Design of a Distributed Multiaxis Motion Control System Using the IEEE-1394 Bus

GuoYing Gu, LiMin Zhu, ZhenHua Xiong, and Han Ding, *Senior Member, IEEE*

***Abstract*—This paper presents a distributed multiaxis motion control system based on the IEEE-1394 bus. In the system design, the Unified Modeling Language is employed to illustrate interactions of the objects required in the system. The developed system consists of a set of smart distributed control nodes connected one by one through the IEEE-1394 bus. Each smart node contains four modules, i.e., an IEEE-1394 interface module, a digital signal processor module, a field-programmable gate array module, and a digital-to-analog converter module. It accomplishes its own control task and coordinates with the others through information exchanges, sampling sensor signals, and controlling actuators. A scheduled communication protocol is proposed according to the criteria in terms of bounded time delay and guaranteed transmission. Time delays arising from data processing and message transmission are analyzed. A platform is built, and experiments are conducted to demonstrate the capabilities of the developed distributed control system for real-time communication and synchronous tracking control, which are required for multiaxis applications. The results verify the feasible application of the IEEE-1394 bus to distributed motion control.**

***Index Terms*—Distributed control system, IEEE 1394, real-time communication, smart motion control node, synchronous servo control.**

I. Introduction

ALONG WITH the rapid development of industrial automation and computer integrated manufacturing systems, the number of sensors and actuators that should be coordinated increases quickly in many areas like computer numerical control (CNC) machines, flexible manufacturing systems, and robots. The requirement for multiaxis control [1] arises whenever the axes move together to realize coordinated operations. For instance, a CNC machining center generally has five or even more axes to be servo actuated, and a humanoid robot generally has more than 30 DOFs to be controlled simultaneously. From a multiaxis control perspective, the distributed system architecture is of practical interest [2]. Such a conception of distribution is put forward on the application of real-time communication technology for information exchanges and synchronization.

Manuscript received June 10, 2009; revised October 15, 2009 and December 24, 2009; accepted February 2, 2010. Date of publication March 1, 2010; date of current version November 10, 2010. This work was supported in part by the National Natural Science Foundation of China under Grant 91023047 and in part by the Science and Technology Commission of Shanghai Municipality under Grants 09520701700, 09JC1408300, and 08JC1411600.

The authors are with the State Key Laboratory of Mechanical System and Vibration, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: guguoying@sjtu.edu.cn; zhulm@sjtu.edu.cn; mexiong@sjtu.edu.cn; hding@sjtu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2010.2044127

In a distributed control system, a large amount of control information is exchanged between the host and distributed nodes. The important and challenging issues are how the control information is accurately and instantly transmitted and how each individual controller synchronously executes the control information. Thus, the real-time control network becomes an emerging technology that attracts research attention for distributed control systems [3]–[8]. The networked control technology (NCT) with real-time transmission rate provides several benefits such as efficiency, flexibility, and reliability. NCTs also reduce the problems of wiring connection and transmit-length limitation. During the past decade, many different networked technologies like control area network (CAN), process field bus, factory instrumentation protocol, and serial real-time communication specification (SERCOS) have been presented in distributed motion control systems [8]–[13]. Recently, several researchers [14]–[16] focus on data-compression methods with reducing transmission information and quantization problems at a limited channel rate. However, these research works suffer from drawbacks. The first one is that some of their developments are not based on the standardized protocol stack like IEEE standards. The second one is that the bandwidth of these field buses is limited and becomes the bottleneck to support high-speed communication for multiaxis control. Although the standardized Ethernet has been used, the nondeterminism of carrier-sense multiple access with collision detection protocols is inherently not suitable for a real-time control system due to the random networked delay [8], [14]. More attention should be paid to eliminate collisions and guarantee determinism.

Compared with the aforementioned NCTs, IEEE 1394 [17]–[19] can completely overcome the existing drawbacks. First, IEEE 1394, established in 1995, is an industry data interface to standardize data structure and communication. A relative higher bandwidth (maximum of 400 Mb/s in IEEE 1394a and maximum of 3.2 Gb/s in IEEE 1394b) can give enough flexibility and expandability in multiaxis control systems. Two types of basic protocol data unit: asynchronous (guaranteed delivery) and isochronous (guaranteed bandwidth) are well suitable for almost all control requirements in real-time control systems. All data, using data-strobe encoding, is sent along the IEEE-1394 serial bus in serious 4 B (32 b is called a *quadlet*), encoded together with their clock signal onto two nonreturn-to-zero bus signals. This improves transmission reliability by ensuring that only one of the two signals changes in each data bit period. Second, conforming to the ISO/IEC 13213 (ANSI/IEEE Standard 1212) and control and status register (CSR) architecture with 64-b fixed addressing, the IEEE-1394 family accommodates 63 nodes at most for a signal bus bridge. The cable

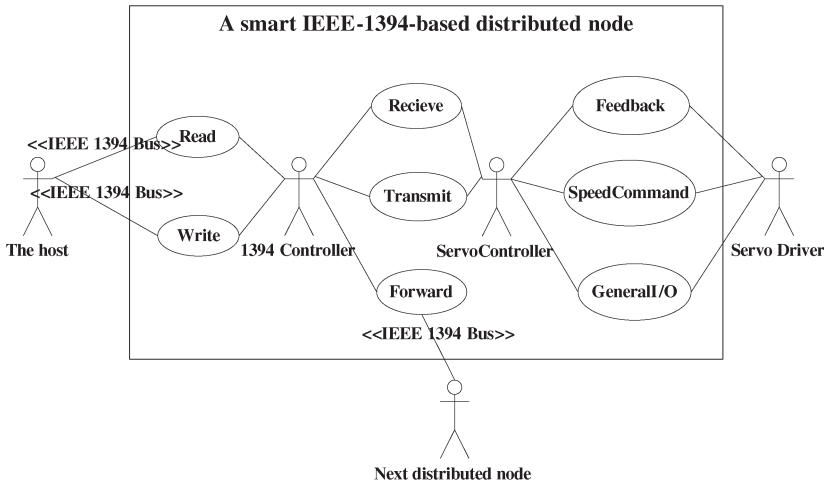

Fig. 1. Use-case model for the distributed control system.

medium allows up to 16 physical connections with standard cables up to 4.5 m in length for a total cable length of 72 m. Finally, as a high-performance serial bus, the IEEE-1394 family defines a serial data-transfer protocol with a reduced-layer model (including transaction layer, link layer, and physical layer) and kinds of arbitration to satisfy real-time communication. A flexible bus management, which recognizes the configuration to optimize the performance of the serial bus, is also provided to offer connectivity between different devices. This assures easy setup, maintenance, diagnostics, and bus supervision without including a personal computer (PC) or any other bus controller. Therefore, the IEEE-1394 family performs excellent network interconnection as modern IEEE standard parallel buses but at a much lower cost [20]. In such a scenario, the IEEE-1394 bus takes full advantages of high speed, deterministic transmission, low cost, and robust and wide industrial support and gives high expandability and openness without bundling to any manufactory.

Nowadays, IEEE-1394 interfaces are widely used in computer graphics systems and consumer electronics industries and increasingly found even inside large workstations and servers to interface peripherals [21]–[23]. Until now, few publications about the application of IEEE 1394 to practical distributed multiaxis control systems are available. Therefore, an IEEE-1394 interface has been studied in this paper to satisfy the requirements of high-speed and real-time information exchanges in a distributed control system. Based on the developed IEEE-1394 interface, this paper also develops an event/time-triggered communication protocol to ensure the distributed control system obtain better servo-control performance.

With the rapid development of microelectronics, microprocessor, and very large scale integrated technologies, digital signal processors (DSPs) and field-programmable gate arrays (FPGAs) are widely adopted to realize a real-time motion control system with a shorter cycle time. DSP, integrating a variety of sophisticated power-electronic peripherals, has been adopted to implement multiple functions for motion control [24]–[26]. This application of DSP core not only simplifies the design process but also offers the capability of incorporating various extra features. FPGA with deterministic computation times through parallel architectures can create custom hardware logical circuits that cannot be realized by DSP. FPGA has been widely used in areas of motion control [27]–[30] to structure reconfigurable systems and to develop high-performance algorithms. These microprocessor-based systems fill the flexibility, performance, power dissipation, and development cost gap. In this paper, DSP is used for servo-control loop and real-time communication with protocols, while FPGA is adopted for sampling sensor signals and controlling actuators.

The structure of this paper is given as follows. Section II is dedicated to the system modeling. Section III illustrates the system architecture and hardware design, software structure, and user-level communication protocol. Section IV outlines the experimental implementation. In the last section, a conclusion and future research perspectives are given.

II. System Modeling

Multiaxis motion control is a real-time control task that maintains an ongoing and timely interaction with the environment. The unified modeling language (UML) is an object-oriented and object-based modeling language [31] and is suitable for modeling real-time systems [32]. UML provides a form of abstraction that facilitates both problem understanding and solving. Although modeling a complex real-time control system needs several software tools [32], here, we are concerned with the functional specifications and the communication relationships among different modules of the developed system, which can be well handled by UML. Therefore, UML is employed to illustrate the interactions of the objects required in the developed system and describe an instance model of an integrated framework, including architecture and behavior descriptions. As shown in Fig. 1, a use-case model is constructed to capture the significant features and instances of the designed system and to illustrate the communication relationships among them. All the distributed nodes have the same structure and are connected one by one with the direction pointing toward the host. In this model, two distributed nodes are given as an example. The host executes coordinated control functions (such as multiaxis interpolation) and transmits control commands to the distributed nodes through the IEEE-1394 bus. Each distributed node contains a 1394 controller and a motion controller.

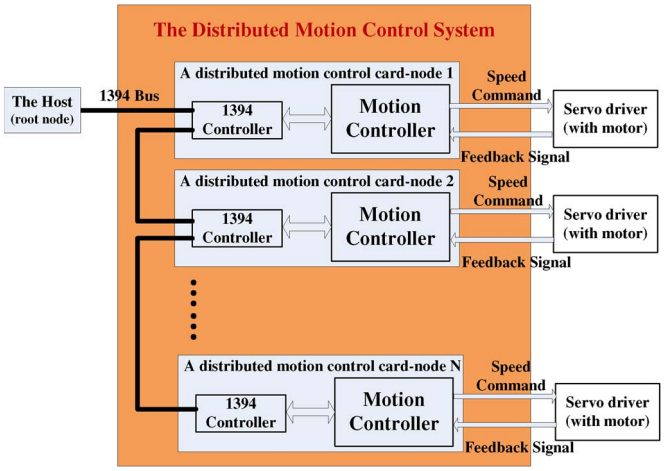

Fig. 2. Topology of the distributed multiaxis motion control system.

The 1394 controller performs IEEE-1394 interfaces in terms of information exchange and transmission, while the motion controller is responsible for servo-loop control, sampling sensor signals, and controlling actuators. With the requirement of real-time control, the main issue is how to effectively perform information exchanges within a predefined period. The time-division multiple-access (TDMA) methodology is adopted to reserve network access for every distributed node at a specific time slice [33]. Supposing that the host controls N distributed nodes, the key consideration is the communication time slice $(T_{\rm cs})$ for every distributed node, which can be expressed in the following:

$$T_{\rm cs} = \frac{\alpha T_s}{N} \tag{1}$$

where T_s is the predesigned communication period and α, varying from zero to one, is the ratio of communication time slice.

III. Design and Development

The development of a distributed control system requires a concurrent design approach that can effectively integrate the interactions between real-time and non-real-time requirements and the corresponding technique. In this paper, the proposed system is designed in three steps [2], [13]: system architecture with the functional specification of different elements, hardware design with a set of modules and their communication accesses (connections and protocols), and software structure with standard modules. In this section, the real-time communication quality of the IEEE-1394 bus is also discussed.

A. System Architecture

Generally speaking, the architecture of a distributed system should be universal and open enough, but an individual user just handles software with a given performance to achieve an open modular design [34]. The developed system contains a set of smart distributed control nodes connected one by one through the IEEE-1394 bus. Each smart node contains modules with different functions. The topological architecture of the system is shown in Fig. 2. In this design, the host implements trajectory interpolation, while the distributed nodes perform position-loop control in terms of speed and parallelism. End users can modify and update their specific kinematics-dependent interpolation only by changing the software in the host without violating the integrity and reliability of the distributed system. Position-loop control, and even velocity-loop control, can also be integrated in the host [33] for soft-CNC architectures. However, this is beyond the scope of this paper. The main purpose of this paper focuses on how the distributed nodes, using the IEEE-1394 bus, receive interpolation data synchronously and realize multiaxis control simultaneously.

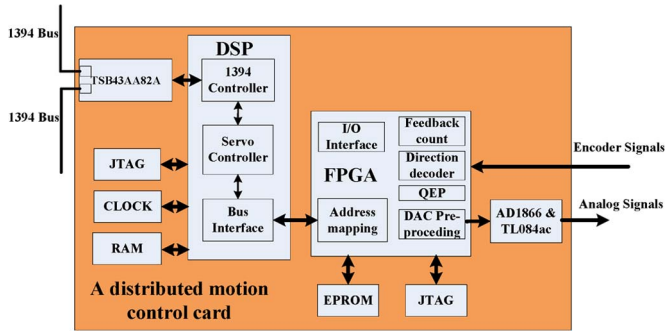

Fig. 3. Hardware structure of a distributed motion control card.

B. Hardware Design

The hardware design is obtained by mapping of the functional distribution onto the concurrent and modular structures. The existing hardware structures of motion control systems include the following: 1) purely adopting a DSP; 2) using a DSP and application-specific integrated circuit; 3) utilizing both a DSP and a FPGA; and 4) using multiple DSPs [24], [35]–[37]. The single DSP-based structure cannot fully take advantage of some advanced control algorithms due to the limited computational capability. An FPGA with deterministic computation times through parallel architectures can create custom hardware logical circuits that cannot be realized by a DSP. The utilization of FPGA also enhances the computation capability and relieves the load of DSP. The hardware configuration of FPGA is simplified just through modifying the software codes. Therefore, the size of the control system can also be reduced. In this paper, the combination of DSP and FPGA, which is the most popular control hardware structure [24], is adopted. The FPGA is just considered as an external device of the DSP, and both of them have the same clock cycle. In order to realize concurrent and modular designs, the DSP accesses the control information through a configurable register interface without changing the codes of the FPGA.

Subsequently, the overriding consideration is the issue of which motor the system should be oriented toward. Due to the applications of servo control, an ac permanent-magnet synchronous motor (PMSM) is preferred. The standard analog drive is adopted to ensure a problem-free interaction between the proposed system and PMSM drivers from different manufacturers [3]. In this paper, the PMSM drivers of Panasonic Company are adopted, and they operate in velocity control mode.

As shown in Fig. 3, the hardware structure of each distributed node include the following: an IEEE-1394 interface module,

a DSP module, an FPGA module, and a digital-to-analog converter (DAC) module.

1) IEEE-1394 Interface: The IEEE-1394 chip TSB43AA82A [38] from Texas Instruments (TI), compatible with IEEE 1394-1995 and IEEE 1394a standards, integrates both PHY and link layer with two connectors. This chip provides all necessary CSRs and three first in/first out (FIFO) for sending and receiving packets. In case of asynchronous transaction, *quadlets* are written to the asynchronous transmit FIFO to transmit a packet, depending on the configuration of the 1394 controller embedded in the DSP. To read a receiving packet, *quadlets* are read from the asynchronous receive FIFO. This chip also implements arbitration, transaction manager, ping function, autoresponse, and other functions, which can satisfy high-speed and reliable communication. In order to realize communication properly, three phases, namely, bus initialization, tree identification, and self-identification, must be done first during the cable configuration. In the developed system, the 1394 controller embedded in the DSP module performs this work by configuring the *ConfigRom* of the TSB43AA82A chip.

2) DSP Module: The specific DSP device chosen for this paper is a TMS320F2812 [39] chip from TI, which provides a maximum of 150 MIPS and lots of on-chip resources compared to the proposed requirements. This module consists of three functions. The first one is a 1394 controller, which configures the TSB43AA82A chip and implements the transaction-layer application interface according to the IEEE-1394 standard. This controller is dedicated to real-time communication with the IEEE-1394 chip and the DSP. The second one is a servo controller, which contains a servo loop with a proportional–integral differential (PID) filter with velocity and acceleration feedforward compensation. In particular, other advanced algorithms can easily be updated to improve the control performance because it is realized as a standard software interface function. The third one is a bus interface with the FPGA module through address mapping. The DSP accesses the control information through the configurable register interface without changing the codes of the FPGA. With the purposes of optimizing performance, increasing reliability, and improving synchronization in the proposed system, an event/time-triggered software structure is developed to ensure real-time communication and synchronous control, which will be discussed later.

3) FPGA Module: In this paper, the FPGA device EP2C35F484I8 [40] from ALTERA is used to improve the control performance in closed-loop motion control by replacing the very time-consuming software process with cost-effective hardware solutions. This module consists of several functions in terms of quadrature encoder pulse (QEP) circuit, feedback count, direction decoder, address mapping, DAC preprocessing circuit, and input/output (I/O) interface. The QEP circuit is designed to capture multiple encoder signals, and the feedback count circuit is adopted to improve the accuracy of orientation. The direction decoder is applied to determine the rotary direction of the motors. The DAC preprocessing circuit realizes dividing frequency for the DAC module. The DSP bus interface circuit provides signal communication between the FPGA and DSP through address mapping. The I/O interface implements interconnections with servo drivers and sensors.

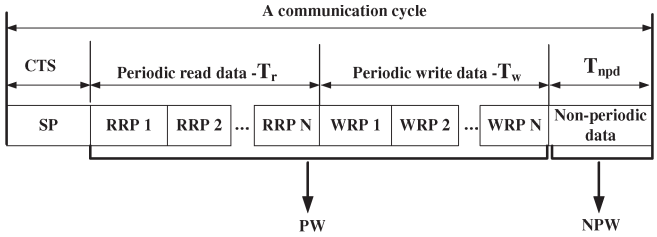

Fig. 4. Time schedule of HPSCP.

4) DAC Module: This module is developed to convert digital signals to analog signals and amplify the analog signals for servo drivers. Two 16-b dual serial-input and voltage-output AD1866R devices [41] are adopted to convert digital signals to analog signals, and a four-channel amplifier TL084ac [42] with the capabilities of low power consumption, high input impedance, and low total harmonic distortion is used to amplify the analog signals to 10 V.

C. HPSCP

The goal of this protocol is to provide high-efficiency communication quality of the IEEE-1394 bus to guarantee timely data transmission between the host and smart distributed nodes.

Although the communication time slice has been defined in (1), the synchronization of all network-based nodes is a consensus problem on the real-time system [43]. Since each node has its own time base, a common clock must be set up for the distributed system. A requester/responder mechanism is proposed to realize communication between the host (the master) and distributed nodes (the slaves). The global time reference is transmitted periodically by the host, while the distributed nodes recognize the reference time and actuate the individual axis synchronously based on the event/time-triggered protocols. In the following paragraph, the specifications of the time/event-triggered high-performance serial communication protocol (THPSCP) are discussed in detail. To realize THPSCP, the following three principles must be obeyed.

1) At the initialization phase, the host finishes the transmission test first and defines the exact time delay for each distributed node. The remote host manages all communication rules based on the time-triggered [44] mechanism with respect to the temporal behavior.
2) During an information communication cycle, the host communicates with each distributed node at a fixed time slice, including transmitting a global time reference, one-time read operation, and one-time write operation for the periodic data. An unfixed time slice is left for the nonperiodic data if necessary. Both the periodic and nonperiodic data are packed into telegram codes with a *Node_ID* [38] and assigned fixed block data.
3) Based on the event/time-triggered structure, the distributed nodes recognize the global reference time, sample the position signals, and update the actuation command. In particular, the distributed nodes never actively access the bus without the master's request.

According to these principles, the time schedule of the developed protocol is shown in Fig. 4. In case of a communication

cycle (a TDMA round), a synchronous packet (SP) is transmitted by the host at the beginning. The SP transmission delay between two nodes is a constant time slice, which is detected by the *ping* packet [38]. All distributed nodes recognize the reception of SP-triggered messages as a global time reference. Such a concept is formed based on the event-triggered structure of the distributed nodes. The SP-triggered messages convey two operations for each distributed node: sampling encoder signals and updating the last received command to servo drivers after the specific time. Subsequently, the protocol defines two windows: periodic window (PW) for real-time tasks and non-PW (NPW) for non-real-time tasks. The host transmits read-requester packets to the distributed nodes one by one. In its turn, each distributed node sends respond packets (RPs) with sampled data back to the host. Then, the host transmits write-requester packets to the distributed controllers, which, in its turn, send RPs back. At last, a remaining time slice is provided for the nonperiodic information transfer during NPW. Thus, the total time slice for each distributed node can be calculated by the following:

$$T_{\rm cs} = (T_r + T_w + T_{\rm npd} + CTS)/N. \tag{2}$$

D. Software Structure

The software structure of the proposed system has two modules, which are implemented on the host and each distributed node, respectively.

The software of the host implemented in a real-time operation system or a real-time microprocessor gives every process a fair share of computation time [45]. Based on HPSCP, the host consists of three threads: 1) sending the SP packet at a predefined period; 2) receiving packets with sensor signals from the distributed nodes; and 3) sending packets with motion commands to the distributed nodes. By this means, real-time communication is realized within a specified time. Otherwise, the host executes coordinated control functions, such as trajectory planning and interpolation. Hence, the α in (1) is selected as 0.5 in this paper.

The software of each distributed node is implemented on the DSP, which can fully satisfy the hard real-time requirement. Two modes, namely, lower level timer interrupt mode and higher level event interrupt mode, are designed. In the timer interrupt mode, the servo controller with a PID servo-loop-control filter [46] is executed to track the desired trajectory. The control parameters (static compensation, constant of proportionality, etc.) of the PID controller can be updated by the host through the IEEE-1394 bus to change the servo-control performance. The event interrupt mode with a higher priority is enabled once the IEEE-1394 chip receives a packet sent by the host. The 1394 controller in the DSP reads this packet and decodes the command. If the command is an SP-triggered message, the 1394 controller waits for a specific time slack, measured in the initialization phase, to ensure that every distributed node has a similar time criterion. Then, the 1394 controller immediately captures the current encoder signals and updates the previously received control command applied to the servo controller by using the shared-memory mechanism. If the command is a read packet, the DSP transmits a read response packet with current sampled encoder signals. If the command is a write packet, the DSP transmit a write response packet without valid data.

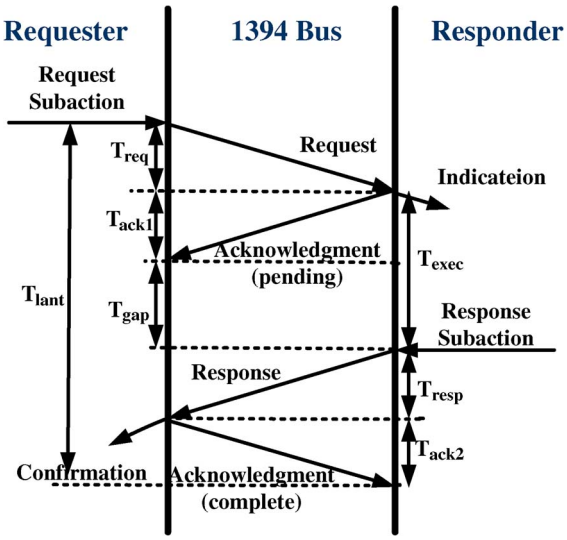

Fig. 5. Schematic runtime structure of the IEEE-1394 asynchronous transaction.

E. Real-Time Analysis

It is an elementary problem to evaluate the average latency and the jitter of an asynchronous transaction during a predefined communication period, such as in fault-tolerant transmission [47]. In order to measure the average packet transfer time (latency) for an asynchronous transaction, the timing constraint of the asynchronous transaction service is discussed first. As shown in Fig. 5, a dynamic runtime structure of the asynchronous transaction is schematically illustrated. Time delays arising from data processing and message transmission [7] are analyzed. In case of a read or write asynchronous operation, the time delay includes the require time ($T_{\rm req}$), the response time ($T_{\rm resp}$), the acknowledge time ($T_{\rm ack1}$ or $T_{\rm ack2}$), and the operation execution time ($T_{\rm exec}$). The total time delay (latency) is expressed by the following:

$$T_{\rm lant} = T_{\rm req} + T_{\rm ack1} + T_{\rm exec} + T_{\rm resp} + T_{\rm ack2}. \tag{3}$$

In IEEE-1394 cable operation mode, sending of an asynchronous packet is partitioned into four phases: arbitration phase (ArbP), data-transfer phase (DTP), acknowledgment phase (AckP), and packet-gap phase. As presented in the standard of IEEE 1394 [17], a set of equations can be derived as follows:

$$ArbP\_time = \left(0.023 + \frac{14}{Bandwidth}\right) \times N + \frac{Gap\_Count \times 4}{Bandwidth} + 1.63 \tag{4}$$

$$DTP\_time = \left(0.023 + \frac{14}{Bandwidth}\right) \times N + a + \frac{DataSize \times 8}{Bandwidth} \tag{5}$$

$$AckP\_time = 0.25 + \left(0.023 + \frac{14}{Bandwidth}\right) \times N \tag{6}$$

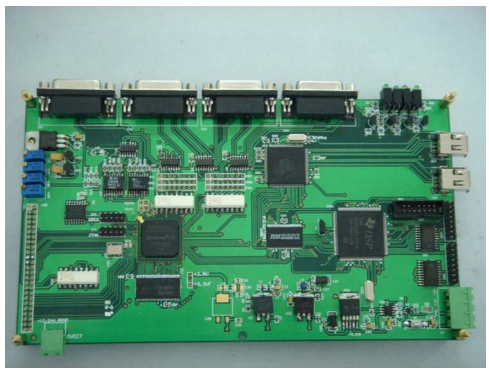

Fig. 6. Smart distributed multiaxis motion control node consisting of a DSP, an FPGA, an IEEE-1394 interface chip, a DAC, an operational amplifier, and servo-driver interfaces.

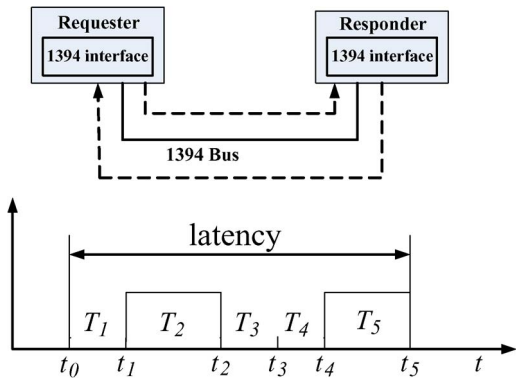

Fig. 7. Two-node communication test and latency scenario along the timeline.

where Gap_Count is the arbitration timer setting; N is the number of distributed nodes; $DataSize$ measured in terms of bytes is the data packet size, including data payload and overhead; and $Bandwidth$ is the data rate of the IEEE-1394 bus. Therefore, the theoretical latency can be estimated using the following:

$$T_{\text{req}} = ArbP\_time + DTP\_time \tag{7}$$

$$T_{\text{resp}} = ArbP\_time + DTP\_time \tag{8}$$

$$T_{\text{ack1}} = T_{\text{ack2}} = AckP\_time. \tag{9}$$

Based on the developed HPSCP, the host communicates with the distributed nodes several times during a predefined period. The required time for the subaction and arbitration reset gaps should be taken into account to determine the maximum number of nodes connected to a single bus. As discussed in [17], the sum of times for the subaction and arbitration reset gaps can be obtained using the following:

$$T_{\text{sum}} = \frac{82 + Gap\_count \times 48}{Bandwidth}. \tag{10}$$

Hence, the number of distributed nodes can be calculated by (11). Since each node can control maximally four axes, the limit of the number of control axes is $4N$

$$N = \frac{0.5 \times T_s - CTS}{2 \times (T_{\text{lant}} + T_{\text{sum}})}. \tag{11}$$

IV. Design and Development

Experiments are implemented to evaluate the performance of the designed system. In the developed platform, three smart distributed motion control cards are set up. A smart card is shown in Fig. 6. Considering future research, an Altera Cyclone II EP2C35F484I8 FPGA was adopted, although the controller circuits only took up approximately 10% of the total FPGA logic elements. The logic synthesis and fitting in FPGA used Quartus II from Altera, and the simulation and experiment in the DSP adopted Code Composer Studio from TI. In order to measure the communication time intuitively, an MSO4032 Mixed Signal Oscilloscope from Tektronix was used.

A. Real Time Performance

To analyze the timing constraint of the developed network, an experiment with two nodes is designed, as shown in Fig. 7. Before analysis, several definitions are discussed as follows.

1) t_0—The requester applies the bus.
2) t_1—The requester starts transmitting data.
3) t_2—The responder receives a packet and executes the commands.
4) t_3—The responder applies the bus.
5) t_4—The responder starts transmitting a response packet.
6) t_5—The requester receives the response packet.

According to (3)–(9) and Fig. 7, the following set of equations can be derived:

$$T_{\text{lant}} = T_1 + T_2 + T_3 + T_4 + T_5 \tag{12}$$

$$T_1 + T_2 = T_{\text{req}} + T_{\text{ack1}} \tag{13}$$

$$T_4 + T_5 = T_{\text{resp}} + T_{\text{ack2}}. \tag{14}$$

For evaluating the physical transmission delay between two nodes, a *ping* packet was used, and the transmission delay was adopted to synchronize the multiple nodes with the same time criteria. In our prototype, standard cables up to 4.5 m in length were used to connect two nodes, and the data rate of IEEE 1394 was 393.216 MHz. The overhead of each packet was 20 B. Once the *ping* packet was transmitted by the requester, the responder would immediately transmit a *Node_ID* packet in response, and this latency was detected automatically by the CSR of the requester. In this system, the average *ping* packet latency between two nodes was 300 ns. The latency would increase by 100 ns if another node was added in the system. In case of write transaction, a series of asynchronous write experiments had been done between the requester and the responder. The experimental results, detected by the DSP timer (150 MHz), from more than 10 000 times with different data are outlined in Table I. In this experiment, there was only one slave node (responder). N was selected as one. *Gap_count* was chosen as the most value of 63 with consideration of the worst

TABLE I
COMPARISON OF THEORETICAL AND EXPERIMENTAL TIMES

Data payload (Bytes)	$T_{lant}(\mu s)$ experiment	$T_1+T_2(\mu s)$ experiment	$T_1+T_2(\mu s)$ theory
4	48.7	6	3.19
16	48.9	6.3	3.44
32	57.9	6.6	3.76
64	68.1	7.5	4.41
128	99.9	9.1	5.71
256	158.5	11.5	8.32
512	246.0	16.9	13.53
1024	460.0	27.4	23.94

TABLE II
COMPARISON OF TRANSACTION LATENCY WITH THE SAME DATA PAYLOAD

Data payload (Bytes)	Developed system $T_{lant}(\mu s)$	M. Omar *et al.* $T_{lant}(\mu s)$
4	48.7	61/58
32	57.9	75/70
64	68.1	90/70
128	99.9	118/108
256	158.5	172/166
512	246.0	277/272

temporal behavior. The theoretical time was obtained through (4)–(9) and (13) as follows:

$$(T_1+T_2)_{\text{theory}} = 3.11 + \frac{Payload \times 8}{393.216}. \tag{15}$$

Since the experimental time was detected by the program on the DSP, the experimental results are a little larger than the theoretical ones. We can validate the correctness and real-time performance of communication via the IEEE-1394 bus. The main latency is T_3, which is consumed by the DSP to access the IEEE-1394 interface chip and execute the commands. In the designed system, time-consumed asynchronous mode is used to access the 1394 chip. The latency can be lower if direct memory access (DMA) bus mode [38] is adopted.

Compared with the related research of Sarker *et al.* [48], the developed system, using a real-time user-level protocol, consumes shorter transaction time with the same data payload, as reported in Table II. Through the test result, we can compare the relative performance of this system with that of the CAN-based system. As discussed in [48], the transaction time of the CAN-based system with 8-B data payload is 10 500 μs, where the CAN bandwidth is 1 Mb/s. Our proposed system just consumes 51.9 μs with the same data payload in the worst case, which is 1/202 of that of the presented CAN-based system.

B. Servo-Control Performance

A two-axis distributed control system is shown in Fig. 8. The experimental hardware is shown in the following: three smart distributed motion control cards, two servo drivers (Panasonic Company), and two ac servo motors (Panasonic Company) with pulse encoders. One smart distributed motion control card, used as the host, performed two-axis trajectory interpolation and transmitted the interpolation data to the distributed controllers every other T_s through the IEEE-1394 bus. The other two control cards controlled two motors, respectively. Both the basic DSP clock frequency and the basic FPGA clock frequency were 150 MHz. The encoder signal capture circuits could reduce the signal sampling period on the order of nanoseconds, and the maximum frequency of the input encoder signals could reach 30 MHz. In combination with the communication interface and general I/O signal processing, the position-loop period was chosen as 100 μs.

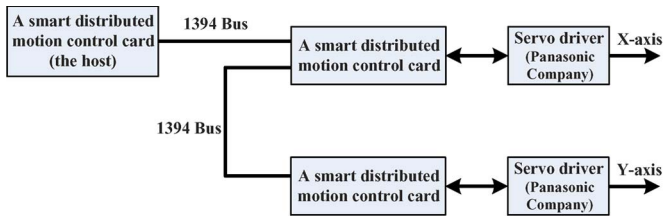

Fig. 8. Architecture of a two-axis motion control system.

Since the interpolation period in our CNC system was 4 ms, the communication period T_s was chosen as 4 ms in this test. Communication messages with a smaller period would result in high network traffic load, which could increase the possibility of data loss and induce the worse control performance. When the interpolation period in the CNC system approaches 1 ms in the future, the control performance should be guaranteed with the same data messages. Therefore, we also select the period as 1 ms in our test.

In this paper, the data payload was 12 B. An anticlockwise circle, whose radius was 320 mm, was chosen to determine the tracking performance of this developed prototype. When T_s was selected as 4 ms, the execution time of the experiment was 48.656 s, and the maximum velocity of the motors reached 2500 mm/min. When the communication period T_s was chosen as 1 ms, the maximum velocity of the motors reached 10 000 mm/min, and the execution time was reduced to 12.164 s at the cost of degrading the tracking performance. Fig. 9 shows the simulation trajectory (solid) and experimental trajectory (dashed) with different predesigned communication periods. The contour error is also shown in Fig. 10. When the communication period is 4 ms, the average value of error is 0.0036 mm, and the maximal value is 0.0187 mm. When the communication period is 1 ms, the average value of error is 0.0049 mm, and the maximal value is 0.0219 mm. The average experimental contour error using SERCOS, presented by Guo *et al.* [49], is about 0.0448 mm. The error of our system is just below 10% of that from the SERCOS-based system. With the comparison of the two experiments, we validate the performance of the developed system in terms of real-time communication and synchronous servo control. Otherwise, we just adopt the basic PID control law for position-loop control. The contour error will be decreased if an advanced control law is implemented.

V. SUMMARY AND OUTLOOK

In this paper, an IEEE-1394-based distributed multiaxis motion control system, along with a real-time user-level communication protocol, has been developed. A platform has been

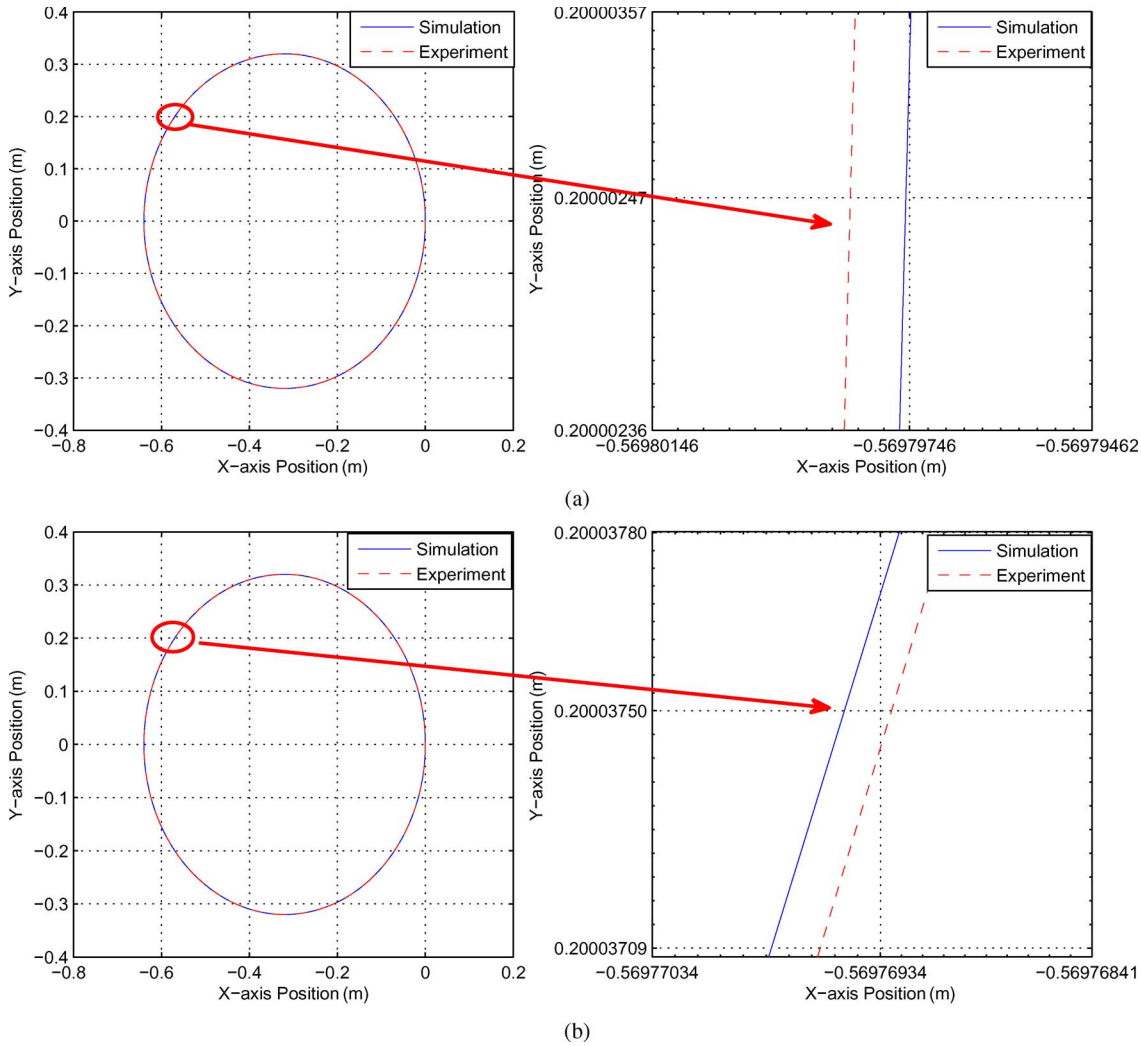

Fig. 9. (Solid) Simulation circle curve and (dashed) experimental results with different T_s's. (a) $T_s = 1$ ms. (b) $T_s = 4$ ms.

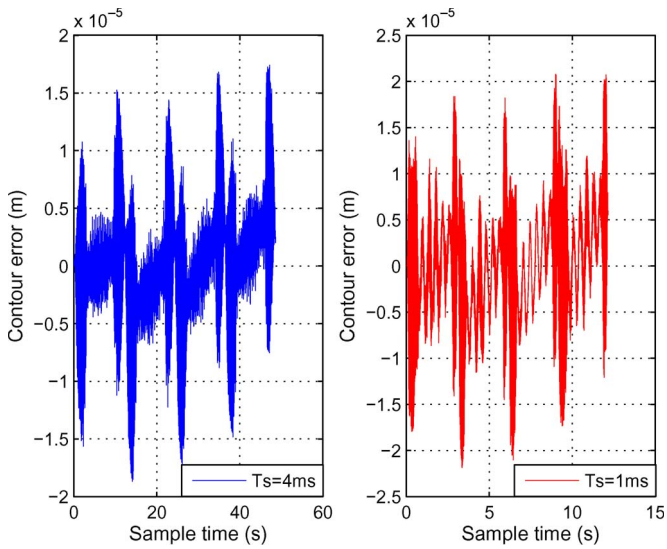

Fig. 10. Contour error with different T_s's.

built, and experiments have been conducted to demonstrate the capabilities of the developed distributed control system for real-time communication and synchronous tracking control, which are required for multiaxis applications. The results verify the feasible application of the IEEE-1394 bus to distributed motion control. Several distinct features of this paper are summarized as follows.

1) The modular hardware and software architectures of a control network have been developed based on the IEEE-1394 bus. The design and implementation of the distributed control system have been described in detail.
2) A scheduled communication protocol has been proposed according to the criteria in terms of bounded time delay and guaranteed transmission. Time delays arising from data processing and message transmission have been analyzed theoretically and verified by experiments.
3) A controller structure allowing both timer and event interrupts has been developed to meet the requirements of real-time communication and synchronous servo control.

Our future work will focus on the development of DMA bus mode to access the IEEE-1394 chip, the improvement of control performance, and the development of real-time IEEE-1394 interface applied in PCs.

ACKNOWLEDGMENT

The authors would like to thank X.-J. Sheng and Dr. J.-H. Wu for the valuable discussions with them, and the anonymous reviewers for their insightful comments and valuable suggestions.

REFERENCES

[1] J. U. Cho, Q. N. Le, and J. W. Jeon, "An FPGA-based multiple-axis motion control chip," *IEEE Trans. Ind. Electron.*, vol. 56, no. 3, pp. 856–870, Mar. 2009.

[2] G. Yasuda, "Distributed autonomous control of modular robot systems using parallel programming," *J. Mater. Process. Technol.*, vol. 141, no. 3, pp. 357–364, Nov. 2003.

[3] T. J. Li and Y. Fujimoto, "Control system with high-speed and real-time communication links," *IEEE Trans. Ind. Electron.*, vol. 55, no. 4, pp. 1548–1557, Apr. 2008.

[4] A. Flammini, D. Marioli, E. Sisinni, and A. Taroni, "Design and implementation of a wireless fieldbus for plastic machineries," *IEEE Trans. Ind. Electron.*, vol. 56, no. 3, pp. 747–755, Mar. 2009.

[5] F. Kanehiro, Y. Ishiwata, H. Saito, and K. Akachi, "Distributed control system of humanoid robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2006, pp. 2471–2477.

[6] Y. Tipsuwan and M. Y. Chow, "Control methodologies in networked control systems," *Control Eng. Pract.*, vol. 11, no. 10, pp. 1099–1111, Oct. 2003.

[7] J. Lian, F. L. Moyne, and D. Tilbury, "Network design consideration for distributed control systems," *IEEE Trans. Control Syst. Technol.*, vol. 10, no. 2, pp. 297–307, Mar. 2002.

[8] S. Y. Lin, C. Y. Ho, and Y. Y. Tzou, "Distributed motion control using real-time network communication techniques," in *Proc. 3rd Int. Power Electron. Motion Control Conf.*, Aug. 2000, vol. 2, pp. 843–847.

[9] W. Prodanov, M. Valle, and R. Buzas, "A controller area network bus transceiver behavioral model for network design and simulation," *IEEE Trans. Ind. Electron.*, vol. 56, no. 9, pp. 3762–3771, Sep. 2009.

[10] J. Kjellsson, A. E. Vallestad, R. Steigmann, and D. Dzung, "Integration of a wireless I/O interface for PROFIBUS and PROFINET for factory automation," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 4279–4287, Oct. 2009.

[11] B. Li, T. Y. Wang, R. J. Y. Xu, and D. Jing-Chuan, "Research and design of embedded reconfigurable mechatronics system based on field bus," in *Proc. 2nd IEEE/ASME Int. Conf. Mechatronic Embedded Syst. Appl.*, Aug. 2006, pp. 1–5.

[12] A. Flammini, P. Ferrari, E. Sisinni, D. Marioli, and A. Taroni, "Sensor interfaces: From field-bus to ethernet and Internet," *Sens. Actuators A, Phys.*, vol. 101, no. 1, pp. 194–202, Sep. 2002.

[13] D. P. Song, F. Divoux, and T. Lepage, "Design of the distributed architecture of a machine-tool using FIP fieldbus," in *Proc. Int. Conf. Appl.-Specific Syst., Architectures Processors*, 1996, pp. 250–260.

[14] W. Hu, G. P. Liu, and D. Rees, "Networked predictive control system with data compression," in *Proc. IEEE Int. Conf. Netw., Sensing Control*, Apr. 2007, pp. 52–57.

[15] H. Ishii and T. Basar, "Remote control of LTI systems over networks with state quantization," *Syst. Control Lett.*, vol. 54, no. 1, pp. 15–31, Jan. 2005.

[16] N. Elia and S. K. Mitter, "Stabilization of linear systems with limited information," *IEEE Trans. Autom. Control*, vol. 46, no. 9, pp. 1384–1400, Sep. 2001.

[17] *IEEE Standard for a High Performance Serial Bus*, IEEE Std. 1394-1995, 1996.

[18] *IEEE Standard for a High Performance Serial Bus—Amendment 1*, IEEE Std. 1394a-2002, 2002.

[19] *IEEE Standard for a High Performance Serial Bus—Amendment 3*, IEEE Std. 1394a-2006, 2006.

[20] T. Shea, J. Mead, P. Cerniglia, and C. Degen, "Evaluation of IEEE 1394 serial bus for distributed data acquisition," in *Proc. Particle Accelerator Conf.*, May 12–16, 1997, vol. 2, pp. 2502–2504.

[21] T. Yago, H. Takano, and N. Tokura, "Loop prevention for IEEE1394-1995 and IEEE1394a-2000 networks," *IEEE Trans. Consum. Electron.*, vol. 48, no. 1, pp. 135–142, Feb. 2002.

[22] J. Kim, S. Lee, Y. Jeon, and S. Choi, "Residential HDTV distribution system using UWB and IEEE 1394," *IEEE Trans. Consumer Electron.*, vol. 52, no. 1, pp. 116–122, Feb. 2006.

[23] J. Y. Oh, J. H. Park, G. H. Jung, and S. J. Kang, "CORBA based core middleware architecture supporting seamless interoperability between standard home network middlewares," *IEEE Trans. Consum. Electron.*, vol. 49, no. 3, pp. 581–586, Aug. 2003.

[24] X. Shao, D. Sun, and J. K. Mills, "A new motion control hardware architecture with FPGA-based IC design for robotic manipulators," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2006, pp. 3520–3525.

[25] W. Shireen, M. S. Arefeen, and D. Figoli, "Controlling multiple motors utilizing a single DSP controller," *IEEE Trans. Ind. Electron.*, vol. 18, no. 1, pp. 124–130, Jan. 2003.

[26] T. N. Chang, B. Cheng, and P. Sriwilaijaroen, "Motion control firmware for high-speed robotic systems," *IEEE Trans. Ind. Electron.*, vol. 53, no. 5, pp. 1713–1722, Oct. 2006.

[27] R. Dubey, P. Agarwal, and M. K. Vasantha, "Programmable logic devices for motion control—A review," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 559–566, Feb. 2007.

[28] E. Monmasson and M. N. Cirstea, "FPGA design methodology for industrial control systems—A review," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1824–1842, Aug. 2007.

[29] Y. F. Chan, M. Moallem, and M. Wang, "Design and implementation of modular FPGA-based PID controllers," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1824–1842, Aug. 2007.

[30] A. M. Lienhardt, G. Gateau, and T. A. Meynard, "Digital sliding-mode observer implementation using FPGA," *IEEE Trans. Ind. Electron.*, vol. 54, no. 4, pp. 1865–1875, Aug. 2007.

[31] B. Selic, "Turning clockwise: Using UML in the real-time domain," *Commun. ACM*, vol. 42, no. 10, pp. 46–54, Oct. 1999.

[32] H. He, Y. F. Zhong, and C. L. Cai, "Unified modeling of complex real-time control systems," in *Proc. Conf. Des., Autom. Test Eur.*, 2005, vol. 1, pp. 498–499.

[33] D. G. Shi, B. R. Wang, and H. E. Wu, "A novel design of distributed servo system based on rearranging functions between host controller and servo driver," in *Proc. IEEE Int. Conf. Autom. Logistics*, Aug. 2007, pp. 480–484.

[34] N. A. Erol, Y. Altintas, and M. R. Ito, "Open system architecture modular tool kit for motion and machining process control," *IEEE/ASME Trans. Mechatronics*, vol. 5, no. 3, pp. 281–291, Sep. 2000.

[35] E. J. Bueno, A. Hernandez, F. J. Rodriguez, C. Giron, R. Mateos, and S. Cobreces, "A DSP-and FPGA-based industrial control with high-speed communication interfaces for grid converters applied to distributed power generation systems," *IEEE Trans. Ind. Electron.*, vol. 56, no. 3, pp. 654–669, Mar. 2009.

[36] H. C. Huang and C. C. Tsai, "FPGA implementation of an embedded robust adaptive controller for autonomous omnidirectional mobile platform," *IEEE Trans. Ind. Electron.*, vol. 56, no. 5, pp. 1604–1616, May 2009.

[37] L. Vachhani, K. Sridharan, and P. K. Meher, "Efficient FPGA realization of CORDIC with application to robotic exploration," *IEEE Trans. Ind. Electron.*, vol. 56, no. 12, pp. 4915–4929, Dec. 2009.

[38] *TSB43AA82 (iSphynx II) Data Manual*, T.I. Inc., Dallas, TX, Jul. 2001. [Online]. Available: http://www.ti.com

[39] *TMS320F2810, TMS320F2812 Digital Signal Processors Data Manual*, T.I. Inc., Dallas, TX, Jul. 2003. [Online]. Available: http://www.ti.com

[40] *Cyclone II Device Handbook*, A. Inc., San Jose, CA, Feb. 2007. [Online]. Available: http://www.altera.com

[41] *Single Supply Dual 16-Bit Audio DAC AD1866* Datasheet Rev. 0*, A.D. Inc., Norwood, MA. [Online]. Available: http://www.analog.com

[42] *TL08x JFET-Input Operational Amplifiers Datasheet*, T.I. Inc., Dallas, TX, Nov. 1992. [Online]. Available: http://www.ti.com

[43] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.

[44] L. Almeida, P. Pedreiras, and J. G. Fonseca, "The FTT-CAN protocol: Why and how," *IEEE Trans. Ind. Electron.*, vol. 49, no. 6, pp. 1189–1201, Dec. 2002.

[45] L. Dozio and P. Mantegazza, "Real time distributed control systems using RTAI," in *Proc. 16th IEEE Int. Symp. Object-Oriented Real-Time Distrib. Comput.*, May 2003, pp. 11–18.

[46] S. Cong and Y. Liang, "PID-like neural network nonlinear adaptive control for uncertain multivariable motion control systems," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 3872–3879, Oct. 2009.

[47] A. Albert, "Comparison of event-triggered and time-triggered concepts with regard to distributed control systems," in *Proc. Embedded World*, 2004, pp. 235–252.

[48] M. O. F. Sarker, C. H. Kim, S. Baek, and B. J. You, "An IEEE-1394 based real-time robot control system for efficient controlling of humanoids," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 9–15, 2006, pp. 1416–1421.

[49] W. Guo, Z. Y. Chen, and C. X. Li, "Investigation on full distribution CNC system based on SERCOS bus," *J. Syst. Eng. Electron.*, vol. 19, no. 1, pp. 52–57, Feb. 2008.

GuoYing Gu received the B.E. degree (with honors) in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2006, where he is currently working toward the Ph.D. degree in mechanical engineering in the State Key Laboratory of Mechanical System and Vibration, School of Mechanical Engineering.

His research interests include mechatronics and motion control.

LiMin Zhu received the B.E. (with honors) and Ph.D. degrees in mechanical engineering from Southeast University, Nanjing, China, in 1994 and 1999, respectively.

From November 1999 to January 2002, he was a Postdoctoral Research Fellow with Huazhong University of Science and Technology, Wuhan, China. In March 2002, he was appointed as Associate Professor with the Robotics Institute, Shanghai Jiao Tong University, Shanghai, China, where he has been a Professor since August 2005 with the School of Mechanical Engineering and is currently with the State Key Laboratory of Mechanical System and Vibration. He was a Visiting Scholar with Monash University, Clayton, Australia (from September 1997 to May 1998), and the City University of Hong Kong, Kowloon, Hong Kong (from December 2000 to March 2001). His research interests include mechatronics, computer vision, computer-aided design (CAD)/computer-aided manufacturing (CAM), and mechanical signature analysis.

ZhenHua Xiong received the B.E. and M.E. degrees from the Department of Aircraft Design, Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1995 and 1998, respectively, and the Ph.D. degree from the Department of Electrical and Electronic Engineering, The Hong Kong University of Science and Technology, Kowloon, Hong Kong, in 2002.

He is currently an Associate Professor with the School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China, where he is also with the State Key Laboratory of Mechanical System and Vibration. His research interests include motion control and electronics manufacturing.

Han Ding (SM'01) received the Ph.D. degree from Huazhong University of Science and Technology (HUST), Wuhan, China, in 1989.

Supported by the Alexander von Humboldt Foundation, he was with the University of Stuttgart, Stuttgart, Germany, from 1993 to 1994. From 1994 to 1996, he was with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, Singapore. From 1997 to 2001, he was a Professor with HUST. He joined Shanghai Jiao Tong University, Shanghai, China, in September 2001, where he is currently a "Cheung Kong" Chair Professor (special appointment of the Yangtze River Scholar Award Plan) with the School of Mechanical Engineering and the Director of the Robotics Institute. His research interests include robotics, manufacturing automation, smart sensors, and intelligent maintenance.

Dr. Ding was the recipient of the National Distinguished Youth Scientific Fund of China (formerly Premier Fund) in 1997.