highly liquid

MIDI CPU

Firmware Version 1.4

User Manual

Updated 2014-04-03

Additional documentation available at:

http://highlyliquid.com/support/

Table of Contents

1.0 Overview

Users are encouraged to read ***MIDI CPU Hardware User Manual*** before continuing with this document.

1.1 About The MIDI CPU

The MIDI CPU is a multipurpose MIDI input/output device. The MIDI CPU accepts input at the *control terminals* and MIDI IN port. Output is generated at the MIDI OUT port. Depending on configuration, certain control terminals may also generate output signals. See Theory of Operation for more information.

The MIDI CPU is configured via MIDI System Exclusive (SysEx) message. Control Terminal Configuration messages contain information related to individual control terminals. Global Configuration messages contain information related to the MIDI CPU as a whole. Configuration Examples and Technical Support are provided and may be adapted for specific applications.

SysEx messages can also be used to examine the current MIDI CPU configuration. See Configuration Retrieval.

1.2 About This Document

Throughout this document, SysEx message contents are shown in hexadecimal format. Hexadecimal numbers either are described as “hex” or are followed by the 'h' suffix. Decimal format should be assumed for all other numbers.

Key to character formatting:

- Clickable links to tables and figures appear in **blue**.
- Clickable links to sections within the document are underscored.
- Names of specific MIDI message types generated by the MIDI CPU appear in **red**.

Sections of the manual that have changed significantly since the previous firmware version are highlighted in yellow.

1.3 Additional Resources

Instructions for using a PC to create and send SysEx messages can be found at:

http://highlyliquid.com/support/library/midi-sysex.html

Support inquiries should be posted at the Highly Liquid support forums. Project examples and other information is posted at the forums:

http://forum.highlyliquid.com/

2.0 Theory of Operation

MIDI CPU functionality can be understood in terms of automatically repeating Tasks performed by the MIDI CPU and the resulting Data which is used to generate MIDI output.

2.1 Tasks

After power-up and initialization, the MIDI CPU continuously performs the following tasks: **MIDI Merge, Matrix Select Signal Generation, Control Input Monitoring**, **MIDI Message Generation**, and **LED Control.**

2.1.1 MIDI Merge

Messages received by the MIDI CPU at the MIDI IN port are echoed in real time at the MIDI OUT port, interspersed with any locally generated messages. This allows the MIDI CPU to be combined with another MIDI controller or with additional MIDI CPU units to function as a single unit with a single MIDI OUT port.

All messages received by the MIDI CPU at the MIDI IN port are echoed at the MIDI OUT port with the exception of MIDI **SysEx** messages and MIDI "**Active Sense**" messages. **Active Sense** messages are generated locally per the MIDI specification.

The MIDI CPU employs an efficient merge algorithm:

- Typical latency between an incoming message and the echoed output approaches the theoretical minimum of approximately 300µs.
- Merged messages and locally generated messages share the same "running status" when possible, resulting in a compact output data stream.
- Merged messages preempt locally-generated messages, allowing successful processing of a saturated data stream.

2.1.2 Matrix Select Signal Generation

When configured as a *matrix select output*, a MIDI CPU control terminal generates a matrix select logic signal. Matrix select outputs are used together with logic inputs in order to monitor a switch matrix.

2.1.3 Control Input Monitoring

The MIDI CPU continuously monitors each control terminal configured as an analog or logic input. Each input signal is used in up to two ways:

- The current value of the input, updated in real time, can be incorporated into outbound MIDI messages. See Data.
- A change to the input state can trigger the generation of a MIDI message.

2.1.4 MIDI Message Generation

The MIDI CPU generates MIDI messages based on user configuration. Message generation is triggered by changes to control terminal input. See Control Terminal Configuration and Global Configuration. Data acquired during operation can be incorporated into the generated messages. See Data.

2.1.5 LED Control

In addition to the built-in and optional remote Activity LED, the MIDI CPU can control multiple matrixed indicator LEDs and/or 7-segment LED displays. See ***MIDI CPU Hardware User Manual*** for wiring information. See Logic Output LED Control Modes for more details.

2.2 Data

The MIDI CPU has several internal memory locations called *data registers* that contain real-time data from control terminal inputs or that control some aspect of MIDI CPU behavior. Each data register contains up to a single byte (8 bits) and is identified by a unique hex *address*.

When the MIDI CPU generates a MIDI message, information in the message (note number & velocity, CC number & value, program number, etc) can be formed from fixed values and/or the contents of data registers.

Each data register is described below. Control Terminal Configuration and Global Configuration determine how the MIDI CPU uses the data registers.

2.2.1 Logic Input Data

Logic input data registers contain the current states of any control terminals configured as logic inputs. See **Table 2.2.1-a**. If the control terminal input state is low (0V or grounded), the corresponding data register bit is '0'. If the control terminal input state is high (V_{REG} or unconnected), the corresponding bit is '1'. For a control terminal not configured as a logic input, the value of the corresponding logic input data register bit is unknown.

Table 2.2.1-a: Logic Input Data Registers

Register Address	Control Terminal #							
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
00h	7	6	5	4	3	2	1	0
01h	15	14	13	12	11	10	9	8
02h	23	22	21	20	19	18	17	16

2.2.2 Analog Input Data

Each *analog input data register* contains a 7-bit representation of the input voltage on the corresponding control terminal. See **Table 2.2.2-a**.

For a control terminal not configured as an analog input, the corresponding analog input data register contains the value 0x7F.

The minimum and maximum value for each register is user-configurable. See Data Register Configuration.

Table 2.2.2-a: Analog Input Data Registers

Register Address	Control Terminal #
03h	16
04h	17
05h	18
06h	19
07h	20
08h	21
09h	22
0Ah	23
0Bh	10
0Ch	11
0Dh	9
0Eh	12
0Fh	8
10h	13

2.2.3 Multipurpose Data

Each *multipurpose data register* can contain a 7-bit or 8-bit value.

If a control terminal pair is configured for rotary encoder input, the 7-bit value in the corresponding data register 11h-1Ch is incremented or decremented by input activity. See **Table 2.2.3-a**.

Control terminals configured for certain modes use values in data registers 1Eh-20h to from MIDI messages as described in **Table 2.2.3-b**.

Data registers 21h-28h are automatically updated with certain values during operation by the MIDI CPU. See **Table 2.2.3-c**.

If the MIDI CPU control terminal configuration leaves any multipurpose register unused, the register is available for "general purpose" use: the register value can be altered via arithmetic operations and the register contents can be used to form MIDI or LED output. For bitwise operations, the register is treated as an 8-bit value. For other arithmetic operations, the register is assumed to contain a 7-bit value.

The power-up value, minimum & maximum value, and round-robin behavior for each register is user-configurable. See Data Register Configuration.

Table 2.2.3-a: Multipurpose Data Registers 11h-1Ch Primary Function

Register Address	Primary Function
11h	Encoder data (control terminals 0,1)
12h	Encoder data (control terminals 2,3)
13h	Encoder data (control terminals 4,5)
14h	Encoder data (control terminals 6,7)
15h	Encoder data (control terminals 8,9)
16h	Encoder data (control terminals 10,11)
17h	Encoder data (control terminals 12,13)
18h	Encoder data (control terminals 14,15)
19h	Encoder data (control terminals 16,17)
1Ah	Encoder data (control terminals 18,19)
1Bh	Encoder data (control terminals 20,21)
1Ch	Encoder data (control terminals 22,23)

Table 2.2.3-b: Multipurpose Data Registers 1Eh-20h Primary Function

Register Address	Primary Function
1Eh	Note velocity for modes Matrix Select: Note On / Note Off and Matrix Select: Note Toggle
1Fh	CC "on" value for modes Logic Input Trigger: Control Change (CC) Toggle, Matrix Select: Control Change (CC) On/Off, and Matrix Select: Control Change (CC) Toggle
20h	CC "off" value for modes Logic Input Trigger: Control Change (CC) Toggle, Matrix Select: Control Change (CC) On/Off, and Matrix Select: Control Change (CC) Toggle

Table 2.2.3-c: Multipurpose Data Registers 21h-28h Primary Function

Register Address	Primary Function
21h	*Note number* of the last **Note On** message generated by the MIDI CPU
22h	*Note velocity* of the last **Note On** message generated by the MIDI CPU
23h	*CC number* of the last **Control Change (CC)** message generated by the MIDI CPU
24h	*CC value* of the last **Control Change (CC)** message generated by the MIDI CPU
25h	*Program number* of the last **Program Change** message generated by the MIDI CPU
26h	Reserved for future use.
27h	Reserved for future use.
28h	Reserved for future use.

2.2.4 Configuration Layer Control Register

Each control terminal is associated with multiple configuration Layers. A configuration layer is globally enabled or disabled by setting or clearing the corresponding bit of the *Configuration Layer Control Register*. See **Table 2.2.4-a**.

The power-up value for the register is user-configurable. See Data Register Configuration.

Table 2.2.4-a: Configuration Layer Control Registers

Register Address	Configuration Layer			
	Bit 3	Bit 2	Bit 1	Bit 0
1Dh	03h	02h	01h	00h

2.2.5 Indirect Address Registers

Registers 29h and 2Ah can be used together to implement indirect addressing of MIDI CPU data registers.

Register 29h is a "pointer" that can contain the address of a different register between 00h and 28h. Any action using the address 2Ah actually takes place at the address contained in register 29h. There is no physical register at location 2Ah. See **Table 2.2.5-a**.

The power-up value, minimum and maximum value, and round-robin behavior of register 29h is user-configurable. See Data Register Configuration.

Table 2.2.5-a: Indirect Address Register

Register Address	Function
29h	Indirect Pointer
2Ah	Indirect Data Access

3.0 Control Terminal Configuration

Each MIDI CPU control terminal is associated with a *control terminal configuration* which is defined by one or more *configuration chunks*. The user sets the configuration for each control terminal via MIDI SysEx message. Configuration is retained when the MIDI CPU is disconnected from a power supply.

The MIDI CPU acknowledges an incoming configuration message via the Activity Indicator LED.

New users may benefit from first reviewing the Configuration Examples and Technical Support.

The factory default configuration for each control terminal is described in Factory Default Control Terminal Configuration.

3.1 SysEx Message Format

Figure 3.1-1 shows the format of the *Control Terminal Configuration SysEx Message*. The message consists of a fixed header, a variable length body, and a fixed footer.

The body of the message consists of a layer number followed by one or more 6-byte configuration chunks, each specifying a behavior for a single MIDI CPU control terminal. The configuration of control terminals not specified in the message will remain unchanged.

When the MIDI CPU configuration is updated via a MIDI SysEx message, the device will reset, followed by the LED self-test. If no LED activity is observed, then the attempted configuration was unsuccessful.

Figure 3.1-1: Control Terminal Configuration SysEx Message Format (Hex)

Header (6 bytes)	**Layer (1 byte)**	**Configuration Chunk (repeat as desired)**						**Footer (1 byte)**
F0 00 01 5D 04 01	*yy*	*nn*	*tt*	*mm*	*ch*	*d0*	*d1*	F7

3.1.1 Layer (*yy*)

Configuration *layers* allow a single control terminal to be associated with multiple functions. Use of multiple layers is optional and can vary by control terminal.

There are 4 control terminal configuration layers. Valid values for *yy* are 00h, 01h, 02h, or 03h.

A control terminal's layer 00h configuration directs its initialization at power-up. Configuration layers 01h-03h, if used, must specify the same category of operation (logic input, analog input, matrix select output, or LED common output) as layer 00h. Specify Logic Input: Data Only mode for configuration layers which are not needed.

During operation, the MIDI CPU will attempt to execute the configuration for each layer 00h-03h in sequence, ignoring any configuration layers disabled via the Configuration Layer Control Register.

3.1.2 Control Terminal Identifier (*nn*)

Parameter *nn* specifies the MIDI CPU control terminal associated with the configuration chunk. Values 00h thru 17h correspond to control terminal numbers 0 thru 23.

3.1.3 Transition (*tt*)

Control terminals configured in Logic Input Trigger Modes can trigger one event on the input signal transition from high to low, and a different event on the transition from low to high. For example, a control terminal might trigger a **Note On** message when the input signal changes from high to low, and a **Note Off** message when the signal changes from low to high. For Logic Input Trigger Modes only, the value of *tt* associates the configuration chunk with one transition or the other. See **Table 3.1.3-a.**

When configuring a control terminal for any mode other than a Logic Input Trigger Mode, use the value *tt* = 00h.

Table 3.1.3-a: Logic Input Trigger Mode Transitions

tt	Associated Transition
00h	High to Low (Falling Edge)
01h	Low to High (Rising Edge)

3.1.4 Mode (*mm*)

Parameter *mm* specifies a function for the control terminal specified by *nn*. Valid range is 00h to 7Fh. The meanings of *d0* and *d1* depend on the value of *mm*.

Each control terminal mode falls into one of three categories:

- **Logic Input:** Control terminal is configured as a high-impedance digital logic input with a built-in pull-up resistor. The input state is either a binary '1' or '0' that is accessible from the Logic Input Data Register associated with the control terminal. Some modes cause a MIDI message to be generated when the input state changes.
- **Analog Input:** Control terminal is configured as an analog input. The input signal is continuously converted to a 7-bit digital value that is accessible from the Register associated with the control terminal. Some modes cause a MIDI message to be generated when the input state changes.
- **Logic Output:** Control terminal is configured as a digital logic output. The resulting signal can be used together with logic inputs to monitor a switch matrix, or can be used to drive an LED matrix.

Wiring examples for each category can be found in ***MIDI CPU Hardware User Manual***.

Individual modes are described in sections 3.2 thru 3.5.

3.1.5 MIDI Channel (*ch*)

Some values of *mm* associate the control terminal with a channel-specific MIDI message. For these modes, the value of *ch* specifies which MIDI channel is used. *ch* can direct the MIDI CPU to generate messages using either a fixed channel value or the channel value specified by the MIDI Channel Selection Jumper. See **Table 3.1.5-a**.

The value of *ch* is ignored for modes which are not associated with a channel-specific MIDI message.

Table 3.1.5-a: MIDI Channel by *ch* Value

ch	MIDI Channel
00h	From MIDI Channel Selection Jumper
01h	1
02h	2
03h	3
04h	4
05h	5
06h	6
07h	7
08h	8
09h	9
0Ah	10
0Bh	11
0Ch	12
0Dh	13
0Eh	14
0Fh	15
10h	16

3.1.6 Data (d*0, d1*)

Depending on the value of *mm, d0* and *d1* can specify fixed values, data register addresses, or other information which determines the behavior of the control terminal. Details are provided in the description of each mode.

3.2 Logic Input Trigger Modes

Logic Input Trigger Modes configure the control terminal specified by *nn* as a logic input. Changes to input state may trigger MIDI output or other events.

3.2.1 Logic Input Trigger: Note Off

A **Note Off** message (status 80h) is generated when the control terminal input state changes. Note numbers are subject to MIDI Number Remapping.

Note: **Note Off** velocity data is ignored by most MIDI devices.

For efficient operation, do not configure the MIDI CPU to send **Note Off** messages. Instead, use **Note On** messages with a fixed velocity value of 00h.

mm	***d0* Meaning**	***d1* Meaning**
40h	Fixed note number	Fixed note velocity
41h	Data register address for note number	Fixed note velocity
42h	Fixed note number	Data register address for note velocity
43h	Data register address for note number	Data register address for note velocity

3.2.2 Logic Input Trigger: Note On

A **Note On** message is generated when the control terminal input state changes. Note numbers are subject to MIDI Number Remapping.

Note: A **Note On** message with a velocity of 00h is equivalent to a **Note Off** message.

mm	***d0* Meaning**	***d1* Meaning**
44h	Fixed note number	Fixed note velocity
45h	Data register address for note number	Fixed note velocity
46h	Fixed note number	Data register address for note velocity
47h	Data register address for note number	Data register address for note velocity

3.2.3 Logic Input Trigger: Aftertouch

An **Aftertouch** message is generated when the control terminal input state changes. Note numbers are subject to MIDI Number Remapping.

mm	*d0* Meaning	*d1* Meaning
48h	Fixed note number	Fixed pressure value
49h	Data register address for note number	Fixed pressure value
4Ah	Fixed note number	Data register address for pressure value
4Bh	Data register address for note number	Data register address for pressure value

3.2.4 Logic Input Trigger: Control Change (CC)

A **Control Change (CC)** message is generated when the control terminal input state changes.

mm	*d0* Meaning	*d1* Meaning
4Ch	Fixed CC number	Fixed CC value
4Dh	Data register address for CC number	Fixed CC value
4Eh	Fixed CC number	Data register address for CC value
4Fh	Data register address for CC number	Data register address for CC value

3.2.5 Logic Input Trigger: Control Change (CC) Toggle

A **Control Change (CC)** message is generated when the control terminal input state changes. The CC value alternates between the global CC "on" value and the global CC "off" value in Multipurpose Data registers 1Fh and 20h.

mm	*d0* Meaning	*d1* Meaning
56h	Fixed CC number	Ignored
57h	Data register address for CC number	Ignored

3.2.6 Logic Input Trigger: Program Change

A **Program Change** message is generated when the control terminal input state changes.

For modes 52h and 53h, the program number is the result of adding the values specified by *d0* and *d1*.

mm	*d0* Meaning	*d1* Meaning
50h	Fixed program number	Ignored
51h	Data register address for program number	Ignored
52h	Fixed program number	Data register address for program number addend
53h	Data register address for program number	Data register address for program number addend

3.2.7 Logic Input Trigger: Channel Pressure

a **Channel Pressure** message is generated when the control terminal input state changes.

mm	*d0* Meaning	*d1* Meaning
54h	Fixed pressure value	Ignored
55h	Data register address for pressure value	Ignored

3.2.8 Logic Input Trigger: Pitch Wheel

A **Pitch Wheel** message is generated when the control terminal input state changes.

Pitch Wheel messages contain 14 bits of position data. The high-order bits (bits 13 thru 7) have a large impact on pitch change and the low-order bits (bits 6 thru 0) have a small impact.

For *mm* = 5Ah, it is recommended that *d0* = 00h.

mm	*d0* Meaning	*d1* Meaning
58h	Fixed value for position bits 6-0	Fixed value for position bits 13-7
59h	Data register address for position bits 6-0	Fixed value for position bits 13-7
5Ah	Fixed value for position bits 6-0	Data register address for position bits 13-7
5Bh	Data register address for position bits 6-0	Data register address for position bits 13-7

3.2.9 Logic Input Trigger: System Exclusive (SysEx)

An arbitrary complete **SysEx** message is generated when the control terminal input state changes.

The generated **SysEx** message consists of the standard "SysEx Begin" status byte (F0h) followed by data from the SysEx Data Buffer as specified by *d0* & *d1* and closed with the standard "SysEx End" status byte (F7h).

mm	*d0* Meaning	*d1* Meaning
5Ch	SysEx buffer starting address	Data length (bytes)

3.2.10 Logic Input Trigger: SysEx Begin Fragment

The beginning fragment of an arbitrary **SysEx** message is generated when the control terminal input state changes.

The generated **SysEx** fragment consists of the standard "SysEx Begin" status byte (F0h) followed by data from the SysEx Data Buffer as specified by *d0* & *d1*. The fragment must be followed by raw data via Logic Input Trigger: SysEx Raw Data Byte and a SysEx end fragment via Logic Input Trigger: SysEx End Fragment in order to complete the SysEx message.

mm	*d0* Meaning	*d1* Meaning
5Dh	SysEx buffer starting address	Data length (bytes)

3.2.11 Logic Input Trigger: SysEx Raw Data Byte

A single SysEx data byte is generated when the control terminal input state changes. The data must be preceded by a SysEx begin fragment via Logic Input Trigger: SysEx Begin Fragment and must be followed by a SysEx end fragment via Logic Input Trigger: SysEx End Fragment.

mm	*d0* Meaning	*d1* Meaning
5Eh	Data Register Address for Data Byte	Ignored

3.2.12 Logic Input Trigger: SysEx End Fragment

The end fragment of an arbitrary **SysEx** message is generated when the control terminal input state changes.

The generated **SysEx** fragment consists of data from the SysEx Data Buffer as specified by *d0* & *d1* followed by the standard "SysEx End" status byte (F7h). The fragment must be preceded by a SysEx begin fragment via Logic Input Trigger: SysEx Begin Fragment and optionally by raw data via Logic Input Trigger: SysEx Raw Data Byte.

mm	*d0* Meaning	*d1* Meaning
5Fh	SysEx buffer starting address	Data length (bytes)

3.2.13 Logic Input Trigger: MIDI Start

A **MIDI Start** message is generated when the control terminal input state changes.

mm	*d0* Meaning	*d1* Meaning
64h	Ignored	Ignored

3.2.14 Logic Input Trigger: MIDI Stop

A **MIDI Stop** message is generated when the control terminal input state changes.

mm	*d0* Meaning	*d1* Meaning
65h	Ignored	Ignored

3.2.15 Logic Input Trigger: MIDI Start / MIDI Stop Toggle

Alternating **MIDI Start** and **MIDI Stop** messages are generated when the control terminal input state changes.

mm	*d0* Meaning	*d1* Meaning
66h	Ignored	Ignored

3.2.16 Logic Input Trigger: MIDI Clock

A **MIDI Clock** message is generated when the control terminal input state changes.

mm	*d0* Meaning	*d1* Meaning
67h	Ignored	Ignored

3.2.17 Logic Input Trigger: Analog Continuous Note Start

If the control terminal specified by *d0* is configured in Analog Input: Continuous Note mode, note generation begins.

mm	*d0* Meaning	*d1* Meaning
68h	Analog Input Control Terminal #	Ignored

3.2.18 Logic Input Trigger: Analog Continuous Note Stop

If the control terminal specified by *d0* is configured in Analog Input: Continuous Note mode, note generation halts.

mm	*d0* Meaning	*d1* Meaning
69h	Analog Input Control Terminal #	Ignored

3.2.19 Logic Input Trigger: Encoder Continuous Note Start

If the control terminal pair specified by *d0* is configured in Encoder Input: Continuous Note mode, note generation begins. For *d0*, use the lower of the two control terminal numbers in the pair.

mm	*d0* Meaning	*d1* Meaning
6Ah	Encoder Input Control Terminal #	Ignored

3.2.20 Logic Input Trigger: Encoder Continuous Note Stop

If the control terminal specified by *d0* is configured in Encoder Input: Continuous Note mode, note generation halts. For *d0*, use the lower of the two control terminal numbers in the pair.

mm	*d0* Meaning	*d1* Meaning
6Bh	Encoder Input Control Terminal #	Ignored

3.2.21 Logic Input Trigger: MIDI Reset

A **MIDI Reset** message is generated when the control terminal input state changes.

mm	*d0* Meaning	*d1* Meaning
7Eh	Ignored	Ignored

3.2.22 Logic Input Trigger: Data Register: Increment

The register specified by *d0* is incremented by the amount *d1*. No MIDI event is triggered.

mm	*d0* Meaning	*d1* Meaning
70h	Data Register Address	Increment Value

3.2.23 Logic Input Trigger: Data Register: Decrement

The register specified by *d0* is decremented by the amount *d1*. No MIDI event is triggered.

mm	*d0* Meaning	*d1* Meaning
71h	Data Register Address	Decrement Value

3.2.24 Logic Input Trigger: Data Register: Sum Values

The contents of register *d0* is summed with the contents of register *d1* and the result is stored in *d0*. No MIDI event is triggered.

mm	*d0* Meaning	*d1* Meaning
72h	"Sum" Data Register Address	"Addend" Data Register Address

3.2.25 Logic Input Trigger: Data Register: Copy Value

The contents of register *d1* is stored in register *d0*. No MIDI event is triggered.

mm	*d0* Meaning	*d1* Meaning
73h	"To" Data Register Address	"From" Data Register Address

3.2.26 Logic Input Trigger: Data Register: Store Value

The value specified by *d1* is stored in the register specified by *d0*. No MIDI event is triggered.

mm	*d0* Meaning	*d1* Meaning
74h	Data Register Address	Value

3.2.27 Logic Input Trigger: Data Register: Clear / Set / Toggle Bit

A bit in the register specified by *d0* is cleared, set or toggled when the control terminal input state changes. No MIDI event is triggered.

The value of *d1* specifies both the bit to be altered, and whether the bit should be cleared, set, or toggled. Values 00h-07h cause the bit numbered *d1* to be cleared. Values 08h-0Fh cause the bit numbered (*d1*-8) to be set. Values 10h-17h cause the bit numbered (*d1*-16) to be toggled.

mm	*d0* Meaning	*d1* Meaning
76h	Data Register Address	Clear / Set / Toggle; Bit #

3.2.28 Logic Input Trigger: Global Refresh

For each control terminal, the MIDI CPU generates the MIDI output that would result from a *change to* the control terminal's current input state, but does not alter the contents of any MIDI CPU data registers.

Use this mode to “sync” a downstream device to the current MIDI CPU state.

mm	***d0* Meaning**	***d1* Meaning**
7Dh	Ignored	Ignored

3.2.29 Logic Input: Data Only

The following mode configures the control terminal as a logic input. The input state is available as Logic Input Data, but no event is triggered by state changes.

mm	***d0* Meaning**	***d1* Meaning**
7Fh	Ignored	Ignored

3.3 Logic Input Rotary Encoder Modes

Logic Input Rotary Encoder Modes configure the control terminal specified by *nn* as a logic input. For proper operation, both control terminals in an encoder input pair must be given the same configuration. See **Table 2.2.3-a** for a list of control terminal encoder input pairs. When the attached rotary encoder is adjusted, the value in the control terminal pair's corresponding data register is incremented or decremented.

The initial value upon power-up for each data register is user configurable. See Data Register Configuration.

3.3.1 Encoder Input: Continuous Note

A continuous stream of MIDI notes is generated. The encoder data register for the control terminal determines the note number of the current note. Must be used in conjunction with modes Logic Input Trigger: Encoder Continuous Note Start and Logic Input Trigger: Encoder Continuous Note Stop.

If note generation has begun, input from the rotary encoder simultaneously causes the previous MIDI note to be canceled with a **Note Off** message and a new note to be initiated with a **Note On** message. Note numbers are subject to MIDI Number Remapping.

mm	***d0* Meaning**	***d1* Meaning**
10h	Ignored	Fixed note velocity
11h	Ignored	Data register address for note velocity

3.3.2 Encoder Input: Control Change (CC)

A **Control Change** message is generated when input is received from the rotary encoder.

mm	***d0* Meaning**	***d1* Meaning**
14h	Fixed CC number	Data register address for CC value
15h	Data register address for CC number	Data register address for CC value

3.3.3 Encoder Input: Program Change

A **Program Change** message is generated when input is received from the rotary encoder.

mm	***d0* Meaning**	***d1* Meaning**
16h	Data register address for program number	Ignored

3.3.4 Encoder Input: Channel Pressure

A **Channel Pressure** message is generated when input is received from the rotary encoder.

mm	*d0* Meaning	*d1* Meaning
17h	Data register address for pressure value	Ignored

3.3.5 Encoder Input: Data Register: Increment / Decrement

When clockwise input is received from the encoder, the register specified by *d0* is incremented by the value *d1*. When counter-clockwise input is received from the encoder, the register specified by *d0* is decremented by the value *d1*. No MIDI event is triggered.

mm	*d0* Meaning	*d1* Meaning
1Ah	Data Register Address	Increment / Decrement Value

3.3.6 Encoder Input: Data Register: Copy Value

The contents of register *d1* is stored in register *d0* when input is received from the encoder. No MIDI event is triggered.

mm	*d0* Meaning	*d1* Meaning
1Ch	“To” Data Register Address	“From” Data Register Address

3.3.7 Encoder Input: Data Register: Store Value

The value specified by *d1* is stored in the register specified by *d0* when input is received from the encoder. No MIDI event is triggered.

mm	*d0* Meaning	*d1* Meaning
1Dh	Data Register Address	Value

3.3.8 Encoder Input: Data Register: Clear / Set / Toggle Bit

A bit in the register specified by *d0* is cleared, set or toggled when input is received from the encoder. No MIDI event is triggered.

The value of *d1* specifies both the bit to be altered, and whether the bit should be cleared, set, or toggled. Values 00h-07h cause the bit numbered *d1* to be cleared. Values 08h-0Fh cause the bit numbered (*d1*-8) to be set. Values 10h-17h cause the bit numbered (*d1*-16) to be toggled.

mm	***d0* Meaning**	***d1* Meaning**
1Eh	Data Register Address	Clear / Set / Toggle; Bit #

3.3.9 Encoder Input: Data Only

When input from the rotary encoder is received, the value in the corresponding data register is adjusted, but no MIDI message is generated.

mm	***d0* Meaning**	***d1* Meaning**
1Fh	Ignored	Ignored

3.4 Analog Input Modes

Analog Input Modes configure the control terminal specified by *nn* as an analog input. Changes to input state may trigger MIDI output or other events.

Only certain control terminals can be configured as analog inputs – see ***MIDI CPU Hardware User Manual***.

3.4.1 Analog Input: Continuous Note

A continuous stream of MIDI notes is generated. The analog data register for the control terminal determines the note number of the current note. Must be used in conjunction with modes Logic Input Trigger: Analog Continuous Note Start and Logic Input Trigger: Analog Continuous Note Stop.

If note generation has begun, change to the analog input simultaneously causes the previous MIDI note to be canceled with a **Note Off** message and a new note to be initiated with a **Note On** message. Note numbers are subject to MIDI Number Remapping.

mm	***d0* Meaning**	***d1* Meaning**
00h	Ignored	Fixed note velocity
01h	Ignored	Data register address for velocity

3.4.2 Analog Input: Aftertouch

Change to the analog input signal causes an **Aftertouch** message to be generated. Note numbers are subject to MIDI Number Remapping.

mm	***d0* Meaning**	***d1* Meaning**
02h	Fixed note number	Data register address for pressure value
03h	Data register address for note number	Data register address for pressure value

3.4.3 Analog Input: Control Change (CC)

Whenever the analog input signal changes, a **Control Change** message is generated.

mm	***d0* Meaning**	***d1* Meaning**
04h	Fixed CC number	Data register address for CC value
05h	Data register address for CC number	Data register address for CC value

3.4.4 Analog Input: Program Change

Whenever the analog input signal changes, a **Program Change** message is generated.

mm	*d0* Meaning	*d1* Meaning
06h	Data register address for program number	Ignored

3.4.5 Analog Input: Channel Pressure

Whenever the analog input signal changes, a **Channel Pressure** message is generated.

mm	*d0* Meaning	*d1* Meaning
07h	Data register address for pressure number	Ignored

3.4.6 Analog Input: Pitch Wheel

Whenever the analog input signal changes, a **Pitch Wheel** message is generated.

mm	*d0* Meaning	*d1* Meaning
08h	Ignored	Data register address for position value

3.4.7 Analog Input: Data Register: Copy Value

Whenever the analog input signal changes, the contents of register *d1* is stored in register *d0*. No MIDI event is triggered.

mm	*d0* Meaning	*d1* Meaning
0Ch	“To” Data Register Address	“From” Data Register Address

3.4.8 Analog Input: Data Register: Store Value

Whenever the analog input signal changes, the value specified by *d1* is stored in the register specified by *d0*. No MIDI event is triggered.

mm	*d0* Meaning	*d1* Meaning
0Dh	Data Register Address	Value

3.4.9 Analog Input: Data Register: Clear / Set / Toggle Bit

Whenever the analog input signal changes, a bit in the register specified by *d0* is cleared, set or toggled. No MIDI event is triggered.

The value of *d1* specifies both the bit to be altered, and whether the bit should be cleared, set, or toggled. Values 00h-07h cause the bit numbered *d1* to be cleared. Values 08h-0Fh cause the bit numbered (*d1*-8) to be set. Values 10h-17h cause the bit numbered (*d1*-16) to be toggled.

mm	***d0* Meaning**	***d1* Meaning**
0Eh	Data Register Address	Clear / Set / Toggle; Bit #

3.4.10 Analog Input: Data Only

Analog input is converted to a 7-bit value and stored in the analog data register corresponding to the control terminal, but no MIDI message is generated.

mm	***d0* Meaning**	***d1* Meaning**
0Fh	Ignored	Ignored

3.5 Logic Output Matrix Select Modes

A *Logic Output Matrix Select Mode* causes the associated control terminal to periodically generate a matrix select output signal, allowing the MIDI CPU to monitor the states of switches in a switch matrix.

Only certain control terminals can be configured as logic outputs – see ***MIDI CPU Hardware User Manual***.

Modes with a negative ("−") pulse polarity generate 0V pulses. Data lines are automatically "pulled up" via the MIDI CPU on-board pull-up resistors.

Modes with a positive ("+") pulse polarity generate pulses of voltage V_{REG}. For these modes, Control Terminals 8-15 Pull-Up Resistor must be disabled for any control terminals used for matrix data. External pull-down resistors must be added for the matrix data control terminals.

Due to ease of implementation, negative pulse polarity is strongly recommended for applications where the user can choose the configuration of the switch matrix.

When a control terminal is configured in a Logic Output Matrix Select Mode, the following "select cycle" will repeat continuously:

1. Control terminal begins in a high-impedance (input) state. (For "positive pulse" modes, the terminals begins in an active low-output state.)
2. A select pulse from the control terminal is initiated.
3. The MIDI CPU takes a "snapshot" of the switch state data register associated with the mode.
4. Control terminal returns to high-impedance (input) state. (For "positive pulse" modes, the terminals returns to an active low-output state.)
5. The Matrix Data Bitmask is applied to the snapshot.
6. MIDI messages are generated according to any changed switch states.
7. The output remains inactive (high-impedance) while select pulses are generated by other matrix select control terminals.
8. Cycle repeats with step 1.

The frequency of the select cycle (or "matrix sampling rate") varies depending on MIDI CPU configuration and input activity. The typical rate approaches 1,000 select cycles per second.

3.5.1 Matrix Select: Note On / Note Off

When a switch state changes from high to low, a **Note On** message is generated. When a switch state changes from low to high, a **Note Off** message is generated.

The note number for each switch is determined by the value of *d0*. Switch state data register bit 0 generates messages with note number *d0,* bit 1 generates messages with note number *d0* + 1, bit 2 generates messages with note number *d0* + 2, etc. Note numbers are subject to MIDI Number Remapping.

Note velocity is determined by the value in Multipurpose Data register 1Eh.

mm	Pulse Polarity	Switch State Data Register	*d0* Meaning	*d1* Meaning
2Ch	–	00h	Note number for data bit 0	Matrix Data Bitmask
2Dh	–	01h	Note number for data bit 0	Matrix Data Bitmask
2Eh	–	02h	Note number for data bit 0	Matrix Data Bitmask
2Fh	+	01h	Note number for data bit 0	Matrix Data Bitmask

3.5.2 Matrix Select: Note Toggle

Alternating **Note On** and **Note Off** messages are generated when a switch state changes from high to low.

The note number for each switch is determined by the value of *d0*. Switch state data register bit 0 generates messages with note number *d0,* bit 1 generates messages with note number *d0* + 1, bit 2 generates messages with note number *d0* + 2, etc. Note numbers are subject to MIDI Number Remapping.

Note velocity is determined by the value in Multipurpose Data register 1Eh.

mm	Pulse Polarity	Switch State Data Register	*d0* Meaning	*d1* Meaning
30h	–	00h	Note number for data bit 0	Matrix Data Bitmask
31h	–	01h	Note number for data bit 0	Matrix Data Bitmask
32h	–	02h	Note number for data bit 0	Matrix Data Bitmask
33h	+	01h	Note number for data bit 0	Matrix Data Bitmask

3.5.3 Matrix Select: Control Change (CC) On/Off

When a switch state changes, a **Control Change** message is generated.

The CC number for each switch is determined by the value of *d0*. Switch state data register bit 0 generates messages with CC number *d0,* bit 1 generates messages with CC number *d0* + 1, bit 2 generates messages with CC number *d0* + 2, etc.

When a switch state changes from high to low, the CC "on" value is used. When a switch state changes state from low to high, the CC "off" value is used. CC on/off values are determined by the global CC "on" value and the global CC "off" value in Multipurpose Data registers 1Fh and 20h.

mm	Pulse Polarity	Switch State Data Register	*d0* Meaning	*d1* Meaning
34h	–	00h	CC number for data bit 0	Matrix Data Bitmask
35h	–	01h	CC number for data bit 0	Matrix Data Bitmask
36h	–	02h	CC number for data bit 0	Matrix Data Bitmask
37h	+	01h	CC number for data bit 0	Matrix Data Bitmask

3.5.4 Matrix Select: Control Change (CC) Toggle

When a switch state changes from high to low, a **Control Change** message is generated.

CC number for each switch is determined by the value of *d0*. Switch state data register bit 0 generates messages with CC number *d0,* bit 1 generates messages with CC number *d0* + 1, bit 2 generates messages with CC number *d0* + 2, etc.

The CC value alternates between the global CC "on" value and the global CC "off" value in Multipurpose Data registers 1Fh and 20h.

mm	Pulse Polarity	Switch State Data Register	*d0* Meaning	*d1* Meaning
38h	–	00h	CC number for data bit 0	Matrix Data Bitmask
39h	–	01h	CC number for data bit 0	Matrix Data Bitmask
3Ah	–	02h	CC number for data bit 0	Matrix Data Bitmask
3Bh	+	01h	CC number for data bit 0	Matrix Data Bitmask

3.5.5 Matrix Select: Program Change

When a switch state changes from high to low, a **Program Change** message is generated.

The program number for each switch is determined by the value of *d0.* Switch state data register bit 0 generates messages with program number *d0,* bit 1 generates messages with program number *d0* + 1, bit 2 generates messages with program number *d0* + 2, etc.

mm	Pulse Polarity	Switch State Data Register	*d0* Meaning	*d1* Meaning
3Ch	–	00h	Program number for data bit 0	Matrix Data Bitmask
3Dh	–	01h	Program number for data bit 0	Matrix Data Bitmask
3Eh	–	02h	Program number for data bit 0	Matrix Data Bitmask
3Fh	+	01h	Program number for data bit 0	Matrix Data Bitmask

3.5.6 Matrix Data Bitmask

Logic Output Matrix Select Modes employ a *bitmask* specified by the value of *d1*.

When processing data from the switch state data register, the MIDI CPU can ignore certain data bits as directed by the bitmask. Bits with a bitmask value of '0' are ignored. Values for *d1* and the corresponding bitmasks are shown in **Table 3.5.6-a**.

When a switch matrix does not employ all 8 control terminals associated with a data register, the correct bitmask prevents the unused control terminals from interfering with matrix monitoring. The unused control terminals can therefore be configured for other purposes.

Table 3.5.6-a: Matrix Data Bitmasks

d1	Data Bitmask							
	Bit 7	**Bit 6**	**Bit 5**	**Bit 4**	**Bit 3**	**Bit 2**	**Bit 1**	**Bit 0**
00h	1	1	1	1	1	1	1	1
01h	1	1	1	1	1	1	1	0
02h	1	1	1	1	1	1	0	0
03h	1	1	1	1	1	0	0	0
04h	1	1	1	1	0	0	0	0
05h	1	1	1	0	0	0	0	0
06h	1	1	0	0	0	0	0	0
07h	1	0	0	0	0	0	0	0
08h	0	1	1	1	1	1	1	1
09h	0	0	1	1	1	1	1	1
0Ah	0	0	0	1	1	1	1	1
0Bh	0	0	0	0	1	1	1	1
0Ch	0	0	0	0	0	1	1	1
0Dh	0	0	0	0	0	0	1	1
0Eh	0	0	0	0	0	0	0	1

3.6 Logic Output LED Control Modes

The following modes configure the control terminal specified by *nn* for logic output.

The MIDI CPU can control up to 64 LEDs arranged in an 8x8 matrix. The matrix can be comprised of a mixture of individual indicator LEDs, 7-segment numerical displays, and 8-segment "bar graph" displays.

LED control functionality requires that at least one control terminal (0-7) be configured for LED Data Output and at least one control terminal (8-15) be configured for LED Common Output.

Control terminals configured for LED control output are subject the electrical limitations described in ***MIDI CPU Hardware User Manual***.

3.6.1 LED Data Output

The control terminal generates a control signal that drives an attached LED matrix. Must be used in conjunction with control terminals configured for LED Common Output.

Note: Only control terminals 0-7 can be configured for LED data output. If one or more control terminals are configured for LED data output, control terminals 0-7 cannot be configured as matrix select outputs.

mm	***d0* Meaning**	***d1* Meaning**
2Bh	Ignored	Ignored

3.6.2 LED Common Output

The control terminal generates a "common" control signal for up to 8 LEDs arranged in either a common-anode or common-cathode configuration. The data signal for each LED is automatically generated by control terminals configured for LED Data Output.

The contents of the data register at address *d1* are formatted as specified by *d0*.

Note: Only control terminals 8-15 can be configured for LED common output.

mm	***d0* Meaning**	***d1* Meaning**
2Ah	LED Output Format	LED State Register

3.6.3 LED Output Format

LED common output mode employs an *output format* specified by the value of *d0*. The output format encodes all of the following:

- The polarity of the attached LED array ("common anode" or "common cathode")
- Whether or not the common signal is externally inverted
- The *display type* for the value in the LED state register

When a *common anode* format is used, the control terminal drives the anodes of up to 8 LEDs. Each LED cathode must be connected to a different MIDI CPU control terminal configured for LED Data Output.

When a *common cathode* format is used, the control terminal drives the cathodes of up to 8 LEDs. Each LED anode must be connected to a different MIDI CPU control terminal configured for LED Data Output.

Because of the electrical limitations of the MIDI CPU, LED common output signals may be externally buffered by the user. *Inverted common* output formats are available for applications where "inverting" buffering methods are employed.

The following LED display types are available:

8-Bit Indication: The state of each LED reflects the corresponding bit in the LED state register. 1 = on; 0 = off.

3-Bit Indication: The 3 lowest-order bits in the LED state register are treated as an integer. The LED matching the integer is illuminated.

Unsigned 7-Segment Display (00h = "0"): The LED state register value is interpreted as an integer. Use these formats to display an unsigned value between 0 and 127.

Unsigned 7-Segment Display (00h = "1"): The LED state register value is interpreted as an integer. Use these formats to display an unsigned value between 1 and 128.

Unsigned 7-Segment Display (hex): The LED state register value is interpreted as an integer. Use these formats to display an unsigned hex value between 00h and 7Fh.

Signed 7-Segment Display (40h = "0"): The LED state register value is interpreted as an integer. Use these formats to display a signed value between -64 and 63.

Bar Graph: The LED state register value is interpreted as an integer. When the state register value is 0, only LED0 is illuminated. Each increase of 16 to the state register value causes the illumination of an additional LED.

The LED state summary for each display type is shown in **Table 3.6.3-a**. For any group of LEDs sharing the same LED common output, LED0-LED7 correspond to LED data output from control terminals 0-7. **Figure 3.6.3-1** shows the standard segment labels for a 7-segment digit.

Output formats and corresponding *d0* values are shown in **Table 3.6.3-b**. and **Table 3.6.3-c**.

Table 3.6.3-a: LED State Summary by Display Type

Display Type	LED State by LED State Register Value "v"							
	LED7	**LED6**	**LED5**	**LED4**	**LED3**	**LED2**	**LED1**	**LED0**
8-Bit Indication	v bit 7	v bit 6	v bit 5	v bit 4	v bit 3	v bit 2	v bit 1	v bit 0
3-Bit Indication	v%8 = 7	v%8 = 6	v%8 = 5	v%8 = 4	v%8 = 3	v%8 = 2	v%8 = 1	v % 8 = 0
7-Segment	off	seg. G	seg. F	seg. E	seg. D	seg. C	seg. B	seg. A
Bar Graph	v ≥ 112	v ≥ 96	v ≥ 80	v ≥ 64	v ≥ 48	v ≥ 32	v ≥ 16	on

Figure 3.6.3-1: 7-Segment Display Segment Labels

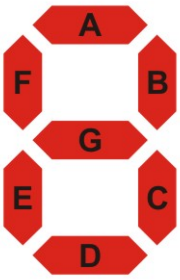


Table 3.6.3-b: LED Common Output Formats (Part 1 of 2)

d0	Common	Display Type
00h	Anode, Non-Inverted	8-Bit Indication
01h	Anode, Inverted	
02h	Cathode, Non-Inverted	
03h	Cathode, Inverted	
04h	Anode, Non-Inverted	Unsigned 7-Segment Display (00h = “0”) Hundreds Column
05h	Anode, Inverted	
06h	Cathode, Non-Inverted	
07h	Cathode, Inverted	
08h	Anode, Non-Inverted	Unsigned 7-Segment Display (00h = “0”) Tens Column
09h	Anode, Inverted	
0Ah	Cathode, Non-Inverted	
0Bh	Cathode, Inverted	
0Ch	Anode, Non-Inverted	Unsigned 7-Segment Display (00h = “0”) Ones Column
0Dh	Anode, Inverted	
0Eh	Cathode, Non-Inverted	
0Fh	Cathode, Inverted	
10h	Anode, Non-Inverted	Bar Graph
11h	Anode, Inverted	
12h	Cathode, Non-Inverted	
13h	Cathode, Inverted	
14h	Anode, Non-Inverted	Signed 7-Segment Display (40h = “0”) Sign Column
15h	Anode, Inverted	
16h	Cathode, Non-Inverted	
17h	Cathode, Inverted	
18h	Anode, Non-Inverted	Signed 7-Segment Display (40h = “0”) Tens Column
19h	Anode, Inverted	
1Ah	Cathode, Non-Inverted	
1Bh	Cathode, Inverted	
1Ch	Anode, Non-Inverted	Signed 7-Segment Display (40h = “0”) Ones Column
1Dh	Anode, Inverted	
1Eh	Cathode, Non-Inverted	
1Fh	Cathode, Inverted	

Table 3.6.3-c: LED Common Output Formats (Part 2 of 2)

d0	Common	Display Type
20h	Anode, Non-Inverted	3-Bit Indication
21h	Anode, Inverted	
22h	Cathode, Non-Inverted	
23h	Cathode, Inverted	
24h	Anode, Non-Inverted	Unsigned 7-Segment Display (00h = “1”) Hundreds Column
25h	Anode, Inverted	
26h	Cathode, Non-Inverted	
27h	Cathode, Inverted	
28h	Anode, Non-Inverted	Unsigned 7-Segment Display (00h = “1”) Tens Column
29h	Anode, Inverted	
2Ah	Cathode, Non-Inverted	
2Bh	Cathode, Inverted	
2Ch	Anode, Non-Inverted	Unsigned 7-Segment Display (00h = “1”) Ones Column
2Dh	Anode, Inverted	
2Eh	Cathode, Non-Inverted	
2Fh	Cathode, Inverted	
30h	Anode, Non-Inverted	Unsigned 7-Segment Display (hex) 16s Column
31h	Anode, Inverted	
32h	Cathode, Non-Inverted	
33h	Cathode, Inverted	
34h	Anode, Non-Inverted	Unsigned 7-Segment Display (hex) Ones Column
35h	Anode, Inverted	
36h	Cathode, Non-Inverted	
37h	Cathode, Inverted	

3.6.4 LED Output Examples

The figures below show four different configurations for a (2 common) x (8 data) LED matrix. Additional groups of 8 LEDs can be driven with the additional common lines, up to a total of 64 LEDs.

Figure 3.6.4-1: LED Matrix, Non-Inverted Common Anode

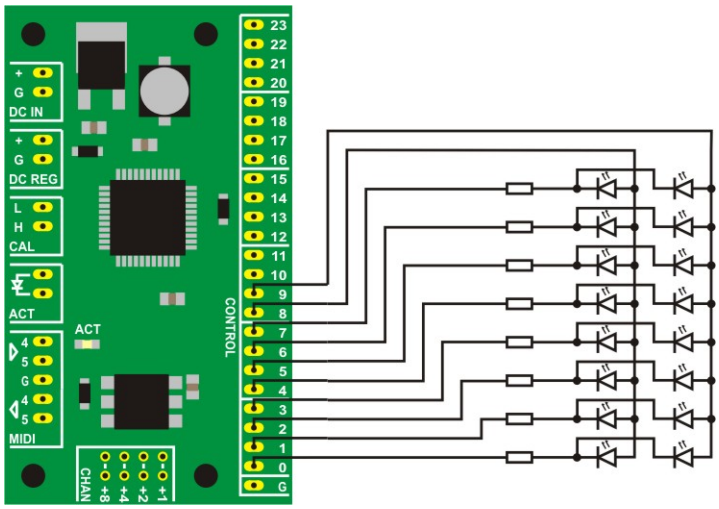


Figure 3.6.4-2: LED Matrix, Inverted Common Anode

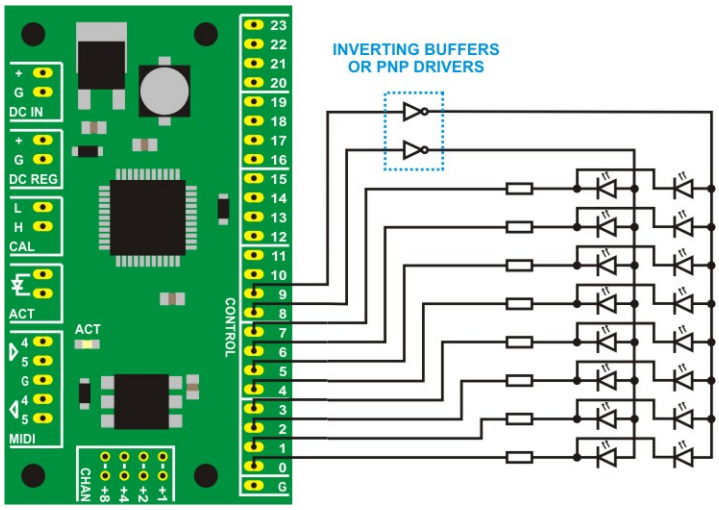


Figure 3.6.4-3: LED Matrix, Non-Inverted Common Cathode

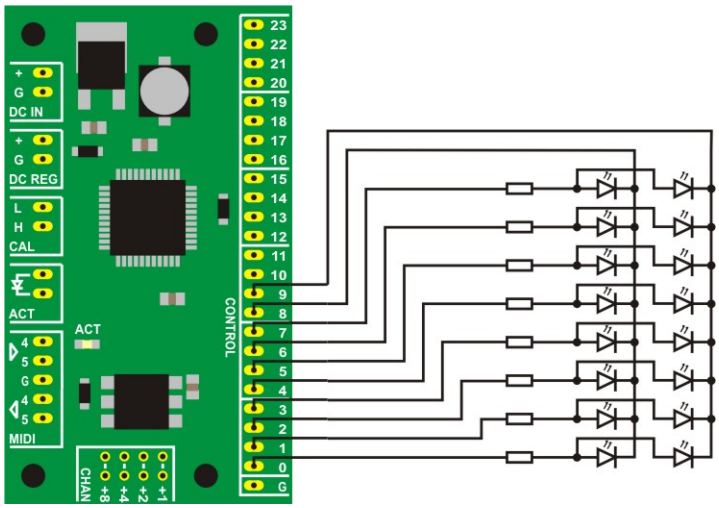
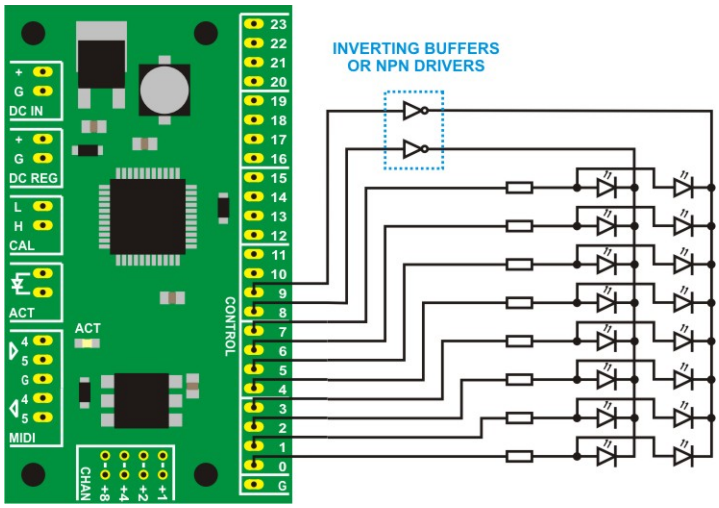


Figure 3.6.4-4: LED Matrix, Inverted Common Cathode



3.7 Factory Default Control Terminal Configuration

Figure 3.7-1: Factory Default Control Terminal Configuration: Layer 00h (Part 1 of 3)

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 01	Fixed Header
00	Configuration Layer 0
00 00 44 00 3C 7F	Control Terminal 0: Transition High to Low Logic Input Trigger: **Note On**: Note 60, Velocity 127
00 01 44 00 3C 00	Control Terminal 0: Transition Low to High Logic Input Trigger: **Note On**: Note 60, Velocity 0 (Velocity 0 = **Note Off**)
01 00 44 00 3D 7F	Control Terminal 1: Transition High to Low Logic Input Trigger: **Note On**: Note 61, Velocity 127
01 01 44 00 3D 00	Control Terminal 1: Transition Low to High Logic Input Trigger: **Note On**: Note 61, Velocity 0 (Velocity 0 = **Note Off**)
02 00 44 00 3E 7F	Control Terminal 2: Transition High to Low Logic Input Trigger: **Note On**: Note 62, Velocity 127
02 01 44 00 3E 00	Control Terminal 2: Transition Low to High Logic Input Trigger: **Note On**: Note 62, Velocity 0 (Velocity 0 = **Note Off**)
03 00 44 00 3F 7F	Control Terminal 3: Transition High to Low Logic Input Trigger: **Note On**: Note 63, Velocity 127
03 01 44 00 3F 00	Control Terminal 3: Transition Low to High Logic Input Trigger: **Note On**: Note 63, Velocity 0 (Velocity 0 = **Note Off**)
04 00 44 00 40 7F	Control Terminal 4: Transition High to Low Logic Input Trigger: **Note On**: Note 64, Velocity 127
04 01 44 00 40 00	Control Terminal 4: Transition Low to High Logic Input Trigger: **Note On**: Note 64, Velocity 0 (Velocity 0 = **Note Off**)
05 00 44 00 41 7F	Control Terminal 5: Transition High to Low Logic Input Trigger: **Note On**: Note 65, Velocity 127
05 01 44 00 41 00	Control Terminal 5: Transition Low to High Logic Input Trigger: **Note On**: Note 65, Velocity 0 (Velocity 0 = **Note Off**)
06 00 44 00 42 7F	Control Terminal 6: Transition High to Low Logic Input Trigger: **Note On**: Note 66, Velocity 127
06 01 44 00 42 00	Control Terminal 6: Transition Low to High Logic Input Trigger: **Note On**: Note 66, Velocity 0 (Velocity 0 = **Note Off**)
07 00 44 00 43 7F	Control Terminal 7: Transition High to Low Logic Input Trigger: **Note On**: Note 67, Velocity 127
07 01 44 00 43 00	Control Terminal 7: Transition Low to High Logic Input Trigger: **Note On**: Note 67, Velocity 0 (Velocity 0 = **Note Off**)

Figure 3.7-2: Factory Default Control Terminal Configuration: Layer 00h (Part 2 of 3)

SysEx Data (Hex)	Meaning
08 00 44 00 44 7F	Control Terminal 8: Transition High to Low Logic Input Trigger: **Note On**: Note 68, Velocity 127
08 01 44 00 44 00	Control Terminal 8: Transition Low to High Logic Input Trigger: **Note On**: Note 68, Velocity 0 (Velocity 0 = **Note Off**)
09 00 44 00 45 7F	Control Terminal 9: Transition High to Low Logic Input Trigger: **Note On**: Note 69, Velocity 127
09 01 44 00 45 00	Control Terminal 9: Transition Low to High Logic Input Trigger: **Note On**: Note 69, Velocity 0 (Velocity 0 = **Note Off**)
0A 00 44 00 46 7F	Control Terminal 10: Transition High to Low Logic Input Trigger: **Note On**: Note 70, Velocity 127
0A 01 44 00 46 00	Control Terminal 10: Transition Low to High Logic Input Trigger: **Note On**: Note 70, Velocity 0 (Velocity 0 = **Note Off**)
0B 00 44 00 47 7F	Control Terminal 11: Transition High to Low Logic Input Trigger: **Note On**: Note 71, Velocity 127
0B 01 44 00 47 00	Control Terminal 11: Transition Low to High Logic Input Trigger: **Note On**: Note 71, Velocity 0 (Velocity 0 = **Note Off**)
0C 00 44 00 48 7F	Control Terminal 12: Transition High to Low Logic Input Trigger: **Note On**: Note 72, Velocity 127
0C 01 44 00 48 00	Control Terminal 12: Transition Low to High Logic Input Trigger: **Note On**: Note 72, Velocity 0 (Velocity 0 = **Note Off**)
0D 00 44 00 49 7F	Control Terminal 13: Transition High to Low Logic Input Trigger: **Note On**: Note 73, Velocity 127
0D 01 44 00 49 00	Control Terminal 13: Transition Low to High Logic Input Trigger: **Note On**: Note 73, Velocity 0 (Velocity 0 = **Note Off**)
0E 00 44 00 4A 7F	Control Terminal 14: Transition High to Low Logic Input Trigger: **Note On**: Note 74, Velocity 127
0E 01 44 00 4A 00	Control Terminal 14: Transition Low to High Logic Input Trigger: **Note On**: Note 74, Velocity 0 (Velocity 0 = **Note Off**)
0F 00 44 00 4B 7F	Control Terminal 15: Transition High to Low Logic Input Trigger: **Note On**: Note 75, Velocity 127
0F 01 44 00 4B 00	Control Terminal 15: Transition Low to High Logic Input Trigger: **Note On**: Note 75, Velocity 0 (Velocity 0 = **Note Off**)

Figure 3.7-3: Factory Default Control Terminal Configuration: Layer 00h (Part 3 of 3)

SysEx Data (Hex)	Meaning
10 00 44 00 4C 7F	Control Terminal 16: Transition High to Low Logic Input Trigger: **Note On**: Note 76, Velocity 127
10 01 44 00 4C 00	Control Terminal 16: Transition Low to High Logic Input Trigger: **Note On**: Note 76, Velocity 0 (Velocity 0 = **Note Off**)
11 00 44 00 4D 7F	Control Terminal 17: Transition High to Low Logic Input Trigger: **Note On**: Note 77, Velocity 127
11 01 44 00 4D 00	Control Terminal 17: Transition Low to High Logic Input Trigger: **Note On**: Note 77, Velocity 0 (Velocity 0 = **Note Off**)
12 00 44 00 4E 7F	Control Terminal 18: Transition High to Low Logic Input Trigger: **Note On**: Note 78, Velocity 127
12 01 44 00 4E 00	Control Terminal 18: Transition Low to High Logic Input Trigger: **Note On**: Note 78, Velocity 0 (Velocity 0 = **Note Off**)
13 00 44 00 4F 7F	Control Terminal 19: Transition High to Low Logic Input Trigger: **Note On**: Note 79, Velocity 127
13 01 44 00 4F 00	Control Terminal 19: Transition Low to High Logic Input Trigger: **Note On**: Note 79, Velocity 0 (Velocity 0 = **Note Off**)
14 00 44 00 50 7F	Control Terminal 20: Transition High to Low Logic Input Trigger: **Note On**: Note 80, Velocity 127
14 01 44 00 50 00	Control Terminal 20: Transition Low to High Logic Input Trigger: **Note On**: Note 80, Velocity 0 (Velocity 0 = **Note Off**)
15 00 44 00 51 7F	Control Terminal 21: Transition High to Low Logic Input Trigger: **Note On**: Note 81, Velocity 127
15 01 44 00 51 00	Control Terminal 21: Transition Low to High Logic Input Trigger: **Note On**: Note 81, Velocity 0 (Velocity 0 = **Note Off**)
16 00 44 00 52 7F	Control Terminal 22: Transition High to Low Logic Input Trigger: **Note On**: Note 82, Velocity 127
16 01 44 00 52 00	Control Terminal 22: Transition Low to High Logic Input Trigger: **Note On**: Note 82, Velocity 0 (Velocity 0 = **Note Off**)
17 00 44 00 53 7F	Control Terminal 23: Transition High to Low Logic Input Trigger: **Note On**: Note 83, Velocity 127
17 01 44 00 53 00	Control Terminal 23: Transition Low to High Logic Input Trigger: **Note On**: Note 83, Velocity 0 (Velocity 0 = **Note Off**)
F7	Fixed Footer

Figure 3.7-4: Factory Default Control Terminal Configuration: Layers 01h, 02h, 03h

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 01	Fixed Header
[01, 02, 03]	Configuration Layers 01h, 02h, 03h
[00..17] 00 7F 00 00 00	For each control terminal 0 thru 23: Transition High to Low Logic Input Data Only
[00..17] 01 7F 00 00 00	For each control terminal 0 thru 23: Transition Low to High Logic Input Data Only
F7	Fixed Footer

4.0 Global Configuration

The MIDI CPU has several user-configurable settings which are not associated with individual control terminals. Each setting, associated SysEx message, and "factory default" value is described below.

The MIDI CPU acknowledges an incoming configuration message via the Activity Indicator LED.

4.1 Matrix Note Velocity

The velocity of MIDI notes generated by control terminal modes Matrix Select: Note On / Note Off and Matrix Select: Note Toggle is determined by the value of Multipurpose Data register 1Eh. To change the power-up value of this register, use the Data Register Configuration SysEx message.

4.2 Control Change (CC) “On/Off” Values

The CC value of messages generated by control terminal modes Logic Input Trigger: Control Change (CC) Toggle, Matrix Select: Control Change (CC) On/Off, and Matrix Select: Control Change (CC) Toggle is determined by the value of Multipurpose Data registers 1Fh and 20h. To change the power-up value of these registers, use the Data Register Configuration SysEx message.

4.3 Data Register Configuration

Analog Input Data registers, Multipurpose Data registers, Configuration Layer Control Register, and Indirect Pointer register have several user-configurable characteristics that can be changed via the MIDI SysEx message described in **Figure 4.3-1**.

Value ***nn*** corresponds to data register address. The valid range for *nn* is 03h-29h.

Value ***ii*** sets the initial value for the register at power-up. This value is ignored for analog input data registers.

Values ***mn*** and ***mx*** determine the minimum and maximum values for the register. For a given register, minimum and maximum values are only enforced when:

- An analog signal is converted to a digital value and stored in the register
- Encoder input increments or decrements the value register value
- Increment, decrement, or sum operations modify the register value

Value ***rr*** specifies the "round robin" behavior of the register. When *rr* = 00h, any attempted increase (decrease) to the register past the value *mx* (*mn*) will fail. When *rr* = 01h, increases (decreases) past the value *mx* (*mn*) cause the value to "wrap" thru the minimum (maximum) values. This value is ignored for analog input data registers.

Figure 4.3-1: Data Register Configuration SysEx Message (Hex)

Header (6 bytes)	Register Configuration (Repeat as Desired)	Footer (1 byte)
F0 00 01 5D 04 04	*nn ii mn mx rr*	F7

Figure 4.3-2: Factory Default Data Register Configuration

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 04	Header
[03..10] 00 00 7F 00	Analog Input Data register 03h thru 10h: *mn* = 0, *mx* = 127, *rr* = 0.
[11..1C] 00 00 7F 00	Multipurpose Data register 11h thru 1Ch: *ii* = 0, *mn* = 0, *mx* = 127, *rr* = 0.
1D 0F 00 0F 00	Configuration Layer Control Register 1Dh: *ii* = 0Fh.
1E 7F 00 7F 00	Multipurpose Data register 1Eh (matrix note velocity): *ii* = 127, *mn* = 0, *mx* = 127, *rr* = 0.
1F 7F 00 7F 00	Multipurpose Data register 1Fh (CC "on" value): *ii* = 127, *mn* = 0, *mx* = 127, *rr* = 0.
20 00 00 7F 00	Multipurpose Data register 20h (CC "off" value): *ii* = 0, *mn* = 0, *mx* = 127, *rr* = 0.
[21..28] 00 00 7F 00	Multipurpose Data register 21h thru 28h: *ii* = 0, *mn* = 0, *mx* = 127, *rr* = 0.
29 00 00 28 00	Indirect Pointer register: *ii* = 0, *mn* = 0, *mx* = 28h, *rr* = 0.
F7	Footer

4.4 Note Transposition

The MIDI CPU can transpose locally generated notes by an amount that is adjustable in "real time". Note transposition is configured via the SysEx message described in **Figure 4.4-1**.

If the register at address *aa* contains the value *vv,* note numbers are increased by the value (*vv* - 40h). This allows a range of transposition values between -64 (*vv* = 00h) and +63 (*vv* = 7Fh). When *vv* = 40h, no transposition takes place.

To disable note transposition, use the value *aa* = 7Fh.

The MIDI CPU transposes note numbers before performing any MIDI Number Remapping.

Figure 4.4-1: Note Transposition SysEx Message (Hex)

Header (6 bytes)	Transposition Register Address (1 byte)	Footer (1 byte)
F0 00 01 5D 04 0B	*aa*	F7

Figure 4.4-2: Factory Default Note Transposition

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 0B 7F F7	Note transposition is deactivated.

4.5 MIDI Number Remapping

The MIDI CPU can remap any locally generated note number, CC number, or program number to another before MIDI output is generated. This can be useful for unusual switch matrix configurations or for changing the musical scale of MIDI output.

Remapping for each message type is independently activated via SysEx message. See **Figure 4.5-1** and **Table 4.5-a**.

Figure 4.5-1: MIDI Number Remapping Flags SysEx Message (Hex)

Header (6 bytes)	Remap Flags (1 byte)	Footer (1 byte)
F0 00 01 5D 04 0A	*ff*	F7

Table 4.5-a: MIDI Number Remapping Flags Hex Values

ff (Hex)	Program Remap	CC Remap	Note Remap
00	off	off	off
01	off	off	**on**
02	off	**on**	off
03	off	**on**	**on**
04	**on**	off	off
05	**on**	off	**on**
06	**on**	**on**	off
07	**on**	**on**	**on**

When the MIDI CPU generates a message type for which remapping is active, the (note, CC, or program) number resulting from control terminal configuration is used as a look-up address for the *MIDI Number Remap Table*. The value contained in the table at that address is substituted for the (note, CC, or program) number in the outgoing message.

By default, each value within the table is equal to its address: therefore, no remapping takes place. Values in the table may be updated via SysEx as described in **Figure 4.5-2**. The SysEx message includes the remap value for each MIDI number in order from numbers 0 to 127.

To return the MIDI Number Remap Table to its default (no remap) state, use the SysEx message described in **Figure 4.5-3**.

Figure 4.5-2: MIDI Number Remapping Table SysEx Message (Hex)

Header (6 bytes)	MIDI Number Remap Table Contents (128 bytes)	Footer (1 byte)
F0 00 01 5D 04 05	*n0 n1 n2 ...*	F7

Figure 4.5-3: MIDI Number Remapping Table Reset SysEx Message (Hex)

Complete Message (7 bytes)
F0 00 01 5D 04 06 F7

Figure 4.5-4: Factory Default MIDI Number Remapping Flags

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 0A 00 F7	Remapping is deactivated for all message types.

Figure 4.5-5: Factory Default MIDI Number Mapping

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 05 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 6E 7F F7	Each number is mapped to itself. Effectively, no remapping takes place.

4.6 SysEx Data Buffer

The MIDI CPU is capable of sending arbitrary **SysEx** messages in response to control terminal input. See the Logic Input Trigger: System Exclusive (SysEx) control terminal mode.

The data payload of the generated **SysEx** message is formed from the contents of the *SysEx Data Buffer*. To update the contents of the buffer, send a SysEx message as shown in **Figure 4.6-1**. The SysEx data is written to the SysEx Data Buffer starting at address 00h. If fewer than 128 bytes are specified, the remainder of bytes in the SysEx Data Buffer is filled with the value 7Fh.

Figure 4.6-1: SysEx Data Buffer Contents SysEx Message (Hex)

Header (6 bytes)	SysEx Data Buffer Contents (up to 128 bytes)	Footer (1 byte)
F0 00 01 5D 04 07	*d0 d1 d2 ...*	F7

Figure 4.6-2: Factory Default SysEx Data Buffer Contents

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 07 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D 5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D 6E 7F F7	The SysEx data buffer contains arbitrary data.

4.7 ADC Configuration

The MIDI CPU uses an analog-to-digital converter (ADC) to measure the voltages on control terminals configured as analog inputs. Measurements from the ADC are processed based on settings from the SysEx message shown in **Figure 4.7-1**.

Figure 4.7-1: ADC Configuration SysEx Message (Hex)

Header (6 bytes)	Analog Input Threshold (1 byte)	Analog Input Smoothing (1 byte)	Reference Voltage (1 byte)	Footer (1 byte)
F0 00 01 5D 04 08	*tt*	*ss*	*rr*	F7

4.7.1 Analog Input Threshold (*tt*)

When a control terminal is configured as an analog input, the MIDI CPU first converts the analog signal to a 10-bit digital value, and then reduces the 10-bit value to the 7-bit resolution employed by MIDI messaging.

Due to "noise" resulting from the analog-to-digital conversion process and/or the instability of an analog input signal, analog input values may fluctuate excessively, causing the MIDI CPU to generate unwanted quantities of MIDI messages. The *Analog Input Threshold* is the minimum quantity by which any 10-bit analog input value must change in order to trigger message generation.

The Analog Input Threshold acts on the 10-bit input value, but is ignored whenever the resulting 7-bit input value reaches its minimum (0) or maximum (127) value.

4.7.2 Analog Input Smoothing (*ss*)

The MIDI CPU uses a rolling average algorithm to smooth the data waveform resulting from analog to digital conversion. The value of *ss* determines the weight of the rolling average compared to the immediate analog input value. Larger values of *ss* result in a smoother waveform. Smaller values of *ss* result in a waveform that responds more quickly to input changes. The valid range for *ss* is 00h to 04h.

4.7.3 Analog Reference Voltage (*rr*)

When one or more MIDI CPU control terminals are configured as analog inputs, the MIDI CPU uses two *reference voltages* during the analog-to-digital conversion process. These voltages, called "V_{REF-}" and "V_{REF+}", represent the analog voltages that are converted to the digital values 00h and 7Fh, respectively.

The MIDI CPU can use its own voltage regulator to supply the reference voltages, or it can use external reference voltages connected by the user.

External reference voltages can be useful when:

- Lower-noise analog-to-digital conversion is required
- The analog input signal does not span the entire range between MIDI CPU ground and V_{REG}.

When *rr* is set to 00h, the MIDI CPU will use its on-board ground as V_{REF-}, and its on-board V_{REG} as V_{REF+}.

When *rr* is set to 01h, the MIDI CPU will use the external V_{REF-} and V_{REF+} connected by the user to control terminals 18 and 19.

See ***MIDI CPU Hardware User Manual*** for external reference voltage wiring diagrams and for additional details regarding V_{REG}.

Regardless of the reference voltages used, analog input voltages must not exceed the limits described in ***MIDI CPU Hardware User Manual***.

Figure 4.7.3-1: Factory Default ADC Configuration

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 08 06 04 00 F7	Analog Input Threshold = 6; Analog Input Smoothing = 4; Reference Voltage = On-Board.

4.8 Active Sense On/Off

During periods of inactivity, a MIDI device can send **Active Sense** messages to notify the receiving device that the MIDI link is still operating. **Active Sense** messages are automatically generated at the rate of approximately 4 per second when no other MIDI data is being generated. This function is not required for most applications.

MIDI CPU **Active Sense** generation can be enabled or disabled using the SysEx message shown in **Figure 4.8-1**. When *aa* = 00h, **Active Sense** is disabled. When *aa* = 01h, **Active Sense** is enabled.

Figure 4.8-1: Active Sense On/Off SysEx Message (Hex)

Header (6 bytes)	Active Sense (1 byte)	Footer (1 byte)
F0 00 01 5D 04 09	*aa*	F7

Figure 4.8-2: Factory Default Active Sense

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 09 00 F7	Generation of **Active Sense** messages is disabled.

4.9 Control Terminals 8-15 Pull-Up Resistor

All MIDI CPU control terminals feature on-board pull-up resistors. The pull-up resistors for control terminals 8-15 can be disabled.

The pull-up resistors can be enabled or disabled using the SysEx message shown in **Figure 4.9-1**. Bits 3-0 of *ph* and *pl* correspond to control terminals 8-15 as shown in **Table 4.9-a**. When a bit is set to 1, the pull-up on the corresponding control terminal is disabled. Bits 6-4 of *ph* and *pl* are ignored.

Figure 4.9-1: Control Terminals 8-15 Pull-Up Resistor SysEx Message (Hex)

Header (6 bytes)	Pull Up States (2 bytes)	Footer (1 byte)
F0 00 01 5D 04 0C	*ph pl*	F7

Table 4.9-a: Pull-Up Resistor Control Bits

Byte	Control Terminal Pull-Up by Bit						
	6	**5**	**4**	**3**	**2**	**1**	**0**
ph	-	-	-	15	14	13	12
pl	-	-	-	11	10	9	8

Figure 4.9-2: Factory Default Pull-Up States

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 0C 00 00 F7	All pull-up resistors are enabled for control terminals 8-15.

4.10 MIDI Messaging Throttle

Normally, the rate of MIDI CPU local MIDI message generation is limited only by the data rate of the MIDI hardware interface.

If desired, the rate of local message generation can be reduced using the SysEx message shown in **Figure 4.10-1**. Larger values of *mt* result in a slower messaging rate. When *mt* = 7Fh, messages are generated at a rate of roughly two per second. When *mt* = 00h, the messaging rate is not restricted beyond the limitations of the hardware interface.

This setting does not affect the rate of output for messages sent as part of the MIDI CPU MIDI Merge function.

Figure 4.10-1: MIDI Messaging Throttle SysEx Message (Hex)

Header (6 bytes)	Pull Up States (2 bytes)	Footer (1 byte)
F0 00 01 5D 04 0D	*mt*	F7

Figure 4.10-2: Factory Default MIDI Messaging Throttle

SysEx Data (Hex)	Meaning
F0 00 01 5D 04 0D 00 F7	MIDI messaging rate limited only by MIDI hardware interface data rate.

5.0 Configuration Retrieval

At any time, the current MIDI CPU configuration can be retrieved via *SysEx Configuration Retrieval Messages*. When the MIDI CPU receives a SysEx Configuration Retrieval Message at its MIDI IN port, it will output the associated configuration data in the form of a SysEx Configuration Message.

The format of each SysEx Configuration Retrieval message is shown in **Table 5-a**.

Table 5-a: SysEx Configuration Retrieval Messages

SysEx Configuration Retrieval Message (Hex)	Resulting Output
F0 00 01 5D 04 00 01 00 F7	Control terminal configurations, all terminals, layer 00h. Formatted as Control Terminal Configuration SysEx Message.
F0 00 01 5D 04 00 01 01 F7	Control terminal configurations, all terminals, layer 01h. Formatted as Control Terminal Configuration SysEx Message.
F0 00 01 5D 04 00 01 02 F7	Control terminal configurations, all terminals, layer 02h. Formatted as Control Terminal Configuration SysEx Message.
F0 00 01 5D 04 00 01 03 F7	Control terminal configurations, all terminals, layer 03h. Formatted as Control Terminal Configuration SysEx Message.
F0 00 01 5D 04 00 04 00 F7	Data Register Configuration SysEx Message
F0 00 01 5D 04 00 05 00 F7	MIDI Number Remapping Table SysEx Message
F0 00 01 5D 04 00 07 00 F7	SysEx Data Buffer Contents SysEx Message
F0 00 01 5D 04 00 08 00 F7	ADC Configuration SysEx Message
F0 00 01 5D 04 00 09 00 F7	Active Sense On/Off SysEx Message
F0 00 01 5D 04 00 0A 00 F7	MIDI Number Remapping Flags SysEx Message
F0 00 01 5D 04 00 0B 00 F7	Note Transposition SysEx Message
F0 00 01 5D 04 00 0C 00 F7	Control Terminals 8-15 Pull-Up Resistor SysEx Message
F0 00 01 5D 04 00 0D 00 F7	MIDI Messaging Throttle SysEx Message
F0 00 01 5D 04 00 7D 00 F7	Firmware Version SysEx Message
F0 00 01 5D 04 00 7F 00 F7	Configuration dump: all of the configuration data listed above, formatted as a series of SysEx messages.

6.0 MIDI Channel Selection Jumper

Depending on Control Terminal Configuration, the *MIDI Channel Selection Jumper* can be used to select the MIDI channel for messages sent or received by the MIDI CPU. See ***MIDI CPU Hardware User Manual*** for diagrams.

When none of the channel jumpers are shorted, the channel setting is 1. Shorting the "+1" jumper adds 1 to the channel setting, the "+2" adds 2 to the channel setting, etc.

For example, if both the "+2" and "+4" jumpers are shorted, the resulting channel setting is 1 + 2 + 4 = 7.

7.0 Activity Indicator LED

The MIDI CPU Activity Indicator LED performs several functions:

- **Self Test:** Upon power-up or device reset, the Activity LED lights briefly before normal operation begins.
- **Configuration Update Indication:** When the MIDI CPU configuration is updated via a MIDI SysEx message, the device will reset, followed by the LED self-test. If no LED activity is observed, then the attempted configuration was unsuccessful.
- **MIDI Activity Indication:** The LED blinks when locally generated or merged MIDI events are sent via the MIDI CPU MIDI Out port.
- **Firmware Update Mode Indication:** The LED will remain lighted while the MIDI CPU awaits a firmware update SysEx message.
- **Firmware Update Error:** The LED blinks continuously when the receipt of a firmware update SysEx message has failed, or when the firmware has become corrupt.

8.0 Firmware Update

MIDI CPU firmware can be upgraded via MIDI SysEx message. Firmware update files, when available, can be downloaded from: http://highlyliquid.com/midi-cpu

8.1 Firmware Update Procedure

1. Download and unzip the firmware update SysEx message. The resulting file should have the extension ".syx".
2. Disconnect power to the MIDI CPU.
3. Connect the "CAL L" terminal to ground.
4. Re-connect power to the MIDI CPU. The Activity LED will remain lighted.
5. Send the firmware update SysEx message to the MIDI CPU. Transmission of the message may require several seconds.
6. Upon completion of the update, the Activity LED will deactivate.
7. Disconnect the "CAL L" terminal.
8. Disconnect power to the MIDI CPU.
9. If updating from version 1.1 or earlier, power up the MIDI CPU, and send the following SysEx message: F0 00 01 5D 04 0B 7F F7
10. If updating from version 1.3 or earlier, power up the MIDI CPU, and send the following SysEx message: F0 00 01 5D 04 04 03 00 00 7F 00 04 00 00 7F 00 05 00 00 7F 00 06 00 00 7F 00 07 00 00 7F 00 08 00 00 7F 00 09 00 00 7F 00 0A 00 00 7F 00 0B 00 00 7F 00 0C 00 00 7F 00 0D 00 00 7F 00 0E 00 00 7F 00 0F 00 00 7F 00 10 00 00 7F 00 11 00 00 7F 00 12 00 00 7F 00 13 00 00 7F 00 14 00 00 7F 00 15 00 00 7F 00 16 00 00 7F 00 17 00 00 7F 00 18 00 00 7F 00 19 00 00 7F 00 1A 00 00 7F 00 1B 00 00 7F 00 1C 00 00 7F 00 1D 0F 00 0F 00 1E 7F 00 7F 00 1F 7F 00 7F 00 20 00 00 7F 00 21 00 00 7F 00 22 00 00 7F 00 23 00 00 7F 00 24 00 00 7F 00 25 00 00 7F 00 26 00 00 7F 00 27 00 00 7F 00 28 00 00 7F 00 29 00 00 28 00 F7

8.2 Notes

- When updating firmware, use a MIDI CPU power supply voltage of 6.0V or greater.
- Do not interrupt the transmission of the firmware update SysEx message.
- A continuously blinking Activity LED indicates an error in the transmission of the firmware update SysEx message.
- If a continuously blinking Activity LED is observed, repeat the procedure from the beginning.

8.3 Firmware Version SysEx Message

During normal operation, the current MIDI CPU firmware version can be retrieved by sending the SysEx message shown in **Figure 8.3-1**. In response, the MIDI CPU will send a SysEx message as shown in **Figure 8.3-2**. The key to firmware versions is shown in **Table 8.3-a**.

Figure 8.3-1: Firmware Version Retrieval SysEx Message (Hex)

Complete Message (9 bytes)
F0 00 01 5D 04 00 7D 00 F7

Figure 8.3-2: Firmware Version SysEx Message (Hex)

Header (6 bytes)	Version (1 byte)	Footer (1 byte)
F0 00 01 5D 04 7D	*vv*	F7

Table 8.3-a: Firmware Versions

vv (Hex)	Firmware Version
0	1.0
01	1.1
02	1.2
03	1.3 (beta releases)
04	1.2a
05	1.3 (production version)
06	1.4 (beta releases)
07	1.4 (production version)
08 thru 7F	Reserved for future versions.

9.0 Configuration Examples and Technical Support

Numerous MIDI CPU project examples can be found at the MIDI CPU support forum. Please visit the support forum to discuss your project:

http://forum.highlyliquid.com/