# AN EMBEDDED EPICS CONTROLLER BASED ON ETHERNET/SERIAL BOX

G.Y. Jiang, L.R. Shen, SSRF, Shanghai 201800, P. R.China

## *Abstract*

The SSRF (Shanghai Synchrotron Radiation Facility) control system takes the Ethernet as backbone. All kinds of serial devices such as vacuum pumps are connected to Linux IOCs via a kind of Ethernet/serial box made by Moxa company. In the pre-research stage of SSRF, the old model of this Ethernet/serial box was only a simple Ethernet/serial protocol converter which was functioned by firmware. Aim to this, we have developed several kinds of EPICS device drivers based on NetDev for our serial devices.

Recently, Moxa company has upgraded the converter by replacing old arm9 CPU with a more powerful Intel Xscale CPU. It supports MontaVista Linux as its embedded OS, also cross-compiler is provided to make further development available. Since we have decided to use the new model of converter in our facility finally, we manage to port EPICS IOC core on MontaVista Linux and implement the same function on the new converter as old one's to avoid modifying existent EPICS device driver. By these, the dedicated Linux IOC can be omitted and the whole system can be more efficient and expandable.

Details of the necessary integration work and initial operation experience will be discussed in this paper.

## INTRODUCTION

The SSRF is a third generation light source designed to produce high brightness and flux soft X-ray and hard X-ray in the energy region of 0.1~40keV located in Shanghai, China. It consists of a 150MeV linac, a 3.5GeV booster, a 3.5GeV storage ring and 62 beam lines and experimental stations [1]. The whole project is under construction and the user operation is scheduled to start in April 2009.

SSRF control system uses the EPICS (Experimental Physics and Industrial Control System) as the software environment. Usually EPICS can be divided into IOC (Input Output Controller) and OPI (Operator Interface) parts [2]. IOC is a platform that originally assumes to use a VME front-end computer as the hardware and VxWorks as the underlying target real-time operating system. However, from the version of EPICS 3.14.x, the IOC can run on various operation systems such as Linux, RTEMS, etc. Meanwhile, small-sized, dedicated instruments/controllers with Ethernet connectivity have become popular in accelerator controls [3]. These two reasons made us choose Ethernet as control network and used numbers of Linux soft IOCs to construct our system. Also, to simplify the controls of all serial devices, several tens of Ethernet/serial box was adopted to interface all serial devices to soft IOCs without using VME IOC and other I/O cards.

## PROTOTYPE OF SERIAL DEVICE CONTROLS AT SSRF

### *Nport5610*

In the pre-research stage of SSRF, Nport5610, which is a commercial Ethernet/serial box made by Moxa company [4] was chosen. It is composed of an arm9 CPU, a small lcd screen, buttons, 8/16 serial ports and two 10/100M Ethernet ports. Each Nport5610 has an IP address on the Ethernet. Users are allowed to modify its IP address and serial port parameters basically through telent/http or on the panel. Data flow from Ethernet is forward to each serial devices in the way of "IP:PORT". For example, to access serial device on port 4001, the socket program will address "IP:4001". Thus, we follow this rule in EPICS, i.e., a typical input filed in EPICS db file is like this:

```
@Hostname(Port)#command
```

### *Application of NetDev at SSRF*

To be controlled by soft IOCs, EPICS asynchronous device/driver support should be designed as all these serial devices are slow response devices. The main difference in designing these device/driver supports is that what to send/receive depends on each of the devices, whereas how to send/receive, as far as all these devices concern is common.

With the advantage of EPICS community, we introduced NetDev [5] which is a common asynchronous device driver framework developed originally by KEK to fulfil our request. NetDev is composed of device specific modules, asynchronous I/O Library module and common driver support module. The lowest layer, common driver support module, manages I/O request queue in MPF (Message Passing Facility) by carefully designing multi tasks and avoiding race condition on soft IOC.

The middle layer asynchronous I/O library encapsulates technical details of an asynchronous device support of EPICS in two main functions, a generic initialisation function and a generic read/write function. All I/O requests will be put into the queue in MPF mentioned above.

In the upper layer, device specific modules, by adding/modifying functions of link field parser, command constructor and response parser, we created several device driver supports, following Table 1 lists all these drivers developed in SSRF now:

Table 1: Device driver supports of serial devices in SSRF

| Device | Maker | Protocol |
|---|---|---|
| Ion pump power | Custom | TCP/UDP |
| supply controller | | |
| Vacuum gauge | Varian | TCP/UDP |
| Dabo power supply controller | Custom | TCP/UDP |
| Hwhr step motor controler | Custom | TCP/UDP |
| Monitor module of VME cabinent | ELMA | TCP |

Need to be mentioned that besides serial devices, NetDev is also used as device drivers of several kinds of PLCs in SSRF such as Omron, Yokogawa and Simens.

### *Shortage of Nport5610*

During the period of test, we noticed that there was an abnormal connection time out error when using the driver of dabo ps controller and Nport5610. The data transfer for this controller is very rapid (0.1s/time), while the data size is small (4 bytes/8 bytes). Netstat command of Linux showed that data were accumulated on the socket of soft IOCs. Finally we located the problem was aroused from Nport5610. In that period, we could do nothing with Nport5610 as it is like a black box and all the software in firmware is hidden to us for commercial secret reasons. We also considered of changing another model of Ethernet/serial box produced by Digi International Inc., but they are more expensive.

## ETHERNET/SERIAL BOX UC7400

### *Hardware*

With the suggestion of engineers from Moxa, we changed our focus to their new model of Ethernet/serial box named as UC7400 which is an upgraded product with an intel xscale CPU of 266Mhz, 128MB SDRAM of main memory, 32MB flash memory, 8/16 serial ports, buttons and a lcd panel. With an additional PCMCIA card, it can support wireless network. Figure 1 shows the new Ethernet/serial box UC7400:

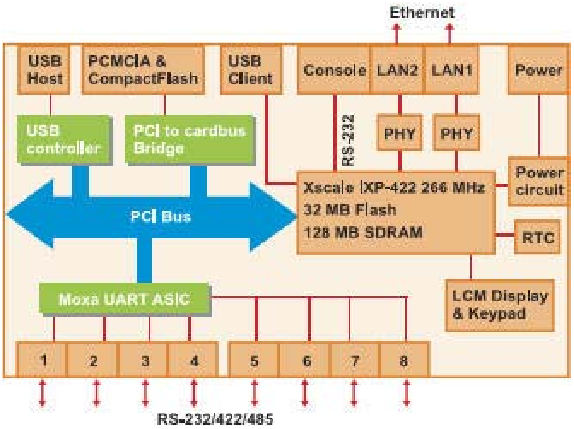


Figure 1: Hardware block diagram of UC7400.

As UC7400 using a 266Mhz CPU which is more powerful than those old VME CPU boards such as VME2302 [6], we started to consider about the possibility to run EPICS IOC core on it instead of using it as only an Ethernet serial protocol converter.

### *Software*

The most attractive characteristic to us is that UC7400 has a MontaVista Linux [7] embedded in it. MontaVista Linux is designed for the scalability, dependability and performance required for intelligent devices. It not only supports the broadest range of processor architectures used in high-end embedded systems, including x86, PowerPC, ARM, SH, and MIPS, but also includes a comprehensive set of development tools. The producer MontaVista is the undisputed leading provider of commercial-grade Linux development platforms for intelligent devices and communications infrastructure. In addition, Moxa is register user of MontaVista Linux that means they can acquire complete, end-to-end support from source code level.

UC7400 is using a kernel which version is 2.4.18_mvl30-ixdp425. Coming with UC7400, a cross-compile toolset includes GNU gcc/g++ which version is 3.3.2 for intel xscale is also available. The develop procedure is like this: cross-compiler toolset is installed on a redhat Linux host, after compilation, the program is downloaded to target via network and executed. With supplying abundant Linux function library and the scalable Linux components, UC7400 is suitable for designing custom programs by ourselves.

As mentioned in introduction that there is a recent trend toward Linux and multi-platform compliance of the latest EPICS base of version R3.14. Above reasons encouraged us to port the EPICS IOC core onto MontaVista Linux platform to make UC7400 embedded EPICS controller. It should decrease large number of dedicated Linux IOCs since we will use tens of Ethernet/serial box in whole control system. Furthermore, all the software on Ethernet/serial box becoming transparent will help us identify the problem easily when error occurs.

### *Porting*

MontaVista Linux is sufficiently like redhat Linux makes porting it a feasible thing to do. The porting was done by following steps:

- Creating 3 new makefiles for new architecture Linux-xscale by using Linux-x86 makefiles as template and doing some modifications of compiler switches.
- Trivial modifications of several EPICS base source codes such as ipAddrToAsciiAsynchronous.cpp.

After this, we used a Linux PC as the host machine and cross-complied the source code of EPICS base3.14.8.2 on host. Since the UC7400 supports NFS protocol, when controller booting, we utilized NFS to mount EPICS base and user application files which exported by host machine, environment variables such as LD_LIBRARY_PATH were set automatically in bash shell of client machine to be pointed to right place. CF card also gives the possibility to download these files to UC7400 and run the EPICS application locally if additional large CF card is installed. Users can interact with UC7400 from a remote computer through telnet and finish the development comfortably.

### *Proxy*

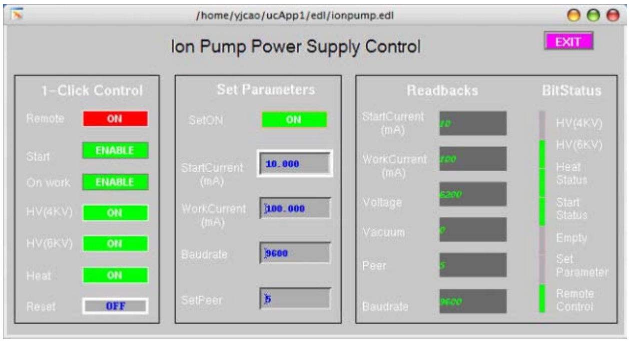
In the old model of Ethernet/serial box Nport5610, firmware resident function converts data flow between Ethernet and serial port. In the new UC7400, the Moxa does not supply this program. Because we didn't want to do further modification to the accomplished EPICS driver of serial devices, we should write the code to implement this function and call it "proxy". This proxy program would supply three parts of functions:

- Nonblock socket recv/send.
- Multi threads for managing multi serial ports.
- Can read configure file of each serial port from file system and save logs to file system.

We implemented these three parts of functions separately in three c files named as socket.c, readcfg.c and proxy.c. Encapsulating all the socket related functions in one file and use it as a library gives the feasibility of developing other network program on UC7400 in future. proxy.c main program uses pthread library to manage the multi threads which respect to serial ports operations.

### *Embedded EPICS Controller on UC7400*

JJJ ion pump power supply controller which was listed in Table 1 is a domestic product produced for SSRF after our work was finished. It was a good chance for us to repeat the general procedure of driving a new device in EPICS from the beginning: creating a new driver (based on netDev), writing EPICS db file and drawing a panel (with edm). The EPICS records for this controller include ai, mbbidirect, bo, stringout, longin, longout. The figure 2 illustrates the control panel displayed on an OPI:



/home/yjcao/ucApp1/edl/ionpump.edl

Ion Pump Power Supply Control

1-Click Control
Remote ON
Start ENABLE
On work ENABLE
HV(4KV) ON
HV(6KV) ON
Heat ON
Reset OFF

Set Parameters
SetON ON
StartCurrent (mA) 10.000
WorkCurrent (mA) 100.000
Baudrate 9600
SetPeer 5

Readbacks
StartCurrent (mA) 10
WorkCurrent (mA) 100
Voltage 5200
Vacuum 0
Peer 5
Baudrate 9600

BitStatus
HV(4KV)
HV(6KV)
Heat Status
Start Status
Empty
Set Parameter
Remote Control

Figure 2: Control panel for JJJ ion pump ps controller.

Finally with this power supply controller, we did a simple CPU loading test of UC7400. When no EPICS applications running on it the CPU status was 0.8-1.4% using. With connecting 16 ports serial devices to UC7400 and IOC core running on it at the same time, the values increased to around 35%. Also we have checked the memory usage, about 40M memory was used for MontaVista Linux itself among 128M memory, there were enough free memory left for EPICS applications.

## STATUS

The idea of embedded EPICS controller based on UC7400 was first proposed in the end of 2006. In March, 2007, all the work about porting and migration has been accomplished. In this summer, some tests were carried out concerning the serial devices of vacuum in linac controls. Furthermore, the SSRF booster commissioning is scheduled in the coming October, we are going to replace more Nport5610 with UC7400. Our target is to use only UC7400 in storage ring. Ps, besides serial devices, PLCs used in SSRF can also benefit from the work in this paper as they can be controlled by this embedded EPICS controller too.

## CONCLUSION

We have completed extensive testing and evaluation of this embedded EPICS controller based on MontaVista Linux. It has proved to be robust, versatile and reliable under all operational conditions and well suited to our requirements. Practically, the Ethernet/serial box UC7400 is convenient for the quick setting, and considerably enhances the degree of freedom of the installations with the compact controller size and wireless characteristic. Since we can monitor the controller status and operate the controller from the remote places, we can enhance the controllability of the distributed controllers.

## AKNOWLEDGEMENT

The authors would like to thank Mr. Odagiri at the KEKB control group for developing the NetDev. They would also like to thank Mr. Yu and other engineers from Moxa company for sharing their experience of using UC7400.

## REFERENCES

[1] Z. T. Zhao et. al., "PROGRESS OF THE SHANGHAI SYNCHROTRON RADIATION FACILITY", APAC' 04, Gyeongju, Mar. 22-26, 2004.

[2] M. Kraimer et.al, "EPICS: Porting iocCore to Multiple Operating Systems," ICALEPCS'99, Trieste ,Italy ,Oct .1999 .

[3] Kenneth, Jr. et. al., "EPICS for PDAs", ICALEPCS 2001, San Jose, Nov. 27-30, 2001.

[4] http://www.moxa.com/

[5] K. Furukawa et. al., "Implementation of EPICS Device Support for Network-Based Controllers", ICALEPCS 2001, San Jose, Nov. 27-30, 2001.

[6] G. Waters, et.al, "TRIUMF/ISAC EPICS IOCs Using a PC104 Platform", ICALEPCS'2003, Gyeongju, Korea, Oct. 2003.

[7] http://www.mvista.com/