulse Width, Period, and Phase delay, and determine the shape of output waveform. In this instance, the values of various parameters are set as follows: phase delay is 0, amplitude is 100, pulse width is 20, and period is 5. The final output result of CAN communication is shown in Figure 9.

Take "Ramp" as the data source and add a "Ramp" module to the model from Simulink → Sources library. Ramp is used to generate the signals that constantly increase or decrease, and it can generate the signal which changes at a designated speed starting from designated time and value. Its parameters of slope, start time, and initial output decide the features of output signal. In this instance, the values of these

RETRACTED

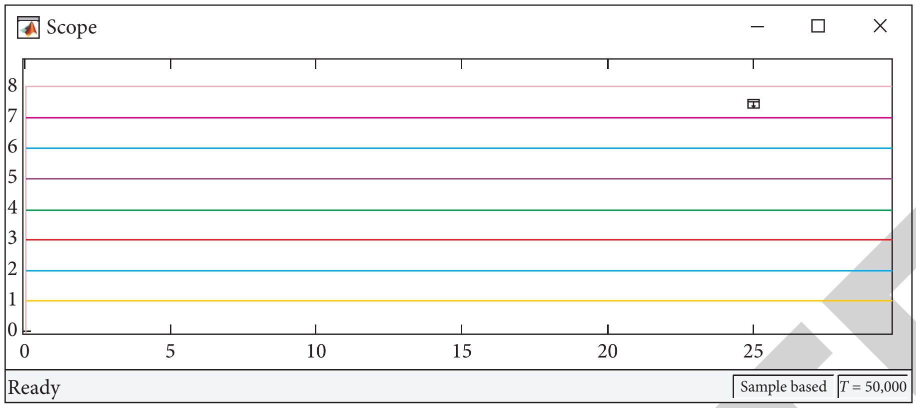


Figure 5: Output result of CAN communication in constant.

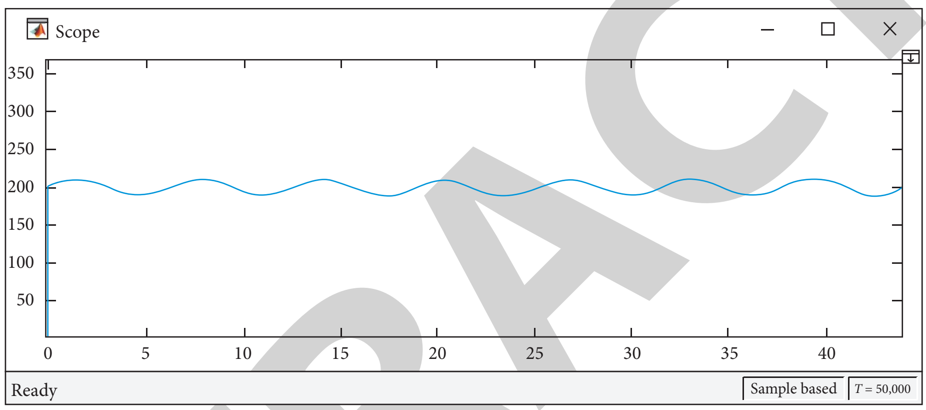


Figure 6: Output result of CAN communication of sine wave.

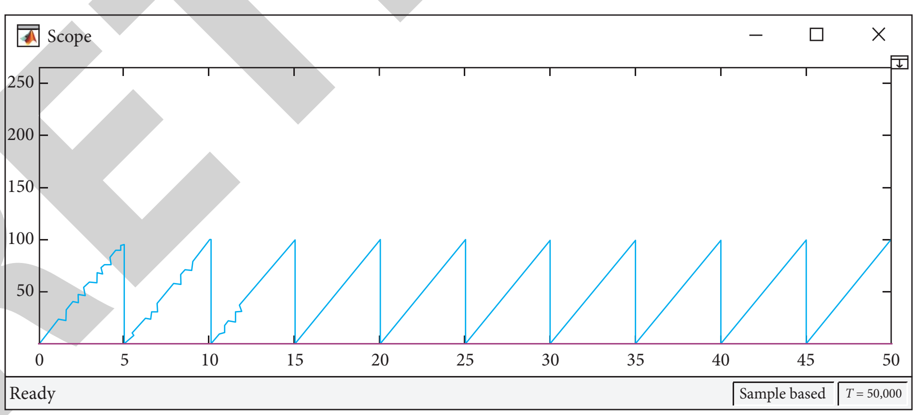


Figure 7: Output result of CAN communication of repeating sequence.

RETRACTED

parameters are set as follows: slope is 5, start time is 15, and initial output is 5. The final output result of CAN communication is shown in Figure 10.

Use "Signal Generator" as the data source and from Simulink → Sources library, add a model of Signal Generator to the model. Signal Generator can be used to generate four different waveforms: sine wave, square wave, sawtooth wave, and random wave. The equation of wave function is

$$Y(t) = \text{Amp} * \text{Waveform}(\text{Freq}, t). \quad (6)$$

In this instance, sine wave is selected, and the values of parameters are set as follows: amplitude is 1 and frequency is 1. The final output result of CAN communication is shown in Figure 11.

Use "Chirp Signal" as the data source and from Simulink → Sources library, add a "Chirp Signal" to the model.

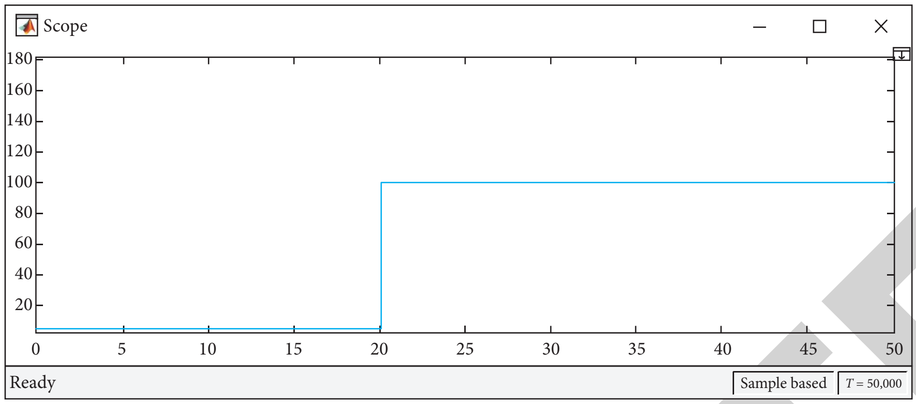


Figure 8: Output result of CAN communication of step.

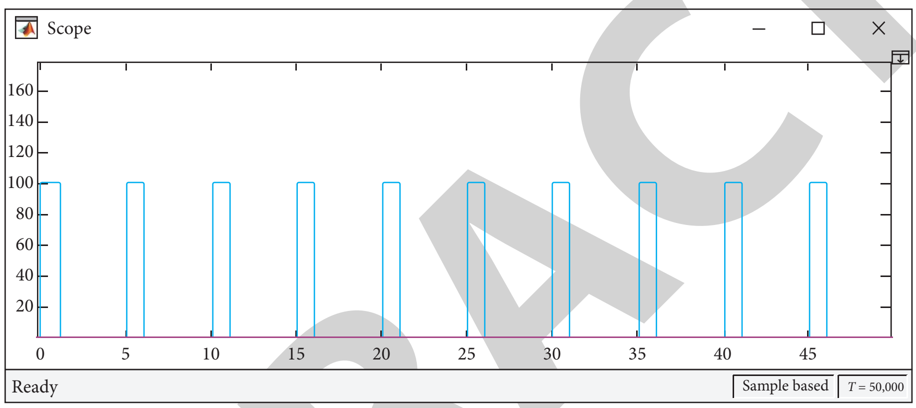


Figure 9: Output result of CAN communication of pulse generator.

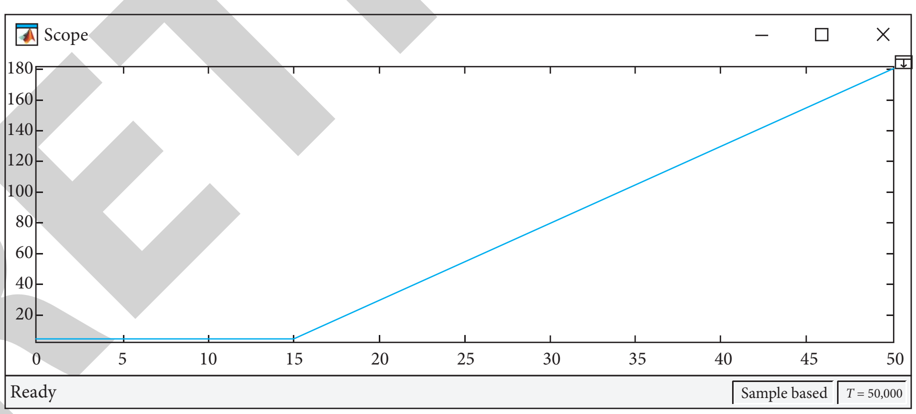


Figure 10: Output result of CAN communication in ramp.

This module is used to generate the waves, the frequency of which constantly increases. The parameters of initial frequency, target time, and frequency at target time decide its output. In this instance, the values of parameters are set as follows: initial frequency is 0.1, target time is 100, and frequency at target time is 5. The final output result of CAN communication is shown in Figure 12.

Through the above experimental results, it can be observed that the IoT-oriented network communication model not only has low consumption, but also solves the front-end

RETRACTED

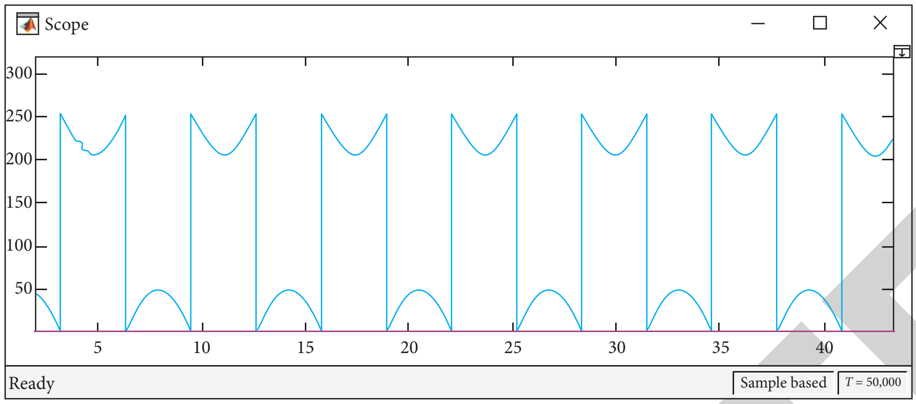


Figure 11: Output result of CAN communication of signal generator.

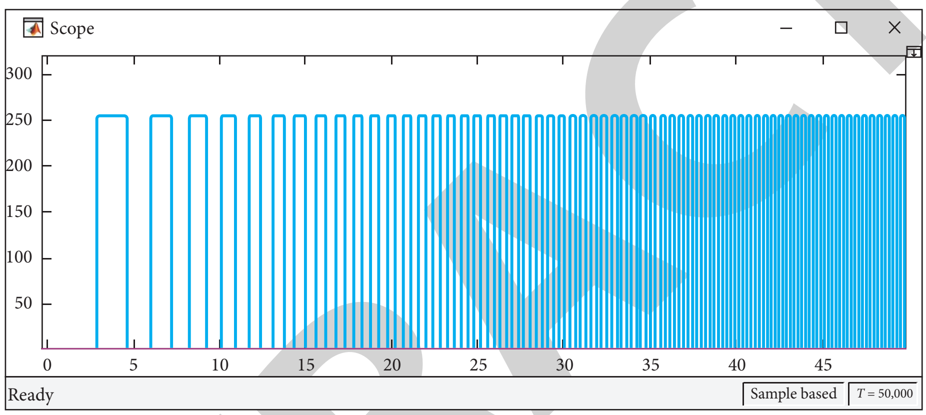


Figure 12: Output result of CAN communication of chirp signal.

communication compatibility issues caused by different communication protocols.

6. Conclusions

IoT is an extension of the Internet, which extends to the communication network between objects through wireless network. It goes through various sensor devices and it can perform real-time collection, remote monitoring, connection operations, and interaction information on the device. It can constitute a new-type network with Internet technology. Interface circuit is an important link in CAN bus network and its reliability and security have directly affected the operation of the entire communication network. This paper has summarized several key issues to be noted in the design of CAN interface circuit, grasped the key in the design, and proposed an IoT-oriented universal CAN bus design, which has enhanced the quality and performance of multi-interface circuit and ensured the secure and reliable operation of CAN bus. The simulation experiment has proven that the proposed design scheme is effective.

Data Availability

The raw/processed data required to reproduce these findings cannot be shared at this time as the data also form part of an ongoing study.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Jiangsu Basic Research Program (Natural Science Foundation) Project (Grant no. BK20191393), and Jiangsu Agriculture Science and Technology Innovation Fund (JASTIF, CX(20)3038).

References

[1] C.-Z.-E. Li and Z. W. Deng, "The embedded modules solution of household Internet of Things System and the future development," *Procedia Computer Science*, vol. 166, pp. 350–356, 2020.

[2] X. Huang, "Intelligent remote monitoring and manufacturing system of production line based on industrial Internet of Things," *Computer Communications*, vol. 150, no. 15, pp. 421–428, 2020.

[3] B. Diène, J. J. P. C. Rodrigues, O. Diallo, E. H. M. Ndoye, and V. V. Korotaev, "Data management techniques for internet of things," *Mechanical Systems and Signal Processing*, vol. 138, no. 8, p. 106564, 2020.

[4] J. H. Nord, A. Koohang, and J. Paliszkiewicz, "The internet of things: review and theoretical framework," *Expert Systems with Applications*, vol. 133, no. 1, pp. 97–108, 2019.

RETRACTED

[5] M. K. Ishak and F. K. Khan, “Unique message authentication security approach based controller area network (CAN) for anti-lock braking system (ABS) in vehicle network,” *Procedia Computer Science*, vol. 160, pp. 93–100, 2019.
[6] X. Kong, Y. Zheng, M. Ouyang et al., “Signal synchronization for massive data storage in modular battery management system with controller area network,” *Applied Energy*, vol. 197, no. 1, pp. 52–62, 2017.
[7] Y. Jiang, “Combination of wearable sensors and internet of things and its application in sports rehabilitation,” *Computer Communications*, vol. 150, no. 15, pp. 167–176, 2020.
[8] S. Mubeen, J. Mäki-Turja, and M. Sjödin, “Integrating mixed transmission and practical limitations with the worst-case response-time analysis for controller area network,” *Journal of Systems and Software*, vol. 99, no. 1, pp. 66–84, 2015.
[9] M. B. N. Shah, A. R. Husain, S. Punekkat, and R. S. Dobrin, “A new error handling algorithm for controller area network in networked control system,” *Computers in Industry*, vol. 64, no. 8, pp. 984–997, 2013.
[10] Y. Shoukry, H. Shokry, and S. Hammad, “Distributed dynamic scheduling of controller area network messages for networked embedded control systems,” *IFAC Proceedings Volumes*, vol. 44, no. 1, pp. 1959–1964, 2011.
[11] S. Mubeen, J. Mäki-Turja, and M. Sjödin, “MPS-CAN analyzer: integrated implementation of response-time analyses for controller area network,” *Journal of Systems Architecture*, vol. 60, no. 10, pp. 828–841, 2014.
[12] M. S. Manshahia, “Grey wolf algorithm based energy-efficient data transmission in internet of things,” *Procedia Computer Science*, vol. 160, pp. 604–609, 2019.
[13] S. Zeadally, F. K. Shaikh, A. Talpur, and Q. Z. Sheng, “Design architectures for energy harvesting in the Internet of Things,” *Renewable and Sustainable Energy Reviews*, vol. 128, no. 8, p. 109901, 2020.
[14] G. Cena, I. Cibrario Bertolotti, T. Hu, and A. Valenzano, “On a family of run length limited, block decodable codes to prevent payload-induced jitter in Controller Area Networks,” *Computer Standards & Interfaces*, vol. 35, no. 5, pp. 536–548, 2013.
[15] M. Park and X. Piao, “Enhanced utilization bound of rate-monotonic scheduling in controller area networks,” *Information Processing Letters*, vol. 114, no. 11, pp. 603–606, 2014.
[16] S. M. Nekooei, G. Chen, and R. K. Rayudu, “Automatic design of fuzzy logic controllers for medium access control in wireless body area networks-an evolutionary approach,” *Applied Soft Computing*, vol. 56, no. 7, pp. 245–261, 2017.
[17] X. Zhu, H. Zhang, D. Cao, and Z. Fang, “Robust control of integrated motor-transmission powertrain system over controller area network for automotive applications,” *Mechanical Systems and Signal Processing*, vol. 58-59, pp. 15–28, 2015.
[18] S. E. Marx, J. D. Luck, S. K. Pitla, and R. M. Hoy, “Comparing various hardware/software solutions and conversion methods for Controller Area Network (CAN) bus data collection,” *Computers and Electronics in Agriculture*, vol. 128, no. 10, pp. 141–148, 2016.
[19] W. Wei, H. Song, W. Li, P. Shen, and A. Vasilakos, “Gradient-driven parking navigation using a continuous information potential field based on wireless sensor network,” *Information Sciences*, vol. 408, no. 2, pp. 100–114, 2017.
[20] Y. Tao, L. Dong, and L. Chai, “Design of special communication protocol for electric rudder system based on CAN bus,” *Aviation Precision Manufacturing Technology*, vol. 53, no. 6, pp. 22–26, 2017.
[21] G. Breaban, M. Koedam, S. Stuijk, and K. Goossens, “Time synchronization for an emulated CAN device on a multi-processor system on chip,” *Microprocessors and Microsystems*, vol. 52, pp. 523–533, 2017.
[22] C.-Y. Wan, A. T. Campbell, and L. Krishnamurthy, “Pump-slowly, fetch-quickly (PSFQ): a reliable transport protocol for sensor networks,” *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 862–872, 2005.
[23] Y. Sun, Y. Weng, B. Luo et al., “Gesture recognition algorithm based on multi-scale feature fusion in RGB-D images,” *IET Image Processing*, 2020.
[24] Q. Pu and X. Wu, “Research on wireless routing algorithm based on congestion control,” *Communication Technology*, vol. 49, no. 3, pp. 312–317, 2016.
[25] R. Alaei, P. Moallem, and A. Bohlooli, “Statistical based algorithm for reducing bit stuffing in the controller area networks,” *Microelectronics Journal*, vol. 101, no. 7, p. 104794, 2020.
[26] J. Bauer, R. Helmke, A. Bothe, and N. Aschenbruck, “CAN’t track us: adaptable privacy for ISOBUS controller area networks,” *Computer Standards & Interfaces*, vol. 66, no. 10, p. 103344, 2019.
[27] Z. Chen, D. Chen, Y. Zhang, X. Cheng, M. Zhang, and C. Wu, “Deep learning for autonomous ship-oriented small ship detection,” *Safety Science*, vol. 130, p. 104812, 2020.
[28] L. Dong, W. Wu, Q. Guo, M. N. Satpute, T. Znati, and D. Z. Du, “Reliability-aware offloading and allocation in multilevel edge computing system,” *IEEE Transactions on Reliability*, pp. 1–12, 2019.
[29] R. Ma, L. Zhang, G. Li, D. Jiang, S. Xu, and D. Chen, “Grasping force prediction based on sEMG signals,” *Alexandria Engineering Journal*, vol. 59, no. 3, p. 1135, 2020.
[30] W. Wei, X. Fan, H. Song, and J. Yang, “Imperfect information dynamic stackelberg game based resource allocation using hidden Markov for cloud computing,” *IEEE Transactions on Services Computing*, vol. 11, no. 1, pp. 78–89, 2016.

RETRACTED