## Appendix A

# DP/4 MIDI SysEx Implementation Specification - version 1.02

## 1 Introduction and Overview

This document describes the MIDI System Exclusive (SysEx) communication protocol used when the DP/4 is communicating with an external computer or device (EXT). The protocol is designed to support the implementation of data storage, management and editing programs running on EXT, and so this information is especially relevant to designers and programmers of such programs. The commands described here allow editor/librarian programs to acquire, modify and replace DP/4 preset and system control data.

### 1.1 Universal System Exclusive Device Inquiry Message

The DP/4 supports the MIDI Device Inquiry message which allows instruments and computers to ascertain the identity of the unit(s) to which they are connected via MIDI. The DP/4 responds to the following Identity Request message by sending an Identity Reply message. The DP/4 will respond to the inquiry if the channel information in the message contains either the Device ID of the DP/4 or the All Channel Broadcast code (7Fh), but the message should not contain both.

| | | |
|---|---|---|
| 11110000 | F0 | System Exclusive status byte |
| 01111110 | 7E | Non Real Time message code |
| 0000nnnn | 0n | nnnn = Device ID number |
| – or – | | |
| 01111111 | 7F | All Channel Broadcast code |
| 00000110 | 06 | General Information message code |
| 00000001 | 01 | Identity Request message code |
| 11110111 | F7 | End of System Exclusive |

### 1.2 System Exclusive Device Identity Reply Message

The following Identity Reply message contains information about the DP/4, and is transmitted in response to an Identity Request.

| | | |
|---|---|---|
| 11110000 | F0 | System Exclusive status byte |
| 01111110 | 7E | Non Real Time message code |
| 0000nnnn | 0n | Base MIDI channel number |
| 00000110 | 06 | General Information message code |
| 00000010 | 02 | Identity Reply message code |
| 00001111 | 0F | ENSONIQ manufacturer's code |
| 01000000 | 40 | Signal Processor Product Family ID code - LSByte |
| 00000000 | 00 | Signal Processor Product Family ID code - MSByte |
| 00000000 | 00 | DP/4 Family Member (Model ID) code LSByte |
| 00000000 | 00 | DP/4 Family Member (Model ID) code MSByte |
| 00000000 | 00 | Software revision information: |
| 00000000 | 00 | — (first two bytes not used by DP/4 - always zero) |
| 0vvvvvvv | vv | — Major Version Number (integer portion) |
| 0vvvvvvv | vv | — Minor Version Number (decimal fraction portion) |
| 11110111 | F7 | End of System Exclusive |

## 2 MIDI System Exclusive Packets

A System Exclusive message can be viewed as a packet of information which appears in the form of a MIDI data stream. Each packet can be divided into three sections or pieces. The first and last packet pieces form the frame for the message. The Message contains the commands described in section 3. Every Message must be preceded with a SysEx head and followed with a SysEx tail. A complete packet looks like this:

SysEx Head.............................Message .........................SysEx Tail

### 2.1 MIDI System Exclusive Packet Head

This is the common MIDI system exclusive header which must be used on all system exclusive messages sent to and from the DP/4. These five bytes are always sent preceding the message portion of the packet. The DP/4 Model ID Code in this header is the common for all compatible members of the family to allow exchange of information. All messages which are not recognized by the DP/4 will be ignored. In all subsequent message descriptions, "*[ header ]*" represents this five byte structure.

| | | |
|---|---|---|
| 11110000 | F0 | System Exclusive status byte |
| 00001111 | 0F | ENSONIQ Code |
| 01000000 | 40 | Signal Processor Family ID Code |
| 00000000 | 00 | DP/4 Model ID Code |
| 0000dddd | 0d | dddd = Device ID (Base MIDI channel) number |

The 5-byte SysEx header is always followed by the Message type code byte:

| | | |
|---|---|---|
| 0mmmmmmm | 0m | mmmmmmm = Message Type code (see section 3) |

### 2.2 MIDI System Exclusive Packet Tail

For every head there must be a tail. The tail follows the message portion, and is the last byte of every complete SysEx packet.

| | | |
|---|---|---|
| 11110111 | F7 | End of System Exclusive |

### 2.3 Message Format

The Message Type code in the packet head defines the message format. The specific format depends on the message type. There may be additional bytes which provide more information about the message, and in some cases there may also be a bulk data segment. Within these bulk data segments, 8 bit data bytes must be transmitted and received using the 7 bit data format of MIDI, where the MSB of these data bytes must always be a zero, so each of the 8 bit data bytes is converted to two 4 bit nybbles for transmission. These nybbles each sent as bytes whose upper four bits are all zero. This is a description of the format of all nybblized data bytes within the packet frame as they are transmitted or received via MIDI. The details of each message are given in section 3.

| | |
|---|---|
| 0000HHHH | H = Hi 4 bits of data byte - transmitted first |
| 0000LLLL | L = Lo 4 bits of data byte |

This represents how the 8 bit byte HHHHLLLL would be transmitted.

### 2.4 Receiver Errors

If a message received by the DP/4 is not understood, or is incomplete, then an informative error message will be displayed and an error message will be sent as described in section 3.2. Errors typically occur when the MIDI In cable is accidentally disconnected during a long dump message (such as an All Presets Dump message).

# 3 Message Types

This section describes the various messages to be used between EXT and DP/4. The message type code corresponds to the last byte of the system exclusive packet head described in section 2.1.

## 3.1 Command Messages (Message Type = 01)

Command messages are used to transmit simulated front panel user actions *to* the DP/4, but they are not transmitted *from* the DP/4. These command messages employ a generalized variable size message format in which the command code and the command-specific data bytes are transmitted using the nybblized message format described in section 2.3. Remember, the message type code is part of the header and is sent as a byte, but the command code is considered part of the data segment and must be sent as two nybbles. Successful transactions are followed by an error message indicating that no error occurred (ACKnowledge). This ACK also indicates that the system is ready for the next transaction.

### 3.1.1 Parameter Change Command

This message allows any effect parameter to be edited under remote control. When this message is received, the system will automatically be put into Edit (or System) mode and the specified unit will become active. The unit number in the message (specified using the following codes) indicates which logical group contains the parameter to be edited.

| *Unit* | *Number* | *Unit* | *Number* |
|---|---|---|---|
| A | 0 | D | 3 |
| B | 1 | Config | 4 |
| C | 2 | System | 5 |

The parameter number must be legal within the specified group, with unit parameters based on effect type and config parameters based on Input Config. The new parameter value is always transmitted as a 16 bit word. If the parameter only uses an 8 bit value, then the MSByte should be transmitted as zero. Illegal values will not change the value in the system, and an error message will be returned to indicate the problem.

**Parameter Change Command**

Message code: 01 (Command message)
Command code: 01
Message size: 17 MIDI bytes

```
[header ]       DP/4 SysEx Header
00000001        Command Message code
Note: the following 5 bytes must each be transmitted as nybbles!
    00000001    Parameter Change command code
    00000uuu    uuu = unit number (0..4)
    0ppppppp    ppppppp = parameter number (0..127)
    mmmmmmmm    mmmmmmmm = parameter value MSByte
    11111111    11111111 = parameter value LSByte
11110111        End of Exclusive status byte
```

*Example (in hexadecimal notation, assuming MIDI Device ID = 01 transmitted as 00):*

```
     Header       ID  Msg   Cmd    Unit   Param   — Value —     EOX   Unit C; Param 03
F0 0F 40 00 00   01   00 01  00 02  00 03   00 00   07 0F    F7    Value = 127 (7Fh)
```

### 3.1.2 Virtual Button Command

EXT can simulate button presses from the front panel of the DP/4 by sending this command. The rear panel foot switches can also be simulated using this command. Sending the listed logical button numbers will simulate a single button being held down. Button up commands require the button logical numbers as listed below *with an offset of 128 (80h) added.* The button number follows the command code byte in the message. Remember to send a button up command for every button down command that is sent. Note: a brief delay (100 msecs) between button commands, or at least pairs of button commands, is recommended to allow time for the DP/4 to do any required internal processing and update the display..

| *Logical Number* | *Front Panel Button Name* | *Logical Number* | *Front Panel Button Name* |
|---|---|---|---|
| 0 | A (Unit A) | 7 | System |
| 1 | B (Unit B) | 8 | Left arrow |
| 2 | C (Unit C) | 9 | Right arrow |
| 3 | D (Unit D) | 10 | Cancel |
| 4 | Config | 11 | Write |
| 5 | Select | 12 | Foot switch 1 |
| 6 | Edit | 13 | Foot switch 2 |

**Virtual Button Command**

Message code: 01 (Command message)
Command code: 02
Message size: 11 MIDI bytes

*[ header ]* DP/4 SysEx Header
`00000001` Command message code
*Note: the following 2 bytes must* ***both*** *be transmitted as nybbles!*
`00000010` Virtual Button command code
`s000nnnn` s = button state (0 = down; 1 = up); nnnn = button logical number (0..13)
`11110111` End of Exclusive status byte

*Example (in hexadecimal notation, assuming MIDI Device ID = 01 transmitted as 00):*

```
     Header      ID   Msg    Cmd     Button  EOX
F0 0F 40 00 00   01   00 02   00 01   F7          B (Unit B) button down
F0 0F 40 00 00   01   00 02   08 01   F7          B (Unit B) button up
```

### 3.1.2 Virtual Knob Command

This command is used to simulate the rotation of the data entry knob on the front panel of the DP/4. Each command message received will effectively turn the knob up to 63 increments (clockwise) or decrements (counter-clockwise), as specified by the number and direction information in the message. If the specified number of "clicks" causes the knob to wrap (change direction) internally, an error message is returned.

**Virtual Knob Command**

Message code: 01 (Command message)
Command code: 03
Message size: 11 MIDI bytes

*[ header ]* DP/4 SysEx Header
`00000011` Command message code
*Note: the following 2 bytes must* ***both*** *be transmitted as nybbles!*
`00000011` Virtual Knob command code
`d0nnnnnn` d = direction (1 = clockwise/increment; 0 = counter-clockwise/decrement); nnnnnn = number of increments or decrements (0..63)
`11110111` End of Exclusive status byte

## 3.2 Error Messages (Message Type = 02)

Error messages are transmitted by the DP/4 when an error occurs while processing any of the messages described in section 3. DP/4 error messages contain a single byte of data. This byte indicates the error status of the last transaction. After a command message or a dump message is received, an error message is sent when the DP/4 has completed the processing of the received message or if an error occurs during the reception of the message. This feature is intended to aid in the implementation of "handshaking" schemes with external devices. *Please note that error messages are not sent in response to a valid dump request; instead the DP/4 transmits the requested dump*. The following table lists the error codes that may be sent by the DP/4:

| *Error Code* | *Error Description* |
|---|---|
| 0 | Acknowledge (ACK) — No error. |
| 1 | Receiver time out (> 1 second since last byte of message was received). |
| 2 | EOX (End of Exclusive F7h) was received when command code data was expected. |
| 3 | DP/4 was still processing previous command. |
| 4 | EOX received when other data was expected. |
| 5 | The byte received after the data block was not EOX. |
| 6 | Command message contained an invalid or illegal argument. |
| 7 | Parameter value in command was illegal. |
| 8 | Button number in command was illegal. |
| 9 | Knob value in command was illegal. |
| 10 | The preset received by the DP/4 was not automatically loaded because its type was incompatible with the current configuration. |

Note that the DP/4 only *transmits* these error messages. If they are received by the DP/4, they will be ignored.

### Error Message

| | |
|---|---|
| Message code: | 02 |
| Message size: | 8 MIDI bytes |

| | |
|---|---|
| *[ header ]* | DP/4 SysEx Header |
| `00000010` | Error message code |
| `0eeeeeee` | `eeeeeee` = error code (see above listing) |
| `11110111` | End of Exclusive status byte |

## 3.3 Request Messages

The DP/4 will respond to the following dump request messages by transmitting the requested bulk dump. There is no error message sent if the request is legal; the bulk dump serves as the confirmation response. Note: The DP/4 receives but does not transmit the following Dump Request messages.

### 3.3.1 Single Preset Dump Request (Message Type = 16)

The DP/4 will dump the specified internal RAM preset using the bulk dump message described in section 3.4 when it receives this message.

#### Single Preset Request Message

| | |
|---|---|
| Message code: | 16 |
| Message size: | 9 MIDI bytes |

| | |
|---|---|
| *[header ]* | DP/4 SysEx Header |
| `00010000` | Single preset request message code |
| `000000tt` | `tt` = preset type (0..3) |
| `0nnnnnnn` | `nnnnnnn` = preset number (0..99) |
| `11110111` | End of Exclusive status byte |

### 3.3.2 Preset Bank Dump Request (Message Type = 17)

The DP/4 will dump the specified internal RAM preset bank using the bulk dump message described in section 3.4 when it receives this message.

Note that it is possible to request preset banks from ROM by setting the RAM/ROM select code bits in the byte containing the preset bank type code. This enables external devices to get a dump of the factory ROM presets, which is not possible any other way. The ROM bank dump will have the same format as the normal preset bank dumps from RAM.

| *Select Code* | *Select Function* |
|---|---|
| 00 | RAM Preset banks (normal setting) |
| 01 | ROM Preset banks |
| 10 | Alternate ROM banks ("hidden" in ROM and used to initialize RAM banks) |

#### Preset Bank Request Message

| | |
|---|---|
| Message code: | 17 |
| Message size: | 8 MIDI bytes |

| | |
|---|---|
| *[header ]* | DP/4 SysEx Header |
| `00010001` | Preset bank request message code |
| `00ss00tt` | `ss` = RAM/ROM select code (0..2); `tt` = preset bank type (0..3) |
| `11110111` | End of Exclusive status byte |

### 3.3.3 All Presets Dump Request (Message Type = 18)

The DP/4 will dump all RAM preset memory using the bulk dump message described in section 3.4 when it receives this message.

#### All Preset Banks Request Message

| | |
|---|---|
| Message code: | 18 |
| Message size: | 7 MIDI bytes |

| | |
|---|---|
| *[ header ]* | DP/4 SysEx Header |
| `00010010` | All preset banks request message code |
| `11110111` | End of Exclusive status byte |

### 3.3.4 System Parameters Dump Request (Message Type = 19)

The DP/4 will dump all of the system global parameters using the bulk dump message described in section 3.4 when it receives this message.

#### System Parameters Request Message

| | |
|---|---|
| Message code: | 19 |
| Message size: | 7 MIDI bytes |

| | |
|---|---|
| *[ header ]* | DP/4 SysEx Header |
| `00010011` | System parameters request message code |
| `11110111` | End of Exclusive status byte |

### 3.3.5 All Presets with System Parameters Dump Request (Message Type = 20)

The DP/4 will dump all internal RAM preset memory and the system global parameters using the bulk dump message described in section 3.4 when it receives this message.

#### All Preset Banks And System Parameters Request Message

| | |
|---|---|
| Message code: | 20 |
| Message size: | 7 MIDI bytes |

| | |
|---|---|
| *[ header ]* | DP/4 SysEx Header |
| `00010100` | All preset banks and system parameters request message code |
| `11110111` | End of Exclusive status byte |

### 3.3.6 Edit Buffer Request Message (Message Type = 21)

The DP/4 will dump the current contents of the "edit buffer" using the bulk dump message described in section 3.4 when it receives this message. This system context information is useful for external editors needing to obtain the current status of all units.

#### Edit Buffer Request Message

| | |
|---|---|
| Message code: | 21 |
| Message size: | 7 MIDI bytes |

| | |
|---|---|
| *[ header ]* | DP/4 SysEx Header |
| `00010101` | Edit buffer request message code |
| `11110111` | End of Exclusive status byte |

## 3.4 Dump Messages

Bulk data dump messages are transmitted using the message format described in section 2.3. The message type byte, which is part of the system exclusive header, is given in decimal along with the name of the dump message. The actual data structures for presets and the system parameters are described in Section 4. The MIDI data byte lengths are listed in decimal for each message type.

### 3.4.1 Single Preset Dump (Message Type = 32)

This dump contains all data from a single preset (refer to section 4).
When this message is transmitted from the DP/4, it contains the data from the specified RAM preset. When this message is received by the DP/4, the new preset will be saved into the specified preset location in the internal RAM. *Remember that the specified destination preset is over-written by the incoming data and its previous contents are lost.* If this message is used to send a preset to a generic MIDI SysEx recorder, the destination preset number is embedded in the message. When the dump is reloaded into the DP/4, that same location will be replaced by the incoming preset data.

Note that the preset will be automatically "selected" loaded after it is received by the DP/4 if the preset type is compatible with the current configuration, based on the current active unit. The following table shows, for each preset type, which units may be active and which configuration is required if the preset is to be automatically selected.

| *Preset Type* | *Code* | *Active Unit* | *Configuration* | *Units Loaded* |
|---|---|---|---|---|
| Config | 3 | Config | <any> | A, B, C, D & Config |
| 4 Unit | 2 | A, B, C, D | One Source | A, B, C, D |
| 2 Unit | 1 | AB, CD | Two Source | AB or CD |
| 2 Unit | 1 | CD | Three Source | CD |
| 1 Unit | 0 | A, B, C, D | Four Source | A or B or C or D |
| 1 Unit | 0 | A, B | Three Source | A or B |

The DP/4 returns an error message if the current configuration does not support "selecting" the received preset's type into the active unit. In all cases, the DP/4 saves the received preset data into the designated RAM preset location and displays a reception complete confirmation message.

| *Preset Type* | *Data segment sizes* | | *Total Message size* |
|---|---|---|---|
| | *Bytes* | *Nybbles* | *MIDI Bytes* |
| One Unit Preset | 51 | 102 | 111 |
| Two Unit Preset | 87 | 174 | 183 |
| Four Unit Preset | 158 | 316 | 325 |
| Config Preset | 163 | 326 | 335 |

**Single Preset Dump Message**

| | |
|---|---|
| Message code: | 32 |
| Message size: | 9 MIDI bytes plus Data segment size (see above listing) |

```
[ header ]    DP/4 SysEx Header
00100000      Single preset dump message code
000000tt      tt = preset type code (0..3)
00nnnnnn      nnnnnn = destination RAM preset number (0..49)
...
[ data ]      Data segment (nybblized)
...
11110111      End of Exclusive status byte
```

### 3.4.2 Preset Bank Dump (Message Type = 33)

Each Preset Bank Dump message contains the data comprising the 50 User Presets. The actual size of the dump is dependent on the preset bank type. These dumps are only received into the RAM preset banks of the DP/4, but may be sent from RAM or ROM depending on the request message.

| *Preset Bank Type* | *Data segment sizes* | | *Total Message size* |
|---|---|---|---|
| | *Bytes* | *Nybbles* | *MIDI Bytes* |
| One Unit Preset Bank | 2550 | 5100 | 5108 |
| Two Unit Preset Bank | 4350 | 8700 | 8708 |
| Four Unit Preset Bank | 7900 | 15800 | 15808 |
| Config Preset Bank | 8150 | 16300 | 16308 |

**Preset Bank Dump Message**

| | |
|---|---|
| Message code: | 33 |
| Message size: | 8 MIDI bytes plus Data segment size (see above listing) |
| *[ header ]* | DP/4 SysEx Header |
| `00100001` | Preset bank dump message code |
| `000000tt` | `tt` = preset type (0..3) |
| ... | |
| *[ data ]* | Data segment (nybblized) — 50 presets |
| ... | |
| `11110111` | End of Exclusive status byte |

### 3.4.3 All Presets Dump (Message Type = 34)

The All Preset Banks Dump message contains four concatenated 50 preset dumps, in the following order: One Unit Presets Bank, Two Unit Presets Bank, Four Unit Presets Bank and finally the Config Presets Bank. The size of the data segment is the sum of the four individual bank dump sizes.

| *Dump Component* | *Data segment sizes* | | *Total Message size* |
|---|---|---|---|
| | *Bytes* | *Nybbles* | *MIDI Bytes* |
| One Unit Preset Bank | 2550 | 5100 | |
| Two Unit Preset Bank | 4350 | 8700 | |
| Four Unit Preset Bank | 7900 | 15800 | |
| Config Preset Bank | 8150 | 16300 | |
| Total: | 22950 | 45900 | 45907 |

**All Preset Banks Dump Message**

| | |
|---|---|
| Message code: | 34 |
| Message size: | 7 MIDI bytes plus Data segment size (see above listing) |
| *[ header ]* | DP/4 SysEx Header |
| `00100010` | All preset banks dump message code |
| ... | |
| *[ data ]* | Data segment (nybblized) — Four concatenated 50 preset banks |
| ... | |
| `11110111` | End of Exclusive status byte |

### 3.4.4 System Parameters Dump (Message Type = 35)

| *Dump Type* | *Data segment sizes* | | *Total Message size* |
|---|---|---|---|
| | *Bytes* | *Nybbles* | *MIDI Bytes* |
| System Parameters | 1312 | 2624 | 2631 |

**System Parameters Dump Message**

| | |
|---|---|
| Message code: | 35 |
| Message size: | 7 plus data segment size = 2631 MIDI bytes |
| *[ header ]* | DP/4 SysEx Header |
| `00100011` | System parameters dump message code |
| ... | |
| *[ data ]* | Data segment (nybblized) — 1312 bytes of system parameters |
| ... | |
| `11110111` | End of Exclusive status byte |

### 3.4.5 All Presets with System Parameters Dump (Message Type = 36)

The All Preset Banks with System Parameters Dump message comprises the data segment from the All Preset Banks Dump concatenated with the System Parameters. The data is sent in the following order: One Unit Presets Bank, Two Unit Presets Bank, Four Unit Presets Bank, Config Presets Bank, and finally the System Parameters. The size of the data segment is the sum of the four individual bank dump sizes plus the system parameters data segment size. Refer to the individual message descriptions for more information of the structure of the data.

| *Dump Component* | *Data segment sizes* | | *Total Message size* |
|---|---|---|---|
| | *Bytes* | *Nybbles* | *MIDI Bytes* |
| One Unit Preset Bank | 2550 | 5100 | |
| Two Unit Preset Bank | 4350 | 8700 | |
| Four Unit Preset Bank | 7900 | 15800 | |
| Config Preset Bank | 8150 | 16300 | |
| System Parameters Bank | 1312 | 2624 | |
| Total: | 24262 | 48524 | 48531 |

**All Preset Banks with System Parameters Dump Message**

| | |
|---|---|
| Message code: | 36 |
| Message size: | 7 plus data segment size = 48531 MIDI bytes |
| *[ header ]* | DP/4 SysEx Header |
| `00100100` | All preset banks with system parameters request message code |
| ... | |
| *[ data ]* | Data segment (nybblized) — Four preset banks concatenated with system parameters |
| .... | |
| `11110111` | End of Exclusive status byte |