A High Speed Redundant IO Bus for Energy Power Controller System

Qinghui Lou1*, Liguo Sun1, Haisong Lu1, Weifeng Xu1, Zhebei Wang1, Dan Cai1, Xiangjian Shi1

1 NR Electric Co., Ltd, Nanjing, 211102, China
* Corresponding author’s e-mail: louqinghui@nrec.com

ABSTRACT. This paper designs and implements a High Speed Redundant IO Bus for Energy Power Controller System. The physical layer adopts multi-point low-voltage differential signal standard. This bus has the characteristics of high real-time, high throughput and easy expansion. The controller communicates with IO module by A/B bus alternately, monitors link status in real time and collects IO module data. Non real time slots can be used to control non real time messages for IO modules such as time synchronizing and memory monitoring. The controller ARM core runs QNX real-time operating system, and transmits the message needed to communicate with IO modules to the FPGA through DMA. After receiving the message, the FPGA parses the message and automatically fills in the CRC check code and frame end flag at the end of the message. When the FPGA receives the data feedback from the IO module, it performs CRC verification. If the verification passes, it fills the corresponding module receiving buffer. Otherwise, it fills the CRC verification error flag in the register of the corresponding IO module to reduce the load of the arm core.

1. Introduction
Control system, including distributed control system (DCS), programmable logic controller (PLC), etc., is a combination of computer technology and automation control technology, in power, petrochemical, metallurgical, chemical, etc. It has a wide range of applications in large industries [1]. Compared with the booming fieldbus (such as Profibus, FF, CAN, LonWorks, etc. [2] [3]), the development of the local bus of the control system appears to be relatively slow. The reason is mainly that the control system and the local input/output (IO) module communication are relatively closed, involving the architecture of the control system and all the IO modules, so each manufacturer keeps the protocol confidential, making other manufacturers unable to get their communication information. The paper [4] proposed a communication protocol between controller and IO based on general object oriented substation event (GOOSE), which limits its scope of use due to the use of network communication. The paper [5] proposed a communication bus based on PCI Express, but because of the compatibility, reliability and scalability of computer serial buses (such as PCI Express, Universal Serial Bus (USB), IEEE1394), The requirements of the control system application are not well met [6]-[9].

Aiming at the problems of the above bus, this paper proposes a high-speed and reliable redundant IO communication bus based on field programmable gate array (FPGA). The physical layer uses multi-point low-voltage differential signals (M-LVDS [10]) standard, realizes the real-time data transmission of the data link layer and the monitoring of the channel link status, ensuring the reliability and real-time performance of data transmission. The IO module uses direct memory access (DMA) to analyze the
communication packets of the universal asynchronous receiver transmitter (UART). The detailed design is cited from paper [11].

2. Overall System Plan
The current mainstream control system architecture can be generally divided into four levels: the field instrument layer, the control device unit layer, the factory (workshop) layer, and the enterprise management layer [12], where the control unit layer includes the controller and the IO module. The interconnection between the controller and the IO module is the hub of data transmission of the whole system. It has very high requirements in real-time, reliability and scalability. The efficiency of the bus directly affects the performance of the entire control system [1]. The controller designed in this paper adopts the scheme of active/standby redundancy, and the IO unit adopts the A/B redundant IO bus scheme. The active and standby controllers communicate with the IO module through the A/B redundant bus, but only the active controller sends the query data and control messages to the IO module alternately through the A/B redundant bus, and the standby controller passes the A/B redundancy. The bus monitors the response data of the IO module to determine the continuity of the link. After the standby controller is upgraded to the active controller, the original active controller no longer sends query data and control packets, and the newly-based controller sends query data and control packets. The system structure is shown as in Fig. 1, wherein the IO module can be analog input (AI), analog output (Analog Output, AO), digital input (DI), digital output (Digital Output, DO) or some other special IO module. A pair of controllers can carry 6 redundant IO buses, and each IO redundant bus can mount 10 IO modules. Therefore, the controller designed in this paper can mount 60 local IO modules.

![Fig.1 Main structure of the system](image)

3. Protocol Design And Implementation
The real-time and reliability of communication between the controller and the IO module mainly depends on the communication rate of the physical layer and the scheduling strategy of the data link layer. The communication protocol designed in this paper specifies the information exchange mode between the controller and the IO module. This protocol complies with the open system interconnection reference model (OSI/RM) and uses the application layer, data link layer and physical layer. The application layer implements the real-time data or non-real-time data analysis of the IO module, the data link layer implements a cyclic redundancy check (CRC), and the physical layer implements the codec function of the serial port data.
Each IO module designed in this paper has the same communication interface and frame structure. The IO module includes a micro control unit (MCU), which realizes data acquisition and analysis through DMA, and detailed reference [11]. The controller uses Xilinx's Zynq chip, which has two (advanced risc machine, ARM) chips and one FPGA. The ARM chip runs the QNX real-time operating system and extends the 6-way redundant IO bus through the FPGA. Each bus realizes 5Mbps high-speed serial communication, including M-LVDS transmission/reception module, 10bit codec module and CRC check module. The QNX real-time operating system interacts with the FPGA through configuration registers and DMA.

3.1 Physical Layer Protocol
The physical layer specifies the mechanical, electrical, functional, and procedural characteristics of the communication between the controller and the IO module. This communication protocol adopts the multi-point low-voltage differential signaling (M-LVDS) standard to implement a single-master multi-slave communication mode. M-LVDS has the characteristics of high-speed transmission, low voltage swing, anti-interference, low power consumption, multi-point communication, and fault safe isolation [1]. Because M-LVDS has the above characteristics, the IO modules can be online hot-plugged to establish or disconnect real-time communication with the controller, and the communication functions of other IO modules on the IO bus will not be affected. The hot-plugged function of the bus improves the reliability and scalability of the whole system.

3.2 Data Link Layer Protocol
The data link layer is responsible for performing link layer protocol data unit (PDU) interaction between the controller and the IO module on the link. The data link layer defines the protocol data units in details as shown in Fig2.

![Protocol data unit](image)

**Fig.2 Protocol data unit**

3.3 Application Layer Protocol
The application layer protocol mainly includes 3 parts:

1) The controller groups the query or control message sent to the IO module according to the configuration file of the IO module;
2) Parses the real-time data sent by the IO module, and sends the data to fill in the IO receiving buffer;
3) Group or parse non-real-time packets.

Since the QNX real-time operating system transmits the message delivered to the IO module to the FPGA through DMA, it is required to design the issued message format to ensure the correctness of the data received by the FPGA. The data format defined by ARM to FPGA is shown in Fig 3. The “info” uses 6 bits as the subnet mask. Each bit represents each branch of the 6 buses. For example, 0x1 means sending on the first bus, 0x10 means sending on the fifth bus, and 0x3f means that all the 6 branches will send. The “len_byte” represents total length of the data packet in this frame. The “content” is the
data link layer protocol data unit (PDU), but there is no CRC checksum and frame tail flag. These two items will be automatically added by FPGA to reduce the load on the ARM chip. Because sending data by DMA will lead to DMA interrupts, after the FPGA receives the data, it does not transmit the data to the QNX real-time operating system through DMA, but stores the data in the global data area by means of the circular buffer. This design has two advantages: 1) Reducing the number of DMA interrupts, the QNX real-time operating system does not need to generate an interrupt to process data after the FPGA receives each IO module data; 2) IO modules under 6 busses can be processed uniformly, and the data of the IO module can form a cross-section, which is beneficial to the algorithm page processing.

Fig. 3 Data format of ARM to FPGA

| Data format of ARM to FPGA |
|---------------------------|
| Frame head flag           |
| info                      |
| len_byte                  |
| content                   |
| SUM_CHC                   |
| Frame tail flag           |

4. **Software Design**

The software program mainly includes:

1) IO module configuration file analysis;
2) Serial port register configuration;
3) Serial port driver configuration;
4) FPGA register configuration;
5) Real-time message transmission;
6) Non-real-time message transmission.

After the controller is powered on, the parameter configuration will be set first. After the setting is completed, the FPGA generates a 5ms pulse signal at a fixed period. After the QNX real-time operating system captures this pulse signal, it performs packetization and analysis of real-time and non-real-time messages.

4.1 **Real-time Message Transmission**

After the QNX real-time operating system captures the timing interrupt generated by the FPGA, it first analyzes the message that the FPGA transmits to the application layer through the circular buffer, and fills the real-time acquisition data of the IO module into the memory area of the IO module according to the analysis result. Then the controller groups control or query messages sent to the IO module according to the IO module configuration information. Finally, after scanning 60 IO modules, the formed sent message will be transmitted to the FPGA through DMA. After the FPGA receives the message, it parses it in the format listed in Table 2, and fills in the CRC checksum and frame tail flag at the end of the link protocol data unit (PDU). Please refer to Fig. 4 for the specific flow.
4.2 Non-Real Time Message Transmission
Because the FPGA generates 5ms pulse signal at a fixed period to receive and send real-time messages, it is required to consider adding non-real-time messages to these real-time messages, e.g.: time synchronization, IO module memory query, specific IO module upgrade program, etc. These non-real-time messages will not affect transmission of real-time messages. The design scheme of this paper is: If the branch has non-real-time data, a frame of non-real-time message will be added at the end of the real-time message transmission period of this branch. If the branch has no non-real-time data, time slot of this branch will be ignored. With this scheme, conflicts between real-time message transmission and non-real-time message transmission can be effectively resolved.

5. Performance Analysis

Through the bus time slot diagram in Fig. 5, the communication between the controller and the IO module through the A / B redundant bus can be introduced in details.

In this figure: T1 represents the time when the QNX real-time operating system parses the data sent by the IO module, T2 represents the packetization time when the QNX real-time operating system issues the query or control packet according to the IO module configuration file, and T3 represents the time...
that DMA sent data from the ARM chip to the FPGA. After actual measurement, the processing time of T1, T2 and T3 all reaches microsecond level.

When T3 is reached, the FPGA analyzes the delivered data message according to the format in Tab.2, and automatically adds a CRC checksum and a frame tail flag to issue a query or control message. In theory, the communication time can reach a cycle of 5ms, because T1, T2, and T3 do not occupy the bus bandwidth time. If this cycle is a message sent by the A bus, then the next cycle is switched to the B bus to send a message. In this way, switching and sending can not only reduce the communication load of the IO module, but also monitor the status of the A / B bus communication link. However, the disadvantage of this operation is to process the data fed back by the IO module. It needs to wait for 2 query cycles (i.e., 10ms). If the A-bus or B-bus communication link is disconnected, it will cause the entire branch query period to double. However, the controller and the IO module are grouped in a panel cabinet, and the controller will alarm when the communication link is disconnected, so this situation can be avoided well.

6. Conclusion
This paper presents the design and implementation of a high-speed redundant IO bus based on FPGA. The system combines the advantages of the embedded ARM chip and FPGA perfectly, and realizes the development of embedded platform combining FPGA and ARM chip, which improves the real-time performance, reliability and stability of the system. This design can guarantee the real-time concurrent processing of the 6-channel IO module bus, and it does not occupy the processing time of the ARM chip, so it has excellent value and promotion significance.

References
[1] MI Xiao-ling, HUANG Wen-jun, JIN Jian-xing, SHI Yi-ming. Design and implementation of FPGA based high-speed bus for control system [J]. Journal of Zhejiang University, 2011, 45(11):2043-2049.
[2] CHEN Ji-ming, WANG Zhi, SONG Ye-qiong. Study on schedule problem of aperiodic message traffic in foundation fieldbus [J]. Journal of Zhejiang University, 2003, 37(03):273-277.
[3] YANG Xianhui Fieldbus technology and its application [M]. Beijing: Tsinghua University Press, 2008.
[4] ZANG Feng, XU Weifeng, WU Bo, NIU Honghai. Development of Industrial Process Controller Based on GOOSE Protocol [J]. Process Automation Instrumentation, 2016, 37(6):51-54.
[5] JIANG Qunxing, HU Lisheng. Research on High-speed Communication Bus Technology of Host Controller in Nuclear-level Control System [J]. MICROCOMPUTER APPLICATIONS, 2010, 26(4):6-8.
[6] DHAWAN S K. Introduction to PCI Express a new high speed serial data bus[C] //Nuclear Science Symposium Conference Record. Fajardo: IEEE, 2005, (2):687-691.
[7] DONG Xinwei, WU Guozhong, ZHAO Rongxiang. Analysis and design of power electronic building blocks’ communication interface [J]. Journal of Zhejiang University, 2007, 41(11):1857-1861.
[8] DEPARI A, FLAMMINI A, MARIOLI D, et al. USB sensor network for industrial applications[J]. Instrumentation and Measurement, 2008, 57(7):1344-1349.
[9] SCHOLLES M, FROMMHAGEN K, KLEINMANN L. IEEE1394 “FireWire” system design for industrial and factory automation applications[C] //IEEE Proceedings of the 8th IEEE International Conference on Emerging Technologies and Factory Automation. AntibesJuan les Pins: IEEE, 2001, (2): 627-630.
[10] ANSI/TIA/EIA-899. Electrical characteristics of multipoint-low-voltage differential signaling (MLVDS) interface circuits for multipoint data interchange[S]. USA TIA/EIA, 2000.
[11] NIU Honghai, ZANG Feng, ZHOU Xugui. Design and Implementation of High Speed UART Based on DMA [J]. Process Automation Instrumentation, 2018, 39(9):45-48.
[12] WANG Changli. Current Status and Development Trends of Distributed Control System [J]. ELECTRIC AGE, 2004, (01):82-86.