Roland

MIDI PROCESSING UNIT

MPU-401

TECHNICAL REFERENCE MANUAL

MIDI PROCESSING UNIT

MPU-401

TECHNICAL REFERENCE MANUAL

[CPU Version 1.5A 5/31/85]

[This Manual Version 1.5 5/29/85]

Warning

This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC Rules. Only Computer certified to comply with the Class B limits may be attached to this equipment. Operation with non-certified computing device is likely to result in interference to radio and TV reception.

Radio and Television Interference

The equipment described in this manual generates and uses radio-frequency energy. If it is not installed and used properly, that is, in strict accordance with our instructions, it may cause interference with radio and television reception.

This equipment has been tested and complies with the limits for a Class B computing device in accordance with the specifications in Subpart J, Part 15, of FCC rules. These rules are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that the interference will not occur in a particular installation, especially if you use a "rabbit ear" television antenna. (A "rabbit ear" antenna is the telescoping-rod type usually contained on TV receivers.)

You can determine whether your computer is causing interference by turning it off. If the interference stops, it was probably caused by the computer or its peripheral devices. To further isolate the problem:

- Disconnect the peripheral devices and their input/output cables one at a time. If the interference stops, it is caused by either the peripheral device or its I/O cable. These devices usually require shielded I/O cables. For Roland peripheral devices, you can obtain the proper shielded cable from your dealer.

If your computer or its peripheral devices does cause interference to radio or television reception, you can try to correct the interference by using one or more of the following measures:

- Turn the TV or radio antenna until the interference stops.
- Move the computer or its peripheral devices to one side or the other of the TV or radio.
- Move the computer or its peripheral devices farther away from the TV or radio.
- Plug the computer or its peripheral devices into an outlet that is on a different circuit than the TV or radio. (That is, make certain the computer or its peripheral devices and the radio or television set are on circuits controlled by different circuit breakers or fuses.)
- Consider installing a rooftop television antenna with coaxial cable lead-in between the antenna and TV.

If necessary, you should consult your dealer or an experienced radio/television technician for additional suggestions. You may find helpful the following booklet, prepared by the Federal Communications Commission.:

"How to Identify and Resolve Radio-TV Interference Problems"

This booklet is available from the U.S. Government Printing Office, Washington, DC 20402, stock number 004-000-00345-4.

TABLE OF CONTENTS
==========

1. INTRODUCTION

1.1

The Roland MPU-401 MIDI PROCESSING UNIT is designed using a CPU and a custom LSI hand-shake controller for the MIDI bus system.

The MPU-401 connects to a HOST computer via the DATA bus and uses handshaking and interrupts. The MPU-401 may be used as a memory-mapped or I/O mapped device.

Roland recognizes the difficulties and weaknesses of using a simple UART or 'DUMB' interface to the MIDI bus and has decided to set out on a program to development this 'intelligent' microprocessor based MIDI interface. This led to the development of a special LSI handshaking controller which drastically reduces the parts count in the interface. Connection to almost any kind of personal computer is now possible due to the simplicity of this hardware interface. The use of interrupts as opposed to polled port techniques greatly adds to the potential of MIDI software applications.

1.2 ABOUT THIS MANUAL

This reference manual is for the version 1.5A ROM of the Roland MPU-401 MIDI PROCESSING UNIT only. Differences between this ROM version and earlier ones are listed in the section 'MPU VERSION DELTA GUIDE'. All references in this manual to 'MPU-401' are applicable to all MPU units (i.e. MPU-APL, MPU-MSX, etc.) unless markes with '(!)'. ITEMS WITH THIS MARK APPLY TO THE MPU-401 ONLY.

1.3 FEATURES OF THE MPU-401

a> The use of an intelligent interface for the MIDI system allows the HOST computer to perform other tasks while the MPU-401 is recording and/or playing. The MPU-401 operates as a 'background' processor using interrupts to the HOST computer to send recorded data to the HOST's memory and requesting data from the HOST to send out on the MIDI OUT port.

b> Because of the above, operations such as disk I/O (if the HOST is using a DMA controller), screen updates, user interface, or any other application functions may be run concurrently while recording or playing MIDI data.

c> The MPU-401 functions as an 8 track MIDI data recorder, looking after all aspects of the recording and playing of musical data, except for HOST memory management. Each track can be polyphonic and multichanneled. The only jobs that the HOST computer must do to look after the MPU-401 are: 1. The sending and receiving of commands to the MPU-401 and; 2. The reading and writing of MIDI data to memory, and organization of memory.

d> The MPU-401 looks after all external SYNC functions, including MIDITIMING CLOCKS ($F8), a Roland DIN SYNC output, an FSK tape sync port (!), and internal timing control. The internal 'TIMEBASE' may be set to one of seven different divisions per beat; 48, 72, 96, 120, 144, 168 or 192.

e> The most commonly used functions are enabled at POWER UP as DEFAULTS, but these default values may be changed on the fly by the HOST. For example, the default TEMPO is 100 beats per minute (BPM), but this may be changed in real time to any tempo within the range of 8 BPM to 250 BPM. Also, the RELATIVE TEMPO and RATE of TEMPO CHANGE (GRADUATION) may be changed in real time, thus allowing the HOST to become the 'CONDUCTOR' of the MPU-401 by automatically changing tempo while recording or playing.

f> The MPU-401 contains 1 RECORD COUNTER; 8 PLAY COUNTERS; 1 CONDUCTOR timer for timing special data; a MIDI IN TABLE which flags each incoming MIDI NOTE ON and clears it upon receiving NOTE OFF or ALL NOTES OFF; 4 CHANNEL REFERENCE TABLES which flag all MIDI NOTE ONs played by the MPU-401 and clears them when sending a NOTE OFF, an ALL NOTES OFF or a STOP PLAY; a 8 PLAY BUFFERS which hold the next MIDI note event to be played in each track; a PROGRAM CHANGE BUFFER which holds the last MIDI PROGRAM CHANGE since the most recent stop; a MIDI Channel filter to allow only selected MIDI channels to pass to the host; a MIDI MESSAGE FILTER for screening out selected MIDI controllers; a UART MODE for applications not requiring the intelligence of the MPU-401; a programmable audio METRONOME which will play any time signature; a synchronization system for MIDI sync, Roland SYNC 24 and FSK (!) type TAPE SYNC; and a full set of intelligent commands which may be used by a HOST application to create any type of MIDI system.

1.4 A NOTE TO PROGRAMMERS

Programming the MPU-401 for various MIDI applications is not a difficult endeavor. Required however is a working knowledge of the following:

a) A sound understanding of programming in the language of one's choice and at least a basic knowledge of the computer being written for.

b) The ability to understand hexadecimal (base 16) arithmetic.

c) An understanding of interrupt handling by the host computer.

d) A thorough knowledge of the entire MIDI 1.0 protocol.

e) A good understanding of the MPU-401 interface by reading this reference in its entirety.

2. Connection to the HOST

The MPU-401 connection to the HOST computer may be memory or I/O mapped. Connection is made to 8 DATA BUS lines, 1 RD*, 1 WR*, 1 RESET*, 1 SELECT*, 3 ADDRESS, 1 DSR*, 2 GROUND and 2 VCC of 5 volt dc.
These connections are made to a 25 pin female type 'D' connector.

Pin Connections

Signal	Pin #	Direction	Signal	Pin #	Direction
D0	1	I/O	A1	18	I
D1	14	I/O	A2	6	I
D2	2	I/O	RD*	9	I
D3	15	I/O	WR*	22	I
D4	3	I/O	CS*	10	I
D5	16	I/O	DSR* #	23	O
D6	4	I/O	RES*	11	I
D7	17	I/O	+5vdc	12 & 24	I
A0	5	I	GND	13 & 25	I

Note...
\# The DSR* (Data Set Ready) output is OPEN COLLECTOR. It is usually connected to the HOST's INTERRUPT input.

Signal Definition

CS*	A2	A1	A0	RD*	WR*	:	Description	
0	x	x	0	0	0	:	ILLEGAL	
0	x	x	0	0	1	:	Read data	(DATAPORT)
0	x	x	0	1	0	:	Write data	(DATAPORT)
0	x	x	0	1	1	:	No operation	
0	x	x	1	0	0	:	ILLEGAL	
0	x	x	1	0	1	:	Read status	(STATPORT) #
0	x	x	1	1	0	:	Write a command	(COMPORT)
0	x	x	1	1	1	:	No operation	
0	0	0	x	x	x	:	Port # 0	
0	0	1	x	x	x	:	Port # 1	
0	1	0	x	x	x	:	Port # 2	
0	1	1	x	x	x	:	Port # 3	
1	x	x	x	x	x	:	Not selected	

Note...
\# STATPORT bit 7 : DSR* bit 6 : DRR*
Normally the A1 and A2 must be zeroes, as the hardware in the MPU-401 is connected as PORT 0. (default)
The hardware in the MPU-401 can be changed if the HOST needs to use more than 2 (up to 4) sets of the MPU-401s as port 0 - 3.

This page shows the defined names of data, commands and status used by the MPU-401. These words (MPU MESSAGE, MPU MARK, etc.) are used often in this manual.

WHILE RECORDING

receives

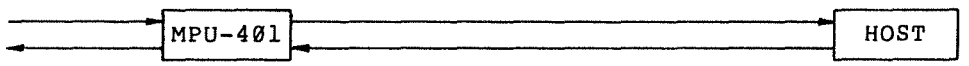
MIDI MESSAGE from MIDI IN

sends

STATUS..... to STATPORT

1. a MIDI MESSAGE with a leading timing byte
2. an MPU MARK with a leading timing byte
3. an MPU MESSAGE

.. to DATAPORT



transmits

MIDI MESSAGE to MIDI OUT

receives

an MPU COMMAND..... from COMPORT

1. a MIDI MESSAGE with a leading timing byte
2. an MPU MARK with a leading timing byte
3. an OVER FLOW

... from DATAPORT

WHILE PLAYING

DATA IN STOP MODE

receives

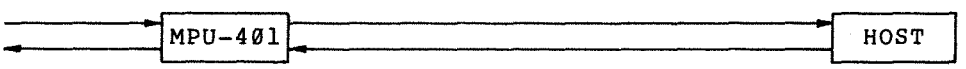
MIDI MESSAGE from MIDI IN

sends

STATUS..... to STATPORT

1. a MIDI MESSAGE
2. an MPU MESSAGE

... to DATAPORT



transmits

MIDI MESSAGE to MIDI OUT

receives

an MPU COMMAND..... from COMPORT

a MIDI MESSAGE..... from DATAPORT

WANT TO SEND DATA

---

3. The MPU DATA FORMAT

The MPU-401 sends data to the HOST while RECORDING, and receives data from the HOST while PLAYING in the format shown here. The HOST should read or send the exact number of bytes defined. Reading or sending fewer than expected will cause the MPU-401 to experience a 'loop problem' as it waits for a certain sized data packet.

1st BYTE	2nd BYTE	3rd BYTE	4th BYTE	DESCRIPTION
$F8 *				MPU MESSAGE timing value = 240
$00 - $EF*	$F8 **			MPU MARK
"	$F9 **			MPU MARK
"	$FC **			MPU MARK
"	$80 - $BF ***	$00 - $7F	$00 - $7F	MIDI MESSAGE
"	$E0 - $EF ***	$00 - $7F	$00 - $7F	****
"	$C0 - $DF ***	$00 - $7F		****
"	$00 - $7F	$00 - $7F ****		MIDI data bytes
"	$00 - $7F			(MIDI running status)

Next bytes are sent from the MPU-401 only. See Section 4.1

$FC,$FD				MPU MESSAGES
$F0-$F7,$F9				MPU data request
$FF	$F0 - $FF	($00 - $7F ..)		MPU MIDI mess. follows

NOTES ON MPU DATA FORMAT:

* LEADING TIMING BYTE

- The first byte (values from $00 to $EF) are TIMING VALUES which precede standard MIDI MESSAGES or MPU MARKS.

- While recording: when the MPU-401 receives a MIDI message, the timing value is copied from the MPU-401's RECORD COUNTER which is then increased by the internal clock. Then the

received MIDI message bytes will follow. The RECORD COUNTER will be cleared after the process. If the MPU-401 does not receive MIDI messages and the record counter reaches 240, it will send out the MPU MESSAGE $F8 'timing overflow'. (See section 4).

- Normally in the DATA IN STOP mode, leading timing bytes are not sent. (See MPU commands $8B and $34.)

- While playing: when the MPU-401 requests the next track data, the HOST MUST send it data; either a MIDI MESSAGE or an MPU MARK with a leading timing byte. The MPU-401 puts the timing value into the PLAY COUNTER of the track, then the counter is decreased by the internal clock. When the counter becomes zero the MPU-401 sends the MIDI message out or executes the mark, and requests the next data for that track. If the leading timing value is zero, the MPU-401 will transmit the message out and request the next data immediately.

- While playing: when the MPU-401 requests next track data, if the HOST sends an $F8 (timing overflow), the play counter of the track in the MPU-401 is set to 240. When the counter reaches zero, the MPU-401 will send nothing and request the next track data.

- When the HOST wants to transmit a MIDI message to MIDI OUT using the MPU COMMAND of $D0 to $D7 (Want To Send Data), the data must be sent in standard MIDI format but without a leading timing byte.

** MPU MARKS

- Do not confuse MPU-401 MARKS with MPU MESSAGES which are explained in the next section.

- MPU MARKS are sent while recording and may be used while playing by the HOST. They are preceded with a LEADING TIMING byte, and do not have any following bytes.

- They are not used in the DATA IN STOP mode, and can not be used with the WANT TO SEND DATA commands ($D0 to $D7 and $DF).

$F8 NO OPERATION

- While recording: the MPU-401 does not sent out this mark.

- While playing: the MPU-401 will send nothing out the MIDI OUT port if sent this mark and then will request next track data.

$F9 MEASURE END

- While recording: the MPU-401 sends Measure Ends at the rate determined by the $E6 command, and is enabled with the $8D command.

- While playing: the metronome counter in the MPU-401 will reset upon receiving an $F9 and the metronome will play its primary accent if in accented mode ($85) (!). Nothing is sent to the MIDI OUT port of the MPU-401.

$FC DATA END

- While recording: when the MPU-401 receives a STOP RECORD command from the HOST, it sends the host this mark with a leading timing byte.

- While playing: if the MPU-401 receives this mark in a track as the response to a REQUEST TRACK DATA ($F0 - $F7), the MPU-401 will shut down the track after the time given by the leading timing byte, and will stop requesting data in that track.

- When all tracks are finished playing: the MPU-401 sends an $FC message, without the leading timing byte, to let the HOST know all tracks are finished playing. The HOST must send a STOP PLAY command. (See section 5.)

*** MIDI STATUS

- While recording: the MPU-401 creates MIDI RUNNING STATUS, so MIDI status bytes are not always sent. This means that the HOST must remember the running status while receiving the MIDI messages. The HOST can otherwise misread messages, and cause the MPU-401 'loop trouble' if the host tries to accept too many or too few bytes. Running status is used to keep memory usage to a minimum and to avoid sending unneeded data over the MIDI bus.

- While playing: the MPU-401 keeps RUNNING STATUS for each track so that the status byte is not always necessary. The same caution must be used to avoid looping problems, but if the data received from the MPU-401 is sent back exactly, this will not happen.

**** MIDI THIRD BYTE

- With two byte MIDI messages such as $Cn and $Dn, only 1 byte of data is sent while recording or is requested while playing. (n is a value from $Ø to $F which indicates any MIDI channel 1 to 16).

4. MPU-401 MESSAGES

While recording, values from $00 to $EF are TIMING VALUES which precede MIDI MESSAGES or MPU MARKS. In the DATA IN STOP mode ($8B), they are the status bytes ($80 - $EF) and data bytes ($00 - $7F) of MIDI messages. (See section 3).

Values from $F0 to $FF are not MIDI MESSAGES. They are messages send by the MPU-401 as follows:

4.1 SINGLE BYTE MESSAGES

$F8 TIMING OVERFLOW

- While recording: when the RECORD TIMING COUNTER reaches 240 ($F0), an $F8 is substituted to indicate the value, then the record counter is cleared.

- While playing: when the MPU-401 requests next track data or conductor, this message can be used for loading value of 240 to play counter by the HOST. It should be used alone, and not be confused with timing bytes. After the play counter has zeroed out, the MPU-401 will request more data to play.

$FC ALL END

- When all tracks are finished playing, this message is sent to the HOST without a leading timing byte. The MPU-401 will still remain in the PLAY mode, but will not request next track data. This message is used as a flag to tell the host that all tracks have finished playing and a stop sequence should be done by the HOST.

$FD CLOCK TO HOST

- This rate is set by $E7, and enabled by $95. (Default = every 1/8th note). The $FD MESSAGE can be used by the host program as a counter for lead-ins, for regulating a graphic metronome or any function done in exact increments of beats.

$FE ACK (ACKNOWLEDGE OF COMMAND)

- The ACK is used by the MPU-401 to let the HOST know that any commands sent have been properly received and will be acted on. As an example, when the HOST sends a SET TEMPO command to the MPU-401, there is a possibility that the MPU-401 will simultaneously be sending a byte of incoming MIDI data. In that case the MPU-401 would want the HOST to read this byte and its following bytes, even while the HOST is waiting to send the rest of the command. This case could cause both the MPU-401 and the HOST go into 'loop' trouble. It must be possible for the HOST to accept incoming data while sending a

command and then receive the ACK.

- When the HOST sends a command, it has to wait to get a data byte and check if it is an ACK ($FE). If the byte is not ACK, the HOST must accept the data byte and following bytes first, then wait for ACK again.

- An ACK will not be sent back upon sending a SYSTEM RESET to leave the UART MODE ($3F).

4.2 REQUEST-DATA MESSAGES (MPU REQUEST)

$F0 - $F7 TRACK DATA REQUEST

- While playing: the MPU-401 requests next data of all active tracks by sending these commands. When the play counter of a track becomes zero, the MPU-401 will send messages of $F0 to $F7 which correspond with track number 1 to 8, to the HOST.

MESSAGE	:	$F0	$F1	$F2	$F3	$F4	$F5	$F6	$F7
TRACK	:	1	2	3	4	5	6	7	8

- It then expects the next data for that particular track from the HOST as one of the following:

1. a leading timing byte and a MIDI VOICE MESSAGE.
2. a leading timing byte and an MPU MARK.
3. an $F8 (timing overflow).

$F9 CONDUCTOR DATA REQUEST

- While playing: when the CONDUCTOR is set ON and the counter of the MPU-401 CONDUCTOR zeros out the MPU-401 will send an $F9 to the HOST.

- It then expects a leading timing byte and an MPU command from the HOST. However, the MPU command should be sent as a data byte through DATAPORT. The MPU-401 will load the timing value to the CONDUCTOR play counter, then the counter will be decreased by the internal clock. The MPU-401 will not send an ACK for commands sent as CONDUCTOR data.

- When the counter reaches zero, the MPU-401 sends another $F9 to the HOST, and then executes the latest command.

- The HOST can also use an MPU MESSAGE $F8 for loading value 240 to the counter, an MPU MARK $FC with a leading timing byte for the end of the conductor data, an MPU MARK $F9 with a leading timing byte for measure end, and an MPU MARK $F8 with a leading timing byte for NO OPERATION.

- When MPU COMMANDS $DØ to $DF which require a following data byte (such as WANT TO SEND DATA) are sent to the conductor, only the leading timing byte and command should be sent at the first CONDUCTOR REQUEST. When the timer has counted down and it is time to execute the command, the CONDUCTOR will send another CONDUCTOR REQUEST ($F9) at which time the data for the command should be sent followed by the next timing byte and command. (See Section 7.3 NOTE ** and Section 8 PLAY A TRACK WITH CONDUCTOR : ON)

4.3 DATA FOLLOWED MESSAGE

$FF SYSTEM MESSAGE

- MIDI SYSTÉM EXCLUSIVE, COMMON or REAL TIME messages are preceded by the SYSTEM MESSAGE MARK $FF.

a. MIDI EXCLUSIVE data begins with an $FØ, and ends when EOX ($F7) or any other MIDI STATUS is received.
b. MIDI COMMON messages are as specified.
c. MIDI REAL TIME messages are single bytes.
d. MIDI TIMING CLOCK ($F8) is never sent to the HOST.
e. ACTIVE SENSING ($FE) and the MIDI message RESET ($FF) are never sent to the HOST.

1st byte	2nd byte	3rd byte	4th byte	Description
$FF	$FØ	($ØØ-$7F ...)	EOX $F7	MIDI Exclusive Mess.
$FF	$F2	$ØØ-$7F	$ØØ-$7F	SongPosition Pointer
$FF	$F3	$ØØ-$7F		Song Select
$FF	$F6			Tune Request
$FF	$FA,$FB,$FC			MIDI Real Time mess.

5. MPU COMMANDS

A table of all the MPU COMMANDS are listed in the back of this reference manual. Be careful to never send an undefined command number to the MPU.

5.1 Mode Commands $00 to $2B

Command Format:
(binary): 00aabbcc
where: aa, bb and cc control:

1. aa (Bits 4 and 5) : Record MIDI data from MIDI IN.
2. bb (Bits 2 and 3) : Play MIDI data to MIDI OUT.
3. cc (Bits 0 and 1) : Transmit MIDI REAL TIME messages ($FA, $FB, $FC) to MIDI OUT.

NOTE: Any one command may contain combinations of these 3 categories.

Function Description.

Bit 5 4

0 0 : no function.

0 1 : STOP RECORDING
The MPU-401 stops increasing the record counter, then it stops sending MIDI messages received from MIDI IN to the HOST.
It sends an MPU MARK $FC (End of Data) with a leading timing byte to the HOST. Then the record counter is cleared.

1 0 : START RECORDING or RECORD STAND-BY
This prepares the MPU-401 to begin recording sequence. For RECORD STAND-BY (no MIDI START) with REAL TIME AFFECTION : ON the MPU will wait until it receives either a MIDI START ($FA) or MIDI CONTINUE ($FB) from MIDI IN or another command from the HOST which has MIDI START or CONTINUE before clearing the record and play counters and starting. This command is also used for punch-in sequences where the MPU may already be playing.

START RECORDING:

a. The MPU-401 clears the record counter. It then sends MIDI PROGRAM CHANGE messages for all acceptable channels which have been received from MIDI IN to the HOST since the last RECORD or PLAY STOP. This can be defeated by sending a CLEAR PM command ($B9) after setting acceptable channels ($EE and $EF). Each message is sent with a dummy leading timing byte of $00. If no MIDI PROGRAM CHANGES were received then nothing will be sent to the HOST until new data is sent to the MIDI IN of

the MPU-401.

b. The MPU-401 will begin to increase its internal RECORD COUNTER. If it receives MIDI VOICE messages of any acceptable channels from MIDI IN it will send them to the HOST. Each message is sent with a leading timing byte which is calculated by the record counter based on the current TIME BASE of the MPU-401. The record counter will then be cleared. This means that the leading timing byte indicates the timing value from the last MIDI message which was sent to the HOST, to the message which the MPU-401 is now going to send.

If no MIDI messages come for a long time, the MPU-401 will send an MPU MESSAGE $F8 (timing overflow) when the record counter reaches to 240 and the counter will be cleared.

While recording: the HOST should read the exact number of bytes defined by the MPU-401 data format. (See section 3.)

When the MPU-401 receives an ALL NOTES OFF message and all notes are OFF in that channel, it will not send any message to the HOST. However, if some notes are ON in the MIDI IN TABLE the MPU-401 will turn OFF these notes in the table and create NOTE OFF messages to send to the HOST, but the ALL NOTES OFF message itself will not be sent.

Normally, any MODE messages such as OMNI ON / OFF, MONO, POLY are not passed to the HOST. However the command $35 will allow them to be passed.

If the MPU-401 is set to MIDI SYNC mode with the $82 command it will begin recording at a tempo based on the incoming MIDI CLOCKS ($F8s). If the MPU-401 is in FSK sync mode, it will wait for an incoming tape sync tone from the TAPE IN jack.

RECORD STAND-BY:

The MPU-401 executes 'a' from above and then waits for another command which has a MIDI START or CONTINUE from the HOST, or if in REAL TIME AFFECTION : ON mode waits for a MIDI START ($FA) or CONTINUE ($FB) message from MIDI IN.

1 1 : other command area.

Bit 3	2	
0	0	: no function.
0	1	: STOP PLAYING: The MPU-401 will discontinue decreasing the play counters so that it will not request any additional track data.

It will transmit to the MIDI OUT port of the MPU-401 the NOTE OFF messages (with velocity zero) of the notes which have been turned on while in PLAY mode. The CHANNEL REFERENCE TABLES will be cleared.

1 0 : START PLAYING
The MPU-401 will start to decrease the PLAY COUNTERS for all active tracks. If the counter reaches zero, it will transmit the MIDI message which has been received from the HOST to MIDI OUT. It will then send the MPU MESSAGE TRACK DATA REQUEST ($F0 - $F7) and will wait for the HOST to send the next MIDI data. The HOST should send the exact number of bytes defined by the MPU data format. (See section 3.)

By clearing the PLAY COUNTERS ($B8) before the START PLAY, the MPU-401 will not transmit a MIDI message from the PLAY BUFFER at first, but will immediately send a TRACK DATA REQUEST to the HOST. This is the most typical sequence for START PLAY. When PLAY is stopped, the PLAY BUFFERS (timing and data) are not cleared so a PLAY CONTINUE can be possible.

An ALL NOTES OFF message is created in the MPU-401 when all notes are turned OFF in the channel.

1 1 : no function.

Bit 1 0
---

0 0 : no function.

0 1 : Transmits a MIDI STOP ($FC) to MIDI OUT.
Resets DIN SYNC signal to low.
Stops the FSK TONE modulation in TAPE OUT. (!)
Saves METRONOME COUNTER, MEASURE END COUNTER and CLOCK TO HOST COUNTER for next CONTINUE PLAY.

1 0 : Transmits a MIDI START ($FA) to MIDI OUT.
Resets DIN SYNC CLOCK. Sets DIN SYNC signal to high.
Starts modulation of FSK TONE at TAPE OUT. (!)
Clears METRONOME COUNTER, MEASURE END COUNTER and CLOCK TO HOST COUNTER.

1 1 : Transmits a MIDI CONTINUE ($FB) to MIDI OUT.
Resets DIN CLOCK.
Sets DIN SYNC signal to high.
Starts modulation of FSK TONE at TAPE OUT.
Sets DIN CONTINUE signal to high then reset to low.
Restores METRONOME COUNTER, MEASURE END COUNTER and CLOCK TO HOST COUNTER.

This table shows each bit mapped possibility for the START, STOP, and CONTINUE commands of the MPU-401.

	COMMAND	MIDI	PLAY	RECORD	FUNCTION
	$00	-	-	-	Do Not Use
	$01	Stop	-	-	
	$02	Start	-	-	
	$03	Continue	-	-	
	$04	-	Stop	-	
*	$05	Stop	Stop	-	STOP PLAY
	$06	Start	Stop	-	
	$07	Continue	Stop	-	
	$08	-	Start	-	
	$09	Stop	Start	-	
*	$0A	Start	Start	-	START PLAY
*	$0B	Continue	Start	-	CONTINUE PLAY
	$0C - $0F	-	-	-	Do Not Use
	$10	-	-	Stop	
*	$11	Stop	-	Stop	STOP RECORD
	$12	Start	-	Stop	
	$13	Continue	-	Stop	
	$14	-	Stop	Stop	
*	$15	Stop	Stop	Stop	Stop Over-Dub
	$16	Start	Stop	Stop	
	$17	Continue	Stop	Stop	
	$18	-	Start	Stop	
	$19	Stop	Start	Stop	
	$1A	Start	Start	Stop	
	$1B	Continue	Start	Stop	
	$1C - $1F	-	-	-	Do Not Use
*	$20	-	-	Stand-by	RECORD STAND-BY **
	$21	Stop	-	Stand-by	
*	$22	Start	-	Start	START RECORD
*	$23	Continue	-	Start	CONTINUE RECORD
	$24	-	Stop	Stand-by	
	$25	Stop	Stop	Stand-by	
	$26	Start	Stop	Start	
	$27	Continue	Stop	Start	
	$28	-	Start	Stand-by	**
	$29	Stop	Start	Stand-by	
*	$2A	Start	Start	Start	Start Over-Dub
*	$2B	Continue	Start	Start	Start Over-Dub
	$2C - $2F	-	-	-	Do Not Use

NOTES:

* Codes marked with '*'s are the most commonly used ones. The others are usually not used alone.

** If the MIDI timing condition has already been started by another MPU COMMAND (such as $ØA), or by MIDI START ($FA) or CONTINUE ($FB) from MIDI IN (with REAL TIME AFFECTION : ON), then these commands will turn RECORD ON. Otherwise, RECORD will go into STANDBY until the MIDI timing condition is set by another MPU COMMAND or until MIDI START or CONTINUE is received (with REAL TIME AFFECTION : ON only).

5.2 SWITCHES FOR INITIALIZATION

These commands switch the functions in only one direction. To escape from these modes, only the RESET command ($FF) can be used. Normally these modes are chosen when the HOST's application program starts only.

$30 ALL NOTES OFF : OFF

- Disables the MPU-401 from sending out a MIDI ALL NOTES OFF message ($Bn $7B $00) which is created by the MIDI IN and CHANNEL REFERENCE TABLES when all notes are turned OFF.
- Disables the MPU-401 from sending out any MIDI ALL NOTES OFF message received from MIDI IN.

$32 NO REAL TIME

- No MIDI real time messages ($F8, $FA, $FB, $FC) are created to the MIDI OUT port of the MPU-401. Normally (before this command is sent) MIDI CLOCKS ($F8) are send to the MIDI OUT port of the MPU-401 at all times.

$33 ALL THRU : OFF

- This command effects the way command $88 will work. In its default mode, $88 is THRU OFF only for acceptable (or unfiltered) channels (see command $EE and $EF).
- Using the $33 command makes $88 THRU OFF for all channels.
- Also, this command disables sending MIDI IN SYSTEM COMMON messages to MIDI OUT.

$34 WITH TIMING BYTE : ON

- A dummy leading timing byte of zero is sent with all MIDI VOICE messages while in DATA IN STOP mode.
- When DATA IN STOP mode is active ($8B) MIDI VOICE messages sent to the HOST normally do not have a leading timing byte associated with them.

- This command allows use of one routine to process all incoming MIDI data sent to the HOST rather than one to process MIDI data with timing bytes in RECORD mode, and another to process MIDI data without timing bytes in DATA IN STOP mode.

$35 MODE MESS : ON

- Normally the MPU-401 masks all MODE MESSAGES received from MIDI IN. This command allows the host to see all MODE MESSAGES (OMNI ON / OFF, POLY, MONO, LOCAL ON / OFF) except ALL NOTE OFF ($Bn, $7B, $00). The 'n' is a value from 0 to $F which indicates MIDI channel number for the mode message.
- This mode would be used if the HOST was to be used as a sound generator, synthesizer module, or diagnostic tool.

$37 EXCLUSIVE THRU : ON

- This allows SYSTEM EXCLUSIVE messages from MIDI IN to pass to MIDI OUT of the MPU-401. The command $97 will cancel this mode.
- On power up, the MPU-401 does not let MIDI SYSTEM EXCLUSIVE messages to pass through to the MIDI OUT.

$38 COMMON TO HOST : ON

- This will allow the MPU-401 to send MIDI System Common messages (Song Select, Song Position Pointer and Tune Request) received from MIDI IN to the HOST. They are sent with an MPU SYSTEM MESSAGE ($FF) which will precede these messages. (See section 4.3).
- In this mode, when REAL TIME AFFECTION is ON ($91), and a Song Position Pointer ($F2) or a SONG SELECT ($F3) is received from MIDI IN, the MPU-401 will automatically clear the internal PLAY COUNTERS (the same as MPU-401 command $B8).

$39 REAL TIME TO HOST : ON

- This allows the MPU-401 to send MIDI Real Time messages (i.e Start, Continue and Stop) received from MIDI IN to be sent to the HOST. They are sent with an MPU SYSTEM MESSAGE ($FF) which will precede these messages. (See section 4.3)

5.3 SPECIAL FUNCTION SWITCH

The next command will set the MPU-401 into a simple UART mode in which it will no longer recognize any of the MPU-401 commands except SYSTEM RESET ($FF). All functions, timers and tables are disabled.

$3F UART mode.

- The MPU-401 will act as a simple UART. This function is used when the programmer wishes to disable all MPU functions, so that the MIDI bus can directly be accessed by the HOST. For example, during debug sessions of MIDI hardware / software, or using the MIDI bus just as a method of data transmission.

To return from this mode back to normal MPU-401 operation use the SYSTEM RESET command ($FF). The MPU-401 will not return an ACK ($FE) in this one case.

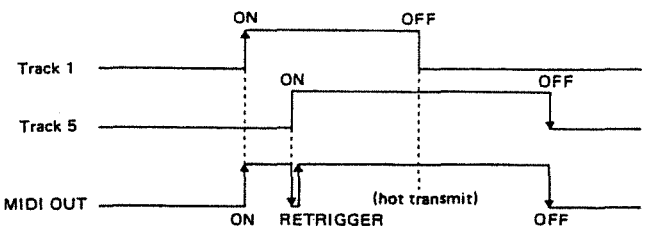
5.4 CHANNEL REFERENCE TABLES

Some polyphonic synthesizers use an assignment system which assigns the same note to the same voice module. To compensate for this reason the CHANNEL REFERENCE TABLES are incorporated into the MPU-401's structure.

While playing, when MIDI NOTE ON events of the same key number from different tracks using the same MIDI channel OVERLAP, the CHANNEL REFERENCE TABLES are used to retrigger the note by sending a NOTE OFF and immediately sending a new NOTE ON.

The CHANNEL REFERENCE TABLES are also used to wait until the notes of all tracks become OFF before transmitting a NOTE OFF message to the MIDI OUT of the MPU-401. (See diagram below.)

Overlap Timing Diagram



There are four REFERENCE TABLES provided in the MPU-401 so the pitch overlapping of up to four MIDI channels may be supervised by the MPU-401. The least significant 4 bits refer to the channel number (1 to 16).

LSBs :	0	1	2	3	4	5	6	7	8	9	$A	$B	$C	$D	$E	$F
CHANNEL :	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

These CHANNEL REFERENCE TABLES may be turned -ON- or -OFF- by using the commands from $98 to $9F. For example, if the reference table A has been switched OFF by $98 command, it will be switched ON automatically by using any $40 - $4F command.

$40 - $4F
: Sets CHANNEL REFERENCE TABLE A to control the MIDI channel defined by LSBs of the command.

$50 - $5F
: Sets CHANNEL REFERENCE TABLE B to control the MIDI channel defined by LSBs of the command.

$60 - $6F
: Sets CHANNEL REFERENCE TABLE C to control the MIDI channel defined by LSBs of the command.

$70 - $7F

Sets CHANNEL REFERENCE TABLE D to control the MIDI channel defined by LSBs of the command.

5.5 SWITCHES

$80 INT CLOCK

- This allows use of the INTERNAL clock of the MPU-401 by the HOST for recording or playing. INT CLOCK is used if the MPU-401 is to be the master clock of the MIDI system. This is the default mode on power up. (See commands $C0 to $C8 and $E0 to $E2).

$81 FSK CLOCK (!)

- The MPU-401 will sync its internal timers with an incoming FSK tone from the TAPE IN. The MPU-401 will generate an FSK tone to the TAPE OUT automatically whenever a command with MIDI START or MIDI STOP is sent (in either RECORD or PLAY). This tone can be read back in FSK mode to synchronize current recording or playback with previously recorded tracks on tape.

$82 MIDI CLOCK

- The MPU-401 expects MIDI CLOCK ($F8) from the MIDI IN port, and syncs its internal clock with the received MIDI CLOCKS.

$83 METRONOME : ON - WITHOUT ACCENTS

- This command turns the audio metronome ON but with no secondary accents. (See section 1).

$84 METRONOME : OFF

- Turns the MPU-401 metronome tone OFF.

$85 METRONOME : ON - WITH ACCENTS (!)

- This command turns the audio metronome ON with primary accents (on the beat) and secondary (on the bar) accents.

$86 BENDER : OFF

- This switch prevents PITCH WHEEL (pitch bender) and CONTINUOUS CONTROL messages from the instrument connected to MIDI IN from reaching the HOST. Use this command when the HOST program does not want MIDI CONTROLLER messages (to save RAM), and the TRANSMITTER connected to the MIDI IN can not cancel these messages.

$87 Bender : ON

- This allows PITCH WHEEL and CONTINUOUS CONTROL messages to reach the HOST.

This table shows the effect of commands $86 and $87 on incoming MIDI VOICE MESSAGES:

Voice Message		$86	$87
$8n	note off event	ON	ON
$9n	note on event	ON	ON
$An	note after touch	OFF	ON
$Bn	control change		
	Ø - 63	OFF	ON
	64 - 95	ON	ON
$Cn	program change	ON	ON
$Dn	channel after touch	OFF	ON
$En	pitch wheel	OFF	ON

n's are LSBs ($Ø to $F) indicating MIDI channels 1 to 16.

$88 MIDI THRU : OFF

- Disables sending MIDI IN VOICE messages of acceptable channels (See $EE and $EF.) to MIDI OUT.
- For disabling all MIDI THRU channel voice messages, use the $33 command.

$89 MIDI THRU : ON

- Enables transmission of all MIDI IN VOICE messages to MIDI OUT.

- The MIDI IN TABLE supervises all MIDI IN NOTE EVENTS (ON / OFF) for verification, such as:

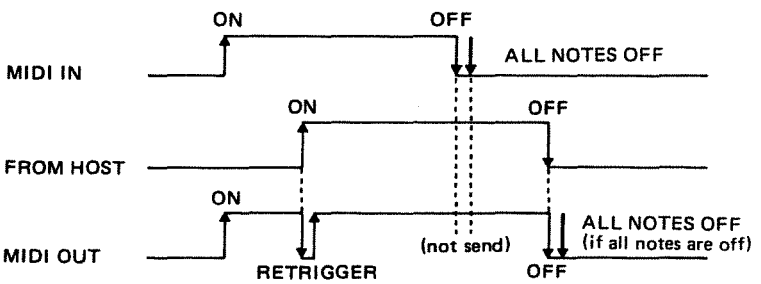
a. If a note has been turned ON in the MIDI IN TABLE for a received NOTE ON, the MPU-401 will retrigger the note (transmit a NOTE OFF then a NOTE ON).

b. If a note in the table has not been turned ON for a received NOTE OFF, the MPU-401 will not transmit the NOTE OFF.

c. If the MPU-401 is playing and NOTE ON events of the same key number and channel come from the HOST and MIDI IN, the CHANNEL REFERENCE TABLES and MIDI IN TABLE are used to RETRIGGER the NOTE and to wait before sending NOTE OFF messages to MIDI OUT until the NOTES in both the HOST and MIDI IN turn OFF.

d. While the MPU-401 is playing, if any notes are ON in the REFERENCE TABLES when an ALL NOTE OFF message is received from the MIDI IN port, the ALL NOTE OFF is not passed to the MIDI OUT.

e. When a MIDI ALL NOTES OFF is received from MIDI IN the MIDI IN TABLES are cleared and each NOTE OFF is sent to MIDI OUT instead of the ALL NOTES OFF itself.



$8A DATA IN STOP MODE : OFF

$8B DATA IN STOP MODE : ON

- When this switch is ON the MPU-401 will send MIDI VOICE messages without a leading timing byte to the HOST while RECORD is in STOP. This mode is useful for interrupt based non-sequencing functions (i.e. channelizers, etc.) where recording is not used.
- Used along with the $34 command a leading timing byte of zero will be sent with the MIDI VOICE messages. This is useful during a LEAD-IN so any keys struck prior to the START RECORD can be stored by the HOST.

$8C SEND MEASURE END : OFF

$8D SEND MEASURE END : ON

- If this switch is ON, MEASURE END MARKS ($F9) with leading timing bytes are sent while in RECORD mode. The MPU-401 calculates the time of the measure end based on the TIME SIGNATURE which is calculated by the number of metronome beeps per measure and by the number of MIDI clocks per metronome. (See $E4 and $E6 commands.)

$8E CONDUCTOR : OFF

$8F CONDUCTOR : ON

- While in PLAY mode: with the CONDUCTOR : ON the MPU-401 will request CONDUCTOR data by sending MPU MESSAGE $F9. When the HOST receives the $F9, it must send conductor data which should consist of:

  a. A leading timing byte of when the command should be executed followed by:
  b. A command byte, but SENT VIA THE DATAPORT as data.
  c. Any following data byte(s) if the command requires them. (e.g. Change tempo, etc.)

- The CONDUCTOR : ON command should only be sent BEFORE the CLEAR PLAY COUNTERS command ($B8) has been sent.

$90 REAL TIME AFFECTION : OFF

$91 REAL TIME AFFECTION : ON

- The MPU-401 will respond to MIDI REAL TIME messages ($FA, $FB, $FC) from MIDI IN. These will affect the RECORD and PLAY functions of the MPU-401 in any sync mode.
- With REAL TIME AFFECTION : ON the MPU-401 will automatically start and stop recording and playback when the appropriate MIDI REAL TIME MESSAGE is received. It is possible to enable

and disable REAL TIME AFFECTION while in RECORD or PLAY modes.
- This command also allows MIDI REAL TIME MESSAGES ($FA, $FB, $FC) to pass through from MIDI IN to MIDI OUT of the MPU-401.

$92 FSK TO INTERNAL

- Sets the resolution of the FSK TAPE SYNC clock to the INTERNAL TIMEBASE of the MPU-401. (See commands $C2 to $C8). The default of the TIMEBASE resolution is 120 divisions per beat.

$93 FSK TO MIDI

- Sets the resolution of the FSK TAPE SYNC clock to MIDI clock rate, which is 24 per beat. Both this command and command $92 affect both the FSK IN and FSK OUT clock rates.

$94 CLOCK TO HOST : OFF

$95 CLOCK TO HOST : ON

- When this switch is ON, the MPU-401 CLOCK TO HOST ($FD) is sent to the HOST at a rate determined by the $E7 command.

$96 EXCLUSIVE TO HOST : OFF

$97 EXCLUSIVE TO HOST : ON

- If this switch is ON, the MPU-401 will send MIDI EXCLUSIVE messages received from the MIDI IN port to the HOST. The messages are always preceded with the MPU-401 SYSTEM MESSAGE MARK ($FF). (See section 4.3)
- This command is used when the HOST wants to handle MIDI EXCLUSIVE messages from MIDI such as sequencer data which has been saved in another system, tone parameters sent from synthesizers etc. To send a MIDI EXCLUSIVE message, use the WANT TO SEND SYSTEM MESSAGE command ($DF).
- When the HOST uses MIDI EXCLUSIVE messages received from MIDI IN the HOST may also want to send some MIDI EXCLUSIVE messages to MIDI OUT. For this reason this command will automatically cancel the EXCLUSIVE THRU ON switch ($37).

$98 CHANNEL REFERENCE TABLE A : OFF
$99 CHANNEL REFERENCE TABLE A : ON

$9A CHANNEL REFERENCE TABLE B : OFF
$9B CHANNEL REFERENCE TABLE B : ON

$9C CHANNEL REFERENCE TABLE C : OFF
$9D CHANNEL REFERENCE TABLE C : ON

$9E CHANNEL REFERENCE TABLE D : OFF
$9F CHANNEL REFERENCE TABLE D : ON

- The CHANNEL REFERENCE TABLES are used by the MPU-401 while playing back track data. If more than one if the same MIDI key number is sent over the same MIDI channel (either from multiple tracks or from 'merged' data) the REFERENCE TABLES will automatically turn the note off and the retrigger it again. With larger polyphonic synthesizer systems connected to MIDI OUT this may not be desired, so the option of shutting the TABLES off is provided by the above commands.

5.6 READING DATA

When these commands are received, the MPU-401 will send 1 byte of data back to the HOST. The HOST MUST read the MPU-401 data byte in order to continue the activity of either.

$A0 - $A7
: Requests PLAY COUNTER value of tracks 1-8.

$AB
: Requests RECORD COUNTER then clears it.

$AC
: Requests VERSION.

  - This will return a byte indicating the VERSION number in BCD such as $15 for the version 1.5.

$AD
: Requests REVISION.

  - This will return a byte indicating the REVISION such as 1 for 'A', 2 for 'B', etc. If there is no REVISION number, the MPU-401 will return a 0.

$AF
: Requests TEMPO.

  - This will return the current TEMPO setting of the MPU-401. In MIDI SYNC or TAPE SYNC (!) modes, the tempo that the MPU-401 is being set at from the external source is given. This is useful for displaying tempo to the user while being externally controlled. The Tempo value is in Beats Per Minute.

5.7 CLEAR FUNCTIONS

$B1 Resets RELATIVE TEMPO

- Resets the RELATIVE TEMPO to the ratio 1 / 1 which is equivalent to the command $E1 $40.

$B8 CLEAR PLAY COUNTERS

- If the HOST wishes to begin playing MIDI data from the beginning of the tracks, all track PLAY counters must be cleared. This command clears PLAY counters of all tracks and the CONDUCTOR. It also clears the PLAY MAP so that the MPU-401 will not transmit any messages for the first time. Use this command before any START PLAY sequence.
- If PLAY is to begin from the point it last stopped (continue) do not use this command so the MPU-401 will remember the old PLAY counter values.
- Clearing the PLAY COUNTERS is typically done at the end of the START PLAY sequence, just before START PLAY or START OVER-DUB.

$B9 CLEAR PLAY MAP

- When this command is received, the MPU-401 will send NOTE OFF messages to MIDI OUT based on the current state of the CHANNEL REFERENCE TABLES and clear the NOTE ONs in these tables. It also clears the PLAY TABLES which hold the timing and MIDI message to be sent next by each channel.
- This command is necessary when the HOST wishes to send data to MIDI OUT by using the WANT TO SEND DATA commands ($D0 to $D7) either while PLAY is stopped or on inactive tracks. The MIDI NOTE ON and OFF messages sent by the WANT TO SEND DATA commands make the CHANNEL REFERENCE TABLES set and clear. This is similar to the effect that MIDI VOICE MESSAGES and MIDI ALL NOTES OFF from MIDI IN has setting and clearing the MIDI IN TABLE.
- This command also clears the PROGRAM CHANGE TABLE which stores all program changes while the MPU-401 is not recording.

$BA Clears the RECORD COUNTER.

- This command is not normally used since the record counter is automatically cleared when RECORD is started.

5.8 SET TIMEBASE

$C2 - $C8 Sets Timebase.

- These commands select the internal TIMEBASE of the MPU-401. Only one may be selected at a time. TIMEBASE refers to the number of clock pulses the counters will use per beat.

COMMAND :	$C2	$C3	$C4	$C5	$C6	$C7	$C8
TIMEBASE :	48	72	96	120	144	168	192

- It is usually recommended to use a TIMEBASE of at lest 96 for any real time sequencer applications to allow enough resolution of the incoming performance data.

5.9 WANT TO SEND DATA

When sent these commands, the MPU-401 expects a MIDI VOICE MESSAGE without a leading timing byte. For example, to send MIDI NOTE ON #69 on with velocity value #55 in channel 1 the HOST would send $90 $45 $37. Each message must contain the proper number bytes defined by MIDI.

$D0 - $D7 WANT TO SEND DATA

- These commands allow the HOST to transmit a single MIDI VOICE MESSAGE on a track of the MPU-401. The least significant 3 bits of the command number refer to the track number (1 to 8).

TRACK	1	2	3	4	5	6	7	8
LSBs	0	1	2	3	4	5	6	7

- WANT TO SEND DATA allows the HOST to send a MIDI VOICE MESSAGE independently of the PLAY process in the MPU-401. A HOST application program may wish to send a special MIDI MESSAGE while the MPU-401 is not playing, or send music data based on an internal timer in the host program. When the routine using the WANT TO SEND DATA commands has stopped, the CLEAR PLAY MAP command ($B9) can be used to automatically transmit MIDI NOTE OFF messages of all notes which have been turned ON during the process.
- The MIDI VOICE MESSAGE which follows the command must be in standard MIDI format. MIDI RUNNING STATUS can be used for each track. (See Section 3.)
- These commands must not be used on ACTIVE TRACKS while the MPU-401 is playing.

MESSAGE	1st BYTE	2nd BYTE	3rd BYTE
Note Off	$80-$8F	$00-$7F	$00-$7F
	$90-$9F	$00-$7F	$00
Note On	$90-$9F	$00-$7F	$01-$7F
Note After Touch	$A0-$AF	$00-$7F	$00-$7F
Control Change	$B0-$BF	$00-$7F	$00-$7F
Program Change	$C0-$CF	$00-$7F	
Ch After Touch	$D0-$DF	$00-$7F	
Pitch Wheel	$E0-$EF	$00-$7F	$00-$7F

$DF WANT TO SEND SYSTEM MESSAGE

- This command lets the HOST transmit the SYSTEM EXCLUSIVE or COMMON message which follows the command to the MIDI OUT port of the MPU-401.
- The SYSTEM EXCLUSIVE messages must be terminated with EOX ($F7) or by any other MIDI STATUS bytes.

MESSAGE	1st BYTE	2nd BYTE	3rd BYTE		END BYTE
Exclusive	$F0	$00-$7F	$00-$7F	.....	$F7
Song Position	$F2	$00-$7F	$00-$7F		
Song Select	$F3	$00-$7F			
Tune Request	$F6				

5.10 SET CONDITIONS AND VALUES

When these commands are received, the MPU-401 expects 1 byte of DATA sent from the HOST. The HOST must always send this byte.

$E0 SET TEMPO

- Sets the TEMPO of the MPU-401's internal clock in Beats Per Minute.

$E1 RELATIVE TEMPO

- This command can be used in PLAY mode to change the tempo of the MPU-401 temporarily. The default is $40 which produces a ratio of 1/1. The rate of TEMPO change is predetermined by GRADUATION ($E2).

- The range of possible tempos determined by the commands $E0 and $E1 will differ depending on the INTERNAL TIMEBASE as follows:

TIMEBASE:	48	72	96	120	144	168	192
Max Tempo	240	240	240	240	208	179	179
Min Tempo	32	16	16	8	8	8	8

$E2 GRADUATION

- Sets the rate of RELATIVE TEMPO change.

0 = IMMEDIATE 1 = SLOWEST $FF = FASTEST

$E4 MIDI/METRO

- This command sets the Number of MIDI CLOCKS ($F8) per METRONOME BEEP. For example, the value 24 will make the metronome beep once every 1/4 note.

$E6 METRO/MEAS

- This command sets the number of beats per measure, based on the number of MIDI CLOCKS/METRONOME BEEP. For example, when the metronome is set to 1/4 notes and the value 4 is sent following this command it will set a measure at 4 beats.

$E7 INT * 4 / CLOCK TO HOST

- Sets the Number of INTERNAL CLOCKS per CLOCK TO HOST. The value following this command will set the rate of CLOCK TO HOST. An $FD will be sent every N divisions of internal clock, where N equals B divided by 4, where B is following byte of the command.
- The default value is 240. This equals 60 internal clocks for every $FD sent to the HOST or twice per beat.
- The MPU MESSAGES ($FD) are sent when the CLOCK TO HOST switch is ON ($95). The $FDs can be used for any purpose such as:
  a. To control the timing of event in the HOST such as a beep or graphic metronome.
  b. To count lead-in measures before recording or playing.

$EC ACTIVE TRACKS ON / OFF.

- Bits Ø to 7 of the byte following this command set tracks 1 through 8 to PLAY when set to a 1. When the bit is reset to Ø, the track is set to OFF.

BIT	:	Ø	1	2	3	4	5	6	7
TRACK	:	1	2	3	4	5	6	7	8

- ACTIVATE TRACKS itself does not start the MPU-401 data requests, but selects the active tracks after a START PLAY is sent. The active tracks will remain for additional PLAY STARTS, but it is recommended to set ACTIVE TRACKS before each PLAY.
- While playing: the MPU-401 will request the HOST to send data of the active tracks by an MPU MESSAGE $FØ - $F7. The HOST must send data as formatted. (See Section 3.)
- The ACTIVATE TRACKS ON/OFF command will not affect the MPU-401 while playing. The new track settings will become effective after the CLEAR PLAY COUNTERS ($B8) and a new START PLAY is sent.
- The MPU-401 automatically shuts off the the track upon receipt of an MPU MARK $FC with a leading timing byte.

$ED SEND PLAY COUNTER ON / OFF

- Sets tracks whose play counter values are sent when RECORD ON is sent while PLAYING.
- Bits Ø to 7 allow tracks 1 to 8. Use of this command is similar to ACTIVE TRACKS ON / OFF command. If RECORD ON is to be sent to the MPU-401 while it is playing (punch-in) it may be important to know the current value of play counters for calculating your position in the measure.

$EE ACCEPTABLE CHANNELS 1 to 8 ON / OFF

$EF ACCEPTABLE CHANNELS 9 to 16 ON / OFF

- The byte following these two commands is a BIT-MAPPED byte which is used to turn ON or OFF any MIDI channel whose voice messages are to be accepted in RECORD mode or DATA IN STOP mode. (1 = ON, Ø = OFF)

BIT	:	Ø	1	2	3	4	5	6	7	
CHANNEL	:	1	2	3	4	5	6	7	8	(for $EE)
		9	1Ø	11	12	13	14	15	16	(for $EF)

- MIDI messages in the channels which are turned ON by these commands are not transmitted to MIDI OUT if the MIDI THRU is turned OFF by the command $88.
- ACCEPTABLE CHANNELS is the same as a MIDI channel filter and is very useful for real-time applications which must be able to reject selected MIDI channels.

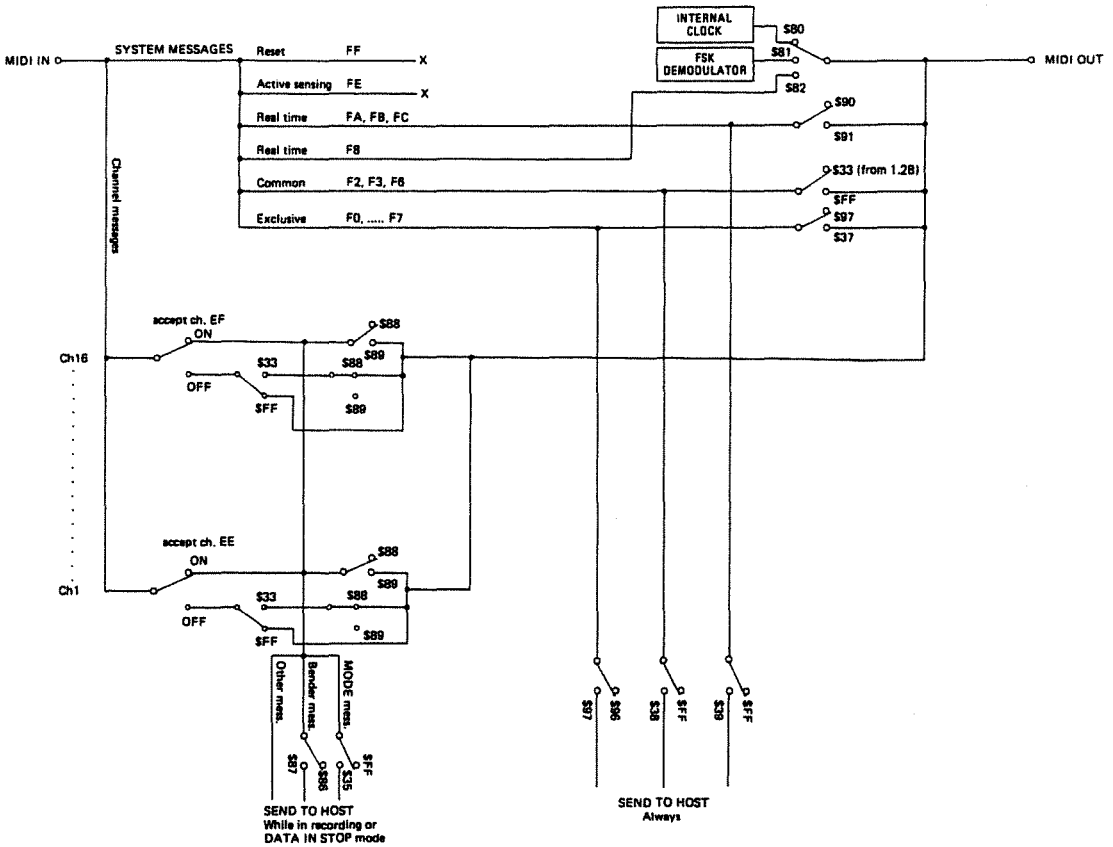
5.11 RESET

$FF RESET

- Resets the MPU-401 to its POWER-UP condition. All switches and values will be cleared to default conditions. This command is useful for returning the MPU-401 to its power-up condition. All buffers are reset by this command. The MPU-401 returns an ACK ($FE) when sent this command.

5.12 MIDI MESSAGE FLOW

The following table shows the effect that MPU commands have on the flow of MIDI through the system:



6. POWER UP DEFAULTS

The following list shows the power-up condition of all MODES, FUNCTION SWITCHES and FUNCTION VALUES of the MPU-401.

MODES

MIDI REAL TIME MESSAGE	:	STOP
PLAY	:	STOP
RECORD	:	STOP

FUNCTION SWITCHES

FUNCTION	CONDITION	EQUIVALENT COMMAND
Sync mode	internal	$80
Metronome	off	$84
Bender	off	$86
MIDI THRU	on	$89
Data in stop	off	$8A
Send measure end	on	$8D
Conductor	off	$8E
Real Time Affection	on	$91
FSK resolution	internal	$92
CLOCK TO HOST	off	$94
Exclusive to HOST	off	$96
Reference tables	on	$99,$9B,$9D,$9F

The INITIALIZATION SWITCHES ($30 to $39) only function by commands. The default of the system is opposite to their function (REAL TIME : ON, THRU : OFF (etc.).

DEFAULT VALUES

FUNCTION	VALUE	EQUIVALENT COMMAND
Time base	120	$C5
Tempo	100	$E0 $64
Relative tempo ratio	1/1	$E1 $40
Graduation	0	$E2 $00
MIDI / METRO	12	$E4 $0C
METRO / MEAS	8	$E6 $08
INT / HOST CLOCK	60	$E7 $F0
Active track number	none	$EC $00
Send play counter	none	$ED $00
Acceptable channels	all	$EE $FF
		$EF $FF
Reference Table A	ch 1	$40
Reference Table B	ch 2	$51
Reference Table C	ch 3	$62
Reference Table D	ch 4	$73

7. OTHER FUNCTIONS

7.1 METRONOME OPERATION

The MPU-401 provides an audible metronome for use by the musician during recording. The beeps reflect the number of MIDI CLOCK divisions set by the byte following the MIDI/METRO command $E4.

(!) The metronome has two modes of operation. In the first mode, set by command $85, the metronome produces three different pitches. The highest pitch is on the downbeat of each bar. The rest of the beats in the measure will alternate between two lower pitches.

(!) The second mode is selected with command $83. The metronome will now produce only two different pitches. A higher one on the downbeat of the measure and a second tone for the remainder of the measure. This will be useful for both triplet meters and odd meters where the division of accents is not easily defined.

The metronome is stopped with the command $84.

The rate of the metronome (quarter note, eighth note, triplet, sixteenth, etc.) is selected by the byte following the MIDI/METRO command ($E4). For example, since MIDI clock is always 24 per beat, if the command $E4 $0C (decimal 12) is sent, the metronome beeps every 1/8 note, or 2 times per beat (once every 12 MIDI clocks). The rate of the downbeats is selected by the byte following the METRO/MEASURE command ($E6). For example, if the meter is four beats to the measure and the metronome is set to eighth notes ($E4 $0C) then the command $E6 $08 will set the downbeat to every four beats (every 8 metronome beeps).

The METRONOME COUNTER, the MEASURE COUNTER and the CLOCK TO HOST COUNTER are initialized by either of the METRONOME : ON commands ($83 or $85) if the MPU-401 is not running. If the MPU has been started by either a MIDI START from the HOST or a MIDI START ($FA) while REAL TIME AFFECTION is ON, these counters are NOT initialized by METRONOME : ON and the metronome will still be synchronized to the beat and the bar.

(!) A METRONOME OUTPUT JACK is provided on the MPU-401 to send out an audio signal of the internal metronome beeps that can be sent to any amplifier or mixing board.

7.2 DIN SYNC

The MPU-401 provides a Roland Sync 24 output (24 pulses per quarter note, 5 volt positive pulse). This output can be used to synchronize any external devices that conform to the Roland DIN Sync 24 standard (all Roland Drum machines and sequencers in addition to many other electronic musical products).

7.3 THE CONDUCTOR

While playing, if the CONDUCTOR is set ON ($8F) the MPU-401 will send the MPU MESSAGE $F9 to the HOST to request conductor data in the same manner as the REQUEST TRACK DATA messages ($F0 - $F7). When the MPU MESSAGE CONDUCTOR DATA REQUEST ($F9) is sent, the MPU-401 expects data in the DATAPORT from the HOST as follows:

	1st byte	2nd byte	3rd byte
Commands	$00 - $EF timing byte	$00 - $C8 MPU command	
Commands *	$00 - $EF timing byte	$D0 - $DF MPU command	
Commands **	$00 - $EF timing byte	$E0 - $EF MPU command	$00 - $FF following byte
Null	$00 - $EF timing byte	$F8 Null	
Measure End	$00 - $EF timing byte	$F9 Measure End	
Data End	$00 - $EF timing byte	$FC Data End	
Over Flow	$F8 timing over flow		

NOTES:

* The data bytes for these commands have to be sent when the commands are executed (when the HOST receives next $F9).

** These commands have to be sent with a following data byte.

The conductor allows MPU-401 commands to be sent, with a leading timing byte, in synchronization with music that is being played or recorded.

If, for example, the data bytes $78 $E1 $80 were sent after a CONDUCTOR REQUEST ($F9), it would cause the MPU-401 internal tempo to double after one quarter note. $78 is the leading timing byte indicating value 120 or one quarter note. $E1 will be executed as the MPU-401 command SET RELATIVE TEMPO. $80 is the following data byte of the command which indicates the rate being double of normal (2/1). So the conductor is very useful for tempo variations within a song (accelerando, ritardando, etc). It can also be use to change the metronome accents for music with mixed time signatures.

8. SAMPLE COMMAND SEQUENCES

These are example sequences meant to illustrate the use of the command set of the MPU-401. They show only a few of the ways to use the system.

There are only four ways to access the MPU-401:

```
1>  SEND A COMMAND        Sn Cmd
2>  RECEIVE A REQUEST     Rc Req
3>  SEND DATA             Sn Dta
4>  RECEIVE DATA          Rc Dta
```

The command sequences shown will use these four elements with the abbreviations on the right. Abbreviations for data will be:

```
$tb       timing byte     (value between $00 and $EF)
$md       MIDI data       (any MIDI data or status from $00 to $EF)
```

NOTES:

1> The MPU-401 sends DATA and REQUESTS to the HOST in the same manner. They are shown as seperate functions in these examples to make their use clear. An MPU REQUEST is either a TRACK DATA REQUEST or a CONDUCTOR REQUEST. The HOST must decide if a message sent by the MPU-401 is a request or is data. See 'RECEIVING MIDI DATA FROM THE MPU-401' near the end of this section for more detail on this.

2> The '$md' symbol is used to represent a MIDI message set, not a single MIDI data byte. For example, a MIDI NOTE ON could be sent as $90 $44 $86. These three bytes would be shown as a single '$md' in this section.

RECORD A TRACK (INTERNAL SYNC)
This example will record a track with the metronome on.

```
Sn Cmd    $85       ;  turn on the metronome
Sn Cmd    $22       ;  RECORD START MIDI START - FSK (!) and DIN ON
                    ;  this command will clear the record counter,
                    ;  clear the metronome counter, clear the beat
                    ;  counter, send an $FA to MIDI OUT and start
                    ;  recording.

Rc Dta    $tb       ;  the MPU-401 will now begin sending data
Rc Dta    $md       ;  with timing bytes to the HOST. They
  :                 ;  should be stored to memory.
  :
  :

Sn Cmd    $11       ;  RECORD STOP MIDI STOP - FSK (!) and DIN OFF
                    ;  this will stop the recording, send an $FC to
                    ;  MIDI OUT and make the MPU-401 send a final
                    ;  timing byte and DATA END MARK to the HOST.

Sn Cmd    $84       ;  turn the metronome off

Rc Dta    $tb       ;  final timing byte
Rc Dta    $FC       ;  MPU MARK   'DATA END'
```

PLAY TRACKS (INTERNAL SYNC)

A bit-mapped byte is created with bits for tracks to play set to 1. For this example we will play back tracks 1,2, and 4 (00001011 or $0B).

```
Sn Cmd    $EC        ;  ACTIVATE TRACKS
Sn Dta    $0B        ;  tracks 1,2, and 4
SnCmd     $B8        ; CLEAR PLAY COUNTERS

Sn Cmd    $0A        ;  START PLAY  MIDI START - FSK (!) and DIN ON

Rc Req    $F0        ;  TRACK DATA REQUEST for track 1
Sn Dta    $tb        ;  send first timing byte
Sn Dta    $md        ;  and MIDI data

Rc Req    $F1        ; TRACK DATA REQUEST for track 2
Sn Dta    $tb        ; send timing and data
Sn Dta    $md
  :
  :
Rc Req    $F3        ; TRACK DATA REQUEST for track 4
Sn Dta    $tb        ; send timing and data for track 4
Sn Dta    $md
  :
  :                  ;  continue sending data as requested
  :
Sn Dta    $tb        ;  send final timing byte of a track
SnDta     $FC        ; and the DATA END mark.  If this is the last
                     ;  track the MPU-401 will send an $FC to the HOST.

RcDta     $FC        ; MPU-401 sends an  ALL END MESSAGE when the HOST
                     ;  has sent it's last DATA END mark

Sn Cmd    $05        ;  STOP PLAY  MIDI STOP - FSK (!) and DIN OFF
                     ; the MPU-401  will stop sending TRACK DATA
                     ; REQUESTS, will send an $FC to MIDI OUT, and will
                     ;  save the metronome and beat  counters for a
                     ;  CONTINUE PLAY.
```

NOTES:

1> If playing from the beginning, CLEAR PLAY COUNTERS must ALWAYS be done before a START PLAY. The command ($B8) should be sent after ACTIVATE TRACKS and before the START PLAY as shown above.

2> The MPU-401 will generate a MIDI ALL NOTES OFF message when all the notes of a channel are shut off except after sending ALL NOTES OFF : OFF ($30) command.

3> When STOP PLAY is sent, any notes still ON in the CHANNEL REFERENCE TABLES will be sent as NOTE OFFs to MIDI OUT.

4> The HOST may send MIDI RUNNING STATUS to the MPU-401. When the MPU generates an ALL NOTES OFF ($B0) and the next byte from the HOST is RUNNING STATUS, the MPU-401 will automatically generate the correct status byte.

5> The MPU-401 will send the ALL END MESSAGE ($FC) when all tracks have come to an end. If the HOST sent a STOP PLAY ($05) before the MPU-401 ends playing all tracks this time the MPU-401 would not send this message.

```
OVERDUB (INTERNAL SYNC)
     This example will record a track while playing another track back.

Sn Cmd     $EC         ; ACTIVATE TRACKS
Sn Dta     $01         ; track 1

Sn Cmd     $B8         ; clear the PLAY COUNTERS

Sn Cmd     $2A         ; START RECORD  START PLAY  MIDI START - FSK (!) and
                       ; DIN ON

;  The  MPU-401 will now send both TRACK DATA REQUESTS along with  data.
;  Upon receiving an interrupt,  the HOST should look at the DATAPORT of
;  the  MPU-401.   If  the incoming message is from $F0 to $F7 it  is  a
;  TRACK DATA REQUEST, otherwise it will be data from the MIDI IN of the
;  MPU-401.   The HOST's interrupt routine should recognize this and  go
;  to the proper routine to either send or receive data.
;  Playback could stop before end of record.   There is no need to  send
;  any STOP until it is time to stop recording.

Sn Cmd     $15         ;  STOP RECORD  STOP PLAY  MIDI STOP FSK (!) and DIN
                       ;  OFF

Rc Dta     $tb         ; get final timing byte from recording
Rc Dta     $FC         ; get MPU DATA END MARK
```

RECORD WITH LEAD-IN (INTERNAL SYNC)

Most often there should be a one or two measure lead-in with metronome to prepare the musician for recording. During this time it is a good idea to store any MIDI data that may come through. The first notes played may be just ahead of the down beat of the first bar and would be lost if the HOST is not recording. For this possibility there is the DATA IN STOP mode. The WITH TIMING BYTE command ($34) will generate a leading timing byte of $00 for all data recorded during the lead-in. CLOCK TO HOST is used to count the length of the lead-in measures, and will be shut off during record in this example.

```
Sn Cmd    $34          ; WITH TIMING BYTES to make lead-in data compatible
                       ; with recorded data
Sn Cmd    $95          ; CLOCK TO HOST : ON to count lead-in
Rc Dta    $FD          ; wait for CLOCK mark before continuing. This one
                       ; is not counted in HOST's lead in counter
Sn Cmd    $85          ; METRONOME :  ON (clears the metronome  counter,
                       ; measure counter and CLOCK TO HOST counter)
Sn Cmd    $8B          ; DATA IN STOP : ON for any data during lead-in
  :
  :
Rc Dta    $FD          ;  CLOCK TO HOST (count down)
  :
  :
Rc Dta    $tb ($00)    ;  store MIDI DATA.   Timing bytes are always $00
                       ;  in DATA IN STOP MODE
Rc Dta    $md
  :
  :
Rc Dta    $FD          ; CLOCK TO HOST (count down)
  :
  :
Rc Dta    $FD          ; if final CLOCK TO HOST - start record

Sn Cmd    $22          ; START RECORD  MIDI START - FSK (!) and DIN ON
Sn Cmd    $94          ; CLOCK TO HOST : OFF
Sn Cmd    $8A          ; DATA IN STOP : OFF
  :
  :
Rc Dta    $tb          ; record incoming MIDI data
Rc Dta    $md
  :
  :
Sn Cmd    $11          ; STOP RECORD  MIDI STOP - FSK (!) and DIN OFF
Rc Dta    $tb          ; final timing and DATA END MARK
Rc Dta    $FC
Sn Cmd    $84          ; METRONOME : OFF
```

NOTES:

1> It is also possible to use the CLOCK TO HOST $FDs while recording or playing at all times to control a visual metronome or other functions in the HOST.

PLAY A TRACK WITH CONDUCTOR : ON (INTERNAL SYNC)

The CONDUCTOR is used to as an event manager during RECORD and PLAY. In this example the CONDUCTOR is used to send a SET TEMPO command to the MPU-401 in the middle of a measure. The command and following data are sent as DATA with a leading timing byte through the DATAPORT when the CONDUCTOR REQUEST is received.

```
Sn Cmd    $EC       ; ACTIVATE TRACKS
Sn Dta    $01       ; track one is set
Sn Cmd    $8F       ; CONDUCTOR : ON  (always before $B8 command)
Sn Cmd    $B8       ; CLEAR PLAY COUNTERS
Sn Cmd    $0A       ; START PLAY  MIDI START

Rc Req    $F0       ; TRACK DATA REQUEST
Sn Dta    $tb       ; send timing byte
Sn Dta    $md       ; and MIDI data
  :
  :
Rc Req    $F9       ; CONDUCTOR REQUEST
Sn Dta    $tb       ; send timing
Sn Dta    $E0       ; send SET TEMPO to the DATAPORT
Sn Dta    $3F       ; send tempo data of 63 to DATAPORT
  :
  :
Rc Req    $F9       ; when CONDUCTOR times out it will execute the
                    ; command and send the next REQUEST
  :
  :
  :
```

NOTES:

1> The CONDUCTOR can use WANT TO SEND DATA (commands $D0 to $D7) to send MIDI data to the MIDI OUT port of the MPU-401. It should only send data on non ACTIVE tracks if the MPU-401 is in PLAY mode. When using WANT TO SEND DATA the following MIDI DATA bytes are sent when the CONDUCTOR times out after the command and sends another $F9. The MIDI DATA is sent along with the next timing byte (or overflow) and the next command.

2> The CONDUCTOR : ON command must always be sent before the CLEAR PLAY COUNTERS command ($B8).

3> See section 7.3 for more information on the CONDUCTOR.

MIDI SYNC MODE

The MPU-401 can also be set up as a MIDI slave unit to other MIDI clocks. In MIDI SYNC mode, the MPU-401 can be made to record or play back at the request of external devices such as drum machines or other sequencers. Care must be taken so that external control will not 'grab' the MPU-401 from the HOST. REAL TIME AFFECTION will help avoid this problem. It is best to keep REAL TIME AFFECTION : OFF until the beginning of the record routine. Just before beginning it can be set ON and disabled again when record is finished.

RECORD A TRACK (MIDI SYNC)

This example will record a track when commanded from an external MIDI device.

```
Sn Cmd     $82        ; MIDI SYNC MODE
Sn Cmd     $EC        ; ACTIVATE TRACKS
Sn Dta     $00          ;All tracks off. (REAL TIME AFFECTION : ON will
                        ; automatically play any active tracks when
                        ; MIDI START or CONTINUE is received. This will
                        ; prevent this from happening unwantedly.
Sn Cmd     $91        ; REAL TIME AFFECTION : ON
Sn Cmd     $20        ; RECORD STAND-BY

;     The MPU-401 now waits for a MIDI REAL TIME byte $FA (start) or $FB
;     (continue) to come in from the MIDI IN port.  At that time it will
;     start the record timer,  set FSK (!)  and DIN ON and begin sending
;     data to the HOST.

Rc Dta     $tb        ; MIDI data to be recorded
Rc Dta     $md
  :
  :
  :
;     The MPU-401 will continue to send data to the host until a MIDI
;     REAL TIME byte $FC (STOP) is received from the MIDI IN port or
;     until the HOST sends a STOP RECORD.  In this example an $FC is
;     received by the MPU-401 to the MIDI IN port. It is not sent to the
;     host unless the HOST has set RT TO HOST : ON ($39) upon
;     initialization.

Rc Dta     $tb        ;  final timing byte and DATA END MARK.
Rc Dta     $FC
Sn Cmd     $11        ; STOP RECORD  MIDI STOP - FSK (!) and DIN OFF
Sn Cmd     $90        ; REAL TIME AFFECTION OFF
```

NOTES:
1> There are several variations for recording in MIDI SYNC MODE. The example above allows the MPU-401 to begin recording automatically by use of the REAL TIME AFFECTION : ON command. By using RT TO HOST : ON ($39) the incoming MIDI REAL TIME bytes may be sent to the HOST. Each REAL TIME byte is preceded by the MPU SYSTEM MESSAGE $FF. The MPU-401 may be

operated by the HOST in MIDI SYNC mode by not using REAL TIME AFFECTION : ON and reading incoming MIDI REAL TIME START, CONTINUE and STOP. This technique is not as recommendable as using REAL TIME AFFECTION : ON since there is a slight chance that a MIDI TIMING CLOCK ($F8) could be missed putting the system slightly out of sync.

2> Placing the MPU-401 into MIDI SYNC MODE means the internal counters will all be synchronized to the incoming MIDI TIMING CLOCK tempo.

3> It is possible to turn REAL TIME AFFECTION : OFF while recording. This is useful if the application wants the MPU-401 to start automatically from an incoming MIDI START ($FA) or MIDI CONTINUE ($FB), but not be stopped by MIDI STOP ($FC). For example, a MIDI STOP could possibly come in the middle of a measure, but the HOST will want the final measure to be complete. This can be accomplished by turning REAL TIME AFFECTION : OFF right after the MIDI START ($FA) has been received (REAL TIME TO HOST must be turned on when the system is initialized). When the MIDI STOP ($FC) is received, the host can continue recording until the end of the measure and then perform a stop record routine.

PLAY TRACKS (MIDI SYNC)

In this example tracks 1 and 3 will be played back in by MIDI SYNC by using REAL TIME AFFECTION : ON.

```
Sn Cmd    $82        ;  MIDI SYNC MODE
Sn Cmd    $EC        ;  ACTIVATE TRACKS
Sn Dta    $05        ;  tracks 1 and 3
Sn Cmd    $91        ;  REAL TIME AFFECTION: ON

;    The MPU-401 will now wait for an incoming MIDI REAL TIME START
;   ($FA) or CONTINUE ($FB) and begin playback automatically.  If $FA
;    is received,  the  PLAY  COUNTERS are cleared automatically by the
;    MPU-401.

Rc Req    $F0        ;  TRACK DATA REQUEST for track 1
Sn Dta    $tb        ;  send first timing byte
Sn Dta    $md        ;  and MIDI data
  :
  :
Rc Req    $F2        ; TRACK DATA REQUEST for track 3
Sn Dta    $tb        ; send timing and data for track 3
Sn Dta    $md
  :
  :
  :                  ;  continue sending data as requested
  :
Sn Dta    $tb        ;  send final timing byte of a track
SnDta    $FC         ; and the DATA END mark.  If this is the last
                     ;  track the MPU-401 will send an $FC to the HOST.

RcDta     $FC        ; MPU-401 sends a ALL END MESSAGE when last track
                     ;  has send it's DATA END MARK

;    If all tracks are finished before the MPU-401 receives a MIDI STOP
;    ($FC), the MPU-401 will send an ALL END MESSAGE ($FC).  The ALL END
;    MESSAGE should not get confused with the MIDI MESSAGE since both
;    are $FC.  The MIDI STOP will only be sent to the HOST with the REAL
;    TIME TO HOST : ON command ($39) and then only with a preceding $FF
;    (SYSTEM MESSAGE) byte.  If all tracks are finished the HOST may
;    either do nothing, may reset address pointers to the top of the
;    tracks again while waiting for a new MIDI START, or may choose to
;    send a STOP PLAY command.  In this case we do not wish to send a
;    MIDI STOP to MIDI OUT of the MPU-401 since there is still another
;    sequencer working.

Sn Cmd    $04        ; STOP PLAY (very important)
Sn Cmd    $90        ; REAL TIME AFFECTION : OFF
```

NOTES:

1> When REAL TIME AFFECTION is set ON, incoming $FAs will automatically clear the PLAY COUNTERS.
2> It is important to send the STOP PLAY command at the end of a MIDI sync sequence.

OVERDUB (MIDI SYNC)

This example will record a track while playing another track back in MIDI SYNC mode.

```
Sn Cmd    $82       ; MIDI SYNC MODE
Sn Cmd    $EC       ; ACTIVATE TRACKS
Sn Dta    $01       ; track 1
Sn Cmd    $91       ; REAL TIME AFFECTION : ON

Sn Cmd    $20       ; STAND-BY RECORD

;   The MPU-401 will wait for an incoming MIDI START ($FA) or MIDI
;   CONTINUE ($FB) before beginning the record/play process. FSK (!) and
;   DIN will go ON.
;   The MPU-401 will now send both TRACK DATA REQUESTS along with data.
;   Upon receiving an interrupt, the HOST should look at the DATAPORT of
;   the MPU-401. If the incoming message is between $F0 and $F7 it is a
;   TRACK DATA REQUEST, otherwise it will be data from the MPU-401. The
;   HOST's interrupt routine should recognize this and go to the proper
;   routine to either send or receive data.      Upon receiving a MIDI
;   STOP ($FC) the MPU-401 will send a final timing byte an DATA END
;   MARK ($FC).  In REAL TIME AFFECTION : ON the HOST can either wait
;   for a MIDI CONTINUE ($FB) or may choose to stop the overdub.

Sn Cmd    $15       ; STOP RECORD  STOP PLAY  MIDI STOP

Rc Dta    $tb       ; get final timing byte from recording
Rc Dta    $FC       ; get DATA END mark

Sn Cmd    $90       ;  REAL TIME AFFECTION :  OFF  in the event that any
                    ;  more MIDI STARTS come when the HOST does not wish
                    ;  to PLAY or RECORD.
```

USE OF FSK MODE (!)

When a START RECORD or START PLAY with MIDI START is sent to the MPU-401 it will automatically begin to modulate the FSK (Frequency Shifted Key) pilot tone through the TAPE OUT jack. This modulated tone is for recording onto the spare track of a multitrack tape recorder. The tape is rewound and the recorded tone is played back into the MPU-401's TAPE IN jack. By putting the MPU-401 into FSK MODE the internal counters are tied to the FSK decoder. Recording, playing back and overdubbing are done exactly as they are in internal mode, but with the addition of the FSK SYNC MODE command ($81) before the START.

In FSK MODE the MPU-401 will not put out MIDI REAL TIME CLOCK bytes ($F8) until it receives the modulated FSK tone. MIDI START ($FA), CONTINUE ($FB), and STOP ($FC) are sent out when the HOST sends the START and STOP commands to the MPU-401. The HOST should not send a START to the MPU-401 until the pilot tone of the FSK is being read back into the TAPE IN jack.

When the FSK RESOLUTION is set to internal ($92) there will not be any FSK signals put out through the TAPE OUT jack while the MPU-401 is in FSK mode. This should have no effect on an application since a tape sync signal is created while in internal sync mode and is of no use while synchronizing.

RECEIVING MIDI DATA FROM THE MPU-401

While in DATA IN STOP or RECORD modes, the MPU-401 will be sending timing bytes, MIDI data, MPU MESSAGES and MPU MARKS to the HOST. When the HOST is first interrupted, the registers of the computer are saved to the stack and the DATAPORT of the MPU-401 should be read. What the HOST receives will either be DATA or an MPU MESSAGE. The following procedure is a logical example for interrupt handling routine.

```
          1. Read STATPORT.
                if DSR = 1 interrupt is not from MPU-401. Go to
                                original interrupt handler.
          2. save registers on stack.
          3. read DATAPORT.
intl:     4. check data.
               a. if data < $F0,          time value followed by MIDI DATA
               b. if $F0 <= data <= $F7 TRACK DATA REQUEST
               c. if data = $F8,          TIMING OVERFLOW
               d. if data = $F9,          CONDUCTOR REQUEST
               e. if data = $FD,          CLOCK TO HOST
               f. if data = $FC,          ALL END
               g. if data = $FF,          MIDI system message
          5. execute appropriate routine based on results of step 3.
          6. restore registers from stack.
          7. return from interrupt.
```

possible actions for step 5:

a. save the time value and MIDI data being input. Read again by using 'get data' routine determined by the MPU format and save system.
   - The following procedure may assist in accepting the data without running into a 'loop problem' with the MPU-401.

   a-1. Store the leading TIMING BYTE
   a-2. Get the next byte of from the DATAPORT
   a-3. If the data is a MIDI STATUS BYTE ($80 to $EF):
        i> Use the upper four bits as an offset into a table to decide how many bytes to accept. Each MIDI STATUS will have a set number of bytes which follow.
        ii> Store the number from this table.
        iii> Use this number to loop through a 'get data' routine the proper number of times to get and store the entire MIDI packet.

   a-4. If it is MIDI RUNNING STATUS ($00 to $7F):
        i> Use the number derived from the above routine, less 1 for no STATUS byte.
        ii> Proceed as 'iii' above.

a-5. If it is an MPU MARK ($F9, $FC):
   i> An $F9 (Measure End) should be stored for playback like any other MIDI data. It has a leading timing value and is a single byte.
   ii> If it is an MPU DATA END MARK $FC then go to a record stop routine.

b. Go to play routine and play from track indicated by 3 LSBs of the message. i.e. $F0= play request track 1, $F1= play request track 2, $F2= play request track 3, etc.. Send N bytes by using 'put data' routine, where N is determined by the MPU format.
   - The following procedure can be used when sending play data.
   b-1. Get and send TIMING BYTE from memory and look at next byte.
   b-2. Use the method from RECEIVING MIDI DATA FROM THE MPU-401 to determine the number of bytes to send to the DATAPORT of the MPU-401 based on RUNNING STATUS of the MIDI data being sent.
   b-3. It will be necessary to keep a data size value for each track.

c. save the timing overflow data being input.
d. send conductor data.
e. use clock to host for COUNT IN before RECORD or other purposes.
f. all tracks are finished playing, do actions required.
g. read N times required by the MIDI format.
   - MIDI SYSTEM MESSAGES have no timing byte and are typically not stored in memory. When the HOST receives an $FF the next byte (or bytes) will be the system message. (See section 4.3).

The 'int1' is a label for fixed_entry from program A and program D of next section.

NOTES:
1> MPU MESSAGE $F8 (OVERFLOW) should not be confused with timing bytes. It is never used to precede MIDI data.
2> The MPU MARK $F9 (MEASURE END) is enabled from command $8D and disabled with command $8C. When used, they should be stored and sent back like any MIDI data.
3> It is advisable to not send commands to the MPU from within your interrupt routine itself. The potential for errors from crossed ACKs and data is small but possible. If an interrupt such as a CLOCK TO HOST which may signal a START RECORD is received it may be safer to set a flag, return from the interrupt and then execute the command.
4> This routine assumes that data has been stored into memory in a 'raw' fashion as it was generated by the MPU-401. In fact, there are numerous methods for storing and recalling musical data.

9. PROGRAMMING SUBROUTINES FOR THE MPU-401

In the examples to follow:

DSR = bit 7 of MPU status byte (Data Set Ready, active low).
DRR = bit 6 of MPU status byte (Data Receive Ready, active low).

The MPU status byte is read from STATPORT.

All MPU commands are sent to COMPORT.

All MPU data bytes are sent to or read from DATAPORT.

*** IMPORTANT ***

After the reset, DATAPORT of the MPU-401 must be read to clear the DSR, before enabling interrupts.

A. Sending a command to the MPU-401.

This routine can be used when the HOST wants to send a command to the MPU-401.
To send a command:

1. test DRR (bit 6) of STATPORT.
2. if bit 6 = 1, then goto step 1.
3. disable interrupts.
4. send command to COMPORT of the MPU-401.
5. test DSR (bit 7) of STATPORT.
6. if bit 7 = 1, then goto step 5.
7. read DATAPORT.
8. if data = acknowledge ($FE), then step 9.
   else no acknowledge. so:
   a. call interrupt service routine.
      - using software interrupt (machine dependent).
   b. goto step 5.
9. enable interrupts.
10. return to caller.

6502 EXAMPLE

```
DATAPORT:       equ     $C080+n0        ; n=slot #
COMPORT:        equ     $C081+n0        ; n=slot #
STATPORT:       equ     $C081+n0        ; n=slot #
DSR:            equ     $80
DRR:            equ     $40

TEMP            ds      1
combfr          ds      1
ack             equ     $FE
intl            equ     fixed_entry     ; in the section 8

  ; send command thru combfr

Sn.Cmd:         lda     STATPORT        ; get status
                and     #DRR            ; test bit6
                bne     Sn.Cmd          ; if bit 6=1 then
                                        ; keep trying
                sei                     ; mask interrupts
                lda     combfr          ; get command
                sta     COMPORT         ; send to the MPU-401
Sn.Cmd2:        lda     STATPORT        ; get status
                bmi     Sn.Cmd2         ; if bit 7=1 then keep
                                        ; checking
                lda     DATAPORT        ; get data from MPU
                cmp     #ack            ; is it an acknowledge?
                bne     Sn.Cmd3         ; take if not ACK.
                cli
                rts

; If no ACK ($FE) is received we must generate a software interrupt
; by simulating the result of an interrupt on the stack

Sn.Cmd3:
                sta     temp            ; save accumulator
                lda     #>Sn.Cmd2       ; return to sendl.1
                                        ; when return from
                                        ; interrupt service rtn.
                pha                     ; by pushing address of
                lda     #<Sn.Cmd2       ; return on stack
                pha
                lda     #$04            ; push processor status
                pha                     ; with interrupt disabled

                     txa                ; this is the same as
                     pha                ; interrupt
                     tya
                     pha
                     lda  temp
                     pha                ; adjust stack level
                jmp     intl            ; jump to fixed entry
                                        ; interrupt service
                                        ; routine
```

B. Sending data to the MPU-401.

This program can be used to send one data byte which is:

1. following to the commands ($E0 - $EF).
2. the part of data following to the commands ($D0 - $DF).
3. the part of a message which is requested by MPU messages such as 'REQUEST NEXT TRACK DATA' or 'REQUEST CONDUCTOR DATA'.

To send data:

1. test DRR (bit 6) of STATPORT.
2. if bit 6 = 1, then return to step 1.
3. else load the data byte to send.
4. send it to DATAPORT.
5. exit routine.

6502 EXAMPLE
============

```
; data is in accumulator

Sn.Dta:     tax                     ; save accumulator
Sn.Dta2:    lda     STATPORT        ; read status
            and     #DRR            ; test bit 6
            bne     Sn.Dta2         ; if bit 6=1
                                    ; then not ready
            txa                     ; restore accumulator
            sta     DATAPORT        ; send to the MPU-401
            rts                     ; return to caller
```

C. Reading data from the MPU-401.

This program is used in the program D and E.
To read data:

1. test DSR (bit 7) of STATPORT.
2. if bit 7 = 1, then return to step 1.
   else continue.
3. read data from DATAPORT.
4. exit routine.

6502 EXAMPLE
============

```
Rc.Dta:     lda     STATPORT        ; read status,
                                    ; bit 7 = minus flag
            bmi     Rc.Dta          ; if minus flag = 1,
                                    ; then not ready
            lda     DATAPORT        ; else get data
            rts                     ; return to caller
```

D. Sending a command to the MPU-401 that returns a single byte response.

This program can only be used for sending a command ($A0 - $AF) and reading a byte.
To send the command and read the data:

1. test DRR (bit 6) of STATPORT.
2. if bit 6 = 1 then goto step 1.
3. disable interrupts.
4. send command to COMPORT of the MPU-401.
5. test DSR (bit 7) of STATPORT.
6. if bit 7 = 1 then goto step 5.
7. read DATAPORT.
8. if data = acknowledge ($FE) then step 9.
   else no acknowledge, so:
   a. call interrupt service routine
      - using software interrupt (machine dependent).
   b. goto step 5.
9. call Rc.Dta subroutine.
10. enable interrupts.
11. return to caller.

6502 EXAMPLE
============

```
  ; send command thru combfr

Sn.Cmd:         lda       STATPORT       ; get status
                and       #DRR           ; test bit6
                bne       Sn.Cmd         ; if bit 6=1 then
                                         ; keep trying
                sei                      ; mask interrupts
                lda       combfr         ; get command
                sta       COMPORT        ; send to the MPU-401
Sn.Cmd2:        lda       STATPORT       ; get status
                bmi       Sn.Cmd2        ; if bit 7=1 then keep
                                         ; checking
                lda       DATAPORT       ; get data from MPU
                cmp       #ack           ; is it an acknowledge?
                bne       Sn.Cmd3        ; take if not ACK.
                jsr       Rc.Dta         ; get info.
                cli
                rts

; If no ACK ($FE) is received we must generate a software interrupt
; by simulating the result of an interrupt on the stack

Sn.Cmd3:
                sta       temp           ; save accumulator
                lda       #>Sn.Cmd2      ; return to send1.1
                                         ; when return from
                                         ; interrupt service rtn.
                pha                      ; by pushing address of
                lda       #<Sn.Cmd2      ; return on stack
                pha
```

```
lda       #$Ø4          ; push processor status
pha                     ; with interrupt disabled

     txa                ; this is the same as
     pha                ; interrupt
     tya
     pha
     lda  temp
     pha                ; adjust stack level
jmp       intl          ; jump to fixed entry
                        ; interrupt service
                        ; routine
```

10. ROM VERSIONS DELTA GUIDE

The version 1.5A ROM reflects numerous improvements and refinements to the concept and architecture of the MPU-401. Some changes are based on function problems, and others on reconsideration of process. This guide can be used by those people needing to create software for MPU units done previous to the version 1.5A release (5/31/85).

A. FOR VERSION NUMBERS BELOW 1.2B

1. - Can not receive SONG POSITION POINTER or SONG SELECT.
2. - No VERSION REQUEST or REVISION REQUEST commands.

B. FOR VERSION 1.2B AND BELOW

3. - SYSTEM COMMON messages either received from MIDI IN of the MPU or sent by the HOST during play would cause problems if the next MIDI message played was RUNNING STATUS.
4. - EXCLUSIVE TO HOST : ON ($97) should not be used.

C. FOR VERSION 1.3

5. - Add ALL NOTES OFF : OFF command ($30). (See section 5.2)
6. - Add COMMON TO HOST : ON command ($38) and REAL TIME TO HOST : ON command ($39). (See section 5.2). On previous version MIDI SYSTEM COMMON and MIDI REAL TIME messages received to the MPU-401 are always sent to the HOST.
7. - If a MIDI SYSTEM COMMON message is sent by using the WANT TO SEND SYSTEM COMMAND ($DF) and the same MIDI message is received to the MPU at the same time (because of MIDI THRU on an external device), the incoming MIDI message will be ignored.
8. - When EXCLUSIVE THRU ON is set, the MPU can not send any data while SYSTEM EXCLUSIVE data is passing to MIDI OUT. However, because early YAMAHA DX-7s send $F0 $43 (not followed by EOX) as an ACTIVE SENSING byte the MPU version 1.3 and 1.3A looks for this sequence and will continue to send MIDI data to the MIDI OUT port after 20 ms if no other EXCLUSIVE data is received.
9. - If the HOST sends more than seven consecutive MIDI RUNNING STATUS bytes, the MPU will generate the STATUS to MIDI OUT to prevent any possible device error by the receiver.
10. - The MPU will ignore $F7 (MIDI END OF EXCLUSIVE) if it is not paired with an $F0 (MIDI EXCLUSIVE).
11. - If REAL TIME AFFECTION is set ON ($91) and COMMON TO HOST is set ($38) the MPU will automatically clear the PLAY COUNTERS when it receives a MIDI SONG POSITION POINTER or SONG SELECT.
12. - The size of the MIDI IN BUFFER of the MPU while in UART MODE ($3F) is enlarged to approximately 1,700 bytes from 85 bytes.
13. - SET TEMPO ($E0) and RELATIVE TEMPO ($E1) are ignored if the MPU is not in INTERNAL SYNC MODE.

14. - When no clock is received from TAPE IN in FSK SYNC mode, requests TEMPO should not be used.

D. FOR VERSION 1.4

15. - A correction to timing in MIDI SYNC MODE is made. Previous versions would ignore one incoming MIDI CLOCK ($F8) if MIDI START ($FA) was too close to the clock. It was also possible in earlier versions to respond slowly to MIDI STOP ($FC) if it was too close to the clock byte.
16. - If the HOST has set EXCLUSIVE TO HOST : ON ($97) but is reading the DATAPORT so slowly that the MIDI IN BUFFER starts to overflow, the MPU will immediately generate an $F7 (EOX).
17. - A problem that occured only in this version caused all FOLLOWING DATA bytes of $00 to be changed to $01 for commands $E0 to $EF.
18. - When all tracks are finished playing and the MPU sends the MPU MESSAGE $FC (ALL END) the HOST MUST send a stop command to the MPU. (See section 4.1).

E. FOR VERSION 1.4A

19. - In all versions of the MPU, the METRONOME COUNTER and MEASURE COUNTER are saved to special background buffers if the MPU is in MIDI SYNC MODE. If a MIDI CONTINUE ($FB) is received from MIDI IN or the HOST sends a MIDI CONTINUE these buffers are recalled so sequencing timing may be continued. These buffers are now automatically cleared when the MPU receives MIDI SONG POSITION POINTER ($F2) or SONG SELECT ($F3). These buffers are also cleared when the HOST sends the CLEAR PLAY COUNTERS command ($B8).
20. - A correction is made to TEMPO REQUEST ($AF) when the FSK RESOLUTION was set to INTERNAL ($92) and the MPU was in FSK SYNC MODE. Previous versions returned the internal clock tempo.

F. FOR VERSION 1.4B

21. - A correction is made for record mode in MIDI SYNC or FSK SYNC modes. If START RECORDING is sent to the MPU and no clock is received before sending a STOP RECORDING the MPU will send a DATA END MARK ($FC) with leading timing byte of $00 to the HOST. On versions 1.4 and 1.4A the MPU sent nothing, and would send the MARK after switching back to INTERNAL SYNC mode.

G. FOR VERSION 1.4 TO 1.5

22. - When in INTERNAL SYNC MODE and REAL TIME AFFECTION is set ON, stand-by recording mode should not be used.

H. FOR VERSION 1.5

23. - MIDI REAL TIME BYTES $FA, $FB, and $FC are only allowed to pass to MIDI OUT from MIDI IN of the MPU while REAL TIME AFFECTION is set ON.
24. - If REAL TIME AFFECTION is set OFF, incoming MIDI REAL TIME messages ($FA, $FB, $FC) will not effect to SYNC OUT and FSK OUT.
25. - A correction is made for a problem that affected WANT TO SEND DATA ($DØ to $D7). In previous versions the first request made after STOP RECORDING or STOP PLAYING was occasionally ignored.
26. - A correction is made to a problem affecting STOP RECORDING. In previous versions if STOP RECORDING was sent directly at the measure end, the MPU MARK $F9 (MEASURE END) would not be sent.

I. FOR VERSION 1.5A

27. - RUNNING STATUS (from MPU-4Ø1 to HOST) is cleared when START RECORDING and STOP RECORDING. On versions 1.5 and below RUNNING STATUS is cleared only when START RECORDING.

DELTA GUIDE TABLE

		1.2 A	1.2 B	1.3	1.3 A	1.4	1.4 A	1.4 B	1.5	1.5 A
1	SPP or SS	X	received							
2	Request Ver and Request Rev	X	O							
3	Running status problem after SYSTEM COMMON	X		O						
4 16	EXCLUSIVE to HOST-ON	do not use		problematic		O				
5	ALL NOTES OFF-OFF	X		O						
6	SYSTEM mess to HOST	always		when COMMON to HOST-ON, RT to HOST-ON						
7	same COMMON	recognized		ignored						
8	20ms, after $F0, $43	waiting		escape		waiting				
9	generate MIDI status	X		O						
10	EOX not paired with EXCLUSIVE	recognized		ignored						
11	clear PLAY COUNTER	X		when received SPP or SS						
12	MIDI IN buffer in UART mode	85 Bytes		1700 Bytes						
13	SET TEMPO, REL TEMPO when not in INT. SYNC mode	recognized		ignored						
14	request TEMPO	do not use		O						
15	$F8 close to $FA, $FB, $FC	problematic				OK				
17	MPU com. ($E0 to $EF) when data in 0	O				changed	O			
18	STOP com. after ALL END	not necessary				must send				
19	background METRO and MEAS COUNTER when clear PC	not cleared					clared			
20	TEMPO request when FSK SYNC mode	problematic					OK			
21	DATA END MARK when no clock is coming	O				nothing		O		
22	STAND-BY RECORDING in INT. SYNC, RT aff-ON	O				do not use				O
23 24	THRU $FA, $FB, $FC MIDI STOP while RT aff-OFF	always								only RT aff-ON
25	first WSD after play	problematic								OK
26	final MEASURE END MARK	not sent								sent
27	clear running status to HOST	when REC. START								REC. START /STOP

Reference Glossary

page

Acceptable Channel : 36
While in RECORD mode or DATA IN STOP mode, MIDI messages in the acceptable channels received from MIDI IN are sent to the HOST. At power up all channels are acceptable. The MPU commands $EE and $EF with a following data byte for each can be used for setting each channel acceptable or not.

ACK :
Acknowledge.

Acknowledge : 13
An MPU MESSAGE ($FE) sent to the HOST when an MPU command has been received and is going to be executed.

Active Track : 36
Any track which will send REQUEST TRACK DATA when START PLAY is sent.

After Touch : 33
Some keyboards support detection of keyboard pressure after keys have been struck. These values are transmitted by MIDI VOICE messages. (See MIDI message.)

ALL NOTES OFF : 21
A MIDI MODE message which can be used to turn OFF all notes in the receiver which have been turned ON by MIDI IN.

Baud :
Rate of serial communications. Bits per second.

Bender : 24
Lever or wheel on keyboards which changes the pitch of any notes being played. These values can be transmitted by MIDI VOICE messages.

Bit :
A single unit of binary logic.

BPM :
Beats per Minute.

Buffer :
Temporary storage space for data.

Byte :
8 bit number which can indicate values from Ø to 255, or $ØØ - $FF in hexadecimal

Channel :
Refering to the MIDI channel scheme. The MIDI protocol allows for VOICE and MODE messages to be sent on 16 discrete channels.

Channel Message :
Part of MIDI messages including VOICE and MODE messages.

Channel Reference Table : 23
A system inside the MPU for keeping track over all NOTE ON and NOTE OFF events to prevent the possibility of notes hanging if the MIDI messages are interrupted.

Clear PC : 31
Clear Play Counters.

Clear RC : 31
Clear Record Counter.

CLK :
Clock.

Clock : 9
This refers to the method which the MPU-4Ø1 uses to keep track of time between MIDI events.

CLOCK TO HOST : 13
An MPU MESSAGE ($FD) sent to the HOST at regular intervals when enabled.

CNT :
Counter.

Command : 16
In this reference manual it is used to mean an MPU command that is a command to the MPU from the HOST.

Common : 22
Part of MIDI SYSTEM messages.

COMPORT : 7
MPU-401's port which the HOST can send MPU commands to.

Conductor : 42
Data structure in the HOST which consists MPU commands with a leading timing byte to control the MPU-401 in sequence.

CONT :
Continue.

CONTINUE : 16
A MIDI REAL TIME message ($FB). To resume recording or playing.

Control Change : 33
Part of MIDI VOICE messages which shows movement in controllers such as modulation wheels, damper pedals and so on.

Data :
1. Data byte. (bit 7 = 0) Content of a MIDI message which is defined by a MIDI status byte.
2. Content of an MPU command which is defined by the command.

DATA IN STOP : 27
A mode in which the MPU-401 sends MIDI VOICE messages in acceptable channels received from MIDI IN to the HOST (normally without leading timing bytes).

DATAPORT : 7
Bi-directional port of the MPU-401 for data transfer.

Default : 39
The power-up settings for functions or values on computers and computer devices.

DRR * : 7
Data Receive Ready (active low). The HOST can read this signal on bit 6 of STATPORT. When the DRR * is low the HOST can send a command or a data byte.

DSR * : 7
1. Data Send Ready (active low). The HOST can read this signal on bit 7 of STATPORT or DSR * line. The MPU-401 wants to send a byte to the HOST when DSR * is low.
2. Line for DSR * signal.

EOX : 34
MIDI End Of Exclusive ($F7). MIDI protocol byte sent at the end of an exclusive message.

Exclusive : 22
One of MIDI SYSTEM messages which is intended for specific purpose such as bulk data transfer, sending note parameters, acknowledge of receiving bulk data etc. This message includes an $F0, an ID number, data bytes and end with an EOX.

Exclusive THRU : 22
An MPU-401's mode in which the Exclusive messages from MIDI in are transmitted to MIDI OUT.

Exclusive to HOST : 28
An MPU-401's mode in which the Exclusive messages from MIDI IN are sent to the HOST.

FSK:
Frequency Shifted Key. A reliable method for representing clock data on magnetic tape.

Graduation : 34
A smooth change of tempo.

Hexadecimal :
Method of counting (base 16) typically used in computer programming.

decimal	: 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
hexadecimal	: 0	1	2	3	4	5	6	7	8	9	$A	$B	$C	$D	$E	$F

decimal	: 16	32	48	64	..	128	..	255	256
hexadecimal	: $10	$20	$30	$40	..	$80	..	$FF	$100

HOST :
This refers to the controlling device connected to the MPU-401. (i.e. a computer!)

ID number :
Identification number byte which follows a status byte $F0 to specify the exclusive message. This number can only be obtained from the MIDI committee. The Roland ID number is $41.

INT :
Internal.

Internal clock : 24
Clock controlled by internal timebase and tempo value of the MPU commands.

Internal timebase : 32
The number of divisions of the internal clock per quarter note which can be set by MPU command.

INT/HOST clock : 35

A command to set the rate of CLOCK TO HOST.

I/O :
Input / Output.

I/O mapped :
A method of interfacing peripheral devices to computers using special I/O instructions in the computers instruction set for information transfer.

Lead-in : 48
Time before RECORD begins (usually 1 or 2 measures) which allows a musician to hear the metronome and adjust to the tempo before actual recording starts.

Leading timing byte : 9
Timing byte preceding a MIDI message or an MPU MARK.

LSB :
Least significant bit(s).

ME :
Measure End.

Measure end : 11
An MPU MARK sent to the HOST to indicate the end of a measure (based on metronome).

Memory mapped :
A method of interfacing peripheral devices to computers which allows the peripherals to be accessed using load-from-memory or store-to-memory operations.

Message :
Term which consists one or more bytes for an event, defined by MIDI format or MPU format.

Metronome : 41
The sound reference on the MPU-401 which is audible when the METRONOME ON command $83 or $85 is sent.

MIDI :
Musical Instrument Digital Interface.

MIDI bus :
Hardware of MIDI for transfer. DIN cable, DIN connectors and circuitry for transmitter / receiver specified by MIDI protocol.

MIDI channel :
MIDI is capable of sending musical data over 16 different channels. They are identified by the last four bits of the MIDI STATUS BYTE.

MIDI CLOCK : 24
A MIDI REAL TIME ($F8) message sent over the MIDI bus to synchronize other equipment. It is sent at a rate of 24 divisions per beat.

MIDI IN :
DIN female connector to receive MIDI messages.

MIDI IN TABLE : 26
Note table which supervises all incoming MIDI note events to the MPU.

MIDI message :
MIDI format message with which any MIDI information is sent with over the MIDI bus.

MIDI OUT :
DIN female connector to transmit MIDI messages.

MIDI status :
A byte which leads data bytes of a MIDI message to indicate the type of MIDI message and its channel number.

MIDI THRU :
An output on a MIDI device which sends an exact copy of the information coming into the device on the MIDI IN input.

Mode :
1. MIDI - Refers to whether MIDI receivers are in 1. OMNI ON or OFF, and 2. POLY or MONO mode.
2. MPU-401 - Condition of the MPU-401 such as RECORD, PLAY, STOP or UART mode etc.

MODE MESSAGE :
Part of MIDI CHANNEL messages.

MONO :
1. One of MIDI MODE messages.
2. A MIDI receiver mode, in which only one voice per channel can be recognized by the receiver.

MPU command : 16
Information sent from the HOST to COMPORT of the MPU-401 which controls the operation of the MPU-401.

MPU MARK : 11
A message with a leading timing byte created by or recognized in the MPU-401.

MPU message :
A message created in the MPU-401 to send a function to the HOST. (Without leading timing byte.)

MSB :
Most significant bit(s).

NOTE OFF event : 33
A MIDI message indicating a note on the keyboard has been released.

NOTE ON event : 33
A MIDI message indicating a note on the keyboard has been pressed.

OMNI OFF :
1. One of MIDI MODE messages.
2. A MIDI receiver mode, in which voice messages in the receiver's channel are recognized.

OMNI ON :
1. One of MIDI MODE messages.
2. A MIDI receiver mode, in which all voice messages in all channels are recognized by the receiver.

PLAY : 16
1. An MPU mode.
2. Transfer of MIDI VOICE messages from the HOST to MIDI bus through the MPU-401.

Play buffer : 5
Data storage area in the MPU-401 for next MIDI message transmission while PLAY mode.

Play counter : 10
Counter used in PLAY mode for timing REQUEST NEXT TRACK DATA commands to the HOST.

PLAY TABLE : 31
Note table which supervises all PLAYING note events being sent legally.

POLY :
1. One of MIDI MODE messages.
2. A MIDI receiver mode. In POLY mode, polyphonic note events in each channel can be recognized.

Program :
Information stored in a computer which controls its operation.

Program change : 33
Part of MIDI VOICE messages which is used to change the tone colors or other functions which are programmed in the receivers.

Punch-in : 16
To update recorded music or music data from an intermediate point.

Real time :
One of MIDI SYSTEM messages used for controlling time functions.

Record : 16
1. An MPU mode.
2. Transfer of MIDI messages, which are received from MIDI IN, with leading timing bytes to the HOST.

Record counter : 9
Counter used in RECORD mode to get timing value for each period between MIDI messages.

Reference table : 23
Channel reference table.

Relative tempo : 34
An MPU command with following byte which controls the internal tempo relatively.

Request :
An action taken by a computer peripheral which has information to either send to or receive from the HOST.

RT :
MIDI Real Time message(s).

Running status :
Current MIDI status which does not need to be sent for VOICE and MODE messages.
If many MIDI VOICE and MODE messages of the same type and the same channel are sent over the MIDI bus, only the first message needs a status byte. With all the rest, a 'running status' is assumed.

Rx :
Receiver.

Sequence data :
A series of MIDI information with timing values stored in the HOST's memory, which comprise a 'song'.

Stand-by recording : 16
A mode of the MPU-401 which is ready to record, waiting for a start command from either the HOST or MIDI IN.

START : 18
One of the MIDI REAL TIME messages ($FA) . Start playing from the beginning of a song.

STATPORT : 7
The MPU-401 port that the HOST can read the MPU-401's status byte which includes DRR * and DSR *.

Status :
1. MIDI status byte (bit 7 is 1) which defines type of the MIDI message.
2. Signal sent out to STATPORT for the HOST. (i.e. DRR * and DSR *)

STOP : 18
1. A MIDI REAL TIME messages ($FC). Stop playing or recording
2. A mode that the MPU-401 is not PLAY or RECORD mode.

System :
MIDI SYSTEM message - SYSTEM EXCLUSIVE messages, SYSTEM COMMON messages or SYSTEM REAL TIME messages. See MIDI message.

System message :
Part of MIDI messages.

Tempo : 34
The speed of the MPU-401's internal or external clock which controls recording and playback of music.

THRU :
MIDI THRU

Timebase : 32
The number of clock pulses per beat.

Time signature :
The number of beats per measure over the type of note which represents one beat.

Timing byte :
The present value of the MPU-401 clock that accompanies all MIDI messages or MPU marks transfered between HOST and MPU-401. See Leading Timing Byte.

Timing value :
Timing byte.

Track :
A designated path for MIDI messages in the MPU-401 and the HOST.

TX :
Transmitter.

UART : 22

Universal Asynchronous Receiver and Transmitter. The device which makes computer serial communication possible.

Voice message :

Part of MIDI CHANNEL messages. NOTE ON, NOTE OFF, AFTER TOUCH, CONTROL CHANGES, PROGRAM CHANGES and BENDER messages. They always belong to one of MIDI channels.

WSD : 33

Want to Send Data. An MPU command which tells the MPU-401 to transmit a MIDI message following the command.

$:

Sign of a number described in HEXADECIMAL.

MPU Command Table

MPU Commands

V1.5

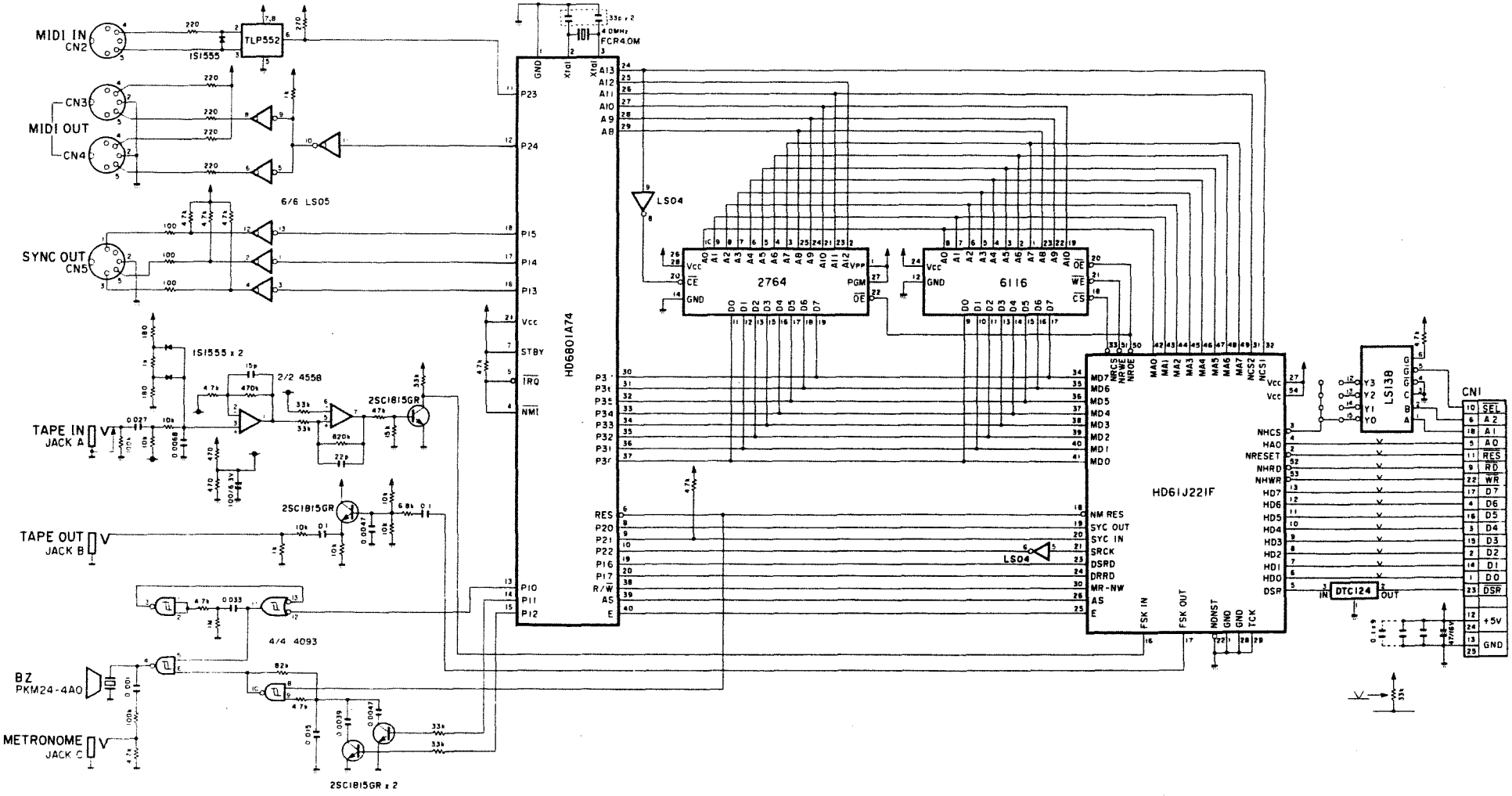
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00		MIDI Stop	MIDI Start	MIDI Cont		MIDI Stop	MIDI Start	MIDI Cont		MIDI Stop	MIDI Start	MIDI Cont				
10	Stop Rec					Stop	Play			Start	Play					
20	Rec															
30	All note OFF		No RT	THRU OFF	With Timing Byte	Mode Mess		Exclu thru on	Common to Host on	RT to Host on	In UART mode, $FF: come back					UART
40	Set ch of Ref table A: 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
50	Set ch of Ref table B															
60	Set ch of Ref table C															
70	Set ch of Ref table D															
80	Clock Sync/Mode: INT	FSK	MIDI	w/o Acc on	Metronome: off	on	Bender to Host: off	on	THRU: off	on	Data in Stop: off	on	Send ME: off	on	Conductor: off	on
90	RT Aff: off	on	FSK Reso: INT	24	CLK to Host: off	on	Exclu to Host: off	on	Ref A: off	on	Ref B: off	on	Ref C: off	on	Ref D: off	on
A0	Req Play CNT: Tr1	Tr2	Tr3	Tr4	Tr5	Tr6	Tr7	Tr8				Read & Clear RC	Req Version	Req Revision		Req Tempo
B0		Clear Rel							Clear PC	Clear PM	Clear RC					
C0	INT Time Base:	48	72	96	120	144	168	192								
D0	Want to Send Data: Tr1	Tr2	Tr3	Tr4	Tr5	Tr6	Tr7	Tr8								WSD System
E0	Set Tempo	Rel Tempo	Gradu-ation		MIDI/ Metro		Metro/ Meas	INT *4/ H. CLK					Act Tr 1–8	Send Play CNT	Acc CH 1–8	Acc CH 9–16
F0																Reset

MPU Messages

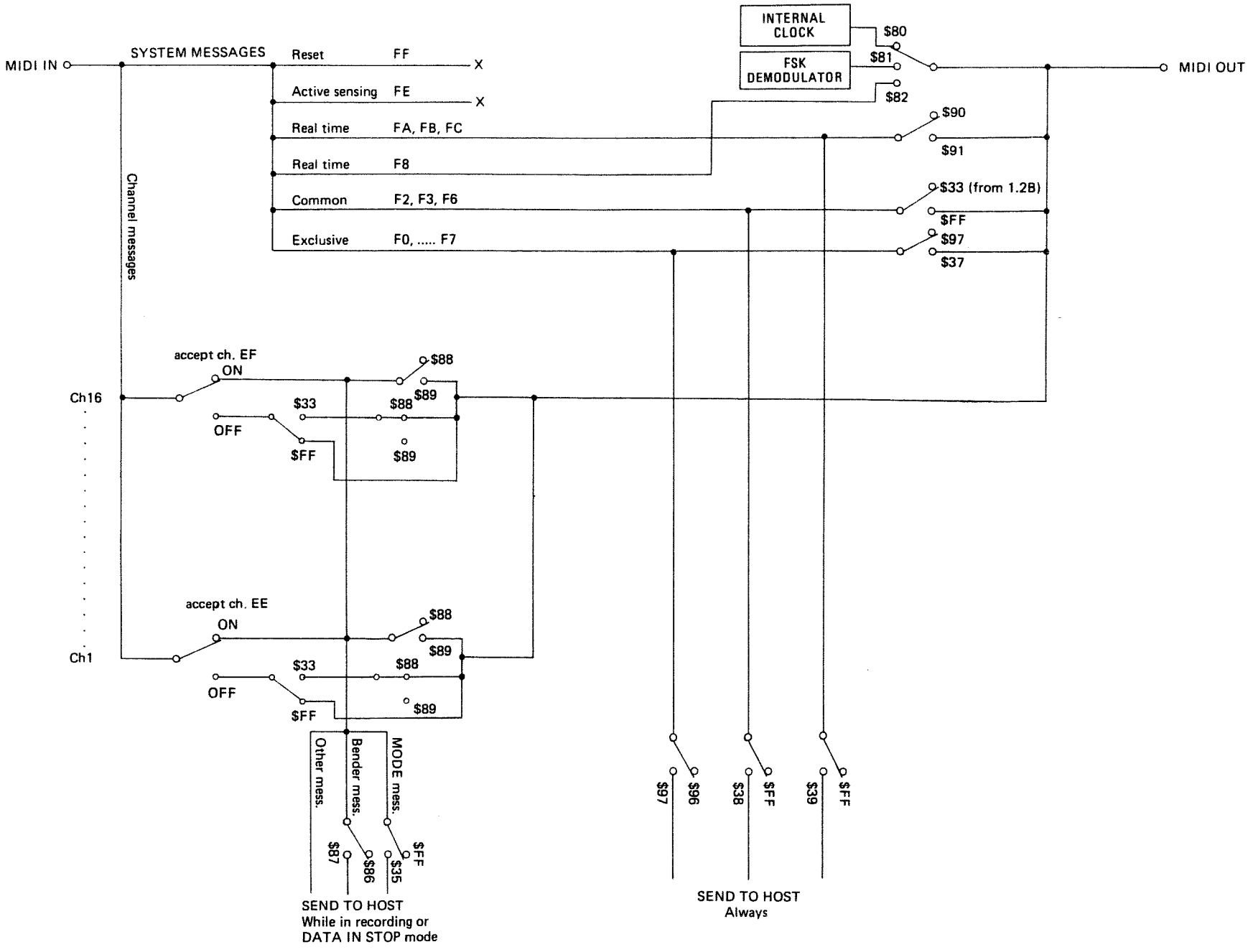
F0	TRACK DATA REQ (With Time Value): Tr1	Tr2	Tr3	Tr4	Tr5	Tr6	Tr7	Tr8	Over Flow	Req Cond			Data End	H. CLK	ACK	SYS

Default Condition	
MIDI Play Rec	Stop
No RT	: off
With Timing	: off
Mode Mess	: off
Active Sensing	: off
Exclu Thru	: off
Common to Host	: off
Real Time to Host	: off
Ref Table	
A	: CH-1, on
B	: CH-2, on
C	: CH-3, on
D	: CH-4, on
Sync Mode	: INT
Metro	: off
THRU	: on
Data in Stop	: off
Send ME	: on
Conductor	: off
Real Time Affection	: on
FSK Reso	: INT
CLK to Host	: off
Exclu to Host	: off
INT Time BASE	: 120
Tempo	: 100
Rel Tempo	: $40
MIDI/Metro	: 12
Metro/Meas	: 8
INT x 4/H	: 240
ActiveTr	: NONE
Send Play CNT Tr	: NONE
Acceptable CH	: ALL

MPU-401 Circuit Diagram



MIDI message flow



Roland®

2701040200

UPC 2701040200

Roland®