OPTICAL VIDEODISC PLAYER
MODELS PR-7820-1 AND PR-7820-2
PROGRAMMING REFERENCE GUIDE

LaserDisc™

PIONEER®

U3

INTERACTIVE PROGRAMMING... The most exciting and versatile communications medium of the future is already a reality. It is the optical videodisc system which puts audio/visual programming in the hands of viewers. For the first time, audiences are no longer limited to passive watching of television. Now they can personalize the programming. The can get involved.

The term for this capability is INTERACTIVE PROGRAMMING. It opens the way to a new kind of communications: high quality pictures and sound tailored expressly for viewers to interact with in businesses, in schools, in homes. The result is communications with a new dimension in persuasion and influence... with a new ability to educate and entertain.

INTERACTIVE PROGRAMMING. For program producers, it represents the audio-visual capability—and the challenge—of the 1980's.

TABLE OF CONTENTS

INTRODUCTION

For commercial, industrial and institutional users of the optical videodisc system from Pioneer Video, Inc. (PVI), the stage is set for revolutionary change in media programming. By combining the latest in television and sound techniques with microcomputer technology, the system allows programmers to tailor audio-visual messages in unique ways:

First, program producers can combine sound and pictures in ways that viewers have not heard or seen before. The videodisc carries the finest quality television signal and high quality sterophonic sound.

Second, producers can create audio-visual programs not just for viewers to watch, but to interact with, in much the same way that a computer user interacts with a data file. The interactive videodisc program can have as many variations as it has viewers.

This means that a single, pre-recorded videodisc program can be resequenced and redirected to serve a variety of audiences. It can be designed to educate at each viewer's learning rate, for example, or it can sell at each consumer's persuasion level. Properly formatted, the optical videodisc brings outstanding new capability in business, in industrial and institutional training, in entertainment, and in merchandising. In short, the videodisc offers a new communications horizon for program producers.

The components of the optical videodisc system are the computer-based industrial videodisc player, PR-7820, and its hand-held Remote Control Unit (RCU) or "keypad." By using the RCU's command buttons, the programmer can control the player's microcomputer, redirecting and resequencing the play of an original videodisc program.

The PVI optical videodisc system is versatile and easy to use. However, for program producers to fully master the rich menu of functions, a data processing discipline should be coupled with creative insight. This manual is designed for the programmer involved in developing application support code for the PR-7820 using the RCU. Information is presented on the player's computer structure, software support, and the commands to program videodisc displays.

For customers who are not staffed with people possessing adequate programming skills, Pioneer Video, Inc. also offers a full range of program development services. Please contact your Pioneer Video, Inc. Marketing Representative for details.

Note: This publication addresses digital programming for videodisc players Models PR-7820-1 and PR-7820-2. A Programming Reference Guide for Model PR-7820-3 is available from Pioneer Video, Inc. as publication TP103.

REMOTE CONTROL UNIT OPERATION

KEYPAD COMMAND BUTTONS

STOP stops the player for a specific period of time (expressed in tenths of seconds) and then continues. Command appears as "WAIT" on the display screen.
EXAMPLE: 30 WAIT stops the program for three seconds.

INPUT prepares the player to respond to keypad input (as in answering a multiple-choice question).
EXAMPLE: 3 INPUT prepares the program for four possible inputs ("0," "1" and "2" and "other").

SEARCH commands a search for a specific frame.
EXAMPLE: 4500 SEARCH sends the player to frame 4500.

AUTOSTOP commands the player to play until a specific frame is reached.
EXAMPLE: 2500 AUTOSTOP plays until frame 2500.

PROGRAM sets the memory location at 0000. If a memory address precedes the command entry, the player will index to that address.
EXAMPLE: 106 PROGRAM puts the player in program mode, starting at memory location 0106.

END takes the player out of program mode.

DEC REG decrements or decreases the value of a specific register by one with each play-through.
EXAMPLE: 6 DEC REG decreases the value of Register 6 by one with each play-through.

SLOW FORWARD or SLOW REVERSE puts the player in slow motion.
EXAMPLE: 6000 SLOW FWD puts the player in slow motion until it reaches frame 6000.

STEP FORWARD or STEP REVERSE steps the player forward or backward one frame.
EXAMPLE: STEP FWD advances the player one frame.
RECALL selects a register number.
EXAMPLE: 1 RECALL selects Register 1.

HALT takes the player out of AUTOMATIC control mode and returns it to manual mode.
EXAMPLE: HALT is last command of a program.

STORE establishes a value for a selected register.
EXAMPLE: 4 STORE sets the selected register to a value of four.

BRANCH redirects the program to another memory location.
EXAMPLE: 76 BRANCH sends the program to memory location 0076.

AUDIO 1 and AUDIO 2 control respective sound tracks on the videodisc. Even number precedents turn sound off; odd number precedents turn sound on.
EXAMPLE: 1 AUDIO 2 turns on track 2.

RUN puts player in AUTOMATIC mode for operation according to stored instructions. RUN also turns audio tracks on and frame display off.

DISPLAY controls on/off display of videodisc frame numbers.

REJECT halts program execution, returns videodisc to load position and stops disc rotation. Normally used in MANUAL operation to allow removal of videodisc from player.

PLAY instructs play of a video sequence until the program stops the player.

RANDOM ACCESS MEMORY

The PR-7820's microprocessor contains the basic operating system of the player and also provides one kilobyte (1024 bytes) of random access memory (RAM). The 1024 bytes of address space contain both the application program's commands and registers—a coexistence that allows RAM to be used for program storage ***and*** as a group of registers that the program can reference. Depending on how the 1024 bytes are used, they are addressed differently. Although a program can use all 1024 bytes of RAM, a program dump on the videodisc contains exactly 1022 bytes, not 1024. See Appendix D for an explanation of videodisc programming.

As memory, programs begin at location 0 (or some other user-specified address) and proceed one byte at a time to locaton 1023. Registers, on the other hand, occupy RAM in two-byte increments; they begin at location 1022 and proceed to location 0. There are 512 registers, addressed 0 through 511. Register 0 occupies memory addresses 1022 and 1023 while Register 511 occupies 0 and 1. The programmer should understand the use of memory because registers and stored programs can overlay one another.

Program data is stored in a coded format (see Appendix A) with each digit and command occupying one byte. Register data is stored in a 16-bit binary format. For example, the value 1536 stored within a program occupies four bytes having the hexadecimal format OFAF4F6F. Stored in a register, 1536 occupies two bytes and equals 0600.

ERASABLE PROGRAMMABLE READ-ONLY MEMORY

The microprocessor also contains erasable programmable read-only memory (EPROM), which controls and interprets the user's program, handles input, and controls videodisc playback. In effect, the interpreter is a built-in program that decodes user commands keyed in with the hand-held Remote Control Unit (RCU) while the PR-7820 is operating in either MANUAL or AUTOMATIC mode. No provision is made to use the microprocessor instruction set directly.

PROGRAM INTERPRETATION

A series of commands combine to create a program. In memory, a program resembles one long character string that is processed by the interpreter. Beginning at the location specified on the RUN command, the interpreter inspects each element for a command code. When the command code is located, an internal command processor performs the desired function. Characters preceding the command code are either the numeric address field or are ignored. The interpreter inspects the next character in the string following each action of the command processor.

For example the commands

1000 SEARCH
2000 AUTOSTOP

become the following hexadecimal codes: 0F3F3F3FF78F3F3FF3. Starting with the first byte (0F), the interpreter scans the character string until the command code for SEARCH, F7, is found. The SEARCH command processor, using the preceded code digits as an address, instructs the player to position the videodisc at frame 1000.

In AUTOMATIC mode the player is interpreting stored programs, and all commands except PLAY are executed in order before new interpretations can begin. Most commands entered from the RCU at this point would interrupt program interpretation and return the player to MANUAL mode. From MANUAL, a RUN command must be given to return the player to AUTOMATIC, where interpretation will resume at the address specified (or implied) by the RUN command. Thus when the videodisc is being moved or played, interpretive execution is usually suspended until the player enters freeze frame.

In the example of SEARCH and AUTOSTOP command usage above, any command following AUTOSTOP is not executed until frame 2000 is played. An exception to this procedure is the PLAY command: it instructs the player to play a video sequence until the program stops the player. All other play-type commands generate an ending frame address.

UTILIZING REGISTERS

In addition to executing individual commands, the interpreter assists in utilizing registers. For example, two commands, SEARCH and AUTOSTOP, require target frame numbers. If a frame number is not specified, the interpreter obtains it from the contents of the active register—the only one of 512 possible registers whose contents can be used or modified at any given time. By varying the contents of the active register, or by designing a new active register, the programmer can execute a series of instructions in a loop and access different frames on the videodisc. By using active registers, the programmer may save memory space. This is because addresses in registers are in binary, rather than coded, format. However, program space may be required to load registers (see the STORE command).

Initially, Register 0 is the active register and contains the value 1. Later any register can be designated as active by specifying it as the operand field of a RECALL command. The interpreter activates successive registers whenever one of a certain group of commands is executed, even if the command does not use the register. The group is STORE, SEARCH and AUTOSTOP.

By using the successive registers with the start and stop frame numbers for several scenes, and using one RECALL command to activate the register at the beginning of the group, the interpreter will automatically activate successive registers each time a SEARCH or AUTOSTOP command is given. This eliminates the need for several RECALL commands. Register use with the STORE command works in similar fashion. These programming techniques can help reduce program size.

SECTION 3/COMMANDS

Commands are functional instructions entered into the PR-7820 via the keypad of the RCU to provide user control and /or interaction of videodisc contents. (See Appendix D for notes on programs loaded from the videodisc and via the PR-7820's External Computer Port.) Depending on the operational mode of the player, commands may be acted upon immediately or stored as part of a program to be run later. In MANUAL mode, for example, the player reacts to the user's commands. In PROGRAM mode, entries are stored in the player's microprocessor memory. When the RUN command is given, the player is placed in the AUTOMATIC mode and operates according to stored instructions.

Commands can be entered at any time. Some commands may have different functions depending on the mode of player operation. Commands required to develop and execute an application program fall into two categories:

- Program Input
- Program Execution

The format of these command categories is the same: usually a numeric operand followed by a command. The operand is a numerical value representing a frame number, a branch address, a register number, or a time designator. An operand is usually specified prior to the command, but it may be implied. An implied operand resides in the active register or has a default value.

SECTION 3/COMMANDS

PROGRAM INPUT COMMANDS

COMMAND: PROGRAM
FORMAT: (Optional Address) PROGRAM
FUNCTION: Places the player in the PROGRAM mode. Applications can be entered, modified or reviewed beginning at the address specified, or at 0 if the address is omitted. Addresses over 1023 are taken modulo 1024 (1024 equivalent to 0).

Entries following the PROGRAM command are stored in successive memory locations, with each digit occupying one byte. The video screen will display the memory address and the location's contents. An END command will terminate the player's PROGRAM mode.

Multiple applications can exist in the player's memory at the same time. Each can be entered using the PROGRAM/END command sequence, and each may have a separate starting address on the PROGRAM command.

A group of commands starting at address NNN can be reviewed by entering NNN PROGRAM. Successive pressing of the PROGRAM button displays memory contents one byte at a time. A review is terminated by the END command.

Stored information can be edited or updated during a program review by substituting a new command to replace the byte being reviewed. The user must be careful not to overlay needed commands or leave unnecessary data, since command lengths can vary from one to six bytes. For complex modifications, the programmer should either use a patch area or rewrite the section of code.

COMMAND: END
FORMAT: END
FUNCTION: Terminates the PROGRAM mode and returns the player to the MANUAL mode. The command is used with the PROGRAM command for application entry, modification or review.

COMMAND: RUN
FORMAT: (Optional Address) RUN
FUNCTION: Instructs the microprocessor to start an application program at a particular memory address, if specified, or at location 0. The RUN command transfers the player from MANUAL to AUTOMATIC mode. The command turns audio tracks 1 and 2 on, turns the frame number display off, and makes Register 1 the active register.

When the player is in the PROGRAM mode, the RUN button becomes the branch command.

The HALT command overrides a program and returns the player from AUTOMATIC to MANUAL. The player may continue in motion, according to the latest program instruction. To stop the play action, press the STOP command.

COMMAND: HALT
FORMAT: HALT
FUNCTION: The HALT command used in a program returns the player from AUTOMATIC to MANUAL. The player may continue in motion in accordance with its latest instruction. The only fully reliable way for a user-viewer to interrupt a program and return to MANUAL mode is to enter HALT (numeric key entry) HALT—for example, HALT 6 HALT.

PROGRAM EXECUTION COMMANDS

The following commands can be entered with the RCU when the player is in the PROGRAM mode. All are useful in developing applications:

AUDIO 1, AUDIO 2
AUTOSTOP
BRANCH
DECREMENT REGISTER
FRAME DISPLAY
INPUT
RECALL
REJECT
SEARCH
SLOW FORWARD/REVERSE
STEP FORWARD/REVERSE
STOP (WAIT)
STORE

COMMAND: AUDIO 1, AUDIO 2
FORMAT: (Optional Number) AUDIO 1
(Optional Number) AUDIO 2
FUNCTION: AUDIO 1 controls audio track 1, and AUDIO 2 controls audio track 2. When the command is preceded by an even number (i.e., 0, 2, 4) sound for that particular track is turned off. An odd preceding number (1 or 3) turns sound on. If there is no preceding number, the AUDIO command acts as a toggle switch to turn sound off or on. The RUN command turns both audio tracks on. During AUTOMATIC mode, the user can still toggle between AUDIO 1 and AUDIO 2 at any time, undetected by the program.

EXAMPLE: Nine commands will program play from frame 1000 to frame 300 with audio track 1 off and track 2 on, and from frame 3000 to frame 5000 with audio track 1 on and audio 2 off. Assume that Registers 20, 21 and 22 contain values 1000, 3000 and 5000, respectively. Here is how the program is entered:

OPERAND	COMMAND	PURPOSE
20	RECALL	Activates Register 30
0	AUDIO 1	Turns off track 1
1	AUDIO 2	Turns on track 2
	SEARCH	Displays frame 1000
	AUTOSTOP	Tells player to play to frame 3000
	AUDIO 1	Turns track 1 on unless the user has toggled AUDIO 1.
	AUDIO 2	Turns track 2 off unless the user has toggled AUDIO 2.
	AUTOSTOP	Tells player to play to frame 5000
	HALT	Returns player to manual mode

COMMAND: AUTOSTOP
FORMAT: (Optional Frame Number) AUTOSTOP
FUNCTION: Starts normal speed play of a scene from the current frame number to the frame number associated with AUTOSTOP. When the last frame is reached it is displayed in a freeze-frame mode. If no frame address is specified, the frame number is taken from the active register. AUTOSTOP is usually used after SEARCH or another AUTOSTOP command. The active register is always deactivated and the next sequential register is activated, even if the frame address is specified. If the frame address is less than the current frame address, the player will position itself at the frame number associated with AUTOSTOP, possibly displaying other scenes until the desired frame is reached.
EXAMPLE: Six commands are required to play the first 1000 frames of a sequence, delay five seconds, and play the next 1500. Assume that Registers 20, 21 and 22 contain values of 2000, 3000 and 4500, respectively. Here is how the program is entered:

OPERAND	COMMAND	PURPOSE
20	RECALL	Activates Register 20
	SEARCH	Displays frame 2000
	AUTOSTOP	Tells player to play to frame 3000
50	STOP	Tells player to wait five seconds
	AUTOSTOP	Tells player to play to frame 4500
	HALT	Returns player to manual mode

COMMAND: BRANCH
FORMAT: (Optional Address) BRANCH
FUNCTION: Alters the sequential flow of command execution by directing the interpreter to take its next data from the specified address. Sequential interpretation of the program resumes from that new location.

Branches are generally used under the following conditions:

Loop Control—following a DEC REG command

Path Selection—with a series of BRANCH commands following each input command, and where each branch corresponds to a user input.

Direct Branch—following a sequence of commands to transfer control to another location. If the BRANCH address is not present, 0 is assumed.

EXAMPLE:

OPERAND	COMMAND	PURPOSE
	BRANCH	Transfers control to Location 0
256	BRANCH	Transfers control to location 256
0	BRANCH	Transfers control to location 0
0613	BRANCH	Transfers control to location 613

COMMAND: DECREMENT REGISTER
FORMAT: (Register Number) DEC REG
FUNCTION: Decreases the contents of the specified register by one. DEC REG is often followed by one BRANCH command to provide loop control. The register previously loaded with a positive number is reduced by one when DEC REG is executed. If the result is greater than 0, the next sequential command is executed; when equal to 0 the interpreter skips all commands until one branch command is passed, then executes the next. There are no negative numbers. A zero decremented remains a zero. (Caution: a DEC REG with no operand should be avoided.) DEC REG does not update the active register. The commands following DEC REG do not have to be branches.
EXAMPLE: Assuming that registers 21 through 30 contain different frame numbers, the following code will play five scenes and then halt.

LOCATION	OPERAND	COMMAND	PURPOSE
500	20	RECALL	Activates register 20
503	5	STORE	Sets loop counter to 5
505		SEARCH	
506		AUTOSTOP	
507	20	DEC REG	
510	505	BRANCH	Greater than 0 – play next scene
514		HALT	Terminates program

COMMAND: DISPLAY
FORMAT: (Optional Number) DISPLAY
FUNCTION: Controls the display of the videodisc frame number. An even number operand (i.e., 0) turns the frame number off, while an odd number operand (1) turns it on. The DISPLAY command used without an operand acts as a toggle switch; it will turn the frame display on if it is off, and off if it is on. The run command turns the display off. The user may toggle the display at any time, undetected by the program.
EXAMPLE: The following code ensures frame numbers will be displayed only while the first sequence is playing:

OPERAND	COMMAND	PURPOSE
1	DISPLAY	Turns on frame number display
1000	SEARCH	Positions player at frame 1000 with display on
2000	AUTOSTOP	Plays to frame 2000 with display on
	DISPLAY	Turns off frame number display
3000	AUTOSTOP	Plays to frame 3000 with display off
	HALT	Terminates program

COMMAND: INPUT
FORMAT (Integer 1 to 9) INPUT
FUNCTION: Allows user interactivity with the player operating in AUTOMATIC mode by permitting entry of a numeric digit. The digit entry becomes an index to a list of BRANCH commands following INPUT and directs the flow of the application. Up to ten BRANCH sequences may follow INPUT where the last one is the default entry.

The integer designation (1-9) in the operand field is the total number of allowed responses not counting the "other" response. Thus when coding an INPUT for three expected entries (0, 1 and 2) and one "other" category (3 to 9), the numeral 3 is the integer operand. Following INPUT there should be three groups of commands, each ending with a BRANCH command.

To execute the BRANCH sequences, the user enters 0 to select the first group; 1 for the second group; and 2 for the third group. Any "other" number (3 through 9) causes all three groups to be skipped.

EXAMPLE 1: The following sequence of commands will display frame 1000 and accept user input for three anticipated entries and the "other" entry:

LOCATION	OPERAND	PURPOSE
100	1000	SEARCH
105	3	INPUT
107	0132	BRANCH
112	138	BRANCH
116	144	BRANCH
120	5000	SEARCH
125	30	STOP
128	100	BRANCH
132	1500	AUTOSTOP
137		HALT
138	2000	AUTOSTOP
143		HALT
144	2500	AUTOSTOP
149		HALT

If the user enters 0 in the above example, frames 1000 to 1500 are played. A 1 entry plays frames 1000 through 2000; a 3 entry plays frames 1000 through 2500. The "other" entry causes frame 5000 to be played for three seconds, then a return to frame 1000 and the INPUT command. Note that the first entry after an input command should begin with a zero. But SEARCH has a meaning different than 0 SEARCH, and STORE is different than 0 STORE.

EXAMPLE 2: Another way of writing the same thing showing groups of commands after INPUT:

LOCATION	OPERAND	COMMAND
100	1000	SEARCH
105	3	INPUT
107	01500	AUTOSTOP
113		HALT
114		BRANCH
115	2000	AUTOSTOP
120		HALT
121		BRANCH
122	2500	AUTOSTOP
127	143	BRANCH
131	5000	SEARCH
136	30	STOP
139	100	BRANCH
143		HALT

COMMAND: RECALL
FORMAT: (Optional Register Number) RECALL
FUNCTION: Activates the specified register. When the operand is omitted, one of two things will happen: (1) If the current-register pointer has not been updated since the last RECALL command, RECALL will activate the next register, or (2) If the pointer has been updated—with SEARCH, AUTOSTOP or STORE commands—it will activate the current register.

COMMAND: REJECT
FORMAT: REJECT
FUNCTION: Halts execution of the application program, returns the videodisc to the load position and stops disc rotation.

This command is given to allow removal of the videodisc. REJECT can be entered while the player is in the PROGRAM mode; however, this will also reject the videodisc. The REJECT command will still be stored in memory.

COMMAND: SEARCH
FORMAT: (Optional Frame Number) SEARCH
FUNCTION: Positions the player at a frame number specified in the operand or, if no operand is specified, a frame number that is contained in the active register. When located, the designated frame is displayed in freeze-frame mode.

Search time is a complicated function of distance moved and player position. Therefore frame access times may vary. If the SEARCH command fails to position the player at the designated frame, the player will continue the search. Continued failure will cause the player to seek out the next higher frame number, and repeat the process if necessary. A SEARCH command specifying a frame address higher than the last address on the videodisc may result in videodisc REJECT.

EXAMPLE: To display frame 1500 using an operand, the programmer simply enters 1500 SEARCH.

To use the active register to display frame 1500, assuming frame 1500 is in Register 20, the steps are:

OPERAND	COMMAND	PURPOSE
20	RECALL	Activates Register 20
	SEARCH	Displays 1500

COMMAND: SLOW FORWARD/REVERSE
FORMAT: (Optional Frame Number) SLOW FWD/REV
FUNCTION: Plays frames forward or backward until the specified frame number is reached. Playing speeds range from "still" frames to normal 30 frames-per-second, depending on the setting of the speed-control slide on the front of the player.

The SLOW FORWARD/REVERSE commands suppress audio and, on completion, leave the player in freeze frame. For practical applications, a specific frame number in the operand is required.

EXAMPLE: Beginning at frame 26000, play backward in slow motion 1000 frames:

OPERAND	COMMAND	PURPOSE
26000	SEARCH	Displays frame 26000
25000	SLOW REV	Plays backward 1000 frames
	HALT	

COMMAND: STEP FORWARD/REVERSE
FORMAT: STEP FWD/REV
FUNCTION: Moves the player forward or backward one frame displayed in freeze mode. STEP FORWARD/REVERSE can be used to view a series of still frames or to provide programmed slow motion effects.
EXAMPLE: Starting at the current frame, a program for viewing the next 10 frames as a series of stills displayed eight seconds each can be written:

LOCATION	OPERAND	COMMAND
0	20	RECALL
3	10	STORE
6		STEP FWD
7	80	STOP (WAIT)
10	20	DEC REG
13	6	BRANCH
15		HALT

COMMAND: STOP (WAIT)
FORMAT: (Integer) STOP
FUNCTION: The STOP button produces a command that is displayed as "WAIT" when entered during PROGRAM mode. The command causes a freeze-frame pause for a specified period of time. The integer specifies the time in tenths of seconds.
EXAMPLE: A programmed freeze for 5.2 seconds is coded 52 STOP

COMMAND: STORE
FORMAT: (Optional Decimal Number) STORE
FUNCTION: Places a value in the active register. The value stored is either the decimal number in the operand field or, if there is no operand, the current frame number. This command allows the program to remember where it is playing, go to another sequence, and later resume playing where it left off. The STORE function is useful following a PLAY command or some STEP FWD/REV commands.

Completion of the STORE function automatically activates the next register. Values over 65000 may not be utilized correctly. In loading a program from the videodisc, the programmer should be aware that registers may be preloaded directly without use of STORE commands (see Appendix D).
EXAMPLE: Placing the current frame number in Register 20 and the value 5000 in Register 21 is achieved with the following code:

OPERAND	COMMAND	PURPOSE
20	RECALL	Activates Register 20
	STORE	Stores current frame number and activates Register 21
5000	STORE	Stores value in Register 21. activates Register 22

EXCEPTIONAL PROGRAM COMMANDS

Three exceptional program commands—PLAY, NULL and LOAD—cannot be entered directly by a button on the RCU. However, knowledge of special techniques allows their use. PLAY is a particularly important command that gives substantially different application capabilities.

COMMAND: PLAY
FUNCTION: The PLAY command tells the player to present video material, starting at the current frame position, until the play mode is changed.

The PLAY command code can be inserted into the program by using the XXX RECALL, 253 STORE command sequence, where XXX is the register equivalent of two memory addresses and 253 is the decimal equivalent of the command codes for NULL, PLAY (00FD) (see Appendix A). Other command combinations can be used when setting up PLAY. For example, a SEARCH, PLAY sequence can be constructed by specifying 63485 (F7FD) in the STORE command.

When the PLAY command is interpreted from a stored program in AUTOMATIC mode, the video sequence commences and the program continues being interpreted. This capability sets PLAY apart from the other play-type commands and provides the programmer another dimension of flexibility. PLAY is terminated when one of the following commands is executed: AUTOSTOP, REJECT, SEARCH, SLOW FORWARD/REVERSE, STEP FORWARD/REVERSE, or STOP. The HALT command ends the program execution but does not terminate PLAY.

Essentially, PLAY provides the capability to present video material on a time basis rather than a frame basis. EXAMPLE 1 allows the user to select one of three lengths of time for which a scene is to be shown. EXAMPLE 2 on page 16 lets the user decide on a real-time basis when to terminate the playing of a scene. Here, an INPUT command follows PLAY and as soon as a key on the RCU is pressed, a command is executed that terminates PLAY but continues program execution.

EXAMPLE 1: This code permits user control of the time that a scene will play. Assume each pass through the DEC REG, BRANCH loop takes about 0.4 seconds:

LOCATION	OPERAND	COMMAND	PURPOSE
0	498	RECALL	Points to locations 26, 27
4	253	STORE	Inserts NULL, PLAY commands in program
8	1	RECALL	Activates counter register
10	2	INPUT	Waits for user response
12	037	BRANCH	0-play for 20 times through loop
16	43	BRANCH	1-play for 35 times through loop
19	50	STORE	2-play for 50 times through loop
22	1500	SEARCH	Displays frame 1500
27	0	PLAY	
29	1	DEC REG	Time loop for play command
31	29	BRANCH	
34	49	BRANCH	Ends play
37	20	STORE	Sets loop counter to 20
40	22	BRANCH	Play
43	35	STORE	Sets loop counter to 35
46	22	BRANCH	Play
49	10	STOP	Terminates play command
		•	
		•	
		•	
		HALT	End program

EXAMPLE 2: User termination of a video sequence via an RCU entry. Assume Register 2 contains frame number 1500:

LOCATION	OPERAND	COMMAND	PURPOSE
0	505	RECALL	Points to locations 12, 13
4	63485	STORE	Inserts SEARCH, PLAY command sequence
10	2	RECALL	Activates register 2;
12	00		SEARCH, PLAY executed
14	1	INPUT	Waits for user response
16	020	BRANCH	
20	10	STOP	Terminates play
23	1	RECALL	Activates register 1
25		STORE	Saves current frame number
		•	
		•	
		•	
	1	RECALL	Accesses saved frame number
		SEARCH	Resumes play from point of interruption; up to 7000
	7000	AUTOSTOP	
		•	
		•	
		•	
		HALT	

COMMAND: NULL
FUNCTION: This is any one-byte code that is interpreted to mean "do nothing," or "continue interpreting the next sequential byte."

The RCU does not contain a NULL button, but the code 00 is treated by the interpreter as a NULL command. Leading (non-significant) zeros on operands will generally serve as bytes that "do nothing," and they are easily entered via the RCU. The zeros (or NULLs) are useful to erase an erroneous command from an already existing program or to save space for later modifications. Placing a NULL under program control may also be useful to "wipe out" or change some existing command.

EXAMPLE: A NULL, NULL command could be placed at XXX, the register number corresponding to the program locations into which the NULL commands are to be placed. The following commands would store the NULL, NULL:

```
XXX     RECALL
0       STORE
```

EXAMPLE: Place the command sequence NULL, RECALL at program location 40 and 41, which is equivalent to Register 491. The hex equivalent for the command sequence NULL, RECALL is 007F, or 127 decimal (see Appendix A). Thus the following command can be used:

```
491     RECALL
127     STORE
```

Similarly any command pair can be placed in an even-odd byte pair or register.

COMMAND: LOAD
FUNCTION: This command is used after the videodisc has been positioned on a frame that has a program dump lead-in on AUDIO 2 track. If the interpreter finds valid lead-in data, it attempts to load the following 1022 bytes of data, which will span about 50 frames of AUDIO 2. The data is recorded at 5000 bits (625 bytes) per second. Including lead-in data, approximately 150 audio frames are needed to hold the whole program dump. After a successful LOAD, the interpreter will take its next commands starting at location 0 (see Appendix D).

If a LOAD is unsuccessful, the player will usually repeat the load process. Sometimes failure to load puts the player into PLAY in MANUAL mode.

Only 1022 of 1024 memory bytes are loaded, and the "missing" two bytes are not contained in the program dump at all, so Register 0 is not overlaid by a LOAD. This allows one register of communication between different loaded programs.

The hex code for LOAD is CC, which is 204 decimal. Thus a NULL, LOAD could be put in locations 2 and 3 with these commands:

```
510     RECALL
204     STORE
```

A LOAD, HALT is represented by decimal 52415.

EXAMPLE: The lead-in **starts before** frame 150, and the data starts **after** frame 150.

```
150     SEARCH
        LOAD
```

An application program may require subroutines to handle repetitive command sequences. The interpreter permits the existence of subroutines, but their use is complicated and one method requires a double branching process. For example: The calling routine sets up a BRANCH command to the desired return address, (say, 200) at some fixed, preselected location (say 0). Thus, using the above example, the subroutine exit branches to the command at location 0 which in turn branches to 200.

Setting up the branch to location 200 requires the understanding of the command code structure and the relationship of registers to memory addresses: for example, Register 511 is memory locations 0 and 1 and register 0 is memory locations 1022 and 1023. The command code for a branch location to 200 is 8F3F3FCF. Thus memory locations 0 and 1 must be set to 8F3F while locations 2 and 3 must equal 3FCF.

The decimal equivalent of 8F3F is 36671 while the decimal equivalent of 3FCF is 16335. The command sequence in the calling routine required to set up at location 0 a branch to 200 is:

510	RECALL
16335	STORE '0 BRANCH' at locations 2 and 3
36671	STORE '20' at locations 0 and 1.

The exit command in the subroutine is:

0	BRANCH

SECTION 5/SAMPLE PROGRAM

The flowchart in Figure1 illustrates some of the program capabilities of the PR-7820 player. The program starts by playing a motion sequence from frame 1000 to frame 3000, then shows a question at still frame 3500. The user has the choice of three responses:

- Pressing key 1 shows a motion sequence in slow motion, from frame 4000 to frame 6000. When completed, the question at frame 3500 is shown again.
- Pressing key 2 shows three timed stills at frames 7000, 7010, and 7020. Each still remains displayed for 3 seconds (30 tenth seconds) before automatically going to the next still. When all three timed stills have been displayed, the question at frame 3500 is shown again.
- Pressing key 3 shows three step stills at frames 8000, 8001, and 8002. Each still remains displayed until the viewer presses key 1 to step forward to the next still. When key 1 is pressed while the third step still is displayed, the question at frame 3500 is shown again.

Key 0 and keys 4-9 are ignored. Figure 2 explains the RCU command sequences for the sample program.

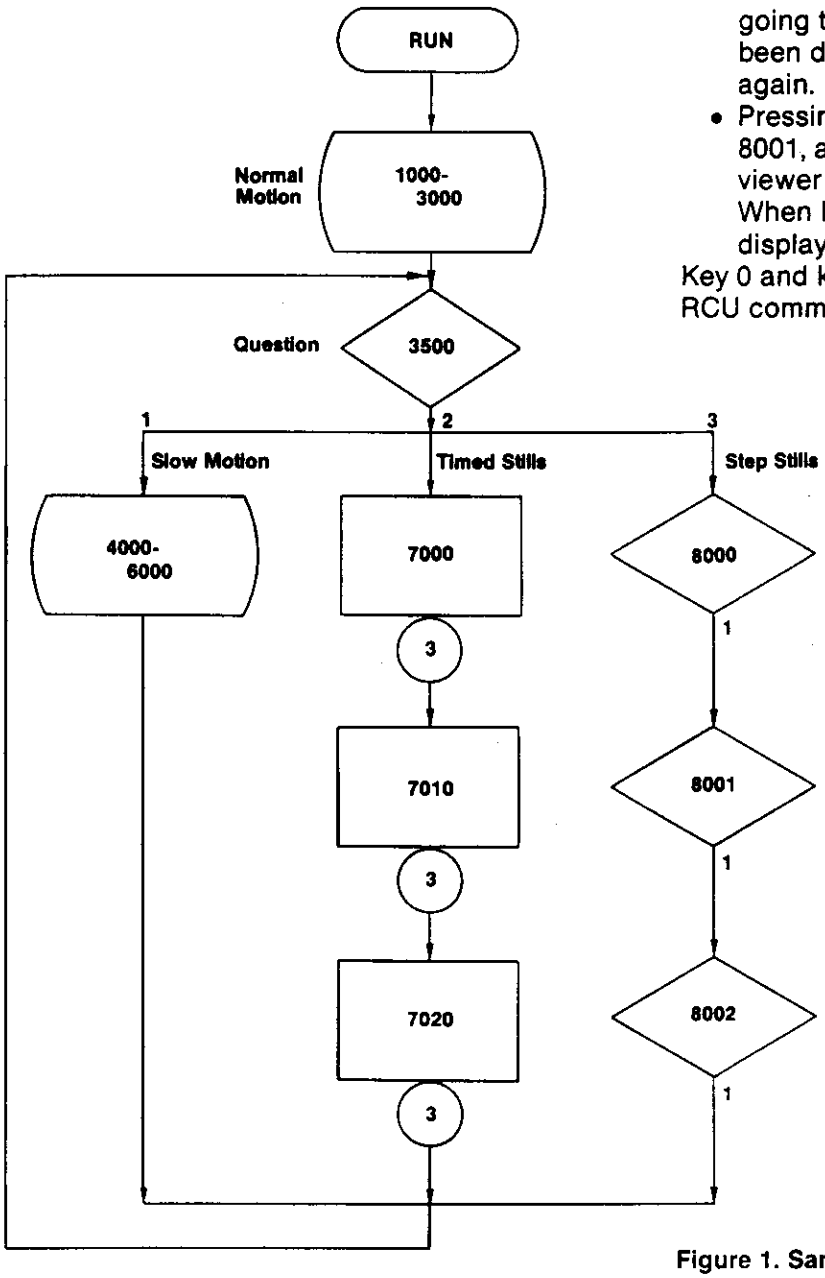

Figure 1. Sample Program Block Diagram

SECTION 5/SAMPLE PROGRAM

LOCATION	OPERAND	COMMAND	PURPOSE
		PROGRAM	Enter PROGRAM mode
0	1000	SEARCH	Display first frame of motion sequence
5	3000	AUTOSTOP	Play motion sequence at normal speed
10	3500	SEARCH	Display question frame
15	4	INPUT	Wait for viewer to press key
17	015	BRANCH	Key 0 = ignored
21	4000	SEARCH	Display first frame of motion sequence
26	6000	SLOW-FWD	Play motion sequence in slow motion
31	10	BRANCH	Key 1 = slow motion
34	7000	SEARCH	Display first timed still
39	30	STOP	Wait for 3 seconds
42	7010	SEARCH	Display second timed still
47	30	STOP	Wait for 3 seconds
50	7020	SEARCH	Display last timed still
55	30	STOP	Wait for 3 seconds
58	10	BRANCH	Key 2 = timed stills
61	67	BRANCH	Key 3 = step stills
64	15	BRANCH	Keys 4-9 = ignored
67	1	RECALL	Set register 1 active
69	3	STORE	Use register 1 as counter
71	8000	SEARCH	Display first step frame
76	2	INPUT	Wait for viewer to press key
78	076	BRANCH	Key 0 = ignored
82	88	BRANCH	Key 1 = step forward
85	76	BRANCH	Keys 2-9 = ignored
88	1	DEC-REG	Subtract one from register 1
90		STEP-FWD	Step forward one frame
91	76	BRANCH	Register 1 = 1
94	10	BRANCH	When register 1 = 0, go to question
		END	Exit program mode to manual mode

Figure 2. Sample Program Step Sequence

APPENDIX A/COMMAND AND INTEGER INPUT CODES

COMMAND/DIGIT	HEXADECIMAL INPUT CODE	DECIMAL EQUIVALENT
AUDIO 1	F4	244
AUDIO 2	FC	252
AUTOSTOP	F3	243
BRANCH	CF	207
DEC REG	F0	240
DISPLAY	F1	241
HALT	BF	191
INPUT	F8	248
LOAD*	CC	204
NULL	00	0
PLAY*	FD	253
RECALL	7F	127
REJECT	F9	249
SEARCH	F7	247
SLOW FWD	F2	242
SLOW REV	FA	250
STEP FWD	F6	246
STEP REV	FE	254
STOP/WAIT	FB	251
STORE	F5	245
0	3F	63
1	0F	15
2	8F	143
3	4F	79
4	2F	47
5	AF	175
6	6F	111
7	1F	31
8	9F	159
9	5F	95

***See Exceptional Programming Commands for use and entry of these commands.**

APPENDIX B/ADDRESS GENERATION TABLE

The following tables are useful in coding a branch command for subroutine. They provide the decimal equivalents for the digit codes 00 through 99 and the decimal equivalents for the digit code 0-9 followed by the BRANCH command.

EXAMPLE: Place at location 0 a BRANCH to location 592.

OPERAND	COMMAND	PURPOSE
510	RECALL	The digit 2 and branch command are stored in locations 2 and 3.
36815	STORE	36815 is the decimal equivalent for the digit code for 2 and the branch command code.
44895	STORE	44895 is the decimal equivalent to the digit codes for 59 and is placed in locations 0 and 1.

APPENDIX B/ADDRESS GENERATION TABLE

DIGITS	EQUIV'S	DIGITS	EQUIV'S	DIGITS	EQUIV'S	DIGITS	EQUIV'S
00	16191	25	36783	50	44863	75	8111
01	16143	26	36719	51	44815	76	8047
02	16271	27	36639	52	44943	77	7967
03	16207	28	36767	53	44879	78	8095
04	16175	29	36703	54	44847	79	8031
05	16303	30	20287	55	44975	80	40767
06	16239	31	20239	56	44911	81	40719
07	16159	32	20367	57	44831	82	40847
08	16287	33	20303	58	44959	83	40783
09	16223	34	20271	59	44895	84	40751
10	3903	35	20399	60	28479	85	40879
11	3855	36	20335	61	28431	86	40815
12	3983	37	20255	62	28559	87	40735
13	3919	38	20383	63	28495	88	40863
14	3887	39	20319	64	28463	89	40799
15	4015	40	12095	65	28591	90	24383
16	3951	41	12047	66	28527	91	24335
17	3871	42	12175	67	28447	92	24463
18	3999	43	12111	68	28575	93	24399
19	3935	44	12079	69	28511	94	24367
20	36671	45	12207	70	7999	95	24495
21	36623	46	12143	71	7951	96	24431
22	36751	47	12063	72	8079	97	24351
23	36687	48	12191	73	8015	98	24479
24	36655	49	12127	74	7983	99	24415

DIGIT PLUS BRANCH COMMAND CODE	DECIMAL EQUIVALENT
0B	16635
1B	4047
2B	36815
3B	20431
4B	12239
5B	45007
6B	28623
7B	8143
8B	40911
9B	24527

APPENDIX C/REGISTER TO MEMORY CONVERSION TABLE

REG	MEM LOC	REG	MEM LOC	REG	MEM LOC	REG	MEM LOC
000	1022	050	0922	100	0822	150	0722
001	1020	051	0920	101	0820	151	0720
002	1018	052	0918	102	0818	152	0718
003	1016	053	0916	103	0816	153	0716
004	1014	054	0914	104	0814	154	0714
005	1012	055	0912	105	0812	155	0712
006	1010	056	0910	106	0810	156	0710
007	1008	057	0908	107	0808	157	0708
008	1006	058	0906	108	0806	158	0706
009	1004	059	0904	109	0804	159	0704
010	1002	060	0902	110	0802	160	0702
011	1000	061	0900	111	0800	161	0700
012	0998	062	0898	112	0798	162	0698
013	0996	063	0896	113	0796	163	0696
014	0994	064	0894	114	0794	164	0694
015	0992	065	0892	115	0792	165	0692
016	0990	066	0890	116	0790	166	0690
017	0988	067	0888	117	0788	167	0688
018	0986	068	0886	118	0786	168	0686
019	0984	069	0884	119	0784	169	0684
020	0982	070	0882	120	0782	170	0682
021	0980	071	0880	121	0780	171	0680
022	0978	072	0878	122	0778	172	0678
023	0976	073	0876	123	0776	173	0676
024	0974	074	0874	124	0774	174	0674
025	0972	075	0872	125	0772	175	0672
026	0970	076	0870	126	0770	176	0670
027	0968	077	0868	127	0768	177	0768
028	0966	078	0866	128	0766	178	0666
029	0964	079	0864	129	0764	179	0664
030	0962	080	0862	130	0762	180	0662
031	0960	081	0860	131	0760	181	0660
032	0958	082	0858	132	0758	182	0658
033	0956	083	0856	133	0756	183	0656
034	0954	084	0854	134	0754	184	0654
035	0952	085	0852	135	0752	185	0652
036	0950	086	0850	136	0750	186	0650
037	0948	087	0848	137	0748	187	0648
038	0946	088	0846	138	0746	188	0646
039	0944	089	0844	139	0744	189	0644
040	0942	090	0842	140	0742	190	0642
041	0940	091	0840	141	0740	191	0640
042	0938	092	0838	142	0738	192	0638
043	0936	093	0836	143	0736	193	0636
044	0934	094	0834	144	0734	194	0634
045	0932	095	0832	145	0732	195	0632
046	0930	096	0830	146	0730	196	0630
047	0928	097	0828	147	0728	197	0628
048	0926	098	0826	148	0726	198	0626
049	0924	099	0824	149	0724	199	0624

APPENDIX C/REGISTER TO MEMORY CONVERSION TABLE

REG	MEM LOC	REG	MEM LOC	REG	MEM LOC	REG	MEM LOC
200	0622	250	0522	300	0422	350	0322
201	0620	251	0520	301	0420	351	0320
202	0618	252	0518	302	0418	352	3018
203	0616	253	0516	303	0416	353	0316
204	0614	254	0514	304	0414	354	0314
205	0612	255	0512	305	0412	355	0312
206	0610	256	0510	306	0410	356	0310
207	0608	257	0508	307	0408	357	0308
208	0606	258	0506	308	0406	358	0306
209	0604	259	0504	309	0404	359	0304
210	0602	260	0502	310	0402	360	0302
211	0600	261	0500	311	0400	361	0300
212	0598	262	0498	312	0398	362	0298
213	0596	263	0496	313	0396	363	0296
214	0594	264	0494	314	0394	364	0294
215	0592	265	0492	315	0392	365	0292
216	0590	266	0490	316	0390	366	0290
217	0588	267	0488	317	0388	367	0288
218	0586	268	0486	318	0386	368	0286
219	0584	269	0484	319	0384	369	0284
220	0582	270	0482	320	0382	370	0282
221	0580	271	0480	321	0380	371	0280
222	0578	272	0478	322	0378	372	0278
223	0576	273	0476	323	0376	373	0276
224	0574	274	0474	324	0374	374	0274
225	0572	275	0472	325	0372	375	0272
226	0570	276	0470	326	0370	376	0270
227	0568	277	0468	327	0368	377	0268
228	0566	278	0466	328	0366	378	0266
229	0564	279	0464	329	0364	379	0264
230	0562	280	0462	330	0362	380	0262
231	0560	281	0460	331	0360	381	0260
232	0558	282	0458	332	0358	382	0258
233	0556	283	0456	333	0356	383	0256
234	0554	284	0454	334	0354	384	0254
235	0552	285	0452	335	0352	385	0252
236	0550	286	0450	336	0350	386	0250
237	0548	287	0448	337	0348	387	0248
238	0546	288	0446	338	0346	388	0246
239	0544	289	0444	339	0344	389	0244
240	0542	290	0442	340	0342	390	0242
241	0540	291	0440	341	0340	391	0240
242	0538	292	0438	342	0338	392	0238
243	0536	293	0436	343	0336	393	0236
244	0534	294	0434	344	0334	394	0234
245	0532	295	0432	345	0332	395	0232
246	0530	296	0430	346	0330	396	0230
247	0528	297	0428	347	0328	397	0228
248	0526	298	0426	348	0326	398	0226
249	0524	299	0424	349	0324	399	0224

APPENDIX C/REGISTER TO MEMORY CONVERSION TABLE

REG	MEM LOC	REG	MEM LOC	REG	MEM LOC
400	0222	450	0122	500	0022
401	0220	451	0120	501	0020
402	0218	452	0118	502	0018
403	0216	453	0116	503	0016
404	0214	454	0114	504	0014
405	0212	455	0112	505	0012
406	0210	456	0110	506	0010
407	0208	457	0108	507	0008
408	0206	458	0106	508	0006
409	0204	459	0104	509	0004
410	0202	460	0102	510	0002
411	0200	461	0100	511	0000
412	0198	462	0098		
413	0196	463	0096		
414	0194	464	0094		
415	0192	465	0092		
416	0190	466	0090		
417	0188	467	0088		
418	0186	468	0086		
419	0184	469	0084		
420	0182	470	0082		
421	0180	471	0080		
422	0178	472	0078		
423	0176	473	0076		
424	0174	474	0074		
425	0172	475	0072		
426	0170	476	0070		
427	0168	477	0068		
428	0166	478	0066		
429	0164	479	0064		
430	0162	480	0062		
431	0160	481	0060		
432	0158	482	0058		
433	0156	483	0056		
434	0154	484	0054		
435	0152	485	0052		
436	0150	486	0050		
437	0148	487	0048		
438	0146	488	0046		
439	0144	489	0044		
440	0142	490	0042		
441	0140	491	0040		
442	0138	492	0038		
443	0136	493	0036		
444	0134	494	0034		
445	0132	495	0032		
446	0130	496	0030		
447	0128	497	0028		
448	0126	498	0026		
449	0124	499	0024		

MEM LOC = 1022-(2*REG)
So, the Register uses memory locations MEM LOC and MEM LOC + 1.

PROGRAMMING FROM VIDEODISC

Any series of 1022 bytes can be loaded into the PR-7820's memory from audio track 2 on the videodisc. There are no invalid or prohibited bytes. This "program dump" on the disc is loaded to fill bytes 0 through 1021, leaving Register 0 (bytes 1022, 1023) undisturbed. This method of entering a program into the player differs from program entry via the RCU or External Computer Interface. All 256 byte patterns may be loaded from the disc, so that all possible commands may be directly loaded (even PLAY, NULL, LOAD) and all registers (except Register 0) can be set to any value desired.

This exceptional attribute of register load from the videodisc may save program space and avoid some run-time delays that occur when the program loads registers itself.

Manually checking out a program that takes advantage of register and command pre-load is a two-step process. To accomplish the same thing via the RCU, first enter manual commands, or enter and run a special program that sets up the registers and the "unenterable" commands. Then the rest of the program can be entered via the RCU. Such programs should be kept as simple as possible and carefully tested.

Any program that exists on the disc as a "program dump" can be loaded under program control when the LOAD command is interpreted. There can be many dumps on the disc, each loading another as needed. Note that Register 0 is not reset or loaded by the LOAD command, so it may serve to pass information between program segments. If the LOAD command encounters a valid leader to a program dump, but fails to load properly, it will try again and again, and may have to be manually stopped.

PROGRAMMING WITH THE EXTERNAL COMPUTER INTERFACE

Commands entered via the External Computer Interface are interpreted exactly as if they had come from the RCU. The ENTER line acts like a "finger press" on a button, the duration of the finger press being significant for the SCAN FWD/REV commands. Note that the SCAN FWD/REV commands move about 300 frames per second when done from STOP or STEP FWD/REV modes. They move about 4000 frames per second from the PLAY or SLOW FWD/REV mode. SCAN FWD = HEX F0 and SCAN REV = F8. (Caution: SCAN FWD/REV commands that move the player past the recorded material on the videodisc may cause the disc to begin spinning in reverse.)

Only 30 different bytes may be generated from the RCU, but all 256 may be entered on the ECI data lines. Five of these bytes are treated specially and four cannot be entered casually as part of program register data. The FF generates no data on the lines (all lines float high), and is the equivalent of the byte normally read by the PR-7820 when no entry is being made (ENTER high, no button press). The PROGRAM command (DF) is treated as a request to skip over a byte in memory. This is similar to RCU entry procedures. The END command (EF) terminates PROGRAMMING mode, just as with the RCU, so it is not entered into memory. The 00 byte is converted to an FF by the microcode, so it looks as if no command has been given. Last, the PLAY command (FD) is translated into a 00 as it is entered as part of a program—otherwise it is treated as the PLAY command.

Note that all five of these bytes (00, FF, FD, DF, EF) could be entered as part of a program from a program dump on disc. In order to load the same program into the player via the computer interface for testing, one must set registers to appropriate values. For example: 510 RECALL 65533 STORE would set FF at location 2 and FD at location 3.

For more details see TP-102, "Installation/ Maintenance Manual for Universal External Interface RS232C."