Measurement of master node delay in networked control systems

Yongmeng Huang1,2, Shuai Ji1,2, Zuguang Huang3, Chengrui Zhang1,2 and Tianliang Hu1,2

Abstract
Master node delay is an unavoidable factor in a networked control system (NCS), which will lead to performance deterioration of the system, and large delay may even cause instability. In order to measure the master node delay in NCSs, its influence factors are analyzed and a non-perturbative black-box measurement method is proposed. In this method, a FPGA-based (Field-Programmable Gate Array) measurement device is developed, which sends out testing frames to master node periodically and triggers the master node to send out a response frame for every testing frame. Then the measurement device can get the measurement delay value by calculating the time interval between sending the testing frame and receiving the response frame, which consists of master node delay and some other delay that can be calculated precisely. Therefore, the accurate master node delay can be obtained by further processing the measurement delay value. A set of experiments were carried out and the results show that the proposed method can effectively measure the master node delay without a thorough understanding about its hardware and operating system software, and is not constrained by the type of operating system. Besides, as the master node delay can be measured exactly in the actual working condition without perturbing its user program by using this method, the results can reflect the real-time performance of the master node accurately. So that, it can provide a direct reference for choosing an appropriate master node for NCSs.

Keywords
the master node delay, non-perturbative measurement, networked control system, real time ethernet

Introduction
Networked control system (NCS) is a kind of spatially distributed system wherein the control loop is closed through communication network. NCSs can break the limit of point-to-point control and achieve remote operations, which makes it possible to design large-scale systems. Compared to conventional control systems, NCSs have advantages in low cost of installation, flexibility in system implementation, ease of maintenance, efficient resource allocation and so on. However, the network delay is inevitable when control and status data are exchanged between master and slave nodes, which will degrade the performance of control systems and even cause system instability.

There are two main types of network delay in a NCS. One is commonly called the network-induced delay, which can be divided into the propagation delay through the network medium and the transmission delay of the packets bits from the queue and into the link. The other is the processing delay of each component in NCSs due to its limited processing speed, such as the master node delay. Figure 1 shows a typical NCS model with the network delay.4 Where, \( \tau_{nw} \) refers to the network induced delay from sensor to controller, \( \tau_{ca} \) refers to the network induced delay from controller to actuator. The master node delay \( \tau_{m} \) is the total time consumed by the master controller between receiving data from sensor and sending out control data to actuator. All these delays are involved in the control loop, when the total delay is smaller than control cycle, the state space equation of NCS is shown in equation (1), where \( r \) is the total delay in NCS including \( \tau_{nw} \), \( \tau_{ca} \) and \( \tau_{m} \). It can be seen that these delays will degrade dynamic performance and the stability of NCS, and add lots of difficulties in the development of high performance NCS.

In traditional communication networks which are not real-time, the network-induced delay is much larger than the master node delay because of the data collision, transmission rate and so on. As a result, the previous literature, whether review

1School of Mechanical Engineering, Shandong University, Jinan, P.R. China
2Key Laboratory of High Efficiency and Clean Mechanical Manufacture at Shandong University, Ministry of Education, Jinan, P.R. China
3Geneerted Machine Tool Engineering Research Institute Co., Ltd, Beijing, P.R. China

Corresponding author:
Shuai Ji, School of Mechanical Engineering, Shandong University, 17923 Jingshi road, Jinan 250061, P.R. China.
Email: jizhongthe@126.com

Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).
articles, or research papers about the stability of NCS,\textsuperscript{3,6} linear
system,\textsuperscript{7} nonlinear system\textsuperscript{8} and software defined network real-
time system,\textsuperscript{9} mainly focused on $\tau_{sc}$ and $\tau_{ca}$.

However, data collision of real-time Ethernet (RTE), such as
EtherCAT, PROFINET IRT, Ethernet Powerlink, etc. Can be
avoided,\textsuperscript{10,11} and their bandwidth is as high as 100Mbps or even
1000Mbps, so the network-induced delay $\tau_{sc}$ and $\tau_{ca}$ have been
greatly reduced. Besides, these delays can be calculated ac-
curately based on the frame length and network topology. Some
works\textsuperscript{12-15} about the performance comparison of different RTEs
provided the calculation methods of the minimum communi-
cation cycle of mainstream RTEs in common scenarios.

In fact, standard personal computer (PC) with real-time
operating system (RTOS) as the master node is becoming
more and more popular for its high computing ability,
powerful integrated development environment and friendly
human-machine interaction. For example, Windows oper-
ating system, equipped with a real-time extension software
such as TwinCAT,\textsuperscript{16} Codesys\textsuperscript{17} or Kithara,\textsuperscript{18} is competent to
be the master node of a NCS. As the RTOS on standard PC is
responsible for multi task, the master node delay is inevi-
table, and even larger than the network-induced delay.
Hence, the master node delay $\tau_c$ will account for an in-
creasing proportion of network delay. Master node delay
will affect the minimum communication cycle, dynamic
performance and stability of NCSs, so analysis and mea-
surement of the master node delay is needed, and a con-
venient and effective measurement method for the master
node delay is proposed in this paper.

The paper is organized as follows. In Section 2, the factors
causing the delay of PC-based master node are analyzed.
Section 3 introduces the details of the proposed measurement
method. Then, the development of the measurement device is
given in Section 4. Section 5 is a set of experiments which
give a performance comparison of different master nodes. At
last, a conclusion is presented in Section 6.

Analysis for master node delay
The master node usually works as the controller in NCSs,
while the sensors and actuators are all slave nodes. Hence, the
master node delay should be the duration between the master
node receiving an upstream status frame and sending out a
downstream control frame.

As shown in Figure 2, the master node delay is determined
by both hardware and software, of which hardware delay is
influenced by both CPU architecture and Design of mother-
boards and software delay includes operating system (OS) delay
and algorithms efficiency. The OS delay can be further divided
into Interrupt delay, Preemption delay, Task switching delay,
Signal alternating delay, Deadlock release delay and so on.

There are software and hardware methods for measuring
OS delay. The software methods mainly employ code in-
strumentation\textsuperscript{19} or embedded software testing tool.\textsuperscript{20} The
accuracy of these methods is not very high, because the op-
erating system computational resources are occupied by both
application program and testing software, and the precision of
system clock is limited. Besides, a thorough understanding on
OS principles is vital to modify system-level code.

The hardware methods\textsuperscript{21-23} usually employ a signal
generator to trigger the master node under test to produce a
single output, and then an oscilloscope is used to measure the
response delay between the input and output signals. Al-
though the measurement result of hardware method is more
accurate than that of software, it is not convenient for sta-
tistical analysis and long-term measurement.

However, all the methods above measure the real-time
performance factors separately, and the actual delay of the
master node is a combination of coupling factors. Besides, the master node delay caused by hardware is difficult to quantify independently. So it is difficult to obtain accurate master node delay with a white-box measurement method, and a non-perturbative black-box measurement method is proposed in this paper.

**Measurement method for the master node delay time**

In order to make the measurement result close to the actual working condition as more as possible, a NCS with master-slave structure is adopted, as shown in Figure 3. Where, a standard PC with RTOS or non RTOS works as master node under the test.
and the measurement device based on FPGA works as a slave node. The measurement process is implemented in a FPGA chip, where high-speed clock and hardware solution can ensure the accuracy of measurement results.

The proposed measurement method regards the master node as a black box, and the measurement device periodically sends upstream frames to the master node and receives the downstream response frames. At the same time, a high-precision timer designed in the FPGA records the sending and receiving time, so that the master node delay can be calculated. Acting as a trigger event, receiving an upstream frame makes the master node to implement data reception, data analysis, computational algorithm, downstream frame sending, etc. The whole process not only involves the delays caused by OS and computational algorithm, but also includes hardware factors. As a result, the master node delay of each communication cycle can be measured with high precision. Besides, the proposed method does not require any extra underlying system operation, which greatly reduces the difficulty of measurement and can be easily used in different types of OS.

**Measurement Process**

As shown in Figure 4, $T_{\text{measure}_{k-1}}$ and $T_{\text{measure}_k}$ denote the measurement delay of cycle $k-1$ and cycle $k$, respectively.

| Preamble | SFD | Dest MAC | Source MAC | Ethernet Type | Data and Pad | FCS |
|----------|-----|----------|------------|---------------|--------------|-----|
| 7 bytes  | 1B  | 6 bytes  | 6 bytes    | 2 bytes       | 46-1500 bytes | 4 bytes |

**Figure 6.** Master node delay algorithm.

**Figure 7.** Frame structure according to IEEE802.3.

**Figure 8.** Program architecture of master node.
and the measurement device based on FPGA works as a slave node. The measurement process is implemented in a FPGA chip, where high-speed clock and hardware solution can ensure the accuracy of measurement results.

The proposed measurement method regards the master node as a black box, and the measurement device periodically sends upstream frames to the master node and receives the downstream response frames. At the same time, a high-precision timer designed in the FPGA records the sending and receiving time, so that the master node delay can be calculated. Acting as a trigger event, receiving an upstream frame makes the master node to implement data reception, data analysis, computational algorithm, downstream frame sending, etc. The whole process not only involves the delays caused by OS and computational algorithm, but also includes hardware factors. As a result, the master node delay of each communication cycle can be measured with high precision. Besides, the proposed method does not require any extra underlying system operation, which greatly reduces the difficulty of measurement and can be easily used in different types of OS.

### Measurement Process

As shown in Figure 4, $T_{\text{measure}_{k-1}}$ and $T_{\text{measure}_k}$ denote the measurement delay of cycle $k-1$ and cycle $k$, respectively. $T_{\text{cycle}}$ denotes the communication cycle. The measurement process is shown in Figure 5.

A detailed description of the measurement process is as follows:

1. At the beginning, set $T_{\text{cycle}}$ as the communication cycle of this time of measurement and set $T_{\text{measure}_0}$ as the initial value 0 which will not affect the statistical results. Then the master node is responsible for initializing the communication.
2. After initialization, the measurement device switches to cycle communication state. In this stage, the measurement device will send an upstream frame with the measurement result of last communication cycle $T_{\text{measure}_{k-1}}$ cyclically according to the set period, and start an internal high-precision timer when sending out the upstream frame.
3. After receiving the upstream frame, the master node parses $T_{\text{measure}_{k-1}}$, calculates the master node delay $T_{\text{master}_{k-1}}$, and writes the value into a file for further analysis. At the same time, the master node sends a downstream frame to the measurement device to respond the upstream frame.
4. The measurement device parses the received downstream frame online, and latches its internal high precision timer at the time of receiving the downstream frame. So that the measurement delay $T_{\text{measure}_k}$ of this cycle can be obtained.
5. Repeat steps (b) – (d) to collect sufficient data for statistical analysis.

This method is mainly aimed at standard PC-based master node, which has powerful computing capability. Therefore,
Table 2. Table of software experimental conditions.

| No. | Hardware platform | Operating system | Driver |
|-----|-------------------|------------------|--------|
| 1   | Laptop            | Win7, 32bits     |        |
| 2   | Laptop            | Ubuntu14.04.5 Desktop (32bits) +RT3.18.59 | Libcap |
| 3   | IPC1              | Ubuntu14.04.5 Desktop (32bits) +RT3.18.59 | Libcap |
| 4   | IPC1              | Ubuntu14.04.5 Desktop (32bits) +RT4.1.40 | Libcap |
| 5   | IPC1              | Ubuntu14.04.5 Server (32bits) +RT4.1.40 | Libcap |
| 6   | IPC2              | WinCE6.0         | Minipro driver |
| 7   | Laptop            | Win7, 32bits + kithara KRTS packet | KRTS packet |
| 8   | IPC1              | Win7, 32bits + kithara KRTS packet | KRTS packet |

The measurement results are transmitted to the master node for subsequent calculation and data statistics just as step (c) shown.

The master node delay algorithm

The direct measurement result $T_{measure,k}$ contains the propagation delay, transmission delay and the device delay of the slave node, so subsequent operation is needed to calculate the exact master node delay, that is, the master node delay algorithm shown in Figure 6.

As shown in Figure 4, $T_{master,k}$ is the master node delay in cycle $k$

$$T_{master,k} = T_{measure,k} - D_{rx} - D_{tx} - T_{pro} \times 2 \quad (2)$$

Where, $T_{measure,k}$ is the direct measurement delay time value measured by the measurement device in a communication cycle, which will be encapsulated into the upstream frame and sent to the master node. $T_{pro}$ is the time delay caused by the propagation of data frames over the network, which is mainly related to the length of the network cable $L_{cable}$. The propagation delay is $4.8$–$5.3$ ns/m for category 5 cable. Then

$$T_{pro} = L_{cable} \times 5 \text{ ns} \quad (3)$$

$D_{rx}$ and $D_{tx}$ represent the delays caused by the measurement device when sending the upstream frame and receiving the downstream frame, respectively. The delays mainly consist of the delay of physical layer chip, $D_{PHY_{rx}}$ and $D_{PHY_{tx}}$, and the delay of data frame transmission, $T_{frame_{rx}}$ and $T_{frame_{tx}}$. Different physical layer chips have different $D_{PHY_{rx}}$ and $D_{PHY_{tx}}$, which can be obtained from its datasheets or through experimental testing. Then

$$D_{rx} = D_{PHY_{rx}} + T_{frame_{rx}} \quad (4)$$

$$D_{tx} = D_{PHY_{tx}} + T_{frame_{tx}} \quad (5)$$

The network under test is the Fast Industrial Ethernet whose bandwidth is 100Mbp/s, so the transmitting rate is 80ns/byte. According to the frame structure of standard Ethernet shown in Figure 7, both upstream and downstream data frames take the Data and Pad region as the minimum value 46 bytes. Then, the length of data frame $L_{frame}$ and the delay of data frame transmission $T_{frame}$ can be obtained.

$$L_{frame} = L_{Pre} + L_{STD} + L_{DestMAC} + L_{SourceMAC} + L_{Type}$$

$$+ L_{Data} + L_{PCS} = 7 + 1 + 6 + 6 + 2$$

$$+ 46 + 4 = 72 \text{ byte}$$

Then, the length of data frame

$$L_{frame} = 72 \text{ byte} \times 80 \text{ ns/byte}$$

$$= 72 \text{ byte} \times 80 \text{ ns/byte}$$

$$= 5760 \text{ ns}$$

According to formula (4) – (7), both $T_{frame_{tx}}$ and $T_{frame_{rx}}$ can be calculated to be 5760ns, and the sum is 11,520ns. Besides, $T_{master,k}$ can be calculated by combining the formula (2) – (7).

$$T_{master,k} = T_{measure,k} - 2 \times L_{cable} \times 5 - D_{PHY_{rx}} - D_{PHY_{tx}} - 11520 \text{ ns}$$

Development of measurement device

The proposed measurement method in this paper adopts the master-slave structure. The slave node is developed based on a FPGA chip, whose hard real-time property can realize high precision measurement. For the master node, a program is necessary for standard PC with RTOS or non RTOS to respond the frame from slave nodes.

Program Architecture of Master Node

The program running on master node is developed with C++, which consists of two parts: communication driver and control task program, as shown in Figure 8, communication driver and control task are two threads started by the master node. They cooperate with each other to implement the tasks of receiving and sending frames and processing measured values through events and semaphores.

The control task is responsible for the parse of upstream frames, the storage of measured values and the packaging of downstream frames. In practice, all computational algorithms will be running on the master node under test, if which can be added to the program, the measurement results can be closer to the actual situation and more referential.

The communication driver is responsible for receiving and sending data frame, controlling of communication state machine and driving network card. In this part, communication function is realized by calling system API functions provided by RTOS employed by the master node under test. With this modular software design, the method can be implemented on different types of OS with only little extra work.

Development of measurement device node

The measurement device works as a slave node in NCSs, which is developed based on a FPGA chip from Altera,
whose advantages are hard real-time, flexible, high-speed and reconfigurable. The device not only acts as a slave node to communicate with the master node, but also can obtain nanosecond resolution results with the high-precision timer in FPGA. Besides, the Nios II soft core technology is used to facilitate the analysis of the downstream frame and the preparation of the upstream frame.

The module of the FPGA program is described in detail as shown in Figure 9:

a. High precision timer in the Delay measurement module will be started when FPGA sends an upstream frame and latched when FPGA receives a response downstream frame from master node. The value of the timer will be sent to Nios II CPU.

Figure 11. Histogram of experiments data.
b. Nios II CPU encapsulates the Delay value into upstream frame when its value is updated cyclically.
c. Cycle load module generates a signal with a set time interval to trigger the Utransfer module to send upstream frame with the measured value to master node.

**Experiment**

In order to verify the proposed method, a set of experiments are designed to compare the master node delay of three different RTOSs (Windows 7 + Kithara, Windows CE 6.0 and Linux with RT-PREEMPT patch). Besides, a master node with Windows 7 is adopted as a reference for conventional non-real-time operating system. At the same time, industrial personal computer (IPC) and laptop are employed to compare the influence of hardware platform on the master node delay.

**Experimental environment building**

As shown in Figure 10, the hardware platforms of the master nodes are two IPCs and one laptop Thinkpad E430C (abbreviated as IPC1, IPC2, laptop) whose hardware information is shown in Table 1.

As shown in Table 2, different OSs and communication drivers have been employed, where communication driver is an important uncertain factor can influence the master node delay.

Some cross-comparison of the measurement results on these platforms are provided to show the measuring performance of the proposed method.

The right side of each image in Figure 10 is the measurement device, whose FPGA chip is EP4CE10F17C8N from Altera and PHY chip is LAN8710A from SMSC. According to LAN8710A’s datasheet, the total sending and receiving delay of LAN8710A is 320ns.

\[ D_{\text{PHY}} \text{tx} + D_{\text{PHY}} \text{rx} = 320 \text{ ns} \]  

(9)

The master nodes under test are connected to the measurement device with a CAT-5 cable, and the cable length \( L_{\text{cable}} \) is 1m. \( T_{\text{master},k} \) can be calculated with formula (8).

\[ T_{\text{master},k} = T_{\text{measure},k} - 11850 \text{ ns} \]  

(10)

Eight different experimental conditions as shown in Table 2 were established, and each measurement lasted for about 40 min with the communication period of 4 milliseconds. There are about 600,000 delay values collected for every time of measurement.

**Analysis of experimental results**

The histograms of these eight measurements results are shown in Figure 11. The X-axis is the value of the master node delay and the Y-axis is the percentage. Because there are lots of uncertain factors for master node delay, so the tested results are random, which may be particularly high for the poor real-time performance. As a result, the master node delay is widely distributed on the X-axis. There are always some long-time delays of small proportion in the overall sample, which results in a big blank on the right side of the original diagram. Thus, in order to show the results clearly, the data-intensive distribution areas are intercepted in Figure 11.

As shown in Table 3, the maximum delay time is one of the most important indicators in practice, which determines the minimum control cycle of NCSs. Too large delay means the long control period, which is intolerable in high-precision control systems. The mean value and standard deviation synthetically reflect the overall delay and data consistency of the experimental samples. The standard deviation also reflects the jitter of the PC-based master node. When the maximum delay value meets the requirement, they can help to choose more stable master node. The last column shows the ratio of data excessive 1 millisecond, which is usually set as the communication cycle in the field of industrial control.

For Windows 7, the maximum delay is about 200 ms and the values are scattered, which means it cannot meet the real-time requirement for a master node in NCSs.

For RT Ubuntu with a RT-PREEMPT patch, the maximum delay is much smaller than Windows 7. However, it is still in millisecond level and the jitter is high. It is supposed the communication driver reduces its real-time performance, which limits the minimum communication cycle.

For WinCE 6.0, although the mean delay is larger than RT Ubuntu, its maximum is about 700 microseconds, which is better and conforms to the actual situation.

For Windows + Kithara, the maximum delay is about 100 microseconds, which has the best real-time performance. The standard deviation is much smaller than other OSs, which means the jitter of Windows + Kithara is also the best.

In addition, (2), (3) and (7), (8) in Figure 11 can also show the influence of hardware platform to the master node delay.

**Conclusions**

In NCSs, master node delay is inevitable, especially for the PC-based control system. This study analyzed the factors lead to the master node delay in NCSs, and proposed a non-perturbative black-box measurement method. As the delay measurement is implemented on a FPGA-based slave node, the master node can be tested without perturbation and all the complex factors of master node are consisted in the measured value as a black box, so that high accuracy can be achieved.
A set of experiments have been conducted, where different master nodes with different OSs were employed to verify the proposed method. And a cross-comparison of the measurement results have been concluded. Therefore, besides the master node delay measurement method, this paper also gives a reference for selecting an appropriate master node of NCSs.

Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the National Key R&D Program of China under Grant 2019YFB2006200.

ORCID iD
Yongmeng Huang https://orcid.org/0000-0001-7379-7469

References
1. You K-Y and Xie L-H. Survey of recent progress in networked control systems. Acta Automatica Sinica 2013; 39(2): 101–117.
2. Gupta RA and Mo-Yuen Chow MY. Networked control system: overview and research trends. IEEE Trans Ind Elect 2010; 57(7): 2527–2535.
3. Branicky MS, Phillips SM and Wei Z. Stability of networked control systems: explicit analysis of delay. In: Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No.00CH36334), Chicago, IL, USA, 28–30 June 2000, pp.2352–2357.
4. Zhang L, Gao H and Kaynok O. Network-induced constraints in networked control systems-a survey. IEEE Trans Ind Inform 2013; 9(1): 403–416.
5. Chow MY and Tipsuwan Y. Network-based control systems: a tutorial. In: IECON’01. 27th Annual Conference of the IEEE Industrial-Electronics Society (Cat. No.37243), Denver, CO, USA, 29 November-1 December 2001, pp.1593–1602.
6. Sun X-M, Wu D, Liu G-P, et al. Input-to-state stability for networked predictive control with random delays in both feedback and forward channels. IEEE Trans Ind Elect 2014; 61(7): 3519–3526.
7. Liu B, Liu Y, Li W, et al. Modeling and control of networked control systems with random network-induced delay and packet-dropout. In: 2017 12th IEEE Conference on Industrial Electronics and Applications(ICIEA), Siem Reap, Cambodia, 18–20 June 2017, pp.786–791.
8. Pang Z-H, Liu G-P, Zhou D, et al. Data-based predictive control for networked nonlinear systems with network-induced delay and packet dropout. IEEE Trans Ind Elect 2016; 63(2): 1249–1257.
9. Kumar R, Hasan M, Padhy S, et al. End-to-end network delay guarantees for real-time systems using SDN. In: 2017 IEEE Real-Time Systems Symposium (RTSS), Paris, France, 5–8 December 2017, pp.231–242.
10. Wollschlaeger M, Sauter T and Jaspeme J. The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0. IEEE Ind Elect Mag 2017; 11(1): 17–27.
11. Jaspeme J, Schumacher M and Weber K. Limits of increasing the performance of industrial Ethernet protocols. In: 2007 IEEE Conference on Emerging Technologies and Factory Automation (EFTA 2007), Patras, Greece, 25–28 September 2007, pp.17–24.
12. Robert J, Georges J, Rondeau E, et al. Minimum cycle time analysis of Ethernet-based real-time protocols. Int J Comput Comm Control 2012; 7(4): 744–758.
13. Seno L, Vitturi S and Zunino C. Real time Ethernet networks evaluation using performance indicators. In: 2009 IEEE Conference on Emerging Technologies & Factory Automation, Mallorca, Spain, 22–25 September 2009, pp.1–8.
14. Cena G, Seno L, Valenzano A, et al. Performance analysis of Ethernet Powerlink networks for distributed control and automation systems. Comp Stand Inter 2009; 31(3): 566–572.
15. Prytz G. A performance analysis of EtherCAT and PROFINET IRT. In: 2008 IEEE International Conference on Emerging Technologies and Factory Automation, Hamburg, Germany, 15–18 September 2008, pp.408–415.
16. Bermudez-Ortega J, Besada-Portas E, Lopez-Orozco JA, et al. Developing web & TwinCAT PLC-based remote control laboratories for modern web-browsers or mobile devices. In: 2016 IEEE Conference on Control Applications (CCA), Buenos Aires, Argentina, 19–22 September 2016, pp.810–815.
17. Liu J, Fu WP, Zhou L, et al. Design and application of a communication system based on Codesys. Adv Mater Res 2014; 1044-1045: 1113–1117.
18. Kithara. Future-proof Software Architecture with Kithara RealTime Suite[EB/OL], 2018. -02-28[2019-06-19]. https://kithara.com.cn/downloads#whitepaper-slideshow.
19. Shen J, Wu Q, Li X, et al. Research of the real-time performance of operating system. In: 2009 5th International Conference on Wireless Communications, Networking and Mobile Computing, Beijing, China, 24–26 September 2009, pp.1–4.
20. Wang H, Zhu X and Wang Y. The hardware probe connection research of CodeTEST based on ARM. In: 2010 International Forum on Information Technology and Applications, Kunming, China, 16–18 July 2010, pp.333–335.
21. Aroca R and Caurin G. A real time operating systems (RTOS) comparison. In: The 6th Work-shop on Operating System, Bento Gonalves, Brazil, 2009, pp.2441–2452.
22. Barbalace A, Luchetta A, Manduchi G, et al. Performance comparison of VxWorks, Linux, RTAI, and Xenomai in a hard real-time application. IEEE TRANSACTIONS NUCLEAR SCIENCE 2008; 55(1): 435–439.
23. Hambarde P, Varma R and Jha S. The survey of real time operating system: RTOS. In: 2014 International Conference on Electronic Systems, Signal Processing and Computing Technologies, Nagpur, India, 9–11 January 2014, pp.34–39.