MACKIE HUI MIDI protocol

The results of a 2-day reverse-engineering-session
by theageman.

5/1/2010

www.ssei-online.de

Content

Preface..... 3
Transmitting data..... 4
Ping..... 4
Text..... 4
4-character channel and 'SELECT-ASSIGN' text displays..... 4
2*40 character main display..... 5
VU-Meters..... 5
Timecode display..... 6
V-Pot rings..... 6
LEDs..... 7
Faders..... 8
Relays..... 8
Click..... 8
Beep..... 8
Recieving data..... 10
Ping..... 10
Switches..... 10
V-Pots..... 10
Jog wheel..... 11
Faders..... 11
Footswitches..... 12
System reset..... 12
Addendum..... 13
Small display character set..... 13
Large display character set..... 13
Hardware layout..... 14

Preface

Even though i kindly asked Mackie several times to give me some information about this 13-year-old product, i never recieved that information. But since this device communicates via MIDI, everybody can sniff this protocol. So i decided to publish my experimentation-results here.

This is the result of a 2-day reverse-engineering-session on a MACKIE HUI (firmware version 1.45). It's so sad that every developer has to spend this time (ok, perhaps only a few hours for an experienced developer) only because he doesn't get the the chance to obtain an sdk by Mackie. We are talking about a 13 Year old product (May, 2010). By the way: the MIDI implementation for the Emagic Logic Control is available for free (even for end users).

Anyway, i *think* i covered every possible aspect in this text. But i'm not really sure. So if you find a feature, i missed, please feel free to extend this document with your knowledge.
Please note that this is for informational purposes only, without any warranty.

Transmitting data

All MIDI data displayed here, is hexadecimal.

There are many sysex bulks, used to transport data larger than 3 bytes. All these sysex bulks start with a header (`f0 00 00 66 05 00`) and end with 'end of sysex' (`f7`). Whenever you find a `<hdr>` in this text, just substitute that with `f0 00 00 66 05 00`.

This header is composed of:

`f7` : start of sysex data

`00 00 66` : manufacturer id (mackie)

`05` : product id (hui)

`00` : i even don't know, if this is part of a sysex header. I could look it up, but i'm too lazy ;)

Have fun,

theageman.

Ping

It is important to send a ping in regular intervals. This will keep the HUI in online mode. If the HUI doesn't get a ping for about 2 seconds, it will go offline.
Sending a ping is quite simple (note on, key 0, velocity 0):

```
90 00 00
```

The HUI will respond to a ping with a ping-reply:

```
90 00 7f
```

This indicates an existing connection to the HUI. If you don't get a ping-reply, the HUI is probably not connected to the computer anymore.

The funny thing is, that you actually don't have to send a ping at all, to control the HUI. If you never send a ping, the HUI won't go offline and you can control everything *EXCEPT* the faders!

But since you probably want to control the faders, just send the ping every second or so.

Text

4-character channel and 'SELECT-ASSIGN' text displays

```
       f0 00 00 66 05 00 10 yy gg hh ii jj f7
- or -
       <hdr> 10 yy gg hh ii jj f7

       yy : 0..7 = channel 1..8
            8    = SELECT-ASSIGN
       gg,
```

```
hh,
ii,
jj : four MACKIE HUI characters (see addendum)
```

All characters should be in the range `00..7f`, otherwise there might be some scrolling chars/undefined behavior.

2*40 character main display

The display is divided into 8 zones (`0..7`):

```
0000000000111111111122222222223333333333
4444444444555555555566666666667777777777
```

Text can be sent for up to 4 zones simultaneously (perhaps even more? I didn't try out more than 4 zones.):

```
f0 00 00 66 05 00 12 z1 g0 g1 g2 g3 g4 g5 g6 g7 g8 g9
                    [z2 h0 h1 h2 h3 h4 h5 h6 h7 h8 h9]
                    [z3 i0 i1 i2 i3 i4 i5 i6 i7 i8 i9]
                    [z4 j0 j1 j2 j3 j4 j5 j6 j7 j8 j9] f7
```

- or -

```
<hdr> 12 z1 g0 g1 g2 g3 g4 g5 g6 g7 g8 g9
        [z2 h0 h1 h2 h3 h4 h5 h6 h7 h8 h9]
        [z3 i0 i1 i2 i3 i4 i5 i6 i7 i8 i9]
        [z4 j0 j1 j2 j3 j4 j5 j6 j7 j8 j9] f7
```

`[...]` denotes optional data.
`z1..z4`: the above mentioned zones (`00..07`). They can be in any order.
`g0..g9`: 10 (ten) MACKIE HUI-2 characters (see addendum).
`h0..h9`: 10 (ten) MACKIE HUI-2 characters (see addendum).
`i0..i9`: 10 (ten) MACKIE HUI-2 characters (see addendum).
`j0..j9`: 10 (ten) MACKIE HUI-2 characters (see addendum).

Please note that this display has a different character set! All characters should be in the range 10..7f, otherwise there might be some scrolling chars/undefined behavior.

VU-Meters

Format:

```
a0 0y sv

y : channel (0..7)
s : side (left/right)
    s = 0 : side = left
    s = 1 : side = right
v : value (0..c)
    v = c : signal >=   0dB; red (clip)
    v = b : signal >=  -2dB; yellow
    v = a : signal >=  -4dB; yellow
    v = 9 : signal >=  -6dB; yellow
    v = 8 : signal >=  -8dB; green
    v = 7 : signal >= -10dB; green
    v = 6 : signal >= -14dB; green
    v = 5 : signal >= -20dB; green
    v = 4 : signal >= -30dB; green
    v = 3 : signal >= -40dB; green
    v = 2 : signal >= -50dB; green
```

```
v = 1 : signal >= -60dB; green
v = 0 : signal <  -60dB; all leds off
```

Timecode display

The TC display consists of eight 7-segment displays (called digits here). To keep MIDI data bandwidth as low as possible data gets transmitted with lsb first. Every digit except the first one (the rightmost) has a decimal point (dp). It is possible to send up to 8 digits to the surface within one sysex frame.

Format:

```
f0 00 00 66 05 00 11 y0 [y1 [y2 [y3 [y4 [y5 [y6 [y7]]]]]]] f7
```

- or -

```
<hdr> 11 y0 [y1 [y2 [y3 [y4 [y5 [y6 [y7]]]]]]] f7
```

where `y0` is the rightmost digit (lsb) and `y7` is the leftmost digit (msb).
`[...]` denotes optional data.

Valid values for `y0..y7` are :

```
00 : '0'     10 : '0.' (only valid for y1..y7)
01 : '1'     11 : '1.' (only valid for y1..y7)
02 : '2'     12 : '2.' (only valid for y1..y7)
03 : '3'     13 : '3.' (only valid for y1..y7)
04 : '4'     14 : '4.' (only valid for y1..y7)
05 : '5'     15 : '5.' (only valid for y1..y7)
06 : '6'     16 : '6.' (only valid for y1..y7)
07 : '7'     17 : '7.' (only valid for y1..y7)
08 : '8'     18 : '8.' (only valid for y1..y7)
09 : '9'     19 : '9.' (only valid for y1..y7)
0a : 'A'     1a : 'A.' (only valid for y1..y7)
0b : 'b'     1b : 'b.' (only valid for y1..y7)
0c : 'C'     1c : 'C.' (only valid for y1..y7)
0d : 'd'     1d : 'd.' (only valid for y1..y7)
0e : 'E'     1e : 'E.' (only valid for y1..y7)
0f : 'F'     1f : 'F.' (only valid for y1..y7)
```

Notes: There might be some more combinations i didn't check out. When the surface loses the connection to a host for example, it displays 'OFF-LINE'. But i think the internal mcu manages this type of message.

V-Pot rings

The V-Pot rings consist of 11 leds (numbered `0..a`). The center-led is number 5.

Format:

```
b0 1y vv

y : channel (0..7)/param (8..b)

v =  0 : ...........   v = 10 : ...........   v = 20 : ...........
v =  1 : *.........   v = 11 : ******.....   v = 21 : *.........
v =  2 : .*........   v = 12 : .*****.....   v = 22 : **........
v =  3 : ..*.......   v = 13 : ..****.....   v = 23 : ***.......
v =  4 : ...*......   v = 14 : ...***.....   v = 24 : ****......
v =  5 : ....*.....   v = 15 : ....**.....   v = 25 : *****.....
v =  6 : .....*....   v = 16 : .....*.....   v = 26 : ******....
v =  7 : ......*...   v = 17 : .....**....   v = 27 : *******...
v =  8 : .......*..   v = 18 : .....***...   v = 28 : ********..
```

```
v =  9 : ........*..   v = 19 : .....****..   v = 29 : *********..
v =  a : .........*.   v = 1a : .....*****.   v = 2a : **********.
v =  b : ..........*   v = 1b : .....******   v = 2b : ***********

v = 30 : ...........
v = 31 : .....*.....
v = 32 : ....***....
v = 33 : ...*****...
v = 34 : ..*******..
v = 35 : .*********.
v = 36 : ***********
v = 37 : ***********
v = 38 : ***********
v = 39 : ***********
v = 3a : ***********
v = 3b : ***********
```

There is also a small led under the encoder that can be turned on by adding `40` to `v`.

LEDs

The HUI is divided into 29(decimal) zones (`00..1d`). Each zone can have up to 8 ports. It seems that the internal multiplexers/demultiplexers are 8 bit devices. All leds have the same zone/port pair as the corresponding button. For example the led for the 'next bank' button is: zone 0a, port 3 (`0a/3`). The button is also at `0a/3`.

To control a led you must have selected the right zone first. After selecting a zone you can control all 8 ports within that zone:

Zone select:

```
b0 0c zz
zz(zone) = 00..1d
```

Switch on port:

```
b0 2c 4p
p(port) = 0..7
```

Switch off port:

```
b0 2c 0p
p(port) = 0..7
```

It is also possible to control multiple port within the same zone using 'running status':

```
b0 2c 00 2c 01 2c 02 2c 03 2c 04 2c 05 2c 06 2c 07
```

switches off ports `0..7` in the pre-selected zone.

The zone/port selection can be combined as well:

```
b0 0c 08 2c 40 2c 41 2c 42
```

This will select zone `08` and switch on ports `0..2`!

But the HUI will always send a zone select/port control pair if you press a button. That is 6 bytes for a button press and another 6 bytes for a button release - 12 bytes in total.

For information regarding the zones and ports please see addendum.

Faders

Faders consist of 2 different devices. Every fader has a switch and a motor. The switch will be closed when you touch the fader. When you release the fader, the switch will be opened. When you move the fader it will send fader positions.

Sending a fader position is quite simple.

Format:

```
b0 0z hi
b0 2z lo
```

Where z denotes the zone (the same as the fader number) - `0..7`
hi and lo are in the range `00..7f` (7 bit values).
The total resolution is therefor 14 bit (`0..3fff = 0..16383 dec`).
As far as I know, the internal resolution of the faders are 9 bit, so the HUI will set the 5 least significant bits of the lo-value to zero.

The format for sending fader positions can be shortened using 'running status' to:

```
b0 0z hi 2z lo
```

Relays

There are two controlable relays in the hui. They have the same zone/port as the corresponding footswitches.
Zone: `1d`
Relay1/footswitch1 : `port 0`
Relay2/footswitch2 : `port 1`

For example to switch on relay2 you would have to send:
`b0 0c 1d` (to select zone `1d`, if not already selected)
`b0 2c 41` (switch on `port 1`)

To turn it off, send:

```
b0 2c 01
```

Click

The HUI features a small click sound used to indicate button presses etc. To create that sound, send:

```
b0 0c 1d (select zone 1d)
b0 2c 42 (switch on port 2)
```

This is the only port that doesn't have to be switched off afterwartds. You may send as many 'switch on'-commands as you like, without sending a 'switch off'-command in between.

Beep

The HUI also features a beeper for indicating error conditions etc. To switch on the beeper, send:

```
b0 0c 1d (select zone 1d)
b0 2c 43 (switch on port 3)
```

The beeper will sound until you send a 'switch off'-command:

```
b0 2c 03
```

Recieving data

Ping

As long as you recieve a ping-reply from the HUI (`90 00 7f`) the HUI is still online. But remember that you will have to send a ping first in order to recieve a ping reply in regular intervals. Sending a ping (`90 00 00`) should be done about every second.

Switches

There is a significant difference between transmitting port switches and receiving port switches. To select a zone you would send something like `b0 0c zz`. When the HUI selects a zone (because the user pressed a button for example), it sends `b0 0f zz` (where `zz` denotes the zone). To *send* a port switch, you would say something like `b0 2c yp`, where `y` would be `0` to turn off, `y` would be `40` to turn on, and p would be the port in question (`0..7`). But you will *receive* something like `b0 2f yp` if the user presses or releases a button on the HUI.

To summarize this:

Command	Transmit	Receive
Zone select	b0 0c zz	b0 0f zz
Port on	b0 2c 4p	b0 2c 4p
Port off	b0 2c 0p	b0 2f 0p

The HUI will always send a complete pair of 'zone select'/'port on/off' pair to the computer. For example, when the user presses the solo button on the fifth channelstrip, the HUI will send:

```
b0 0f 04  - meaning : select zone 04
b0 2f 43  - meaning : port 3 switched on
```

After the user releases the button, the HUI will send:

```
b0 0f 04  - meaning : select zone 04
b0 2f 03  - meaning : port 3 switched off
```

As far as i can tell, the HUI *never* makes use of a 'running status'.

For information regarding zones and port, please have a look at HUIZONES.txt.

V-Pots

V-Pots send its data using delta valus.

Format:

```
b0 4p vv
```

where `p` is the V-Pot number and `vv` is the delta value. `p` can be anything from `0` to `c`. This is just a linear mapping of the V-Pots from left to right. That means, when `p` equals `c` then the 'scroll'-V-Pot has been operated.

vv denotes the delta value.
If vv>40 then delta=vv-40.
If vv<40 then delta=-vv.
vv never seems to equal 40.

I was able to obtain deltas as high as 2d (by turning the knobs real fast).

Jog wheel

The jog wheel sends its data using delta values as well.

Format:

```
b0 0d vv
```

vv denotes the delta value.
if vv>40 then delta=vv-40
if vv<40 then delta=-vv
vv never seems to equal 40.

delta seems to be in the range [-f..-1,1..f].

Faders

As already mentioned in HUIREFTX.txt, the faders have 2 functions. The first function is 'touch fader' and 'release fader'. And the second function is 'move fader'.

A complete sequence for altering a fader looks like this:

```
touch fader
move fader
move fader
.
.
.
move fader
release fader
```

This behavior gives you the chance, to manage automation. When, for example, your software constantly moves the fader according to recorded automation data, but still shall be able to update that data using fader movements made by the user, your software can react to 'touch fader'-commands and 'release-fader'-commands.

Format:

```
'touch fader':
b0 0f 0z
b0 2f 40

'release fader':
b0 0f 0z
b0 2f 00

'move fader':
b0 0z hi
b0 2z lo
```

where z is the corresponding zone (the fader number, 0..7) and

`value = (hi << 7) + lo`, giving a range of `0..3fff`.

Footswitches

Footswitch#	Close	Open
1	`b0 0f 1d b0 2f 40`	`b0 0f 1d b0 2f 00`
2	`b0 0f 1d b0 2f 41`	`b0 0f 1d b0 2f 01`

System reset

Whenever the HUI is turned on or off, it sends one ore more `ff`. This is MIDI slang and means 'System reset'.

Addendum

Small display character set

Hex	Display	Hex	Display	Hex	Display	Hex	Display
00	ì	10	è	20	Space	30	0
01	↑	11	Æ	21	!	31	1
02	→	12	æ	22	"	32	2
03	↓	13	Å	23	#	33	3
04	←	14	å	24	$	34	4
05	¿	15	Ä	25	%	35	5
06	à	16	ä	26	&	36	6
07	Ø	17	Ö	27	'	37	7
08	ø	18	ö	28	(	38	8
09	ò	19	Ü	29	)	39	9
0a	ù	1a	ü	2a	*	3a	:
0b	Ň	1b	°C	2b	+	3b	;
0c	Ç	1c	°F	2c	,	3c	<
0d	ê	1d	ß	2d	-	3d	=
0e	É	1e	£	2e	.	3e	>
0f	é	1f	¥	2f	/	3f	?
40	@	50	P	60	`	70	p
41	A	51	Q	61	a	71	q
42	B	52	R	62	b	72	r
43	C	53	S	63	c	73	s
44	D	54	T	64	d	74	t
45	E	55	U	65	e	75	u
46	F	56	V	66	f	76	v
47	G	57	W	67	g	77	w
48	H	58	X	68	h	78	x
49	I	59	Y	69	i	79	y
4a	J	5a	Z	6a	j	7a	z
4b	K	5b	[	6b	k	7b	{
4c	L	5c	\	6c	l	7c	|
4d	M	5d	]	6d	m	7d	}
4e	N	5e	^	6e	n	7e	~
4f	O	5f	_	6f	o	7f	▒

Large display character set

Hex	Display	Hex	Display	Hex	Display	Hex	Display
00		10	l1	20	Space	30	0
01		11	l2	21	!	31	1
02		12	l3	22	"	32	2
03		13	l4	23	#	33	3
04		14	full	24	$	34	4
05		15	r4	25	%	35	5
06		16	r3	26	&	36	6
07		17	r2	27	'	37	7
08		18	r1	28	(	38	8
09		19	♪	29	)	39	9
0a		1a	°C	2a	*	3a	:
0b		1b	°F	2b	+	3b	;
0c		1c	▼	2c	,	3c	<
0d		1d	►	2d	-	3d	=
0e		1e	◄	2e	.	3e	>
0f		1f	▲	2f	/	3f	?
40	@	50	P	60	`	70	p
41	A	51	Q	61	a	71	q
42	B	52	R	62	b	72	r
43	C	53	S	63	c	73	s
44	D	54	T	64	d	74	t
45	E	55	U	65	e	75	u
46	F	56	V	66	f	76	v
47	G	57	W	67	g	77	w
48	H	58	X	68	h	78	x
49	I	59	Y	69	i	79	y
4a	J	5a	Z	6a	j	7a	z
4b	K	5b	[	6b	k	7b	{
4c	L	5c	¥	6c	l	7c	|
4d	M	5d	]	6d	m	7d	}
4e	N	5e	^	6e	n	7e	→
4f	O	5f	_	6f	o	7f	←

Hardware layout

Zone	Port 0	Port 1	Port 2	Port 3	Port 4	Port 5	Port 6	Port 8
00	fader	select	mute	solo	auto	v-sel	insert	rec/rdy
01	fader	select	mute	solo	auto	v-sel	insert	rec/rdy
02	fader	select	mute	solo	auto	v-sel	insert	rec/rdy
03	fader	select	mute	solo	auto	v-sel	insert	rec/rdy
04	fader	select	mute	solo	auto	v-sel	insert	rec/rdy
05	fader	select	mute	solo	auto	v-sel	insert	rec/rdy
06	fader	select	mute	solo	auto	v-sel	insert	rec/rdy
07	fader	select	mute	solo	auto	v-sel	insert	rec/rdy
08	ctrl/clt	shift/ad	editmode	undo	alt/fine	option/a	edittool	save
09	mix	edit	transprt	mem-loc	status	alt		
0a	<- chanl	<- bank	chanl ->	bank ->				
0b	output	input	pan	send e	send d	send c	send b	send a
0c	assign	default	suspend	shift	mute	bypass	recrdyal	
0d	down	left	mode	right	up	scrub	shuttle	
0e	talkback	rewind	fast fwd	stop	play	record		
0f	|<rtz	end>|	on line	loop	qck pnch			
10	audition	pre	in	out	post			
11	input 3	input 2	input 1	mute	discrete			
12	output 3	output 2	output 1	dim	mono			
13	0	1	4	2	5	.	3	6
14	enter	+						
15	7	8	9	-	clr	=	/	*
16	timecode	feet	beat	rudesolo				
17	plug in	pan	fader	sendmute	send	mute		
18	trim	latch	read	off	write	touch		
19	phase	monitor	auto	suspend	create	group		
1a	paste	cut	capture	delete	copy	separate		
1b	f1	f2	f3	f4	f5	f6	f7	f8/esc
1c	ins/para	assign	select 1	select 2	select 3	select 4	bypass	compare
1d	fs/rlay1	fs/rlay2	click	|beep				

Zone-names:

Zone	Name
0	channel strip 1
1	channel strip 2
2	channel strip 3
3	channel strip 4
4	channel strip 5
5	channel strip 6
6	channel strip 7
7	channel strip 8
8	keyboard shortcuts
9	window
0a	channel selection
0b	assignment 1
0c	assignment 2
0d	cursor movement/mode/scrub/shuttle
0e	transporter main (big switches)
0f	transporter loop/rtz/end
10	transporter punch
11	monitor input
12	monitor output
13	num pad 1
14	num pad 2
15	num pad 3
16	timecode leds (no associated buttons)
17	auto enable
18	auto mode
19	status/group
1a	edit
1b	function keys
1c	parameter edit
1d	click/beep/relay/footswitch (no associated buttons or leds)