SuperBox II

Exhibit and Kiosk Controller

Programmer's Manual

Visual Database Systems

Scotts Valley, CA 95066 (408) 439-6890 FAX (408) 439-6894

IMPORTANT LD-V4400/8000 SET-UP REQUIREMENTS

Before the **SuperBox II** can communicate with Pioneer LD-V8000 and LD-V4400 players, the player's baud rate must be set for 4800. If you are unfamiliar with your player and its settings, please refer to the player's operating manual under the section referring to "On-Screen Function Switches."

A brief summary of the procedure for on-screen switch setting is as follows:

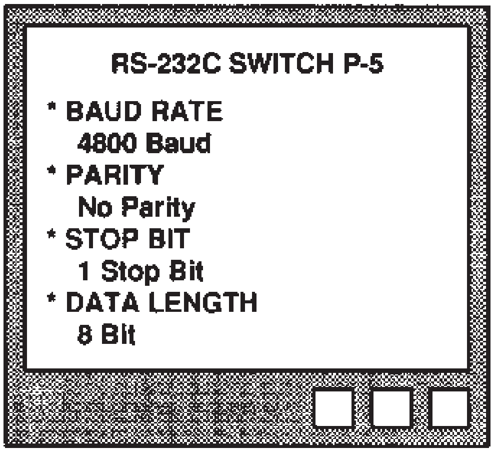
1. Power up the player by pressing and holding the "DISPLAY" key while turning on the power. The on-screen display will read "KEY OPERATION P-0." This page explains which buttons on the front panel or player remote will allow you to locate, select and set the function switches.

2. After the player has powered up, reset the player's function parameters to their default setting. This is done by pressing the "OPEN/CLOSE" key.

3. Using the "SCAN FWD" key, go to the on-screen page "RS-232 SWITCH P-5." (See illustration below).

4. Use the "STEP FWD" and "STEP REV" keys to set this page as follows:
   BAUD RATE = 4800
   PARITY = No Parity
   STOP BIT = 1 Stop Bit
   DATA LENGTH = 8 Bit

5. **VERY IMPORTANT.** After properly setting the RS-232 Switch page as shown below, save the settings by pressing the "DISPLAY" key. Do not power down the player until you have saved your settings.



RS-232C SWITCH P-5

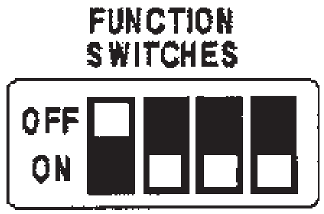
*** BAUD RATE**
4800 Baud
*** PARITY**
No Parity
*** STOP BIT**
1 Stop Bit
*** DATA LENGTH**
8 Bit

Refer to your player's operating manual for a complete description of function settings.

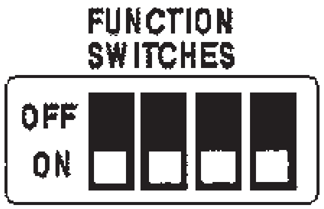
IMPORTANT FUNCTION SWITCH SETTINGS

For Pioneer CLD-V2800, CLD-V2600, CLD-V2400, CLD-V2200, LD-V4200 players

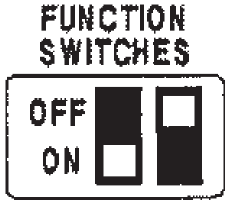
*Before the **SuperBox** can communicate with Pioneer players, the player's baud rate must be set correctly. The **SuperBox** communicates at 4800 baud, 8 data bits, 1 stop bit and no parity. The Functions switch settings below are the recommended settings for these players.*



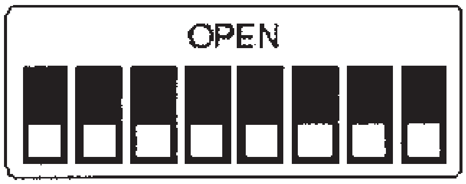
CLD-V2800



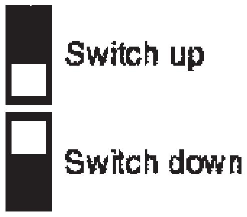
CLD-V2600



CLD-V2400 CLD-V2200



LD-V4200



Visual
Database
Systems

SuperBox II Product Configuration Changes

As we try to continually better support our customer's needs, we are making the following changes to the SuperBox II product line:

Base Product

The SuperBox II will now ship in a standard configuration as follows:

- SuperBox II base unit with firmware for current Pioneer serial videodisc players.
- SBII/P15 player cable.
- SuperBox II power supply.
- 8k EEPROM user's memory (unless 8k programmed EPROM specified).

Manufacturer's suggested price (qty 1-10) $600.00 each

SBII Videoswitch Option

Adding a videoswitch at time of original SBII purchase.

- Includes additional SBII/P15 player cable.

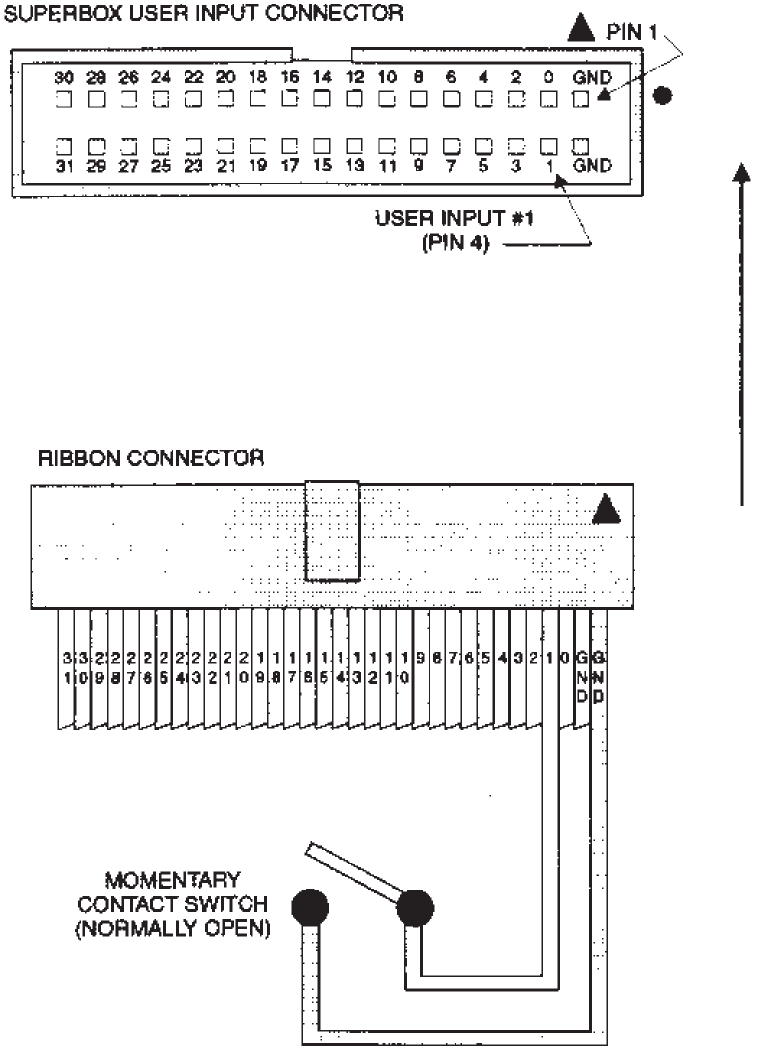
Manufacturer's suggested price (qty 1-10) $100.00 each

These changes are effective August 1, 1994

Visual
Database
Systems

SuperBox II User Input Wiring Diagram

The diagram below illustrates a simple user input connection for Button 1. Ground (GND) is available at both pins 1 and 2. Button 1 is located at pin 4. The corresponding ribbon connector shows pin 1 (typically indicated by a red stripe on the ribbon cable) and pin 4 in a normally open circuit. Momentary shorting of these two contacts will trigger Button 1.


SUPERBOX USER INPUT CONNECTOR
PIN 1
30 28 26 24 22 20 18 16 14 12 10 8 6 4 2 0 GND
31 29 27 25 23 21 19 17 15 13 11 9 7 5 3 1 GND
USER INPUT #1
(PIN 4)
RIBBON CONNECTOR
31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 GND GND
MOMENTARY
CONTACT SWITCH
(NORMALLY OPEN)

March 8, 1995

SuperBox II User Output Wiring Diagram

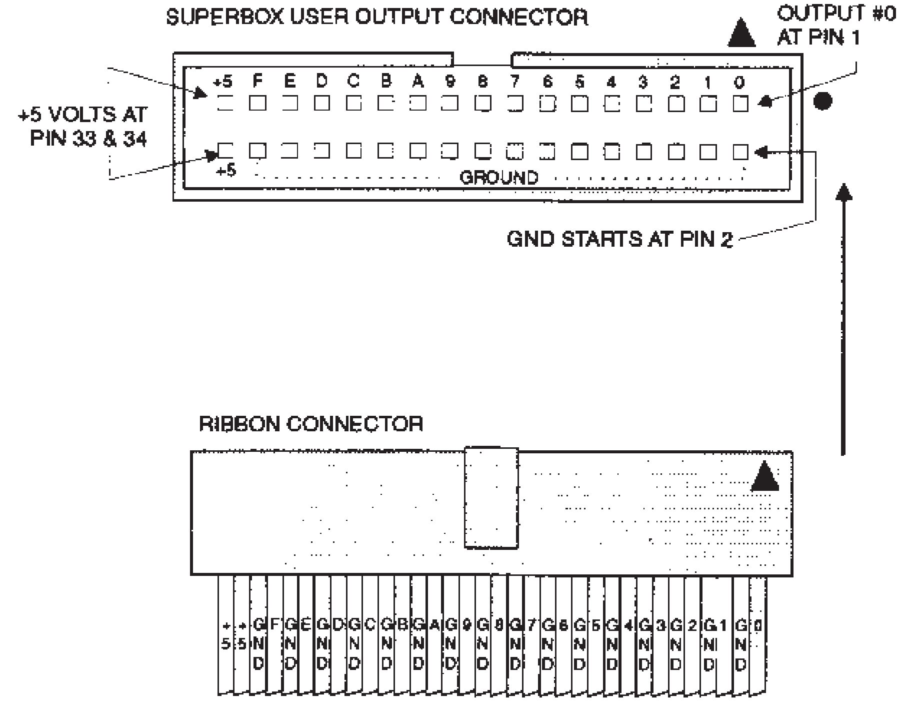
The diagram below illustrates the user output connections for SuperBox II.

Ground (GND) is available at all pins along the bottom row of the connector (even numbered pins), except for pin 34 which is a +5 volt output.

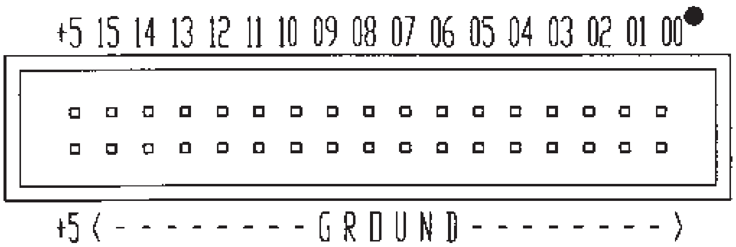
Output 0 is located at pin 1 and the remaining User Outputs 1-9, A-F are also located along the top row of the connector (odd numbered pins). The last pin, 33, is a +5 volt output.

The maximum load on the +5 volt pins must not exceed 200mA.

The corresponding ribbon connector shows all the output connections in series as they are on the ribbon cable.

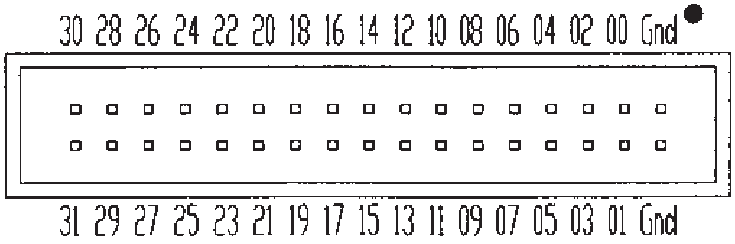

SUPERBOX USER OUTPUT CONNECTOR
OUTPUT #0
AT PIN 1
+5 F E D C B A 9 8 7 6 5 4 3 2 1 0
+5 VOLTS AT
PIN 33 & 34
+5
GROUND
GND STARTS AT PIN 2
RIBBON CONNECTOR
+5 +5 GND F GND E GND D GND C GND B GND A GND 9 GND 8 GND 7 GND 6 GND 5 GND 4 GND 3 GND 2 GND 1 GND 0

March 8, 1995



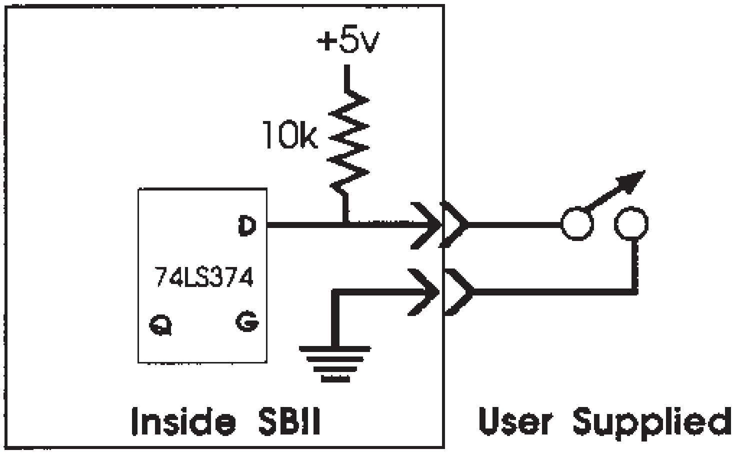
SuperBox II User Output (Output Lines)

AMPMODU 87478-4
mateswith AMPMODU polarized connectors
AMP-LATCH and Ribbon cable connectors

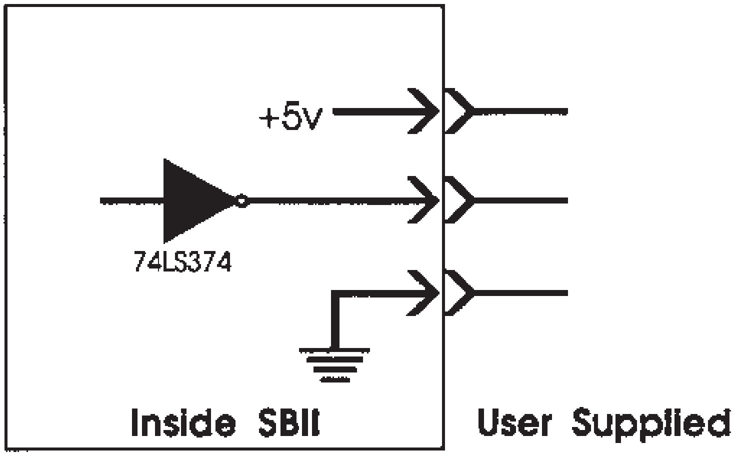

30 28 26 24 22 20 18 16 14 12 10 08 06 04 02 00 Gnd
31 29 27 25 23 21 19 17 15 13 11 09 07 05 03 01 Gnd

SuperBox II User Input (Button Numbers)

AMPMODU 87478-4
mateswith AMPMODU polarized connectors
AMP-LATCH and Ribbon cable connectors



SuperBox II User Input circuit
(1 of 32)



SuperBox II User Output circuit
(1 of 16)

Visual Database Systems
614 Bean Creek Road
Scotts Valley, CA 95066
(408) 438-8396

SuperBox II Rev B Engineering Change Notice

EEPROM Write-disable Jumper Now Installed

With all SuperBox II production starting in April, 1990, we have installed a jumper and shorting block on the circuit board at location JP3, near EEPROM 0. This location is well marked on the board. Removing this jumper disables the EEPROM write line and should eliminate false writes to the EEPROM on system startup power glitches.

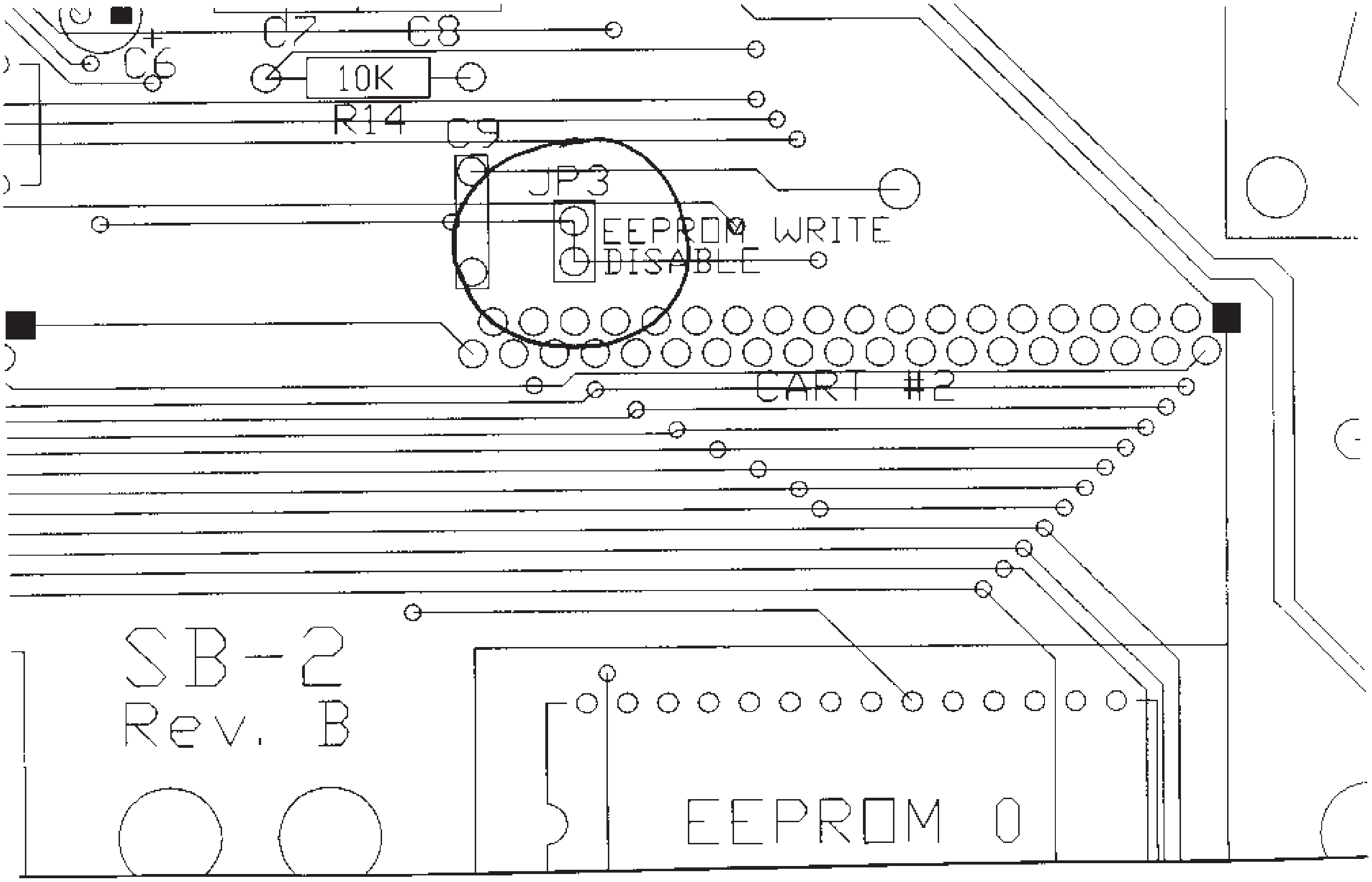
To gain access to the jumper, it is necessary to remove two screws on each side of the top cover. Moderate caution should be exercised to prevent static discharge when handling the board. The shorting block's position can be changed with the power turned on or off.

To program the EEPROM the shorting block must be in position between the two pins, removing the shorting block disables the write line. Store the shorting block on one of the pins when the write line is disabled. The SuperBox II is shipped with the write line enabled unless we have pre-programmed the box for you.

We recommend that you disable the write line after development of your program before installing it in its permanent location. The ability to program the EEPROM can be quickly regained by installation of the shorting block.

Earlier production versions of the SuperBox II also have this jumper position marked but did not have the jumper installed. This can be done be a competent electronics technician. The procedure is to cut the trace outlined in position JP3, then installa connector and shorting block. (Molex styles 70343 and 7859)

April, 1990



Additions & Corrections to the SuperBox User's Manual

Player firmware upgrade fixes one problem, causes another

In late '87 Pioneer released a new version of firmware in it's LD-V4200 player to fix a bug found in playing very short clips with an autostop. This new release, (150202) responds differently to the opening sequence of commands sent in firmware version 1.06 and earlier. Specifically, the player does not remove the "PLAYER SPINUP" displayed as a part of the SuperBox's startup sequence. Version 1.08 and higher of the SuperBox firmware corrects this problem and can be supplied upon request. The problem may also showup in your application code and appears to be caused by combining certain commands on a single line, specifically after displaying text on the screen, sending the command:

```
CS1PR
[
LINE OF TEXT
```

will not clear the screen and will cause an error on the text display. Sending the commands on separate lines will work correctly.

Sending the LD-V4200 an "<Esc>X?" will return the player's firmware version number. For instance at VDS our pre-release LD-V4200 returns 150200 but the newest players will return 150202. These players will show slightly different behaviour with certain commands.

When calling us with technical questions please know which player model you own. This can be done with the SuperBox in pass-through mode by sending an <Esc>X?

Typography style problem in manual

In all places in the manual (pages 15,17,18,21) where <Esc> BTN ##: is seen there is a space before BTN ##: There should not be a space, as can be seen on page 32.

Page 9 - c. Pass through mode

Remember when a carriage return is sent to the player a response is returned from the player to the computer. This response is either an "R" or an error code like "E04" followed by a carriage return. No line feed is sent by player so the responses will appear as follows:

```
2345PL                                  'sent in pass through will look like
R345PL                                  'once the response is returned from player.
```

Page 26 - Set AUTO-EXECUTE

Remarks: auto-execute times must be at least 2 minutes apart

```
<Esc>X?
AUTO # 1  TIME : 13:10  BTN : 0
AUTO # 2  TIME : 13:12  BTN : 1
AUTO # 3  TIME : 13:14  BTN : 2
AUTO # 4  TIME : 13:15  BTN : 3
Ok
```

In this case buttons 0,1,2 will work but button 3 will not get executed.

Page 30 - Add and subtract example is on page 71 (not page 60)

Page 30 - Frame chapter example is on page 75 (not page 63)

Page 34 - SUBROUTINE button 1 should be typed in as follows:

```
<Esc>B1=
100SE
[
#10(&30)                    ;here the #10(  ) is added so 10 still frames
123SE                       ; will be seen
[
```

Page 54 - CLEAR SCREEN

The newest version LD-V4200 players respond differently to this command. In the past the following was perfectly valid:

```
CS1PR
[
THIS IS LINE 1
[
```

However in the most recent models of the LD-V4200 this must be written as follows:

```
CS
[
1PR
[
THIS IS LINE 1
[
```

Page 24 - TIME is changed to the following

```
<Esc>T
CURRENT TIME : 06:48
ENTER NEW TIME (HH:MM) : 16:00
CURRENT DATE : 11/20/00
CURRENT TIME : 16:00
Ok
```

Page 23 - DATE format is changed to the following:

```
<Esc>D

CURRENT DATE : 11/20/00
ENTER NEW DATE (MM/DD/YY) : 11/20/87
CURRENT DATE : 11/20/87
CURRENT TIME : 16:00
Ok
```

Page 23 - CLEAR USAGE LOG the command is now:

```
<Esc>CUSE                   (no longer <Esc>C)
```

Page 24 - LOG TITLE format is changed to:

```
<Esc>L
LABEL : KIOSK #3 IN LA
ENTER NEW LABEL : VDS SUPERBOX, V1.07
LABEL : VDS SUPERBOX, V1.07
Ok
```

SBCOMM available as a programming aid.

SBCOMM is a program designed to aid in programming the SuperBox. It has several features not available to any other communications programs. The program is available from VDS for $150.00 which includes a PC-S/SB cable. The Main menu looks like this:

```
SBOXCOMM  V1.00   Port: COM1   (c) 1987 - Visual Database Systems

          OPTIONS
                               1) Communicate with the SuperBox
                               2) DownLoad a Text File to the SuperBox
                               3) DownLoad an EEPROM Dump File to the SuperBox
                               4) UpLoad SuperBox Button Contents to a Text File
                               5) UpLoad SuperBox EEPROM Contents to a File

                               0) Exit to DOS


                                      Response : (1,2,3,4,5 or 0) :
```

Format of the EEPROM Dump files is same as explained on pages 19 and 20.

The TEXT file download format enables programmers to write programs using any word processor and allows comments to be added between buttons.

```
                              ' This is a simple text file that can be loaded to the SB42
                              ' using a text editor, I like Borland's SideKick because it
                              ' can be popped up any time
                              ' button 0 sets reg. C for "black" searches
                              ' reg. A user area display turned off
                              ' delay 18.0 seconds 9 times


BTN 0:
16RC
[
0RA
[
#9(!180!)
                              ' there must be two blank lines before the next button
                              ' is started.
                              ' Button 1 plays a segment of video over and over again


BTN 1:
45005SE
[
53966PL
[
                              ' This is the end of the text file
```

New Commands Added

Clear all memory

A new command has been added to the SB42 command set, "<Esc>CALL", this clears all the EEROM memory to "FF". Use this with caution as it will wipe out anything that is stored in SuperBox memory.

Show commands

The following command are now available for SuperBox 42 starting with EPROMS dated 12/3/87 ver 1.08b (with clock calendar/usage log option).

\SU	Show usage log using player character generator.
\SDn	Show date on line n. n = 1 to 8
\STn	Show time on line n. n = 1 to 8

Wait Command

The wait command is useful to delay such as when displaying the usage log on the screen.

\W	Wait for press of any button before continuing.

Changes to Superbox in version 1.10

1) Increase # of user variables from 10 To 100 ($0-$99)

2) Add conditional execution based on variables

```
?LeftSide Operator RightSide| CONDITIONALLY EXECUTED STRING |
   LeftSide: $V, $V1 + $V2, $V1-$V2, $V + nnnnn, $V-nnnnn, +, +C, +F, +T, D, MO, Y, MI, H
   Operators: =, >, > =, <, < =, < >
   RightSide: $V, $V1 + $V2, $V1-$V2, $V + nnnnn, $V-nnnnn, nnnnn, +, +C, +F, +T
   Where: $V, $V1, $V2 is any Variable (0-99) nnnnn is any Number (0-65535)
          +, +C, +F, +T Causes the current frame/chapter/time to be read from the player (SB4200 only).
          D, MO, Y, MI, H Use the Date, Month, Year, Minutes, and Hours from the clock.
```

Example:

```
?$0 > 100|F1000SE                ;IF $0 IS GREATER THAN 100
[                                ;THEN SEARCH TO FRAME 1000
2000PL                           ;AND PLAY TO FRAME 2000
[
|                                ;END OF CONDITIONAL STRING
5000SE                           ;CONTINUE FROM HERE AFTER CONDITIONAL
[
?$23 < > $10-$5|1PR
[
TEST SCORE $23
[
|
```

Note : Nesting is possible

```
?0 = $1|?$0 = 100|DO THIS COMMAND STRING||
   If both $0 and $1 = 100 do the command string
```

3) Goto a label within the current button with a fixed number or variable

```
Lnnnnn  or  L$V
100SE
[
>L$20                              ;GOTO LABEL POINTED TO BY VBL 20
\L1000,1000PL                      ;IF VBL 20 = 1000, PLAY TO 1000
[
>L5000                             ;IF VBL 20 = 5000, SINGLE STEP TO 1000
\2000,#900(!10!SF
[
)
\L5000,10000SE                     ;CONTINUE HERE AFTER STEP/PLAY
[
```

Note: The label may be either above or below the GOTO LABEL command, but it it must be within current button. If the specified label is not found, button execution terminates and control returns to the main loop waiting for a button press.

4) Variable length delays using !$v! where: V = 0-99.

5) Call a variable subroutine using &$v where: V = 0-99.

6) Do a loop a variable number of times: #$v( )

The Following Commands are valid only in SuperBox II

The introduction of SuperBox II continues compatibility with the existing SuperBox command set and adds additional capabilities. These include additional memory (to 56k bytes), support of two videodisc players, 16 user controlled output lines, a printer port option and a video switch option.

1) Turn on the printer from the computer for development use

```
<Esc>P1<CR>                        Turn ON the printer
<Esc>P0<CR>                        Turn OFF the printer
```

2) Print a line from a button using: \P"

```
\P"THIS LINE IS PRINTED.
\P"THE TOTAL IS : "$0" POINTS."    Uses variable within text
\P                                 P followed by <CR>. Newline
```

3) Print the date or time using \PD AND \PT.

```
\PD                                Date printed as MM\DD\YY
\PT                                Time printed as HH:MM
```

4) Print a special character \P\nnn.

Where : nnn = 0 - 255, decimal code of ASCII character

```
\P\27                              Sends ESCAPE character to printer.

\P\12                              Sends FORM FEED character to printer.
```

5) Select active player commands

```
{ or \P1                           Selects Player #1 as the active player

} or \P2                           Selects Player #2 as the active player

}2000SE                            ;Search TO 2000 ON #2. Don't wait for status

{1000SE                            ;Search TO 1000 on player #1

[                                  ;Wait until search complete
5000PL                             ;Play to 5000 on #1
[                                  ;Wait until play complete
}5000PL                            ;Now play to 5000 on player #2
```

6) Select video source commands

```
\V1                                Select Video Input #1
\V2                                Select Video Input #2
```

7) Player and video source toggle commands

```
\TP                                Change player selection to other player
\TV                                Change Video Source to the other input
```

8) Use of variables to set user outputs

There are 16 TTL level user outputs available in the SuperBox II. These can be SET / CLEARED either using the number of the output in HEX (0-F) or by assigning a variable with the output number in DECIMAL (0-15). The level can also be set either using "0" / "1" or a variable whose value if 0 will clear the output, otherwise will set the output.

```
~50~B1~F0                          Constants used for both # and level
~5$3~B$4~F$1                       Constants used for #, variable for level
~$2,0~$30,1~$1,0                   Variables used for #, constant for level
~$6$2~$3$57~$1$2                   Variables used for both # and level
```

TABLE OF CONTENTS

I. Introduction

Congratulations! You have the simplest to use, yet most powerful **Programmable Videodisc Controller** available today. Visual Database Systems is proud to introduce the concept of "Level II in a box" to the interactive video community, and to share the results of our experiences with videodisc control.

The SuperBox is a stand alone dedicated controller designed to facilitate the use of all the features of the Pioneer brand LaserVision videodisc players. The SuperBox is designed to startup automatically. When the power is turned on the player will be start executing the SuperBox program.

The SuperBox interfaces to either user supplied momentary contact switches, a remote control unit, or optionally to a touchscreen. For operation without user intervention, an automatic execution mode based on time of day is available as an option which will also provide usage statistics.

To configure a fully functional exhibit, you will need a Pioneer LD-V4200 videodisc player, an NTSC monitor with video and audio cables, a videodisc and a source of power. Your SuperBox is pre-loaded with a demonstration program that will run when you plug everything together. See Appendix 4 for a complete description of the demonstration program.

Read section II.1 "Getting Started Quickly" to test your SuperBox and start creating your own interactive video.

I.1. WHERE CAN THE SuperBox BE USED

The SuperBox was specifically designed for low cost point of information applications. We at VDS wish to encourage all your new ideas. The SuperBox is not waterproof so please do not leave it outdoors. Some typical applications are museum kiosks, "Point Of Purchase" kiosks and restaurant kiosks.

I.2. THE BUTTON CONCEPT

The button concept is the key to understanding how the SuperBox works with the videodisc player. Each "button program" sends a string of commands to the videodisc player. The buttons can also be associated with sectors on a touch screen.

Each time a button is pressed the commands stored in that button are sent to the videodisc player, and the desired video segments are played. Multiple choice answers are easily handled by assigning variables to several buttons, and then assigning "button program" numbers to those variables. Then the value of the variables can be changed within the button programs.

I.3 Product Description

The SuperBox is designed to automatically begin operation upon application of power. The typical configuration has the controller, videodisc player and monitor all on a single power strip. When power is applied simultaneously to the SuperBox and player, the SuperBox will do the following: 1.) wait 5 seconds for the disc to be "set"; 2.) Send a spin up command to the player; and 3.) Display a opening screen with the time and date if available and then automatically execute the program, in button 0. If the player is not detected, the box will be in Command Mode and wait for user intervention.

If a computer or terminal is attached to the SuperBox during normal operation, the commands sent to the player are echoed to the computer's RS232 port along with the player's responses. This is a valuable aid in debugging an errant program segment.

The SuperBox controller is based on an Intel 8031 micro-controller. The hardware supports 2 RS232C serial ports, 32 TTL input lines, 4 additional reserved TTL level I/O lines, 2k or 8k bytes of EEPROM (Electrically Erasable, Programmable Read Only Memory), for button program storage; 8k bytes of EPROM (Erasable Programmable Read Only Memory), for videodisc player controlling firmware; 2k or 8k bytes of RAM memory which may be battery backed up and include a clock/calendar, log and autostart buttons.

The same hardware configuration is used to support several different Pioneer players. To accommodate their differences, the Firmware EPROM is changed. This capability allows an easy path to future enhancements or customization.

The SuperBox's programming language is simple and powerful. The commands include all of the tools you need to program, edit and execute videodisc commands. These are used to create segments called "buttons" or "button programs" but they can also be thought of as pre-defined program labels. Additional commands allow looping, subroutines and delays.

Programming the SuperBox is easy. First, communication is established with a terminal or host computer running as a terminal emulator. Once you are in communication with the box, you can enter, examine, test and modify the short programs (that are to be executed when requested by the user through the button interface or optional touchscreen and automatic execute capabilities).

User type input is either from one of the 32 user input pins from a touchscreen, or possibly from a remote control unit. The user input pins are designed to be used with momentary contact switches. Switches are normally open, momentary contact type, and can be triggered by traditional push button, VDS remote control unit or any other closure such as a burglar alarm sensor to detect the presence of a customer.

I.4. FIRMWARE UPDATES

Because the SuperBox is firmware based, many features may be added with relative ease, and custom requirements are relatively easy to handle. Our final goal is to release the ideal Videodisk controller - at a low cost and easily programmed.

Version #	**Features added**
1.02	Variables added Frame, grab and search capability
1.04	Buttons enabled and disabled function.
1.05	Read clock only if clock chip installed.
1.06	a) Intel format files for SuperBox program dumps and program loads. b) Enable or disable specific buttons. c) Delays are now echoed to computer in tenths of seconds, instead of a hex code. d) Method of setting Maximum Number of Buttons is changed. e) Time and chapter, grab and search capability This version has the same program interface as the new SBC42 (program cartridge version).
1.07	a) User control outputs, to allow TTL control of 4 external devices. b) Remote control firmware added.

1.5. RELATED PUBLICATIONS

a. "PIONEER LD-V4200 OWNERS MANUAL"
This manual contains a description of LD-V4200 computer interfacing, and also some information on its commands and operation. Available from:

Pioneer Communications,of America, Inc.,
600 East Crescent Avenue,
Upper Saddle River, NJ 07458.

b. EIA Standards RS232C - "INTERFACE BETWEEN DATA TERMINAL EQUIPMENT AND DATA COMMUNICATION EQUIPMENT EMPLOYING SERIAL BINARY DATA INTERCHANGE" This document is available from

Electronic Industries Association
Engineering Department
Standards Sales Office
2001 Eye Street, N.W.
Washington, DC 20006

1.6. ACKNOWLEDGMENTS

We refer to several different products in this manual; the following lists each one and its company:
* SuperBox and WISP are trademarks of Visual Database Systems.
* Turbo Basic, SuperKey and Sidekick are registered trademarks of Borland International
* CrossTalk is a registered trademark of Microstuf Inc.
* IBM PC/XT ,IBM AT are registered trademarks of International Business Machines Corp.
* LOTUS 123 is a registered trademark of Lotus Development Corp.

II.1 Getting Started FAST

This section will show you how to connect the SuperBox to your player, start it up and see a quick preview of what your new controller can accomplish. Next we will show you how to connect a terminal or PC and observe the conversation that the box is having with the player. Finally we will lead you through the steps required to write your own simple program.

Before you start, check the material that you received to be sure that you have everything you need.

SuperBox Parts List

1. SuperBox SB42 or SB42TS
2. Wall mount power supply
3. Short cable labeled SB/LDV4200
3. Packet containing a DB37S connector
   a DB37 hood and a short wire with sockets

User Supplied Items

1. Pioneer LD-V4200 videodisc player
2. NTSC color monitor
3. Video and audio cables
4. CAV mode videodisc
5. (recommended) A switched power strip

To program the SuperBox you also need

1. A personal computer and communication program
   or a terminal capable of 4800 baud
2. A connecting cable (see Appendix 1)

II.2. The Button Interface

The button interface will be called the User Interface. You must supply your own switches to actuate the button programs.

The large DB-37 connector is the User interface (UI), 32 of the 37 pins on this connector are associated with a button program number. See Appendix I for the complete pin-out description. When a pin associated with a button program is touched to pin 37 (ground) - that button program will be executed.

II.3. Hooking it all Together

1. Connect the videodisc player and monitor and verify that they are working correctly together. Plug the player into the power strip if available.

2. Turn the player's power off at the power strip, open the player's front switch access cover and flip all of the switches UP and replace the panel. Connect the SB/LDV4200 cable to the connector on the rear of the player and the connector marked PLAYER EXTERNAL CONTROL on the SuperBox.

3. Connect the wire with sockets by attaching one end to pin 37 on the User Interface (DB37 connector) leave the other end dangling for a moment. Connect the power supply to the SuperBox and plug it into the power strip. Turn on the power strip or apply power to the player and SuperBox simultaneously, or power player up first.

4. To watch a simple demonstration of the SuperBox in a stand alone operation. When prompted on the screen, touch the free end of the wire to the correct number pin on the user interface and simulate a button contact. Don't leave the wire connected, just touch and then let it dangle. See appendix I, page 38 for details of pin outs.

5. Congratulate yourself, you have just completed installation of an interactive videodisc system!

II.4. The videodisc Player

Most videodisc players support both Constant Angular Velocity (CAV), and Constant Linear Velocity (CLV) videodiscs. However, The type of videodisc determines what player functions are available. For instance CLV discs do not allow still frames to be seen.

The SuperBox is limited only by the videodisc player command set. Please note that some commands apply to CAV format videodiscs and others to CLV videodiscs. Thus, any command that is supported by the videodisc player command set is also supported by the SuperBox. For a complete list of the commands for the Pioneer LD-V4200 player, see Appendix III.

II.5. The Computer

The computer is not needed during normal SuperBox operation. However, a computer is needed to access various functions, such as creating and editing the program strings. This is best done by using a communications program, like Crosstalk, and using the SuperBox's commands -- found in section III.2 . The computer is also needed to set time and date, view the usage log, to set the autobutton feature.

VDS presently uses the RS-232 port on a IBM-PC, the connection is made with the VDS serial cable **(P.N. PC/SB)**. The communication parameters must be set as follows:

4800	BAUD rate
8	Data bits
no	Parity
1	Stop bit

The communications parameters (baud, no. bits, parity) are set in the firmware EPROM and are not modifiable by the user. The firmware completely determines the controller's functionability.

III.1 Modes of Operation

These are four modes of operation for the SuperBox:
 a. Command mode or CMD mode
 b. Program mode
 c. Pass through mode
 d. and, Execute/Touch Screen or ETS mode

a. Command Mode

At power up, if a player **is not** connected, the SuperBox will be in CMD Mode and the "OK" prompt will be sent to the computer. Command mode is the "IDLE" mode for the SuperBox. It is achieved by sending an <Esc> with the SB42 or by sending an <Esc>CMD with the SB42-TS -- this will be echoed as **eCMD.**-- and also disables the Touch Screen. Once in CMD mode all the SuperBox commands may be used and seen on/from the computer.

b. Program mode

Program mode uses the - prompt. Any character or string of characters sent to the SuperBox in this mode will be stored in EEROM. All player commands (page 41) and all SuperBox functions and variables (page 26) are valid.

c. Pass Through Mode

Pass Through Mode allows the programmer to access the videodisc player commands directly from the computer. In this mode all commands sent to the SuperBox will be "passed through" , untouched, to the player, and all player responses and error codes will be returned to the computer.

d. Execute/Touch Screen Mode

At power up, if a player **is connected**, the SuperBox will be in ETS Mode and and button 0 will be executed -- if it exists. A button press or switch closure will also put the SuperBox into ETS mode. When in this mode the program in the SuperBox will execute until it is stopped. How a program is halted depends on the SuperBox model. With the SB42 an <Esc>bb will execute button **bb** and put the SuperBox into ETS mode. With the SB42-TS an <Esc>R will put the SuperBox into ETS mode, and the Touch Screen will be re-enabled. For the touch screen to work a mask must be enabled. This is usually done at the top of a button or by using the **<Esc>Smm** command.

III.2 Summary of Commands for the SuperBox 4200

<Esc>bb	Executes button bb from computer. (bb= 0-2000)
<Esc>B?	Display unused buttons.
<Esc>Bbb**?**	Display button bb contents.
<Esc>Bbb=	Clear button bb, and input new contents.
<Esc>Bbb-	Edit contents of button bb. (See note 1.)
<Esc>Bbb,	Append to end of button bb.
<Esc>CBTN	Clear contents of all buttons.
<Esc>E	Dump EEROM contents. (See note 2.)
<Esc>I	Input EEROM contents. (See note 2.)
<Esc>M	Display all valid buttons and Masks.
<Esc>MB	Display all valid buttons.
<Esc>Nn	Set maximum button number to n (n=0,1,2,3,4,5,6)
<Esc>P	Enterpass through mode.<Esc>exits Passthru mode.

Commands are available with the clock/calendar versions only:

<Esc>C	Clear usage log.
<Esc>D	Display and change date
<Esc>L	Display and change label
<Esc>T	Display and change time.
<Esc>U	Output usage log data.
<Esc>Xxx	Input auto-execute button xx (xx=0-15).
<Esc>X?	Display active auto executes- Clock option.

Commands are available with the touchscreen versions only:

<Esc>CINIT	Re-initializes SuperBox at next power-up
<Esc>CMSK	Clears all mask data
<Esc>CMD	Enter Command Mode
<Esc>MM	Displayall valid masksectors
<Esc>R	Exit Command Mode and return to touch screen mode
<Esc>Smm	Enables touch screen mask , mm=0-99.
<Esc>Smm?	Display Mask mm.
<Esc>SmmC	Clear Mask mm.
<Esc>Smm,ssC	Clear Sector ss and sectors greater than ss, of Mask mm.

<Esc>Smm,ss=ULx,ULy,LRx,LRy,bb   Creates a mask sector

Input mask sector data as follows:

Smm - Mask number
ss - sector number, sector 0 must exist
ULx - Upper Left X-coordinate
ULy - Upper Left Y-coordinate
RLx - Right Lower X-coordinate
RLy - Right Lower Y-coordinate
bb - Button to be performed when the sector is touched

<Esc>Zbb	Sets mask memory area. All buttons from bb to Button Maximum are used for mask data.

Functions used in Button Programs

"string" Surround character strings in print (PR) commands.

!ddddd! Delay ddddd tenths of seconds. ddddd = 1 to 65535.

:bbl.bb2 Disable buttons from bbl to bb2.

;bbl.bb2 Enable buttons from bbl to bb2.

#ddddd(commands) Loop ddddd times repeating commands between ( ). Nesting of loops is NOT permitted.

&bb Call button bb as a subroutine then return. Subroutines are limited to 1 level deep.

[Wait for the player to return a response status.

>bb Branch to button bb.

<M> Activates mask M

, (comma) Delimiter for variable assignments. Preceding variable not sent to player.

~pb (tilde) allows pin number p (p = 0-3), to be set by bit **b** (b = 0-1, were 0 is ground and 1 is TTL 5 volts)

Variable Operations used in Button Programs

Vn,V0,V1,V2 are variable numbers between 0 and 9.
Valid variable values 0 < xxxxx < 99999.

$Vn=xxxxx	Assigns xxxxx to variable Vn.
$V0=$V1	Assigns value of $V1 to $V0.
$V0=$V1+xxxxx	Assign xxxxx plus $V1 to $V0.
$V0=$V1-xxxxx	Assign difference of xxxxx and $V1 to $V0.
$V0=$V1+$V2	Assign sum of $V1 and $V2 to $V0.
$V0=$V1-$V2	Assign difference of $V2 and $V1 to $V0.
$Vn=+	Assign current frame number to $Vn. **SEE Note 3.**
$Vn=+xxxxx	Assign current frame number plus xxxxx to $Vn.
$Vn=-xxxxx	Assign current frame number minus xxxxx to $Vn.
$V0=+$V1	Assign current frame number plus $V1 to $V0.
$V0=-$V1	Assign current frame number minus $V1 to $V0.
$Vn	Send value of the variable to player.
$V0+$V1	$V0 and $V1 are added and result sent to player.
$V0-$V1	$V1 is subtracted from $V0 and result sent to player.
$Vn+xxxxx	Add xxxxx to the value of $Vn and send result to player.
$Vn-xxxxx	Subtract xxxxx from $Vn and send result to player.
+xxxxx	Add xxxxx to the current frame number.
-xxxxx	Subtract xxxxx from the current frame number.
+$Vn	Read present frame number and add value of $Vn to it.
-$Vn	Read present frame number and subtract $Vnfrom it.

Notes:

1. As many as 2000 buttons may be set. However, since more buttons mean less BYTES/BUTTON a minimum of 15 BYTES/BUTTONare allowed. Therefore with 8K memory 500 buttons is the maximum, and, with 2K 100 buttons is the maximum.

2. Intel HEX format is used. Once program is written, it can be dumped to a file and loaded again to the SuperBox from that file.

3. Assign a current chapter or time code number, place a "C" or a "T" after the "+" . +, +F, +C, +T, +C$Vn, are all valid.

4. Editing a button with <Esc>Bbb-
   <Ctrl>X Terminates the string at the current character.
   <Ctrl>C Changes current character to a carriage return <CR>.
   <CR>   Retains the current data in this location.
   A <CR> in the button contents is displayed as "c".

III.3 Description for the SuperBox 4200 Commands

This section describes the SuperBox commands needed to control the Pioneer LD-V4200 Laser disc player. For details on the LD-V4200 player commands see Appendix III.

The SuperBox prompts:

1. Ok This prompts user that the SB42 is waiting and ready to be programmed or viewed. This prompt is seen after the power up sequence, after leaving Pass Thru mode, and when a button is finished executing.

2. - This prompt indicates that the box is in Program mode. Any string sent to this prompt will be stored in memory. If a command within the string is not a valid player command then when that button is pushed and the command is sent, the player will return an error response (see Appendix III), and proceeds with the command. To leave Program mode send a <CR> on a line by itself.

3. ^ - INVALID ENTRY This tells the user that extra Escape characters have been received. The SuperBox ignores these extra characters and performs function.

When a button is pressed or executed from the computer, the program will be interrupted from the computer when <Esc> is pressed, and the OK prompt will be returned.

Each SB42 command is initiated by sending <Esc> before the command, and for all practical purposes <Esc> is considered part of the command.

Note :
1. in the examples the **BOLD FACED** commands are user inputs in response to prompts.

2. The single quote mark denotes program comments. These comments cannot be stored in the SuperBox.

EXECUTE BUTTON <Esc>bb

Function Allows the button **bb** to be executed from the computer.

Remarks Allows programmer to verify a program without using the User Interface. Goes into Execute mode (ETS), press <Esc> to get back to Ok prompt.

DISPLAY UNUSED BUTTONS <Esc>B?

Function Displays all available buttons

Example

```
<Esc>B?<CR>
16 BUTTONS ENABLED  ( 0 - 15 )
512 BYTES / BUTTON
TABLE OF UNUSED BUTTONS :

                            5    6    7
       11        13   14   15

Ok
```

VIEW BUTTON <Esc>Bbb?<CR>

Function

<Esc>Bbb? asks the SB to send the program stored in specified button number (bb), back to the host computer.

Syntax

<Esc>B(integer)**?<CR>** sent to SB

Remarks

The BUTTON VIEW command shows user what is stored in a specified button number.

The button number (bb) can be an integer value between 0 an N. The Maximum number allowed is determined by the MAXIMUM BUTTON command (<Esc>Nbb). The default maximum is 16, buttons 0 to 15. Only valid button numbers may be programmed.

BYTES USED : can be greater than bytes available, if the button program overwrites the next button.

Example

```
'This example demonstrates how to VIEW a valid
' button - in this case button 2

<Esc>B2?<CR>              ' to view button 2
 BTN 2:
100SE
[
200PL
[

BYTES USED :  16 / 255
Ok
```

BUTTON INPUT <Esc>Bbb=<CR>

Function

<Esc>Bbb= clears button bb and puts SuperBox into Program mode, and makes it ready to accept commands in the specified button number.

Syntax

```
<Esc>B(integer)=<CR>                                 sent to SB
<CR><LF>-                                      returned to host
```

Remarks

This command erases what was previously written in the specified button location. Sending an <Esc>B1=<CR><CR> would thus clear button # 1.
If the length of program stored in a button is longer than memory alotted for each button, the program simply runs over the memory alotted for the next button, and disables that next button.

If button 0 is so long that it overruns into button 1 the <Esc>B1= will recreate button 1, thus killing button 0.

Example

```
' This example demonstrates how to program a valid
' button - (in this case button 2)
' Once this is complete each time Button 2 is pushed
' these codes will be sent to the player

Note : the  -  indicates Program Mode

<Esc>B2=<CR>               ' erase and program button 2
 - 100SE<CR>               ' enter "search for frame100"
 - [<CR>                   ' wait for "R"
 - 200PL<CR>               ' enter "play to frame 200"
 - [<CR>                   ' wait for "R"
 - <CR>                    ' send <CR> to end button
                           ' program
```

BUTTON EDIT <Esc>Bbb-<CR>

Function

<Esc>Bbb- allows the user to change each character in the button.

Syntax

<Esc>B(integer)-**<CR>** sent to SB

Remarks

The BUTTON EDIT command first displays the entire program stored in the specified button. Then displays the first character and prompts for change. **Only one character may be entered per line.** If no change is to be made the <CR> will move to next character making no change. Send an <esc><CR> to exit program mode. or keep sending the <CR> (until the end of the program is reached) to exit program mode.

Sending a <Ctrl>X will truncate the command at the current character.

Example

```
' this example demonstrates how to CHANGE the search
' frame in button 2 to 123 from 100

note : In this case the - indicates Program Mode but
       only one character can be sent. And, not a
       string when using the <Esc>Bbb= command,

<Esc>B2?<CR>              ' to view button 2
 BTN 2:
100SE
[
200PL
[

1 - <CR>                  ' leave first chr. alone
0 - 2<CR>                 ' change 0 to 2
0 - 3<CR>                 ' change 0 to 3
S - <CR>                  ' send <CR> 13 times to leave
E - <CR>                  ' change mode
          .
          . 13 times    or use <Esc><CR> to exit
          .
            until exit
```

BUTTON APPEND <Esc>Bbb,<CR>

Function **<Esc>Bbb,** allows the user to add commands to the end of the specified button.

Syntax **<Esc>B**(integer),**<CR>** sent to SB

Remarks The BUTTON APPEND command first displays the entire program stored in the specified button. Then moves to the end of the button program and gives the program prompt, thus putting the SB back into Program mode. The button number (bb) can be an integer value between 0 and 99. The Maximum number allowed is determined by the MAXIMUM BUTTON command (<Esc>Nbb).

Example

```
' This example demonstrates how to ADD the goto
' button 0 after completing button 2

<Esc>B2,<CR>              ' to view button 2
   BTN 2:
  123SE
  [
  200PL
  [


- >0                      ' go to button 0
- <CR>                    ' end the APPEND
```

CLEAR ALL BUTTONS <Esc>CBTN

Function To clear all Button Programs

Remarks Once this command is sent the Unused Button flag is set to "FF" for all buttons. The "OK" prompt is displayed when this is complete.

Function **<Esc>E** dumps the memory of the EEROM to the host computer. This can be captured to a file and then sent, using Input EEROM, to another SuperBox

Syntax **<Esc>E<CR>** sent to SB

TURN CAPTURE ON NOW -

PRESS ANY KEY TO BEGIN EEROM DUMP returned to host

Remarks The dump function allows a direct memory dump. However, the program is almost impossible to edit in the HEX format, therefore the DUMP is mostly used to save a working program on disc -- use CrossTalk's CA ON at the Command line

The EEROM Dump command sends INTEL format HEX codes. Should the communication program miss a character the SuperBox uses a Checksum Error check and will not allow the program to be input back to the box.

The Intel format uses the first 9 characters to represent the memory address, the next 64 characters are HEX data, and the last two characters are the checksum.

Example 'This example demonstrates how to Dump

```
<Esc>E<CR>
TURN CAPTURE ON NOW -

PRESS ANY KEY TO BEGIN EEROM DUMP
:200000003E330DFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF7E
:20002000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFDF
:20004000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF3C323E43533144533150520D5B204407
:2000600049534E45592053494E4720414C4F4E47530D5B3250520D5B2A2A2A2A2A2A2A2ACE
:200080002A2A2A2A2A2A2A2A2A2A2A2A0D5B3450520D5B5E20534E4F572057484954450D4E
:2000A0005B3550520D5B5E20434C4153534943530D5B3650520D5B5E205745535445524E84
            .
            .
            .
            .
:201F40000401819000FF002C000400698100FF002BFF030400FF00FF001E00030300FF0000
:201F6000FF000100030290A200FF0023000301819000FF0022000300698100FF00210004C0
:201F80000400FF00FF0001000203AAC800FF000A00020290A200FF0032000201819000FF43
:201FA0000028000200698100FF001E00000800FF00FF0009000007ABB8007F0008000006E9
:201FC0009BAB007F00070000058A9B007F00060000047B8A007F0005000003697B007F0092
:201FE0000040000025869007F00030000014858007F00020000003348007F0001811C990242
:00000001FF

Ok
```

Function **<Esc>I** allows a previously dumped program to be loaded into the SuperBox from a computer.

Syntax **<Esc>I** sent to SB

BEGIN SENDING EEROM (INTEL HEX FORMAT) FILE NOW

Remarks

Example ' Demonstrates how to send a ????????.DMP file

```
<Esc>I<CR>
BEGIN SENDING EEROM (INTEL HEX FORMAT) FILE NOW
```

using the **SEnd** in CrossTalk type at the command line:
Command?**SE a:\SB42\DEMO.DMP**

then the following will scroll on the computer:

```
:200000003E330DFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF7E
:20002000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFDF
:20004000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF3C323E43533144533150520D5B204407
:2000600049534E45592053494E4720414C4F4E47530D5B3250520D5B2A2A2A2A2A2A2ACE
:200080002A2A2A2A2A2A2A2A2A2A2A0D5B3450520D5B5E20534E4F572057484954450D4E
:2000A0005B3550520D5B5E20434C4153534943530D5B3650520D5B5E205745535445524E84
            .
            .
            .
            .
:201F40000401819000FF002C000400698100FF002BFF030400FF00FF001E00030300FF0000
:201F6000FF000100030290A200FF0023000301819000FF0022000300698100FF00210004C0
:201F80000400FF00FF0001000203AAC800FF000A00020290A200FF0032000201819000FF43
:201FA0000028000200698100FF001E00000800FF00FF0009000007ABB8007F0008000006E9
:201FC0009BAB007F00070000058A9B007F00060000047B8A007F00050000003697B007F0092
:201FE0000040000025869007F00030000014858007F00020000003348007F0001811C990242
:00000001FF
```

VIEW ALL BUTTONS <Esc>M<CR>

Function **<Esc>M** shows programs in all enabled and valid buttons

Syntax **<Esc>M<CR>** sent to SB

Remarks The VIEW ALL command will show all buttons that are enabled, and contain valid command data.

Example 'This example demonstrates how VIEW ALL BUTTONs

```
<Esc>M<CR>                    ' to view all

 BTN 0:
CS120SP100SE
[
<1>>90

 BTN 1:
$1=1$2=1CS200SE
[
<4>!400!>0
.
.
.
 BTN 98:
$0=123>12
```

MAX. NUMBER OF BUTTONS **<Esc>N**(argument)**<CR>**

Function **<Esc>N** creates a specified number of buttons, and allocates amount of memory available in each button.

Syntax **<Esc>N**(integer)**<CR>** sent to SB

Remarks The MAXIMUM NUMBER OF VALID BUTTONS Sets the maximum number of buttons allowed and the number of bytes of memory allowed per button. Allowable values for (argument) are 0,1,2,3,4,5,6.

The MAXIMUM NUMBER OF VALID BUTTONS to be set must be done before programming sessions. This function determines how much memory is allocated to the buttons.

$$\text{Button Memory} = \frac{\text{Available Memory (2k or 8K)}}{\text{Maximum Number of Buttons}}$$

So the more buttons that are used the less memory is available to each as seen here:

Value of	Maximum	Number of bytes per button		
N	# buttons	2 K option	8 K option	16 K option
6	2000	--	--	
5	1000	--	--	15
4	500	--	15	31
3	250	--	31	80
2	100	15	80	255
1	32	64	255	511
0	16 (minimum)	128	511	1022

Example

```
' Maximum is set to default of 16 buttons
 ' but gets changed here.

<Esc>N1<CR>                      ' set valid buttons
32 BUTTONS ENABLED ( 0 - 31 )
255 BYTES / BUTTON                  ' with 8k option
```

CLEAR USAGE LOG Clock option **<Esc>C<CR>**

Function

<Esc>C clears the usage log, if the clock option is installed, otherwise an error message is returned.

Syntax

```
<Esc>C<CR>                              sent to SB
<CR><LF>                          returned to host
```

Remarks

The CLEAR USAGE LOG command allows user to reset the log when needed. The usage log remembers the button pushes for the last 150 days.

Example

```
'This example demonstrates how to CLEAR usage log

<Esc>C<CR>            ' to clear
```

DATE Clock Option **<Esc>D<CR>**

Function

<Esc>D displays date and prompts for change

Syntax

```
<Esc>D<CR>                              sent to SB
```

Remarks

The DATE function only works if Clock option is installed, the format is mm/dd/yy, and invalid formats return error messages.

Example

```
'This example demonstrates how to set the DATE

<Esc>D<CR>                    ' to set date
CURRENT DATE : 01/01/87       ' hit <CR> to keep this date or
ENTER NEW DATE : 04/04/87<CR> ' sets new date and

CURRENT TIME : 17:40          ' shows time
```

LOG TITLE

Clock Option **<Esc>L**(string)**<CR>**

Function

<Esc>L To give the usage log a title

Syntax

```
<Esc>L(string)<CR>                             sent to SB
<CR><LF>                                 returned to host
```

Remarks

The LOG TITLE only works if Clock option is installed. Twenty characters is the maximum length of the title. The title will appear whenever the usage log is viewed

Example

'This example demonstrates how to set the LOG Title

```
<Esc>L<CR>
ENTER TITLE : KIOSK #3 in LA<CR>     ' to set title to read
                               ' "KIOSK #3 IN LA"

' to clear the title
<Esc>L<CR>
ENTER TITLE :  space <CR>
```

TIME

Clock Option **<Esc>T<CR>**

Function

<Esc>T displays TIME and prompts for change

Syntax

```
<Esc>T<CR>                                       sent to SB
```

Remarks

The DATE function only works if Clock option is installed. The format is mm/dd/yy, and invalid formats are ignored.

Example

'T%his example demonstrates how to set the time

```
<Esc>T<CR>                   ' to set TIME
CURRENT TIME : 17:40<CR>    ' hit <CR> if OK
ENTER NEW TIME : 12:00<CR>   ' sets new time

CURRENT DATE : 4/4/87

CURRENT TIME : 12:00
```

Function **<Esc>U** give usage summary for enabled buttons. Available with clock option only.

Syntax **<Esc>U<CR>** sent to SB

Remarks The VIEW USAGE LOG command will show a tally of all buttons pushed since the log was last cleared. The number of pushed buttons during morning, afternoon and evening are also tallied.

Example 'this example demonstrates howto view the log

```
<Esc>U<CR>                            ' to view usage log
USAGE SUMMARY FOR :  KIOSK # 3 IN LA
 BTN  0: 0004                          ' button 0 pushed 4 times
 BTN  1: 0154                          ' button 1 - 154 times
 BTN  2: 0000                          ' etc.....
 BTN  3: 0008
     .
     .
     .
 BTN   #:  0014                       ' here "#" is the last
                                      ' button enabled using
                                      ' <Esc>N
04/04/87  0338 0000 0000              ' 338 pushed in morning
                                      ' 0 pushed in afternoon
                                      ' 0 pushed in evening
```

SET AUTO-EXECUTE Clock Option **<Esc>Xxx<CR>**

Function **<Esc>X**bb allows enabled buttons to be given times to start playing.

Syntax **<Esc>X**(integer)**<CR>** sent to SB

Remarks The SET AUTO-EXEC. command is used to have a predetermined program (defined by a button) start up and play any a given time of day. When such a time occurs the SB42 will wait until a button is finished executing before it will send the auto-execute button, specified in the <Esc>X function.
There are a maximum of 16 autoexecute functions.

Example ' This example demonstrates how to SET Auto-execute

```
<Esc>X15<CR>                         ' SET AUTO EXEC # 15
AUTO #15
  TIME : 12:40<CR>                   ' TO INTERRUPT AT 12:40
  BTN : 2                            ' DO BUTTON # 2
```

VIEW AUTO-EXECUTE Clock Option **<Esc>X?<CR>**

Function **<Esc>X?** asks the SB to display the Auto-execute functions currently enabled.

Syntax **<Esc>X?<CR>** sent to SB

Remarks TheVIEW AUTO-EXEC command shows the user what is stored in all (16 maximum) autoexecute functions.

Example ' This example demonstrates how to VIEW Auto-execute

```
<Esc>X?<CR>
AUTO # 15 TIME : 12:40 BTN : 2
```

CREATE MASK TS option **<Esc>Smm=<CR>**

Function

<Esc>Smm= is used to create touch screen areas of touch and assign branch to locations when a valid area is touched.

Syntax

<Esc>Sm,s=x1,y1,x2,y2,b<CR>
m - integer that represents the mask number
s - integer that represents the sector number
x1- integer coordinate for the top left corner
y1- integer coordinate for the top left corner
x2- integer coordinate for the bottom right corner
y2- integer coordinate for the bottom right corner
b - integer that gives the button to branch to

The range for x is 0-255
The range for y is 0-255

Upper left Co-ordinates are x=0 y=0
Lower right Co-ordinates are x=255 y=255

Remarks

Mask memory is determined by using <Esc>Zbb. If more mask memory is needed (MASK MEMORY FULL) then another button must be used.

If sector areas exist then the low sector number has precedence over the larger numbers.

All buttons should have an associated mask.

The last sector of each mask should include the entire screen (0,0,255,255) and go to a valid button. This will ensure accidental touches during a segment have a branch to location.

Example

```
' This example shows how to set mask 1  to be the
' whole screen . When screen is touched button 1 is
' performed

<Esc>S1,0=0,0,255,255,1
```

VIEW ALL MASKS TS Option **<Esc>MM<CR>**

Function **<Esc>MM** shows all enabled touch screen MASK sectors

Syntax **<Esc>MM<CR>** sent to SB

Remarks View all Masks will show the data stored for each mask. UL is the Upper Left corner, and LR is the Lower Right corner. BTN = shows which button is to be performed when the sector of the enabled mask is touched. Sector 0 has priority over sector 1 and sector 1 has priority ovewr sector 2, etcetera, should the sectors overlap.

Example

<Esc>MM<CR>

```
MASK 0:
 SECTOR 0 : UL=(0,51) ; LR=(127,72) ; BTN = 1
 SECTOR 1 : UL=(0,72) ; LR=(127,88) ; BTN = 2
 SECTOR 2 : UL=(0,88) ; LR=(127,105) ; BTN = 3
 SECTOR 3 : UL=(0,105) ; LR=(127,123) ; BTN = 4
 SECTOR 4 : UL=(0,123) ; LR=(127,138) ; BTN = 5
 SECTOR 5 : UL=(0,138) ; LR=(127,155) ; BTN = 6
 SECTOR 6 : UL=(0,155) ; LR=(127,171) ; BTN = 7
 SECTOR 7 : UL=(0,171) ; LR=(127,184) ; BTN = 8
 SECTOR 8 : UL=(0,0) ; LR=(255,255) ; BTN = 9

99 TOTAL MASK SECTORS ENABLED
35 MASK SECTORS FREE
```

VIEW MASK MEMORY TS Option <Esc>Z

Function **<Esc>Z** Shows the mask memory allocation.

Example:

```
99 TOTAL MASK SECTORS ENABLED
35 MASK SECTORS FREE
```

Ok

SET MASK MEMORY TS Option <Esc>Zbb<CR>

Function **<Esc>Zbb** allocates from button **bb** to button max. (as set by <Esc>Nn) as the memory area for touch screen sectors.

Remarks When the Maximum number of buttons is changed make sure the Mask Memory is not affected.

ADD and SUBTRACT + and -

Function

The Plus and Minus signs have different functions depending on what preceeds the symbol.

1. +/- precedes a number and a player command then the current frame number is remembered and the value of the number is added or subtracted and the player command is executed.
2. +/- is also used to add the value two variables.

Syntax

1. **+10SE** searches current frame plus 10
2. **+C$6SE** adds value of variable 6 to present chapter number. If a chapter is valid then the player searches for it.
3. **$7=$6+$1** adds contents of 1 and 6 and stores the value in variable 7.

Remarks

The +/- can be located anywhere in the program string. They cannot be used in pass-thru mode

Example

See Appendix IV, page 60, see Button 7 in the sample program.

FRAME, CHAPTER or TIME GRAB +x and -x

Function

The +/- can also be used to get the value of the current frame number, chapter number or time code number. This number may be incremented decremented, or variables added/subtracted from it.

Syntax

+, +F, +C, +T and -, -F, -C, -T

Remarks

If F,C,T are omitted the default is F, thus a frame grab will occur.

Example

Appendix IV page 63, button 20

```
+50SE        ' searches current frame plus 50
+T$4SM       ' places a stop marker at the current
             ' time code plus the value in variable 4
-C3SE        ' searches to current chapter minus 3
```

DELAY |ddddd|

Function: |ddddd| gives a delay in tenth of seconds before issuing the next command.

Syntax: |ddddd| -ddddd represents tenths of seconds from 0 to 65535.

Remarks: The DELAY function is used to create delays between commands. The largest value that can be specified is 65535 . Each dot viewed on the computer during the delay command represents 1 second.

Example:

```
100SE            ' search for frame 100
[                      ' wait for "R"
|35| SF                ' wait 3.5 secs. then step forward
[                      ' wait for "R"
```

DISABLE BUTTON :

Function: : is used to disable all or some of the pins on the user interface (UI). Thus the associated button programs will not execute.

Syntax: :bbi.bbf, bbi is the first button in range
bbf is the last button in range to be disabled.

Remarks: When a button is executed either from the button pad or a touch screen, the button program is automatically DISABLED. This is done to ensure that a button cannot be pushed while the Superbox is waiting for the Player ready status. This is especially true when using the text overlay capabilities of the LD-V4200.

To interrupt a command sequence the UI must be ENABLED first.

Once the button is disabled it can always be executed from the terminal.

Example: see ENABLE BUTTON

ENABLE BUTTON ;

Function ; is used to ENABLE all or some of the button programs. This allows the viewer to interrupt the current video segment by using the user interface.

```
Syntax      ;bbi.bbf,    bbi is the first button in range
                         bbf is the last button in range to be
                         enabled.

Examples    ;,           Enables all button programs
            ;1,          Enables button # 1
            ;1.4,        Enables buttons 1 through 4
            ;,:3,        Enables all buttons, then disables button # 3
            ;1,;10.20    Enables buttons 1,10 to 20

            :,           Disables all button programs
            :1,          Disables button # 1
            :1.5,        Disables buttons 1 through 5
            :,;4,        disables all buttons, then enables Button # 4

            BTN 1:
            ;1,;3.6,100SE              '  buttons 1,3,4,5,6 enabled
            [
            150MF
            [
            BYTES USED : 22 / 255

            BTN 2:
            ;1.2,200SE                 ' buttons 1,2 enabled
            [
            250MF
            [
            BYTES USED : 19 / 255

            BTN 3:
            ;,:1.3,300SE               ' buttons 1,2,3 disabled
            [                          ' all others enabled
            350MF
            [
            BYTES USED : 21 / 255

            BTN 4:
            :,;5.31,400SE              ' buttons 5 thru 31 enabled
            [                          ' all others disabled
            450MF
            [
            BYTES USED : 22 / 255

            BTN 5:
            ;5.6,500SE                 ' buttons 5,6 enabled
            [
            550MF
            [
```

LOOP #ddddd(string)

Function #ddddd(string) will execute the string the number times specified by the Argument.

Syntax

```
#ddddd(string) - ddddd represents number of times
                   to loop.
                 - string can be any player command or
                   SuperBox function.
```

Remarks The LOOP function can not be nested. A variable cannot determine the number of time to loop.

Example

```
' this example shows how step through 10 frames with
 ' a 5.0 second pause on each one

#10(↓50↓SF                  ' wait5.0 sec. and step fwd.
[                           ' wait for "R"
)                           ' repeat 10 times
```

SUBROUTINE &bb

Function **&bb** will execute the string stored in the button specified by the Argument, and then return to the next command or function.

Syntax **&bb** bb represents the BUTTON to be executed.

Remarks The SUBROUTINE function is to be used for repetitive commands. It can not be nested.

Example

```
' this example shows how step through 10 frames with
' a 5.0 second pause on each one. using  button 1 as
' the button to be pushed and button 30 as the
' subroutine.
<Esc>B1=<CR>               ' to program button 1
- 100SE<CR>                ' search frame 100
- [<CR>                    ' wait for "R"
- &30<CR>                  ' do button 30
- 123SE<CR>                ' search frame 123
- [<CR>                    ' wait for "R"
- <CR>                     ' end program for button 1

<Esc>B30=<CR>              ' to program button 30
- 150!SF<CR>                ' wait 5.0 sec. then step fwd.
- [<CR>                    ' wait for "R"
- <CR>                     ' end program for button 30
```

GOTO >bb

Function >bb will execute the string stored in the button specified by the Argument.

Syntax >bb - bb represents the BUTTON to be executed.

Remarks The GOTO function is to be used to branch to the specified button. The only logical spot to use this function is at THE END of a button program.

Example

```
' This example shows how to step back and forth between
' two frames using STEP FORWARD and STEP REVERSE with
' 1/2 second delays between each.

<Esc>B28=<CR>                ' to program button 28
- ↓5↓SF<CR>                  ' wait .5 sec., step fwd
- [<CR>                      ' wait for "R"
- >29<CR>                    ' goto button 29
- <CR>                       ' end program for button 1


<Esc>B29=<CR>                ' to program button 29
- ↓5↓SR<CR>                  ' wait.5 sec.and step Rev.
- [<CR>                      ' wait for "R"
- >28<CR>                    ' goto button 28
- <CR>                       ' end program for button 30
```

This technique can be used to create two screens of menu where choices on screen 1 would be buttons 1-8 and choices on screen two could be 9-16. A third screen could be created saying something like "PLEASE MAKE A CHOICE"
"PRESS A BUTTON"
"1 - 16"

WAIT FOR RESPONSE [

Function [waits for the player to return an "R" or some other return code such as error codes.

Purpose The reason for this function is that the videodisc player will perform commands as they are sent, thus, in order not to interrupt a command the SuperBox must Wait for the player response.

MASK <mm>

Function <mm> sets the touch screen mask to be used

Syntax <mm> mm is the mask that is now enabled

Remarks This function is only valid in command (CMD) mode. It sets the mask that will be enabled when the SuperBox returns to ETS mode.

USER OUTPUT CONTROL ~pb

Function ~pb is used to control external devices, such as video tape recorders, video switches or whatever can be imagined.

Syntax ~pb , p is the output pin 0, 1, 2, 3
b is the logic level 0 or 1

Remarks When the SuperBox is powered up pins 9, 10, 28, 33 on the User Interface (UI) are TTL logic level high, thus they have a voltage reading of 5 Volts.

p=0 refers to the UI pin 9
p=1 refers to the UI pin 10
p=2 refers to the UI pin 28
p=3 refers to the UI pin 33

Example

```
BTN 1:
100SE              ' SEARCH FOR 100
[
~30,200PL          ' pin 33 on UI turned low - ground
[
~31,>2             ' pin 33 on UI turned high - 5 volts
```

RECALL $n

Function

$n allows the value of the variable to be used as the address or argument used by any of the player commands or by the SB42 GOTO function.

Example

```
' this example shows how a variable can be used to
' search a frame number

<Esc>B0=
- $1=10<CR>                  ' assigns 10 to variable 1
- $1SE<CR>                   ' searches to frame 10
- [<CR>                      ' wait for "R"
- $1=$1+1,4PR
- [
- VARIABLE 1 = $1            ' will print Variable 1 = 11
- [                          ' on line 4
- >$1                        ' goto button 11
```

STORE $Vn=

Function

$Vn= is used to set the value of the variable specified by the argument.

Remarks

The STORE functions may be placed anywhere on the program string. They must be ended with a comma. For readability they can be separated by commas but this uses up memory unneccessarily.

Example

```
$1=10$2=11$3=12,        ' stores 10 in variable 1
                        '        11             2
                        '        12             3
```

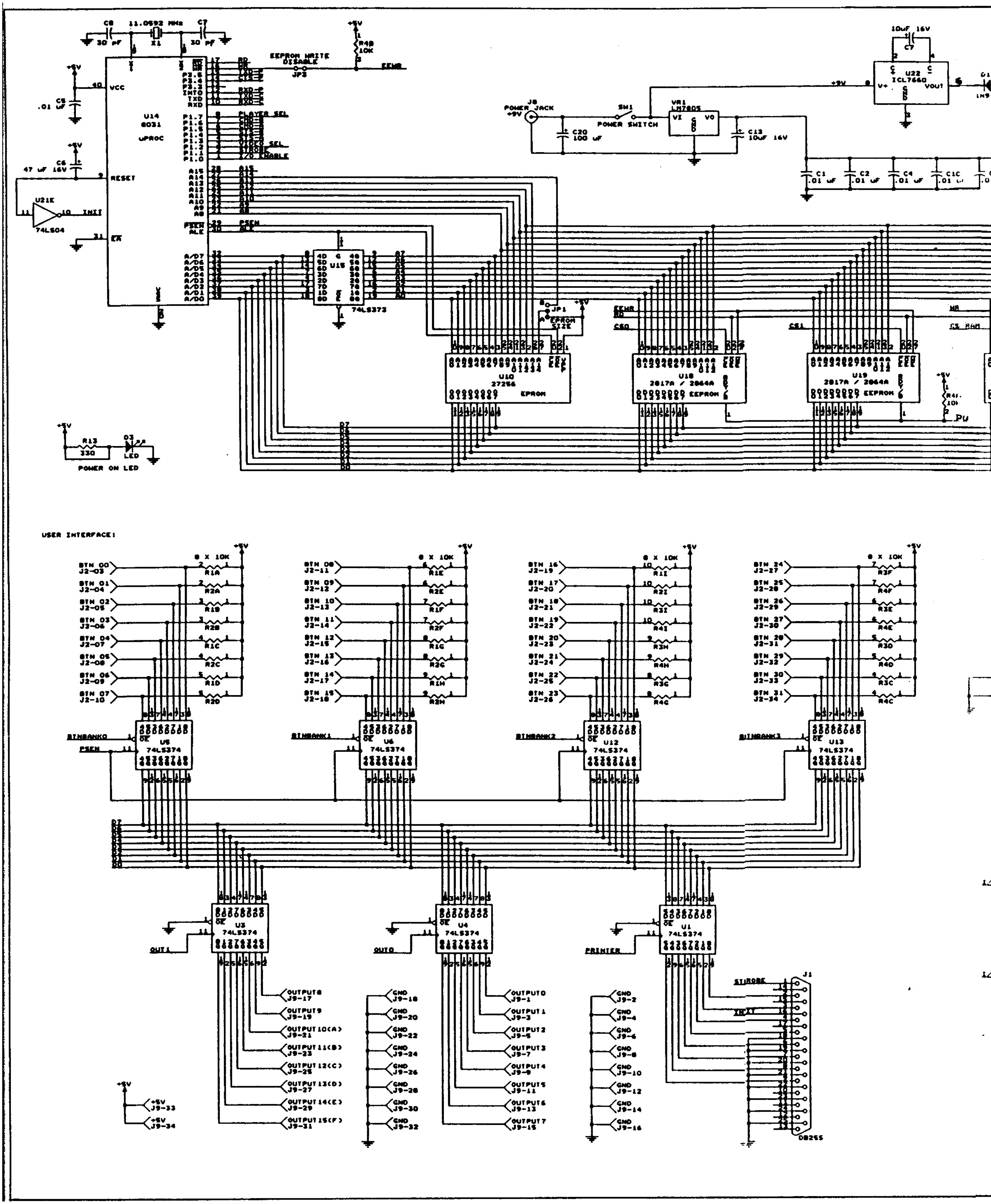
Pages 38-42 are omitted (obsolete).

The SuperBox has some features that help with troubleshooting. When power is applied to SB42 and the LD-V4200 at the same time, the SB42 does the following:

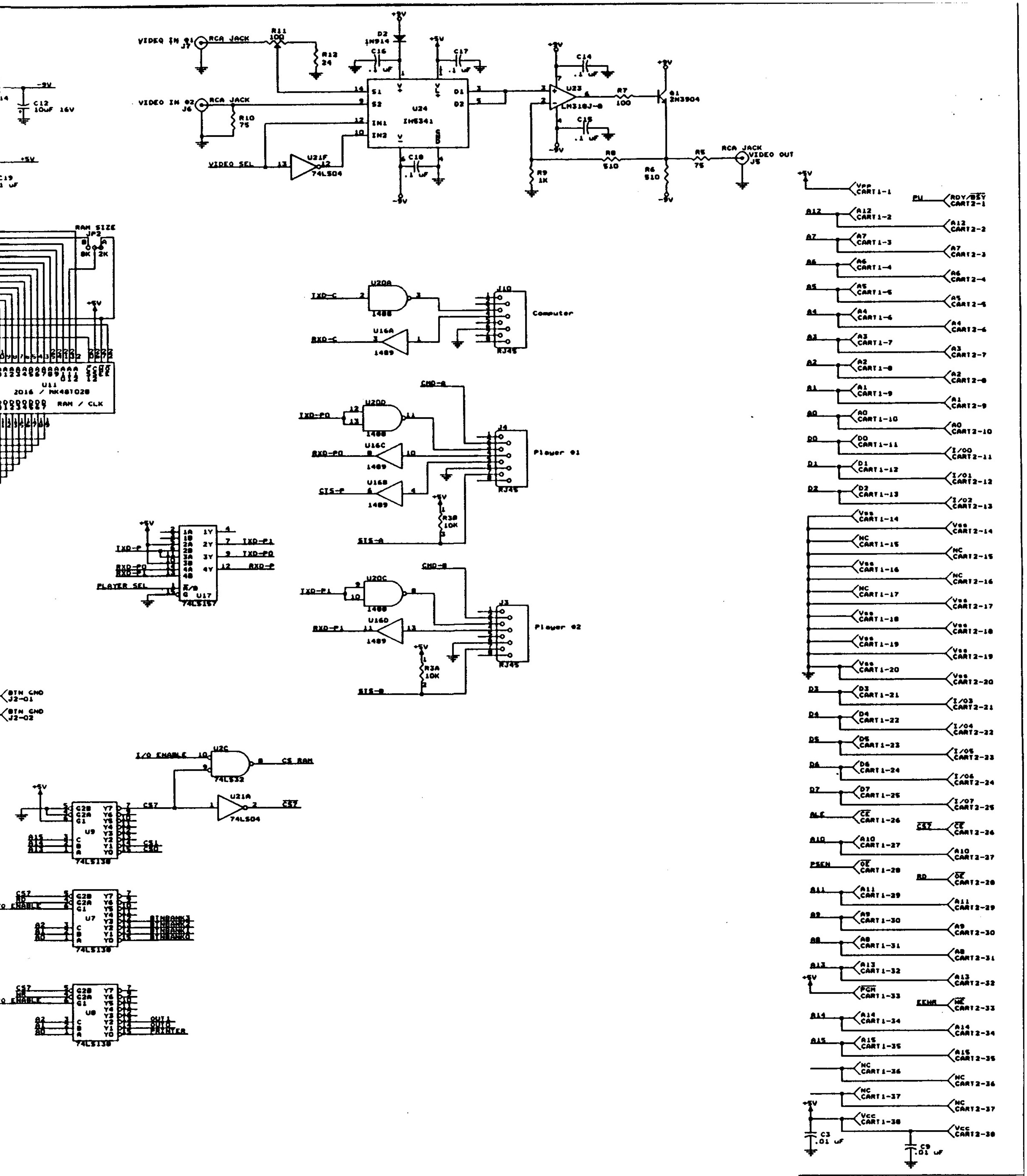
a. Power LED should be on when power is applied
b. waits 5 seconds for disc to set
c. sends SA command to spin up player
d. prints "SPIN UP" on the screen until play light on the front panel lights up.
e. prints VDS introduction and our phone number (408)-438-8396
f. if clock option is present then time and date are displayed
g. then button 0 is performed

If none of this happens try the following:

1 Check that there is a working videodisc in player, as damaged discs may not work properly.
2 Check that videodisc player operates without SB42.
3 Check Video cable is connected to player and monitor.
4 Check that the function switches on front panel of LD-V4200 are all up.
5 Check SB/LD-V4200 cable is securely attached to player and the SB42 Ext. Player Control port (the outside DB-9 connector).
6 Check Power supply voltages - see Appendix I.



C8
11.0592 MHz
C7
30 pF
X1
30 pF
+5V
R48
10K
EEPROM WRITE DISABLE
JP3
EEWR
C5
.01 uF
VCC
U14
8031
uProc
RD
WR
P3.5
P3.4
P3.3
P3.2
INT0
TXD
RXD
P1.7
P1.6
P1.5
P1.4
P1.3
P1.2
P1.1
P1.0
PLAYER SEL
VIDEO SEL
STROBE
I/O ENABLE
C6
47 uF 16V
RESET
U21E
74LS04
INIT
EA
A15
A14
A13
A12
A11
A10
A9
A8
PSEN
ALE
A/D7
A/D6
A/D5
A/D4
A/D3
A/D2
A/D1
A/D0
U15
74LS373
JP1
EPROM SIZE
EEWR
RD
CS0
CS1
WR
CS RAM
U10
27256
EPROM
U18
2817A / 2864A
EEPROM
U19
2817A / 2864A
EEPROM
D7
D6
D5
D4
D3
D2
D1
D0
R13
330
D3
LED
POWER ON LED
J8
POWER JACK
+9V
SW1
POWER SWITCH
VR1
LM7805
VI
VO
C20
100 uF
C13
10uF 16V
10uF 16V
C7
U22
ICL7660
V+
VOUT
+9V
C1
.01 uF
C2
.01 uF
C4
.01 uF
C1C
USER INTERFACE:
8 X 10K
BTN 00 J2-03
BTN 01 J2-04
BTN 02 J2-05
BTN 03 J2-06
BTN 04 J2-07
BTN 05 J2-08
BTN 06 J2-09
BTN 07 J2-10
R1A
R2A
R1B
R2B
R1C
R2C
R1D
R2D
BTN 08 J2-11
BTN 09 J2-12
BTN 10 J2-13
BTN 11 J2-14
BTN 12 J2-15
BTN 13 J2-16
BTN 14 J2-17
BTN 15 J2-18
R1E
R2E
R1F
R2F
R1G
R2G
R1H
R2H
BTN 16 J2-19
BTN 17 J2-20
BTN 18 J2-21
BTN 19 J2-22
BTN 20 J2-23
BTN 21 J2-24
BTN 22 J2-25
BTN 23 J2-26
R1I
R2I
R3I
R4I
R3H
R4H
R3G
R4G
BTN 24 J2-27
BTN 25 J2-28
BTN 26 J2-29
BTN 27 J2-30
BTN 28 J2-31
BTN 29 J2-32
BTN 30 J2-33
BTN 31 J2-34
R3F
R4F
R3E
R4E
R3D
R4D
R3C
R4C
BTNBANK0
PSEN
U5
74LS374
BTNBANK1
U6
74LS374
BTNBANK2
U12
74LS374
BTNBANK3
U13
74LS374
OE
U3
74LS374
OUT1
U4
74LS374
OUT0
U1
74LS374
PRINTER
OUTPUT8 J9-17
OUTPUT9 J9-19
OUTPUT10(A) J9-21
OUTPUT11(B) J9-23
OUTPUT12(C) J9-25
OUTPUT13(D) J9-27
OUTPUT14(E) J9-29
OUTPUT15(F) J9-31
GND J9-18
GND J9-20
GND J9-22
GND J9-24
GND J9-26
GND J9-28
GND J9-30
GND J9-32
OUTPUT0 J9-1
OUTPUT1 J9-3
OUTPUT2 J9-5
OUTPUT3 J9-7
OUTPUT4 J9-9
OUTPUT5 J9-11
OUTPUT6 J9-13
OUTPUT7 J9-15
GND J9-2
GND J9-4
GND J9-6
GND J9-8
GND J9-10
GND J9-12
GND J9-14
GND J9-16
+5V J9-33
+5V J9-34
STROBE
INIT
J1
DB25S



VISUAL DATABASE SYSTEM

614 Bean Creek Road
Scotts Valley, Ca 95066
(408)438-8396

Title

SB-2 SUPERBOX CONTROLLER