OPC (OLE for Process Control) based Calibration System for Embedded Controller

Ming Cen
School of Automation, Chongqing University of Posts and Telecommunications, Chongqing, P. R. China
Email: m_cen0104@sina.com

Qian Liu, Yi Yan
School of Automation, Chongqing University of Posts and Telecommunications, Chongqing, P. R. China
Email: liuqian.122678@163.com, yanyi210@126.com

***Abstract*—During embedded software development of complex control system, the calibration is an important approach to obtain optimal parameters of embedded software. Currently, typical calibration systems are with poor adaptability for various controllers which have different communication interfaces and calibration protocols. In order to solve the problem, an improved architecture for embedded controller calibration system is proposed to support more communication buses and protocols. In this architecture, by introducing OPC (OLE for Process Control) technology, the host software of calibration system is separated into OPC server and client. The OPC server masks the difference of various calibration protocols and detail of communication devices, and provides a unified access interface for controller parameters. The OPC client calibrates and acquires the parameters though calling the interface provided by the OPC server. By the method, when communication device or calibration protocol is varied, only the corresponding OPC server is required to be replaced. Then the details of communication devices and calibration protocols are no longer considered while developing a calibration system, and the generality and openness of the calibration system are enhanced greatly. The calibration system corresponding to the architecture was applied to an engine controller to verify the effectiveness of the method presented.**

***Index Terms*—Embedded software, Embedded controller, Calibration system, OPC technology**

I. INTRODUCTION

The wide application of embedded system in nearly all of industrial control fields, such as automotive, aerospace, military and other manufacturing, was grown extremely. For example, the safety, comfort and efficiency of modern automotive mainly depend on various embedded electronic control systems, so modern automotives are equipped with more and more parts involving embedded controllers called electronic control units (ECU), including powertrain, chassis and body control, etc [1, 2]. In new energy vehicles especially, such as hybrid electric vehicles and pure electric vehicles, the number of ECUs is increasing greatly.

Nowadays, electronics makes 90% of the innovations in automotive industry, and 80% out of that is from software [3]. Because of increasing complexity and functionality of the systems, the size of software in modern automotive raised continuously, and almost 50-70% of the development costs of the ECUs are software costs [4]. For example, some cars contain more than 50 controllers, more than 600,000 lines of code, three different bus systems and approximately 150 messages and 600 signals [5]. On the other hand, due to the high pressure of time and cost, the productivity of the software development is of great importance for automotive manufacturers. It is similar in other industrial control fields also.

To reduce the increasing software development costs and time-to-market, and enhance the reliability, a series of approaches are applied to improve the efficiency of the software development and the quality. One of the solutions is to increase reuse of development results, including requirements, models, functions, and software components. For example, AUTOSAR (AUTomotive Open System ARchitecture) provides a group of embedded software architecture and interfaces standardization to facilitate the reuse of software components between different vehicle platforms, OEMs and suppliers [6]. The other is to manage the software development and maintenance processes, including requirements engineering, design, coding, software and systems integration, quality assurance and maintenance. The international standard, IEC 61508 and ISO 26262, provide an automotive safety lifecycle to develop a safety-related system [7, 8].

Other than ordinary embedded software, because of the complexity of automotive or other industrial control systems, the performance of embedded software highly dependents on the working parameters, including that of controlled objects and controllers. The working parameters of embedded controllers must be determined

doi:10.4304/jsw.7.12.2866-2873

and optimized by calibration and matching experiment from trial to finalizing of the products. So calibration is one of the key technologies in the development of embedded controllers. The calibration system with high efficiency and adaptability can improve the development efficiency and the quality of the embedded controllers greatly [9].

Currently, calibration systems mostly follow the ASAM standard architecture. ASAM [10, 11] is a standard system defined by Association for Standardization of Automation and Measuring Systems (ASAM). It provides a standard interface for measurement, calibration and fault diagnosis of automotive ECUs and embedded controllers of other industrial field. In this architecture, the communication interfaces, such as serial, CAN, USB and Ethernet, and the calibration protocols, including CCP (CAN Calibration Protocol), XCP (eXtended Calibration Protocol) and KWP2000 are supported. Many calibration systems were introduced. For example, the calibration system based on serial communication had been used to calibrate the parameters of the vehicular battery management system [12]. The hybrid electric vehicle ECU calibration system based on CCP was developed with Labview, and implemented the calibration and measurement of ECU parameters by calling the driver library of CAN communication adapter [13]. The gas engine controller calibration system based on CCP was developed with Borland C++ builder, and the connector of the host and the ECU was USB-CAN adapter [14]. There were other applications of CCP protocol in automotive controller also [15]. The KWP2000 protocol was used for diesel engine calibration system, and the serial port was as the connector of calibration system and high-pressure common rail diesel engine [16]. To support different types of ECUs with different parameters, a calibration system architecture supporting user interface customizing and reconstruction is provided [17].

However, the calibration systems in above solutions are strongly coupled with a specifically communication device or calibration protocol. Once the communication device or calibration protocol changed, the calibration system must be updated correspondingly. Therefore, the solutions are insufficient to suit different calibration protocols and hardware interfaces, and the generality and openness of the calibration system are restricted greatly.

An improved architecture of calibration system is presented to solve the problem, in which the different calibration protocols and communication devices are encapsulated with OPC technology to provide unified data access interface of ECUs. The development of OPC server and client of calibration system are discussed and verified also.

II. ARCHITECTURE OF CALIBRATION SYSTEM BASED ON MIDDLEWARE

A. *ASAM architecture of Calibration System*

ASAM working Group defines the conception of MCD (Measurement, calibration and Diagnostics) model, and provides a corresponding standard architecture to direct the development of calibration system. The typical calibration system accorded with the ASAM standard architecture is shown in Figure 1.

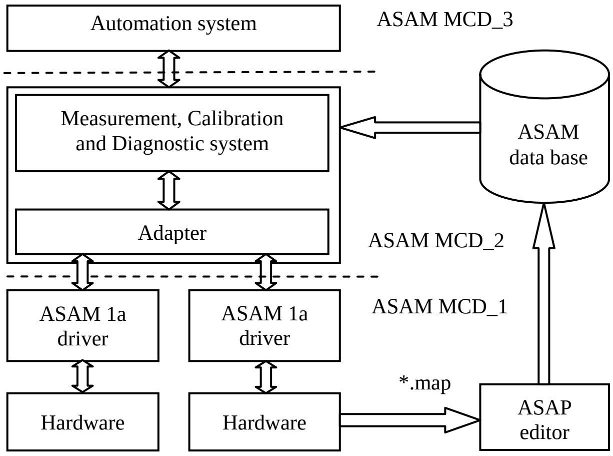


Figure 1. ASAM standard architecture of calibration system

In this architecture, there are three interfaces or protocols called ASAM MCD-1, ASAM MCD-2 and ASAM MCD-3. The calibration and measurement parameters of the ECU are originated from map file (*.map), which generated by the compiler of the micro controller firstly. The map file is transformed to ASAM MCD-2 data base file (*.a2l) by the ASAP editor. According to ASAM MCD-2 data base, the host of MCD system can upload or download parameters to the ECU by ASAM MCD-1 interface to implement calibration and measurement.

In the architecture, the host software user interface of calibration system is strong coupled with calibration protocol and bottom hardware layer generally. For example, a calibration system supporting serial port can not be used for the calibration of the ECUs which have CAN interface only. The host calibration software based on USB-CAN adapter can not support PCI-CAN adapter also. Furthermore, when the calibration protocol is changed to XCP, the original calibration system based on CCP would be failure. Once the calibration protocol or hardware interface layer is changed, the calibration system should be updated accordingly. So the calibration system can not meet the requirement of customizing and reconstruction according to the changing of ECUs.

The calibration system which supports host software interface customizing and automatically generating can provide the reconstruction of user interface, and the adaptability for different parameters or ECUs is improved [17]. For lack of the standard of abstract communication interface to encapsulate the different bottom hardware and calibration protocols, the host software is coupled with bottom hardware or calibration protocol layer yet. So it is very necessary to introduce a mechanism to shield the difference of communication hardware layer and protocol layer, and provides a unified data access interface to improve the generality and adaptability of calibration system.

B. *Architecture of Middleware based Calibration System*

There are mainly two approaches to mask different bottom hardware and calibration protocols, namely standard API and middleware technology. standard APIs can encapsulate the detail of communication hardware and calibration protocol, and provide unified access interface for calibration host software. The scheme of calibration system based on API is shown in Figure 2. But in fact, for lack of API specification, the API interface is varied for differnt communication device. If the driver interface or device are changed, the calibration system is required to be updated also.

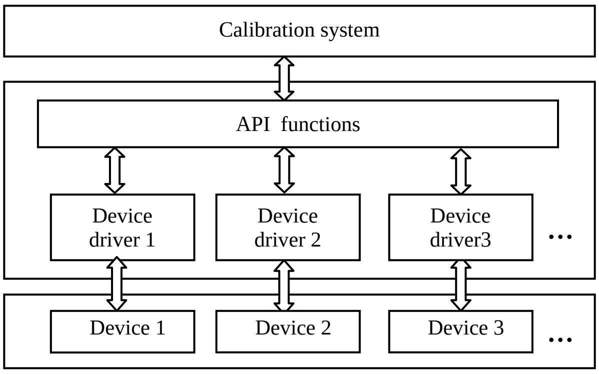


Figure 2. Scheme of calibration system based on API

Middleware technology [18] can provide unified interface to shield the detail of bottom communication devices and calibration protocols also. Once the device driver or communication protocol changed, only the middleware is required to be updated, and the calibration host software keeps the same. Then the development time and cost of calibration system are both reduced, and the development efficiency is improved to raise the development of embedded software. The architecture of calibration system based on middleware is shown in Figure 3.

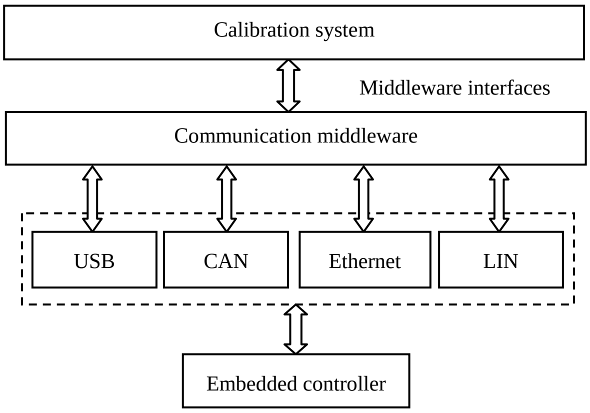


Figure 3. Architecture of calibration system based on middleware

The middleware of calibration system can provide a unified standard interface to shield the difference of network communication interfaces such as USB, Ethernet, CAN, LIN, etc, as well as the difference of calibration protocols such as CCP, XCP, and KWP2000 and so on. So the adaptability and generality of calibration system would be improved greatly.

III. CALIBRATION SYSTEM BASED ON OPC TECHNOLOGY

A. *Selection of Calibration System Middleware*

There are two basic middleware resolutions. The one is COM/DCOM (Component Object Model/Distributed COM) standard proposed by Microsoft Corporation [19, 20], and the other is CORBA (Common Object Request Broker Architecture) standard proposed by OMG (Object Management Group) [21]. The OPC is a COM/DCOM based industrial standard which specifies the communication of real-time plant data between control devices from different manufacturers. By OPC technology, the problem of heterogeneity for bottom device driver design is solved effectively, and the development costs and incompatibilities between different devices are reduced. Meanwhile, the unified interface shielded the difference of bottom devices and improved the performance of industrial control systems. But this resolution can only be used in windows platform. CORBA is an open industry standard developed by OMG. The resolution provides the capability of platform independence, program language independence and transparent message passing, but it is too huge or too complicated.

Currently, the hosts of calibration system are largely PC and industrial computer, and they are based on Windows platform mostly. Considering that the scale of calibration system is small generally, the OPC approach is more suitable for proposed method than that of CORBA.

B. *Architecture of OPC based General Calibration System*

The calibration of embedded control system is a process that adjusts the control parameters of embedded controller to optimize the system working state. A typical calibration system should have the functions of data acquisition, display, modification and storage, etc. Beyond the basic features, the presented calibration system is required to satisfy different calibration parameters of embedded controllers, HMI (human-machine interface), calibration protocols and detail of bottom devices based on various communication buses. To achieve the goals above, the calibration host software must be with sufficient generality and adapbility.

The configuration technique based approach introduces a XML (eXtensible Markup Language) file as interface to describe the configuration of HMI and parameters, and separates the calibration software as editing environment and running environment. The editing environment provides a visualized development interface to customize the manifestation of calibration parameters as XML configuration file, and the running environment parses the XML file to generate the HMI of calibration host software automatically. Once the parameters or embedded controller are updated, only the customizing of HMI in the editing environment is necessary, any new update is not required. Then adaptability for different calibration parameters and HMI is obtained.

Ulteriorly, by integrating the configuration and OPC approaches, a new solution with more generality and adapbility can be introduced to adapt various calibration protocols, bottom devices and communication buses. Then the generality of calibration system is further improved. The architecture of the corresponding calibration system is shown in Figure 4.

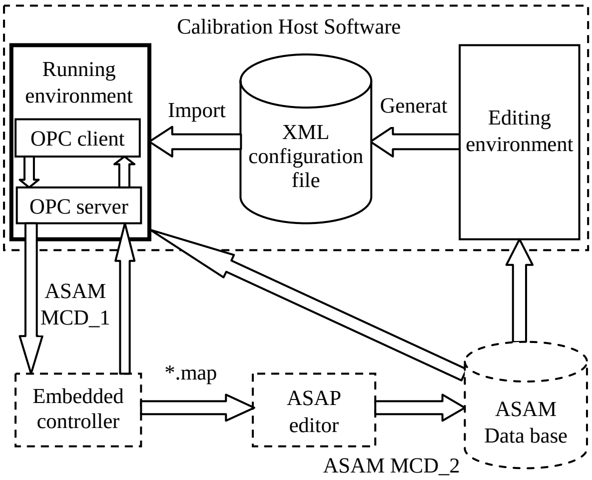


Figure 4. Architecture of OPC based general calibration system

In the architecture, the calibration system running environment is separated into OPC server and client. The OPC server encapsulates calibration protocols and the driver of communication device, and provides a unified data access interface of embedded controller for the OPC client. Then the OPC client would be unrelated to the details of calibration protocol and communication device, and acquires and modifies the parameters of embedded controller through the OPC interface. The structure of running environment based on OPC is shown in Figure 5.

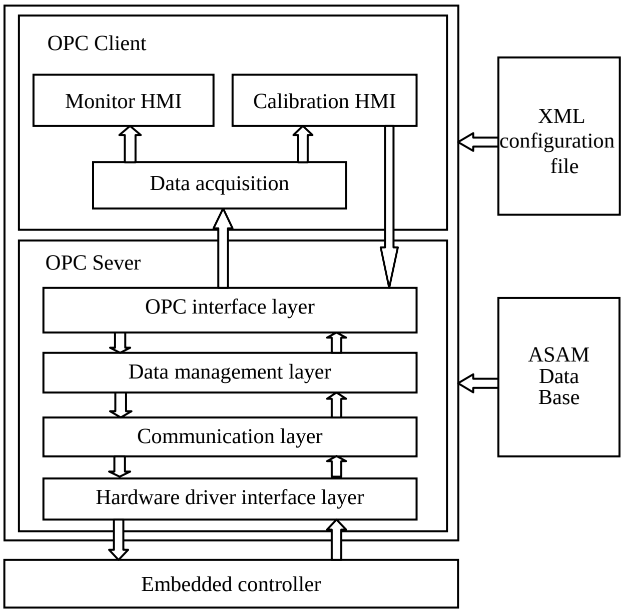


Figure 5. Structure of calibration system based on OPC

In the structure, like the user interface layer of the conventional calibration system, the OPC client provides HMI to implement measurement and calibration. Corresponding to measurement and calibration function respectively, there are two types of HMI that is the monitor interface and the calibration interface. The monitor interface is used for displaying the running state of the embedded controller in various forms, and the calibration interface are used for manipulating calibration parameters generally. Similarly, like the communication layer of the conventional calibration system, the OPC server accesses the parameters of the embedded controller by CCP or XCP protocol, but it provides the unified communication interface to upper OPC client. The OPC server consists of communication hardware driver, calibration protocol, data buffer and OPC interface. With the standard OPC interface, the detail of hardware driver and calibration protocol is concealed.

Furthermore, because an OPC client can access multiple servers simultaneously, the client and server of OPC based calibration system can be deployed at different network nodes to support distribute calibration. The OPC client can access data from different embedded controllers with different protocol and hardware concurrently. So repeated development of calibration system for different embedded controller is reduced, and the universality of the calibration system is improved to enhance the development efficiency of the embedded software.

C. *Design of OPC Server*

As the same in the field of industrial control, the OPC server of calibration system is designed to shield the difference of bottom devices. Standard OPC server is a COM component according with OPC specification, an industrial standard established by OPC foundation. As the most foundational OPC specification, data access specification defines a mechanism for real-time data communication [20]. According to the functions, the OPC server can be divided into four layers, communication interface layer, communication layer or calibration protocol stack, data management and OPC interface layer. The OPC server structure of calibration system is shown in Figure 6.

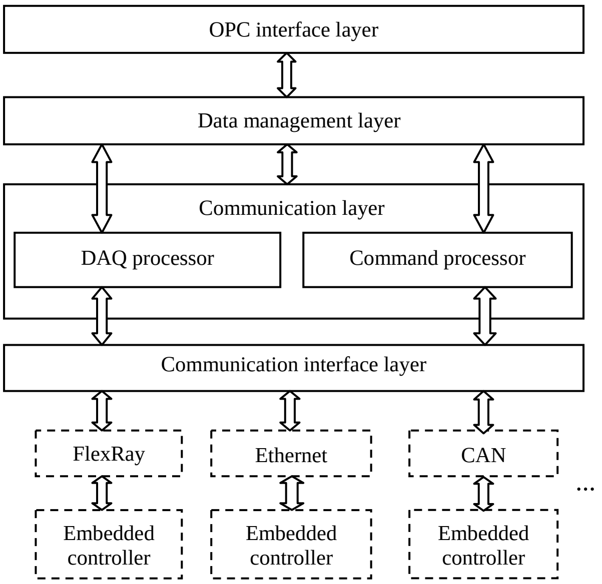


Figure 6. Structure of calibration system OPC server

The communication interface layer is the lowest layer of the OPC server, which includes different driving interfaces provided by different physical layer devices and different communication interfaces provided by various buses. Because the OPC client accesses data of embedded controller by OPC interface rather than driver of bottom device, even if the device or driver changed, only the corresponding OPC server is required to be replaced, and it is not necessary to update the calibration system to adapt to new hardware. Then the calibration system can support various devices and buses, such as CAN, LIN, Ethernet, FlexRay and so on.

The communication layer encapsulates calibration protocols. There are two typical calibration protocols supporting different buses, CCP and XCP. Similarly, the OPC client does not access calibration protocol stack directly, so only the corresponding OPC server is required to be replaced when calibration protocol is varied. Then the generality of calibration system is improved correspondingly.

The data management layer provides the management for two kinds of data, one is metadata from ASAM database, and the other is measurement and calibration data from embedded controller. According to the ASAM architecture, the metadata from ASAM database describes the information of the parameters of embedded controller, such as variable name, data type, address and so on. The OPC server configures the DAQ-ODT (Data AcQuisition - Object Descriptor Table) tables of embedded controller with the metadata, and then the controller can send specified data to the calibration host automatically according to DAQ-ODT configuration. The latter provides a data buffer to temporarily store the real time measurement data from the controller as well as the calibration data from the OPC client. Essentially, the data buffer is equivalent to a map of the embedded controller memory.

The OPC interface layer provides standard interfaces according with OPC data access specification. By the OPC interface, the calibration data from client can be sent to the controller, and the measurement data from the controller can be acquired also. Then the detail of communication devices and calibration protocols are concealed completely.

D. *Design of OPC Client*

The goal of calibration system client is to provide a friendly HMI and effective communication capability. The former can be implemented by user interface customizing and configuration, and the latter can be implemented by accessing embedded controller through OPC Server.

The OPC client of calibration system consists of three layers, HMI, data management and OPC interface layer. The OPC client model is shown in Figure 7.

The OPC interface layer of the client provides an access mechanism to OPC server. By the interfaces, the callback mechanism is implemented to achieve bidirectional communication between client and server. Then the OPC client can send calibration data to the controller, and acquire the measurement data of the controller from OPC server asynchronously by polling or publish/subscribe mode.

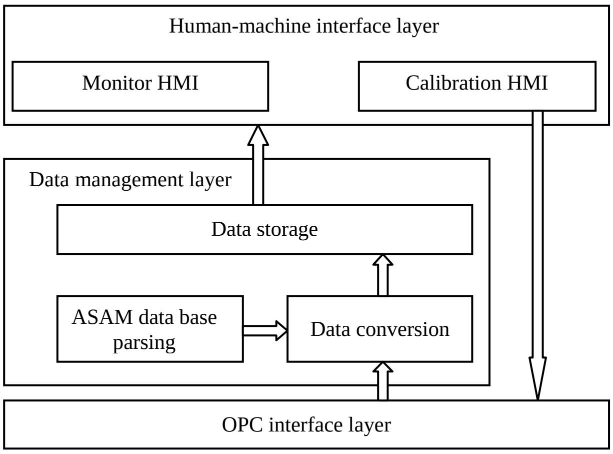


Figure 7. Model of calibration system client

The client side data management layer parses ASAM database to obtain calibration metadata, and provides a data buffer to store the real time measurement data of embedded controller from OPC server and the calibration data. Similarly, the data buffer is corresponding to the memory of embedded controller, but the data is converted according to calibration metadata.

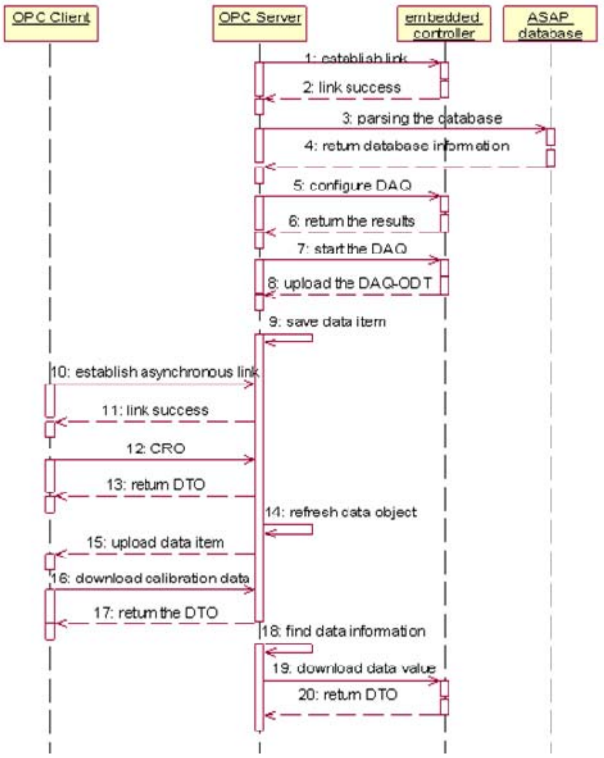


Figure 8. Sequence diagram of OPC based calibration system

The human-machine interface layer includes calibration and monitor interfaces to display calibration and measurement parameters respectively. Calibration users can modify and calibration parameters through calibration interfaces, then the modified parameters should be sent to the server through OPC interface timely. Then OPC server sends the data to embedded controller after some data conversion. The monitor interfaces update the display of measurement data periodically by various styles, such as scopes, meters or grids.

The OPC client and server described above are integrated to realize the host software of calibration system. The sequence diagram of interaction process between OPC server and client is shown in Figure 8. During the calibration process, the OPC server connects to the embedded controller firstly, and configures the DAQ-ODT tables of the controller according to parsed calibration metadata, then starts the DAQ command to obtain the controller parameters continuously. Secondly, the OPC client connects to the server, and begins to perform measurement and calibration.

IV. TEST AND APPLICATION

To verify the effectiveness of the method presented, the calibration system according with the method is developed, and used for the calibration of an engine ECU. In the calibration bench shown in Figure 9, an industrial PC with USB interface is as host computer, and an USB-CAN adapter is used to connect the host and the ECU.


Calibration host USB-CAN Engine ECU

Figure 9. Calibration bench for engine ECU

The OPC server is corresponding to the USB-CAN adaptor, and CCP is used as calibration protocol. Both the OPC server and client of calibration host software are installed at the industrial PC.

In calibration experiment, the OPC server runs firstly, and a series of initialization and configuration operations are executed, such as importing the ASAM database to obtain meta data of calibration parameters, connecting to the engine ECU and configurating DAQ-ODT tables. Then measurement data of the ECU running state is sent to the OPC server continuously.

Subsequently, the OPC client starts and connects to the OPC server. Then the ASAM data base and XML configuration file are imported to generate the calibration and monitor interfaces automatically, and the measurement data are acquired and displayed continuously. At the same time, the control parameters of engine ECU can be updated at HMI, sent to OPC server by OPC interface, and send to ECU afterwards to achieve the calibration. The corresponding host software HMI of engine ECU calibration system is shown in Figure 10.

By means of the calibration bench and corresponding measurement instruments and equipments, the engine ECU is calibrated to obtain a group of optimal control parameters. The main parameters include fuel injection pulse width, ignition advance angle and so on. Because the fuel injection pulse width influences the air-fuel ratio directly, the calibration of the parameter is one of the most important parts for engine ECU. Figure 11 shows the calibration results of fuel injection pulse width as map graph, where blue-axis denotes the engine speed and the red-axis denotes intake manifold pressure. Figure 12 is the corresponding fuel injection pulse width table.

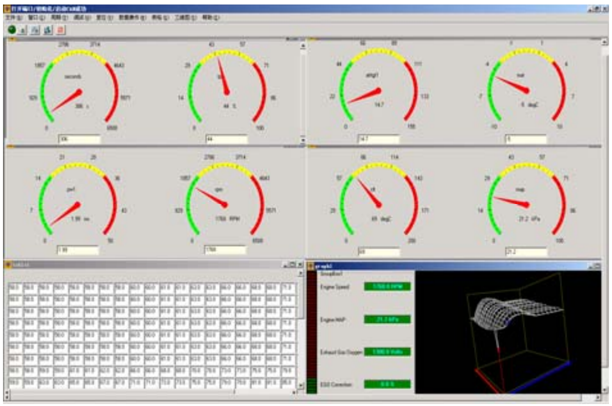


Figure 10. The client interface of ECU

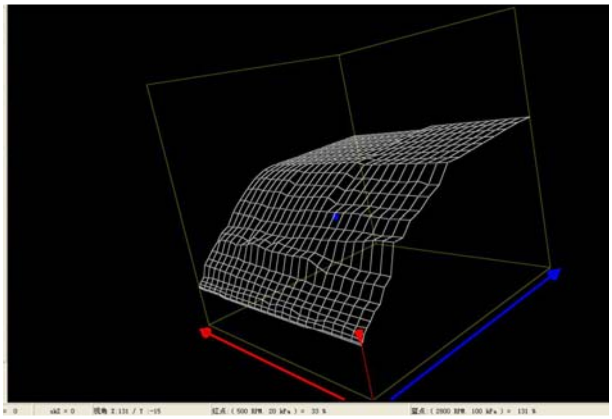
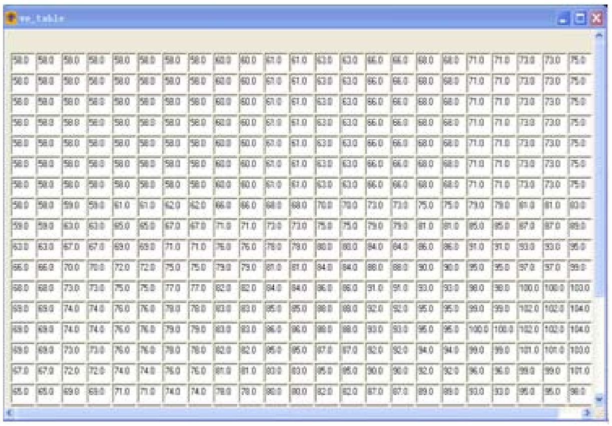


Figure 11. MAP graph of fuel injection pulse width



ve_table

58.0	58.0	58.0	58.0	58.0	58.0	58.0	58.0	60.0	60.0	61.0	61.0	63.0	63.0	66.0	66.0	68.0	68.0	71.0	71.0	73.0	73.0	75.0
58.0	58.0	58.0	58.0	58.0	58.0	58.0	58.0	60.0	60.0	61.0	61.0	63.0	63.0	66.0	66.0	68.0	68.0	71.0	71.0	73.0	73.0	75.0
58.0	58.0	58.0	58.0	58.0	58.0	58.0	58.0	60.0	60.0	61.0	61.0	63.0	63.0	66.0	66.0	68.0	68.0	71.0	71.0	73.0	73.0	75.0
58.0	58.0	58.0	58.0	58.0	58.0	58.0	58.0	60.0	60.0	61.0	61.0	63.0	63.0	66.0	66.0	68.0	68.0	71.0	71.0	73.0	73.0	75.0
58.0	58.0	58.0	58.0	58.0	58.0	58.0	58.0	60.0	60.0	61.0	61.0	63.0	63.0	66.0	66.0	68.0	68.0	71.0	71.0	73.0	73.0	75.0
58.0	58.0	58.0	58.0	58.0	58.0	58.0	58.0	60.0	60.0	61.0	61.0	63.0	63.0	66.0	66.0	68.0	68.0	71.0	71.0	73.0	73.0	75.0
58.0	58.0	58.0	58.0	58.0	58.0	58.0	58.0	60.0	60.0	61.0	61.0	63.0	63.0	66.0	66.0	68.0	68.0	71.0	71.0	73.0	73.0	75.0
58.0	58.0	59.0	59.0	61.0	61.0	62.0	62.0	66.0	66.0	68.0	68.0	70.0	70.0	73.0	73.0	75.0	75.0	79.0	79.0	81.0	81.0	83.0
59.0	59.0	63.0	63.0	65.0	65.0	67.0	67.0	71.0	71.0	73.0	73.0	75.0	75.0	79.0	79.0	81.0	81.0	85.0	85.0	87.0	87.0	89.0
63.0	63.0	67.0	67.0	69.0	69.0	71.0	71.0	76.0	76.0	78.0	78.0	80.0	80.0	84.0	84.0	86.0	86.0	91.0	91.0	93.0	93.0	95.0
66.0	66.0	70.0	70.0	72.0	72.0	75.0	75.0	79.0	79.0	81.0	81.0	84.0	84.0	88.0	88.0	90.0	90.0	95.0	95.0	97.0	97.0	99.0
68.0	68.0	73.0	73.0	75.0	75.0	77.0	77.0	82.0	82.0	84.0	84.0	86.0	86.0	91.0	91.0	93.0	93.0	98.0	98.0	100.0	100.0	103.0
69.0	69.0	74.0	74.0	76.0	76.0	79.0	79.0	83.0	83.0	85.0	85.0	88.0	88.0	92.0	92.0	95.0	95.0	99.0	99.0	102.0	102.0	104.0
69.0	69.0	74.0	74.0	76.0	76.0	79.0	79.0	83.0	83.0	86.0	86.0	88.0	88.0	93.0	93.0	95.0	95.0	100.0	100.0	102.0	102.0	104.0
69.0	69.0	73.0	73.0	76.0	76.0	78.0	78.0	82.0	82.0	85.0	85.0	87.0	87.0	92.0	92.0	94.0	94.0	99.0	99.0	101.0	101.0	103.0
67.0	67.0	72.0	72.0	74.0	74.0	76.0	76.0	81.0	81.0	83.0	83.0	85.0	85.0	90.0	90.0	92.0	92.0	96.0	96.0	99.0	99.0	101.0
65.0	65.0	69.0	69.0	71.0	71.0	74.0	74.0	78.0	78.0	80.0	80.0	82.0	82.0	87.0	87.0	89.0	89.0	93.0	93.0	95.0	95.0	98.0

Figure 12. Fuel Injection Pulse Width Table

Replacing the USB-CAN adaptor by a PCI-CAN adaptor, developing corresponding OPC server for new adaptor, and repeating the calibration experiment, the same result as above is obtained. It is shown that while the bottom hardware interface changed, only the middleware is required to be updated, and the rest of calibration host software keeps the same. Subsequently, the vehicle test verified the effectiveness of the calibration parameters also. So the calibration system according with the method presented can adapt to different bottom hardware.

The adaptability of the calibration system to different communication buses and calibration protocols can be verified similarly. So the method presented improves the adaptability and generality, and the development cost and time of calibration system are reduced. Finally, the development efficiency and quality of embedded controller software are enhanced.

V. CONCLUSIONS

Calibration is one of the key technologies in the development of the software of embedded controller. The existing calibration system is strongly coupled with a specifically communication device or calibration protocol. It is difficult to meet the fast, efficient and reliable development requirement of the embedded software.

The OPC based calibration system presented is separated into OPC server and client to isolate the bottom hardware and calibration protocol with the rest. The OPC server masks the difference of various calibration protocols and detail of communication devices, and provides a unified data access interface. Once the hardware or calibration protocol changed or updated, only the OPC server is required to be updated correspondingly, and the rest of calibration system keeps the same. So the adaptability and openness of the calibration system is enhanced to improve the development efficiency of embedded software in industrial control fields.

ACKNOWLEDGMENT

This work is supported by Science and Technology Project of Chongqing Municipal Education Commission under the Grant No. KJ110521, and Chunhui Project of the Ministry of education of China under the Grant No. z2009-1-63019.

REFERENCES

[1] Kiesel Rainer, Streubühr Martin, Haubelt Christian, Löhlein Otto and Teich Jürgen, "Calibration and validation of software performance models for pedestrian detection systems," In *Int. Conf. Embedded Comput. Syst.: Archit., Model. Simul.*, IC-SAMOS, pp.182-189, 2011

[2] Huizong Feng, Ming Cen, Yu Zhang, Jianchun Jiang and Huasheng Dai, "A weak coupled calibration system architecture for electronic control unit," In *IEEE Veh. Power Propul. Conf., VPPC*, pp.1-4, 2008

[3] Bernd Hardung, Thorsten Kölzow and Andreas Krüger, "Reuse of software in distributed embedded automotive systems," In *Fourth ACM Int. Conf. Embedded Softw.*, EMSOFT, pp.203-210, 2004

[4] Manfred Broy, "Challenges in automotive software engineering," In *Int. Conf. Software Eng.*, ICSE'06, pp.33-42, 2006

[5] Klaus Grimm, "Software technology in an automotive company - Major challenges," In *Int. Conf. Software Eng.*, pp.498-503, 2003

[6] Th. Scharnhorst, H. Heinecke, K.-P. Schnelle and H. Fennel, et al. "AUTOSAR - Challenges and achievements 2005," *VDI Berichte*, no.1907, pp.395-408, 2005

[7] Ron Bell, "Introduction & revision of IEC 61508," *Measurement and Control*, vol.42, no.6, pp.174-179, 2009

[8] B. Dion and J. Gartner, "Efficient development of embedded automotive software with IEC 61508 objectives using SCADE drive," *VDI Berichte*, no.1907, pp.237-247, 2005

[9] M. Beham, M. Etzel, D. L. Yu, "Development of a new automatic calibration method for control of variable valve timing," *Proc. Inst. Mech. Eng. Part D J. Automob. Eng.*, vol.218, pp.707-718, 2004

[10] ASAM, "AE MCD-2MC: ASAP2 interface specification v.51, " *http://www.asam.net*, 2009

[11] S. Bienk, "ASAM ODX: syntax as semantics," In *Int. Conf. Software Eng.*, pp.583-592, 2008

[12] John Chatzakis, Kostas Kalaitzakis, Nicholas C. Voulgaris and Stefanos N. Manias, "Designing a new generalized battery management system," *IEEE Trans. Ind. Electron.*, vol.50, pp.990-996, 2003

[13] C. M. Vong, P. K. Wong and H. Huang, "Case-based reasoning for automotive engine electronic control unit calibration," In *Int. IEEE Conf. Inf. Autom., ICIA*, pp.1380-1385, 2009

[14] Shiwei Yang, Lin Yang and Bin Zhuo, “Developing a multi-node calibration system for can bus based vehicle,” In *IEEE Int. Conf. Veh. Electron. Saf., ICVES*, pp.199-203, 2006
[15] G. S. King, R. Peter Jones, Andrew D. Bailey, “Application of systems modeling and simulation in the discrete ratio automatic transmission calibration process for an automotive,” *ME Dyn Syst Control Div Publ DSC*, vol.72, pp.935-944, 2003
[16] Xiaojun Fang, Yinnan Yuan, Jiayi Du, Xing Wu and Ming-quan Jia, “Development of ECU calibration system for electronic controlled engine based on Labview,” In *Int. Conf. Electr. Inf. Control Eng., ICEICE - Proc.*, pp.4930-4933, 2011
[17] Ming Cen, Yi Yan and Huasheng Dai, “General calibration system architecture of automotive electronic control unit,” *Journal of computers*, vol.5, pp.1894-1898, 2010
[18] G. Blair, A. T. Campbell and D. C. Schmidt, “Middleware technologies for future communication networks,” *IEEE Network*, vol.18, pp.1-4, 2004
[19] Al Chisholm, “Technical overview of the OPC data access interfaces,” *ISA TECH EXPO Technol. Update*, vol. 2, no.1, pp.63-72, 1998
[20] Renjie Huang and Feng Liu, “Research on OPC UA based on electronic device description,” In *IEEE Conf. Ind. Electron. Appl., ICIEA*, pp.2162-2166, 2008
[21] F. G. Chatzipapadopoulos, M. K. Perdikeas and I. S. Venleris, “Mobile agent and CORBA technologies in the broadband intelligent network,” *IEEE Commun. Mag.*, vol.38, pp.116-124, 2000

Ming Cen received the PhD degree in Optical Engineering from Graduate University of the Chinese Academy of Sciences. He had worked on a number of projects related to automation and communication techniques. His research interests include information fusion, target tracking and recognition, embedded system and intelligent vehicle.

Qian Liu and Yi Yan are Master Degree Candidates of Chongqing University of Posts and Telecommunications in Control Theory and Control Engineering. The main research interest is automotive electronics and embedded system.