

Background

What is synchronization?

Synchronization is said to exist when you make two pieces of equipment agree on time or tempo and position info. You can establish synchronization between Cubase and a number of other types of devices, including tape recorders and video decks, but also MIDI devices that “play back”, such as other sequencers, drum machines, “workstation sequencers” etc.

When you set up a synchronization system, you must decide which unit is the master. All other devices are then slaved to this unit, which means they will adjust their playback speed to the master’s.

⚠ For a description of the VST System Link feature (with which you can synchronize separate computers running Cubase or Nuendo for example), see “[Working with VST System Link](#)” on [page 431](#).

Cubase as slave

When a synchronization signal is coming in to Cubase, from another device, this device is the master and Cubase is the slave. Cubase will adjust its playback to the other device.

Cubase as master

When you set up Cubase to transmit synchronization information to other devices, Cubase is the master and the other devices are the slaves; they will adjust their playback to Cubase.

Cubase – both master and slave

Cubase is a very capable synchronization device. It can operate as both a master and a slave at the same time. For example, Cubase might be slaved to a tape recorder transmitting timecode, while at the same time transmitting MIDI Clock to a drum machine, acting as a master for that.

Synchronization signals

Basically there are three types of synchronization signals for audio: timecode, MIDI clock and word clock.

Timecode (SMPTE, EBU, MTC, VITC etc.)

Timecode appears in a number of guises. No matter which “format” it has, it always supplies a “clock on the wall” type of synchronization, that is, a synchronization related to hours, minutes, seconds and two smaller units called “frames” and “subframes”.

- LTC (SMPTE, EBU) is the audio version of timecode. This means that it can be recorded on the audio track of an audio or video recorder.
- VITC is the video format timecode, i.e. it is stored in the actual video image.
- MTC is the MIDI version of timecode, transmitted via MIDI cables.
- ADAT sync (Alesis) is only used with the ASIO Positioning Protocol, see “[About the ASIO Positioning Protocol \(APP\)](#)” on [page 427](#).

For the ASIO Positioning Protocol, other high precision timecode formats may also be supported.

Format recommendations for timecode – without ASIO Positioning Protocol

- When synchronizing your system to external timecode, via a synchronizer, the most common timecode format is MTC. Contrary to some reports you might have heard, MTC delivers good precision for external sync. This is due to the fact that the operating system can “time stamp” incoming MIDI messages, which increases precision.

Format recommendations for timecode – with ASIO Positioning Protocol

- LTC and VITC are the formats with the highest precision and are recommended when available.
- MTC is the next best option and probably the most common choice, since few audio hardware solutions have built-in LTC or VITC readers. However, LTC and VITC offer even higher precision when available.

MIDI Clock

MIDI Clock is a tempo-based type of synchronization signals, i.e. it is related to the number of “beats per minute”. MIDI Clock signals are suitable for synchronizing two devices that agree on tempo, such as for example Cubase and a drum machine.

⚠ MIDI Clock is not suitable as a master sync source for an application like Cubase. Therefore Cubase will transmit MIDI Clock signals to other devices, but it will not receive MIDI Clock.

Word Clock

Word clock is basically a replacement for the sample rate clock in for example an audio card. Word clock hence runs at the same rate as the sample rate in the audio, 44.1kHz, 48kHz etc.

Word clock does not contain any position information, it is only a “simple” signal for clocking the audio at its sample rate.

Word clock comes in many formats, analog on coaxial cable, digital as part of an S/PDIF, AES/EBU or ADAT audio signal, etc.

Synchronizing the transport vs. synchronizing audio

How timing is handled in a non-synchronized system

Let’s first look at the situation where Cubase is not synchronized to any external source:

Any digital playback system has an internal clock that ultimately affects the playback speed and stability, and PC audio hardware is no exception. This clock is extremely stable.

When Cubase is playing back with no external synchronization, all playback is internally synchronized to the internal digital audio clock.

Synchronizing Cubase’s playback

Let’s assume now that we use external timecode synchronization with Cubase. For example, we might synchronize playback to a tape recorder.

Timecode coming from an analog tape recorder will always vary slightly in speed. Different timecode generators and different tape recorders will also supply timecode with slight differences in speed. In addition, the shuttling of tape mechanisms due to overdubs and re-recordings can cause the physical tape to wear and stretch, which affects the speed of the timecode.

If you use a synchronizer that generates word clock and set up Cubase to sync to incoming timecode, it will vary its overall playback speed to compensate for such fluctuations in the speed of the timecode – that’s the whole purpose of synchronization.

What happens with the digital audio?

The fact that Cubase’s playback is synchronized to the timecode does not affect the playback of the digital audio. It still relies on the perfectly stable, built-in clock in the audio hardware.

As might be expected, problems will appear when the perfectly stable digital audio gets related to the slightly varying speed of a system synchronized to timecode.

The playback timing of each event will not be in total accordance with the tape or the MIDI playback, since the playback speed of the audio is determined by the digital audio hardware’s built-in clock.

Resolving to word clock

The solution to this problem is to use one external clock for all components in the system. One master clock is used to derive whatever type of clock signal each component in the system needs. For example, something called a house clock can be used to generate sample rate clocks for the digital audio hardware and timecode for Cubase. This ensures that all components in the system use the same reference source for their timing.

Synchronizing digital audio to external clocks running at sample rate is often called “resolving” or “synchronizing to word clock”.