M&06/5/COMSC/BP2/ENG/TZ0/XX/CS

COMPUTER SCIENCE
CASE STUDY

For use in May 2006 and November 2006

INSTRUCTIONS TO CANDIDATES

- Case study booklet required for higher level paper 2 and standard level paper 2 computer science examinations.

2206-7001 ; 8806-7001

MIDI – Musical Instrument Digital Interface.

MIDI is the use of computer programs and sound cards to play numerous musical parts with realistic sounds.

History

Before MIDI was invented a composer had to get a whole band together in order to hear what a composition sounded like. This was time-consuming, expensive and difficult to arrange. But it had to be done after every re-write in order to hear the latest version. MIDI makes it possible to hear a piece of music without the need for musicians who are capable of learning to play the music fast. MIDI makes it possible for one person to:

- Create music that would require many musicians to play.
- Engineer difficult passages.
- Play the music.
- Edit the music.
- Post the music.
- Create music even if one can't play an instrument.

MIDI requires fewer people, less equipment and no studio facilities. Instead of hiring musicians, equipment and studios to record a composition after every change, the composer can record each track into the computer whenever it is convenient. Once in the computer the music can be edited often until it is perfect. Corrections can be made to individual notes so that new versions of the music do not require the musicians to replay all the 'good' sections again. Difficult sections can be played in at a slow speed, or even engineered, and replayed at full speed without affecting the sound quality. Additionally, the whole piece can be edited if needed. For example, every note could be raised by a semitone at the press of a button. This would normally require the whole piece to be re-recorded. Replays can be heard immediately after each edit and multiple versions can be saved for comparison. This leaves the composer free to compose music instead of spending large amounts of time organizing recording sessions.

The cost of MIDI technology is falling and performance is rising. So much so that very high quality equipment is now available to the general public. By comparison, the cost of using a studio will keep pace with wage, rent and equipment costs. MIDI technology is, in effect, a "Band in a Box" that is available 24 hours a day, 7 days a week. When the composer has an inspirational idea, the band is there, ready and waiting to show the composer what the idea will actually sound like. The idea is also saved for future reference.

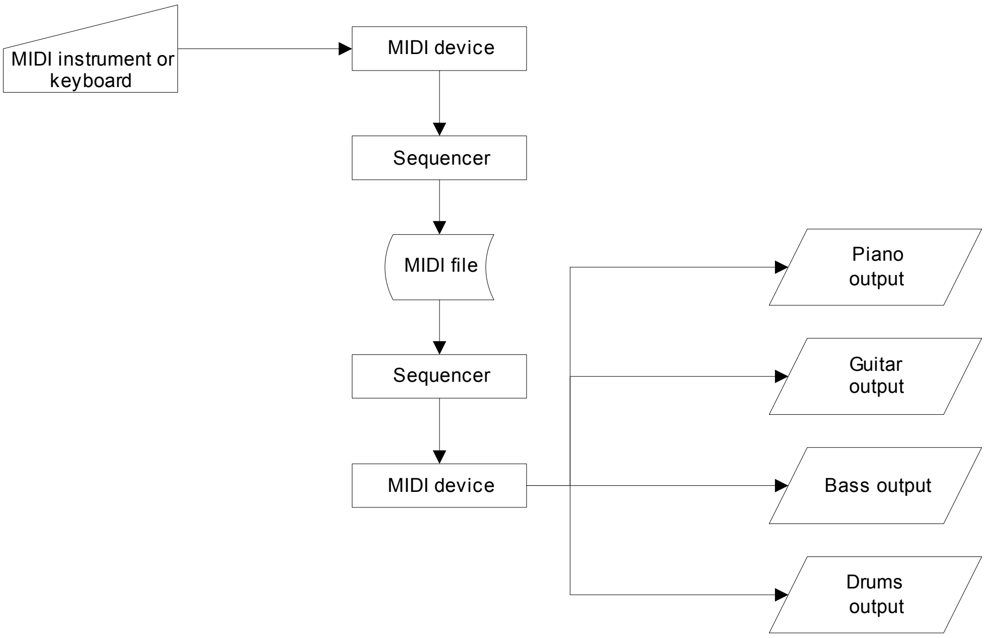
Input to a MIDI file is usually via a MIDI keyboard but other instruments can be modified to generate MIDI data. A guitar, for example, usually generates an analog signal but, fitted with the right sensors, it can generate digital MIDI data. This same data can be used to drive a sound module that is set up to produce the sound of a harp, or a flute, or a piano. In fact anything, the sound of a frog croaking can be used as the sound source and suddenly the frog is singing the composer's tune. If you can play one MIDI instrument you can produce the sound of any real instrument and more - in effect creating music that is impossible using conventional instruments.

MIDI emerged alongside synthesisers. Musicians wanted to control them with computers and at the same time one of the first home computers, the Commodore 64, became available to the public. The Commodore had a built-in analog synthesiser chip and musicians began to use it. The PC had no built in chip but it wasn't long before a slot-in card with a built in chip was available. It came with a language suitable for transferring data, MIDI, which was adopted by many manufacturers and the musical revolution had begun. Software, called a sequencer, was developed to make controlling the chip easy, MIDI interfaces were added to many devices and the MIDI Manufacturers Association was created to establish and control industry wide protocols.

Computerised control allows the musician to:

- Control electronic instruments remotely.
- Control electronic instruments automatically.
- Control more than one instrument.
- Combine sounds.
- Use one sound module to create several instruments.
- Create music that is physically impossible to play.

The instructions for controlling MIDI devices can be batch processed (a pre-recorded list of instructions) or processed in real time (instructions generated during performance).



How it works

When a musician uses a conventional instrument to create a note, it always starts with an action such as blowing, plucking, pressing, bowing or hitting. This is called a ‘NOTE-ON’ event in MIDI. The sound can be stopped sometime later when the musician ceases to execute the ‘NOTE-ON’ event. Stopping the sound in this way is a ‘NOTE-OFF’ event. The sound will also have a ‘PITCH’ corresponding to the underlying frequency of the sound wave. A high ‘PITCH’ note such as that produced by a whistle will have a high frequency.

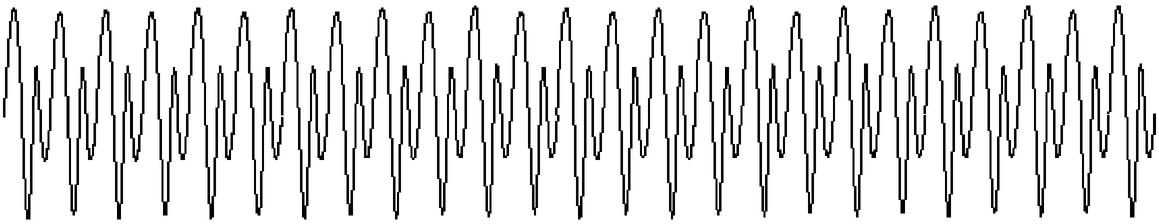


Figure 1: High Frequency

A low ‘PITCH’ note such as that produced by a rumble will have a low frequency.

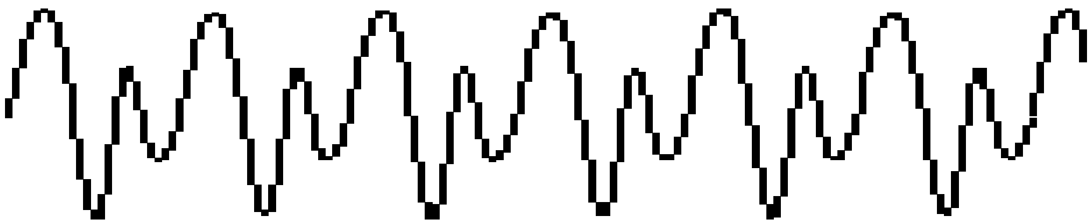


Figure 2: Low Frequency

The sound will also have a ‘VOLUME’ or amplitude depending on how loud it is. A quiet sound will have a low ‘VOLUME’ and a loud sound will have a high ‘VOLUME’. The speed at which the note builds up from zero volume to full volume is called the ‘ATTACK’. Something that is hit, such as a drum, reaches full volume almost straight away and is said to have a fast ‘ATTACK’. The gentle build up volume that can be produced by bowing a cello has a slow ‘ATTACK’. The speed at which a note fades away is called the ‘DECAY’. A drum beat fades away quickly and so has a fast ‘DECAY’, whereas a piano note, so long as the key remains pressed, loses volume slowly and so has a slow ‘DECAY’. There are other PARAMETERS that can be applied to a note but these are the main ones and collectively they define the ENVELOPE of the note.

Figure 3: Parameters of a note. Courtesy of MIDI Manufactures Association – Used with permission.

When the envelope parameters are combined with a soundwave the result is a note.

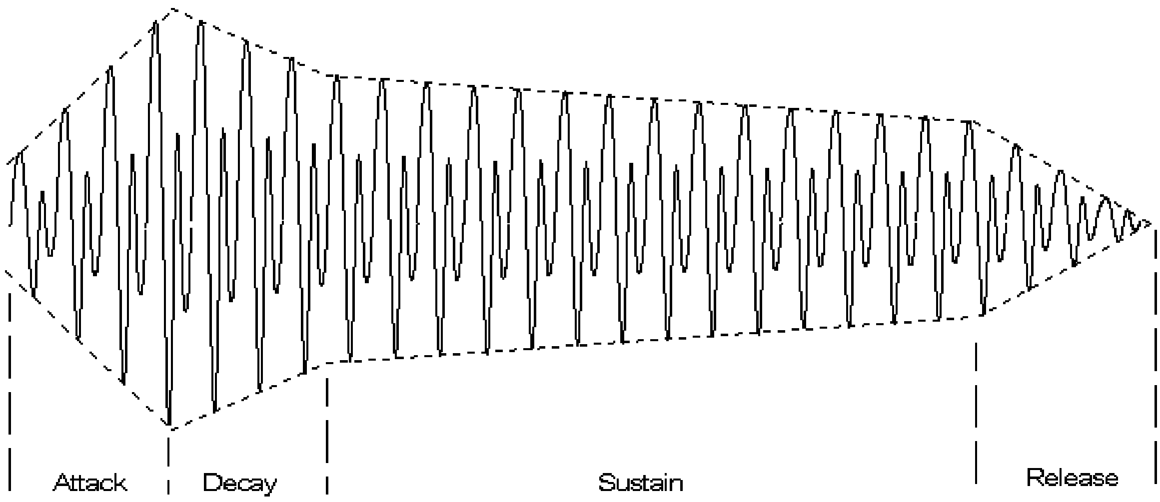


Figure 4: A sound wave modified by an envelope. Courtesy of MIDI Manufactures Association – Used with permission.

A musical note can therefore be stored as a list of integers representing all of its parameters. Storing the data this way makes it easy to manipulate. For example, to increase the volume of a note just increase the value of the Volume parameter. Or to change the note itself just change the Pitch value. It also produces very small files compared with other methods of recording. This makes them very suitable for use in small devices and for transmission. For example a tune can easily be stored on a small chip such as those found in greeting cards or delivered as part of an on-line computer game. And because it's all just numbers it is easy to handle digitally without the attenuation affects suffered by analog signals.

The sound wave inside the envelope can be created in two ways. Frequency Modulation (FM) synthesis and Wavetable synthesis.

With FM synthesis, a minimum of two periodic signals are mixed together and their parameters adjusted to create a sound. These sounds are not the sounds of real instruments but can be made to sound similar. FM synthesis is better for 'electronically' created sounds.

If real instrument sounds are needed then digital sampling systems are used to store high quality sound samples digitally, and then replay them as needed. Because samples require large amounts of memory, looping, pitch shifting, mathematical interpolation, and digital filtering can be used to reduce the amount of sampling required. This type of synthesis is called "wavetable" synthesis and can be thought of as a "table" of sounds which may be looked up and used. One-Shot Sounds such as short drum sounds are stored as a single sample which is played once through with no processing to save memory.

Data Compression can be used to improve the signal-to-noise ratio for some samples. The dynamic range of the sound samples is reduced and then decompressed during playback to restore the dynamic range. There are several techniques which may be used to compress the dynamic range of a signal.

Because the data for each instrument is delivered on a different track it can be transmitted between devices on different channels and every aspect of the sound can be easily edited and other parameters easily added. Not only can performance parameters be added (‘slides’, ‘slurs’, ‘pull offs’ etc) but colouration (‘echo’, ‘phase’, ‘sustain’ etc) can be applied.

In comparison, a digitised recording of an analog instrument has to be sampled about 44,000 times per second. This means the amplitude of the sound wave is measured 44,000 times each second and the value stored as a number. That’s 44,000 numbers being saved for each second of performance, and that has to be done for each instrument. It is also more difficult to edit afterwards. If instruments have been mixed to produce a final performance it is very difficult to pull them apart again and the results may be unpredictable. This makes it more demanding on resources and takes control away from the composer.

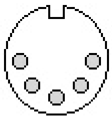
For example, if you play a note on a keyboard, and hold it for 1 minute, the NOTE-ON and NOTE-OFF MIDI messages will be stored in 6 bytes. To do the same thing with a digital recording will take 44,000 * 60 samples because all of the sound between the start of the note and the end of the note has to be recorded. MIDI records the musician’s actions, not the sound so it just has to save the events. The spaces between events are not recorded. MIDI takes about 10 Kbytes per minute of performance to store and digital audio takes about 10 Mbytes per minute of performance to store.

The major disadvantage of MIDI is that it cannot handle the singing of words. When the voice is used as an instrument (‘oohs’ and ‘aahs’) a sample of the sound can be made and played back as a parameter in a MIDI message. But because the words of a song are changing continuously they cannot be stored as a MIDI message parameter.

Communication

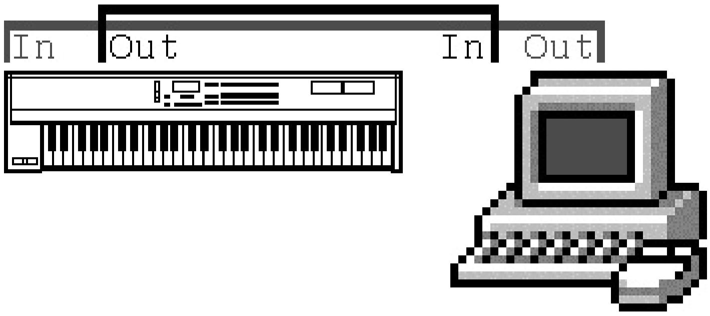
MIDI data flows in one direction (unidirectional) and does not handshake the target device before transmission (asynchronous). This means that the target device must be dedicated to the source and be available at all times to receive data. Data flows at a rate of 31.25 Kbits/s and is transmitted as 10-bit bytes: a start bit, 8 data bits and a stop bit. Signals are generated by MIDI controllers or sequencers.

MIDI controllers are devices that are attached to musical instruments that generate MIDI data as the musician is playing the instrument (real time processing). They are electronic circuits connected to keys, pick ups or microphones. A MIDI sequencer is a device which allows MIDI data to be captured, stored, edited and replayed (batch processing). A sequencer can be a computer application or a dedicated hardware device that can do everything a software version can do but it does it with electronic circuits and not computer program instructions. The hardware version is therefore faster and of a higher quality but the software version is easier to update or upgrade. The messages are transmitted via a MIDI OUT connection and received by a MIDI IN connection. There will often be a MIDI THRU connection available for messages that need to be passed from machine to machine without being altered in any way.



The connectors themselves are called 5 pin DIN jacks and separate connectors are used for IN, OUT and THRU signals. The internal wiring between pins means that general purpose DIN-to-DIN leads cannot be used and special MIDI leads have to be used. They look the same but the internal pin-to-pin connections are different. When a computer is connected to a keyboard MIDI connection, the keyboard is used to create performance data, the computer is used to store, edit and send it back to the keyboard. When the keyboard re-receives the MIDI data, it uses its onboard sound chips to reproduce the sounds.

Figure 5:



A single message sent between machines will represent a single instruction such as a NOTE-ON command. A typical message will often be 3 bytes in length: a status byte and 2 data bytes, For example, if a musician presses middle c on a keyboard the midi circuits inside the keyboard will send the numbers 144 60 64 to the MIDI OUT connectors. 144 means NOTE-ON. 60 means middle c. 64 is the velocity of the key press. When the key is released the numbers 128 60 64 will be sent. If the MIDI IN connector of a second device were connected to the same MIDI OUT connector, the same note would be played. If the first device were set to a piano sound, and the second device set to an organ sound, the result would be a piano and an organ playing the same note. Linking devices together like this is called 'Daisy Chaining'. Communications are serial and, when many machines are linked together, many events may need to be triggered at the same time. On the first note of a bar, for example, 10 instruments all playing a 5 note chord will generate 50 NOTE-ON messages needing to be delivered at the same time. Because the system is serial the 50 events will have to be triggered one after another. However, as it takes about 1ms to deliver a MIDI message, all 50 events can be triggered in 50 ms. To the human ear they sound simultaneously. Where delays may be audible a feature called 'running status' can be used to combine similar events in a single message. Messages will be either system or channel (voice and mode) messages.

Channel messages can be of type:

- Note On,
- Note Off,
- Velocity,
- Aftertouch
- Sustain,
- Key Pressure,
- Channel Pressure,
- Pitch Bend,
- Program Change.

System messages can be of type:

- Bank Select,
- RPN / NRPN,
- Channel Mode Messages,
- System Messages,
- System Common Messages,
- System Real Time Messages,
- System Exclusive Messages.

MIDI data can be stored in three formats. Format 0 stores all of the MIDI sequence data in a single track. This makes the data file small and uses less bandwidth for transmission but it is difficult to edit because all the parts are mixed. Format 1 files store MIDI data as a collection of tracks. The files are bigger than format 0 but they are easy to edit as all the different channels have been preserved. Format 2 files can store several channels using independent patterns. Format 1 is the most often used.

Messages can be transmitted along 16 channels and devices can be set to respond to a pre-selected channel. Middle c, NOTE-ON transmitted on channel 1 will be 144 60 64. The same note on channel 2 will be 145 60 64. Four bits within the message bit stream select the channel. This means that data sent around a daisy chain can be accepted or ignored by devices on the transmission path depending on which channels they are set to receive data. In effect, the devices can all be playing different notes using the data being transmitted around the system.

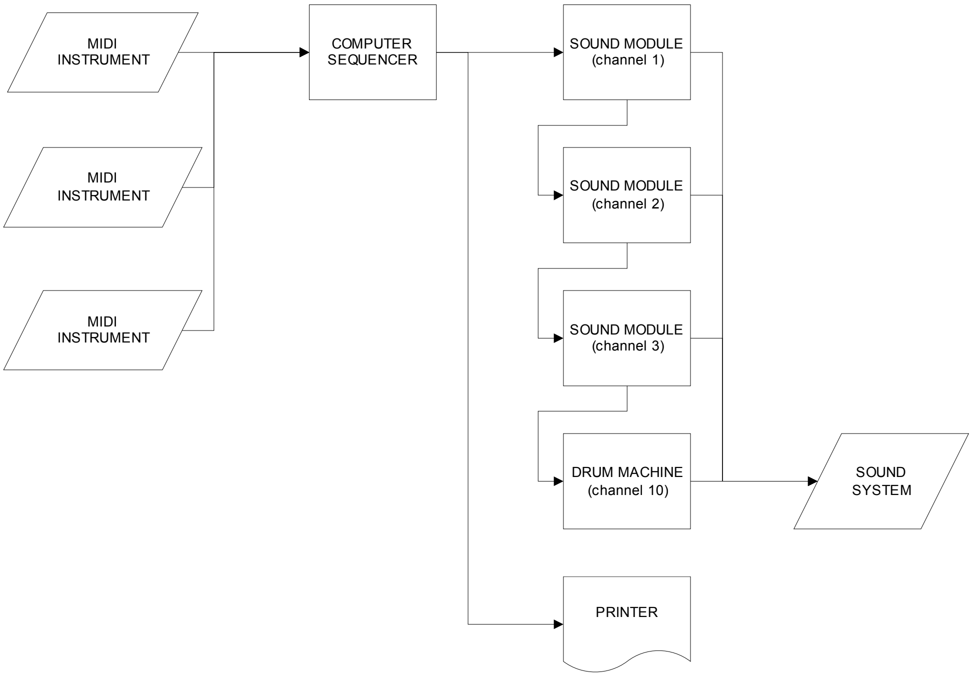


Figure 6:
Courtesy of MIDI Manufactures Association – Used with permission

The sounds produced by each sound module will depend on what sound patches have been assigned to each channel. The General MIDI system utilises channel 10 for drums and all the other channels for chromatic instruments. Instrument sounds are grouped into sets. For example, program numbers 1-8 are piano sounds, 9-16 are tuned percussion sounds, 17-24 are organ sounds, 25-32 are guitar sounds, etc. There are 128 sounds available. Channel 10 is therefore different from the rest as each note will produce the sound of a different instrument (bass drum, cymbal, clap) rather than the same sound produced at a different pitch.

Although three different input devices are shown above, often only one is used: a keyboard. The computer could be a small dedicated sequencer. And if the sound module were a type that could produce different sounds (polyphonic multi-timbral) on different channels at the same time then the system would look like:

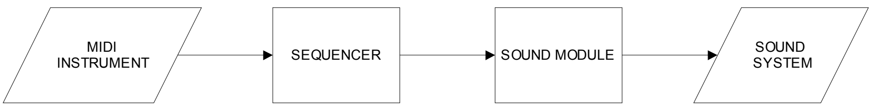


Figure 7:
Courtesy of MIDI Manufactures Association – Used with permission

Even a rudimentary system like this can reproduce the sound of an entire band. The sound card in a PC has been of limited quality in the past but better quality cards can be added, making a PC-based home recording studio look the following:

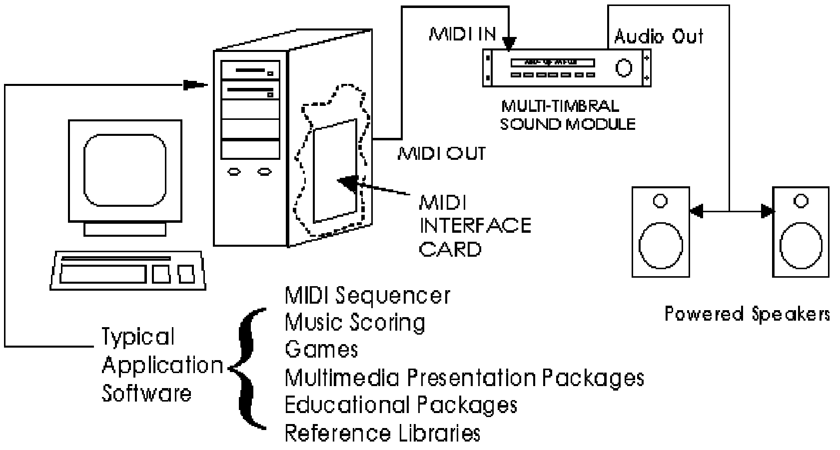


Figure 8: **A PC-Based MIDI System**. Courtesy of MIDI Manufactures Association – Used with permission.

Configuration
Computer applications address hardware devices through the use of drivers. These provide applications software with a common interface to access hardware simply. When a MIDI interface or synthesiser is installed in a computer a suitable device driver has to be loaded. In Windows a MIDI Mapper applet is used to receive messages from an application, which then sends the messages to the device driver.

Conclusion

The storage efficiency and ease of editing makes MIDI an attractive tool for generation of sounds in multimedia applications, computer games, portable devices or high-end karaoke equipment. It also opens up its own avenues for musical expression to the musically literate and illiterate alike. One such application uses the electronic systems found in burglar alarms to generate the MIDI messages. Proximity sensors are used to detect movement within a performance space. The speed of movement and positioning is used to generate pitch and volume data. Computers are then used to normalise this data to a set of rules and to generate other signals that are used to create backing. Rather than the dancer having to follow the music, the dancer can thus create to music and never be out of step.