Presented at 1982 Nuclear
Science Symposium, Washington, DC,
19-22 October 1982

BNL 32173

Conf-82-011--26

A GENERAL-PURPOSE STEPPING MOTOR-ENCODER
POSITIONING SUBSYSTEM WITH STANDARD
ASYNCHRONOUS SERIAL-LINE INTERFACE*

F. W. Stubblefield and J. L. Alberi[†]

Brookhaven National Laboratory
Upton, New York 11973

October, 1982

*This research was supported by the U. S. Department of Energy:
Contract No. DE-AC02-76CH00016.

[†]Present address: Bell Labs., Lincroft, NJ 07738.

By acceptance of this article, the publisher and/or recipient acknowledges
the U. S. Government's right to retain a nonexclusive, royalty-free license
in and to any copyright covering this paper.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

A GENERAL-PURPOSE STEPPING MOTOR-ENCODER
POSITIONING SUBSYSTEM WITH STANDARD
ASYNCHRONOUS SERIAL-LINE INTERFACE*

ENL--32173

F. W. Stubblefield and J. L. Alberi[†]

DE83 004634

Brookhaven National Laboratory
Upton, New York 11973

Abstract

A general-purpose mechanical positioning subsystem for open-loop control of experiment devices which have their positions established and read out by stepping motor-encoder combinations has been developed. The subsystem is to be used mainly for experiments to be conducted at the National Synchrotron Light Source at Brookhaven National Laboratory. The subsystem unit has been designed to be compatible with a wide variety of stepping motor and encoder types. The unit may be operated by any device capable of driving a standard RS-232-C asynchronous serial communication line. An informal survey has shown that several experiments at the Light Source will use one particular type of computer, operating system, and programming language. Accordingly, a library of subroutines compatible with this combination of computer system elements has been written to facilitate driving the positioning subsystem unit.

Introduction

Many devices to be incorporated into experiments at the National Synchrotron Light Source will require automated positioning. Included among these devices will be beam line elements such as collimating slits, monochromators, and mirrors and spectrometer components such as sample changers, goniometers, and detector transports. It is obvious that the dynamical mechanical properties of such a wide variety of devices would vary over such a wide range that devising a single scheme for their control by closed loop servo techniques would be extremely difficult. A more reasonable approach is to control as many of the devices as possible in open loop fashion with stepping motors and position encoders. This approach has been taken with full recognition of the usual disadvantages of stepping motors, i.e., relatively low shaft torque and power, high power consumption, and susceptibility to vibration during movement. The effects of these disadvantages can be minimized, however, by designing a subsystem which allows the user to choose the basic operating parameters of individual stepping motors. Thus, the method chosen for submitting positioning commands to the subsystem should lend itself easily to submitting commands to establish stepping motor parameters of operation.

Since the subsystem is to be used in a number of different types of experiments, priority has been placed on making it easy to maintain. Operation of the unit can be made visible to the user or maintainer by having a large amount of status information displayed in alphanumeric form in real time, i.e., simultaneously with operation of the unit. This method has been applied by having each major part of the subsystem generate a video display of its status. One of these parts generates a real-time display of the current and target positions of all stepping motor-encoder pairs in the subsystem. Thus a display which would normally have to be supplied by the user is provided by the subsystem.

As an additional aid to device maintenance, the unit has been designed so that it is capable of being operated by very simple driving devices. Any device which generates ASCII⁽¹⁾ characters over a standard RS-232-C⁽²⁾ asynchronous serial line can be used to drive the motor controller unit. Thus a simple ASCII character terminal can be used to drive the unit during maintenance procedures. Also, most computers which are known to be in the process of being built into experiment stations at the Light Source provide input/output ports for RS-232-C asynchronous communication lines. Thus all of these computer systems may be used to drive the motor control unit.

An informal survey has shown that most of the experiment stations which will incorporate one of the motor control units utilize Digital Equipment Corporation (DEC) PDP-11/34 or LSI-11/23 computers and the DEC RSX-11M operating system. Furthermore, experiment control and data acquisition programs for these stations will be written in the FORTRAN⁽³⁾ programming language. For this reason, a library of FORTRAN-callable subroutines has been written to facilitate driving the unit from one of these computer-operating system-programming language combinations.

The motor controller unit itself consists of a chassis box which occupies 10 $\frac{1}{2}$ " vertical height of a standard 19" NEEMA⁽⁴⁾ rack. The unit will control up to 15 stepping motor-encoder pairs. The unit also includes three video monitors for display of status information. While the size and mounting requirements of these monitors are not rigorously defined, the standard unit includes a Satchell-Carlson 6M917T triple video monitor which requires 7" vertical height of a standard 19" rack. The motor controller unit is shown mounted in a rack in Fig. 1.

The forms which input to and output from the motor control unit take are presented in the following sections. Succeeding sections outline the internal structure of the unit. Concluding sections describe the subroutines to be built into users' driving programs, list the experiments utilizing the motor control unit and their corresponding stepping motor and encoder types, and briefly mention the status of the operational units.

Intrasystem Placement

The motor control unit occupies a position within an experiment control and data acquisition system between the main control computer on one side and a set of stepping motor power supplies and position encoders on the other. Thus two types of input to and output from the motor control unit must be considered. The "front end" of the controller is defined to be toward the main control computer (or other driving device); a serial communication line connects the driving device with the motor control unit front end. The video displays of controller status may also be considered to be front end output, since the status information is to be interpreted by the user or maintainer operating the driving device.

The "back end" of the controller is defined to be toward the stepping motor power supplies (translators) and position encoders (usually shaft encoders). Output

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information disclosed herein, or represents that its use would not infringe upon privately owned rights. This report is the property of the United States Government and is loaned to your agency; it and its contents are not to be distributed outside your agency.

ment of

38.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

END

Fig. 1. Motor controller unit, including video display subunits, mounted in rack.

to the power supplies consists of two transistor-transistor-logic (TTL) compatible pulse streams; input consists of three TTL-compatible voltage levels. The interface with the position encoders is not concisely defined, primarily because the unit has been designed to accommodate the whole range of position encoder readout types. The motor control unit is considered to extend, via RS-422⁽³⁾ serial communication lines, to subunits termed encoder serializer modules. Then the interface between the serializer modules and the position encoders consists of a number of parallel TTL-compatible voltage levels or a single TTL-compatible pulse stream. In an additional configuration, the position encoder itself may be absent, and the serializer module interfaces with the stepping motor translator directly.

Front End Input/Output

A computer or less complex device drives the motor control unit by means of a standard RS-232-C asynchronous serial communication line operated in full duplex mode. The bytes which make up the transmission elements are coded as ASCII characters. The ASCII characters are grouped into lines containing a maximum of 64 characters each. Lines containing less than the maximum number of characters are delimited by control characters, usually a <carriage return> <line feed> pair. A complete transaction between driving device and motor controller unit consists of three lines:

- (1) a request line is sent from the driving device to the motor controller;

- (2) the motor controller returns a response line to the driving device. The response line may be empty, i.e., it may consist of only the delimiting control characters;
- (3) an error message line is returned to the driving device by the motor controller. The error line is usually empty.

The number and type of control characters used to delimit the lines of ASCII characters which make up a transaction depend on the characteristics of the device driving the control unit. The motor control unit accommodates different types of driving devices by changing its interaction mode.

Driving Device Interaction Mode

The least complex device which may be used to drive the motor controller unit is a standard ASCII character terminal; the most complex is a small- or medium-scale computer which executes an operating system having extensive capabilities for driving character-oriented peripheral devices. The control unit facilitates interaction with this range of driving devices by employing different device interaction modes.

Manual Interaction Mode. The motor control unit operates in manual interaction mode when the driving device is an ASCII character terminal. It is assumed that a user or maintainer is manually typing commands at the terminal keyboard. The motor control unit responses are designed to facilitate request line entry and to make the response and error message lines readable on the terminal display. In particular, the motor control unit in manual interaction mode executes transactions in the following manner:

- (1) characters typed in the request line are echoed back to the driving terminal as they are received;
- (2) the <rubout> or <delete> control character function is implemented. Typing one of these characters causes the previously typed character to be removed from the request line buffer;
- (3) the request line is terminated upon receipt of a <carriage return> control character. A <carriage return> <line feed> control character pair is echoed back to the driving terminal;
- (4) the response line is terminated with a <carriage return> <line feed> character pair;
- (5) the error message line is terminated with a <carriage return> <line feed> character pair.

Thus all lines of a transaction appear sequentially in easily readable form on the terminal display and the user/maintainer has facilities for correcting typing mistakes when composing the request line.

Processor Interaction Mode. In normal operation, the motor control unit is driven by a computer; when this is the case, the unit operates in processor interaction mode. In processor interaction mode, the manner in which the lines of a transaction are handled and the number and sequencing of delimiting control characters are always dependent on the operating system and programming language executing in the driving computer. Different combinations of operating system/programming

language are accommodated by defining various submodes of processor interaction mode. So far two submodes have been implemented. Because of the physical and logical structure of the unit, additional submodes would be very easy to implement. However, regardless of submode, in processor interaction mode, the unit takes the following actions during transactions:

- (1) characters transmitted in the request line are not echoed back to the driving device;
- (2) the request line is terminated upon receipt of a <carriage return> control character;
- (3) the response line is terminated with a <carriage return> character.

XL Driver Submode. As will be described in more detail in following sections, one combination of computer and operating system used to drive the motor controller is a DEC PDP-11/34 or LSI-11/23 computer running under supervision of the DEC RSX-11M operating system. The logical facility of RSX-11M used for serial communication with the motor control unit is the "XL" message-oriented communication device. An appropriate submode of interaction has been defined for interacting with this type of device. In XL driver submode, the response and error message lines are treated as follows:

- (4) the error message line is terminated with a <carriage return> <line feed> control character pair;
- (5) no wait time interval is inserted between transmission of the response line and transmission of the error message line (i.e., no wait time in addition to the normal asynchronous serial line byte transmission time);
- (6) no echoing of the response and error message lines back to the motor control unit is expected.

Terminal Driver Submode. The terminal driver submode is provided for the case where the motor control unit is driven by a computer executing programs written in the FORTRAN language and transaction lines are transmitted and received using the formatted READ/WRITE statement capabilities of this language. The FORTRAN logical unit used to interact with the motor controller is assigned to a character terminal physical device. Most operating system and FORTRAN object time (or execution support) system combinations require that the motor controller support this type of FORTRAN input/output by performing the following operations:

- (4) the <line feed> prompt control character at the start of the request line is ignored and removed from the request line;
- (5) transmission of the response line back to the computer does not begin until a <line feed> character prompt has been received. Response line characters echoed back to the motor controller are verified against the transmitted characters but are otherwise ignored;
- (6) transmission of the error message line back to the computer does not begin until a <line feed> prompting character

has been received. Error message line characters echoed back to the motor controller are verified against the transmitted characters but are otherwise ignored;

- (7) the error message line is terminated with a <carriage return> control character.

Regardless of the particular type and number of control characters which delimit them, the transaction line contains subgroups of ASCII characters recognizable as English-language mnemonics, words, or numbers. Thus the lines may be checked in real time (e.g., from the video display) by a maintainer.

Characters within the request line must specify three basic types of information: (1) function codes, a set of mnemonics which specify the controller operation to be performed; (2) motor names, character strings for labelling the motor-encoder pairs; and (3) position coordinates, ASCII-coded position values with respect to various coordinate systems used to describe the motor axes. The response line contains information requested by the driving device and supplied by the motor control unit. Typically this information consists of ASCII-coded current position values and mnemonics which indicate the current status of the motor-encoder pairs or the motor control unit itself.

The error line contains a short message describing any error detected by the unit during its attempt to perform a requested function. As mentioned above, this line will be empty if no error has been detected.

Position Specification Mode

The format of some information, specifically the motor labels and the position values, in the request and response lines depends on the current position specification mode of operation of the motor control unit. Three position specification modes have been provided. The modes correspond to increasingly complex methods for visualizing and specifying the motor axis coordinate systems. The three position specification modes are listed below in order of increasing complexity. The format of function codes and status indicators is independent of position specification mode.

Physical Absolute Mode. In physical absolute mode, each motor has a name consisting of a two-digit decimal number associated with its position of physical connection to the control system. The coordinate system provided in this mode is the direct reading of the position encoder. This number has a range of 0 to $(2^{24}-1)$.

Physical Scale Mode. In physical scale mode, the motor naming convention is the same as in physical absolute mode, but the coordinate system is different. On many devices a physical scale may be associated with a motion in order to provide a visual check of device position. The physical scale is usually in engineering or scientific units as opposed to encoder units. The motor control unit maintains a physical scale coordinate system for each motor axis and associates the physical scale with encoder readings via a linear transformation. The linear transformation requires that a zero point and scale factor be specified.

Logical Experiment Mode. In logical experiment mode, each motion (motor) is referred to by a logical name which may consist of up to six characters. The mapping of logical names to physical motor numbers is performed by the control unit. The motor names have meaning within the context of the experiment which is being performed,

rather than the physical layout of the motors and the devices they drive. The same statement applies to the coordinate system used in this mode. Logical experiment coordinates refer to position-related axes used to describe the experiment being performed. There will usually be no physical scale on the device which corresponds to this coordinate system. A zero point and scale factor must also be specified for each motion coordinated in this mode. It is expected that the motor control unit will be used in logical experiment position specification mode most of the time.

The motor control unit also generates front end output in the form of three video displays. These displays may be interpreted by the user to ascertain the positional status of his experiment or may be utilized by a maintainer as an important aid during subsystem repair or maintenance. The first of these displays shows the current status of the motor control unit itself. This status information includes the current interaction and position specification modes. Also, each ASCII character received by the unit is displayed before any additional processing is performed on it and before it is interpreted within the request line. Once the response and error message lines have been composed, each character in these lines is displayed just prior to its transmission back to the driving device. Thus the subsystem maintainer may quickly check whether the serial line input/output from/to the driving device is correct.

The second video display consists of a list of all motor-encoder pairs controlled by the unit. The label (name or number) and status of each motor-encoder pair are shown along with its current and target position values. This display is primarily for the benefit of the user; it allows the user to verify that his devices are being positioned correctly throughout the course of an experiment. Since such a real-time display of device position is usually required for any experimental setup, the motor controller subsystem automatically provides a function for which hardware and software would otherwise have to be added to the main control computer (driving device).

The third video display may be used to exhibit additional information about the stepping motor power supplies or the position encoders themselves. In particular, a more detailed status of the motor driving operations, including the states of limit switches associated with the motor motion, can be provided, as can a more direct display of encoder readout in its natural format. In general, the three displays correspond roughly to the three internal parts of the controller unit to be listed below. Additional details of each display will be given in the subsection on each part.

Back End Input/Output

The motor controller unit interfaces with stepping motor power supplies (translators) via five optically isolated, TTL-compatible signal lines. Two output lines carry pulse streams which cause the power supplies to move the motors, one step per pulse. The two streams correspond to clockwise and counterclockwise motor motion. The total number of pulses determines the total motor travel during a move operation; the pulse rate determines the instantaneous stepping velocity. These pulse streams are suitably ramped up to and down from a maximum stepping velocity at the beginning and end of motor travel. Three input signal lines supply voltage levels which specify the status of clockwise and counterclockwise position limit switches and the home position switch. The home position gives a reference mechanical position by which incremental encoders or pseudo encoders

(implemented by counting stepping motor pulses) can be initialized.

As mentioned above, the motor control unit is considered to extend to the encoder serializer modules. Internally, the control unit communicates with each serializer module via an RS-422 (differential signal pair) asynchronous serial line operated at 9600 baud in full duplex mode. The internal transactions between the control unit and the serializer modules consist of one character (the request character) transmitted to the module and one character (the response character) returned to the control unit. Each encoder is considered to have a maximum resolution of 24 bits or three 8-bit bytes. Five different request characters may be used to read out an encoder:

- (1) initialize character. This request character instructs the serializer module to initialize the encoder and its readout. Usually, no action is required in the case of absolute encoders. The internal scaler (up-down counter) used to accumulate pulses from incremental encoders or pseudo encoders (motor pulses) is set to zero when this character is received. The initialize character is echoed back to the control unit as the response character;
- (2) latch character. This request character instructs the serializer module to latch an encoder reading into an internal buffer. In the case of incremental encoders and absolute encoders without integral gating electronics, the serializer module latches at least two readings into internal buffers in order to ensure that the encoder readings is not in the process of changing during the latching process. The latch character is echoed back to the control unit as the response character;
- (3) transmit low-order byte character. The serializer module extracts the low-order eight bits of the encoder reading from its internal buffer and transmits this bit pattern, as the response character, back to the control unit;
- (4) transmit middle-order byte character. The response is the same as for (3), above, except that the encoder reading middle byte is transmitted back to the control unit;
- (5) transmit high-order byte character. The high-order byte of the encoder reading is transmitted back to the control unit.

Internal Structure

The main body of the motor control subsystem consists of a standard chassis box (Plessey Peripheral Systems PM-1150/5) modified to provide "start" and "halt" control signals to more than one microprocessor. A connector panel has been added to the back of the chassis box. The panel contains 9- and 15-pin "D" type connectors for interfacing with the serial lines to the encoder readout serializer modules and with the TTL-compatible lines to the motor translators.

The chassis box occupies 10 $\frac{1}{2}$ " vertical height of a standard 19" rack and supplies power, cooling, and structural support to three sections of hardware components. Each of the three sections contains both components of common type and components of function-specific type. The primary common component of each section is a microprocessor, a DEC LSI-11/2 single-board level processor. The common components also include a variable size local memory array, a serial line unit for driving a diagnostic terminal, a video display generator module, and a buss terminator module which contains a line time clock.

Hardware components specific to the function performed by each section will be mentioned in the text sections below which describe the individual functions.

Communication among the three sections is accomplished via a shared memory subsystem.^{6,7} This subsystem consists of three shared memory access ports, one for each section, and a potentially large module (many megabytes) of memory accessed via a DEC Modified UNIBUS bussing scheme. The shared memory module may contain any type of UNIBUS-compatible memory. The motor control unit shared memory contains 8,192 bytes of read/write memory (RAM) of metal oxide semiconductor (MOS) type and 196,608 bytes of ultraviolet light erasable, programmable read-only access memory (EPROM). The RAM memory is used for communicating function request/response parameters between pairs of processors; the EPROM memory contains images of operating systems and programs to be executed by the processors.

Software for execution by the processor within each section has a common structure which includes as major components an operating system, a module of trap (software interrupt) service routines, and a FORTRAN main program-overlay set. The interaction of the various components can be explained by describing the bootstrap procedure for the motor control unit as a whole.

At motor control unit bootstrap, each processor automatically executes a short (400(8) bytes) bootstrap program residing, as read-only memory code, in the module which terminates the processor's local buss (Q-buss). The bootstrap program sets an appropriate page number into the page number register of its processor's shared memory access port and loads a short (1000(8) bytes) system loader program from the read-only portion of the shared memory module into processor local memory. Execution of the system loader program is begun; the loader program transfers two modules from shared memory to processor-local memory: an image of a DEC RSX-11S operating system and a module of synchronous system trap (SST) service routines. The operating system could be specially tailored (in terms of device drivers and task execution support routines) for each processor, but in the present case a single image is used for all three processors. The main feature of this operating system is support of a driver for a console (diagnostic) terminal. The module of synchronous system trap service routines supports execution of FORTRAN language main programs and overlays by loading these components from the shared memory module on demand. The operating system (12 Kbytes) and SST service routines module (4 Kbytes)⁸ occupy the lowest part of processor address space.

The system loader transfers program control to the RSX-11S operating system. In turn, the operating system initializes itself and automatically begins execution of a small dummy task built into and loaded along with the operating system image. The dummy task executes only one instruction, a software interrupt, or synchronous system trap, to an SST service routine

which loads and starts execution of the processor's FORTRAN main program. As far as the operating system is concerned, the dummy task executes continuously throughout operation of the motor control unit.

It is the FORTRAN main program, along with its associated overlays, which performs the motor controller subfunctions required of each section. Typically the main program periodically examines either the input serial line port buffer or the shared memory function request area until a function request has been received. The program then decodes the function request and performs the indicated function. The main program and each overlay are restricted to a maximum length of 16,384 bytes.

The logical space between addresses 140 000(8) and 157 777(8) is used as a window for mapping into shared memory physical space. The last (highest) 8,192 bytes of address space correspond to the standard DEC external device registers.

The hardware and software belonging to each section perform a part, or subfunction, of the motor controller subsystem function. The subfunctions bear a hierarchical relationship to each other, in the sense that a processor at one level in the hierarchy may request lower level functions from the processor at the next lower level, but may not request functions of higher level processors. The subfunction performed by each section is discussed below in order of decreasing level in the hierarchy.

Protocol Handler Section

The protocol handler section resides at the highest level in the functional hierarchy. This section handles serial line communication with the experiment control device and decodes the submitted function codes. The protocol handler is also responsible for converting the ASCII fields of motor label and position value into internal binary representation (and vice versa for returned output information). As mentioned above, this section also generates a video display of the status of the motor controller as a whole, in particular, the real-time display of serial line ASCII characters as they are received and transmitted.

The only component of subfunction-specific hardware present in this section is a DEC DLV-11J quadruple asynchronous serial line port. Two of the ports provided by this module are utilized; the first is connected to the experiment control device, while the second supports input/output from/to a manual controller box⁸ (a special purpose driving device).

The primary function of the FORTRAN main program executing in this section is to support these serial line ports by setting up device interrupt service routines appropriate to the current driving device interaction mode and submode of the motor control unit. These interrupt service routines are responsible for receiving, delimiting, and transmitting transaction lines and raising a flag when a complete request line has been received. The main program decodes the words of the request line and determines whether or not it must call one of its overlays in order to perform the requested function. The FORTRAN main program-overlay set code uses very little of the text conversion and formatting capabilities of the language. The special cases of these operations unique to the motor control unit have been identified and assembly language level subroutines written to perform them. Typically, the logical variable manipulation and subroutine calling features of the FORTRAN language are stressed. These features facilitate setting up and modifying the overall

(complex) control algorithm for a section.

The protocol handler section also maintains the position specification mode parameter of the unit. This parameter is used to keep track of the coordinate systems associated with each motor axis and generate the video display of axis current and target position values. The protocol handler program uses FORTRAN floating point arithmetic routines to reduce all logical experiment mode and physical scale mode coordinates to encoder readings (physical absolute mode). If a submitted function code specifies a motor positioning operation, the protocol handler passes the position parameters, in 24-bit binary integer form, to the next lower section in the hierarchy, the motor driver section.

Motor Driver Section

The motor driver section uses current and target position value information (in 24-bit fixed point binary representation) to compute motor travel values and generate the required number of step pulses for the motor power supplies. Special-purpose hardware present in this section consists of subsections which perform control operations for each individual motor. Each subsection is capable of handling complete motor move operations (with acceleration, deceleration, and backlash takeup) once it has been provided with motor total travel and maximum stepping velocity parameters. These subsections also handle homing operations and readout of limit switch and home switch status.

The subsection consists of an Intel 8085A 8-bit microprocessor and an Intel 8155 multifunction integrated circuit which contains a 256-byte read/write memory and three parallel input/output ports. The subsection also includes a generic 2716 EPROM for program storage and an Advanced Micro Devices 8253-5 programmable counter/timer for generation of the stepping pulse stream. The parallel-line ports accept the limit switch voltage levels as input; limit switch status change also interrupts the subsection processor so that the stepping pulse stream being generated can be stopped. All counter/timer output is optically isolated from the motor translators; all parallel port input is optically isolated from the limit switch power supplies.

Each subsection communicates with the motor driver processor by having a portion of its local RAM memory addressable as an array of external device registers on the Q-buss of that processor.⁹ Structurally, the electronics for four subsections is contained on a single hex-width DEC-compatible printed circuit board.

The motor driver section generates a video display of the motor current and target position values for each motor axis. The positions are exhibited as decimal integers and thus correspond to physical absolute mode coordinates. The display also includes status mnemonics for the axis position limit switches. If a current position value is required for an axis, the motor driver section submits, via the shared memory, a function request to the encoder reader section.

Encoder Reader Section

The encoder reader section occupies the lowest level in the functional hierarchy. This section employs DEC DLV-11J asynchronous serial line port modules, as function-specific hardware, to read out the encoders. The single-character request/response protocol detailed above is used to obtain three-byte encoder readings, formatted according to specific encoder type. The encoder reader section converts these readings into internal binary representation before passing them back up to higher level processors. Encoder readings may be

obtained on demand or in a continuous mode. This section generates a video display of current encoder readings both in decimal integer form and in a binary coded form which corresponds to the actual bit pattern read from the encoder.

Subsystem Driver Routines

Including the Instrumentation Division prototype, six motor control units have been constructed. Five of the units have been assigned to a test or experiment station; all of these stations employ a DEC PDP-11/34 or ISI-11/23 computer, the DEC RSX-11M operating system, and the FORTRAN high-level programming language. Accordingly, a set of FORTRAN-callable subroutines has been written to facilitate driving the motor control unit from this computer system configuration.

Motor control functions are initiated by calling one of the subroutines at the highest level in the set. These subroutines are also written in FORTRAN; the requested function is mnemonically contained in the subroutine name. The FORTRAN subroutines invoke lower level subroutines written in DEC assembly language to fill a line buffer with ASCII-coded words representing the function code and other parameters, typically a motor label and position coordinate, required to completely specify the function. A particular type of RSX-11M execution support software interrupt, the "QIO" or "queue input/output transfer" trap, is invoked to queue up the output/input operation required to transmit/receive transaction lines to/from the motor control unit. These input/output transfer requests are placed in the queue of a serial line physical device which is logically treated by the RSX-11M operating system as an "XL message-oriented communications device". Thus the operating system must have an "XL" device driver built into it in order that the subroutine package be used.

Assembly language level subroutines decode the response and error message lines returned by the unit and return the resulting ASCII strings to higher-level FORTRAN routines. The FORTRAN "DECODE/ENCODE" facilities are used for converting position coordinates from ASCII form to internal binary floating point representation. In particular, the package uses the following standard variables:

IMTRNO, the motor-encoder pair number. This integer variable assumes values from 1 to 15(10);

CURPOS or TGTPOS, the motor-encoder pair current or target position value. Coordinate positions are maintained as floating point numbers within a main program or overlay and converted to ASCII strings appropriate to the motor control unit current position specification mode by the support subroutines. The motor controller almost always operates in "logical experiment" position specification mode when being driven by this package;

LMOVNG, LMVTTC, LMVTHM, LATIGT, LATHOM, LATCWL, LATCCL, LATMNL, LATMXL, LURIGT, and LURHOM, motor encoder pair status flags. These logical variables are returned with .TRUE. values if the motor-encoder pair has the status ("moving", "moving towards target", etc.) indicated by the variable name;

LFTNSF, a logical variable returned with .TRUE. or .FALSE. to reflect the outcome (successful or not) of the requested function;

IERRPT(64), the returned error message. This byte array will contain an error message to be printed if the function requested of the motor control unit is not successful.

Experiment Station Translators,
Motors, and Encoders

The design objective of making the motor control unit compatible with many types of translators and encoders has been realized. The motors, translators, and encoders present at one test station and three experiment stations are briefly listed following. The Small-Angle X-ray Scattering Station being constructed at Brookhaven's Biology Department is to have thirteen motions automated. Table I gives details of the mechanical positioning hardware at this station. The sixth unit is a Light Source spare and has not been assigned to an experiment.

(3) X-ray topography, NSLS and State University of New York at Stony Brook. Twelve motions are being automated with SLO-SYN type ST103 and ST105 translators. These translators drive SLO-SYN series M061, M062, M091 and M092 motors at 3000 and 5000 steps/second, respectively. The motions have their positions encoded with BEI incremental encoders.

(4) Test facility, Instrumentation Division. Two axes are used for testing the motor control unit. The first employs a SLO-SYN TBM105-6309 translator to drive a SLO-SYN M063-FC09 motor. The axis is coordinated with a BEI 5V241 12-bit absolute encoder. The second axis utilizes a SLO-SYN STM1800 translator logic card with external power supplies, a SLO-SYN SS25-1002 motor, and

MOTION LABEL	MOTION DESCRIPTION	TRANSLATOR TYPE†	MOTOR TYPE	ENCODER TYPE‡	COMMENTS
A	Beam follower, horizontal direction, moves entire spectrometer in order to follow incident beam	ST103	M092-FC09-E	5VN232BGL	
B	Beam follower, vertical direction, moves entire beam spectrometer in order to follow incident beam	TM600U	M172-FD306	5VN232BGL	High-power stepping motor, 0.8 HP
C	Direction along spectrometer arm, detector to sample spacing	TBM105-9214	M092-FD310	5VN232BGL	
D	Direction perpendicular to spectrometer arm, detector scan motion	TBM105-6209	M062-FC09-E	5VN232BGL	
E	Spectrometer carriage travel, spectrometer distance from last optical output element	TBM105-9322	M093-FD301-E	5VN232BGL	
F	Sample position, vertical direction	TBM105-6209	M062-FC09-E	5VN232BGL	
α	Spectrometer arm, detector rotation about horizontal axis	TBM105-1230	M112-FJ326	5V682HBC	Encoder resolution is 16 bits
ω	Sample rotation about horizontal axis	ST103	M093-FC11	ITEK 16/23 MICRO	Micro size encoder to fit inside rotary table
λ	Monochromator, determines wavelength	ST101	M062-FD03	NONE	
ϕ	Sample rotation, $\perp$ to χ	ST101	M062-FD03	NONE	
χ	Sample rotation, $\perp$ to ω	ST101	M062-FD03	NONE	
Attenuator	Rotation of attenuator disks wheel into incident beam	TBM105-6209	M062-FC09	NONE	
Sample changer	Rotation of samples wheel	Included	Included	NONE	Translator and motor built into sample changer

TABLE I. STEPPING MOTORS AND ENCODERS FOR POSITIONING DEVICES AT
BNL BIOLOGY DEPARTMENT SMALL-ANGLE X-RAY SCATTERING STATION

†All translators are Superior Electric SLO-SYN models unless otherwise noted.

‡All encoders are Baldwin Electronics, Inc. models unless otherwise noted.

(1) X-ray crystallography, BNL Chemistry Department. This unit drives six axes of a Huber goniostat by means of six Superior Electric SLO-SYN¹² translators and motors. All translators are 5000 step/second open chassis type, series TBM105. The motors are types M061-FD08 and M091-FD09. No encoders are used; the encoder serializer modules count motor step pulses. This unit is fully operational.

(2) Materials analysis using EXAFS, BNL Department of Energy and Environment. The motor control unit is being connected to six MESUR-MATIC¹³ series 340 motors, two of which are driven by Model 3640A Microstepper translators. The motors are normally 200 steps/revolution; the microstepping translators multiply this resolution to 1600 steps/revolution. This station utilizes BEI¹⁴ series 5VN270 incremental encoders.

a BEI 5V270BZ 10-bit incremental encoder. The test facility has been in operation since May of 1981.

Future Work

Three improvements to the motor control unit are being considered. The first would modify the output of the motor driver subsections so that, as an option, a stepping pulse stream and clockwise/counterclockwise direction voltage level would be produced instead of two pulse streams. This would add other types of translators to the list of types with which the control unit is directly compatible. The second improvement consists of adding a control line from the subsections which would turn translator power off when the motor is not stepping (assuming that the driven load will not back-drive the motor). The third improvement is more complex and will

be undertaken if high-speed multiple-axes scans are required. This change will involve additional software in all three sections of the motor control unit and will essentially speed up the propagation of a step command received over the serial line through the protocol handler section to the motor driver section.

References

1. American Standard Code for Information Interchange.
2. EIA Standard RS-232-C, "Interface Between Data Terminal Equipment and Data Communication Equipment Employing Serial Binary Data Interchange", Electronic Industries Association, 2001 Eye Street, N. W., Washington, DC 20006 (1969).
3. (FOR)mula (TRAN)slation, a standard compiled high-level programming language originally developed by the International Business Machines Corporation.
4. National Electronic Equipment Manufacturers Association.
5. EIA Standard RS-422, "Electrical Characteristics of Balanced Voltage Digital Interface Circuits".
6. D. G. Dimmiller and W. H. Hardy, II, IEEE Trans. Nuc. Sci. NS-24, No. 1, (1977) 469.
7. J. L. Alberi, "Multiport Memory Subsystem II", BNL 29056 (1978).
8. J. L. Alberi and F. W. Stubblefield, IEEE Trans. Nuc. Sci. NS-28, No. 1, (1981) 351.
9. J. L. Alberi, IEEE Trans. Nuc. Sci. NS-29, No. 1, (1982) 84.
10. J. L. Alberi and F. W. Stubblefield, IEEE Trans. Nuc. Sci. NS-29, No. 1, (1982) 49.
11. F. W. Stubblefield and J. L. Alberi, "An Experiment Control and Data Acquisition System for the Biology Small-Angle X-ray Scattering Station at the National Synchrotron Light Source", these Proceedings.
12. SLO-SYN is a registered trademark of the Superior Electric Company, Bristol, Connecticut 06010.
13. MESUR-MATIC Electronics Corp., 50 Grove Street, Salem, Massachusetts 01970.
14. BEI Electronics, Inc., Digital Products Division, 1101 McAlmont Street, Little Rock, Arkansas 72203.