RS-232 Control
of the
Advantage MSP11/MSP22

Biamp Systems, 10074 S.W. Arctic Drive, Beaverton, Oregon 97005 U.S.A. (503) 641-7287
an affiliate of Rauland-Borg Corp.

Introduction

This document contains information for the serial control of the Advantage MSP11 and the Advantage MSP22. Specifically, this document tries to inform those looking to write their own software controls for the Advantage MSP (in this document, the term "Advantage MSP" refers to both the MSP11 and MSP22). It is assumed that the reader has some familiarity with standard programming practices, binary and hexadecimal numbers, the ASCII character set, asynchronous serial data connections, and RS-232 interfaces.

Decimal, Binary, and "Pseudo-hex" Numbers

This document uses three different numerical notations. The first is the decimal notation. Whenever it is used, a "d" will appear after the number..

8 Bit binary numbers are the second format used in this paper. These numbers will be followed by "b" after their usage. If a specific bit is being referred to, the numbers will be preceded by the word "bit."

To transmit an 8 bit binary number to the Advantage MSP, hexadecimal notation is used. Hexadecimal numbers are arrived at by splitting the number into two halves. One half consists of the first four binary digits (most significant nibble) while the other consists of the last four binary digits (least significant nibble). 2 nibbles form a byte, which takes on a decimal value of 0 to 255. Each half is then assigned a hexadecimal value. Since the binary values range from 0 to 15, usually values from 10 to 15 are given the alphabetic letters from A to F.

However, the Advantage MSP does not utilize standard hex format. Instead, the Advantage MSP uses what is known as "pseudo-hex." Simply put, instead of using the letters A, B, C, D, E and F the Advantage MSP uses : ; < = > and ?, respectively. All it takes to arrive at the new notation for hex values 10 to 15d is to add 30 to the old ASCII values. In this paper, [pseudo-hex] will appear after the use of a pseudo-hex character. The changes are traditional hex are summed up below:

	Nibble Conversion		
Decimal	Hex	Pseudo-hex	Binary
0	0	0	0000
1	1	1	0001
2	2	2	0010
3	3	3	0011
4	4	4	0100
5	5	5	0101
6	6	6	0110
7	7	7	0111
8	8	8	1000
9	9	9	1001
10	A	:	1010
11	B	;	1011
12	C	<	1100
13	D	=	1101
14	E	>	1110
15	F	?	1111

Serial Interface - Data Communications Parameters

The Advantage MSP communicates through its serial port at four different baud rates: 2400, 9600, 19200, and 38400. The factory default setting is 9600 baud. Changing this rate is accomplished through BiampWin. The Advantage MSP communicates with 8 data bits, no parity, and 1 stop bit. The Advantage MSP utilizes a subset of the standard 7-bit ASCII character set.

Control

The Advantage MSP has an RS-232-compatible serial port which allows it to be controlled by a computer or by a third party system controller (such as those provided by AMX® or Crestron®). The Advantage MSP offers the following two methods of serial control:

- Control Button Emulation. This method of control emulates Biamp's standard infrared remote control transmitter or wall-mount remote control panel. Using this method, single ASCII characters sent to the device's serial port cause the device to behave as if a Biamp remote controller were attached. While Control Button Emulation is simple to perform, it only provides basic and "one-way" control of the Advantage MSP - it allows the user to send simple commands *to* the Advantage MSP, but it does not provide any mechanism for requesting status information *from* the Advantage MSP.

- Advanced Control. Advanced control provides a command set which allow "two-way" control of the Advantage MSP. Using Advanced Control commands, a system may request status information *from* the device as well as send commands *to* the device. Communication occurs with the Advantage MSP using the Advantage MSP's serial port.

Control Button Emulation

Control Button Emulation is the simplest form of serial control of the Advantage MSP. This method of operation allows the user to emulate the operation of a standard Biamp remote control transmitter.

For each button on a standard Biamp remote control, there is a corresponding ASCII character. In order to emulate a remote control button, the transmitting system simply transmits the corresponding ASCII character to the Advantage MSP's serial port. Each character received by the Advantage MSP will be echoed back out the serial port.

The standard Biamp remote control devices never exceed a transmission rate of 9 characters per second. If the controlling system wishes to perform Control Button Emulation at a rate of greater than 20 characters per second (50 msec per character), flow control should be implemented by waiting for the echo of each character before transmitting the next character. At slower speeds, flow control should not be necessary.

The following table summarizes the ASCII character codes for Control Button Emulation corresponding to each of the 40 remote control buttons supported by the Advantage MSP. These button codes are also summarized on the ASCII code chart provided at the end of this manual. The remote control buttons on the standard Biamp transmitter are numbered from left to right going from bottom to top with the lower left-hand button being button number 1.

Using BiampWin, it is possible to program the MSP to respond to these commands.

button 1	'B'	(0x42)		button 21	'V'	(0x56)
button 2	'C'	(0x43)		button 22	'W'	(0x57)
button 3	'D'	(0x44)		button 23	'X'	(0x58)
button 4	'E'	(0x45)		button 24	'Y'	(0x59)
button 5	'F'	(0x46)		button 25	'Z'	(0x5A)
button 6	'G'	(0x47)		button 26	'['	(0x5B)
button 7	'H'	(0x48)		button 27	'\'	(0x5C)
button 8	'I'	(0x49)		button 28	']'	(0x5D)
button 9	'J'	(0x4A)		button 29	'^'	(0x5E)
button 10	'K'	(0x4B)		button 30	'_'	(0x5F)
button 11	'L'	(0x4C)		button 31	'`'	(0x60)
button 12	'M'	(0x4D)		button 32	'b'	(0x62)
button 13	'N'	(0x4E)		button 33	'c'	(0x63)
button 14	'O'	(0x4F)		button 34	'd'	(0x64)
button 15	'P'	(0x50)		button 35	'e'	(0x65)
button 16	'Q'	(0x51)		button 36	'f'	(0x66)
button 17	'R'	(0x52)		button 37	'g'	(0x67)
button 18	'S'	(0x53)		button 38	'h'	(0x68)
button 19	'T'	(0x54)		button 39	'i'	(0x69)
button 20	'U'	(0x55)		button 40	'j'	(0x6A)

Simple vs Addressable

The simple method of control button emulation is to send any one of the control button characters through the serial port to the MSP. The disadvantage to this method is that every device hooked into the MSP will also hear the command. If any of the other devices have been programmed with this particular character, they will also respond.

To avoid this problem, the MSP allows addressable control button emulation. By using the control-button-emulation command, on page 22, control button commands are sent directly to a specific device.

Advanced Control

The Advanced Control command set includes more powerful commands to allow more flexible control of the Advantage MSP. Unlike Control Button Emulation (which is basically a one-way control mechanism) advanced control commands allow the MSP to return information through the serial port,. The following list summarizes the commands available using Advanced Control, including the ASCII command character associated with each command:

!	execute-command *(execute selected command)*
.	set-baud *(set communications speed)*
+	sleep-for-10-seconds *(sleep for 10 seconds, ignoring all communication)*
/	get-version *(retrieve the model information and firmware version date)*

Each Advanced Control command requires at least two parameter bytes (four pseudo-hex characters) to be sent prior to the command character. Each command will be explained in detail on the following pages.

The MSP differ from some other Advantage products in that it uses one command byte (the '!' character) to control several different functions. The execute-command incorporates the utility of several commands by having different parameters control its function.

Some of the commands cause the Advantage MSP to return information through the serial port. For each string of information returned to the serial port, the Advantage MSP terminates the string by transmitting the ASCII carriage return character (0x0D - represented in this document as ↵).

As mentioned earlier, the Advantage MSP will echo all characters it receives, regardless of whether or not the characters are valid commands or parameters. Characters greater than 0x7F are reserved and should not be transmitted to the serial port. The Advantage MSP utilizes a subset of the standard ASCII character set. The following characters have meaning to the Advantage MSP:

character	hexadecimal	operation
ASCII control characters	(0x00 - 0x1F)	no operation
ASCII **SPACE** character	(0x20)	no operation
! thru **/**	(0x21 - 0x2F)	Advanced Control commands
0 thru **?**	(0x30 - 0x3F)	pseudo-hex parameters for Advanced Control commands
@	(0x40)	Control Button Emulation Repeat Code
A	(0x41)	no operation
B thru **`**	(0x42 - 0x60)	Control Button Emulation commands (buttons 01 - 31)
a	(0x61)	no operation
b thru **j**	(0x62 - 0x6A)	Control Button Emulation commands (buttons 32 - 40)
k thru **z**	(0x6B - 0x7A)	no operation
{ thru **DEL**	(0x7B - 0x7F)	no operation
0x80 thru 0xFF	(0x80 - 0xFF)	RESERVED

Device Type Bitmask, Device Number Bitmask, and Device Model Bitmask

In a system which has more than one Advantage product connected together, the device type bitmask and device number bitmask command parameters provide a mechanism to individually address a particular device (or a combination of devices). Every command in the advanced control command set requires that a device type bitmask and a device number bitmask be transmitted as the last two parameter bytes before transmitting the command character itself. These two bitmask parameters bytes provide a device addressing capability to specify which of the devices in the system should execute the command. All devices which are not specifically addressed by these two bitmask values will ignore the command.

The device type bitmask parameter byte supports up to eight distinct device types - one bit per device type. The eight device types are:

0x01 [hex] (bit 0) Biamp Advantage DRC 4+4 digital remote control
0x02 [hex] (bit 1) Biamp Advantage EQ28X digitally-controlled graphicEQ
0x04 [hex] (bit 2) Biamp Advantage SPM522D stereo preamp/mixer
0x08 [hex] (bit 3) Biamp Advantage PMX84 programmable matrix switch
0x10 [hex] (bit 4) (reserved for future products)
0x20 [hex] (bit 5) (reserved for future products)
0x40 [hex] (bit 6) (reserved for future products)
0x80 [hex] (bit 7) Advanced Products, such as the Biamp Advantage MSP

The Advantage MSP will only respond to advanced control commands if bit 7 of the device type bitmask parameter byte is a '1'. A command may be directed to more than one device type in the system by setting all of the corresponding bits in the device type bitmask to '1's. If only advanced equipment is being addressed (EQ2828/8 DRI, MSP, and DDL12) 80 is the only bitmask required to use.

The device number bitmask parameter byte supports up to sixty-four distinct device numbers:

0x00 [hex] Select Device Number 0
0x01 [hex] Select Device Number 1
0x02 [hex] Select Device Number 2
0xFF [hex] Select Device Number 63

A particular Advantage MSP will only respond to advanced control commands if the device number bitmask parameter byte corresponds to its own device number.

For instance, the bitmask 8007 serves to talk only to advance product (**80**) number 7 (**07**).

The device model bitmask is a special number that is reserved exclusively for the Advantage MSP. Unlike the device type bitmask, which can refer to an entire line of products, the device model bitmask is reserved just for one device. To retrieve this setting, the get-version command can be used.

! execute-MSP-command

description:

The execute-MSP-command byte causes the MSP to change its operating parameters. To control each of the varied functions of the MSP, there are several MSP command numbers that are associated with the execute-MSP-command. In the standard format, the MSP command number is the 4 bytes preceding the checksum , device type, and device model bitmasks. The command has control over the gain manager, the input and output levels, the crossovers, delays, gain matrix, and equalizer.

0x28 → 0x2A gain manager bypass commands

description:

These commands are used to control the gain manager function, allowing simple ways to turn on and off individual sections of the GM.

syntax of commands:

The commands in this section share the following parameters:

zz = device number
yy = reserved for checksum (no action parameter)
aa = channel number (01 or 02)
bb = byte containing command status
(bit set = under command control, bit cleared = ignored by command)
bit 0 - reserved
bit 1 - Auto Silence Hold status
bit 2 - Soft Gate status
bit 3 - Limiter status
bit 4- Compressor status
bit 5 - Leveler status
bit 6 - reserved
bit 7 - Gain Manager status

commands:

0x28 Bypass Any GM Block (channel #, bitfield)
40d Bypasses multiple sections of the GM with one command
Command Structure: ***bbaa*0028*yy*80*zz*!**

0x29 Enable Any GM Block (channel #, bitfield)
41d Enables multiple sections of the GM with one command
Command Structure: ***bbaa*0029*yy*80*zz*!**

0x2A Toggle Bypass of Any GM Block (channel #, bitfield)
42d Toggles multiple GM section bypass settings with one command
Command Structure: ***bbaa*002:*yy*80*zz*!**

examples:

command:	response:
02020028008001!	***(none)***

Here, an MSP (device number 1) is instructed to bypass the Auto Silence Hold (bit 1) of channel 2.

command:	response:
0601002900800:!	***(none)***

This command instructs MSP device number 10d (**0:** [pseudo-hex]) to activate the soft gate and the ASH (bits 1 and 2) of channel 1.

command:	response:
8002002:008002!	***(none)***

When executed, this command toggles the gain manager status (bit 7) on MSP device 2, channel 2. Depending on the status of the GM when called, this will either enable or disable all sections of the GM.

comments:

The gain manager bit will override the status of the other bits. For instance, if this bit is used to have the gain manager disabled, all sections of the GM will be disabled also. Although the GM status overrides the settings of the 5 GM sections, it does not *overwrite* the 5 sections.

The MSP11 only responds to commands directed to channel 1.

0x16 → 0x1B gain manager threshold commands

description:

This command group is used to set or adjust the threshold levels of the different sections of the gain manager.

syntax of commands:

The commands in this section share the following parameters:

zz	=	device number	
yy	=	reserved for checksum	(no action parameter)
aa	=	channel number (01 or 02)	
bb	=	code for threshold	
	bits 0 - 7: (range)		
		0x06 - +18 dBu	(-6dBFS)
		0x7F - -103 dBu	(-127dBFS)

commands:

0x16 Set Limiter Threshold (channel #, Threshold)
22d Sets Threshold of Limiter section of GM
Command Structure: ***bbaa*0016*yy*80*zz*!**

0x17 Set Compressor Threshold (channel #, Threshold)
23d Sets Threshold of Compressor section of GM
Command Structure: ***bbaa*0017*yy*80*zz*!**

0x18 Set Leveler Threshold (channel #, Threshold)
24d Sets Threshold of Leveler section of GM
Command Structure: ***bbaa*0018*yy*80*zz*!**

0x19 Set Soft Gate Threshold (channel #, Threshold)
25d Sets Threshold of Soft Gate section of GM
Command Structure: ***bbaa*0019*yy*80*zz*!**

0x1A Set Auto Silence Hold Threshold (channel #, Threshold)
26d Sets Threshold of Auto Silence Hold section of GM
Command Structure: ***bbaa*001:*yy*80*zz*!**

0x1B Set Dynamic Silence Hold Threshold (channel #, Threshold)
27d Sets Threshold of Dynamic Silence Hold section of GM
Command Structure: ***bbaa*001;*yy*80*zz*!**

examples:

command:	response:
0601001600800:!	***(none)***

06 corresponds to a threshold of +18dBu (+24dBu - 6dBu = +18dBu). Thus, this command sets +18 dBu as the threshold value for the limiter section of channel 1's of the gain manager. The device number of this MSP is 10.

comments:

Setting threshold values above +18dBu (0x00 to 0x05) will cause undesirable side effects.

MSP11 only responds to channel 1 commands.

0x1C → 0x21 gain manager response time commands

description:

Using these commands adjusts the amount of time it takes for different gain manager sections to respond to signal input.

syntax of commands:

The commands in this section share the following parameters:

zz = device number
yy = reserved for checksum (no action parameter)
aa = channel number (01 or 02)
bbbbbb= code for response time
bits 0 - 14: (range)
0x1FFF - 8191ms
0x0001 - 1ms
bit 15:
1 - release time
0 - attack time
bits 16 - 23:
0x00 - reserved

commands:

0x1C Set Limiter Response Time (channel #, Response Time)
28d Sets Response Time of Limiter section of GM
Command Structure: ***bbbbbbaa*001<*yy*80*zz*!**

0x1D Set Compressor Response Time (channel #, Response Time)
29d Sets Response Time of Compressor section of GM
Command Structure: ***bbbbbbaa*001=*yy*80*zz*!**

0x1E Set Leveler Response Time (channel #, Response Time)
30d Sets Response Time of Leveler section of GM
Command Structure: ***bbbbbbaa*001>*yy*80*zz*!**

0x1F Set Soft Gate Response Time (channel #, Response Time)
31d Sets Response Time of Soft Gate section of GM
Command Structure: ***bbbbbbaa*001?*yy*80*zz*!**

0x20 Set Auto Silence Hold Response Time (channel #, Response Time)
32d Sets Response Time of Auto Silence Hold section of GM
Command Structure: ***bbbbbbaa*0020*yy*80*zz*!**

0x21 Set Dynamic Silence Hold Response Time (channel #, Response Time)
33d Sets Response Time of Dynamic Silence Hold section of GM
Command Structure: ***bbbbbbaa*0021*yy*80*zz*!**

examples:

command:	response:
00807801001<008003!	***(none)***

In this example, the limiter response time for channel 1 is set on MSP number 3. By breaking the statement up into its bit representation, (0000 0000 1000 0000 0111 1000) it becomes apparent that the release time has been set to 120 ms.

command:	response:
00007901001<008003!	***(none)***

In this example, the limiter response time for channel 1 is set on MSP number 3. By breaking the statement up into its bit representation, (0000 0000 0000 0000 0111 1001) it becomes apparent that the attack time has been set to 121 ms.

comments:

MSP11 only responds to channel 1 commands.

0x2B → 0x31 & 0x76 input/output gain control commands

description:

The commands control the level of the input and output faders for the Advantage MSP.

syntax of commands:

The commands in this section share the following parameters:

zz	=	device number	
yy	=	reserved for checksum	(no action parameter)
aa	=	channel number (01 or 02)	

commands:

0x2B Mute Output (channel #)
43d Mutes output for specified channel
Command Structure: ***aa*002;*yy*80*zz*!**

0x2C Unmute Output (channel #)
44d Unmutes output for specified channel
Command Structure: ***aa*002<*yy*80*zz*!**

0x2D Toggle Output Mute (channel #)
45d Toggles output mute for specified channel
Command Structure: ***aa*002=*yy*80*zz*!**

0x2E Volume Down (channel #)
46d Decreases output level by one step
Command Structure: ***aa*002>*yy*80*zz*!**

0x2F Volume Up (channel #)
47d Increases output level by one step and unmutes output, if muted
Command Structure: ***aa*002?*yy*80*zz*!**

examples:

command:	response:
01002<008008!	***(none)***

This unmutes channel 1 of MSP device number 8.

command:	response:
01002?008005!	***(none)***

For MSP device number 5, this command increases channel 1's output level by one step. If channel 1 is muted, it unmutes it.

syntax of command:

This command has the following parameter:

bb - bitfield for volume operations
- bit 0 - channel 1 action
  - 0 - no action
  - 1 - perform channel 1 action
- bit 1 - channel 1 volume up/down bit
  - 0 - channel 1 volume down
  - 1 - channel 1 volume up
- bit 4 - channel 2 action
  - 0 - no action
  - 1 - perform channel 2 action
- bit 5 - channel 2 volume up/down bit
  - 0 - channel 2 volume down
  - 1 - channel 2 volume up

command:

0x76	Volume Both (channel #, bitfield)
118d	Performs Output Level adjustments on one or both channels
	Command Structure: ***bbaa*0076*yy*80*zz*!**

examples:

command:	response:
03010076008001!	*(none**)***

By breaking up the bitfield (0000 0011), it can be seen that this command sets the volume of channel 1 up one step. This command is directed to MSP device number 1. The channel number ***(aa)*** is ignored in this command, but must be set to 01 for comaptibility.

command:	response:
33010076008001!	*(none**)***

By breaking up the bitfield (0011 0011), it can be seen that this command sets the volume of channel 1 and 2 up one step. This command is directed to MSP device number 1. The volume both command is not particularily useful for MSP11's, since the volume up / down commands provide more appropriate control.

syntax of command:

This command has the following parameter:

bb = input gain (range)
0x00 - +20dBu
0x14 - 0dBu

command:

0x30 Set Input Gain (channel #)
48d Sets input gain for specified channel
Command Structure: ***bbaa*0030*yy*80*zz*!**

example:

command: response:
02010030008001! *(none)*

The input gain for channel 1 on MSP device number 1 has been set to +18dBu (+20 dBu - 2dBu = +18dBu)

syntax of command:

This command has the following parameter:

bb = output gain
bits 00-06
0x00 - no output
0x01 - -48dBu
0x 02 - -42dBu
0x 03 - -36dBu
0x 04 - -30dBu
0x 05 - -28dBu
0x 06 - -26dBu
0x 07 - -24dBu
0x 1F - 0dBu
bit 07 - bypass status
1 - bypassed
0 - enabled

command:

0x31 Set Output Gain(channel #)
49d Sets Output Gain for specified channel
Command Structure: ***bbaa*0031*yy*80*zz*!**

examples:

command:	response:
030100310008001!	*(none)*

This command causes the peak output level of channel 1 to -12 dBu, for MSP device number 1.

comments:

The MSP11 responds only to channel 1.

0x38 → 0x3B gain matrix commands

description:

The order, levels and connections of the gain matrix are controlled by these commands.

syntax of commands:

The commands in this section share the following parameters:

zz	=	device number	
yy	=	reserved for checksum	(no action parameter)
b	=	matrix number	
	0x0	- matrix #1	
	0x1	- matrix #2	
c	=	branch number	
	0x0	- input 1 to output 1	
	0x1	- input 2 to output 1	
	0x2	- input 3 to output 1	(if units are linked)
	0x3	- input 4 to output 1	(if units are linked)
	0x4	- input 1 to output 2	
	0x5	- input 2 to output 2	
	0x6	- input 3 to output 2	(if units are linked)
	0x7	- input 4 to output 2	(if units are linked)
dd	=	branch gain	
	0x00	- 0dB	
	0x7F	- -127dB	
!	=	execute-command character	

command:

0x38 Set Branch as Inverting(matrix #, branch #)
56d Sets the specified branch as inverting
Command Structure: ***cb*010038*yy*80*zz*!**

0x39 Set Branch as Non-Inverting(matrix #, branch #)
57d Sets the specified branch as non-inverting
Command Structure: ***cb*010039*yy*80*zz*!**

0x3B Set Branch Gain(matrix #, branch #, gain)
59d Sets Gain of specified branch
Command Structure: ***ddcb*01003;*yy*80*zz*!**

examples:

command: response:
11010038008005! *(none)*

This commands sets the input 2 to output 1 branch as inverting.

command: response:
184002003:008004! *(none)*

The branch gain for channel 1 has been set to -24dBu (0dB - 24dB = -24dB). The particular branch for this case is matrix 1, input 1 to output 2.

comments:

As there is no matrix in the MSP11, these commands have no effect.

0x48 → 0x4B delay commands

description:

These commands adjust and set the delay for individual channels of the Advantage MSP.

syntax of commands:

The commands in this section share the following parameters:

zz	=	device number	
yy	=	reserved for checksum	(no action parameter)
aa	=	channel number (01 or 02)	
bbbb	=	Delay settings	
	0x0000 -	no delay	
	0x0001 -	1/48 ms	(20.8us)
	0x0002 -	2/48 ms	(41.6us)
	0xFEFF -	65279/48 ms	(1.36 s)
cc	=	Delay bypass setting	
	bit 2 -	1 - Bypassed	
		0 - Enabled	
	all other bits -	reserved	
!	=	execute-command character	

command:

0x48 Bypass Delay(channel #)
72d Bypasses Delay for specified channel
Command Structure: ***aa*0048*yy*80*zz*!**

0x49 Enable Delay(channel #)
73d Enables Delay for specified channel
Command Structure: **aa 0049 yy 80 zz !**

0x4B Set Delay(channel #, delay value)
75d Sets Delay value for specified channel
Command Structure: **bbbb aa 004; yy 80 zz !**

examples:

command:	response:
010048008002!	*(none)*

This command bypasses the delay for channel 1, MSP device number 2.

command:	response:
0022010048008002!	*(none)*

This command sets channel 1's delay to .7083 (34/48) ms, for the same MSP as before, device number 2.

comments:

The MSP11 and MSP22 have propagation delays caused by the A/D and D/A converters of approximately 1.33 ms. All delay values used with these commands are in addition to the propagation delay. Setting the delay to 0x0000 will provide 0/48 + 1.33ms of delay. It is not possible to avoid the 1.33 ms of delay.

MSP11 will only respond to channel 1 commands.

0x50 → 0x51 crossover commands

description:

These commands enable and disable the crossovers for specific channels.

syntax of commands:

The commands in this section share the following parameters:

zz	=	device number
yy	=	reserved for checksum
aa	=	channel number (01 or 02)
!	=	execute-command character

command:

0x50	Bypass Crossover(channel #)
80d	Bypasses Crossover section for specified channel
	Command Structure: ***aa*0050*yy*80*zz*!**
0x51	Enable Crossover(channel #)
81d	Enables Crossover section for specified channel
	Command Structure: ***aa*0051*yy*80*zz*!**

examples:

command:	response:
020051008008!	*(none)*

This command enables channel 2's crossover for MSP device number 8.

comments:

As there is no crossover in the MSP11, these commands have no effect on that device.

By default, channel 1 is the HPF section of the crossover and channel 2 is the LPF section. This cannot be changed. Each filter can be independently bypassed using the appropriate channel.

0x60 → 0x61 eq commands

description:

These commands enable and disable the equalizer for specific channels

syntax of commands:

The commands in this section share the following parameters:

zz	=	device number
yy	=	reserved for checksum
aa	=	channel number (01 or 02)
!	=	execute-command character

0x60 Bypass EQ(channel #)
96d Bypasses EQ section for specified channel
Command Structure: ***aa*0060*yy*80*zz*!**

0x61 Enable EQ(channel #)
97d Enables EQ section for specified channel
Command Structure: ***aa*0061*yy*80*zz*!**

examples:

command:	response:
010061008005!	*(none)*

This command enables channel 1's equalizer section of MSP device number 5.

comments:

The MSP11 only responds to channel 1 commands.

0x7E & 0x78 remote control commands

Description:

Various remote control commands to recall presets or perform remote actions.

Syntax of commands:

The commands in this section share the following parameters:

zz	=	device number	
yy	=	reserved for checksum	(no action parameter)
bb	=	preset number (1-20)	
!	=	execute-command character	

command:

0x7E Recall System Preset(channel #,preset #)

126d Recalls all system parameters from the specified preset number to the current settings

Command Structure: ***bbaa*007>*yy*80*zz*!**

syntax of command:

This command has the following parameter:

bbbbbb - button number (0-329)

0d	-	PowerUp button
1-40d	-	Normal Buttons
41d	-	Reserved
42-57d	-	Logic input closures 1-16
58-73d	-	Logic input openings 1-16
74-329d	-	Logic input binary mode 0-255
330-16x10^6	-	Reserved

command:

0x78 Execute Button (button #)

120d Executes the actions associated with the specified button definition or logic input definition.

Command Structure: ***bbbbbb*010078*yy*80*zz*!**

examples:

command:	response:
0501007>008003!	*(none)*

This command recalls preset 5 on the MSP, device number 3.

command:	response:
0000270100780080002!	***(none)***

This command activates button 39 on MSP number 2.

comments:

. set-baud

Description:

The set-baud rate command allows the user to specify the baud rate at which the Advantage MSP operates. The units operate at 2400, 9600, 19200, and 38400 baud. In order to specify which of these baud rates to use, the Advantage MSP refers to them by the numbers 0,1,2 and 3; respectively.

Syntax of Command:

*rrii*80*dd*.

where

rr	=	Baud rate (**0** to **3**)
ii	=	Compliment of selected baud rate (< to **?** [pseudo-hex])
80	=	Device type bitmask for Advantage MSP
dd	=	Device number bitmask (1 to 63d; **00** to **3?** [pseudo-hex])
.	=	set-baud command character

Syntax of response:

no response

Example:

command:	response:
00??8002.	***(none)***

This command changes the baud of the Advantage MSP (device number 2) to 2400 (mode **00** [pseudo-hex]).

Comments:

Changing the baud value will immediately disconnect the user from the Advantage MSP until the user has changed the baud of the device connected to serial port also. Therefore, this command can be dangerous and is not recommended.

+ sleep-for-10-seconds

Description:

The sleep-for-10-seconds command allows the Advantage MSP to ignore all communication for 10 seconds,. During this 10 seconds of sleep, the Advantage MSP will not respond to nor echo any commands that it receives.

Syntax of Command:

80*dd*+

where

80	=	Device type bitmask for the Advantage MSP
dd	=	Device number bitmask (1 to 63d; **00** to **3?** [pseudo-hex])
+	=	sleep-for-10-seconds command character

Syntax of response:

no response

Example:

command:	response:
800;+	***(none)***

This example causes the Advantage MSP (device number 11d) to sleep for 10 seconds.

Comments:

/ get-version

Description:

The get-version command causes the Advantage MSP to return the model identification code and firmware version to the user. The firmware version is the release date, in the American format *mmddyy*. It is important to note that the Advantage MSP will return this date in "decimal" format, **not** pseudo-hex.

Syntax of Command:

80*dd*/

where

80	=	Device type bitmask for Advantage MSP
dd	=	Device number bitmask (1 to 63d; **00** to **3?** [pseudo-hex])
/	=	get-version command character

Syntax of response:

16*mmddyy*↵

where

16	=	Model i.d. for Advantage MSP22 [pseudo-hex]
mm	=	2 digit decimal month character
dd	=	2 digit decimal day character
yy	=	2 digit decimal year character

Example:

command:	response:
800=/	**16060598↵**

This command asks a Advantage MSP, number 13d, (**0=** [pseudo-hex]) to return its model i.d. and firmware date. In this case, the model i.d. is **16** [pseudo-hex] and firmware date is 6/5/98.

Comments:

ASCII Code Chart

ls

000. 0x00	016. 0x10	032. 0x20	048. 0x30	064. 0x40	080. 0x50	096. 0x60	112. 0x70
NUL	**DLE**	**(space)**	**0** nibble 0x0	**@** repeat code	**P** button 15	**`** button 31	**p** select 1,3
001. 0x01	017. 0x11	033. 0x21	049. 0x31	065. 0x41	081. 0x51	097. 0x61	113. 0x71
SOH	**DC1**	**!** vol limits	**1** nibble 0x1	**A**	**Q** button 16	**a**	**q** select 2,3
002. 0x02	018. 0x12	034. 0x22	050. 0x32	066. 0x42	082. 0x52	098. 0x62	114. 0x72
STX	**DC2**	**"** do-button	**2** nibble 0x2	**B** button 01	**R** button 17	**b** button 32	**r** select 1,2,3
003. 0x03	019. 0x13	035. 0x23	051. 0x33	067. 0x43	083. 0x53	099. 0x63	115. 0x73
ETX	**DC3**	**#** do-volume	**3** nibble 0x3	**C** button 02	**S** button 18	**c** button 33	**s** select 4
004. 0x04	020. 0x14	036. 0x24	052. 0x34	068. 0x44	084. 0x54	100. 0x64	116. 0x74
EOT	**DC4**	**$** define-preset	**4** nibble 0x4	**D** button 03	**T** button 19	**d** button 34	**t** select 1,4
005. 0x05	021. 0x15	037. 0x25	053. 0x35	069. 0x45	085. 0x55	101. 0x65	117. 0x75
ENQ	**NAK**	**%** get-preset	**5** nibble 0x5	**E** button 04	**U** button 20	**e** button 35	**u** select 2,4
006. 0x06	022. 0x16	038. 0x26	054. 0x36	070. 0x46	086. 0x56	102. 0x66	118. 0x76
ACK	**SYN**	**&** get/set-volume	**6** nibble 0x6	**F** button 05	**V** button 21	**f** button 36	**v** select 1,2,4
007. 0x07	023. 0x17	039. 0x27	055. 0x37	071. 0x47	087. 0x57	103. 0x67	119. 0x77
BEL	**ETB**	**'**	**7** nibble 0x7	**G** button 06	**W** button 22	**g** button 37	**w** select 3,4
008. 0x08	024. 0x18	040. 0x28	056. 0x38	072. 0x48	088. 0x58	104. 0x68	120. 0x78
BS	**CAN**	**(** do-logic	**8** nibble 0x8	**H** button 07	**X** button 23	**h** button 38	**x** select 1,3,4
009. 0x09	025. 0x19	041. 0x29	057. 0x39	073. 0x49	089. 0x59	105. 0x69	121. 0x79
HT	**EM**	**)** do-preset	**9** nibble 0x9	**I** button 08	**Y** button 24	**i** button 39	**y** select 2,3,4
010. 0x0A	026. 0x1A	042. 0x2A	058. 0x3A	074. 0x4A	090. 0x5A	106. 0x6A	122. 0x7A
LF	**SUB**	***** get-status	**:** nibble 0xA	**J** button 09	**Z** button 25	**j** button 40	**z** select 1,2,3,4
011. 0x0B	027. 0x1B	043. 0x2B	059. 0x3B	075. 0x4B	091. 0x5B	107. 0x6B	123. 0x7B
VT	**ESC**	**+** sleep 10 sec.	**;** nibble 0xB	**K** button 10	**[** button 26	**k** select none	**{**
012. 0x0C	028. 0x1C	044. 0x2C	060. 0x3C	076. 0x4C	092. 0x5C	108. 0x6C	124. 0x7C
FF	**FS**	**,** read memory	**<** nibble 0xC	**L** button 11	**** button 27	**l** select 1	**¦**
013. 0x0D	029. 0x1D	045. 0x2D	061. 0x3D	077. 0x4D	093. 0x5D	109. 0x6D	125. 0x7D
CR	**GS**	**-** write memory	**=** nibble 0xD	**M** button 12	**]** button 28	**m** select 2	**}**
014. 0x0E	030. 0x1E	046. 0x2E	062. 0x3E	078. 0x4E	094. 0x5E	110. 0x6E	126. 0x7E
SO	**RS**	**.** set defaults	**>** nibble 0xE	**N** button 13	**^** button 29	**n** select 1,2	**~**
015. 0x0F	031. 0x1F	047. 0x2F	063. 0x3F	079. 0x4F	095. 0x5F	111. 0x6F	127. 0x7F
SI	**US**	**/** get version	**?** nibble 0xF	**O** button 14	**_** button 30	**o** select 3	**DEL**