CHAPTER 12

Synchronization

Of course, it's a safe bet to say that music and audio itself, in all its various forms, is an indispensable part of almost all types of media production. In video postproduction, digital video editors, digital audio workstations (DAWs), audio and video transports, automated console systems and electronic musical instruments routinely work together to help create a finished soundtrack (Figure 12.1). The underlying technology that allows multiple audio and visual media to operate in tandem (so as to maintain a direct time relationship) is known as *synchronization* or *sync*.

Strictly speaking, synchronization occurs when two or more related events happen at precisely the same relative time. With respect to analog audio and video systems, sync is achieved by interlocking the transport speeds of two or more machines. For computer-related systems (such as digital video, digital audio and MIDI), synchronization between devices is often achieved through the use of a timing clock that can be fed through a separate line or is directly embedded within the digital data line itself. Within such an environment, it's often necessary for analog and digital devices to be synchronized together; resulting in a number of ingenious forms of communication and data translation systems. In this chapter, we'll explore the various forms of synchronization used for both digital and analog devices, as well as current methods for maintaining sync between media types.

[caption]**FIGURE 12.1** Example of an integrated audio production system.

TIMECODE

Maintaining relative sync between media devices doesn't require that all transport speeds involved in the process be constant; however, it's critical that they maintain the same relative speed and position over the course of a program. Physical analog devices, for example, have a particularly difficult time achieving this. Due to differences in mechanical design, voltage fluctuations and tape slippage, it's a simple fact of life that analog tape devices aren't able to maintain a constant playback speed, even over relatively short durations. For this reason, accurate sync between analog and digital machines would be nearly impossible to achieve over any reasonable program length without some form of timing lock. It therefore quickly becomes clear that if production is to utilize, multiple forms of media and record/playback … sync is essential.

The standard method of interlocking audio, video and film transports makes use of a code that was developed by the Society of Motion Picture and Television Engineers (SMPTE, www.smpte.org). This timecode (or SMPTE timecode) identifies an exact position within recorded media or onto tape by assigning a digital address that increments over the course of a program's duration. This address code can't slip in time and always retains its original location, allowing for the continuous monitoring of tape position to an accuracy of between 1/24th and 1/30th of a second (depending on the media type and frame rates being used). These divisional segments are called *frames*, a term taken from film production. Each audio or video frame is tagged with a unique identifying number, known as a "timecode address". This eight-digit address is displayed in the form 00:00:00:00, whereby the successive pairs of digits represent hours:minutes:seconds:frames or HH:MM:SS:FF (Figure 12.2).

[caption]**FIGURE 12.2** Readout of a SMPTE timecode address in HH:MM:SS:FF.

The recorded timecode address is then used to locate a position on hard disk, magnetic tape or any other recorded media, in much the same way that a letter carrier uses a written address to match up, locate and deliver a letter to a specific, physical residence (i.e., by matching up the address, you can then find the desired physical location point, as shown in Figure 12.3a). For example, let's suppose that a time-encoded analog multitrack tape begins at time 00:01:00:00, ends at 00:28:19:00 and contains a specific cue point (such as a glass shattering) that begins at 00:12:53:19 (Figure 12.3b). By monitoring the timecode readout, it's a simple matter to locate the precise position that corresponds to the cue point on the tape and then perform whatever function is necessary, such as inserting an effect into the sound track at that specific point ... CRAAAASH!

[caption]**FIGURE 12.3** Location of relative addresses: (a) postal address analogy; (b) timecode addresses and a cue point on longitudinal tape.

TIMECODE WORD

The total of all time-encoded information that's encoded within each audio or video sync frame is known as a *timecode word*. Each word is divided into 80 equal segments, which are numbered consecutively from 0 to 79. One word covers an entire audio or video frame, such that for every frame there is a unique and corresponding timecode address. Address information is contained in the digital word as a series of bits that are made up of binary 1's and 0's.

In the case of an analog, a SMPTE signal is electronically encoded in the form of a modulated square wave. This method of encoding information is known as biphase modulation. Using this code type, a voltage or bit transition in the middle of a half-cycle of a square wave represents a bit value of 1, while no transition within this same period signifies a bit value of 0 (Figure 12.4). The most important feature about this system is that detection relies on shifts within the pulse and not on the pulse's polarity or direction. Consequently, timecode can be read in either the forward or reverse play mode, as well as at fast or slow shuttle speeds.

[caption]**FIGURE 12.4** Biphase modulation encoding.

<Insert DIY logo here>
DIY box format

Try this: SMPTE Timecode

1. Go to the Tutorial section of www.modrec.com, click on SMPTE Audio Example and play the timecode soundfile. Not my favorite tune, but it's a useful one!
2. The 80-bit timecode word is subdivided into groups of 4 bits (Figure 12.5), whereby each grouping represents a specific coded piece of information. Each 4-bit segment represents a binary-coded decimal (BCD) number that ranges from 0 to 9. When the full frame is scanned, all eight of these 4-bit groupings are read out as a single SMPTE frame number (in hours, minutes, seconds and frames).

Sync Information Data

An additional form of information that's encoded into the timecode word is sync data. This information exists as 16 bits at the end of each timecode address word. These bits are used to define the end of each frame. Because timecode can be read in either direction, sync data is also used to tell the device which direction the tape or digital device is moving.

[caption]**FIGURE 12.5** Biphase representation of the SMPTE timecode word.

Timecode Frame Standards

In productions using timecode, it's important that the readout display be directly related to the actual elapsed time of a program, particularly when dealing with the exacting time requirements of broadcasting. Due to historical and technical differences between countries, timecode frame rates may vary from one medium, production house or region of origin to another. The following frame-rates are available:

- *30 fr/sec (monochrome U.S. video):* In the case of a black-and-white (monochrome) video signal, a rate of exactly 30 frames per second (fr/sec) is used. If this rate (often referred to as non-drop code) is used on a black and-white program, the timecode display, program length and actual clock-on-the-wall time would all be in agreement.
- *29.97 fr/sec (drop-frame timecode for color NTSC video):* The simplicity of 30 fr/sec was eliminated, however, when the National Television Standards Committee (NTSC) set the frame rate for the color video signal in the United States and Japan at 29.97 fr/sec. Thus, if a timecode reader that's set up to read the monochrome rate of 30 fr/sec were used to read a color program, the timecode readout would pick up an extra 0.03 frame for every second that passes. Over the duration of an hour, the timecode readout would differ from the actual elapsed time by a total of 108 frames (or 3.6 seconds). To correct for this difference and bring the timecode readout and the actual elapsed time back into agreement, a series of frame adjustments was introduced into the code. Because the goal is to drop 108 frames over the course of an hour, the code used for color has come to be known as drop-frame code. In this system, two frame counts for every minute of operation are omitted from the code, with the exception of minutes 00, 10, 20, 30, 40 and 50. This has the effect of adjusting the frame count, so that it agrees with the actual elapsed duration of a program.
- *29.97 fr/sec (non-drop-frame code):* In addition to the color 29.97 drop-frame code, a 29.97 non-drop-frame color standard can also be found in video production. When using non-drop timecode, the frame count will always advance one count per frame, without any drops. As you might expect, this mode will result in a disagreement between the frame count and the actual clock-on-the-wall time over the course of the program. Non-drop, however, has the distinct advantage of easing the time calculations that are often required in the video editing process (because no frame compensations need to be taken into account).
- *25 fr/sec EBU (standard rate for PAL video):* Another frame rate format that's used throughout Europe is the European Broadcast Union (EBU) time-code. EBU utilizes SMPTE's 80-bit code word but differs in that it uses a 25 fr/sec frame rate. Because both monochrome and color video EBU signals run at exactly 25 fr/sec, an EBU drop-frame code isn't necessary.
- *24 fr/sec (standard rate for film work):* The medium of film differs from all of these in that it makes use of an SMPTE timecode format that runs at 24 fr/sec.

From the above, it's easy to understand why confusion often exists as to which frame rate

should be used on a project. Basically, if you are working on an in-house project that doesn't incorporate time-encoded material that comes from the outside world, you should choose a rate that both makes sense for you and is likely to be compatible with an outside facility (should the need arise).

For example, electronic musicians who are working in-house in the US will often choose to work at 30 fr/sec. Those in Europe have it easy, because on that continent 25 fr/sec is the logical choice for all music and video productions. On the other hand, those who work with projects that come through the door from other production houses will need to take special care to reference their time-code rates to those used by the originating media house. This can't be stressed enough: If care isn't taken to keep your timecode references at the proper rate and relative address times (while keeping degradation to a minimum from one generation to the next), the various media might have trouble syncing up when it comes time to put the final master together - and that could spell BIG trouble.

TIMECODE WITHIN DIGITAL MEDIA PRODUCTION

Given that SMPTE exists in a digitally encoded data form, current-day digital professional media devices are able to accept and communicate SMPTE directly without too much trouble. Professional camera, film, controllers and editing systems are able to directly chain and synchronize SMPTE using a multitude of complicated, yet standardized methods that make use of both digital- and analog-style timecode data streams.

Of course, there are a wide range of approaches that can be taken when media devices (cameras, video editing software and field audio recorders) are to be synchronized together. These can range from "shoots" that make use of multiple cameras and separate field recorders, which are "locked" to a single timecode source on a set ... all the way down to a simple camera and digital hand recorder, with audio that can be manually synced up within the digital editor, without the use of timecode at all. The types of equipment and the ways that they deal with the technology of sync are ever-changing. Therefore it's important to keep abreast of current technology, read the manuals (about how connections and settings can best be made) and dive into the study of visual media production.

Broadcast Wave File Format

Although digital media devices and software are able to import, convert and communicate using the language of SMPTE timecode (in all its various flavors), the folks at the EBU (European Broadcast Union) saw the need to create a universal audio file format that would include timecode data within all forms of audio and visual media production. The result was the Broadcast Wave Format (BWF). Broadcast Wave is in most ways completely compatible with its Microsoft Wave counterpart, with the exception that it is able to embed metadata (information about the recorded content ... photo, take#, date, technical data, etc.) as well as SMPTE timecode address data. The inclusion of such important content and timecode information means that the time-related information will actually be imbedded within the file itself, allowing soundfiles that are imported into a video or audio editor to automatically snap to their appropriate timecode position. Obviously, Broadcast Wave can be a huge time saver within the production and post-production process.

MIDI TIMECODE

In earlier times, the synchronization of audio devices to other video and/or audio devices was a very expensive proposition, far beyond the budget of most project or independent production houses. Today, however, an easy-to-use and inexpensive standard makes use of MIDI to transmit sync and timecode data throughout a connected production system (Figure 12.6). This has made it possible for even the most budget-minded bedrooms to be able to synchronize media devices and software using timecode.

[caption]**FIGURE 12.6** Many time-based media devices in the studio can be cost effectively connected via MIDI timecode (MTC).

MIDI timecode (MTC) was developed to allow electronic musicians, project studios, video facilities and virtually all other production environments to cost effectively and easily translate timecode into time-stamped messages that can be transmitted over MIDI data lines. Created by Chris Meyer and Evan Brooks, MIDI timecode allows SMPTE-based timecode to be distributed throughout the MIDI chain to devices or instruments that are capable of synchronizing to and executing MTC commands. MIDI timecode is an extension of the MIDI standard, making use of existing sys-ex message types that were either previously undefined or were being used for other, non-conflicting purposes.

Since most modern recording systems include MIDI in their design, there's often no need for external hardware when making direct connections. Simply chain the MIDI data lines from the master to the appropriate slaves within the system (via physical cables, USB or virtual internal routing). Although MTC uses a reasonably small percentage of MIDI's available bandwidth (about 7.68% at 30 fr/sec), it's customary (but not at all necessary) to separate these lines from those that are communicating performance data when using physical MIDI cables. As with conventional SMPTE, only one master can exist within an MTC system, while any number of slaves can be assigned to follow, locate and chase to the master's speed and position. Because MTC is easy to use and is often included free in many system and program designs, this technology has grown to become the most straightforward and commonly used way to lock together such devices as DAWs, external devices and basic analog and video setups.

MIDI Timecode Messages

The MIDI timecode format can be divided into two parts:

- Timecode
- MIDI cueing

The timecode capabilities of MTC are relatively straightforward and allow devices to be synchronously locked or triggered to SMPTE timecode. MIDI cueing is a format that informs a MIDI device of an upcoming event that's to be performed at a specific time (such as load, play, stop, punch-in/out, reset). This protocol envisions the use of intelligent MIDI devices that can prepare for a specific event in advance and then execute the command on cue.

MIDI timecode is made up of three message types:

- *Quarter-frame messages:* These are transmitted only while the system is running in real or variable speed time, in either forward or reverse direction. True to its name, four quarter-frame messages are generated for each timecode frame. Since 8 quarter-

frame messages are required to encode a full SMPTE address (in hours, minutes, seconds and frames: 00:00:00:00), the complete SMPTE address time is updated once every two frames (In other words, MIDI timecode actually has half the resolution accuracy of its SMPTE timecode counterpart). Each quarter-frame message contains 2 bytes. The first byte is F1, the quarter-frame common header; the second byte contains a nibble (four hits) that represents the message number (0 through 7) and a nibble for encoding the time field digit.

- *Full messages:* Quarter-frame messages are not sent in the fast-forward, rewind or locate modes, because this would unnecessarily clog a MIDI data line. When the system is in any of these shuttle modes, a full message is used to encode a complete timecode address. After a fast shuttle mode is entered, the system generates a full address message and then places itself in a pause mode until the time-encoded slaves have located to the correct position. Once playback has resumed, MTC will again begin sending incremental quarter-frame messages.
- *MIDI cueing messages:* MIDI cueing messages are designed to address individual devices or programs within a system. These 13-bit messages can be used to compile a cue or edit decision list, which in turn instructs one or more devices to play, punch in, load, stop, and so on, at a specific time. Each instruction within a cueing message contains a unique number, time, name, type and space for additional information. At the present time, only a small percentage of the possible 128 cueing event types have been defined.

SMPTE/MTC Conversion

Although MIDI timecode connections can be directly made between compatible MIDI devices, a SMPTE-to-MIDI converter is required to read incoming LTC SMPTE timecode and convert it into MIDI timecode (and vice versa) for other device types. These conversion systems are available as a stand-alone device or as an integrated part of an audio interface or multiport MIDI interface/patch bay/synchronizer system (Figure 12.7).

[caption]**FIGURE 12.7** SMPTE timecode can often be generated throughout a production system, possibly as either LTC or as MTC via a capable MIDI or audio interface.

TIMECODE PRODUCTION IN THE ANALOG AUDIO AND VIDEO WORLDS

Fortunately for us, most digital editing systems (such as digital video and audio workstations) are able to communicate timecode in a relatively seamless and straightforward manner (at least at a basic level, often only requiring that the various systems be set to the same framerates, etc). Synching analog-to-analog or analog-to-digital devices, on the other hand, is often far less straightforward and needs to be understood ... at least at a fundamental level.

Timecode that's recorded onto an analog audio or video cue track of an older-style video tape recorder is known as longitudinal timecode (LTC). LTC encodes a bi-phase timecode signal onto an analog track in the form of a modulated square wave at a bit rate of 2400 bits/sec. The recording of a perfect square wave onto a magnetic audio track is difficult, even under the best of conditions. For this reason, the SMPTE standard has set forth an allowable rise time of 25 ±5 microseconds for the recording and reproduction of valid code. This tolerance requires a signal bandwidth of at least 15 kHz, which is well within the range of most professional audio recording devices. Variable-speed timecode readers are often able to decode timecode information at shuttle rates ranging from 1/10th to 100 times normal playing

speed, which is often necessary when monitoring videotape at slow or near-still speeds.

Because LTC can't be read at speeds slower than 1/10th to 1/20th normal play speed, therefore whenever a VTR is used (which is uncommon these days), a character generator will be used to burn time-code addresses directly into the video image of a worktape copy. This superimposed readout allows the timecode to be easily seen and identified, even at very slow or still picture shuttle speeds.

It's nice to keep in mind, however, that in most modern-day production settings, LTC code won't be necessary whenever digital video and audio devices are involved.

TIMECODE REFRESH AND JAM SYNC

Longitudinal timecode operates by recording a series of square-wave pulses onto magnetic tape. As you now now know, it's somewhat difficult to record a square waveform onto analog magnetic tape without having the signal suffer moderate to severe waveform distortion. Although timecode readers are designed to be relatively tolerant of waveform amplitude fluctuations, such distortions are severely compounded when code is copied from one analog recorder to another over one or more generations.

Should the quality of a copied SMPTE signal degrade to the point where the synchronizer can't differentiate between the pulses, the code will disappear and the slaves will come to a stop. For this reason, a timecode refresher (Figure 12.8) has been incorporated into most timecode synchronizers and various MIDI interface devices that have sync capabilities. Basically, this process (known as jam sync) reads the degraded timecode information from a previously recorded track and then refreshes and regenerates the square wave back into its original shape, so it can be freshly recorded to a new track and accurately read by another device.

[caption]**FIGURE 12.8** Jam sync is used to restore distorted SMPTE when copying code from one machine to another.

Jam sync also refers to the synchronizer's ability to output the next timecode value, even though the next valid value has not appeared at its input. The generator is then said to be working in a freewheeling fashion, since the generated code may not agree with the actual recorded address values; however, if the dropout occurs for only a short period, jam sync can often detect or refresh the lost signal (This process is often useful when dealing with dropouts or undependable code from audio tracks on an analog video machine.) Two forms of jam sync options are available:

- Freewheeling
- Continuous

In the freewheeling mode, the receipt of timecode causes the generator's output to initialize when a valid address number is detected. The generator then begins to count in an ascending order on its own, ignoring any deterioration or discontinuity in code and producing fresh, uninterrupted SMPTE address numbers. Continuous jam sync is used in cases where the original address numbers must remain intact and shouldn't be regenerated as a continuously ascending count. After the reader has been activated, the generator updates the address count for each frame in accordance with incoming address numbers and outputs an identical,

regenerated copy.

SYNCHRONIZATION USING SMPTE TIMECODE

In order to achieve a frame-by-frame timecode lock between multiple audio, video or film analog transports, it's necessary to use a device or integrated system that's known as a synchronizer. The basic function of a synchronizer is to control one or more tape, computer-based or film transports (designated as slave machines) so their speeds and relative positions are made to accurately follow one specific transport (designated as the master).

The use of a synchronizer within a project studio environment (Figure 12.9) often involves a multiport MIDI interface that includes provisions for locking an analog audio or video transport to a digital audio, MIDI or electronic music system by translating LTC SMPTE code into MIDI timecode. In this way, one simple device can cost effectively serve multiple purposes to achieve lock with a high degree of accuracy. Systems that are used in video production and in higher levels of production will often require a greater degree of control and remote-control functions throughout the studio or production facility. Such a setup will often require a more sophisticated device, such as a control synchronizer or an edit decision list (EDL) controller.

[caption]**FIGURE 12.9** Example of timecode sync production using a simple MIDI interface synchronizer (possibly one that's already designed into an audio interface) within a studio setting.

SMPTE OFFSET TIMES

In the real world of audio production, programs or songs don't always begin at 00:00:00:00 (as might easily happen when using a video or audio workstation with an in-house project). Let's say that you were handed a recording that needed a synth track to be laid down onto track 7 of a song that goes from 00:11:24:03 to 00:16:09:21. Instead of inserting more than 11 minutes of empty bars into a MIDI track on your synched DAW, you could simply insert an offset start time of 00:11:24:03. This means that the sequenced track will begin to increment from measure 1 at 00:11:24:03 and will maintain relative offset sync throughout the program.

Offset start times are also useful when synchronizing devices to an analog or videotape source that doesn't begin at 00:00:00:00. As you're probably aware, it always takes a bit of time for an analog audio transport to settle down and begin playing (this wait time often quadruples whenever a videotape transport is involved). If a program's timecode were to begin at the head of the tape, it's extremely unlikely that you would want to start a program at 00:00:00:00, since playback would be delayed and extremely unstable at points near this time. Instead, most programming involving an analog audio or video media is striped with an appropriate pre-roll of anywhere from 10 seconds to 2 minutes. Such a pre-roll gives any analog transports ample time to begin playback and sync up to the master time-code source.

In addition, it's often wise to begin the actual production or first song at an offset or SMPTE start time of 00:01:00:00 (some facilities set the start offset at 01:00:00:00). This minimizes the possibility that the synchronizer will become confused by rolling over at midnight. That's to say, if the content starts at 00:00:00:00 (midnight), the pre-roll would be in the 23:59:00:00 range and the synchronizer would try to rewind the tape backwards to find 00:00:00:00

(rolling the tape backwards off the reel) instead of rolling forward. Not always fun in the heat of a production!

DISTRIBUTION OF SMPTE SIGNALS

Generally, when analog media devices are synced together, connections will need to be made between each transport and the synchronizer. These include lines for the LTC timecode reproduce track and the control interface (which often uses the Sony 9-pin remote protocol for giving the synchronizer full logic transport and speed-related feedback information). LTC signal lines can be distributed throughout the production system in much the same way that any other audio lines are distributed. They can be routed directly from machine to machine or patched through audio switching systems via balanced, shielded cables or unbalanced cables. It should be noted that because the timecode signal is bi-phase or symmetrical, it's immune to cable polarity problems.

Timecode Levels

One problem that can plague systems using timecode is crosstalk. This happens when a high-level signal leaks into adjacent signal paths or analog tape tracks. Currently, no industry standard levels exist for the recording of timecode onto magnetic tape or digital tape track; however, the levels shown in Table 12.1 can help you get a good signal level while keeping distortion and analog crosstalk to a minimum.

Table 12.1 Optimum Timecode Recording Levels

Tape Format	Track Format	Optimum Recording Level
ATR	Edge track (highest number)	–5 to –10 VU
Digital device	Highest number track or dedicated timecode I/O ports	–20 dB

Note: If the VTR is equipped with automatic gain compensation (AGC), override the AGC and adjust the signal gain controls manually.

REAL-WORLD APPLICATIONS USING TIMECODE AND MIDI TIMECODE

Before we delve into the many possible ways that a system can be set up to work in a timecode environment, it needs to be understood that each system will often have its own particular personality and that the connections, software and operation of one system might totally differ from those of another. This is often due to factors such as system complexity and the basic hardware types that are involved, as well as the type of hardware and software systems that are installed in a DAW. Larger, more expensive setups that are used to create television and film soundtracks will often involve extensive timecode and system interconnections that can easily get complicated.

Fortunately, the use of MIDI timecode and digital systems has greatly reduced the cost and complexity of connecting and controlling a synchronous pro and project studio system down to levels that can be easily managed by both experienced and novice users. Having said these things, I'd still like to stress that solving synchronization problems will often require as much intuition, perseverance, insight and art as it will technical skill. For the remainder of this chapter, we'll be looking into some of the basic concepts and connections that can be used to get your system up and running. Beyond this, the next best course of action will be to consult

your manuals, seek help from an experienced friend or call the tech department about the particular hardware or software that's giving both you and your system the willies.

Master/Slave Relationship

Since synchronization is based on the timing relationship between two or more devices, it follows that the logical way to achieve sync is to have one or more devices (known as slaves) follow the relative movements of a single transport or device (known as the master). The basic rule to keep in mind is that there can be only one master in a connected system; however, any number of slaves can be set to follow the relative movements of a master transport or device (Figure 12.10).

[caption]**FIGURE 12.10** There can be only one master in a synchronized system; however, there can be any number of slaves.

Generally, the rule for deciding which device will be the master in a production system (during the pre-planning phase) can best be determined by asking a few questions:

- What type of media is the master timecode media recorded on?
- Which device will provide the most stable timing reference?
- Which device will most easily and cost-effectively serve as the master?

If the master comes to you from an outside source, asking lots of questions about the source specs will most likely solve many of your problems. If the project is in-house and you have total say in the matter, you might want to research your options more fully, to make the best choice for your facility. The following sections can help give you insights into which devices will best serve as the master within a particular system.

Video's Need for a Stable Timing Reference

Whenever a video signal is copied from one machine to another, it's essential that the scanned data (containing timing, video and user information) be copied in perfect sync from one frame to the next. Failure to do so will result in severe picture breakup or, at best, the vertical rolling of a black line over the visible picture area. Copying video from one machine to another generally isn't a problem (because the video recording device that's doing the copying normally provides sync from the playback machine within the picture itself). Video postproduction houses, however, often simultaneously use any number of video and audio workstations, switchers and edit controllers during the production and editing of a single program. Mixing and switching between these combined sources without a stable sync source would almost certainly result in chaos ... again, with the end result being a very unhappy client.

Fortunately, referencing all of the video, audio and timing elements to an extremely stable timing source (called a black burst or house sync generator) will generally resolve this sync nightmare. This reference clock serves to synchronize the video frames and timecode addresses that are received or transmitted by nearly "every" video-related device in a production facility, so the leading frame edge of every video signal occurs at exactly the same instant in time (Figure 12.11). By resolving all video and audio devices to a single black burst reference, you're assured that relative frame transitions, speeds and TC addresses throughout the system will be consistent and stable.

[caption]**FIGURE 12.11** Example of a system whose overall timing elements are locked to a black burst reference signal.

Digital Audio's Need for a Stable Timing Reference

The process of maintaining a synchronous lock between digital audio devices or between digital and analog systems differs fundamentally from the process of maintaining relative speed between analog transports. This is due to the fact that a digital system generally achieves synchronous lock by adjusting its playback sample rate (and thus its speed and/or pitch ratio), so as to precisely match the relative playback speed of the master transport. Therefore, whenever a digital system is synchronized to a time-encoded master, a stable timing source is extremely important in order to keep jitter (in this case, an increased distortion due to rapid pitch shifts) to a minimum. In other words, the source's program speed should vary as little as possible to prevent any degradation in the digital signal's quality. As such, a digital audio system that's working within a video production environment would also benefit from the above mentioned house sync timing source.

Video Workstation or Recorder

Since video is often an extremely stable timing source, a digital video editor (or even analog video device) would be a stable timing source within a connected production system. This process still shouldn't be taken lightly, because the timecode must (in most cases) conform to the timecode addresses on the original video or working master. Basically, the rule of thumb is: If you're working on a project that was created out of house, always use the code that was provided by the original production team. Striping your own code or erasing over the original code with your own would render the original timing elements useless, because the new code wouldn't relate to the original addresses or include any timing variations that might be a part of the original master source. In short, make sure that your working copy includes a SMPTE track that a regenerated copy of the original code! Should you overlook this, you might run into timing and sync troubles, either immediately or later in the post-production phase - factors that will definitely lead to premature hair and client loss.

Digital Audio Workstations

A computer-based DAW can often be set to act as either a master or slave. This will ultimately depend on the software and the situation, because most professional workstations can be set to chase (to follow or be triggered by) a master timecode source, as well as generate timecode (often in the form of MIDI or SMPTE timecode within a higher-end system).

Most modern DAWs include support for displaying a video track (Figure 12.12) within a session (both as a separate video screen that can be displayed on the monitor desktop and in the form of a video thumbnail track that appears within the track view). Of course, the video track provides important visual cues for tracking live music, accurately placing automation moves and effects (sfx) at specific hitpoints within the scene or for adding music sweetening. This feature allows audio to be built up within a DAW environment without the need for synching to an external device at all. As you might expect, the use of recorded tracks, software instruments and internal mixing capabilities, tracks can easily be built up, spotted and mixed - all inside the box.

[caption]**FIGURE 12.12** Most high-end DAW systems are capable of importing a videofile directly into the project session window. (Courtesy of Apple Computers, Inc., www.apple.com)

Routing Timecode to and from Your Computer

From a connections standpoint, most DAW, MIDI and audio application software packages are flexible enough to let you choose from any number of available sync sources (whether connected to a hardware port, MIDI interface port or virtual sync driver). All you have to do is assign all of the slaves within the system to the device driver that's generating the system's master code (Figure 12.13). In many cases, it's best to have your DAW or editor generate the master code for the system with the appropriate settings and timecode address times.

[caption]**FIGURE 12.13** Cubase/Nuendo Sync Preferences dialog box. (Courtesy of Steinberg Media Technologies GMBH, www.steinberg.net)

Analog Audio Recorders

In many audio production situations, whenever an analog tape recorder is connected in a timecode environment, this machine will most often want to act as the master in an LTC environment. Although this might be counter-intuitive, it's far easier and less expensive for an analog recorder to output a master SMPTE code, than to be controlled in an external slave relationship. This is because special and expensive equipment is generally required to continuously adjust the regulator's speed (using a DC capstan servo), so as to maintain a synchronous relationship to the master SMPTE address.

A Simple Caveat

The above guidelines are just that - guidelines. As you might expect, each and every setup will be slightly different, and might require that you come up with a novel solution to a quirky problem. Again, the internet is full of insights and solutions from those who have already gone down that long and treacherous path. Be warned though. It's important that you prepare and make your decisions wisely, lest a problem raise its ugly head at a later and crucial time.

KEEPING OUT OF TROUBLE

Here are a few guidelines that can help save your butt when using SMPTE and other timecode translations during a project:

- Familiarize yourself with the hardware and software involved in a project "BEFORE" the session starts.
- When in doubt about frame rates, special requirements or anything else, for that matter ... ask! You (and your client) will be glad you did.
- Fully document your timecode settings, offsets, start times, etc.
- If the project isn't to be used in-house, ask the producer what the proper frame rate should be. Don't assume or guess it.
- When beginning a new session (when using a tape-based device), always stripe the master contiguously from the beginning to end before the session begins. It never

hurts to stripe an extra tape, just in case.

- Whenever analog machines are involved, start generating new code at a point after midnight (i.e., 00:01:00:00 or 01:00:00:00 to allow for a pre-roll). If the project isn't to be used in-house, ask the producer what the start times should be. Don't assume or guess it.
- Never dub (copy) timecode directly. Always make a refreshed (jam synched) copy of the original timecode (from an analog master) before the session begins.
- Disable noise reduction and AGC (Automatic Gain Control) on analog audio tracks (on both audio and video devices).
- Work with copies from the original production video, and make a new one when sync troubles appear.
- It's not unusual for the timecode to be read incorrectly (when short dropouts occur on the track, usually on videotape). When this happens, you might set the synchronizer to freewheel once the transports have initially locked.

In closing, I'd like to point out that synchronization can be a simple procedure or it can be an extremely complex one, depending on your requirements and the type of equipment that's involved. A number of books and articles have been written on this subject. If you're serious about production, I suggest that you do your best to keep up on it. Although the fundamentals often remain the same, new technologies and techniques are constantly emerging. As always, the best way to learn is simply by reading and then jumping in and doing it.