Глава VI. Музыкальные средства MSX BASIC

> У меня под руками оказался жалкий спинет; и вскоре я начал писать ноты, вот и все.
>
> *—Дж. Верди*

MSX BASIC включает в себя Музыкальный Макро–Язык (MML), позволяющий управлять четырёхканальным звуковым генератором компьютера. Этот генератор работает как независимый процессор: если музыкальные или другие звуковые эффекты запрограммированы, то они запускаются параллельно с использованием основной программы.

Звукогенератор имеет три независимых выходных канала (A, B, C), допускается тем самым трёхголосный полифонический выход. Звуковой сигнал формируется генератором регулярных сигналов, частота которых (высота звука) может программироваться, либо широкополосным генератором шумов. Амплитуда сигнала(громкость) может устанавливаться для каждого сигнала отдельно или модулироваться генератором огибающей.

MML — это Музыкальный Макро–Язык («Musical Macro–Language»), в некотором смысле аналогичный GML ("Graphics Macro Language").

MML использует для записи нот символы:

C	D	E	F	G	A	B
до	ре	ми	фа	соль	ля	си

которые вызывают исполнение соответствующей ноты в текущей октаве.

Для обеспечения хроматизма игры ноты можно модифицировать добавлением символов: «#», «+» — *диез*, «-» — *бемоль*, т.е. «D#» и «D+» это ре–диез, «G-» соль–бемоль и т.д. Запомните: «B+» соответствует «C» *той* же октавы.

После символов C, D, E, F, G, A, B, Вы можете указать *длительность* ноты — любое целое число k, принадлежащее отрезку [1,64] (длительность ноты получается по формуле 1/k), так, например,

- C1 — *целая* нота «до»,
- C2 — *половинная* нота «до»,
- C4 — *четвертная* нота «до».

К сожалению, длительность ноты в этом случае не может задаваться с помощью переменной. После указания ноты и длительности можете добавить необходимое вам количество *точек* («.»), первая из которых удлинит ноту наполовину, а роль последующих аналогична их роли на нотном стане (вторая точка добавляет половину длительности, определяемой первой точкой).
Например, нота «C4..» имеет длительность 1/4+1/8+1/16=7/16.

Если длительность не указана, то компьютер играет «в четвертях».

В качестве символов–разделителей нот можно использовать либо точку с запятой, либо пробелы, либо ноты вообще не разделять, например: «C;D;E» или «C D E» или «CDE».

VI.1. Описание команд MML

Приведём список команд MML. Каждая команда представлена одиночной ключевой буквой (L, O, N, T, V, S, M, R), за которой следует один числовой параметр (аргумент).

1. *Команда* Lk («Length» — «длительность»), где k — целое число, принадлежащее отрезку [1,64], устанавливает текущую длительность для всех последующих нот.

   Если длительность не будет задана или не будет указана после ноты, то по умолчанию k=4 (четверть).

   Код длительности звука равный 1 соответствует целой длительности, 2 — половинной, 4 — четвертной, 8 — восьмой, 16 — шестнадцатой, 32 — тридцать второй, 64 — шестьдесят четвёртой. Таблица из [76] , стр. 105.

Команда MML	Нота
L1	
L2	
L4	
L8	
L16	
L32	
L64	

Например, команда L8 установит длительность звука равной восьмой.

Заметим,что «C32» равносильна «L32C», но учтите, что команда L действует и на все последующие ноты до её «сброса» оператором BEEP (см.раздел VI.2.) или командой L с другим аргументом.

Команда Lk называется командой задания длительности *группы* нот.

Текущую длительность можно задать и с помощью переменной (см.ниже).

2. *Команда* Ok («Octave» — «октава»), где k — целое число, принадлежащее отрезку [1,8], позволяет выбрать октаву. Диапазон MML 8 октав (наименьшая октава O1, а наибольшая — O8).

   Октава O4 устанавливается компьютером по умолчанию (она соответствует первой октаве рояля).

   Напомним, что C — самая низкая нота, а B — самая высокая нота в октаве. Вы можете указать новую октаву в любой момент времени, когда вам потребуется нота вне текущей октавы.

   Команда O влияет на все последующие ноты, пока октава не будет переопределена.

3. *Команда* Nk («Note» — «нота»), где k — целое число, принадлежащее отрезку [1,96], позволяет кодировать ноты целыми числами.

   Самая низкая нота — «N1», самая высокая — «N96»,при этом «O4C» и «N36» — это одна и та же нота! Вообще, доступна« *любая* нота, т.е. «N37» — это нота «C+» 4–й октавы, а не нота «D» 4–й октавы!

   Пример.
   061-01.bas
   061-01.bas

   ```
   10 FOR I=1 TO 96:CLS:PRINT"Номер ноты";I:PLAY"N=I;"
   20 FOR D=1 TO 300:NEXTD:NEXTI
   ```

   Для записи нот вне текущей длительности, Вы должны записать символ »;« после номера ноты и только потом указать её длительность, например: "O4C8" — соответствует "N36;8". Таким образом, можно задавать индивидуальную длительность.

4. *Команда* Tk («Tempo» — «темп»), где k — целое число, принадлежащее отрезку [32,255], устанавливает темп исполнения музыки (по умолчанию k=120). Это значение задаёт количество четвертей, проигрываемых в минуту.

   Например, команда T120 установит разновидность темпа Allegro («скоро»).

   Забегая вперёд, отметим, что оператор BEEP сбрасывает текущий темп на темп, принятый по умолчанию.

5. *Команда* Vk («Volume of sound» — «громкость звука»), где k — целое число, принадлежащее отрезку [0,15],устанавливает громкость исполнения музыки. По умолчанию k=8.

   Громкость сохраняется до следующей команды V (или оператора BEEP).

Заметим, что команда V «сбрасывает» значения аргументов команд S и M (см.ниже пункты 6 и 7) на их значения по умолчанию. Приведём примеры: 061-02.bas
061-02.bas

```
10 FOR V=0 TO15:CLS:?"Volume";V
20 PLAY"V=V;C":FOR D=1 TO 500
30 NEXTD:NEXTV
```

061-03.bas
061-03.bas

```
10 BEEP
20 PLAY"V15O5":FOR N=1TO10
30 PLAY"CE":NEXTN
```

6. MML позволяет работать с восемью различными *волновыми пакетами*, или *формами волны*. Форма волны определяет параметры звука с точки зрения изменения его громкости (амплитуды). Так, при одной форме волны звук, постепенно нарастающий по громкости, внезапно резко обрывается, другой волновой пакет соответствует обратной ситуации. Можно также получить с помощью компьютера трели, свист и другие интересные звуковые эффекты. Однако,одновременно допускается пользоваться лишь одной из восьми форм, и разным звуковым каналам не могут соответствовать различные пакеты.

   Команда Sk («Shape» — «форма»), где k — целое число, принадлежащее отрезку [0,15], устанавливает форму волны («конверта»). Параметр k (цикл модуляции) определяет несущую частоту звукового генератора. По умолчанию k=1.

   Ниже показаны допустимые формы волновых пакетов [1].

   - S=0,1,2,3,9
   - S=4,5,6,7,15
   - S=8
   - S=10
   - S=11
   - S=12
   - S=13
   - S=14

Выполнив следующую программу, Вы услышите особенности звучания каждого из восьми пакетов.

Пример [76]. *Восемь* волновых пакетов
061-04.bas
061-04.bas

```
20 CLS:A$="M500O4L4DF#ARAF#D"
40 FOR I=1 TO 8
50    READ S$:PRINT"Волновой пакет: ";S$:PLAY S$:PLAY A$
80    FOR J=1 TO 2000:NEXT J
90 NEXT I
100 END
110 DATA s1,s4,s8,s10,s11,s12,s13,s14
```

Все пакеты можно подразделить на два типа: *«непрерывные»* и *«скачкообразные»*.

В непрерывных пакетах звук изменяется непрерывно. К ним относятся пакеты 8, 10, 12, 14.

В «скачкообразных» пакетах происходит однократная модуляция. При этом всякий раз, когда требуется получить данный звук, необходимо записывать номер пакета в 13–й регистр Программируемого Звукового Генератора.

Форма огибающей сбрасывается только оператором BEEP, команда L командой V или меняется командой S с другим значением аргумента.

Чистые обертоны звучат при S = 1, 4, 8, 10, 11, 12, 13, 14.

Напомним, что *музыкальный звук* — комплекс основного тона и гармонических обертонов или частичных тонов. Обертоны звучат слабее основного тона, слитно с ним и на слух не распознаются. Наличие и сила каждого из них определяет окраску или тембр звука.

Команда S в основном используется совместно с командой M для создания музыкальных эффектов. К сожалению, нет ни руководств, ни правил, которые можно бы было здесь применить, и все, что можно сделать — это экспериментировать при помощи компьютера!

Важным моментом является то, что команда S «сбрасывает» громкость, установленную с помощью команды V. Заметим, что для одного и того же канала синтезатора звуков команды S и M нельзя использовать одновременно с командой V: или только команды S и M, или только команда V!

7. *Команда* Mk («Modulation» — «модуляция, изменение частоты»), где k — целое число, принадлежащее отрезку [0,65535], позволяет установить несущую частоту звукового генератора (период «конверта», выбранного командой S). По умолчанию k=255.

   Код периода «конверта» (значение k) можно вычислить по формуле:

   ```
   k = 6991.3×Период конверта (в секундах)
   ```
   .

   Например, команда M13982 устанавливает длительность звука равную 2 сек. Действительно, 6991.3×2 ≈ 13982.

   Из приведённой формулы ясно, что чем меньше значение k, тем меньше длительность звука (выше частота модуляции).

   Оператор BEEP сбрасывает значение аргумента команды M на значение по умолчанию.

   Примеры [76].

   - 1)
     061-11.bas
     061-11.bas

     ```
     10 ' Изменение частоты
     20 A$="S8O4L12CGAO5CDGA"
     30 PLAY"M200":PLAY A$
     40 PLAY"M1000":PLAY A$
     50 PLAY"M5000":PLAY A$
     60 END
     ```

   - 2)
     061-12.bas
     061-12.bas

     ```
     10 ' Звуки (1)
     20 PLAY"T250L4"
     ```

```
30 FOR I=1 TO 4
40    PLAY"S14M400O4CAB"
50    PLAY"S8M600O6CO3CO2C"
60    PLAY"S14M400O4CO5BAO4CA"
70    PLAY"S8M600O4CO2CO3C"
80 NEXT I
90 END
```

- 3)
  061-13.bas
  061-13.bas

```
10 'Звуки (2)
20 PLAY"T120L2"
30 PLAY"S13M9000O3F+RGACD"
40 PLAY"S13M9000O3F+RGACO2A"
50 FOR I=1 TO 4
60    PLAY"S1M1500L64O5F+GACO4A"
70    PLAY"S11M1900L64O3F+GACO2A"
80 NEXT I
90 END
```

- 4)
  061-14.bas
  061-14.bas

```
10 CLS:INPUT"Начало модуляции";X
20 INPUT"Конец модуляции";Y:INPUT"Шаг модуляции";Z
30 CLS:FOR S=1 TO 8:READ I:PRINT"Форма";I
40 FOR M=X TO Y STEP Z:BEEP:PRINT"Модуляция";M
50 PLAY"S=I;M=M;B":FOR D=1TO500:NEXTD:NEXTM:NEXTS
60 DATA 0,4,8,10,11,12,13,14
```

Как уже говорилось, имеется 8 «конвертов», которые могут быть выбраны командой S. Период каждого «конверта» можно уменьшить, уменьшив значение аргумента команды M (другими словами, количество «пилочек» в единицу времени увеличивается). Думается, что после «прослушивания» следующих двух операторов, разница между различными «конвертами» станет очевидной:

```
PLAY"S8M900cdefgab"
PLAY"S10M6000cdefgab"
```

.

Однако, если значение аргумента команды M становится «слишком большим», то период также возрастает и различие предыдущих операторов исчезает.

Попытаемся убедить Вас в этом. Рассмотрим операторы:

```
PLAY"S8M6000cdefg"
```

.

и

```
PLAY"S10M6000cdefg"
```

К нашему удивлению, звучит одинаковая «музыка»! Однако, изобразим «конверты», соответствующие каждому оператору:

```
          ▲
          |                         |
S8 M6000  |                         |
          |_________________________|▶
     ▲ исполняемая музыка
          |                         |
S10 M6000 |                         |
          |_________________________|▶
       исполняемая музыка
```

... и «тайное становится явным»! Однако учтите, что длительность исполняемого фрагмента можно легко изменить командой L.

8. *Команда* Rk («Rest» — «пауза»), где k — целое число, принадлежащее отрезку [1,64], генерирует специальную «молчащую» ноту, *паузу* (звук отсутствует!). По умолчанию k=4.

Код длительности паузы равный 1 соответствует целой длительности, 2 — половинной, 4 — четвертной, 8 — восьмой, 16 — шестнадцатой, 32 — тридцать второй, 64 — шестьдесят четвёртой.

Например, команда R4 установит паузу четвертной длительности. Таким образом:

R1	R2	R4	R8	R16	R32	R64

Вы можете модифицировать R таким же образом,как,например,ноту «C» или любую другую ноту: для задания длительности, увеличенной на свою половину, используется код длительности с точкой (допускается любое количество точек. Так, код 1. соответствует целой длительности с точкой,

- код 2. соответствует половинной с точкой,
- код 4. соответствует четвертной с точкой,
- код 8. соответствует восьмой с точкой,
- код 16. соответствует шестнадцатой с точкой,
- код 32. соответствует тридцать второй с точкой,
- код 16. соответствует шестьдесят четвёртой с точкой.

Например, команда R2 установит паузу половинной с точкой длительности.

Выдерживая при кодировании полную длительность каждого звука, мы получим исполнение legato.

Для исполнения non legato выдерживается 7/8 указанной длительности звука, а 1/8 приходится на паузу. В некоторых случаях выдерживается от 7/8 до 15/16 указанной длительности звука, а от 1/8 до 1/16 приходится на паузу.

Для исполнения staccato выдерживается 3/4 указанной длительности звука, а 1/4 приходится на паузу, например:

```
"C5R16C5R16C5R16"
```
.

В приведённой ниже таблице приведены коды основных длительностей, составляющих при исполнении legato указанную длительность, при исполнении non legato — 7/8 и 1/8 указанной длительности, при исполнении staccato — 3/4 и 1/4 указанной длительности.

Название длительности	**legato**	**non legato**		**staccato**	
	1 длительности	**7/8 длительности**	**1/8 длительности**	**3/4 длительности**	**1/4 длительности**
Целая с точкой	1+2	1+4+16	8+6	1+8	4+8
Целая	1	2+4+8	8	2+4	4
Половинная с точкой	2+4	2+8+32	16+32	2+16	8+16
Половинная	2	4+8+16	16	4+8	8
Четвертная с точкой	4+8	4+16+64	32+64	4+32	16+32
Четвертная	4	8+16+32	32	8+16	16
Восьмая с точкой	8+16	—	—	8+64	32+64
Восьмая	8	16+32+64	64	16+32	32
Шестнадцатая с точкой	16+32	—	—	—	—
Шестнадцатая	16	—	—	32+64	64

Например, чтобы закодировать какую–либо ноту четвертной длительности при исполнении non legato, мы должны установить согласно этой таблице длительность звука равную сумме восьмой, шестнадцатой и тридцать второй (7/8 четвертной длительности) и длительность паузы равную тридцать второй (1/8 четвертной длительности).

9. *Команда* XA; (символ «;» обязателен!), где A — строковая переменная, выполняет «подпрограмму» на MML, которая определяется значением строковой переменной.

Пример.
061-05.bas
061-05.bas

```
10 A$="DCF"
20 PLAY "V15XA$;V6XA$;"
```

Подпрограммы могут быть вложенными, что упрощает запись повторяющихся данных. Все повторяющиеся части звуковой партии могут быть записаны в символьных переменных и в случае необходимости вызываются командой X.

Внимание ! Аргумент команд L, O, V, T, M, S, N можно задавать именем переменной, используя конструкцию вида:

```
Команда = Имя переменной;
```

(наличие символа »;« обязательно!).

Например, команда T=k; установит, в зависимости от значений переменной k (32≤k≤255), разновидность темпа от Larghissimo («очень протяжно») до Prestissimo («предельно быстро»).

Заметим, что это эффективный способ сочинения новых мелодий!

В качестве символов–разделителей команд можно использовать либо точку с запятой (для удобства чтения), либо пробелы, либо вообще не разделять команды. Например: «L8;O5;V6» или «L8 O5 V6» или «L8O5V6».

Как уже упоминалось, BEEP (или ошибка в программе) сбрасывает значения аргументов всех команд на значения по умолчанию.

Забегая вперёд, отметим, что значения аргументов команд M, O, L, T, V хранятся в системной области ОЗУ (Оперативного Запоминающего Устройства) компьютера в ячейках с номерами &HF97A, &HFB50, &HFB51, &HFB52, &HFB53 соответственно.

VI.2. Оператор PLAY. Функция PLAY. Оператор BEEP

А Вы ноктюрн сыграть могли бы
На флейте водосточных труб?

—В. Маяковский

Синтаксис оператора:

```
PLAY [A][,B][,C]
```
,

где:

- PLAY («to PLAY» — «играть») — служебное слово;
- A, B, C — строковые выражения, значения которых содержат команды MML.

Оператор PLAY содержит одну, две или три строки (для одного, двух или трёх голосов), поэтому можно добиться одновременного звучания не более *трёх* голосов! Приведём примеры:

1. Аккорд ми–минор(4–я октава).
   062-01.bas
   062-01.bas

```
10 A$="L2M10000S1E"
20 B$="L2M10000S1G+"
30 C$="L2M10000S1B"
40 PLAY A$,B$,C$
```

2. Аккорд до–мажор(4–я октава).
   062-02.bas
   062-02.bas

```
10 A$="L4M8000S1C"
```

```
20 B$="L4M8000S1E"
30 C$="L4M8000S1G"
40 PLAY A$,B$,C$:GOTO 40
```

3. 062-03.bas
062-03.bas

```
10 A$="O4CDO3BO4E2R4"
20 B$="O4EFDG2R4"
30 C$="O4GAGO5C2R4"
40 PLAY A$,B$,C$
```

Если для одного из трёх голосов указана пустая строка, то данный голос будет «молчать». Например:
062-04.bas
062-04.bas

```
10 PLAY"O4L8CDEFGAB"
20 PLAY"L8CC+DD+EFF+GG+AA+B"
```

Будет исполнена гамма до–мажор и хроматический ряд в 4–й октаве.

Когда выполняется оператор PLAY, строки, соответствующие каждому голосу (содержащие максимум 255 символов) записываются в трёх так называемых *очередях*. Затем они автоматически исполняются, в то время как основная программа может «заниматься» чем–либо другим. После выполнения команды макроязыка удаляются из очередей. Когда все очереди пусты, музыка прекращается.

Далее, может выполняться новый оператор PLAY, но его выполнение, как правило, должно начаться сразу же после считывания из очереди последней команды предыдущего оператора PLAY, и даже в этом случае загрузка новых очередей вызовет задержку.

Следующая программа наглядно демонстрирует описанный выше эффект очереди. Замысел состоял в том, чтобы последовательное исполнение десяти нот одновременно сопровождать индикацией на дисплее названия ноты. Как легко убедиться, названия нот появляются почти одновременно в виде списка, без всякой связи с их исполнением.

Пример.
062-051.bas
062-051.bas

```
30 CLS
40 PLAY"T255O5L16"
50 FOR I=1 TO 10
60    READ A$:PRINT"Название ноты: ";A$:PLAY A$
90 NEXT I
100 END
110 DATA C,A,E,F,G,B,D,G,E,C
```

Наиболее простое решение проблемы сводится к организации цикла временной задержки внутри временного основного цикла FOR. Период задержки необходимо согласовать с темпом и длительностью звучания нот.

В предложенном примере проблема будет решена, если Вы вставите строку:
062-052.bas
062-052.bas

```
85 FOR J=1 TO 100:NEXT
```

В MSX BASIC имеется функция, выясняющая, имеются данные в очередях или нет. Эту функцию (PLAY) не следует путать с оператором PLAY.

Функция PLAY определяет, есть ли ещё ждущая в очереди несыгранная музыка. Её общий вид:

```
PLAY (голос)
```
,

где *голос* — арифметическое выражение, целая часть значения которого принадлежит отрезку [0,3]. Функция сообщает, есть ли ещё музыка в очереди и в процессе игры в качестве фоновой задачи, причём `PLAY(0)` сообщает, есть ли ещё несыгранная музыка, `PLAY(1)` проверяет только голос A, `PLAY(2)` — голос B, `PLAY(3)` — голос C. Функция возвращает 0, если музыки в очереди нет, если же в очереди «что-нибудь» есть и играется, то функция возвращает (-1).

Приведём пример использования функции `PLAY`:

```
10 PLAY A$,B$,C$
20 IF PLAY(0) THEN 20 ELSE READ A$:PLAY A$
  'Выполнение программы не продолжится, пока музыка не будет сыграна!
```

Оператор `BEEP` используется для вывода звукового сигнала продолжительностью 0.04 с частотой 1.3 Гц и инициализации синтезатора звуков: установки всех его регистров в начальное состояние и очистку очередей всех его каналов от последовательности музыкальных команд оператора `PLAY`. Его синтаксис:

```
BEEP
```
,

где `BEEP` («бип», звукоподражание) — служебное слово.

Оператор `BEEP` нетрудно моделировать оператором `PLAY`:

```
PLAY"O6 E64"
```

или

```
PLAY"M225 V8 L64 O6 T120 S1 E"
```

естественно без инициализации синтезатора звука!

Таким образом, оператор `BEEP` можно применять в следующих случаях.

Во-первых, оператор `BEEP` используется для сигнализации о наличии ошибок в программе пользователя, т.к. любая ошибка, допущенная в программе на языке MSX BASIC, влечёт за собой выполнение `BEEP` с одновременным сообщением об ошибке.

Например:

```
10 INPUT"Номер месяца";MN
20 IF MN>12 OR MN<1 THEN BEEP:GOTO 10 ELSE PRINT DAY$(MN)
```

Несколько звонков подряд дадут длительный сигнал и привлекут, если требуется Ваше внимание, например:

```
FOR I=1 TO 10:BEEP:NEXT
```

Во-вторых, для предоставления пользователю возможности прекратить в любой момент воспроизведение музыки или звуковых эффектов (в командном режиме наберите `BEEP` или `PRINT CHR$(7)`, или нажмите клавиши CTRL+G).

Определённая усидчивость и аккуратность позволят вам перевести на MML любую партитуру из трёх или менее голосов. При правильной организации программы компьютер может исполнять музыкальное произведение пока «работают» другие части Вашей программы, т.е. как бы аккомпанировать компьютеру (это так называемое *фоновое* исполнение).

Взгляните на следующий пример. Возможно, получаемое качество звучания далеко от совершенства, тем не менее оно даёт представление о том, каких результатов Вы можете добиться.

Пример [76]. *Фоновая* музыка
062-06.bas
062-06.bas

```
20 CLS:KEYOFF:R=RND(-TIME)
40 'Описание подстрок
50 A$="o5df#ga":B$="o5gbo6cd":C$="o5ao6c#de"
80 D$="xa$;xc$;xb$;xa$;xb$;xc$;"
90 PLAY"v10t120l16"
100 PLAY D$
110 IF NOT(PLAY(1)) THEN 100
120 'Генерация случайных строк
130 W$=""
140 FOR I=1 TO 5
150    X=INT(RND(1)*26):W$=W$+CHR$(65+X)
170 NEXTI
180 PRINT W$;
190 GOTO 110
```

Теперь разберём подробнее организацию музыкальной программы и некоторые *особенности*, которые следует учесть при её составлении.

Оператор BEEP или ошибка в программе останавливают исполнение музыки.

Два последовательных оператора

```
PLAY A$:PLAY B$
```

всегда образуют короткую паузу между концом музыкальных данных первого оператора PLAY и началом музыкальных данных следующего оператора PLAY.

Оператор PLAY позволяет исполнять ноты с чёткими перерывами между ними — staccato, без перерывов — legato и с короткими перерывами, вполне достаточными для того, чтобы отличить звуки друг от друга — non legato. Как уже говорилось, исполнение staccato достигается только за счёт того, что выдерживается 3/4 номинальной длительности ноты и 1/4 оставляется на паузу. Для исполнения legato выдерживается полная длительность каждого звука, в результате чего создаётся ощущение, что соседние звуки плавно переходят один в другой. При исполнении non legato выдерживается 7/8 номинальной длительности звука, а 1/8 длительности приходится на паузу:

```
PLAY"L9 C R64 D R64 E R64 F R64"
```
.

Как уже упоминалось, паузу можно организовывать с помощью специальной «молчащей» ноты R (см.раздел VI.1.), только надо помнить, что текущая длительность не влияет на длительность «ноты» R. После R надо указывать её нестандартную длительность, например:

```
PLAY"L8 CDEF R8 GAB"
```
.

Если после R в данном случае не была бы указана длительность (8), то несмотря на текущую длительность L8, «нота» R в действительности имела бы длительность по умолчанию (4). Однако это не лучший способ для игры staccato и non legato. Гораздо проще обратиться за помощью к PSG (Программируемому Звуковому Генератору) (см. раздел VI.3.).

Если научиться умело пользоваться командами S и M, то вам будут доступны любые приёмы игры, тембры и разнообразные эффекты. Так, например, если аргумент команды S равен 1, то приёма staccato можно добиться, использовав диапазон частот 700÷3000. Если же аргумент команды S равен 1, а аргумент команды M находится в диапазоне 3000÷8000, то будет звучать non legato. При значении аргумента команды M, большего 8000, звучание нот переходит на legato. Если же аргумент команды S отличен от 1, то диапазон M, конечно, изменяется, но не очень сильно. Зато возникают новые эффекты, которыми можно разнообразить Ваше произведение. Приведём ряд примеров:

- 1)
  062-11.bas
  062-11.bas

  ```
  10 FOR I=4 TO 30 STEP 2:PLAY"SM1000L=I;O4G":NEXT:GOTO10
  ```

В этой программе параметр цикла I используется для обозначения длительности звучания ноты G (соль) в третьей октаве, причём с возрастанием I длительность звучания становится всё короче. В результате выполнения программы Вы услышите «музыку», напоминающую звук скачущего пластмассового шарика.

- 2)
062-12.bas
062-12.bas

```
5 'Пулеметная очередь!
10 Z=RND(-TIME):FOR I=1TO60
20 X=INT(RND(1)*65)
30 IF X<4 THEN 20
40 PLAY"S9M800L=X;O1G":NEXT
```

- 3)
062-13.bas
062-13.bas

```
5 '"Морзянка"!
10 Z=RND(-TIME):FOR I=1TO60
20 X=INT(RND(1)*65)
30 IF X<4 THEN 20
40 PLAY"S9M800L=X;O7C":NEXT
```

К тому же имеется возможность использовать трезвучие на 3 различных тембрах, что может придать Вашей мелодии ритмический оттенок, а также появляется возможность уместно использовать фон. Пример использования ритмического оттенка покажем на отрывке мелодии Л.Книппера «Полюшко-поле»: 062-07.bas
062-07.bas

```
10 A$="sm25000o5l4c2o4ao5co4b2ge":B$="sm2500l4o3aaaagggg"
20 C$="sm2500l4o3eeeeeeee":PLAY A$,B$,C$
```

Пример использования фона («Перепелочка», Белорусская народная песня): 062-08.bas
062-08.bas

```
10 A$="sm20000l8o5e4ao6co5b4aabo6co5b4a4a4":B$="sm20000l2o4adea"
20 C$="sm20000l2o4efg+e":PLAY A$,B$,C$
```

Конечно, использование только трёх голосов является недостатком PSG, но умелый подбор тембра, нахождение новых обертонов и эффектов компенсируют этот недостаток.

Необходимо запомнить и то, что использовав S для различных приёмов игры, Вы потеряете весь контроль громкости с помощью V. «Перебирая» аргументы команды S при определённой частоте, Вы не всегда сможете услышать заметные отличия тембров. При другой частоте и тех же значениях аргумента команды S эти отличия будут уже явными, поэтому поиск закономерностей для сочетания S и M становится похожим на прогнозирование погоды. Но интересующий Вас тембр или эффект звучания можно найти за довольно короткий промежуток времени. Вы и сами заметите, что явно отличных друг от друга звучаний не так уж и много!

Обратите также внимание на следующие *тонкие моменты*:

- α) Значения аргументов команд MML не уничтожаются оператором NEW, поэтому новую программу следует начинать с конкретного определения текущих значений аргументов команд L, M, S, O, T. Если этого не сделать, то не следует надеяться на значения аргументов, принятые по умолчанию, т.к. восприниматься будут «старые» значения аргументов. Например, последовательно выполните:

  - гамма (legato)
  062-141.bas
  062-141.bas

```
NEW
Ok
10 PLAY"cdefgab"
run
Ok
```

- гамма (отрывисто)
  062-142.bas
  062-142.bas

```
NEW
Ok
10 PLAY"sm500cdefgab"
run
Ok
```

- гамма (отрывисто)
  062-143.bas
  062-143.bas

```
NEW
Ok
10 PLAY"cdefgab"
run
Ok
```

- β) Запомните, что все текущие значения аргументов команд MML лучше всего определять в программе *явно*:
  062-15.bas
  062-15.bas

```
10 A$="S M5000 T150 O4 L4":B$="S M5000 T150 O4 L4":C$="S M5000 T150 O4 L4"
20 PLAY A$,B$,C$
```

Приведём программы, по которым компьютер исполнит несколько разных по сложности музыкальных произведений.

- 1)
  062-21.bas
  062-21.bas

```
20 '"Во саду ли,в огороде" (Русская народная песня).
30 FOR I=1 TO 2
40 PLAY"T100sm800O6L8ffeedddeffeedddR8ccO5bbaaabO6ccO5bbaaa",
   "T100sm800O4L8dfaedfdfdfaedfaR8aeeg+aeaeaeeg+aee",
   "O4L8T100sm800daac+dadadaac+dafR8acebacacacebacc"
50 A$="T100s2m2000O5L16fedfedc+edO4abO5c+dc+defedfedc+eddddcO4BaO5CO4bag+
   baef+g+ag+abO5cO4baO5cO4bag+bA4"
60 B$="t100s2m2000O4L16ddffaaeeddffddffddffaaeeddffaaeeeeg+g+aaeeaaeeaaeeeeg+g+ccee"
70 C$="t100s2m2000O4L16ddaaR16r16c+c+ddaaddaaddaaR16r16c+c+ddaaaacceebbaaccaaccaacceebbeecc"
80 PLAY A$,B$,C$:PLAY"O4a4","O4E4","O4c4":NEXT
```

- 2)
  062-22.bas
  062-22.bas

```
10 '"Гопак" (Украинский народный танец).
20 FOR I=1 TO 2
30 A$="sm2000L8O5T150D4O4AR16g16f+ef+dO3aO4df+aL16gf+geL8f+dO5ddO4aR16g16"
40 B$="sm2000t150O3L8A4D4ac+df+df+df+ac+df+df+df+"
50 C$="sm2000t150O3L8f+4f+4r8er8aR8AR8AR8ER8AR8AR8A"
60 PLAY A$,B$,C$
70 A$="O4L8f+ef+dO3aO4df+aL16gf+geL8D4ggB4aO5cO4B4":B$="O3L8ac+df+df+df+ac+d4gbO4g4f+f+g4":
   C$="O3L8r8ER8AR8AR8AR8EF+4R8DD4dAD4"
80 PLAY A$,B$,C$
90 A$="O4L8bO5d4O4bO5l16cdcO4al8bgdgb4aO5cO4b4bO5d4O4bO5cO4B16A16g4":
   B$="O4L8gb4O3bdf+g4df+O4g4O3df+O4g4gO3bgbdf+O4d4":
   C$="O3L8gO4g4O3ddad4gdO4d4dad4gdgdR8ab4"
100 PLAY A$,B$,C$
110 NEXT-
```

- 3)
  062-23.bas
  062-23.bas

```
2 '"Неаполитанская песенка" (отрывок) П.И.Чайковский.
5 A$="sm1500l6o6t112":B$="sm1500l6o6t112":C$="sm1500l6o6t112"
6 PLAY A$,B$,C$
10 FOR I=1 TO 2
20 A$="l8o5e-ce-ce-ce-l16fe-dcco4b-b-b-o5co4b-b-b-o5co4b-b-b-o5co4b-a-g"
30 B$="l8o3r8a-o4co3a-o4co3a-o4co3a-o4co3e-b-e-b-e-b-e-b-"
40 C$="o4l8r8r8e-r8e-r8e-r8e-r8e-r8e-r8e-r8e-"
50 PLAY A$,B$,C$
60 A$="o4l16b-a-a-a-b-a-a-a-b-a-a-a-b-a-gfe-e-fga-b-o5cde-8o3b-8o4e-8"
70 B$="o2l8b-o4do2b-o4do2b-o4do2b-o4do3e-b-r4r8gb-"
80 C$="o3l8r8fr8fr8fr8fr8gr4r8e-g"
90 PLAY A$,B$,C$
100 A$="l8o5e-ce-ce-ce-l16fe-dcco4b-b-b-o5co4b-b-b-o5co4b-b-b-o5co4b-a-g"
110 B$="l8o3b-8a-o4co3a-o4co3a-o4co3a-o4co3e-b-e-b-e-b-e-b-"
120 C$="o4l8o3go4r8e-r8e-r8e-r8e-r8e-r8e-r8e-r8e-"
130 PLAY A$,B$,C$
140 A$="o4l16b-a-a-a-b-a-a-a-b-a-a-a-b-a-gfe-e-fga-b-o5cde-e-fga-b-o6cde-8r8e-8r8"
150 B$="o2l8b-o4do2b-o4do2b-o4do2b-o4do3e-r8b-r8o4gr8fr8gr8o3b-r8"
160 C$="o3l8r8fr8fr8fr8fo3gr8o4dr8e-r8dr8cr8o3e-r8"
170 PLAY A$,B$,C$
180 NEXT
190 PLAY "l8o4e-","l8o4g","l8o4b-"
```

Более того, в музыкальную программу можно ввести элементы случайности. Они давно уже используются в *алеаторике* (лат.»aleator« — «азартный игрок»), где исполнителю время от времени предоставляется возможность самому выбирать, что играть.

В 1751 году в Англии музыкант Уильям Гейс написал сатирическое руководство «Искусство сочинять музыку исключительно новым способом, пригодным для самых захудалых талантов». Там предлагалось, обмакнув в чернила щётку, провести по ней пальцем. К брызгам–кляксам на заблаговременно подложенной нотной бумаге оставалось подрисовать хвостики, чтобы получились «полноценные ноты». В ту пору предлагались и другие неожиданные способы сочинения музыки. Не избежал заразительного увлечения даже Моцарт. В 1793 году он пишет «Руководство, как при помощи двух игральных костей сочинять вальсы, не имея ни малейшего представления о музыке и композиции».

В XVII в. было создано немало музыкальных пьес «на случайную тему», и некоторые из них приписывают Моцарту и Гайдну. Например, в шестнадцатитактовой части можно предусмотреть 11 различных вариантов для каждого такта, что можно представить в виде таблицы из 16 столбцов по 11 тактов в каждой, а исполнителю дано выбирать каждый такт, бросая игральные кости (по–видимому, музыкант должен заранее бросать кости 16 раз, чтобы не останавливаться перед каждым новым тактом). На нотной записи одного произведения этот способ был назван «несложным методом сочинения бесконечного числа менуэтов на три счета». «Бесконечное» — это, разумеется, слегка преувеличено, так как на самом деле их $11^{16} \approx 4.595 \times 10^{16}$ (46 квадриллионов). В произведениях подобного рода тщательно следили за тем, чтобы любые варианты тактов, выбранных в соседних столбцах, музыкально сочетались друг с другом и чтобы результат случайного выбора был вполне приемлем для слушателей XVIII в. В современной же музыке возможности случайного выбора все более приближаются к истинно случайным. Конечно, большинство из полученных таким образом пассажей совершенно неинтересны, но встречаются и запоминающиеся отрывки.

Пример. Программа «Композитор».
062-16.bas
062-16.bas

```
10 KEYOFF:PLAY"t100s1m10000l8","t100s1m10000l16"
20 SCREEN1:LOCATE 10,10:PRINT"Пьеска!"
30 DATA "o4gbo5d","o3gbo4dgf+g","o5dco4a","o4fgegce"
40 DATA "o4f+o5dc","o3ao4co3f+ao4df+","o4a#b.a16","o4go3gbo4dgf"
50 DATA "o4g+bo5d","o4efedo3bg+","o5co4ba","o3ao4aedce"
60 DATA "o4bag","o4egcecc+","o4bar8","o4dgf+dco3a"
70 DATA "o4gbo5d","o3gbo4dgeb","o5dco4a","o3ao4aeace"
80 DATA "o4f+o5dc","o4do5dco4af+a","o4a+bb","o3o4gf+gfg"
90 DATA "o5cde","o4egdgc+g","o5do4bg","o4dbgdo3bg"
100 DATA "o4gaf+","o3bo4dcdo3ao4d":Y=1
110 N=16:DATA"o4g","o3g8":Z=RND(-TIME)
```

```
120 DIM A$(N),B$(N),C$(N),D$(N),E(N),R(N)
130 FOR I=1 TO N:READ A$(I),B$(I):E(I)=I:NEXT
140 X=INT(RND(1)*15)+1
150 IF E(X)<>0 THEN R(Y)=E(X):Y=Y+1:E(X)=0 ELSE 140
160 IF Y<>16 THEN 140 ELSE C$(N)=A$(N):D$(N)=B$(N)
170 FOR I=1 TO N-1:C$(I)=A$(R(I)):D$(I)=B$(R(I)):NEXT
180 FOR J=1 TO N:PLAY C$(J),D$(J):NEXT
```

VI.3. Оператор SOUND

Этот оператор позволяет квалифицированному пользователю управлять Программируемым Звуковым Генератором (PSG). Имеется четырнадцать 8–битовых регистров,пронумерованных от 0 до 13, каждый из которых предназначен только для записи и может содержать значения в диапазоне от 0 до 255 (см.таблицу).

Номер регистра	Содержимое	Значение	Значение на паре регистров
0	Частота канала(голоса) A	0÷255 0÷15	0÷4095
1			
2	Частота канала(голоса) B	0÷255 0÷15	0÷4095
3			
4	Частота канала(голоса) C	0÷255 0÷15	0÷4095
5			
6	Частота шума	0÷31	
7	Коммутация каналов для генерации звука	0÷63	
8	Громкость канала A	0÷15	При значении 16 устанавливается режим управляемой громкости.
9	Громкость канала B	0÷15	
10	Громкость канала C	0÷15	
11	Частота огибающей («конверта») (управление частотой изменения громкости сигнала)	0÷255 (младш.биты)	0÷65535
12		0÷255 (старш.биты)	
13	Форма огибающей («конверта») (управление законом изменения громкости)	0÷15	

Оператор `SOUND` обеспечивает прямой доступ к регистрам звукового генератора. Синтаксис оператора:

```
SOUND R,V
```
,

где:

- `SOUND` («звук») — служебное слово;
- R — арифметическое выражение, целая часть значения которого принадлежит отрезку [0,13] и определяет номер звукового регистра;
- V — арифметическое выражение, целая часть значения которого принадлежит [0,255] и определяет новое содержимое звукового регистра, номер которого определяется параметром R.

1. *Определение частоты звучания* Частотой звука управляют 6 регистров (номера 0,1,2,...,5). Данные, которые необходимо поместить в регистры, могут быть определены из уравнения:

$$\frac{1789772.5(\text{Гц})}{16\times\text{частота(Гц)}} = 256 \times \text{содержимое}\left\lceil \begin{matrix}1\\3\\5\end{matrix}\right\rceil + \text{содержимое}\left\lceil \begin{matrix}0\\2\\4\end{matrix}\right\rceil .$$
регистров регистров

Все получаемые частоты являются делителями числа 1789800, которое выражает в герцах (числе колебаний в секунду)

значение частоты звукового генератора. Это число составляет примерно половину тактовой частоты микропроцессора Z80, равной 3579545 Гц.

Например, если мы хотим получить звук частотой 300 Гц из канала A, то содержимое регистров 0 и 1 (голос A!) определяется из уравнения:

```
1789772.5
————————— ≈ 373 = 256*1+117  .
 16×300
```

А теперь запишем с помощью оператора SOUND число 117 в регистр 0, а 1 — в регистр 1:

```
SOUND 0,117:SOUND 1,1
```

.

В случае канала B (используйте теперь регистры 2 и 3!) операторы примут вид:

```
SOUND 2,117:SOUND 3,1
```

.

Поскольку ширина полосы звукового канала не превышает на практике 10000 Гц, минимальное значение, которое может быть введено в пару регистров, задаётся величиной следующего выражения

```
 1789772.5
————————— ≈ 11.19
 16×10000
```

Таким образом, 11 надо поместить в регистр 0, а 0 — в регистр 1. Звук уже в значительной степени смягчился, но ещё «кое–что» слышно (разумеется, предел слышимой частоты зависит как от источника звука, так и от восприятия).

Минимальная частота определяется по следующей формуле:

```
1789772.5/16/4095=27.31 Гц
```

Предложим Вашему вниманию программу вычисления содержимого рассматриваемых регистров для любых заданных частот [76].
063-001.bas
063-001.bas

```
20 SCREEN0:KEY OFF
30 CLS
40 PRINT "Вычисление частоты"
50 INPUT "Введите частоту в Гц";HZ
60 IF HZ<28 OR HZ>55932! THEN 30
70 TMP=1789800#/(16*HZ)
80 CT=INT(TMP/256)
90 FT=TMPMOD256
100 PRINT "Частота";TAB(17);":";HZ;"Гц"
110 PRINT "Параметр точной настройки:";FT
120 PRINT "Параметр грубой настройки:";CT
130 PRINT"Будете ли продолжать (Y-да, N-нет)?";
140 A$=INKEY$:IF A$="" THEN 140
150 IF A$="Y" OR A$="y" THEN 30
160 END
```

Составим программу решения обратной задачи: определения частоты по содержимому регистров.
063-002.bas
063-002.bas

```
20 DEF FNFC(A,B)=INT((1789800#/((256*A)+B))/16)
30 CLS
40 PRINT "Вычисление частоты по значениям регистров"
50 INPUT"Параметр точной настройки:";FT
60 IF FT<0 OR FT>255 THEN BEEP:GOTO 50
70 INPUT "Параметр грубой настройки:";CT
80 IF CT<0 OR CT>15 THEN BEEP:GOTO 70
90 ' Исключение деления на нуль
100 IF CT=0 AND FT=0 THEN BEEP:GOTO 50
110 PRINT "Частота: ";FNFC(CT,FT);"Гц"
```

```
120 PRINT "Будете ли продолжать (Y-да, N-нет)?";
130 A$=INKEY$:IF A$="" THEN 130
140 IF A$="y" OR A$="Y" THEN 30
150 END
```

2. *Определение частоты шума*

   Акустический *шум* — это беспорядочные звуковые колебания, характеризующиеся случайным изменением амплитуды и частоты. Частота шума определяется содержимым 6-го регистра. Следующее соотношение определяет связь между данными в 6-м регистре и частотой:

$$\text{Данное} = \frac{1789772.5\ (\text{Гц})}{16 \times \text{частота шума(Гц)}}.$$

   Например, оператор `SOUND 6,15` означает, что частота шума около 7457 Гц.

3. *Микширование* (смешивание) *звука* и *шума*. Оно определяется значением, записанным в 7-й регистр.

Шум			Звук		
Канал C	**Канал B**	**Канал A**	**Канал C**	**Канал B**	**Канал A**
32	16	8	4	2	1

   Сложите числа, которые соответствуют выбранному каналу, вычтите результат из 63 и запишите полученное в регистр 7. Например, если Вы хотите получить звук голосов A и B и одновременно звук и шум голоса C, то проведите несложное вычисление 63-(2+1+4+32)=24 и используйте оператор `SOUND 7,24`.

   Пример.
   063-003.bas
   063-003.bas

```
10 SOUND 0,47:SOUND 1,1  'Частота канала A = 200 Гц.
30 SOUND 2,140:SOUND 3,0 'Частота канала B = 800 Гц.
50 SOUND 4,56:SOUND 5,0  'Частота канала C =2000 Гц.
70 SOUND 7,56 'Выбор звука из каналов A, B и C.
80 SOUND 8,9:SOUND 9,10:SOUND10,11'Выбор громкости канала.
```

   Выполните эту программу. А затем включите в программу строку:

```
65 SOUND 6,31 'Определена частота шума.
```

   и замените строку 70 на строку

```
70 SOUND7,48'Звук и шум из канала A, звук из каналов B и C.
```

   В результате генерируется звук, смешанный с шумом.

   Укажем ещё один способ заполнения регистра 7.

   В каждом из каналов возможен следующий выходной сигнал: отсутствие звука, только звук, только шум, звук и шум.

   Первые два бита регистра 7 должны быть двоичным числом 10, следующие три запрещают (1) или разрешают (0) выход шума, последние три запрещают (1) или разрешают (0) выход звука.

   Обратите внимание на то, что порядок каналов обратный!

   Пример.

```
              CBA CBA
SOUND 7, &B10 110 011
       шум звук
```

Канал A	шум, т.к. бит, соответствующий шуму из канала A — нулевой
Канал B	отсутствие шума и звука

Канал C	звук, т.к. бит, соответствующий звуку из канала C — нулевой

4. Звуковые эффекты, связанные с *формой волны* Оператор `SOUND` позволяет получать звуковые эффекты, достигаемые при помощи совместного действия команд S и M. Форма волны («конверта») определяется значением, записанным в регистр 13 (то же самое значение равно аргументу команды S). Частота огибающей «конверта» определяется данными, находящимися в регистрах 11 и 12, при помощи следующего уравнения:

```
 1789772.5(Гц)        ⎡ содержимое    ⎡ содержимое
─────────────── = 256 × ⎨ регистра  +  ⎨ регистра    .
256×частота(Гц)       ⎣ 12            ⎣ 11

Например, если частота=10 Гц, то
1789772.5
────────── ≈ 699=256×2+187,
 256×10
```

и операторы: `SOUND 12,2:SOUND 11,187` позволят получить желаемый результат!

Минимальное значение частоты, следовательно, равно 0.107 Гц, что приближённо соответствует периоду 9.4 секунды. Максимальное значение оценивается около 6991.3 Гц и тем самым принадлежит к слышимым частотам.

Пример. Программа, позволяющая генерировать звук движущегося паровоза.
063-004.bas

063-004.bas

```
10 FOR I=6 TO 13:READ J:SOUND I,J:NEXT I
20 DATA 31 ◄──────── Частота шума.
30 DATA 7 ◄──────── Шум из каналов A,B и C.
40 DATA 16,16,16 ◄── Изменение громкости каналов A,B и C.
50 DATA 71,2 ◄──────── Период "конверта" равен 12 Гц.
60 DATA 14 ◄──────── Форма волны равна 14.
```

5. Определение *громкости каналов* Регистры с 8 по 10 устанавливают громкость:

   - регистр 8 — канала A ,
   - регистр 9 — канала B,
   - регистр 10 — канала C.

Установка лежит в пределах от 0 до 15 (15 соответствует максимальной громкости).

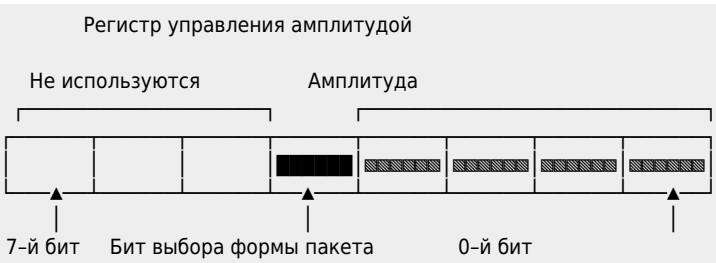
Если содержимое регистров равно 16, устанавливается режим автоматически изменяющейся громкости (громкость будет контролироваться генератором огибающей).

Примеры.

```
SOUND 8,&B00010000:SOUND 9,&B00001111

Канал A: Громкость контролируется генератором огибающей.
Канал B: Громкость равна 15.
```

Давайте посмотрим, в чем же причины «антагонизма» команд S и V. Дело в том, что при выполнении команды S в соответствующий регистр управления амплитудой записывается число 16 = 10000_2 . Взгляните на рисунок...



При этом в четвёртый бит попадает единица, в младшие биты — нули (старшие три бита не используются), следовательно, все установленные значения амплитуд сбрасываются. При выполнении же команды V (параметр V находится в диапазоне от 0 = 0000_2 до 15 = 1111_2) значение четвёртого бита всегда будет сброшено на 0.

6. *Регистр выбора формы пакета*

Отметим, что хотя параметр S может принимать значения от 0 до 15, некоторым из них соответствуют одинаковые пакеты.

В чем причина этого?

Выбор *формы пакета* осуществляется при записи требуемого значения в регистр 13.Значащими в нем являются лишь *четыре младших* разряда. Однако в ряде ситуаций значения двух младших разрядов не играют роли, поэтому при записи в регистр чисел, не превышающих семи, в результате может получаться одна и та же форма пакета.

Числам, большим семи, однозначно соответствуют *различные* формы волновых пакетов.

Всегда необходимо помнить, что оператор BEEP изменяет содержимое звуковых регистров!

Для иллюстрации возможностей оператора SOUND, рассмотрим следующие примеры:

- 1)
  063-01.bas
  063-01.bas

```
5 'Звук бомбардировщика.
6 'Формирование различных частот в трёх каналах имитирует "биения" в шуме пропеллеров,
7 'вращающихся с разными скоростями.
10 BEEP:FOR I=0 TO 13:READ V:SOUND I,V:NEXT:END

┌──────────────────────────────┐
│ 100 DATA 200, 14, 220, 14, 240, 14 │
│ 110 DATA 0                   │
│ 120 DATA &B10111000          │
│ 130 DATA 15, 15, 15          │
│ 140 DATA 0, 0, 0             │
└──────────────────────────────┘
```

  А теперь замените выделенный фрагмент на:

  - α)
    063-011.bas
    063-011.bas

```
5 'Звук м о р я
100 DATA 0,0,0,0,0,0
110 DATA 30
120 DATA &B10110111
130 DATA 16,0,0
140 DATA 0,90,&B1110
```

  - β)
    063-012.bas
    063-012.bas

```
5 'Вертолет 1
100 DATA 0,0,255,14,255,14
110 DATA 10
120 DATA &B10000001
130 DATA 8,8,16
140 DATA 0,20,&B1100
```

- 2)
  063-02.bas
  063-02.bas

```
5 'Вертолет 2
10 BEEP:FOR I=0 TO 13:READ V
20 SOUND I,V:NEXT
50 FOR I=5120 TO 300 STEP -10
60 SOUND 11,I MOD 256
70 SOUND 12,I\256
80 FOR J=0 TO 20:NEXT J,I
```

```
100 DATA 0,0,255,14,255,14
110 DATA 10
120 DATA &B10000001
130 DATA 8,8,16
140 DATA 0,20,&B1100
```

- 3)
  063-03.bas
  063-03.bas

```
5 ' Звук бомбы
10 FOR I=0 TO 13:READ V
20 SOUND I,V:NEXT
40 FOR I=50 TO 200:SOUND 0,I
60 A=5^5:NEXT
80 SOUND 12,100:SOUND 13,9
90 SOUND 7,&B10011111:END
100 DATA 50,0,0,0,0,0
110 DATA 30
120 DATA &B10111110
130 DATA 10,0,16
140 DATA 0,0,0
```

Для прекращения звука достаточно выполнить оператор BEEP или сбросить регистры 8–10 на 0.

- 4)
  063-04.bas
  063-04.bas

```
10 FOR I=1 TO 12:SOUND I+1,I+82:NEXT
20 FOR I=1 TO 8:SOUND I+1,I+81:NEXT '**** Шум морского прибоя
50 FOR X=0 TO 13:SOUND X,0:NEXT 'Очистка всех регистров
100 SOUND 7,62:SOUND 8,15:SOUND 9,16:SOUND 10,16
110 SOUND 12,16:FOR X=48 TO 170:A=5^5^5 '25ms задержка
120 SOUND 0,X:NEXT:SOUND 8,0 '********* Эффект падающей бомбы
130 SOUND 8,0:SOUND 6,15:SOUND 7,7:SOUND 12,16
140 FOR X=8 TO 10:SOUND X,16:NEXT
150 SOUND 13,0:FOR X=1 TO 500:NEXT '********** Эффект взрыва
160 SOUND 1,1:SOUND 7,46:SOUND 8,15:SOUND 9,9
170 FOR X=80 TO 33 STEP -1:POKE 0,X:NEXT
180 X=TAN(1):X=TAN(1)  '2·175ms задержка
190 FOR X=55 TO 40 STEP -1
200 POKE 0,X:A=5^5:NEXT '12ms задержка
210 FOR X=40 TO 100:POKE 0,X:NEXT
220 SOUND 8,0:SOUND 9,0 '********************* Эффект свиста
230 END 'Вызывает STOP, далее BEEP и звучание прекращается.
```

Примеры 5÷10 заимствованы из [76].

- 5)
  063-05.bas
  063-05.bas

```
10 'С и р е н а
20 SOUND 0,255:SOUND 1,0:SOUND 8,15:SOUND 7,&B00111110
60 'Нарастание высоты звука
70 FOR I=252 TO 170 STEP -2:SOUND 0,I:NEXT
100 'Убывание высоты звука
110 FOR I=170 TO 252 STEP 1:SOUND 0,I:NEXT
140 GOTO 70
```

Для генерации завывающего звука используется регистр, позволяющий изменять высоту исполняемой ноты. В данном примере высота периодически равномерно, круто нарастает, после падает до прежнего уровня.

- 6)

063-06.bas
063-06.bas

```
10 ' Ш у м о в о й  эффект
20 SOUND 6,0:SOUND 7,&B00000111
40 'Убывание частоты шума (проверить убывание)
50 FOR I=1 TO 31
60    SOUND 8,10:SOUND 9,10:SOUND 10,10
70    SOUND 6,1
80    FOR J=1 TO 50:NEXT J
90    SOUND 8,0:SOUND 9,0:SOUND 10,0
100   FOR J=1 TO 25:NEXT J
110 NEXT
120 GOTO 50
```

Для одного канала генерируется шум, частота которого непрерывно меняется от высокого значения до низкого. Такой эффект достигается при изменении содержимого регистра 6.

- 7)
063-07.bas
063-07.bas

```
10 ' З в у к о в ы е  эффекты ("тема пришельцев")
20 SOUND0,252:SOUND1,0:SOUND8,16:SOUND11,200:SOUND 12,2:SOUND 13,10
80 SOUND 7,&B00111110
90 PRINT"Нажмите любую клавишу..."
100 A$=INPUT$(1)
110 'Эффект модуляции:изменение частоты случайным образом
120 SOUND 12,1
130 SOUND 0,RND(1)*255
140 GOTO 130
```

Здесь звучание основного тона сопровождается эффектом модуляции. При нажатии клавиши регистр 12 вызывает рост периода волнового пакета, а частота звука (регистр 0) изменяется случайным образом. Этот пример хорошо демонстрирует широчайший диапазон генерируемых звуков.

- 8)
063-08.bas
063-08.bas

```
10 '"Перевернутые" звуки. Выбирается другая форма пакета, при которой амплитуда медленно растет,
11 'а затем резко  падает. Используются все три звуковых канала,
12 'причем частота одного из них слегка "расстроена".
20 R=RND(-TIME)
30 FOR I=0 TO 13:READ A:SOUND I,A:NEXT I
60 FOR I=1 TO 1000:NEXT
70 X=RND(1)*250
80 IF X<30 THEN 70
90 Y=RND(1)*10+1
100 SOUND 0,X:SOUND 1,Y:SOUND 2,X:SOUND 3,Y
120 SOUND 4,X+5:SOUND 5,Y:SOUND 13,13
140 GOTO 60
150 DATA 62,2,60,2,60,2,0,56,16,16,16,120,30,13
```

- 9) звуки барабана можно имитировать, создавая шумовой эффект с помощью «скачкообразного» пакета. Для имитации ударов турецкого барабана используется тот же пакет, но на этот раз модулируется не шум, а звуковой тон. В зависимости от нажатой клавиши программа выдаёт то или иное звучание. Регистр 7 позволяет выделить получаемый звук.
063-09.bas
063-09.bas

```
10 'Звуки  б а р а б а н а
20 SCREEN 0,0,0
30 FOR I=0 TO 12:READ S:SOUND I,S:NEXT I
70 PRINT  "Нажмите 'Б' для барабана"
80 PRINT  "Нажмите 'Т' для турецкого барабана"
90 A$=INPUT$(1)
100 IF A$="Т" OR A$="т" THEN SOUND 7,&B00111000:SOUND 13,1
```

```
110 IF A$="Б" OR A$="б" THEN SOUND 7,&B00000111:SOUND 13,1
120 GOTO 90
130 DATA 140,7,140,7,140,7,15,255,16,16,16,190,7
```

- 10) программа воспроизводит бой часов. Звук генерируется с помощью набора трёх частот, настроенных с некоторым рассогласованием. «Скачкообразный» пакет создаёт впечатление удара в колокол, а затухание звука достигается за счёт длинного модуляционного цикла. При изменении частоты тона можно добиться звучания, напоминающего колокольный звон, звучание треугольника (музыкальный инструмент) или даже звук удара по металлической трубе.
  063-10.bas
  063-10.bas

```
10 'Б о й  ч а с о в
20 ' Настройка звукогенератора
30 FOR I=0 TO 12:READ S:SOUND I,S:NEXT I
60 WIDTH 40:CLS:KEY OFF
70 PRINT "Часы бьют":PRINT
80 SOUND 7,56
90 FOR I=0 TO 12
100    SOUND 13,0
110    PRINT I;
120    FOR J=1 TO 1200:NEXT J
130 NEXT I
140 PRINT:PRINT
150 PRINT TAB(1);"12 часов":PRINT "и все отлично!"
160 END
170 DATA 229,0,97,0,115,0,0,63,16,16,16,190,120
```

- 11) *«Паровоз»* (автор программы студент 5–го курса Никитин А.Н.)
  063-11.bas
  063-11.bas

```
10 SOUND 7,&B00000111:SOUND 6,20
20 FOR I=8 TO 15:SOUND 8,I:SOUND 9,I:SOUND 10,I:GOSUB 80:FOR L=1 TO 50:NEXT:NEXT
30 FOR I=6 TO 13:READ A:SOUNDI,A:NEXT
40 DATA 31,7,16,16,16,250,5,14
50 FOR M=5 TO 2 STEP -1:RESTORE 70:FOR J=1 TO 12:FOR I=1 TO 200:NEXT:READ A:
   SOUND 11,A:SOUND 12,M:NEXT:NEXT
60 SOUND 7,&B00001100:SOUND 0,78:SOUND 1,6:SOUND 2,78:SOUND 3,6:SOUND 13,14:
   SOUND 11,11:SOUND 12,0:FOR L=1 TO 1500:NEXT:BEEP:END
70 DATA 225,200,180,160,150,125,100,80,50,40,20,10
80 SOUND 0,156:SOUND 1,12:SOUND 2,156:SOUND 3,12:SOUND 4,115:SOUND 5,9:RETURN
```

Итак, оператор SOUND представляет возможность программам, написанным на языке MSX BASIC, получать звуки, дополнительные к тем, которые воспроизводятся с помощью оператора PLAY.

И наконец, о «…двух ролях,которые могут играть компьютерные программы. В первой роли программа вычисляет то, что она вычисляет, и ничего более. Во второй роли программа снабжается одной–двумя звуковыми инструкциями по соседству с каждым внутренним циклом, внешним циклом и условной конструкцией (оператор IF). Во втором случае программа как бы поёт песню о задаче, которую решает. Тот, кто часто слушает свою излюбленную программу (будь она развлекательной или производственной), через некоторое время научится чувствовать по звуку, как она работает. Нет сомнений в том, что некоторые ошибки в новых версиях программы можно будет выявить на слух» [19].

VI.4. Примеры музыкальных программ

Отметим, что во всех примерах программ, приведённых в данном разделе, рекомендуется перед запуском программы выполнить команду BEEP. Последняя инициализирует звуковую микросхему, устанавливая её регистры по умолчанию в первоначальное состояние.

1. *«Игра в лошадки»* (П.И.Чайковский)
   064-01.bas
   064-01.bas

```
NEW
Ok
10 PLAY"s10m1800t200o4l8f+gf+bbbo5c+dc+ddd",
   "s10m2800o4l8t200ddddddb-b-b-aaa","s10m2800l8t200o3abag+g+g+gggf+f+f+"
20 PLAY"v10s1m2800o4l8f+gf+bbbo5c+dc+ddd",
   "v11s1m2800o4l8ddddddb-b-b-aaa","s1m2800l8o3abag+g+g+gggf+f+f+"
30 PLAY"V12o5f+gf+gggd+ed+eeec+dc+dddo4bo5c+o4baaa",
   "V12o5e-e-e-dddccco4bbbbbbaaag+g+g+ddd",
   "sm2800o4ccco3bbbaaag+g+g+gggf+f+f+eeef+f+f+"
40 PLAY"v13o5f+gf+gggd+ed+eeec+dc+dddo4bo5c+o4baaa",
   "v13o5e-e-e-dddccco4bbbbbbaaag+g+g+ddd",
   "o4ccco3bbbaaag+g+g+gggf+f+f+eeef+f+f+"
50 PLAY"sm500o4a+ba+f+f+f+f+f+f+f+f+f+bo5co4bo5c+c+c+d+ed+eeed+ed+o4bbbbbb",
   "m3000o4f+f+f+o3bbba+a+a+bbbo4aaabbbbbbbbbbbbbeeed+d+d+",
   "m3000o4c+c+c+o3f+f+f+eeedddd+d+d+o4gggo3f+f+f+gggo4f+f+f+o3bbbaaa"
60 PLAY"o4bbbef+eaaag+g+g+aaaf+gf+bbbo5c+dc+dddo4f+gf+bbbo5c+dc+dddf+gf+ggg",
   "o4eeedddeeededc+c+c+dddeeeb-b-b-aaadddeeeb-b-b-aaao5e-e-e-ddd",
   "o3gggg+g+g+aaaeeeeeeaaag+g+g+gggf+f+f+aaag+g+g+gggf+f+f+o4ccco3bbb"
70 PLAY"v13o5d+ed+eeec+dc+dddo4bo5c+o4baaaf+gf+gggd+ed+eeeo3bo4c+o3bo4dddag+adddf+gf+ggg",
   "v13o5ccco4bbbb-b-b-aaag+g+g+ddde-e-dddccco3bbbr8br8aaao4c+c+c+o3f+f+f+o4e-e-e-ddd",
   "o3aaag+g+g+gggf+f+f+e+e+e+f+f+f+aaagggf+f+f+eeee+e+e+f+f+f+gggdddo4ccco3bbb"
80 PLAY"o5c+dc+dddf+gf+gggo6c+dc+dddd","o4gggf+f+f+o5e-e-e-dddgggf+f+f+f+",
   "o3b-b-b-aaaaaadddeeedddd"
```

2. *«Лезгинка»* (Грузинский народный танец)
   064-02.bas
   064-02.bas

```
NEW
Ok
10 A$="T180L4O4":B$="T180L4O4":C$="T180L4O4":PLAY A$,B$,C$
20 A$="M35000S1":B$="M3500S1":C$="M3000S1":PLAY A$,B$,C$
30 A$="R8O5L8E4FGA16G16FED4G":B$="O4L4CECEDO5D":C$="O4L4R4GR4GR4O4B"
40 PLAY A$,B$,C$
50 A$="R8O5L8D4EFG16F16EDE4C4":B$="O3L4BO4GDGO4CG":C$="O4L4R4BR4BR4E"
60 PLAY A$,B$,C$
70 A$="R8O5L8E4FGA16G16FED4G":B$="O4L4CECEDO5D":C$="O4L4R4GR4GR4O4B"
80 PLAY A$,B$,C$
90 A$="R8O5L8D4EFG16F16EDE4E4":B$="O3L4BO4GDGO4Cb":C$="O4L4R4BR4BR4g+"
100 PLAY A$,B$,C$
110 'Вторая часть
120 A$="O4L8R8A4O5CEF16E16DCO4B4O5E4":B$="O3L4AO4CO3AO4CO3BO4B"
130 C$="O4L4R4ER4ER4G+":PLAY A$,B$,C$
140 A$="O4L8R8B4O5CDE16D16CO4BO5C4O4A4":B$="O3L4BEO4EG+O3AO5C"
150 C$="O4L4R4G+R4BR4O4E":PLAY A$,B$,C$
160 A$="O4L8R8A4O5CEF16E16DCO4B4O5E4":B$="O3L4AO4CO3AO4CO3BO4B"
170 C$="O4L4R4ER4ER4G+":PLAY A$,B$,C$
180 IF K=1 THEN 220
190 A$="O4L8R8B4O5CDE16D16CO4BAR8O5D4":B$="O3L4BEO4EG+O3AO4B"
200 C$="O4L4R4G+R4BR4O4g4":PLAY A$,B$,C$
210 IF K=0 THEN K=1:GOTO 10
220 A$="O4L8R8B4O5CDE16D16CO4BAR8O5e4":B$="O3L4BEO4EG+O3AO4C"
230 C$="O4L4R4G+R4BR4O4a4":PLAY A$,B$,C$:END
```

3. *«Менуэт из сонаты 13»* (Й.Гайдн)
   064-03.bas
   064-03.bas

```
NEW
Ok
2 FOR I=1 TO 2
3 PLAY"sm15000l8t180o4a4a4a4a2o5dc+",
  "s1m15000l8t180o4f+4f+4f+4f+4f+4f+4",
  "s1m15000l8t180o4d4d4d4d4d4d4d4"
4 PLAY"o5c+o4bbo5ggo4bbaa4.o5a","l4o4ggggf+","l4o4ddddd"
5 PLAY"o5agg4.f+","l8ago5d4c+4","l8ago4b4a4"
6 PLAY"o5f+ee4.d","f+eo4b4a4","f+eo4g4f+4"
7 PLAY"o5c+eeggf+","c+eo4c+4d4","o5c+eeged"
8 PLAY"o5e2","o5c+2","o4a4e4o3a4"
```

```
9 NEXT
10 FOR I=1 TO 2
11 PLAY"o4ao5c+c+4.o4aao5c+eggf+","a4o4g4g4e4c+4d4"
12 PLAY"o5f+ee4.g","f+4o4b4b4","f+4o4g4g4"
13 PLAY"f+ef+4.gf+eebbdc+2r4","f+4b4b4f+4b4b4a4e4o3a4o4","f+4g4g4f+4g4g+4"
14 NEXT
15 PLAY"l8o4a4a4a4a2o5sm5000dc+","l8o4f+4f+4f+4f+4f+4sm5000f+4","l8o4d4d4d4d4d4d4sm5000d4"
16 PLAY"o5c+o4bbo5ggo4bbasm15000a4.o5a","l4o4ggggsm15000f+","l4o4ddddsm15000d"
17 PLAY"sm8000o5agg4.f+","o4ago5d4c+4","o4agb4a4"
18 PLAY"o5f+ee4.d","o4f+eb4a4","f+eg4f+4"
19 PLAY"o5dbb4.aagg4.f+","o4dbg4f+4age4d4"
20 PLAY"sm2000o5f+eeddc+d1","v15o4g4f+4e4sm2000d4o3a4o4d2","v15o3a4a4a4o4"
```

4. *«Итальянская песенка»* (П.И.Чайковский)
   064-04.bas
   064-04.bas

```
NEW
Ok
20 GOSUB 370:GOSUB 120
30 GOSUB 40:END
40 A$="M5000S11":B$="M5000S1":C$="M5000S1"
70 PLAY A$,B$,C$
80 A$="T150L4O4":B$="T150L4O4":C$="T150L4O4"
110 PLAY A$,B$,C$
120 A$="R8L8F+GAB.F#16AF+g":B$="O3L8DF+f+af+f+c+aa"
125 C$="O3L8R8aar8aaR8ee"
130 PLAY A$,B$,C$
140 A$="L8O4E4R8R8gabO5D.c+16":B$="O3L8AEEc+EEAee"
145 C$="O3L8R8c+c+r8c+c+R8c+c+"
150 PLAY A$,B$,C$
160 A$="O4L8BO5c+O4aa4R8R8f+g":B$="O3L8Df+f+Af+f+Df+f+"
165 C$="O3L8R8AAR8AAR8AA"
170 PLAY A$,B$,C$
180 A$="O4L8AB.f+16Af+GE4R8R8GA":B$="O3L8Af+f+c+c+c+Ac+c+c+c+c+"
185 C$="O3L8R8AAR8AAR8AAR8AA"
190 PLAY A$,B$,C$
200 A$="O4L8BO5c+.O4A16O5D4R8D4R8":B$="O3L8Ac+c+Df+f+af+f+"
215 C$="O3L8R8EER8AAR8AA"
210 PLAY A$,B$,C$
220 A$="O4L8R8R8f+B4AAGEC+R8R8R8EO5c+4O4B"
223 B$="O3L8Df+f+Af+f+c+c+c+Ac+c+c+c+c+Ac+c+"
226 C$="O3L8R8AAR8AAR8EER8EER8EER8EE"
230 PLAY A$,B$,C$
240 A$="O4L8BAf+D4R8DR8A":B$="O3L8Df+f+Af+f+Df+f+"
245 C$="O3L8R8AAR8AAR8AA"
250 PLAY A$,B$,C$
260 A$="O5L16Dc+edc+O4Bbag+abaL8O5c+4R8cO4Ba"
265 B$="O3L8Af+f+c+c+c+Ac+c+c+c+c+":C$="O3L8R8AAR8EER8EER8EE"
270 PLAY A$,B$,C$
280 A$="O4L8gf+ebaf+D4R8":B$="O3L8Ac+c+Df+f+Af+f+"
285 C$="O3L8R8EER8AAR8AA"
290 PLAY A$,B$,C$
300 A$="O4L8R8R8f+B4aagec+4R8":B$="O3L8Df+f+Af+f+c+c+c+Ac+c+"
305 C$="O3L8R8AAR8AAR8EER8EE"
310 PLAY A$,B$,C$
320 A$="O4L8R8R8EO5c+4O4BBAf+D4R8DR8AL16O5Dc+edc+O4B"
323 B$="O3L8Ac+c+ac+c+df+f+af+f+df+f+f+f+f+"
326 C$="O3L8R8EER8EER8AAR8AAR8AAR8AA"
330 PLAY A$,B$,C$
340 A$="O4L16a+bO5c+O4bgf+L8E4R8R8d+egO3BO4c+E4R8R4R8D2"
343 B$="O3L8Gbbgbbac+c+Ac+c+Dc+c+Dc+c+A2"
346 C$="O3L8R8DDR8DDR8EER8EER8EER8EEf+2"
350 PLAY A$,B$,C$
360 RETURN
370 A$="M5000S3":B$="M1000S3":C$="M1000S3"
400 PLAY A$,B$,C$
410 A$="T150L4O4":B$="T150L4O4":C$="T150L4O4"
440 PLAY A$,B$,C$
450 RETURN
```

5. Перейдём к сочинению *«функциональной»* музыки [22].

   Будем создавать мелодию в до–мажор, что соответствует игре только по белым клавишам пианино (все ноты без диезов и бемолей). Более того, используем только 9 клавиш:

Нота	E	F	G	A	B	C	D	E	F
Октава	O4	O4	O4	O4	O4	O5	O5	O5	O5

   По желанию можно легко расширить диапазон, в котором звучит мелодия. Кроме того, можно выбрать другую тональность, указав на нотах знаки «-» и «+».

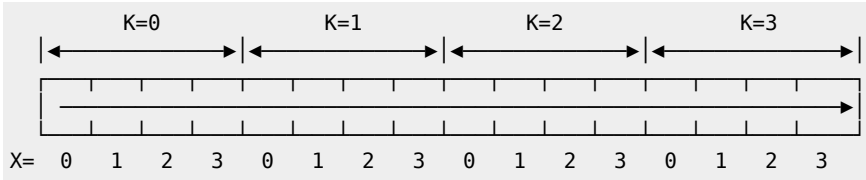
   Используем две функции: одну для построения мелодии, другую для создания ритма. Обе функции зависят от двух переменных K и X.

   Вычисляется значение функции FNF, определяется остаток (положительный) от деления на 9. Полученное число, принадлежащее отрезку [0,8], рассматривается как индекс массива N, что позволяет выбрать ноту.

   Значение функции FNR используется как коэффициент повторения данной ноты. Например, если это значение равно 2, то очередная нота подклеивается к переменной M$, содержащей мелодию, два раза (строки 210–230).

   Когда построение мелодии закончено, она исполняется (строка 250).

   Способ «вытягивания» нот в линию поясним рисунком.



   Замысловатая функция FNF, конечно подобрана. Испробуйте более простые функции. Если включить в определение функций стандартную функцию RND, компьютер начнёт «сочинять».
   064-05.bas
   064-05.bas

```
NEW
Ok
10 DATA o4e,o4f,o4g,o4a,o4b,o5c,o5d,o5e,o5f
20 DIM N$(8)
30 FOR I=0 TO 8
40    READ N$(I)
50 NEXT I
60 DEF FNF1(X)=(X+4)^2-10
70 DEF FNF2(X)=(2*X+1)^2+(X+1)*(1+(-1)^(X+1))/4+1
80 DEF FNF3(X)=3*((X+2)^2+X+2)/2-2
90 DEF FNF4(X)=5*(X^2+SGN(X)+1)
100 DEF FNS1(K)=1-K*(3-K)
110 DEF FNS2(K)=1-2*INT(K/2)
120 DEF FNS3(K)=(-1)^K
130 DEF FNPW(X,K)=1.5+(-1)^K*(X-1.5)
140 DEF FNR(X,K)=2*INT(1/(4-X))+INT((X+1)*(K+1)/16)+1
150 DEF FNF(X,K)=FNF1(FNPW(X,K))+FNS1(K)*FNF2(FNPW(X,K))+
    FNS2(K)*FNF3(FNPW(X,K))+FNS3(K)*FNF4(FNPW(X,K))
160 M$=""
170 FOR K=0 TO 3
180    FOR X=0 TO 3
190       I=FNF(X,K)MOD9:D=FNR(X,K)
200       IF I<0 THEN I=I+9
210       FOR J=1 TO D
220          M$=M$+N$(I)
230       NEXT J
240    NEXT X,K
250 PLAY M$
```

6. Программа *«Музыкальный* редактор» [22] служит для перевода нотной записи в буквенную. После запуска программы на экране рисуется нотный стан. По нему можно перемещать указатель. Установка ноты производится нажатием на клавишу управления курсором →. После установки исправить ноту *нельзя* (это недостаток редактора, однако для его устранения нужно значительно усложнить программу). До установки можно перемещать указатель по вертикали. Название ноты будет автоматически формироваться по положению на нотном стане. Можно придать ноте бемоль (знак «-») или диез

(знак «+»), а также длительность, задаваемую одной цифрой (1 — целая, 2 — половинная, 4 — четверть, 8 — восьмая). Если длительность не указана, то подразумевается четверть. Порядок ввода фиксирован:после длительности вводить знаки «-» и «+» нельзя.

Пока клавиша → не нажата, можно сбросить длительность и знаки «-» и «+», сдвинув указатель по вертикали. Программа заканчивает свою работу после нажатия на клавишу Ввод ⏎, музыка исполняется, а на экран выдаётся буквенная запись мелодии.

В массиве N$ сформированы тройки символов:октава и нота от 03C до 05B (строки 40–120). Нотный стан рисуется в строках 160–180. Мелодия накапливается в переменной M$. Запись конкретной ноты создаётся в переменной W$. Для обеспечения достаточного объёма памяти под строки использован оператор CLEAR.

В строке 210 в переменную F$ поступает символ нажатой клавиши. Если были нажаты вертикальные стрелки, то переменной W$ присваивается начальное значение, задающее ноту (строки 230–240).

При установке знаков «-», «+» и длительности используются признаки P1 и P2. Если P1=0, знаки «-» и «+» ещё не ставились, и их можно ставить (строка 250). Если P2=0, длительность ещё не указывалась (строка 260).
064-06.bas
 064-06.bas

```
NEW
Ok
10    '**** Подготовка ****
20 CLEAR 600
30 OPEN "GRP:" AS #1
40 DATA C,D,E,F,G,A,B
50 DIM N$(20)
60  FOR I=0 TO 2
70      FOR J=0 TO 6
80         READ A$
90         N$(I*7+J)="O"+STR$(I+3)+A$
100     NEXT J
110     RESTORE
120 NEXT I
130 SCREEN 2,0
140 SPRITE$(0)=CHR$(192)+CHR$(192)
150 M$="":X=S:Y=118:K=7:W$=N$(K)
160 FOR I=0 TO 4
170     LINE(0,104-I*16)-(255,104-I*16)
180 NEXT I
190 PUT SPRITE 0,(X,Y),,0
200    '**** Перемещение курсора ****
210 F$=INPUT$(1)
220 IF X>250 GOTO 370
230 IFASC(F$)=30ANDK<20THENY=Y-8:K=K+1:W$=N$(K):P1=0:P2=0:GOTO 190
240 IFASC(F$)=31ANDK>0THENY=Y+8:K=K-1:P2=0:W$=N$(K):P1=0:GOTO 190
250 IF(F$="+"ORF$="-")ANDP1=0ANDP2=0 THEN W$=W$+F$:P1=1:GOTO 210
260 IFVAL(F$)<>0ANDP2=0THENW$=W$+F$:P2=1:GOTO 210
270 IF ASC(F$)=28 THEN 290
280 IF ASC(F$)=13 GOTO 370 ELSE 210
290    '**** Установка ноты ****
300 PSET(X,Y-4),4
310 PRINT #1,"♪"
320 P1=0:P2=0
330 X=X+8:M$=M$+W$+" "
340 PLAY W$:W$=N$(K)
350 GOTO 190
360    '**** Окончание работы ****
370 PLAY M$
380 SCREEN 0
390 PRINT M$
400 END
```

В заключении данной главы приведем весьма полезную таблицу:

Октава	**Название ноты и её номер в команде N**											
1	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
	—	1	2	3	4	5	6	7	8	9	10	11

Октава	Название ноты и её номер в команде N											
2	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
	12	13	14	15	16	17	18	19	20	21	22	23
3	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
	24	25	26	27	28	29	30	31	32	33	34	35
4	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
	36	37	38	39	40	41	42	43	44	45	46	47
5	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
	48	49	50	51	52	53	54	55	56	57	58	59
6	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
	60	61	62	63	64	65	66	67	68	69	70	71
7	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
	72	73	74	75	76	77	78	79	80	81	82	83
8	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
	84	85	86	87	88	89	90	91	92	93	94	95
9	C											
	96											

7. *Дополнение.*
   «Соловей» (музыка народная)
   064-07.bas
   064-07.bas

```
10 SOUND7,&B10001000: SOUND8,15:SOUND1,0
20 J=RND(1)*25:U=RND(1)*15:E=RND(1)*20:FORC=0TOJ:FORI=0TO20:SOUND0,I:
   NEXT:NEXT:FORC=0TOU:FORI=0TO45:SOUND0,I:NEXT:NEXT:FORC=0TOE:FORI=30TO
   0STEP-1:SOUND0,I:NEXT:NEXT
30 GOTO 20
```

Диск с примерами

Загрузить образ диска

Открыть диск в WebMSX

---

1)
[101], [76] (стр. 107 Figure 7.2 Sound envelopes), [94] (Figure 5.8 Setting the wave forms of the envelopes)

https://sysadminmosaic.ru/msx/basic_dialogue_programming_language/006

2023-02-19 16:19