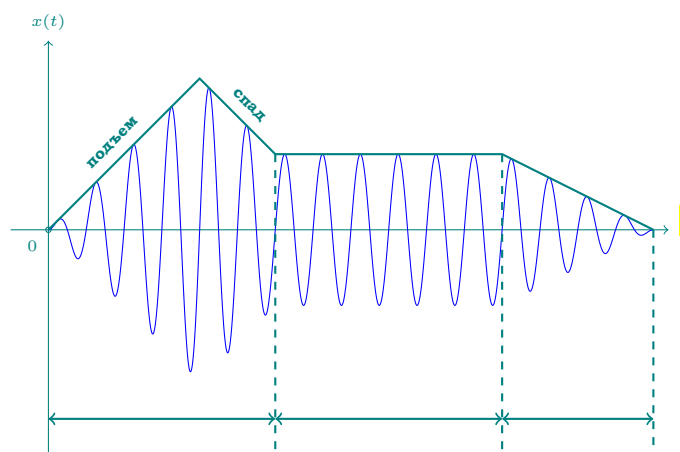


# Московский авиационный институт (государственный Т<sub>Е</sub>Хнический университет)

Факультет прикладной математики

Кафедра вычислительной математики  
и программирования

Методы и средства Мультимедиа. Звук



Преподаватель: О. В. Казанцев  
Студент: И. К. Никитин

Москва, 2011

# Содержание

<b>Введение</b>	<b>5</b>
<b>1 Основные сведения о звуковых волнах</b>	<b>6</b>
1.1 Характеристики . . . . .	6
1.2 Закон Вебера-Фехнера для звука . . . . .	7
1.3 Логарифмическая шкала . . . . .	7
1.4 Примеры . . . . .	8
1.5 Спектральное представление звука . . . . .	8
<b>Искажения и эффекты</b>	<b>11</b>
<b>2 Линейные искажения</b>	<b>11</b>
2.1 Линейное искажение . . . . .	11
2.2 Искажения в многоканальных системах . . . . .	11
<b>3 Нелинейные искажения. Помехи и шумы</b>	<b>13</b>
3.1 Примеры нелинейных искажений . . . . .	13
3.1.1 Перегрузка . . . . .	13
3.1.2 Интермодуляционные . . . . .	13
3.1.3 Биение . . . . .	14
3.2 Помехи . . . . .	14
<b>4 Цифровой способ представления звука</b>	<b>15</b>
4.1 Аналогово-цифровое преобразование . . . . .	15
4.1.1 Фильтрация . . . . .	16
4.1.2 Дискретизация . . . . .	16
4.1.3 Квантование . . . . .	17
<b>5 Динамическая обработка звука</b>	<b>18</b>
5.1 Компрессор и лимитер . . . . .	19
5.2 Гейт и экспандер . . . . .	20
<b>6 Частотная коррекция звукового сигнала</b>	<b>22</b>
6.1 Частотные фильтры . . . . .	22
6.1.1 Фильтр низких частот . . . . .	23
6.1.2 Фильтр высоких частот . . . . .	23
6.1.3 Полосовый фильтр . . . . .	24

6.2 Эквалайзер . . . . .	25
<b>7 Пространственные и модуляционные эффекты</b>	<b>26</b>
7.1 Хорус, Фленджер, Фазер . . . . .	26
7.2 Эхо . . . . .	28
7.3 Реверберация . . . . .	29
<b>Синтез звука</b>	<b>31</b>
<b>8 Аддитивный синтез звука</b>	<b>32</b>
<b>9 Субтрактивный синтез звука</b>	<b>33</b>
<b>10 Частотной модуляция</b>	<b>34</b>
10.1 Генератор, управляемый кодом . . . . .	34
10.2 Частотная модуляция . . . . .	35
10.3 Синтез . . . . .	35
<b>11 Нелинейный синтез звука</b>	<b>37</b>
<b>12 Таблицы волн</b>	<b>38</b>
12.1 WT-синтез . . . . .	38
12.2 Сэмплерные синтезаторы . . . . .	39
<b>13 Физическое моделирование</b>	<b>40</b>
<b>14 Звуковые платы</b>	<b>41</b>
14.1 Состав звуковой платы . . . . .	41
14.2 Блок записи и воспроизведения . . . . .	42
14.3 Блок синтезатора . . . . .	42
14.4 Блок DSP . . . . .	43
14.5 Блок интерфейсов . . . . .	43
14.6 Блок микшера . . . . .	44
<b>MIDI-интерфейс</b>	<b>45</b>
<b>15 MIDI-интерфейс</b>	<b>45</b>
15.1 Назначение MIDI . . . . .	45
15.2 Аппаратная реализация MIDI . . . . .	46
15.3 Разъем DIN-5 (СГ-5) . . . . .	47

<b>16 Протокол MIDI</b>	<b>48</b>
16.1 Адресация в MIDI . . . . .	48
16.2 Типы MIDI-сообщений . . . . .	49
16.2.1 Канальные сообщения о звуке . . . . .	49
16.2.2 Канальные сообщения о режиме . . . . .	50
16.2.3 Системные сообщения . . . . .	50
<b>17 Стандарты MIDI-систем</b>	<b>51</b>
17.1 General MIDI . . . . .	51
17.2 General Synthesis . . . . .	51
17.3 Extended General MIDI (XG) . . . . .	52
17.4 Все вместе . . . . .	53
<b>Запись и передача</b>	<b>54</b>
<b>18 Секвенсоры</b>	<b>54</b>
18.1 Простейшая студия . . . . .	54
18.2 Студия для записи «живого» исполнения . . . . .	54
18.3 Студия для многоканальной записи . . . . .	55
18.4 Студия для многоканальной записи с секвенсором . . . . .	56
18.5 Секвенсор . . . . .	57
18.5.1 Паттерновые (шаговые) . . . . .	57
18.5.2 Линейные . . . . .	58
<b>19 Маскирование</b>	<b>59</b>
19.1 Иллюстрации эффектов маскирования . . . . .	60
19.1.1 Граница слышимости в тишине . . . . .	60
19.1.2 Маскирование тоном . . . . .	60
19.1.3 Маскирование для различных частот . . . . .	61
19.1.4 Временное маскирование . . . . .	61
19.2 Сжатие звука . . . . .	62
<b>20 Формат MP3</b>	<b>63</b>
20.1 Алгоритм сжатия MP3 . . . . .	63
20.2 Схема кодера MP3 . . . . .	63
20.3 Режимы кодирования стерео . . . . .	64
20.4 Психоакустические форматы . . . . .	65
<b>Программирование звука</b>	<b>66</b>

<b>21 Основные программные интерфейсы</b>	<b>66</b>
<b>22 Программный интерфейс ММЕ</b>	<b>66</b>
22.1 Способы кодирования звука . . . . .	67
22.2 Формат потока . . . . .	68
22.3 Структура потока . . . . .	68
22.4 Системные особенности . . . . .	69
22.4.1 Несколько процессов . . . . .	69
22.4.2 Wave Mapper . . . . .	69
22.4.3 Устройства . . . . .	70
22.5 Алгоритм взаимодействия . . . . .	72
<b>23 Программный интерфейс DirectSound</b>	<b>74</b>
23.1 Назначение, структура, особенности . . . . .	74
23.2 Аппаратная поддержка . . . . .	74
23.3 Звуковые буферы . . . . .	75
23.3.1 Аппаратные и программные . . . . .	75
23.3.2 Первичный и вторичные . . . . .	76
23.4 Уровни взаимодействия . . . . .	80
23.5 Наборы свойств . . . . .	81
23.6 Идентификация устройств . . . . .	81
23.7 Системные особенности . . . . .	81
23.8 Алгоритм взаимодействия . . . . .	82
23.8.1 Воспроизведение . . . . .	82
23.8.2 Запись . . . . .	83
<b>Предметный указатель</b>	<b>84</b>

# Введение

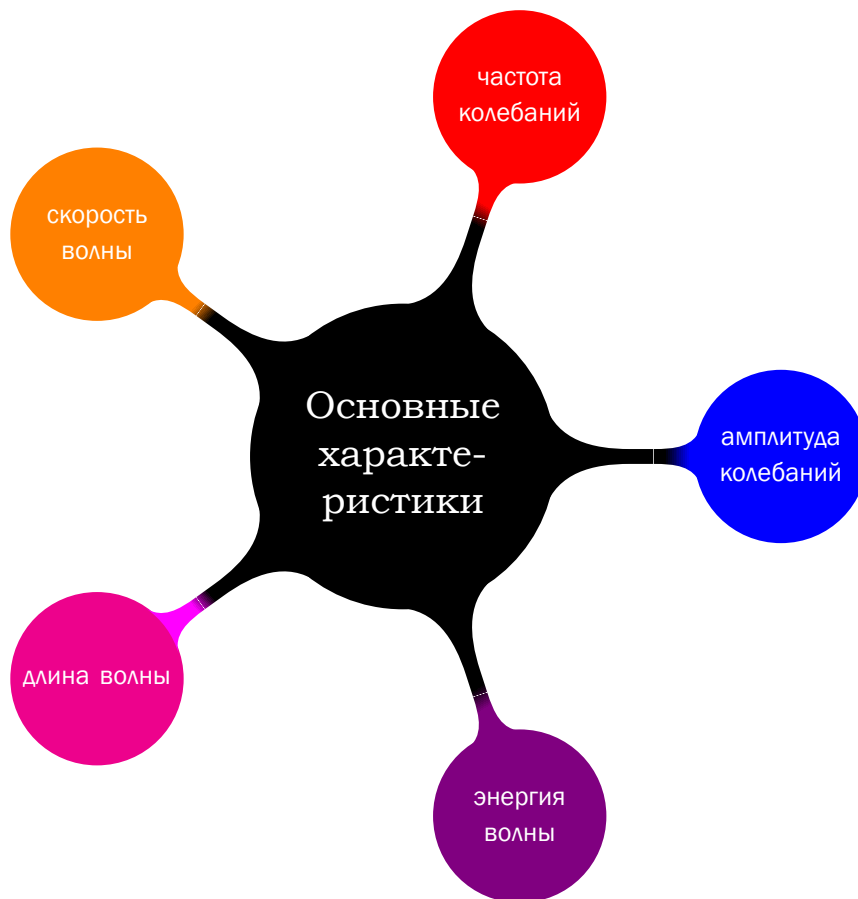
Методичка основана на курсе лекций, который читает О. В. Казанцев в рамках курса лекций «Методы и средства Мультимедиа». Эта вторая версия лекций, в ней исправлен ряд опечаток, немного изменена верстка.

Здесь нужно несколько слов самого О. В. Казанцева.

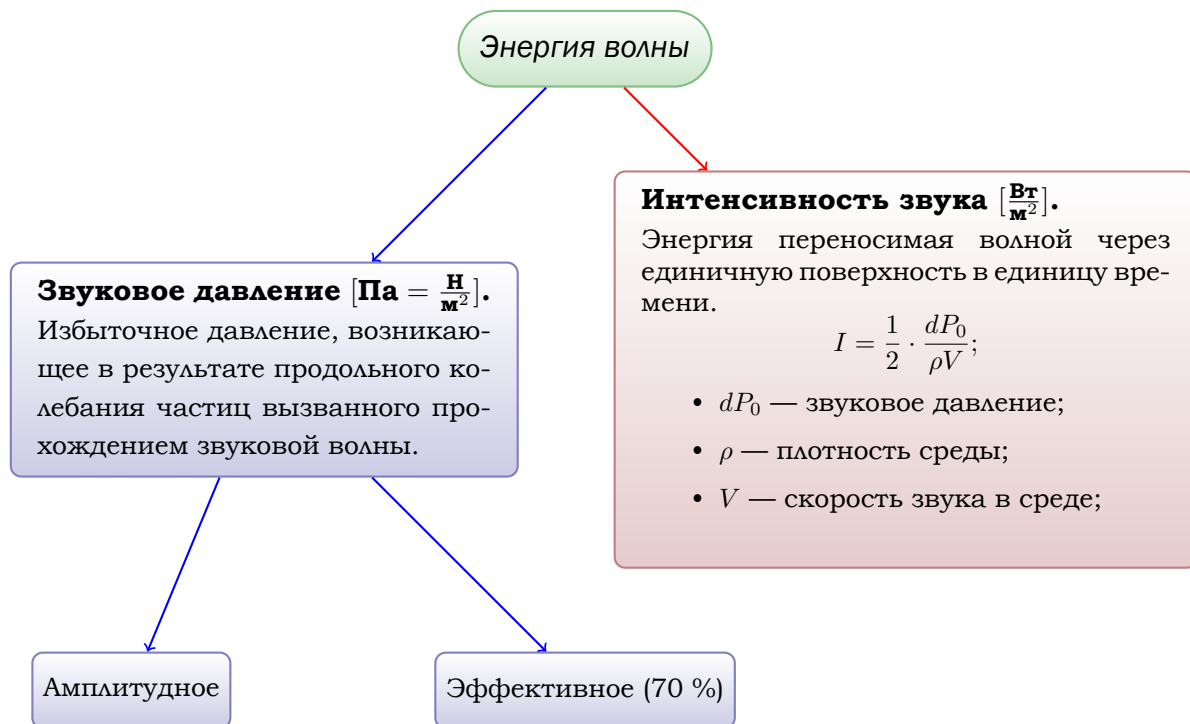
# 1. Основные сведения о звуковых волнах

## 1.1. Характеристики

**Звуковая волна** — процесс распространения в сплошной среде объемных деформаций.



Забавно, заметить, если длинна волны больше расстояния между ушами человека, то он не сможет определить откуда идет звук.



**Порог слышимости** — минимальное значение интенсивности звука, воспринимаемым человеческим ухом.

Порог слышимости зависит от частоты звуковой волны. Минимальное значение лежит в частоте 2 кГц и составляет  $10^{-12} [\frac{Вт}{м^2}]$ .

**Порог болевого ощущения** — интенсивность звука вызывающего болевые ощущения.

Порог болевого ощущения **не зависит** от частоты звуковой волны. Значение составляет  $10 [\frac{Вт}{м^2}]$ .

## 1.2. Закон Вебера-Фехнера для звука

**Теорема 1** (Закон Вебера-Фехнера). *Слух одинаково оценивает равные относительные изменения силы звука.*

## 1.3. Логарифмическая шкала

$$L = \log \frac{I}{I_0}$$

- $L$  — интенсивность в Беллах;
- $I$  — интенсивность;
- $I_0$  — порог.



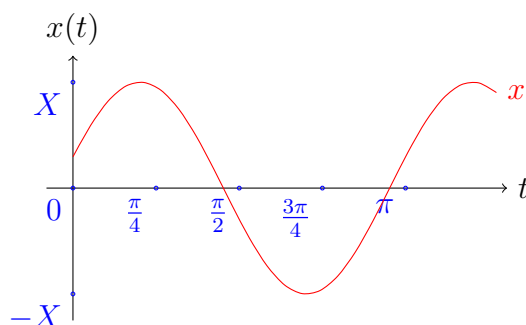
## 1.4. Примеры

Шум	Громкость
Фоновый	10 дБ
Транспорт	70 дБ
Оркестр	90 дБ
Наушники	100 дБ
Реактивный двигатель	120 дБ
Болевой порог	130 дБ

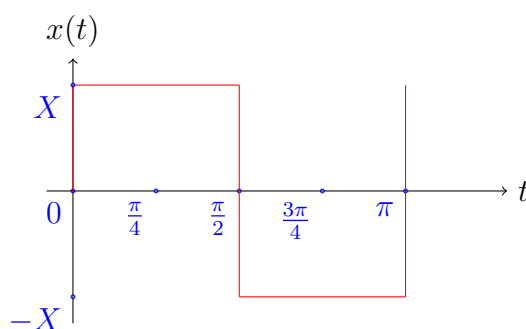
## 1.5. Спектральное представление звука

Звук представим в виде спектра:

$$x(t) = X \cdot \sin(2 \cdot \pi \cdot f \cdot t + \varphi);$$



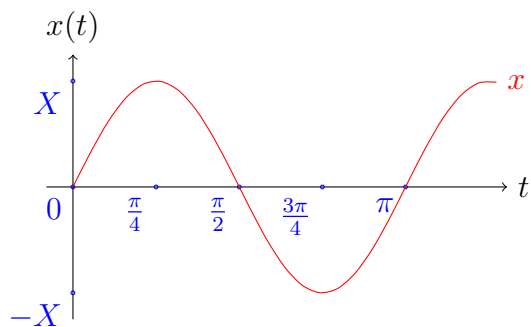
Даже белый шум представим как сумма гармоник, но не синусоидальных, а интегральных.



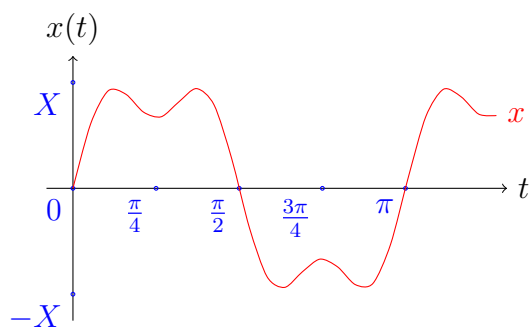
$$Y(x) = \sum_{n=0}^{\infty} \frac{1}{2k+1} \cdot A \cdot \sin(2 \cdot (2n+1) \cdot \pi F x)$$

- $A$  — амплитуда;
- $f$  — частота.

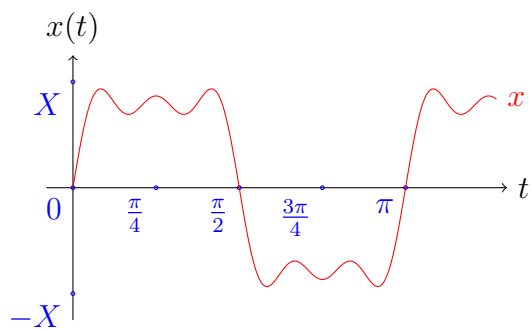
Изобразим первый член ряда:



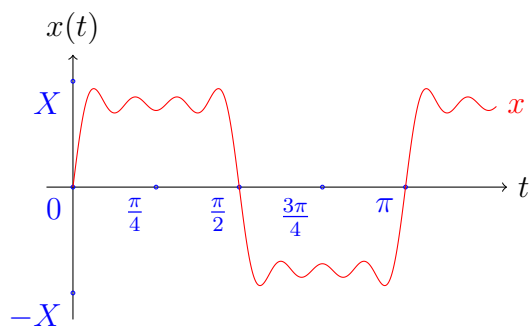
Изобразим два члена ряда:



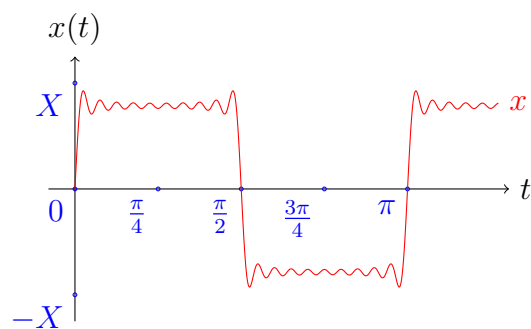
Изобразим три члена ряда:



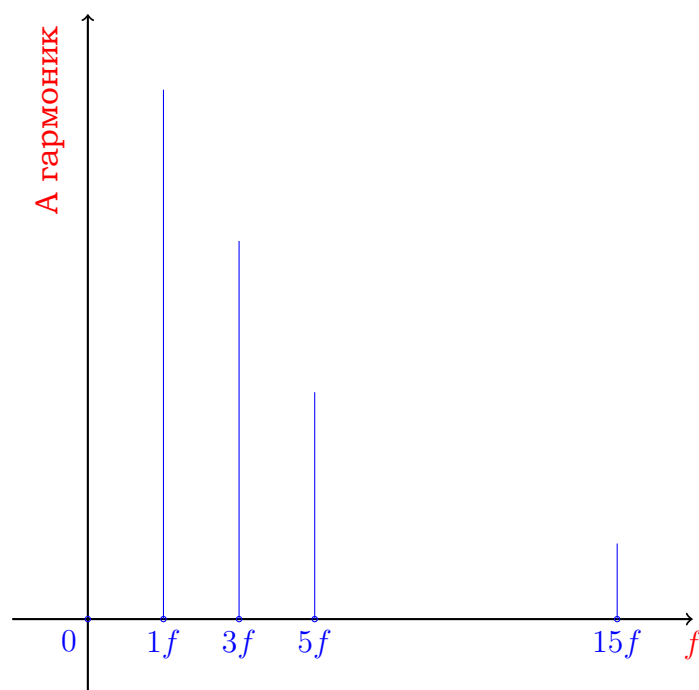
Изобразим четыре члена ряда:



Изобразим десять членов ряда:



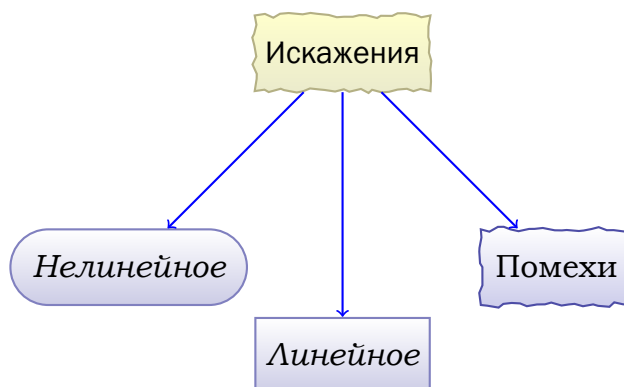
Наблюдаемые горбы (осцилляции) называются **эффектом Гиббса**.  
Таким образом можно изобразить звуковой спектр на графике:



# Искажения и эффекты

## 2. Линейные искажения

**Искажение** — изменение формы сигнала после прохождения через электроакустические устройства.



### 2.1. Линейное искажение

**Линейное искажение** — искажение, которое проявляется в неодинаковом усилении передачи или воспроизведении различных гармоник, составляющих звуковой сигнал, независимо от их уровня.

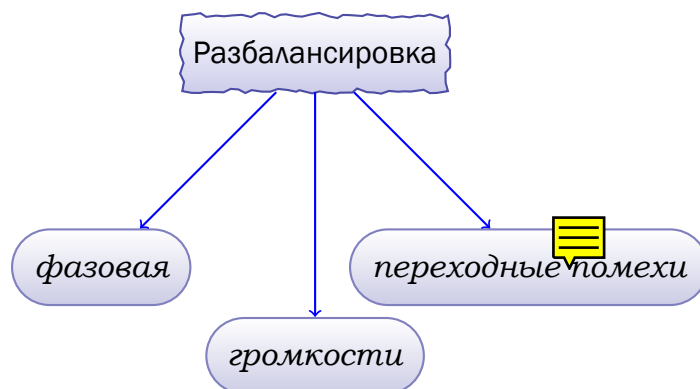
Для случая спектра происходит изменение коэффициентов ряда Фурье. Изменяется только амплитуда набора гармоник.

При измерении искажений используется логарифмическая формула:

$$Y(f) = 20 \lg \frac{Xf}{Xf_0}$$

### 2.1.1 Искажения в многоканальных системах

Искажения в этих системах чаще всего симметричны. Выделяют 3 вида разбалансировки системы.



**Разбалансировка громкости** — явление, при котором, какой-то канал многоканальной системы передает звук громче.

**Фазовая разбалансировка** — явление, при котором, различные каналы многоканальной системы передают сигнал с разной скоростью.

**Переходные помехи** — явление, при котором, сигнал из одного канала многоканальной системы проникает в другой канал.

### 3. Нелинейные искажения. Помехи и шумы

**Нелинейное искажение** — заключается в изменении набора гармоник, обычно выше определенной частоты.

Особенность: степень проявления зависит от амплитуды входного сигнала.

Коэффициент гармоник:

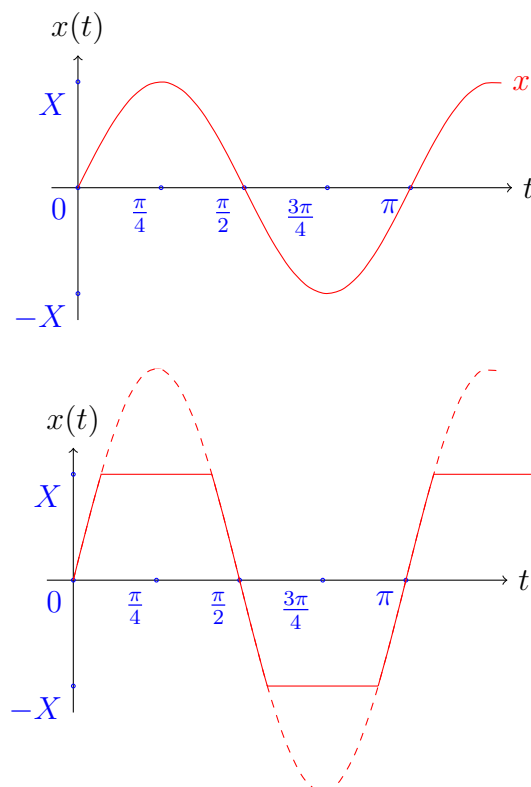
$$K = \frac{\sqrt{x_1^2 + x_2^2 + \dots + x_n^2}}{x_0} \cdot 100\%$$

- $x_0$  — амплитуда выходного сигнала;
- $x_1 \dots x_n$  — амплитуды наведенных гармоник.

#### 3.1. Примеры нелинейных искажений

##### 3.1.1. Перегрузка

Это искажение часто используется для «обогащения» звука, например для электрогитар.



##### 3.1.2. Интермодуляционные

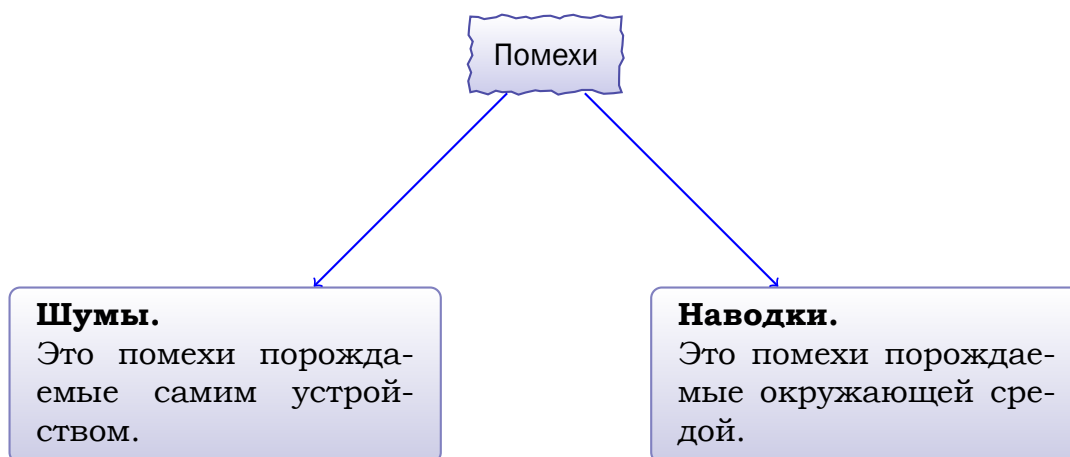
Более мощный низкочастотный сигнал вызывает амплитудную модуляцию у более слабого высокочастотного сигнала.

### 3.1.3. Биение

Биение — это появление разностной частоты при воздействии на нелинейную систему двух сигналов с близкими частотами.

## 3.2. Помехи

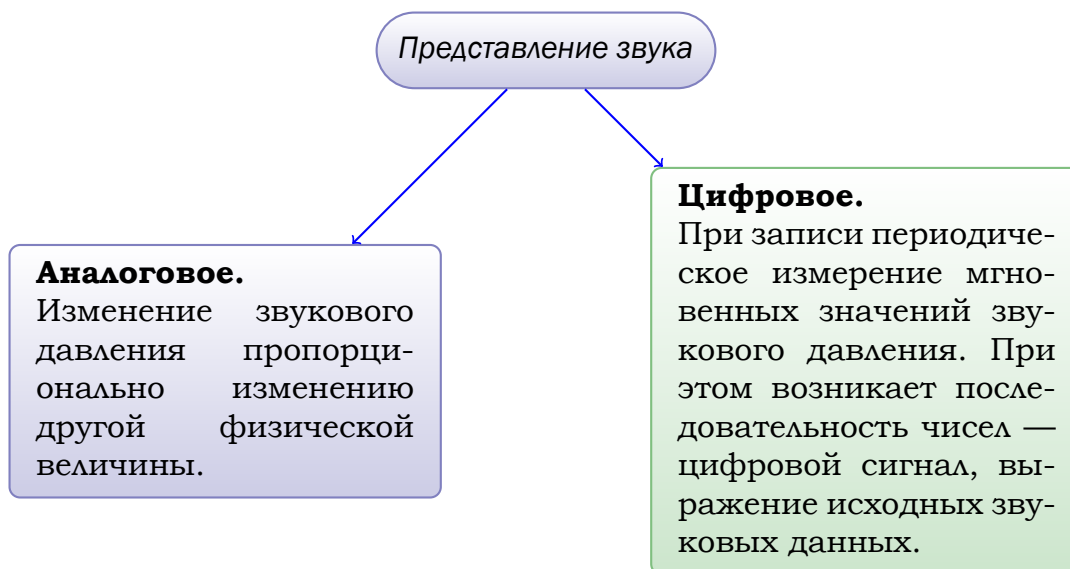
Помехи порождают в выходном сигнале гармоники, которые не зависят от входного сигнала.



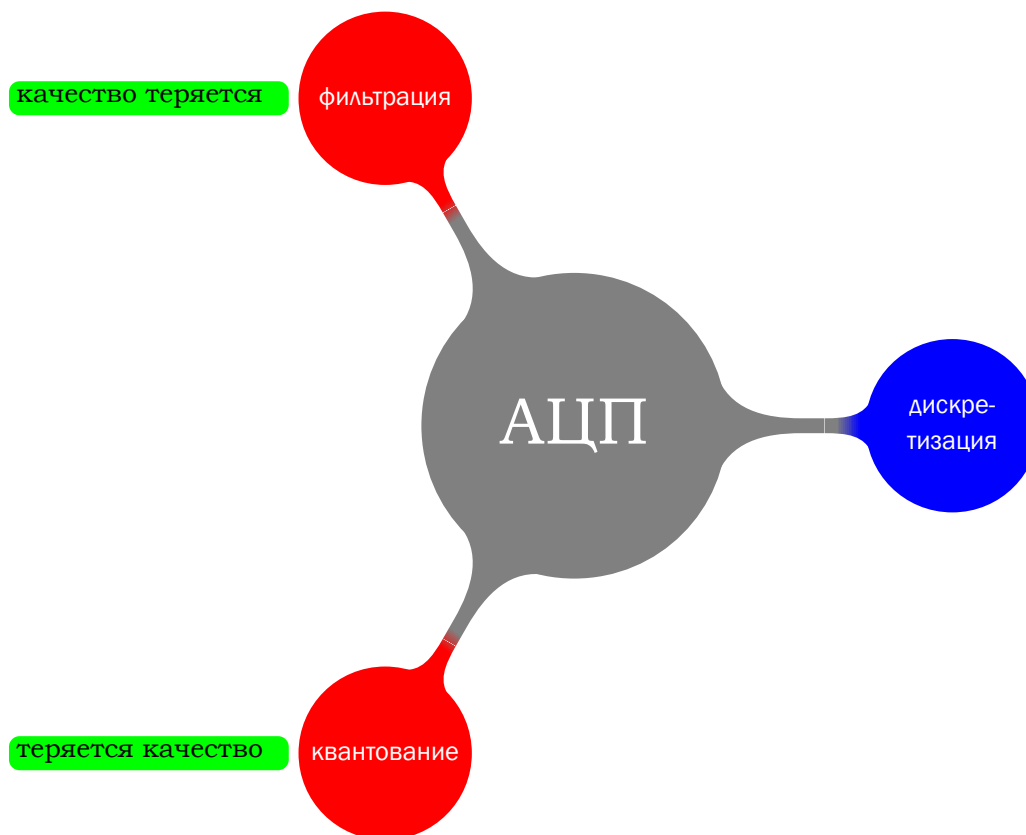
Уровень шумов и наводок можно оценить с помощью формулы:

$$U = 20 \cdot \log \frac{U_n}{U_{\max}}.$$

## 4. Цифровой способ представления звука




### 4.1. Аналогово-цифровое преобразование



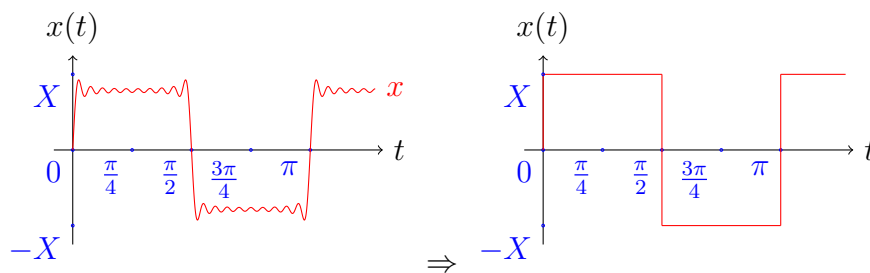


### 4.1.1. Фильтрация

**Фильтрация** — отсечение всех сигналов, частоты  которых выше чем частота, которая требуется для преобразования.

Граница на которой фильтр срезает частоту чаще всего имеет ширину 2 кГц.

**Оверсэмплинг** — фильтрация на высокой частоте.



### 4.1.2. Дискретизация

**Дискретизация** — процесс взятия отсчетов непрерывного во времени сигнала в равно отстоящие друг от друга временных точках. **Интервал дискретизации** — интервал времени через который производится взятие отсчетов.

**Теорема 2** (Котельникова-Найквиста-Шенона). Сигнал, спектр частот, которого занимает область частот до  $F_{\max}$  Может быть **полностью** представлен дискретными отсчетами с интервалами не превышающими

$$\frac{1}{2F_{\max}}$$

- частота 44.1 кГц — стандарт для Audio CD;
- частота 48.0 кГц — профессиональный стандарт.

### 4.1.3. Квантование

**Квантование** — процесс измерения мгновенных уровней сигнала, полученных в результате дискретизации, с точностью, ограниченной количеством разрядов, используемых для записей значений.

Есть 3 стандарта хранения отсчетов:

- 1 бит — 256 уровней сигналов;
- 16 бит — 65536 уровней сигналов;
- 32 бит — 4294967296 уровней сигналов.

С явлением квантования связано понятие **шум квантования**. Он рассчитывается по формуле:

$$P = -(6N + 1.8), \text{ где } N — \text{ количество разрядов.}$$

## 5. Динамическая обработка звука

Динамическая обработка служит для изменения **динамического диапазона сигнала**.

**Динамический диапазон сигнала** — разница между самым громким и самым тихим звуком.

Чем шире диапазон, тем больше разница между самым тихим и самым громким звуком и наоборот. Динамические процессоры в основном подключаются «в разрыв».

Основные виды динамических обработок это:



## 5.1. Компрессор и лимитер

Задача компрессора состоит в том, что бы **сжимать динамический диапазон** обрабатываемого сигнала. Компрессор понижает уровень громких звуков и повышает уровень тихих.


Лимитер тоже сжимает динамический диапазон, но в отличие от компрессора делает это жестко — **не позволяет** сигналу **превышать** определенный уровень.

Основные параметры:

- **Порог**<sup>1</sup> — уровень сигнала, при котором срабатывает обработка.
- **Отношение**<sup>2</sup> — величина уменьшения сигнала при превышения порога.

Например, 2 : 1 означает, что при превышении порога сигнал должен быть уменьшен вдвое. У лимитера этот параметр не регулируется (бесконечность).

Компрессор с отношением  $n : 1$ ,  $n \geq 10$  работает как лимитер.

- **Атака**<sup>3</sup> — скорость срабатывания компрессора.
- **Затухание**<sup>4</sup> — скорость восстановления компрессора.
- **Усиление**<sup>5</sup> — уровень общего усиления сигнала на выходе.  Задается в децибелах, отражающих увеличение или ослабление сигнала. Сигнал не должен превышать порог срабатывания.
- **Жесткое или мягкое колено**<sup>6</sup> — определяет жесткость срабатывания (отношение достигает своего значения сразу или плавно).

---

<sup>1</sup>Threshold.

<sup>2</sup>Ratio.

<sup>3</sup>Attack.

<sup>4</sup>Release.

<sup>5</sup>Gain.

<sup>6</sup>Hard knee, soft knee.

## 5.2. Гейт и экспандер

Это обработка, противоположная лимитеру. Если лимитер отсекает самые громкие звуки, то гейт отсекает самые тихие.

Гейт пропускает только те сигналы, уровень которых превосходит заданный порог, остальные отбрасывает. В основном предназначен **для борьбы с шумами** и паразитными сигналами (звук соседнего барабана).

Основные параметры:

- **Порог**<sup>7</sup> — уровень сигнала, при котором срабатывает обработка.
- **Отношение**<sup>8</sup> — определяет насколько должен уменьшаться сигнал, уровень которого ниже порога. Чаще всего полное ослабление.

Например, 40 Дб — это практически полное ослабление.

- **Атака**<sup>9</sup> — скорость срабатывания компрессора.
- **Затухание**<sup>10</sup> — скорость восстановления компрессора.

**Экспандер** прибор очень похожий на гейт. Отличие состоит в том, что гейт *понижает сигнал ниже порога на определенную величину*, а экспандер понижает сигнал в заданном отношении.

То есть, если у задано отношение 2 : 1, то при недостатке 10 дБ, сигнал будет понижен на 20 дБ, а если сигнал недостает 2 дБ, то сигнал будет понижен на 4 дБ. Соответственно у экспандера отношение называется ratio.

---

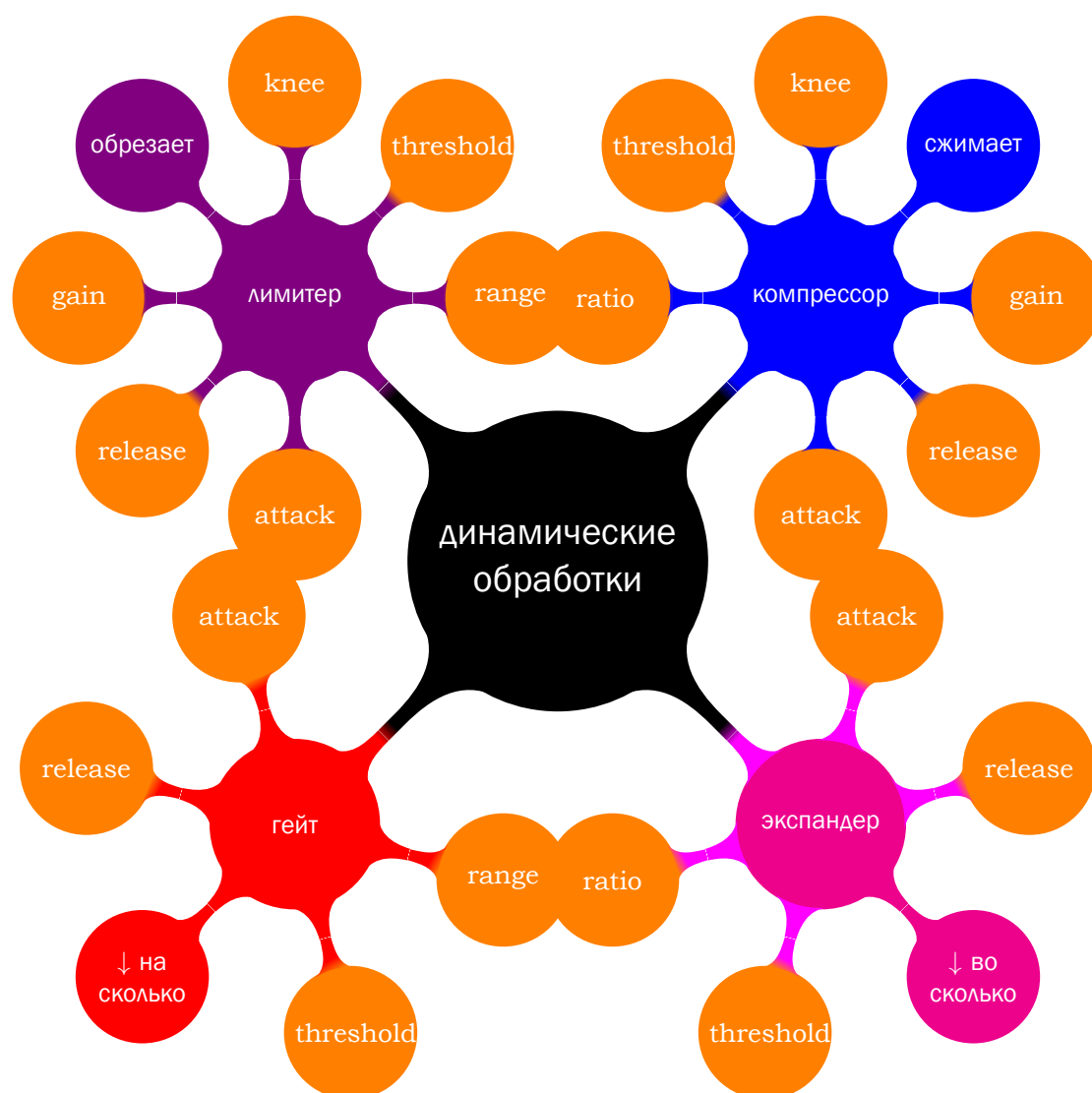
<sup>7</sup>Threshold.

<sup>8</sup>Range.

<sup>9</sup>Attack.

<sup>10</sup>Release.

Представим сказанное в виде mind map.

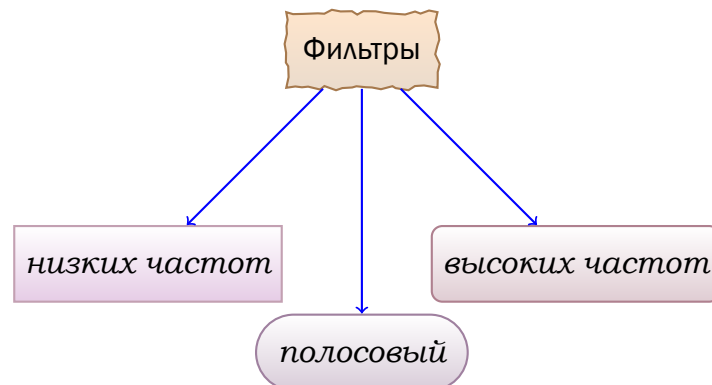


## 6. Частотная коррекция звукового сигнала

Основные виды обработок:

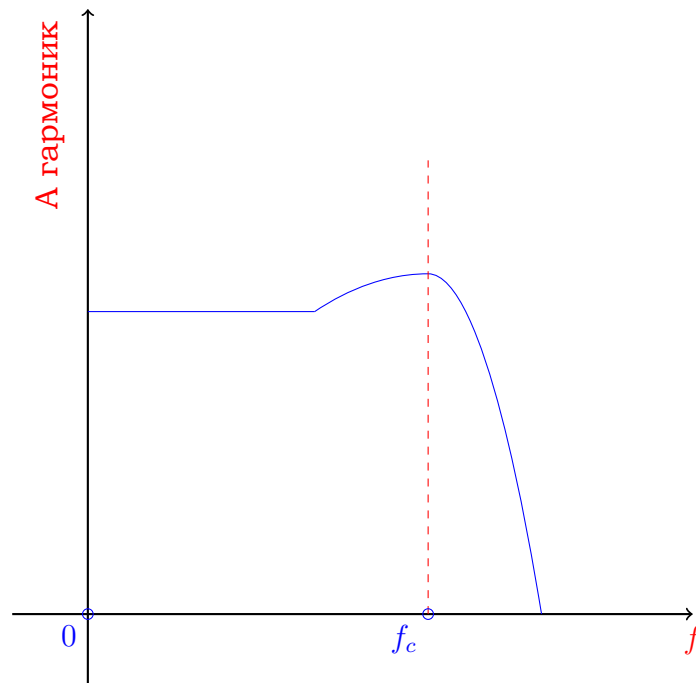
- ФНЧ (LPF);
- ФВЧ (HPF);
- Полосовый фильтр(BPF);
- Режекторный (Notch filter) — полосовый фильтр, работающий в минус;
- Графический эквалайзер;
- Параметрический эквалайзер;
- Параграфический эквалайзер — гибрид параметрического и графического эквалайзера.

### 6.1. Частотные фильтры



### 6.1.1. Фильтр низких частот

Фильтр низких частот<sup>11</sup> — отфильтровывает все, что ниже заданной частоты.



$f_c$  — частота среза

### 6.1.2. Фильтр высоких частот

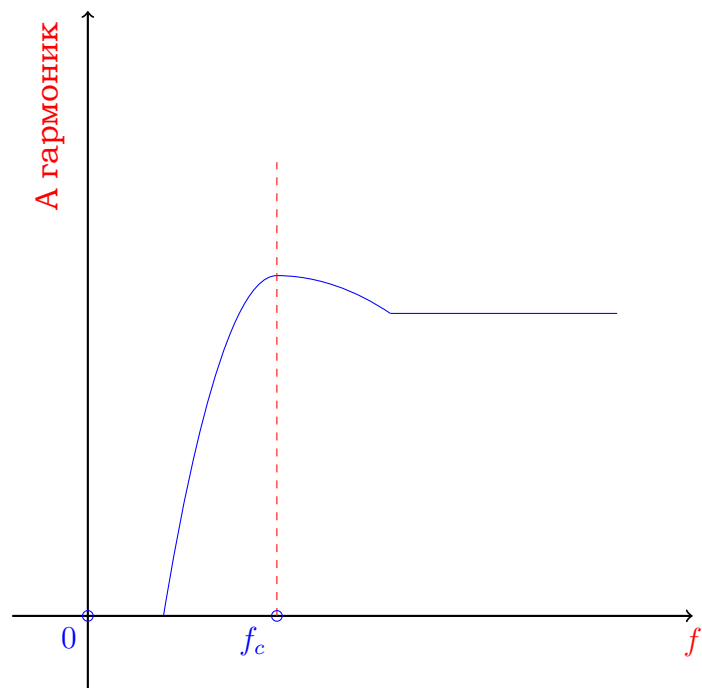
Фильтр высоких частот<sup>12</sup> — отфильтровывает все, что выше заданной частоты. Часто используется для подавления частот, которые не воспринимаются человеком. Эти частоты перегревают устройства.

---

<sup>11</sup>LPF — Low Path Filter.

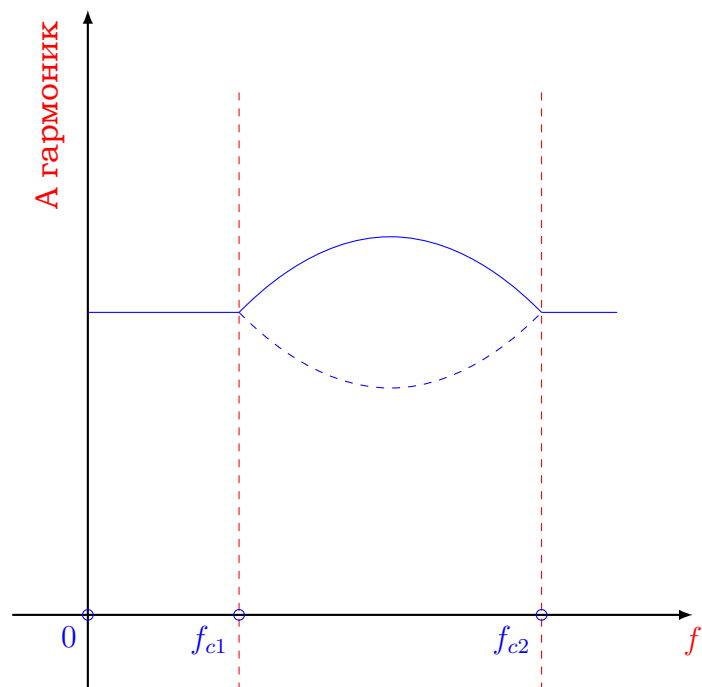
<sup>12</sup>HPF — High Path Filter.





$f_c$  — частота среза

### 6.1.3. Полосовый фильтр

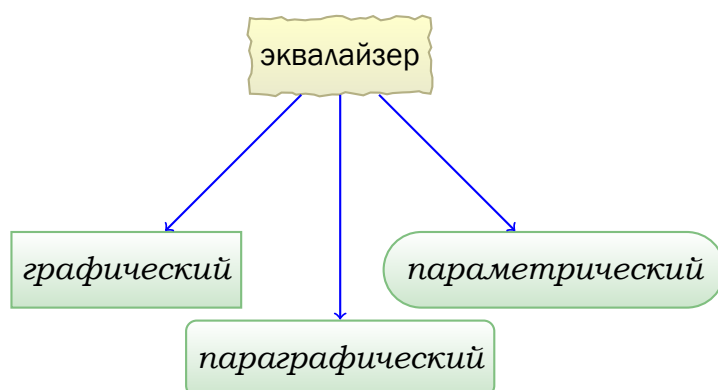


Частотный фильтр, в случае понижения, не срезает полосу до нуля. Он только уменьшает интенсивность определенных частот.

## 6.2. Эквалайзер

**Эквалайзер** — устройство или компьютерная программа, которая позволяет выравнивать амплитудно-частотную характеристику звукового сигнала, то есть корректировать его (сигнала) амплитуду избирательно, в зависимости от частоты. Эквалайзер обычно используется для частотной компенсации прибора.

Можно воспринимать эквалайзер как совокупность нескольких фильтров.



**Графический** — эквалайзер с жестко определенным набором фильтров.

Графический эквалайзер имеет определенное количество регулируемых по уровню частотных полос, каждая из которых характеризуется постоянной рабочей частотой, фиксированной шириной полосы вокруг рабочей частоты, а также диапазоном регулировки уровня (одинаковый для всех полос).

**Параметрический** — эквалайзер с кастомизированным набором фильтров. Параметрический эквалайзер удобно использовать для исправления конкретного дефекта.

Каждая полоса параметрического эквалайзера имеет три основных регулируемых параметра:

- **центральная**<sup>13</sup> **частота** в герцах (Гц);
- **добротность**<sup>14</sup> — безразмерная величина;
- **уровень усиления** или **ослабления** выбранной полосы в децибелах (дБ).

**Параграфический** — эквалайзер графического типа с регулировкой добротности.

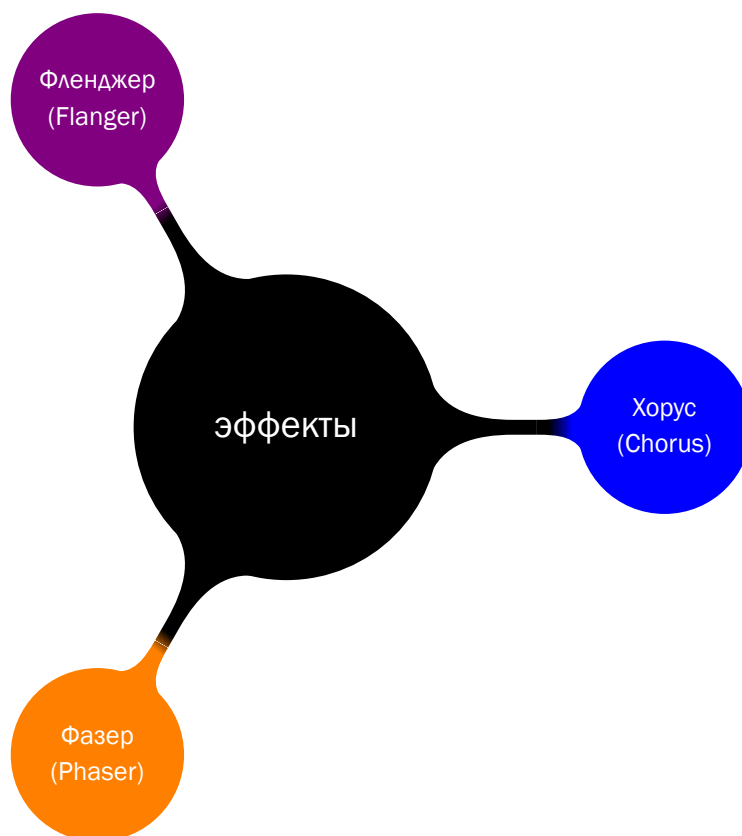
---

<sup>13</sup>Рабочая.

<sup>14</sup>Ширина рабочей полосы вокруг центральной частоты, обозначается буквой «Q».

## 7. Пространственные и модуляционные эффекты

### 7.1. Хорус, Фленджер, Фазер



Модуляционные эффекты основаны на задержке сигнала, вызывающей эффект изменения высоты тона.

Для хора, фленджера, фазера задержка очень маленькая, порядка десятков миллисекунд. Задержка сигнала может изменяться во времени. Модулируется эта величина при помощи **низкочастотного генератора**.

Эффект	Задержка [мс]
Фазер	1 – 6
Фленджер	7 – 15
Хорус	15 – 90

Основные параметры:

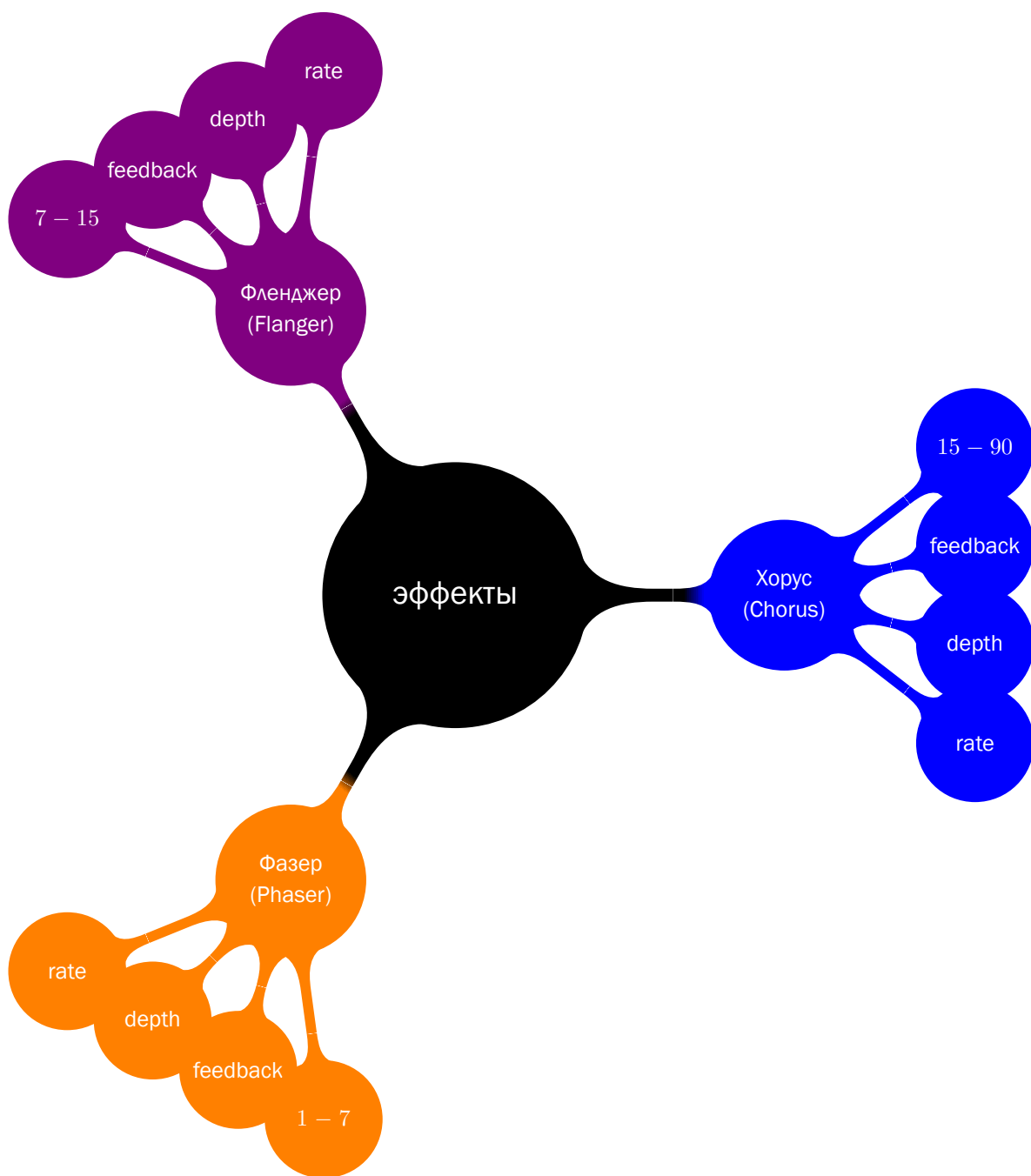
- **частота**<sup>15</sup> — частота модулирующего генератора;
- **глубина**<sup>16</sup> — величина отклонения тона;

---

<sup>15</sup>Rate.

<sup>16</sup>Depth.

- **обратная связь**<sup>17</sup> — величина обработанного сигнала, подаваемого на вход. Определяет число повторов.



---

<sup>17</sup>Feedback.

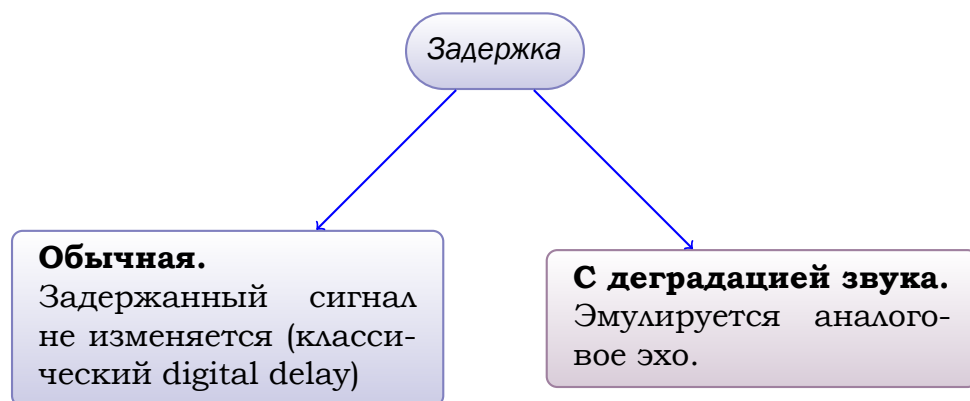
## 7.2. Эхо

**Эхо (Delay)** — задержка исходного сигнала с повтором.

Существует множество алгоритмов:

- одиночный повтор;
- многократный повтор;
- повтор с изменением панорамы;
- повтор с разными величинами задержки для правого и левого каналов.

Величина задержки очень большая от 200 мс до нескольких секунд.



Основные параметры:

- **время**<sup>18</sup> — интервал времени между повторами;
- **обратная связь**<sup>19</sup> — величина обработанного сигнала, подаваемого на вход. Определяет число повторов.

---

<sup>18</sup>Time.

<sup>19</sup>Feedback.

## 7.3. Реверберация

**Реверберация** — это подражание естественным отражениям звуковых волн в помещении. Реверберация применяется для имитации акустики окружающего пространства. Представляет из себя совокупность большого числа задержек исходного сигнала с разным временем.

Алгоритмы формирования таких задержек достаточно сложны и зависят от того, что моделируется.

Время задержки варьируется от десятком до сотен миллисекунд. Задержка как таковая на слух не ощущается (в отличие от эха). Воспринимается как придание некоторого объема звуковому сигналу.

Основные параметры:

- Тип и размер помещения<sup>20</sup> — определяет алгоритм реверберации и величину задержек.

Основные типы:

- ▶ room,
  - ▶ hall,
  - ▶ stadium,
  - ▶ cathedral,
  - ▶ bathroom,
  - ▶ plate,
  - ▶ spring и т.д.
- Время<sup>21</sup> – время звучания реверберационного хвоста (не путать с временем задержки, как у delay).
  - Задержка начала<sup>22</sup> — определяет расстояние от источника звука до ближайшей стены, то есть время, через которое начнется реверберация.

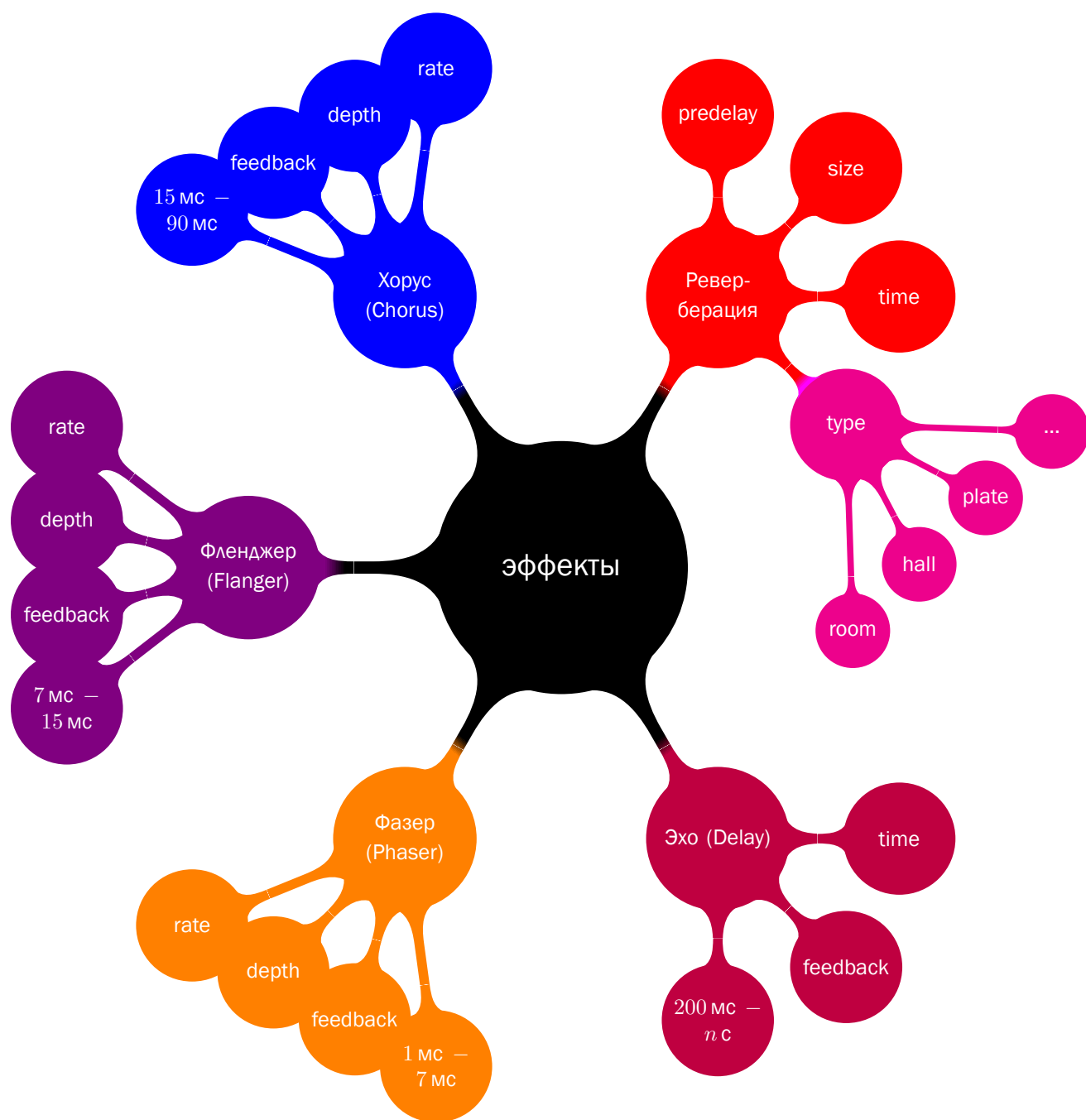
В зависимости от алгоритма могут еще задаваться параметры смешивания отраженных сигналов и их деградацию, обусловленную поглощающими материалами помещения.

---

<sup>20</sup>Room size/type.

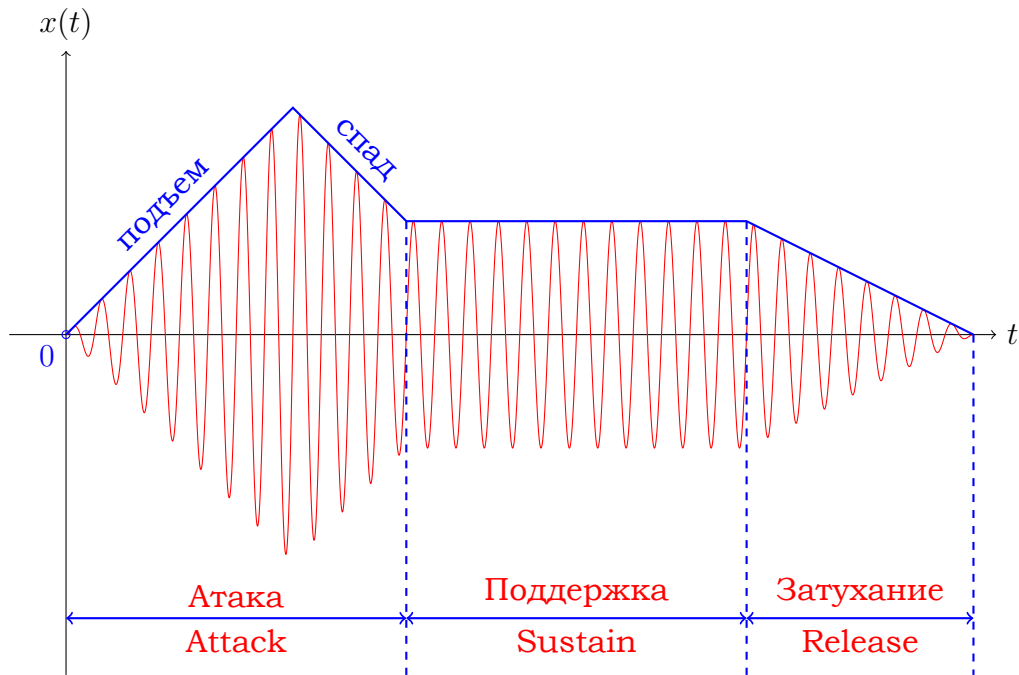
<sup>21</sup>Time.

<sup>22</sup>Predelay.



## Синтез звука

Звук любого инструмента имеет вид:



**Атака** — начальная фаза образования звука.

**Поддержка** — фаза образования звука, следующая после атаки. Во время поддержки формируется ощущение высоты звука.

**Затухание** — участок уменьшения сигнала.

- 1) Фаза атаки наиболее ярко выражена для барабанов. Момент удара палочкой, как раз, и есть фаза атаки.
- 2) Фаза затухания у различных инструментов может быть, как очень короткой (орган), так и очень длинной (арфа).

Некоторые фазы могут отсутствовать.



## 8. Аддитивный синтез звука

Результирующий звук формируется путем сложения нескольких исходных звуковых волн. Формируется  $N$  гармоник с частотами  $f_1(t) \dots f_n(t)$ , и амплитудами  $A_1(t) \dots A_n(t)$ . Гармоники не обязательно должны быть кратными. И гармоники и амплитуды зависят от времени. Эти гармоники складываются. Получаем синтезируемый сигнал.

Для получения всех точек звукоряда нужно несколько сотен составляющих гармоник.

Для генерации синусоидального сигнала используются ГУК — генераторы, управляемые кодом.

## 9. Субтрактивный синтез звука

Метод субтрактивного синтеза звука является вторым по популярности. Синтез состоит из нескольких этапов:

- 1) Создается сигнал богатый гармониками.

Данный сигнал должен содержать максимальное количество гармоник. Чаще всего в качестве такого сигнала используют последовательность коротких прямоугольных импульсов. Иногда, чтобы еще сильнее обогатить сигнал используют пилообразные и треугольные импульсы.

- 2) Выбор нужного набора гармоник из исходного сигнала.

Интересно заметить, что подобный механизм «звукообразования» используется при формировании речи человека.

## 10. Частотной модуляция

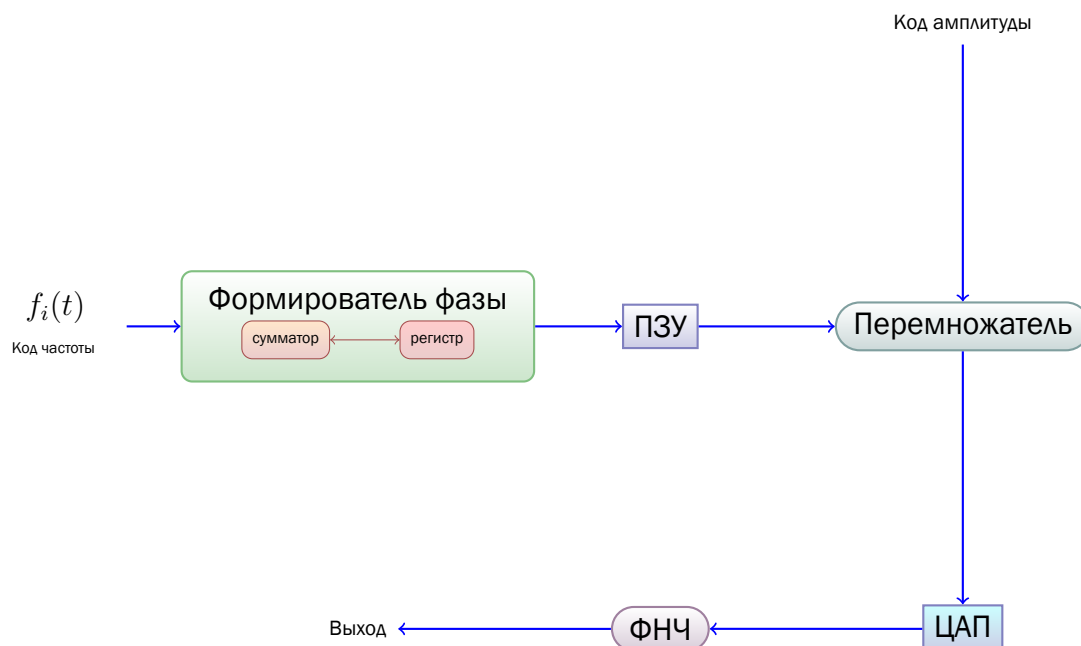
### 10.1. Генератор, управляемый кодом

Напомним, что **ГУК**<sup>23</sup> — генераторы, управляемые кодом. Он используется для генерации синусоидального сигнала. Составляющие:

- формирователь фазы;
  - ▶ сумматор;
  - ▶ регистр;
- ПЗУ;

В ПЗУ записаны отсчеты функции  $(\sin x)$  для одного периода. Точки одинаково удалены по времени.

- перемножитель параллельных кодов;
- ЦАП — цифро-аналоговый преобразователь;
- ФНЧ — фильтр низких частот;



<sup>23</sup>NCO — Numeric Controlled Oscillator.

## 10.2. Частотная модуляция

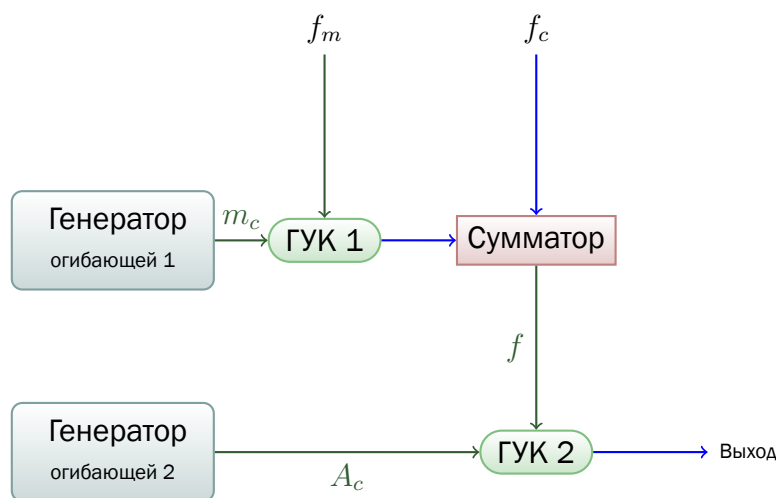
**Частотная модуляция** — процесс модуляции, при котором мгновенная частота несущего сигнала изменяется под воздействием модулирующего сигнала, а отклонение несущей частоты от среднего значения пропорционально амплитуде модулирующего сигнала.


$$x(t) = A_c \cdot \sin(2\pi t [f_c + m_f \cdot \sin(2\pi f_m t)])$$

- $A_c$  — амплитуда несущей частоты;
- $f_c$  — несущая частота;
- $m_f$  — индекс частотной модуляции;
- $f_m$  — модулирующая частота.

Изменяя только  $m_f$  можно варьировать спектр  $x(t)$  в широких пределах. Это используется для ЧМ-синтеза.

## 10.3. Синтез



Минимальное число ГУК для такой схемы должны равняться 6. 

Важно заметить, что на аналоговом механизме такая схема работать не будет.

**Достоинство:**

Универсальность. Можно получить любое звучание.

**Недостаток:**

Сложность реализации.

ЧМ-синтез был основным методом синтеза в середине 80-х годов XX века.

Это метод реализован в Native Instruments FM8.

## 11. Нелинейный синтез звука

Нелинейный синтез (и как частный случай, метод волновой формы) часто выступает как дополнение к субтрактивному синтезу.

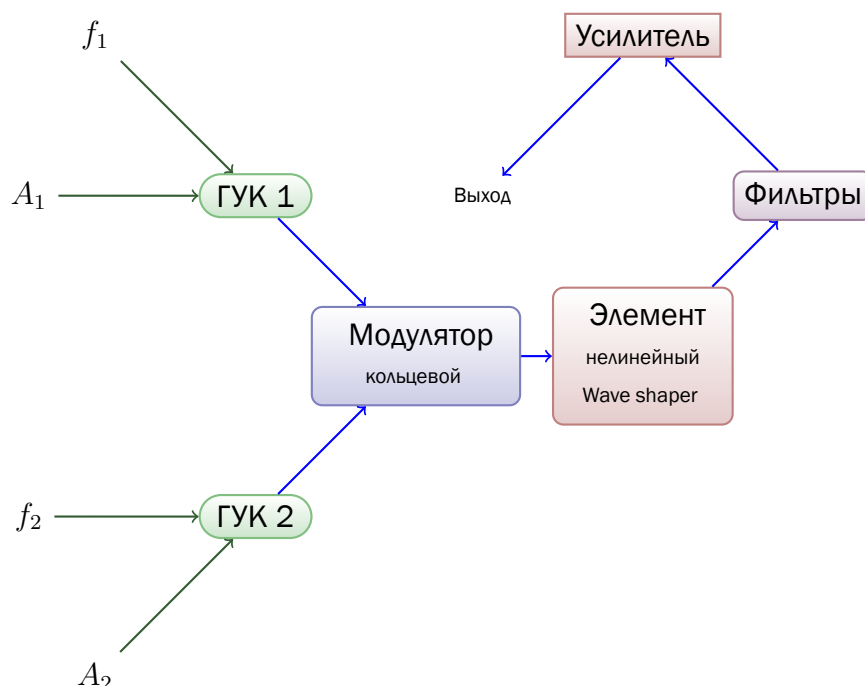
Для синтеза одного инструмента используется сигнал одного генератора. Сигнал инструмента получается в результате нелинейных искажений гармоник генератора. Синусоидальный сигнал от генератора пропускают через нелинейный элемент и на выходе получают сигнал с той же частотой, но с другой фактурой (амплитуда, «рисунок») гармоник. Таким образом можно получить спектры, характерные для тех или иных инструментов.

### Достоинство:

Простота.

### Недостаток:

Нельзя раздельно управлять амплитудой и спектром.



**Кольцевая модуляция** — умножение спектров сигналов. В данном контексте это сложение генераторов.

## 12. Таблицы волн

Этот метод синтеза также называют WT-синтез<sup>24</sup>. На данный момент наиболее популярен.

### 12.1. WT-синтез

**WT-синтез** — метод синтеза, основанный на воспроизведении заранее записанного в цифровом виде звучания инструментов — сэмплов. Для изменения высоты звука сэмпл проигрывается с разной скоростью.

Атака, поддержка, затухание сэмплируются отдельно. Это необходимо, чтобы не изменился характер звучания.

В сложных (и дорогих) синтезаторах используется параллельное проигрывание нескольких сэмплов на одну ноту. Обычно такие сэмплы играют на разных уровнях громкости. Такой метод называется — **многослойным сэмплированием**.

Инструменты с короткой поддержкой семплируются полностью для всех вариантов звучания. Для большинства инструментов семплируется полностью только атака и затухание. Поддержка (вернее ее маленькая часть) семплируется всего лишь в нескольких наиболее отличимых вариантах, а при воспроизведении проигрывается в цикле и с различным ускорением.

**Достоинства:**

- реалистичность;
- простота.

**Недостатки:**

- жесткий набор сэмплированных тембров;
- большие объемы памяти для хранения сэмплов.

**MultiSample** — метод сэмплирования, при котором используется не одна волновая таблица, а последовательность в несколько десятков сэмплов.

Такой метод например, реализован в модуле NanoWave. Это модуль для Native Instruments Reactor.

---

<sup>24</sup>Wave Table.



## 12.2. Сэмплерные синтезаторы

Не надо путать WT-синтез с старым «сэмплерным методом»<sup>25</sup>. В сэмплерных синтезаторах атака и затухание реализуются predetermined сэмплом. Его длительность невозможно изменить. И только поддержка может быть произвольной длины.



<sup>25</sup>Его еще называют Sample Based синтез.



## 13. Физическое моделирование

**Синтез на основе физического моделирования** — метод синтеза, использующий математические модели образования звуков реальных музыкальных инструментов, для генерации волновых форм в цифровом виде.

Любой музыкальный инструмент состоит из:

- возбудитель — например, струна;
- резонатор — например, верхняя и нижняя деки гитары.

### Достоинство:

Реалистичность.

### Недостаток:

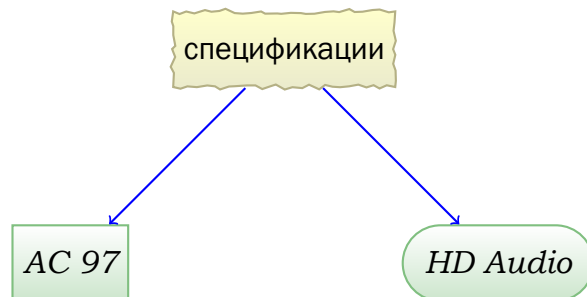
В алгоритм вшиваются конкретные инструменты, которые потом невозможно изменить.

Примеры программного обеспечения:

- IK Multimedia Amplitube X-Gear;
- Logana;

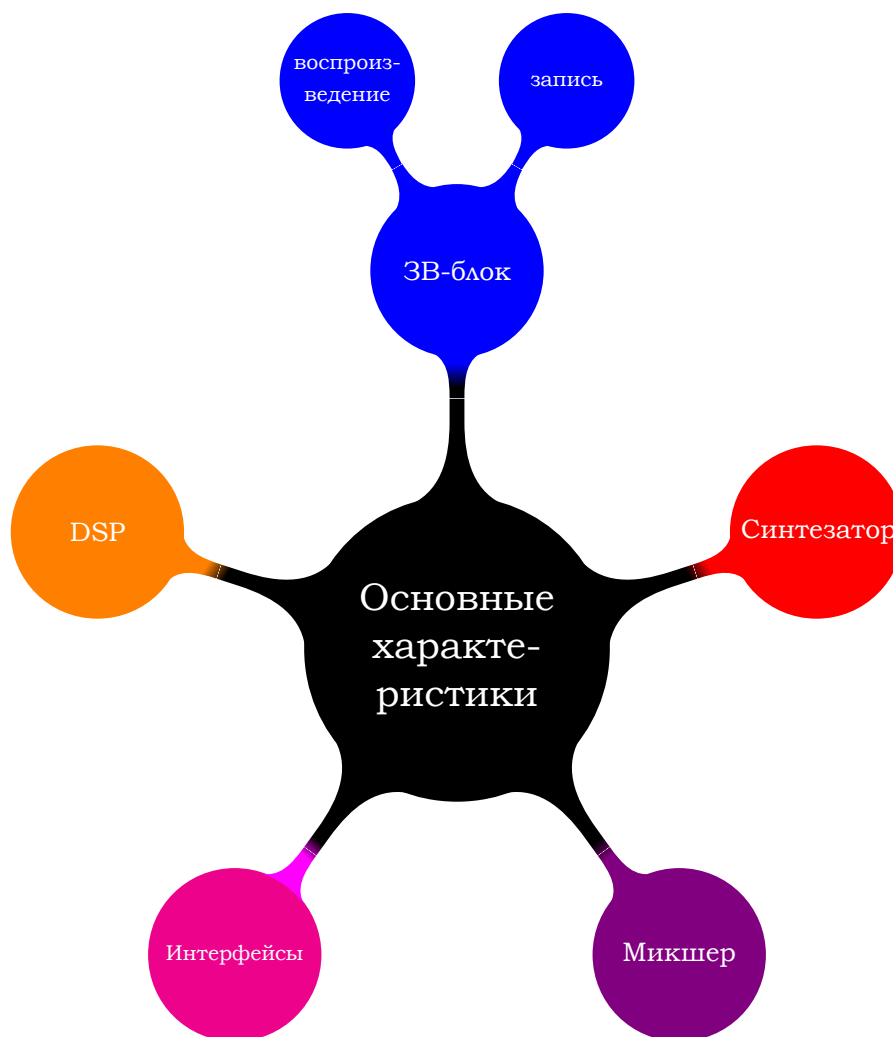
## 14. Звуковые платы

Для звуковых существует две спецификации:



Речь идет о цифро-аналоговых и аналогово-цифровых спецификациях.

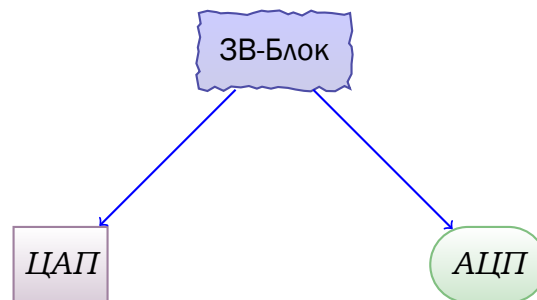
### 14.1. Состав звуковой платы



Большинство звуковых плат являются встроенными. Встроенные звуковые платы страдают задержкой звука. Для профессиональной записи используют внешние многоканальные платы.

## 14.2. Блок записи и воспроизведения

Блок записи и воспроизведения — самый важный блок звуковой платы. Он управляется своим собственным драйвером и не связан с блоком DSP и блоком синтезатора.



Важные части:

- цифро-аналоговый преобразователь (ЦАП);
- аналогово-цифровой преобразователь (АЦП).

Основные характеристики преобразователей:

- тип;
- разрядность.

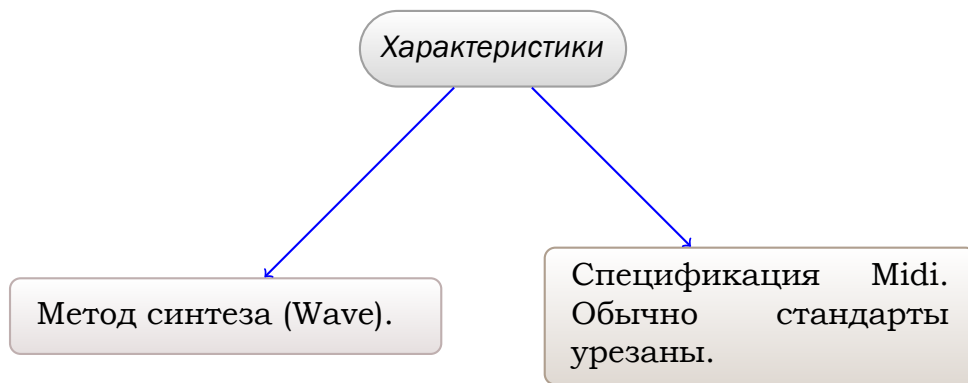
Важные характеристика самого блока:

- частота дискретизации;
- количество входных и выходных каналов.

## 14.3. Блок синтезатора

Этот блок предназначен для синтеза произвольных звуковых сигналов, в том числе *голоса* и *музыкальных инструментов*.

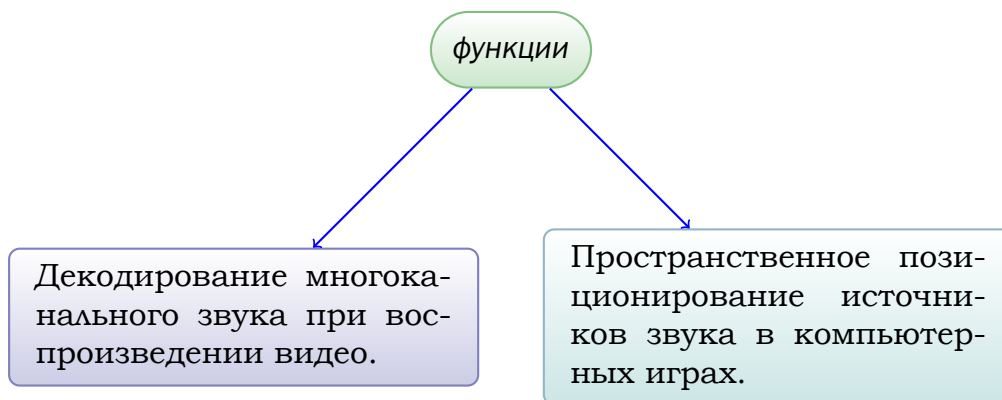
В современных звуковых платах этот блок отсутствует. Его поставляет только фирма Creative Technology.



## 14.4. Блок DSP

Digital signal processor (DSP) — цифровой сигнальный процессор.

Используется для обработки аудиоданных в цифровом виде. Блок DSP «разгружает» основной процессор во время звуковой обработки.



## 14.5. Блок интерфейсов

Большинство звуковых плат поддерживают:

- USB;
- FireWire;
- PCI;
- PCI-express;
- MIDI — на встроенной звуковой плате нужны специальные адаптеры;
- S/P-DIF — для передачи данных в цифровом формате.

## 14.6. Блок микшера

Этот блок нужен для сведения входных и выходных аудио-сигналов и регулировки их уровня. Важное свойство — поддержка внутренней коммутации. Таким образом карта может записывать сама на себя.

## ➤ MIDI-интерфейс ➤

### 15. MIDI-интерфейс

**MIDI**<sup>26</sup> — это цифровой интерфейс музыкальных инструментов.

Создан в 1982 г. ведущими фирмами-производителями музыкального оборудования — Yamaha, Roland, Korg, E-mu и др. Изначально был предназначен для замены принятого в то время стандарта управления с помощью аналоговых сигналов. Впоследствии интерфейс MIDI стал стандартом де-факто в области электронных музыкальных инструментов и компьютерных синтезаторов.

#### 15.1. Назначение MIDI

Главное назначение MIDI — хранение и передача информации в нотной записи:

- управление электронными музыкальными инструментами в реальном времени;
- запись MIDI-потока, формируемого при игре исполнителя, на носитель данных с последующим редактированием и воспроизведением;
- синхронизация различной аппаратуры.

Существуют устройства, управляемые только через интерфейс MIDI. Наиболее распространенным таким устройством является тон-генератор.

**Тон-генератор** — это устройство, предназначенное для синтеза и управляемое только через MIDI-интерфейс.

На самом деле, это обычный синтезатор, только без клавиатуры.

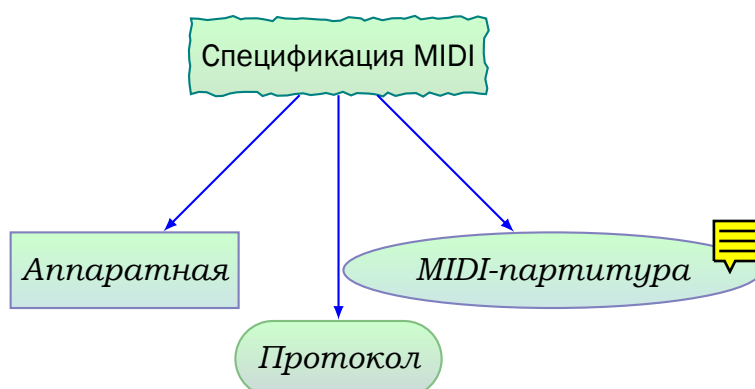
Тон-генератор с достаточными возможностями управления может очень точно воспроизвести звучание инструмента по заданному MIDI-потoku.

---

<sup>26</sup>Musical Instrument Digital Interface.

Для формирования MIDI-сообщений используют MIDI-контроллеры:

- клавиатура;
- педаль;
- рукоятка с несколькими степенями свободы;
- ударная установка (с датчиками способа и силы удара);
- струнный инструмент;
- духовой инструмент.



В настоящее время эти понятия стали совершенно самостоятельными: по аппаратному интерфейсу могут передаваться данные любого формата, а MIDI-формат может применяться только для обработки партитур, без вывода на устройство синтеза.

## 15.2. Аппаратная реализация MIDI

Аппаратная реализация интерфейса MIDI представляет собой обычный последовательный асинхронный интерфейс типа «токовая петля». Скорость передачи данных  $\approx 31250$  бит/с.

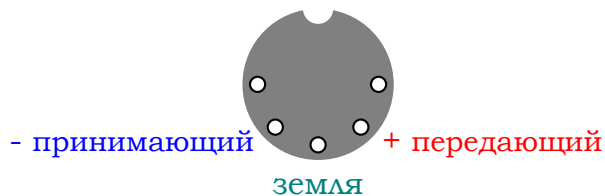
Интерфейс реализуется активным передатчиком с уровнем 5 мА. Для соединения используется двухжильный экранированный<sup>27</sup> кабель длиной не более  $\approx 15$  м, нечувствительный к наводкам извне<sup>28</sup>.

<sup>27</sup>Экран необходим только для защиты от излучаемых помех.

<sup>28</sup>Это потому, что «токовая петля», наводки просто гасят друг друга.

## 15.3. Разъем DIN-5 (СГ-5)

Разъем для MIDI-устройства:

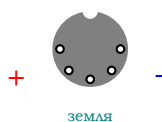


Это разъем типа female DIN-5 (СГ-5, «мама»).

Оставшиеся 2 контакта не используются.

Каждый инструмент имеет три соединительных разъема:

- In (вход);



- Out (выход);



- Thru (копия сигнала с In через буфер).



Один MIDI-передатчик допускает подключение до четырех приемников. Так, что можно создавать сеть MIDI-устройств, выстраивая их по цепочке, в нескольких направлениях.



## 16. Протокол MIDI

MIDI-протокол является событийно-ориентированным.

Обмен данными осуществляется при помощи сообщений. Сообщения — блоки данных произвольной длины. Каждое сообщение является командой для музыкального инструмента.

Стандарт предусматривает 16 независимых и равноправных логических каналов. Внутри канала действуют свои режимы работы.

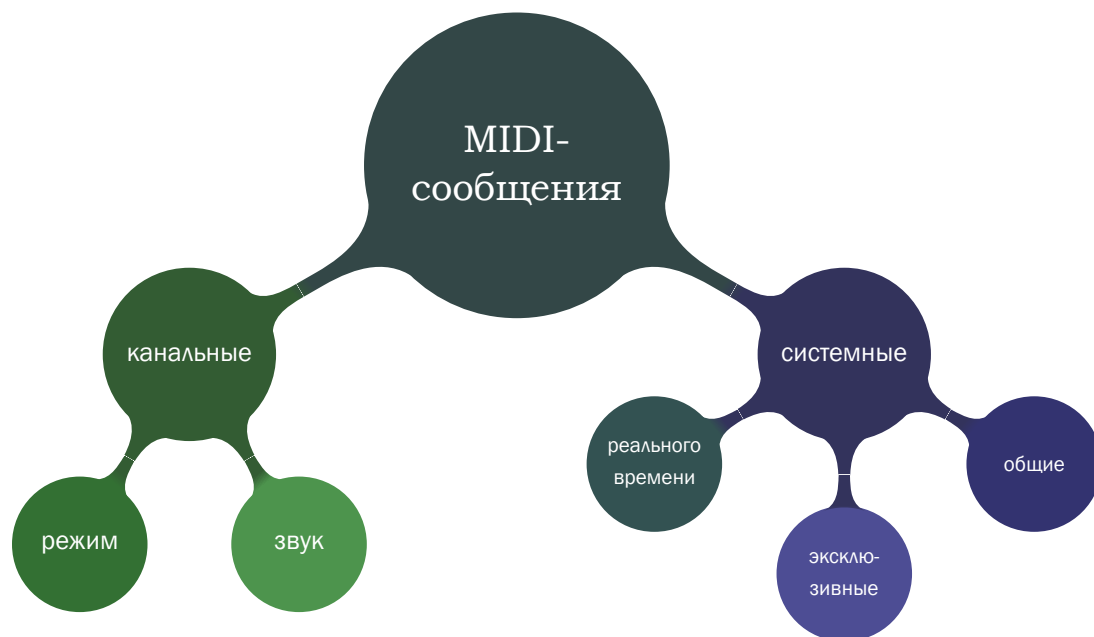
### 16.1. Адресация в MIDI

Адресация в MIDI не является однозначной. Несколько инструментов могут быть настроены на один и тот же MIDI-канал. В таком случае канальное сообщение может быть послано группе инструментов.

Изначально технология была предназначена для однотембровых инструментов. Однотембровые инструменты могли воспроизводить звук только одного тембра в каждый момент времени. Каждому инструменту присваивался свой номер канала. Последнее давало возможность многотембрового исполнения. С появлением многотембровых инструментов они стали поддерживать несколько каналов. Современные инструменты поддерживают все 16 каналов и могут иметь более одного MIDI-интерфейса. Сейчас каждому каналу обычно назначается свой тембр, называемый по традиции инструментом, хотя возможна комбинация нескольких тембров в одном канале.

У каждого MIDI устройства есть режим **OMNI ON**. В этом режиме оно перестает различать каналы.

## 16.2. Типы MIDI-сообщений



**Канальные сообщения** предназначены для передачи команд и параметров отдельным MIDI-устройствам сети в целях управления звучанием музыкального инструмента и определения реакции музыкального инструмента на сообщения.

**Системные сообщения** — это сообщения предназначенные для управления всеми MIDI-устройствами сети.

### 16.2.1. Канальные сообщения о звуке

- Note Off (выключение ноты);
- Note On (включение ноты);

В качестве параметров команды передаются номера клавиш и ускорение, с которым была нажата или отжата клавиша.

- Key Pressure (Polyphonic Aftertouch, давление на клавишу);
- Channel Pressure (Channel Aftertouch, давление в канале);

Например, чем сильнее давим на клавишу, тем громче она звучит.

- Control Change (смена значения контроллера);
- Program Change (смена программы (тембра, инструмента));
- Pitch Bend Change (смена значения Pitch Bend).

### 16.2.2. Канальные сообщения о режиме

- Omni Off (выключение режима всех сообщений);
- Omni On (включение режима всех сообщений);
- Poly/Mono:
  - ▶ в режиме Poly можно брать аккорды,
  - ▶ в режиме Mono срабатывают самые верхние ноты из нескольких нажатых;


Забавно заметить, но именно режим Mono был долгое время единственным для советских синтезаторов класса «Электроника».

- Local Control Off (выключение режима локального управления);
- Local Control On (включение режима локального управления);
- All Notes Off (сообщение о потере сигнала).

### 16.2.3. Системные сообщения

- System Exclusive (SysEx, системное исключительное сообщение);
- Song Position Pointer (указатель позиции в партитуре);
- Song Select (выбор партитуры);
- Tune Request (запрос подстройки);
- Timing Clock (синхронизация по времени);
- Start (запуск игры по партитуре);
- Continue (продолжение игры по партитуре);
- Stop (остановка игры по партитуре);
- Active Sensing (проверка соединений MIDI-сети);
- System Reset (сброс всех устройств сети);
- EOX (End Of SysEx, конец системного исключительного сообщения).

## 17. Стандарты MIDI-систем

MIDI-файл не передает полного оригинального звучания инструментов. MIDI-системы в основном используются как ающие обучающие системы и для одинакового звучания музыки в играх.

Набор команд синтезаторов в MIDI-системах непереносим.

### 17.1. General MIDI


Стандарт General MIDI (GM) появился 1991 году.

Стандарт вводит несколько ограничений, благодаря которым MIDI-файл легче переносить:

- 128 стандартных инструментов;
- выделение 10-го канала под ударные инструменты.

Основные контроллеры

- (1) модуляция<sup>29</sup>;
- (7) громкость инструмента<sup>30</sup>;
- (10) панорама<sup>31</sup> — правый-левый канал.

Изначально этот стандарт был предназначен ен для узкого круга инструментов (скрипки, пианино, ударные).

**Проблема:**

Сложно реализовать синтезатор без Wave Table.

### 17.2. General Synthesis

Стандарт General Synthesis (GS) был предложен фирмой Roland в 1991 году. Он полностью поддерживает General MIDI.

Используется банк инструментов.

Введено дополнительно 98 звуков<sup>32</sup>.

Введены дополнительные контроллеры:

- (91) реверберация;
- (93) хорус<sup>33</sup>.

---

<sup>29</sup>Modulation.

<sup>30</sup>Volume.

<sup>31</sup>Pan.

<sup>32</sup>Все, из того что производила фирма.

<sup>33</sup>Chorus.

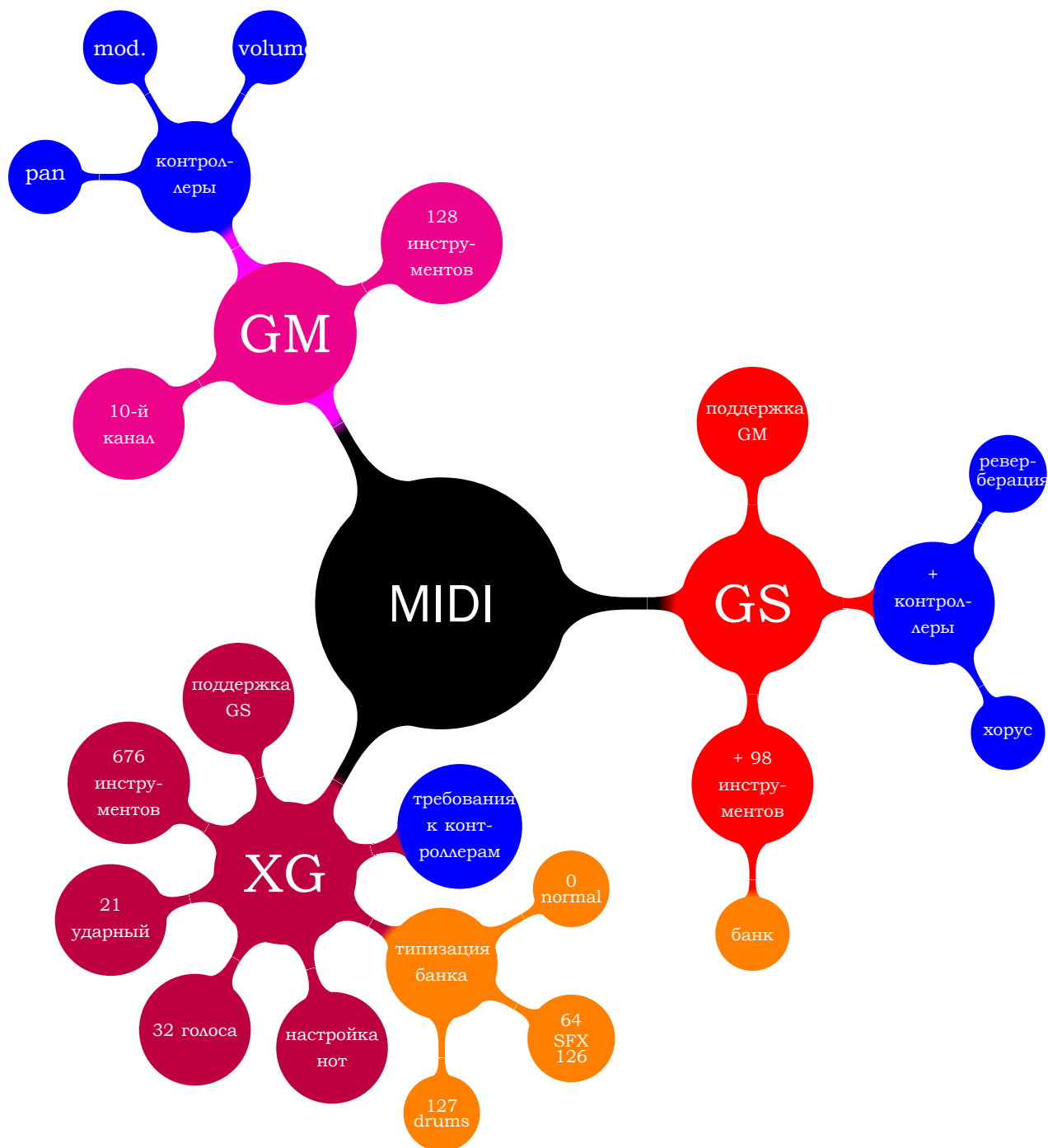
### 17.3. Extended General MIDI (XG)

Стандарт был предложен фирмой Yamaha. Это наиболее жесткий стандарт MIDI. Он полностью поддерживает General MIDI и General Synthesis. Особенности:

- 676 инструментов;
- 21 набор ударных инструментов;
- 32 голоса, полифония;
- расширенные требования к различным контроллерам;
- есть возможность настройки каждой ноты звукоряда;
- введена типизация банка инструментов:
  - ▶ (0) normal — мелодичные инструменты;
  - ▶ (64) SFX — специальные эффекты;
  - ▶ (126) SFX — специальные эффекты;
  - ▶ (127) drums — ударные.

Extended General полностью поддерживается только Yamaha и *почти* полностью переносим.

## 17.4. Все вместе

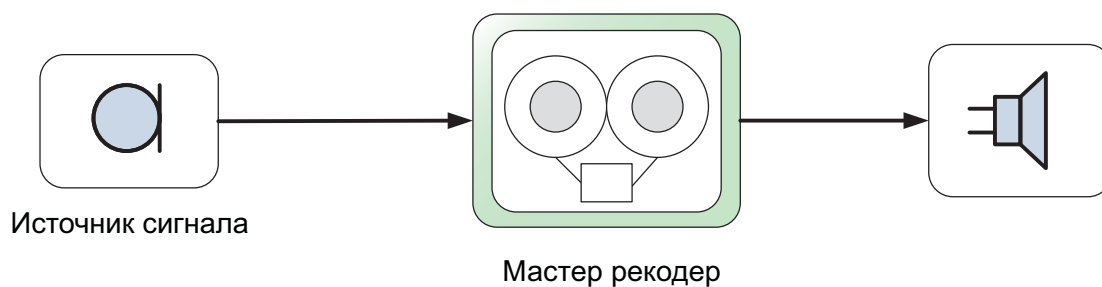


## ✧ Запись и передача ✧

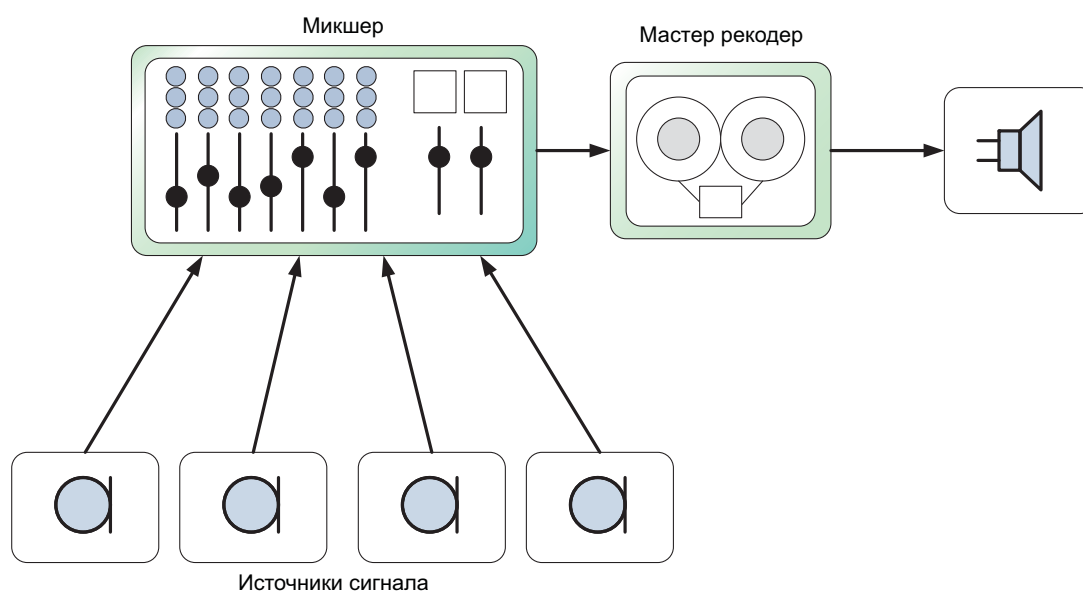
### 18. Секвенсоры

Индустрия звукозаписи развивается с 30-х годов XX века. Технологию записи музыкальных произведений можно представить с помощью следующих простых схем:

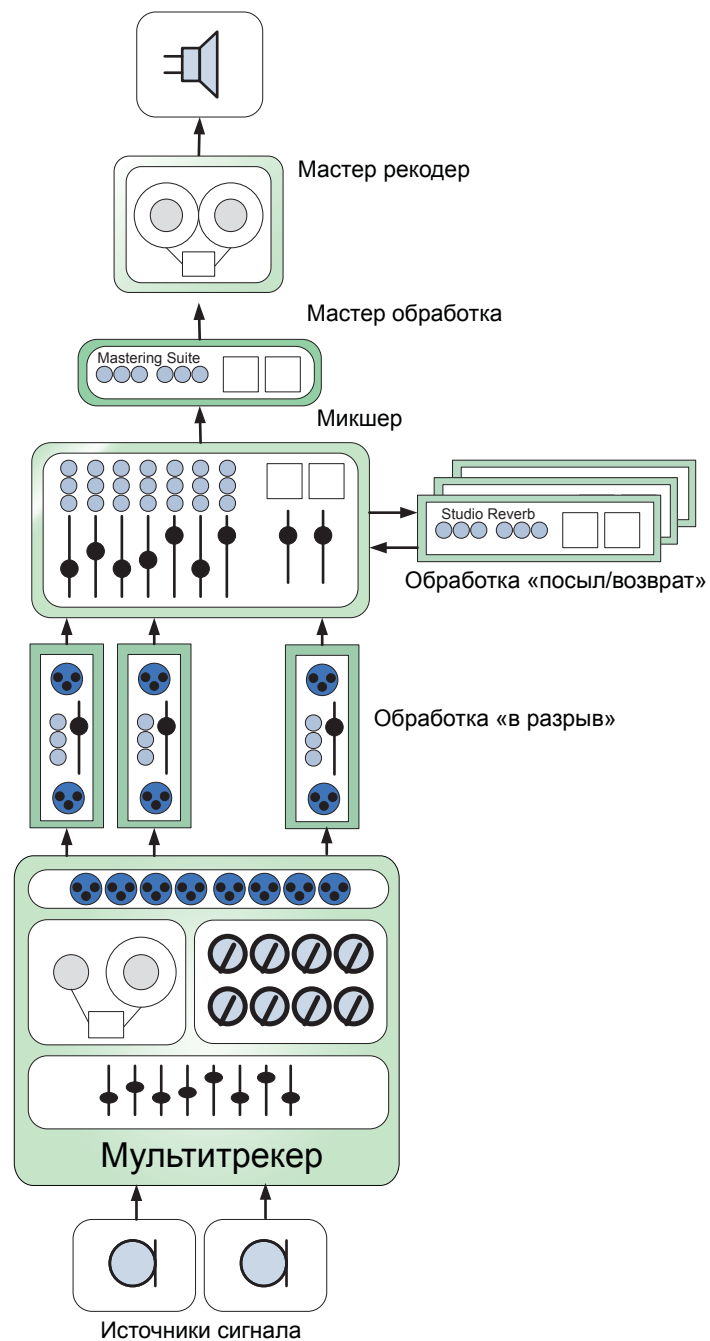
#### 18.1. Простейшая студия



#### 18.2. Студия для записи «живого» исполнения

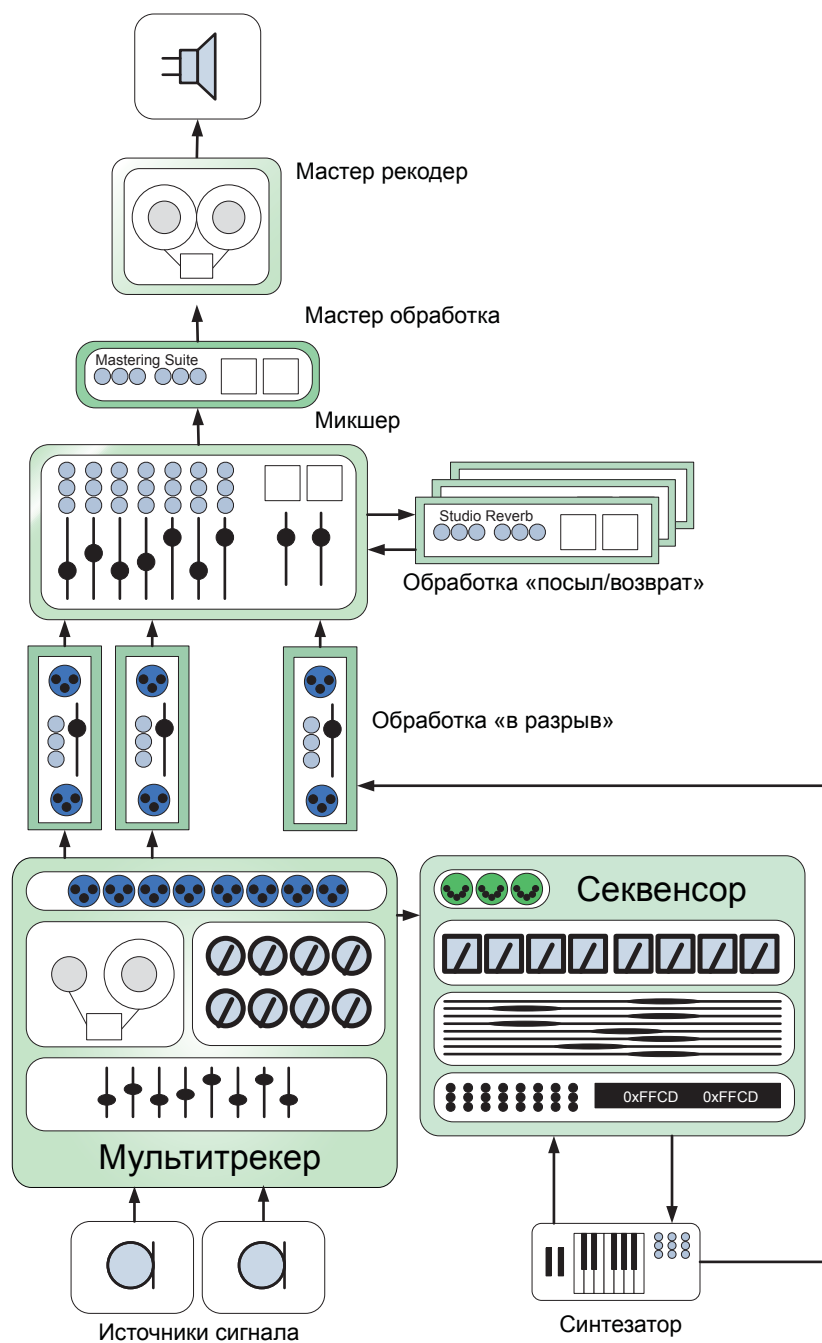


### 18.3. Студия для многоканальной записи



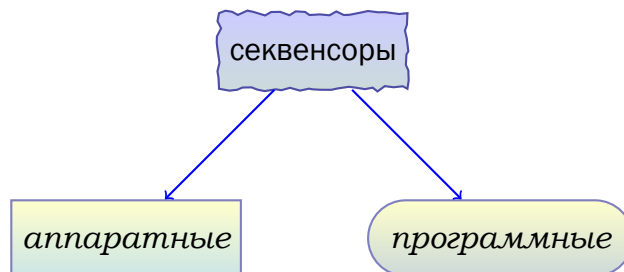


## 18.4. Студия для многоканальной записи с секвенсором



## 18.5. Секвенсор

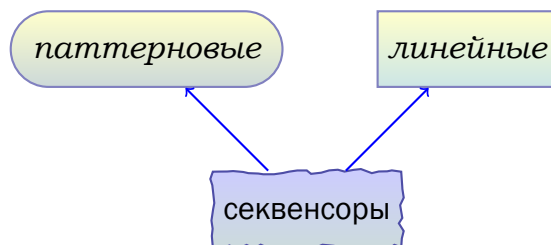
**Секвенсор** — аппаратное или программное устройство для записи и воспроизведения MIDI-сообщений.



Музыкальный синтезатор 80-х годов имел 3 аналоговых блока:

- генератор;
- фильтр;
- усилитель.

Все они управлялись напряжением. Первые секвенсоры использовались для координации работы этих блоков. Такие аппараты состояли из ряда потенциометров и могли подавать различное напряжение циклически. В простейшем случае секвенсор перебирал одну последовательность из трех (по числу блоков) напряжений. Эта идея воплотилась в современных паттерновых секвенсорах. На данный момент для управления музыкальными инструментами используется стандарт DCB-Roland.



### 18.5.1. Паттерновые (шаговые)

Особенности:

- управление ручками или тумблерами (для аппаратных);
- наличие паттернов, которые хранятся в банке звуков (эти паттерны программируются, а не записываются);
- не позволяет играть аккорды;
- достаточно громоздкие и сложные.

Наиболее яркий представитель — шаговый секвенсор Matrix в ПО Propellerhead Reason.

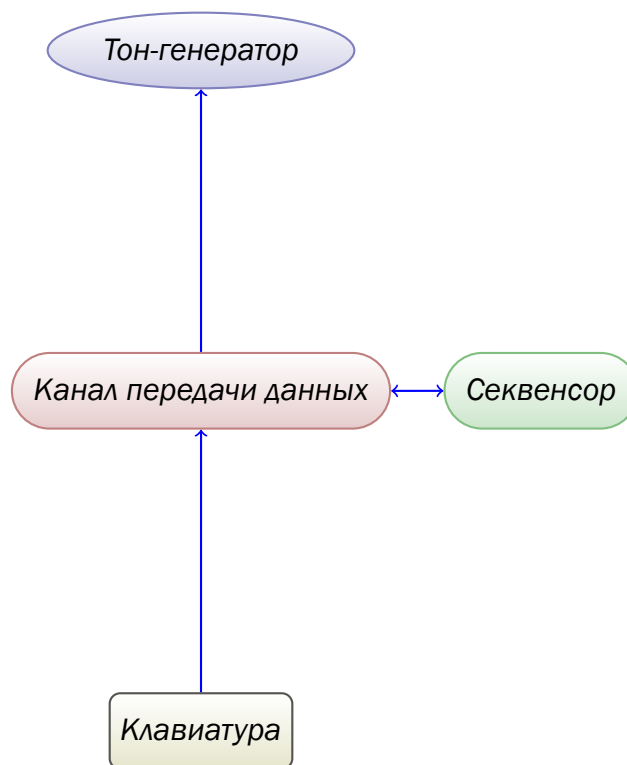
## 18.5.2. Линейные

Особенности:

- есть 16 каналов для приема сообщений;
- есть 16 дорожек для записи MIDI сообщений;

Связь дорожек и каналов прямая. В зависимости от того какой канал «играется», запись производится на дорожку с таким же номером. На различные каналы может подаваться информация от различных инструментов.

- позволяет играть аккорды;



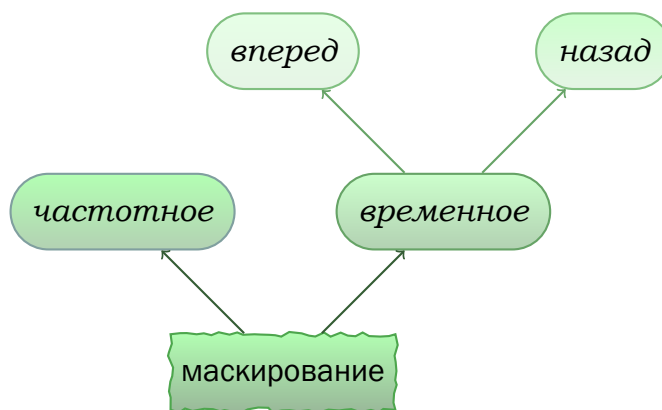
Наиболее яркие представители:

- секвенсор в Ableton Live;
- NoteWorthy Composer;
- Steinberg Cubase;
- Cakewalk Sonar.



## 19. Маскирование

При восприятии звука человеком наблюдается эффект маскирования. Более сильные сигналы преобладают над менее сильными, *маскируя* тем самым тихие звуки, **попадающие в тот же диапазон частот**.



**Частотное маскирование** — эффект, при котором один звук маскирует другие, более слабые, звуки. Диапазон частот, в пределах которого один звук может маскировать другой, в соответствии с концепцией Флетчера, называется критической полосой.

**Временное маскирование** — эффект, при котором звук большой амплитуды маскирует другие звуки, предшествующие ему во времени или следующие за ним.

**Маскирование назад** — эффект, при котором звук большой амплитуды маскирует другие звуки, только предшествующие ему. Промежуток времени, в пределах которого оно действует, составляет 5-50 миллисекунд.

**Маскирование вперед** — эффект, при котором звук большой амплитуды маскирует другие звуки, только следующие за ним. Промежуток времени, в пределах которого оно действует, составляет 50-200 миллисекунд.

Заметим, что человеческая система восприятия звука имеет ограниченное разрешение. Это разрешение зависит от частоты звука. Равномерное, с точки зрения восприятия человеком, измерение частоты может быть выражено в единицах ширины критических полос. Их ширина составляет менее 100 Гц для нижних слышимых частот и более 4 кГц — для наиболее высоких. Весь частотный диапазон может быть разделен на 25 критических полос.

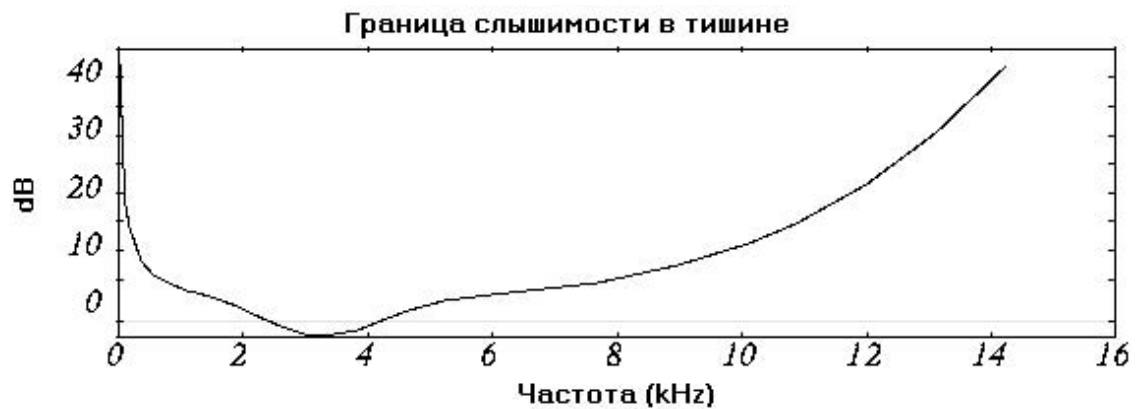
## 19.1. Иллюстрации эффектов маскирования

Иллюстрации взяты с сайта [websound.ru](http://websound.ru).

### 19.1.1. Граница слышимости в тишине

Был проведен эксперимент:

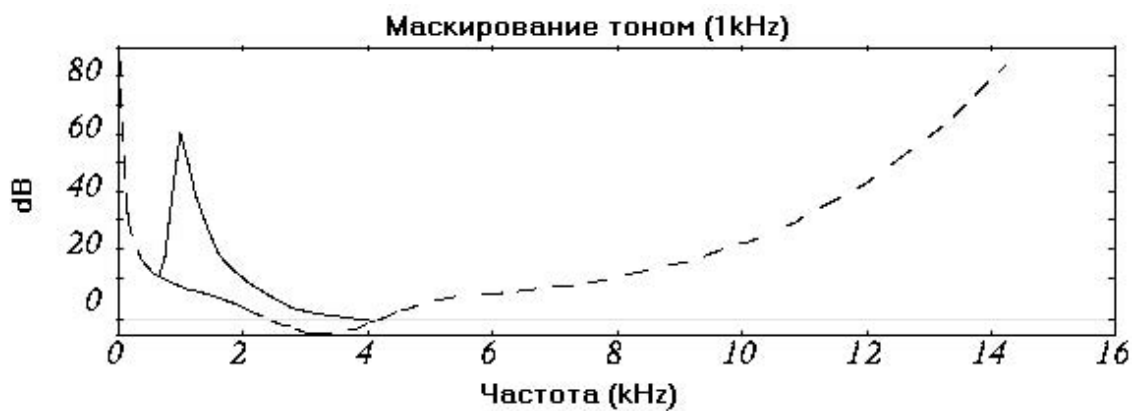
Слушатель находится в тихой комнате. Громкость тона частотой 1 кГц повышают до уровня когда он становится слышимым.



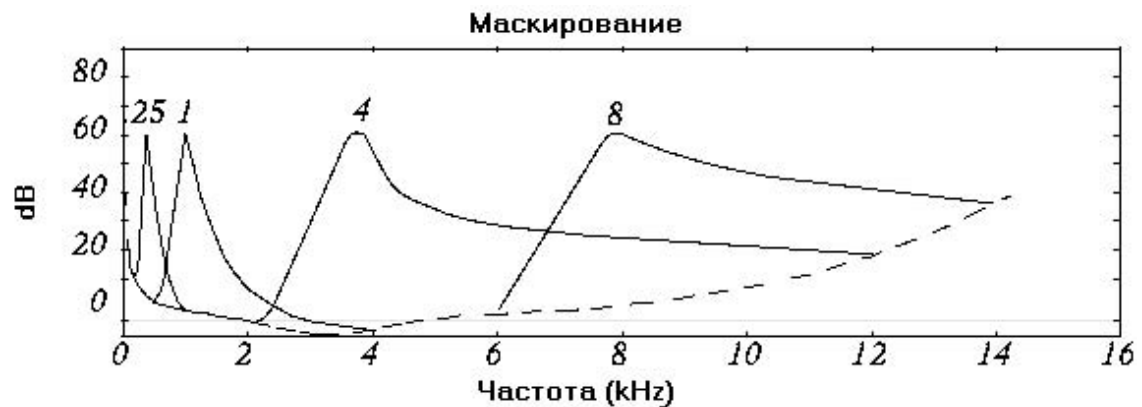
### 19.1.2. Маскирование тоном

Был проведен эксперимент:

Воспроизводится тон частотой 1 кГц (маскирующий сигнал), с фиксированной громкостью — 60 дБ). Воспроизводится тестовый (маскируемый) тон с различной громкостью и частотой.



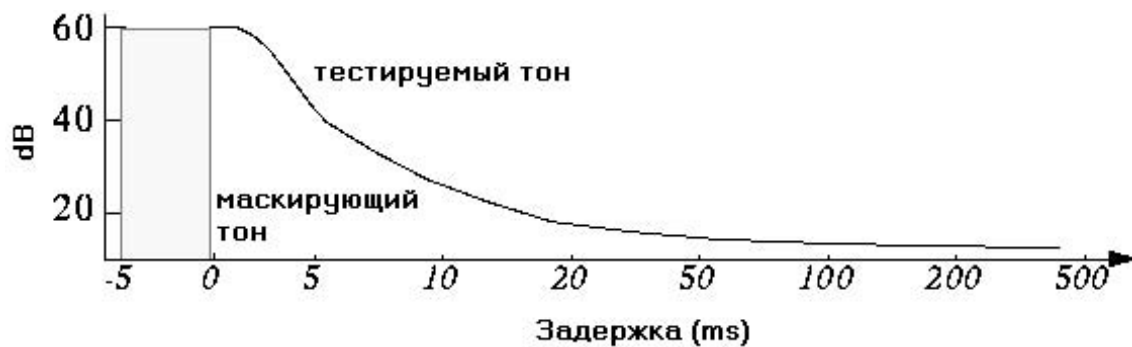
### 19.1.3. Маскирование для различных частот



### 19.1.4. Временное маскирование

Был проведен эксперимент:

Воспроизводится маскирующий тон (1 кГц) с фиксированной громкостью — 60 дБ. Воспроизводится тестовый тон (1.1 кГц) с фиксированной громкостью — 60 дБ. В этом случае тестовый тон не слышен (он замаскирован). Маскирующий тон отключают, затем, после небольшой **отключают** тестовый тон. Время задержки уменьшают до тех пор пока тестовый тон еще слышен. В результате получается следующий график.



## 19.2. Сжатие звука

При сжатии звука не имеет смысла оставлять в записи звуки, которые находятся ниже порога слышимости, поэтому любые алгоритмы сжатия должны отбрасывать соответствующие данные.

Чтобы реализовать эту идею на практике, алгоритм должен использовать психоакустическую модель.

**Психоакустическая модель** — математическое описание восприятия звуков ухом и головным мозгом с учетом критических полос. Алгоритм сжатия получает дополнительную возможность отбросить некоторые данные.

В то же время, маскировку можно использовать более эффективно. Поскольку она, кроме некоторых компонентов сигнала, скрывает шум, в ней можно скрыть *шум квантования*.

Приведем краткое описание алгоритма сжатия:

```
1: расщепить сигнал на полосы частот; /* блок фильтров */
2: Для каждой полосы выполняем:
3:   вычислить средний уровень сигнала;
4:   подставить значения в психоакустическую модель;
5:   определить порог маскировки;
6:   /** Предполагается, что маскирующую кривую в каждой полосе можно
    аппроксимировать одним значением. **/
7:   Если сигнал целиком опускается ниже порога маскировки,
    то:
8:     отвергнуть полосу;
9:     продолжить цикл;
10:  иначе:
11:    квантовать сигнал грубо;
12:    /** Сигнал квантуется с использованием меньшего количества битов
    за счет маскировки шума квантования. **/
```

Примерно такой алгоритм сжатия используется в формат хранения и передачи аудиосигнала MP3.

## 20. Формат MP3

**MP3** — сокращение от MPEG<sup>34</sup> Layer 3. Это формат хранения и передачи аудиосигнала в цифровой форме. В MP3 используется алгоритм сжатия с потерями.

Формат был разработан компанией Fraunhofer IIS при спонсорстве компании Thomson. Изначально разрабатывался для передачи аудиоданных через Интернет с высокой скоростью и является потоковым. Позднее MP3 был утвержден как часть стандартов сжатого видео и аудио MPEG 1 и MPEG 2.

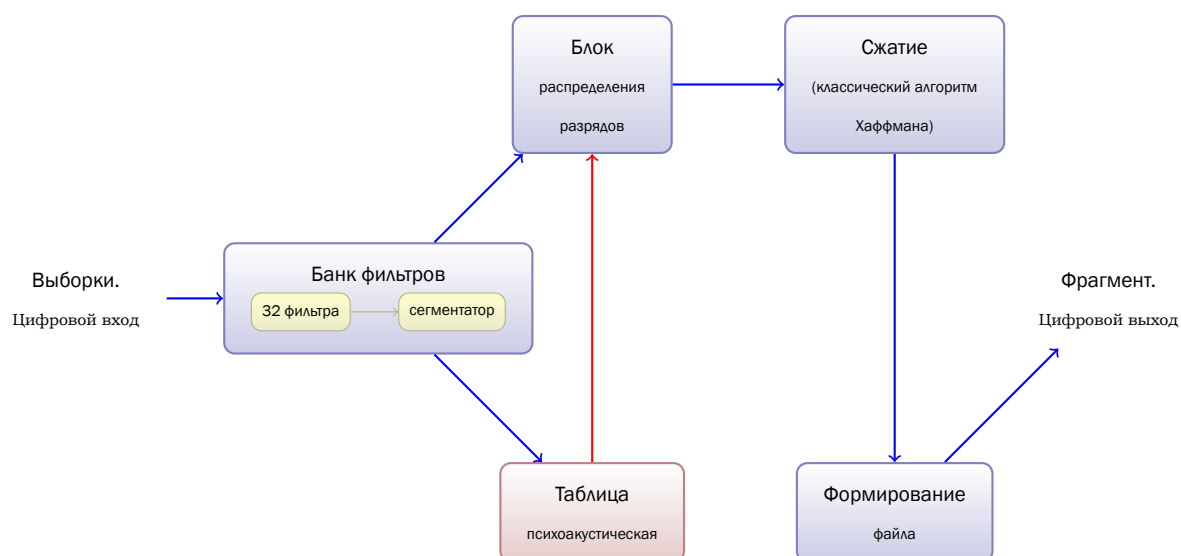
### 20.1. Алгоритм сжатия MP3

Алгоритм сжатия MP3 использует особенности слуха:

- абсолютные ограничения слуха (20 Гц – 20 кГц);
- частоты маскирования и ширину критических полосы;
- последовательное (временное) маскирование;
- воспроизводимую энтропию<sup>35</sup>.

*Зачем кодировать звуки, которые все равно не будут услышаны?*

### 20.2. Схема кодера MP3



<sup>34</sup>Moving Pictures Expert Group

<sup>35</sup>Степень информационной значимости гармоник.



Файл состоит из нескольких фрагментов (фреймов) МРЗ. Такая последовательность фрагментов называется **элементарным потоком**.

## 20.3. Режимы кодирования стерео

Существуют несколько методов кодирования стерео аудио информации<sup>36</sup>.

- 1) **Dual channel**<sup>37</sup> — два абсолютно независимых («совсем разных») канала. Битрейт делится на два канала.

Может использоваться для речевого сопровождения на разных языках.

- 2) **Standard stereo** — два канала. Битрейт варьируется в зависимости от сложности сигнала в каждом канале.

- 3) **Joint stereo**<sup>38</sup> — основан на использовании избыточности стерео-информации.

- **MS Stereo.** Кодироваться на левый и правый канал, а их суммарная и разностная составляющие. Разностный канал в некоторых случаях (Lame encoder) кодируется с меньшим битрейтом.
- **Intensity Stereo (MS/IS Stereo).** Кодироваться суммарная составляющая. Вместо разностной составляющей, кодируется отношение мощностей сигнала на разных каналах.

Особенность: повышается качество кодирования на особо низких битрейтах, но происходит потеря фазовой информации, теряется любой противофазный сигнал.

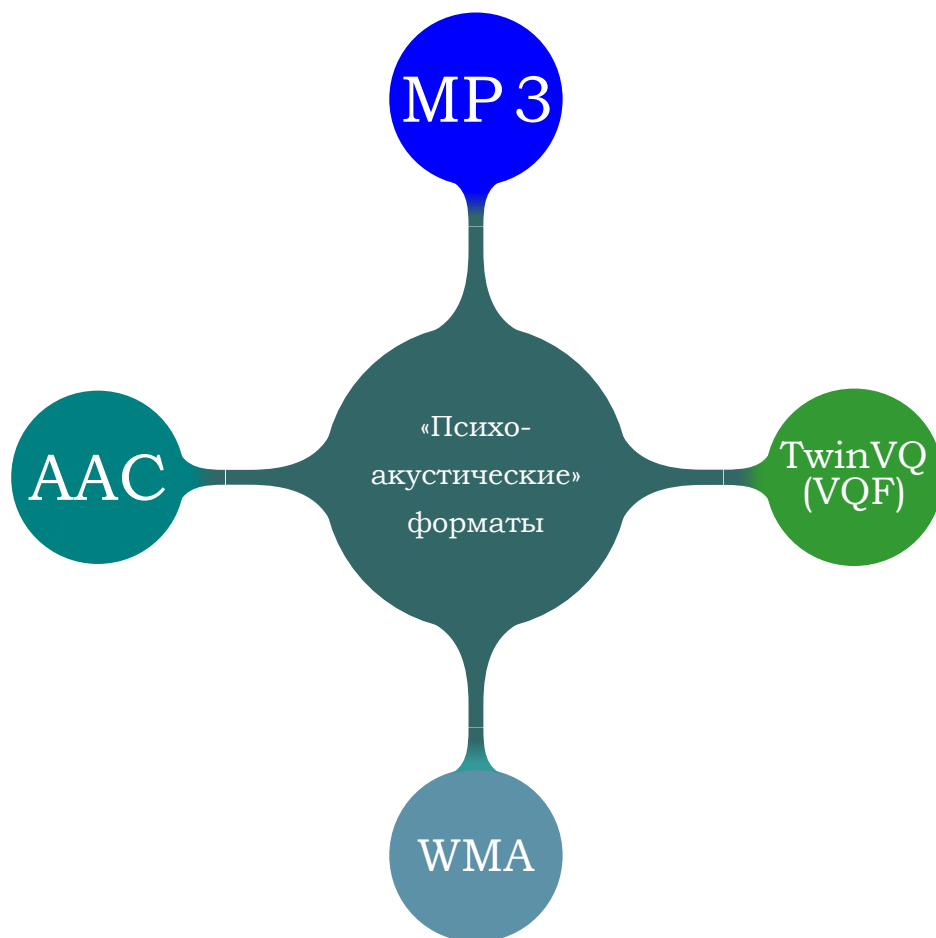
---

<sup>36</sup>В стандарте ISO11172-3 (MPEG-1 Layer 1,2,3).

<sup>37</sup>Двухканальное стерео.

<sup>38</sup>Объединенное стерео

## 20.4. Психоакустические форматы



- **AAC**<sup>39</sup> — разрабатывался как преемник MP3 компанией Fraunhofer при участии AT&T, Sony, NEC и Dolby. Проект не был доведен до конца. На данный момент существует большое число несовместимых друг с другом форматов на основе AAC<sup>40</sup>.
- **TwinVQ** — формат, разработанный компанией NTT<sup>41</sup>. Считается старейшим «конкурентом» MP3. Лицензией на право распространения этого формата владеет фирма Yamaha.
- **WMA**<sup>42</sup> — формат сжатия аудиоданных от компании Microsoft.

---

<sup>39</sup>Advanced Audio Coding.

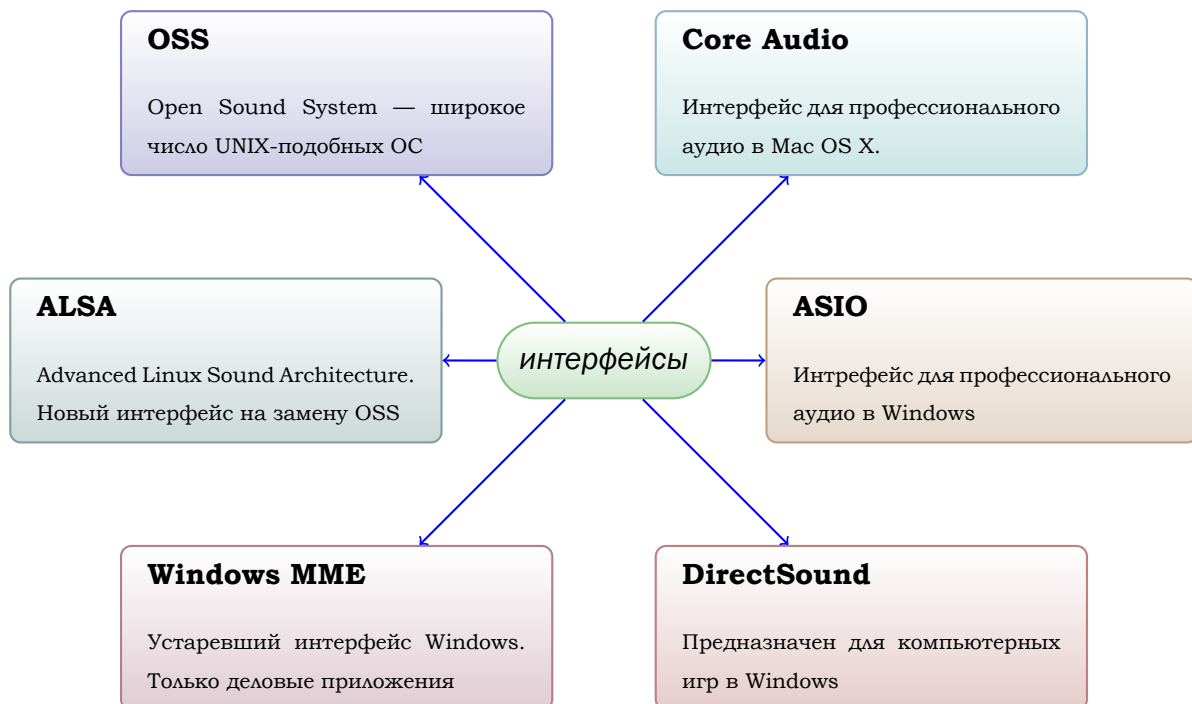
<sup>40</sup>«Семейство форматов AAC».

<sup>41</sup>Nippon Telegraph and Telephone.

<sup>42</sup>Windows Media Audio.

## ➤ Программирование звука ➤

### 21. Основные программные интерфейсы



### 22. Программный интерфейс MME

Первоначально интерфейс со звуковыми устройствами был введен в Windows 3.x под названием MME<sup>43</sup>. Звуковые устройства в Windows относятся к классу Multimedia/Audio.

<sup>43</sup>MultiMedia Extension — мультимедийное расширение.



Взаимодействие приложения с драйвером организуется в виде взаимного **обмена потоками** звуковых данных в реальном времени. Для переноса потоков между приложением и звуковым драйвером используется **звуковой буфер**. Звуковые буферы создаются приложением, и затем передаются драйверу:

- **пустые** — для устройств ввода;
- **заполненные** звуковыми данными — для устройств вывода.

Применяется концепция *связанной цепочки программных буферов*.

## 22.1. Способы кодирования звука

При работе со звуковыми адаптерами чаще всего используется традиционный способ цифрового кодирования РСМ<sup>44</sup>.

Ряд мгновенных значений звуковой амплитуды, следующих друг за другом с частотой дискретизации, *представляется рядом чисел выбранной разрядности*. Значения пропорциональны величине амплитуды. Именно в таком виде звуковой поток снимается с выхода АЦП или подается на вход ЦАП.

Однако, наряду с предельной простотой, РСМ обладает **существенной избыточностью**:

Звук передается настолько точно, насколько это возможно при выбранных параметрах оцифровки. На первое место выходит задача минимизации скорости и объема звукового потока, отдельными параметрами точности и качества можно пренебречь.

<sup>44</sup>Pulse Code Modulation - импульсно-кодовая модуляция, ИКМ.

Обрабатывать звук в PCM способен *любой* звуковой адаптер!


## 22.2. Формат потока

**Формат потока** — совокупность основных параметров потока.

- способ кодирования — это главный параметр, он же признак формата<sup>45</sup>;
- частота дискретизации;
- количество каналов;
- разрядность отсчета.

## 22.3. Структура потока

**Блок** — наименьшая единица звукового потока. Размер каждого буфера должен быть кратен размеру блока.

В PCM блоком считается набор отсчетов, передаваемых за один период частоты дискретизации, то есть — один отсчет для монофонических потоков, два — для стереофонических,  так далее. Отсчеты могут быть 8-разрядными, 16-разрядными, 32-разрядными. Современные звуковые адаптеры могут использовать 18-, 20- и 22-разрядные отсчеты. Отсчет выравнивается по старшей границе трех- или четырехбайтового слова, а лишние младшие разряды заполняются нулями. Трехбайтовые слова почти не используются и заменяются четырехбайтовыми.

С момента запуска потока драйвер отслеживает текущую позицию записи или воспроизведения, которая в любой момент может быть запрошена приложением. Для этого адаптеры используют очередь между преобразователями и встроенным процессором.

Драйвер отслеживает позицию путем подсчета количества звуковых блоков потока, переданных от приложения к устройству или наоборот.

---

<sup>45</sup>format tag

## 22.4. Системные особенности

### 22.4.1. Несколько процессов

Звуковая подсистема Windows допускает работу с устройством **нескольких** процессов (клиентов) **одновременно**. Многие звуковые устройства поддерживают более одного клиента; устройство вывода смешивает проигрываемые клиентами звуковые потоки, а устройство ввода — «тиражирует» записываемый поток для всех подключенных клиентов.

Устройство, драйвер которого поддерживает **не более одного клиента**, не может быть повторно открыто до тех пор, пока клиент не закроет его. При попытке повторно открыть такое устройство, звуковая подсистема возвращает ошибку, сигнализирующую о том, что устройство занято.

### 22.4.2. Wave Mapper

Для упрощения реализации основных операций со звуком, Windows содержит службу переназначения — Wave Mapper. В Windows может быть установлено более одного звукового устройства. Существует понятие стандартного системного устройства ввода и стандартного системного устройства вывода.

В Windows имеется подсистема сжатия звука — АСМ<sup>46</sup>. При помощи АСМ возможно взаимное преобразование звуковых форматов — как внутри групп, так и между ними. Служба АСМ может использоваться, как автономно, через собственный отдельный интерфейс, так и автоматически службой Wave Mapper.

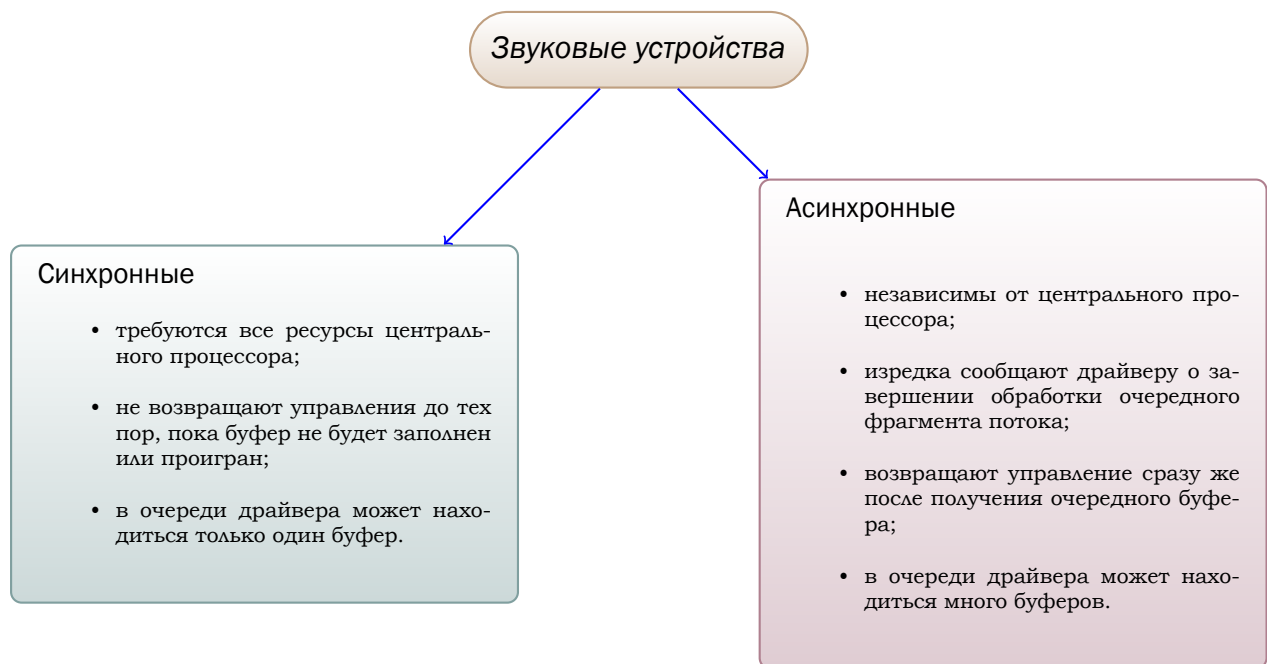
Подсистема сжатия реализована в виде набора кодеков<sup>47</sup>, специальных драйверов АСМ, которые и занимаются непосредственно переводом звука из одного формата в другой. АСМ активизирует нужные кодеки по запрошенным форматам, снабжая их необходимыми параметрами.

---

<sup>46</sup>Audio Compression Manager.

<sup>47</sup>ACM Codec

### 22.4.3. Устройства



При завершении обработки каждого буфера драйвер устанавливает в его заголовке **флаг готовности**, по которому приложение может определить, что драйвер *освободил данный буфер*.

Для **асинхронных устройств** гораздо более эффективным способом возврата буфера является **уведомление (notification)**. Драйвер:

- либо вызывает заданную функцию приложения,
- либо активизирует событие (event),
- либо передает сообщение заданному окну или задаче (thread) приложения.

В параметрах функции, или сообщения передается также указатель заголовка буфера.

Звуковая подсистема нумерует установленные устройства, начиная с 0. При установке нового устройства или удалении существующего нумерация изменяется. Во время работы программы в системе могут появиться или исчезнуть звуковые устройства. Вместо номера звукового устройства может использоваться ключ (handle) ранее открытого устройства. Система автоматически определяет, какое именно значение передано интерфейсной функции.

При открывании каждого звукового устройства система возвращает его идентификатор, или ключ (handle), по которому затем происходит вся оставшаяся работа с устройством. Формально идентификаторы устройств ввода и вывода имеют различные типы

- HWAVEIN;
- HWAVEOUT.

Оба они эквивалентны типу HWAVE, который может использоваться для создания универсальных функций, не зависящих от типа устройства.

Ключи звуковых устройств не имеют ничего общего с ключами файлов, событий, окон, задач и т.п.

Если программе «безразлично», с каким конкретно устройством она будет работать, либо работа ведется только со стандартным системным устройством, программа может ориентироваться только на службу переназначения. В противном случае программа определяет количество имеющихся в системе устройств ввода и/или вывода при помощи функций **GetNumDevs**.

При необходимости программа может запросить параметры и имена звуковых устройств при помощи функций GetDevCaps — например, чтобы сформировать меню доступных устройств для пользователя или найти устройство, удовлетворяющее заданным требованиям.



## 22.5. Алгоритм взаимодействия

Рассмотрим упрощенный алгоритм взаимодействия программы и звуковой подсистемы:

```
1: открыть (
2:     <устройство> ,
3:     <формат звукового потока> ,
4:     <способ уведомления о выполнении запрошенных операций>
5: );
6: буферы ← создать( <количество> );
7: заполнить_заголовки( <буферы> );
8: Если <сразу подготовить к передаче> то:
9:     подготовить_к_передаче( <буферы> ); /* Prepare */
10: Если <цикл записи> то:
11:     Пока <запись> выполняем:
12:         заполнить очередь драйвера буферами /* AddBuffer */
13:         записать поток /* Start */
14:         /** В этот момент драйвер запускает АЦП адаптера, и звуковые отсчеты начинают поступать в первый буфер из очереди. **/
15:         получить уведомление от драйвера;
16:         определить размер данных; /* dwBytesRecorded */
17:         обработать записанные данные;
18:         освободить буфер;
19:         передать буферы приложению;
20: Если <цикл воспроизведения> то:
21:     Пока <воспроизведение> выполняем:
22:         заполнить буферы звуковыми данными;
23:         передать буферы драйверу устройства вывода; /* Write */
24:         /** После получения первого же буфера драйвер запускает ЦАП адаптера, который начинает извлекать звуковые отсчеты. **/
25:         воспроизвести буфер полностью; /* dwBufferLength */
26:         освободить буфер;
27:         передать буферы приложению;
28:     освободить буферы; /* Unprepare */
29: закрыть устройство; /* Close */
```

При необходимости приостановить движение потока вызывается функция *Stop/Pause*. При этом устройство ввода сразу же возвращает очередной буфер приложению — возможно, заполненный лишь частично. Не полностью проигранный буфер устройства вывода остается в очереди. Остальные буферы устройств обоих типов также остаются в очереди и включаются в работу только после перезапуска потока функциями *Start/Restart*.

Для устройств вывода, поддерживающих **расширенные функции управления**, программа может регулировать громкость звука функцией *SetVolume*, а также изменять высоту тона и скорость воспроизведения функциями *SetPitch* или *SetPlaybackRate*. Более общим способом регулировки громкости является обращение к микшеру (mixer), который является устройством класса **Aux**.

Для аварийного прерывания обработки потока используется функция *Reset*, немедленно останавливающая процесс записи или воспроизведения и возвращающая все буферы из очереди приложению.

## 23. Программный интерфейс DirectSound

### 23.1. Назначение, структура, особенности

Подсистема DirectSound обеспечивает приложениям практически непосредственный доступ к аппаратуре звукового адаптера. Предоставляет модель современного звукового адаптера, предельно приближенная к реальности, с минимальным уровнем абстракции.

Подсистема DirectSound построена по объектно-ориентированному принципу в соответствии с моделью СОМ<sup>48</sup> и состоит из набора интерфейсов.


Каждый интерфейс отвечает за объект определенного типа:

- устройство;
- буфер;
- службу уведомления и т.п.

Интерфейс — набор управляющих функций, организованных в класс объектно-ориентированного языка.

DirectSound не поддерживает звуковые форматы, отличные от PCM. Назначение DirectSound — исключительно эффективный вывод звука.


Основные преимущества DirectSound:

- задание нескольких  источников звука;
- объемный звук (в DirectSound3D это преимущество усилено).

### 23.2. Аппаратная поддержка

DirectSound всю возможную работу старается переложить на аппаратуру адаптера.

Однако, стоит заметить, что приложениям, использующим большое количество одновременно звучащих источников или сложную трехмерную картину, имеет смысл упрощать свою модель при отсутствии средств аппаратного ускорения, иначе накладные расходы могут существенно снизить общую производительность системы.

При отсутствии каких-либо звуков у адаптера DirectSound  эмулирует их на синтезаторе адаптера. Для создания реалистичной звуковой картины DirectSound нуждается в информации о расположении звукоизлучателей — громкоговорителей или наушников — относительно слушателя.

---

<sup>48</sup>Component Object Model — модель объектов-компонентов, или составных объектов.

## 23.3. Звуковые буферы

Большинство существующих звуковых адаптеров использует для обмена звуком с центральным процессором звуковые буферы, представляющие собой участок памяти, в который заносятся звуковые данные.

Обычно буфер — кольцевой.

Адаптер и его драйвер работают параллельно с разными частями буфера, стараясь следовать друг за другом и не создавать конфликтов. Если их работа согласованна — получается непрерывное движение сколь угодно длительного звукового потока.

DirectSound предоставляет приложению почти прямой доступ к аппаратным буферам адаптера.

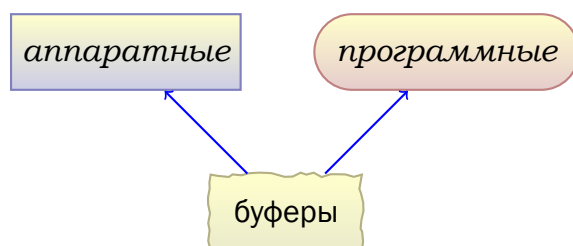
В отличие от модели ММЕ: вывод коротких и повторяющихся звуков значительно упрощается, а вывод длительных непрерывных звучаний несколько усложняется.

### 23.3.1. Аппаратные и программные

Различные адаптеры используют буферы разного типа:

- Классические адаптеры типа Sound Blaster, Windows Sound System и совместимые с ними используют буфер в основной памяти компьютера с доступом через DMA<sup>49</sup>.
- Адаптеры архитектуры Hurricane (Turtle Beach Tahiti, Fiji и совместимые) используют буфер в собственной (on-board) памяти, который доступен в виде «окна» в диапазоне адресов внешних устройств.
- Существуют также адаптеры со встроенным буфером, доступ к которому осуществляется через порты ввода-вывода; обычно так работают таблично-волновые синтезаторы.

В зависимости от размещения и способа управления различают аппаратные и программные буферы.



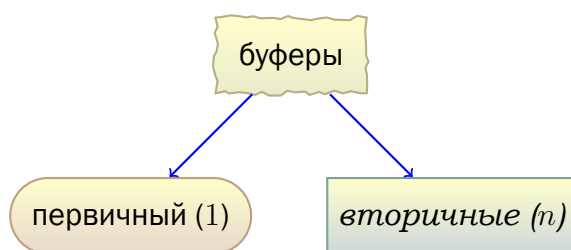
<sup>49</sup>Обмен информации между устройствами, без использования ЦП.

**Аппаратный буфер** — буфер, к которому адаптер имеет прямой доступ. Такой буфер располагается либо в памяти самого адаптера, либо в основной памяти с обращением через DMA.

**Программный буфер** — буфер, к которому адаптер имеет доступ через процессор. Программные буферы всегда располагаются в основной памяти.

В документации по DirectSound аппаратными называют только те буферы, которые находятся в памяти адаптера, и нередко путают термин «hardware» в отношении размещения буфера и способа смешивания звука.

### 23.3.2. Первичный и вторичные



Если в архитектуре адаптера один из аппаратных буферов является основным, его называют первичным (primary). Остальные буферы, занимающие подчиненное положение, называются вторичными (secondary).

Обычно звуки из вторичных буферов смешиваются воедино в первичном буфере, откуда и поступают на ЦАП адаптера.

Для адаптеров, работающих только с одним буфером, он и является первичным. Вторичные буферы могут быть только программными и управляются самой подсистемой DirectSound.

Для современных многоканальных адаптеров PCI, имеющих несколько равноправных каналов вывода звука, первичный буфер недоступен, зато некоторое количество вторичных может быть аппаратными, и управление звуками в них осуществляет непосредственно сам адаптер. Чаще всего приложению не требуется использовать первичный буфер.

В типовой схеме взаимодействия для каждого источника звука создается свой вторичный буфер (в DirectSound часто отождествляются понятия «источник звука» и «вторичный звуковой буфер»). Впоследствии приложение в нужные моменты включает и выключает звучание источников, меняет текущую позицию в звуке, параметры звучания и т.п.

Вторичные буферы могут иметь произвольные размеры, которые задаются приложением при их создании. Даже если смешивание выполняет DirectSound — оно осуществляется на уровне ядра.

Прямой доступ к первичному буферу возможен только в исключительных случаях. При этом запрещается использование вторичных буферов — то есть приложение теряет возможность описывать независимые источники звука. Зато наличие доступа к первичному буферу гарантирует, что все изменения в звуковых данных будут услышаны максимально быстро. Однако первичный буфер имеет фиксированный размер, выбираемый драйвером DirectSound, и размер этот достаточно мал.

Для того чтобы успевать вписывать звук в первичный буфер, приложение должно иметь высокий уровень приоритета. Но даже в этом случае Windows не гарантирует нужной скорости.

Вторичный буфер может быть статическим (static) и потоковым (streaming). **Статические буферы** предназначены для постоянных звуков, цифровое представление которых не меняется либо меняется достаточно редко.

**Потоковые буферы** ориентированы на часто изменяемые звуки, как правило — на представление длительного звукового потока, который по частям «прогоняется» через буфер.

Статические и потоковые буферы различаются только тем, что подсистема старается в первую очередь делать аппаратными статические буферы, загружая их в память адаптера.

Таким образом, постоянные и короткие звуки оказываются в распоряжении адаптера. Достаточно лишь дать команду, чтобы они включились в общее звучание.

Приложение может явно указывать при создании буфера тип памяти для его размещения.

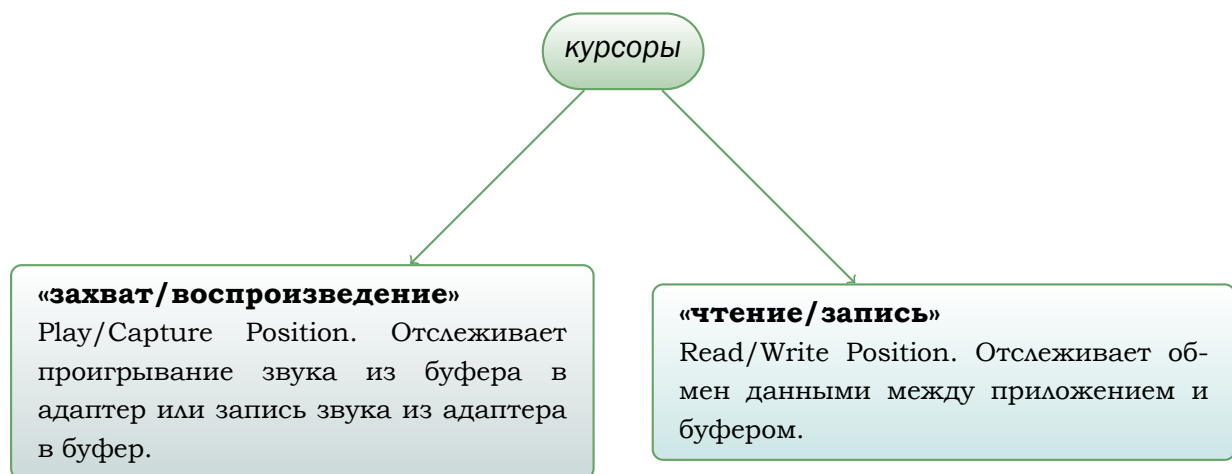
DirectSound оптимизирует использование вторичных буферов в порядке их создания приложением. Источники звука, созданные в первую очередь, имеют приоритет в использовании аппаратных средств. Буферы, созданные первыми, подсистема старается по возможности загружать в память адаптера, предоставлять им каналы DMA. При исчерпании аппаратных ресурсов DirectSound переходит на самостоятельную, программную обработку оставшихся буферов.

Поскольку каждый вторичный буфер описывает независимый источник звука, подсистема предоставляет средства управления режимами звучания источника:

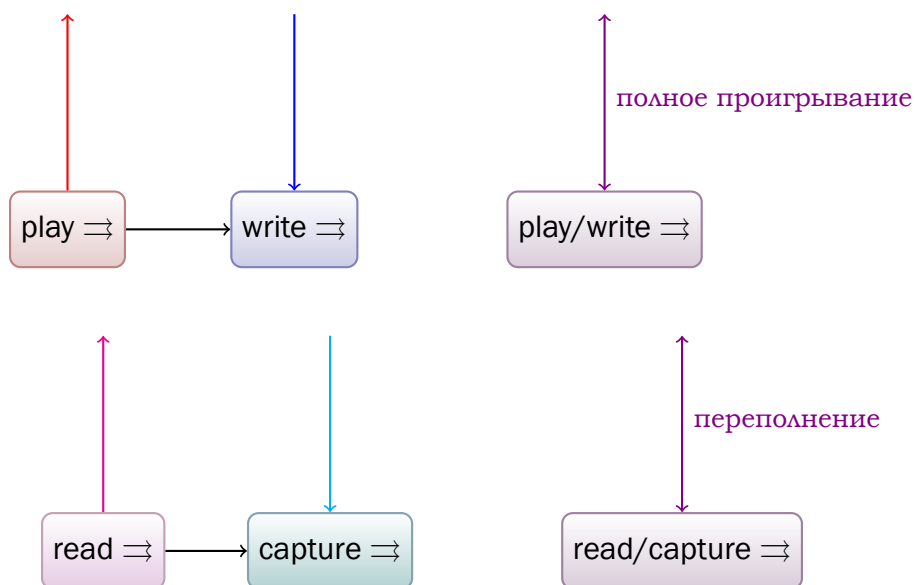
- для базовых источников DirectSound доступно управление:
  - ▶ громкостью,
  - ▶ панорамой,
  - ▶ частотой дискретизации;
- для источников DirectSound3D еще:
  - ▶ пространственными координатами,
  - ▶ направленностью,
  - ▶ скоростью движения.

Набор необходимых для источника методов управления задается при создании буфера и позволяет подсистеме оптимально связывать буферы с аппаратными ресурсами. Впоследствии доступны только заказанные методы управления; для изменения набора необходимо **уничтожить** буфер **и создать** его заново.

DirectSound использует для адресации в звуковых буферах понятие *текущих позиций*, или курсоров.



Позиция воспроизведения (play) следует за позицией записи (write) в буфер, а позиция чтения (read) — за позицией захвата (capture). Достижение позицией воспроизведения позиции записи означает полное проигрывание буфера воспроизведения, при этом начинают воспроизводиться «старые» данные, которые приложение не успело перезаписать. Достижение позицией захвата позиции чтения означает переполнение буфера захвата, и последующие данные накладываются на «старые», которые приложение не успело извлечь из буфера.



Достижение одной из заданных позиций в звуковом буфере считается событием в подсистеме DirectSound. Для запроса уведомления о наступлении таких событий приложение может использовать специальный интерфейс IDirectSoundNotify, создавая соответствующие ему следящие объекты.

При достижении указанных позиций, следящий объект активизирует (set) заданные объекты события (event objects), которые могут быть опрошены приложением непосредственно, либо может быть создана отдельная задача, ожидающая активизации одного или нескольких объектов событий.



## 23.4. Уровни взаимодействия

DirectSound вводит четыре уровня взаимодействия (cooperation levels) приложений между собой и звуковым адаптером. Когда несколько приложений **одновременно** используют один и тот же адаптер, соотношение уровней взаимодействия определяет их приоритетность в использовании аппаратуры и создании звучания.

- 1) **Обычный (normal)** уровень фиксирует формат первичного буфера адаптера:

- 22050 Гц;
- стерео;
- 8-разрядные отсчеты.

Форматы вторичных буферов преобразуются в этот формат, и при переключении приложений подсистеме нет необходимости изменять формат первичного буфера.

На этом уровне достигается наибольшая универсальность и эффективность. Качество звука в таком формате посредственно и не допускается уплотнение внутренней памяти адаптера.

- 2) **Приоритетный (priority)** уровень позволяет приложению устанавливать формат первичного буфера и уплотнять внутреннюю память адаптера. Предоставляет приоритетный доступ к *аппаратным ресурсам*, когда окно приложения становится *активным* (foreground).

Если происходит переключение между приложениями этого уровня, установившими различные форматы первичного буфера — подсистема вынуждена переключать форматы, для чего необходим перезапуск адаптера, нередко порождающий **щелчки** и тому подобные помехи.

- 3) **Исключительный (exclusive)** уровень подобен приоритетному, но на время активности окна приложения ему предоставляется исключительный доступ к адаптеру, и звучание источников всех остальных приложений заглушается (но не останавливается).
- 4) Уровень доступа к **первичному буферу** (write-primary) разрешает приложению прямую запись в первичный буфер адаптера. На этом уровне приложение может работать только с первичным буфером, активизация вторичных буферов запрещена.

Этот уровень доступен только для устройств, имеющих специализированный DirectSound-драйвер.

## 23.5. Наборы свойств

DirectSound вводит понятие набора свойств (property set) — параметров, описывающих виды обработки звука. При помощи набора свойств можно описать:

- параметры зала;
- голоса исполнителя;
- манеры пения;
- звучания инструментов и т.п.

При наличии необходимых средств обработки одну и ту же звуковую картину можно представлять в разных ракурсах, активизируя нужные наборы свойств.

В данное время эти виды обработки почти не поддерживаются; наборы свойств введены в основном на будущее.

## 23.6. Идентификация устройств

В отличие от подсистем MME, идентификация устройств в DirectSound следует правилам COM и использует GUID (Globally Unique Identifier — идентификатор, уникальный в мировом масштабе). Любой объект COM имеет свой идентификатор, по которому приложения могут обращаться к нему. Идентификаторы доступных устройств приложение получает в процессе перебора (enumeration) устройств заданного класса.

## 23.7. Системные особенности

Интерфейсы DirectSound доступны для платформ Windows 98 или 2000 и выше.

Стоит заметить, что интерфейсы DirectSound и его модификации для Windows Vista являются базовыми. Поддержки MME в Vista уже нет. Очевидно, нет ее и в Windows 7.

## 23.8. Алгоритм взаимодействия

Рассмотрим упрощенный алгоритм взаимодействия программы и звуковой подсистемы:

### 23.8.1. Воспроизведение

```
1: идентификатор ← «устройство по умолчанию»;
2: Если <конкретное устройство> то:
3:   идентификатор ← перебор; /* DirectSoundEnumerate */
4:   устройство /* IDirectSound */ ← создать( <идентификатор> );
5:   /** Созданный объект устройства может быть опрошен методом GetCaps, возвращающим его характеристики и возможности. Таким образом может быть, например, найдено минимально и оптимально подходящее для целей приложения устройство из всех имеющихся в системе. **/
6:   установить уровень взаимодействия; /* SetCooperativeLevel */
7:   Если <уровень != normal > то:
8:     создать первичный буфер; /* CreateSoundBuffer */
9:     задать его формат; /* SetFormat */
10:    /** На обычном уровне взаимодействия формат первичного буфера фиксирован — 22 050 Гц, стерео, восемь разрядов. **/
11:    Для всех <источники звука> выполняем:
12:      создать вторичные буферы; /* CreateSoundBuffer */
13:      Пока заполнение данными выполняем:
14:        указатели доступных участков буфера ← Lock;
15:        Пока <есть потерянный буфер> /* от методов Lock или Play */
16:          выполняем:
17:            Restore;
18:            заполнить участки данными;
19:            UnLock;
20:        /** При желании приложение может разделить один и тот же экземпляр звучания между несколькими объектами буферов, создавая объекты-копии методом DuplicateSoundBuffer. **/
21:      Если <определить звучащий фрагмент> то:
22:        GetCurrentPosition;
23:      Если <запуск воспроизведения> то:
24:        Play;
25:      Если <запуск с фрагмента> то:
26:        SetCurrentPosition;
27:      Если <остановка воспроизведения> то:
28:        Stop;
29:      Если <изменить частоту дискретизации> то:
30:        SetFrequency( <частота> );
31:      Если <изменить громкость> то:
32:        SetVolume( <громкость> );
33:      Если <изменить положение на панораме> то:
34:        SetPan( <координаты> , <ориентация> , <скорость> );
35:      уничтожить объекты буферов; /* Release */
36:      уничтожить объекты устройств;
```

### 23.8.2. Запись

```
1: идентификатор ← «устройство по умолчанию»;
2: Если <конкретное устройство> то:
3:   идентификатор ← перебор; /* DirectSoundCaptureEnumerate */
4: устройство /* IDirectSoundCapture */ ← создать ( <идентификатор>
   );
5: создать буфер захвата; /* CreateCaptureBuffer */
6: задать его формат; /* CreateCaptureBuffer */
7: Пока заполнение данными выполняем:
8:   указатели доступных участков буфера ← Lock;
9:   Пока <есть потерянный буфер> выполняем:
10:    Restore;
11:    заполнить участки данными;
12:    UnLock;
13: Если <запуск записи> то:
14:   запустить захват звука; /* Start */
15: Если <остановка записи> то:
16:   остановить запись; /* Stop */
17: уничтожить объекты буферов; /* Release */
18: уничтожить объекты устройств;
```

Функции перебора требуют указания локальной перебирающей функции приложения (callback), которая будет вызываться для каждого доступного устройства. Перебирающая функция может либо самостоятельно выбрать подходящее устройство, либо сформировать полный список устройств, из которого пользователь сделает выбор по своему усмотрению.

Приложение может разделить один и тот же экземпляр звучания между несколькими объектами буферов, создавая объекты-копии методом **DuplicateSoundBuffer**. Изменяя параметры звучания, можно получать различные звуки на основе одной и той же оцифровки, не расходуя дополнительную память.

Приложение может воспользоваться интерфейсом уведомления **IDirectSoundNotify**, запрашивая его у объектов тех буферов, для которых требуются уведомления, и заказывая установку заданных объектов программных событий (event objects) по достижении определенных позиций в буфере.



# Предметный указатель

44.1 кГц [16](#)

48.0 кГц [16](#)

## А

AAC [65](#)

Ableton Live [58](#)

ACM [69](#)

ACM Codec [69](#)

Advanced Audio Coding [65](#)

ALSA [66](#)

Amplitube X-Gear [40](#)

ASIO [66](#)

AT&T [65](#)

attack [19](#), [20](#)

Audio CD [16](#)

## В

BPF [22](#), [24](#)

## С

Cakewalk Sonar [58](#)

chorus [26](#), [51](#)

COM [74](#)

Core Audio [66](#)

Creative [42](#)

Creative Technology [42](#)

Cubase [58](#)

## Д

DCB-Roland [57](#)

delay [28](#)

с деградацией [28](#)

digital delay [28](#)

depth [27](#)

DirectSound [66](#), [74](#)

GUID [81](#)

курсоры [78](#)

capture [78](#)

play [78](#)

read [78](#)

write [78](#)

помехи [80](#)

уровни [80](#)

щелчки [80](#)

Dolby [65](#)

DSP [43](#)

dual channel [64](#)

## Е

E-mu [45](#)

Extended General MIDI [52](#)

## Ф

feedback [27](#), [28](#)

Fiji [75](#)

FireWire [43](#)

flanger [26](#)

FM-синтез [35](#)

FM8 [36](#)

Fraunhofer IIS [63](#), [65](#)

## Г

gain [19](#)

General MIDI [51](#)

General Synthesis [51](#)

## Н

hard knee [19](#)

HPF [22](#), [23](#)

Hurricane [75](#)

## И

IDirectSoundNotify [79](#)

IK Multimedia Amplitube [40](#)

## Ј

joint stereo [64](#)

intensity [64](#)

ms [64](#)

ms/is [64](#)

## К

knee

hard [19](#)

soft [19](#)

Korg [45](#)

## Л

Lame encoder [64](#)

Logana [40](#)

LPF [22](#), [23](#), [34](#)

## М

Microsoft [65](#)

MIDI [43](#), [45](#), [51](#)

chorus [51](#)

Extended General [52](#)

General [51](#)

General Synthesis [51](#)

GS [51](#)

Modulation [51](#)

Pan [51](#)

Volume [51](#)

## *М* (продолжение)

MIDI 43, 45, 51  
  XG 52  
  протокол 48  
  реверберация 51  
  системы 51  
  файл 51  
MIDI-GS 51  
MIDI-контроллеры 46  
MIDI-системы 51  
MME 66  
  HWAVE 71  
  HWAVEIN 71  
  HWAVEOUT 71  
  блок 68  
  формат потока 68  
MP3 63  
  dual channel 64  
  joint stereo 64  
    intensity 64  
    ms 64  
    ms/is 64  
  Lame encoder 64  
  standard stereo 64  
  поток 64  
  файл 64  
MPEG 63  
  1 63  
  2 63

## *Н*

NanoWave 38  
Native Instruments 36, 38  
NCO 34  
NEC 65  
Notch filter 22  
NoteWorthy 58  
NTT 65  
NWC 58

## *О*

OSS 66

## *Р*

PCI 43  
PCM 67, 74  
phaser 26  
predelay 29  
Propellerhead Reason 57

## *Р*

range 20  
rate 27  
ratio 19, 20  
Reactor 38  
release 19, 20

## *Р* (продолжение)

Roland 45, 51, 57  
Roland Sonar 58  
room size 29  
room type 29

## *С*

S/P-DIF 43  
S/PDIF 43  
Sample Based синтез 39  
soft knee 19  
Sonar 58  
Sound Blaster 75  
standard stereo 64  
Steinberg Cubase 58

## *Т*

Thomson 63  
threshold 19, 20  
time 28, 29  
Turtle Beach Tahiti 75  
TwinVQ 65

## *U*

USB 43

## *V*

VA 40  
Virtual Analog 40  
VQF 65

## *W*

Wave Mapper 69  
wave shaper 37  
Windows 81  
  7 81  
  Vista 81  
Windows Media Audio 65  
Windows Sound System 75  
WMA 65  
WT-синтез 38, 51

## *Y*

Yamaha 45, 52, 65

## *А*

АЦП 15, 67

## *В*

Вебер 7

## *Г*

ГУК 32, 34  
Гиббс 10

## И

ИКМ 67

## К

Котельников 16

## Н

Найквист 16

## Ф

ФВЧ 22, 23

ФНЧ 22, 23, 34

Фехнер 7

Флетчер 59

## Х

Хаффман 63

## Ц

ЦАП 34, 67

## Ч

ЧМ-синтез 35

## Ш

Шенон 16

## а

алгоритм Хаффмана 63

аппаратный буфер 76

атака 19, 20, 31, 38

## б

банк инструментов 51, 52

биение 14

блок 68

блок синтезатора 42

буфер 67, 75

аппаратный 76

вторичный 76

первичный 76

поточный 77

программный 67, 76

статический 77

воспроизведения 78

захвата 78

## в

волна

звуковая 6

время 28, 29

вторичный буфер 76

## г

гейт 20

глубина 27

графический эквалайзер 22, 25

## д

давление звуковое 7

динамический диапазон 18

дискретизация 16

интервал 16

## ж

жесткое колено 19

## з

задержка начала 29

закон

Вебера-Фехнера 7

затухание 19, 20, 31, 38

зв-блок 42

звук

давление 7

интенсивность 7

звуковая волна 6

звуковая плата

DSP 43

FireWire 43

MIDI 43

PCI 43

S/P-DIF 43

S/PDIF 43

USB 43

АЦП 42

ЦАП 42

зв-блок 42

синтезатор 42

звуковое давление 7

## и

интенсивность звука 7

интервал дискретизации 16

искажение 11

биение 14

интермодуляционное 13

линейное 11

наводки 14

нелинейное 13

перегрузка 13

помехи 12, 14

разбалансировка 12

шумы 14

## К

квантование 17  
колесо  
жесткое 19  
мягкое 19  
кольцевая модуляция 37  
кольцевой буфер 75  
компрессор 19  
критическая полоса 59, 63

## Л

лимитер 19

## М

маскирование 59, 63  
вперед 59  
временное 59  
назад 59  
частотное 59  
метод волновой формы 37  
модуляция  
кольцевая 37  
частотная 35  
мягкое колесо 19

## Н

нелинейный синтез 37

## О

обратная связь 27, 28  
оверсэмплинг 16  
отношение 19, 20

## П

параграфический эквалайзер 22, 25  
параметрический эквалайзер 22, 25  
паттерновый секвенсор 57  
первичный буфер 76  
перегрузка 13  
переходные помехи 12  
поддержка 31, 38  
подключение «в разрыв» 18  
полоса критическая 59  
полосовый фильтр 22, 24  
помехи 14  
наводки 14  
переходные 12  
шумы 14  
порог 19, 20  
слышимости 7  
болевой 7  
поточковый буфер 77  
преобразование аналогово-цифровое 15  
программный буфер 76  
психоакустическая модель 62  
психоакустическая таблица 63

## Р

разбалансировка  
громкости 12  
помехи 12  
фазовая 12  
реверберация 29, 51  
режекторный фильтр 22

## С

секвенсор 57  
паттерновый (шаговый) 57  
линейный 58  
синтез 31  
FM 35  
Sample Based 39  
VA 40  
WT 38, 51  
ЧМ 35  
нелинейный 37  
субтрактивный 33  
физическое моделирование 40  
синтез звука 31  
синтезатор 51  
статический буфер 77  
субтрактивный синтез 33  
сэмпл 38  
атака 38  
multi 38  
затухание 38  
многослойный 38  
поддержка 38

## Т

теорема  
Котельникова-Найквиста-Шенона 16  
Котельникова-Шенона-Найквиста 16  
токовая петля 46

## У

усиление 19

## Ф

фазер 26  
фильтр 22  
высоких частот 22, 23  
низких частот 22, 23, 34  
полосовый 22, 24  
режекторный 22  
частотный 22  
фильтрация 16  
фленджер 26  
формат потока 68  
формирователь фазы 34

## Х

хорус 26



## ц

цепочка буферов 67

## ч

частота 27

частотная модуляция 35

частотное маскирование 59

## ш

шаговый секвенсор 57

шум квантования 17, 62

## э

эквалайзер 25

графический 22, 25

параграфический 22, 25

параметрический 22, 25

экспандер 20

эффект

chorus 26

delay 28

flanger 26

phaser 26

хорус 26

модуляционный 26

реверберация 29

фазер 26

фленджер 26

эхо 28

эффект Гиббса 10

эхо 28

с деградацией 28

digital delay 28

## Список литературы

- [1] Крапивенко А.В., «Технологии мультимедиа и восприятие ощущений», Москва М.: БИНОМ. Лаборатория знаний, 2009;
- [2] Крапивенко А.В., «Методы и средства обработки аудио- и видеоданных», Москва М.: «Вузовская книга», 2010;
- [3] Психоакустика: материалы с сайта [websound.ru](http://websound.ru);
- [4] Программный интерфейс ММЕ: материалы с сайта [rsdn.ru](http://rsdn.ru);
- [5] Программный интерфейс DirectSound: Журнал «Компьютер Пресс», сайт: [compress.ru](http://compress.ru).
- [6] Протокол MIDI: Евгений Музыченко, «Описание интерфейса MIDI», сайт: [opennet.ru](http://opennet.ru)