DEVELOPMENT OF AN INTELLIGENT MOTOR CONTROL UNIT WITH ETHERNET CONNECTIVITY

T. Masuda*, T. Fukui, R. Tanaka, K. Yanagida
SPring-8, Hyogo 679-5198, Japan

Abstract

We developed a new motor control unit (MCU) with network connectivity, and installed the MCUs for the SPring-8 linac pulse motor control by reengineering VME systems. We still hold VMEbus system, but we replaced VME motor control boards by the MCUs to provide local controllability for machine experts without spoiling processing speed. We adopted a PC architecture industrial controller ND-MCU, which has 200MHz SH-4 CPU and three PCI slots, and commercial-available PCI motor control boards were installed to the slots.

Routine operation sequences of a pulse motor such as initialization, extraction, insertion and movement are defined and stored as embedded procedures in the MCU. Because the sequences are performed as concurrent tasks for each axis of a motor, we can operate a maximum of twelve axes at the same time. The MCU holds conversion equations between pulse counts and physical values to achieve motor operations by using physical values only. Since the embedded processes run on the MCU, the MCU can perform the same operations both to remote control via network and to local control by a touch panel.

We report the new motor control unit by focusing on its functions and advantages of the operations.

INTRODUCTION

The SPring-8 1-GeV linac works as an injector not only for an 8-GeV SPring-8 storage ring but also for a 1.5-GeV NewSUBARU storage ring [1]. Since May 2004, SPring-8 started top-up operation in which electron beams are continuously injected into the SPring-8 storage ring to keep stored beam current approximately constant [2]. And since September 2004, SPring-8 started the simultaneous top-up injections to two storage rings of SPring-8 and NewSUBARU. So the injector linac was highly required to have stability and reliability.

Toward the realization of the top-up operation, we needed to re-engineer the linac control system hardware to obtain enough reliability, stability and availability [3]. In the old linac control system, VME computers were placed close to the linac equipment such as modulators and magnet power supplies, and direct input/output (I/O) boards were used to control the linac equipment. Total 25 VME computers were distributed along the length of the linac.

VME pulse-motor control (PMC) boards were used in the old control system to control many pulse-motor drivers to operate phase shifters, attenuators, beam slits, and wire grid monitors (WGMs). The PMC boards brought us fast controllability, but on the other hand local controllability of the motor drivers was sacrificed at maintenance time. Local operations of the motor drivers were performed either by attaching a serial terminal to the VME CPU, or using a network-connected terminal. When the control system was updated, we considered achieving the local controllability of the motor controllers. We developed new network connectable motor control unit, MCU.

*masuda@spring8.or.jp

MOTOR CONTROL UNIT

The MCU was developed based on a PC-based industrial controller ND-MCU [4]. The MCU is a 4U height, 19”-rack mountable and diskless system. The controller has a CPU of 200MHz SH-4 (SH7751R) [5], a 10Base-T/100Base-Tx Ethernet interface, and three PCI slots. A real-time operating system (OS) NORTi [6], which follows the μiTRON4.0 [7] specification, is used as the OS of SH-4. For a local operation, the MCU has a 4” LCD touch panel on the front panel. A picture of the MCU is shown in Fig. 1.

Figure 1: A picture of the MCU.

For a choice of a PCI motor control board, we took account of function compatibility and hardware interface compatibility with the VME PMC board. We chose a PCI-7414V PCI board [8] from some commercial available products as a result of control test of some kinds of motor drivers used in the linac. The PCI-7414V independently drives four axes of pulse motors by using pulse outputs that are electrically isolated with GMR (Giant Magneto Resistance) isolator. As the results, the MCU can independently control a maximum of twelve axes. The main specification of the PCI-7414V is listed in Table 1.

We designed the MCU to satisfy good local controllability and remote operation, and to achieve the following features. First, the MCU can perform complicated fixed-sequences such as initialization, extraction and movement, and the sequences can be

modified and downloaded via a network, if necessary. Secondly, the MCU can provide intuitive operation for the local and remote control by using physical value. Thirdly, the MCU can provide seamless and equivalent operation between the local and remote control.

Table 1: Specifications of a PCI-7414V motor control board.

Number of axes	4 (individual control available)
Pulse output	CW/CCW or OUT/DIR output mode Max. 6.5Mpps output pulse rate Output counts: -134217728 ~ 134217727 GMR isolation +5V DC output (differential line driver)
Encoder input	Incremental encoder inputs (A/B/Z-phases) Max. 1MHz input pulse rate Counter length: 28bits High-speed optical isolation +5V DC input
Other signal output	4-bit general purpose outputs +5V ~ +48V DC output Max. 100mA output current
Other signal input	+/- EL and the ORG signal inputs available 12-bit general purpose inputs Optical isolation +5V~+48V DC input

Fixed Sequences Execution

The MCU executes complicated fixed-sequences according to equipment control instructions. Table 2 shows the currently embedded sequences in the MCU.

Table 2: Developed fixed sequences embedded in the MCU

Sequence	Action
initialize	Determine the origin of an axis with a low speed drive
extract	Drive a pulse motor at a low speed to the extraction limit of an axis.
insert	Drive a pulse motor at a low speed to the insertion limit of an axis.
nominal	Return to the origin of an axis with a trapezoid drive.
move to	Move to the specified position of an axis with a trapezoid drive.

The MCU has execution buttons for the local operation corresponding to the fixed sequences in an operation panel as shown in Fig. 2. For example, when we need to determine the origin of the motor driver, we touch the init button on the panel. Then, the MCU starts the embedded initialization sequence. Thus, we can execute the complicated motor control by using the fixed sequences without any difficulties even for the local operations.

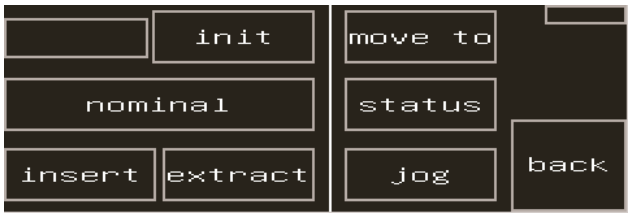

Figure 2: An operation panel on the MCU LCD touch panel.

We designed the MCU to work as a socket server for the remote control. We can remotely instruct the MCU to execute a fixed sequence by sending a corresponding command via a network. The feature simplifies remote operations, and reduces the number of network communications with the MCU. Consequently, the MCU provides fast network controllability.

The MCU concurrently executes the fixed sequence for each axis on the NORTi OS, so that it can execute a maximum of 12 tasks concurrently for the local and remote operations. This feature reduces the total throughput of the sequence executions.

Intuitive Operation by Using Physical Values

In order to provide intuitive operation by using physical value, the MCU convert pulse counts and encoder counts into physical values, and vice versa by using embedded conversion equations. We can specify and download up to ten equations with ten arbitrary coefficients into the MCU in advance. We need to set three equations for each axis from the embedded equations via a network. The three equations are a conversion of pulse counts into physical values, a conversion of encoder counts into physical values, and a conversion of physical values into pulse-counts. At the same time, we have to specify the coefficients for each conversion equation via network.

The conversion equations are useful to set the destination and read the current position by physical value for local and remote controls. This feature greatly improves the local operability, especially in the case of a complicated conversion equation.

Seamless and Equivalent Operability

Since the MCU holds both output pulse counts and encoder counts on the PCI-7414V PMC boards, we can switch between the local and the remote operations without losing the current positions. This feature of the MCU brings us seamless operability between the local and remote control. The local operations and the remote operations are completely equivalent.

Local Operation

Two operation modes are prepared for the local control of the MCU. One is a setup mode of the MCU and the other is a drive mode for each axis.

In the setup mode, we can set up operation conditions of the MCU such as IP address and subnet mask of a network interface, output pulse rates, and logic level of

end-limit signals. Except for the network interface, we can also set up them from the remote.

In the drive mode, the MCU provides jog control for the local operation in addition to the embedded sequences as shown in Table 2. We can drive each axis by using fast and slow control buttons in a jog control panel.

Since we need to monitor the drive status of each axis during the execution of the embedded sequence, we prepare a status panel to display output pulse counts, encoder counts, limit switches status, result of initialization and so on as shown in Fig. 3.

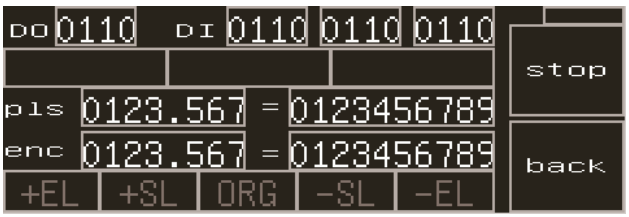

Figure 3: A status panel on the MCU LCD touch panel.

Remote Operation

We designed socket interfaces which define command formats for the network control using TCP protocol. A format of action commands to the MCU is written as follows:

cmd:#board:#axis arg1 arg2 … ,

where,

cmd : command,
#board : board number (A, B, C),
#axis : axis mumber (1, 2, 3, 4),
arg1, arg2…: arguments of the command.

Table 3: The main commands of the MCU socket interfaces.

Command	Function
move_to	Move to the specified physical position. Execute move to sequence
exec	Execute the fixed sequence of *initialize*, *insert*, *extract* and *nominal*.
stop	Stop the current executing sequence and pulse output.
phy_conv	Choose a conversion equation of pulse counts into physical values.
phy_conv?	Get a current conversion equation of physical values into pulse counts.
encphy_conv	Choose a conversion equation of encoder counts into physical values.
pulse_conv	Choose a conversion equation of physical values into pulse counts.
position?	Get the current positions and status.
param_set	Set operation parameters of an initial speed, a drift speed, an acceleration time and a deceleration time.
save	Save current operation parameters in a flash ROM.

In query commands to the MCU, a question mark accompanies cmd is used as follows:

cmd?:#board:#axis arg1 arg2 … .

The MCU returns the result (*ok* or *fail*) to the client process, which sent a control command. The main commands are listed in Table 3.

The commands are categorized into three groups according to their functions. One is a group of the commands for the fixed sequence embedded in the MCU like *initialize*. Second is a group of the commands for choosing conversion equations between the pulse counts and physical values such as *phy_conv* and *pulse_conv* commands. And third is a group of the other commands such as getting a current status of the MCU, setting an output pulse rate and so on such as *position?* and *save* commands.

The MCU supports a maximum of five TCP connections for client processes. Typical round trip time between a socket client process and the MCU to get a current status via fast Ethernet is about 90msec, which is almost consumed inside the MCU.

APPLICATION TO THE LINAC CONTROL SYSTEM

We newly installed twenty MCUs at the upgrade time of the linac control system [3]. In order to keep connectivity with all the kinds of present motor drivers in the linac, two kinds of interface-boxes were newly prepared between the motor drivers and the PCI-7414V boards. At the beginning, we worked to fix and optimize the MCU software, but now the MCU system is substantially reliable and stable.

Since the MCU holds the current positions of each axis, we can continue the MCU operations without the initialization even after operation client computers were down or rebooted.

When a trouble happened to a motor driver, machine experts investigated the motor driver by the local control of the MCU. In that case, the embedded sequences were very helpful and convenient because we were able to instruct the MCU to execute the sequences without any special knowledge about differences between motor drivers such as gear ratio. The MCUs are instructed to keep the differences as the given parameters of the execution commands in advance.

SUMMARY

We have succeeded in developing the intelligent MCU to achieve good local controllability without spoiling remote controllability. The MCUs work well with substantial reliability and stability.

In order to achieve good local controllability, we designed the MCU to provide the capabilities of fixed sequences execution and intuitive operation by using physical values. These features actually bring us simple and intuitive local controllability requiring no special knowledge about motor drivers. Also the features succeed in simplifying remote operations. We can reduce a

number of communications with the MCU by using the embedded sequences in the MCU. Concurrent execution of the fixed sequences reduces total throughput of the remote operations.

Since the MCU holds current positions in it, we succeed in achieving seamless and equivalent operations between the local control and the remote control. This feature of the MCU enables us to operate the linac continuously even though client computers are down or rebooted. Consequently, the MCU greatly contributes to enhancing availability of the linac operation together with its reliability and stability.

The MCU has enough flexibility and expandability due to its adaptive feature of PC architecture and embedded software. The MCU PCI boards are not limited to the pulse-motor control boards only; rather we can build various kinds of controllers by combination of other kinds of PCI boards.

REFERENCES

[1] A. Ando *et al*., "Isochronous storage ring of the NewSUBARU project", J. Synchrotron Rad. 5 (1998) pp.342-344.

[2] H. Tanaka *et al*., "Top-up Operation at SPring-8 – Towards Maximizing the Potential of a 3r Generation Light Source", Proc. of EPAC'04, Lucerne, Switzerland, 2004, p. 222.

[3] T. Masuda *et al*., "Upgrade of the SPring-8 Linac Control by Re-engineering the VME Systems for Maxmizing Availability", Proc. of ICALEPCS'03, Gyeongju, Korea, 2003, p.295.

[4] Nichizou Electronic & Control Corp., http://www.ndssf.co.jp/

[5] Renesas Technology Corp., http://www.renesas.com/

[6] MiSPO Co., Ltd., http://www.mispo.co.jp/ (Japanese only)

[7] ITRON project, http://www.ertl.jp/ITRON/home-e.html/

[8] Interface Corp., http://www.interface.co.jp