M5000

DIGITAL AUDIO MAINFRAME

MIDI System-Exclusive Documentation

Revision 2.00

(16/06/97)

Documents all features in application-software APL115.M5K

OF DENMARK

t.c. electronic

1 Introduction

1.1 Overview

The M5000 sends and receives system-exclusive messages (sysex). The sysex-protocol gives you access to the following:

- Each individual parameter in each algorithm
- System-parameters (audio-routing, formats, sample-rates, meters etc.)
- System-information (software version, installed options)
- Preset-handling (preset-transfers)
- Preset-selection (to recall presets without needing to know about MIDI-channels)

1.1.1 The Parameter-Access Dump

The Parameters-Access dump is used for almost all communication to and from the M5000, and is therefore the most extensive part of the protocol. Parameters that are specific to individual algorithms are described in chapter 5. Parameters that are general to the system (DSP-cards) are described in chapter 5.

There are many different parameter-types in the M5000, including milliseconds, hertz, decibels, tables and character-strings. It would be nearly impossible to describe and list each parameter-type in detail, which is why TC supplies a C programming-interface to assist you in displaying the correct value for each individual parameter-type (eg. “10 kHz” or “50%”). The programming-interface consists of two files, “CLASS.C” and “CLASS.H”. The two files can be downloaded from the TC User-Club BBS on the following phone-numbers:

Denmark: +45 - 86 21 75 99
USA: +1 - 805 373 1828

Please refer to chapter 0 for information on how to use the interface.

1.1.2 System-Configuration/Info Dumps

Information about the system (software version/installed options/DSP-cards etc.) can be requested from the M5000. This is described in chapter 0.

1.1.3 Preset-Handling Dumps

These dumps provide you the means to store, recall, dump and retrieve presets as well as request information about them. This is described in chapter 0.

1.2 General Format

Sysex-packets are transferred to and from the M5000 using the following general format:

```
Sysex-start   $f0
TC ID         $33
Device#       $00-$7f
Card#         $01-$04
Packet-type   $00-$07

Data specific to the Packet-type

Sysex-end     $f7
```

The Device# must correspond with the Device# set for the M5000 frame.

The Card# refers to the ID of each consecutive DSP-card or layer. A value of 1 refers to the first DSP-card and a value of 2 refers to the second DSP-card. A value of 0 is only used in certain operations that refer to the entire M5000 frame.

The Packet-type signifies the type of packet. Each individual request and dump has it's own unique packet-type. The following packet-types are transmitted and/or recognized by the M5000:

$00	Set parameter(s)
$01	Request parameter(s)
$02	Recall Preset
$03	Request Preset Info
$04	Request System Configuration Info
$05	Preset Info

The M5000 is very tolerant about incomplete or erroneous sysex-packets, but it is still recommended that you keep your packets clean with all values within range. The M5000 allows packet-sizes of any size (which is relevant for requesting a large number of parameters), though you need to obey the precautions regarding the parameter-queue as described in the next chapter.

2 Individual Parameter-Access

2.1 Overview

Each individual parameter in the M5000 has a unique ID. This gives you access to each parameter in each algorithm and general system-parameters, such as Input-gain or Bypass, for instance.

Not all parameters exist at the same time. For example, parameters in the REVERB-3 algorithm don't exist if a PITCH-1 algorithm is running on the DSP-card in question. Trying to set nonexistent parameters will have no effect, and requesting their setting will produce no result.

Note: It is possible to obtain information about the algorithm currently running, to determine which parameters should be polled. Please refer to Chapter 0 for a description of how to obtain information about the algorithm currently running on a DSP-card.

2.2 Parameter-IDs and Values

All parameter-IDs and values in the M5000 are 14 bit wide. In addition, parameter-values are signed, to allow for negative values. The two ranges are as follows:

Parameter-IDs:	0 to 16383 ($0000 to $3fff)
Parameter-values:	-8192 to 8191 (-$2000 to $1fff)

2.2.1 Parameter-IDs

In the following documentation, the 14-bit parameter-IDs are shown as <Par# xxxx>, although their physical placement in the sysex-packet is as follows:

```
Par #xx__     bit 8-13 (MSB First)
Par #__xx     bit 0-7 (LSB Last)
```

...is shown as:

```
<Par #xxxx>
```

In sections 0 and 0 you'll find two C-routines that convert the two MIDI-bytes to a single C-type unsigned short and vice versa.

2.2.2 Parameter-values

In the following documentation, the 14-bit signed parameter-values are shown as <Value #xxxx>, although their physical placement in the sysex-packet is as follows:

```
Value #xx__   bit 8-13 (MSB First, sign in bit 13)
Value #__xx   bit 0-7 (LSB Last)
```

...is shown as:

```
<Value #xxxx>
```

In order to convert these double MIDI-bytes to a single C-type short, the sign bit must be extended from bit 13 to bit 15. In sections 0 and 0 you'll find two C-routines that convert the two MIDI-bytes to a single C-type short and vice versa.

2.3 Requesting Parameter Values

The following sysex-packet allows you to request the setting of a number of parameters. In a single packet, you can request as many or as few parameters as you like.

```
Sysex-start   $f0
TC ID            $33
Device#       xx
Card#            xx
Packet-type   $01 - Request
<Par #xxxx>
<Par #yyyy>
<Par #zzzz>
...
...
Sysex-end     $f7
```

The M5000 replies with a Parameter Dump, as described next:

2.4 Setting Parameter Values

In a single parameter-dump, you can set as many or as few parameters as you like. In order to minimize MIDI-traffic, you should set as many parameters as possible in a single dump.

```
Sysex-start   $f0
TC ID            $33
Device#       xx
Card#            xx
Packet-type   $00 - Dump
<Par #xxxx>
<Par Value>
<Par #yyyy>
<Par Value>
<Par #zzzz>
<Par Value>
...
...
Sysex-end     $f7
```

2.4.1 Truncation and Mutual Dependencies

If a parameter-value is out of range, it is truncated to fit. Please note, that some parameters (such as cross-overs) have floating minimum and maximum values. This scheme follows a fairly simple logic, although you must implement this yourself in order to track the correct value for the user; the M5000 has no way of telling you that a parameter-value has been truncated. These mutual dependencies are described as necessary in conjunction with the parameter-listings in chapter 0 and 0.

2.4.2 Linked Parameters

Some parameters are linked to always contain the same value (a good example of this is the 0dBRef parameter in the DYNAMIC1 algorithm). Generally, you shouldn’t display or manipulate more than one of the linked parameters. With the 0dBRef example, simply choose one of the parameters as the only one.

2.4.3 The Parameter-Queue

The M5000 places all parameters that need to be changed in a queue. Some parameters take a little time to recalculate, while others change instantaneously. The parameters are extracted from the queue as fast as possible.

Note: If you set a parameter that hasn't yet been extracted from the queue, the queue-entry for the given parameter is updated to hold the new value. This means that you don't need to worry about placing delays in the MIDI data-stream while the user is dragging a slider in a patch-editor. Simply transmit the new value for the given parameter every time the slider is moved.

The parameter-queue in the M5000 holds 32 messages. If you are setting up an algorithm like TAPFAC-1 or DYNAMIC1 (which have more than 32 parameters), you must place slight delays in the MIDI data-stream. Some special parameters take some time to recalculate (you will know these parameters from the M5000 front panel). Instead of having a specific delay for each parameter-type, you should simply transmit the parameters at a pace that works.

2.5 Spontaneous Data-Emissions

The M5000 will generally never output parameter-packets spontaneously. However Recall Preset packet will be transmitted if the user recalls a preset via the front-panel or an ATAC.

Meters are never transmitted spontaneously, but must be polled.

2.6 Conversion Routines

2.6.1 convertMIDItoPar

The following routine combines the two MIDI-bytes that identify the parameter-ID and return a single unsigned short:

```
unsigned short convertMIDItoPar(char byte1, char byte2)
{
   return byte2+(byte1 << 7);
}
```

2.6.2 convertPartoMIDI

The following routine derives the two MIDI-bytes that identify the parameter-ID from a single short:

```
void convertMIDItoPar(unsigned short parNo, char *byte1,
                                                 char *byte2)
{
   *byte1=parNo >> 7;
   *byte2=parNo & 0x7f;
}
```

2.6.3 convertMIDItoValue

The following routine combines the two MIDI-bytes that identify the parameter-value and return a single short:

```
short convertMIDItoValue(char byte1, char byte2)
{
   short   value;

   value=(byte2 & 0x7f) + ( (short) (byte1 & 0x7f) << 7);
   if (value & 0x2000) i |= 0xc000; // Extend sign bit
   return value;
}
```

2.6.4 convertValuetoMIDI

The following routine derives the two MIDI-bytes that identify the parameter-ID from a single short:

```
void convertValuetoMIDI(short value, char *byte1, char *byte2)
{
   *byte1=(value >> 7) & 0x7f;
   *byte2=value & 0x7f;
}
```

3 Algorithm-Parameters

3.1 REVERB-1 & REVERB-2

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1000	0	100	CLS_PERCENT
INLEV	1001	0	100	CLS_DB1
OUTLEV	1002	0	100	CLS_DB1
DECAY	1003	3	600	CLS_SEC1
x LOW	1004	1	250	CLS_NON2
x HIGH	1005	1	200	CLS_NON2
DIFFUSE	1006	1	25	CLS_NON0
SHAPE	1007	0	5	CLS_SHAPE0
x SIZE	1008	0	20	CLS_SIZE0
PREDLY	1009	0	2000	CLS_MS1
REVFEED	100A	0	1000	CLS_MS1
HICUT	100B	14	30	CLS_FRQ0
ATT	100C	20	100	CLS_DB1
LO-XOVR	100D	0	30	CLS_FRQ0
HI-XOVR	100E	0	30	CLS_FRQ0
INITLEV	100F	0	100	CLS_DB1
REVLEV	1010	0	100	CLS_DB1
RWIDTH	1011	0	100	CLS_PERCENT
I-XFEED	1012	0	1	CLS_OFFON

These last 4 parameters are only available in REVERB-2:

REVDIFF	1013	0	100	CLS_PERCENT
BUILDUP	1014	0	100	CLS_NON0
IATTACK	1015	0	100	CLS_DB1
IDECAY	1016	0	100	CLS_DB1

3.2 REVERB-3

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1500	0	100	CLS_PERCENT
INLEV	1501	0	100	CLS_DB1
OUTLEV	1502	0	100	CLS_DB1
DECAY	1503	3	300	CLS_SEC1
x LOW	1504	1	250	CLS_NON2
x LOMID	1505	1	200	CLS_NON2
x HIGH	1506	1	200	CLS_NON2
DIFFUSE	1507	1	99	CLS_NON0
LO-XOVR	1508	0	23	CLS_FRQ0

LM-XOVR	1509	10	25	CLS_FRQ0
HI-XOVR	150A	20	30	CLS_FRQ0
PREDLY	150B	1	200	CLS_MS1
DISTANS	150C	0	15	CLS_NON0
HICUT	150D	14	30	CLS_FRQ0
ATT	150E	20	100	CLS_DB1
MODRATE	150F	1	200	CLS_NON0
MODDPTH	1510	0	100	CLS_PERCENT
DIFTYPE	1511	0	4	CLS_DIFF0

3.3 NONLIN-1

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1600	0	100	CLS_PERCENT
INLEV	1601	0	100	CLS_DB1
OUTLEV	1602	0	100	CLS_DB1
PREDLY	1603	0	500	CLS_MS0
ATTACK	1604	0	500	CLS_MS0
HOLD	1605	10	500	CLS_MS0
RELEASE	1606	0	500	CLS_MS0
LOCUT	1607	0	20	CLS_FRQ0
HICUT	1608	16	30	CLS_FRQ0
DIFFUSE	1609	0	25	CLS_NON0
PREDIFF	160A	0	100	CLS_NON0
DIFTYPE	160B	0	3	CLS_PREDIFF
WIDTH	160C	0	100	CLS_PERCENT

3.4 CHORUS-1

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1100	0	100	CLS_PERCENT
INLEV	1101	0	100	CLS_DB1
OUTLEV	1102	0	100	CLS_DB1
PHASE	1103	0	2	CLS_PHASE1
DELAY	1104	1	670	CLS_MS0
FB	1105	0	99	CLS_PERCENT
SPEED	1106	0	40	CLS_SPEEDS0
DEPTH	1107	0	100	CLS_PERCENT
FBLOCUT	1108	0	4	CLS_LOCUTS
FBHICUT	1109	0	4	CLS_HICUTS
HICUT	110A	14	30	CLS_FRQ0
ATT	110B	20	100	CLS_DB1

3.5 DELAY-1

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1200	0	100	CLS_PERCENT
INLEV	1201	0	100	CLS_DB1
OUTLEV	1202	0	100	CLS_DB1
LDELAY	1203	1	670	CLS_MS0
RDELAY	1204	1	670	CLS_MS0
FB	1205	0	99	CLS_PERCENT
FBLOCUT	1206	0	4	CLS_LOCUTS
FBHICUT	1207	0	4	CLS_HICUTS
HICUT	1208	14	30	CLS_FRQ0
ATT	1209	20	100	CLS_DB1

3.6 DELAY-2

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1300	0	100	CLS_PERCENT
INLEV	1301	0	100	CLS_DB1
OUTLEV	1302	0	100	CLS_DB1
DELAY1	1303	1	670	CLS_MS0
DELAY2	1304	1	670	CLS_MS0
HICUT	1305	14	30	CLS_FRQ0
ATT	1306	20	100	CLS_DB1
LEVEL1	1307	0	100	CLS_DB1
PAN1	1308	0	100	CLS_PANL
LEVEL2	1309	0	100	CLS_DB1
PAN2	130A	0	100	CLS_PANR
SPEED	130B	0	40	CLS_SPEEDS0
DEPTH	130C	0	100	CLS_PERCENT
PHASE	130D	0	2	CLS_PHASE1
INV-PAN	130E	0	1	CLS_ONOFF
FB1	130F	-100	100	CLS_PERCENT
FB2	1310	-100	100	CLS_PERCENT
XFB12	1311	-100	100	CLS_PERCENT
XFB21	1312	-100	100	CLS_PERCENT
LOFB	1313	20	100	CLS_DB1
HIFB	1314	20	100	CLS_DB1
LOXOVR	1315	0	30	CLS_FRQ0
HIXOVR	1316	0	30	CLR_FRQ0

3.7 REVPITCH

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1700	0	100	CLS_PERCENT
INLEV	1701	0	100	CLS_DB1
OUTLEV	1702	0	100	CLS_DB1
PITCH1	1703	-12	12	CLS_NON0
FINE1	1704	-50	50	CLS_NON0
PITCH2	1705	-12	12	CLS_NON0
FINE2	1706	-50	50	CLS_NON0
LEVEL1	1707	0	100	CLS_DB1
PAN1	1708	0	100	CLS_PANL
LEVEL2	1709	0	100	CLS_DB1
PAN2	170A	0	100	CLS_PANR
HICUT1	170B	14	30	CLS_FRQ0
ATT1	170C	20	100	CLS_DB1
HICUT2	170D	14	30	CLS_FRQ0
ATT2	170E	20	100	CLS_DB1
FB1	170F	0	100	CLS_PERCENT
FB2	1710	0	100	CLS_PERCENT
XFB12	1711	0	100	CLS_PERCENT
XFB21	1712	0	100	CLS_PERCENT
AMBMIX	1713	0	100	CLS_PERCENT
PREDLY	1714	0	1500	CLS_MS1
SHAPE	1715	0	6	CLS_SHAPE0
SIZE	1716	0	20	CLS_SIZE0
PITCDLY	1717	10	40	CLS_MS0
PITCCFT	1718	5	100	CLS_NON0

3.8 PITCH-1

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1800	0	100	CLS_PERCENT
INLEV	1801	0	100	CLS_DB1
OUTLEV	1802	0	100	CLS_DB1
PITCH1	1803	-12	12	CLS_NON0
FINE1	1804	-1200	1200	CLS_NON0
PITCH2	1805	-12	12	CLS_NON0
FINE2	1806	-1200	1200	CLS_NON0
LEVEL1	1807	0	100	CLS_DB1
PAN1	1808	0	100	CLS_PANL
LEVEL2	1809	0	100	CLS_DB1
PAN2	180A	0	100	CLS_PANR

HICUT1	180B	14	30	CLS_FRQ0
ATT1	180C	20	100	CLS_DB1
HICUT2	180D	14	30	CLS_FRQ0
ATT2	180E	20	100	CLS_DB1
FB1	180F	0	100	CLS_PERCENT
FB2	1810	0	100	CLS_PERCENT
XFB12	1811	0	100	CLS_PERCENT
XFB21	1812	0	100	CLS_PERCENT
DELAY1	1813	0	310	CLS_MS0
DELAY2	1814	0	310	CLS_MS0
DGSPEED	1815	5	50	CLS_NON2
POLYSPD	1816	5	50	CLS_NON0
POLYDLY	1817	5	18	CLS_NON0
DGFILT	1818	0	3	CLS_DGFILTS

3.9 PITCH-2

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1900	0	100	CLS_PERCENT
INLEV	1901	0	100	CLS_DB1
OUTLEV	1902	0	100	CLS_DB1
PITCH	1903	-12	12	CLS_NON0
FINE	1904	-1200	1200	CLS_NON0
FB	1905	0	100	CLS_PERCENT
DELAY	1906	0	310	CLS_MS0
HICUT	1907	14	30	CLS_FRQ0
ATT	1908	20	100	CLS_DB1
DGSPEED	1909	5	50	CLS_NON2
POLYSPD	190A	5	50	CLS_NON0
POLYDLY	190B	5	18	CLS_NON0
DGFILT	190C	0	3	CLS_DGFILTS

3.10 TAPFAC-1

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1B00	0	100	CLS_PERCENT
INLEV	1B01	0	100	CLS_DB1
OUTLEV	1B02	0	100	CLS_DB1
SCALE	1B03	1	100	CLS_PERCENT
PREDLY	1B04	0	1000	CLS_MS0
WIDTH	1B05	0	100	CLS_PERCENT
LASTTAP	1B06	1	18	CLS_NON0
CURTAP	1B07	1	18	CLS_NON0

DELAY1	1B08	0	6230	CLS_MS0
DELAY2	1B09	0	6230	CLS_MS0
DELAY3	1B0A	0	6230	CLS_MS0
DELAY4	1B0B	0	6230	CLS_MS0
DELAY5	1B0C	0	6230	CLS_MS0
DELAY6	1B0D	0	6230	CLS_MS0
DELAY7	1B0E	0	6230	CLS_MS0
DELAY8	1B0F	0	6230	CLS_MS0
DELAY9	1B10	0	6230	CLS_MS0
DELAY10	1B11	0	6230	CLS_MS0
DELAY11	1B12	0	6230	CLS_MS0
DELAY12	1B13	0	6230	CLS_MS0
DELAY13	1B14	0	6230	CLS_MS0
DELAY14	1B15	0	6230	CLS_MS0
DELAY15	1B16	0	6230	CLS_MS0
DELAY16	1B17	0	6230	CLS_MS0
DELAY17	1B18	0	6230	CLS_MS0
DELAY18	1B19	0	6230	CLS_MS0
LEVEL1	1B1A	0	100	CLS_PERCENT
LEVEL2	1B1B	0	100	CLS_PERCENT
LEVEL3	1B1C	0	100	CLS_PERCENT
LEVEL4	1B1D	0	100	CLS_PERCENT
LEVEL5	1B1E	0	100	CLS_PERCENT
LEVEL6	1B1F	0	100	CLS_PERCENT
LEVEL7	1B20	0	100	CLS_PERCENT
LEVEL8	1B21	0	100	CLS_PERCENT
LEVEL9	1B22	0	100	CLS_PERCENT
LEVEL10	1B23	0	100	CLS_PERCENT
LEVEL11	1B24	0	100	CLS_PERCENT
LEVEL12	1B25	0	100	CLS_PERCENT
LEVEL13	1B26	0	100	CLS_PERCENT
LEVEL14	1B27	0	100	CLS_PERCENT
LEVEL15	1B28	0	100	CLS_PERCENT
LEVEL16	1B29	0	100	CLS_PERCENT
LEVEL17	1B2A	0	100	CLS_PERCENT
LEVEL18	1B2B	0	100	CLS_PERCENT
PAN1	1B2C	0	20	CLS_PANG
PAN2	1B2D	0	20	CLS_PANG
PAN3	1B2E	0	20	CLS_PANG
PAN4	1B2F	0	20	CLS_PANG
PAN5	1B30	0	20	CLS_PANG
PAN6	1B31	0	20	CLS_PANG
PAN7	1B32	0	20	CLS_PANG
PAN8	1B33	0	20	CLS_PANG
PAN9	1B34	0	20	CLS_PANG
PAN10	1B35	0	20	CLS_PANG
PAN11	1B36	0	20	CLS_PANG
PAN12	1B37	0	20	CLS_PANG
PAN13	1B38	0	20	CLS_PANG
PAN14	1B39	0	20	CLS_PANG
PAN15	1B3A	0	20	CLS_PANG
PAN16	1B3B	0	20	CLS_PANG
PAN17	1B3C	0	20	CLS_PANG
PAN18	1B3D	0	20	CLS_PANG

LOCUT	1B3E	0	17	CLS_FRQ0
LOATT	1B3F	20	100	CLS_DB1
HICUT	1B40	17	30	CLS_FRQ0
HIATT	1B41	20	100	CLS_DB1
SPEED	1B42	0	40	CLS_SPEEDS0
DEPTH	1B43	0	100	CLS_PERCENT

3.11 AMBIENCE

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1C00	0	100	CLS_PERCENT
INLEV	1C01	0	100	CLS_DB1
OUTLEV	1C02	0	100	CLS_DB1
SHAPE	1C03	0	5	CLS_SHAPE0
SIZE	1C04	0	20	CLS_SIZE0
PREDLY	1C05	0	1000	CLS_MS1
WIDTH	1C06	0	100	CLS_PERCENT
LOCUT	1C07	0	17	CLS_FRQ0
LOATT	1C08	20	100	CLS_DB1
HICUT	1C09	17	30	CLS_FRQ0
HIATT	1C0A	20	100	CLS_DB1
SPEED	1C0B	0	40	CLS_SPEEDS0
DEPTH	1C0C	0	100	CLS_PERCENT
PDLYMUL	1C0D	0	1	CLS_DLYMUL

3.12 DYNAMIC1

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1A00	0	100	CLS_PERCENT
INLEV	1A01	0	100	CLS_DB1
OUTLEV	1A02	0	100	CLS_DB1
BALANCE	1A03	0	100	CLS_PANL
LOWCUT	1A04	0	21	CLS_FRQ1
LMXOVR	1A05	0	29	CLS_FRQ2
MHXOVR	1A06	0	29	CLS_FRQ3
SOFTCLIP	1A07	0	1	CLS_ONOFF
Low Section → COMTHR	1A08	13	124	CLS_DB1
COMRATIO	1A09	0	15	CLS_RATIO1
COMATCK	1A0A	0	15	CLS_DYNATCK
COMREL	1A0B	0	15	CLS_DYNDEC
LIMTHR	1A0C	76	100	CLS_DB1
LIMRATIO	1A0D	0	1	CLS_RATIO3
LIMATCK	1A0E	0	15	CLS_LIMATCK
LIMREL	1A0F	0	15	CLS_DYNDEC

	EXPTHR	1A10	2	100	CLS_CLS_DB1
	EXPRATIO	1A11	0	15	CLS_RATIO2
	EXPATCK	1A12	0	15	CLS_DYNATCK
	EXPREL	1A13	0	15	CLS_DYNDEC
	EXPRANGE	1A14	20	100	CLS_DB1
	LEVEL	1A15	63	124	CLS_DB1OFF18
	CREST	1A16	0	8	CLS_CREST
	DELAY	1A17	0	250	CLS_MS1
	LIMDLY	1A18	0	250	CLS_MS1
	SFTKNEE	1A19	0	1	CLS_ONOFF
	METERS	1A1A	0	5	CLS_MTRRES
	REF0DB	1A1B	64	100	CLS_DB1
Mid Section →	COMTHR	1A1C	13	124	CLS_DB1
	COMRATIO	1A1D	0	15	CLS_RATIO1
	COMATCK	1A1E	0	15	CLS_DYNATCK
	COMREL	1A1F	0	15	CLS_DYNDEC
	LIMTHR	1A20	76	100	CLS_DB1
	LIMRATIO	1A21	0	1	CLS_RATIO3
	LIMATCK	1A22	0	15	CLS_LIMATCK
	LIMREL	1A23	0	15	CLS_DYNDEC
	EXPTHR	1A24	2	100	CLS_CLS_DB1
	EXPRATIO	1A25	0	15	CLS_RATIO2
	EXPATCK	1A26	0	15	CLS_DYNATCK
	EXPREL	1A27	0	15	CLS_DYNDEC
	EXPRANGE	1A28	20	100	CLS_DB1
	LEVEL	1A29	63	124	CLS_DB1OFF18
	CREST	1A2A	0	8	CLS_CREST
	DELAY	1A2B	0	250	CLS_MS1
	LIMDLY	1A2C	0	250	CLS_MS1
	SFTKNEE	1A2D	0	1	CLS_ONOFF
	METERS	1A2E	0	5	CLS_MTRRES
	REF0DB	1A2F	64	100	CLS_DB1
High Section →	COMTHR	1A30	13	124	CLS_DB1
	COMRATIO	1A31	0	15	CLS_RATIO1
	COMATCK	1A32	0	15	CLS_DYNATCK
	COMREL	1A33	0	15	CLS_DYNDEC
	LIMTHR	1A34	76	100	CLS_DB1
	LIMRATIO	1A35	0	1	CLS_RATIO3
	LIMATCK	1A36	0	15	CLS_LIMATCK
	LIMREL	1A37	0	15	CLS_DYNDEC
	EXPTHR	1A38	2	100	CLS_CLS_DB1
	EXPRATIO	1A39	0	15	CLS_RATIO2
	EXPATCK	1A3A	0	15	CLS_DYNATCK
	EXPREL	1A3B	0	15	CLS_DYNDEC
	EXPRANGE	1A3C	20	100	CLS_DB1
	LEVEL	1A3D	63	124	CLS_DB1OFF18
	CREST	1A3E	0	8	CLS_CREST
	DELAY	1A3F	0	250	CLS_MS1
	LIMDLY	1A40	0	250	CLS_MS1
	SFTKNEE	1A41	0	1	CLS_ONOFF
	METERS	1A42	0	5	CLS_MTRRES
	REF0DB	1A43	64	100	CLS_DB1
Other Parameters →	PARLNK ◆	1A44	0	1	CLS_ONOFF
	NOMDELAY	1A45	0	250	CLS_MS1

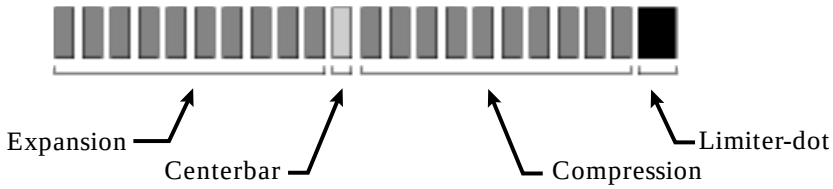
LEV PAGE ◆	1A46	0	2	CLS_CLEVPG
COM PAGE ◆	1A47	0	6	CLS_CCOMPG
LIM PAGE ◆	1A48	0	4	CLS_CLIMPG
EXP PAGE ◆	1A49	0	4	CLS_CEXPPG
LOW METER ♣	1A4A	-32767	32767	CLS_COMMTR
MID METER ♣	1A4B	-32767	32767	CLS_COMMTR
HIGH METER ♣	1A4C	-32767	32767	CLS_COMMTR
LGAIN ♣	1A4D	0	124	CLS_DBF1
MGAIN ♣	1A4E	0	124	CLS_DBF1
HGAIN ♣	1A4F	0	124	CLS_DBF1

Parameters marked with a Fejl! Henvisningskilde ikke fundet. are only used for internal front-panel operations within the M5000 and have no effect on the audio-signal.
Parameters marked with a Fejl! Henvisningskilde ikke fundet. are read-only. Setting their value has no effect.

3.12.1 Meters

The Low, Mid and High meters indicate compression, expansion and limiting. The meters must be polled manually by your software. The M5000 makes sure that you always receive the highest value since the last poll (or lowest, if the expander is active), so all you have to do is poll the meters and display the bargraph. The M5000 is ready with new meter-settings 25 times a second, although you may choose to poll at a lower rate. It is recommended that you poll all 3 meters with the same request-packet in order to minimize MIDI-traffic.

The M5000 supplies you with information to display a meter that looks exactly like the meter on the M5000 display. The meter is 21 segments wide with a center at bar #11. Expansion causes the meter to move left, while compression causes the meter to move right. Limiting causes a dot to appear in the rightmost corner of the meter. Compression and expansion are mutually exclusive, and thus never happen simultaneously:



The M5000 scales the meters according to the METER-resolution for each band (eg. Par #$1A1A for the low band). Don't confuse this parameter with the actual METER-readout (eg. $1A4A for the low band). If the meter-resolution is set to 5dB, then the compression section of the meter shows 5dB of compression, and the expansion section shows 5dB of expansion. See section 0 for a description of how the Meter-resolution parameters affect each other.

3.12.2 Meter Code-Example

The METER-readout parameter contains all necessary information on how to draw the meter. The limiter-dot is stored in the sign-bit (bit 15, if you've converted the parameter-value to a C-type short). You must first extract this bit and mask it off, before calculating the meters.
The compression meter moves in the range 0 up to 127, and the expansion meter moves in the range 256 down to 128. You should only display the first 10 segments in each range. An inactive meter is signified with a compression of 0.

The following demo-code gives a general outline of the decoding process:

```
#define  NOTHING -1

short    compression = NOTHING;
short    expansion = NOTHING;
short    limiting = FALSE;
short    masked;

// Parameter-value is passed in ‘value’

if (value & 0x8000)
    limiting = TRUE;
masked = value & 0xff;
if (masked < 128)
{
    compression = masked;
    if (compression > 10) compression = 10;
}
else
{
    expansion = 256 - masked;
    if (expansion > 10) expansion = 10;
}
if (compression != NOTHING)
    // Draw compression
if (expansion != NOTHING)
    // Draw expansion
if (limiting)
    // Draw limiting
```

3.12.3 Mutual Dependencies

- LMXOVR must never be higher than MHXOVR.
- MHXOVR must never be lower than LMXOVR.
- METERS (Meter-resolution) for each band are hard-linked, meaning that they will always contain the same value. You should only choose to display and manipulate one of the parameters (fx. METERS - Low Band).
- 0DBREF follows the same principle as METERS.

In each of the bands, the thresholds of the compressor and expander limit each other. The following criteria must be met for each of the bands:

- Compressor-threshold must never be lower than Expander-threshold.
- Expander-threshold must never be higher than Compressor-threshold.

3.13 TOOLBOX

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
MIX	1d00	0	100	CLS_PERCENT
INLEV	1d01	0	100	CLS_DB1
OUTLEV	1d02	0	100	CLS_DB1

INS-ON	1d03	0	1	CLS_ONOFF
BALANCE	1d04	-30	30	CLS_DBF1
MONO	1d05	0	100	CLS_PERCENT
LRSWAP	1d06	0	1	CLS_ONOFF
PHASE	1d07	0	2	CLS_LRPHASE
DITHER	1d08	0	3	CLS_DITHER
DITHER-TYPE	1d09	0	2	CLS_DITTYP
MS-IN	1d0a	-180	180	CLS_MSANGLE
MS-OUT	1d0b	-180	180	CLS_MSANGLE
FADECURVE	1d0c	0	1	CLS_MFC
FADER	1d0d	-80	0	CLS_DBF0
METER	1d0e	0	1	CLS_INOUT
RANGE	1d0f	0	2	CLS_MRNGE
TICKS	1d10	0	3	CLS_MTICKS
HOLD	1d11	0	2	CLS_MHOLD
LDELAY	1d12	0	3000	CLS_MS1
RDELAY	1d13	0	3000	CLS_MS1
EQTYPE1	1d14	0	3	CLS_EQTYPE
EQFREQ1	1d15	0	192	CLS_EQFREQ
PWIDTH1	1d16	0	16	CLS_WIDTH0
NWIDTH1	1d17	0	16	CLS_WIDTH1
SSLOPE1	1d18	0	3	CLS_SLOPE0
CSLOPE1	1d19	0	1	CLS_SLOPE1
PGAIN1	1d1a	-120	120	CLS_DBF1
NGAIN1	1d1b	0	100	CLS_DB1
SGAIN1	1d1c	-120	120	CLS_DBF1
EQON1	1d1d	0	1	CLS_ONOFF
EQTYPE2	1d1e	0	1	CLS_EQTYPE
EQFREQ2	1d1f	0	240	CLS_EQFREQ
PWIDTH2	1d20	0	16	CLS_WIDTH0
NWIDTH2	1d21	0	16	CLS_WIDTH1
PGAIN2	1d22	-120	120	CLS_DBF1
NGAIN2	1d23	0	100	CLS_DB1
EQON2	1d24	0	1	CLS_ONOFF
EQTYPE3	1d25	0	1	CLS_EQTYPE
EQFREQ3	1d26	0	240	CLS_EQFREQ
PWIDTH3	1d27	0	16	CLS_WIDTH0
NWIDTH3	1d28	0	16	CLS_WIDTH1
PGAIN3	1d29	-120	120	CLS_DBF1
NGAIN3	1d2a	0	100	CLS_DB1
EQON3	1d2b	0	1	CLS_ONOFF
EQTYPE4	1d2c	0	3	CLS_EQTYPE
EQFREQ4	1d2d	112	240	CLS_EQFREQ
PWIDTH4	1d2e	0	16	CLS_WIDTH0
NWIDTH4	1d2f	0	16	CLS_WIDTH1
SSLOPE4	1d30	0	3	CLS_SLOPE0
CSLOPE4	1d31	0	1	CLS_SLOPE1
PGAIN4	1d32	-120	120	CLS_DBF1
NGAIN4	1d33	0	100	CLS_DB1
SGAIN4	1d34	-120	120	CLS_DBF1
EQON4	1d35	0	1	CLS_ONOFF

♣ LPPM	1d36	-32768	32767	CLS_PPM72
♣ RPPM	1d37	-32768	32767	CLS_PPM72
♣ PHASE-CORR	1d38	-32768	32767	CLS_BAR72

Parameters marked with a ♣ are read-only. Setting their value has no effect.
If you wish to display the VU-meters, you should not use the local meters in this algorithm (LPPM and RPPM). A TOOLBOX algorithm has front-panel VU-meters as any other algorithm. If you use those, you only need to write one routine to display meters.

3.13.1 Special Considerations

Each of the bands in the TOOLBOX can be set to a number of equalizer-types. The Low and High bands have 4 selections, while the two Mid bands have only 2 selections. Each type of equalizer has it's own set of unique associated parameters. For example, a Shelve-type has a Slope-parameter, while a Parametric-type has a Width-parameter. As a product of this, the Low-band has 10 parameters, although not all of them are used at the same time. In the M5000, the display is swapped, to show only the relevant parameters. If the Parametric-type parameters are shown on screen, the Shelve-type parameters still exist as seperate identities, although they have no immediate function and aren't displayed. You will need to accomodate for this.

For each band, the EQTYPE, EQFREQ and EQON parameters stay fixed (meaning that they aren't swapped). The rest of the parameters for each band (Width/Slope and Level) are swapped. Please refer to the M5000 front-panel, to see how this works. If possible, you should retain this scheme as opposed to physically changing the type of objects in your software application.

4 System-Parameters

4.1 Overview

System-parameters are parameters just like algorithm-parameters except that they apply to the DSP-card as such or perhaps to the entire M5000 frame. Each slot has it's own standard set of parameters, such as Bypass, but care must be taken, because the parameters have the same numbers in all the slots. It is the slot# that distinguishes between the parameters, not the actual parameter-number.

4.2 System-Parameters

Parameter-name	ID (Hex)	Min (Dec)	Max (Dec)	Class (Type)
SYSMIXMODE	0100	0	2	CLS_MIXMODE
SYSBYPASS	0101	0	1	CLS_ONOFF
SYSGIN	0102	0	100	CLS_DB1
SYSCHANMODE	0103	0	3	CLS_CHNLMODE
SYSPHASE	0104	0	1	CLS_POSNEG
SYSIOMODE	0105	(See		CLS_IOMODE
♣ SYSCURRATE	0108	Section		
SYSMCLOCK	0109	0)		
SYSAIN	010a	-12	12	CLS_DBF0???
SYSAOUT	010b	-18	12	CLS_DBF0???
♣ SYSLOCKSTAT	010c	0	1	CLS_ONOFF
SYSMETERSHOW	0110	0	1	CLS_INOUT
SYSDOTYPE	0111	0	2	CLS_DOTYPE
SYSDOCPY	0112	0	2	CLS_DOCPY
SYSDADEMP	0113	0	1	CLS_ONOFF
SYSR68LEV	0114	0	1	CLS_ONOFF
SYSFSTTRIG	0115	0	1	CLS_ONOFF
SYSMETERL	0401			See section 0
SYSMETERR	0402			See section 0

Parameters marked with a ♣ are read-only.

4.3 VU-Meters

Even though the M5000 front-panel VU-meters only have 10 segments, the meter-information is in fact far more detailed. The SYSMETERL and SYSMETERR parameters contain the actual meter-readout in 1/8 dB steps, which is why steps must be taken to produce a useable meter.

4.3.1 Communication

The meters must be polled manually by your software at 1/25 second intervals. You may choose to poll at a lower rate, although a higher rate won't result in any improvement. It is recommended that you poll both the left- and right meters in the same request-packet in order to minimize MIDI-traffic.

The M5000 cannot guarantee to reply with the meters in the same packet, although the packets will be very close in time. For this reason, your software should be able to handle the left- and right channels separately.

4.3.2 Calculations

Some bits of SYSMETERL and SYSMETERR are reserved, so you should AND with 0x03ff before processing.

The meter-reply contains the actual meter position in 1/8 dB steps. A value of 0 means peak, while a value of 1 means -1/8 dB and so forth. If you wish to produce a meter that shows the amplitude in whole dB steps, simply divide the number by 8.

The DSP-clip flag is found in bit 13, which can be accessed by ANDing with 0x2000. This flag has a built-in timeout, so all you need to do is print it. The M5000 front-panel VU-meters use the 0dB LED to signal DSP-clipping, but this information should be printed separately wherever possible.

4.4 Hardware-Specific Parameter Ranges

Some parameters have ranges that depend on the physical DSP-card configuration.

5 Preset-Handling

5.1 Overview

The M5000 preset-handling facilities allow you to request information about presets (including edit-buffers) as well as transfer presets to/from the M5000. A convenient way of recalling presets via Sysex is also offered.

5.2 Conversions And IDs

A few new data-types are introduced for preset-transfers:

5.2.1 Preset-numbers

When a preset is referred to, it's number and bank is included in the Preset#. A Preset# is always spread over 2 bytes as an unsigned short and is combined/derived with the same methods as used for parameter-numbers.

In the C programming-language, the Preset# is calculated as follows:

```
presetNumber = number + (bank << 12);
```

where bank is one of the following:

```
0:   Current Preset (Edit Buffer)
1:   ROM
2:   RAM
3:   FILE
```

5.2.2 Algorithm-IDs

Each type of DSP-algorithm has a unique ID:

```
1   REVERB1
2   CHORUS
3   REVPITCH
4   REVERB2
5   NONLIN1
6   DELAY1
7   PITCH1
8   PITCH2
9   DELAY2
10  REVERB3
11  SAMPLER
12  AMBIENCE
13  TAPFAC1
14  DYNAMIC1
15  TOOLBOX
16  PAREQ
17  CORE
```

5.2.3 Preset-Names

Preset-names are always 8 ASCII-characters long with the unused character-places padded with spaces.

5.3 Preset Information

The following request-message allows you to request information about a preset or the edit-buffer:

```
Sysex-start   $f0
TC ID            $33
Device#       xx
Card#            Unused, unless edit-buffer is requested
Packet-type   $03 - Request Preset Info
<Preset#>     Preset#
Sysex-end     $f7
```

The M5000 will reply with the following:

```
Sysex-start   $f0
TC ID            $33
Device#       xx
Card#            Unused, unless edit-buffer is replied
Packet-type   $05 - Preset Info
<Name>           Preset-name
<Preset#>     Preset# (original Preset# if edit-buffer)
<Byte>           Algorithm-ID
<Byte>           Edited, 0=FALSE, 1=TRUE
Sysex-end     $f7
```

5.4 Recall Preset

This dump allows you to recall presets along the same path as the rest of your Sysex-communications:

```
Sysex-start   $f0
TC ID            $33
Device#       xx
Card#            xx
Packet-type   $02 - Recall Preset
<Preset#>     Preset#
Sysex-end     $f7
```

Please note, that if the M5000 front-panel is showing the program-recall page, the Preset# will start flashing, because the preset you are in the process of recalling via the M5000 front-panel no longer is the current preset (because of this Sysex-dump).

6 C Programming-Interface

6.1 Overview

The C programming-interface is provided as a means to display the correct value for any given type of parameter. Without this interface, you would have to create all tables and conversions yourself in order to display the value of all parameters.

The interface consists of 2 files, CLASS.C and CLASS.H, which can be downloaded from the TC User-Club BBS. The phone-numbers are listed in the beginning of this manual.

6.2 Using The Interface

The interface is platform-independent and only requires the ANSI Standard Libraries stdio, string and math to be present.

The only routine you need to call is this:

```
void class_GetStr (char *s, WORD idClass, short v);
```

where `s` is an array of `7 chars` to receive the string, `idClass` is the class-number as found in the algorithm-listings and `v` is the value.

Keep in mind that the string isn't automatically null-terminated. You can null-terminate the string by providing an 8-character string to `class_getStr` and then setting the 8th character to 0.

Because this code is taken directly from the M5000 application-software, the result is always printed as a 7-character string which is padded with spaces. You are free to modify the code to display the parameter-text in a less short-hand way, but keep in mind that the CLASS.C and CLASS.H files probably will be updated in the future to support new algorithms. Your CLASS.C and CLASS.H files will always remain compatible with existing algorithms, but if you would like to support new algorithms, you must either make all your modifications again or add the new parameter-types by hand.

Always make sure that the string you provide to `class_getStr` is large enough to contain the reply.

A few of the classes are irrelevant to most applications, but they have been left in the interface to simplify the process of providing it. You should not use classes such as CLS_BAR72, because their character-string reply requires custom-characters that are only available in the M5000.