A new level
of show

Roma
Robot

RU

SYNC

Roma Robot

Sync a New Level of Show

«Автор»
2022

Robot R.

Sync a New Level of Show / R. Robot — «Автор», 2022

Цель этой книги — объяснить то чудо, которое позволяет создать технологии синхронизации, и дать вам инструменты для его осуществления. Шоу-синхронизация является одним из важнейших элементов представления, который не прощает ошибок. Эта книга позволит вам приобрести необходимую уверенность в этой технологии, и вы с лёгкостью сможете реализовывать самые сложные и невероятные проекты с синхронизацией.Вы готовы?Тогда добро пожаловать в увлекательное путешествие.

Содержание

Roma Robot
Sync a New Level of Show

Наследие Donald J. Childs

Дональд… каким он останется в нашей памяти? Театральным художником, уважаемым педагогом, наставником, ценным другом и создателем: его авторству принадлежит художественное оформление огромного количества (более 200) представлений и мюзиклов. Своими трудами он внес большой вклад в развитие мировой шоу-индустрии. Но лучшим из лучших проектов Дональда стали вдохновляющие отношения с супругой и бизнес-партнером по совместительству – Джейн Чайлдс. В 2006 году эта знаменитая чета основала Stagecraft Institute of Las Vegas – восьминедельную интенсивную программу обучения, проходящую на действующих шоу в городе Лас-Вегас с участием лучших специалистов со всего мира. Такой потрясающий симбиоз дает возможность SILV воспитывать новое поколение профессионалов. Все выпускники этого института становятся последователями главной миссии его основателей – нести полученные знания и опыт дальше, делиться ценным багажом информации с другими. Этот ценный, очень заряженный импульс способствует общему развитию шоу-индустрии во всем мире. Я являюсь выпускником и преподавателем SILV. И я следую объявленной миссии: я передаю свои знания и опыт следующим поколениям специалистов. Дональд Дж. Чайлдс покинул этот мир, но его богатое наследие продолжает жить дальше.

Это не энциклопедия

Хочу с самого начала расставить все точки над «i».

Если вы ожидаете, что эта книга является своего рода супер точным справочником, где вы получите невероятно полную информацию о шоу протоколах и их составляющих вплоть до каждого байта, то лучше вам не разочаровывать себя и взять в руки другую книгу, которая сможет удовлетворить ваш интерес в этом направлении.

Эта книга другого характера, она об идее синхронизации. Эта книга не преследует цели свалить на вас тонну информации. Цель этой книги – поделиться с вами идеей и, самое главное, научить вас эту идею использовать. По этой причине я опустил много технических деталей и исторических уточнений. Ни один самый полный технический словарь не научит вас работать с технологией. Я читал такие технические книги, где авторы старались вставить все, что только можно и максимально подробно. После прочтения двух страниц мой мозг настолько переполнялся ненужной информацией, что дальше ничего не мог усвоить. И спрашивается, зачем такие книги нужны? Поставить на полку как справочник? Или все же цель такого рода книг – научить?

Итак, если вам нужна точная энциклопедия, то поищите что-то другое. Ну а если вы хотите научиться идее Синхронизации, то добро пожаловать в увлекательное путешествие.

PROLOGU. A NEW LEVEL OF SHOW

Свет в зале гаснет. Взгляды зрителей в ожидании прикованы к сцене. После небольшой паузы режиссер по рации командует: «Поехали!». Занавес открывается. А потом происходит нечто удивительное. Зритель попадает в волшебный мир, в котором он никогда не был. Магия происходящего захватывает каждого, кто находится в зале. Все вместе замирают от удивления. Вместе плачут. Вместе кричат от восторга. Свет, звук, декорации, артисты создают единый образ, новое пространство, уникальную реальность. Люди нескоро забудут то, что они увидели, то, что они почувствовали.

С древних времен наши предки собирались по вечерам у костров, и охотники рассказывали им захватывающие истории. Со временем это развилось в профессиональную шоу-индустрию с использованием современных технологий. Театральный свет прошел путь от простой свечи до сложных световых приборов, были созданы мощные усиливающие звуковые системы, технологии декораций теперь позволяют создать любой объект. На каждом этапе индустрия развлечений поднимается на новый уровень. И как логичный виток развития, в нее вошла технология синхронизации. Ведь только когда все элементы представления работают вместе, появляется волшебство, встречи с которым зритель с волнением ждет месяцами.

Цель этой книги – объяснить то чудо, которое позволяет создать технологии синхронизации, и дать вам инструменты для его осуществления. Но кое-что нужно подчеркнуть с самого начала. Синхронизация – это сложное направление, которое не терпит непрофессионализма и халтуры. В визуальных выразительных средствах – таких, как свет или видео – небрежное отношение к работе можно прикрыть творческим видением или художественной задумкой. Повесил двести световых приборов Sharpy, накинул эффекты по движению и по цвету – и неискушенный зритель примет это на ура. Шоу-синхронизация является одним из важнейших элементов представления, который не прощает ошибок.

В шоу могут быть использованы самые передовые технологии, а в его основе может лежать гениальный сценарий, но, если в нужный момент произойдет сбой из-за ошибки в синхронизации, толку от всех остальных шоу-систем не будет. Синхронизация требует знаний, работы с протоколами, четкого понимания особенностей каждого интерфейса передачи данных. Необходимо быть готовым ко всем сюрпризам, с которыми можно встретиться на концертной площадке.

Инженер синхронизации – это специалист широкого профиля, который должен иметь представление о каждой шоу-технологии, задействованной в представлении (чтобы их синхронизировать).

В любой области профессионала отличает наличие знаний и опыта. Последний появляется со временем, а для получения знаний нужен источник. Когда я впервые всерьез задумался о синхронизации, начал искать информацию. Я смог найти кое-какие технические справки о протоколах, но без общей концепции, без идеи применения этого на практике в реальных условиях. Существовало много ошибочных предположений относительно различных технологий и их использования (это мне впоследствии пришлось выяснять самому на практике). Информация была разрознена и не структурирована. Основная причина такого положения дел заключалась в том, что не существовало как такового подхода или идеи синхронизации. Не существовало общего учения. Я читал книги. Общался с коллегами. Мы спорили, находили истину. Кто-то делился своим опытом. На моих семинарах студенты задавали вопросы, которые для меня были абсолютно новыми, и это побуждало меня изучать новые направления. Всю собранную информацию я записывал и обязательно проверял на проектах. Что-то работало, а что-то – нет.

Вот почему моя книга – плод труда многих людей. В ней собраны наработки и опыт многих специалистов, причем не только из шоу-индустрии. Если, читая ее, вы найдете то, что

уже не раз где-то применяли, не исключено, что когда-то я был вдохновлен именно вами и вашим трудом. В этой книге также есть мой опыт и мои собственные знания. Все это вместе есть общая идея использования синхронизации, которая наконец обрела структуру и форму. Теперь и у вас появилась возможность познать эту идею.

Вы готовы?

Давайте же поднимем наше шоу на новый уровень!

SYNCHRONIZATION

Несмотря на то, что источников информации об этом направлении немного, я не был первым, кто задался вопросом синхронизации. Художник и писатель Остин Клеон говорил: «Все давным-давно уже придумано и кем-нибудь уже использовано. Если вы считаете, что ваша идея оригинальна, значит, вы просто не нашли первоисточника».

Я точно могу сказать, что моя идея синхронизации неоригинальна, потому что я нашел первоисточник.

Первым, кто задумался о синхронизации выразительных средств в своем творчестве, был гений, композитор, музыкант Александр Николаевич Скрябин. Одна из его идей заключалась в том, что все в этом мире едино. Все существует в тесной связи между собой. И, в частности, звук неразрывно сосуществует со светом.

Апогеем его работы стало первое в мире произведение, для исполнения которого были написаны нотная и световая партитуры. Это музыкальная поэма «Прометей» («Поэма огня»). Он считал, что только звук и свет могут, задействованные вместе, передать всю полноту произведения. Но, к большому сожалению, Скрябину не суждено было увидеть результат своего труда в том виде, в котором он его создал. В те времена не существовало световых приборов, способных воссоздать те эффекты, которые описал Скрябин. Он говорил: «Я очень надеюсь на то, что когда-нибудь человечество сможет увидеть это произведение в своей полной красе и величии, в том виде, как я его задумал». До сих пор «Прометей», или «Поэма огня», считается шедевром. Некоторые художники пытались повторить световые образы, которые описывал Скрябин, но и по сей день никто не смог воссоздать задуманное им для своего произведения световое оформление.

Технологии современного мира шоу-индустрии постоянно развиваются. Когда-то большим достижением считались запись звука на грампластинке и его воспроизведение на граммофоне, а сейчас нас уже не удивить технологиями объемного звучания с абсолютным погружением. Когда-то на спектаклях у каждого светового прибора стоял человек, который управлял его параметрами. А сейчас мы можем на расстоянии сотни метров одной кнопкой изменить состояние тысячи сложных технических световых приборов. Когда-то запуск звука и света на концертах синхронизировался на счет «раз, два, три[1]», сейчас современные технологии позволяют синхронизировать все элементы шоу-представления со сложнейшими сценариями. Прогресс идет вперед, и использование синхронизации становится таким же очевидным, как сохранение шоу-файла светового пульта на USB-носителе.

Но, в отличие от большинства благ прогресса, технология синхронизации может как преобразить наше шоу, так и убить его. Нужно трезво оценивать, где это уместно, а где нет. Вы удивлены? Вы думали, что я, как ярый сторонник синхронизации, буду проповедовать ее использование везде и всюду? Это не так.

Я хочу, чтобы вы поняли, что синхронизация – это не простой инструмент. Его необходимо применять только там, где это нужно (с вами Кэп). Хорошо, так в каких же именно случаях?

Необходимо обеспечить постоянную повторяемость воспроизведения шоу.

Например, спектакль или шоу-представление много раз показывается на конкретной площадке или ездит в тур целый сезон (может, даже не один сезон). Художники и режиссеры

[1] На профессиональном сленге такой способ синхронизации шутливо называется «РазДваТри- Код» – по аналогии с названием «Таймкод».

создают представление, а воспроизводят его операторы. И каждый раз, когда меняется оператор, необходимо его обучать и вводить в спектакль заново. Довольно непростой процесс, который занимает много времени и сил. В случае использования синхронизации вам нужны лишь те люди, которые следят за исправностью разных систем, а вот задача воспроизведения спектакля ложится на шоу-сервер, в котором прописана вся партитура и который синхронизирует все системы. При этом от представления к представлению спектакль или шоу будут всегда воспроизводиться одинаково с высоким качеством, которое изначально закладывали режиссер и художник по свету.

В шоу задействовано большое количество систем, которые работают вместе.

Если у нас большое представление с использованием десятка самых различных служб, которые должны работать вместе, то зачастую концентрация действий на сцене очень высокая, и выпускающий режиссер или шоу-колл не способны все проконтролировать. В этом случае мы можем синхронизировать каждую службу с главным шоу-сервером – и все действия будут выполнены четко по сценарию.

Человеческой реакции недостаточно, чтобы сделать все необходимые переключения аккуратно.

Существует максимально возможная скорость, с которой человек способен нажимать клавиши на пульте. Когда ваше представление открывает динамичное интро, где под музыку за одну секунду происходит с десяток переключений световых сцен, каким бы профессионалом оператор световой консоли ни был, он просто не способен воспроизвести подобное в силу ограниченности человеческих возможностей. В этом случае на помощь приходит синхронизация, которая поможет световой консоли выполнить все команды максимально четко, аккуратно и в нужный момент.

Телевизионное шоу или шоу, где используется прямая трансляция, при которой цена ошибки слишком велика.

Когда мы работаем на шоу-представлении с прямой трансляцией, то нам необходима гарантия успешного воспроизведения нашего шоу всего один раз в момент трансляции. Второй попытки у нас не будет! Синхронизация помогает максимально обезопасить шоу от различных провалов, связанных с человеческим фактором. Яркий пример такого шоу – «Евровидение», трансляция которого ведется на весь мир.

И последнее: мы создаем профессиональное шоу.

Один из обязательных аспектов любого эффектного шоу – единая связь всех выразительных средств: артиста, музыки, света, видео, спецэффектов. В таком случае все элементы представления сходятся в одном большом пазле. Это одно из множества условий, благодаря которому зритель начинает верить в то, что происходит на сцене, и погружаться в тот волшебный мир, который мы с вами создаем!

Синхронизация – это дирижер на представлении. Чтобы симфония нашего шоу зазвучала, все службы должны исполнять свои партии синхронно!

SYNC TYPE CLASSIFICATION

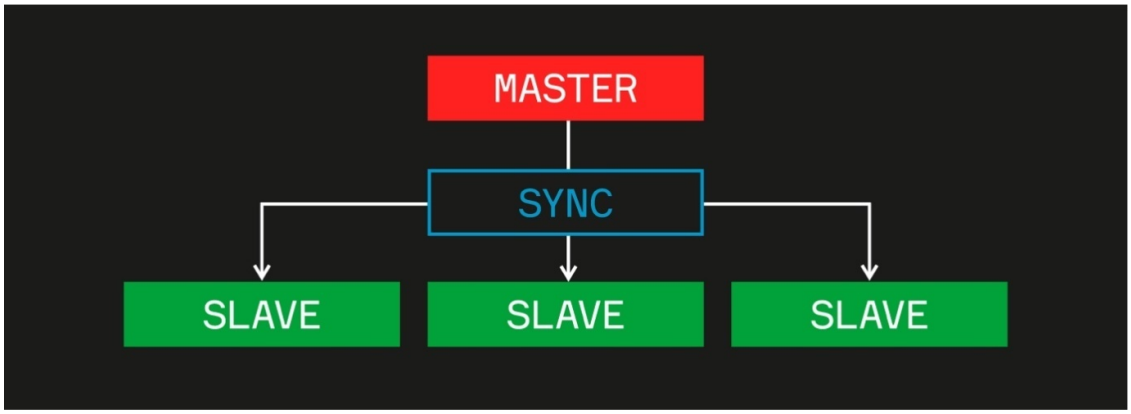
Итак, давайте начнем наше захватывающее путешествие в мир синхронизации. Сперва давайте познакомимся с тем, что нас ожидает и с чем нам придется иметь дело.

Любую систему синхронизации можно описать простой схемой:

MASTER – SYNC – SLAVE

MASTER – это главное управляющее устройство.

SLAVE – это подчиненные исполняющие устройства.



В принципе, этой топологией можно описать любую систему: световая консоль —> световые приборы, звуковой пульт —> система звуковоспроизведения, видеосервер —> видеопроекторы и т. д. Такое базовое определение необходимо для того, чтобы понимать, что чем управляет и как.

Если дать общее определение, то синхронизация – это канал связи между главными и подчиненными системами, который определяется протоколами и типами синхронизации. При взгляде на современную шоу-индустрию мы обнаружим, что количество типов оборудования, производителей, новинок и каждый год, огромное количество. Как же все это соединить в одну систему, спросите вы? Ведь у каждого производителя свои протоколы, оборудование, стандарты. Но все не так страшно. Когда-то производители делали свои системы закрытыми, чтобы они могли работать только с выпущенным ими оборудованием. Сейчас ситуация изменилась в лучшую сторону. Так как технологии не стоят на месте и постоянно разрабатываются новые протоколы и способы передачи данных, производители давно подметили, что намного целесообразнее делать поддержку универсальных протоколов в своих технических и программных решениях. И протоколы синхронизации не стали исключением. Исходя из этого, не имеет значения, насколько разные системы вы используете. Если они умеют работать с протоколами синхронизации, значит, все эти системы могут работать вместе.

Мы обязательно затронем каждую составляющую общей системы MASTER SYNC SLAVE, но сейчас поговорим о классификации протоколов и интерфейсов синхронизации. Поняв это, вы с легкостью сможете определять, какой протокол и какой интерфейс передачи данных необходим для каждой из поставленных задач.

Протоколы синхронизации

Временные	Командные	Генеративные
LTC	MIDI note	MIDI note
MTC	MIDI CC	MIDI CC
VITC	MIDI PC	MIDI PC
TCnet	MSC	MSC
ArtTimeCode	MMC	OSC
OSC Timecode	OSC	ArtNet, sACN
	Time Clock	TCP/UDP
	TCnet	

Итак, как понятно из названия, временной тип протоколов синхронизации обусловлен временем. Это означает, что основная задача такого протокола – кодировать передаваемое время. Все протоколы этой группы могут кодировать данные реального времени, кадры, секунды, минуты, часы и даже дату, в отличие от Time Clock, который находится в другой группе, так как этот протокол кодирует не время, а аудиобит.

Все вы, скорее всего, сталкивались с таким понятием, как BPM (Beats Per Minute). Этот показатель определяет скорость исполнения композиции. По сути, это метроном, который задает такт воспроизведения. Именно он и передается по протоколу Time Clock. Основная задача этого протокола – синхронизировать такт мастер-устройства и подчиненного устройства без какой-либо привязки ко времени.

Некоторые протоколы синхронизации могут передавать не только время таймкода, но и сообщения команд, поэтому они заняли место как в первой, так и во второй группе (например, протокол TCnet).

Хочу еще немного затронуть особенность VITC. Это временной протокол, который пишется на ленту в пространстве между кадрами. Основная сфера его применения – телевидение. В шоу-индустрии не так много систем, которые могут использовать этот протокол, и поэтому он практически не применяется для синхронизации. То же самое и с Time Clock. Из-за того, что он не передает реальное время, широкого применения в шоу-синхронизации этот протокол также не нашел.

Командные типы протоколов передают команды. Все очень просто и понятно. Единственное отличие между ними – это гибкость и вариации команд, которые может передать тот или иной протокол. К примеру, MIDI note может передать только команду, в которой содержится

информация о том, какую ноту воспроизвести и с какой громкостью. Также типы протоколов MIDI CC (Control Change) и MIDI PC (Program Change) очень схожи по простоте с MIDI note. Отличие только в том, что сообщения этих типов меняют настройки звукового синтезатора.

В то время, как MSC (MIDI Show Control) может передать конкретную команду конкретному устройству (которых в сети может быть несколько), MMC (MIDI Machine Control) может передавать команды на воспроизведение, остановку, выбор трека и время старта для профессиональных плееров и генераторов. А вот OSC (Open Sound Control) вообще никак не ограничивается синтаксисом передаваемых команд. Но об этом чуть позже.

Генеративный тип протоколов появился совсем недавно. Как вы могли заметить, в него входят все типы протоколов, которые базируются на командной идеологии. В чем же отличие? Чтобы понять это, давайте снова вернемся к временным протоколам. Все эти протоколы базируются на постоянной онлайн-синхронизации. Это означает, что сообщения о времени поступают от мастер-устройства на исполняющие устройства постоянно с определенной частотой, и синхронизация работает до тех пор, пока идет генерация сообщений. В случае командных типов протоколов синхронизация происходит только в момент отправки сообщения по конкретному событию, которое никак не привязано ко времени, то есть постоянная синхронизация отсутствует.

А вот генеративная синхронизация базируется на идее постоянной синхронизации как временной тип, но реализуется через командные протоколы, то есть сообщения о командах от мастер-устройства к исполняющим устройствам передаются постоянно с определенной частотой. Для чего это может быть нужно? Самый банальный пример – передача звукового уровня по цифровому протоколу и синхронизация определенных событий по громкости. Или передача координат для синхронизации положения луча на сцене.

Мы также можем заметить в генеративной группе такие типы, как ArtNet, sACN и TCP/UDP. ArtNet и sACN – это протоколы, которые используются в световой индустрии, но в силу своей высокой скорости и простоты они нашли широкое применение и в других направлениях шоу-индустрии. Их часто используют в проектах, где необходимо передать генеративные данные.

TCP (Transmission Control Protocol и UDP (User Datagram Protocol)—это низкоуровневые транспортные сетевые протоколы, на которых базируется большая часть всех протоколов, передающихся через Ethernet. Например, ArtNet базируется на протоколе UDP. Позже мы разберем особенности этих протоколов и то, как они помогают нам, когда мы работаем с сетью.

Вообще, TCP/UDP-протоколы нельзя ставить в один список с высокоуровневыми протоколами синхронизации, но тем не менее я включил эти базовые протоколы в генеративный тип, так как существует еще множество протоколов синхронизации, которые могут использоваться в различных направлениях шоу-индустрии, а также может возникнуть необходимость создать новый протокол для реализации необходимых задач.

Один из ярких примеров – система управления ETC Eos, которая позволяет принимать и отправлять UDP-строку. Как это может быть реализовано в системах синхронизации, мы разберем чуть позже, когда будем говорить о технологиях передачи данных стандарта Ethernet, а также когда мы ближе познакомимся со всем функционалом системы управления ETC Eos.

От того, какой тип протокола мы используем, зависит тип синхронизации с конечными клиентами. Чтобы понять принцип работы каждого типа протокола, давайте разберем три простейших примера.

Солнечный день в лесу. Вини Пух занят своими делами. Вини отлично знает, что ровно в час дня наступает время обеда. Утро пролетело быстро и вот на часах ровно час дня и Вини спешит полакомиться медом, ведь такое совсем ни как нельзя пропустить! Это пример временной синхронизации.

Пятачок решил помочь Кролику. Нужно было вскопать целый огород с морковкой. Кролик сказал Пятачку, что закончить свою работу он может, только тогда, когда он придет и даст команду: «Работа закончена». Бедный Пятачок не знает, когда он закончит, морковки столько что ему и за день не управиться, ему приходиться работать и работать, пока Кролик наконец не вернется. Это пример командной синхронизации. Кстати, с Пятачком все в порядке, Кролик все же пришел и дал команду: «Работа закончена!».

Пяточек шел с Вини в гости к Кролику. Вини знал где живет Кролик, а Пятачок нет, поэтому все дорогу мучал Винни вопросами:

Мы пришли?
Нет.
А сейчас?
Нет!
А сейчас?
И сейчас нет!
А сейчас?
Тоже нет!
…

К счастью, дорога к Кролику заняла не много времени и друзья благополучно дошли. В итоге на свой вопрос Пятачок получил утвердительный ответ. Это пример генеративной синхронизации.

Интерфейсы синхронизации

LTC	MIDI	ETHERNET	Bluetooth	SDI
SMPTE	MTC	OSC	BLE-MIDI	VITC
	Time Clock	ArtNet, sACN		
	MIDI note	TCP/UDP		
	MIDI CC	RTP-MIDI		
	MIDI PC	TCnet		
	MSC	ArtTimeCode		
	MMC	OSC Timecode		

Каждый протокол синхронизации для передачи данных использует конкретный физический интерфейс. Более подробно особенности каждого интерфейса мы затронем в главе «Оборудование и коммутация», а сейчас давайте познакомимся с общей классификацией.

Интерфейс LTC (Longitudinal Time Code) – это, по сути, простой балансный аудиокабель, который может передать только линейный таймкод SMPTE. LTC представляет собой закодированный цифровой сигнал, передаваемый по аудиоканалу.

MIDI-интерфейс немного шире по функционалу: это цифровой серийный интерфейс передачи данных. Так как он цифровой, то можно создать для передачи по нему любые протоколы. Что и делали производители много лет назад. В таблице указаны только те протоколы, которые прижились и нашли широкое применение в шоу-индустрии. Исходя из названия большинства протоколов, можно понять, на каком интерфейсе они базируются – MTC (MIDI Time Code), MIDI note, MIDI Show Control.

Ethernet – довольно универсальный интерфейс, который применяется практически везде, особенно в профессиональных сферах. Этот интерфейс используют все протоколы, которые базируются на TCP/UDP. И тут же вы можете заметить новый протокол – RTP-MIDI. По сути, это все тот же MIDI, только передается он уже по Ethernet.

BLE-MIDI очень схож с RTP-MIDI. Его ключевое отличие в том, что этот протокол использует беспроводной интерфейс передачи данных Bluetooth.

SDI – это универсальный коаксиальный интерфейс передачи цифрового видеосигнала, который также передает и VITC-таймкод. Существует некоторое количество оборудования, которое может читать по SDI-интерфейсу таймкод, но в шоу-индустрии для синхронизации он практически не используется, поэтому подробно разбирать его мы не будем.

Подводя итог, можно коротко сказать, что основное различие между протоколом и интерфейсом в том, что протокол— это то, что мы передаем, а интерфейс – через что мы передаем.

Протокол – это правила, по которым описываются сообщения, а интерфейс – это физический канал связи, по которому эти сообщения передаются.

Исходя из этого, совсем не грамотно называть физический XLR- порт синхронизации SMPTE-портом, так как это название стандарта кодирования времени, а физический интерфейс передачи таймкода SMPTE называется LTC. Ведь тот же самый SMPTE, передаваемый по интерфейсу MIDI, называется уже MIDI Time Code (MTC).

И еще одна особенность: SMPTE в оригинальном формате кодирования передается только через LTC. А через все остальные интерфейсы передается его адаптированная под конкретный интерфейс передачи данных форма. Эту особенность мы сможем рассмотреть при сравнении линейного и MIDI-таймкода в следующей главе.

Теперь, когда информация структурирована, давайте разберем особенности каждого протокола и интерфейса: как их генерировать, как их принимать, как избежать ошибок и, самое главное, как гарантированно запустить синхронизацию.

TIME SYNC PROTOCOLS. HOW SMPTE WORKS

Таймкод в шоу-индустрию пришел с телевидения. И называется он SMPTE. Аббревиатура расшифровывается как Society of Motion Picture and Television Engineers. Это ассоциация, которая приняла единый стандарт кодирования времени, разработанный в 1967 году. Чуть позже к ним присоединился EBU (European Broadcasting Union). Сейчас существует множество других стандартов записи таймкода, которые намного удобнее и функциональнее, но старый добрый SMPTE до сих пор жив и широко используется в шоу-индустрии.

LTC (Longitudinal Time Code)

Первая технология записи движущегося изображения появилась в кино. Изображение писалось на кинопленку кадр за кадром. Это был довольно большой прорыв. Позже, в 1920 году, на свет появилось телевидение, которое очень активно развивалось. Но мало кто знает, что долгое время все телевизионные программы передавались в прямом эфире! Тогда не существовало технологий записи и воспроизведения изображения для ТВ-трансляций. Первые ТВ-камеры были сделаны на основе иконоскопа. Это как кинескоп, который используется для формирования изображения, только наоборот: он преобразует видимый свет в электрический сигнал, который напрямую транслировался в эфир. Способов для записи такого сигнала тогда не существовало. Конечно, позже появились специальные технологии записи и трансляции телепередач и фильмов, но принцип оставался тот же самый: кинопроектор проецировал изображение напрямую в телекамеру. Правда, при таком способе качество изображения падало в разы.

AMPEX quadruplex VR-1000A, первый коммерчески выпущенный в конце 1950-х годов видеомагнитофон, использующий поперечно-строчную ленту с открытой катушкой шириной 2 дюйма

В 1956 году на свет появился первый видеомагнитофон (Video Tape Recorder), который мог записывать изображение и звук в аналоговом формате на ленту и воспроизводить эту информацию. Позже появилась необходимость также записывать информацию о времени, которая была бы синхронна с видеоизображением. Так как на видеоленте было несколько дорожек, предназначенных для записи аудио, одна из этих дорожек использовалась для записи таймкода, где в аналоговом формате был записан SMPTE-код. Такой способ записи SMPTE

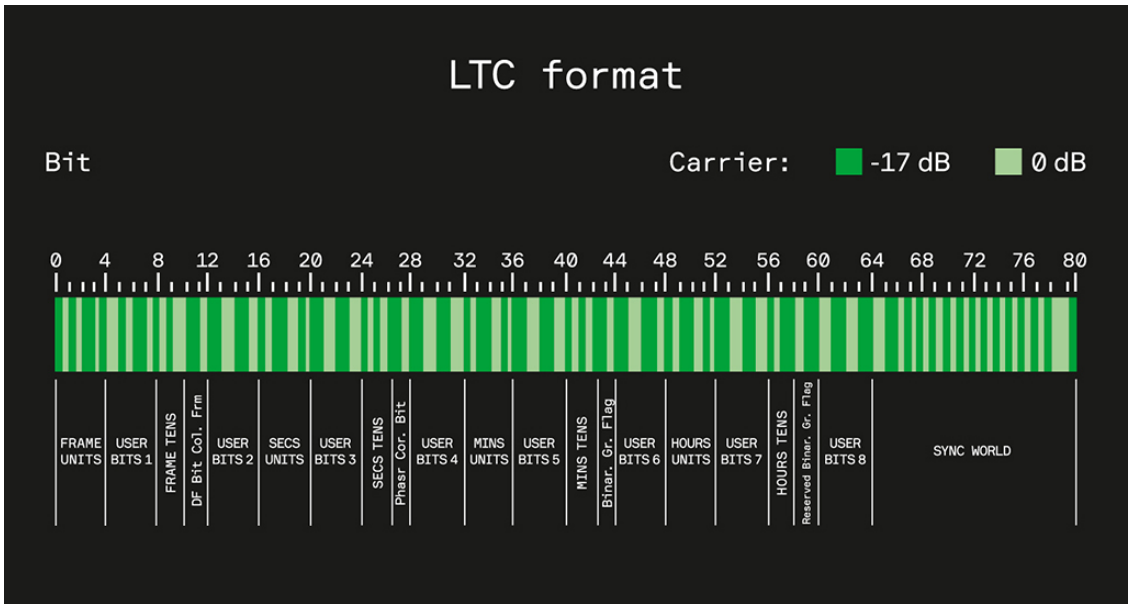
называется Longitudinal Time Code (LTC), или линейный таймкод. Такое название было дано потому, что информация в этом формате записывается бит за битом последовательно, так как по-другому на ленту информацию не записать. Спустя многие годы эта технология записи таймкода начала использоваться в шоу-индустрии.

SMPTE – это время, которое начинается с нуля. Так же, как и в обычной жизни, в SMPTE есть часы, минуты и секунды. Максимальное значение SMPTE – 24 часа, как и в сутках. Но есть и отличие от привычного для нас измерения времени – это кадры. Терминология пришла из кино. Так как одна секунда видео содержит в себе определенное количество кадров, то и временной код определяется видеокадрами. Но количество кадров может быть различным в зависимости от формата видео. Это применимо также для SMPTE, поэтому он может быть разным.

Есть несколько форматов SMPTE: 24fps (frames per second – кадров в секунду), 25fps, 30fps, 29.97fps (он же 30fps drop). Все эти значения, кроме 24 fps, были обусловлены форматом телевещания в конкретной стране – PAL, SECAM, NTSC – которые, в свою очередь, были обусловлены частотой электросети (50 Гц, 60 Гц) и стандартами передачи цветного изображения. 24 fps – это киноформат, так как кино снимается на пленку, где одна секунда содержит в себе 24 кадра.

В самом начале мы с вами договаривались, что в этой книге будет минимум лирических отступлений, но одно все же сделать нужно, так как эта информация нам нужна для понимания различия между линейным и другими форматами таймкода

Простейшей единицей информации в LTC является блок данных, передающий каждый кадр реального времени, поэтому он так и называется – кадр SMPTE. Для кодирования битов в LTC-сигнале используется схема под названием Bi-Phase Mark: нули кодируются одиночным переворотом фазы на границе периода, единицы – двумя переворотами (один на границе периода, другой в половине периода). LTC-кадр имеет длину 80 бит. Структура кадра показана на рисунке.



Время SMPTE кодируется методом BCD (Binary Coded Decimal). В этом методе под каждую десятичную цифру отводятся четыре бита. В кадре на запись времени отводятся 26 бит. Между ними присутствуют дополнительные данные, которые по большей части относятся к параметрам видеокадра, а также данные User Bits, которые в свободной форме могут использоваться для передачи дополнительной пользовательской информации. На данный момент такой

способ передачи дополнительных данных не используется, и информация в этих областях всегда равняется нулю. Завершается блок данных синхрословом (последние 16 бит). Синхрослово используется для определения границы кадра, значение которого фиксировано: 0011 1111 1111 1101.

Если таймкод записан на ленту правильно, то при воспроизведении никаких проблем не возникает, так как информация записана физически. В этом и есть один из секретов надежности LTC.

MTC (MIDI Time Code)

Время прошло, и на телевидении SMPTE стали писать уже в других форматах: VITC, CTL, BITC, Keycode.

Поговорим теперь о другом варианте работы с SMPTE. Следующий интерфейс пришел из мира музыки. Это MIDI. Он очень обширен и позволяет работать с разными форматами данных. Более подробно о MIDI и его структуре мы поговорим в следующей главе, а сейчас затронем формат работы со временем. Называется он MTC (MIDI Time Code).

Первое отличие его от LTC в том, что этот формат полностью цифровой и кодируется в шестнадцатеричной системе исчисления (в то время как LTC кодируется бинарным способом).

Второе: этот формат нельзя записать на носитель (как LTC), он генерируется программно или аппаратно.

Теперь давайте подробнее затронем цифровую часть этого протокола. Существует два типа MTC сообщений: Full-message (полное сообщение) и Quarter-frame (четверть сообщения).

Full-message содержит в себе информацию целого таймкод кадра. Такое сообщение отправляется в определенных случаях: остановка, начало воспроизведения, перемотка и др.

Quarter-frame сообщения отправляются в случае штатного воспроизведения. Как следует из названия, в момент передачи одного таймкод кадра используются четыре Quarter-frame сообщения. Но вот парадокс, чтобы закодировать информацию о полном SMPTE кадре через MTC, необходимо использовать восемь Quarter-frame сообщений. Из этого следует, что, чтобы передать через MIDI информацию о полном кадре таймкода, необходимо потратить время, равное воспроизведению двух кадров SMPTE таймкода. Это означает, что к моменту, когда принимающее устройство декодирует информацию о полном кадре, она уже устареет на два кадра. А также из-за этой особенности принимающее устройство будет получать по MIDI только каждый второй кадр. Чтобы по итогу принимающее устройство смогло корректно интерпретировать принятый таймкод, оно должно сделать поправку на два кадра и также самостоятельно генерировать недостающие кадры. К сожалению, не все производители оборудования реализуют подобные алгоритмы, по этой причине MTC на разных устройствах может работать по-разному.

LTC и MTC – это основные форматы при работе с синхронизацией по времени. Чуть позже мы с вами еще вернемся к вопросу таймкода и разберем несколько альтернативных интерфейсов для кодирования и передачи времени.

EVENT SYNC PROTOCOLS

Поговорим теперь о протоколах, которые передают не время, а сообщения. Они обеспечивают синхронизацию по определенным событиям, а не по времени (в этом случае постоянная синхронизация отсутствует). В начале 70-х годов, в эру зарождения компьютерных технологий, разные компании экспериментировали и придумывали все новые и новые стандарты передачи данных, увеличивая скорость, надежность и дальность их передачи. К этому времени шоу-индустрия нуждалась в более продвинутом и функциональном протоколе синхронизации, так как LTC было уже недостаточно. Настало время появления на свет новых интерфейсов, которые сделали определенный прорыв в шоу-индустрии.

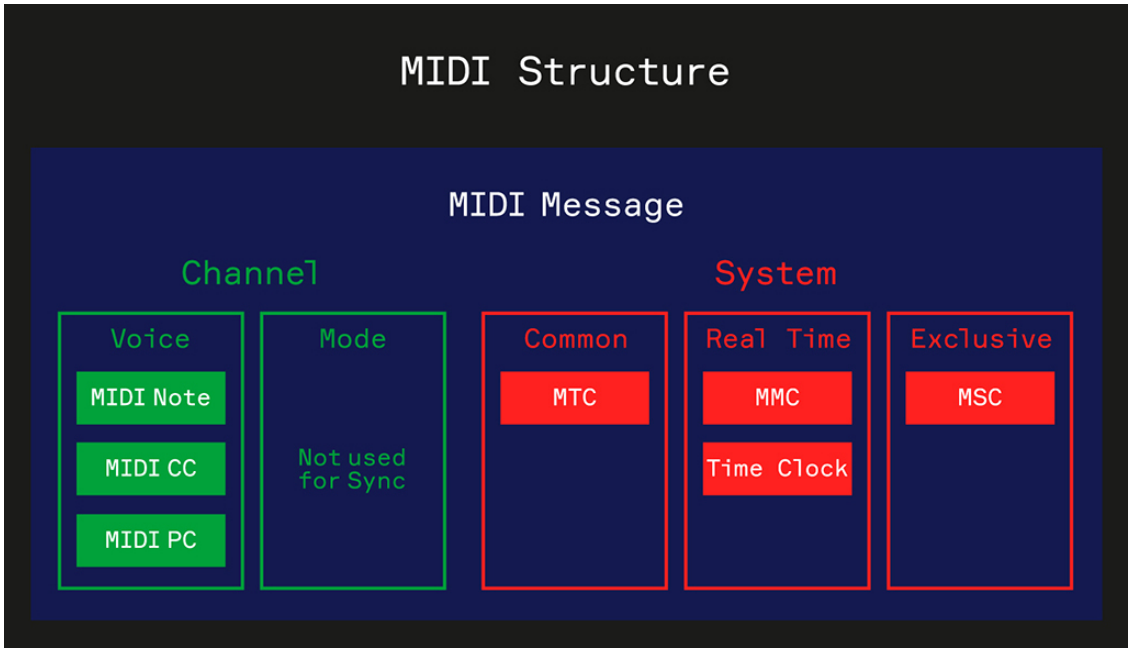
MIDI note / MIDI CC / MIDI PC

Как я уже говорил, новые разработки в мире синхронизации зачастую приходили из музыкальной индустрии. В конце 70-х годов особенное распространение получили музыкальные синтезаторы. Это были электрические музыкальные устройства, которые модулировали определенные звуки, управляемые напряжением.

Для каждой тональности звука существовал свой генератор. Модель каждого синтезатора характеризовалась эффектом или звуком, который она могла модулировать. В те времена рабочее место музыканта состояло из множества различных синтезаторов и управлять всем этим было очень непросто. Потребности росли – и в начале 80-х годов активно развивающиеся технологии в мире электроники подсказали решение этой проблемы. Был сделан шаг в сторону цифрового программного управления. Компании, производившие синтезаторы, смогли удачно договориться о разработке и поддержке единого стандарта интерфейса управления синтезаторами, первая спецификация которого появилась в 1982 году. Смысл заключался в разделении звукообразующего модуля и модуля управления, которые соединялись между собой по цифровому каналу. Протокол, разработанный для этих целей, передавал состояния нажатия клавиши.

Позже его функционал расширили большими возможностями. Благодаря такой системе музыкант мог с одной клавиатуры управлять сразу несколькими синтезаторами. Появилась возможность записывать и воспроизводить сыгранную мелодию. Сложно переоценить влияние нового протокола на музыкальную и шоу-индустрию в целом.

Дамы и господа, позвольте мне представить вам интерфейс синхронизации MIDI Musical Instrument Digital Interface. Несмотря на свой приличный возраст, MIDI до сих пор широко используется в разных направлениях шоу-индустрии.



Язык MIDI состоит только из команд управления и параметров этих команд. Команды в языке MIDI называются сообщениями. Сообщения разделяются на два основных типа: одни управляют звукообразованием, то есть говорят, какую ноту и как громко играть, вторые выполняют служебные функции, то есть регулируют изменения настроек тона генератора и синхронизации.

Сообщения первого типа называются сообщениями канала (Channel Messages).

Сообщения второго типа называются системными (System Messages).

Сообщения канала, в свою очередь, делятся на голосовые (Channel Voice Messages) и сообщения режима канала (Channel Mode Messages).

Системные сообщения делятся на общесистемные (System Common Messages), сообщения реального времени (System Real Time Messages) и эксклюзивные сообщения (System Exclusive Messages).

На изображении выше вы можете увидеть, какой MIDI-протокол к какому типу относится.

При работе непосредственно с MIDI-протоколом мы неоднократно встретим такой параметр, как MIDI Channel – программный канал передачи сообщений. Всего может быть 16 каналов. Этот параметр применим только к сообщениям группы Channel, соответственно, каждое сообщение MIDI note, MIDI CC и MIDI PC несет информацию о том, к какому каналу оно принадлежит. Каналы нужны для разделения в одном потоке нот разных инструментов. Это позволит нам отправить одновременно две одинаковые MIDI-ноты, но для генерации разных музыкальных инструментов.

MIDI note

В сообщении MIDI note содержатся три байта, в которых закодирована информация о номере канала, ноты и ее состояния. Нота может быть активна (Note on) или неактивна (Note off). Каждая нота имеет свой определенный номер – от 0 до 127. Каждый номер соответствует определенной ноте в определенной октаве. Для синхронизации устройств эти ноты могут пере-

давать разные события, с которыми нужно синхронизироваться. Только в отличие от музыкального устройства, принимающее оборудование не генерирует звук, а выполняет определенное действие.

К примеру, можно настроить на световом пульте и на мультимедиа-сервере следующее: при получении ноты до первой октавы первого канала необходимо запустить световую программу и видеоконтент. При получении по MIDI этой ноты мультимедиа-сервер и световой пульт запустят свои программы синхронно. Особенность такого способа в том, что при большом количестве разных команд MIDI note можно запутаться в обозначениях. К тому же при изменении шоу, перемещении CueList внутри пульта нужно следить за ссылками MIDI-команд: будут ли они выполнять то, что нам нужно.

MIDI CC / MIDI PC

Сообщения MIDI CC (Control Change) и MIDI PC (Program Change) очень похожи на сообщения MIDI note. Сообщения этого типа кодируются тремя байтами, где также содержится информация о канале, номере параметра и его состоянии. Используются они для изменения программы с набором инструментов воспроизведения, а также других настроек синтезатора.

Чаще всего при помощи сообщений MIDI CC и MIDI PC производят синхронизацию линейных параметров. К примеру, уровень фейдера или регулятора на MIDI-контроллере. Используя эти сообщения, вы можете передать состояния 128 параметров управления, где каждый канал имеет диапазон от 0 до 127.

MSC (MIDI Show Control)

В определенный момент простого звукового MIDI протокола стало недостаточно для театрально-концертных нужд, и встал вопрос о создании более специализированного протокола управления.

К концу 1989 года Charlie Richmond организовал рабочую группу в составе MMA (MIDI Manufactures Association) и создал общий форум на электронной доске USITT (The United States Institute for Theatre Technology). Созданный в результате проект стандарта был утвержден MMA и JMSC (Japanese MIDI Standard Committee) и 25 июля 1991 года превратился в «Рекомендованную практику RP-002», или иначе – в MIDI Show Control версии 1.0.

Сообщения MIDI Show Control относятся к категории универсальных эксклюзивных сообщений реального времени (Universal Real Time System Exclusive).

В спецификации MSC используется терминология Controller и Controlled Device. Устройство, которое генерирует MSC сообщения, называется Controller, обычно это компьютер со специализированным софтом и MIDI картой. Принимающие устройства, световые пульты, медиа серверы и остальные исполняющие контроллеры— это Controlled Device.

Особенность MSC в том, что сообщения этого формата имеют определенные категории. Они делятся на основные (General Categories) и дополнительные подкатегории. Также есть особая категория All-types, сообщения этого типа транслируются на все типы контроллеров. К общим категориям относятся свет, звук, машинерия, видео, проекция, спецэффекты и пиротехника.

Ко всему этому в MSC сообщении передается информация об устройстве, которое должно получить команду. Каждое устройство внутри одной категории имеет свой собственный ID. Выставляется он вручную в настройках принимающего контроллера.

Внутри одной категории ID всех устройств должны быть разные. Уникальный ID каждого устройства может быть в диапазоне 1–111.

Что нам дает использование ID? Теперь можно каждому принимающему контроллеру отправлять команды, предназначенные конкретно для него, а все остальные контроллеры, которые имеют другую категорию оборудования и ID, эти сообщения будут просто игнорировать.

Также в идеологии MSC есть такое понятие, как Group ID и Broadcast ID.

Каждое устройство может иметь уникальное ID, но при этом быть подписанным на групповые сообщения. Для этого в MSC зарезервированы специальные групповые ID с 112 до 126.

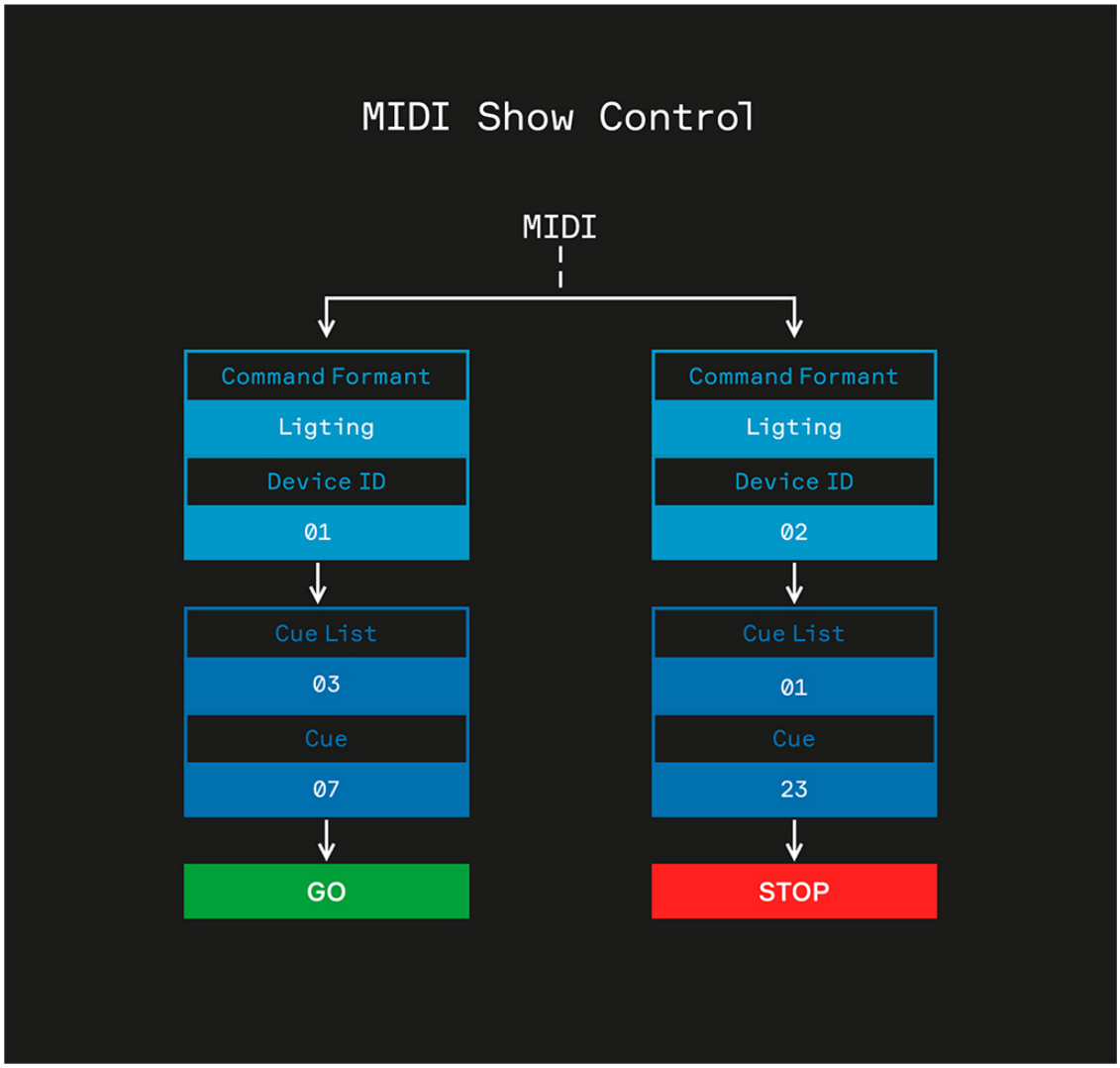
Всего, может быть, 15 независимых групп. Чтобы конкретную команду получили клиенты, подписанные на группу, серверу нужно просто отправить MSC сообщение на один из групповых ID.

Также есть один зарезервированный ID, равный 127 для широковещательных команд. MSC команду, отправленную на этот ID, получат все клиенты.

General Commands
Hex command_format

00 reserved for extensions	06 SET
01 GO	07 FIRE
02 STOP	08 ALL_OFF
03 RESUME	09 RESTORE
04 TIMED_GO	0A RESET
05 LOAD	0B GO_OFF

Дополнительное преимущество MSC перед MIDI note в том, что сообщения этого протокола содержат конкретные команды для действия, а не просто код сообщения, к которому нужно привязывать действие на конечном контроллере. Есть основная группа MSC сообщений – General Commands, которые применимы для всех типов устройств. Помимо General Commands, есть дополнительная группа Sound Commands специально для звуковых систем синхронизации. Для общей синхронизации нам интересны только команды типа General Commands.



Итак, с командами разобрались, куда отправлять команды, тоже

понятно, теперь поговорим о том, к чему эти команды применимы. В MSC сообщениях есть часть кода, в которой зашифрована информация об объекте в контроллере, к которому нужно применить команду.

<Q_number> <Q_list> <Q_path>

Этот код состоит из номера сцены, к которой применять команду (CueNumber), в каком списке находится эта сцена (CueList) и путь туда, где находится список сцен (CuePath). Последнее значение осталось еще с тех времен, когда световые пульты сохраняли свои шоу на флоппи дискеты и пульт мог воспроизводить световые сцены как с внутренней памяти, так и с внешнего носителя. QuePath как раз и указывал, где находится финальная цель сообщения.

Не все сообщения из всех команд MSC несут в себе объект, к которому нужно применить команду, в основном это глобальные команды, как, например, ALL OFF или RESET.

Резюмируя все особенности MSC, можно сказать следующее:

• В MSC сообщении содержится информация о том, к какому типу устройства и к какому по номеру устройству предназначается сообщение: Command Format, Device ID.

• MSC сообщение имеет команду, которую нужно выполнить на принимающем устройстве – Command.

• MSC сообщении есть информация, к какому объекту в принимающем устройстве применить команду – Q Number, Q List, Q Path.

Эти возможности во много раз расширяют функционал MSC перед простыми звуковыми MIDI сообщениями. Это позволяет быстро запрограммировать команды на главном контроллере для синхронизации шоу. При этом нет необходимости в сложных настройках на световых консолях и других принимающих устройствах. Всю необходимую информацию они получают по протоколу управления.

Теперь снова вернемся к истории. MSC v1.0 стал настолько популярным, что позже была разработана новая усовершенствованная версия MSC v1.1. Основное его отличие заключалось в том, что эта версия MSC теперь подразумевало обратную связь между сервером и клиентом. Применение новой версии протокола было нацелено на опасные системы, такие, как, к примеру, механика сцены

В новой идеологии протокола клиент перед тем, как получить данные с сервера, должен отправить ответ серверу о возможности выполнения действия, и после получения команды он также должен сообщить серверу, удалось ли выполнить команду или нет. Такая система подтверждения команд рассчитана на то, чтобы повысить безопасность управления сложными системами. Но кроме явных преимуществ обратной связи с сервером был один недостаток, заключался он в скорости синхронизации. Сервер не мог отправить новую команду до тех пор, пока не был получен ответ о выполнении предыдущей команды. К сожалению, усложненная версия MSC v1.1 так и не получила широкого применения в шоу-индустрии. А вот MSC v1.0 был долгое время очень популярен, пока на его смену не пришли новые протоколы, основанные на современных технологиях передачи данных.

MCC (MIDI Machine Control)

Использование MMC для синхронизации в шоу-индустрии применяется довольно редко. Но мы тем не менее еще можем встретить оборудование, которое может генерировать и принимать MMC сообщения. Этот протокол был разработан для того, чтобы дистанционно управлять профессиональными многодорожечными видео и аудио рекордерами на студиях. Основное назначение MMC команд – передача сообщений разных режимов воспроизведений и записи.

Команды MMC находятся в группе SysEx сообщений. MMC сообщения так же, как и MIDI Show Control, несут в себе информацию о номере устройства Device ID и команду. Все команды, которые можно передать в MMC сообщении, находятся ниже в таблице.

Command ID	Command
01	Stop
02	Play
03	Deferred Play (play after no longer busy)
04	Fast Forward
05	Rewind
06	Record Strobe (AKA [[Punch in/out|Punch In]])
07	Record Exit (AKA [[Punch out (music)|Punch out]])
08	Record Pause
09	Pause (pause playback)
0A	Eject (disengage media container from MMC device)
0B	Chase
0D	MMC Reset (to default/startup state)
40	Write (AKA Record Ready, AKA Arm Tracks)parameters: <length1> 4F <length2> <track-bitmap-bytes>
44	Goto (AKA Locate)parameters: <length>=06 01 <hours> <minutes> <seconds> <frames> <subframes>
47	Shuttle parameters: <length>=03 <sh> <sm> <sl> (MIDI Standard Speed codes)

Как мы видим, некоторые команды могут нести дополнительную информацию. Как, например, команда Goto, которая передает также время, на которое необходимо перейти устройству. Мы не будем подробно разбирать этот протокол, так как он уже устарел и для синхронизации практически не используется. Но мы все равно его затронули, потому что некоторые профессиональные таймкод генераторы поддерживают этот тип сообщений для настройки и старта генерации времени.

OSC (Open Sound Control)

Время идет и технологии не стоят на месте. Протоколу MIDI уже почти сорок лет! И конечно же, ему на замену пришла более современная альтернатива Open Sound Control. Как понятно по названию, и этот интерфейс пришел из мира музыки. Но как и следовало ожидать, другие направления шоу-индустрии быстро его освоили. Сейчас этот протокол настолько популярен в разных направлениях шоу технологий, что позже сами создатели дали ему второе название: Open System Control. Что же в нем особенного? Давайте разберемся.

Этот протокол был разработан в компании CNMAT (Center for New Music and Audio Technologies) программистами Adrian Freed и Matt Wright еще в 1997 году. У них появилась идея создать новый протокол на базе широко используемого интерфейса Ethernet. Главное преимущество его в том, что для работы с этим протоколом не нужно никакого специального оборудования и коммутации, как в случае с другими интерфейсами и протоколами синхронизации. Также, Ethernet – это отдельное направление, которое развивается и поддерживается разными компаниями и институтами. Сейчас технологии позволяют передавать сигнал по Ethernet, по воздуху и оптическому каналу на многие километры. А скорость передачи по сравнению с MIDI в тысячи раз быстрее, при этом соединение надежнее. Все эти преимущества автоматически унаследует протокол OSC. Но кроме ряда преимуществ есть и недостатки. Он стал сложнее, а как следствие, специалисту необходимо иметь больше знаний об этом протоколе для работы с ним. Как раз цель этой главы – дать вам знания для комфортной и уверенной работы с протоколом синхронизации OSC.

Концепт построения сетей

Так как OSC базируется на сетевом интерфейсе Ethernet, то прежде чем работать с этим проколом, необходимо настроить сеть. Я уверен, что вы и так прекрасно ориентируетесь в этом вопросе, и то, что вы прочитаете ниже, для вас будет не ново, тем не менее я рекомендую освежить свои знания в этом направлении.

Первое, с чем необходимо познакомиться, это с группой протоколов TCP/IP (Transmission Control Protocol/Internet Protocol). Концепт работы со всеми протоколами описан уровнями (слоями), через которые должен пройти каждый пакет данных при отправке или получении.

TCP/IP Уровни (Layers)

Уровень 1. Физический уровень

Уровень 1 является физическим уровнем. Это оборудование интерфейсов, включая электронику и провода.

Уровень 2. Дата линк уровень

Уровень 2 тесно связан с оборудованием. Этот уровень занимается тем, что собирает данные в пакеты для отправки через Физический уровень. Также этот уровень отвечает за корректную передачу и прием данных и исправление ошибок. На этом уровне появляется такое понятие, как MAC адрес (Media Access Address[2]).

[2] Если коротко, то каждое сетевое оборудование имеет уникальный MAC адрес, предустановленный на заводе. MAC адрес для каждого устройства является уникальным. При помощи MAC адреса каждое устройство может идентифицировать себя в общей сети для передачи или приема данных. Пример MAC адреса – 98:5A:EB:C9:00:A8.

Уровень 3. Сетевой (Internet) уровень

Это Сетевой уровень, отвечающий за логическое направление данных через сеть.

В большой сети может быть множество логических маршрутов связи между устройствами, по которым пакеты данных могут передаваться. Этот уровень занимается анализом доступных возможностей в сети и выбором подходящего маршрута.

Уровень 4. Транспортный уровень

Этот уровень ответственен за обеспечение того, чтобы низкоуровневые пакеты данных были собраны в их оригинальной форме, правильном порядке и без ошибок.

Сетевой уровень может разбить пакет данных на несколько частей и отправить их разными маршрутами. Поэтому части одного пакета могут быть доставлены не по порядку. Транспортный уровень как раз и решает такие проблемы. Осуществляется это при помощи функционала двух протоколов TCP и UDP.

Уровень 5. Уровень приложения

Этот уровень отвечает уже за передачу «полезных» данных через сеть, с которыми непосредственно имеют дело приложения и программное обеспечение.

Итак, зачем же нам знать все эти уровни и их функционал? Дело в том, что если мы хотим передать что-то через сеть, то прежде нам необходимо выполнить определенные условия, которые регламентирует группа протоколов TCP/IP.

Одно из таких условий – это обязательное использование для каждого клиента сети IP адреса (Internet Protocol Address). Напрашивается резонный вопрос, зачем же использовать дополнительный параметр адреса, если каждое сетевое устройство уже имеет уникальный адрес MAC? Не будем углубляться во все особенности, скажу лишь только, что благодаря IP адресам в нашей сети появляется возможность ее организации, что очень важно, когда мы создаем сложную сеть с определенными правилами. К тому же в отличие от MAC адреса, одно устройство может иметь несколько адресов IP. Какие бывают IP адреса и как их настроить, мы разберем позже.

Кстати, существуют такие типы протоколов, которые используют для подключения только MAC адрес, к примеру, транспортный протокол NetBEUI. Это довольно быстрый протокол, но его главным недостатком является то, что он полностью не маршрутизируемый.

Следующее условие. Для передачи данных мы должны использовать UDP или TCP протокол. По сути, и первый, и второй тип протокола занимается тем, что обеспечивает корректную передачу данных. Но в чем же их отличие?

Принцип работы UDP основан на том, что при отправке данных сервер не беспокоится о том, получил ли клиент эти данные или нет. Он просто отправляет пакет за пакетом в сеть. Отличительная особенность UDP протокола в том, что он очень быстрый.

При использовании протокола TCP клиент сначала должен подключиться к серверу, создав виртуальное подключение. После этого сервер может передать данные клиенту. После получения данных клиентом он отвечает серверу, что пакет данных принят корректно, в противном случае сервер отправляет пакет данных повторно. После завершения сеанса связи клиент должен обязательно корректно отключиться от сервера. Эта схема занимает больше времени на передачу данных, но при этом она более надежна, так как сервер получает обратную связь от клиента.

Но как бы ни был надежен TCP протокол, UDP намного шире используется в шоу-индустрии. Самый яркий пример – это ArtNet протокол, который базируется на UDP. Световой консоли нет необходимости беспокоиться о том, получил ли необходимый пакет данных один из десятка артгейтов. Консоли более важно отправлять пакеты данных в реальном времени для всех участников. И если по какой-то причине один пакет был доставлен некорректно, то это будет менее заметно, чем если все артгейты будут ждать, пока один из клиентов получит корректный пакет. И последнее условие – это использование TCP или UDP портов для передачи или приема данных. Так как потоков данных для одного клиента может быть большое количество, то для того, чтобы эти потоки не перемешивались и были независимы, каждый поток использует свой входящий или исходящий порт, через которые идет обмен данными. Номер порта в UDP и TCP идентифицируется одним числом. Оно может быть в диапазоне от 1 до 65535. Приложение, которое хочет получить или отправить данные, обращается к необходимому TCP или UDP порту. Некоторые порты зарезервированы системой для служебного пользования, и программы пользователя не могут использовать эти порты. К примеру, протокол ArtNet использует порт 6454.

Теперь давайте немного пофантазируем и представим работу курьерской службы и вокзала. Это поможет нам более наглядно понять принцип работы TCP и UDP.

Вокзал имеет свой уникальный адрес местонахождения: город, улица, номер дома – как IP адрес компьютера. На этот вокзал постоянно приходят поезда, как пакеты данных, которые передаются по Ethernet. В поездах приезжают курьеры, работающие на конкретную компанию,

которая в свою очередь имеет жесткий протокол отправления и доставки посылок, как протоколы TCP/IP.

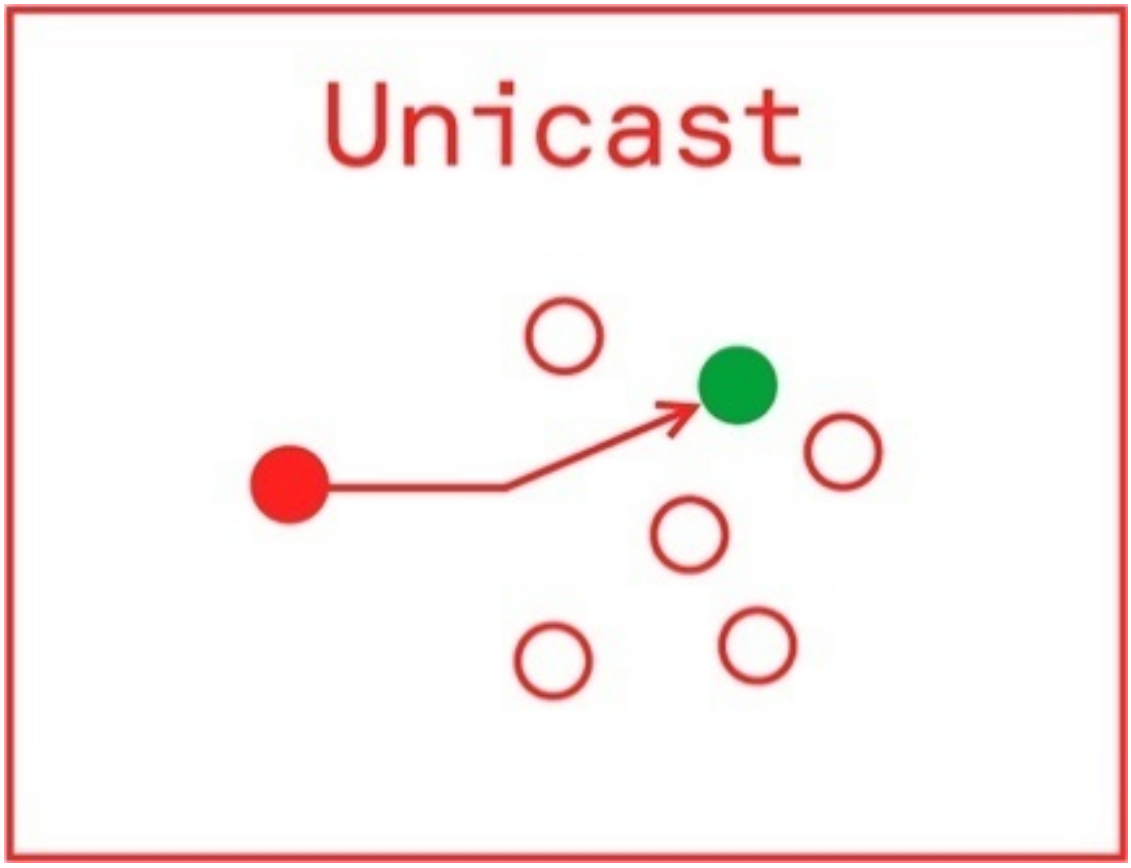
В этом протоколе прописано, что каждый посыльный должен доставлять посылку в отделение своей компании и в конкретное окно приема посылок, каждое окно имеет свой номер, так же, как номер порта UDP или TCP. По итогу посыльный доставляет посылку в конкретное отделение и в конкретное окно с конкретным номером. В отделении этой компании много окон, так как заказчиков, которые хотят получить свои посылки, тоже немало. Когда заказчик хочет получить на руки свою посылку, он приходит в отделение компании и обращается в окно с конкретным номером, к которому он прикреплен. Также поступают и программы на компьютере, когда они хотят прочитать данные с конкретного UDP или TCP порта. Если в окне что-то есть для заказчика, то служащий выдает посылку.

Теперь, когда данные получены программой, наступает следующий этап. Расшифровка данных. Когда заказчик открывает посылку, он ожидает увидеть там, скажем, письмо, которое состоит из букв и знаков препинания, с помощью которых закодировано сообщение. Чтобы его прочитать, заказчик должен знать язык, на котором это письмо написано. Языки кодирования регламентируют протоколы высокого уровня, как, например, ArtNet, sACN, OSC, RTP-MIDI. Наш заказчик – англичанин, и если письмо написано на английском языке, то он сможет прочитать сообщение, написанное в письме. То же самое и с программным обеспечением: если программа понимает язык ArtNet, то, получив пакет данных, закодированных протоколом ArtNet, она сможет прочитать и извлечь данные о DMX уровнях.

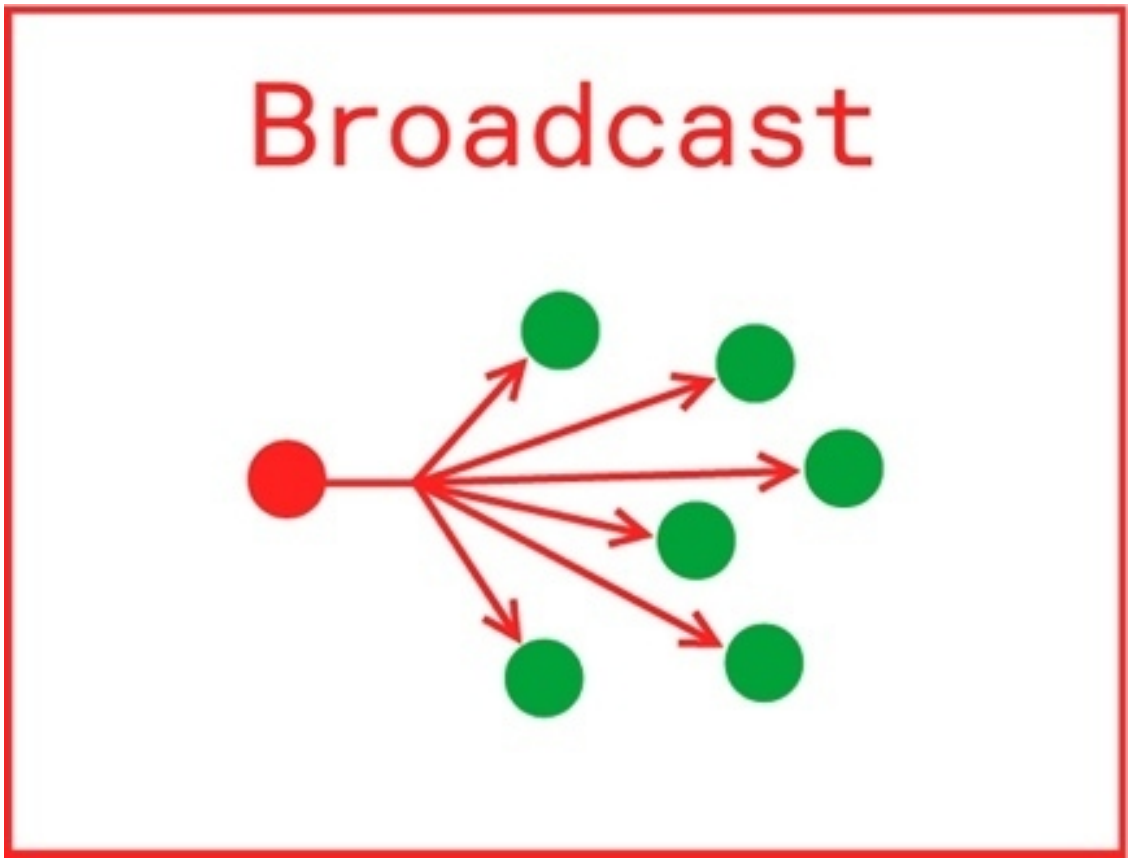
Теперь давайте познакомимся с методами передачи данных, которые основываются на IP адресах.

Существует три распространенных метода, которые имеют место быть в шоу-индустрии: Unicast, Multicast и Broadcast.

Сразу уточню, методов больше и их принципы отличны. Но так как это уже более специализированное направление, которое не имеет точек соприкосновения с нашей индустрии, то информацию о них вы можете найти в более специализированной литературе.



Самый простой метод передачи. В этом методе пакеты данных передаются уникально, т.е. от клиента сети к клиенту. Такой пакет данных может быть передан только одному устройству. Сетевые свитчи (принцип, который мы разберем чуть позже), получая такой пакет данных, переправят его на физический порт конкретного устройства, которое должно получить эти данные.



Широковещательный метод передачи данных, где один пакет данных передается всем участникам сети (или подсети), в независимости от их адреса. По этому принципу очень часто работает протокол ArtNet. Для этого метода в каждом уровне диапазона сетей зарезервирован адрес 255.

Какие бывают подсети и как они ограничиваются, мы разберем чуть позже, когда будем знакомиться с таким понятием, как маска подсети, но уже сейчас я хочу привести пример широковещательных адресов для сети 10.0.0.0, для понимания идеи диапазона широковещания. Ниже я постараюсь избегать терминологии, с которой мы еще не познакомились.

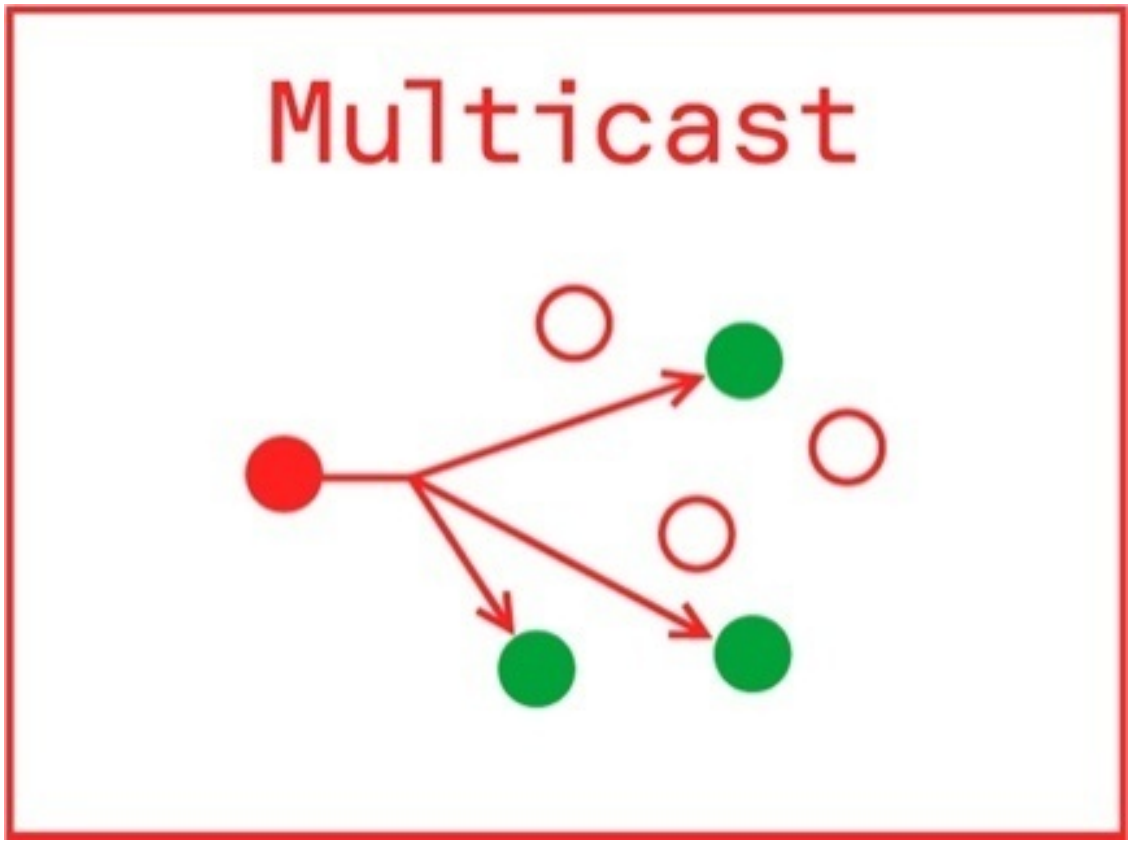
Итак, если отправить пакет данных на адрес 10.0.0.255, его получат все устройства, чей адрес начинается на 10.0.0.x.

Если отправить пакет данных на адрес 10.0.255.255, его получат все устройства, чей адрес начинается на 10.0.x.x.

Если отправить пакет данных на адрес 10.255.255.255, его получат все устройства, чей адрес начинается на 10.x.x.x.

И последний, самый широковещательный адрес – это 255.255.255.255, пакет данных, отправленный на этот адрес, получат все устройства в сети.

Зачастую в консолях управления уже предустановлено системой, насколько широко может открываться вещание того или иного протокола. В частности, хорошо знакомого нам ArtNet.



Технология, которая реже используется в шоу-индустрии и обычно скрыта в управлении от пользователя.

Особенность этого протокола заключается в том, что эта технология подразумевает передачу пакетов данных только тем клиентам, которые сами подписались на получение этих данных.

Для этой технологии передачи данных зарезервирован диапазон адресов 224.0.0.0 – 239.255.255.255.

Один адрес этого диапазона определяет мультикаст-группу. Каждое устройство, которое хочет получать пакеты данных, относящиеся к этой группе, подписывается на эту мультикаст-группу, декларируя в сеть эту информацию. Сетевые свитчи и роутеры регистрируют эти запросы, и когда один из клиентов отправляет данные на мультикаст адрес, свитчи и роутеры рассылают эти данные только тем клиентам, которые подписались на эту группу.

Unicast и Multicast технологии являются более совершенными по сравнению с Broadcast, так как они позволяют разгрузить сеть, не создавая лишних маршрутов, и при этом клиенты, подписываясь или отписываясь от мультикаст-групп вещания, сами решают, получать данные или нет.

Сетевое оборудование

Чтобы построить сеть, нам необходимо иметь то, что должно участвовать в сети. Это могут быть контроллеры, компьютеры, смартфоны, планшеты, консоли управления и многое, многое другое. Давайте разберем простейший пример, в котором у нас есть четыре компьютера.

Итак, чтобы каждый компьютер мог стать участником сети, он должен иметь сетевую карту. Зачастую все компьютеры уже имеют встроенные сетевые карты, за исключением некоторых компактных ноутбуков, где сетевой разъем убирают для экономии места. К примеру,

Mac Book новых моделей толщиной меньше, чем сам разъем Ethernet. Для таких компьютеров есть специальные переходники с Thunderbolt и USB на Ethernet.

Большинство сетевого оборудования использует физический интерфейс стандарта RJ-45. Кстати, существует путаница с названием сетевых разъемов и коннекторов. RJ-45 – это имя стандарта, который описывает конструкцию обеих частей разъема (вилки и розетки) и схемы их коммутации. В то время как сам сетевой разъем имеет имя 8P8C (8 Position 8 Contact).

Сетевые карты бывают разные. Основная характеристика, которая нас должна интересовать, – это скорость передачи и приема данных. Она измеряется в максимальном количестве данных в битах, передаваемых за одну секунду. На данный момент существуют такие скорости, как 10Mbps (Megabit per second), 100Mbps, 1Gbps (Gigabit per second), 2,5Gbps, 5Gbps, 10Gbps, 40Gbps, 100Gbps и 160Gbps. Самые распространенные и часто используемые скорости в оборудовании и сетевых картах – это 100Mbps и 1Gbps. Скорости выше одного гигабита в секунду, это уже более профессиональные стандарты для передачи огромных объемов данных. Для этих скоростей нужны особые сетевые карты и сетевые кабели подходящего стандарта.

С сетевыми картами определились, теперь как подключить в сеть сразу три и более компьютеров? Если бы была необходимость подключить в сеть только два устройства, то нам бы понадобился один сетевой кабель, который бы просто подключили между двумя сетевыми картами. Но давайте подключим четыре компьютера, которые должны стать участниками одной сети. Чтобы соединить их, необходим коммутатор. Существуют разные модели таких устройств, которые отличаются скоростью портов, их количеством и другими особенностями, но я бы хотел выделить три основные типа таких коммутаторов и определить их различия.

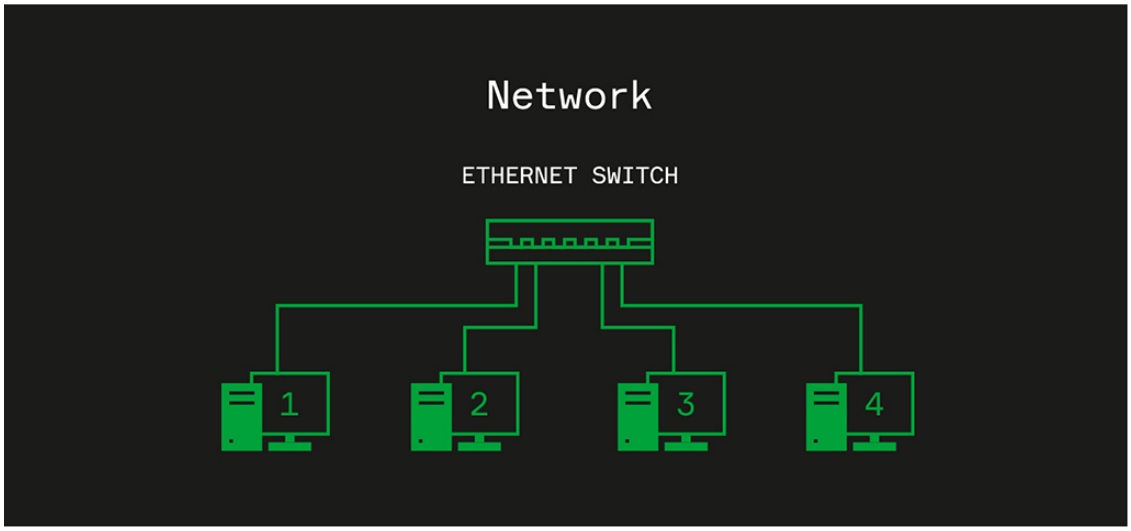


Схема подключения клиентов

Первый тип – *Ethernet Hub*. Хаб представляет из себя простейшее сетевой устройство, которое дублирует все свои порты между собой. Это своего рода сетевой сплиттер. Получив пакет данных с одного порта, хаб дублирует этот пакет на все остальные. Вот и все. Но это плохо, потому что, даже если пакеты данных предназначаются только для одного клиента, то хаб отправит эти пакеты всем клиентам. Это загружает порты и самое главное сеть! Хорошая новость, что простые хабы практически уже не встречаются, так как они попросту, морально устарели.

Следующий тип коммутаторов – это *Ethernet Switch*. Свитч внешне никак не отличим от хаба, но это уже более умное устройство. Оно позволяет грамотно организовать нашу сеть.

При включении свитч опрашивает все свои порты и знает, на каком порте какое устройство и с каким адресом находиться. Что это нам дает? Когда на определенный порт свитча приходит пакет, который предназначается для конкретного адреса, свитч отправляет этот пакет на конкретный порт, где находится адресат. При такой схеме сеть работает намного рациональнее и быстрее.

Но также не могу не затронуть еще одну очень важную особенность свитчей. Существует два типа свитчей: Managed и Unmanaged. Unmanaged более распространены на рынке. Это значит, что такие свитчи не настраиваемы, в них уже предустановлены основные настройки, которые позволяют работать сразу с сетью. Второй тип – Managed, чтобы запустить в работу такой свитч, сначала его необходимо настроить. Такие свитчи работают на более высоком уровне. Они понимают, с какими протоколами работают, могут фильтровать данные или блокировать. Такие свитчи имеют довольно широкий функционал для конкретных задач. Но для работы с такими свитчами нужно иметь более высокий уровень знаний построения сетей и уметь программировать оборудование конкретного производителя. Если у вас таких знаний нет, то я советую использовать неуправляемые свитчи, так как для того чтобы запустить его в работу, вам нужно только подключить его к питанию!

И последний тип коммутаторов – *Router*. Роутер по своему функционалу похож на настраиваемый свитч, его отличие в том, что он помогает настроить правила общения между двумя и более физическими сетями. Самое распространенное использование роутеров – организация сети между глобальной сетью интернет (WAN[3]) и локальной сетью (LAN[4]). Роутер может подключиться к сети интернет по определенным правилам, но при этом создать независимую сеть для локальных пользователей, и также передавать пакеты данных между этими сетями. Также зачастую роутеры имеют службу DHCP (Dynamic Host Configuration Protocol), которая позволяет автоматически раздавать IP адреса клиентам без участия пользователя. О ней мы поговорим чуть позже. Также большинство бытовых роутеров имеют встроенную беспроводную точку доступа Wi-Fi.

Резюмируя особенности всех типов коммутаторов, хочу сказать, что самый надежный тип коммутаторов – не настраиваемые свитчи, так как вы точно можете быть уверенными, что ваши данные дойдут до клиента, и никакие правила и настройки роутеров или управляемых свитчей не смогут заблокировать ваши пакеты данных.

Сетевые настройки

Итак, физически мы подключили четыре компьютера в одну сеть. Теперь второй шаг, нужно настроить сетевую карту на каждом компьютере. Современные сетевые карты настолько автономны, что они сами «дружатся» с оборудованием, которое подключено на другом конце кабеля, будь то другой компьютер или маршрутизатор. Но тем не менее есть настройки, которые необходимо сделать пользователю. А именно указать каждой сетевой карте хотя бы один уникальный IP адрес в локальной сети. Как мы помним, это обязательное условие, которое регламентирует группа протоколов TCP/IP.

Кстати, возвращаясь к различиям между хабом и свитчем, можно еще раз сказать, что в случае с хабом отправленный пакет данных получат все компьютеры, даже с другими IP адресами. Когда такой пакет приходит постороннему компьютеру, сетевая карта должна сначала расшифровать заголовок пакета и понять, ей ли предназначается этот пакет данных. Когда сетевая карта понимает, что этот пакет данных предназначается не для нее, она просто очи-

[3] WAN – Wide Area Network (Глобальная компьютерная сеть).

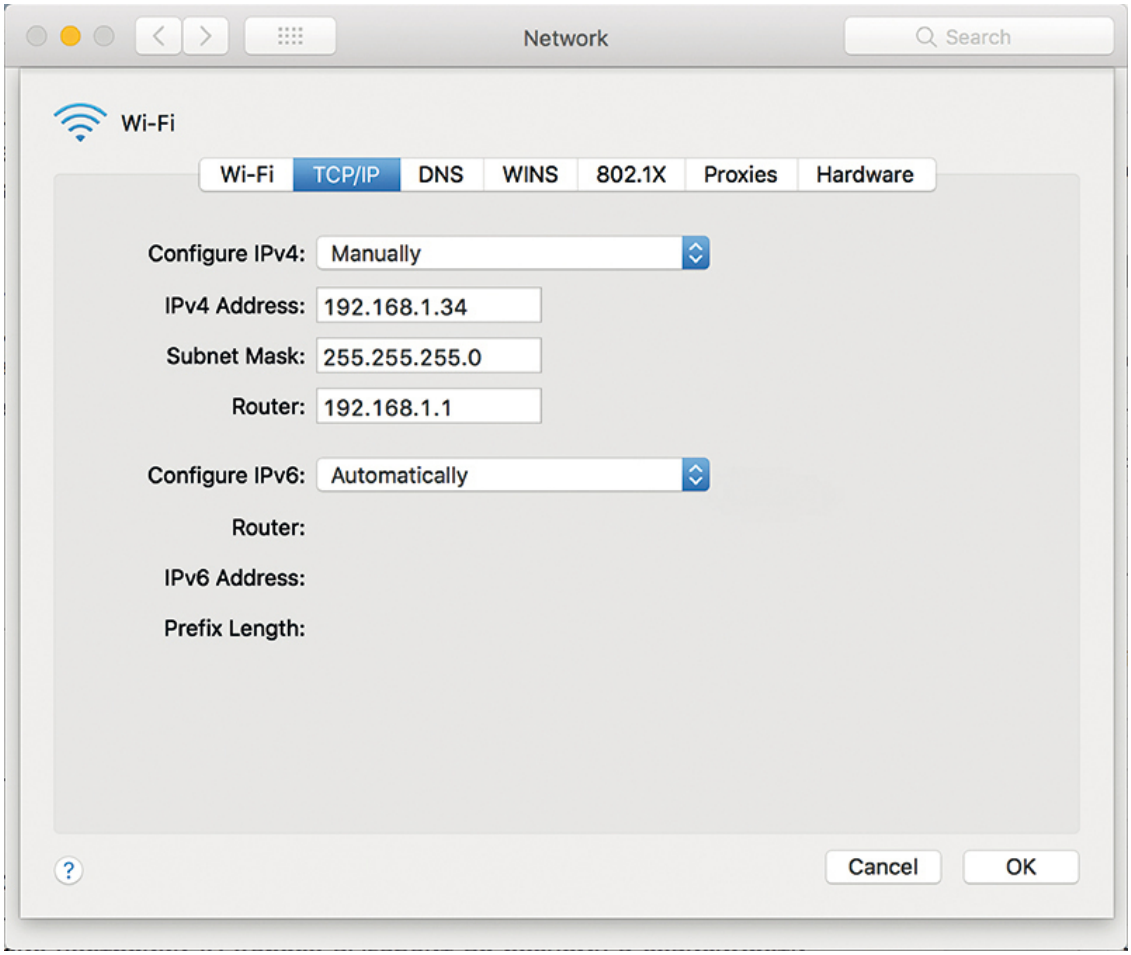
[4] LAN – Local Area Network (Локальная компьютерная сеть).

щает свой буфер памяти, который только что получил этот пакет данных, и готова принимать следующие данные. Но на это требуется время и вычислительные ресурсы сетевой карты. Что не рационально. В случае со свитчем, когда он получает пакет данных, он читает заголовок сообщения и, так как он уже знает, на каком физическом порте сидит клиент с этим IP адресом, свитч перенаправляет этот пакет данных конкретному клиенту с конкретным IP адресом. В то время как все остальные клиенты не расшифровывают чужие сообщения и готовы принимать пакеты, которые предназначены конкретно для них!

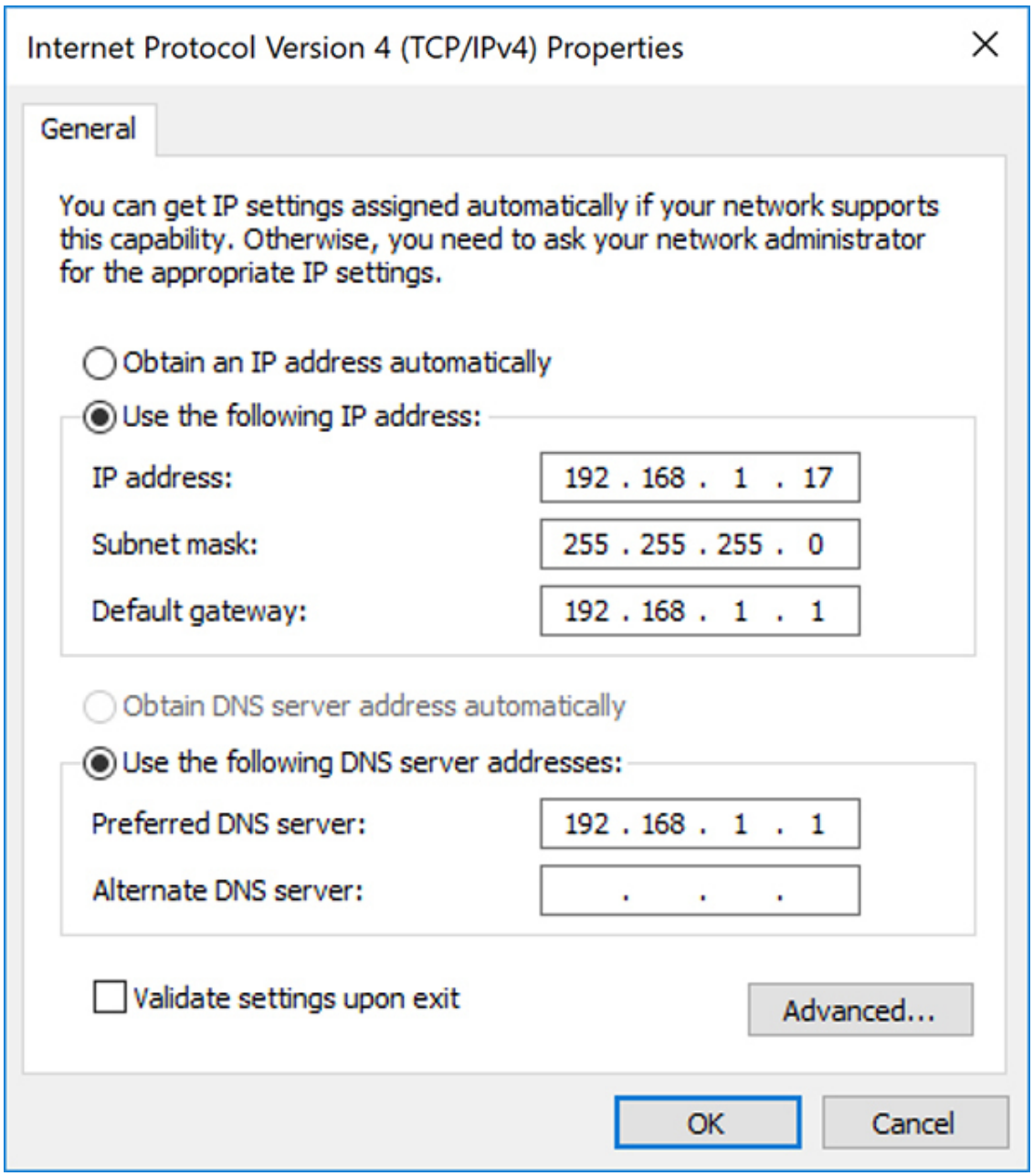
Вернемся к IP адресу сетевой карты. На данный момент существуют две версии IP адресов, IPv6 и IPv4, разница между ними в количестве байт, выделяемых для идентификации адреса или, говоря простым языком, они различаются длиной адреса. Пример IPv4 адреса: 192.168.1.100. Пример IPv6 адреса: FE:BA:76:32:FE:BA:76:32. Вы можете спросить, для чего же придумали более сложную версию IP адресов? Дело в том, что когда на свет появилась глобальная сеть интернет, никто даже не мог себе представить, что объема адресов IPv4 формата может не хватить для всех потребностей глобальной сети, ведь в этом формате мы можем идентифицировать 4,22 миллиарда уникальных адресов.

Но цифровой прогресс рос с огромной скоростью, и количество устройств в сети интернет с каждым годом росло в геометрической прогрессии! Адресов стало не хватать, и тогда ввели новую версию формата IPv6, благодаря которому можно задать 34*1037 адресов! Но так как мы зачастую настраиваем нашу личную локальную сеть, вероятность того, что мы исчерпаем лимит адресов IPv4 стремится практически к нулю, поэтому вполне достаточно использование более простой версии IP адреса.

Довольно теории, давайте уже перейдем к практике, а именно к непосредственной настройке адресов. На разных консолях, устройствах и контролерах интерфейс настройки IP адреса выглядит по-разному в зависимости от каждого производителя. Я предлагаю рассмотреть самые распространенные опции, которые предоставляют операционные системы Windows и MAC OS.



Окно настройки сетевой карты системы MAC OS



Окно настройки сетевой карты системы Windows

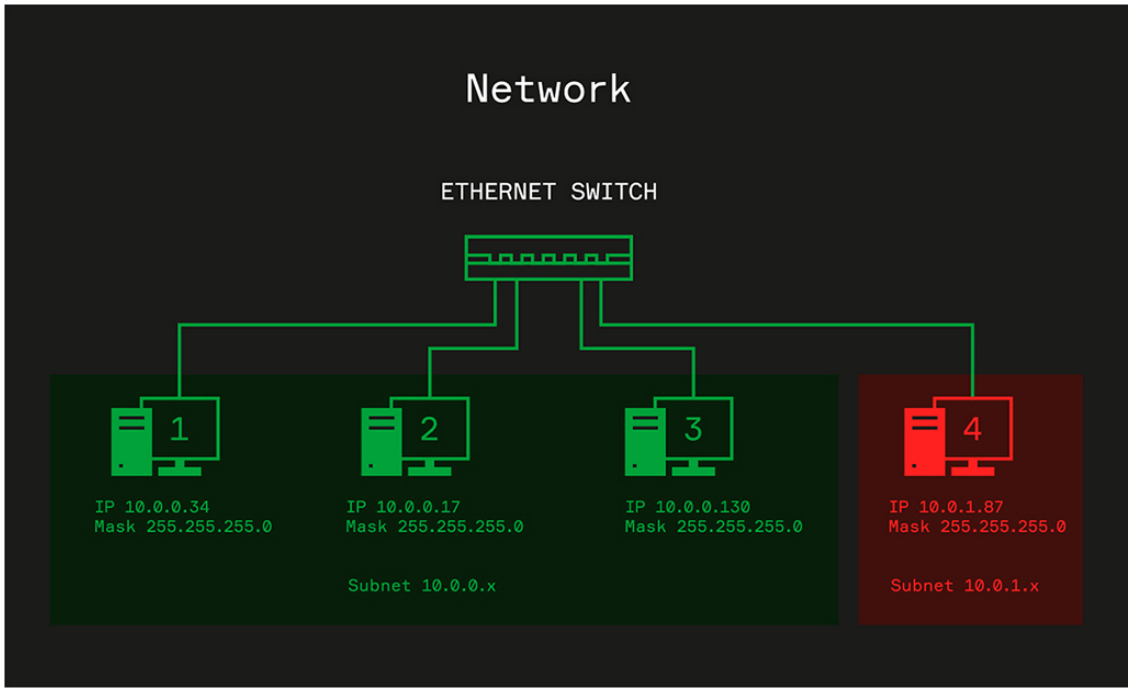
Ниже мы с вами затронем настройки таких параметров сетевой карты, как IP адрес, маска подсети (Subnet mask), Router (Defaulte gateway) и DNS Server.

Начнем с IP адреса. Возникает вопрос, какой диапазон адресов необходимо выбрать для наших устройств в сети? На самом деле, он может быть каким угодно, но все же стоит придерживаться правил, которые устанавливает всемирная организация IANA (Internet Assigned Numbers Authority). Функционал этой организации очень широк, но одна из ее задач – следить за тем, чтобы не было хаоса с адресами в сети интернет. Поэтому они четко регламентировали диапазоны адресов для локальных сетей. Ниже представлены эти диапазоны.

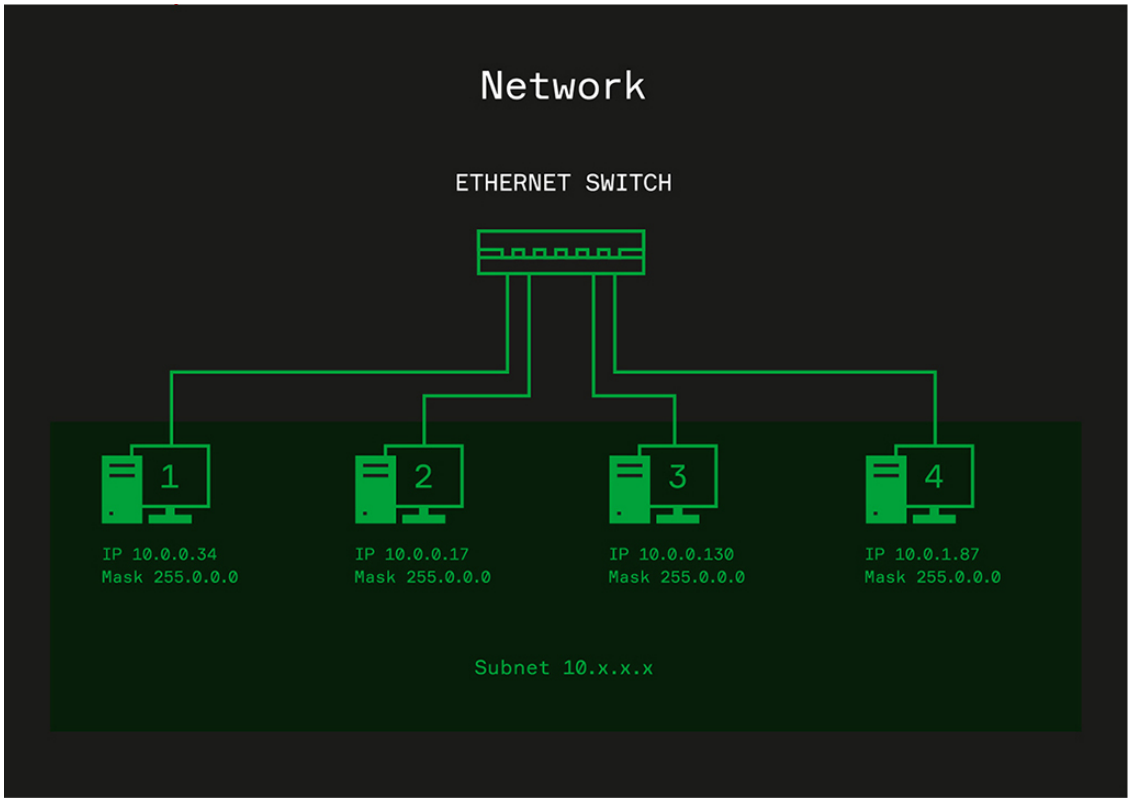
10.0.0.0 – 10.255.255.255
172.16.0.0 – 172.31.255.255
192.168.0.0 – 192.168.255.255

Раньше в этом списке был также диапазон адресов 2.x.x.x, но позже IANA исключила его из списка допустимых адресов локальных сетей. Вы можете выбрать любой из этих диапазонов и устанавливать адреса для ваших устройств и компьютеров, главное только, чтобы диапазон был один и тот же.

Но также немаловажная настройка сетевой карты – маска подсети! Я встречаюсь довольно часто с ситуацией, когда многие специалисты не знают, для чего вообще это нужно! Давайте разберемся. Итак, маска подсети позволяет нам разделить в пределах одного адресного диапазона сеть на несколько независимых подсетей. Разберем на примере.



Возьмем адресное пространство 10.x.x.x. Если мы хотим максимально закрыть нашу сеть, мы устанавливаем маску подсети 255.255.255.0. В этом случае, если мы хотим, чтобы наши компьютеры были в одной подсети и могли друг друга видеть, то их IP адреса должны иметь одинаковые первые три цифры адреса. К примеру, 10.0.0.x. Диапазон адресов нашей подсети будет составлять от 10.0.0.1 до 10.0.0.255. Но если в этом случае один из компьютеров будет иметь IP адрес 10.0.1.87, то он будет находиться в другой подсети (10.0.1.1—10.0.1.255), и для первой подсети он будет не виден.



Как же сделать четвертый компьютер видимым для всех остальных компьютеров? Есть два способа. Первый способ – ввести четвертый компьютер в адресное пространство подсети

10.0.0.x. Для этого нужно изменить IP адреса четвертого компьютера, к примеру, на 10.0.0.87. Обратите внимание, мы просто изменили третью цифру адреса с единицы на ноль.

Или второй способ – расширить нашу подсеть так, чтобы старый адрес четвертого компьютера попадал в нашу подсеть. Для этого достаточно на всех компьютерах расширить маску подсети, поменяв ее с 255.255.255.0 на 255.0.0.0. Теперь диапазон IP адресов нашей подсети составляет от 10.0.0.1 до 10.255.255.255. В этом случае, как мы можем видеть, IP адрес четвертого компьютера 10.0.1.87 попадает в диапазон адресов нашей подсети.

Вообще, я всегда рекомендую максимально открывать вашу рабочую подсеть, потому что в этом случае у вас меньше шансов ошибиться с настройкой IP адресов, и вы с легкостью сможете поднять вашу сеть без долгих согласований IP адресов.

Для начала работы локальной сети сетевой карте достаточно указать только IP адрес и маску подсети. Возникает вопрос, для чего же в сетевой карте присутствуют другие параметры адресов?

Зачастую рабочая локальная сеть изолирована от глобальной сети (интернет). Но в случаях, когда нашей сети необходима возможность доступа к глобальной сети, мы вынуждены познакомиться с еще двумя такими параметрами, как Router (Gateway) и DNS Server.

Как мы уже узнали, для корректной связи между собой двух независимых сетей используются роутеры, которые по определенным правилам перенаправляют пакеты данных из одной сети в другую. Когда локальный компьютер хочет отправить пакет данных на компьютер за пределами локальной сети, т.е. который находится в глобальной сети интернет, то компьютер отправляет такой пакет данных на роутер, а он в свою очередь знает, куда необходимо перенаправить данные. Любое взаимодействие с глобальной сетью интернет из локальной сети и обратно всегда происходит через сетевой роутер.

В настройках сетевой карты, в параметре Router указывается адрес роутера, на который будут отправляться и с которого будут приниматься все пакеты данных глобальной сети интер-

нет. Многие меня спрашивают, что будет, если мы укажем неверный адрес роутера или укажем в этом параметре IP адрес локального компьютера или вообще в нашей сети нет роутера? Все очень просто, на работу локальной сети это никак не повлияет. Если мы укажем в параметре Router несуществующий адрес, то попросту пакеты данных, предназначенные для выхода в глобальную сеть, будут уходить в никуда, и локальный компьютер никогда не получит ответ. Если мы укажем в параметре Router адрес другого компьютера или даже свой собственный, то компьютер, получая запросы и данные, попросту будет их игнорировать, так как он не будет знать, что с ними делать, исключая те случаи, когда на компьютере запущена специальная служба, которая занимается перенаправлением пакетов данных в глобальную сеть. Ведь функционал роутера может быть реализован не только на внешнем устройстве. Когда вопрос стоит больших вычислительных мощностей, на базе компьютеров создаются серверы, выполняющие задачи перенаправления пакетов данных.

Теперь давайте перейдем к следующему параметру сетевой карты. При создании любого интернет сайта ему присваивается уникальный IP адрес, благодаря которому к сайту можно получить доступ. Чтобы пользователям не пришлось запоминать IP адреса всех сайтов, придумали доменные имена, при помощи которых также можно получить доступ к интернет сайту.

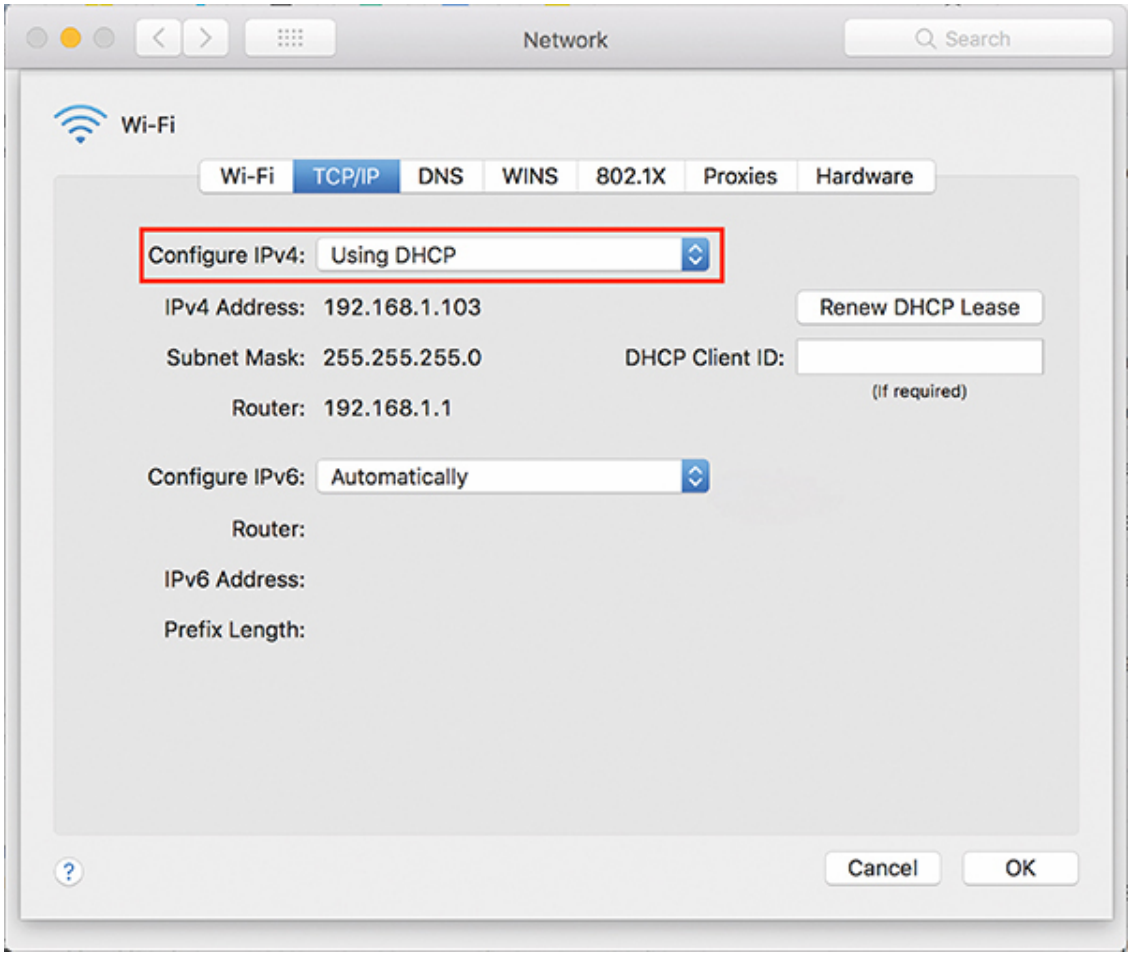
DNS – Domain Name System – «система доменных имен», это некая база, в которой хранятся ассоциации всех имен и IP адресов сайтов. Когда мы вводим в адресной строке имя интернет сайта, то DNS позволяет получить IP адрес сайта, чтобы его открыть. В настройках сетевой карты можно указать либо IP адрес одного из глобальных DNS серверов, либо адрес роутера, который будет заниматься получением адресов DNS сервера. Указание ошибочных или несуществующих адресов DNS никак не скажется на работоспособности локальной сети, но при этом не позволит получать доступ к интернет сайтам в глобальной сети.

Итак, для того чтобы клиент локальной сети имел доступ к глобальной сети интернет, в настройках сетевой карты должны быть указаны корректные адреса Router и DNS Server. В случае, когда активирована служба DHCP, эти адреса устанавливаются автоматически.

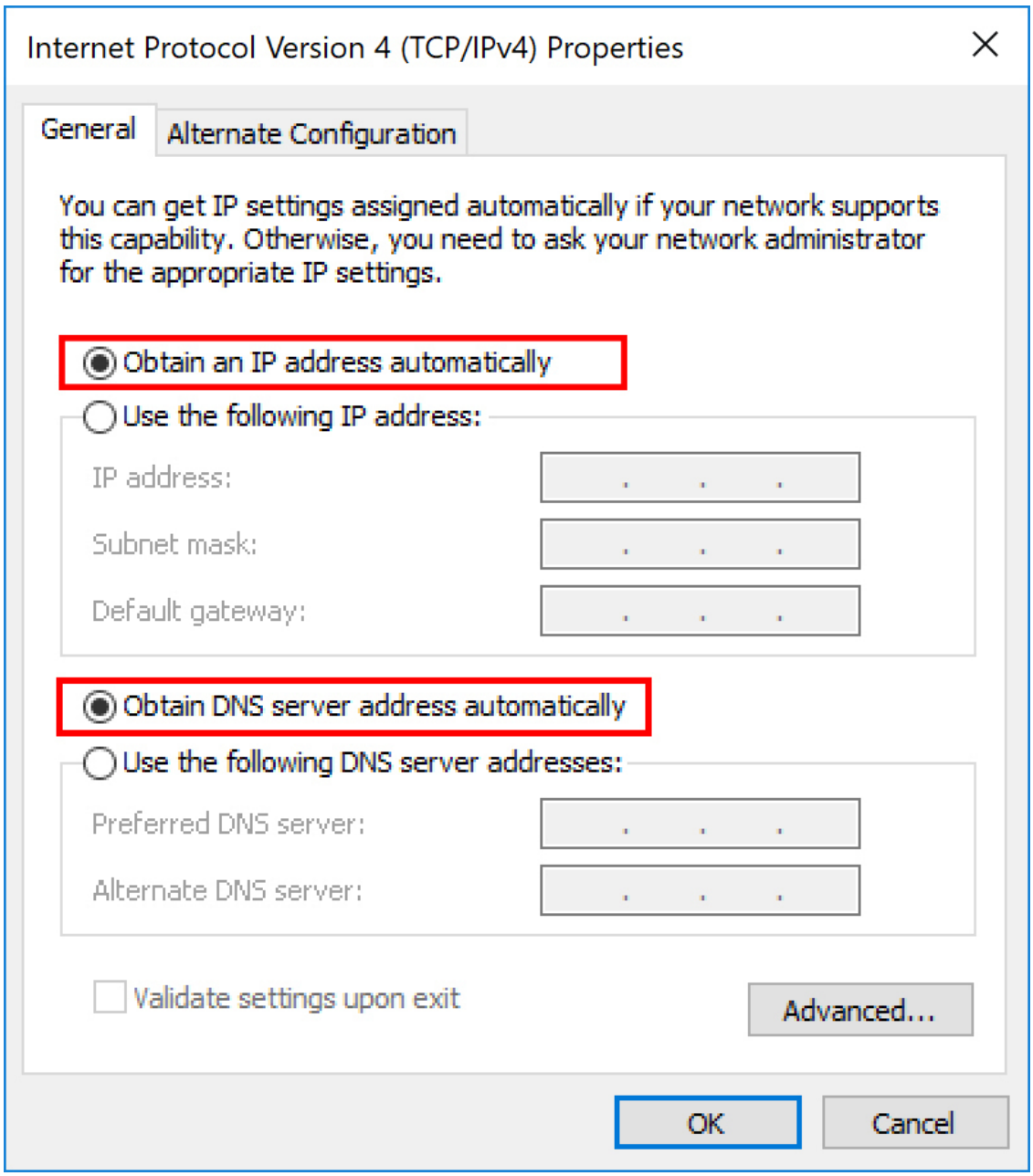
А теперь давайте познакомимся с волшебной службой DHCP (Dynamic Host Configuration Protocol). Это специальная служба, которая может автоматически всем компьютерам сети задать IP адреса и другие настройки сетевой карты так, чтобы все компьютеры были участниками одной подсети. Чтобы волшебство DHCP произошло, нужно соблюсти два условия.

Первое условие: на одном из участников сети должен быть создан и запущен DHCP сервер. Этот сервер можно поднять на Windows и на MAC OS, но очень часто такая служба присутствует на роутере. В этой службе указываются такие параметры как маска подсети и диапазон адресов, которые будут выдаваться новым клиентам сети.

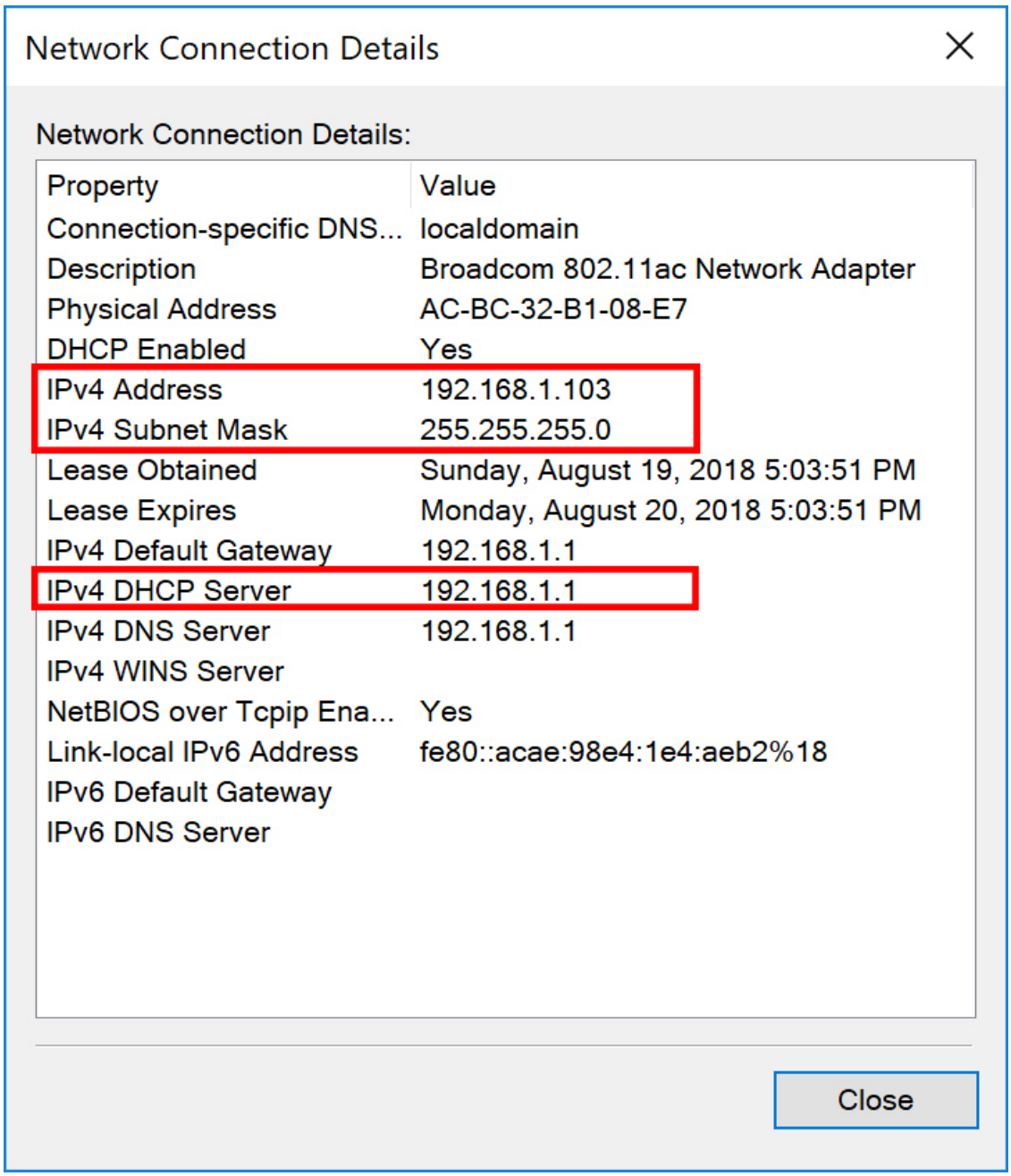
Второе условие: необходимо, чтобы на сетевой карте вашего компьютера была активирована опция автоматического получения настроек сетевой карты от DHCP сервера. Как мы видим в MAC OS системе, эта функция называется Using DHCP. А в Windows – Obtain an IP address automatically и Obtain DNS server address automatically.



Включение функции получения настроек сетевой карты от DHCP сервера на MAC OS

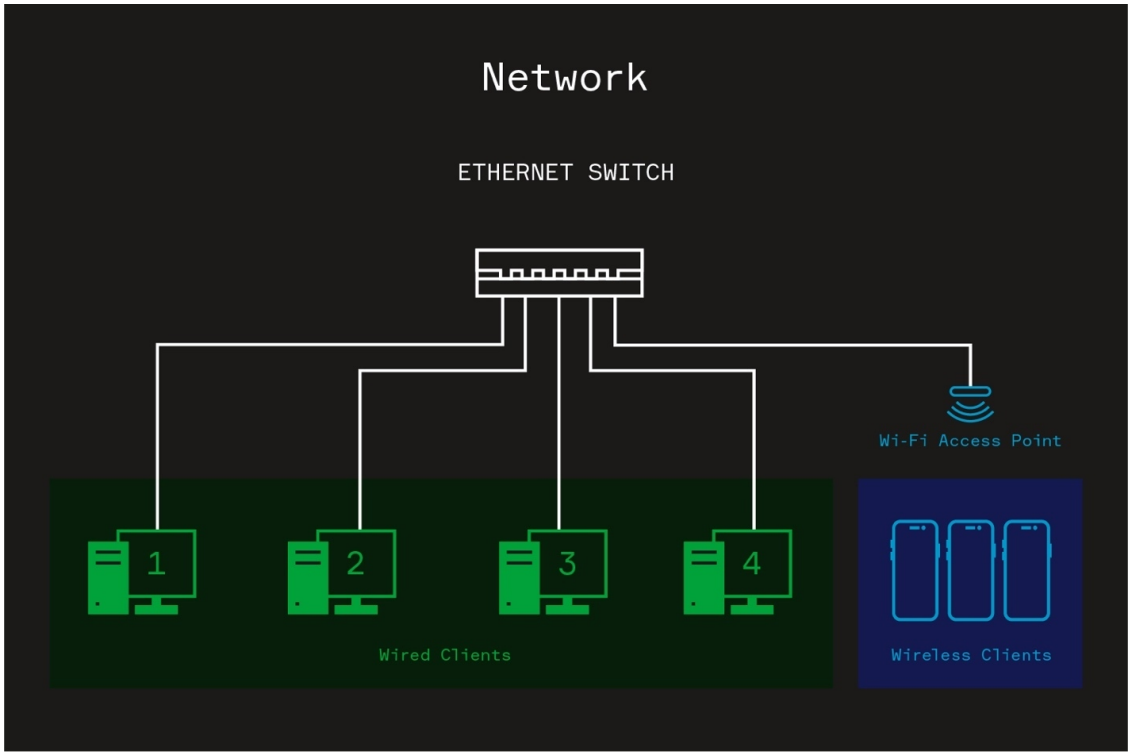


Включение функции получение настроек сетевой карты от DHCP сервера на Windows



Дополнительная информация сетевой карты, где можно увидеть полученные настройки и IP DHCP сервера

Но зачастую, когда мы работаем с синхронизацией, где нам приходится указывать конкретные IP адреса клиентов, которые должны получать наши данные, как, например, в протоколе OSC, автоматическая конфигурация сети – не самый лучший выбор, так как наши устройства не привязываются жестко к IP адресам. У DHCP есть такой параметр, как время аренды IP адреса клиента, по истечению которого DHCP сервер выдает клиенту новый IP адрес, и он может отличаться от старого. Если такое происходит, то наша синхронизация по сети рушится, так как у клиента, которому мы хотим отправить сообщения, теперь новый IP адрес. При статических IP адресах на сетевых картах адрес перманентен, и только пользователь может его изменить. Такая жесткая привязка настроек сетевой карты намного надежнее, что необходимо при настройке синхронизации по сети.



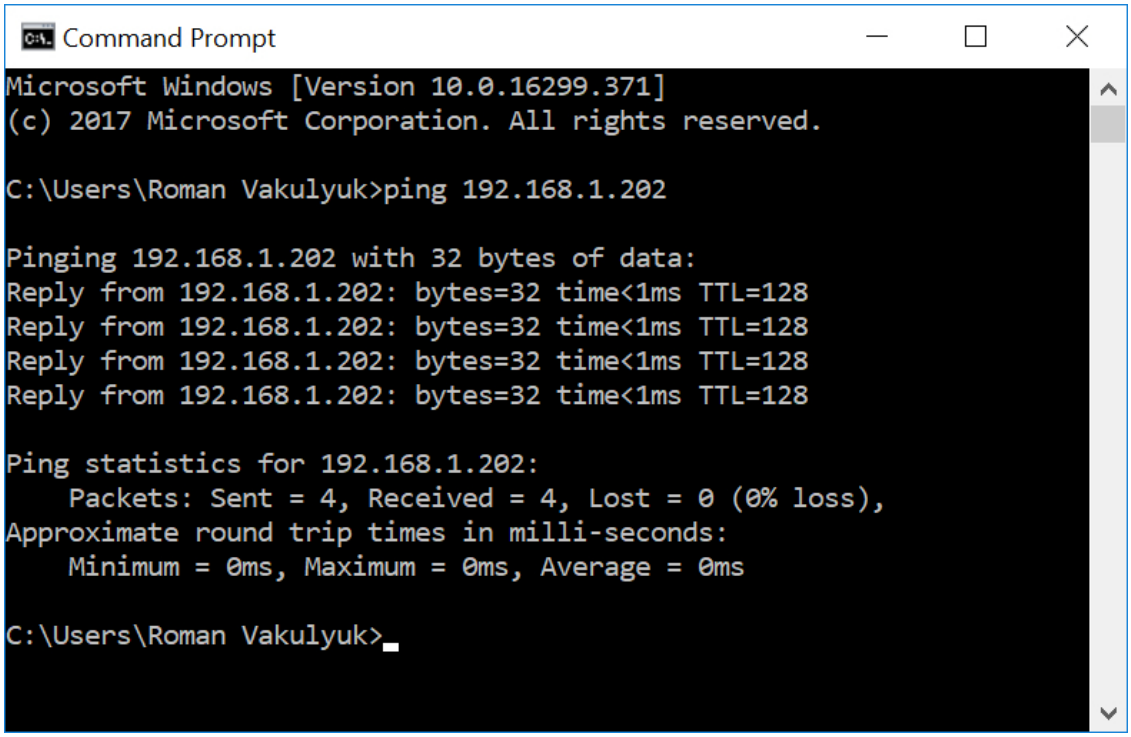
Упомяну также о беспроводной сети Wi-Fi. Часто приходится использовать в синхронизации смартфоны и планшеты, к примеру, для управления сервером синхронизации, которые также должны быть участниками общей сети. В этом случае нам достаточно ввести в нашу сеть беспроводную точку доступа (Wi-Fi Access Point) или Wi-Fi роутер, в котором уже встроена точка доступа. Все настройки Wi-Fi сетевых карт абсолютно идентичны, единственное, что перед тем, как подключиться к беспроводной сети, нам необходимо авторизоваться в ней, введя логин и пароль Wi-Fi, которые заданы в беспроводной точке доступа.

Проверка сетевого подключения

Если все сетевое оборудование настроено правильно и все клиенты подключены к сети, то теперь все участники могут обмениваться данными между собой. Чтобы проверить, действительно ли устройства видят друг друга, можно сделать тест, отправив тестовые пакеты на конкретный IP адрес.

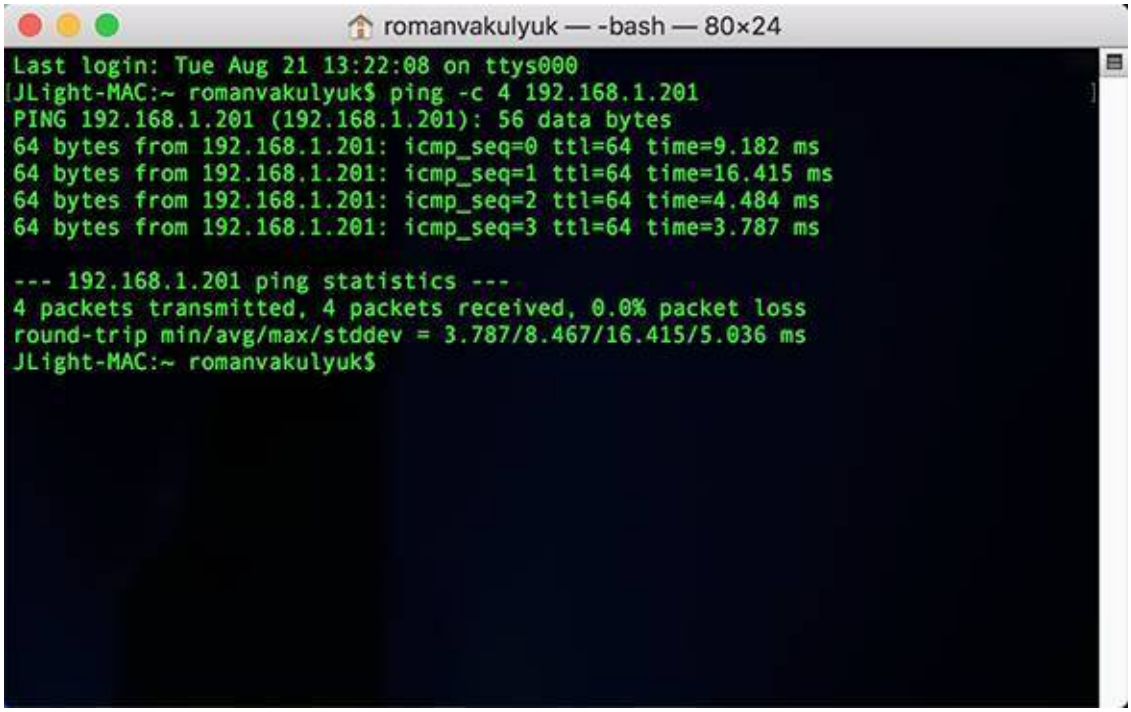
WINDOWS

Для этого в Windows системе необходимо открыть приложение командной строки. Чтобы это сделать, нажмите сочетание клавиш Win+R, в окне «выполнить» введите cmd, далее нажимаем кнопку OK. Перед вами откроется окно командной строки. Чтобы проверить подключение между вашим компьютером и другим клиентом, введите команду *ping 192.168.1.202* и нажмите Enter. Соответственно, вам нужно вести тот IP адрес, связь с которым вы хотите проверить. Если все успешно, то в отчете пинга адреса вы увидите *Received=4; Lost=0*; В противном случае командная строка выдаст вам отчет, где либо частично, либо все пакеты были потеряны. Это означает, что клиент с этим IP адресом недоступен или имеет какие-то неисправности с сетью.



MAC

Чтобы сделать такую же проверку в операционной системе MAC OS, необходимо открыть приложение терминала. Для этого зайдите в папку с вашими установленными приложениями и откройте приложение Terminal. Для проверки соединения с клиентом введите команду ping -c4 192.168.1.201. По итогу вы также получите отчет о пинге адреса. Если все в порядке, то вы не потеряете ни одного пакета, и терминал выдаст сообщение *0.0% packet loss*. Это означает, что соединение работает корректно.



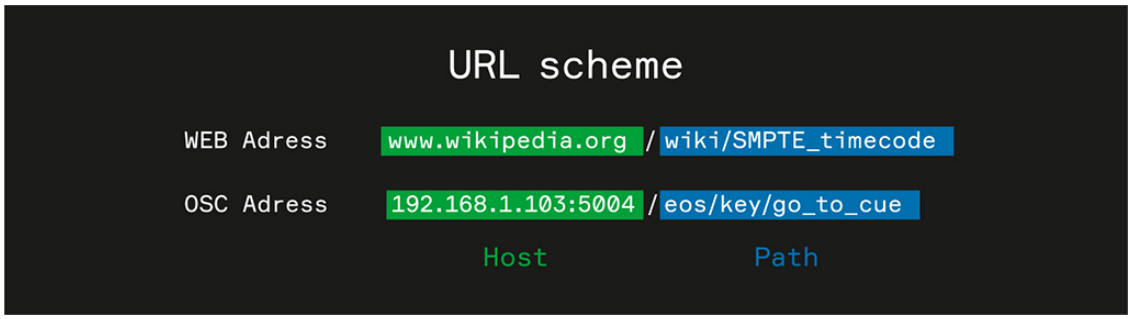
Итак, после того как мы подключили все оборудование, проверили подключения между участниками сети и убедились в том, что все работает корректно, можно приступить к настройке OSC.

Как работает OSC

Основываясь на той информации, которую мы разобрали выше, теперь можно более подробно поговорить об OSC и его структуре. Для передачи данных OSC использует транспортный протокол UDP и TCP. Поэтому для передачи и приема сообщений мы должны указывать порт данных и IP адрес клиента и сервера.

OSC очень похоже на MSC сообщения. Отличие в том, что сами сообщения и адрес клиента не регламентируются протоколом, как в MSC. В OSC регламентируется лишь правило описания адреса и сообщения. Любой производитель и программист могут придумать свои наборы сообщений и передать их через OSC.

Итак, мы хотим с одного компьютера через OSC отправить сообщение на другой компьютер. Для этого нужно указать в сообщении IP адрес получателя и его порт. Указание этих параметров зависит от каждого софта отдельно. Позже мы с вами разберем особенности настройки OSC на разных шоу системах.



Хорошо, OSC сообщение мы доставили в нужный порт, и программа клиента прочитала это сообщение. Но как же программе понять, к чему применить это сообщение? Для этого OSC сообщение содержит адрес назначения внутри программы клиента. Это очень похоже на параметр назначения MSC сообщения, к примеру, как Cuelist или Cue. Только, как я уже сказал выше, OSC не имеет жесткой привязки к синтаксису адреса как в MSC, но тем не менее мы должны соблюдать правила описания адреса, которое использует OSC, а именно URL (Uniform Resource Locator). Эту схему описания адреса пути вы используете каждый раз, когда пользуетесь интернет браузером, чтобы попасть на конкретное место в интернет сайте.

Эти пути назначения сообщения могут быть разнообразны в зависимости от того, какой функционал заложил конкретный производитель. К примеру, если вы хотите отправить OSC сообщение на световую консоль ETC Eos, то путь вашего сообщения должен начинаться с /eos, далее нужно указать группу контролируемых параметров пульта, к примеру, /fader, дальше нужно указать номер фейдера /1, и по итогу мы получим полный путь к конкретному фейдеру, который будет выглядеть как: /eos/ fader/1/. Также мы можем указать путь к группам, к спискам сцен и другому содержимому пульта.

Идем дальше. Теперь по аналогии с MSC вы можете предположить, что дальше в сообщении OSC передается команда! Верно, но тут есть своя особенность: в OSC сообщении передается не команда, а аргумент. В чем же отличие команды от аргумента? Дело в том, что аргу-

мент в OSC сообщении – это некий контейнер, который передает данные определенного типа. В последней версии OSC 1.1 вы можете использовать следующие типы данных:

Int32

Integer 32 bit, этот тип данных может хранить в себе натуральное число в диапазоне от -2 147 483 648 до 2 147 483 647. Этот тип используют, когда нужно передать целочисленный номер, как, например, для идентификации номера страницы или фейдера, так как в пульте не существует фейдеров и страниц с номером, к примеру, два с половиной или три целых четырнадцать сотых.

Float32

Float 32 bit, этот тип данных может хранить в себе действительное число с плавающей запятой в диапазоне от -3.4*1038 до +3.4*1038. Этот способ выражения действительного числа отличается от простого целочисленного типа. Но зато он позволяет закодировать более точные данные. Часто этим типом данных кодируют уровни фейдеров, вы можете определить диапазон фейдера от нуля до единицы, а вот точность позиционирования фейдера в этом диапазоне может быть огромной, но зачастую производители ограничиваются двумя знаками после запятой.

String

Этот тип данных передает строку, закодированную в формате ASCII. С помощью этого типа мы можем передать имя объекта или целое сообщение. Очень часто этот тип используется на системах дистанционного управления по OSC. К примеру, пульт может передать по OSC информацию об имени Cuelist, который назначен на конкретный фейдер.

Blob

Binary Large Object. Это тип данных, который передает оригинальный массив байтов. Очень часто его используют для передачи изображений, звука и видео.

Bool

Boolean – это логический тип данных, который может передать либо ложь, либо истину. Самое распространенное его использование – описание состояния переключателя, который может быть либо включенным (истина), либо выключенным (ложь). На самом деле, в типологии OSC этот тип данных разделен на две части, каждая из которых несет в себе конкретное состояние. Я объединил их воедино, дабы облегчить понимание этих типов.

Impulse

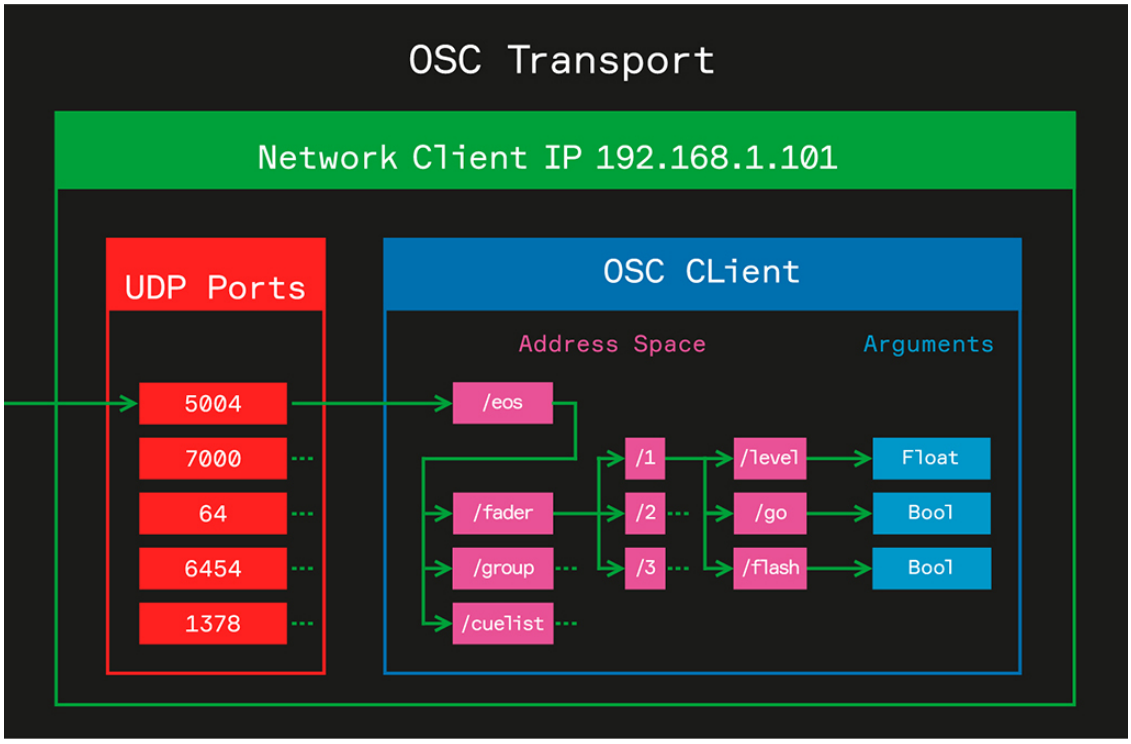
Это не совсем тип данных как таковой, так как он не несет в себе информации о состоянии аргумента, он инициализирует событие. В описании OSC протокола этот тип описывается как Bang. Частое его применение, когда вам нужно передать событие о действии, скажем, открытие страницы или любое другое событие без необходимости передачи конкретной информации в аргументе.

Null

Это пустой тип данных, который не содержит в себе ничего. Используется довольно редко, но как дополнительная опция присутствует.

Итак, давайте еще раз вспомним, из чего состоит OSC сообщение. Первая часть – это IP адрес клиента и номер его порта, на который нужно доставить OSC сообщение. Вторая его

часть – это адрес. И третья часть – аргумент. Схематически это будет выглядеть следующим образом.



Как видно на схеме, чтобы передать состояние кнопки Flash фейдера номер один на световую консоль Eos, мы должны указать сетевой адрес и порт пульта 192.168.1.101:5004. Далее нужно указать адрес необходимой кнопки, состояние которой мы хотим передать, /eos/fader/1/flash и по итогу передать аргумент типа Boolean, если кнопка должна быть нажата, то аргумент равен True, если кнопка отпущена, то аргумент равен False.

Итак, резюмируем особенности OSC протокола.

• OSC протокол базируется на интерфейсе передачи данных Ethernet. А это дает сразу несколько преимуществ. Для передачи такого сигнала мы можем использовать стандартное сетевое оборудование, которое намного распространеннее и доступнее, чем специализированные карты синхронизаций. По Ethernet мы можем передать сигнал практически на неограниченное расстояние, используя при этом разные способы передачи сигнала, как по радиоканалу, по оптике, так и по витой паре.

• OSC использует протокол передачи данных UDP и TCP. Эти протоколы обязывают нас указывать IP адрес и порт клиента, что дает нам множество преимуществ. К примеру, мы можем на одном сетевом клиенте синхронизировать несколько приложений одновременно, используя один и тот же IP адрес, но при этом разные порты. Это также позволяет нам настраивать сложные маршруты, делить OSC сигнал или получать на одного клиента сообщения с разных источников без использование дополнительного оборудования, так как этот функционал уже заложен в сетевых протоколах группы TCP/IP.

• OSC не регламентирует адрес к управляемым параметрам, каждый производитель может создать свою индивидуальную схему, которая будет максимально удобна для управления

конкретным функционалом. OSC регламентирует лишь правила описания этого пути, который базируется на URL системе.

• OSC позволяет передавать по заданному адресу аргумент, который может содержать разные параметры и типы данных. OSC также позволяет передать исходные байты данных для передачи в OSC сообщение изображения, звука и видео.

Я считаю, что OSC – самый функциональный и современный протокол синхронизации, с помощью него можно построить сложнейшие системы генеративной синхронизации с большой скоростью передачи данных. При этом, как я уже говорил, благодаря тому, что этот протокол базируется на физическом интерфейсе Ethernet, OSC наследует все преимущества передачи данных по этому интерфейсу. Что делает его намного привлекательнее перед остальными командными типами протоколов синхронизации.

ALTERNATIVE SYNC PROTOCOLS

В этой главе мы с вами познакомимся с различными протоколами, которые появились на фоне активного развития интерфейсов передачи данных. Некоторые из них базируются на уже хорошо нам известном протоколе MIDI, а некоторые были созданы с нуля для решения определенных задач синхронизации.

RTP-MIDI (Apple MIDI)

Благодаря тому, что интерфейс передачи MIDI полностью цифровой, то без особых изменений пакеты данных MIDI можно передавать через более современные и быстрые интерфейсы. Ассоциация MMA понимала, что MIDI стал довольно популярным стандартом для работы в разных индустриях, но при этом развитие этого стандарта упиралось в технические особенности физического серийного интерфейса, на котором базировался MIDI. И тогда MMA ассоциация начала смотреть в сторону других успешных технологий передачи данных, чтобы уйти от технических недостатков, которые накладывал старый интерфейс передачи данных. И по итогу на свет начали появляться новые технологии передачи MIDI.

RTP (Real-time Transport Protocol) – это протокол высокого уровня, который базируется на UDP, но при этом имеет свои преимущества, которые были специально разработаны для стриминга аудио и видео. Основная его особенность в том, что каждый пакет данных этого протокола имеет в заголовке абсолютное время отправки, которое может прочитать принимающее устройство и определить задержку и порядок доставки сообщений. Такие преимущества идеально подошли для MIDI, и в 2004 году на свет появилась первая версия протокола RTP-MIDI. Позже компания Apple включила этот протокол в состав своих операционных систем и активно дорабатывала его. И как следствие этого, протокол получил второе название Apple MIDI. Позже был написан отдельный драйвер для Windows и Linux, который также позволял использовать протокол в этих системах.

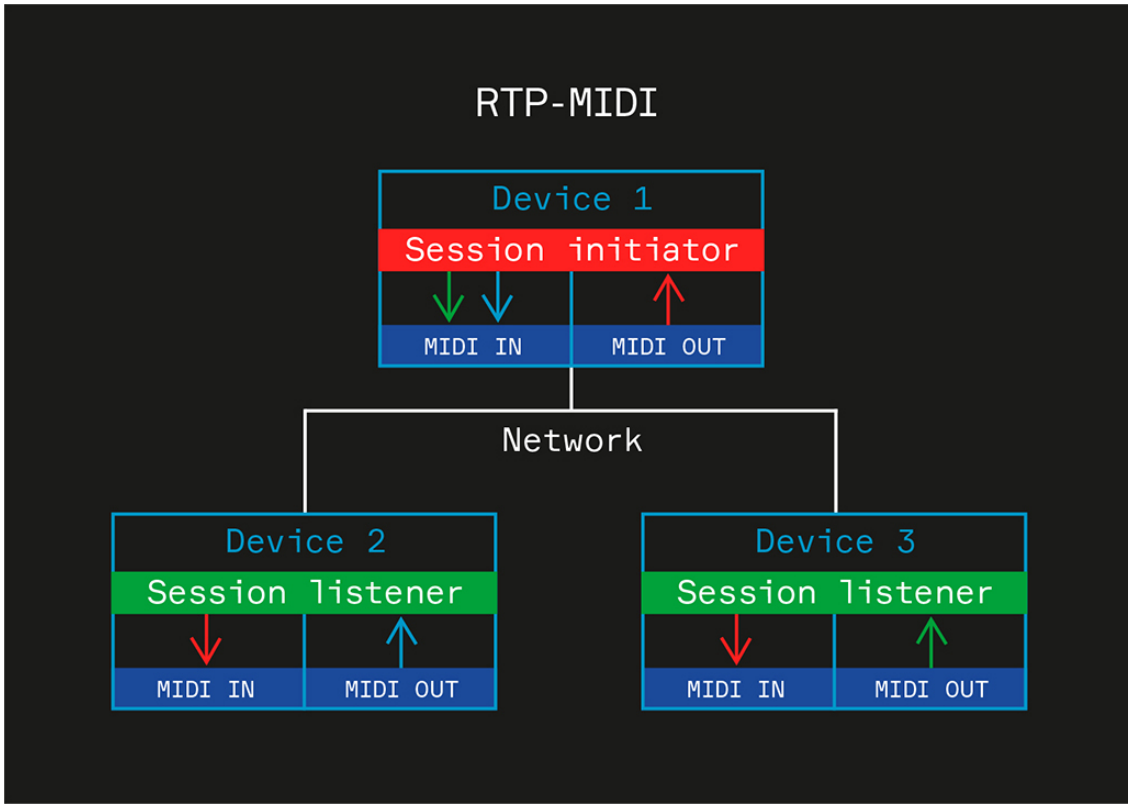
Так же, как и в случае с OSC, этот протокол базируется на физическом интерфейсе Ethernet, и как следствие, RTP-MIDI также наследует все преимущества этого интерфейса передачи данных. Соответственно, чтобы использовать протокол RTP-MIDI, так же как и с OSC, изначально необходимо поднять сеть, в которой будут находиться все наши сетевые клиенты. Как это сделать, мы уже узнали в главе «Концепт построения сетей».

Давайте теперь поговорим об идеологии этого протокола. В RTP- MIDI есть такое понятие, как «Сессия». «Сессия» – это виртуальная среда, к которой могут подключаться клиенты, для того чтобы обмениваться MIDI сообщениями. Для начала в сети должен быть тот, кто создаст эту сессию. Это может быть либо компьютер, либо другое устройство. Создатель сессии будет являться мастер устройством или, говоря терминологией RTP-MIDI, «Инициализатором сессии». После того как в сети будет создана сессия, другие клиенты могут к ней подключиться и стать участниками этой сессии. В сети может быть создано несколько сессий, и они могут работать независимо. Но при этом удобство заключается в том, что клиент сам может выбрать, к какой сессии ему необходимо подключиться. После того как клиент стал участником сессии, в операционной системе появляются виртуальные MIDI порты, которые могут использовать приложения для приема и передачи MIDI сигнала.

Одно из немаловажных преимуществ RTP-MIDI заключается в том, что на уровне этого протокола реализованы схемы разделения и смешивания MIDI сигналов (Split/Merge). Ниже представлена простейшая схема транспорта сообщений между разными участниками сессии, где видно, что устройство номер один является инициализатором сессии, к которому подключены другие участники сети. При отправлении MIDI сообщения с главного устройства (Device 1) это сообщение автоматически дублируется на все остальные клиенты. Но при этом, если сообщения отправят другие клиенты сессии (Device 2 и Device 3), то они будут получены только инициализатором сессии, т.е. устройством номер один. Эти сообщения автоматически будут соединены и направлены на его виртуальный MIDI IN порт.

Так как RTP-MIDI – это по сути лишь способ передачи MIDI через Ethernet, то все, что касается протокола MIDI, остается по-прежнему тем же самым, единственное, что отличается, это способ доставки MIDI сообщений. По этой причине предлагаю разобрать пример того, как

создать RTP-MIDI сессию и как подключить к ней клиентов для обмена сообщениями. Чтобы в будущем вы сами для себя решали, использовать ли физические MIDI карты и коммутацию для работы и экспериментов или использовать сетевую альтернативу RTP-MIDI.



Ниже представлен список операционных систем и название программ, которые добавляют в систему возможность работы с протоколом RTP-MIDI.

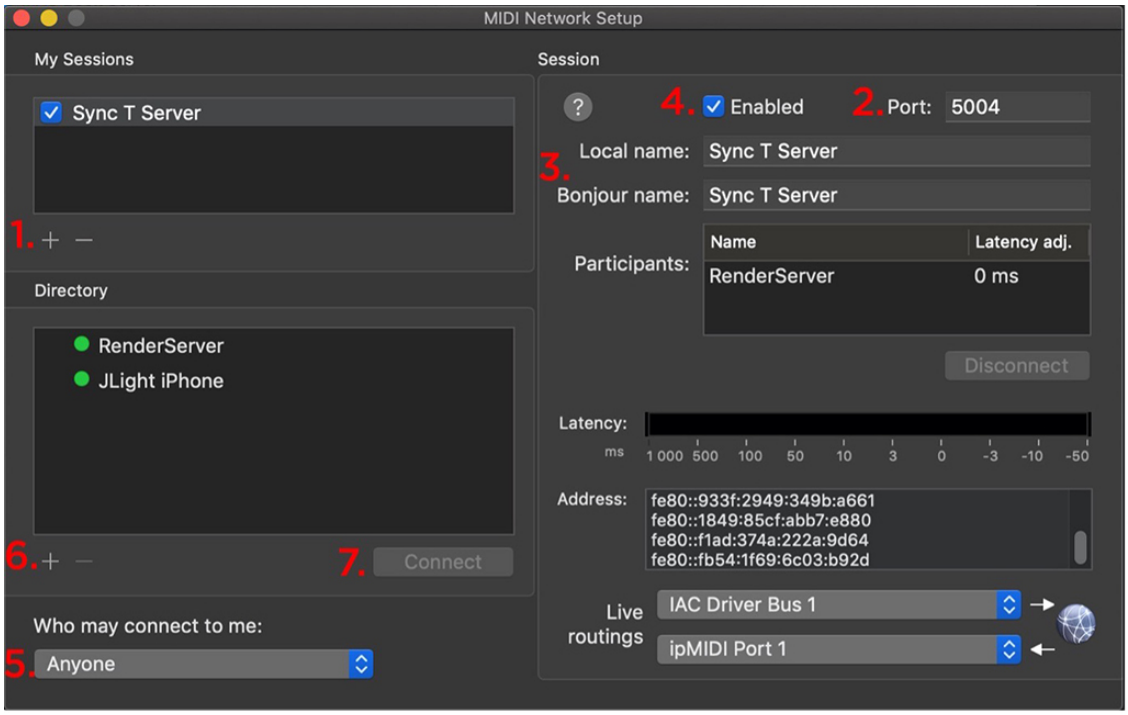
Операционная система	Программное обеспечение
MAC OS	Протокол RTP-MIDI поддерживается операционной системой.
iOS	Протокол RTP-MIDI поддерживается операционной системой.
Windows	rtpMIDI (by Tobias Erichsen)
Android	nmj — Network MIDI for Android (by Humatic)

Некоторые программисты, которые читают эту книгу, возможно, спросят, а где же Linux? Существуют библиотеки, которые позволяют интегрировать поддержку этого протокола во внутрь отдельной программы в момент разработки приложения программистами, эти библиотеки есть для всех операционных систем, в том числе и Linux. Приложения под Windows и Android были написаны программистами-энтузиастами, которые выложили свои программы в открытый доступ, за что им огромное спасибо.

Если вы пользователь устройств Apple, то вам ничего не нужно устанавливать, так как, как я уже говорил, RTP-MIDI уже интегрирован в системы MAC OS и iOS. А вот для других операционных систем нужно скачивать специальные драйверы и программное обеспечение.

Предлагаю создать сессию в системе MAC OS. Приложение для Windows выглядит абсолютно идентично с таким же интерфейсом и функционалом. Итак, чтобы открыть меню для работы с RTP-MIDI в MAC OS, зайдите в папку Applications и откройте приложение Audio MIDI Setup. Если перед вами откроется окно настроек аудио без окна MIDI, в панели меню откройте Window и в выпадающем меню нажмите на опцию Show MIDI Studio, и перед вами откроется окно, содержащее устройства MIDI.

Для работы с RTP-MIDI нам нужно нажать на иконку глобуса в верхнем правом углу. Давайте разберемся, что здесь к чему



Итак, чтобы создать новое подключение, необходимо:

1. Ниже раздела Sessions, с правой стороны окна настройки RTP-MIDI, необходимо указать номер порта, который будет использоваться для передачи и приема этого протокола. Как мы помним, RTP-MIDI использует низкоуровневый протокол UDP, а как следствие, для приема и передачи данных мы должны указать конкретный порт. По умолчанию RTP- MIDI

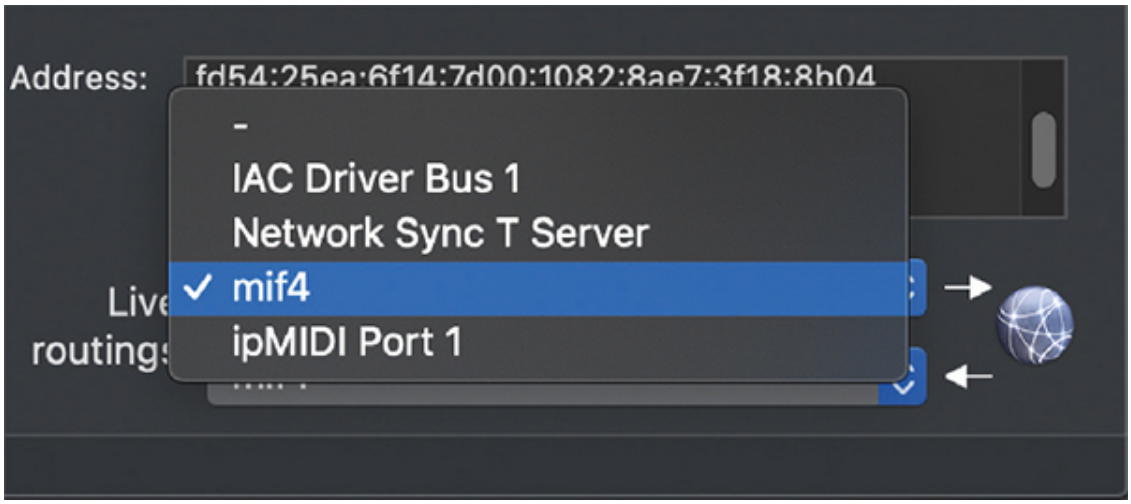
использует порт 5004, но если этот порт уже занят каким-нибудь приложением, мы можем указать любой другой свободный порт.

2. Далее необходимо указать Local name, это имя виртуальной MIDI карты, которая будет использоваться в нашей системе. И также необходимо указать Bonjour name, это имя сессии, которую будут видеть клиенты в сети.

3. Теперь нужно активировать сессию, для этого необходимо поставить галочку напротив Enabled верхней части окна Session.

4. И также необходимо настроить правила безопасности, которые позволяют настроить права подключения к сессии. В нижней части окна настроек RTP-MIDI слева раскройте выпадающий список Who can connect to me: и выберите одну из опций. Так как клиенты сами могут подключаться в сессию, иногда нужно ограничить правила подключения.

5. Если необходимо добавить нового клиента, вручную нажмите кнопку ниже окна Directory и введите имя, адрес и порт клиента, который должен стать участником вашей сессии. В завершении, чтобы добавить клиентов в вашу сессию, выберите устройство в окне Directory и нажмите кнопку Connect. Если клиент успешно добавлен в сессию, он появится в окне Participants:



Список MIDI портов, сигнал с которых приложение RTP-MIDI может перенаправить в сессию

Хотел бы также упомянуть об очень удобной опции, которая позволяет перенаправить потоки MIDI с сессии на физическое устройство и наоборот. Для этого в правой нижней части программы, напротив Live routings, раскройте необходимый список оборудования на вход или выход и выберите MIDI порт, на который необходимо направить поток с сессии или, наоборот, отправить поток в сессию. В этом случае наш компьютер

выступает как конвертер, который преобразует между собой физический MIDI и сетевой RTP-MIDI.

Подводя итог всех особенностей протокола RTP-MIDI, можно сказать:

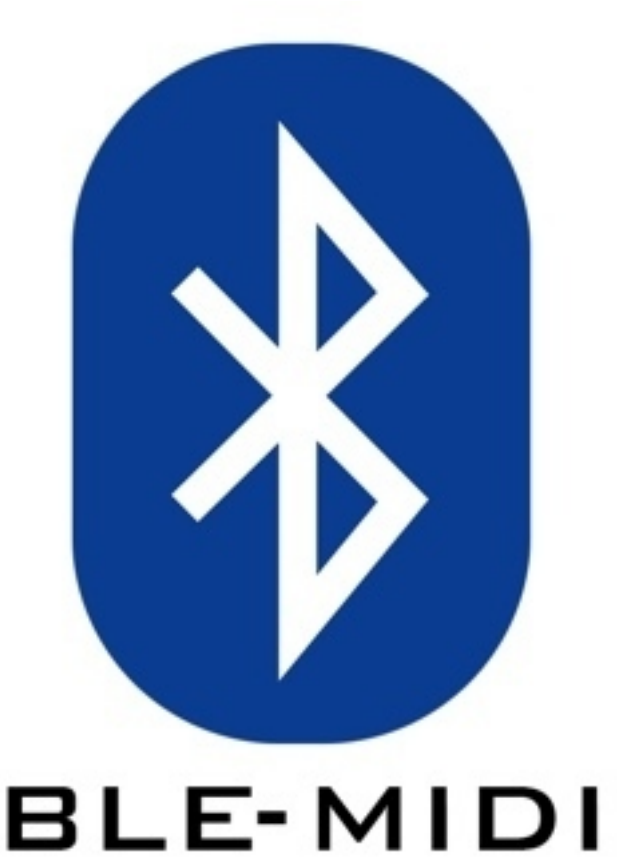
• Первая особенность, этот протокол абсолютно совместим со всеми MIDI протоколами, которые были разработаны для передачи через классический серийный интерфейс данных MIDI.

• Вторая, для работы с RTP-MIDI нет необходимости использовать MIDI карты для передачи сообщений MIDI, так как протокол был разработан для работы через локальную сеть, вам достаточно иметь любое устройство, которое имеет сетевую карту, для подключения к сети.

• Третья особенность вытекает из второй благодаря тому, что этот интерфейс работает через локальную сеть, он наследует все преимущества сетевых технологий передачи данных Ethernet, Wi-Fi и оптоволокна, одни из которых – дальность передачи и скорость сигнала, что является одним из недостатков классического интерфейса MIDI.

• И четвертая, в RTP-MIDI сессии уже реализована идеология соединения и разделения MIDI сигнала между клиентами, что также позволяет оказаться от использования специализированных MIDI сплиттеров и мерджеров.

BLE-MIDI



Bluetooth уже довольно взрослая беспроводная технология, которая активно используется в мобильных устройствах. А так как в мобильных устройствах всегда уделяли внимание экономии ресурсов батареи, то на свет появилась новая, более усовершенствованная версия Bluetooth, которая намного умнее использует ресурсы мобильного устройства. И по итогу новая версия стала называться Bluetooth Low Energy или, сокращенно, BLE, также у него есть и второе название – Bluetooth SMART. Так как новый стандарт стал более привлекателен, производители стали активно внедрять новую технологию в свои устройства. MMA (MIDI Manufacturers Association) не стало исключением, в 2014 разработав новый протокол BLE-MIDI, передаваемый через Bluetooth Low Energy. И как следовало ожидать, компания Apple не заставила себя долго ждать и в этом же году объявила о поддержке протокола BLE-MIDI в своих вновь вышедших релизах операционных систем iOS 8 и OS X 10.10. В течение года компания Apple исправляла ошибки и дорабатывала этот протокол. И в 2015 году ассоциация MMA приняла все их изменения и утвердила финальную версию BLE-MIDI.

Теперь поговорим об особенностях BLE-MIDI. В отличие от RTP- MIDI, он не настолько функционален. К примеру, по этой технологии вы можете настроить подключение только между шестнадцатью устройствами. По причине того, что скорость Bluetooth не настолько велика. В сравнении с RTP-MIDI мы можем создать практически неограниченное количество сессий в одной системе, количество зависит от скорости нашей сети и мощности системы, на которой поднимаются сессии. И также Bluetooth имеет ограниченный радиус действия, до 10 метров прямой видимости. Но есть и плюсы, так же как и с RTP-MIDI, BLE-MIDI отправляет

пакеты данных с абсолютными маркерами времени, что также обеспечивает синхронность приема пакетов данных. Также отличие BLE-MIDI подключения от других Bluetooth соединений в том, что BLE-MIDI использует отличную от стандартного Bluetooth систему передачи данных по радиоканалу, которая обеспечивает минимальные задержки на кодирование данных, передачу их по радиоканалу и декодинг для обеспечения максимальной скорости передачи данных между устройствами.

После этого всего вы можете задать вполне ожидаемый вопрос, для чего же тогда нужен BLE-MIDI, если у него столько недостатков по сравнению с RTP-MIDI? Ответ очень прост. Для работы с BLE-MIDI нам не нужно никакого дополнительного оборудования, не нужно создавать проводную или беспроводную сеть, мы просто включаем встроенный Bluetooth в устройстве, подключаемся к клиенту и начинаем работать. Это очень удобно, когда у нас нет необходимости в создании сложной системы синхронизации. К примеру, нам нужно просто с телефона отправить MIDI команды на компьютер, к которому привязаны определенные действия. Или нам нужно просто проверить нашу систему управления по MIDI, но у нас нет под рукой физически MIDI пульта, мы можем с эмулировать работу MIDI консоли с телефона, подключившись по BLE-MIDI.

Теперь об идеологии подключения BLE-MIDI. Хотя бы одно из BLE-MIDI устройств должно стать инициализатором подключения. В понятии BLE-MIDI это устройство должно начать «рекламировать» свое подключение, чтобы клиент мог его увидеть и подключиться. Как только клиент подключился к устройству, которое инициализировало подключение, мастер устройство перестает «рекламировать» свое подключение и спаривается с клиентом. Теперь устройства могут обмениваться сообщениями.

Также хочется упомянуть еще об одной особенности. Стандартные интерфейсы настройки подключения к Bluetooth устройствам по умолчанию не видят BLE-MIDI оборудование и его подключение. К примеру, когда мы на компьютере или на телефоне хотим подключить новое Bluetooth устройство, мы заходим в стандартное меню поиска новых устройств, находим в списке устройство и подключаем его.

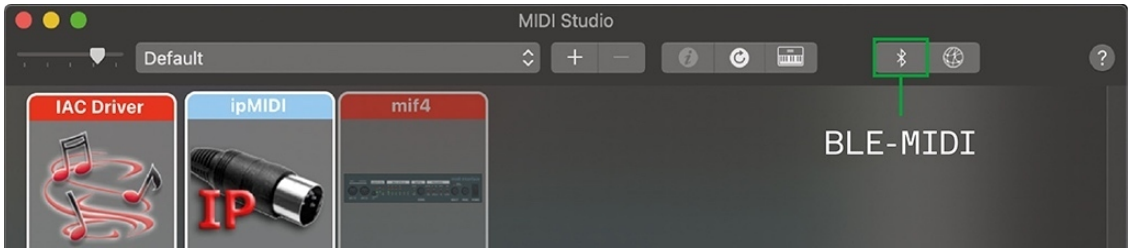
На Apple устройствах через стандартное меню мы не сможем найти BLE-MIDI устройства, а на Windows и Android мы увидим устройство, но либо нам не удастся к нему подключиться, либо подключение через некоторое время будет обрываться. Дело в том, что BLE-MIDI подключения отличаются от стандартных устройств, и для его подключения нам нужно использовать дополнительные программы, которые могут распознавать BLE-MIDI. Обычно на смартфонах и компьютерах устанавливается дополнительный софт, который может обнаружить такие соединения и также создать виртуальные MIDI порты.

Как я уже говорил, Apple – первая компания, которая интегрировала BLE-MIDI в свои операционные системы. В MAC OS уже есть необходимое программное обеспечение, которое позволяет и «рекламировать» подключение, и также видеть и подключаться к BLE-MIDI устройствам. В iOS хоть такая поддержка и есть, но отдельного софта для работы с этим протоколом нет. Apple реализовала возможность подключения оригинальной библиотеки BLE-MIDI от Apple для интеграции работы с этим протоколом в приложениях. Это значит, что если производитель софта хочет открыть в своем приложении возможность работать с этим протоколом, то ему достаточно подключить стандартную библиотеку. В других системах, Windows, Android и Linux, для работы с этим протоколом нужно ставить дополнительное программное обеспечение, которое будет позволять видеть такие подключения и подключаться к ним с эмуляцией виртуальных MIDI портов. Хорошая новость в том, что таких приложений существует не одно, и пользователь сам может выбрать, с каким ему удобнее работать. Пример разных приложений и их возможностей мы разберем в главе «Программное обеспечение для работы с протоколами синхронизации».

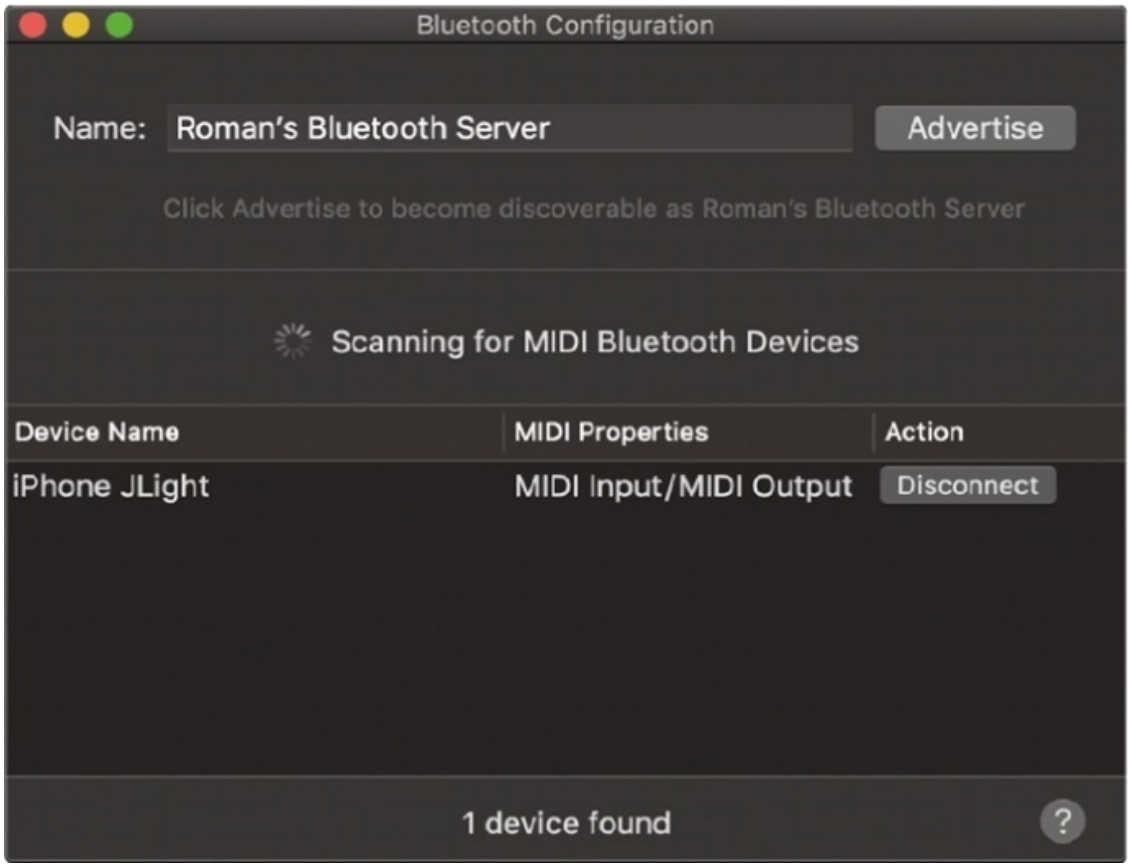
А сейчас предлагаю посмотреть, как устроено приложение для работы с BLE-MIDI в MAC OS.

Чтобы попасть в настройки BLE-MIDI в MAC OS, так же, как и в случае с RTP-MIDI, нам нужно открыть окно MIDI Studio. Напомню, для этого нужно зайти в папку Applications и открыть приложение Audio MIDI Setup. Если перед вами откроется окно настроек аудио, без окна MIDI, в панели меню откройте Window и в выпадающем меню нажмите на опцию Show MIDI Studio. Перед вами откроется окно со всеми MIDI картами, которые подключены к системе, и также вы увидите иконки для доступа к дополнительным меню для работы с технологиями передачи MIDI.

Чтобы открыть меню настройки технологии BLE-MIDI, необходимо нажать на иконку Bluetooth в правой части панели управления.



Как мы видим, здесь все довольно просто. В поле NAME мы можем задать имя BLE-MIDI подключения, которое будут видеть остальные клиенты. Чтобы начать «рекламирование» своего подключения, достаточно нажать кнопку Advertise справа от поля ввода NAME. Ниже мы можем увидеть список устройств, которые также «рекламируют» свое подключение, и также тех клиентов, которые уже подключились к нам. Если мы хотим подключиться как клиент к серверу, нам нужно просто нажать на кнопку Connect напротив того устройства, к которому мы хотим подключиться.



Резюмируя особенности BLE-MIDI протокола, можно сказать:

• Основное предназначение BLE-MIDI – организация простой синхронизации между несколькими устройствами.

• BLE-MIDI – это исключительно беспроводная технология передачи MIDI сообщений, ограниченная дистанцией работы Bluetooth, в среднем это до 10 метров.

• Чтобы увидеть и подключить BLE-MIDI устройства, необходимо приложение, которое поддерживает работу с этим интерфейсом передачи данных.

• Пакеты данных, передаваемые по BLE-MIDI, имеют в заголовке маркеры времени, что позволяет принимающему устройству вычислить задержку на доставку и определить правильный порядок полученных пакетов данных.

• И также, благодаря технологии BLE-MIDI, сообщения поэтому протоколу передаются с минимальными задержками, что делает его вполне надежным и быстрым.

TC Net

Протокол TCNet был создан в 2004 году программистами компании TC Supply. Основное направление этой компании – создание программных продуктов для синхронизации шоу систем с DJ оборудованием, поэтому по большей части весь функционал этого протокола опирается на потребности клубной сферы. Базируется этот протокол на технологии передачи данных Ethernet.

Несмотря на то, что этот протокол синхронизации имеет ряд преимуществ при работе с таймкодом в сравнении с SMPTE, в профессиональной концертной индустрии он широко не используется. Но при этом в клубной сфере это один из самых функциональных протоколов для синхронизации.

По своей сути этот протокол очень схож с RTP-MIDI. Он также работает через транспортный протокол UDP, также каждое устройство рекламирует себя в сети, в одной сети также может быть несколько мастер устройств, и также может быть множество клиентов, которые могут сами выбирать, данные с какого сервера получать.

Если говорить об отличиях, то так как TC Net не был принят ни одной из официальных ассоциаций, то единственная компания, которая поддержала этот протокол, была Pioneer, в сравнении с RTP-MIDI, которую по итогу модернизировала компания Apple, чьи изменения впоследствии были приняты двумя ассоциациями MMA и JMSC. И также компания Apple включила поддержку этого протокола в свои операционные системы, что несомненно явилось решающим фактором в популяризации этого протокола.

Но пожалуй, основное отличие этих двух протоколов в том, что RTP-MIDI передает данные серийного протокола MIDI, а TC Net работает полностью со своими собственными данными. С одной стороны, это плохо, потому что этот протокол полностью несовместим со старыми системами, а с другой стороны, такой протокол передачи данных синхронизации не зажат рамками возможностей старых технологий.

Итак, давайте более подробно разберем, что может протокол TC Net. Несмотря на название самого протокола, он передает информацию не только о таймкоде, но и дополнительно метаданные.

Timecode

Как и следовало ожидать, это более усовершенствованная версия классического SMPTE таймкода.

- Через TC Net можно передать несколько источников таймкода.
- Скорость таймкода может изменяться в пределах 0%—200%.
- Направление таймкода может быть как вперед, так и назад.

Media Information

TC Net может передавать текущую информацию о медиа данных с каждого DJ проигрывателя.

- Трек и аудио метаданные.
- BMP и скорость воспроизведения.
- Данные для построения волновой формы трека.
- Название трека, альбома, исполнителя, изображение обложки альбома.
- Служебные сообщения для управления сторонними программами.

В общей сети TC Net выделяются три типа устройств.

Master

Этот тип устройства может предоставлять в сеть таймкод и отправлять дополнительные мета данные и команды управления.

Slave

Этот тип устройств может принимать данные таймкода, мета данные и команды управления, при этом Slave тип устройств может отправлять только команды управления.

Repeater

Этот тип устройств может принимать и отправлять все данные синхронизации и управления.

Позже мы с вами еще вернемся к этому протоколу, когда будем обсуждать возможные варианты синхронизации DJ оборудования с шоу системами.

ArtTimeCode

Скорее всего, вы уже не в первый раз слышите о такой компании, как Artistic License, и о протоколе передачи данных ArtNet, главная цель которого – передача данных серийного протокола DMX512 через сеть Ethernet. Но мало кто знает, что этот протокол также поддерживает передачу данных таймкода. Модификация протокола ArtNet для передачи таймкода называется ArtTimeCode, также многие для упрощения называют его ArtNet Timecode.

В отличие от предыдущего протокола, ArtNet Timecode передает только данные времени, но при этом он также поддерживает большие отклонения от скорости воспроизведения таймкода и его направления.

Технология передачи таймкода довольна проста и очень схожа с ArtNet протоколом.

• В сети есть только одно мастер устройство, которое может транслировать либо Broadcast, либо Unicast пакеты данных, все остальные клиенты принимают эти данные.

• Мастер устройство не знает, есть ли в сети хоть кто-то, кто может принять данные таймкода, и также мастер устройство никогда не узнает, если из сети пропал один из клиентов.

• В протоколе нет системы контроля корректности порядка доставленных пакетов данных, что не может гарантировать корректное время таймкода на конечном устройстве.

• Но из-за того, что этот протокол максимально прост, общее время задержки между мастер устройством и клиентом минимально.

Протокол ArtNet сейчас широко используется в шоу-индустрии, а вот его собрат ArtTimeCode из-за своих существенных недостатков так и не получил широкого применения в профессиональном оборудовании и программном обеспечении.

OSC Timecode

OSC Timecode появился из среды объектно-визуального программирования нодами (Node[5]). Как яркий пример такого программного обеспечения, можно привести VVVV и TouchDesigner. Так как протокол OSC очень прост в использовании, то очень часто программистами он используется для передачи кадров таймкода между компьютерами и своими серверами. Этот способ получил популярность по двум причинам.

• Для передачи такого таймкода не нужно никакого специализированного оборудования, наподобие звуковых и MIDI карт.

• Ноды (Node[6]) и библиотеки для работы с протоколом OSC присутствуют во всех подобных программных средах разработок. Для работы с другими протоколами синхронизации, базирующимися на технологии Ethernet, подобными RTP-MIDI, TCNet и ArtNet Timecode, программистам необходимо писать на низкоуровневом языке собственные ноды для расшифровки сообщений этих протоколов, что зачастую является большой проблемой, так как необходимо знать на более высоком уровне языки программирования. В случае с OSC подобного не требуется.

Так как OSC является протоколом синхронизации, в котором нет четкого определения самих сообщений, то производители программного обеспечения и оборудования сами определяют их формат. В принципе, любой программист, работая на любой платформе, может придумать свои собственные OSC сообщения для передачи данных синхронизации.

Если мы вспомним суть работы любого таймкода, изученного нами ранее, то увидим, что таймкод – это набор сообщений, передаваемых с определенной частотой, несущих в себе один кадр таймкода, который содержит абсолютное значение времени. В зависимости от протокола, с сообщением времени может передаваться дополнительная информация.

Кадр OSC Timecode представляет из себя одно OSC сообщение, где аргументом является беззнаковая переменная типа Int32, в которой хранится информация таймкода одного кадра. Адрес такого сообщения не регламентирован, в зависимости от каждого проекта или программиста, он может быть придуман произвольно. Самый распространенный и простой адрес – /timecode/. При необходимости использовать независимые источники тайм кода в адрес добавляют номер источника или устройства.

Хочу обратить внимание, что по сути OSC Timecode никак и никем не стандартизирован, поэтому и адреса, и даже форматы аргументов в OSC Timecode сообщениях могут различаться в зависимости от программы, особенно в кастомных. Поэтому информация в этой главе лишь для понимания сути этого протокола таймкода. Если вам придется работать с таким типом протокола, то обязательно уточняйте ключевые параметры таких OSC сообщений.

[5] Node – это программный логический блок, который выполняет определенные задачи, извлечение данных из протоколов, логические вычисления, вывод данных и информации.

[6] Node – это программный логический блок, который выполняет определенные задачи, извлечение данных из протоколов, логические вычисления, вывод данных и информации.

AVB/Dante/MADI

Audio Video Bridging (AVB), Dante, Multichannel Audio Digital Interface (MADI) – все это сетевые протоколы передачи несжатого многоканального цифрового аудиосигнала с минимальными задержками через сеть Ethernet.

Может возникнуть вопрос, каким образом эти цифровые аудио интерфейсы могут использоваться для синхронизации? Давайте вспомним один из самых старых интерфейсов LTC. Как мы уже разбирали, это цифровой протокол, передаваемый через аналоговый аудиоинтерфейс. Для его передачи мы можем воспользоваться оригинальным аналоговым звуковым трактом или его современными цифровыми модификациями. Своего рода, это возрождение старого стандарта в новом цифровом интерфейсе. Наподобие серийного интерфейса MIDI, который переродился в цифровой протокол RTP-MIDI.

Использование AVB/Dante/MADI протоколов для передачи LTC является спорным вопросом, так как кроме очевидных преимуществ есть также и ряд недостатков. Давайте выделим основные плюсы и минусы такого способа передачи LTC.

Преимущества:

• AVB/Dante/MADI использует технологии сети Ethernet, что позволяет передать цифровой сигнал по витой паре или оптическому кабелю на большие расстояния с минимальными задержками и потерями.

• Практически все профессиональные цифровые аудио пульты, звуковые карты, плееры и другое аудио оборудование работает с цифровым протоколом AVB, Dante или MADI.

• Существуют цифровые аудио драйвера AVB/Dante/MADI для компьютеров, которые позволяют в операционной системе транслировать аудио по цифровому протоколу, даже без наличия звуковых карт. Более того, Apple MAC OS по умолчанию может работать с AVB без установки дополнительных драйверов.

Недостатки:

• Используя AVB/Dante/MADI, возможно передать только LTC таймкод.

• При работе с LTC через оборудование, не предназначенное для этого, сложно контролировать корректность передачи сигнала.

• Для подключения физических клиентов LTC таймкода все равно необходимо аудио оборудование для конвертации AVB/Dante/MADI в аналоговый аудиосигнал.

• Для работы с AVB/Dante/MADI необходимы профессиональные знания для использования этих протоколов.

Много лет назад, когда существовал только интерфейс таймкода LTC, особого выбора не было, но сейчас, когда есть множество разных цифровых интерфейсов синхронизации, которые намного надежнее и функциональнее, игнорировать новые разработки и пытаться оживить устаревшие протоколы неразумно.

Когда стоит вопрос использования современных интерфейсов передачи данных в синхронизации, зачастую это обусловлено необходимостью передачи сигнала на большие расстояния, что не могут позволить себе интерфейс MIDI или LTC.

Конечно, можно проявить находчивость, потратить время на изучение протоколов AVB/Dante/MADI и передавать через них LTC, но я задам лишь один вопрос: зачем изобретать велосипед, если все уже летают на космических кораблях? Если все же встал вопрос передачи синхронизации на большие расстояния, куда рациональнее использовать RTP-MIDI, OSC или даже ArtNet Timecode и TCNet.

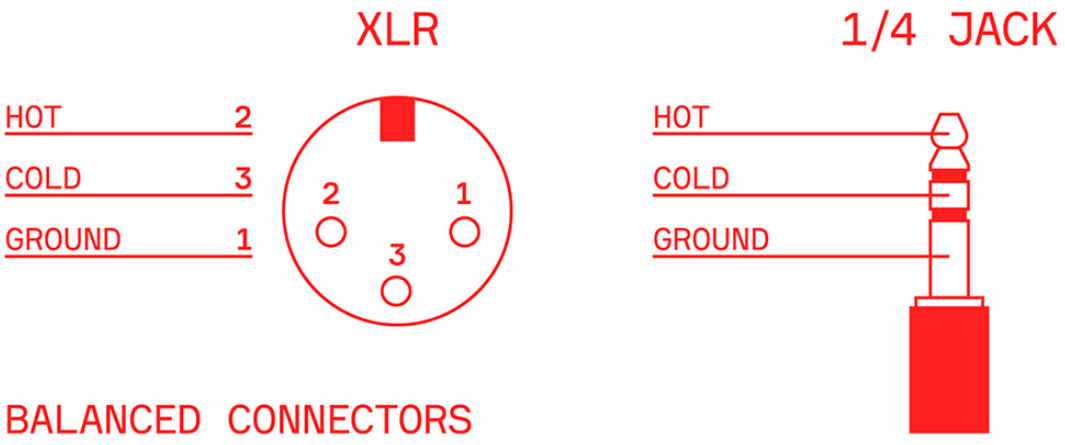
EQUIPMENT AND COMMUTATION

Важным звеном в цепочке синхронизации является оборудование, которое непосредственно генерирует и принимает протоколы синхронизации. Чтобы синхронизация работала корректно, нужно знать, какое оборудование и для каких целей использовать. Также немаловажной составляющей является коммутация, которая, в зависимости от используемого протокола, диктует свои правила эксплуатации.

Все самые распространенные протоколы синхронизации, которые используются в шоу-индустрии, передаются посредством нескольких интерфейсов LTC, MIDI и Ethernet. Мы подробно разберем все технические особенности в работе с каждым из них.

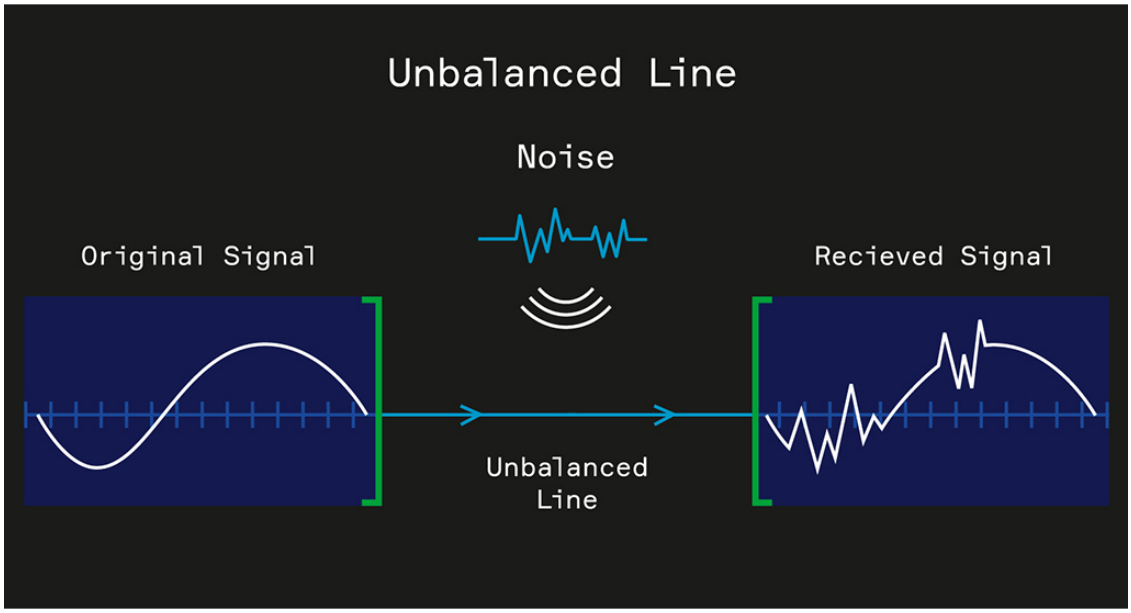
Передача и прием LTC

LTC – это линейный временной код, передаваемый через аналоговый аудиоканал. Частота работы этого сигнала находится в пределах звукового диапазона, поэтому этот сигнал легко воспринимается звуковым оборудованием и его можно воспроизвести на аудиосистеме. Звук, который мы услышим, будет схож со звуком, который издавали раньше DialUP модемы при подключении к интернету, т.к. и в первом, и во втором случае передается цифровой сигнал по аналоговому каналу.



Для генерации LTC сигнала необходима либо звуковая карта с балансными аудиовыходами, либо таймкод LTC генератор. Самый распространенный разъем, который используется для LTC, это звуковой XLR коннектор, так как этот разъем имеет три пина, что необходимо для передачи балансного сигнала. Помимо этого, разъем имеет замок фиксатор, что делает его более надежным. Также в звуке используется балансный 1/4 Jack (TRS) разъем, который также имеет три пина для передачи сигнала. Некоторые производители для передачи, приема LTC также используют этот разъем, что вполне допустимо.

Стандартные звуковые карты с mini-Jack выходами плохо подходят для работы с линейным таймкодом по причине того, что звуковые выходы у таких карт не балансные. Чтобы понять, почему это важно, давайте разберем основные принципы и различия балансного и небалансного типов сигнала.

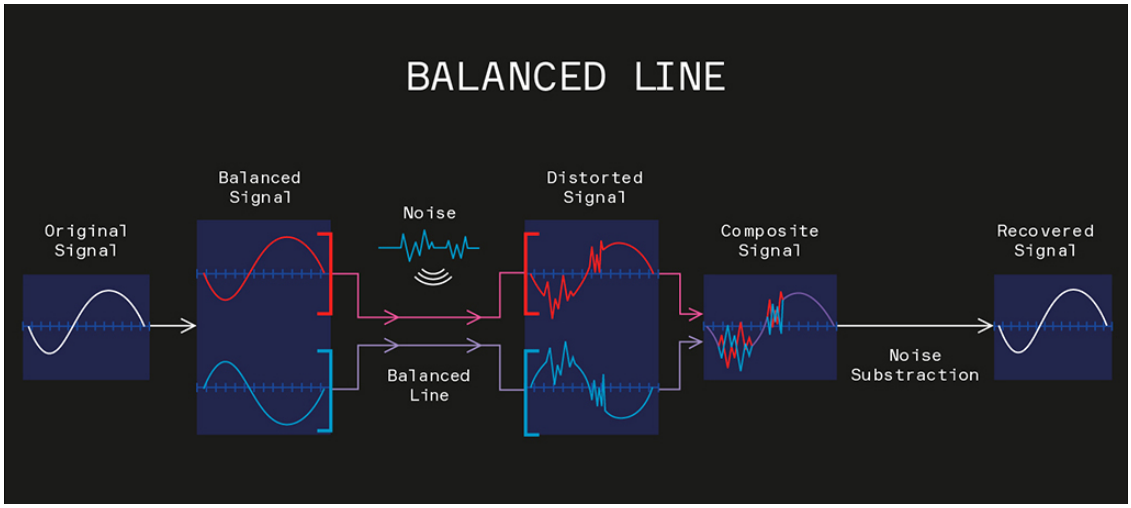


Небалансный (также его называют несимметричный, линейный) сигнал передается по двум проводам: один провод сигнал, другой – земля. Данный способ передачи сигналов отличается исключительной простотой реализации, однако он не способен противостоять помехам на физическом уровне.

Балансный сигнал использует для передачи три провода: два сигнала HOT и COLD (прямой и инверсный) и третий провод земля. Электрическое сопротивление сигнальных проводов по отношению к земле сбалансировано (то есть равно), что нашло отражение в названии. Сигналы по двум проводам балансной линии передаются в противофазе с равной амплитудой относительно земли.

Балансные входы и выходы звуковой карты

Операционный усилитель на балансном входе устройства вычитает из прямого сигнала инверсный, в результате помехи, одинаково наведенные на две фазы такой линии, вычитаются, а полезный сигнал увеличивается по амплитуде в два раза.



В сравнении с небалансным подключением балансное имеет два основных преимущества, обусловленных его техническими особенностями.

Первое преимущество – возможность передавать аналоговый и цифровой сигнал без существенных искажений на значительное расстояние.

Это, прежде всего, обусловлено в два раза большей разностью потенциалов между прямым и инверсным сигналом балансной линии в сравнении с разностью потенциалов между землей и единственным сигналом небалансной линии.

Вторая особенность, тесно связанная с первой, – лучшая в сравнении с небалансным подключением помехоустойчивость, которая достигается таким фактором, как анализ наведенных шумов на две линии сигналов в противофазе.

Небалансное подключение позволяет получать стабильный сигнал без существенных искажений на расстояниях, на практике не превышающих 15—20 метров. А при использовании балансного аудио подключения мы можем передать сигнал на расстоянии 200 метров и более. Но тут же хочу обратить внимание, что это не значит, что мы можем передать LTC на то же расстояние, что и аудиосигнал. Причина тому – ряд условий, которые не позволят нам это сделать.

Если не учитывать эти условия, то без потерь мы можем передать по балансной линии сигнал LTC на расстояние не более 30–50 метров. Чтобы понять, какие условия не могут нам позволить передать этот сигнал на большее расстояние, давайте сперва изучим особенности передачи цифрового LTC по аудиоканалу, а потом вновь вернемся к этому вопросу.

Большинство внешних звуковых карт имеет балансные выходы, что позволяет более качественно работать с передачей линейного таймкода, также эти карты позволяют полноценно работать с двумя или более каналами аудио и при этом на отдельном канале работать с LTC, что невозможно со встроенными двухканальными картами.

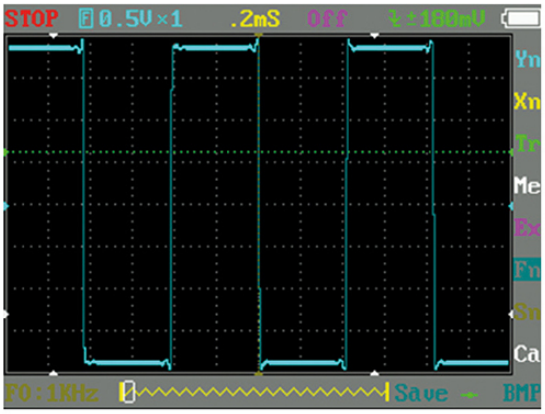
Интерфейс синхронизации LTC изначально был придуман для использования на телестудиях, где и по сей день используется самое профессиональное оборудование.

Longitudinal Time Code был разработан не для передачи через низкокачественный звуковой тракт и тем более не через домашние аудиокарты и разъем mini-Jack. Он был разработан для работы с профессиональными балансными входами и выходами, которыми были оснащены все профессиональные аудио и видеосистемы. Да и передавать LTC на дальние расстояния не было необходимости.

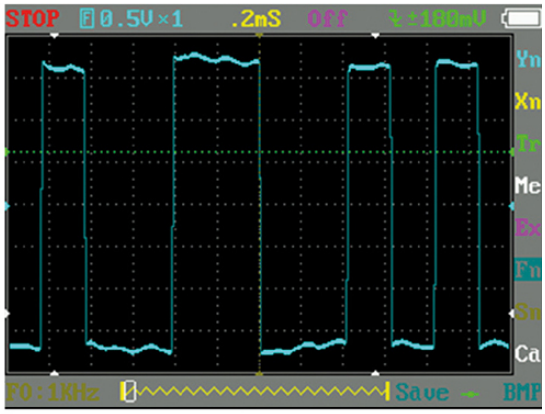
Сейчас в различное оборудование, которое работает с таймкодом, уже внедрены специальные алгоритмы для исправления самого искаженного сигнала, которые позволяют компенсировать многие ошибки при работе с линейным таймкодом. Но это всегда игра в лотерею. Как

только обстоятельства усугубляются факторами, которые мы не сможем спрогнозировать, наша система синхронизации может обрушиться. Поэтому лучше следовать технологиям и стандартам, чтобы иметь такую надежность системы, которая бы обеспечила нам безотказную работу в ста случаях из ста!

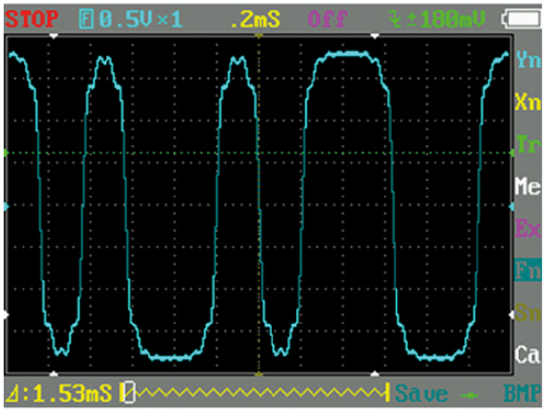
Так как LTC – это цифровой сигнал, передаваемый по аудиоинтерфейсу, то коммутируется он при помощи балансных аудиокабелей, но здесь появляются свои особенности с передачей цифрового сигнала по аудиоканалу.



Digital signal



Digital signal + noise



Distorted digital signal

Distorted low-level digital signal

График SMPTE не синусоидальный, как у аудиосигнала, а квадратный, т.к. сигнал кодируется в бинарной системе. На небольших расстояниях тип сигнала не принципиален. При большой длине аудио кабеля появляются такие факторы, как емкость кабеля и его индукция. Чем это плохо? А тем, что квадратная форма сигнала становится синусоидальной и смещается фаза восхождения сигнала. Это грозит тем, что сильно измененный LTC сигнал попросту не будет читаться принимающим устройством или будет нестабилен, и часть кадров будет теряться, а как следствие этого, синхронизация будет некорректная.

То же самое происходит, если LTC сигнал пропустить через звуковой пульт или аудио сплиттер. Многие специалисты, не до конца понимая принципиальное отличие звукового сигнала от LTC, наивно полагают, что с цифровым сигналом SMPTE можно работать точно так же, как и с обычным аналоговым сигналом, что является грубейшей ошибкой. Так как LTC – это импульсный сигнал с абсолютно другими характеристиками. К примеру, если подать с одинаковым уровнем LTC и аналоговый сигнал, то LTC будет звучать в два раза громче, и на звуковых пультах нормальный уровень LTC будет приниматься как перегруженный, что явля-

ется нормой, потому что это цифровой сигнал. В таких случаях «специалисты» понижают уровень генерируемого LTC сигнала, тем самым понижая качество помехоустойчивости линии.

Так как LTC – это цифровой сигнал, для него очень важен такой параметр, как громкость, так как громкость в аналоговом формате задает цифровую амплитуду между логическим нулем и единицей. Поэтому стоит очень внимательно следить за уровнем выходного сигнала LTC со звуковой карты. Рабочий уровень LTC от +4dBu до +8dBu (1.228v. – 1.95v.), если уровень сигнала опустить значительно ниже, то таймкод либо вообще не будет определяться принимающими устройствами, либо будет нестабилен. Часто по невнимательности на рабочем компьютере опускают общий рабочий уровень звуковой карты в системе, от этого падает не только сигнал аудиоканалов, но и LTC. Добавьте к этому генерацию со встроенной звуковой карты, аудио сплиттер или звуковой пульт, который делит LTC сигнал, и вы получите изрядно потрепанный LTC сигнал, который может доставить вам ряд неудобств при синхронизации.

Если обращаться к оригинальному телевизионным стандарту описывающий LTC, то там можно найти следующую информацию: «Предпочтительный диапазон выходного сигнала 1–2 вольт. Допускается диапазон амплитуд с размахом 0,5–4,5 вольт». Может возникнуть вопрос, почему же так сильно отличаются диапазоны в шоу индустрии и в телевидение? Диапазон 1.228–1.95 вольт, регламентирован звуковой индустрией и стандартом передачи балансного аудио сигнала, так как большинство оборудования и пультов управления используют звуковые модули на LTC портах, для того чтобы можно было программно анализировать входящие аналоговые искажения сигнала, для его последующего исправления. Что невозможно используя цифровые модули, где чётко регламентированы параметры цифрового сигнала. И так же, звуковые карты не способны генерировать LTC сигнал уровнем в 4.5 вольт. Профессиональное таймкод оборудование, которое работает с LTC сигналом, умеет работать, как и с телевизионным, так и с шоу индустриальным стандартом.

При работе с LTC допустимо использование профессиональных звуковых карт со встроенным процессором. В качестве примера можно привести звуковую карту MOTU 828x. На основе встроенного процессора можно организовывать маршруты между входами и выходами карты.

Я часто использую такие карты для разделения LTC сигнала на несколько клиентов, так как после настройки они полностью автономны, и для их работы не нужен компьютер. И к тому же у звуковых карт такого класса обычно очень большие динамические диапазоны работы входов и выходов, у MOTU 828x верхний динамический диапазон входов составляет +24dBu. Это значит, что такая звуковая карта с легкостью справится с перегруженным цифровым LTC сигналом. Также если нужно сгенерировать LTC сигнал для нескольких клиентов, то можно не делить один сигнал на несколько, а сразу с многоканальной звуковой карты генерировать необходимое количество LTC сигналов.

Преимущество такого способа в том, что в нашей рабочей схеме синхронизации уходит промежуточный элемент, что безусловно увеличивает надежность системы.

Теперь пару оговорок. Выше я упомянул о пагубном воздействии использования аналоговых аудио сплиттеров на сигнал LTC.

Но все же, как исключение, на рынке профессиональной техники существуют такие аудио сплиттеры, которые действительно не вредят сигналу LTC, но идентифицировать такие сплиттеры возможно только опытным путем с таймкод генератором и осциллографом. Но это больше редкость, чем правило, так как ни один производитель аудиооборудования не рассчитывает его характеристики для передачи квадратного сигнала LTC.

И следующая оговорка, для работы с LTC также возможно использование некоторых цифровых аудиопультов. Тут, конечно, может появиться некоторое замешательство, так как это опять противоречит вышесказанному про звуковые пульты.

Дело в том, что в новую цифровую эру все больше и больше звуковые аналоговые системы вытесняются цифровыми. Несмотря на то, что и аналоговый звуковой пульт, и цифровой выполняют одну и ту же задачу, работают они принципиально по-разному. В цифровых аудиопультах практически отсутствует аналоговый звуковой тракт, который вносит искажения в цифровой LTC сигнал. И также динамический рабочий диапазон аудиоканалов у цифровых консолей намного выше, чем у полностью аналоговых моделей, что также важно для LTC сигнала. Но выяснить, пригодна ли для работы та или иная модель цифрового пульта, можно только практически и опять при наличии осциллографа, который может показать состояние цифрового сигнала LTC, так как это широко известный факт, что существует ряд среднебюджетных моделей цифровых пультов, которые любят по своему «украшать» звук, что опять не допустимо для LTC.

Когда я работаю на проект, даже если я знаю, что пульт, который стоит у звукового отдела, подходит для работы с LTC, я все равно стараюсь избегать его использования. Потому что, к сожалению, я не могу застраховаться от кривых рук звукорежиссера, который может случайно накинуть на мой LTC канал какой-нибудь процессор обработки или переключиться на пресет, где мой LTC канал окажется закрытым. В этом случае возникает вопрос ответственности. Задача звукорежиссера – сделать так, чтобы звук в зале звучал корректно, это его главный приоритет, и если что-нибудь у него пойдет не так, то о LTC сигнале он будет думать в последний момент.

Поэтому я всегда настаиваю на том, чтобы в системе синхронизации участвовало только то оборудование, которое знаю я и моя команда. В этом случае ответственность за надежность системы будет только на мне.

Так как LTC сигнал может воспроизводиться как аудиодорожка (как мы помним, этот сигнал изначально записывался и воспроизводился с магнитной ленты), то этот LTC сигнал также зависим от скорости воспроизведения. Если скорость воспроизведения будет выше или ниже изначальной, при которой LTC был записан, то по итогу изменяется и скорость потока данных. Что в конечном итоге неизбежно влияет на качество синхронизации. Некоторые устройства могут компенсировать незначительные изменения в скорости воспроизведения, но если скорость LTC таймкода выходит за пределы допустимых границ, то либо устройство начинает терять кадры, либо определять некорректное время, что в любом случае приводит к потери синхронизации.

А теперь давайте вернемся к теме передачи сигнала на большие расстояния. А именно об условиях, которые нам могут позволить или не позволить передать LTC по балансной линии на расстоянии больше 50 метров.

Чтобы понять, о чем идет речь, давайте разберем передачу обыкновенного аудиосигнала на расстояние 200 метров. Рабочий уровень аудиосигнала обычно составляет 0dBu (усредненное значение), соответственно, при передаче сигнала по кабелю мы ожидаем на выходе линии получить такого же уровня сигнал. Но при передаче сигнала на 200 метров мы получим существенные потери уровня сигнала.

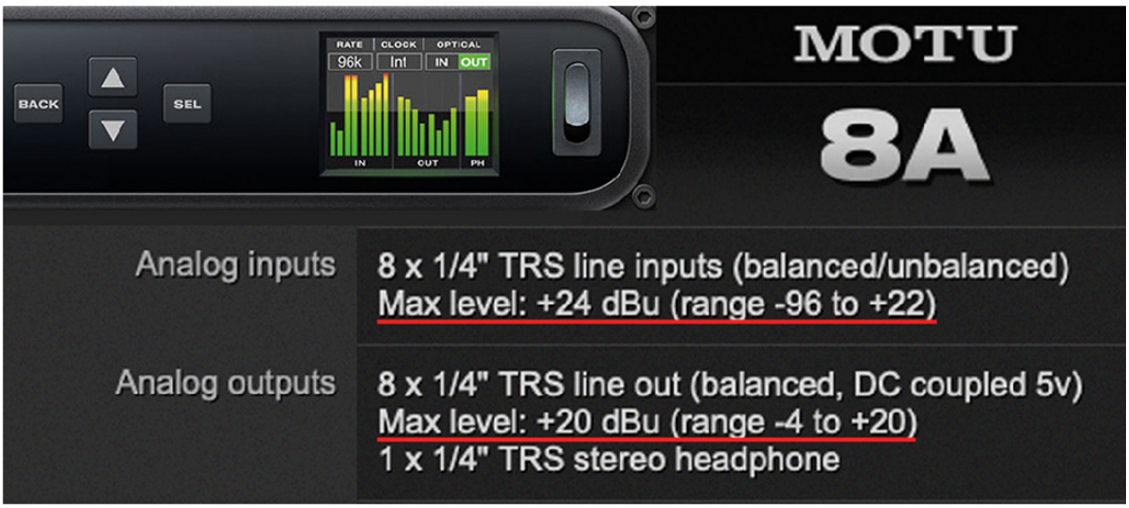
Чтобы восстановить аудиосигнал на конце линии, необходимо поднять громкость отправляемого сигнала до такого уровня, чтобы он смог компенсировать потери сигнала на расстоянии 200 метров. Но для этого нужно использовать оборудование, которое имеет хорошие показатели динамического диапазона аудиовыходов.

В случае если оборудование, которое отправляет сигнал, не имеет необходимых мощностей, то при попытке поднять уровень сигнала выше возможностей оборудования звуковой тракт такого устройства будет просто перегружен и, как следствие, сигнал будет искажен.

Или другой вариант, если невозможно поднять уровень сигнала в начале линии, то на конце линии используют устройство с высоким показателем чувствительности входов. Такие

устройства спокойно могут распознать сигнал с большими потерями и усилить его до необходимого уровня.

Чтобы понимать, о каких параметрах динамического диапазона и чувствительности идет речь, я приведу характеристики звуковой карты MOTU 8A, которая находится в профессиональном сегменте подобных карт, и чьи параметры являются довольно внушительными.



Но, к сожалению, не всегда заявленная информация совпадает с реальностью. В основном этим грешат аудиокарты бюджетного сегмента. Звуковые карты такого класса в основном получают питание по USB. Такие карты действительно могут иметь фантастические показатели динамических диапазонов на аналоговых выходах. Но у нее попросту может не хватить мощности питания, чтобы разогнать все свои выходы до заявленных показателей. Классический порт USB компьютера предоставляет всего 0,5 ампер. Напряжение питания по USB —5 вольт. Итого мы можем получить максимум 2,5 ватт мощности для всех портов звуковой карты.

А теперь сравним MOTU 8A, которая имеет внешний блок питания 15 вольт, с максимальной силой тока 1 ампер. Итого 15 ватт! Чувствуете разницу?

Никогда звуковая карта, которая работает только от USB, не сможет дать таких же показателей динамического диапазона, как профессиональные карты. Звуковые карты профессионального сегмента всегда имеют внешнее питание, что обеспечивает необходимый запас мощности для всех аналоговых выходов!

Хочу обратить внимание, что вышеуказанные вычисления грубы. Я не брал в расчет затраты на питание процессоров и другой электроники. Финальная разница расчетов настолько велика, что этим можно было пренебречь.

Вернемся к нашей теме передачи аудиосигнала по балансной линии 200 метров. Теперь давайте представим ту же самую ситуацию, но уже с сигналом LTC. К примеру, у нас есть сервер синхронизации, который предоставляет LTC и, скажем, устройство, которое должно принять LTC сигнал по балансной линии.

Ни один световой пульт, ни одна карта синхронизации конечных клиентов не имеют необходимых показателей чувствительности своих LTC портов, чтобы принять LTC через линию в 200 метров и более! И к тому же мы получим поверх ослабленного сигнала искажения квадратной формы LTC, так как кабель такой длины имеет внушительные показатели собственной индукции и емкости.

При таких условиях мы не можем передать LTC на 200 метров и уж тем более дальше!

У нас есть два пути исправить эту ситуацию.

Первый путь. Как и в случае со звуком, нам необходимо поднять уровень генерируемого сигнала эквивалентно потерям на линии. А для этого необходимо либо использовать профессиональные звуковые карты с хорошими показателями выходного динамического диапазона, либо таймкод генераторы/контроллеры, которые позволяют регулировать отдельно уровни сигналов на LTC портах.

И второй путь. Установить на конце линии либо звуковую карту с хорошими показателями чувствительности входных портов, чтобы усилить сигнал LTC до рабочего, либо установить на конце линии таймкод анализатор/решейпер, который сможет принять низкий уровень LTC, исправить все его искажения, поднять до рабочего уровня и отдать клиентам. Позже мы разберем функционал одного из таких устройств.

А вообще, я советую использовать одновременно два этих пути, тогда надежность системы будет намного выше.

Теперь мы выяснили, что просто так на 200 метров по балансной линии LTC не передать, LTC – не звук, и он требует больше внимания.

Итак, давайте подведем итог и еще раз определим те факторы, от которых зависит качество LTC сигнала:

• Использование для генерации и передачи сигнала небалансных звуковых карт и коммутации. Для обеспечения более устойчивого к помехам сигнала необходимо использование профессиональных балансных звуковых карт и LTC генераторов.

• Низкий уровень LTC сигнала. Цифровой LTC сигнал зависим от громкости воспроизведения или генерации. Нормальный диапазон LTC сигнала от +4dBu до +8dBu. И также при передаче на большие расстояния высокий уровень LTC обеспечивает лучшую помехоустойчивость.

• Использование для усиления и деления LTC сигнала звукового

• оборудования. А именно звуковых сплиттеров и звуковых микшерных пультов, которые впоследствии искажают квадратную форму сигнала в синусоидальную звуковую форму.

• Измененная скорость воспроизведения LTC аудиодорожки, которая отличается от оригинальной скорости, при которой таймкод был создан.

После этого напрашиваешься вопрос: если использование аудио-микшерных пультов и сплиттеров нежелательно, что же тогда применять для деления и усиления LTC сигнала?

На первый взгляд может показаться, что использование неспециализированного оборудования для LTC вызвано отсутствием соответствующих технических решений на рынке, но это не так. Так как LTC – один из старейших протоколов в направлении синхронизации, то производители также трудились над созданием специального оборудования для работы с этим интерфейсом.

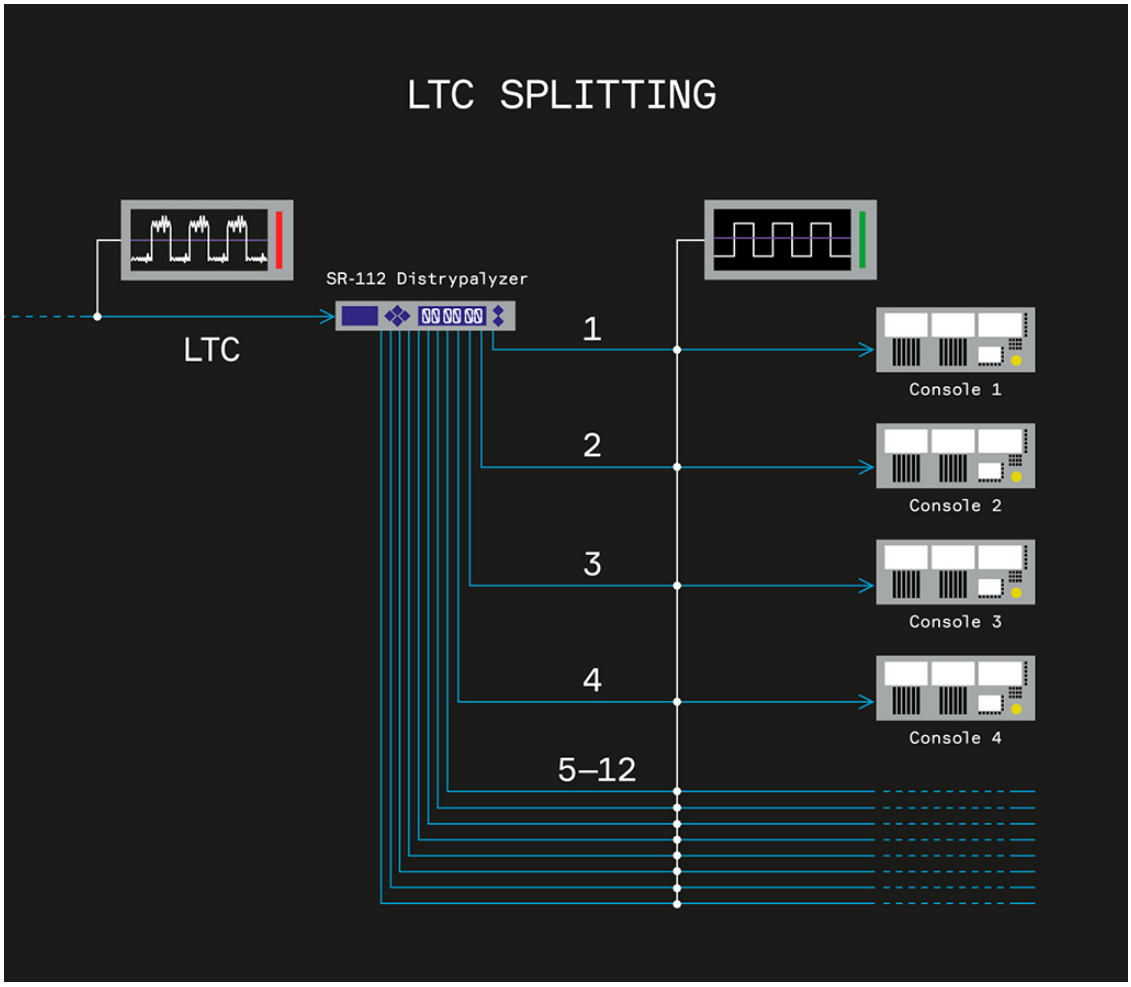
Ниже я не буду приводить список всех возможных моделей, которые существуют в нашей индустрии. Из каждого класса устройств мы разберем самого яркого представителя, который даст нам полное понимание об устройствах этого типа.

Brainstorm SR-112 Distripalyzer

Позвольте мне представить вам Brainstorm SR-112 Time Code Distripalyzer. Это многофункциональное устройство, которое работает только с LTC сигналом.

Первая его функция – сплиттер. Это устройство позволяет принимать один LTC сигнал через XLR-female и цифровым способом разделять его на 12 источников, из которых два XLR, два Jack 1/4(TRS) и остальные восемь передаются через разъем DB-25, распиновка (pin out configuration) точно такая же, как у производителей Tascam и Avid. В магазинах можно отдельно приобрести DB-25 male to 8 XLR male break-out кабель.

Также SR-112 позволяет индивидуально для каждого выхода настроить уровень сигнала, что очень полезно при передаче LTC на большие расстояния. Подняв уровень генерации, мы сможем компенсировать потери на линии.



Вторая его функция – восстановление формы сигнала (Reshaping Distorted TimeCode). Существует множество причин, которые могут исказить LTC сигнал, SR-112 позволяет исправить форму и уровень сигнала до родного квадратного. Также в настройках этого устройства можно задать параметры выходного LTC сигнала, тем самым создать максимальную совместимость между SR-112 и клиентами.

Третья функция – генерация таймкода. У SR-112 есть множество опций, как он может генерировать таймкод.

Он может начать генерацию собственного таймкода согласно параметрам входящего сигнала, но после этого он отвязывается от источника. И даже если скорость воспроизведения LTC изменится или фреймы начнут прыгать не по порядку, SR-112 все равно будет генерировать свой таймкод, до тех пор пока LTC не восстановится или не будет превышен интервал отсутствия таймкода.

Это всего лишь один из пяти возможных сценариев генерации и регенерации таймкода, которая возможна на SR-112. Более подробную информацию можно найти в мануале.

И четвертая полезная функция— анализ таймкода. На корпусе этого устройства есть Ethernet разъем, через который SR-112 можно подключить к сети. Через сеть возможно не только его управление, а также просмотр и экспорт отчетов работы SR-112, где можно увидеть по времени, когда таймкод пришел, какие ошибки были с ним, когда началась регенерация и многое, многое другое.

Rosendahl mif4

Также одно из самых часто используемых устройств при работе с таймкодом – Rosendahl mif4.

Это универсальное таймкод устройство, которое предназначено для исправления, анализа, конвертирования и генерирования таймкода через разные физические интерфейсы.

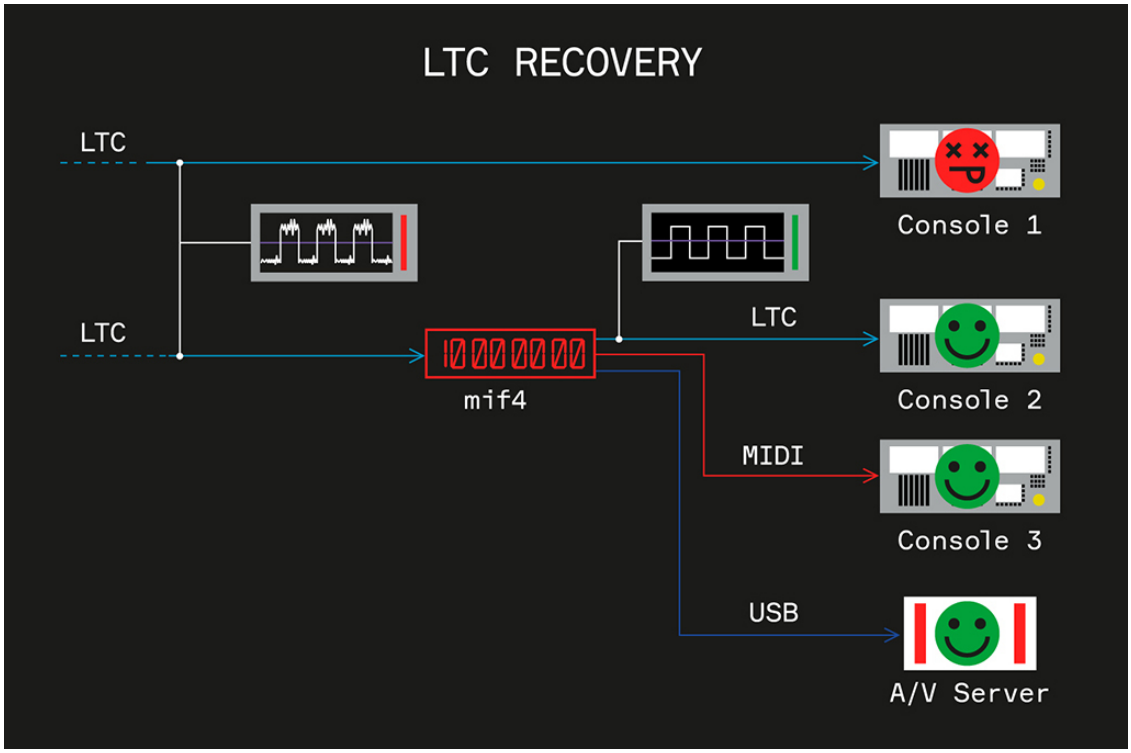
Давайте более подробно разберем его возможности и как его можно использовать при работе с синхронизацией.

Первая его особенность в том, что mif4 может принимать таймкод через MIDI, LTC, USB и SDI интерфейсы

При подключении к компьютеру через USB, mif4 определяется как простейшее MIDI устройство, которое можно использовать для программной генерации таймкода или его приема. Благодаря этому mif4 можно использовать как таймкод виджет для световых и видеопультов.

Вторая полезная функция – автоматическое конвертирование и генерация таймкода на выходные порты устройства. Мы можем через USB с компьютера генерировать таймкод, и он будет ретранслироваться на MIDI и LTC выходы. Так же это работает и с другими источниками. Если мы получаем таймкод на LTC вход, то mif4 ретранслирует его на USB, MIDI и LTC выход. Если подать таймкод на MIDI вход, устройство его продублирует на LTC, MIDI и USB.

Третье преимущество – возможность mif4 генерировать свой собственный таймкод на все выходы устройства. Параметры генерации можно задать вручную, через панель управления устройства или при помощи внешних MMC (MIDI Machine Control) сообщений.



Четвертая особенность – анализ таймкода и его скорости воспроизведения. Как мы уже разобрали выше, непостоянная и неоригинальная скорость воспроизведения также может влиять на качество генерации. mif4 показывает такие ошибки, что очень удобно при поиске отклонений в генерации таймкода.

Пятая функциональная особенность – возможность исправления искаженного сигнала. Если на устройство приходит слишком низкий уровень сигнала LTC или он искажен наведенными помехами, то клиент попросту не увидит таймкод. В таких случаях между принимающим устройством и линией LTC достаточно установить mif4, который способен прочитать самый плохой сигнал, исправить его и отдать клиенту нормальный LTC.

На схеме приведен простейший пример, как можно использовать mif4 для исправления сигнала и разделения его на несколько клиентов.

KISSBOX

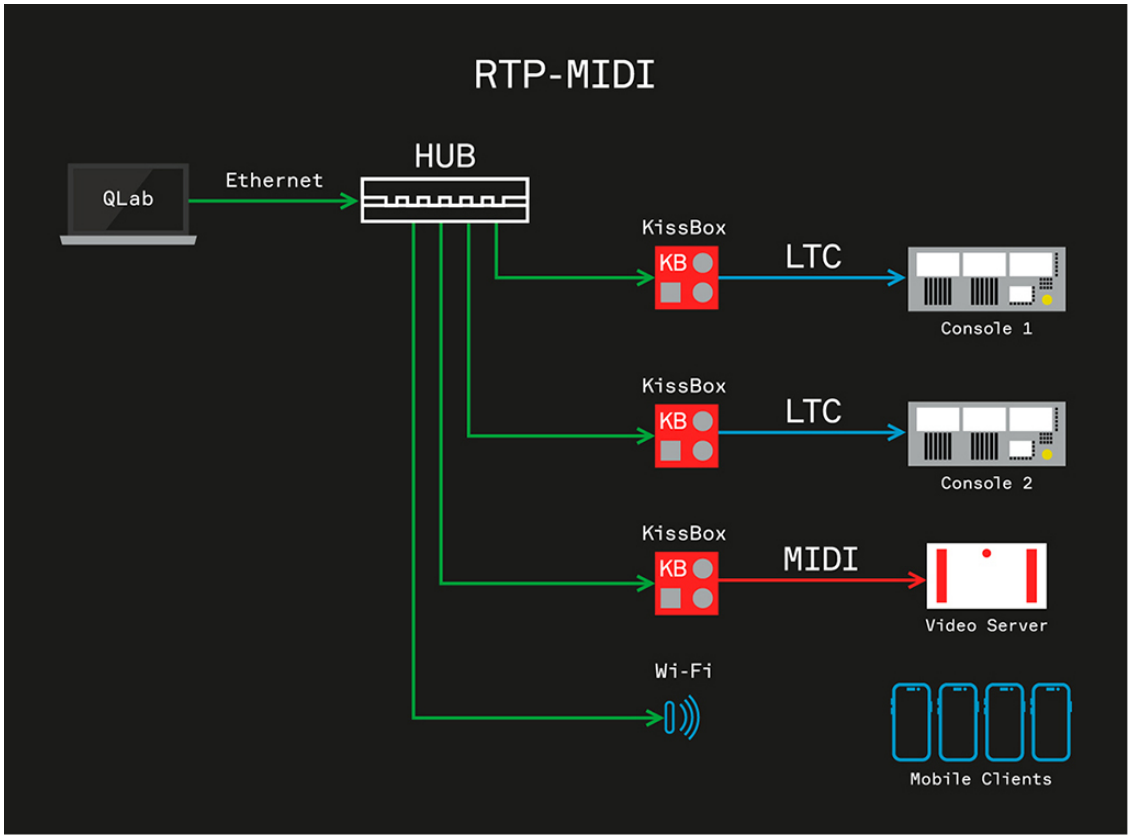
Порой приходится работать на проектах, где необходимо передать сигнал синхронизации на довольно большое расстояние, больше, чем позволяет балансная линия. Или мы хотим улучшить помехоустойчивость и надежность. В этом случае необходимо использовать дополнительное оборудование и технологии передачи данных.

Можно воспользоваться такими звуковыми протоколами передачи данных, как DANTE, MADI или AVB, но тогда для этого нам понадобится отдельный звукоинженер, который сможет поднять такую сеть и следить за ней во время проекта.

Или вы можете использовать специализированные технические решения, созданные для этих целей. К примеру, KISSBOX TC2TR. Кстати, KISSBOX была первой компанией, которая сделала поддержку протокола RTP-MIDI в своих технических решениях. KISSBOX TC2TR позволяет конвертировать LTC в MTC и передавать сигнал по протоколу RTP-MIDI через сеть.

KISSBOX TC2TR

В этом случае мы можем либо генерировать с шоу сервера таймкод через RTP-MIDI в сеть и принимать его на контроллер KISSBOX, либо вообще использовать два устройства KISSBOX, чтобы передать или принять LTC. В этом случае, в зависимости от того, используем ли мы Ethernet кабель, оптоволокно или Wi-Fi, мы можем передать наш LTC сигнал на расстояние более чем 200 километров.



Используя протокол RTP-MIDI, мы можем транслировать протоколы синхронизации нескольким клиентам. Это преимущество также применимо и к оборудованию KISSBOX. Имея один источник, мы можем доставить LTC сигнал нескольким клиентам сразу.

Передача и прием MIDI

В отличие от LTC, MIDI – это полностью цифровой сигнал, который передается и принимается через цифровые интерфейсы и порты. Формат сообщений, технология передачи MIDI сигнала, электрические параметры – все это регламентировано в открытых документах рекомендованной практики при работе с MIDI, которая в разные годы была утверждена ассоциациями MMA и AMEI. Конечно же, это функциональный плюс, который уменьшает шансы на ошибку, но тем не менее, как и любой протокол, этот стандарт любит, когда с ним работают правильно.

MIDI был разработан как серийный интерфейс передачи данных. За основу MIDI был взят стандарт серийных интерфейсов UART (Universal Asynchronous Receiver-Transmitter). Самый близкий собрат MIDI – последовательный интерфейс RS-232. Очень похожая история с DMX512, который базируется на стандарте RS-485, используемый для управления промышленными контроллерами. MIDI интерфейс для подключения использует разъем DIN 41524 (DIN 5-pin 180°). Все устройства используют MIDI разъем «мама». А MIDI кабели с двух сторон используют коннекторы «папа». MIDI интерфейс имеет только один однонаправленный канал связи. По этой причине каждый MIDI порт имеет свое направление, IN или OUT.

Теперь давайте поговорим об особенностях MIDI интерфейса, которые накладывают определенные условия при работе с ним. Первая особенность MIDI в том, что этот интерфейс не рассчитан на передачу данных на большие расстояния. Он был разработан с учетом коммутации оборудования в пределах звуковой студии. Максимальная длина кабеля, при которой падения сигнала будут незначительны, от 10 до 15 метров. Но со временем MIDI вышел за пределы звуковых студий, и возникла потребность в больший расстояниях. Производители начали выпускать различное оборудование для увеличения дистанций использования MIDI. В основном это были MIDI усилители, которые ставились в разрыв линии, тем самым продлевая дальность MIDI еще на 15 метров. К вопросу передачи MIDI сигнала на большие расстояния мы вернемся чуть позже.

С развитием MIDI индустрии потребности в работе с этим интерфейсом постоянно возрастали, и как следствие того, развивался и рынок оборудования, который предлагал разные возможности в работе с MIDI.

Одна из самых распространенных потребностей – это подключение нескольких управляемых устройств к одному ведомому. Для решения этой задачи есть несколько вариантов. Давайте разберем подробно каждый из них.

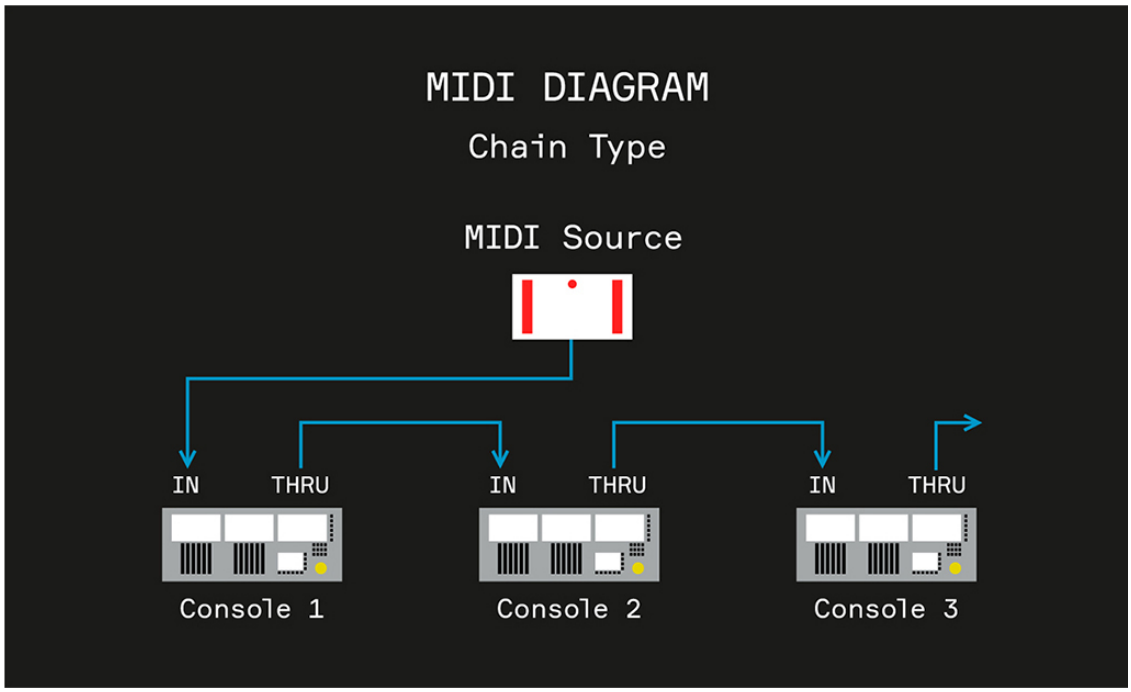
Первый и самый простой способ – подключить все устройства последовательно. В большинстве случаев оборудование, которое поддерживает работу с MIDI, имеет три порта MIDI: IN, OUT и THRU.

• IN – Порт для подключения входного сигнала. Используется устройством для приема MIDI сообщений от других устройств.

• OUT – Порт для подключения выходного сигнала. Используется устройством для передачи генерируемых MIDI сообщений другим периферийным MIDI устройствам.

• THRU – Порт, который логически дублирует данные, приходящие на порт IN.

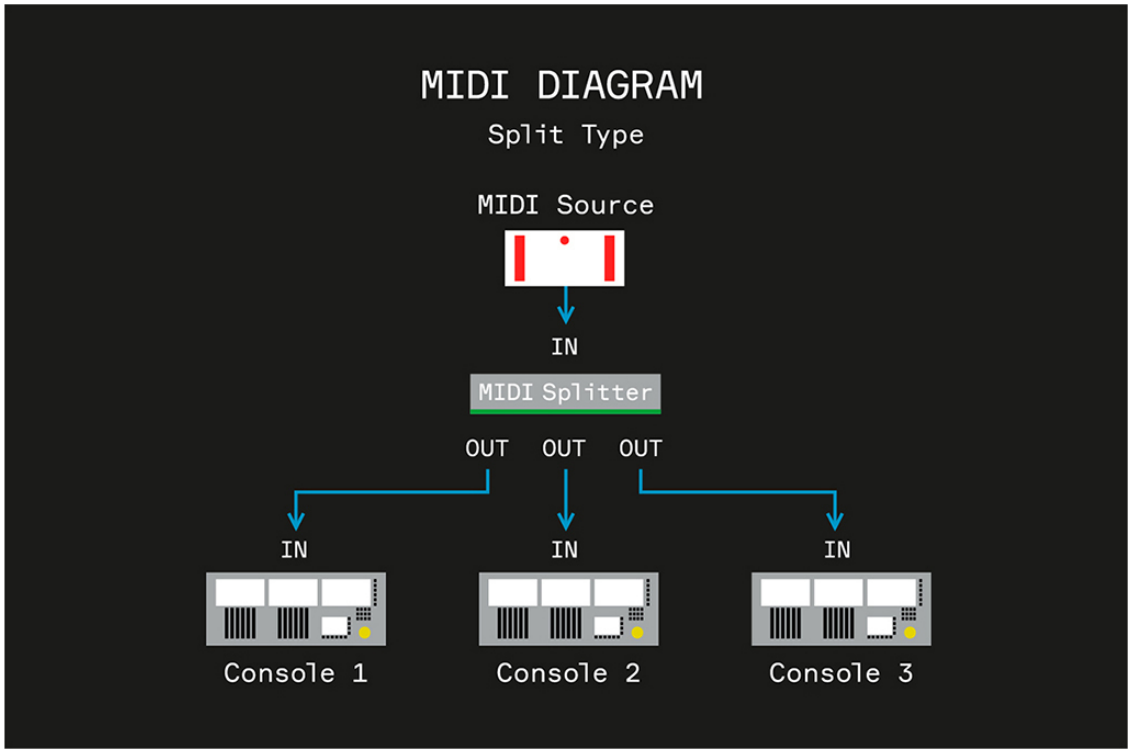
Для подключения последовательно нескольких MIDI устройств к одному мастер устройству необходимо использовать порты IN и THRU, соединяя устройства по типу последовательной цепи. Одно из главных преимуществ такого подключения в том, что для реализации этого метода нам не нужно дополнительное оборудование.



Недостаток такого решения заключается в том, что, как и в варианте любого последовательного подключения, при обрыве линии между консолями все последующие потребители в цепи также теряют сигнал.

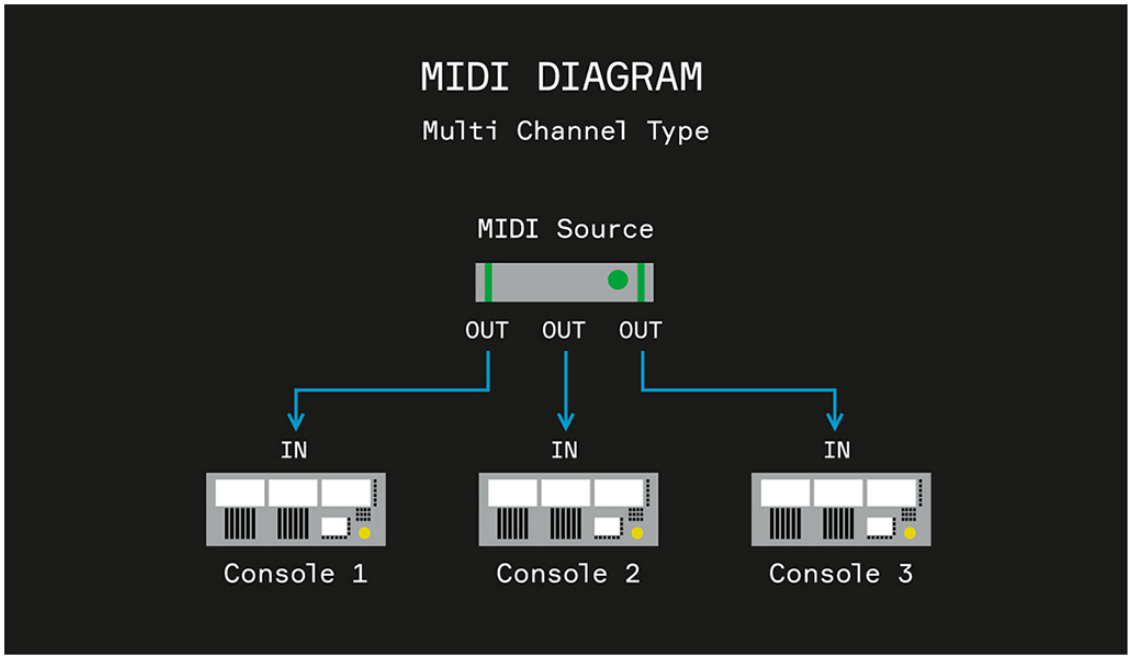
Второй способ— разделить MIDI при помощи специального MIDI сплиттера.

Преимущество такого варианта в том, что каждый потребитель подключается независимо друг от друга, а это значит, что при выходе из строя одного из потребителей это никак не повлияет на прием MIDI сигнала другими клиентами.



Недостаток такого подключения лишь один: при отключении питания у сплиттера мы теряем подключение со всеми потребителями.

И третий вариант, самый функциональный, это использовать для генерирования MIDI сигнала многоканальные MIDI карты. Помимо преимуществ второго варианта, добавляется также возможность генерирования независимых MIDI сообщений для каждого клиента. Насколько это полезно? Представим, что у вас есть видео и световой пульт, и в один момент нужно синхронно запустить видео контент и световую сцену. У нас есть два разных MIDI сообщения, которые на двух пультах запускают необходимые сцены.



При наличии многоканальной MIDI карты возможно отправить две разные команды на разные порты. Достаточно для каждого сообщения назначить свой MIDI порт многоканальной карты. В итоге мы сможем вызвать световую и видеосцену синхронно, используя разные команды.

Единственный недостаток такого способа заключается в том, что стоимость таких MIDI карт выше в сравнении со стандартными одноканальными картами или MIDI сплиттерами.

В качестве примера многоканальной MIDI карты можно привести MOTU MIDI Express XT. Эта карта предоставляет восемь независимых MIDI входов и восемь MIDI выходов. Помимо этого, эта карта имеет собственный процессор, который позволяет без участия компьютера настраивать внутренние маршруты между MIDI портами. К примеру, эту карту можно сконфигурировать как MIDI сплиттер или как мэрджер MIDI сигналов. И также дополнительно у этой карты есть LTC вход и выход, который позволяет генерировать и принимать линейный таймкод SMPTE.

Также для решения вопросов передачи сигнала MIDI на большие расстояния и деления его на несколько клиентов мы можем использовать технологию RTP-MIDI.

В предыдущей главе мы уже затронули устройство KISSBOX TC2TR, который, используя протокол RTP-MIDI, может передавать не только LTC сигнал, но и классический MIDI по Ethernet. И опять, дальность передачи MIDI этим способом будет напрямую зависеть от того, какой канал связи вы будете использовать: оптику, витую пару или WiFi. Кстати, если вам нужно передавать через RTP-MIDI только MIDI без использования LTC, то у KISSBOX есть устройства, которые передают только MIDI. KISSBOX MIDI2TR имеет на своем борту четыре MIDI порта, два входящих и два исходящих. А KISSBOX CM-MIDI2, кроме миди портов, еще имеет возможность подключения к компьютеру через USB.

KISSBOX MIDI2TR

Передача и прием Ethernet

Так как сейчас технология Ethernet является одной из самых распространенных для построения локальных и глобальных сетей, в том числе и в шоу-индустрии, предлагаю затронуть основы и особенности оборудования и разные способы передачи сигнала. На самом деле, если подойти более серьезно к этому вопросу, то можно найти достаточное количество хорошей литературы, где более подробно описано то, о чем мы сейчас с вами будем говорить. Но тем не менее, тех знаний, которые вы получите сейчас, вполне хватит, чтобы ориентироваться в этой теме и понимать, какие существуют способы создания сети по технологии Ethernet и в чем особенность каждого из них.

Витая пара

Один из самых доступных и распространенных способов соединить несколько участников в одну сеть— использовать стандартную витую пару. Классическая витая пара имеет восемь проводов, которые скручены попарно в четыре пары. Отсюда и название этого кабеля. Способ передачи сигнала по витой паре схож со способом передачи в балансной аудиолинии. Что-бы помехи на каждую пару проводов наводились максимально одинаково, каждая пара скручена вместе. В этом случае вычитание помех будет максимально эффективно. В зависимости от скорости передачи данных и режима работы сетевой карты используется определенное количество пар. К примеру, для передачи данных на скорости 100 Mbps достаточно две пары, а вот начиная со скорости 1 Gbps необходимо использовать уже все четыре пары. Как вы уже успели заметить, витая пара использует стандартный разъем 8P8C (RJ45). Итак, ничего сложного, идем дальше.

Конец ознакомительного фрагмента.