DISTRIBUTED IO SYSTEM BASED ON ETHERNET POWERLINK UNDER THE EPICS ARCHITECTURE

X. Sun, Y. Song, G. Liu*, NSRL, USTC, Hefei, Anhui 230029, China

Abstract

Ethernet POWERLINK (EPL) is a communication profile for Real-Time Ethernet. The communication profile meets real-time demands for the distributed system composed of multiple controllers. EPICS is a wildly used distributed control system in large scientific facilities. We design a distributed IO system based on EPL under the EPICS architecture and establish the prototype system composed of a PC and several FPGA boards. In this system, an EPICS driver based on openPOWERLINK is developed to monitor the system status. In this paper, the communication mechanism of EPL, the design of system architecture, the implementation of EPICS driver and the test results of prototype system will be described.

INTRODUCTION

Ethernet POWERLINK (EPL) is an open source real-time Ethernet built on standard Ethernet absolutely and allows data transfer with predictable timing and precise synchronization.

EPICS is a set of Open Source software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as particle accelerators, telescopes etc [1].

There are real-time demands in millisecond level in accelerator control system, such as interlock system. Therefore we design a distributed IO system based on EPL under the EPICS architecture. In order to monitor the system status, we develop an EPICS driver based on openPOWERLINK(oplk). A prototype system composed of a PC and several FPGA boards is established, and we test the communication cycle time and the response time of the system.

OVERVIEW OF THE EPL PROTOCOL

The EPL protocol uses a technique called Slot Communication Network Management (SCNM) to ensure that there are no collisions during physical network access of any of the networked nodes thus providing deterministic communication. There are two kinds of nodes in the EPL network: the EPL Managing Node(MN) and the Controlled Nodes(CN). There are five different message types, namely SoC, PReq, PRes, SoA, and ASnd, shown in Fig 1. These EPL messages are exchanged in the specific part of the communication [2].

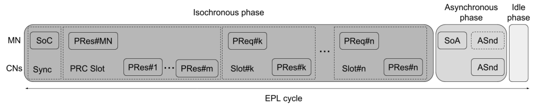


Figure 1: The communication cycle of EPL standard mode.

The basic EPL communication cycle is divided into three phases as shown in Fig. 1. In the Isochronous phase, the MN sends a SoC frame to all CNs via Ethernet multicast at first, then the PReq frame is sent to every configured and active CN by MN. The PReq frame is received only by the addressed CN, and this addressed CN responds to this request with a PRes multicast frame. The Asynchronous phase of the EPL communication cycle is used to exchange acyclic data, it starts with a SoA multicast frame. The Idle phase is the remaining time interval between the end of the Asynchronous phase and the beginning of the next cycle.

SYSTEM ARCHITECTURE

A distributed IO system is designed based on EPL under the EPICS architecture, Fig. 2 is the system hardware architecture. The network topology of this system is a linear topology, each node is connected in sequence. This system consists of seven nodes : one MN and six CNs. The MN is a PC with a RT Linux. The IOC application runs on the MN and monitors the IO status of each CN. The CN is the DE2-115 FPGA development board supporting EPL communication and Hub function. The IO functions are implemented on the DE2-115 FPGA development board. The MN collects all input signals from CNs through EPL PRes frame and sends the output signals to the CN by EPL PReq frame according the algorithm.

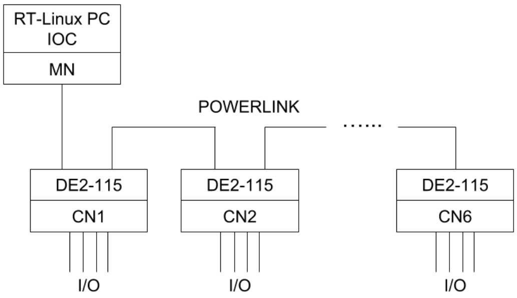


Figure 2: Hardware architecture of the distributed IO system.

EPICS DRIVER

Oplk is an open source industrial Ethernet stack implementing the EPL protocol, and it is supported on several processor platforms including Intel X86 and Altera NiosII, and operating systems including Linux and Windows. Under Linux, oplk has two kinds of operating modes: kernel module and user space. An EPICS driver is developed based on oplk, and it only runs in user space now [3].

* gfliu@ustc.edu.cn

Figure 3 is the software structure of the EPICS driver. The driver consists of the device support and the driver support. The driver support realizes the functions of MN by calling the functions implemented in oplk. The driver support periodically puts the received data into the buffer "Recv Buf" and sends data in the buffer "Send Buf" to the CNs. The device support conveys the data in buffers to records. Only the standard record types AI, AO, DI and DO are supported by the device support now [3].

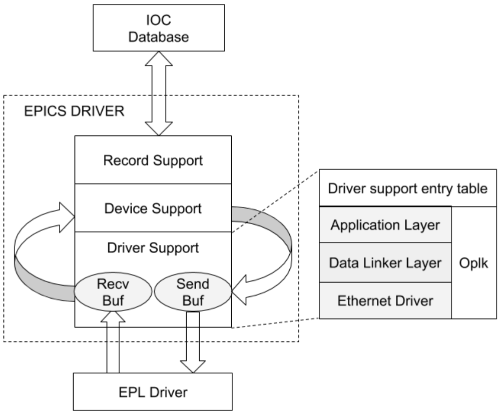


Figure 3: Software structure of the EPICS driver.

PROTOTYPE SYSTEM AND PERFORMANCE TEST

According to the hardware architecture shown in Fig. 2, we set up a prototype system. This is shown in Fig. 4 It is composed of one PC and six FPGA boards. The OS of the PC is Centos7.4.1708 with real-time patch rtai-3.10.75. IOC application runs on the PC, and the function of MN is integrated within the IOC application. The model of the FPGA board is DE2-115, and it acts as CN. The EPL speed is 100Mbps. The oscilloscope and the Hilscher netAnalyzer are the tools for the performance test.

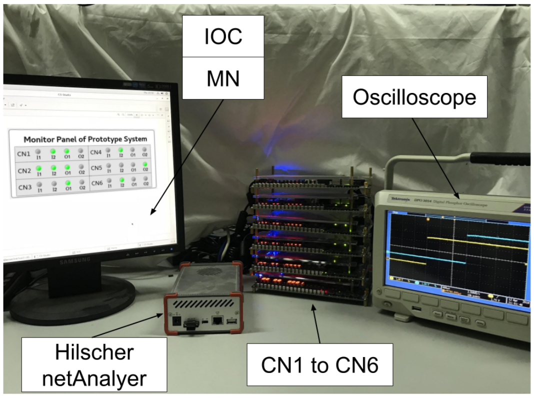
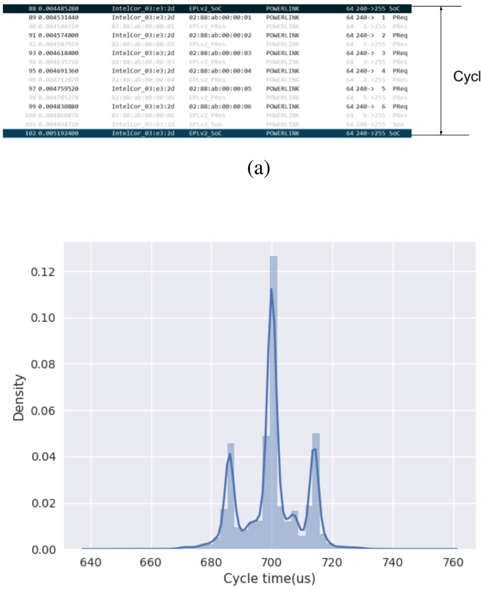


Figure 4: Photo of the prototype system.

Test of Cycle Time

openCONFIGURATOR is an open-source toolkit for easy configuration of the EPL network. The cycle time of the prototype system is set to 700µs by openCONFIGURATOR. In order to test the cycle time, the Hilscher netAnalyzer is used to capture the EPL frames. The cycle time can be calculated from the interval between two consecutive SoC frames, as shown in Fig. 5(a). About 27000 cycle times are calculated from the EPL frames, and the mean value of cycle time is about 703s, it is closed to the configured cycle time. Figure 5(b) is the probability density distribution curve of cycle time fitted by the python program, the peak of the density is around the value Cycle time = 700µs.



```
 88 0.004485280   IntelCor_03:e3:2d   EPLv2_SoC           POWERLINK   64 240->255 SoC
 89 0.004531440   IntelCor_03:e3:2d   02:88:ab:00:00:01   POWERLINK   64 240->  1  PReq
 90 0.004540310   02:88:ab:00:00:01   EPLv2_PRes          POWERLINK   64   1->255  PRes
 91 0.004574800   IntelCor_03:e3:2d   02:88:ab:00:00:02   POWERLINK   64 240->  2  PReq
 92 0.004587910   02:88:ab:00:00:02   EPLv2_PRes          POWERLINK   64   2->255  PRes
 93 0.004618400   IntelCor_03:e3:2d   02:88:ab:00:00:03   POWERLINK   64 240->  3  PReq
 94 0.004635710   02:88:ab:00:00:03   EPLv2_PRes          POWERLINK   64   3->255  PRes
 95 0.004691360   IntelCor_03:e3:2d   02:88:ab:00:00:04   POWERLINK   64 240->  4  PReq
 96 0.004712870   02:88:ab:00:00:04   EPLv2_PRes          POWERLINK   64   4->255  PRes
 97 0.004759520   IntelCor_03:e3:2d   02:88:ab:00:00:05   POWERLINK   64 240->  5  PReq
 98 0.004785270   02:88:ab:00:00:05   EPLv2_PRes          POWERLINK   64   5->255  PRes
 99 0.004830880   IntelCor_03:e3:2d   02:88:ab:00:00:06   POWERLINK   64 240->  6  PReq
100 0.004860870   02:88:ab:00:00:06   EPLv2_PRes          POWERLINK   64   6->255  PRes
101 0.004894720   IntelCor_03:e3:2d   EPLv2_SoA           POWERLINK   64 240->255  SoA
102 0.005192400   IntelCor_03:e3:2d   EPLv2_SoC           POWERLINK   64 240->255 SoC
```

(a)

(b)

Figure 5: (a) Frames captured by the Hilscher netAnalyzer, (b) Probability density distribution curve of cycle time fitted by the python program.

Test of Response Time

The response time is the time difference between the arrival time of the input signal and the output signal. It consists the process time of input/output signals on the FPGA boards, the data communication time, cable propagation delay and hub delay. The data communication time is the main part, the others are in 1-10µs levels.

For the prototype system, the communication process is shown in Fig. 6. According to the EPL communication sequence, the CN1 is the first communication CN and the CN6 is the last one in a cycle. So the maximum communication time ($T_{comm-Max}$) is the time difference between T_{input} and T_{output}. $T_{comm-Max}$ can calculated from the frames captured by the Hilscher netAnalyzer, it ranges from 905µs to 1040µs [4].

When the oscilloscope is used to test the response time, the figure shown on the oscilloscope is not stable. The reason

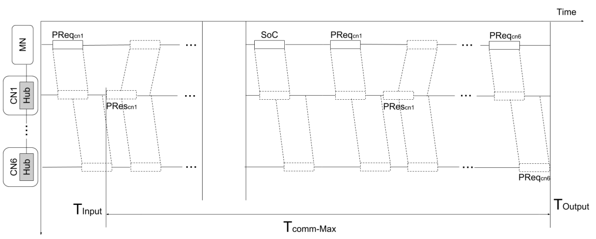


Figure 6: Space-time diagram of the response process for the EPL operating in the standard mode. T_{input} is the moment before CN1 sends the frame PRes, T_{output} the moment after CN6 receives the frame PReq, $T_{comm-Max}$ is the maximum response time of the prototype system.

is that response time is variable. Fig. 7 is the test result with the mode "single trigger" of the oscilloscope, the response time is about 1070µs, it is close to the range of $T_{comm-Max}$.

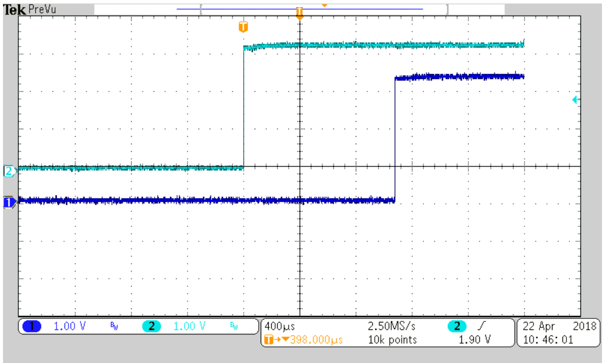
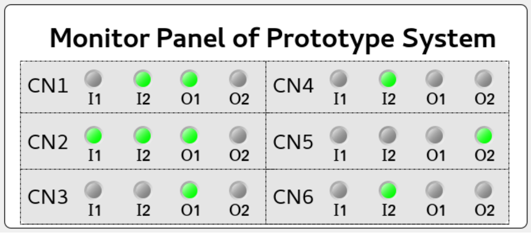


Figure 7: Test result of response time by oscilloscope with mode "single trigger". The ch2 is the input signal of CN1, the ch1 is the output signal of CN6.

Monitor Panel

The IOC application integrates the functions of MN, so the IO data of CNs are monitored by the records. Figure 8 is the monitor panel developed by CSS/BOY. Each CN has two digital input signals and two digital output signals. The green color represents logic "1", the gray color represents logic "0". The response time of the prototype is around 1ms, the EPICS record process time is not fast enough, so the monitoring of the records is only the snapshot of the IO status.



Monitor Panel of Prototype System

	I1	I2	O1	O2		I1	I2	O1	O2
CN1	⚪	🟢	🟢	⚪	CN4	⚪	🟢	⚪	⚪
CN2	🟢	🟢	🟢	⚪	CN5	⚪	⚪	⚪	🟢
CN3	⚪	⚪	🟢	⚪	CN6	⚪	🟢	⚪	⚪

Figure 8: The monitor panel of the prototype system.

CONCLUSION

We design a distributed IO system based on EPL under the EPICS architecture and establish the prototype system composed of a PC and six FPGA boards. An EPICS driver is developed to monitor the IO status of each CN with EPICS records. The cycle time of the prototype system is about 703µs, and the response time is around 1ms. The EPICS driver runs in user space now. When it can run in kernel module, the cycle time can be shorter, and the response time will be less than 1ms.

REFERENCES

[1] EPICS, https://epics.anl.gov/

[2] Ethernet Powerlink, https://www.ethernet-powerlink.org/

[3] openPOWERLINK, http://openpowerlink.sourceforge.net/web/

[4] Ferrari. P, A. Flammini, and S. Vitturi. "Response times evaluation of PROFINET networks", in *Industrial Electronics, 2005. ISIE 2005. Proceedings of the IEEE International Symposium*, Vol. 4. IEEE, 2005, pp. 1371-1376.