EtherCAT Based Robot Modular Joint Controller

Conference Paper · August 2015

DOI: 10.1109/ICInfA.2015.7279562

Liu Zhaoming
Chinese Academy of Sciences
7 PUBLICATIONS **37** CITATIONS
SEE PROFILE

Nailong Liu
China Academy of Space Technology (CAST)
9 PUBLICATIONS **9** CITATIONS
SEE PROFILE

Tao Zhang
Chinese Academy of Sciences
12 PUBLICATIONS **14** CITATIONS
SEE PROFILE

Long Cui
Shenyang Institute of Automation, Chinese Academy of SciencesChinese Academ…
24 PUBLICATIONS **29** CITATIONS
SEE PROFILE

Proceeding of the 2015 IEEE
International Conference on Information and Automation
Lijiang, China, August 2015

EtherCAT Based Robot Modular Joint Controller

Zhaoming Liu[1 2], Nailong Liu[1 2], Tao Zhang[2], Long Cui[2], Hongyi Li[2]
[1]*State Key Laboratory of Robotics*
Shenyang Institute of Automation (SIA), Chinese Academy of Sciences (CAS)
Shenyang, Liaoning Province, P. R. China
[2]*University of Chinese Academy of Sciences*
Beijing, P. R. China
Email: liuzhaoming@sia.cn

Abstract –**EtherCAT technology is originally developed by Beckhoff. EtherCAT is by and large the fastest industrial Ethernet technology, and it also synchronizes with nanosecond accuracy. EtherCAT sets new standards for real-time performance and topology flexibility. In this paper, the development of EtherCAT based robot modular joint controller is introduced. This paper describes the hardware and software design, such as PDI design, Slave Stack Code, CiA402 Driver Profile, device description file design and so on. At last, experiments are implemented to test the controller.**

Index Terms – EtherCAT, robot, real-time

I. Introduction

With the development of the industry 4.0, the speed of integration of industrialization and informatization has been accelerated. Intelligent manufacturing system has become more and more intelligent and flexible[1]. In order to develop intelligent manufacturing systems, we need more information of sensors and automatic devices.

For a long time, in order to replace factory workers industrial robots have been widely adopted, which have improved automatic level of manufacturing industry and labor productivity. In the environment of intelligent manufacture, robots which are the key point in the integration of industrialization and informatization will play a significant role. However, several questions exist in the current industrial robots. Firstly, the fieldbus communication speed is insufficient to meet the requirements of large data transmission[2]. Secondly, most of robots based on position control are lack of the force sensing information[3]. These problems limit the application of robots in the intelligent manufacturing system.

Based on the analysis of present industrial robots, improving communication speeds and increasing the feedback information are the essential tasks for industrial robots. Hence, it is important to design a robot module joint controller with high speed Field Bus. In contrast to other Field Bus, EtherCAT has obvious advantages of a transfer of 125 bytes over 100Mbit/s Ethernet[4]. Based on precise clock synchronization, the EtherCAT circuit coupler accuracy is 23ns[5]. Hence, EtherCAT is the right Field Bus desired by robot joint controller.

In [6], an EtherCAT slave module is designed. In [7], the EtherCAT network for multi-axis smart driver is developed. And [8] introduces the application of EtherCAT in parallel robot control system. In this study, we propose to develop an EtherCAT based robot modular joint controller with more complete structure. Thus, our EtherCAT based robot modular joint controller consists of DSP core controller, EtherCAT communication unit, motor drive unit, HALL encoder circuit, encoder circuit, torque sensor circuit, RS232 interface and so on. Fig. 1 shows the function module of joint controller. In addition, we also have designed the control program, EtherCAT drive program and control algorithm of brushless DC motor.

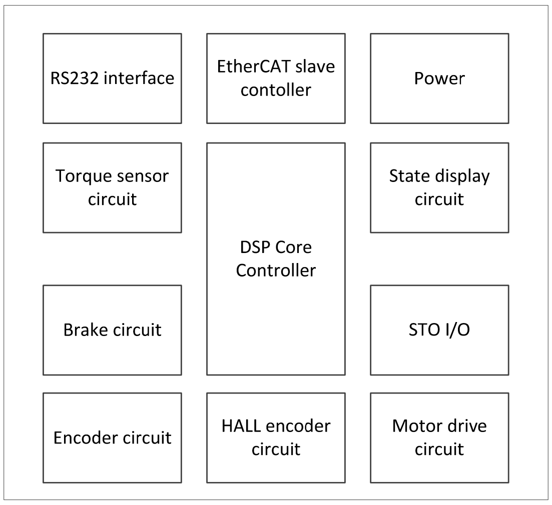


Fig. 1 Function module of joint controller.

This paper consists of 6 sections including this introduction. Section 2 briefly introduces basic concepts of EtherCAT. Section 3 presents the hardware design of the joint controller. Section 4 introduces the design of EtherCAT slave stack code. Section 5 describes CiA402 drive profile and device description file. Section 6 introduces experiments of the joint controller.

II. Overview of EtherCAT

EtherCAT is an industrial communication technology that is specifically standardized in the IEC standard 61158[1]. EtherCAT protocol only has the physical layer, data link layer and application layer in OSI layer model. In EtherCAT, the data frame does not stop and process data is extracted and inserted on the fly. All data of different slave devices are collected in one data frame, and the frame can be edited as it arrives at the corresponding device. Owing to processing by hardware and no CSMA/CD mechanism existing, only a few nanoseconds' delay existences on the entire network and the delay is very stable.

EtherCAT protocol uses a standard IEEE 802.3 Ethernet frame as a reserved type of 0x88A4, which enables the EtherCAT master to transmit frames with standard UDP/IP packet. Based on Ethernet technology, implementation of a master does not require any other special hardware. However, the EtherCAT slave needs specially designed hardware. An EtherCAT frame includes 2 bytes header and 44-1498 bytes data section. Fig. 2 shows the EtherCAT frame structure. The details description of EtherCAT frame can be found in [9].

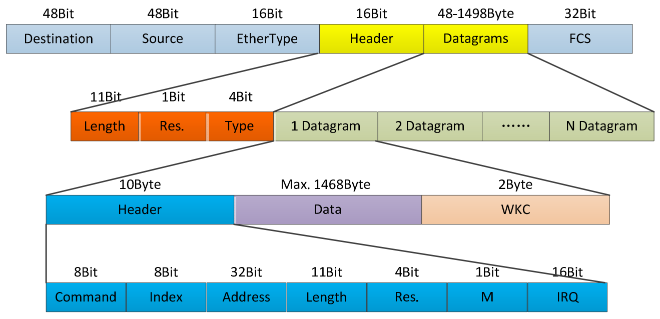


Fig. 2 The EtherCAT frame structure

EtherCAT Slave Controller(ESC) is an ASIC product of Beckhoff, and ET1100 is one of the ASIC products[10]. It manages the EtherCAT communication with an interface between the EtherCAT Field Bus and slave application. ET1100 supports a wide range of applications. For example, it may be used as a 32 bits Digital I/O node without external logic using Distributed Clocks, or as a part of the complex local controller designed with up to 4 EtherCAT communication ports. The ESC works on the data link layer of EtherCAT system. It takes care of the communication infrastructure, which includes link control, access to transceivers, addressing, SyncManager configuration and management, process data interface configuration and so on.

For local host controller, DSP, of the joint controller, ESC is used as an external memory that can generate interrupt. And the Process Data Interface(PDI) realizes the connection between local controller and ESC. Through reading or writing corresponding memory regions on the ESC, the local controller can implement a data transmission. Fig. 3 shows the general functionality of ESC.

An ET1100 has an address space of 64K bytes. The first block of 4K bytes(0x0000-0x0FFF) is dedicated for registers. The Process Data RAM starts at address 0x1000. The memory space is read and written by EtherCAT frame or PDI. Table I is the overview of address space of ET1100.

ESC must be initialized at startup, but configuration information cannot be stored inside of it. Hence, ESC requires an external EEPROM memory chip to store the configuration data. The EEPROM will first be read to acquire internal information and ESC would not start working until it completed configuration.

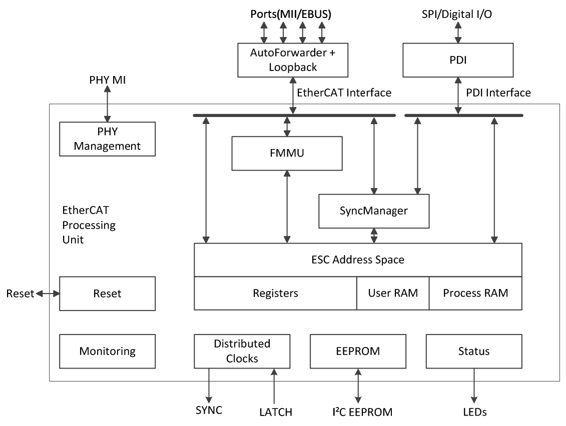


Fig. 3 EtherCAT Slave Controller(ESC) block diagram

TABLE I
ESC ADDRESS SPACE

Address	Description	Address	Description
0x0000-0x0009	ESC Information	0x0600-0x06FC	FMMU Configuration
0x0010-0x0013	Station Address	0x0800-0x087F	SyncManager
0x0020-0x0031	Write Protection	0x0900-0x090F	DC-Receive Times
0x0040	ESC Rest	0x0910-0x0935	DC-Time Loop Control Unit
0x0100-0x0111	Data Link Layer	0x0980	DC-Cyclic Unit Control
0x0120-0x0135	Application Layer	0x0981-0x09A7	DC-SYNC Out Control
0x0140-0x0153	PDI	0x09A8-0x09CF	DC-Latch In Unit
0x0200-0x0223	Interrupt	0x09F0-0x09FF	DC-SM Event Times
0x0300-0x0313	Error Counters	0x0E00-0x0EFF	ESC Specific
0x0400-0x0443	Watchdogs	0x0F00-0x0F1F	Digital I/O Input Data
0x0500-0x050F	EEPROM Interface	0x0F80-0x0FFF	User RAM/Extended ESC Features
0x0510-0x051B	MII Management Interface	0x1000-0xFFFF	Process Data RAM

III. HARDWARE DESIGN OF THE JOINT CONTROLLER

In this section, the design of Process Data Interface(PDI) between ESC and DSP is mainly introduced. According to the architecture of EtherCAT system, hardware design of the joint controller can be divided into three parts: physical layer, data link layer and application layer. In physical layer, standard RJ45 connectors, pulse transformers and standard PHYs are indispensable. ET1100 is a chip for data link layer where the chip handles EtherCAT frame all by hardware. The application layer controller is TMS320F28335 type of DSP produced by TI company. The core part of joint controller is this DSP, which controls brushless DC motor and reads the force feedback signal. The MII interface is between the physical layer chip and ESC. In addition, the ESC processes EtherCAT frames and provides data for local host controller via SPI. Fig. 4 shows the overall structure of joint controller hardware.

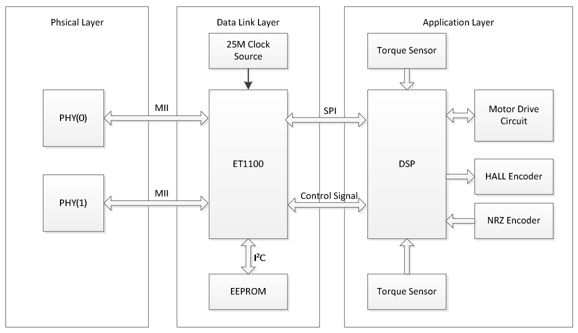


Fig.4 Overall structure of joint controller hardware

ESC supports several types of PDI, which can be configured as SPI, parallel interfaces or digital I/O interfaces by the registers of 0x0140~0x0141. For high quality motor control, the SPI mode is utilized for process data interface. Fig. 5 shows the connection diagram of SPI master and slave. In this mode, SPI master and slave are DSP and ESC respectively. An SPI access starts while the master asserts SPI_SEL and terminates by taking back SPI_SEL. Each SPI access consists of an address phase and a data phase. In the address phase, the master transmits the address to be accessed and the command. In the data phase, output data is presented by the slave and input data is transmitted by the master. During accessing to SPI slave, the master has to cycle SPI_CLK eight times for each byte transfer.

The PDI operation is related to the synchronization mode of EtherCAT system. In the periodic communication mode of EtherCAT, there are three modes of synchronization, which are FreeRun, SM-Synchron and DC-Synchron respectively. In FreeRun mode, the slave does not generate interrupt signal on SPI_IRQ. And the local host controller generates interrupt in itself to access ESC periodically. In SM-Synchron mode, the event of local controller is synchronized as data exporting or entering. The SPI slave will generate interrupt signal on SPI_IRQ to trigger the DSP to operate process data when ESC receives the frame from the EtherCAT master. In DC-Synchron mode, the event of local controller is synchronized by SYNC event that is controlled by event of distributed clocks system of EtherCAT. Based on distributed clocks, all of the ESCs will generate SYNC event simultaneously. However, the process data must be exchanged before the operation of local controller in this mode.

Finally, it is necessary to emphasize EEPROM_LOADED signal. This signal can be set only if the data of EEPROM have been loaded into ESC correctly. The signal provides conveniences to detect the state of ESC.

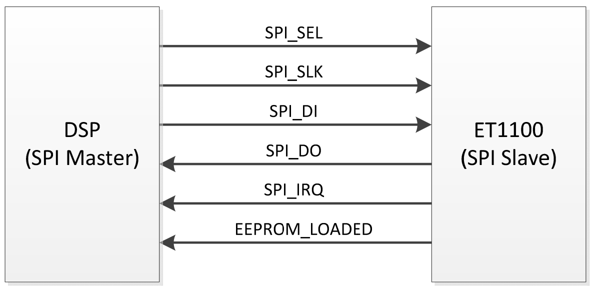


Fig.5 Connection diagram of SPI master and slave

IV. Design Of Slave Stack Code And Application Program

Fig.6 shows the structure of EtherCAT slave stack which is consisted of hardware access, generic EtherCAT stack and user application. The hardware access layer provides the PDI and hardware abstraction functions and macros for ESC access. The generic EtherCAT stack provides the functions that handle the process data, maxilbox data and EtherCAT state machine. And the CoE protocol is applied in the joint controller.

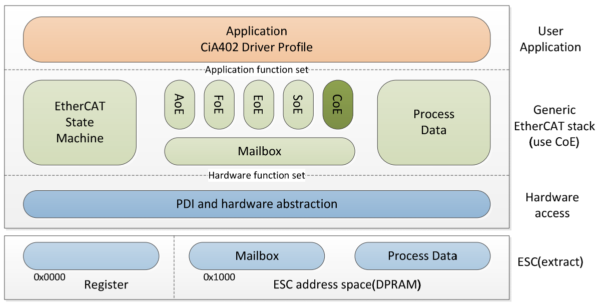
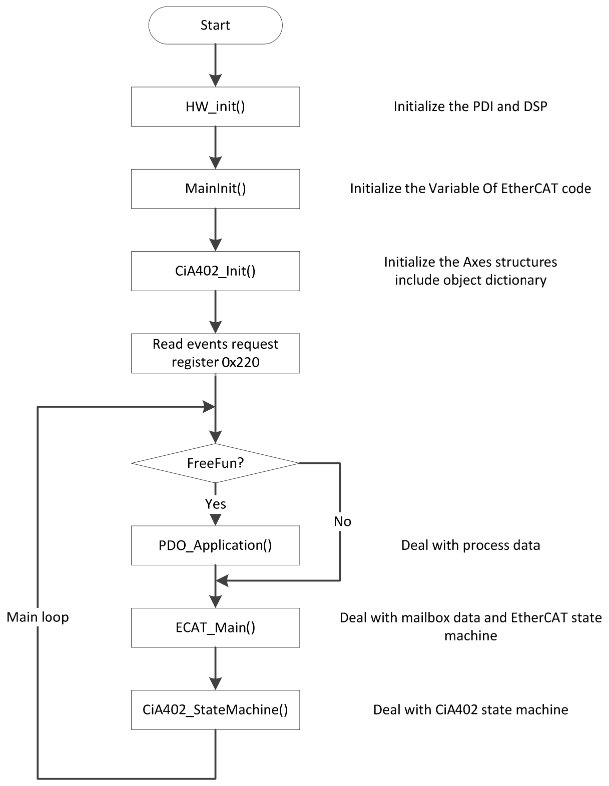


Fig. 6 EtherCAT slave stack

Fig. 7 shows the flow chart of application layer program. Before entering into the main loop, a series of initialization have been executed. Firstly, program initializes the hardware such as PDI and DSP interrupt. Secondly, variables of EtherCAT protocol are initialized. Finally, CoE objects dictionary initialization is executed. Then the program enters into the Mainloop to handle process data and conversion of state machine.


Start
HW_init()
Initialize the PDI and DSP
MainInit()
Initialize the Variable Of EtherCAT code
CiA402_Init()
Initialize the Axes structures include object dictionary
Read events request register 0x220
FreeFun?
Yes
No
PDO_Application()
Deal with process data
Main loop
ECAT_Main()
Deal with mailbox data and EtherCAT state machine
CiA402_StateMachine()
Deal with CiA402 state machine
Fig. 7 Flow chart of application program

V. CiA402 Driver Profile And Device Description File In XML Format

The CANopen over EtherCAT(CoE) protocol is utilized to configure motion controller and exchange data objects on application level. CiA402 drive profile is specified by IEC for motion control industry. And it makes a clear specification for the structure function of servo drive system. CiA402 drive profile is mapped to EtherCAT. This implementation provides interface between the motion controller application and communication layer. In this paper, EtherCAT application layer is designed according to CiA402 drive profile. Fig.8 shows the CiA402 servo controller based on EtherCAT.

Object dictionary is the critical point of CoE protocol, and it is the description of whole servo controller. Each of the objects consists of the 16 bits index and the data. Objects with the index of 0x6000~0x67FE are specified for CiA402 drive profile.

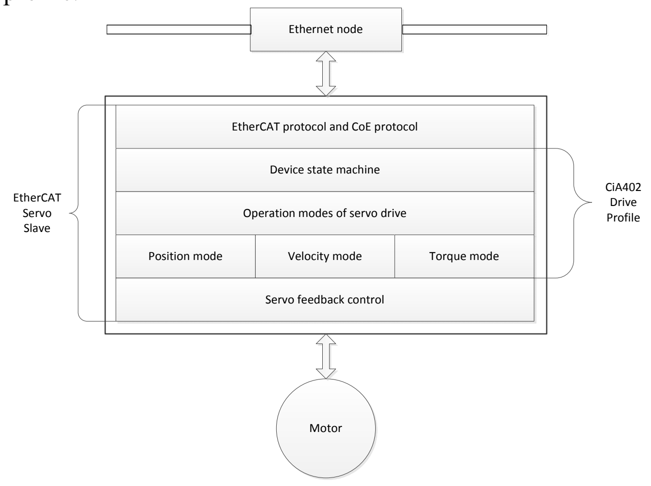


Fig.8 CiA402 servo drive based on EtherCAT

Drive state machine plays an important role in CiA402 drive profile. The state machine describes the device status and the possible control sequence of the drive. The motor cannot run, until the state machine has been switched correctly. Fig. 9 shows the device state machines and their relationship. State transactions are requested by setting the object with index 0x6040 called ControlWord. NOT_READY_TO_SWITCH_ON is an internal state in which communication is enabled only at the end, and the user cannot monitor this state. SWITCH_ON_DISABLED is the minimum state to which a user can switch. In this state, the drive initialization is completed. The drive can perform transitions 0 and 1 after initiation automatically. The high voltage may be applied to the drive in the state of READY_TO_SWITCH_ON, however the drive operation is still disabled. SWITCHED_ON is the state that the power amplifier is ready but the drive operation is disabled all the same. The drive function can be enabled as the state is switched to OPERATION_ENABLE. No matter what state of the drive is, the state can be transmitted to FAULT in case of a fault occurring.

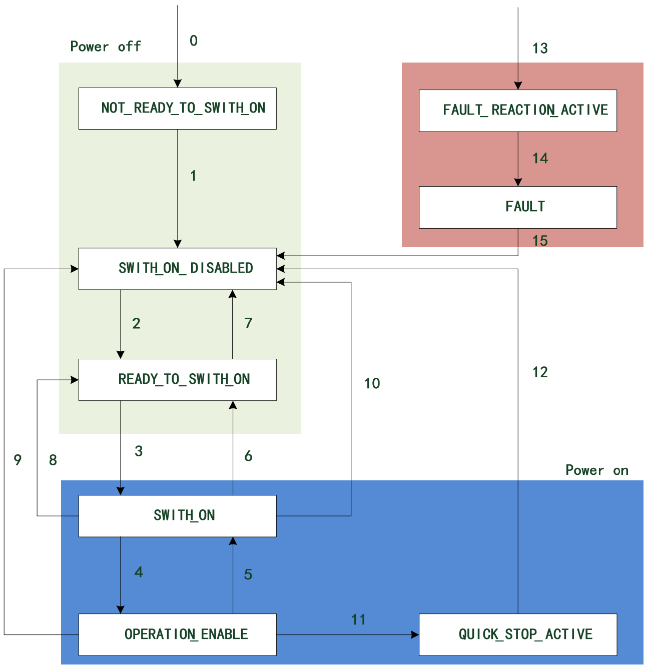


Fig. 9 State machine of servo drive

Through the device description file of slave, the EtherCAT master can acquire the information of slave and configure it. The device description file of EtherCAT system is generated by XML Schema language[11]. Fig. 10 shows the general structure of a device description file.

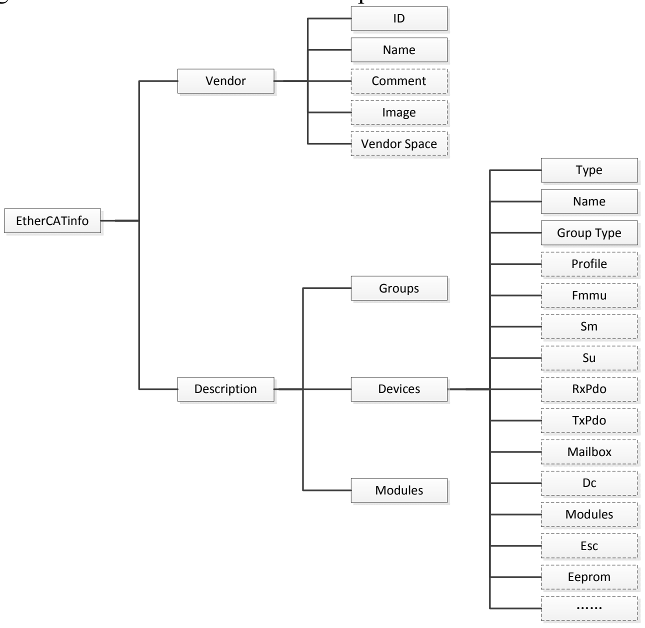


Fig. 10 General structure of device description file

The EtherCATInfo element is the root element of EtherCAT slave device description file[12]. The element of Vendor describes the identity of the device vendor including its name and EtherCAT Vendor ID assigned by EtherCAT Technology Group(ETG). The Description element describes EtherCAT device via the elements Groups, Devices and Modules. The Groups and Devices are main elements of slave device description. Through the element Groups, similar devices can be assigned to one group, and the element Devices describes the device with its EtherCAT features such as SyncManagers, FMMUs, and Object Dictionary. However the element Modules is optional, according to the Modules Device Profile, it is used when the slave device is structured.

The design of this paper according to the object dictionary of CiA402, and the object dictionary is described in the element of Profile. As shown in Fig. 11, the ChannelInfo describes information of protocol, and the Dictionary which contains two elements of DataTypes and Objects describes content of object dictionary. The DataTypes are the data type of corresponding objects. Moreover, the description file also includes the information of EEPROM, which is in the element of Eeprom.

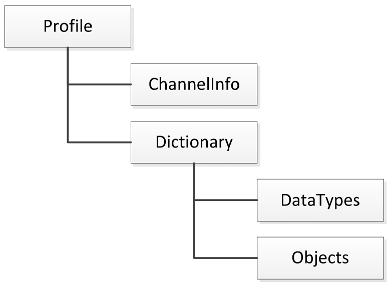


Fig. 11 The element of Profile

VI. Experiment And Discussion

The test system contains EtherCAT master and slave. The master includes a PC, standard Ethernet interfaces card and TwinCAT software system. Two experiments for CiA402 state transform and real-time task period have been implemented successfully in order to verify performance of the joint controller.

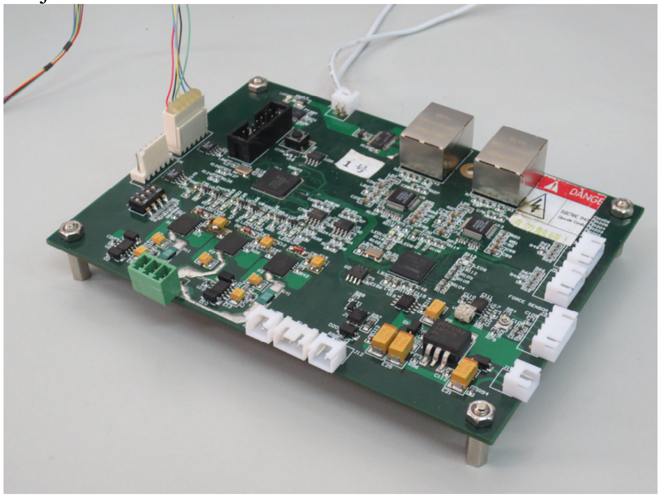
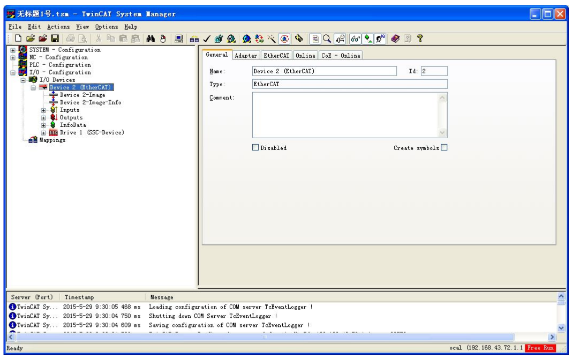


Fig. 12 EtherCAT based robot modular joint controller

In the first experiment, CiA402 state machine transform has been executed. Firstly, TwinCAT System Manager in master PC was started. Secondly, right mouse click on "I/O Devices" and select "Scan Devices", after successful device scan Fig. 13 was shown on the screen.



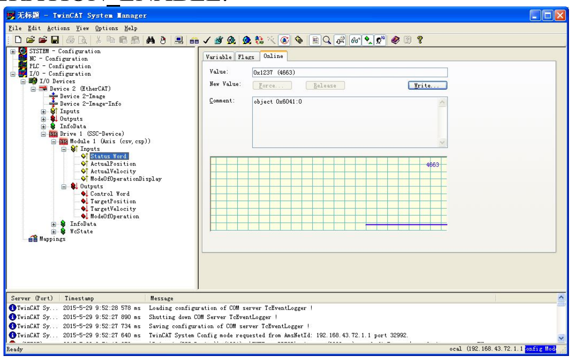
```
Name:     Device 2 (EtherCAT)        Id: 2
Type:     EtherCAT
Comment:

Server (Port)   Timestamp                      Message
TwinCAT Sy...   2015-5-29 9:30:05 468 ms       Loading configuration of COM server TcEventLogger !
TwinCAT Sy...   2015-5-29 9:30:04 750 ms       Shutting down COM Server TcEventLogger !
TwinCAT Sy...   2015-5-29 9:30:04 609 ms       Saving configuration of COM server TcEventLogger !

ocal (192.168.43.72.1.1
```

Fig. 13 Successful device scan

Click on the "FreeRun" button, then the system enters Operation Mode. Assign a series of values to ControlWord(0x6040), which are 0x06, 0x07 and 0x0F. Then the feedback of StatusWord(0x6041) is 0x4663, which signifies the state of OPERATION_ENABLE. Fig. 14 shows that the drive device has been switched to the state of OPERATION_ENABLE.



```
Value:      0x1237 (4663)
New Value:
Comment:    object 0x6041:0

Server (Port)   Timestamp                      Message
TwinCAT Sy...   2015-5-29 9:52:28 578 ms       Loading configuration of COM server TcEventLogger !
TwinCAT Sy...   2015-5-29 9:52:27 890 ms       Shutting down COM Server TcEventLogger !
TwinCAT Sy...   2015-5-29 9:52:27 734 ms       Saving configuration of COM server TcEventLogger !
TwinCAT Sy...   2015-5-29 9:52:27 640 ms       TwinCAT System Config mode requested from AmsNetId: 192.168.43.72.1.1 port 32992.

ocal (192.168.43.72.1.1
```

Fig. 14 Drive device in OPERATION_ENABLE

In the second experiment, a timing analyzer NANL-B500E-RE, the product of Hilscher, has been used to analyze the performance of EtherCAT system. The experiment has been carried out with a real-time task period of 1000μs. Fig. 15 shows the timing analysis of the experiment. Almost every sample concentrates on the point of 1000μs. Obviously, the round trip time delay of EtherCAT system is very stable. These results show that the performance of EtherCAT system is remarkable.

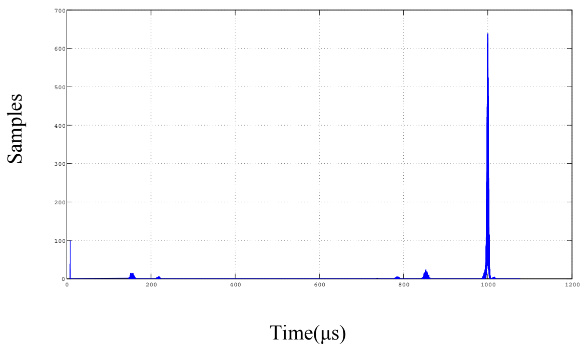


Fig. 15 Timing analysis of EtherCAT system

VII. Conclusions

This paper presents the EtherCAT based robot modular joint controller. With the excellent performance of the real-time EtherCAT technology, it will be applied by developers in industry robots widely.

ACKNOWLEDGEMENT

This work is supported by ITER Program of MOST, Grant 2012GB102005 and supported by the National Science Foundation of China under contact: 61172145.

REFERENCES

[1] A. Kluth, J. Jäger, A. Schatz, and T. Bauernhansl, "Method for a Systematic Evaluation of Advanced Complexity Management Maturity," Procedia CIRP, vol. 19, pp. 69-74, 2014.

[2] H. Xing, H. Jia, and L. Yanqianga, "Motion Control System using SERCOS over EtherCAT," Procedia Engineering, vol. 24, pp. 749-753, 2011.

[3] L. Minzhou, F. Jian, Z Jianghai, "The Development and the Application of the Industrial Robot Technology," Machine Buiiding & Automation, vol.1, pp. 1-4, 2015.

[4] M. Cereia, I. C. Bertolotti, and S. Scanzio, "Performance of a real-time EtherCAT master under Linux," Industrial Informatics, IEEE Transactions on, vol. 7, pp. 679-687, 2011.

[5] G. Cena, I. C. Bertolotti, S. Scanzio, A. Valenzano, and C. Zunino, "Evaluation of EtherCAT distributed clock performance," Industrial Informatics, IEEE Transactions on, vol. 8, pp. 20-29, 2012.

[6] C. Kang, Y. Pang, C. Ma, and C. Li, "Design of EtherCAT slave module," in 2011 IEEE International Conference on Mechatronics and Automation, ICMA 2011, August 7, 2011 - August 10, 2011, Beijing, China, 2011, pp. 1600-1604.

[7] J. H. Park, S. Lee, K. C. Lee, and Y. J. Lee, "Implementation of IEC61800 based EtherCAT slave module for real-time multi-axis smart driver system," in International Conference on Control, Automation and Systems, ICCAS 2010, October 27, 2010 - October 30, 2010, Gyeonggi-do, Korea, Republic of, 2010, pp. 682-685.

[8] J.-H. Kim, S. Lim, and I.-K. Jung, "EtherCAT based parallel robot control system," in 1st International Conference on Robot Intelligence Technology and Applications, RiTA 2012, December 16, 2012 - December 18, 2012, Gwangju, Korea, Republic of, 2013, pp. 375-382.

[9] "EtherCAT introduction," ETG, 2012[Online]. Available: http://www.ethercat.org/

[10]"Hardware data sheet ET1100-EtherCAT slave controller, Ver. 1.8," Beckhoff Automation GmbH, 2010[Online]. Available: http://www.beckhoff.com/

[11]W3C. "XML Schema Part 2: Datatypes." Internet: http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/datatypes.xml

[12]"EtherCAT Slave Information-Specification, Ver. 1.0.7", ETG, 2012[Online]. Available: http://www.beckhoff.com/