

THEATRE DESIGN & TECHNOLOGY MAGAZINE

1992 SUMMER ISSUE - SOUND COLUMN

MIDI SHOW CONTROL 1.0 - THE NEW STANDARD

By Charlie Richmond

All types of technical elements in live shows will soon enjoy the universal ease and power of standardised control which MIDI has for years given complex musical instrument setups. MIDI Show Control (MSC) has recently become a new international standard that promises to do just this. Born of a proposal put forth by Andy Meldrum of Vari*Lite in early 1989, discussions amongst key manufacturers of high tech show equipment took place throughout 1990 and 1991 via the USITT CallBoard network's MIDI Forum. The culmination of this work was a proposal that gained the approval of the MIDI Manufacturers Association (MMA) and the Japan MIDI Standards Committee (JMISC), becoming MIDI 1.0 Recommended Practice RP-002 on 25 July 1991.

MSC is an extension of the basic MIDI 1.0 specification and remains completely compatible with all past and current MIDI equipment. MSC follows the rules for standard Universal Real Time System Exclusive (SysEx) messages, which has been part of MIDI from the beginning. Potentially, this means that a single MIDI control network can be used for communication between all MIDI equipment in very large productions without confusion or complication. Moreover, substantial advantages will be realised by allowing all this equipment to intercommunicate. MSC is an interconnection standard for use between intelligent "Controllers" and similarly intelligent "Controlled Devices" and is not meant to provide or replace lower level communication such as DMX512 between controllers and relatively unintelligent "slaves."

In a lighting control application, for example, MSC would be used between the stage manager's show control computer and the lighting console or perhaps between two lighting consoles operating together (or both, even using the same MIDI network) but not between the lighting console and the dimmers. Typically, the person controlling a show operates a show control computer connected via MIDI to a variety of controllers of various technical systems including lighting, sound, rigging, mechanics, pyro, smoke, lasers, multimedia equipment and more. In essence, the MIDI network becomes an electronic communication equivalent of the headset system. Software and systems implementing MSC and specifically for this application are already available and in use.

This article describes only the basics of the standard - the complete MSC 1.0 standard is publicly available from the MMA and is essential for anyone seriously interested in MSC implementation. A good working knowledge of the MIDI 1.0 standard, also sold by the MMA, is additionally helpful. The self-described purpose of MSC is "to allow MIDI systems to communicate with and to control dedicated intelligent control equipment in theatrical, live performance, multi-media,

audio-visual and similar environments." For maximum speed, simplicity, reliability and cost-effectiveness, this first version of MSC runs in an 'open-loop' environment, requiring no response from the Controlled Device. A 'closed-loop' version of MSC, meeting the needs of fail-safe control under the most critical conditions, is currently under development. Again, from the standard, "The guiding philosophy behind live performance control is that, as much as possible, failures of individual Controlled Devices should not impair communications with other Controlled Devices."

In any case, MSC is not certified as the sole control determinant under life-threatening conditions. In these situations, the most rigid safety procedures and systems must always be present. As the standard clearly states, "MIDI Show Control is not intended to tell dangerous equipment when it is safe to go; it is only intended to signal what is desired if all conditions are acceptable and ideal for safe performance. Only properly designed safety systems and trained safety personnel can establish if conditions are acceptable and ideal at any time."

In the following documentation, as well as in MIDI specifications generally, transmitted data is expressed in its hex form, often including an 'H' immediately after each byte to confirm its hex nature. For a full explanation of hex notation specifically relating to MIDI implementation, please refer to the MIDI 1.0 specification, since the subject is well treated and beyond the scope of this article. MSC is completely contained within the Universal Real Time System Exclusive message format. All messages of this type begin with F0H (= start of SysEx) followed by 7FH (= Universal Real Time header) then a user defined destination address in the range 00H-7FH (= device_ID No.) and 02H (= MSC message header) - all in front of the command variables and related data in the message. The message is terminated by F7H (= end of SysEx) and the maximum length of any MSC message is 128 bytes.

Within these parameters, a number of variables may be sent as follows (in hex):

```
F0 7F <device_ID> 02 <command_format> <command> <data> F7
```

Command_format tells the addressed device what kind of message is being sent, including general categories of lighting, sound, machinery, video, projection, process control, pyro and all-types. A number of sub-categories are defined (except under all-types) and there is room for future expansion along with the ability to address an entire group or just a specific sub-category (see Table 1).

TABLE 1

Hex command_format

00 reserved for extensions

01 Lighting (General Category)

02 Moving Lights

- 03 Colour Changers
- 04 Strobes
- 05 Lasers
- 06 Chasers

- 10 Sound (General Category)
- 11 Music
- 12 CD Players
- 13 EPROM Playback
- 14 Audio Tape Machines
- 15 Intercoms
- 16 Amplifiers
- 17 Audio Effects Devices
- 18 Equalizers

- 20 Machinery (General Category)
- 21 Rigging
- 22 Flys
- 23 Lifts
- 24 Turntables
- 25 Trusses
- 26 Robots
- 27 Animation
- 28 Floats
- 29 Breakaways
- 2A Barges

- 30 Video (General Category)
- 31 Video Tape Machines
- 32 Video Cassette Machines
- 33 Video Disc Players
- 34 Video Switchers
- 35 Video Effects
- 36 Video Character Generators
- 37 Video Still Stores
- 38 Video Monitors

- 40 Projection (General Category)
- 41 Film Projectors
- 42 Slide Projectors
- 43 Video Projectors
- 44 Dissolvers
- 45 Shutter Controls

- 50 Process Control (General Category)
- 51 Hydraulic Oil
- 52 H2O
- 53 CO2
- 54 Compressed Air
- 55 Natural Gas
- 56 Fog
- 57 Smoke
- 58 Cracked Haze

- 60 Pyro (General Category)
- 61 Fireworks
- 62 Explosions
- 63 Flame

64 Smoke pots

7F All-types

The command tells the Controlled Device with a matching device_ID and command_format what the Controller wishes it to do.

Commands may be simple or complex, with the Controlled Device's response designed to be appropriate for the application. A number of data variables may follow each command so that the most complex devices can be controlled while simpler Controlled Devices simply ignore the information they do not use. Variables are sent in the order of generic to specific, with null characters (00H) separating each one. Each Controlled Device may use the data variables of each command in its own way, but 1024 Standard Generic Control Numbers have been defined for lighting - with 14,000 more undefined.

The first group of commands defined are: GO (01H); STOP (02H); RESUME (03H); TIMED_GO (04H), which is like GO with an execution time; LOAD (05H), which puts a cue into standby, ready to execute immediately; and GO_OFF (0BH), which starts a transition or fade to an off state. Each can optionally contain a cue number, a cue list and a cue path as data. SET (06H) defines the value of a numbered Generic Control and may include an execution time; FIRE (07H) triggers a numbered macro; ALL_OFF (08H) turns everything off; RESTORE (09H) restores everything as it was prior to ALL_OFF; RESET (0AH) presets to the top of the show.

The second group of commands are specific to computer controlled theatrical sound systems: GO/JAM_CLOCK (10H), which forces the auto follow clock timer to the time of the auto follow cue; STANDBY_+ (11H) and STANDBY_- (12H), which load into standby the next or previous cue, respectively; SEQUENCE_+ (13H) and SEQUENCE_- (14H), which load into standby the next or previous parent (using the whole integer value of the cue only) cue; START_CLOCK (15H), STOP_CLOCK (16H), ZERO_CLOCK (17H), SET_CLOCK (18H), which start, stop, reset and set the clock timer to a particular time; MTC_CHASE_ON (19H) and MTC_CHASE_OFF (1AH), which lock and unlock the clock timer to incoming MIDI Time Code; OPEN_CUE_LIST (1BH) and CLOSE_CUE_LIST (1CH), which include in the show or exclude from the show a specified Cue List and the cues it contains; and OPEN_CUE_PATH (1DH) and CLOSE_CUE_PATH (1EH), which include in the show or exclude from the show a specified Cue Path.

A standard command data format for cue, cue list and cue path numbers is part of the specification. Numbers including decimal points are transmitted as ASCII characters (0-9 = 30H-39H; "." = 2EH). For many commands, cue numbers are optional, as are Cue List and Cue Path data, and the Controlled Device may simply discard data that it does not support. In this way, the standard provides a tremendous amount of flexibility for complex systems while allowing the same commands to be used with much simpler devices. Time data is represented in the standard full form MIDI Time Code (MTC) format, which is basically an expanded implementation of SMPTE Time Code with accuracy to 1/100 of a

frame (330 microseconds).

As an example, here's how an MSC message would actually look (in hex):

```
F0 7F 61 02 42 04 60 02 1E 0F 63 31 33 35 2E 36 00 33 36 2E 36
00 35 39 F7
```

The breakdown of this message is as follows:

F0 7F	Universal Real Time System Exclusive message header
61	Device No. 97 (61 Hex)
02	MSC sub ID No. 1 header
42	Slide Projectors command_format
04	TIMED_GO command
60 02 1E 0F 63	00 hours 02 minutes 30 seconds 15 frames 99 subframes
31 33 35 2E 36	Cue 135.6
00	delimiter
33 36 2E 36	Cue List 36.6
00	delimiter
35 39	Cue Path 59
F7	End of SysEx

At the MIDI transmission rate of 31,250 bits per second these 25 bytes will take 8 milliseconds to send, which will create no perceivable delay. An example of a full slide projector implementation of this protocol could be that cues would correspond to individual slide numbers, cue lists to currently loaded slide trays and cue paths to slide trays that can be mechanically loaded by some automatic mechanism. A slide projector which does not support the TIMED_GO command could respond by simply projecting slide 135 in the currently loaded slide tray using a default fade time. A fully computerized slide projector control system connected to a number of slide projectors with multiple trays per projector could, however, respond to all aspects of this message.

We are convinced that MSC has a bright future since MIDI has been largely responsible for the enormous proliferation of musical equipment and technology since 1985. MSC contains the same features that have allowed MIDI to do this: it is powerful, easy to use, inexpensive, and, most importantly, it is a standard. Much equipment already in the field uses some form of non-standard MIDI implementation since customers have been asking for the equivalent of MSC for years. Most of this equipment can easily incorporate MSC with a simple software upgrade. Obviously it will be more difficult to upgrade older gear, but not impossible. In this case it may be more cost effective to use a low cost MSC add-on unit, interfaced with a few modifications, than to discard this still useful equipment. In fact, until everyone supports MSC, we will even have to use similar interfaces in new projects but we definitely look forward to when everything can simply be connected together with standard MIDI cables and communicate perfectly right from the start!

Copyright ©1992 Charlie Richmond