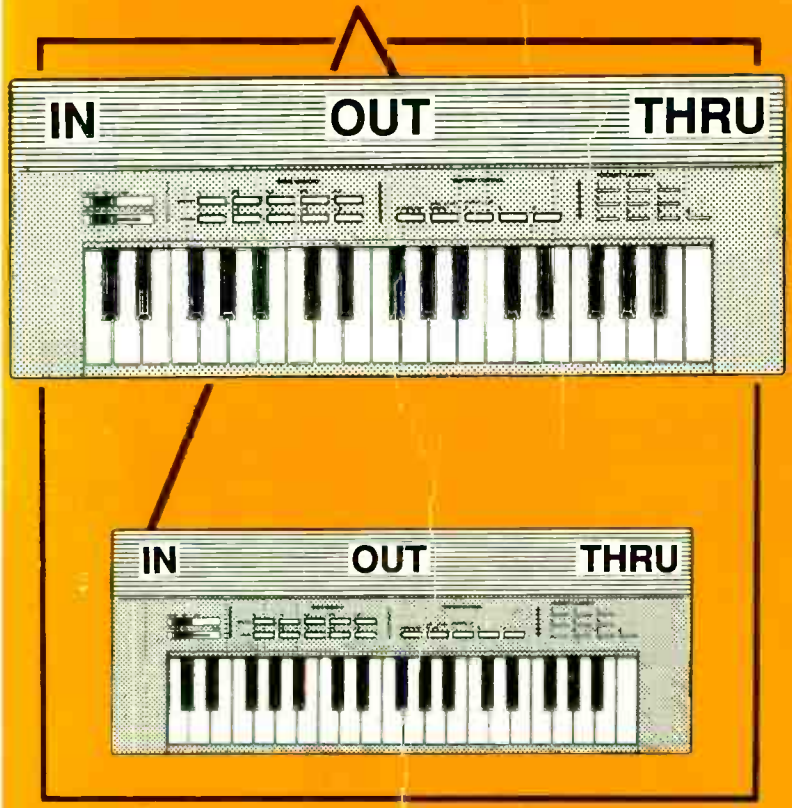
A Beginners Guide to MIDI

R.A. PENFOLD



A BEGINNERS GUIDE TO MIDI

Other Titles of Interest

BP182 MIDI Projects
BP247 More Advanced MIDI Projects
BP329 Electronic Music Learning Projects

A BEGINNERS GUIDE TO MIDI

By

R. A. PENFOLD

BERNARD BABANI (publishing) LTD
THE GRAMPIANS
SHEPHERDS BUSH ROAD
LONDON W6 7NF
ENGLAND

Please Note

Although every care has been taken with the production of this book to ensure that any projects, designs, modifications and/or programs, etc., contained herewith, operate in a correct and safe manner and also that any components specified are normally available in Great Britain, the Publishers do not accept responsibility in any way for the failure, including fault in design, of any project, design, modification or program to work correctly, or to cause damage to any other equipment that it may be connected to or used in conjunction with, or in respect of any other damage or injury that may be so caused, nor do the Publishers accept responsibility in any way for the failure to obtain specified components.

Notice is also given that if equipment that is still under warranty is modified in any way or used or connected with home-built equipment then that warranty may be void.

First Published – June 1993

British Library Cataloguing in Publication Data
Penfold, R. A.
Beginner's Guide to Midi
I. Title
780.285416

ISBN 0 85934 331 6

Printed and Bound in Great Britain by Cox & Wyman Ltd, Reading

Preface

MIDI is an acronym, and I have seen various explanations for the meaning of each letter. These include "Music Industry Digital Interchange", and even "Music Systems Digital Interface"! MIDI does in fact stand for "Musical Instrument Digital Interface", and its function is to permit electronic musical instruments to "talk" to each other. To be strictly accurate, MIDI can be used with any units that can operate as part of an electronic music system. This includes such things as computers, sequencers, audio mixers, and even lighting controllers.

MIDI is mainly used for what is effectively playing instruments by remote control. By playing one instrument you can simultaneously play several slave instruments. Alternatively, complex pieces of music can be recorded track-by-track into a sequencer, and then played back into the instruments of the system in order to reproduce the complete piece. A wide range of facilities are available via MIDI, and it is not restricted to simple on/off switching of notes. In theory, absolutely any desired facility could be implemented via MIDI.

It would be misleading to claim that it is possible to exploit MIDI to the full with no understanding of the technology involved. However, it is definitely possible to largely exploit MIDI's music making potential without delving into the precise way in which it functions. The essential knowledge required is an understanding of how MIDI systems are interconnected, the standard facilities that are available, and how they are used in practice. These are the topics covered in this book. You do not need to have any previous knowledge of MIDI, computing, etc., in order to use this book, but some background knowledge of electronic musical instruments is assumed.

R. A. Penfold

Contents

Chapter 1

CONNECTIONS

In days gone by, linking two synthesisers successfully was often a difficult process. There was a lack of proper standardisation which meant that the connecting leads often had to be specially made up, and tailored to suit the two particular instruments that were to be interconnected. It was usually the case that once the two instruments were connected together your troubles were just beginning! If the two instruments were produced by the same manufacturer, then you might be in luck. If not, it was quite likely that they would not operate properly together. Often a simple interface device and another lead would get things up and running, but in some cases there was no easy way of getting the two instruments to communicate properly.

Standardisation

MIDI came into being largely as a result of this lack of standardisation. New and much more powerful electronic musical instruments were being developed in the early 1980s, and they needed something beyond the simple methods of interfacing that had been used up until that time. These provided only very basic control of a synthesiser, and when applied to polyphonic instruments required large numbers of interconnections.

There was a need for something much more sophisticated that used just a twin connecting lead. This really meant some form of computer style digital interface. Synthesisers having microprocessor control were starting to appear, and so some form of computer-like interface was more appropriate than the old style synthesiser interfaces.

An advantage of such an interface is that it can handle quite complex dialogues between the two instruments. All a digital interface does is to provide a means of swapping numbers between two or more devices. This might not seem to be particularly useful, but in a practical system the numbers are codes which can mean anything you like. With suitable

coding it is possible to provide control of any desired aspect of an instrument, or any device which is used in electronic music systems (audio mixers, digital effects units, etc.). Polyphonic control of notes, pitch bending, changes to the sound generator circuits, or anything else, can all be carried down the same twin lead cable.

There was an urgent need for a truly standard digital interface that would avoid all the problems of incompatibility that had existed up until then. These problems were potentially going to multiply a thousandfold. With each manufacturer producing its own sophisticated form of digital interface it would be virtually impossible to successfully link two devices from different manufacturers. Interfacing such units would probably require a unit that would first overcome any differences in the hardware. It would then have to translate the code numbers from one instrument into the method of coding used by the other. Such a unit is quite feasible, but would probably be nearly as complex and expensive as the instruments themselves.

Fortunately, a number of the main electronic musical instrument producers agreed on a properly standardised digital interface. Both the hardware and the software (method of coding) were standardised to a high degree. This means that it is possible to successfully link any two MIDI equipped devices by simply connecting them together using a standard "off the shelf" lead. Obviously some instruments are much more sophisticated than others, but any two MIDI units should be able to operate properly together in at least a very basic fashion. In the vast majority of cases, especially if the units involved are reasonably modern, they will work together in a very sophisticated fashion.

Loop The Loop

We have been talking in terms of two instruments (or other devices) being connected together, but MIDI actually enables any number of units to be interconnected. There are three types of port on units that have a full MIDI interface. These are marked "IN", "OUT", and "THRU". It is not a requirement of the MIDI standard that these should all be present, and in the early days of MIDI "THRU" sockets were

something of a rarity. For reasons that will soon become apparent, this made it difficult to connect everything together in a large MIDI system.

All three types of port use the same type of socket, which is a 5 pin (180 degree) DIN type. This is also known as a 5 pin DIN socket type "A". Although the sockets are 5 way types, there are actually only two connecting wires in a MIDI cable. There is also a screen connection, but this is simply an earthed metal sheath over the other two wires. This sheath is needed to prevent radio interference from being radiated by the cable, and it does not carry a signal. The screen only connects to anything at one end of the system, and does not provide an electrical link between units in the system. Things are arranged this way as it helps to avoid problems with "hum" and "earth" loops.

At every MIDI input there is a device called an opto-isolator. Basically all this does is to couple the input signal through to the main circuit using infra-red light, so that there is no direct electrical connection between units in the system via the MIDI cables. This again helps to prevent problems with "hum" and "earth" loops, although these can still occur due to interconnections via the audio and mains wiring. The opto-isolation is still very worthwhile though, since it does at least mean that MIDI will not add to any existing problems with loops through the audio or mains wiring.

With only three wires (including the screen) and five pins on the connectors, there are obviously two pins on each socket that are left unused. These are reserved for possible future use, and should be left unconnected. If you wish to make up your own MIDI leads, for each one you will need two 5 pin (180 degree) DIN plugs, and some screened twin cable. In my experience any normal screened twin lead seems to be adequate for MIDI leads. An inexpensive "overall" screened twin type should be perfectly satisfactory. Note that MIDI is only guaranteed to operate using connecting cables up to 15 metres (about 50 feet) long. This should be adequate for most purposes, including systems used in "live" performances. In practice, reliable operation with longer cables is probably possible provided good quality cable is used.

The interconnections in a MIDI cable are as follows:–

Pin 4 on the first plug connects to pin 4 on the second plug
Pin 5 on the first plug connects to pin 5 on the second plug
Pin 2 on the first plug connects to pin 2 on the second plug via the screen of the cable
Pins 1 and 3 are left unconnected on both plugs.

The pins of DIN plugs and sockets are identified by numbers which are moulded into their plastic bodies. However, the numbers are very small due to the limited space which is available, and you might need a magnifier in order to see them.

There is no need to produce your own MIDI leads as they are readily available at reasonable prices. Some of these ready-made leads seem to provide interconnections between pins 1 and 3, possibly in anticipation of a "super" MIDI interface which will need these extra connections. At present they provide no useful purpose, but will not do any harm. Note that audio leads which have the correct 5 pin (180 degree) DIN connectors do not usually provide the correct interconnections for MIDI use. A normal audio lead of this type uses a form of cross connection, and will not work at all in a MIDI system.

The MIDI specification does actually allow for an alternative type of connector to be used. This is a very high quality three way connector known as the "XLR" type. This is a much tougher type of connector than the 5 way DIN variety, and it is only likely to be found in some very up-market MIDI equipment. It was the intention that this type of connector should be used in equipment which had to stand up to the rigours of "life on the road". Where this type of connector is fitted to equipment, it is a requirement of the MIDI standard that the manufacturer should make available adaptors so that ordinary (5 pin DIN type) MIDI leads can still be used with the equipment if desired. In practice it is unlikely that you will ever use any gear which is fitted with XLR connectors at its MIDI ports.

A large percentage of MIDI systems are based on computers running MIDI software, which usually means a sequencer of

some kind, plus (perhaps) librarian or other utility programs. Few computers are equipped with built-in MIDI interfaces, the Atari ST series of computers being the main exception. This means that most computers are only usable in a MIDI system if they are fitted with an add-on MIDI interface unit. These are available for most of the popular computers, but you should be careful to ensure that the add-on MIDI interface you intend to use is fully compatible with the MIDI software that you will be running.

Chain Links

In a MIDI system there is always a controller and one or more slave units. The controller is the device which you use to send the MIDI messages that control the other units in the system. This would normally be a synthesiser or other keyboard instrument, a dedicated sequencer, or a computer running a sequencer or other MIDI program. However, it could be something more exotic such as a MIDI guitar, or indeed anything that has a MIDI "OUT" socket.

In a simple system which just consists of a master controller and one slave instrument, the "OUT" socket of the master instrument connects to the "IN" socket of the slave instrument. This most basic of MIDI setups is shown in Figure 1.1. With a system of this type you are limited to having the slave instrument mimic the playing on the master instrument, but this can provide much improved sounds. The slave instrument is set to give a different sound to the master instrument. For example, the main instrument could be set to give a piano sound, while the slave instrument could be set up for a "strings" sound. With many instruments there are built-in filter facilities which enable a slave instrument to ignore certain types of MIDI message. This filtering can be used to produce small but useful differences between the two instruments.

A further slave instrument can be connected into the system by connecting the "THRU" socket of the first slave to the "IN" socket of the second slave instrument. In fact any number of instruments can be connected into the system using this method of connection, with the "THRU" socket of one instrument being connected to the "IN" socket of the next

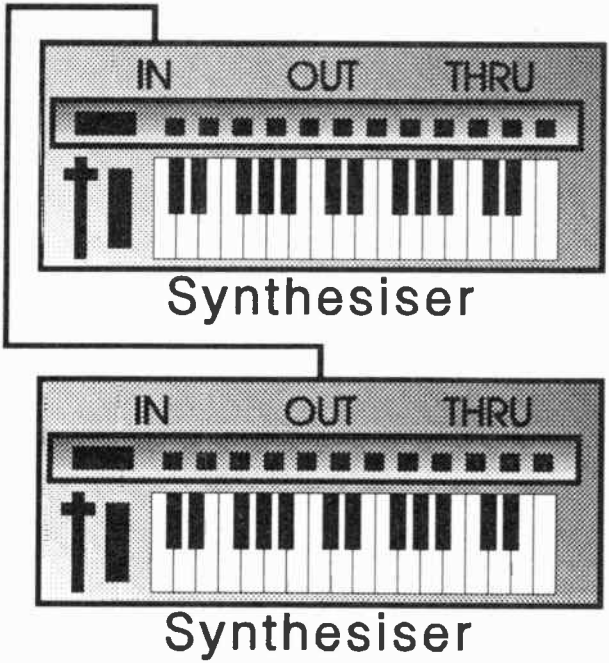


Fig. 1.1 *The most basic of MIDI systems. The slave instrument (top) mimics the playing on the master instrument*

instrument in the chain. Figure 1.2 shows a "chain" system of this type which has a master instrument plus three slave units. A "THRU" socket is a form of output, but it simply provides a replica of the input signal. The input signal from the controller is therefore coupled through each instrument and on to the next one in the chain.

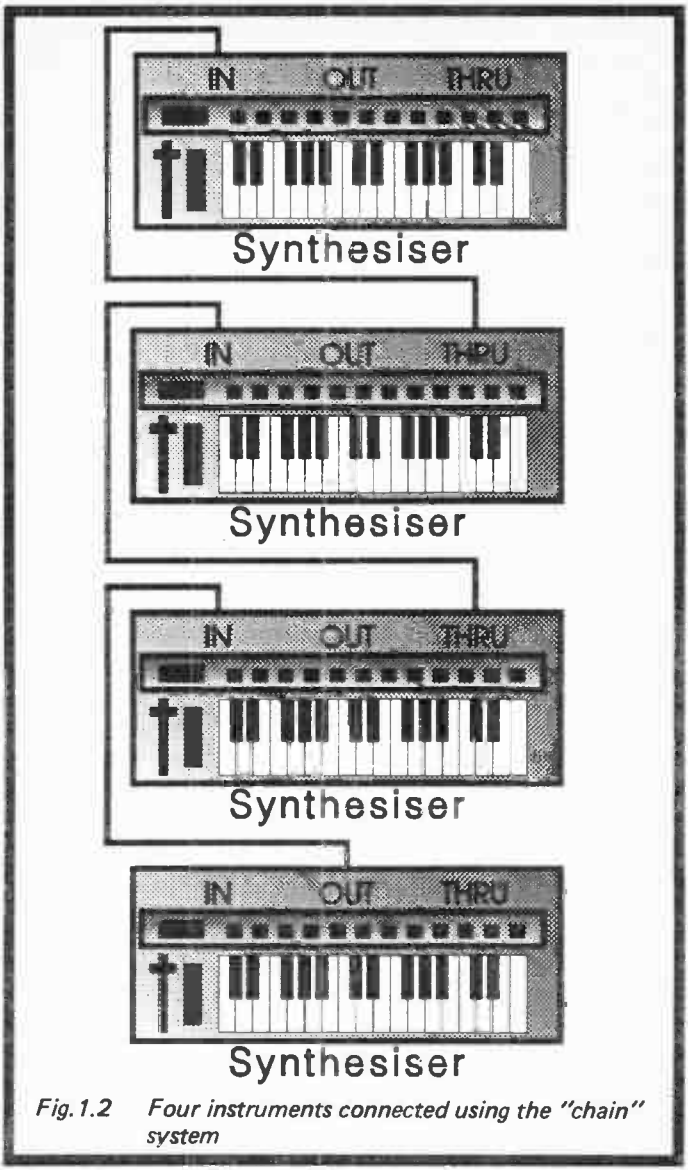


Fig. 1.2 Four instruments connected using the "chain" system

MIDI Delays

Although it is theoretically possible to have any number of instruments wired together using the chain system of connection, in reality there is a practical limit to the number of devices that can be used. The MIDI signal tends to be degraded slightly by each transition from an "IN" socket to a "THRU" type. This is due to the fact that the opto-isolators at MIDI inputs are not particularly fast in electronic terms. In fact they are quite slow by digital electronics standards. This produces a small amount of distortion to the signal as it passes from an "IN" socket to a "THRU" type, and in a long chain of instruments the distortion could accumulate so much that the instruments towards the end of the chain would not be able to decode the signal properly.

The MIDI specification makes it clear that you can not simply go on chaining more and more instruments together without the system failing to function properly at some stage. However, it does not give any guidance as to the maximum number of instruments that can safely be used in a chain system. In practice, the maximum number of devices that can be used before reliability becomes problematic is probably quite large. I have certainly never experienced this so-called "MIDI delay" problem. It should certainly be possible to have half a dozen or so instruments in a chain system, which is more than enough for most of us.

There seems to be a certain amount of confusion about MIDI delays. Some MIDI users seem to be under the impression that this problem manifests itself in the form of a perceptible delay in the time taken for notes played on the master instrument to reach the instruments at the end of the chain. This is not the case, and for equipment that conforms to the MIDI specification there should be a delay of only a millionth of a second or so through each instrument. Even with a hundred instruments in the system, this would not give a significant delay.

The problem is strictly one of signal degradation, and if you should experience this problem, it will become apparent when the final instrument in a chain system fails to operate reliably. The problem is most likely to manifest itself in the form of notes occasionally being left switched on. In a very

bad case an instrument would simply show something along the lines of "MIDI ERROR" on its display panel, and refuse to do anything further until the problem is solved.

Star System

There is an alternative to the chain system in the form of the "star" method of connection. Diagrams in magazines and books often show this system in the form of a controller at the centre, surrounded by five or six slave units, giving a star-like appearance to the system. It is from this that the "star" name is derived, but practical systems are usually far from star-like. Figure 1.3 shows the star equivalent to the setup of Figure 1.2, with no attempt to make the system live up to its name!

A star system has the advantage of avoiding problems with MIDI delays. No units in the system receive their input signal after it has passed from an "IN" socket to a "THRU" type. Consequently there are no delays to cause problems. A star system in the form shown in Figure 1.3 is only possible if the MIDI controller has multiple outputs. A few MIDI controllers are equipped with several "OUT" sockets, but most are not. There is a way around this problem though. All that is needed is a simple device called a "THRU box". This has one "IN" socket and several "THRU" outputs, which enables it to drive a number of devices from the source signal.

Figure 1.4 shows a star system which is much the same as the one featured in Figure 1.3, but it utilizes a THRU box. Obviously this system is not totally free from delays, since every slave unit receives a signal that has passed from an "IN" socket to a "THRU" type. However, each unit receives a signal which has only undergone one such journey, and this means that any distortion of the signal will be far too small to be of any significance.

Which method of connection should you use for your MIDI system, the star or chain system? If your instruments have "THRU" sockets, then there is probably no point in using anything other than the chain system. This is very simple and straightforward, requiring no extra hardware. Even if one slave unit in the system lacks a "THRU" socket, it

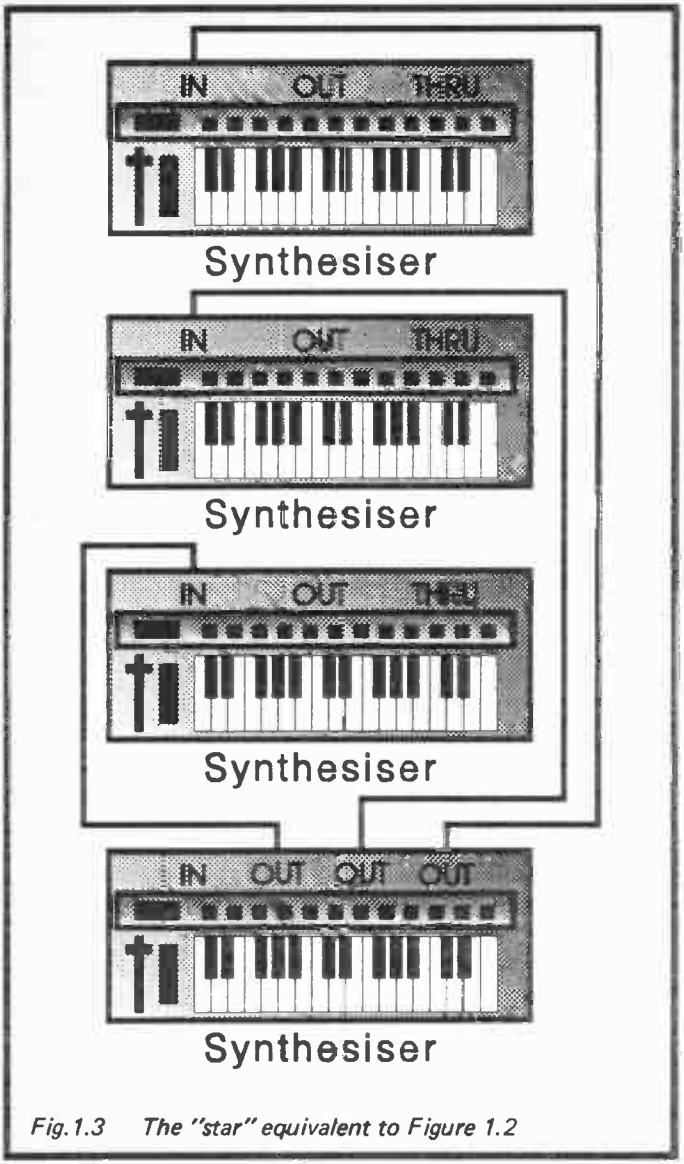


Fig. 1.3 The "star" equivalent to Figure 1.2

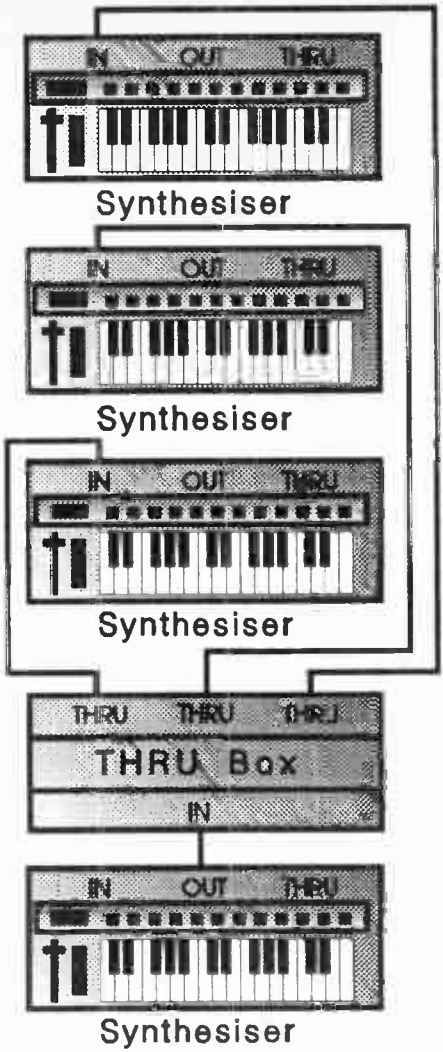


Fig.1.4 The "star" system usually requires the addition of a THRU box

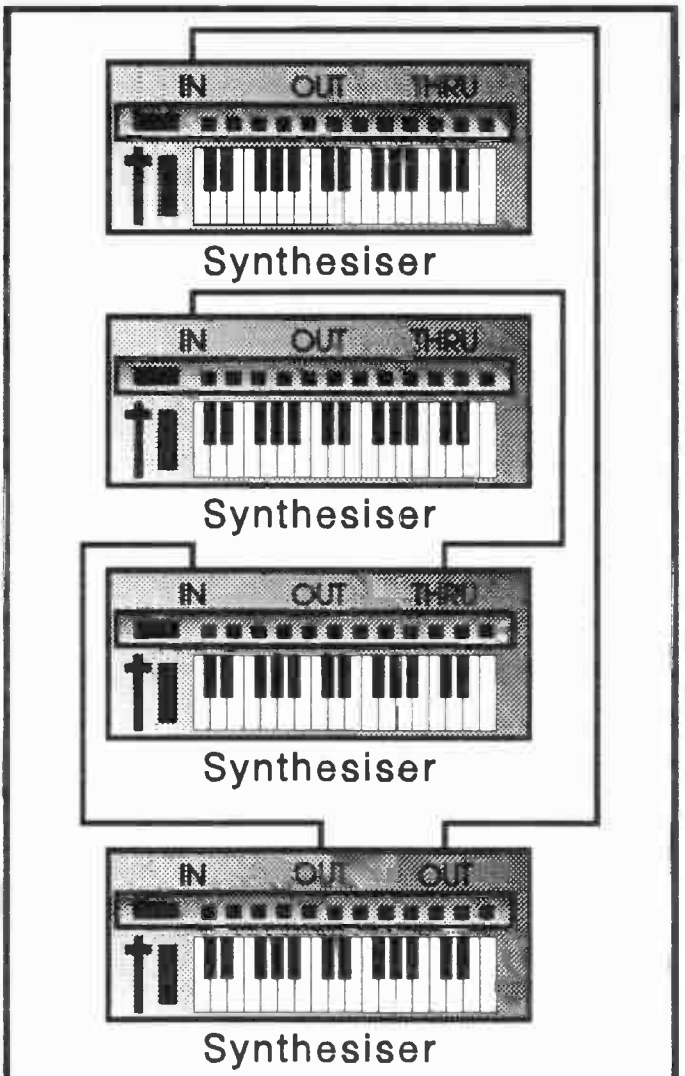


Fig. 1.5 This system uses a combination of the "star" and "chain" systems

is still possible to use the chain system. Simply have the unit which lacks the "THRU" socket at the end of the chain.

If two or more slave units lack a "THRU" socket there is only one choice, and the star system must be adopted. If the controller lacks sufficient "OUT" sockets, a THRU box must be added to the system. "THRU" sockets are often absent on older MIDI equipment, but provided you are using reasonably modern equipment it is unlikely that you will be forced into using the star system. If you use the chain system and experience problems with MIDI delays, the star system should be used instead. It is unlikely that you will have any problems with the chain system though.

It is worth making the point that it is quite possible to use a combination of the star and chain systems. In the setup of Figure 1.5 for example, the middle two instruments are connected to the master unit using the chain method. The other slave instrument (the one at the top) is driven from a separate output socket, and it is effectively in a star system. When designing a MIDI system the important thing is to ensure that the controller's output connects through to each slave unit by some means or other. The system should then work properly.

Sequencing

A system which will be used for real-time sequencing needs a slightly different setup to the basic master/slave arrangement. The controller must still have its output fed to the "IN" socket of each slave unit in the system, and this can be achieved using the star or chain method of connection. In the case of a step-time sequencer this is all that is needed, since the sequences are entered into the sequencer using a computer type keyboard, a computer "mouse", or some form of alphanumeric keypad. Real-time sequencers are used far more than the step-time variety, and they require a MIDI keyboard instrument, or just a plain MIDI keyboard, to drive the "IN" socket of the sequencer. It is then possible to play tracks on the keyboard and record them using the sequencer.

Figure 1.6 shows a typical real-time sequencer setup. This is basically the same as a system which has a master controller plus some slave instruments, but there is a link from the

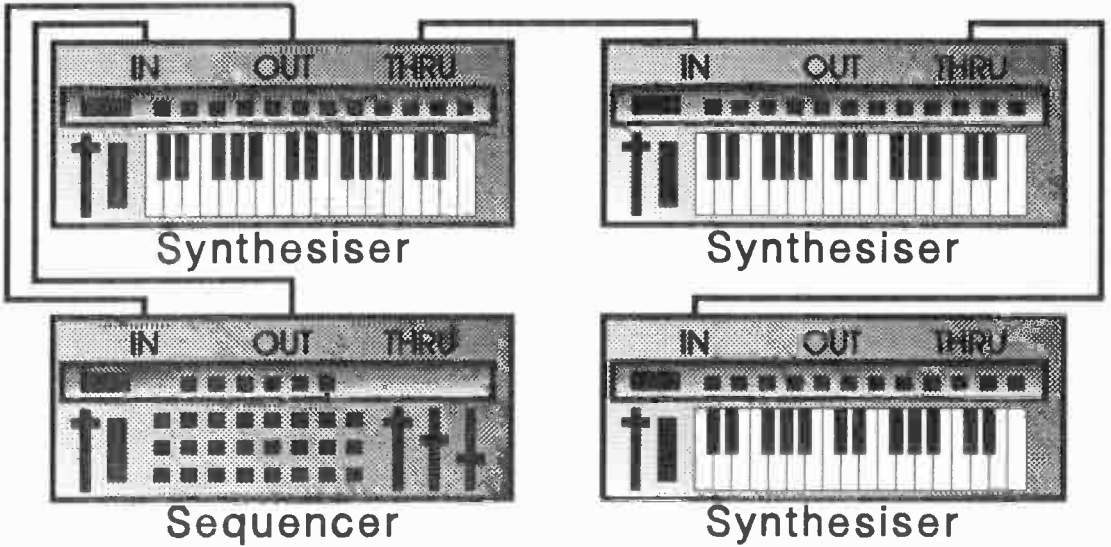


Fig.1.6 A system for real-time sequencing

"OUT" socket of the first synthesiser (the one at the top left) to the "IN" socket of the sequencer. Notes are played on the keyboard of the first synthesiser, and the resultant MIDI signals are recorded using the sequencer. The piece is built up track by track, and it is then played-back into the synthesisers.

If the MIDI keyboard is a dedicated keyboard unit, rather than the keyboard of a synthesiser or other instrument, there may seem to be no point in having it in the chain of devices fed from the output of the sequencer. There are no sound generator circuits to sequence! In reality most MIDI keyboards are quite sophisticated pieces of equipment which can often respond to certain message types from a sequencer or other controller. It is therefore a good idea to include the keyboard in chain of slave devices. This leaves your options open, and it only requires one extra MIDI lead.

There is a potential problem with this basic sequencer setup in that it does not provide any means of playing the slave instruments direct from the keyboard. It is often helpful to have one of the slave instruments play a track while it is being played on the keyboard and recorded using the sequencer. This can only be achieved with some slight rewiring, such as unplugging the lead from the "OUT" socket of the sequencer, and connecting it to the "THRU" socket instead. In most cases this is unnecessary, because most sequencers have a "THRU" facility. When set to this mode, the data received on the "IN" socket is echoed on the "OUT" socket. This feeds the output of the keyboard through to the instruments in the system.

The hardware alternative is to use a device called a "merge" unit in the manner shown in Figure 1.7. The merge unit takes the signals from the "OUT" and "THRU" sockets of the sequencer, and couples them through to the slave instruments. With a full merge unit it is actually possible to feed it with signals on both inputs simultaneously, and it will combine them into a single coherent MIDI output signal. This makes it possible to play along "live" with a sequence that is being played-back. Some merge units have more than two inputs, but any inputs that are not needed can simply be left unused.

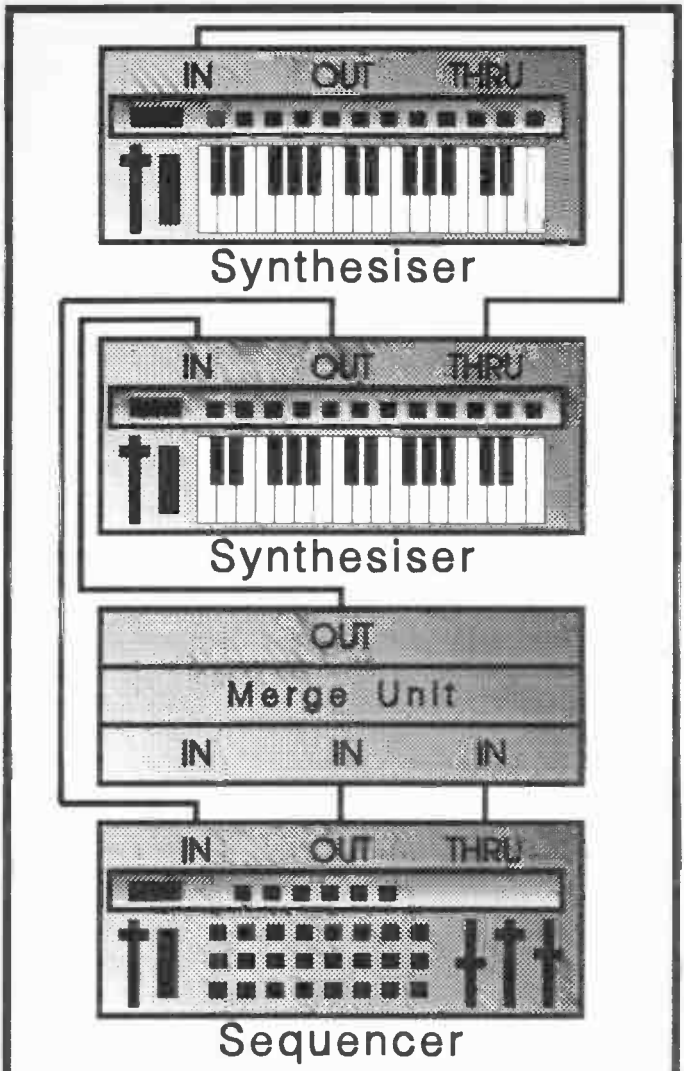


Fig.1.7 A sequencer system using a merge unit. Many sequencers have a built-in merge facility

In general there can only be one controller in a MIDI system. In a basic sequencer setup there are actually two controllers, which are the keyboard and the sequencer. This is acceptable only because there are effectively two MIDI systems. One is comprised of the keyboard controller and the sequencer as the slave unit. The other is comprised of the sequencer as the master unit and everything else as a slave to this. Problems arise when an attempt is made to control the slave units from both the keyboard and the sequencer. This gives what is really two controllers with what has become a single system. Whenever this situation occurs, the only solution is to use some form of merge unit to effectively combine the two controllers into a single signal source.

Patching Up

With a complex MIDI system it might be worthwhile using a patch bay. The basic idea is to have everything in the system with both its "IN" and "OUT" sockets connected to the patch bay. Figure 1.8 shows a mini patch bay system which should give you the general idea. Patch bays are mainly used in complex systems having about half a dozen instruments, and are not generally worthwhile in simple MIDI systems. The "THRU" sockets of the instruments are left unconnected because using a patch bay gives what is a form of star system. The patch bay effectively operates as a sort of ultra-versatile THRU box.

What connects to where is controlled by the patch bay, and any "IN" socket can be connected through to any "OUT" socket or sockets. The system can therefore be quickly reconfigured using the controls of the patch bay unit. Most MIDI patch bays seem to be quite complex pieces of equipment which offer sophisticated merging facilities. They often have other capabilities, but this is something that varies significantly from one unit to another. Any sophisticated patch bay can store a number of preset configurations. These units can be switched from one configuration to another via messages from the MIDI controller. This enables the system to be reconfigured in the middle of a sequence, or even during a "live" performance.

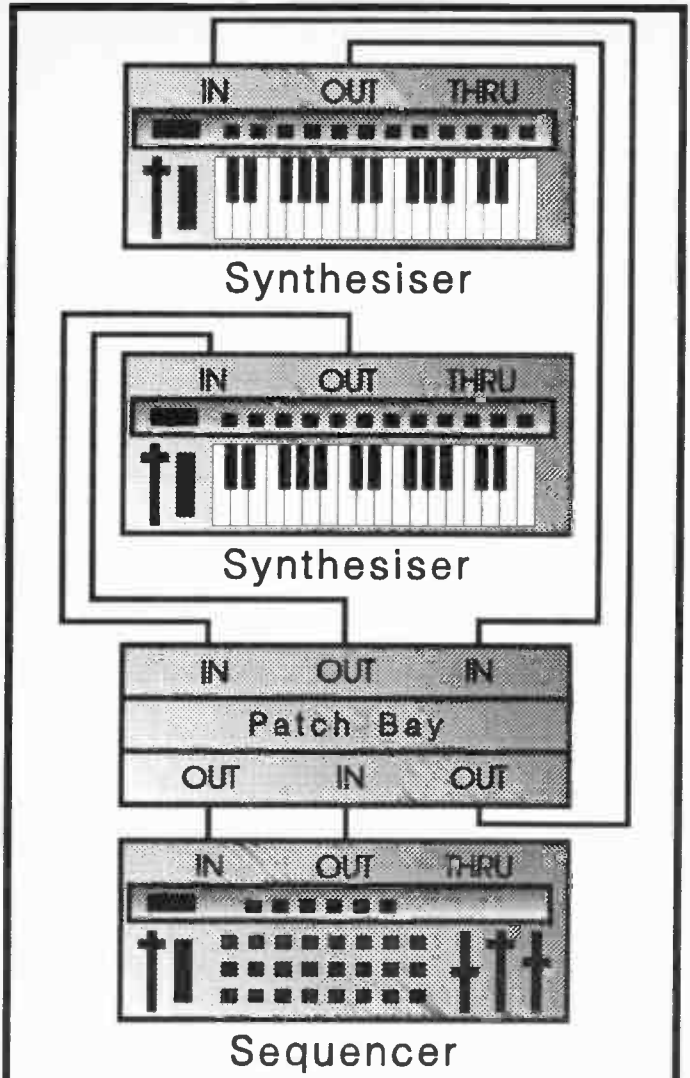


Fig. 1.8 The interconnections for a simple system based on a patch bay

Patch bays give great versatility, but tend to be something less than completely straightforward in use. To get the best from a patch bay you will usually need to use some careful advance planning. Many users now prefer to use a system which has multiple MIDI outputs, and this is certainly the type of "mega" MIDI system which I find the most usable. I must emphasise here that I am talking about a MIDI controller which has several MIDI outputs which are totally independent, and carry different signals. Figure 1.9 shows a basic system of this type.

This may look rather like an ordinary star system, but it is really far more versatile. Using a single MIDI output it is possible to connect any number of instruments into the system. In practice there is a limit to the number of instruments that can usefully be driven from a single MIDI output. MIDI has facilities which permit individual control over the instruments in the system, but this is one respect in which the designers of the original MIDI system did not really allow sufficient "future proofing". With a system based on a number of modern instruments a single MIDI output can be rather limiting for the more advanced user. Having several individually controllable outputs enables a large number of instruments to be easily controlled, and controlled completely independently of each other if required.

Small Beginnings

Probably the best advice for newcomers to MIDI is to start off with a very simple and straightforward system using one of the standard methods of interconnection. Expand or modify the system as and when the need arises. Try not to get carried away with ever more complex but impractical setups. It is very easy to design clever systems having wires going here, there, and everywhere. Managing to use such systems is likely to prove difficult, if the system is usable at all. Keeping everything simple and straightforward provides you with a system that should work well, and which should be easy to use.

Summary

MIDI is a computer-style digital interface which basically just

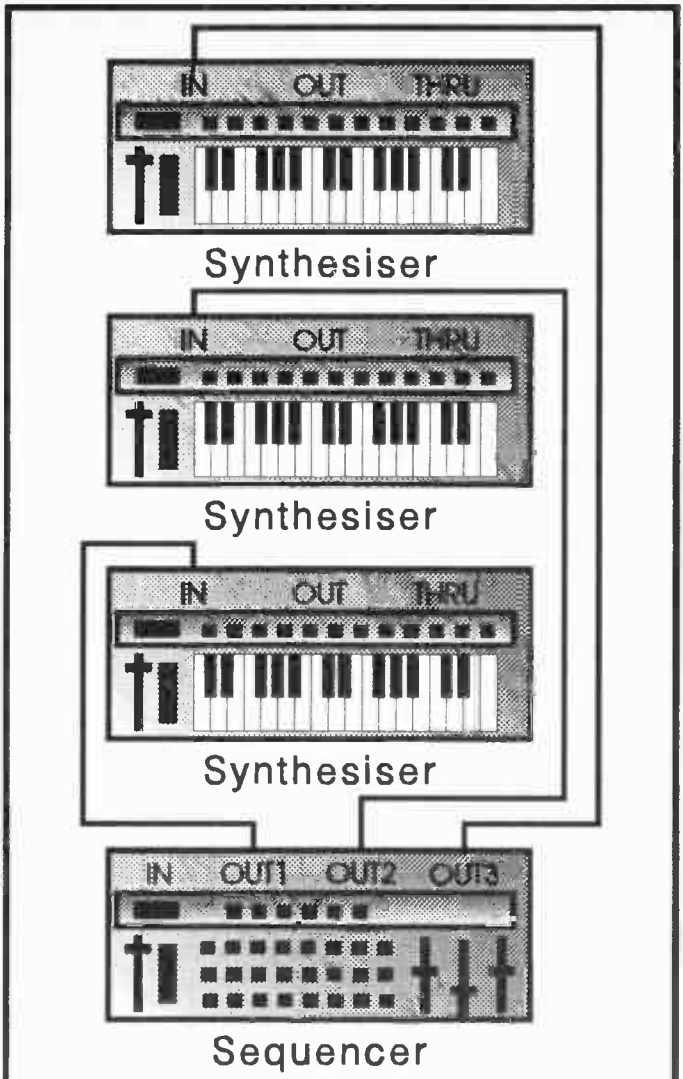


Fig. 1.9 A system based on a sequencer having multiple outputs

enables a series of numbers to be transferred from one device to another. However, the numbers are codes which carry useful information.

Any two MIDI units should work properly together as MIDI is fully standardised.

MIDI has facilities that permit sophisticated control of today's multi-timbral polyphonic instruments.

The interconnections are via cables fitted with 5 way (180 degree) DIN plugs. Ready-made MIDI leads should be available from any shop which sells electronic music equipment.

Opto-isolation at MIDI inputs helps to minimise problems with "hum" loops.

There are three types of MIDI port, which are the "IN", "OUT", and "THRU" varieties.

A THRU socket simply provides an output signal that is a replica of the signal received on the "IN" socket.

The two standard methods of interconnection are the "star" and "chain" types.

The chain method requires THRU sockets, the star system requires a controller having several outputs (or a THRU box).

It is acceptable to use a combination of the star and chain systems if desired, but there is normally no point in using anything other than the straightforward chain system.

MIDI devices can be slaved to only one controller unless a merge unit is used to combine the output signals from two or more controllers.

With a complex system a patch bay may be a worthwhile proposition, but multiple MIDI outputs probably offer greater versatility.

MIDI "delay" is unlikely to be a problem unless you have a vast number of instruments "chained" together.

Keep it simple!

Chapter 2

MIDI MODES

MIDI modes is a topic which seems to cause a fair amount of confusion, particularly for beginners. This is not an aspect of MIDI that you can ignore, even if you only intend to use MIDI in a fairly basic fashion. If you do not understand MIDI modes, then you are almost certain to run into difficulties sooner rather than later. Fortunately, MIDI modes are really quite straightforward, and are not difficult to master. Although MIDI modes may at first seem to be an unnecessary complication, they do add to MIDI's versatility.

Having different modes available makes it easier to use sophisticated units with much more basic ones, which these days generally means that it makes it easier to use modern equipment with early MIDI units. It also means that complex devices can effectively be downgraded to more simple units if you only need to use them in a fundamental fashion. This makes them easier and more straightforward to use. MIDI modes also enable sophisticated instruments in complex systems to be fully exploited.

With most users now having quite complex instruments which they use in sophisticated ways, it is probably fair to say that MIDI modes are rather less important than they once were. You may well end up always using your instruments in the same mode, and this mode may well be an unofficial mode! However, it is still useful to have the additional modes available in case you should need them.

Getting The Message

The only difference between one MIDI mode and another is the way in which channel messages are handled. There are two types of MIDI message, which are the channel and system types. System messages are directed to everything in the system. This is not to say that everything in the system will actually respond to a system message, but every device in the system should at least give it serious consideration. Each device will then act on the message if it is designed and set up

to do so, or ignore it if it is not. A channel message is different, and unless it is on a suitable channel, it will simply be discarded at an early stage by the "intelligence" in the control circuits of a MIDI device.

There are sixteen MIDI channels, which are simply numbered 1 to 16. These channels are to a large extent notional, as they are not carried by separate connecting wires. System messages and channel messages all go down the same pair of connecting wires. All that the basic hardware of a MIDI interface achieves is to send a series of numbers from one device to another. These numbers are coded in a rigidly standardised fashion, so that their meaning is correctly interpreted by any MIDI device that receives them.

System messages may actually be ignored by a device even if they are on the correct channel. Messages are ignored when a device simply does not have the ability to respond to the message correctly, the message is not relevant to that particular device, or because a unit has simply been set up by the user so that it will ignore that particular type of message. This selective ignorance set by the user is called MIDI filtering. It is important to realise that although MIDI has provision for a wide range of facilities, it is not a requirement that every piece of equipment should implement the full range.

Most MIDI messages are three numbers long, although they can be as short as one number, or in the case of one special type of message they can consist of a string of many thousands of numbers. The first number in a MIDI message indicates the message type (switch a note on, switch a note off, etc.). In the case of a system message that is all it indicates, but with channel messages it also indicates the MIDI channel number. Any gadget which receives a MIDI message therefore works out from the header number, the message type and, where appropriate, the channel number. In the case of a channel number, it responds to the message only if the message is on a suitable channel, and it is something appropriate to that particular device.

So why bother with the added complication of a chanelling system? In use each channel is normally used for a different instrument sound. With several instruments and

no system of channelling, they will all play very much the same thing. In most cases this is not what is required, and it is necessary to have each instrument playing a different part of the score. With a system of channelling, and each instrument or voice of an instrument on a separate channel, fully independent control of each instrument or voice is possible.

MIDI modes govern the way in which receiving devices respond to channel messages. You may sometimes encounter pieces of equipment or (more probably) MIDI computer software that enables a mode to be set when transmitting. This is not a strictly valid use of MIDI modes, and as such it is best to avoid using them in this way as far as possible. As the MIDI specification makes quite clear, MIDI modes only govern the way in which received messages are treated.

There are four MIDI modes, plus a sort of unofficial fifth mode. The names of the four official modes have been changed since the original MIDI specification was drawn up, which probably accounts for much of the confusion over MIDI modes. The confusion has been aided further by the use of mode numbers as well as the names. As the old names are still used by many MIDI users, the old names, the new names, and the mode numbers will all be given here. In general in this book though, only the mode numbers will be used, since these are now almost certainly the most popular method of differentiating between the various MIDI modes.

Mode 1
Omni On/Poly
Previously Omni Mode

This is the most basic of the modes, and if a unit is set to this mode it will respond to a suitable MIDI message on any of the sixteen channels. In effect, MIDI channels are totally ignored, and channel messages are treated as system types. The "omni on" part of the name indicates that channel numbers are ignored.

The "poly" part of the name indicates that this mode supports polyphonic operation. In other words, there can be several notes playing at once. An important point to note here is that MIDI does not lay down any minimum or maximum

number of notes that instruments should be able to play simultaneously. This is something that will vary from one instrument to another, but most polyphonic instruments can accommodate at least six notes at once. Some can play 32 or more notes simultaneously. The instruction manuals for your instruments should indicate the polyphonic capabilities of your instruments. It is up to you to determine the limitations of your instruments and to ensure that they are not exceeded.

Mode 1 is intended to be a sort of universal mode that all MIDI devices should have, whereas all the other modes are optional. The idea of this mode is that it enables any MIDI device to operate with any other MIDI unit. Devices in this mode can only operate in a rather basic fashion, but they can nevertheless provide a useful level of functionality. The original intention was that all MIDI units should have this as their default mode (i.e. it should be the mode adopted at switch-on). Some MIDI equipment still does, but it is increasingly common for MIDI units to include memory circuits that store all or most of the control settings at switch-off. With these units the start-up mode is usually whatever mode the device happened to be in last time it was switched off. Consequently, there is no longer any guarantee that a device will default to this mode at switch-on.

I suppose that these days mode 1 tends to be regarded as a relic of the past which is no longer of much practical value. It certainly lacks the capabilities of some other modes, but it does still have its uses. If you have problems getting devices in the system to "talk" to each other it is often useful to switch the receiving devices to mode 1. If the problem is simply due to a mismatch in the transmitting and receiving channels, then this will get the devices communicating. If not, then you know that there is a more complex problem to sort out.

Mode 1 is also useful if you only need to use MIDI equipment in a fairly basic way. For example, if you simply need to have one instrument slaved to another instrument, mode 1 may well suffice. However, even with a pretty fundamental setup of this type you might still find that mode 1 is inadequate. One possible problem is where the master instrument has split keyboard operation. In other words, the top section

of the keyboard operates on one MIDI channel, while the lower section operates on a separate channel. In fact some keyboards can be "split" into three or more sections with each section operating on a different channel.

Clearly a slave unit operating in mode 1 will simply ignore the keyboard splits, and treat all incoming messages the same. This might be adequate for your requirements, or you might need a more sophisticated method of operating. Mode 1 will suffice if you genuinely need to use MIDI in a very basic way, but if you need to do anything even slightly clever, one of the more advanced modes will be needed.

Mode 2
Omni On/Mono
No Previous Name

This mode is little used in practice, and is not really very useful. Like mode 1, it has "omni on", and consequently this mode does not implement channelling. The "mono" part of the name indicates that only monophonic operation is permitted. Monophonic operation means that only one note at a time can be played. With MIDI channels ignored and only monophonic operation permitted, this mode is, to say the least, very limiting. In fact it would seem to be of no practical value at all.

It was probably included in the MIDI specification to accommodate analogue synthesisers, most of which provided only monophonic operation (and few of which were MIDI equipped). Such instruments could not provide true mode 1 operation as they lacked polyphonic operation. They would therefore default to what was effectively mode 2 instead. Many modern instruments can be set for operation on mode 2, but there would seem to be no point in using them in this mode.

Mode 3
Omni Off/Poly
Previously Poly Mode

Mode 3 is a powerful mode, and is actually the most powerful of the official modes. The "omni off" part of the name indicates that this mode does implement channelling.

Consequently, a device in mode 3 must be set for operation on a particular channel, and it will then only respond to channel messages that are on that channel. If no channel has been set by the user, the device will operate on the default channel. This is usually channel 1, but with many modern instruments it will be the channel stored in memory on the last occasion that the device was used in an omni off mode. As indicated by the "poly" part of the name, this mode provides polyphonic operation.

The reason for mode 3's power is that it enables a sequencer to independently sequence up to sixteen polyphonic instruments (one on each MIDI channel). It provides the user with what is effectively a MIDI controlled band, or even a full symphony orchestra if that is what you want. Even with just a single MIDI output this mode is potentially so powerful that a tremendous amount of skill is needed in order to fully exploit it. Multiple MIDI outputs and mode 3 enables 32 or more polyphonic instruments to be individually controlled, which should be sufficient to handle any "real world" music.

Mode 3 is clearly one that should be given serious consideration if you need to produce complex music using a MIDI system. It can also be very useful in more simple setups though. Suppose that you have two synthesisers with one acting as a slave to the other. The master instrument could have split keyboard operation, with the lower section on channel 1 and the upper section on channel 2. If you wanted the slave instrument to follow only the part played on the lower section of the keyboard, playing a double bass sound perhaps, it would merely be necessary to switch it to mode 3 operation on channel 1. It would then ignore the notes played on the upper section of the keyboard as these would be on channel 2. Of course, if desired you could have a third instrument to follow the upper part of the keyboard. It would be set to mode 3 operation on channel 2. Although MIDI in general (and the more complex modes in particular) tend to be associated with sequencing, they are also very useful for "live" performances.

Most modern instruments can be set for mode 3 operation on any desired channel. If you will be using older instruments

it is advisable to check the MIDI implementation chart, since early MIDI equipped instruments are often far less accommodating. In fact many had a "Ford" choice, whereby you could use the instruments on any channel you liked, provided it was channel 1! Having one instrument of this type in a system is not a major limitation, since the other devices can be switched to operate on channels other than channel 1. Having two or more units that will only operate on channel 1 is obviously a more serious limitation.

Mode 4
Omni Off/Mono
Previously Mono Mode

This mode is still often referred to by its old "mono" mode name. It tends to be regarded by many as the most powerful of the official MIDI modes, and in the past it has been very popular with sequencer users. Strictly speaking though, it is not as powerful as mode 3. As the omni off part of the name indicates, this mode implements channel numbers. As the mono part of the name indicates, it only provides monophonic operation.

On the face of it, the monophonic limitation renders this mode of little practical use. In reality it is a very useful mode, because it is only monophonic in the sense that each channel is limited to playing one note at a time. The all-important point to bear in mind is that an instrument in mode 4 can operate on the basis of having its voices spread across several consecutive channels. An instrument which provides eight note polyphony in mode 3, may well give up to eight channel operation on mode 4, with a different voice on each channel, and simultaneous monophonic operation on each channel. In both cases the instrument is providing eight note polyphonic operation, albeit in rather different ways. Some modern instruments can provide a different voice on each of the sixteen MIDI channels.

The power of this mode stems from the fact that you can have what is effectively a different instrument on each channel. You can have (say) a piano sound on channel 1, an organ sound on channel 2, a guitar sound on channel 3, a flute sound on channel 4, and so on. The ability of an instrument to

provide several different sounds at once is known as multi-timbral operation.

Mode 4 is more limiting than mode 3 because it permits only monophonic operation on each channel, whereas mode 3 provides full polyphonic operation. On the other hand, few MIDI users can afford to work on the basis of one polyphonic instrument on each of the sixteen channels, or anything approaching this. Electronic musical instruments are relatively cheap these days, but a bank of a dozen or so assorted synthesisers and sound samplers would still cost a great deal of money. Certainly the cost of such a setup is well beyond anything most MIDI users could give serious consideration to. Mode 4 enables each MIDI channel to be occupied by what is effectively a different instrument, but at a vastly lower cost, provided you can accept the limitation of monophonic operation on each channel.

An important point to remember when dealing with MIDI modes is that there is no rule which states that all the instruments in the system have to be set to the same mode. Indeed, in most cases a mixture of modes is likely to give the best results. In particular, for sequencer users a mixture of modes 3 and 4 is likely to provide the greatest versatility. If a piece of music has a polyphonic part, perhaps for a piano or organ sound, then that part would be provided by an instrument set to mode 3, and operating on its own MIDI channel. If the accompaniment consisted of six monophonic parts, then these could be provided by a suitable instrument set to mode 4. A single instrument set for mode 3 operation plus one set to mode 4 is quite a potent combination, and one which can accommodate some very effective sequences. If resources will stretch to a couple of mode 3 instruments and a couple more operating in mode 4, so much the better. With a setup of this type controlled by a good sequencer you can do virtually anything you like.

Mode 4 is popular with guitarists. With a guitar synthesiser in mode 4 each string can operate on a separate channel, giving much greater versatility. Pitch bend, etc., can be applied to each string individually. It is perhaps worth mentioning here that it is a good idea to check the "fine print" in MIDI implementation charts for any instruments which you intend

to use in mode 4. Particularly with older instruments, you sometimes find that an instrument supports mode 4, but that certain types of channel message affect all the channels. Pitch bend and overall (channel) aftertouch are the two types of message which are most likely to operate in what is effectively "omni on" while everything else operates in "omni off". Strictly speaking, this method of operation is not within the MIDI specification, but I suppose that it is much better to have mode 4 with a few strings attached than it is to have no mode 4 available at all.

With mode 4 operation it is possible to set the base channel to any sensible channel. For instance, with an instrument set for operation on eight channels, you could set the base channel to channel 5. The instrument would then operate on channels 5 to 12. Setting the base channel to channel 10 would not be possible, since this would make the highest channel number 17. With most instruments it is not essential to assign all the voices to channels, and an eight channel instrument can, for instance, be set to occupy only four MIDI channels.

Multi-modes

There is a curious omission in the MIDI 1.0 specification in that there is no fifth mode called something like "omni off/multi". Such a mode would permit an instrument to operate polyphonically on up to sixteen channels, with a different voice on each channel. A sort of polyphonic version of mode 4. As such, it would be the most powerful MIDI mode. Probably the reason that such a mode was not included was that the instruments of the time (around 1980 to 1981) were very crude by modern standards, and an instrument that could make reasonable use of an "omni off/multi" mode probably did not exist. In any event, such an instrument was not the type of thing you would find in the music shops of the day, and there seemed to be no prospect of such instruments coming along in the near future.

Even so, it seems strange not to have included such a mode from the start as it would have helped to "future-proof" MIDI. Also, logically there would seem to be a good argument for such a mode in order to keep things tidy and complete.

As more and more powerful electronic musical instruments have been developed, the original four MIDI modes have gradually become inadequate. With even a 16-note polyphonic, multi-timbral instrument, modes 3 and 4 do not permit the unit to be fully exploited. With a few instruments starting to offer around 32-note polyphony and 16-voice multi-timbral operation, modes 3 and 4 have become very wasteful indeed. With such an instrument mode 3 gives 32-note polyphony on one channel, but how often would 32 notes at once really be needed? Mode 4 on the other hand, gives monophonic operation on sixteen channels, which means that no more than 16 notes at a time can be played, and that half the instrument's capacity is wasted. It would often be much better if the instrument could be used on the basis of something like 2-note polyphony on 16 channels, or 4-note polyphony on eight channels.

Today, even some of the less expensive instruments would be unnecessarily limited if they were restricted to operation on the four standard MIDI modes. Where something beyond modes 3 and 4 is required, manufacturers give their MIDI devices special modes which have been known by various names in the past. They are generally known as "multi-modes" these days, and are all basically the same. They are essentially variations on the missing "omni off/multi" mode, or mode 5 as it would have been if it had ever been officially implemented. Virtually all modern electronic musical instruments have some form of multi-mode. It has to be emphasised that these modes are not covered by the MIDI standard, and that in points of detail these modes vary somewhat from one instrument to another.

With the more simple multi-modes you have (say) an 8-note polyphonic instrument which can operate with 4-note polyphony on two channels, or 2-note polyphony on four channels. There may be a mode which offers something along the lines of 5-note polyphony on one channel, plus monophonic operation on another three channels. The more options that are available, the more efficiently the voices of the instrument can be exploited.

Although these modes might seem to be outside the MIDI specification, this is not really the case. In effect, an

instrument using a multi-mode is two or more instruments in mode 3. These instruments are actually in the same box, but from the MIDI point of view it does not matter whether everything is in one box or each channel has its own separate "box of tricks". The end result is exactly the same either way. Incidentally, a channel of a multi-mode instrument is sometimes referred to as a "virtual" instrument.

The best multi-modes are the ones which have dynamic note allocation. Suppose that an instrument has 20-note polyphony and 16-voice multi-timbral operation. With dynamic voice allocation you could have up to 20-note polypony on each of the 16 voices, but there would also be an overall limit of 20-note polyphony. Obviously it would not be possible to have 16 notes playing on one voice/channel with 16 notes playing on another voice/channel at the same time. This would be 32-note polyphony, which would exceed the 20-note polyphony capability of the instrument.

On the other hand, you could have 16 notes playing on one voice/channel, followed immediately by 16 notes on another voice/channel. In fact this would leave a spare capacity of four notes which could be used on other channels. With dynamic note allocation the control circuits of the instrument automatically allocate notes to the sound generator circuits of the instrument, and make adjustments to these circuits, so that optimum use of the instrument can be made without the user having to implement any quick mode changes. The user only has to set the required sound for each MIDI channel, and ensure that no attempt is made to play too many notes at once. The control circuits of the instrument make sure that everything is sorted out correctly inside the instrument.

For complex sequencing work a multi-mode is obviously a great asset. Whether you have two instruments of modest capabilities, or several sophisticated instruments, multi-modes ensure that you can fully exploit the system. Remember that an instrument with a multi-mode can be used in conjunction with instruments that only support modes 3 and 4. If you are going to undertake complex sequencing work, the essential starting point is to give some careful thought to which instrument will play each track, and what mode should be used for each instrument.

Summary

There are 16 MIDI channels which are simply numbered 1 to 16.

The main purpose of MIDI channels is to permit several instruments (or voices of instruments) to be controlled independently. This permits extremely complex multi-part pieces of music to be produced using a sequencer and some MIDI equipped instruments.

MIDI modes govern the way in which received channel messages are handled. They are irrelevant to system messages.

It is incorrect to talk in terms of MIDI modes when dealing with the transmission of MIDI messages – they only apply to receiving devices.

Mode 1 is a sort of universal mode that enables any MIDI device to "talk" to any other MIDI device.

Mode 2 is of no practical value these days.

Mode 3 is a powerful mode which implements channelling and provides polyphonic operation.

Mode 4 enables an instrument to operate with a different voice on each of several MIDI channels. Only monophonic operation is possible on each channel though.

Multi-modes are not covered by the MIDI specification. In effect, multi-modes provide a polyphonic version of mode 4. The exact capabilities of multi-modes vary from instrument to instrument.

An "omni on" mode ignores channel numbers.

An "omni off" mode implements channelling.

A "poly" mode gives polyphonic operation on the selected channel.

A "mono" mode gives monophonic operation on the selected channel or channels.

You do not have to use all the instruments in the same mode. For sequencing work a combination of modes 3 and 4 is often the best choice. The exception is where multi-modes are available. These are usually the best choice for sequencing work.

Read the manuals for your instruments very carefully in order to determine the precise capabilities of any multi-modes that are available.

Multi-modes which have dynamic note allocation are the most powerful modes.

Chapter 3

MIDI MESSAGES

As explained in the previous chapters, basically all a MIDI interface does is to permit a series of numbers to be sent from the controller to one or more slave devices. A rigidly standardised system of coding is used to ensure that the numbers sent from the controller are properly interpreted by each slave unit in the system. It does not matter if every piece of equipment in your MIDI system was produced by a different manufacturer. It does not matter either if some pieces of equipment are old while others are the very latest units with much higher specifications. Any piece of equipment that has a MIDI interface should operate properly in conjunction with absolutely any other pieces of equipment that are fitted with MIDI interfaces.

This contrasts with the pre-MIDI methods of interfacing synthesisers, etc. Connecting together two synthesisers from different manufacturers often required several leads and some additional electronics (if it was possible at all). The degree of control over the slave instruments was very basic by MIDI standards. In most cases only simple on/off switching of notes was possible.

Virtual Universality

Although any MIDI equipped devices should function correctly together, there are a few provisos to bear in mind. Perhaps the most important point of all is that although MIDI enables a sophisticated degree of control over the slave devices, the MIDI specification does not state that every device should actually implement the full set of MIDI functions. Few (if any) real pieces of MIDI gear implement all the possible functions, and in many cases only a small fraction of the available instructions are implemented. In general, modern MIDI devices tend to have extensive MIDI implementations, whereas early pieces of MIDI gear usually have quite sparse implementations. In practice, this means that connecting together two MIDI equipped instruments might not give quite the desired result.

If the slave instrument does not support a certain feature, then clearly it can not be controlled via MIDI from the main instrument. Even if the slave unit does actually have the required feature, it might not be accessible via MIDI. It is important to study the MIDI implementation charts for your MIDI equipment. These charts show exactly which MIDI messages are supported, and which are not. Note that some messages may be available when a unit is sending MIDI messages, but might not be recognised when it is receiving them (or vice versa). Again, the MIDI implementation chart should show clearly which messages can be sent, and which are recognised when receiving. Read any "fine print" very carefully, as there are often a few explanatory notes which detail restrictions or peculiarities in the way that certain MIDI functions are handled. For more information about MIDI implementation charts (and interpreting them) refer to chapter 5 of this publication, where they are considered in detail.

Another possible cause of problems is where two pieces of equipment have a degree of incompatibility that is not MIDI related, or is only partially associated with MIDI. These days there is more to MIDI than simply connecting together two or three instruments and a drum machine, or a sequencer and a couple of instruments. MIDI equipped devices which are currently available include such things as digital effects units, lighting controllers, and audio mixers. There are also one or two more obscure pieces of kit which can be wired into your MIDI system.

The fact that two devices have MIDI interfaces does not necessarily mean that they can usefully swap information via MIDI. The controller must be able to produce the messages that the slave unit requires before any sort of control will be possible. Some of the non-instrument MIDI devices can be controlled using the kinds of message that any MIDI controller can generate, while others are much less accommodating and difficult to control. Even if the right messages can be sent, it is still not completely certain that the slave device can be controlled in a fashion that will be genuinely useful in practice.

To avoid incompatibility problems of these types it is essential to give some careful thought about how you will actually use any MIDI unit that is in any way out of the ordinary. Study the MIDI implementation charts for all the equipment, and the manuals in general, to ascertain whether or not the equipment will really work together as a proper system. Make sure that any expensive MIDI accessory will work properly in your system before you buy it. Do not actually part with any money unless and until you are sure that the unit is properly compatible with the rest of your MIDI system.

Virtual Exclusivity

Although MIDI message codes are rigidly standardised, there is an "escape clause" which enables manufacturers to implement any special features which can not be properly accommodated by the standard MIDI messages. These messages are called system exclusive types, and they are something that we will consider in more detail in the next chapter. With one or two exceptions, you can not use system exclusive messages generated on equipment from one manufacturer to control units from other manufacturers. Indeed, in many cases these messages are specific to one particular instrument (or whatever), and will even be ignored by other pieces of gear from the same manufacturer.

There are two instances in which system exclusive messages might be usable between devices from different equipment producers. There is a form of system exclusive message called the universal type. The only commonly encountered universal system exclusive message is the sample dump type, which is used to send sound samples from one sampler to another, or from some form of data storage system to a sampler. This type of thing is not really applicable to synthesisers, since sound parameter data for one instrument is totally meaningless to most other synthesisers, due to their different methods of sound generation. It works quite well with samplers though, which all deal with much the same sort of data. A sound captured on one sampler is therefore readily interpreted by virtually any other sampler. It clearly makes

sense to have a standard method of exchanging sampled sounds via MIDI.

However, in practice by no means all manufacturers make use of the sample dump standard. Some use a method of swapping sample data that is based very closely on the sample dump standard, but which does not totally adhere to it. In these cases there are usually some additions to the basic sample dump standard, so that the samplers concerned will accept standard samples without any major problems. In most cases, any extra data they send will simply be ignored by an instrument which operates to the true sample dump standard, again giving no major problems when samples are exchanged. In my experience the main problem with swapping sound samples via MIDI is the slowness of MIDI hardware. Transferring large sound samples via MIDI seems to take around 10 to 30 minutes. If another method of transfer is possible it will almost certainly be much quicker to use this alternative.

The second way in which it is possible to use system exclusive messages between units from different manufacturers is where one of the units, in effect, mimics a device from another equipment producer. In practice this usually means that a computer running suitable software generates system exclusive messages and (or) can receive them. By using appropriate software a computer (or other microprocessor based controller) can successfully generate and receive any system exclusive messages.

An important part of the MIDI standard is a stipulation that any system exclusive coding must be published and made freely available to anyone who wants it. Furthermore, the coding can be used by anyone producing so-called third party MIDI equipment or software. System exclusive messages are therefore something less than totally exclusive. On the other hand, if you wish to make extensive use of system exclusive messages, it makes sense not to build up a system where every piece of equipment is from a different manufacturer.

If you use MIDI in a fairly basic way, with (say) a slave instrument being made to simply duplicate the notes played on the master instrument, there should be little difficulty. If

you start using more complex setups and practically every type of MIDI message that is available, you need to proceed a little more carefully. Make sure that the features you wish to use are actually supported by the equipment, and that they can be utilized via MIDI. Do the relevant checking before you buy new pieces of equipment, not afterwards.

MIDI Filters

There are two basic types of MIDI message, which are the channel and system types. A channel message is simply one that includes a channel number. A device in mode 3, mode 4, or a multi-mode (i.e. an "omni off" mode), will only respond to a channel message if it is on a suitable channel. A system message does not include a channel number, and any device in the system will respond to a message of this type. In both cases though, devices will only respond to a message if that message is included in their MIDI implementations.

Also bear in mind that most instruments have some built-in MIDI filters. This simply means that they can be set to ignore certain types of message, or set not to transmit certain MIDI messages. This can be very useful. For example, you might want to use pitch wheel modulation on the master instrument, but you might not want the slave instrument to respond to the pitch wheel. With suitable instruments this can be achieved by either having the master instrument not transmit pitch wheel change messages, or by having the slave instrument filter these messages out and ignore them. However, it means that you have to be careful to ensure that the messages you intend to use are not switched off or filtered out due to an error when setting up the equipment.

Channel Messages

The messages that are of most interest to the majority of users are the channel types. It is possibly true to say that most MIDI users do not use the system messages at all. In everyday use it is certainly the channel messages that the user is involved with for the majority of the time. In this chapter the channel messages are considered in detail. The system messages are described in Chapter 4.

Undoubtedly the most important messages of all are the note on and note off types. These messages are of the same basic form, and carry four pieces of information. The first of these is simply the message type (i.e. switch a note on or switch a note off). The next piece of information is the channel number. For all the channel messages this is a standard MIDI channel number in the range 1 to 16.

The next piece of data is the number of the note to be switched off. There are 128 note values from 0 to 127, and there is an increment by one semitone from one note value to the next. This gives a range of over ten octaves, which should be more than adequate for any practical purposes. It is substantially wider than the compass of a concert grand piano for example. In fact it is a wider range than most MIDI equipped instruments can actually handle. The manuals for your instruments should give details of their note ranges, and should also make it clear what happens if out-of-range note messages are received via the MIDI inputs.

There are two normal methods of handling out-of-range notes. Probably the most common method of all these days is for the right note to be played, but not in the right octave. Instead the instrument plays the note in the nearest octave that it can handle. This clearly has some detrimental effect on the reproduced music, but it minimises the problem of out-of-range notes. The main alternative is for any unplayable notes to simply be ignored, which is rather less satisfactory. For best results you should obviously be careful to arrange things so that no attempt is ever made to exceed the compasses of your instruments.

For the record, middle C is at a note value of 60. You will not normally need to get involved with actual note values. If you play a key of a MIDI keyboard instrument, the appropriate note value will be placed in the note on message that is generated, and the slave instrument will play the correct note. Some computer based sequencers enable you to edit the raw MIDI data if you wish, although there are usually more convenient ways of editing pieces. However, with a sequencer of this type it might be possible to work with note values if that is the way you would like to do things.

The final piece of information in note on and note off messages is the velocity value. This is only applicable to touch sensitive instruments. In other words, it only applies to instruments where playing a key lightly produces a quiet sound, and playing a key heavily produces a loud sound. In the early days of MIDI there were few keyboards that were touch sensitive. These days a new MIDI keyboard instrument would be unlikely to be taken very seriously if it did not have this feature.

However, it is only fair to point out that the touch sensitivity of many budget keyboard instruments is pretty basic. MIDI provides for up to 127 different degrees of loudness, plus silence with a velocity value of zero. This gives such a wide range of volume levels that the player gets the impression that there is an infinite range of volumes available. With many budget keyboards there only seems to be about half a dozen or so different velocity levels. An experienced player will soon realise that the instrument is not fully responding to his or her touch. A limited degree of touch sensitivity is still infinitely better than no touch sensitivity at all.

If an instrument does not support touch sensitivity it will still include velocity data in note on messages. The velocity value will always be the same value though, and is normally a middle value (usually 64). An instrument which does not support touch sensitivity will simply ignore velocity values in note on messages, and will play all notes at full volume.

Velocity values can be used in note off messages, and they then reflect the speed with which keys were released. In terms of the sound produced, a high velocity value could be used to give a more rapid cutoff than a low value, or something of this type. Features of this type do not seem to be used much in practice, if at all. In most cases the velocity value in transmitted note off messages is always zero. With most instruments the velocity values in received note off messages are simply ignored, and have no effect on the sound generator circuits.

Choked-up

There is a slight complication to the standard note on – note off scheme of things in that sequences can consist solely of

note on messages! A note on message which has a velocity value of zero acts as a note off message. The reason for including this facility seems to be little understood, even by those who are fairly expert in MIDI matters. I must admit that I was puzzled by this for many years until I eventually found an explanation tucked away in the MIDI 1.0 specification. It seems that with note on messages it is quite in order to have a complete message followed by sets of just note and velocity values. Although these follow-up messages are incomplete, MIDI equipment will interpret these pairs of data values correctly provided they are preceded by a proper note on message.

This makes it possible to have a long sequence of notes played on the slave instruments using just a single note on message from the controller. The appropriate note and velocity values are used to switch a note on – the same note value plus a velocity value of zero is used to switch the note off again. This may not seem to be of much practical significance, but it does have the advantage of enabling a given sequence of notes to be sent with a lower level of MIDI activity. This is important in modern MIDI systems where large amounts of information have to be sent at times. Due to the limited speed of a MIDI interface, there is a real danger of MIDI choke occurring. MIDI choke is simply where so much data must be sent, that the MIDI interface can not cope with it all.

Exactly how MIDI choke manifests itself in practice is something that varies from one system to another. In a worst case scenario the system could simply grind to a halt. This should not cause any damage to anything in the system, and simply switching off the controller, waiting a few seconds, and then switching on again should enable you to regain control of the system. More usually, MIDI choke will become apparent due to some notes failing to materialise, and (or) the timing of the notes becoming noticeably inaccurate. Anything that reduces the amount of MIDI activity without impairing a sequence in any way is obviously very worthwhile. In the past this facility to send long sequences using a single note on message has been little used. It is now being used in an increasing number of sequencers though, and with MIDI

systems becoming ever more complex, it is perhaps a more valuable feature now than it was a few years ago.

In reality the degree of streamlining it provides is often not as great as it might at first appear. The first part of a note on message contains the MIDI channel number. A long sequence of notes can be sent using a single note on message, but only if the sequence is on a single channel. Otherwise, a new note on message must be sent each time the sequence switches to a different channel. Despite this limitation it is still possible to obtain a worthwhile reduction in MIDI activity using this system. As far as the user is concerned, it makes little difference whether or not the MIDI controller uses this method of note switching. If it does, MIDI choke is less likely to be a problem, but it makes no difference to the way in which you use the system.

Beat Note

Drum machines produce percussive sounds that are mostly of indeterminate pitch. The same is true of the percussion channels of some MIDI sound generation modules and synthesisers. These could operate in mode 4 with one drum sound per MIDI channel, but this would be a very wasteful way of handling things. The drum sounds could easily use up half a dozen or more channels, which would probably leave too few channels for the other instruments.

The more usual approach, and a much more efficient one, is to have all the sounds on one MIDI channel, with each one activated by a different note. With this system you have what is typically about 60 or so percussion sounds available, with fifteen MIDI channels left free for the other instruments in the system. This system also permits easy control of the drum machine or drum channel using a MIDI keyboard or some form of sequencer.

Under Pressure

There are two normal types of touch sensitivity for keyboards. The most common variety is velocity sensing, where the loudness of a note is determined by how hard the relevant key is pressed. To be more precise, it is usually the speed at which the key is moving that controls the loudness of the

note, but in practice the harder you press a key, the faster the downward movement. This method of touch sensitivity, as explained previously, is handled by the velocity values in note on and note off messages.

The second type of touch sensitivity is called aftertouch. This controls the loudness of a note after its initial attack period. If you press the key harder the note gets louder – if you press the key less hard the note gets quieter. This type of touch sensitivity is far less common than the velocity type, and until recently very few instruments included any form of aftertouch in their specifications. In the last few years it has become much more common, and even a few low cost MIDI keyboard instruments have at least a basic form of aftertouch.

Aftertouch is not catered for in note on and note off messages, but is instead implemented by separate messages. This is really the only sensible way of handling things, since the aftertouch value may have to change dozens of times during the course of each note. Unlike velocity sensing, it does not match up neatly on a one value per note basis. The number of aftertouch messages per note will vary considerably, depending on how long each note is sustained, and possibly on how much variation in key pressure occurs during each note. Normally the initial attack phase of a note has to come to an end before the first aftertouch message will be sent. This means that a key has to be pressed for typically about half a second before an aftertouch message will be sent.

It must also be pointed out that aftertouch is not applicable to all types of sound. If an instrument is producing a type of sound that sustains properly when a key is held down (something like a woodwind sound for instance), then aftertouch is applicable. If a short percussive sound, such as a harp type sound is being produced, each note will tend to die away quite rapidly, making aftertouch irrelevant. It should perhaps be pointed out that aftertouch is not included in instruments as an alternative to velocity sensitivity. If an instrument has aftertouch, then it will certainly have velocity sensing as well (but one with velocity sensing will not necessarily have some form of aftertouch). With a harp type sound the aftertouch may be ineffective, but the loudness of each note would be

controlled by velocity sensing, so that a good degree of expression would still be available.

Two-touch

MIDI caters for two types of aftertouch. The more sophisticated of the two is polyphonic aftertouch. This is a highly desirable feature, but it is still something of a rarity. However, it is much less rare than it was a few years ago, and it is now even featured on a few low cost instruments. With this system there is separate aftertouch for each key. This produces MIDI messages that contain code numbers for the message type, the MIDI channel number, the key being played, and the pressure value. The key in question is indicated using the appropriate MIDI note value, and the pressure value is in the normal MIDI data range of 0 to 127. This gives a fine degree of control, and although the volume actually varies in steps, this should not be apparent to the listener.

There is a slight problem with polyphonic aftertouch in a MIDI context, in that it can generate large numbers of messages if a lot of sustained notes are played simultaneously. It is unlikely that polyphonic aftertouch on its own would generate so much MIDI activity that MIDI choke would become a problem, but it is probably best not to make extensive use of this feature at times when other facilities are generating large numbers of MIDI messages.

Overall aftertouch is the more simple variety, and is still probably the more common of the two types, although it seems likely that polyphonic aftertouch will soon be the norm. Overall aftertouch gives a sort of average key pressure for any keys that are being pressed. Changing the sustain level of one note therefore has the effect of changing the dynamics of that note, and any others on the same MIDI channel. Any notes on other MIDI channels should not be affected. This gives only a rather crude control over the dynamics of the notes, but it is still much better than having no aftertouch at all.

We have been discussing velocity sensitivity and aftertouch as they apply to keyboard instruments. For most users of MIDI systems the only way of playing music into the system is

via a piano style keyboard. There are other types of MIDI controller though, such as MIDI guitars and breath controllers. Also, MIDI systems can be controlled by step-time sequencers where notes are entered from a computer style keyboard using a simple method of coding. In the most up-market step-time sequencers the notes can be entered in standard music notation form, with the crotchets, dotted quavers, etc., being placed onto an on-screen stave.

Any form of MIDI controller could be used to control the dynamics of the slave instruments by generating suitable velocity values and aftertouch messages. Even with something like a step-time sequencer which has notation entry of notes, dynamic markings on the score could be converted into MIDI data which would control the dynamics of the instruments. This is not to say that all non-keyboard controllers do provide full dynamic control, but it is a feature of many.

Velocity sensitivity and aftertouch also apply to the increasingly popular MIDI expanders and other sound modules which can only be played via their MIDI interfaces, and which have no built-in keyboard. Such units will obviously not generate any velocity data or aftertouch messages, but many of them will respond to both of these. Of course, where any instrument does not support any form of aftertouch, it will simply ignore any aftertouch messages that it receives.

Pitch Bend

Most synthesisers (and many other electronic musical instruments) have a pitch wheel. Manipulating this outsize control knob produces corresponding variations in the pitch of the notes, or vibrato as it is usually termed. Pitch wheel variations can be sent via MIDI, and a very fine degree of control is possible. However, in order to provide very smooth pitch control it is necessary for large numbers of pitch change messages to be sent. This makes pitch bending a very good way of producing MIDI choke, and it is something that often has to be used in moderation if satisfactory results are to be obtained. With a lot of rapid pitch wheel changes you might find that the variations in pitch from the slave instruments are not as smooth as they might be. In fact you may well be able to hear definite glitches as the pitch occasionally jumps by

significant amounts.

Pitch bend messages contain the appropriate message code, the MIDI channel number, and two pieces of data which together provide a high resolution pitch wheel value. This means that if a series of pitch bend messages are sent on (say) channel 3, any notes being played on that channel will be affected by these messages. There is no way of applying the pitch bend to just one particular note on the channel.

In Control

MIDI has provision for a number of general purpose controls. The control change messages consist of four sections. The first is simply the control change code. The other three are the channel number, the control number, and the new data value for the control (i.e. the new control setting). There are 128 different controls, and these are numbered from 0 to 127. Each control has data value in the range 0 to 127, apart from switch type controls (as will be explained in more detail later).

There are a few complications when dealing with MIDI controls. Unfortunately, these are not of purely academic importance, and it is more than a little useful to understand the basic organisation of the control numbers if you are going to make even occasional use of them. Control numbers from 0 to 63 are all continuous controls. All this means is that they are used for controlling things that require a range of settings, rather than simple on/off switching. In other words, controls such as volume, tone, modulation depth, etc., that would conventionally be altered by means of control knob.

Continuous controls are something less than totally straightforward as they are used in pairs. Control numbers from 0 to 31 are paired with control numbers from 32 to 63 respectively. Table 1 should help to clarify the way in which this system of pairing operates. The lower control numbers are used for coarse control, while the higher number controls are used for fine adjustment. For example, conventionally control number 7 is the master volume control. Therefore, coarse control of the overall volume would be achieved by sending new values to this control number (zero for minimum volume through to 127 for maximum volume). If fine control of the volume is needed, then the data values would be sent to

control number 39 (the one paired with control number 7).

In practice this pairing of controls seems to be little used, since the 128 different settings provided by the coarse controls is usually quite sufficient. In fact some MIDI devices seem to use less than the full resolution provided by a single control. There is a problem in using very fine resolution in that it requires a large number of MIDI messages to be sent in a short space of time. Like pitch bending, this can easily result in MIDI choke. It would be wrong to suggest that no "real world" instruments use the pairing system, but it is certainly something of a rarity. It was probably more common in the early days of MIDI than it is at present. Anyway, reference to the MIDI implementation charts for your MIDI gear should soon reveal whether or not any pairs of controls are used. With luck you will be able to forget the pairing system, with some of the controls from 0 to 31 being used for various purposes, and the controls from 32 to 63 simply being left totally unused.

Although most of the continuous controls operate on the basis of 0 for minimum and 127 for maximum, there are a couple of exceptions. These are control numbers 8 and 10, which are respectively assigned to pan and balance functions. These both operate on the basis of having a value of 64 for the central balanced setting. A value of 0 produces full left hand volume, while a value of 127 gives full right hand volume.

Table 1

Coarse Control No.	*Fine Control No.*
0	32
1	33
2	34
3	35
4	36
.....	.....
27	59
28	60
29	61
30	62
31	63

Switched On

Switch controls are ones which only have on and off settings, and which are used for such things as switching effects on and off. The switch type controls are the ones which have control numbers from 64 upwards. However, as we shall see later, some of these are actually used as special forms of continuous control. Also, some of these controls have special functions or, as yet, no function at all.

The original MIDI specification only required devices to recognise 0 and 127 as valid data values for switch type controls. As one would expect, 0 is used to turn a control off, and 127 is used to switch a control on. Data values other than 0 and 127 should simply be ignored. This scheme of things has been slightly modified in recent up-dates to the MIDI specification. It is now a requirement that data values from 0 to 63 are recognised as "off", while those from 64 to 127 should be recognised as "on". Recently produced instruments should adhere to this new standard, but there are still vast numbers of instruments in circulation that will only accept data values of 0 and 127 for switch type controls.

MIDI controls will often be duplicating functions that are available via the front panel controls. The most obvious example of this is the MIDI master volume control, which will almost invariably be duplicating a function which is available from a conventional volume control on the instrument. The way in which this doubling-up is normally handled is by having the maximum volume controlled by the MIDI control. The front panel control then permits the volume to be set anywhere from zero to the maximum level set by the MIDI master volume control value. The output level from the instrument is therefore a function of both control settings.

The situation is different with most other controls though. Setting other parameters via the front panel controls usually requires a great deal of button pushing, and frequent references to some form of display panel. Once you have finally set the control at the required value, what you have normally done is to set a value stored in the instrument's memory circuit. Altering control settings via MIDI control messages usually gives the same result, with the appropriate memory cell having its value altered to suit the new control setting.

The MIDI controls and the front panel controls then provide different means of achieving the same ends.

It is important to bear in mind that altering the front panel controls of an instrument usually results in a corresponding MIDI messages being transmitted. In the right circumstances this can be very useful, and it obviously provides an easy means of controlling slave units from the master instrument. On the other hand, if you are not careful it is easy to inadvertently alter numerous settings on one or more slave instruments. It could take a long time to unscramble all the slave unit control settings and get things back to where you want them! There should be a control which enables MIDI transmission to be switched off, or possibly a switch that enables MIDI to be switched off completely. Remember to switch off MIDI transmission when you do not require the slave units to follow changes made to the master unit. There should be no problems of this type with the slave units, as their MIDI OUT sockets will not normally be connected to anything.

Assignments

One of the few major shortcomings of the original MIDI 1.0 specification was that it did not lay down definite functions for a range of MIDI controls. As will be explained later, special functions were assigned to control numbers 121 to 127, which effectively became a different form of MIDI message. Apart from this, the only assignment was in a pre-MIDI 1.0 specification which had control number 0 as the pitch wheel control. This was dropped by the time the MIDI 1.0 specification was agreed and published, with pitch wheel modulation being given its own MIDI message type.

The result of this is that there is no true standardisation of MIDI control functions. Some conventions became generally accepted, and these seem to be largely based on the Yamaha DX7 synthesiser which was extremely popular some years ago. Eventually some control assignments were officially laid down in the detailed MIDI specification. However, you can not rely on MIDI equipment strictly adhering to these assignments.

There are two main reasons for this. One is simply that there are large numbers of MIDI devices in current use which

predate the official assignments. An older instrument will not necessarily use a non-standard method of assignment, but it is quite likely to have at least one or two controls which do not follow the current specification. Some instruments actually had assignable controls, where any control number could be assigned to any MIDI controllable feature of the instrument. With a unit of this type it would presumably be possible to set it up so that it would conform to the current scheme of things. However, its default settings are unlikely to conform to the standard assignments.

Another potential cause of non-conformity is that the MIDI specification provides a sort of "get out clause". The specified functions are all concerned with musical effects and general sound control. MIDI, on the other hand, is currently used to control all manner of equipment, such as lighting controllers. Manufacturers of non-musical MIDI equipment can legitimately use the MIDI controls in any way they like, although common sense would dictate that they should try to avoid using the more commonly used control numbers in non-standard ways.

Table 2 gives details of the current assignments for MIDI control numbers. At least, it gives the functions for those control numbers which have so far been assigned to specific functions. A number of control numbers are so far assigned.

Table 2

Control No.	*Function*
0	Bank select
1	Modulation wheel
2	Breath controller
4	Foot controller
6	Data entry (msb)
7	Main volume
8	Balance
10	Panning
11	Expression Controller
12–13	Effect control
16–19	General purpose

Table 2 (Continued)

Control No.	*Function*
64	Damper pedal
65	Portamento
66	Sostenuto
67	Soft pedal
68	Legato foot switch
69	Hold 2
70–79	Sound controllers
80–83	General purpose
91–95	Effects depth 1–5
96	Data increment
97	Data decrement
98	Non-registered parameter number (lsb)
99	Non-registered parameter number (msb)
100	Registered parameter number (lsb)
101	Registered parameter number (msb)
120	All sound off
121–127	Channel mode messages

Remember that continuous controls from 0 to 31 are paired with control numbers from 32 to 63 respectively (only the lower continuous control numbers are included in this table).

If you are going to make use of MIDI controls you really need to study the MIDI implementation charts for your equipment very closely. These should show whether or not each device conforms to the standard. Also, they will show which particular control numbers are actually utilized by each unit. Some instruments make extensive use of MIDI controls while others implement only master volume and perhaps one or two others.

Sound Control

The standard MIDI controls should not be used for adjustments to the sound generator circuits of an instrument. For example, they should not be used for controlling such things as the attack time of an envelope shaper, or the cutoff frequency of a filter. However, there are some special control

numbers which can be used for adjustments to the sound generator circuits. There are only a few control numbers available for sound generator control, but there could be hundreds or even thousands of parameters to adjust.

The way around this problem is to have two controls which are used to select the parameter that you wish to adjust. A further two controls are then used to actually adjust the parameter. Control numbers 98 and 99 are used to select the desired parameter. Using one control to select parameters would only permit up to 128 different parameters to be adjusted. Using two controls for this purpose enables up to 128 x 128 (16384) different parameters to be selected, which should be more than would ever be needed in practice. The parameters are adjusted using controls 6 (coarse) and 38 (fine).

This gives control of what are termed the non-registered parameters. These are non-standardised types where each control can be whatever the equipment manufacturer chooses. There are also registered parameters, which are rigidly standardised. Having a set of standardised sound generator controls is clearly very desirable, since it enables a random collection of MIDI gear to operate together more effectively as a system. Standardisation to the highest degree was a main aim of MIDI when it was being formulated, and it is to a large extent this standardisation that led to MIDI becoming so popular. On the other hand, true standardisation is not really possible with the control of sound generator circuits because there are so many different forms of sound generation currently in use. A set of controls that are relevant to one instrument may well have no relevance at all to most other instruments. Registered parameters are therefore little used in practice, and as yet there are too few registered parameters for this system to be used to any great effect.

Anyway, registered parameters operate in the same basic way as the non-registered variety. The required parameter is selected using control numbers 100 and 101. The selected parameter is then adjusted, as before, using control numbers 6 and 38. So far only five registered parameters have been agreed, and these are detailed in Table 3. These are all of a general nature, and are not concerned with a specific method of sound generation.

Table 3

Registered Parameter No.	*Function*
0	Pitch bend sensitivity
1	Fine tuning
2	Coarse tuning
3	Change tuning program
4	Change tuning bank

It is only fair to point out that some manufacturers make extensive use of MIDI controls to permit adjustments to the sound generator circuits, while others largely ignore this route. The alternative method, and the one which now seems to be favoured by many manufacturers, is to use system exclusive messages. This might seem reasonable, since many of the parameters in question will be specific to the instruments of one manufacturer, and thus of a non-standard nature. It does mean though, that it is not possible to access the sound generator circuits of such an instrument via a general purpose controller, or a computer running a general purpose controller program. Access is only possible via a purpose built controller, or a computer running a program written specifically for use with the instrument in question.

Special Controls

As pointed out previously, some of the MIDI control numbers are assigned to special functions. Control numbers 121 to 127 have special purposes, as listed in Table 4. These are known as the channel mode messages, and are effectively a separate class of message, rather than just variations on normal MIDI controls.

The "reset all controls" message is presumably meant as a quick means of getting back to the default settings (i.e. the settings that the instrument had on leaving the factory). This could be useful if the control settings get well and truly scrambled, and it would be easier to start from scratch. This message differs from normal switch type MIDI controls in that the data value sent is always zero. This is true of most of the channel mode messages.

Table 4

Control No.	*Function*
121	Reset all controls
122	Local control
123	All notes off
124	Omni off
125	Omni on
126	Mono on
127	Poly on

Losing Control

The local control message does operate in the same fashion as an ordinary MIDI switch type control, but it is the only channel mode message that does so. Normally local control is switched on, and what this generally means in practice is that playing notes on the keyboard results in the relevant MIDI messages being generated at the MIDI OUT socket, and the appropriate notes being produced by the instrument's sound generator circuits. If local control is switched off, the appropriate MIDI signals will still appear at the OUT socket, but the sound generator circuits will not produce any notes. The sound generator circuits will respond normally to any MIDI messages received at the MIDI IN socket.

In effect, switching off local control converts an instrument into separate keyboard and sound generator units. In most setups there is probably little advantage in doing this, but there are some possible advantages. This facility is mainly used in real-time sequencing. Consider the basic real-time sequencer setup of Figure 3.1. This looks quite plausible, but there is a potential problem if the sequencer's THRU facility is used (or a merge unit is used to give the same facility). Anything played on the synthesiser's keyboard will produce MIDI messages that will be looped back to the synthesisers MIDI IN socket. This is undesirable, and in practice seems to result in many instruments playing each note twice (once due to the keyboard control, and once due to the messages received via MIDI). This may not seem to be

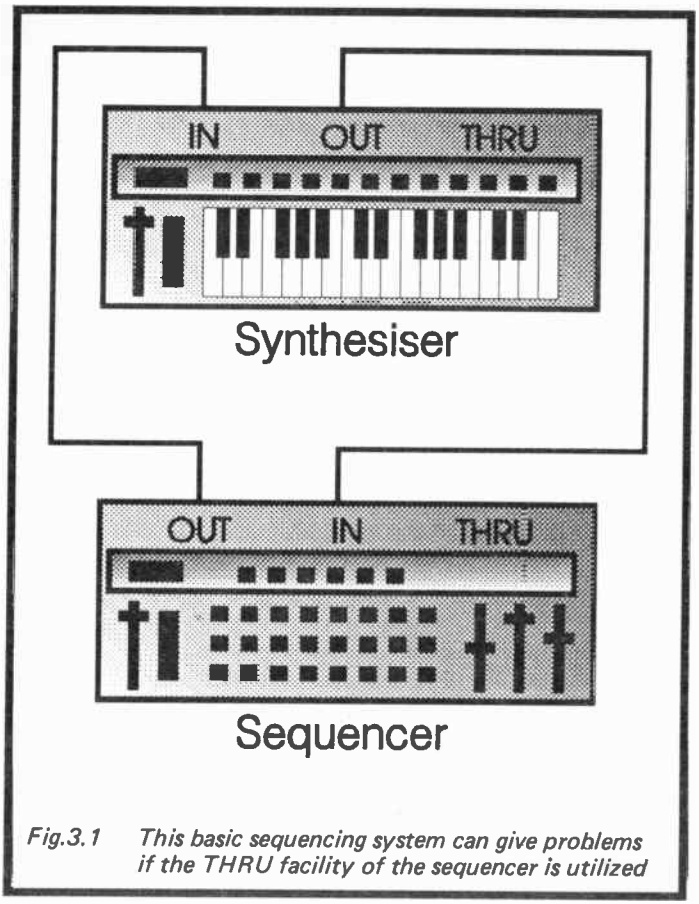


Fig.3.1 This basic sequencing system can give problems if the THRU facility of the sequencer is utilized

important, but with (say) an eight note polyphonic instrument, this effectively reduces it to a four note polyphonic type. Switching local control off effectively gives the setup of Figure 3.2, which clearly avoids problems with notes being played twice.

Although I have been talking here in terms of a keyboard as the normal method of local control, local on/off switching can be applied to MIDI guitars, etc. It obviously does not

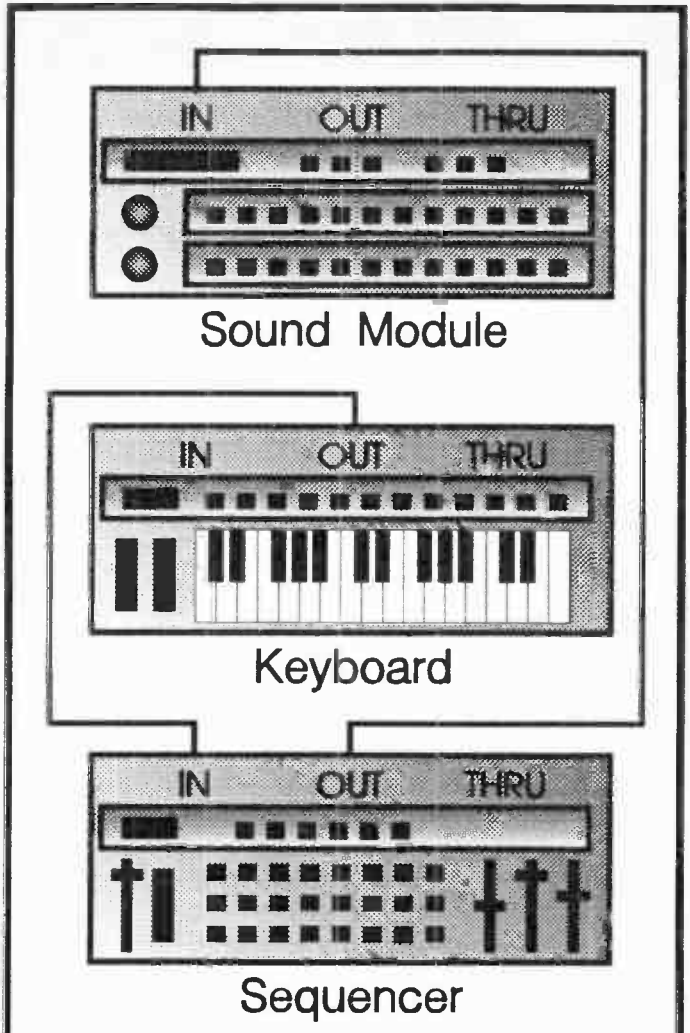


Fig.3.2 Using "local off" effectively splits the sequencer into a separate keyboard and sound module

apply to any form of MIDI sound module which does not have any form of local control to switch off.

The all notes off message is not intended to be used as a normal method of switching off notes. Any note on messages must be followed by corresponding note off messages in due course. It is presumably intended as a sort of last resort method of silencing the sound generator circuits if a fault should occur and notes are left switched on.

Mode switching is accomplished using the remaining four channel mode messages. These do not operate on the basis of one message per mode, but instead they must be used to select the desired combination of omni, poly, and mono. When using these messages it is clearly best to think in terms of the current mode names, rather than the old names or mode numbers (e.g. omni off – mono rather than mode 4 or mono mode). It should then be obvious which messages are needed in order to select the required mode. For example, omni off – poly (mode 3 or poly mode) would obviously require controls 124 (omni off) and 127 (poly on) to be used. Note that the mono mode control (126) can include a data value of other than 0, and that the data value sets the number of channels that are switched to mono mode. A value of 0 sets all the available channels to mono mode. Note also that all mode change messages provide an "all notes off" action, like control number 121.

All Change

The seventh and final type of channel message is the program change variety. In this context a "program" is actually a set sound generator data. This message therefore enables an instrument to be switched between a number of different sounds. For example, you might wish to switch from program 27 (a trombone type sound) to program 45 (a flute-like sound). This message carries a data value in the range 0 to 127, which enables an instrument to be switched between up to 128 different sounds. With an older instrument you might find that the full 128 sounds are not available, for the very good reason that the instrument's memory circuits only have provision for (say) 64 different sounds. With recent instruments though, the full complement of 128 sounds is almost

certain to be available. The only common exception is sound samplers, where massive amounts of memory can be required for each sound. With these instruments there are usually only a few sounds loaded at any one time.

Although in MIDI terms the program numbers are from 0 to 127, some other method of program numbering may be used by the equipment manufacturer. For example, when selecting programs using the front panel controls you might have to operate on the basis of program numbers from 1 to 128, or even something like A – 1 to P – 8. The equipment manuals should make the relationship between the manufacturer's program numbers and the MIDI program numbers perfectly clear. There may even be a simple conversion chart. This is by no means of purely academic importance. Differences in the methods of program numbering could easily result in the wrong programs being selected on the slave instruments unless you are aware of the differences, and make allowances for them.

If used sensibly, program change messages offer a means of fully exploiting synthesisers. Multi-part music can be problematic using a single MIDI output since there is a limitation of sixteen channels. Obviously in many cases something along the line of fifteen different instrument sounds and one channel for the percussion sounds will be more than adequate. However, in some cases it will prove to be inadequate. Being realistic about it, it is probably fair to say that few musicians will need more than fifteen instruments at once plus a percussion section. On the other hand, many MIDI users require more than fifteen different instrument sounds (plus percussion) during the course of a piece. On the face of it, MIDI can not accommodate such a piece, unless two or more separate MIDI outputs are available, effectively giving 32 or more channels.

Multiple MIDI outputs and banks of instruments certainly represents the best way of handling this problem, but it is a solution that few of us can ever hope to afford. Program changes offer a good low cost alternative. The basic idea is to have two sounds on one channel, and to switch between them using program change messages. For this to work properly it is necessary for the two sounds to be available on

the same channel, which in practice generally means that they must be available on the same instrument. Also, it is obviously necessary to pair sounds that will not be needed simultaneously. However, these are not really major limitations, and with a little thought it is usually possible to pair-up sounds in a way that will give the desired result.

You are not limited to two sounds per channel, and with carefully worked out program changes it is perfectly possible to have a dozen or more sounds per channel. In practice it is unlikely that this would ever be necessary, and it is probably best to keep things relatively simple and straightforward, but program changes together with suitable instruments offer tremendous versatility if you should ever need it.

Although program changes were originally intended as a means of switching synthesisers between their various sounds, they are now used as general purpose control messages. Some audio mixers for example, can be switched between various sets of control settings using program change messages. Some digital effects units can also be controlled in a similar fashion, and it is a method of control which could presumably be applied to virtually any MIDI device. It is a convenient method of control since most instruments and sequencers can easily provide program change messages.

Standard MIDI

When using program changes you can have any sounds assigned to any program numbers, within the limitations of the instruments you are using. For the average MIDI user there is probably no advantage in having a standard sound assigned to each program number. This would, to say the least, be rather limiting. On the other hand, there is a potential flaw in having no standard assignments. Suppose that you produce a piece of music using a MIDI sequencer, and that you send this sequence to someone so that they can play it back through their sequencer and MIDI system. Simply playing the sequence through their setup is unlikely to result in your masterpiece being reproduced correctly. Getting the correct result is dependent on having each MIDI channel occupied by the right instrument (or voice of an instrument) set up to give precisely the correct sound. Otherwise, the track that you intended to

be played using a grand piano sound could well be reproduced using a trumpet sound, or even as a series of percussion sounds!

Standard MIDI is a set of standard program number sound assignments. The general idea is that each sequence should start off with a series of program change messages. These set each channel to produce the appropriate sound. There is also a standard set of assignments for the percussion channel. The sequence should therefore be produced with the right sound on each track, and the reproduced music should be as the composer intended. Any program change messages within the sequence should produce a switch to the right sound, and should again give the right end result.

In reality there will be some variations in the sounds from one instrument to another. The saxophone sound from an FM synthesiser will be slightly different to that from an LA synthesiser, which will in turn be somewhat different to the saxophone sound from a sound sampler. Presumably the differences in most cases will be relatively minor though.

For those who need to produce sequences that others can play back correctly on their MIDI systems, standard MIDI is presumably the best thing since MIDI itself. Of course, for many users MIDI is an aid to "live" performances, or the finished result is an audio recording of a sequence. For such users standard MIDI is probably of little relevance.

Summary

The Channel Messages:

Note on
Note off
Pitch bend
Program change
Polyphonic aftertouch
Overall aftertouch
Control change (including MIDI mode messages)

In order to use MIDI effectively you must have at least a basic understanding of all the channel messages.

MIDI caters for a note range of over ten octaves, but most instruments can accommodate substantially less than the full range.

MIDI note on and note off messages can include touch sensitivity information in the form of a velocity value.

Aftertouch is catered for by separate messages, and MIDI supports two types. Polyphonic aftertouch gives individual control over the dynamics of each note. The overall variety applies a sort of average volume to every note playing on that channel.

Aftertouch is not applicable to every type of sound. In general, it applies to sounds which have a long sustain period, and does not apply to brief percussive sounds.

There is a MIDI message for pitch bending, and it provides very high resolution control. Very smooth pitch bending is possible, but produces a lot of MIDI activity. Pitch bending applies to all notes playing on the relevant channel.

There is a limit to the amount of information that can be sent via a MIDI link in a given period of time. Trying to exceed this limit produces MIDI "choke", which will usually become apparent due to poor timing accuracy.

Polyphonic aftertouch and pitch bending produce large numbers of messages, and are therefore likely causes of MIDI "choke".

Instruments can be switched from one sound to another using program change messages. Up to 128 different sounds can be used (or more if several banks of sounds are available).

Program change messages represent a very powerful facility, both for "live" performances and for sequencing work, but their use will often require some careful advance planning.

Standard MIDI is a set of standard sound assignments for the program numbers. The use of standard MIDI is advisable when swapping music in the form of MIDI sequences.

Program change messages are also used as a means of controlling MIDI devices other than instruments (mixers, effects units, lighting units, etc.).

MIDI controls provide a means of adjusting general parameters such as volume, balance, etc.

There are two basic types of MIDI control, which are the continuous type (0 – 63) and the switch type (64 to 127).

Although the continuous controls have provision for high resolution using a system of pairing, this facility is little used

in practice.

Where only low resolution control is required, controls 0 to 31 are used, and controls 32 to 63 are left unused.

Access to the sound generator circuits of an instrument is possible via four special controls. The required parameter is selected using controls 98 and 99. New values for the selected parameter are then sent to control numbers 6 and 38 (or just control number 6 if only low resolution is required).

Some instruments have extensive provision for control via control change messages. Others (including some sophisticated modern units) only implement two or three basic controls.

Control numbers 121 to 127 are used for a special category of control change message. These are called MIDI channel mode messages, and are effectively a different category of message.

MIDI channel mode messages are used for such things as mode changing, switching off all notes in an emergency, and local control switching.

Local control is a facility which should not be overlooked. It permits a keyboard instrument to effectively be converted into a separate keyboard and sound generator module.

Few (if any) MIDI devices implement the full set of MIDI channel messages. Read the MIDI implementation charts for your instruments very carefully to determine what can and can not be achieved via MIDI. MIDI provides excellent compatibility between devices of different types from different manufacturers, but you can not use facilities if the device in question does not support them!

Chapter 4

SYSTEM MESSAGES

System messages are undoubtedly an important form of MIDI message, and they are essential to many MIDI users. On the other hand, they are much less well used than channel messages, and not used at all by a fair proportion of MIDI users. They are probably of more importance to advanced users, particularly musicians who use MIDI professionally, than they are to beginners. In fact it is probably best for beginners to gain some experience with channel messages before delving too deeply into system messages, which tend to be less straightforward in use. It is only fair to point out that although some system messages are very useful indeed, others are little used and have no obvious practical value in a modern MIDI context. It is doubtful whether one or two of the system messages ever were of much practical use.

In this chapter we will consider most of the system messages in a reasonable amount of detail. The discussion of some messages will be something less than complete. This is simply because some of the system messages are quite involved, and really go well beyond the scope of a beginners MIDI book such as this.

Although system messages are directed at every piece of equipment in the system, it is important to realise that this does not mean that everything in the system will actually respond to them. Like channel messages, they can be implemented or omitted, as the equipment manufacturer chooses. It is a matter of checking the MIDI implementation chart to determine which system messages each piece of equipment implements. In many cases only a very few system messages are implemented.

Just In Time

Several of the system messages are concerned with timing, and keeping two or more systems properly synchronised. Having two MIDI systems running side-by-side and synchronised might seem to be doing things the hard way, and I suppose

that in many cases it would not be a worthwhile method of working. Where possible it is probably best to keep everything under the control of a single MIDI controller. With simple systems where there is something like a master instrument and one or two slaves, there is no problem in doing this. It can become more difficult with complex sequencer setups.

The most common reason for using the MIDI timing signals is that the system includes a drum machine which has a built-in sequencer. A sequence is entered into the drum machine, and the MIDI synchronisation signals are then used to keep this sequence (and hence the drum track) in time with the main sequencer. This method of working is perhaps less worthwhile than it once was. With any recent MIDI equipped drum machine there should be no difficulty in controlling the instrument directly from the main sequencer. The usual method is to have the drum machine operating in mode 3 on any spare channel. A range of drum sounds are then available, and are organised on the basis of one sound per note. This makes it very easy to control the drum machine from the sequencer. Many sequencers now have facilities which make it easier to lay down repetitive drum tracks, plus other facilities that make it easy to directly control the drum machine from the main sequencer.

One problem in using the drum machine's sequencer plus synchronisation to the main sequencer is that the main sequences are incomplete. They only provide the correct final output if they are used with a drum machine that is loaded with the correct sequence. This is not particularly convenient, and you have to be careful to work out a system that ensures you can properly match up the main and drum machine sequences. It makes life very difficult if you need to swap sequences with other people. They can only produce the full piece if they have a drum machine equipped with a suitable sequencer.

Synchronisation is also needed where two sequencers are used to produce a masterpiece that is simply too complex for either of the sequencers to handle it in isolation. Such a setup suffers from the same problem as a sequencer plus drum machine and integral sequencer, but more so. Few people are equipped to handle complex MIDI sequences that are in this

twin sequence format. A twin sequencer setup is not very convenient to use either. A single, more sophisticated sequencer having two or more separate MIDI outputs is probably a more practical choice than two sequencers of lesser capability.

Beat The Clock

The messages related to timing and synchronisation are known as "system real-time" messages. Note that only devices which have some form of built-in sequencer can make use of these messages. Consequently, most MIDI devices do not recognise any of these messages. The main timing message is the clock type, and this is a sort of MIDI version of a metronome beat. It is sent at regular intervals, and it is actually sent at a rate of 24 per quarter note. The slave sequencer follows these messages in a way that is analogous to a human player following the beat of a metronome.

In the pre-MIDI days drum machines were kept in synchronisation by a series of electrical pulses sent from the main sequencer. The MIDI clock signals are also analogous to these clock pulses. There is a major difference though, in that the old style drum machines were started and stopped by switching the clock signal on and off. MIDI uses a rather more sophisticated system. The slave sequencer is switched on and off by "start" and "stop" MIDI messages. The string of clock messages is only needed while the sequence is in progress, but it is perfectly acceptable for the clock signal to be sent continuously.

In addition to the "start" message there is a "continue" type. These are not simply the same message with different names, and they provide very different functions. A "start" message has the effect of starting the sequence "from the top". If a sequencer is stopped half way through a piece using a "stop" message, and then it is started again using a "start" message, the sequence will be taken right back to the beginning and it will proceed from there. A "continue" message will result in the sequence continuing from where it left off. If a sequence has yet to start, then a "continue" message would presumably result in it starting from the beginning. However, if a sequence must start "from the

top", it is advisable to play safe and use a "start" message.

There is a further system real-time message in the form of the song position pointer. In the present context the word "song" means a MIDI sequence. A MIDI sequencer must have a built-in counter so that it can keep track of its position within a sequence. The counter simply registers the number of MIDI clock periods that have elapsed since the start of the sequence. A song position pointer message will take the slave sequencer to any desired point in the sequence. The song position pointer message includes the new value for the clock counter, and this is in the range 0 to 16383. Where synchronisation is to be used, the maximum length for a sequence is therefore 16384 MIDI clock signals sent at a rate of 24 per quarter note.

Normally a song position pointer message is used in conjunction with a "continue" type. The song position pointer takes the sequencer to the desired place in the piece, and a "continue" message is then used to start the sequencer playing from that point. These days most sequencers will fairly rapidly move to the indicated point in the sequence, but some sequencers can take a few seconds to reach the desired point in a long and complex sequence. It is therefore advisable to have a reasonable delay between a song position pointer message and a continue type.

Making Sense

The other system messages are known as "system common" messages. One of these is used to implement active sensing, which is not a feature that is used much in practice. It is supported by some early MIDI instruments, but in later instruments it seemed to be very rare indeed. It has had something of a renaissance in recent times, but is still a far from universal feature. Even where it is available, it does not seem to be utilized very often.

The function of active sensing is to detect a fault in the system such as a damaged lead or socket. It then switches off all notes in the slave instruments so that no notes are left switched on. The basic idea is to have active sensing messages sent from the MIDI controller during periods of inactivity. To be more precise, they are inserted as and when

necessary, so that there is never a gap of more than 300 milliseconds (0.3 seconds) between MIDI messages. If a slave unit does not receive a message for more than 300 milliseconds, it will assume that a fault has occurred. It then switches off all its sound generator circuits.

Active sensing is not used by default. The MIDI controller will only send active sensing messages if you set it to the correct mode. Receiving units will not implement active sensing until they receive the first message of this type from the controlling device.

If you have instruments, etc., which support active sensing, then you might find it useful. It probably has most appeal to those who use MIDI for "live" performances, where a fault such as a broken lead could produce very embarrassing results. Active sensing could help to minimise the damage. However, initially you are probably best advised to ignore this complication.

Request Program

The song select message is a rather more useful one, and it can be used with sequencers that can be loaded with several sequences (songs). This message includes the number of the new sequence that is being selected. This number is in the range 0 to 127, which obviously means that this sytem can handle up to 128 different sequences. This is rather more than many sequencers can handle.

Tune request is another of the system common messages. This one is largely obsolete these days, and it is a message which you are unlikely to use in earnest. It should not be confused with the song select message. These are two different messages which have totally different functions. The word "tune" is used here in an instrument sense, and not the melodic sense.

It seems that this message was intended for use with the old analogue synthesisers which tended to slip out of tune, but had automatic retuning circuits that enabled them to be easily and quickly put back into tune. The idea was that if things became noticeably discordant, this message could be sent in order to force each instrument to retune itself. This presumably restored perfect harmony to the system.

Modern instruments operate in a totally different fashion, and they are usually controlled by quartz crystals (as used in electronic watches) which do not suffer from significant drifting. Consequently, unless you collect early MIDI instruments you are unlikely to encounter an instrument which supports this message.

System reset is another system common message which is of no real value these days. All it does is to take everything back to the start-up state, so that the instrument (or other device) is set exactly as it was at switch-on. As modern instruments mostly have memory circuits which result in them starting up more or less where they left off, this message has become largely irrelevant. It does not seem to be implemented on most modern instruments. If a modern unit did implement this message, its result would presumably be to return all the settings back to the default states set at the factory. If you should happen to have a MIDI controller that can generate this message, it is probably not a good idea to try it out to see what happens!

System Exclusive

System exclusive messages are the most important system common types, and are potentially one of the most useful of all MIDI messages. All the other MIDI messages are designed with rigid standardisation in mind. This ensures that there is no problem in building up an effective MIDI system using equipment from a variety of manufacturers. All MIDI equipment uses the same hardware interface, and the same method of message coding. Within the limitations of the individual pieces of equipment, everything is guaranteed to work together properly as an efficient system.

The problem with this approach is that it is rather limiting. If a manufacturer comes up with an innovative new feature it might not be possible to exploit it properly via MIDI. This could result in technical developments leaving MIDI behind, and a return to the "bad old days" when manufacturers tended to go their own way and there was a lack of proper standardisation. In an attempt to make MIDI more "future proof" the original specification included a route for those manufacturers who wished to implement

special features via MIDI. This is the purpose of system exclusive messages.

In order to maintain the highest possible degree of compatibility between MIDI devices, system exclusive messages should not be used to implement anything that can be easily accommodated by other MIDI messages. Switching notes on and off, pitch bending, etc., should be accomplished using the standard MIDI messages, and are not the type of thing that can be accomplished via system exclusive messages.

There is a potential problem with system exclusive messages. There could be dire consequences if a message of this type from a unit made by one manufacturer was fed to a device made by a different manufacturer. The device receiving the message would not be able to decode it properly, and there would seem to be a real risk of such a message scrambling all the control settings of this slave instrument. In practice there is no risk of any MIDI unit trying to decode a system exclusive message unless that message is one that it can use properly. Every system exclusive message includes a manufacturer's identification number. Any device receiving one of these messages checks the manufacturer's identification number, and only responds to the message if this number is the correct one.

In practice a manufacturer's identification number might not be sufficient to prevent a device responding to a message which it can not handle properly. The fact that two instruments are made by the same manufacturer does not necessarily mean that they can usefully swap data via system exclusive messages. Data that makes perfect sense to a synthesiser will be meaningless to a sound sampler, and vice versa. In order to overcome this problem, many manufacturers use a second identification number which is unique to a particular instrument. Unless a system exclusive message contains both the correct manufacturer's and instrument identification numbers, the message is ignored. This ensures that an instrument will only respond to a message that it can fully digest.

It also means that a successful transfer of data via system exclusive messages often requires the sending device to be set up to transfer data to a specific device, and not just set up for operation with equipment from a certain manufacturer. It is

also worth noting that system exclusive messages sometimes involve two-way communications, even though, on the face of it, a straightforward dumping of data from one unit to another is all that is taking place.

Where large amounts of data are being transferred it is common practice for the data to be sent in a series of short bursts (often known as "packets"). The usual arrangement is for the first packet to be sent, after which the receiving device sends an acknowledgement message. Then the sending device transmits the next packet, another acknowledgement message is sent, and so on. Systems of this type almost invariably incorporate error checking systems, and a packet is sent again when an error is detected.

System exclusive messages can, in theory at any rate, be used to implement any desired function. Although they are in a sense "exclusive", the equipment manufacturers must publish details of any system exclusive coding that they use. Details of the coding must be made available to anyone who requires it, and it is free for anyone to use. This makes it possible for so-called third party suppliers to produce software and hardware that can exploit system exclusive messages. In practice there is not a great deal of equipment produced by one manufacturer that can communicate with equipment from a second manufacturer via system exclusive messages. It is mainly software for use with popular computers (PCs, Atari ST, Commodore Amiga, etc.) that exploits this lack of true exclusivity.

With some MIDI equipment system exclusive messages provide tremendous power. With other items of equipment few facilities (if any) are available via this route. It is necessary to carefully study the instruction manuals in order to find out what can be achieved using these messages with your particular equipment. The MIDI implementation charts will show whether or not any system exclusive messages can be transmitted or recognised by each piece of equipment. No details of the available facilities will be included in these charts though. You must study the appropriate section of each manual. If a piece of gear has a range of features that are available via the system exclusive route, there may well be a separate manual covering this aspect of things.

Although I have been talking here about system exclusive messages as a single type of MIDI message, to be strictly accurate there are two types. However, they are always used together, and must not be used singly. One is used to mark the beginning of some system exclusive data, and the other is used to indicate the end of the data. In between these two messages there can be as much (or as little) data as the software writer chooses. Despite this, where large amounts of data must be transferred, it invariably seems to be broken down into numerous packets of data. Each packet is preceded by a "start system exclusive" message, and followed by an "end system exclusive" message. In practice you will normally deal with system exclusive messages as if they were genuinely single messages, and will not need to get involved with the exact protocol used for these data transfers.

It is only fair to point out that system exclusive messages have a reputation for being difficult to use in practice. Many users seem to spend a lot of time before they get their equipment communicating properly using system exclusive messages. I have certainly spent a fair amount of time getting these messages to "deliver the goods". It is often essential to have a lot of things set up on both pieces of equipment before they will "talk" to each other via the system exclusive route.

I would certainly not wish to put anyone off using system exclusive messages. They represent a facet of MIDI that is well worth exploring once you have gained a certain amount of expertise with MIDI, and you have become familiar with the operation of your MIDI equipment. System exclusive messages are definitely not a good starting point for newcomers to MIDI.

Sample Dump

Copying large amounts of data from one instrument to another of the same (or a similar) type is a common use of system exclusive messages. Most sound samplers can swap samples via this route. In fact it is the only way of swapping sound samples via MIDI, since there are no MIDI messages specifically for this function. A sound sample is basically just a long string of numbers, and the numbers for a given sound are much the same, regardless of which manufacturer produced

the sampler. A standardised system of swapping samples via MIDI would therefore seem to be a good idea.

Such a standardised system of swapping sound samples via MIDI does exist, and it is known as the "sample dump standard". This is a form of system exclusive message, or to be more accurate it is a small group of system exclusive messages. Messages which conform to this standard do not carry a manufacturer's identification number though. Instead they carry the sample dump standard identification number. Provided you follow the instruction manuals "to the letter", swapping sound samples using this standard should be quite straightforward. However, there are a few points to bear in mind.

The most important of these is that several manufacturers have their own systems for swapping sound samples via MIDI, and do not use the sample dump standard. Such samplers can only swap samples with units from the same manufacturer, or with computers running specially written software. The instruction manuals for samplers should make it clear whether or not they support the sample dump standard. Some manufacturers use a system that is based on the sample dump standard, but which has some additions to the standard. These should work properly in conjunction with any instrument which conforms to the sample dump standard, but will give one or two additional features if used with most samplers from the same producer.

Some samplers provide higher resolution (and hence better audio quality) than others. If you produce a sample on a 12-bit sampler and then transfer it to a 16-bit type, you will only get 12-bit quality when it is played on the 16-bit machine. Similarly, a 16-bit sample which is transferred to a 12-bit type will only give 12-bit quality when played. In general, sample transfers are only possible if the resolutions of the two samplers are reasonably well matched. Once again, it is a matter of checking the "fine print" in the instruction manuals, which should detail any restrictions.

One final point is that using MIDI for large data transfers of any kind takes a long time. MIDI is very slow by normal electronic and computer standards, and sound samples can contain vast amounts of data. Transferring a large sound

sample via MIDI takes at least a few minutes, and can take up to about half an hour for a very large sample! Even small samples can take a couple of minutes per transfer. Where there is an alternative method of transferring the data, this alternative method is almost certain to be a more practical one.

There are other types of universal system exclusive message, and these are the ones concerned with MIDI time code (MTC). This is a relatively new aspect of MIDI, and it is one that is designed to make it easier to use MIDI equipment with video, cine film, etc., via a synchronisation unit. This is very much into the realm of the professional MIDI studio, and goes well beyond the scope of this publication.

Summary

System messages do not include channel numbers, and everything in the system will contemplate these messages. However, like channel messages, devices will only act on system messages that are appropriate to them.

MIDI real-time system messages are used to synchronise two MIDI sequencers. There are six real-time system messages, as follows:

Clock
Start
Continue
Stop
Active sensing
Reset

The clock messages act as regular timing signals, rather like an electronic equivalent of a metronome.

The start and continue messages are not the same. A start message starts the sequence from the beginning – a continue message results in the sequence carrying on from the current position.

Active sensing can be used to detect an error such as a broken MIDI cable, and shut down the system. It is a MIDI facility that seems to be little used in practice.

The Reset message is of little or no value in a modern MIDI context. Do not try experimenting with this one!

There are five system common messages, as follows:

Song position pointer
Tune request
Song select
Start system exclusive
End system exclusive

The song position pointer enables you to move to the desired point in a sequence. A continue message will then start the sequence from that position.

The song select message enables one of several stored sequences to be selected.

The tune request message is of no value in a modern MIDI context.

System exclusive messages can be used to implement any feature that can not be accommodated properly by the ordinary MIDI messages.

It may not be possible to make much use of system exclusive messages unless you have two or more units from the same manufacturer. However, these messages can be used by suitable units or computer software from third party producers.

Some devices can make extensive use of system exclusive messages, while others have few or no facilities implemented via this route. The manuals for your MIDI equipment should give full details of any facilities available using system exclusive messages (the MIDI implementation charts will simply show whether or not any system exclusive messages are implemented).

System exclusive messages are a very powerful MIDI feature, but they can be a bit difficult to use. It is probably best to leave these until you have gained some experience with MIDI, and you are very familiar with the operation of the units in your system.

The sample dump standard is a special (universal) form of system exclusive message. Not all samplers are designed to use this method of transferring samples though.

MIDI can be very slow when large amounts of data must be transferred from one unit to another. If another method of transfer is possible, it will almost certainly be faster and easier to use.

Chapter 5

USING MIDI

Before connecting up and trying out a new MIDI system there is the all-important task of studying the equipment manuals to determine just what the instruments can and can not achieve via MIDI. In fact the wise MIDI user checks the MIDI specification charts before buying anything. It is very important to realise that few (if any) pieces of MIDI equipment have a complete implementation, or something approximating to it.

Modern instruments tend to be much better in this respect than the early MIDI units, but most MIDI instruments have a few significant omissions in their MIDI implementations. In some cases the omissions are simply facilities that are not applicable. For instance, a unit will obviously not implement most of the system real-time messages and the song position pointer message if it does not have a built-in sequencer. In some cases the omissions are more serious. Pay particular attention to whether a type of message is both transmitted and recognised. You may find something along the lines of a keyboard instrument that implements some form of aftertouch, but that it only transmits the aftertouch messages, and does not respond to any that are received.

Implementation Charts

The "MIDI 1.0 Detailed Specification" lays down a standard arrangement for MIDI implementation charts. All MIDI devices should be supplied with a chart which conforms to the standard layout. Normally the MIDI implementation chart is at the back of the manual.

There are four columns headed "Function", "Transmitted", "Recognised", and "Remarks". The "Function" column is basically a list of MIDI messages, but there are also some additional functions here. For example, the available base channels are given here, as are the note range of the instrument. The "Transmitted" column is marked with an "O" for a feature that is supported when the device sends MIDI messages, or an "X" for features that are not supported when

messages are being sent. The "Recognised" column operates in much the same way, but indicates whether or not features are supported when the device is receiving MIDI messages.

The "Remarks" column can be used to give any helpful additional information. For instance, if the "Function" column lists MIDI control numbers that are supported, then the "Remarks" column could helpfully give the actual function of each one. A "Notes" section at the end of the implementation chart can be used to give further information, such as any peculiarities in the way MIDI messages are handled. Finally, at the end of the chart the modes supported by the instrument should be listed.

The MIDI implementation chart shown here is a dummy which shows the basic layout for these charts, and it contains some typical data. The default basic channel is the channel which the instrument uses at switch-on. If mode 4 is in use, the base channel is the lowest channel (e.g. if channels 5 to 12 are used in mode 4, then channel 5 is the base channel).

In this example chart the default channel is 1 – 16, which does not seem to make much sense. However, the "memorised" entry in the "Remarks" column indicates that the instrument has a memory circuit which remembers the base channel that was in use at the end of the previous session. The default channel is therefore whichever channel you were using when you last switched off the unit. It is increasingly common for instruments to have memory circuits which retain all the settings at switch-off, so that the whole instrument simply carries on where it left off at the end of the previous session.

The "Changed" basic channel entry indicates the channels that the instrument can use. These days virtually all new MIDI equipment can use all sixteen channels, but some older instruments are restricted to only one or two channel operation.

Modes are covered in the next section of the chart. The default mode is the one which the instrument adopts at switch-on. The "Messages" section details the channel mode messages (omni off, poly on, etc.) that the instrument can handle. Few MIDI devices seem to have the ability to send any of these messages, and perhaps rather surprisingly, some will not respond to any of them either. This is not really a

major drawback, since mode changing via MIDI does not have much practical application. The "Altered" section is usually blank these days. It simply shows what mode will be used if the selected mode is not available. Virtually all recent instruments have a full complement of MIDI modes, making this section of the chart superfluous.

"Note Numbers" is an important section, as it shows the range of notes that your instruments can accommodate. The range of notes that can be transmitted is often different to the range that can be recognised. Many instruments seem to be able to play a wider range via MIDI than they can handle via their keyboards. The recognised range of notes is often a bit misleading, as it often suggests that an instrument can play any note from 0 to 127, or something close to this. In most cases the highest notes are transposed downwards by one or two octaves, and the lowest notes are transposed upwards by one or two octaves. The "True Voice" line of the chart shows the range of notes that can be played in the correct octave. If you need to convert MIDI note values into "real" notes, remember that middle C is note number 60, and that there is a semitone increment from one note number to the next.

The next section covers velocity sensitivity. A lack of velocity sensitivity on note off messages is not really much of a drawback, and this is a feature which is relatively rare. Velocity sensitivity for note on messages is a different matter, and its absence has to be viewed as a major shortcoming. The range of velocity values used might be indicated.

Aftertouch is handled in a straightforward manner. "Key's" aftertouch is what is more normally termed polyphonic aftertouch. Similarly, "Ch's" aftertouch is what is more commonly termed overall aftertouch (although it is often referred to as channel aftertouch). The pitch bend entry is also very straightforward (these messages are recognised/transmitted, or they are not).

The control change section of the chart will show which control numbers are utilized, and the "Remarks" column will give the function of each one. Channel mode messages are covered by an earlier section of the chart, and are obviously

Example MIDI Implementation Chart

Function		*Transmitted*	*Recognised*	*Remarks*
Basic (memorised)	Default	1 – 16	1 – 16	
Channel	Changed	1 – 16	1 – 16	
	Default	X	3	
Mode	Messages	X	X	
	Altered	**********	X	
Note		30 – 90	0 – 127	
Number	True Voice	**********	12 – 108	
Velocity	Note On	O	O	
	Note Off	X	X	
Aftertouch	Key's	X	X	
	Ch's	O	X	
Pitch Bender		O	O	
Control wheel	1	X	O	mod
Change	64	X	O	sustain

Prog Change	True #	O **********	O 1 – 127
System Exclusive		O	O
System Common	Song Pos	X	X
	Song Sel	X	X
	Tune	X	X
System Real Time	Clock	X	X
	Commands	X	X
Aux Messages	Local	X	O
	All Notes	X	O
	Active Sen	X	O
	Reset	X	X
Notes		System exclusive can be disabled using front panel controls	
Mode 1:	Omni On, Poly	Mode 2:	Omni On, Poly
Mode 3:	Omni Off, Poly	Mode 4:	Omni Off, Mono

O = Yes X = No

not mentioned in this section.

The first line of the section which deals with program change messages simply shows whether or not these messages are recognised/transmitted. Where applicable, the second line indicates the range of program numbers that are supported. The number range given here is the true MIDI program change number range. In other words, it shows the range of data values in program change messages that the device can handle. As pointed out in Chapter 3, manufacturers do not always use MIDI's 0 to 127 numbering range for identifying programs. This entry is very useful, since it makes it very easy to check the degree of compatibility between different devices, with no ambiguities.

System exclusive messages are covered in the next section of the chart. It merely shows whether these messages are transmitted/recognised, and no details of any available facilities are provided here. These should be detailed in a section of the manual which deals specifically with this topic, or there may be a separate manual covering this aspect of the unit. The "Remarks" column will probably give the appropriate manufacturer's identification number, but for most users this is of only academic importance.

The next two sections cover system common and system real-time messages, and both of these sections are very straightforward. The final section covers the MIDI mode messages that are not concerned with mode changing, or the auxiliary ("Aux") messages as these are sometimes termed. Again this is very straightforward and requires no amplification.

MIDI implementation charts are printed at a (more or less) standard size. The point of this is that it makes it easy to compare two charts if they are placed side-by-side. In particular, it is easy to check the compatibility of the controller and a slave device by comparing the "Transmitted" section of the controller's chart against the "Recognised" section of the slave unit's chart.

Going Live

When a MIDI system is used for a genuine "live" performance (with no sequenced backing track or anything of this kind),

the options are relatively limited. The normal method of working is to have one or two instruments slaved to the main instrument. The latter obviously acts as the controller, and it should therefore be an instrument which has a good MIDI implementation. In particular, it should be able to transmit a wide range of different MIDI message types.

Ideally the keyboard should cover a wide range of notes, and should have both velocity sensing and some form of aftertouch. It is worth considering the use of a MIDI keyboard (i.e. a keyboard unit which does not have any built-in sound generator circuits). These are mostly high quality keyboards which have velocity sensing and aftertouch, plus a wide compass. Also, the majority of these units have facilities which make it easy to generate control change messages, etc.

The slave instruments can be keyboard types, or simply sound modules. Sound modules are the more practical choice since they are smaller and lighter than their keyboard instrument equivalents. They also cost rather less than the keyboard versions. Another point to bear in mind is that the sound module versions of instruments sometimes have better MIDI implementations than their keyboard counterparts.

One way of using a master/slave setup is to simply have the slave unit or units mimic the playing on the master keyboard. The first slave unit is set to give a different sound to the master unit, and any further slave units are also set to give different sounds. This is known as "layered" sounds, and it can give a much better effect than a single instrument playing a single type of sound. It can also sound rather banal if it is overdone, so try not to get carried away with this type of thing. Mode 1 is all that is needed for this very basic form of slaving.

Basic MIDI setups used for "live" performances are generally more effective if the keyboard is set for split operation. In other words, the keyboard is set so that one section operates on one MIDI channel, and the rest operates on another MIDI channel. Many MIDI keyboard instruments, and virtually all dedicated MIDI keyboard units, can be operated in at least one split mode.

Some MIDI keyboards are more sophisticated than others in this respect. You might be limited to something like a

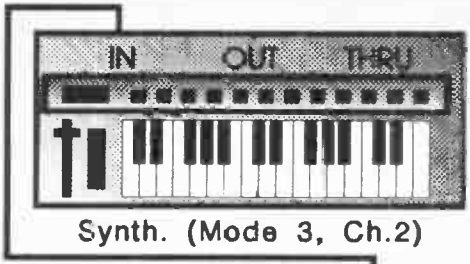
single split at a factory set point on the keyboard, with the lower section on channel 1 and the upper section on channel 2. On the other hand, it might be possible to split the keyboard at any desired point, and to have each section operating on any desired channel. It might even be possible to have a three or four way split, but initially it is probably best to keep things reasonably simple and straightforward.

When an instrument operates in a split mode it can normally have each section of the keyboard playing a different sound, if desired. It is not invariably best to do things this way, and you might find it better to have the master instrument playing just one sound, but with two slave instruments set to produce different sounds. For example, the lower section of the keyboard could be set to operate on channel 1, and it could control a slave instrument set to mode 3 operation on channel 1. The slave instrument would be set to produce a suitable bass sound. The upper section of the keyboard would be set to operate on channel 2, and would control a slave instrument set to mode 3 operation on this channel. Although multi-modes tend to be mainly associated with sequencing, they can be put to good use in a "live" performance situation. In this case a single instrument set to a multi-mode could be used instead of the two mode 3 instruments.

Of course, you might only wish to reinforce the bass or the treble part, and then a single slave instrument on channel 1 or channel 2 (as appropriate) would be used. You might wish to have a single slave instrument double the playing on both sections of the master keyboard. This would be achieved by setting the slave instrument to mode 1 so that it would play notes on any MIDI channel. Try not to use facilities simply because they are there. Only use facilities if they are genuinely useful, and will really improve the particular pieces of music that you perform. Figure 5.1 shows a typical MIDI system for "live" performances, but all four instruments would not necessarily be needed for every piece of music.

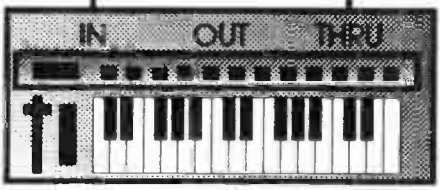
Transposing

It is well worthwhile reading through the manuals for your MIDI equipment as there may well be some useful MIDI related features which you can exploit. In the current context,

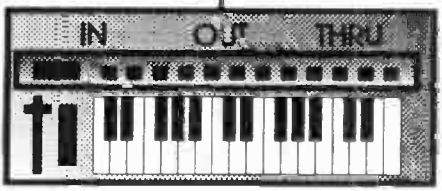


Synth. (Mode 3, Ch.2)

Synth. (Mode 3, Ch.1)



Synth. (Mode 1)



Synth. (Split, Ch.1/2)

Fig.5.1 A typical MIDI system for live performances

one of the most useful of these is the ability to transpose notes up or down by a certain amount. The controller might have the ability to send note messages that are transposed up or down by a certain number of semitones, or the slave devices may well have a similar facility for transposing received notes. For example, a slave instrument playing a bass part might be able to produce that part an octave higher or lower than it is played on the master keyboard.

When using any MIDI equipment, whether for a "live" performance or for sequencing, bear in mind that operating many of the controls on the master unit will result in corresponding MIDI messages being transmitted. For example, changing the master unit to a different program will result in a program change message being sent, and operating the pitch wheel will generate a string of pitch wheel change messages.

This is clearly very useful in a "live" performance context, as it provides an easy means of generating a range of MIDI messages that can be used to control the slave devices. However, you will probably not always require the slave instruments to follow all the adjustments that are made to the controls on the master unit. In fact the music will usually be enhanced quite noticeably if differences between the master and slave instruments can be introduced. Also bear in mind that pitch wheel modulation sounds fine with some sounds, but is not very good with others. Just because a sound is changed on the master unit it does not necessarily mean that you also require the slave instruments to switch to different sounds.

Most instruments now have a few built-in MIDI filtering facilities. This is where the unit can be set to ignore certain types of MIDI message. In the case of the controller it might be possible to switch off certain message types so that they are not transmitted when the appropriate controls are adjusted. Using one or other of these facilities you can have slave units that will not follow pitch wheel changes, program changes, or whatever. Where possible it is better to use filtering on the slave instruments rather than on the controller. This permits some slave units to follow the messages, while others can ignore them. If the messages are not transmitted in the first place, then obviously nothing at all in the system can respond

to them. Be prepared to experiment a little with different sounds and different filters to determine what best suits the music.

When using program change messages you should bear in mind that most instruments can have the same sound assigned to several program numbers. Some instruments even have the ability to copy a set of sound data from one program number to another, which makes it very easy to set up several program numbers with the same sound. The point of this is that slave units can effectively ignore program changes by simply changing to a program number which provides exactly the same sound as the original program number. Alternatively, the controller can be set up with the same sound assigned to several program numbers. The slave units can then be set to different sounds while the controller, in effect, ignores the program changes it is generating.

It is possible to produce some elaborate setups for "live" performances if you wish to do so. It is probably best to start with something more basic though, and to only use more complex systems when you outgrow your original MIDI setup. Various MIDI gadgets are available to aid "live" performances, and it is worth looking through some MIDI equipment catalogues to see what is currently on offer.

For the more advanced "live" performer a MIDI pedal unit is a very worthwhile piece of equipment. These vary considerably in their abilities, and in its most simple form a MIDI pedal will simply transmit a particular message each time the pedal is operated. The more sophisticated MIDI pedals can be programmed to send any desired string of messages each time they are operated. Often these units have several pedals, thus permitting several messages or strings of messages to be produced. Remember that there can only be one controller in a normal MIDI system, and that a merge unit might be needed in order to fully utilize a MIDI pedal unit.

Sequencing

Step-time sequencing uses a setup that is essentially the same as a master/slave system for "live" performances. However, the keyboard controller of the "live" performance system is

replaced with the step-time sequencer unit. A sequence is entered into the sequencer via its computer style keyboard (or whatever), and then played-back by the slave instruments.

A real-time sequencer requires a slightly different arrangement, with a keyboard or keyboard instrument feeding into the IN socket of the sequencer. This enables sequences to be played on the keyboard and recorded using the sequencer. Connection details for real-time sequencing systems were covered in Chapter 1 and will not be considered again here.

A real-time sequencer can be used to record a piece in one go, but they are mainly used for a form of multi-track recording. The music is recorded into the sequencer one part at a time, gradually building up the full piece. Typically a repetitive percussion track would be laid down first, followed by a melody line, and then various tracks to produce the main accompaniment. Obviously the exact scheme used has to be varied somewhat to suit the particular type of music being produced. In every case though, the music is produced track-by-track until the full piece has been built up.

If a track is not "up to scratch", it is simply erased and a replacement is then recorded. Most real-time sequencers have comprehensive editing facilities, and something like an incorrect note is easily corrected. Also, it is usually possible to overdub a sub-standard part of a track. Consequently, it is not usually necessary to completely re-record a track that has a few minor flaws. Most real-time sequencers have some step-time sequencing capability, and it is possible to use a mixture of the two techniques if desired. With all the real-time sequencers I have encountered there is no need to worry about recording each track with the keyboard set to a different MIDI channel. The channelling can be selected using the sequencer, and is easily altered at any stage of the proceedings.

With conventional multi-track recording it is possible to use one instrument to record all the tracks, provided it has a large enough repertoire of sounds. The same technique is not usually possible with MIDI multi-track recording. The classic beginners' mistake is to build up a complex sequence, and then discover that the system is incapable of playing the entire piece. In order to play back the completed piece there must be an instrument (or a voice of a multi-timbral

instrument) to produce each track. With a simple piece, one or two instruments might be sufficient to occupy all the tracks. With a complex piece several instruments will almost certainly be needed in order to reproduce all the tracks properly. Note that this does not only apply to real-time sequencing, and that it is equally germane to step-time sequencing.

This means that careful planning is needed before you start work on a new piece. Give some careful consideration to the pairing of tracks and instruments. Also give careful consideration to the best mode or modes to use. If your instruments support multi-modes, then you will almost certainly need to use these in order to fully exploit your MIDI system's potential. If a track requires an instrument with something like 8-note polyphony, then you will probably have to use an instrument set to mode 3 to provide this track. The other tracks can then be occupied by one or more instruments in mode 4 or a multi-mode. Consider a range of options, and select the one which best suits the particular piece you will be recording. The best setup for one piece of music might not give the best results with a different piece. Figure 5.2 shows a typical MIDI sequencing arrangement.

A real-time sequencer normally records all the MIDI messages that are received from the keyboard instrument, and not just note on and note off messages. It is therefore perfectly possible to record aftertouch and pitch-bend information, as well as program changes, or practically any MIDI messages that can be generated from the keyboard instrument. The only messages that are likely to cause problems are system types, which will almost certainly be filtered out by the sequencer. This is not a major drawback, since it would not normally be necessary to record these anyway.

System exclusive messages are an exception, and it can sometimes be helpful to record these as part of a sequence. Some sequencers do have facilities for recording system exclusive messages, but as pointed out in previous chapters, using this type of message can be problematic. Unless the sequencer specifically supports system exclusive exchanges with the instrument you are using, it will almost certainly not be possible to use these messages in sequences.

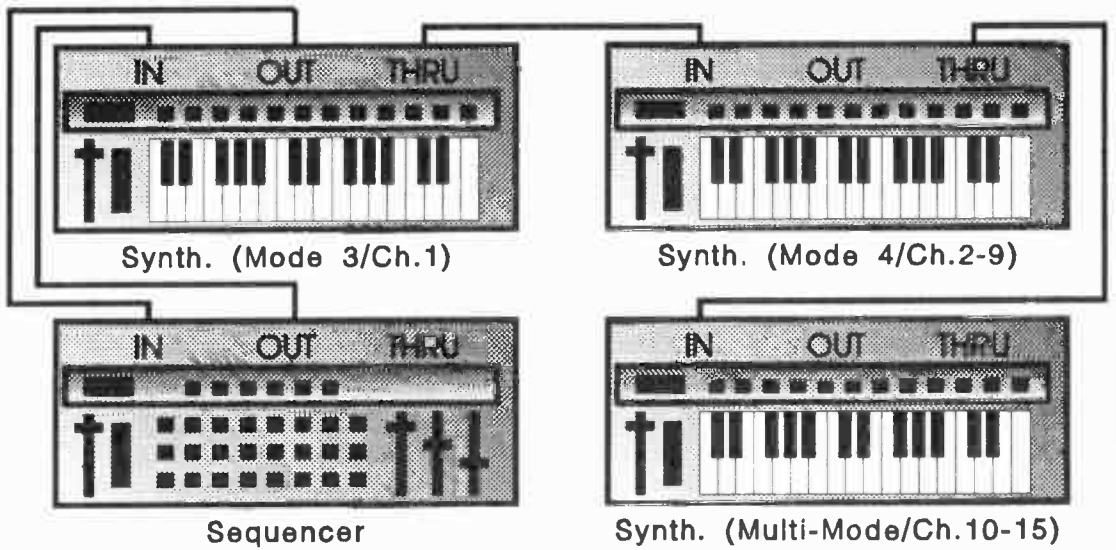


Fig.5.2 A typical MIDI sequencing arrangement

It is worth mentioning that most real-time sequencers permit virtually any type of MIDI message to be inserted into a sequence. You can, for example, record a track and then insert program change messages afterwards. This method is not a practical proposition with all types of message, and it is unlikely that good results would be obtained by adding in dozens of pitch change or aftertouch messages, even if you were willing to spend a lot of time doing this. It is a method that works well with simple one-off messages such as program change and control change messages. Of course, any good step-time sequencer should enable MIDI messages such as these to be inserted as and where necessary.

Troubleshooting

If you have difficulty in establishing contact between two units in a MIDI system, there are two likely causes. We are assuming here that the two units in question do not have any obvious faults, and that they appear to be fine apart from the lack of response via MIDI. If there is evidence of a major fault with one of the units (a display that does not operate at all or only displays random characters, sound generator circuits that are not functioning correctly, etc.), then the unit should obviously be taken to a service centre for a professional repair.

One likely cause of problems is a faulty MIDI cable. Connecting cables are quite reliable if they are well cared for, but they will be extremely unreliable if they are kicked around, tripped over, etc. (which they usually are). It is a good idea to have one or two spare MIDI cables in stock. If you have difficulties in getting a unit to respond properly, you can then replace the cable. This will either cure the problem, or indicate that the problem lies elsewhere.

The second cause is simply something on one of the devices in the system which is not set up correctly. Modern electronic musical instruments tend to have large numbers of push-buttons, and getting at some functions tends to be rather like operating a modern push-button style combination lock! If the units have MIDI on/off switching, make sure that MIDI has not become accidentally switched off. Remember that receiving devices which are set to have "omni off"

(i.e. set to mode 3, mode 4, or a multi-mode) will only respond to messages on a particular channel (or channels). Make sure that the transmitting and receiving units are set to operate on the same channel, or try setting the receiving device to mode 1 so that it will respond to messages on any channel.

When I have experienced problems with a lack of communications in a MIDI system it has almost invariably been due to something not being set up correctly. One potential cause of this problem is that many MIDI controllers will transmit mode change messages when they are switched from one mode to another. This can result in unwanted mode changes being performed in the slave units which eventually results in a confusion and an apparent malfunction of the system. With a large MIDI system it is easy to lose track of which channel (or channels) each instrument is operating on, which can again lead to confusion and apparent malfunctions. It helps if, as far as possible, you work to a system, with each instrument having standard channel assignments which you use whenever possible.

Bear in mind that although most instruments "remember" all manner of settings from one session to the next, some will revert to default channel and mode settings each time they are switched on. Such instruments will normally need to be manually set to the required MIDI mode and base channel each time they are switched on.

Sometimes the problem will not be in the form of a complete lack of response. It might manifest itself as an erratic response from the slave unit, or just one MIDI facility failing to operate properly (no pitch bending for example). A fault in one of the units in the system can not be discounted when this type of problem occurs, but in many cases it is not due to faulty hardware. One possibility is that one of the units has some MIDI filtering switched in, so that the pitch bend messages (or whatever) have no effect. The relevant settings of the MIDI controller and the "faulty" slave unit should therefore be thoroughly checked. Many instruments will "remember" MIDI filter settings from the previous session, which can lead to confusion if you forget that you were using filtering previously.

Where the system behaves erratically, one possibility is a faulty cable which is only providing an intermittent connection. When this occurs the slave device normally gives a "MIDI error" message, or something similar. It is also likely to leave some notes left switched on. If these symptoms occur, it is time to try out the spare MIDI cables!

If the erratic operation manifests itself in the form of poor timing accuracy, plus (possibly) something like pitch bending which is considerably less than smooth, the problem is probably due to MIDI choke. Remember that there is a limit to the number of MIDI messages that can be handled in a given period of time. When building up a complex multi-track recording in a real-time sequencer it is easy to get carried away. Note on and note off messages, even with something like a sixteen track recording, are unlikely to cause MIDI choke on their own. If you use a lot of pitch bending and aftertouch on several channels, MIDI choke is quite likely to occur on occasions.

Many sequencers now do their best to avoid MIDI choke. Basically, this means that the sequencer will give precedence to important messages such as note on and note off types, and if necessary it will omit less important ones such as aftertouch and pitch bend messages. This minimises the damage, but is unlikely to give entirely satisfactory results. It is unlikely that all messages of a particular type will be filtered. Instead, a certain percentage of pitch bend or aftertouch messages will be removed. This can give some rather rough sounding results, and when MIDI choke occurs, it is best to re-record one or more tracks, omitting pitch bending, aftertouch, or whatever is the primary cause of the problem.

Summary

It is essential to check the MIDI implementation charts for your MIDI equipment to determine what facilities are implemented via MIDI.

Few, if any, MIDI devices have something approximating to a full implementation. Some older instruments have only very basic MIDI implementations.

The MIDI implementation charts will only show whether or not any system exclusive messages are supported. Details of

any facilities available using system exclusive messages will be covered elsewhere in the manuals.

MIDI implementation charts are printed at a standard size so that it is easy to compare the "Transmitted" implementation of the controller with the "Recognised" implementations of the slave units.

For "live" performances split keyboard operation will usually give greatly enhanced results.

MIDI filtering and transposition can introduce differences between the master keyboard and the slave units which will often improve results.

Pitch bend, program change, and probably other types of MIDI message can be generated by the master keyboard if the appropriate controls are operated.

A MIDI pedal unit can be very useful for the more advanced "live" performer, but a merge unit might be needed in order to integrate it into the MIDI system properly.

Be careful not to get carried away with sequencer systems – a suitable instrument (or voice of an instrument) is needed to play each track.

With complex sequences some careful planning must be undertaken before you start laying down the first track.

Most types of MIDI message can be recorded using a sequencer, but there can be problems if you try to record system exclusive messages.

In most cases messages such as program change types can be added to tracks using the sequencer's editing controls. This will often be the easiest way of including them in a sequence.

If something in a MIDI system fails to respond, a faulty MIDI cable may be all that is wrong. Another possibility is that one of the units in the system is not set up correctly.

Mode 1 is useful when you are having difficulties getting two devices to communicate with one another, as it avoids problems with channel mismatches.

If a slave unit operates erratically, with notes being left switched on, an intermittent fault in a MIDI cable is the most likely cause of the problem.

If the timing becomes inaccurate at some points in the sequence, possibly with some of the less important MIDI messages being omitted, MIDI choke is almost certainly to

blame.

Some sequencers will do their best to minimise choke problems by filtering out some of the less important messages at times of high MIDI activity. This may still give something less than entirely convincing results, and it is better to avoid MIDI choke in the first place.

Please note following is a list of other titles that are available in our range of Radio, Electronics and Computer books.

These should be available from all good Booksellers, Radio Component Dealers and Mail Order Companies.

However, should you experience difficulty in obtaining any title in your area, then please write directly to the Publisher enclosing payment to cover the cost of the book plus adequate postage.

If you would like a complete catalogue of our entire range of Radio, Electronics and Computer Books then please send a Stamped Addressed Envelope to:

BERNARD BABANI (publishing) LTD
THE GRAMPIANS
SHEPHERDS BUSH ROAD
LONDON W6 7NF
ENGLAND

160	Coil Design and Construction Manual	£2.50
227	Beginners Guide to Building Electronic Projects	£1.95
BP28	Resistor Selection Handbook	£0.80
BP36	50 Circuits Using Germanium Silicon & Zener Diodes	£1.95
BP37	50 Projects Using Relays, SCRs and TRIACs	£2.95
BP39	50 (FET) Field Effect Transistor Projects	£2.95
BP42	50 Simple LED Circuits	£1.95
BP44	IC 555 Projects	£2.95
BP48	Electronic Projects for Beginners	£1.95
BP49	Popular Electronic Projects	£2.50
BP53	Practical Electronics Calculations & Formulae	£3.95
BP56	Electronic Security Devices	£2.95
BP74	Electronic Music Projects	£2.95
BP76	Power Supply Projects	£2.50
BP78	Practical Computer Experiments	£1.75
BP80	Popular Electronic Circuits – Book 1	£2.95
BP84	Digital IC Projects	£1.95
BP85	International Transistor Equivalents Guide	£3.95
BP87	50 Simple LED Circuits – Book 2	£1.95
BP88	How to Use Op-amps	£2.95
BP90	Audio Projects	£2.50
BP92	Electronics Simplified – Crystal Set Construction	£1.75
BP94	Electronic Projects for Cars and Boats	£1.95
BP95	Model Railway Projects	£2.95
BP97	IC Projects for Beginners	£1.95
BP98	Popular Electronic Circuits – Book 2	£2.95
BP99	Mini-matrix Board Projects	£2.50
BP105	Aerial Projects	£2.50
BP107	30 Solderless Breadboard Projects – Book 1	£2.95
BP110	How to Get Your Electronic Projects Working	£2.95
BP111	Audio	£3.95
BP115	The Pre-computer Book	£1.95
BP118	Practical Electronic Building Blocks – Book 2	£1.95
BP121	How to Design and Make Your Own PCB's	£2.50
BP122	Audio Amplifier Construction	£2.95
BP125	25 Simple Amateur Band Aerials	£1.95
BP126	BASIC & PASCAL in Parallel	£1.50
BP130	Micro Interfacing Circuits – Book 1	£2.75
BP131	Micro Interfacing Circuits – Book 2	£2.75
BP132	25 Simple SW Broadcast Band Aerials	£1.95
BP136	25 Simple Indoor and Window Aerials	£1.75
BP137	BASIC & FORTRAN in Parallel	£1.95
BP138	BASIC & FORTH in Parallel	£1.95
BP144	Further Practical Electronics Calculations & Formulae	£4.95
BP145	25 Simple Tropical and MW Band Aerials	£1.75
BP146	The Pre-BASIC Book	£2.95
BP147	An Introduction to 6502 Machine Code	£2.95
BP148	Computer Terminology Explained	£1.95
BP171	Easy Add-on Projects for Amstrad CPC 464, 664, 6128 & MSX Computers	£2.95
BP176	A TV-DXers Handbook (Revised Edition)	£5.95
BP177	An Introduction to Computer Communications	£2.95
BP179	Electronic Circuits for the Computer Control of Robots	£2.95
BP182	MIDI Projects	£2.95
BP184	An Introduction to 68000 Assembly Language	£2.95
BP187	A Practical Reference Guide to Word Processing on the Amstrad PCW8256 & PCW8512	£5.95
BP190	More Advanced Electronic Security Projects	£2.95
BP192	More Advanced Power Supply Projects	£2.95
BP193	LOGO for Beginners	£2.95
BP196	BASIC & LOGO in Parallel	£2.95
BP197	An Introduction to the Amstrad PC's	£5.95
BP198	An Introduction to Antenna Theory	£2.95
BP230	A Concise Introduction to GEM	£2.95
BP232	A Concise Introduction to MS-DOS	£2.95
BP233	Electronic Hobbyists Handbook	£4.95
BP239	Getting the Most From Your Multimeter	£2.95
BP240	Remote Control Handbook	£3.95
BP243	BBC BASIC86 on the Amstrad PC's & IBM Compatibles – Book 1: Language	£3.95
BP244	BBC BASIC86 on the Amstrad PC's & IBM Compatibles – Book 2: Graphics and Disk Files	£3.95
BP245	Digital Audio Projects	£2.95
BP246	Musical Applications of the Atari ST's	£5.95
BP247	More Advanced MIDI Projects	£2.95
BP248	Test Equipment Construction	£2.95
BP249	More Advanced Test Equipment Construction	£3.50
BP250	Programming in FORTRAN 77	£4.95
BP251	Computer Hobbyists Handbook	£5.95
BP254	From Atoms to Amperes	£3.50
BP255	International Radio Stations Guide (Revised 1991/92 Edition)	£5.95
BP256	An Introduction to Loudspeakers & Enclosure Design	£2.95
BP257	An Introduction to Amateur Radio	£3.50
BP258	Learning to Program in C (Revised Edition)	£4.95
BP259	A Concise Introduction to UNIX	£2.95
BP260	A Concise Introduction to OS/2	£2.95
BP261	A Concise Introduction to Lotus 1-2-3 (Revised Edition)	£3.95

BP262	A Concise Introduction to Wordperfect (Revised Edition)	£3.95
BP264	A Concise Advanced User's Guide to MS-DOS (Revised Edition)	£3.95
BP265	More Advanced Uses of the Multimeter	£2.95
BP266	Electronic Modules and Systems for Beginners	£3.95
BP267	How to Use Oscilloscopes & Other Test Equipment	£3.50
BP269	An Introduction to Desktop Publishing	£5.95
BP270	A Concise Introduction to Symphony	£3.95
BP271	How to Expand, Modernise & Repair PC's & Compatibles	£4.95
BP272	Interfacing PC's and Compatibles	£3.95
BP273	Practical Electronic Sensors	£4.95
BP274	A Concise Introduction to SuperCalc5	£3.95
BP275	Simple Short Wave Receiver Construction	£3.95
BP276	Short Wave Superhet Receiver Construction	£2.95
BP277	High Power Audio Amplifier Construction	£3.95
BP278	Experimental Antenna Topics	£3.50
BP279	A Concise Introduction to Excel	£3.95
BP280	Getting the Most From Your PC's Hard Disk	£3.95
BP281	An Introduction to VHF/UHF for Radio Amateurs	£3.50
BP282	Understanding PC Specifications	£3.95
BP283	A Concise Introduction to SmartWare II	£4.95
BP284	Programming in QuickBASIC	£4.95
BP285	A Beginners Guide to Modern Electronic Components	£3.95
BP286	A Reference Guide to Basic Electronics Terms	£5.95
BP287	A Reference Guide to Practical Electronics Terms	£5.95
BP288	A Concise Introduction to Windows3.0	£3.95
BP290	An Introduction to Amateur Communications Satellite	£3.95
BP291	A Concise Introduction to Ventura	£3.95
BP292	Public Address Loudspeaker Systems	£3.95
BP293	An Introduction to Radio Wave Propagation	£3.95
BP294	A Concise Introduction to Microsoft Works	£4.95
BP295	A Concise Introduction to Word for Windows	£4.95
BP297	Loudspeakers for Musicians	£3.95
BP298	A Concise Introduction to the Mac System & Finder	£3.95
BP299	Practical Electronic Filters	£4.95
BP300	Setting Up An Amateur Radio Station	£3.95
BP301	Antennas for VHF and UHF	£3.95
BP302	A Concise Users Guide to Lotus 1-2-3 Release 3.1	£3.95
BP303	Understanding PC Software	£4.95
BP304	Projects for Radio Amateurs and SWLs	£3.95
BP305	Learning CAD with AutoSketch for Windows	£5.95
BP306	A Concise Introduction to Ami Pro 3	£4.95
BP307	A Concise Introduction to QuarkXPress	£4.95
BP308	A Concise Introduction to Word 5.1 on the Macintosh	£5.95
BP309	Preamplifier and Filter Circuits	£3.95
BP310	Acoustic Feedback – How to Avoid It	£3.95
BP311	An Introduction to Scanners and Scanning	£4.95
BP312	An Introduction to Microwaves	£3.95
BP313	A Concise Introduction to Sage	£3.95
BP314	A Concise Introduction to Quattro Pro	£4.95
BP315	An Introduction to the Electromagnetic Wave	£4.95
BP316	Practical Electronic Design Data	£4.95
BP317	Practical Electronic Timing	£4.95
BP318	A Concise User's Guide to MS-DOS 5	£4.95
BP319	Making MS-DOS Work for You	£4.95
BP320	Electronic Projects for Your PC	£3.95
BP321	Circuit Source – Book 1	£4.95
BP322	Circuit Source – Book 2	£4.95
BP323	How to Choose a Small Business Computer System	£4.95
BP324	The Art of Soldering	£3.95
BP325	A Concise Users Guide to Windows3.1	£4.95
BP326	The Electronics of Satellite Communications	£4.95
BP327	MS-DOS One Step at a Time	£4.95
BP328	Sage Explained	£5.95
BP329	Electronic Music Learning Projects	£4.95
BP330	A Concise User's Guide to Lotus 1-2-3 Release 2.4	£4.95
BP331	A Beginners Guide to MIDI	£4.95
BP332	A Beginners Guide to TTL Digital ICs	£4.95
BP333	A Beginners Guide to CMOS Digital ICs	£4.95
BP334	Magic Electronic Projects	£4.95
BP335	Operational Amplifier User's Handbook	£5.95
BP336	A Concise User's Guide to Lotus 1-2-3 Release 3.4	£5.95
BP337	A Concise Users Guide to Lotus 1-2-3 for Windows	£5.95
BP338	A Concise Introduction to Word for Windows	£5.95
BP339	A Concise Introduction to Wordperfect 5.2 for Windows	£5.95
BP340	A Concise Introduction to dBase V	£4.95
BP341	A Concise Users Guide to MS-DOS 6	£5.95
BP342	A Conciser Users Guide to Lotus Improv	£5.95

BERNARD BABANI BP331

A Beginners Guide to MIDI

☐ After a slow start in the early 1980s, MIDI (Musical Instrument Digital Interface) has progressed to the point where it dominates some aspects of electronic music making. It is now an area of electronic music that few can afford to ignore. Many potential users are put off by the technicalities associated with MIDI, but it is not essential to understand the precise way in which MIDI functions in order to use it effectively. In fact many everyday users of MIDI have little idea of how it does what it does.

☐ With the aid of this book, those with no previous knowledge of MIDI will learn how to:-

Connect up a MIDI system that will suit their exact requirements.

Exploit MIDI modes and channels, for both "live" performances and sequencing.

Use the various message types that provide communications between the units in their MIDI systems.

Distinguish between the MIDI messages that are worth exploiting and those which are of no real value to them.

Use the basic MIDI timing and synchronisation facilities.

Exploit MIDI filtering, transposition, and other "extras" that are built into many instruments.

Interpret a MIDI implementation chart.

Avoid problems with MIDI "choke", and troubleshoot when minor problems occur.

☐ In fact this book covers everything you need to know in order to put together a MIDI system and use it effectively.

ISBN 0-85934-331-6

00495

£4.95

9 780859 343312