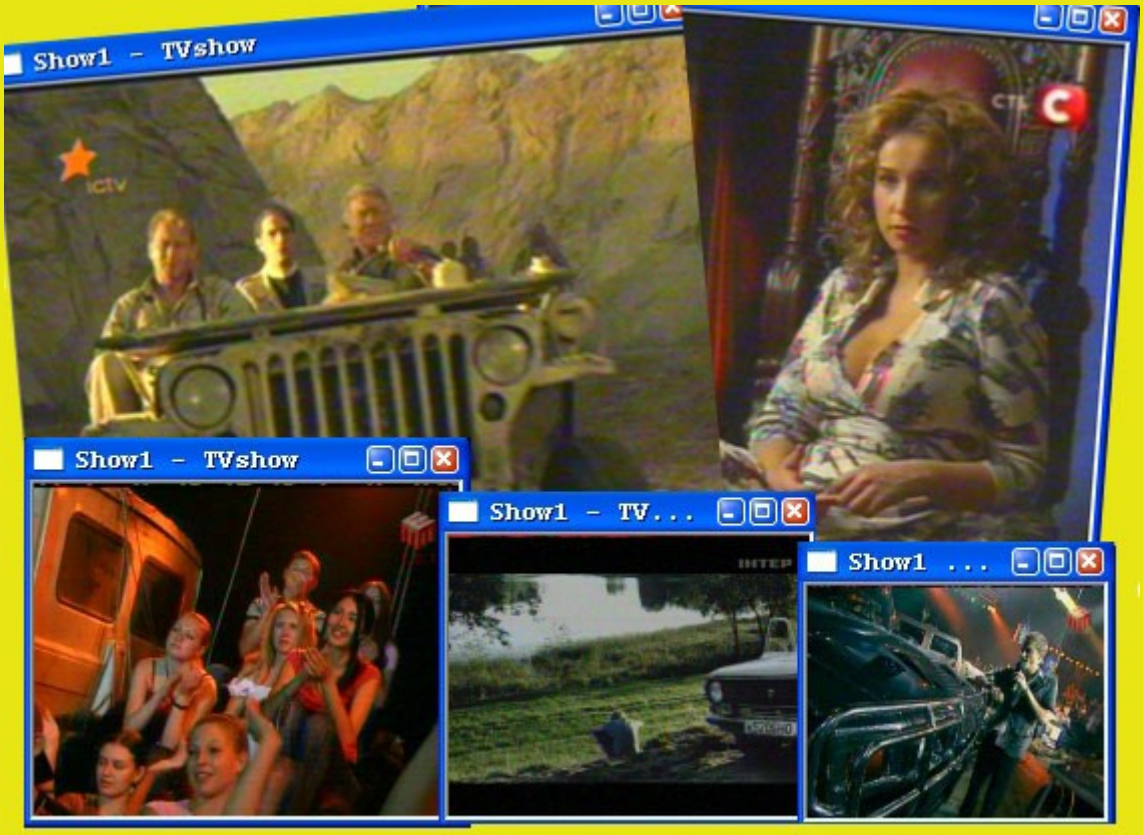
Александр Левчук ©

DirectShow и телевидение



2008

Предисловие

Книга является попыткой обобщить и описать процесс решения реальной задачи, которая часто встречается в практике прикладного программиста. Многим разработчикам неоднократно приходилось выполнять задания в условиях недостаточных знаний и конкретного опыта работы в новой области. Каждый раз в такой ситуации приходится быстро изучать малоизвестные технологии, API, SDK, перечитывать горы англоязычной документации, поскольку книги на русском языке по интересующему вопросу часто недоступны или вообще отсутствуют.

Цель книги предоставить базовые знания и практические приемы программирования приложений DirectShow. На примере создания приложения DirectShow для телевизионной карты PCI Xpert TV-PVR, рассматривается технология DirectShow и ее применение. Карта PCI Xpert TV-PVR представляет собой многофункциональное устройство, что позволяет рассмотреть практически все аспекты и возможности DirectShow по реализации приложений для телевизионных устройств. Рассмотрение телевизионной технологии DirectShow ограничивается рамками используемого устройства, то есть аналоговым телевидением, однако это ограничение не существенно для первоначального изучения, поскольку овладение технологией DirectShow существенно облегчает самостоятельное изучение остальных аспектов технологии, используя документацию DirectX.

Книга будет полезна студентам и опытным программистам. Студентам и начинающим программистам книга демонстрирует весь процесс разработки приложения DirectShow на языке C++, который включает различные средства программирования:

- интерфейс проектируется с применением библиотеки MFC и ATL
- Создание приложения DirectShow иллюстрирует основные приемы применения технологии COM
- обработка данных производится с применением алгоритмов стандартной библиотеки STL
- управление звуковым системным микшером проектируется с использованием низкоуровневой библиотеки winmm.lib
- для работы с системным регистром, привлекаются средства ATL и DirectShow

Опытным программистам, не работавшим ранее с DirectShow, книга поможет быстро освоить приемы программирования приложений DirectShow. В книге описываются все встретившиеся на пути проблемы и

способы их решения. Программирование в среде DirectShow ведется на уровне интерфейсов COM, без использования базовых классов DirectShow. Так достигается лучшее понимание функционирования DirectShow, базовые классы требуют определенных начальных знаний технологии DirectShow.

Книга имеет максимально возможную практическую направленность и сопровождается большим объемом исходных кодов.

Глава 1
Общие принципы построения и использования DirectShow

Предисловие.....1

Глава 1
Общие принципы построения и использования DirectShow.....2

Телевизионная технология Microsoft.....4

.....4

Архитектура DirectShow.....5

Основные свойства фильтров и фильтрового графа.....8
Критерий предпочтения фильтра.....9
Контакты фильтров и типы медиа данных.....10
Носители.....11
Передача данных и медиа выборки 11
Состояния фильтров и фильтрового графа.....12
Синхронизация фильтров.....14
Использование объектов DMO в DirectShow.....16

Соединение фильтров.....17
Согласование медиа типов при соединении фильтров.....18

Согласование транспортировки данных.....19
"Интеллектуальное" соединение фильтров.....21

Представление аппаратных устройств в фильтровом графе.....24

Потоки данных 26
Отображение WDM устройств в локальной системе 29

Сбор видео и звуковой информации.....31
Фильтр сбора видео информации(Capture).....31

Фильтр сбора звука.....33

GraphEdit инструмент визуального построения графа и отладки.....33

Телевизионная технология Microsoft

DirectShow является архитектурой, обеспечивающей запись и воспроизведение мультимедийных потоков данных от различных источников, и входит в состав DirectX 8.1 и DirectX 9.0 SDK. В то же время DirectShow является объединяющей частью, поскольку использует другие компоненты DirectX, такие как DirectDraw, Direct3D, DirectSound для решения основной задачи записи и воспроизведения мультимедийных потоков. DirectShow также частично использует и технологию Windows Media для сжатия медиа потоков.

DirectShow позволяет создавать широкий спектр видео и звуковых приложений, в том числе для телевизионных карт аналогового и цифрового телевидения (DV). Телевизионные устройства должны поддерживаться драйверами WDM (Windows Driver Model) или драйверами широковещательной передачи (Microsoft Broadcast Driver Architecture), сокращенно BDA.

Телевизионная технология Microsoft являясь составной частью DirectShow, не выделена в явном виде в отдельный подраздел и тесно связана с DirectShow общим подходом к решению задач записи и воспроизведения потоков мультимедийных данных. В DirectShow телевизионный источник информации является равноправным среди других источников и подчиняется общим принципам построения приложений DirectShow. DirectShow позволяет быстро создавать TV приложения на платформах Microsoft Windows. Данная технология реализована на основе модели COM и содержит компоненты, предоставляемые Microsoft, а также спецификации для дополнительных компонентов создаваемых сторонними производителями.

Телевизионная технология Microsoft поддерживает все аналоговые и цифровые стандарты, включая ATSC, DVB, NTSC, PAL, SECAM и другие. Также полностью поддерживаются дополнительные телевизионные сервисы, например телетекст. Приложения, реализованные на основе технологии Microsoft, полностью освобождены от необходимости учета специфических деталей связанных с особенностями распространения телевизионного сигнала. Это означает, что приложение может использовать любую сеть распространения сигнала аналоговую, кабельную, цифровую (ATSC, DVB-S, DVB-C, DVB-T) а также любой другой тип сети, если ее производитель поддерживает технологию Microsoft.

Архитектура DirectShow

DirectShow имеет модульную архитектуру, основой которой является отдельный программный модуль, называемый фильтром. Фильтры могут соединяться между собой для выполнения последовательной и параллельной обработки потоков данных. Совокупность соединенных фильтров, выполняющих определенную задачу, называется фильтровым графом. Например, запись телевизионной передачи на диск выполняет фильтровый граф определенной конфигурации, созданный именно для выполнения этой задачи. Он содержит несколько фильтров DirectShow соединенных между собой. Приложение, написанное с применением DirectShow, представляет собой две части, первая часть это само приложение, которое обеспечивает интерфейс с пользователем, а также реализует построение фильтрового графа и управление им. Второй частью приложения является фильтровый граф, который взаимодействует с аппаратурой и каналами передачи данных.

DirectShow реализован на основе COM и любой компонент DirectShow является объектом COM. Все компоненты DirectShow можно разделить на три группы.

- Компоненты, обеспечивающие создание и функционирование фильтрового графа. Компоненты этой группы объединяются под общим названием менеджер фильтрового графа.

- Фильтры, осуществляющие обработку потоков данных, их передачу преобразование и отображение.

- Дополнительные служебные компоненты, обеспечивающие функционирование фильтрового графа.

Таким образом, фильтровый граф представляет собой совокупность фильтров соединенных определенным образом и управляемый в процессе работы менеджером фильтрового графа. Менеджер фильтрового графа также участвует в первоначальном построении графа, и в последующей его перестройке. Эти задачи во многом автоматизированы, но менеджер графа допускает и ручное построение или перестройку графа. Некоторые задачи невозможно решить в режиме автоматического построения и требуется ручное построение участков графа или даже полностью всего графа. После построения графа приложение управляет им посредством передачи команд. Реакцию графа отображают события, которые он может посылать приложению. На рисунке 1 показано взаимодействие приложения и DirectShow.

DirectShow содержит компоненты, которые обеспечивают создание графа и его функционирование. Каждый компонент имеет соответствующий идентификатор CLSID.

К компонентам создающим граф относятся.

- Построитель графа сбора (Capture Graph Builder, CLSID_CaptureGraphBuilder2). Графом сбора (capture) называется граф, производящий сбор звуковой и видео информации для записи и отображения.
- Построитель графа DVD (DVD Graph Builder, CLSID_DvdGraphBuilder). Это вспомогательный объект, обеспечивающий воспроизведение DVD.

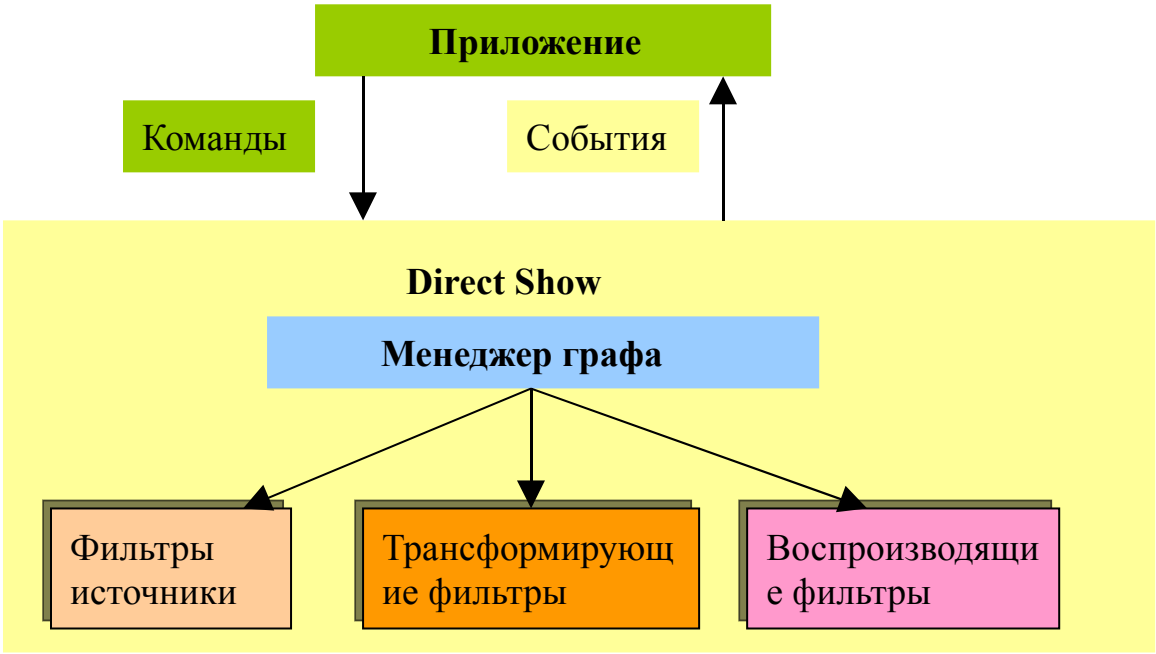


Рис. 1. 1. Взаимодействие приложения и DirectShow.

К компонентам, управляющим функционированием графа, относится менеджер фильтрового графа (Filter Graph Manager, CLSID_FilterGraph). Это центральный элемент фильтрового графа. Менеджер фильтрового графа осуществляет взаимодействие приложения с графом. Приложение передает в фильтровый граф управляющие сигналы и принимает события, а все управление работой и потоками информации выполняет менеджер фильтрового графа. Он предоставляет методы для соединения фильтров, их запуска и остановки, как в целом, так и по частям. Более детальное управление данны-

ми или поведением фильтра, может быть предоставлено только фильтром посредством своих COM интерфейсов. Менеджер графа имеет два идентификатора, один для рабочего потока процесса, в котором существуют все компоненты графа CLSID_FilterGraph, а второй для многопоточного приложения CLSID_FilterGraphNoThread. Для второго случая, поток, создающий граф должен иметь цикл сообщений, кроме того, перед завершением потока он должен освободить все объекты графа.

Дополнительные компоненты обеспечивают функционирование фильтрового графа. К ним относятся:

- Системный перечислитель устройств и фильтров, зарегистрированных в системе (System Device Enumerator, CLSID_SystemDeviceEnum).
- Карта фильтров (Filter Mapper, CLSID_FilterMapper2). Служит для поиска фильтров зарегистрированных в регистре по определенным критериям отбора. Используется менеджером графа и может использоваться приложением.
- Синхронизатор графа (System Reference Clock, CLSID_SystemClock) или ссылочное время. Объект используется менеджером графа для общей синхронизации фильтров и получения отсчетов времени. Менеджер графа использует в качестве источника ссылочного времени любой фильтр в графе, который способен предоставить такой источник. В случае его отсутствия данный объект должен быть создан приложением. Фильтровый граф может функционировать и без источника ссылочного времени.
- Вспомогательный компонент для получения информации о свойствах файлов AVI (Media Property Bag, CLSID_MediaPropertyBag).
- Распределитель памяти для буферов медиа данных (Memory Allocator, CLSID_MemoryAllocator). Используется фильтрами DirectShow.
- Вспомогательный компонент для поиска фильтров с одним входным контактом (SeekingPassThru). Данный объект не должен использоваться приложением

Основные свойства фильтров и фильтрового графа

Фильтровый граф строится по определенным правилам с учетом свойств фильтров, принципов соединения фильтров и принципов передачи потоков данных между фильтрами. Каждый фильтр в графе выполняет определенную операцию с мультимедийным потоком, например:

- Управление устройствами аналогового телеприемника.
- Прием видеопотока из телевизионной карты.
- Чтение мультимедийного файла.
- Передача видеопотока в видеокарту для отображения.
- Передачу звукового потока в звуковую карту для прослушивания и другие задачи.

Существует три типа фильтров:

- Фильтр источник информации (Source Filters). Например, файл AVI на диске представляется как фильтр источник сырых данных. Звуковая карта также является фильтром источником. Любой источник информации для фильтрового графа должен быть представлен своим фильтром или несколькими фильтрами.
- Трансформирующие фильтры (Transform Filters). К этим фильтрам относятся те фильтры, которые каким либо способом преобразуют поток данных. Например, кодируют или декодируют данные или разделяют общий поток на звуковой поток и видеопоток.
- Отображающие фильтры (Renderer Filters). Эти фильтры представляют информацию потребителю, они преобразуют цифровые потоки в зрительную информацию, отображаемую на экране и в аналоговый звук, передаваемый на динамики. Также в эту категорию отнесены фильтры, производящие запись информации на твердый носитель. Такие фильтры называются фильтрами записи (file-writer).

Для выполнения специфической задачи обработки мультимедийного потока может быть разработан пользовательский фильтр с учетом соглашений налагаемых DirectShow.

Каждый фильтр может иметь несколько входов и выходов. Для работы в составе фильтрового графа они соединяются последовательно в разветвленные цепочки. Например, для проигрывания видео файла создается следую-

щая последовательность фильтров, которая содержит фильтры всех трех типов (рис 1.2).

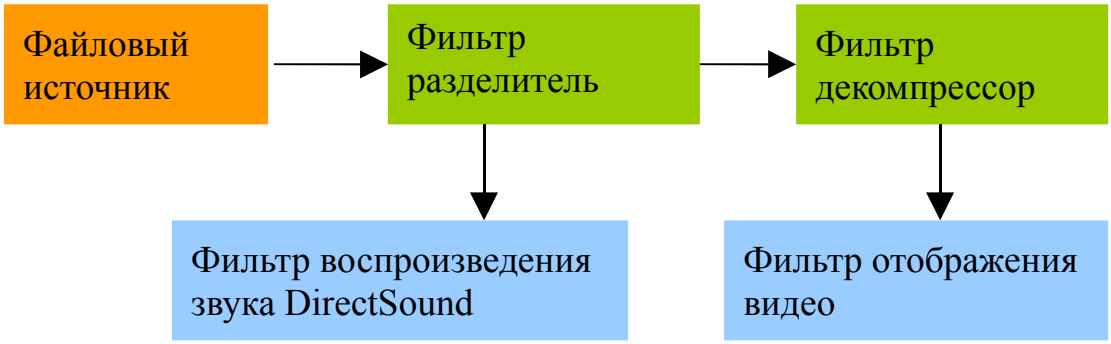


Рис. 1. 2. Пример фильтрового графа.

Критерий предпочтения фильтра

При построении фильтрового графа в автоматизированном режиме, когда выбор фильтров и их установку производит менеджер графа используется критерии предпочтения (Merit). Он позволяет менеджеру графа выбрать из нескольких фильтров одинаковых по назначению наиболее подходящий. При установке фильтров в граф в ручном режиме он не используется и может служить просто для сведения программиста.

Все фильтры в системе регистрируются с критерием предпочтения. При установке фильтра менеджер графа осуществляет перечисление всех фильтров с корректным медиа типом и их отбор в соответствии с критерием предпочтения, начиная с наивысшего значения. Для фильтров имеющих одинаковый критерий предпочтения используются дополнительные критерии отбора.

Константы критерия предпочтения имеют такие значения.

MERIT_PREFERRED	= 0x800000
MERIT_NORMAL	= 0x600000
MERIT_UNLIKELY	= 0x400000
MERIT_DO_NOT_USE	= 0x200000
MERIT_SW_COMPRESSOR	= 0x100000
MERIT_HW_COMPRESSOR	= 0x100050

Фильтры, которые не используются для стандартного режима воспроизведения видео, имеют значение критерия равное MERIT_DO_NOT_USE. Ме-

неджер графа не проверяет такие фильтры. Каждый критерий может иметь промежуточное значение. Например, MERIT_NORMAL + 1.

Контакты фильтров и типы медиа данных

Входы и выходы фильтров именуются контактами (Pin). Контакт является COM объектом и выполняет определенную низкоуровневую обработку потока данных необходимую для соединения фильтров. Контакты подразделяются на входные и выходные. Контакт не является отдельным COM объектом, он не может быть создан вне фильтра, которому принадлежит.

Каждый контакт характеризуется направлением, и принадлежностью к определенному фильтру. Объект контакта содержит медиа тип потока (формат данных), который может быть передан через него. Контакт может поддерживать несколько различных медиа типов потока. Выходной контакт одного фильтра может определять возможность или невозможность подключения к входному контакту другого фильтра. Медиа тип потока данных определяет вид данных и их физическое строение. При прохождении потока данных последовательно от входного контакта первого фильтра до входного контакта последнего фильтра, данные могут преобразовываться из одного формата в другой с помощью трансформирующих фильтров. Число возможных форматов данных в DirectShow ограничено и стандартизовано. В тоже время множество форматов данных не ограничивается константой и может расширяться.

При соединении двух фильтров медиа типы данных на соединяемых контактах должны быть одинаковыми. Если медиа типы различны, то соединение фильтров невозможно. При реализации стандартных задач приложение не нуждается в прямой обработке медиа типов. Менеджер графа выполняет все построение графа автоматически. При ручном построении графа или при совмещении автоматического построения с ручным построением может потребоваться дополнительная обработка медиа типов передаваемых потоков данных.

При обработке медиа типов используется структура AM_MEDIA_TYPE. Она полностью описывает медиа тип потока данных и содержит следующую информацию:

- Главный медиа тип (major type). Он определяется идентификатором (GUID) для основных видов данных. Например, это могут быть звуковые данные или видеоданные, текст и так далее.

- Подтип данных (subtype) определяется идентификатором (GUID) для более точного определения формата внутри главного типа. Например, в

группе видео данных производится деление на подтипы RGB-24, RGB-32, UYVY и так далее. В группе звуковых данных различают подтипы PCM, MPEG-1 и другие.

- Блок формата (format block) определяет детали медиа типа. Блок формата сам является структурой, на которую указывает член pbFormat структуры AM_MEDIA_TYPE. Тип структуры блока формата определяется членом formattype (GUID). Например, для звука это структура WAVEFORMATEX, а для видео VIDEOINFOHEADER или VIDEOINFOHEADER2.

Если блок формата заполнен, главный тип и подтип являются избыточной информацией. Главный тип и подтип удобно использовать для генерации формата данных, не заполняя больших структур блока формата вручную. Для этого достаточно указать главный тип и подтип, а остальные члены заполнить нулями. При соединении фильтров все члены структуры будут установлены автоматически.

Носители

В DirectShow термин носитель или среда (medium) определяет тип связи двух устройств в системе для передачи данных на главном компьютере. Другими словами носитель определяет физический тип шины, соединяющий устройства. Примером таких устройств может быть ТВ приемник и переключатель сигналов, связанные шиной на одной карте. Или ТВ приемник в исполнении внешнего устройства, связанный с компьютером шиной USB. Это также могут быть два устройства на разных картах, связанные шиной PCI. Фильтры режима ядра, обеспечивающие работу устройств, используют тип носителя вместо медиа типа при соединении контактов. Приложению редко требуется анализ среды, только в специфических случаях. Среда описывается структурой REGPINMEDIUM.

Передача данных и медиа выборки

В фильтровом графе передача данных осуществляется всегда в одном направлении, от выходного контакта одного фильтра на входной контакт другого фильтра до поступления потока через цепочку фильтров на конечный воспроизводящий (rendering) фильтр. Медиа тип передаваемых данных между фильтрами определяется реализацией выходного контакта фильтра. Это направление потока данных называется "вниз по потоку" (downstream). Обратная передача (upstream) данных по принципу обратной связи в фильтровом графе не допускается. Попытка соединения фильтров подобным образом приводит к зацикливанию графа и не допускается менеджером графа. В тоже

время графический менеджер допускает просмотр графа с целью его анализа как вниз по потоку, так и в обратном направлении.

Передаваемые данные организуются в медиа выборки (Media Samples).Преобразование простого (сырого) потока данных, поступивших на вход графа в медиа выборки, может выполнять как фильтр источник, так и отдельный фильтр. Медиа выборка это фрагмент потока данных имеющий законченный вид. Медиа выборка в DirectShow это объект COM, который реализует интерфейс IMediaSample2. Для видеоданных медиа выборка представляет один кадр изображения, для звука это определенная совокупность элементарных звуковых выборок.

Процесс передачи медиа выборки от выходного контакта одного фильтра на входной контакт другого фильтра называется транспортом (transport). Транспорт может осуществляться различными способами. Обычно он производится с использованием буферов размещаемых в локальной памяти, тогда он называется транспорт локальной памяти, но может осуществляться непосредственно с контакта на контакт, если обработка потока данных производится устройствами, которые имеют собственную память.

Созданием медиа выборок и выделением памяти для обеспечения транспорта занимается другой COM объект, называемый распределителем памяти. Во время соединения контактов распределитель памяти резервирует буфер для медиа выборки, создает ее и передает ей указатель на буфер памяти. Распределитель памяти создает медиа выборки по мере поступления данных. Когда медиа выборка поступила на входной контакт по запросу следующего фильтра, и буфер памяти освободился, она записывается в список свободных медиа выборок. Если в списке есть свободные выборки, распределитель памяти использует их повторно, что уменьшает расход памяти, ускоряет процесс подготовки данных для транспорта и предотвращает перезапись данных в необработанной выборке. Соединенные контакты используют общий распределитель памяти. Если фильтр не изменяет размер буфера памяти, то для входного и выходного контакта фильтра используется один распределитель и данные обрабатываются в одном буфере без перемещения их из буфера входного контакта в буфер выходного контакта. Раздельные распределители памяти используются, если фильтр увеличивает объем буфера, например при декомпрессии данных.

Состояния фильтров и фильтрового графа

Фильтры могут быть в одном из трех состояний, остановленном, запущенном (активном) и в состоянии паузы. Когда фильтр находится в остановленном состоянии, он не принимает и не обрабатывает медиа выборки. Состояние запуска и паузы для большинства фильтров неразличимы. В этом состоянии они принимают медиа выборки, обрабатывают их и передают на выход. Исключение представляет два фильтра воспроизводящий фильтр и жи-

вой фильтр источник. Термин "живой источник" (Live Source Filter) означает фильтр, принимающий данные в реальном масштабе времени и не изменяющий скорости поступления данных. Например, фильтр сбора данных (capture) телевизионного приемника.

Живой видео фильтр источник при переходе в состояние паузы не продуцирует медиа выборки. Воспроизводящий видео фильтр при переходе из состояния остановки в состояние паузы, принимает одну медиа выборку, блокирует прием последующих выборок и отображает принятую медиа выборку как стоп кадр. Звуковой воспроизводящий фильтр не обрабатывает выборки до момента запуска, поскольку воспроизводить фрагмент звука не имеет смысла.

Менеджер графа управляет переходами фильтров из одного состояния в другое. Правильная последовательность перехода фильтра из состояния запуска в состояние остановки и обратно должна обязательно включать состояние паузы. Целью перевода фильтров в состояние паузы является создание очереди данных, распределенных по всей цепочке фильтров. Это позволяет по команде пуска немедленно начать воспроизведение потока данных с точки остановки. Приложению не обязательно производить вызов метода для перевода графа в состояние паузы, менеджер графа делает это самостоятельно.

Рассмотрим перевод фильтров из состояния запуска в состояние паузы для обычного фильтра источника, например фильтра источника, производящего чтение файла. При переводе фильтров в состояние паузы менеджер графа устанавливает в паузу каждый фильтр, начиная с воспроизводящего фильтра и передвигаясь вверх по потоку до фильтра источника. При такой последовательности перевода фильтров в состояние паузы они продолжают прием, обработку медиа выборок и их передачу, создавая очередь данных из обработанных выборок. В тоже время такой порядок предотвращает потерю выборок и выход фильтров из синхронизации. Фильтр источник для ускорения перевода фильтров в состояние паузы создает новый поток и генерирует новые выборки с максимально возможной скоростью. Поток проталкивает выборки по цепочке фильтров вниз по потоку. Когда первая, полностью обработанная после команды останова, выборка достигнет воспроизводящего фильтра, он заблокирует прием выборок. Фильтр выше по потоку примет следующую выборку и также заблокирует прием. Процесс продолжается до тех пор, пока все фильтры выше по потоку не получат свои выборки, заблокируются, и таким образом создадут очередь данных. В результате, воспроизведение потока может начаться в любой момент без потери данных. Переход в состояние паузы может быть достаточно длительным, если в графе есть фильтр, выполняющий декомпрессию, но это не имеет большого значения, так как перевод в состояние запуска может быть произведен в любой момент времени практически без задержки. Фильтр источник и воспроизводящий фильтр не имею перехода в состояние запуска, они просто продолжают процесс обработки

данных. Воспроизводящий фильтр начинает принимать выборки из очереди данных и воспроизводит их в соответствии с их потоковым временем. Если очередная выборка требует промежуточного хранения до момента воспроизведения, фильтр блокирует прием новой выборки или принимает ее в очередь на рабочем потоке.

Живой источник, например телевизионный приемник, является исключением в данном процессе. При наличии в графе живого источника, не создается очереди данных, поскольку граф может сколь угодно долго, находится в состоянии паузы и устаревшие данные в очереди становятся просто ненужными. Таким образом, живой источник производит выборки только в состоянии запуска.

В состоянии остановки фильтры не принимают никаких выборок. Фильтр источник выключает рабочие потоки, другие фильтры выключают свои рабочие потоки, которые обрабатывали данные. Контакты прекращают распределение памяти.

Синхронизация фильтров

Синхронизацию фильтров в графе осуществляет менеджер графа. Временные отсчеты, формируемые менеджером графа, называются reference clock. Дословно этот термин можно перевести как ссылка на генератор тактовой частоты, более точно по смыслу это генератор тактовой частоты, используемый для синхронизации фильтров. Часто этот термин переводят как ссылочное время. Любой фильтр в графе может использоваться менеджером как источник ссылочного времени, если он предоставляет интерфейс IReferenceClock. Фильтр DirectShow может быть источником ссылочного времени, если он имеет доступ к системному таймеру, например, фильтр звуковой обработки. Если ни один фильтр не имеет интерфейса ссылочного времени, тогда менеджер графа использует объект на основе системного времени, который будет использоваться для синхронизации фильтров. Источник синхронизации может быть выбран приложением и указан менеджеру графа. Это возможность используется в редких случаях, когда к источнику синхронизации предъявляются особые требования.

Номинальное значение тактового интервала синхронизатора равно 100 наносекунд, точное значение может быть и меньше. Получить значение времени можно, вызвав метод IReferenceClock::GetTime. Начало отсчета времени зависит от реализации, поэтому абсолютное значение величины времени не имеет особого значения. Важно определить относительное время от момента старта графа. Метод GetTime всегда должен возвращать монотонно увеличивающуюся величину. Если в качестве синхронизатора используется аппаратный источник, и если в процессе его настройки величина времени уменьши-

лась, GetTime должен возвращать последнее значение до тех пор, пока значение времени аппаратного источника не достигнет возвращаемого значения.

При запуске графа менеджер графа применяет следующий алгоритм выбора синхронизатора.

- Если приложением выбран источник синхронизации, то он и используется.
- Если граф содержит живой фильтр источник (live source filter), который поддерживает интерфейс IReferenceClock, то используется этот фильтр.
- Если граф не содержит живого источника, используется любой фильтр, который поддерживает интерфейс IReferenceClock. Предпочтение отдается соединенным фильтрам. Если в графе есть фильтр, обрабатывающий звук, то выбирается он.
- Если нет соответствующего фильтра, то выбирается системный синхронизатор(System Reference Clock), базирующийся на системном времени.
- Приложение может установить собственный синхронизатор, вызвав метод менеджера графа IMediaFilter::SetSyncSource. Это можно делать, если есть существенные причины для отказа от автоматического выбора синхронизатора. Можно указать менеджеру, не использовать синхронизатор с целью максимального ускорения обработки данных. Для этого метод SetSyncSource вызывается с нулевым параметром. Для восстановления синхронизатора по умолчанию вызывается метод IFilterGraph::SetDefaultSyncSource. Для возврата указателя на источник ссылочного времени, который используется по умолчанию, следует вызвать метод IMediaFilter::GetSyncSource, запросив интерфейс у менеджера графа.

Когда происходит изменение синхронизации, менеджер графа уведомляет фильтры в графе, вызывая метод IMediaFilter::SetSyncSource для каждого фильтра. Приложение не должно вызывать этот метод для каждого фильтра.

В DirectShow имеется два источника синхронизирующего времени абсолютный это ссылочное время и относительный, который называется потоковое время. Потоковое время еще называют временем представления (presentation time). Потоковое время определяется относительно момента запуска графа. Когда граф стартует, потоковое время определяется как разница между ссылочным временем и временем старта. Если граф в состоянии паузы потоковое время запоминается при переходе графа в состояние паузы. После опе-

рации поиска (seek) по потоку потоковое время сбрасывается в нуль. Когда граф остановлен, потоковое время не определено.

Каждая медиа выборка снабжается временным штампом (time stamp), который формируется из потокового времени и нормализуется относительно старта графа. Старт графа принимается за нулевой отсчет времени. Временной штамп указывает, когда выборка была записана, а для режима воспроизведения указывает, когда выборка должна быть воспроизведена. Временной штамп содержит начало выборки и время ее окончания. При старте графа менеджер графа производит запуск каждого фильтра, что приводит к небольшим смещениям во времени старта между фильтрами. Для компенсации этих смещений менеджер фильтра устанавливает время старта с небольшим опережением.

Использование объектов DMO в DirectShow

Все видео и звуковые кодеки в Windows реализованы в архитектуре DMO. Поскольку фильтры DirectX включают также и кодеки, необходимо иметь общее представление об архитектуре DMO. Объект DMO это COM объект, предназначенный для преобразования данных.

Объект DMO может иметь m входов и n выходов. Входы и выходы объекта называются потоками. Поток не является объектом, это просто ссылка на индекс входа или выхода. Количество потоков определяется во время проектирования.

Все данные, обрабатываемые DMO, характеризуются медиа типом. Медиа тип определяет содержание данных и технические характеристики. Например, видео изображение размером 320x240 пикселей, RGB 24 бита это один медиа тип, а звук PCM 44,1 кГц, 16 бит, стерео, это другой медиа тип. Медиа тип описывается структурой DMO_MEDIA_TYPE, которая должна быть заполнена перед началом обработки данных. Обычно поток может обрабатывать несколько медиа типов, или ограничиваться определенными характеристиками одного медиа типа. Некоторые DMO могут обрабатывать широкий диапазон медиа типов. Приложение может произвести перечисление медиа типов для потока и проверить их на соответствие требуемым характеристикам.

Архитектура DMO возлагает ответственность за распределение буфера данных на приложение. Каждый блок памяти входных данных DMO организован как COM объект и поддерживает интерфейс IMediaBufer. Этот интерфейс содержит методы для установки размера буфера и возвращает указатель на буфер и распределенный размер буфера. По умолчанию предполагается

использование входного и выходного буфера данных, также предусмотрена и возможность загрузки выходных данных во входной буфер. Первая возможность реализуется по умолчанию с помощью интерфейса IMediaObject, вторая с помощью интерфейса IMediaObjectInPlace, который может и не поддерживаться объектом.

Приложение DirectShow может использовать объекты DMO с помощью фильтра обертки (DMO Wrapper Filter), который производит сборку нескольких объектов DMO и позволяет использовать их как фильтр DirectShow. Фильтр обертка скрывает все детали реализации и использования DMO, такие как распределение буферов, пересылка данных через DMO и так далее. Поскольку фильтр обертка собирает объекты DMO в единый комплекс, приложение может запрашивать любой интерфейс, предоставляемый DMO. Приложение не должно вмешиваться в процесс обработки потоков, который выполняется DMO. Превращение DMO в фильтр DirectShow выполняется достаточно просто и не требует от программиста значительных усилий.

Соединение фильтров

Соединение фильтров производит менеджер графа. Далее описывается методика, которой руководствуется менеджер графа. Программист не должен применять описываемую методику при соединении фильтров.

Соединение фильтров производится с помощью контактов. Управление соединением фильтров осуществляется посредством интерфейса IPin, который предоставляется на контакте. Каждый соединяемый контакт имеет медиа тип, описываемый структурой AM_MEDIA_TYPE. Приложение при соединении фильтров должно использовать методы менеджера графа и никогда не вызывать методы интерфейсов фильтра или контакта. В описании методов интерфейса IPin (и других) прямо указывается, разрешается приложению использовать данный метод или нет. Если метод не может применяться приложением, то это означает, что он применяется только фильтром или менеджером графа. Соединение фильтров может происходить прямо с указанием соединяемых фильтров или непрямым способом, например при вызове метода, который производит воспроизведение файла. Перед соединением фильтров они должны быть установлены в граф. Это делает либо приложение, либо менеджер графа. При непрямом соединении все управление построением графа передается менеджеру графа. Менеджер графа самостоятельно определяет, какие фильтры должны использоваться в графе, устанавливает их и соединяет. В процессе соединения фильтров осуществляется проверка медиа типа потока, согласование распределения буфера и запрос необходимых интерфейсов.

Процесс соединения двух фильтров производится следующим образом:

- Менеджер графа вызывает метод IPin::Connect на выходном контакте, передавая методу указатель на входной контакт следующего фильтра.
- Если выходной контакт способен к соединению, то он вызывает IPin::ReceiveConnection на входном контакте соединяемого фильтра. Метод проверяет, может ли входной контакт соединиться с медиа типом данных, который предлагает выходной контакт.
- Если входной контакт может принимать данные с предлагаемым медиа типом, то фильтры соединяются.

Все соединения фильтров производятся при остановленном графе. Существует возможность динамического подключения фильтров при запущенном графе. В таком случае фильтр должен поддерживать интерфейсы IPinConnection и IPinFlowControl. Большинство фильтров не поддерживают динамическое соединение. Соединение фильтров обычно производится вниз по потоку, вначале подключается входной контакт первого фильтра, затем выходной контакт первого фильтра подключается к входному контакту второго фильтра и так далее. Все фильтры должны поддерживать такой порядок соединения. Некоторые фильтры поддерживают и обратный порядок соединения. Например, фильтр мультиплексора (Mux) может быть подключен к фильтру файлового писателя перед подключением своего входного контакта.

Согласование медиа типов при соединении фильтров

Когда менеджер графа вызывает метод IPin::Connect он проверяет медиа тип на контакте. Есть несколько вариантов указания медиа типов, которые позволяют производить соединение:

- Медиа тип указан полностью, все поля структуры AM_MEDIA_TYPE заполнены. Если соединяемые фильтры используют такой медиа тип, то контакты его принимают и производится соединение.
- Медиа тип указан частично, поля главного типа, подтипа или формата имеют значение GUID_NULL. Величина идентификатора GUID_NULL в данном случае играет роль подстановочного знака, указывая, что любое значение величины приемлемо. В этом случае контакты производят согласование частичных медиа типов.
- Медиа тип не указан. Указатель на структуру равен NULL, контакты согласовывают любой медиа тип, при котором возможно соединение.

Если контакты соединились, графический менеджер заполняет медиа тип полностью, чтобы ограничить число возможных медиа типов для соединения. В процессе согласования выходной контакт предлагает медиа тип, вызывая на входном контакте метод IPin::ReceiveConnection. Входной контакт принимает или отвергает предложенный медиа тип. Этот процесс повторяется до приема соединения или до исчерпания выходным контактом вариантов доступных медиа типов. Выбор медиа типов, предлагаемых для соединения, зависит от реализации. В DirectShow предлагается перечислять предустановленные медиа типы на входном контакте и сопоставлять их с медиа типами на выходном контакте, используя метод IPin::EnumMediaTypes.

Согласование транспортировки данных

После согласования медиа типов контакты нуждаются в транспорте данных от фильтра к фильтру. В DirectShow нет ограничений на параметры транспорта, достаточно, чтобы оба фильтра поддерживали одинаковую схему транспорта данных. Наиболее употребительная схема транспорта это передача данных через локальную память.

Существует две разновидности транспорта через локальную память, которые поддерживаются DirectShow:

- Первая модель предполагает, что фильтр источник генерирует медиа выборку данных и отправляет их фильтру ниже по потоку. Нижний фильтр пассивно принимает данные и обрабатывает их. Эта модель транспорта данных называется моделью проталкивания (push) данных. Она применяется наиболее часто и использует интерфейс IMemInputPin.
- Вторая модель использует способ передачи данных по запросу. Фильтр источник данных соединяется с фильтром синтаксического разбора (parser filter), который запрашивает данные у фильтра источника. На этот запрос фильтр источник отвечает передачей данных. Эта модель называется моделью выталкивания данных, точнее извлечением данных у источника (pull). Она применяется реже и использует интерфейс IAsyncReader.

В транспорте данных с использованием локальной памяти за распределение буферов данных ответственен COM объект, называемый распределителем. Он предоставляет интерфейс IMemAllocator. Оба контакта участвующие в соединении используют один общий объект распределителя, но выходной контакт определяет использование распределителя. Выходной контакт устанавливает свойства распределителя памяти, которые определяют, сколько буферов будет создавать распределитель, размер каждого буфера и выравнивание памяти на границу. Выходной контакт может быть подчинен требованиям, которые предъявляет к распределителю входной контакт.

В модели транспорта, использующей интерфейс IMemInputPin(push), распределитель при согласовании выполняет следующие действия:

- Выходной контакт может вызвать метод IMemInputPin:: GetAllocatorRequirements для учета требований входного контакта к свойствам буфера. Выходной контакт должен учитывать требования входного контакта, если нет особых условий препятствующих этому.
- Выходной контакт может вызвать метод IMemInputPin::GetAllocator для запроса распределителя с входного контакта.
- Выходной контакт выбирает распределитель. Он может использовать распределитель входного контакта или создать свой.
- Выходной контакт вызывает метод IMemAllocator::SetProperties для установки свойств распределителя. Распределитель может не выполнить установку свойств. Это может произойти, если используется распределитель входного контакта. Распределитель возвращает действительные свойства, как выходные параметры в методе SetProperties.
- Входной контакт должен вызвать IMemAllocator::GetProperties для проверки, что свойства распределителя применимы.
- Выходной контакт отвечает за распределение памяти распределителем и ее освобождение при стартах и остановках графа.

В модели транспорта, использующей интерфейс IAsyncReader (pull), распределитель при согласовании выполняет следующие действия:

- Входной контакт вызывает метод IAsyncReader::RequestAllocator на выходном контакте, в котором указывает требуемые свойства буфера и может предоставить свой распределитель.
- Выходной контакт выбирает распределитель. Он может использовать распределитель входного контакта или создать свой.
- Выходной контакт возвращает распределитель в выходном параметре метода RequestAllocator. Входной контакт должен проверить свойства распределителя.
- Входной контакт отвечает за распределение памяти распределителем и ее освобождение.
- В любое время процесса согласования любой контакт может прервать соединение.

- Если выходной контакт использует распределитель входного контакта, то он может использоваться только для производства выборок для этого входного контакта. Фильтр не может использовать этот распределитель для производства выборок для других своих контактов.

В процессе соединения фильтру может понадобиться произвести повторное соединение своего контакта. Это может произойти в следующих случаях:

- Когда фильтр является фильтром, дублирующим входные данные в несколько выходных потоков без их изменения (Tee filters). Этот фильтр принимает диапазон медиа типов, но медиа типы должны быть одинаковы для всех соединений и поэтому фильтр должен согласовать все выходные соединения после выполнения входного соединения. Это может потребовать повторного соединения контактов.

- Когда фильтр является фильтром, который модифицирует данные в исходном буфере без их копирования в отдельный выходной буфер (trans-in-place filter). Такой фильтр должен использовать тот же самый распределитель для входного и выходного соединения. Первый соединяемый контакт (входной или выходной) согласовывает соединение обычным образом. Когда соединяется второй контакт, первый распределитель может не пройти согласование. В этом случае второй контакт выбирает другой распределитель, и первый контакт повторно соединяется с новым распределителем.

Процесс повторного соединения происходит следующим образом:

- Фильтр вызывает метод IPin::QueryAccept на другом контакте фильтра с новым медиа типом.

- Если вызов метода завершился без ошибки, вызывается метод IFilterGraph2::ReconnectEx для выполнения повторного соединения.

"Интеллектуальное" соединение фильтров

Механизм "интеллектуального" соединения используется менеджером графа при построении и перестроении графа. Он содержит несколько связанных алгоритмов для выбора фильтров при включении их в граф. Алгоритмы "интеллектуального" соединения применяются в следующих методах:

- IGraphBuilder::Render

- IGraphBuilder::AddSourceFilter

- IGraphBuilder::RenderFile
- IGraphBuilder::Connect
- ICaptureGraphBuilder2::RenderStream

Метод Render строит ветвь графа. Он начинает работу с не подключенного выходного контакта, установленного в граф фильтра и строит ветвь графа вниз по потоку, добавляя при необходимости промежуточные фильтры. В каждом шаге алгоритма метод ищет для фильтра другой фильтр, который может с ним соединиться. Если фильтр имеет несколько выходных контактов, поток может разветвляться. Поиск останавливается, когда поток полностью обработан. Если алгоритм блокируется, в какой либо точке, он может вернуться в начало и продолжить работу с другим набором фильтров.

При соединении контактов метод Render работает по следующему сценарию:

- Если контакт поддерживает интерфейс IStreamBuilder, менеджер графа передает входной процесс методу IStreamBuilder::Render, который строит граф до воспроизводящего фильтра. Очень мало контактов поддерживают этот интерфейс.
- Менеджер графа, прежде всего, пытается использовать фильтры, сохраненные в кеше на более ранних шагах процесса соединения. Некоторые фильтры менеджер графа помещает в кеш на предыдущих шагах алгоритма.
- Если фильтровый граф содержит любой фильтр с неподключенным входным контактом, то менеджер графа проверяет его на возможность подключения. Это позволяет ускорить работу метода, предварительно установив нужные фильтры в граф вручную.
- В последнюю очередь проверяются фильтры, зарегистрированные в системе. Поиск осуществляется с помощью карты фильтров, метод IFilterMapper2::EnumMatchingFilters. Соответствие определяется по медиа типу выходного контакта.

При поиске в регистре метод EnumMatchingFilters возвращает фильтры в соответствии с их критерием предпочтения. Фильтры группируются по категориям. Категории фильтров также имеют критерий предпочтения. Категории со значением MERIT_DO_NOT_USE или меньше игнорируются, даже если внутри категории есть фильтры с высоким значением критерия. Первыми

возвращаются фильтры с минимальным значением критерия. Фильтры со значением MERIT_DO_NOT_USE или меньше игнорируются.
Метод AddSourceFilter добавляет фильтр источник для воспроизведения указанного файла. Метод ищет в регистре зарегистрированное расширение файла для фильтра или протокол (такой как http://). Если метод не находит такого соответствия, то читает набор проверочных байт (check bytes) из файла и ищет соответствующий набор в регистре. При нахождении фильтра метод создает его экземпляр, добавляет его в граф и вызывает метод IFileSourceFilter::Load для загрузки файла.

Метод RenderFile строит граф воспроизведения файла по умолчанию, используя имя файла. Метод использует вызов AddSourceFilter для добавления фильтра источника и вызов Render для построения оставшейся части графа.

Метод Connect производит соединение контактов. Метод добавляет при необходимости промежуточные фильтры, используя вариант алгоритма метода Render, описанный выше. Порядок действий метода следующий:

1. Пытается произвести прямое соединение фильтров.
2. Пытается использовать фильтры из кеша.
3. Пытается использовать фильтры в графе.
4. Ищет фильтры в регистре.

Следует отметить, что алгоритм "интеллектуального" соединения хорошо работает для стандартных многократно проверенных задач, например воспроизведения файлов. В более сложных ситуациях, например при построении графа для телевизионной карты алгоритм делает сбои, может произвести неправильное построение графа.

Представление аппаратных устройств в фильтровом графе

Все фильтры DirectShow являются программными компонентами режима пользователя. Для реализации управления аппаратными средствами, фирма Microsoft разработала единую модель WDM (Windows Driver Model) драйвера. В этой модели все драйверы, начиная с первичного драйвера устройства и заканчивая драйвером, связанным с фильтром DirectShow объединены единым замыслом, что позволяет производителю аппаратуры проектировать устройство и первичный драйвер под управление в DirectShow. Такой подход имеет очень много преимуществ. Он дает свободу производителю аппаратных средств, избавляя его от забот связанных с проектированием драйверов для разных платформ, и в тоже время позволяет разрабатывать любые фильтры DirectShow, не опасаясь, что они будут несовместимы с аппаратурой. Если платформа поддерживает DirectShow, то на ней будет работать любое WDM устройство.

Все новые телевизионные устройства используют архитектуру WDM драйвера. Драйверы WDM обеспечивает поддержку всех новых устройств, например USB видеокамер, устройств с интерфейсом 1394 и многое другое. В тоже время DirectShow обеспечивает обратную совместимость с устаревшим программным и аппаратным обеспечением.

Фильтры DirectShow используются как программные компоненты, работающие в режиме пользователя. Они доступны для прикладного программиста без ограничений. Устройства работают в режиме ядра, управляются драйверами и недоступны для программиста в той степени, в какой доступны фильтры DirectShow. Для представления устройств в DirectShow как обычных фильтров используются специализированные оберточные фильтры для каждого устройства. Оберточные фильтры предоставляют свои интерфейсы, которые отражают все возможности конкретного устройства. Вызовы методов интерфейсов со стороны приложения транслируются оберточным фильтром в форму доступную для понимания драйвером устройства и затем передаются ему. Методы фильтров оберток осуществляют управление драйвером, скрывая все детали связи с устройством, работающим в режиме ядра. Все фильтры обертки, кроме VFW Capture и Audio Capture поддерживают настройку свойств драйвера с помощью интерфейса IKsPropertySet. Для программиста управление устройством ничем не отличается от управления любым другим фильтром DirectShow.

Кроме оберточных фильтров DirectShow поддерживает фильтр KsProxy (ksproxy.ax), с помощью которого представляется любой тип потокового устройства в операционной системе Windows. Новые устройства обычно

представляются этим фильтром. Устройство должно поддерживаться драйвером, который проектировался с учетом требований для драйверов WDM(Windows Driver Model®). В этом случае независимый производитель устройств может расширять возможности своего устройства и поддерживать дополнительные функции. Это обеспечивается возможностью фильтра KsProxy объединять COM объекты. Фильтр KsProxy взаимодействует с драйвером устройства таким же образом, как и оберточный фильтр. Число контактов фильтра и число интерфейсов определяются драйвером, с которым связан фильтр. Поскольку фильтр может представлять любое устройство, то он для своего конфигурирования использует набор структур, определяющих свойства устройства. Структуры свойств могут использоваться WDM драйверами и некоторыми фильтрами режима пользователя, например программным декодером MPEG2.

Фильтр представляется в графе под именем устройства, но не под именем KSProxy. Для просмотра фильтров KSProxy установленных в системе можно запустить утилиту GraphEdit, выбрать пункт меню Graph/Insert Filters и в открывшемся окне поискать в категории WDM Streaming установленные в системе устройства. В категории "потоковые системные устройства" может быть, например интегрированное звуковое устройство. Таким образом, каждое устройство представляется своим фильтром, или одно устройство может быть представлено несколькими фильтрами. В диспетчере устройств одна телевизионная карта может представляться как совокупность нескольких устройств разного назначения. Например, телевизионная карта аналогового телевидения может быть представлена несколькими фильтрами, своим для каждого отдельного устройства:

- Фильтр TV Tuner, выполняющий настройку приемника.
- Фильтр TV Audio, обеспечивающий управление приемником звуковых программ.
- Фильтр Analog Video Crossbar, управляющий телевизионным коммутатором сигналов.
- Фильтр Audio Capture, управляющий звуковой картой.
- Фильтр VFW Capture, поддерживающий устаревшие телевизионные карты.

Устройство, представленное фильтром KSProxy, управляется со стороны приложения как любой фильтр DirectShow.

Обобщенная структура телевизионного приемника, так как она представляется в фильтровом графе DirectShow, показана на рисунке 1.5. Подобный граф выполняет запись файлов AVI и обеспечивает предварительный просмотр. Рис. 1.5 показывает некоторые фильтры и их соединения в упрощенном виде. Все соединения фильтров отражают программную логику, но не физическую логику соединения устройств телевизионной карты.

Потоки данных

Устройства WDM поддерживают передачу потоков данных в режиме ядра, когда данные передаются полностью в режиме ядра без передачи их в режим пользователя. Передача потоков данных под управление в режим пользователя и обратно сильно увеличивает загрузку центрального процессора, в то время как пересылка данных в режиме ядра выполняется без участия центрального процессора. Фильтры WDM используют потоки данных в режиме ядра для передачи мультимедийных данных из одного устройства в другое не только размещенных на одной карте, но и на разных картах без копирования данных в основную системную память.

С точки зрения приложения это выглядит, так как будто данные передаются в режиме пользователя из одного фильтра в другой. Реально потоки данных никогда не передаются в режим пользователя полностью до тех пор, пока не будут выданы на отображение в графическую карту или на фильтр высокоуровневой обработки данных, например в компрессор.

В некоторых сценариях обработки, таких как запись файла, требуется переход потока данных в режим пользователя только в одной точке фильтрового графа. При этом не требуется копирование данных на новое место в памяти. Приложение явно управляет передачей потоков данных из одного режима в другой на этапе построения графа, когда например данные передаются из устройства в фильтр, работающий в режиме пользователя. Программист может, не беспокоится по поводу потоков данных режима ядра, информация о них дается в рамках общего ознакомления с принципами передачи данных в DirectShow. Нет никаких условий для программиста требующих учета передачи потоков из одного режима в другой. Все потоки данных управляются, так как будто они находятся в режиме пользователя.

В архитектуре WDM осуществляется оптимизированная связь потоков данных режима ядра с потоками режима пользователя. Для этой цели используется слой аппаратно независимых драйверов, называемых потоковые класс драйверы(Stream class drivers), или сокращено потоковые драйверы.

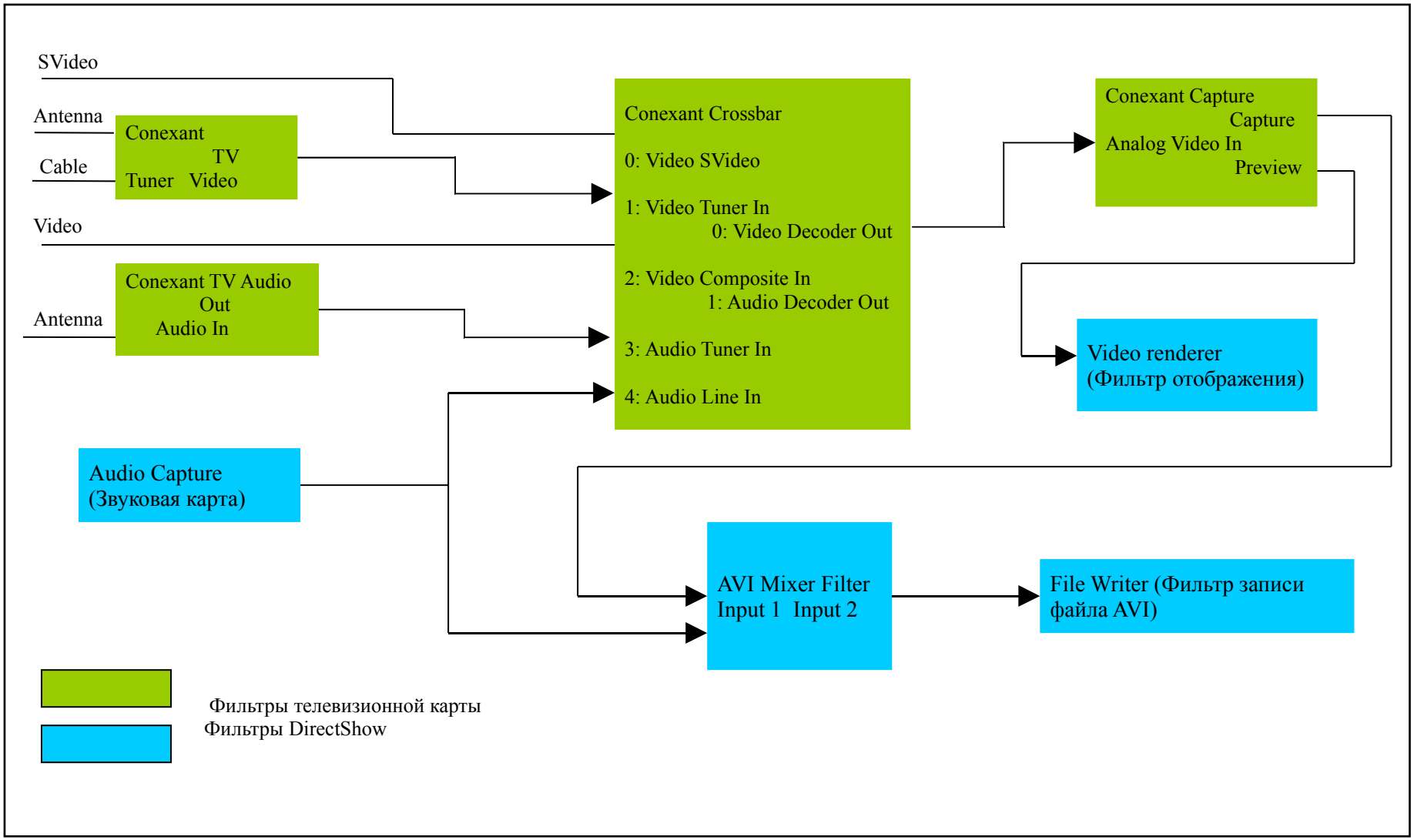


Рис. 1. 3 Фильтровый граф телеприемника.

Потоковые класс драйверы производятся Microsoft. Потоковые драйверы также отвечают за вызовы минидрайверов, которые являются аппаратно ориентированными динамическими библиотеками (DLL) и поддерживают специфические аппаратные функции. Минидрайвер и потоковый драйвер, работая совместно, обеспечивают наименьшее время задержки потока данных при низкоуровневой обработке, а DirectShow обеспечивает высокоуровневую поддержку приложения. Поскольку потоковые драйверы используют единую потоковую модель как для стандартных типов данных так и не стандартных, а также поддерживают передачу данных между драйверами ядра без передачи в режим пользователя, они очень эффективны в применении. Передача данных в режим пользователя из режима ядра производится только тогда, когда необходимо произвести высокоуровневую обработку данных, как например компрессию или воспроизведение потока. Переходы потоков данных из режима ядра и обратно занимают много времени из-за многочисленных проверок параметров, защиты, возможного копирования данных. Использование обработки потоков в режиме ядра значительно уменьшает число передач данных. Когда потоки обрабатываются в режиме ядра, DirectShow только предоставляет средства для управления ими. Например, вся обработка телевизионных данных производится на уровне ядра и переходит в режим пользователя при передаче в компрессор или для записи на диск, или для отображения (воспроизведения).

Таким образом, фильтры DirectShow связаны с устройством через два слоя драйверов. Нижний слой мини драйверов реализует управление конкретным типом устройства и отправляет видеопоток драйверу Stream class. Драйвер видеопотока(Stream class) производит низкоуровневую обработку информации от мини драйверов для каждого типа информации отдельно, формирует видеопотоки, производит дополнительную обработку данных и отправляет их соответствующим фильтрам DirectShow. Общая структура связи фильтров DirectShow и драйверов WDM показана на рисунке 1.3.

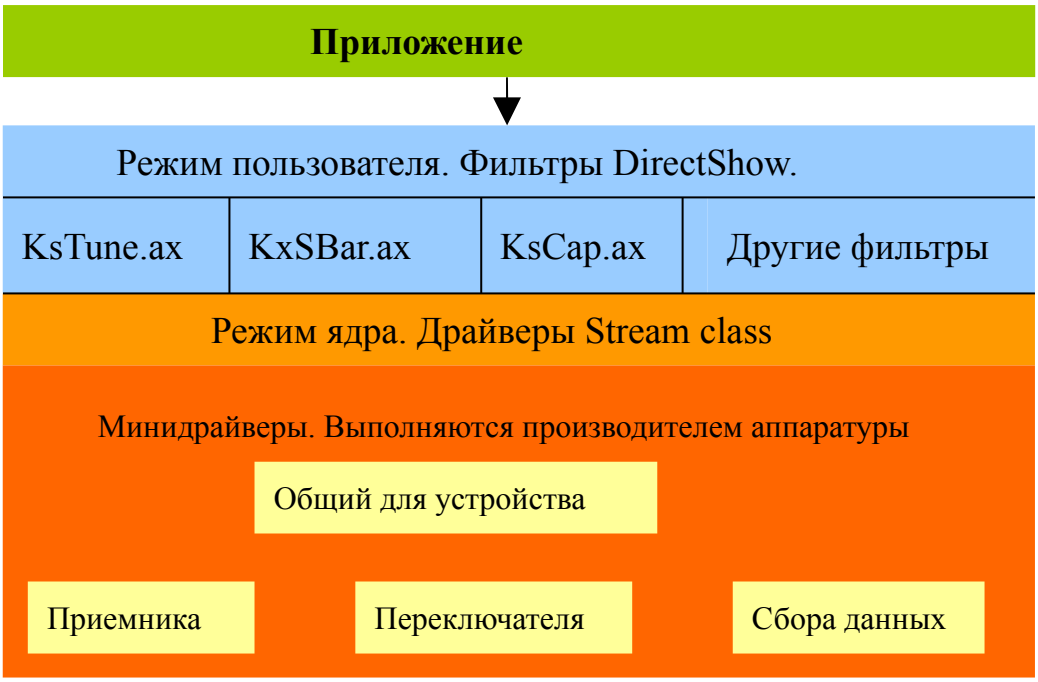


Рис. 1. 4. Связь фильтров DirectShow и драйверов WDM.

Отображение WDM устройств в локальной системе

Отображение WDM устройств в локальной системе может быть рассмотрено на примере аналоговой телевизионной карты TV-PVR 878.

Все аналоговые телевизионные устройства сводятся к двум типам:

- Устройства, передающие данные в системную память по шинам PCI, USB, IEEE 1394 и затем в видеопамять графической карты. Физическая реализация таких устройств выполняется на чипе Bt848 и более новых Bt878.
- Устройства, передающие данные непосредственно в видеопамять графической карты через входной разъем, называемый видеопорт. Для таких устройств, требуется использование фильтра VBI Surface Allocator, который выполняет распределение памяти графической карты. Физическая реализация этих устройств выполняется на чипе Bt829.

Аналоговые фильтры предназначены для управления картой телеприемника. Карта телеприемника широкого назначения обычно содержит следующие устройства:

- Приемник телевизионного сигнала.

- Переключатель сигналов, который может либо отсутствовать, либо может быть несколько переключателей сигналов.
- Устройство сбора данных. Устройство включает в себя видеопроцессор, видеодетектор и аналого-цифровой преобразователь телевизионного сигнала.
- Возможно, отдельный или совмещенный звуковой приемник. Это может быть приемник диапазона FM или другого типа для приема звуковых программ.

Для обеспечения совместимости с DirectShow и другими программными продуктами производители телевизионных карт должны поддерживать драйверы WDM для всех устройств карты.

Все перечисленные устройства телеприемника представляются четырмя компонентами DirectShow: фильтр настройки (TV Tuner filter), фильтр приемника звуковых программ (TV Audio filter), фильтр коммутатора сигналов(Analog Video Crossbar filter), фильтр сбора видеоданных (Video Capture filter). После стандартной установки драйверов телевизионной карты все указанные устройства отображаются в окне диспетчера устройств, как показано на рис 1.4. Пример приведен для карты на чипе Conexant BT878.

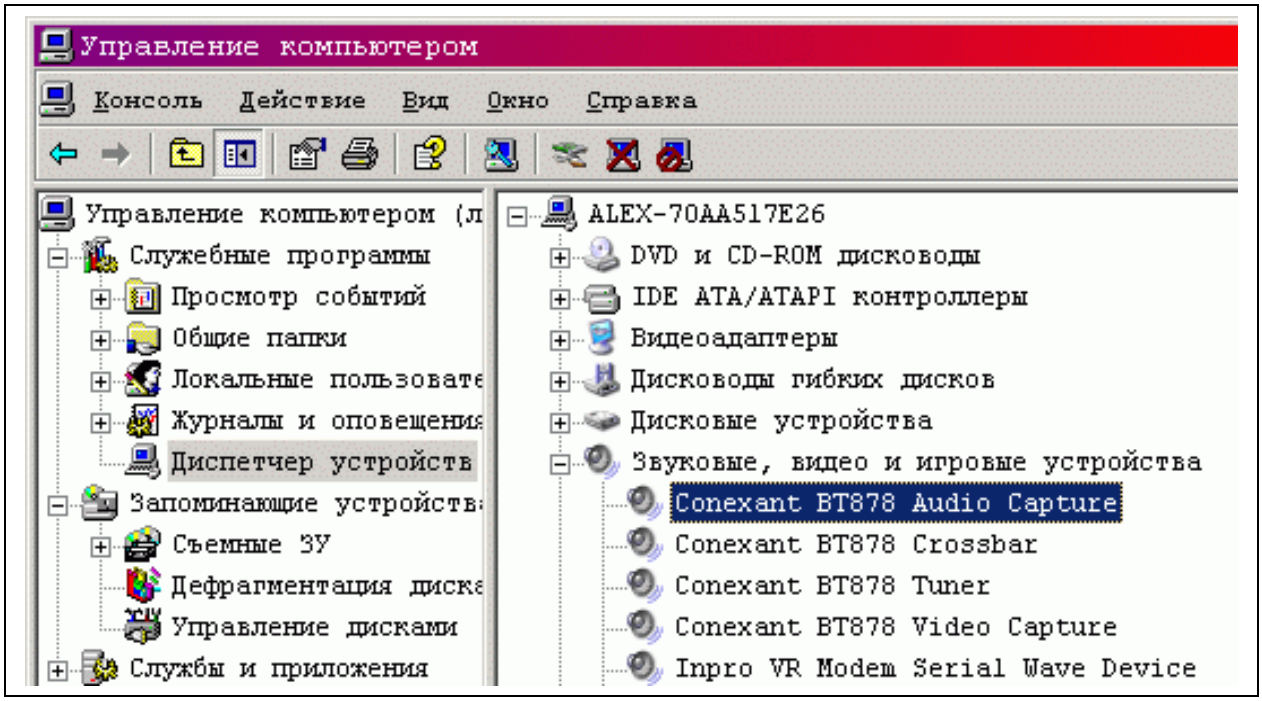


Рис. 1. 5 Отображение устройств телевизионной карты в диспетчере.

Сбор видео и звуковой информации

Термин сбор видео информации(Video Capture), применяемый в документации DirectShow, подразумевает прием видеоинформации от любого устройства, например, видео камеры, телевизионной карты, видеомагнитофона и других устройств для записи и (или) просмотра.

Термин сбор аудио информации(Audio Capture) подразумевает прием звуковой информации поступающей от звуковой карты для записи и (или) воспроизведения.

Большинство новых устройств для сбора видео информации, осуществляют поддержку WDM драйверов. В фильтровом графе DirectShow любое WDM устройство сбора именуется как WDM фильтр сбора видео информации(WDM Video Capture Filter). Фильтр сбора настраивается, основываясь на характеристиках драйвера устройства, и появляется под именем предоставляемым драйвером. На рис. 1.4 можно увидеть этот фильтр под именем Conexant BT878 Video Capture. Все устройства более раннего производства, не WDM архитектуры поддерживаются DirectShow с помощью фильтра сбора VFW Capture Filter.

Фильтр сбора видео информации(Capture)

Фильтр сбора видео информации(Capture), в дальнейшем именуемый как фильтр сбора играет ведущую роль в графе DirectShow. Он имеет некоторые особенности, выделяющие его из ряда других фильтров DirectShow. Фильтр Capture может иметь несколько выходных контактов(Pin) разного назначения. Назначение каждого контакта однозначно идентифицируется его категорией. Каждая категория имеет свой уникальный глобальный идентификатор(GUID).

Основные типы выходных контактов фильтра Capture имеют следующее назначение:

- Контакт записи (Capture), который используется для записи видеопотока в файл. Каждый кадр изображения, поступающий с контакта записи, имеет временной штамп, в отличие от контакта предварительного просмотра. Фильтр может иметь несколько контактов записи, каждый из которых предназначен для информации определенного типа. Например, контакт для записи изображения имеет категорию PIN_CATEGORY_CAPTURE.

- Контакт Preview, который используется для предварительного просмотра видео в окне. Некоторые кадры изображения, поступающие с этого контакта для предварительного просмотра, могут отбрасываться для увеличения производительности при записи файла. С этой же целью изображение с контакта предварительного просмотра не сопровождается временным штампом. Контакт имеет категорию PIN_CATEGORY_PREVIEW.

- Контакт VBI, который служит для обработки информации передаваемой на обратном ходе кадровой развертки (VBI). Состав информации зависит от телевизионного стандарта вещания. Для стандарта NTSC могут передаваться данные интерактивного телевидения (NABTS IP), вплоть до Web страниц и изображений, а также телетекст североамериканского стандарта. Для стандарта PAL этот контакт служит для передачи данных стандартного телетекста (WST).

- Контакт видеопорта (VP), позволяющий передавать видео информацию непосредственно в память видео карты без участия центрального процессора. Если устройство имеет видео порт, то контакт предварительного просмотра отсутствует. Контакт видео порта имеет определенные ограничения. Видео порт всегда использует оверлейную поверхность* для отображения в процессе работы фильтра. Управление предварительным просмотром отсутствует. Переключение кадров производится автоматически, и организовать синхронизацию с другими изображениями трудно. Видео порт имеет категорию PIN_CATEGORY_VIDEOPORT.

Примечание.
Видеопоток может воспроизводиться с использованием специального режима DirectDraw, который называется оверлеем (overlay). Видеоданных в этом случае загружаются сразу в локальную память видеокарты, где она обрабатывается аппаратными средствами самой видеокарты. После обработки буфер оверлея может выводиться на монитор или на телевизор через отдельный разъем.

Существует еще несколько типов контактов, которые могут использоваться на фильтрах сбора. Их характеристика и категории приводятся в приложении в разделе "Категории контактов".

Фильтр сбора никогда не может иметь контакты предварительного просмотра и видеопорта одновременно. Фильтр сбора может иметь входной контакт, который служит для подключения некоторых дополнительных фильтров. Это могут быть следующие фильтры.

- Фильтр настойки приемника (TV Tuner).

- Фильтр приемника звуковых программ (TV Audio).
- Фильтр переключателя входов (Analog Video Crossbar).

В зависимости от модели карты, некоторые из указанных фильтров могут отсутствовать. Эти фильтры представляют отдельные устройства. В графе они соединяются, как и обычные фильтры DirectShow, посредством контактов, но никакие данные через соединение не проходят. При соединении этих фильтров используются не медиа типы, а идентификаторы (GUID) которые идентифицируют носители. Идентификатор носителя принадлежит минидрайверу. Для соединения фильтров носители должны иметь одинаковые идентификаторы.

Фильтр сбора звука

В качестве фильтра сбора звука используется звуковая карта компьютера. Обычно звуковая карта содержит входной смеситель и переключатель сигналов, позволяющий производить запись и воспроизведение звука от различных источников. Звуковая карта в DirectShow представляется звуковым фильтром сбора (Audio Capture Filter). Фильтр использует API waveInXXX для управления драйвером звуковой карты. Кроме фильтра звука DirectShow, драйвером звуковой карты еще управляет и программа системного микшера Sndvol32.exe, которая доступна пользователь без ограничений. Таким образом, фильтр звука образует второй канал управления, что может привести к неожиданным эффектам при записи и требует учета этого обстоятельства. Каждая звуковая карта в системе представлена своим экземпляром фильтра. Входные контакты звукового фильтра не являются контактами DirectShow, они отображают физические контакты звуковой карты. Особенности использования звукового фильтра при записи видео потока будут рассмотрены отдельно.

GraphEdit инструмент визуального построения графа и отладки

В состав DirectShow входит утилита, позволяющая существенно упростить процесс создания и отладки приложения. Она позволяет построить граф в режиме редактора, то есть устанавливать и соединять фильтры, удалять фильтры, запускать граф и просматривать картинку. Для каждого фильтра и контакта можно просмотреть их свойства, а также изменить их, если фильтр позволяет это сделать.

Утилита позволяет:

- Построить граф в режиме редактора, то есть устанавливать и соединять фильтры, удалять фильтры. Это делается в визуальном режиме с использованием перетаскивания.
- Симулировать программный процесс построения графа.
- Вручную соединять фильтры с помощью перетаскивания от выходного контакта фильтра к входному контакту другого фильтра.
- Управлять графом. Режимы пуск, стоп, пауза и поиск.
- Просматривать информацию для всех зарегистрированных фильтров в системе.
- Просматривать страницы свойств фильтров.
- Просматривать медиа типы контактов.
- Тестировать аппаратуру на совместимость с DirectShow.
- Подключаться к внешнему графу для его просмотра. Таким образом, можно увидеть результаты работы своей программы и сохранить их в виде файла редактора графа.

Примерный вид части экрана GraphEdit с построенным графом показан на рисунке 6.

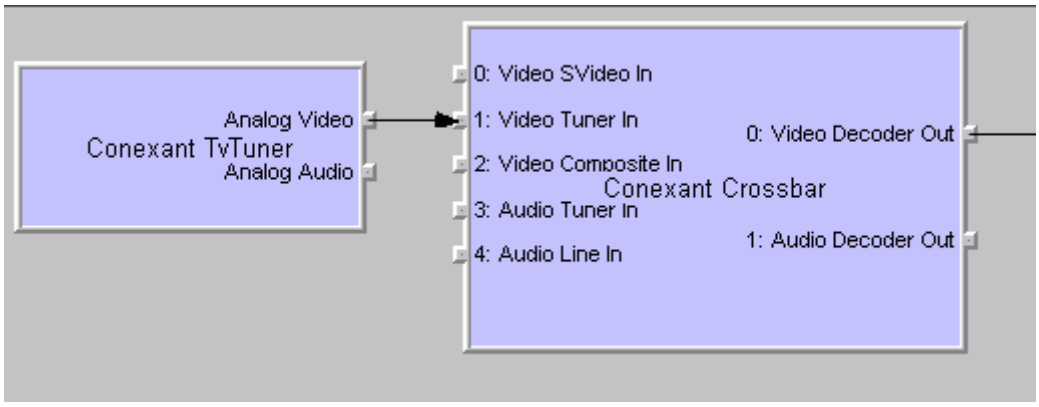


Рис. 1. 6 Вид части экрана GraphEdit.

В режиме отладки можно подключить GraphEdit к приложению и увидеть результат работы. Эта функция доступна только на Windows 2000/XP. Для подключения GraphEdit к приложению необходимо зарегистрировать моникер графа в таблице запущенных объектов (ROT), код приведен ниже.

```
HRESULT CCapture::AddToRot(IUnknown *pUnk, DWORD *pReg)
{
     HRESULT hr;
     IMoniker * pMoniker;
     IRunningObjectTable *pROT;
     if (FAILED(GetRunningObjectTable(0, &pROT)))
          return E_FAIL;

     //создание моникера графа
     WCHAR wsz[256];
     wsprintfW(wsz,L”FilterGraph %08x pid %08x”, (DWORD_PTR)pUnk,
     GetCurrentProcessId());
     hr = CreateItemMoniker(L”!”, wsz, &pMoniker);

     if (SUCCEEDED(hr))
     {
          hr = pROT->Register(0, pUnk, pMoniker, pReg);
          pMoniker->Release();
       }
     pROT->Release();
     return hr;
}
```

Вызов метода можно произвести после создания менеджера графа или после установки в граф хотя бы одного фильтра.

После окончания работы граф следует удалить из ROT.

```
void CCapture::RemoveFromRot(DWORD pdwRegister)
{
     IRunningObjectTable *pROT;

     if (SUCCEEDED(GetRunningObjectTable(0, &pROT)))
     {
          pROT->Revoke(pdwRegister);
          pROT->Release();
     }
}
```

Вызов метода `RemoveFromRot` можно произвести в деструкторе или при закрытии программы в обработчике сообщения WM_CLOSE. Применение GraphEdit для отладки и код вызова методов будет показан далее. Следует отметить, что граф созданный с помощью редактора и вполне работоспособный не всегда может быть реализован программно.